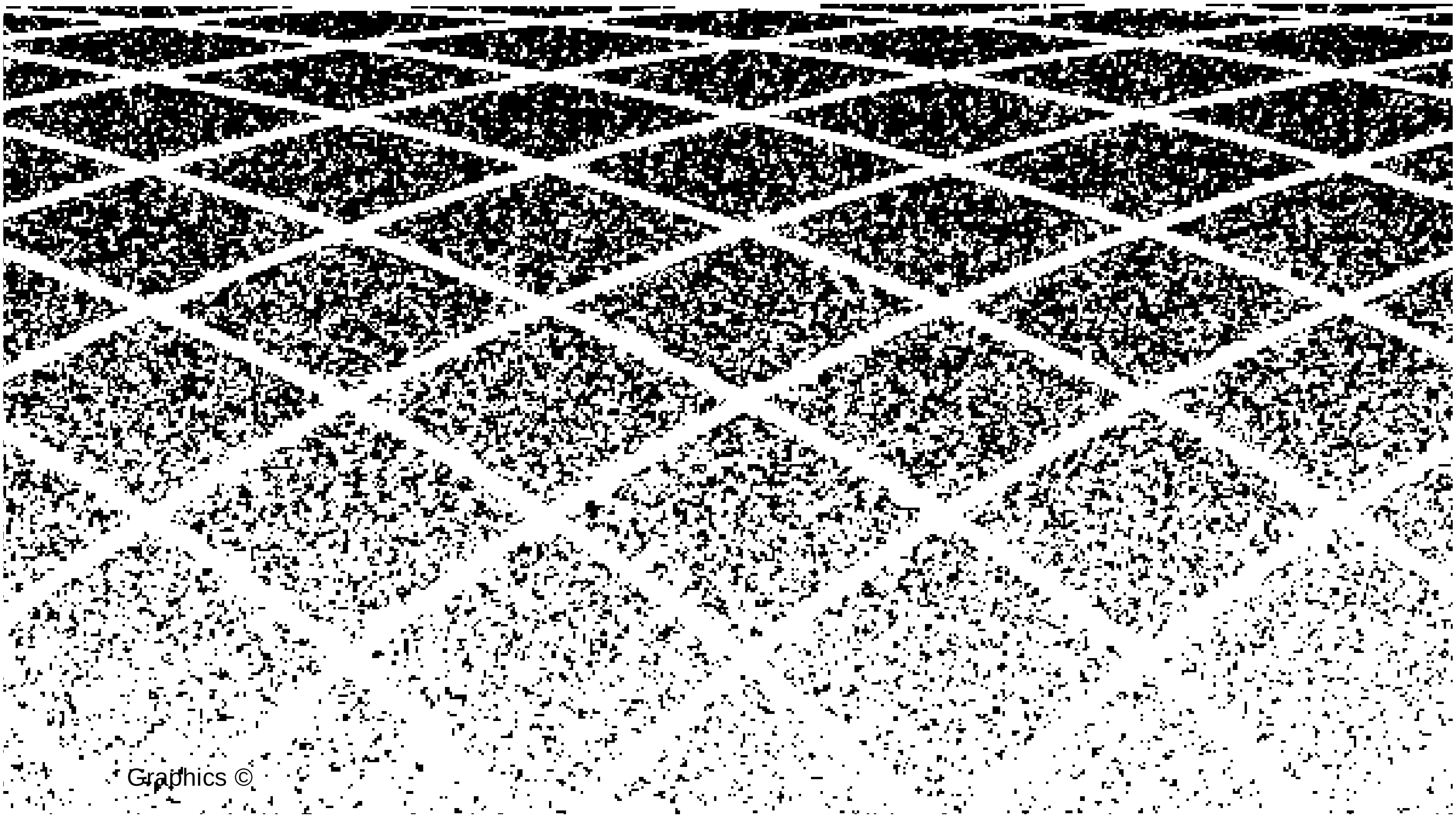
555-230-520
Issue 4
September, 1995

DEFINITY Generic 3 Call Vectoring/Expert Agent Selection (EAS) Guide



Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

About This Document

1

Feature Availability

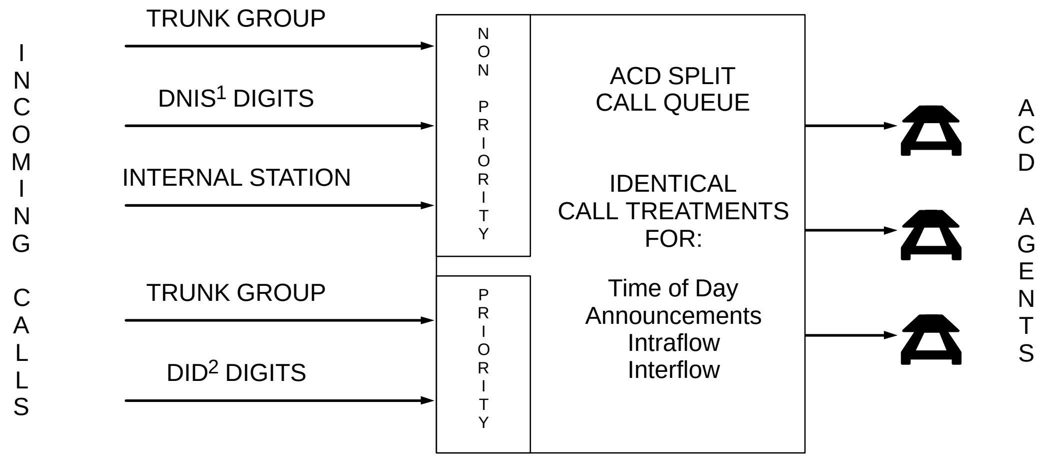
Call Vectoring is available with all Generic 3 releases. Many of the features discussed in this manual are only available with later releases of the DEFINITY switch. Some are optional features and must be purchased separately. See Appendix L for information about the availability of Call Vectoring features.

What is Call Vectoring?

NOTE:
Sample vectors are provided throughout this manual. These vectors are designed to illustrate vectoring features and capabilities. Because they are simplified to clearly demonstrate specific features, they are not complete and should not be used without modification at your call center.

Call Vectoring provides DEFINITY Generic 3 users with a highly flexible approach for managing incoming call traffic to the DEFINITY Generic 3 switch. By using a series of user-defined commands, you can direct or route internal and network calls as desired and thereby determine how these calls are processed. Calls can be directed to on- or off-network destinations, to Automatic Call Distribution (ACD) agents, or to various treatments.Call Vectoring also can be used in conjunction with CallVisor ASAI.

Call Vectoring works hand in hand with and, more importantly, enhances traditional ACD call processing, which may be illustrated as follows:



1. Dialed Number Identification Service
2. Direct Inward Dialing

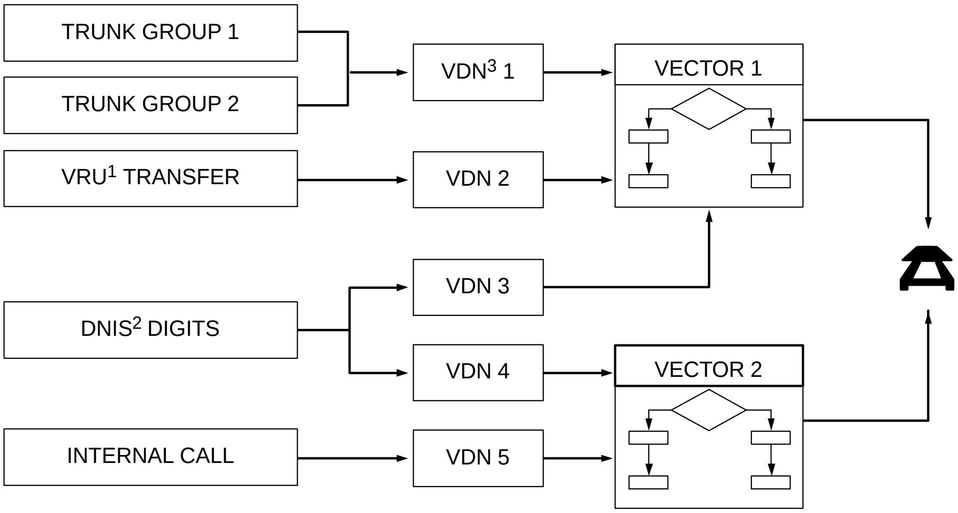
Figure 1-1. Traditional ACD Call Processing

As Figure 1-1 illustrates, the traditional ACD approach is rather limited in the way it handles queued calls (that is, all calls within a specific queue receive identical announcements, intraflow parameters, etc.). Call Vectoring, on the other hand, is flexible in that it permits unique treatments for each call according to a number of factors, including the number the caller dials, the number the caller calls from, the number of calls in queue, and the time of day and/or day of the week. This even applies to all calls that are ultimately handled by the same agent group.

Call Vectoring is comprised of three basic components, including:

- Vector Directory Numbers
- Vectors
- Vector commands

Working together, these components direct incoming and ASAI event reports and requests to the desired answering destinations, and they specify how each call is to be processed. Call Vectoring may be illustrated as follows:



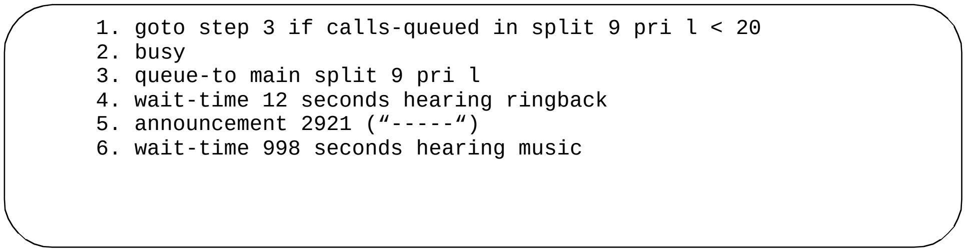
1. Voice Response Unit
2. Dialed Number Identification Service
3. Vector Directory Number

Figure 1-2. Call Vectoring

As Figure 1-2 illustrates, an incoming call to the DEFINITY G3 switch with Call Vectoring enabled is first directed to a Vector Directory Number (VDN). A VDN is an internal telephone number that, in turn, directs the call to a specific call vector. The VDN represents the call type or category (for example, billing, customer service, etc.), and thus, it defines the service desired by the caller. Multiple VDNs may point to the same or to different vectors, depending upon whether the relevant calls are to receive the same or different treatment.

The vector, which is the heart of Call Vectoring, is a set of commands that define the processing of a call. The type of processing that a call receives depends upon the commands included within the vector. For example, a call can be queued and then routed to another destination.

Here’s an example of a vector:



```
1. goto step 3 if calls-queued in split 9 pri l < 20
2. busy
3. queue-to main split 9 pri l
4. wait-time 12 seconds hearing ringback
5. announcement 2921 (“-----“)
6. wait-time 998 seconds hearing music
```

Figure 1-3. Vector Example

Each individual vector can contain up to 32 command steps or instructions. Multiple vectors can be chained together to extend processing capabilities or to process calls to the same or different answering destinations. Any number of calls can use the same multiple vectors and process steps independently. Understanding and planning your Call Center before you begin writing vectors is important. A planning guide is provided in Appendix J.

What is Expert Agent Selection (EAS)?

Expert Agent Selection (EAS) is a feature that allows Call Center managers to provide the best possible telephone service to the caller by matching the needs of the caller with the Skills or talents of the agents. Caller needs and agent Skills are matched via Call Vectoring. All the Call Vectoring features described in this guide can be used with EAS.

Matching the call to an agent with the appropriate Skills reduces transfers and call-holding time. Accordingly, customer satisfaction is increased. Also, since an entire agent group need not be trained at the same time for the same Skills, employee satisfaction is increased.

In addition to matching the Skills required for a call to an agent with one of these Skills, EAS provides other capabilities, as follows:

- **Logical Agent** associates hardware (voice terminal) with an agent only when the agent is logged in. While the agent is logged in, calls to the Agent LoginID are directed to the agent.
- **Direct Agent Calling** allows a user to call a particular agent and have the call treated as an ACD call.

Most EAS administration can be completed prior to enabling EAS. This minimizes the down time for upgrading to EAS. EAS is described in Chapter 10.

Call Vectoring Features

Call Vectoring provides the following features:

- **Basic Call Vectoring** allows you to “program” the type of processing that is given to a telephone call. You program by arranging a set of vector commands in the desired sequence. Depending upon the command, you can do the following:
  - — Place the call in queue until an agent is available to answer the call.
  - — Provide a recorded information or delay announcement to the caller.
  - — Allow the caller to leave a recorded message.
  - — Access a Voice Response Unit (VRU) so that a script can be started.
- **Call Prompting** allows you to collect digits and give some call control to the caller. Specifically, this feature allows the caller using a touch-tone or an internal rotary phone to enter digits that are subsequently processed by the vector. Among other tasks, Call Prompting allows the caller to do the following:
  - — Select one or more options from a menu in order to access recorded information provided by the call center or be routed to the correct split or agent.
  - — Enter an extension to which a call can be routed.
  - — Provide the call center with caller entered data (such as a credit card number) that the center can use to process the call. This data also can be displayed on the agent’s voice terminal.
- **Advanced Vector Routing** allows you to route calls based on three additional conditions:
  - — Rolling Average Speed of Answer for a split, skill, or VDN.
  - — Expected Wait Time for a split (skill) or for a call.
  - — The number of calls that are active in a specified VDN
- **ANI/II Digits Routing** allows you to route calls based on:
  - — The caller identity (ANI) or,
  - — The type of line where the call was originated (II-digits).
- **Look-Ahead Interflow** allows the call center to intelligently off-load some or all calls to another ACD switch. When this feature is enabled, a call arriving at a vector that is unable to handle the call (due to preset limits) can interflow to a switch in a remote location whenever the latter switch is able to receive the call. By using this feature, you can establish a “load balancing” of calls among multiple locations.

To use Look-Ahead Interflow, Basic Call Vectoring and Integrated Services Digital Network-Primary Rate Interface (ISDN-PRI) must be enabled. Private Network Access (PNA) software may also be required.

- **Adjunct Routing** provides you with a means of evaluating calls before the calls are processed. Specifically, this feature allows a DEFINITY PBX to request instructions from an associated adjunct, which is a processor that performs one or more tasks for another processor (the switch, in this case). The adjunct makes a routing decision according to agent availability and/or caller information sent by the switch, and it returns the routing response to the switch. By using this feature, the call center ensures that each call is delivered to the appropriate destination.

  To use Adjunct Routing, Adjunct Switch Applications Interface (ASAI) capabilities and Basic Call Vectoring must be enabled. Adjunct Routing can also be used in conjunction with Call Prompting and Look-Ahead Interflow.

Contents and Organization of the Guide

The *DEFINITY Generic 3 Call Vectoring/Expert Agent Selection (EAS) Guide,* 555-230-520 discusses all facets of Call Vectoring and EAS.

The guide is organized as follows:

- Overview (Chapter 1)
- Tutorial (Chapter 2)
- Reference (Chapters 3 through 11, Appendices A through N)
- Glossary
- Index

The guide first concentrates on illustrating Call Vectoring principles (Chapters 1 through 9). Chapter 10 presents a thorough discussion of EAS, which builds on Call Vectoring. Chapter 11 presents several Call Vectoring and EAS applications. Finally, the appendices, Glossary, and Index provide information and references to both Call Vectoring and EAS topics.

Intended Audience and Use of the Guide

The guide is intended primarily for personnel who opt to use Call Vectoring and/or EAS. You should use this guide as an information source for implementing Call Vectoring and/or EAS. A knowledge of Automatic Call Distribution (ACD) is assumed.

The level of your expertise in Call Vectoring and/or EAS should determine how you use the guide. Users who are unfamiliar with Call Vectoring should read the overview, then study the tutorial. Users who will be using EAS should read Chapter 10. Users who want to learn more about Call Vectoring should review Chapters 3 through 11 in order to get a good grasp of how the Call Vectoring features function. Finally, advanced users of Call Vectoring and/or EAS may only find it necessary to periodically reference a specific appendix or two (such as Appendix A, which contains a set of Call Vectoring/EAS command “manual pages”) to get the information needed.

Users who want to set up a Call Center (EAS and non-EAS) should read Appendix J, and users who want to convert a Call Center to EAS should read Appendix K.

References

The publications listed in this section should be used to supplement the information presented in this guide:

- *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204
- *DEFINITY Communications System Generic 3 CallVisor ASAI Technical Reference*, 555-230-220
- *DEFINITY Communications System Generic 3 Version 4 Implementation*, 555-230-655 or
  DEFINITY Communications System Generic 3 V2/V3 Implementation, 555-230-653
- *DEFINITY Communications System Generic 3 Basic Call Management System (BCMS)*, 555-230-704
- *CentreVu™ Call Management System Release 3 Version 4 Administration,* 585-215-800
  CentreVu™ Call Management System Release 3 Version 4 Reports, 585-215-801
  (or previous version CMS documentation that came with your system)
- *GBCS Products Security Handbook*, 555-025-600.

Tutorial

2

Introduction

This chapter is intended to provide you with a ‘‘practical start’’ in using Call Vectoring. To this end, the chapter presents the basics you need to write a representative vector and to enter it on-line. The last section of the chapter summarizes the benefits of Call Vectoring, and it identifies example vectors in the reference section of the guide that illustrate these benefits.

Entering the Vector On-Line

A vector can be entered on-line via the following two methods:

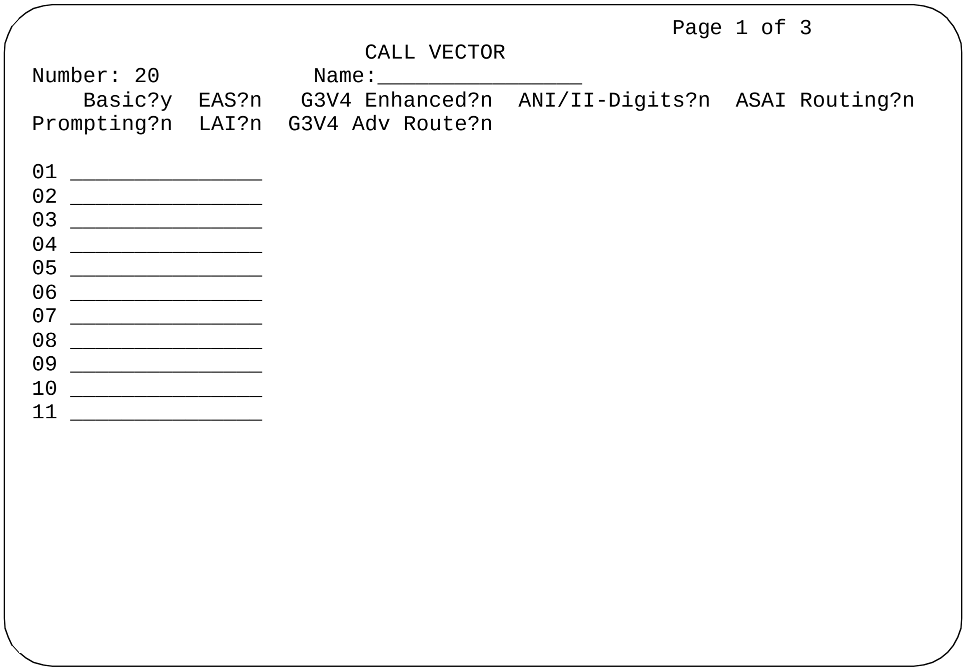
- Basic Screen Administration (on the Manager I or G3-MA)
- Call Management System (CMS)

NOTE:
All references to CMS in this manual, unless otherwise noted, refer to CMS Release 2, CMS Release 3 through Version 2 or CenterVu™ CMS.

The following section discusses the Basic Screen Administration method for entering a vector on-line at your DEFINITY system. For complete details for creating a vector with CMS, consult the *CentreVu™ Call Management System Release 3 Version 4 Administration,* 585-215-800, document (or earlier version CMS documents that you received with your system).

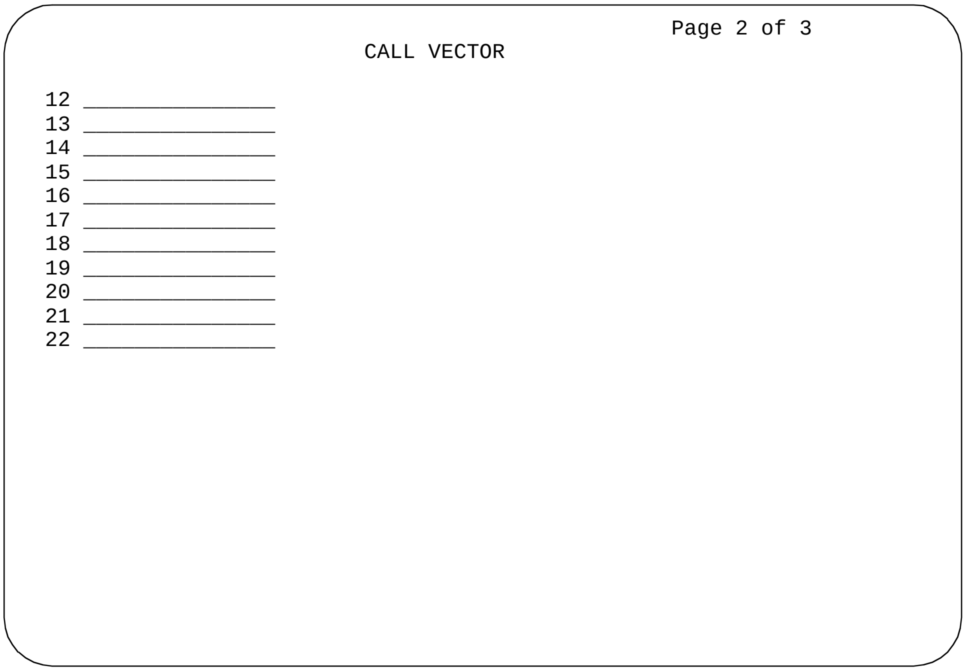
Procedures for Basic Screen Administration

Basic Screen Administration is a function of the DEFINITY Communications System. A vector is entered on-line via Basic Screen Administration by completing the Call Vector Form. This form appears on three screens, as follows:



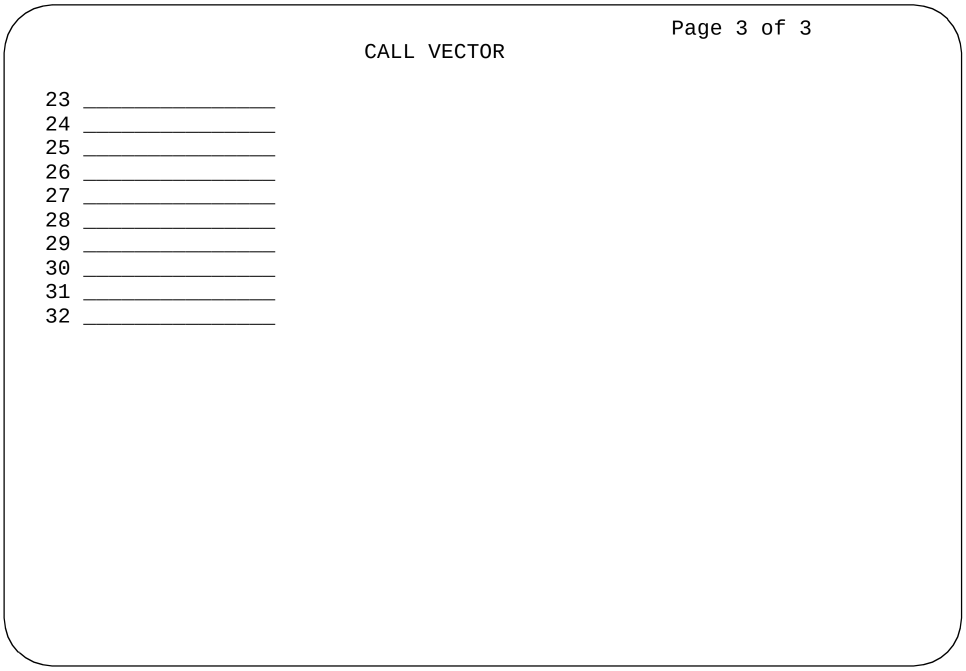
```
                                                 Page 1 of 3
                           CALL VECTOR
Number: 20            Name:________________
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 _______________
02 _______________
03 _______________
04 _______________
05 _______________
06 _______________
07 _______________
08 _______________
09 _______________
10 _______________
11 _______________
```

Figure 2-1.   Call Vector Form



```
                                                  Page 2 of 3
                               CALL VECTOR

  12 _______________
  13 _______________
  14 _______________
  15 _______________
  16 _______________
  17 _______________
  18 _______________
  19 _______________
  20 _______________
  21 _______________
  22 _______________
```

Figure 2-2. Call Vector Form (Page 2 of 3)



```
                                                  Page 3 of 3
                               CALL VECTOR

  23 _______________
  24 _______________
  25 _______________
  26 _______________
  27 _______________
  28 _______________
  29 _______________
  30 _______________
  31 _______________
  32 _______________
```

Figure 2-3. Call Vector Form (Page 3 of 3)

The following list summarizes how you can enter a vector on-line via Basic Screen Administration. For complete details on this process, consult *DEFINITY Communications System Generic 3 Version 4 Implementation,* 555-230-655 *or DEFINITY Communications System Generic 3 V2/V3 Implementation,* 555-230-653.

1. Access the Call Vector Form by executing the “change vector x” command, where *x* is a number between 1 and 256 [G3i], 1 and 48 [G3s PBP, G3vs PBP], or 1 and 512 [G3r]. Use the “change vector” command either to change an existing vector, or to create a new vector.

   If you are not certain of the number or name of a vector, enter the “list vector” command to view a complete list of all vectors that have been administered for your system.

2. Assign a name to your vector by completing the blank next to **Name**. The vector name can contain up to 15 alphanumeric characters.

   NOTE:
   The vector number, which appears next to **Number**, is automatically assigned by the system.

3. Look at the next fields and note where a *y (yes)* appears. These fields indicate the Call Vectoring features and corresponding commands you can use. (The Call Vectoring features are optioned from the Customer Options Screen.) On the other hand, if an *n* appears in one of these fields, you cannot use the corresponding feature. A y in one of the fields indicates the following.

Basic
You can use the Basic Call Vectoring commands. See Chapter 4, "Basic Call Vectoring".

EAS
Expert Agent Selection is enabled. See Chapter 10, "Expert Agent Selection".

G3V4 Enhanced
You can use the G3V4 Enhanced Vector Routing commands and features. See Appendix L, "V4 Feature Availability" for an explanation of which features are included with G3V4 Enhanced Vector Routing.

ANI/II-Digits
You can use the ANI and II-Digits Vector Routing commands. See Chapter 7, "ANI and II-Digits Routing". ANI/II-Digits Routing requires G3V4 Enhanced Vector Routing.

ASAI Routing
You can use the Adjunct Routing command. See Chapter 9, "Adjunct Routing".

Prompting
You can use the Call Prompting commands. See Chapter 5, "Call Prompting".

LAI
Look-Ahead Interflow is enabled. See Chapter 8, "Look-Ahead Interflow".

G3V4 Adv Route
You can use the G3V4 Advanced Vector Routing commands. See Chapter 6, "Advanced Vector Routing".

4. Enter a maximum of 32 vector commands in the blanks next to the step numbers. See Appendix A for a complete description of all Call Vectoring commands.

   NOTE:
   You need not type every letter of each command that you enter. If you type just the first few letters of a command and press RETURN or TAB, the system spells out the entire command.

5. Save the vector in the system by pressing ENTER.

Enhanced Vector Editing (G3V4 and later releases)

Enhanced Vector Editing allows you to insert and delete vector steps while editing a vector on the switch.

To insert a vector step complete the following procedure:

1. On the vector form, press F6 (edit)
2. At the command line, type “i” followed by a space and the number of the step you would like to add. Enter the command.
3. Type the new vector step

When a new vector step is inserted, the system automatically renumbers all succeeding steps and renumbers *goto* step references as necessary.

To delete a vector step complete the following procedure:

1. On the vector form, press F6 (edit)
2. At the command line, type “d” followed by a space and the number of the step you would like to delete. Enter the command.

When a vector step is deleted, the system automatically renumbers all succeeding steps and renumbers *goto* step references as necessary.

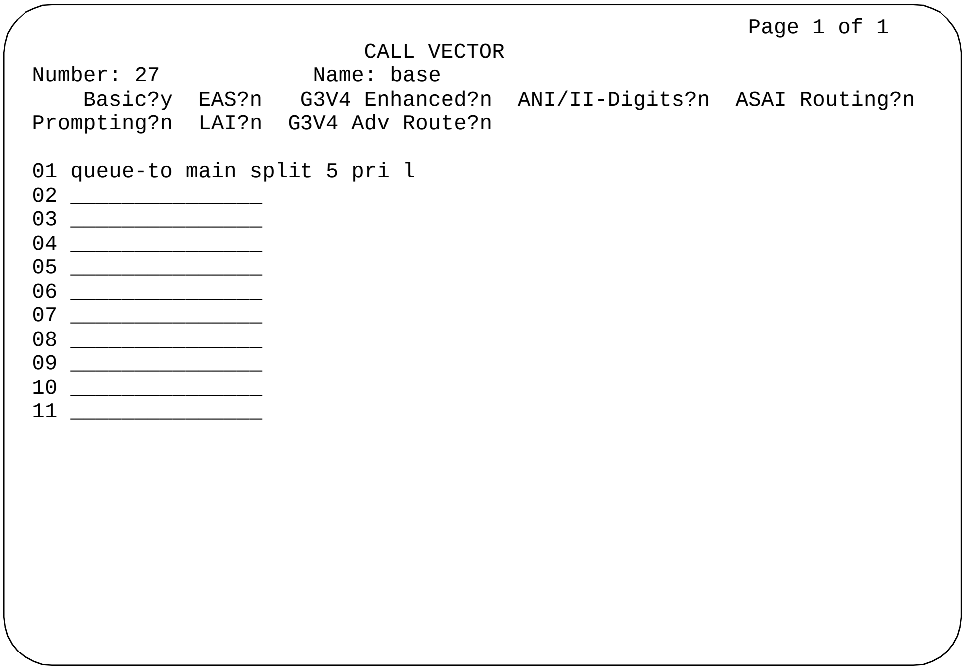
NOTE:
After editing a vector, be certain to verify that the vector will work as you intend it to. This is particularly important if you deleted a step that was the target of a *goto* step.

Constructing a Vector: One Approach

This section is intended to provide you with one logical approach to constructing a vector. In so doing, the section presents a starting vector that consists of one step and then builds upon this vector to produce a new vector that provides additional functions. This ‘‘vector building’’ process continues through several phases until a final ‘‘complete’’ vector is constructed. As each phase is presented, you are introduced to one or more new vector commands and/or approaches to vector processing. While it is not practical to present all such commands and approaches along the way to constructing a single ‘‘final’’ vector, those presented in this tutorial should allow you to get a good grasp of how to use Call Vectoring.

Phase 1: Queuing a Call to the Main Split

If a call cannot be immediately answered by an agent (or operator), the call is usually queued until an agent becomes available. A call can be connected to an available agent or queued via the following vector:



```
                                                                  Page 1 of 1
                          CALL VECTOR
Number: 27            Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 queue-to main split 5 pri l
02 _______________
03 _______________
04 _______________
05 _______________
06 _______________
07 _______________
08 _______________
09 _______________
10 _______________
11 _______________
```

Figure 2-4. Queuing Call to Main Split

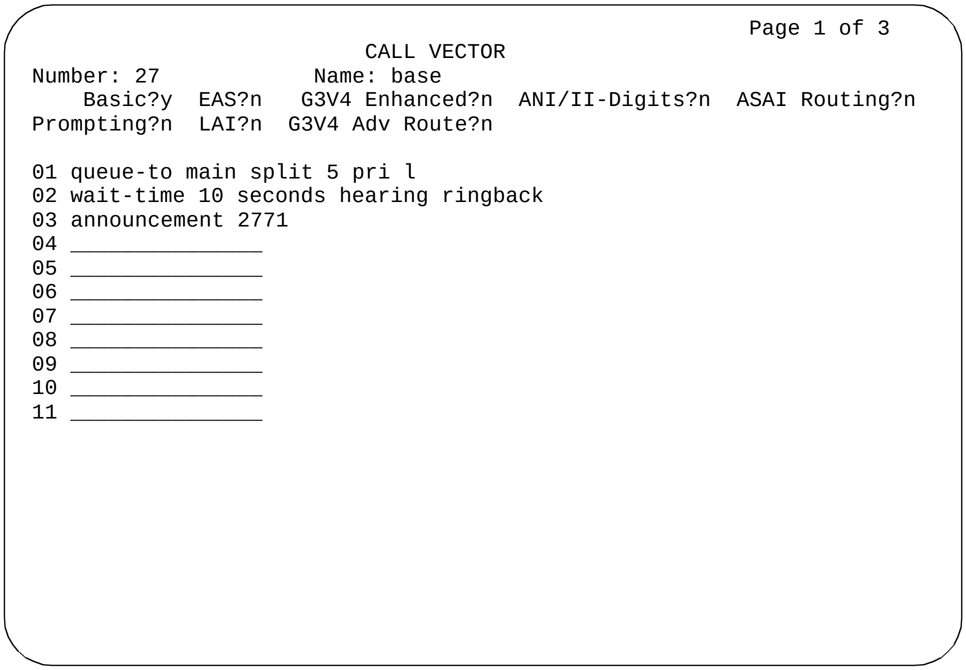
If an agent is available, the *queue-to main split* command automatically sends the call to the agent without queuing the call. However, if no agent is available, the command queues the call to the main split (or group) of agents. Once the call is sent to the main split queue, the call remains there until either it is answered by an agent or some other treatment is provided.

Each call queued to a split occupies one queue slot in that split. Calls are queued sequentially as they arrive according to the assignment of the priority level. In our vector, note the priority level *low* is assigned to the call. The priority level establishes the order of selection for each call that is queued. A call can be assigned one of four priority levels: *top, high, medium,* or *low*. Within a given split (the main split, in our vector), calls are delivered to the agent sequentially as they arrive to the split queue and according to the priority level assigned. Accordingly, calls assigned a *top* priority (if any) are delivered to an agent first, calls that are assigned a *high* priority are delivered second, etc.

Finally, note that the call is queued to Split 5. Split numbers range from 1 to 99 [G3i], from 1 to 24 [G3s PBP, G3vs PBP], and from 1 to 255 [G3r].

Phase 2: Providing Feedback and Delay Announcement

In the last section, we mentioned that a call remains queued until an agent becomes available to answer the call. In the meantime, the caller would no doubt like to hear some feedback assuring him or her that the call is being processed. The following vector provides one solution.



```
                                                              Page 1 of 3
                          CALL VECTOR
Number: 27            Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 queue-to main split 5 pri l
02 wait-time 10 seconds hearing ringback
03 announcement 2771
04 _______________
05 _______________
06 _______________
07 _______________
08 _______________
09 _______________
10 _______________
11 _______________
```

Figure 2-5. Providing Feedback and Delay Announcement

NOTE:
Announcement 2771 could contain this message: ‘‘We’re sorry. All of our operators are busy at the moment. Please hold.’’

The *wait-time* command in Step 2 provides a delay of a specified number of seconds before the next vector step is processed. The time parameter may be assigned any even number in the range of 0 through 998. In our vector, the time specified is 10 seconds.

In addition to the delay period, the *wait-time* command provides the caller with feedback. In our vector, *ringback* is provided. Other types of feedback that can be provided with the *wait-time* command are: silence; system music; or an alternate audio/music source. For more information see, "Delays with Audible Feedback" on page 4-5.

Theoretically, then, the *wait-time* command in our vector provides the caller with 10 seconds of ringback. But what happens if an agent answers the call before the *wait-time* command runs its course? If this happens, the command is terminated (that is, the delay period is ended and the accompanying feedback is stopped). So, returning to our example, let’s presume the call is delivered to an agent after four seconds. In such a case, the following is true:

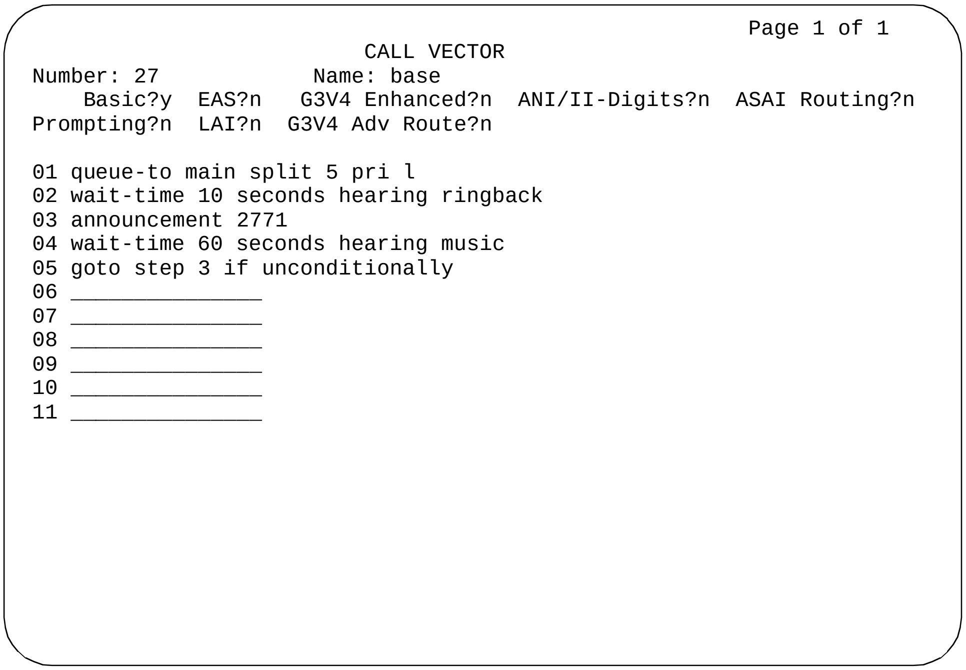
- Caller does not hear the remaining six seconds of ringback, inasmuch as the delivery of the call to the agent is the primary objective.
- Announcement in Step 3 (discussed next) is not played.

If the call is not answered by the time the *wait-time* command in Step 2 is completed, vector processing continues with the *announcement* command in Step 3.

The *announcement* command consists of a recorded message, and it is often used to encourage the caller to stay on the phone or to provide information to the caller. If a call is delivered to an agent during the *announcement* command, the announcement is interrupted. Otherwise, the announcement is played from beginning to end. Thereafter, the call remains in queue until it is answered by an agent or until the caller hangs up. Multiple callers can be connected to an announcement at any time. See “Recorded Announcement” in the *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204, for more information about announcements.

Phase 3: Repeating Delay Announcement and Feedback

The vector in the previous section provides feedback to the caller after the call is queued. However, if the announcement in Step 3 is played, and if the agent does not answer the call soon after the announcement is complete, the caller may end up holding the line for too long a time without receiving any further feedback or treatment. The following vector provides one solution:



```
                                                      Page 1 of 1
                           CALL VECTOR
Number: 27           Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 queue-to main split 5 pri l
02 wait-time 10 seconds hearing ringback
03 announcement 2771
04 wait-time 60 seconds hearing music
05 goto step 3 if unconditionally
06 _______________
07 _______________
08 _______________
09 _______________
10 _______________
11 _______________
```

Figure 2-6. Repeating Delay Announcement and Feedback

The *wait-time* command in Step 4 of this vector provides additional feedback (this time, music) to the caller. If the call is not answered by the time Step 4 completes, the *goto step* command in Step 5 is processed.

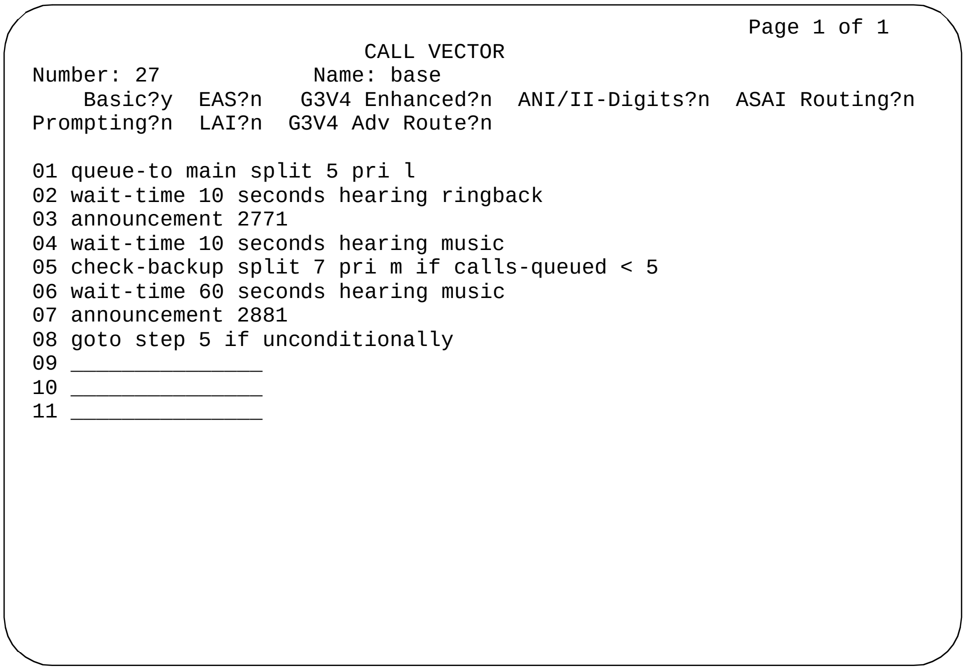
Up to this point, we have discussed and illustrated Call Vectoring commands that cause *sequential flow* (that is, the passing of vector processing control from the current vector step to the next sequential vector step). The *goto step* command is an example of a Call Vectoring command that causes *branching* (that is, the passing of vector processing control from the current vector step to either a preceding or succeeding vector step).

The *goto step* command in Step 5 allows you to establish an announcement-wait loop that continues until the agent answers the call. Specifically, the command makes an unconditional branch to the *announcement* command in Step 3. If the call is not answered by the time the announcement in Step 3 is complete, control is passed to the *wait-time* command in Step 4. If the call is still not answered by the time this command completes, control is passed to Step 5, where the unconditional branch is once again made to Step 3. As a result of the established loop, the caller is provided with constant feedback.

Phase 4: Queuing a Call to a Backup Split

Up to this point, we have dealt with a call queued to one split: the main split. However, Call Vectoring allows a call to be queued to a maximum of three splits simultaneously. If a call is queued to multiple splits, the call has a better chance of being answered more quickly. Multiple split queuing is especially useful during periods of heavy call traffic.

The following vector allows a call to queue to two splits:



```
                                                               Page 1 of 1
                          CALL VECTOR
Number: 27            Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 queue-to main split 5 pri l
02 wait-time 10 seconds hearing ringback
03 announcement 2771
04 wait-time 10 seconds hearing music
05 check-backup split 7 pri m if calls-queued < 5
06 wait-time 60 seconds hearing music
07 announcement 2881
08 goto step 5 if unconditionally
09 _______________
10 _______________
11 _______________
```

Figure 2-7. Queuing Call to Backup Split

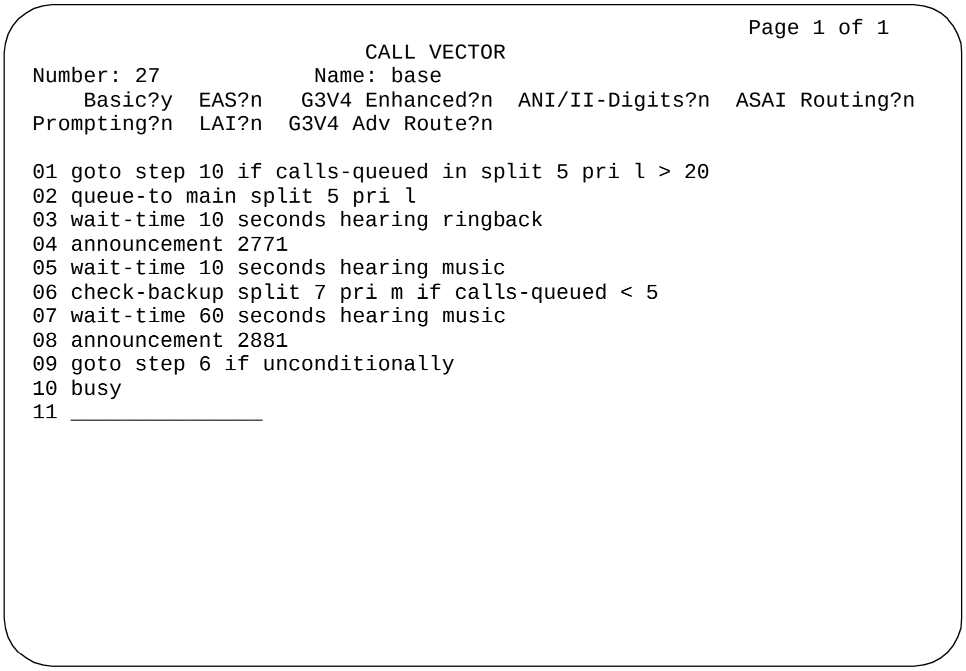
We have already discussed how the *queue-to main split* command in Step 1 queues the call to the main split. If the call is not answered by the time the *wait-time* command in Step 4 completes, the *check-backup split* command in Step 5 attempts to queue the call to backup Split 7 at a medium priority. The condition expressed in the command (*if calls-queued < 5*) determines whether or not the call is to be queued to the backup split. Specifically, if the number of calls currently queued to Split 7 at a medium or higher priority is less than 5, the call is queued to the split. Note that if the call is queued, the call in this case is assigned a *medium* priority instead of a *low* priority, which is assigned if the call is queued by the *queue-to main split* command in Step 1. It is a good practice to raise the priority level in subsequent queuing steps in order to accommodate callers who have been holding the line for a period of time. (We could have even assigned a *high* priority instead of just a *medium* priority in Step 5.)

The *calls-queued* condition is one of seven conditions that can be included in the *check-backup split* command. The other conditions are *unconditionally, average speed of answer (rolling-asa), available agents, staffed agents, expected wait time* and *oldest call waiting*. Some of these conditions are only available with G3V4 and later releases; see Appendix L, ''V4 Feature Availability'' for information. As is true for the *queue-to main split* command, the *check-backup split* command can queue a call at one of four priorities: *low, medium, high* or *top*.

We are including a queuing step within the loop, thus giving the call repeated opportunities to queue (if necessary). The call queues to split 7 only once.

Phase 5: Checking the Queue Capacity

It is a good practice to check the main split queue for the number of calls already queued before allowing another call to queue to the split. The reason for this is that there is a limited number of queue slots assigned to each split. The number of such slots assigned to each split is defined in the queue length field on the hunt group screen. A call that attempts to queue to a split with no available queue slots cannot be queued to that split and, accordingly, the *queue-to main split* command fails. Vector processing would then continue with the next vector step. The following vector contains provisions for checking queue capacity:



```
                                                          Page 1 of 1
                          CALL VECTOR
Number: 27           Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 goto step 10 if calls-queued in split 5 pri l > 20
02 queue-to main split 5 pri l
03 wait-time 10 seconds hearing ringback
04 announcement 2771
05 wait-time 10 seconds hearing music
06 check-backup split 7 pri m if calls-queued < 5
07 wait-time 60 seconds hearing music
08 announcement 2881
09 goto step 6 if unconditionally
10 busy
11 _______________
```

Figure 2-8. Checking Queue Capacity

A check of split 5 is implemented by the *goto step* command in Step 1. In this example, 21 slots are assigned to split 5 (that is, the queue length for split 5 is *21*). Accordingly, the *goto step* command tests whether the split contains more than 20 calls via the condition *if calls-queued in split 5 > 20 pri l*. If this test is successful, control is passed to the *busy* command in Step 10. The *busy* command gives the caller a busy signal and eventually causes the call to drop.

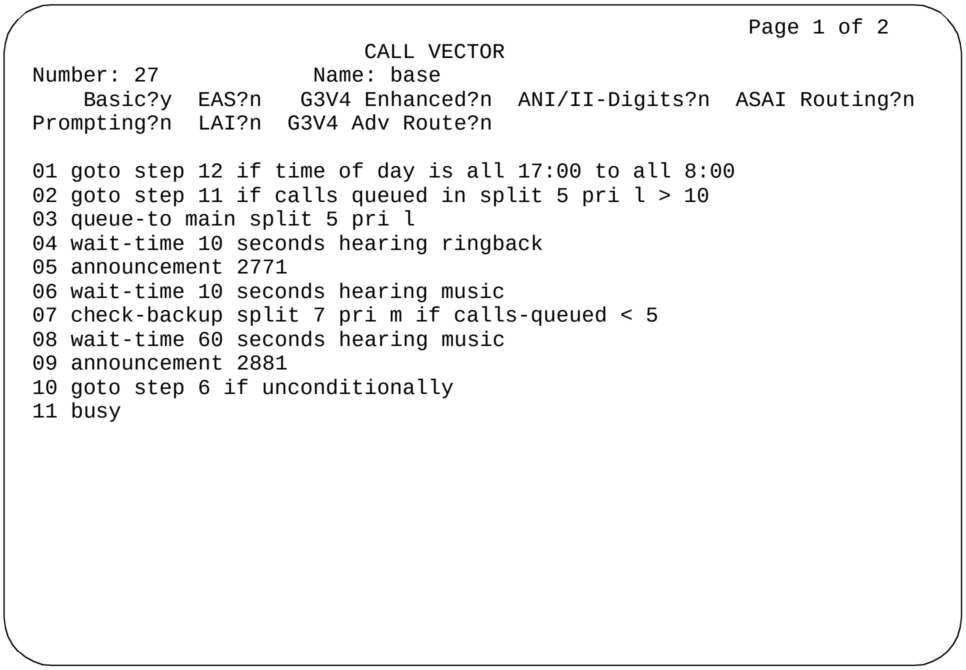
On the other hand, if 20 or fewer calls at a medium priority are already queued to the main split when Step 1 executes, the *queue-to main split* command in Step 2 queues the call, and vector processing continues at Step 3.

NOTE:

Instead of providing the caller with a busy tone if the *queue-to main split* step cannot queue the call, we can queue the call to another split that is designed to serve as a backup split. To do this, we can change the step parameter for the *goto step* command from 10 to 6 (so that the command reads *goto Step 6.....*). In such a case, control is passed from Step 1 to the *check-backup split* step (Step 6). Inasmuch as this queuing step is included within a continuous loop of steps (Steps 6 through 9), continuous attempts to queue the call are now made (if necessary).

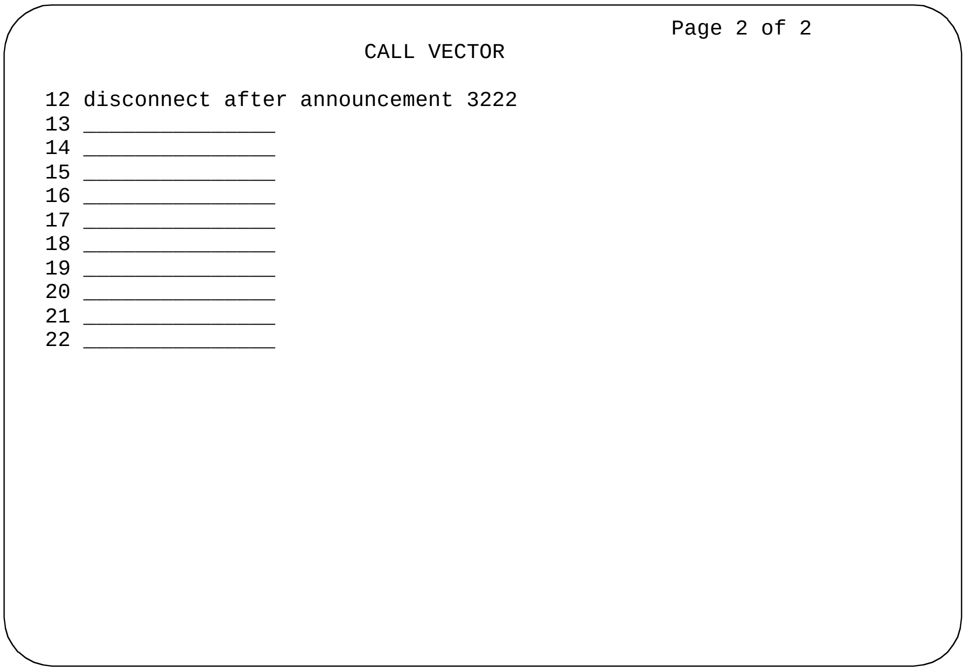
Phase 6: Checking for Non-Business Hours

If a caller calls during non-business hours, you can still provide the caller with some information for calling back during working hours by playing the appropriate recorded message. The following vector illustrates one approach in this regard. This vector would be used for a company that was open seven days a week, from 8:00 A.M to 5:00 P.M., including Saturday and Sunday.



```
                                                              Page 1 of 2
                          CALL VECTOR
Number: 27            Name: base
    Basic?y  EAS?n   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?n  LAI?n  G3V4 Adv Route?n

01 goto step 12 if time of day is all 17:00 to all 8:00
02 goto step 11 if calls queued in split 5 pri l > 10
03 queue-to main split 5 pri l
04 wait-time 10 seconds hearing ringback
05 announcement 2771
06 wait-time 10 seconds hearing music
07 check-backup split 7 pri m if calls-queued < 5
08 wait-time 60 seconds hearing music
09 announcement 2881
10 goto step 6 if unconditionally
11 busy
```

Figure 2-9. Checking for Non-Business Hours (Screen 1 of 2)



```
                                              Page 2 of 2
                              CALL VECTOR

12 disconnect after announcement 3222
13 _______________
14 _______________
15 _______________
16 _______________
17 _______________
18 _______________
19 _______________
20 _______________
21 _______________
22 _______________
```

Figure 2-10. Checking for Non-Business Hours (Screen 2 of 2)

NOTE:
Announcement 3222 could contain this message: “We’re sorry. Our office is closed. Please call back any day between 8:00 A.M. and 5:00 P.M.’’)

The *goto step* command in Step 1 checks if the call arrives during non-business hours. Specifically, if the call arrives between 5:00 P.M. and 8:00 A.M. on any day of the week, the command passes control to Step 12. The *disconnect* command in Step 12 includes and provides an announcement that first gives the caller the appropriate information and then advises him or her to call back at the appropriate time. The command then disconnects the caller.

On the other hand, if the call does not arrive during the specified hours, control is passed to Step 2, and vector processing continues. On step 2, split 5 is checked for calls waiting at priority low and above (that is, for all priorities).

NOTE:
As an alternative to disconnecting callers who place a call during non-business hours, you can allow callers to leave a message by including the *messaging split* command within the vector. See Chapter 4 for more details.

Benefits of Call Vectoring

Coupled with Automatic Call Distribution (ACD), Call Vectoring enables telephone calls to be processed at a faster rate within an intelligent, “real-time” system. As a result, Call Vectoring provides an appreciable cost saving to the user.

The following table summarizes the benefits that Call Vectoring provides. The last column in the table identifies the vector(s) [via the appropriate screen(s)] in the reference portion of the manual that illustrate(s) these benefits.

NOTE:
If the dashed number in the last column is followed by '"F)," the number is a figure number.

Table 2-1. Benefits of Call Vectoring

Category	Call Vectoring Benefits	Figure
Call Treatment	Implement special treatment based on the time of day and the day of the week (for example, providing night service).	4-16, 11-1
	Automatically change treatment according to either how long the call has been waiting or to changing traffic or staffing conditions.	4-13, 4-15, 8-2, 8-3, 11-3, 11-4
	Provide appropriate caller feedback during waiting (for example, music or announcements during heavy calling periods).	4-7, 4-10
	Provide multiple and/or recurring informational or delay announcements that are selected according to the time of day/day of the week, call volume, or staffing conditions.	4-12, 4-13, 8-2, 11-1
	Provide 24 hour/day, 7 day/week automated information announcements.	4-4, 4-5
	Remove selected calls (by providing busy or disconnect)	4-7, 4-8, 4-9, 4-11, 4-15
	Set up and test, in advance, special call treatments for events such as sales, advertising campaigns, holidays, snow days, etc.	4-4, 4-8
	Provide the caller with a menu of choices.	5-6, 5-7, 5-11, 11-3, 11-5
	Execute a VRU script	4-9, 11-4
	Notify callers of their expected delay in queue	6-3, 6-4
	Provide multiple audio/music sources	4-6

Table 2-1. Benefits of Call Vectoring

Category	Call Vectoring Benefits	Figure
Call Routing	Queue calls to up to three splits simultaneously, consequently improving the average speed of answer and agent productivity.	4-10, 11-1, 11-4
	Implement routing to local or distant destinations.	4-13, 5-1, 5-2, 8-2, 11-2, 11-3, 11-4, 11-5
	Connect callers to a voice-mail or messaging system either automatically or at their request.	4-11, 4-12, 11-3, 11-5
	Reduce call transfers by accurately routing callers to the desired destination.	5-1, 5-2, 11-3
	Provide up to four ACD queuing priority levels and the ability to change the queuing priority dynamically, as a result, providing faster service for selected callers.	11-1, 11-3, 11-4
	Reduce agent and/or attendant staffing requirements by: (1) automating some tasks; (2) reducing caller hold time; (3) having agents in one split service multiple call types.	4-4, 4-5, 5-1, 5-11, 5-12, 11-2, 11-3
	Intelligently balance ACD call loads across multiple locations.	8-2, 8-3, 8-5, 11-5
	Determine the expected wait time in ACD queues.	6-3, 6-5
	Limit the number of simultaneous incoming trunk calls to a VDN.	6-7
	Route calls based on the caller's ANI or the type of the originating line.	7-1, 7-3, 7-4
Information Collection	Provide customized and/or personalized call treatment via information collection and messaging.	5-1, 5-6, 5-11, 11-2, 11-3, 11-5
	Collect information for use by an adjunct or by agent display.	5-7, 11-5

Fundamentals of Call Vectoring

3

Introduction

The manner in which a call is processed depends upon a number of components within both the DEFINITY System and the Call Vectoring software. Some of these components include the following:

- Resources available to process a call (for example, agents, splits, software, hardware)
- Vector control flow
- Commands used within the relevant vector(s)

A prudent utilization of these components will produce an effective means of processing telephone calls. This chapter discusses these components, which constitute the fundamentals of Call Vectoring.

Call Management

When a call is placed to a PBX with Call Vectoring activated, the call accesses the appropriate vector(s) via a Vector Directory Number (VDN). A VDN is a ‘‘soft’’ extension number not assigned to an equipment location. Each VDN maps to one vector, and several VDNs may map to the same vector. (The VDN is fully discussed later in this chapter).

Once the call goes to a vector, the call’s routing and treatment are determined by the commands in the vector. Processing starts at the first step and then proceeds usually sequentially through the vector. Any steps left blank are skipped, and the process automatically stops after the last step in the vector.

Call Vectoring allows the chaining of vector steps and vectors. Accordingly, one vector can direct the call to another vector or VDN, which in turn can direct the call to yet another vector, etc. Note, however, that a maximum of 1,000 vector steps can be executed for any call. When a call enters vector processing, a loop counter keeps track of the number of vector steps executed. If the loop counter exceeds 1,000, a *stop* command is executed.

When a call is delivered to an available agent, the agent can see the information associated with the VDN (for example, the VDN name) on his/her display (if present) and, as a result, can respond to the call with knowledge of the service or response required.

In the real world, of course, not every call placed to a site is immediately answered by an agent. (The customer often has fewer agents than the maximum simultaneous call capacity. Therefore, calls will have to be queued.) The following sections discuss how calls are routed and/or queued via Call Vectoring. Subsequent sections discuss agent states, priority levels, caller feedback, and caller control.

Call Flow

Calls enter a vector and execute steps sequentially beginning with step 1, unless there is a *goto* step. Most steps take microseconds to execute. The exception is steps with *announcement*, *wait-time* and *collect digits* commands. A one second wait occurs after every seven executed steps unless an explicit wait has occurred. Note that *wait-time* with 0 seconds is not an explicit wait.

Call Vectoring uses several call flow methods to redirect and/or queue calls. These methods involve the use of the Call Vectoring commands, which are described later in this chapter. The methods for queuing and redirecting calls include the following:

- **Multiple split queuing** allows a call to queue to up to three splits.
- **Intraflow** allows calls unanswered at a split within a predefined time frame to be redirected to one or more other splits on the same switch. If redirection depends upon a condition to be tested, the process is referred to as *conditional intraflow*.
- **Interflow** allows calls directed to a vector to be redirected to an external or non-local split destination. This destination is represented by a number programmed in the relevant vector. Calls can be routed to an attendant (or attendant queue), a local extension, a remote [that is, Uniform Dialing Plan (UDP)] extension, an external number, or a VDN.
- **Look-Ahead Interflow** can be implemented for call centers with multiple ACD locations connected via an ISDN-PRI. This method allows a call to interflow only to a remote location that is better equipped to handle the call. Look-Ahead Interflow (LAI) can occur only when the proper conditions at the receiving switch are met.

- **Adjunct Routing** allows the switch to request a routing destination from an adjunct processor via ASAI. When this feature is enabled, the switch sends the ASAI adjunct a message containing information about the calling party. The adjunct uses this information to determine, from its databases, the best place for the switch to send the call. The adjunct then passes this routing information back to the switch.

Each of these call control flow methods is fully discussed in the upcoming chapters.

Caller Control

Call Vectoring allows for the temporary transfer of call management control to the caller via several means, as follows:

- **Caller-Selected Routing**. If Call Prompting is enabled, the customer can prompt the caller to input information in the form of dialed digits from a touch-tone or local rotary telephone. (A recorded announcement is usually used for prompting purposes.) Once the caller inputs the digits, the call is efficiently and accurately routed to the correct department or destination. This procedure can significantly reduce the number of transferred calls and thus better satisfy the caller’s needs.
- **Messaging** is a means of satisfying customer demand during peak calling periods. The caller can leave a voice message for the customer in the event that the call cannot be or has not yet been answered. When messaging is enabled, control is eventually passed to the Audio Information Exchange (AUDIX) or message service split. AUDIX is a voice mail adjunct that allows a customer to record, edit, forward, and retrieve voice messages to and from callers.

Subsequent chapters discuss these procedures in more detail.

Call Queuing to Splits

Basic Call Vectoring is used primarily to control the call activity of ACD splits. Basic Call Vectoring can queue calls to up to three such splits simultaneously at any one of four priority levels. This process is called *multiple split queuing*. The first split to which a call is queued via this process is called the *main split*, while the second split and the third split (if necessary) are called *backup splits*.

Multiple split queuing serves to provide better service to the caller, and it also enables a better utilization of agents. A call remains queued until either vector processing terminates or the call reaches an agent or another destination. (Vector processing termination is discussed later in this chapter.)

When an agent becomes available in any split to which the call is queued, the following events take place:

- Call begins alerting the agent (or connects if it is automatically answered).

- Call is removed from any other queues. Announcements, music, ringback, or other audio source are also removed.
- Vector processing terminates.

Note that these actions always happen *immediately*, even if the caller is receiving call treatment (for example, hearing an announcement). (Call treatments are discussed later in this chapter.)

Multiple split queuing is illustrated in Chapter 4, "Basic Call Vectoring".

Split Queue Priority Levels

If a call is queued without Call Vectoring enabled, the call is tracked at one of two priority levels: *Medium* and *High*. On the other hand, if a call is queued via Call Vectoring, the call can be assigned one of four priority levels: *Top, High, Medium,* and *Low*. Within each priority level, calls are processed sequentially as they arrive. This is equivalent to a *FIFO* (first-in, first-out) order. A vector can be administered to queue calls at any of the four priority levels.

NOTE:
A direct agent call is always given the highest priority and, as a result, it is always delivered before a call that is directed to a split. (A direct agent call is an ACD call that is directed to a specific ACD agent rather than to any available ACD agent in the split. See "Direct Agent Calling" on page 10-28 for more information.)

NOTE:
If a call is already queued to one or more splits that are currently intended to serve as backup splits, the call could be requeued at the new priority level indicated in the command step. (For further details on requeuing, see Appendix A.)

Agent Work Mode

Call Vectoring can make call management decisions according to real-time agent work modes. These states, *available-agents* and *staffed-agents*, can appear as conditions within the *check backup split* and *goto* Call Vectoring commands (that is, the commands can check for the number of available agents or staffed agents).

For ACD splits, *staffed-agents* represents the number of agents logged-in. *Available-agents* represents the number of agents logged-in *and* ready to receive an ACD call.

For non-ACD hunt groups, *staffed-agents* is synonymous with *administered*, since hunt groups do not have any log-in, log-out, or work modes. *Available-agents* is the number of agents ready to receive a hunt group call.

For ACD calls, an agent's state is further defined by the relevant *work mode*. The following list describes these modes:

- *After-Call-Work Mode* makes the agent unavailable to receive any ACD calls for any split. This mode can be used when the agent is doing ACD call-related work.
- *Auto-In Work Mode* makes the agent available to receive calls and allows the agent to receive a new ACD call immediately after disconnecting from the previous call. When Multiple Call Handling is enabled an agent in Auto-In Work Mode can receive additional ACD calls while still active on a call.
- *Auxiliary-Work Mode* makes the agent unavailable to receive any ACD calls for the specified split. This mode can be used when an agent is performing non-ACD activities, such as going on a break.
- *Manual-In Work Mode* makes the agent available to receive calls and automatically puts the agent into the *After Call Work Mode* after disconnecting from an ACD call. When Multiple Call Handling is enabled an agent in Manual-In Work Mode can receive additional ACD calls while still active on a call.

See the *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204, for a more complete description of agent work modes and Multiple Call Handling.

Calling Party Feedback

The initial feedback a caller hears as the call is being processed by a vector depends upon the origin classification of the call, which can be one of the following:

- Internal (internal call from another Generic 3 user)
- Non-CO (incoming call over a DID or tie trunk over which incoming digits are received)
- CO (incoming call over a CO or automatic type tie trunk over which no digits are received)

For an internal or a non-CO call, the caller hears silence until one of the following vector steps is reached:

- Wait with system music, ringback, or an alternate audio/music source (Caller hears system music, ringing, or the music or audio associated with an administered port.)
- Announcement (Caller hears the announcement.)
- Busy (Caller hears a busy tone.)
- Call alerting an agent or at a station (Caller hears ringing or the agent answering the call.)

For a CO call, the caller hears CO ringback until one of the following vector steps is reached:

- Announcement (Caller hears the announcement.)
- Wait with system music or alternate audio/music source (Caller hears system music, or the music or audio associated with an administered port.)
- Call answered (Caller hears the agent or voice response answering the call.)

For a CO call for which answer supervision has already been supplied (via the processing of an announcement or the issuing of a *wait-time* command), the caller may hear any of the following:

- Announcement when any *announcement* command is processed.
- Ringback, silence, system music, or an alternate audio/music source when a *wait-time* command is processed.
- Busy when a *busy* command is processed.
- Ringback when the call is alerting a station.

Regardless of the call's origin, the caller can expect to hear different forms of the feedback described in this section as the relevant vector steps are processed. Examples of how subsequent caller feedback is provided in the vector appear in Chapter 4, "Basic Call Vectoring" and in several of the following chapters.

Dialed Number Identification Service (DNIS)

In the traditional ACD arrangement, each agent in a given split is trained to answer calls relevant to one specific purpose in an efficient and professional manner. However, ACD managers have recognized the need to enhance this arrangement in which each split is limited to a single call-answering task.

To this end, there is now a split arrangement available in which each group of agents is proficient in dealing with several types of calls. The intent is to service multiple call types with the use of fewer agents overall and with less administrative intervention by the ACD manager. Usual economies of scale come into play here. For example, where five agents might be needed in each of three smaller splits (15 agents total) to handle three types of calls, only 11 or 12 agents might be needed in the combined split.

To aid in providing capabilities such as the one just presented, a network service known as Dialed Number Identification Service (DNIS) is available. DNIS enables a unique multidigit number that is based on the dialed number (of usually four digits) to be associated with the call (sent to a customer's telephone, sent to a host computer with ASAI applications, used to provide different treatments for the call, etc.). The number that is sent depends upon the telephone number dialed by the caller. Each DNIS number in the customer's telephone system can be programmed to route to an ACD split comprised of agents who are proficient in handling several types of calls.

Call Vectoring takes the DNIS number from the network and interprets this number as a VDN. When the call is delivered to the agent terminal, the unique name assigned to the particular VDN is displayed on the agent's terminal. This allows the agent to know the specific purpose of the call. As a result, the agent can answer with the appropriate greeting and be immediately prepared to service the customer.

Vector Processing

If Call Vectoring is in effect, telephone calls are processed by one or more programmed sequences of command steps called vectors.

The following sections provide a general overview of vector processing. To this end, the following topics are discussed:

- Vector Directory Number (VDN)
- Vector control flow
- Programming capabilities

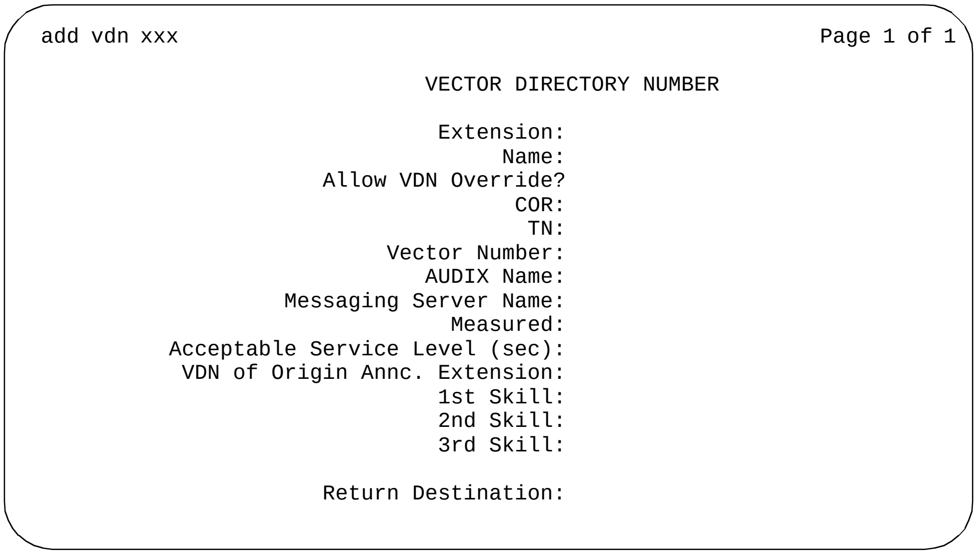
Vector Directory Number

Within Call Vectoring, calls access the appropriate vector(s) via a Vector Directory Number (VDN). A VDN is a ''soft'' extension number that is not assigned to an equipment location. In effect, the digits dialed by a caller or sent to the DEFINITY system from an external network are translated within the system as a VDN.

The VDN points to the vector, and it defines the service desired by the caller. The VDN also serves as the application number, and it allows for specific call-handling and agent-handling statistical reporting within both the DEFINITY Basic Call Management System (BCMS) and the CentreVu™ Call Management System (CMS) for each application handled by the call center.

VDNs are assigned to different vectors for different services or applications that require specific treatments. Any number of VDNs can be assigned to the same vector. As a result, the same sequence of treatments can be given to calls that reach the PBX/ACD via different numbers or from different locations.

The VDN has several properties. These properties are administered by the System Manager on the Vector Directory Number administration form.



```
add vdn xxx                                                   Page 1 of 1
                                VECTOR DIRECTORY NUMBER

                                 Extension:
                                      Name:
                        Allow VDN Override?
                                       COR:
                                        TN:
                             Vector Number:
                                AUDIX Name:
                     Messaging Server Name:
                                  Measured:
           Acceptable Service Level (sec):
            VDN of Origin Annc. Extension:
                                 1st Skill:
                                 2nd Skill:
                                 3rd Skill:

                        Return Destination:
```

- **Extension**. Extension number used to identify the VDN.
- **Name**. Name that is associated with the VDN. This name, which is shown on the agent’s display, is optional and can contain up to 15 characters.
- **Allow VDN Override**. Option that allows the name and other attributes of a subsequently routed to VDN to be used instead of the name and attributes of the current VDN.
- **COR (Class of Restriction)**. 1- or 2-digit number that specifies the COR of the VDN.

  NOTE:
  As a security measure, you can deny incoming callers access to outgoing facility paths by configuring the COR of the VDN to prohibit outgoing access. For details, refer to the *GBCS Products Security Handbook*, 555-025-600.

- **TN**. The Tenant Partition Number for this VDN.
- **Vector Number**. Vector-assigned number that determines which vector is activated when a call comes into a VDN. Several VDNs may send calls to the same vector.
- **AUDIX Name**. Only displayed for G3r. The name of the AUDIX that is associated with the VDN as it appears on the Adjunct Names form.
- **Messaging Server Name**. Only displayed for G3r. The name of the messaging server as it appears in the Adjunct names form.
- **Measured**. Indicates whether data about the VDN is being measured internally by BCMS, externally by CMS, by both BCMS and CMS, or not at all.

- **Acceptable Service Level**. The number of seconds within which calls to this VDN should be answered.
- **VDN of Origin Annc. Extension**. The extension number of the VDN of Origin announcement. Only displayed if VDN of Origin Announcement is optioned.
- **1st/2nd/3rd Skill**. Only displayed with Expert Agent Selection (EAS). Gives the skill numbers associated with the VDN.
- **Return Destination**. The VDN extension number to which an incoming trunk call will be routed if it returns to vector processing after the agent drops the call. Only displayed if VDN Return Destination is optioned.

VDNs can be preassigned to incoming (automatic) trunk groups, or they can be sent in digit form to the PBX/ACD by the public or a private network. The digits sent to the PBX/ACD can come from the serving Central Office (CO) or toll office via the Direct Inward Dialing (DID) feature or DNIS. The digits can also come from another PBX/ACD location via dial-repeating tie trunks, or they can be dialed by an internal caller. For a non-ISDN call, the last four digits of the number are sent to the PBX, while for an ISDN call, the entire 10-digit number is sent.

The last four or five digits of the destination address passed to the PBX/ACD on a DID/DNIS or on a dial tie-trunk call comprise the VDN. Automatic trunks do not pass destination address digits. Instead, each such trunk always routes to a specific incoming destination that is programmed for the corresponding automatic trunk group. The destination can be an attendant queue, an extension, a hunt group number, or a VDN.

VDN Override

VDN Override is an option that allows information about a subsequently routed to VDN (if any) to be used instead of the information about the current VDN. This information includes:

- The name of the subsequent VDN
- Skill sets
- Messaging split command with the “active” entry
- VDN of Origin Announcement
- Tenant number

NOTE:
Throughout this document the “active” VDN is the active called VDN as modified by VDN override rules. The “latest” VDN is the most recent VDN to which the call was routed.

VDN Override can be used in conjunction with a vector that prompts the caller for a particular service. Let’s say, for example, a call is placed to an automobile dealer. Like any such dealer, this one consists of several departments, including

“Sales’’ and ‘‘Parts.’’ Let’s presume the caller wants to talk to someone in “Sales.’’ In such a case, the call comes into the ‘‘Main’’ vector (whose VDN name is ‘‘Main’’) and is eventually routed to the “Sales’’ vector (whose VDN name is “Sales’’). If VDN Override is assigned to the ‘‘Main’’ VDN, the ‘‘Sales’’ VDN name appears on the agent’s display when the call is finally connected to the agent. This process is illustrated in Figure 3-1. In this example, the “Sales” VDN is the active VDN as well as the latest VDN. If VDN override had not been assigned to the “Main” VDN, the agent’s display would have shown “Main.” In this case, “Main” would be the active VDN while “Sales” would be the latest VDN.

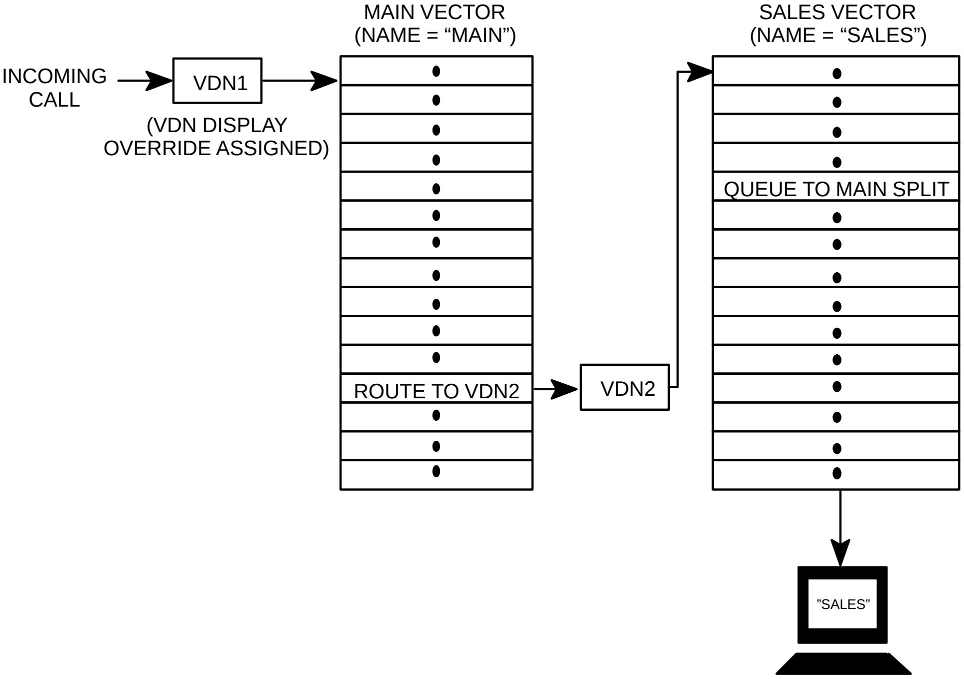


Figure 3-1. VDN Override Assigned to Originally Called VDN

VDN in a Coverage Path

A VDN can be assigned as the last point in a coverage path. Whenever a VDN is assigned as such, a call goes to coverage and can then be processed by Call Vectoring or Call Prompting (if either is enabled). Accordingly, the Call Coverage treatment for the call is extended (that is, coverage can be sent to an external location, or the type of coverage can be controlled by the caller).

VDN in a coverage path is used for a number of applications, including:

- Sending direct agent calls or personal calls to an agent (EAS required)
- Routing coverage calls off-premises via the *route-to* command

- Serving as a coverage point for specific call operations (for example, sending calls to a secretary during the day and to AUDIX at night).

VDN as a coverage point is illustrated in Chapter 4, "Basic Call Vectoring".

Service Observing VDNs

The Service Observing feature provides the option of being able to observe VDNs with G3V3 and later releases. With this option an observer selects a specific VDN and bridges onto calls (one call at a time) that have just started vector processing for that VDN. The observer hears all tones, announcements, music, and speech that the caller and the agent hear and say, including Call Prompting and caller dialing. Also, the observer hears VDN of Origin announcements. Once the system makes an observing connection to a call in vector processing, it maintains the connection throughout the life of the call until the call is disconnected or until the observer hangs up. This is true even if the call is routed or transferred externally. See “Service Observing” in the *DEFINITY Communications System Generic 3 Feature Description,* 555-230-204 for complete information about Service Observing VDNs.

Vector Control Flow

Vector Processing starts at the first step in the vector and then proceeds sequentially through the vector unless a *goto* command is encountered. Any steps left blank are skipped, and the process automatically stops after the last step in the vector.

The Call Vectoring ‘‘programming language’’ provides three types of ‘‘control flow’’ that serve to pass vector-processing control from one vector step to another. Control flow types are described in the following list.

- **Sequential flow** passes vector-processing control from the current vector step to the following step. Most vector commands allow for a sequential flow through the vector.

  NOTE:
  Any vector command that fails automatically passes control to the following step. The success and/or failure criteria for the Call Vectoring commands is discussed in Appendix A.

- **Unconditional branching** *unconditionally* passes control from the current vector step to either a preceding and/or succeeding vector step or to another vector (for example, *goto step 6 if unconditionally*).
- **Conditional branching** *conditionally* passes control from the current vector step to either a preceding and/or succeeding vector step or to a different vector. This type of branching is based on the testing of threshold conditions (for example, *goto vector 29 if staffed-agents in split 6 < 1*).

Each of these control flow types is fully described in the upcoming chapters.

NOTE:
Call Vectoring has an execution limit of 1,000 steps. Once a call enters vector processing, a ‘‘loop counter’’ keeps track of the number of vector steps executed. If the loop counter exceeds 1,000, a *stop* command is executed.

NOTE:
An implicit wait of one second is provided after every seven vector steps if vector processing is not suspended during any one of these steps (see the *wait-time* command manual pages in Appendix A, "Call Vectoring Commands").

Termination vs. Stopping

For the purposes of this guide, the expression *vector processing terminates* means a call has completely left vector processing. This occurs when the call is alerting at an agent’s station, is abandoned by the calling party, receives a forced disconnect or a forced busy, or is successfully routed to an extension or to an off-premises number.

It is important to differentiate between *vector processing termination* and *stopping*, the latter of which is caused by the *stop* command or by the execution of the final step in the vector. Whereas vector processing termination removes the call from the queue if the call is queued, the *stop* command prevents the processing of new vector steps *but leaves the call in queue* as the calling party receives feedback, such as ringback. If vector processing stops and the call is not queued, the call is dropped.

Vector processing termination and the *stop* command are discussed and illustrated later in this guide.

Programming Capabilities

The Call Vectoring commands can perform a number of functions relevant to processing telephone calls. A brief explanation for each of these functions follows.

- **Providing call treatments**. The caller can be provided with a recorded announcement explaining that, at the moment, an agent cannot answer the call for some reason (for example, there are no agents available, the work day is over, etc.). Announcements also provide the caller with instructions and encouragement. Also, audible feedback (silence, ringback, system music, or an alternate audio or music source) or a busy tone can be provided to the caller. Provisions can also be made to delay vector processing a specific number of seconds before the next vector step is executed. Also, when necessary, the call can be disconnected. Finally, a session with voice mail (AUDIX) can be initiated.

- **Routing calls**. Calls not immediately answered by an agent can be queued to one or more splits, as explained earlier in this chapter. A caller can also leave a recorded message if he or she chooses to do so. Finally, a call can be routed to a number programmed in the vector or to digits collected from the caller.
- **Branching/programming**. Branches can be made from one vector step to another such step or to another vector. This can be done unconditionally as well as conditionally. Conditional branching is done according to a number of conditions (for example, number of available agents in a split, number of calls in a split queue, the number of the phone the call is made from, etc.). Finally, vector processing can be stopped when necessary.
- **Collecting and acting on information**. Optionally, touch-tone digits can be collected and serve as the basis for further vector processing (for example, a specific agent can be reached via touch-tone digit(s) entered by the caller).
- **Executing VRU scripts**. Voice scripts housed within a Voice Response Unit (VRU) can be executed for the caller. Voice scripts provide the caller with information or instructions, and the caller can often make an appropriate response thereto (by, for example, entering touch-tone digits).

Command Summary

This section lists and describes the commands used by the Call Vectoring features. The list is meant to help familiarize the reader with these commands. The commands are explained further in Chapter 4 through Chapter 9 and also in Appendix A.

- **Adjunct Routing** is available only when the CallVisor ASAI capabilities and Basic Call Vectoring are optioned. The command causes a message to be sent to an ASAI adjunct requesting routing instructions.
- **Announcement** provides the caller with a recorded announcement.
- **Busy** gives the caller a busy signal and causes termination of vector processing.
- **Check-Backup Split** conditionally checks the status of a split for possible termination of the call to that split. The command either connects to an agent in the split or puts the call into the split’s queue (at the specified queuing priority level) if the condition specified as part of the command is met. A call may be queued to up to three different splits simultaneously.
- **Collect Digits** allows the user to enter up to 16 digits from a touch-tone phone. An optional announcement may be played first.
- **Converse-on Split** integrates Voice Response Units (VRUs) with the DEFINITY switch. Specifically, the command allows voice response scripts to be executed while the call remains in queue, and it allows the passing of data between the DEFINITY switch and the VRU.

- **Disconnect** ends treatment of a call and removes the call from the switch. The command also allows the optional assignment of an announcement that will play immediately before the disconnect.
- **Goto Step** is a branching step that allows conditional or unconditional movement to a preceding or succeeding step in the vector. Conditional branching is determined by a number of factors (for example, number of calls queued in the split, number of staffed agents in the split, etc.).
- **Goto Vector** is a branching step that allows conditional or unconditional movement to another vector. Conditional branching is determined by a number of factors (for example, number of calls queued in the split, number of staffed agents in the split, etc.).
- **Messaging Split** allows the caller to leave a message for a specified extension or the VDN extension (default).
- **Queue-to Main Split** unconditionally queues a call to a split and assigns a queuing priority level to the call in case no agents are available. A call sent with this command either connects to an agent in the split or enters the split's queue.
- **Route-to Digits** routes the call to the destination specified by a set of digits collected from the caller by the previous *collect digits* step.
- **Route-to Number** routes the call to the destination specified by the administered digit string.
- **Stop** terminates the processing of any subsequent vector steps.
- **Wait-Time** is used to specify whether the caller will hear ringback, system music, silence, or an alternate audio or music source while the call is waiting in queue. The command also delays the processing of the next vector step by the specified delay time that is included in the command's syntax.

NOTE:
Complete operation details for the *route-to* commands are included in Appendix G.

Condition Testing within the Commands

As was mentioned in the previous section, a number of the Call Vectoring commands are implemented according to a tested condition that comprises part of the command. In other words, for example, if the condition expressed in the command is *true*, the command action is executed. On the other hand, if the condition expressed in the command is *false*, the command action is *not* implemented, and the next vector step is processed.

The following list provides a set of conditions that might comprise the conditional portion of a Call Vectoring command. The available set of conditions is dependent upon the DEFINITY switch version number and the optional features that have been enabled. See Appendix L, "V4 Feature Availability" for more information. Refer to Appendix A for the syntax of each condition.

- Number of staffed agents in a split (explained earlier in this chapter)
- Number of available agents in a split (explained earlier in this chapter)
- Number of calls queued at a given priority to a split
- Amount of time that the oldest call has been waiting in a split
- Average Speed of Answer for a split or a VDN
- Expected Wait Time for a split or for a call that has entered vector processing
- Number of calls active in a VDN
- Caller identity (ANI)
- Type of originating line (ii-digits)
- Digits entered by the caller
- Time-of-day and day of the week that the call is placed

NOTE:
The syntax for this condition can be illustrated by a couple of examples, as follows: *mon 8:01 to fri 17:00* means ‘‘anytime between 8:01 A.M. Monday through 5:00 P.M. Friday,’’ and *all 17:00 to all 8:00* means ‘‘between 5:00 P.M. and 8:00 A.M. on any day of the week.’’)

Depending upon the condition, specific comparison operators ([for example, < (less than), > (greater than), = (equal to), <= (less than or equal to), >= (greater than or equal to), <> (not equal to)]) and a threshold (that is, a range of accepted numerical entries) might be in effect. Note that some of these comparators are available only with G3V4 and later releases. Again, refer to Appendix A.

The chapters on the Call Vectoring features illustrate condition checking in more detail.

Basic Call Vectoring

4

Introduction

Basic Call Vectoring allows you to ‘‘program’’ the type of treatment a telephone call receives. You can program accordingly by using a set of vector commands.

Vector commands can direct calls to various destinations, such as adjuncts and splits. The commands can also direct calls to various treatments, such as announcements, a forced disconnect, a forced busy, or a delay treatment.

Command Set

The following table illustrates the commands used for Basic Call Vectoring.

Table 4-1. Basic Call Vectoring Command Set

Command Category	Action Taken	Command
TREATMENT	Play an announcement.	*announcement*
	Delay with audible feedback of silence, ringback, system music, or alternate audio/music source.	*wait-time*
	Play a busy tone and stop vector processing.	*busy*
	Disconnect the call.	*disconnect*
	Execute a Voice Response Unit (VRU) script.	*converse-on split*
ROUTING	Queue the call to an ACD split	*queue-to main split*
	Queue the call to a backup ACD split.	*check-backup split*
	Leave a message.	*messaging split*
	Route the call to a number programmed in the vector or to a Service Observing Feature Access Code.	*route-to number*
	Send to an adjunct a message requesting routing instructions for the call.	*adjunct routing*
BRANCHING/ PROGRAMMING	Go to a vector step.	*goto step*
	Go to another vector.	*goto vector*
	Stop vector processing.	*stop*

Functions and Examples

Basic Call Vectoring allows the user to employ a number of different functions to process telephone calls. These functions include the following:

- Providing call treatments
- Routing calls
- Branching/Programming

The following sections explain these functions.

Providing Call Treatments

In this guide, the term *treatment* is used to indicate the type of feedback the caller receives if the caller is not immediately connected to an agent, or if the call center is too busy or not in operation. Basic Call Vectoring provides several types of treatment, as follows:

- Announcements
- Delays with audible feedback
- Busy tone
- Disconnect
- Voice Response Scripts

The sections that follow explain these treatments.

Announcements

If a caller is not able to connect to an agent *immediately*, it is logical to provide the caller with a recorded message in order to accomplish one of the following, depending upon the circumstances:

- Encourage the caller to continue to hold the line.
- Provide the caller with information that will satisfy his or her needs, thereby keeping him or her from waiting a long time for service and also allowing him or her to hang up as soon as possible.

Such a recorded message is referred to as an *announcement*, and it is provided via the *announcement* command.

Whenever a call is connected to an announcement, any previous treatment is discontinued, and answer supervision is sent (unless it has already been provided). If, during an announcement, the call is moved from waiting in a split’s queue to alerting or connecting to an agent’s station, the announcement is disconnected, and the caller hears ringback. When the announcement completes and is disconnected, the caller hears silence until either a vector step with alternate treatment is processed or the call reaches an agent’s station.

Announcements can be classified into three groups, as follows:

- Delay announcements
- Forced announcements
- Information announcements

Delay Announcements

Here is an example of a delay announcement:

```
announcement 2556 (‘‘All our agents are busy.
Please hold.’’)
```

Figure 4-1. Delay Announcement

If the caller does as suggested but ends up waiting an appreciable amount of time without receiving further feedback, he or she may tire of waiting and hang up. To keep the caller on the phone at least a little longer, a supplementary delay announcement similar to the one following might be used:

```
announcement 2557 (‘‘Thanks for holding.  All
our agents are still busy.  Please hold.’’)
```

Figure 4-2. Supplementary Delay Announcement

A delay announcement is usually coupled with a delay step, which is provided by the *wait-time* command (discussed later).

The customer should incorporate as many supplementary delay announcements as he or she deems necessary, given the resources available.

Forced Announcements

There are times when the customer may find it advantageous to have the agents *not* answer calls. Usually, this option is exercised whenever the customer anticipates a barrage of calls concerning an emergency or a service problem of which the customer is already aware. Accordingly, the customer can incorporate an appropriate announcement as the very first step in the vector. Such an announcement is referred to as a *forced announcement*. Here’s an example.

```
announcement 1050 (‘‘We are aware of the current
situation and are working to rectify the problem.  If your
call is not urgent, please call back later.’’)
```

Figure 4-3. Forced Announcement

Information Announcements

Under certain circumstances, the customer may find it necessary to provide the caller with recorded information that, by its very content, resolves a problem with such finality that the caller feels no need to follow up on his or her call. Such a recorded message is referred to as an *information announcement*. An example follows:

```
disconnect after announcement 2918 (‘‘Today has
been declared a snow day.  Please report for work tomorrow
at 8 A.M.’’)
```

Figure 4-4. Information Announcement

Note that the *disconnect* command is used with the announcement. After the announcement, the caller is disconnected, since he or she need not stay on the line any longer.

Delays with Audible Feedback

In presenting an example of a delay announcement earlier in this chapter, we mentioned that this type of announcement is usually coupled with a delay step. A delay step is provided by the *wait-time* command, which allows the caller to remain on hold for at least the number of seconds indicated in the command.

Let’s take another look at our delay announcement. However, this time, let’s couple the announcement with a delay step:

```
announcement 2556 (‘‘All of our agents are busy.
Please hold.’’)
wait-time 20 seconds hearing music
```

Figure 4-5. Delay with Audible Feedback

Here, the caller is allowed to wait at least 20 seconds for the call to be answered by an agent. During this wait period, the caller is provided with system music, which is one type of feedback available via the *wait-time* command.

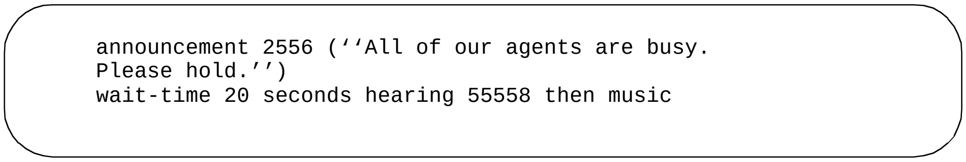
If the delay step is the final effective step in the vector, the audible feedback continues beyond the specified duration. (A “final effective step” in a vector is either the last vector step or a vector step that is followed by a *stop* step.) Under normal circumstances, the audible feedback continues until the call is either answered or abandoned. However, if the call is not queued when vector processing stops, the call is dropped. Feedback also continues while a call is queued to a converse split, that is, any split routed to by a converse-on split command, and while data is being passed to a Voice Response Unit (VRU) (see the "Voice Response Scripts" section later in this chapter). Finally, feedback also continues during the wait period before the connection of an announcement and/or a Touch-Tone Receiver (TTR). (TTRs are used in conjunction with the Call Prompting feature and are discussed in Chapter 5.)

Multiple Audio or Music Sources

With G3V4 and later releases, you can specify an alternative audio or music source for a vector *wait-time* step. This alternative source can be any extension number that is administered on the Announcements/Audio Sources form. For instructions for entering an audio or music source on this form see *DEFINITY Communication System Generic 3 Version 4 Implementation*, 555-230-655 or *DEFINITY Communication System Generic 3 V2/V3 Implementation*, 555-230-653.

With the Multiple Audio/Music Sources feature, you can tailor the *wait-time* feedback to the interests, tastes, or requirements of the audience. You can provide specific types of music or music with overlays of advertising that relate to the service provided by the splits or skills that the vector serves. Or, additional advertising messages can be heard by the caller as they wait for an available agent.

Let’s look again at our delay announcement. However, this time let’s assign an alternate audio/music source to the *wait-time* step.



```
announcement 2556 (‘‘All of our agents are busy.
Please hold.’’)
wait-time 20 seconds hearing 55558 then music
```

Figure 4-6. Delay with Multiple Audio/Music Source Feedback

When the wait-time step is encountered, the caller is connected to extension 55558 for 20 seconds. At the end of 20 seconds the next vector step is executed. The “then” option in the wait-time step specifies what the caller will hear if the caller cannot be connected to the specified source. Or, when the call is waiting in queue, the “then” option specifies what the call will hear if the call is not answered in 20 seconds. In this example, if the call is not answered in 20 seconds, the caller will hear system music until a subsequent *announcement, busy, collect, converse-on, disconnect* or *wait-time* step is encountered.

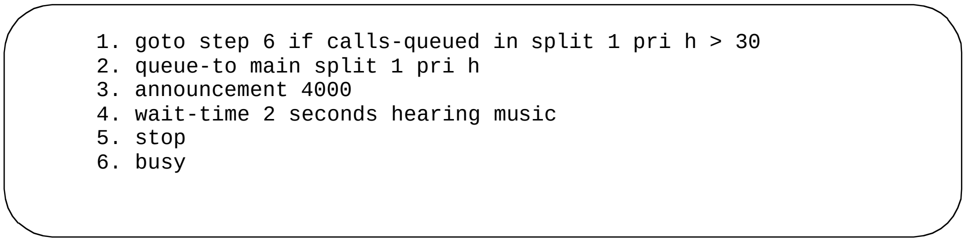
You can specify *music* (system music), *ringback*, *silence*, or *continue* for the “then” option. When continue is specified, the caller continues to hear the alternative audio or music source until it is replaced by a subsequent vector step regardless of the time specified in the *wait-time* step.

Busy Tone

A busy tone and subsequent termination of vector processing are produced via the *busy* command. An exception to this occurs on CO trunks where answer supervision has not been sent. Callers on such trunks do not hear the busy tone from the switch. Instead, these callers continue to hear ringback from the CO. The *busy* command eventually times out and drops the call after 45 seconds. With ISDN PRI, busy tone can be provided from the network switch.

The customer usually forces a busy tone to process a call that arrives at a time when there is a large number of calls queued in the main split, or when the call center is out of service or closed.

The following vector illustrates how you can use the *busy* command:



```
1. goto step 6 if calls-queued in split 1 pri h > 30
2. queue-to main split 1 pri h
3. announcement 4000
4. wait-time 2 seconds hearing music
5. stop
6. busy
```

Figure 4-7. Providing Busy Tone

In this vector, the *goto step* command in Step 1 sends call control to *busy* in Step 6 if the conditions in the former command are met. Specifically, if the number of calls queued at a high priority is greater than 30, the *busy* command is accessed.

Disconnect

The customer can opt to have a call disconnected by incorporating the *disconnect* command. However, the customer should use good judgement in this regard lest he or she unnecessarily infuriate the caller. Also, as a courtesy to the caller, an announcement should be given to the caller before he or she is disconnected under any circumstances.

The *disconnect* command itself has a built-in announcement option. We saw an example of the command when we were discussing information announcements earlier in this chapter. Here's the example again:

```
disconnect after announcement 2918 (‘‘Today has
been declared a snow day.  Please report for work tomorrow
at 8 P.M.’’)
```

Figure 4-8. Disconnecting a Call

This example presents an ideal use of the *disconnect* command. The caller is given recorded information that, by its very content, resolves a problem with such finality that the caller feels no need to follow up on his or her call.

Voice Response Scripts

Voice Response Integration (VRI) is designed to enhance the integration of AT&T's DEFINITY call center systems and to integrate DEFINITY Call Vectoring with the capabilities of voice response units (VRUs), particularly AT&T's CONVERSANT Voice Information System.

VRI can do the following:

- Execute a VRU script while retaining control of the call in DEFINITY vector processing.
- Execute a VRU script while the call remains in the split queue and retains its position in the queue.
- Pool CONVERSANT ports for multiple applications (which previously was possible only when ASAI was present.)
- Use a VRU as a flexible external announcement device.
- Pass data between the DEFINITY switch and a VRU.
- Tandem VRU data through the DEFINITY switch to an ASAI host.

The capabilities in the previous list are provided by the *converse-on split* command, which is an enhancement to the Basic Call Vectoring customer option. The *converse-on split* Call Vectoring step is specifically designed to integrate a VRU with the DEFINITY Automatic Call Distribution (ACD). VRI allows VRU capabilities to be used while keeping control of the call in the DEFINITY ACD. The inclusion of VRUs with vector processing provides the following advantages:

- Access to local and host databases.
- Validation of caller information.

- Text to speech capabilities.
- Speech recognition.
- Increased recorded announcement capacity.
- Audiotex applications.
- Interactive Voice Response (IVR) applications.
- Transaction processing applications.

One of the advantages of VRI is that it allows users to make more productive use of queuing time. For example, while the call is waiting in queue, the caller can listen to product information by using an audiotex application or by completing an interactive voice response transaction. In some cases, it may even be possible to resolve the customer’s questions while the call is in queue. This can help reduce the queuing time for all other callers during peak intervals.

In addition, when Advanced Vector Routing is enabled, the Expected Wait Time for a call can be passed to the VRU. In this way, the caller can be told how much longer they can expect to wait before their call will be answered. See "Expected Wait Time (EWT)" on page 6-2 for a complete description of the EWT feature.

During the execution of a VRU script, if the caller previously queued to an ACD split, the caller retains his/her position in queue. If an agent on the DEFINITY switch becomes available to service the call, the line to the VRU is immediately dropped, and the calling party is connected to the available agent.

Here’s an example of a vector that can access voice response scripts from a VRU.

NOTE:
Recall from Chapter 1 that one or more VDNs can access the same vector. This capability is appropriate for the following example.

```
VDN (extension=1040   name=‘‘car loans’’     vector=40)
VDN (extension=1041   name=‘‘equity loans’’  vector=40)
Vector 40
     1. goto step 10 if calls-queued in split 1 pri h > 30
     2. queue-to main split 1 pri h
     3. announcement 4000
     4. goto step 7 if calls-queued in split 1 pri h < 5
     5. wait-time 0 seconds hearing music
     6. converse-on split 11 pri h passing vdn and none
     7. wait-time 20 seconds hearing music
     8. announcement 4001
     9. goto step 7 if unconditionally
    10. busy
```

Figure 4-9. Accessing Voice Response Scripts

For this example, let’s suppose first that a caller would like to hear information concerning car loans. Let’s also assume the call is queued to split 1 (Step 2) and that vector processing proceeds to Step 6. In such a case, the *converse-on split* command in this step delivers the call to the converse split if there is a queue for the split and the queue is not full, or if a VRU port is available. (Otherwise, vector processing continues at the next vector step.) When the VRU port responds, the step then outpulses VDN 1040 to the VRU via the *passing vdn* parameter included in the command. In turn, the VRU executes the “car loans” voice response script for the caller. Note that it is important to provide a feedback step prior to the converse-on step in case there is a delay in reaching an available converse split port. In this example step 5 provides music for this purpose.

Now, let’s suppose another caller wants information concerning equity loans. In such a case, if everything proceeds according to form, VDN 1041 is outpulsed to the VRU, which in turn executes the ‘‘equity loan’’ voice response script for the caller.

In either case, while interaction with the VRU is taking place, the call remains in the appropriate split’s queue (split 1 in this example). If an agent answers the call while the voice response script is being executed, the voice response script is interrupted, the line to the VRU is dropped, and the caller is connected to the available agent. Once a voice response script starts, no further vector steps are executed until the voice response script is completed.

NOTE:
Refer to Appendix H for a detailed explanation of the call flow for converse—VRI calls.

Besides VDN extensions, the *converse-on split* command can outpulse to the VRU calling party extensions, collected (inputted) caller digits (if Call Prompting is enabled), Expected Wait Time (if Advanced Vector Routing is enabled) call queue positions, a string of a maximum of six digits or asterisks, a pound sign (#), or nothing. Further details are included in Chapter 5, "Call Prompting", Chapter 6, "Advanced Vector Routing" and in Appendix A, "Call Vectoring Commands".

NOTE:
In the previous example, the *calls-queued* condition in the second *goto* step (Step 4) in effect serves as a checkpoint for determining whether or not there is enough time for the voice response script (activated by the *converse-on* step) to be executed. Specifically, if five or more calls are queued to split 1, it is considered feasible to execute the voice response script.

Routing Calls

Basic Call Vectoring offers several means of routing telephone calls, as follows:

- Queuing calls to ACD splits

- Leaving recorded messages
- Sending calls to a vector-programmed number (that represents an internal or external destination)
- Routing calls to a Feature Access Code that will activate Service Observing
- Adjunct Routing

The following sections discuss these routing procedures (except for adjunct routing, which is fully described in Chapter 9).

Queuing Calls to ACD Splits

Calls that come into the Call Vectoring system can be queued to a maximum of three ACD splits. Two commands are used to queue calls to splits.

The *queue-to main split* command queues a call *unconditionally*. The command sends a call to a split and assigns a queuing priority level to the call in case all agents are busy.

The *check-backup split* command *conditionally* checks the status of a split for possible termination of the call to that split. The command either connects the call to an agent in the split or puts the call into the split's queue (at the specified priority level) if the condition specified as part of the command is met.

Multiple Split Queuing

The term *multiple split queuing* refers to the queuing of a call to more than one split at the same time. The following vector helps to illustrate this process.

```
1. goto step 4 if calls-queued in split 1 pri l >= 10
2. queue-to main split 1 pri t
3. wait-time 12 seconds hearing ringback
4. check-backup split 2 pri m if calls-queued < 5
5. check-backup split 3 pri m if calls-queued < 5
6. announcement 3001
7. wait-time 50 secs hearing music
8. goto step 4 if unconditionally
```

Figure 4-10. Multiple Split Queuing

To avoid completing vector processing without queuing the call to a split, it is always good practice to check a split's queue before queuing to that split. If the queue is full, alternate treatment (such as queuing to an alternate split) should be provided. In this vector, if the main split's queue (which has 10 queue slots) is full, the *goto step* command in Step 1 skips the main split and goes directly to step 4 to check the backup splits. Although calls are queued in Step 2 at a top priority, a low priority is specified in step 1 so that calls in queue at all priority levels are counted. If there are 10 or fewer calls in the main split, control is passed to Step 2, where the *queue-to main split* command queues the call to split 1. Once the call is queued, vector processing continues at the next step.

Step 4 contains a *check-backup split* command. (Recall that in the last paragraph we mention that this step is branched to if the main split queue is holding 10 or more calls.) If the call is not answered by the time Step 4 is reached, the *check-backup split* in the step attempts to queue the call to a second split. Specifically, the command first determines whether there are fewer than 5 calls queued to split 2. If so, the command then attempts to connect the call to an agent in the split. If such a connection cannot be made, the command puts the call into the split's queue (at the specified priority level). Vector processing then continues at the next step. On the other hand, if there are 5 or more calls queued to split 2, the command fails, and vector processing continues at Step 5.

Step 5 contains another *check-backup split* command and, accordingly, the process described in the previous paragraph is repeated, with one difference: the queuing attempt is made to split 3 instead of to split 2.

Except for the condition check, the circumstances under which the *check-backup split* command cannot queue a call are identical to those for the *queue-to main split* command.

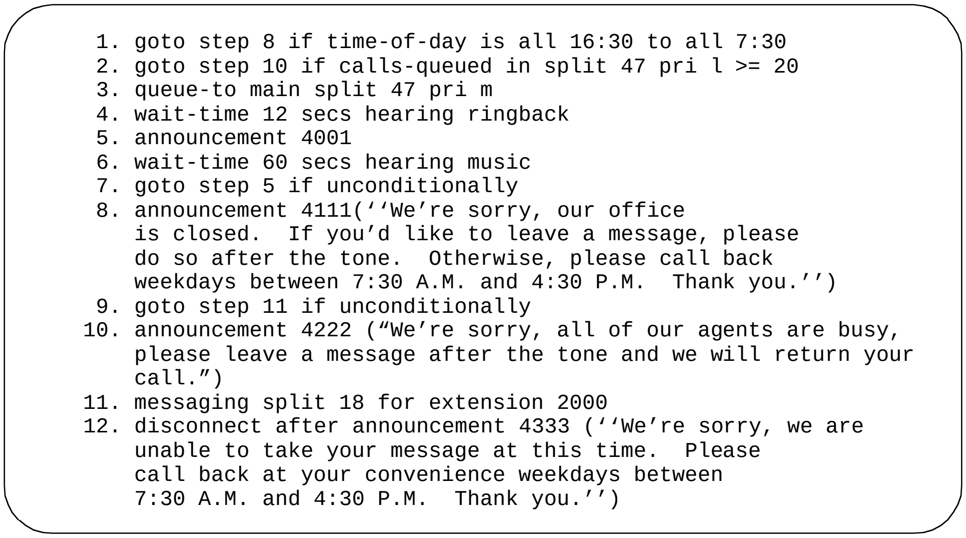
Finally, note that whenever a call is queued to a backup split, the call remains queued to the main split and/or to another backup split (if already queued to either or both of these splits). Once the call is answered in a split to which it is queued, the call is automatically removed from all the other split(s) to which it is also queued.

NOTE:

The *check-backup split, queue-to main split,* and converse-on commands can access *only* those splits that are "vector-controlled." A split is considered "vector-controlled" if *yes* is entered in the **Vector** field of the Hunt Group Administration Screen.

Leaving Recorded Messages

Basic Call Vectoring allows the caller to leave a message for the customer if the agents at the customer site are not available to take telephone calls. This is done with the help of the *messaging split* command. Let’s take a look at an example.



```
 1. goto step 8 if time-of-day is all 16:30 to all 7:30
 2. goto step 10 if calls-queued in split 47 pri l >= 20
 3. queue-to main split 47 pri m
 4. wait-time 12 secs hearing ringback
 5. announcement 4001
 6. wait-time 60 secs hearing music
 7. goto step 5 if unconditionally
 8. announcement 4111(‘‘We’re sorry, our office
    is closed.  If you’d like to leave a message, please
    do so after the tone.  Otherwise, please call back
    weekdays between 7:30 A.M. and 4:30 P.M.  Thank you.’’)
 9. goto step 11 if unconditionally
10. announcement 4222 (“We’re sorry, all of our agents are busy,
    please leave a message after the tone and we will return your
    call.”)
11. messaging split 18 for extension 2000
12. disconnect after announcement 4333 (‘‘We’re sorry, we are
    unable to take your message at this time.  Please
    call back at your convenience weekdays between
    7:30 A.M. and 4:30 P.M.  Thank you.’’)
```

Figure 4-11. Leaving Recorded Message

In this vector, the *goto step* command in Step 1 checks to see if the office is open, and branches to Step 8 if the office is closed. This is done to accommodate calls that are made during non-working hours, when there are no agents available to take telephone calls. Accordingly, Step 8 provides the caller with an appropriate announcement and an opportunity to leave a recorded message.

Step 2 checks to see if split 47’s queue (which has 20 queue slots) is full, and branches to step 10 if it is. Steps 3 to 7 queue the call to split 47 and then give audible feedback to the caller.

If the caller chooses to leave a message, the *messaging split* command in Step 11 is executed. Split 18 in the command is the Audio Information Exchange (AUDIX) split. AUDIX is a voice mail adjunct that allows a customer to record, edit, store, forward, and retrieve voice messages to and/or from callers. Extension 2000 is the mailbox for split 47 (from Step 2), which represents a group of daytime agents.

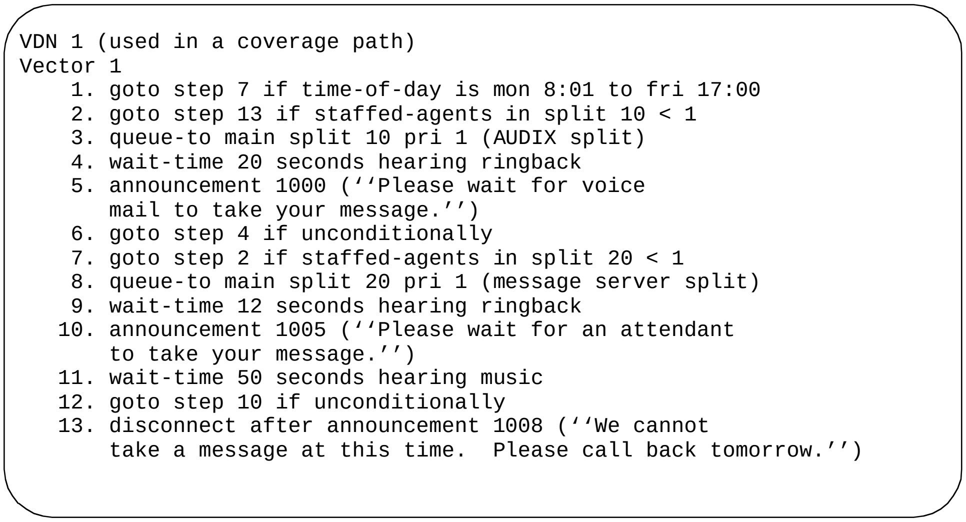
Upon execution of the *messaging split* command, an attempt is made to connect the caller to AUDIX so he or she can leave a recorded message. If the split queue is full, or if the AUDIX link is down, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step, which (as is the case here) usually contains an announcement that provides the caller with the appropriate apology and subsequent directives. If the caller is successfully connected to AUDIX, vector processing terminates, and a message may be left for the specified mailbox (2000, in this case).

Finally, if the supervisor or a group of agents has an Automatic Message Waiting (AMW) Lamp for the mailbox used, and if the lamp lights, the relevant party, upon returning, knows a caller has left an AUDIX message.

Option with the VDN as the Coverage Point

Recall from Chapter 3 that the Vector Directory Number (VDN) can be used as the last point in a coverage path. This capability allows the call to first go to coverage and to then be processed by Call Vectoring and/ or Call Prompting. The capability also allows you to assign AUDIX or the Message Server to a vector-controlled hunt group and to therefore enable access to these servers via a *queue-to main split* or *check-backup split* command. The result of all this is that call handling flexibility is enhanced.

Here's a vector, for which the VDN serves as a final coverage point, that allows the caller to leave a recorded message.



```
VDN 1 (used in a coverage path)
Vector 1
    1. goto step 7 if time-of-day is mon 8:01 to fri 17:00
    2. goto step 13 if staffed-agents in split 10 < 1
    3. queue-to main split 10 pri 1 (AUDIX split)
    4. wait-time 20 seconds hearing ringback
    5. announcement 1000 (‘‘Please wait for voice
       mail to take your message.’’)
    6. goto step 4 if unconditionally
    7. goto step 2 if staffed-agents in split 20 < 1
    8. queue-to main split 20 pri 1 (message server split)
    9. wait-time 12 seconds hearing ringback
   10. announcement 1005 (‘‘Please wait for an attendant
       to take your message.’’)
   11. wait-time 50 seconds hearing music
   12. goto step 10 if unconditionally
   13. disconnect after announcement 1008 (‘‘We cannot
       take a message at this time.  Please call back tomorrow.’’)
```

Figure 4-12. Leaving Recorded Messages (VDN as the coverage point option)

In Steps 3 and 8 of the vector, the caller is given the option of leaving a recorded message. However, in accord with our discussion at the beginning of this section, the *queue-to main split* command instead of the *messaging split* command is used in each case. The advantage here is that the call is actually *queued* to the AUDIX split or to the message server split. On the other hand, a *messaging split* command does not queue the call to the split; instead (if successful), it simply connects the caller to the split so the caller may leave a message for the specified extension. However, termination to the split may turn out to be unsuccessful due to a factor that cannot be “checked” by vector processing. (For example, the AUDIX link might be down, or all AUDIX ports might be out of service.)

As a result of the queuing process, a wait-announcement loop can be included after each *queue-to main split* step, and the appropriate loop can be executed until the call is actually terminated to either an AUDIX voice port or to an available message service agent. In this vector, Steps 4 through 6 comprise the first wait-announcement loop, and Steps 10 through 12 comprise the second such loop.

Sending Calls to a Vector-Programmed Number

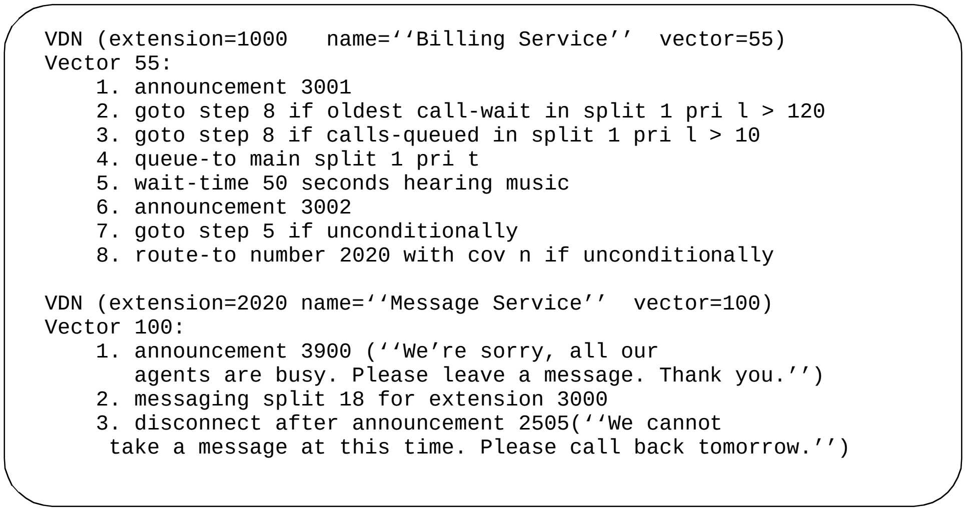
Earlier in this chapter, we mentioned calls can be queued to a maximum of three splits. Calls can also be routed to a programmed number in the vector via a process known as *interflow*.

Interflow

Interflow is a process that allows calls that are directed or redirected to one split to be redirected to an internal or an external destination. For Basic Call Vectoring, this destination is represented by a number programmed in the vector. The number is always included in the *route-to number* command, and it may represent any of the following destinations:

- Attendant (or attendant queue)
- Local extension
- Remote (that is, UDP) extension
- External number
- VDN

The following vectors illustrate how interflow is used:



```
VDN (extension=1000   name=‘‘Billing Service’’  vector=55)
Vector 55:
    1. announcement 3001
    2. goto step 8 if oldest call-wait in split 1 pri l > 120
    3. goto step 8 if calls-queued in split 1 pri l > 10
    4. queue-to main split 1 pri t
    5. wait-time 50 seconds hearing music
    6. announcement 3002
    7. goto step 5 if unconditionally
    8. route-to number 2020 with cov n if unconditionally

VDN (extension=2020 name=‘‘Message Service’’  vector=100)
Vector 100:
    1. announcement 3900 (‘‘We’re sorry, all our
       agents are busy. Please leave a message. Thank you.’’)
    2. messaging split 18 for extension 3000
    3. disconnect after announcement 2505(‘‘We cannot
     take a message at this time. Please call back tomorrow.’’)
```

Figure 4-13. Call Interflow

In the first vector, a branch is made to Step 8 from Step 2 if the condition in the latter step (*oldest call-wait in split 1 > 120 seconds*) is true. If the condition is false, a branch is made to Step 9 from Step 3 if the condition in the latter step (*calls-queued in split 1 > 10*) is true. If that condition is also false, the call is queued (Step 4), and a wait-announcement loop becomes effective (Steps 5 through 7).

If a successful branch to Step 8 is made from Step 2, the *route-to number* command is executed. The destination number (2020) in this particular command is a VDN. Accordingly, vector processing terminates in the first vector and begins at the first step of the second vector, to which the VDN points.

Once processing control is passed to the second vector, the caller is provided with the appropriate announcement (Step l). Thereafter, upon execution of the *messaging split* command in Step 2, the system attempts to either queue the call to the message service split or else terminate the call to a message service agent or to an AUDIX voice port. If one of these attempts succeeds, the caller may leave a message. If none of the attempts succeed, the command fails, and vector processing continues at the next vector command (usually an announcement explaining that the necessary connection could not be made).

Service Observing

Vector initiated Service Observing is available with G3V4 and later releases. For a complete description of Service Observing see the *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204.

Service Observing vectors allow users to observe calls either from a remote location or a local station. A Service Observe button is not required. The use of a Service Observing vector limits users to listen-only or listen-talk observing. The observer cannot toggle between the two states.

Service Observing vectors can be used to observe physical extensions, EAS logical agent LoginIDs, and VDNs.

The calling permissions of the COR assigned to the Service Observing VDN in conjunction with the “can be observed” settings of the COR assigned to the destination determine what agents, terminals, or VDNs can be observed. For additional information about the security requirements with Service Observing vectors see Appendix I, "Security Issues"

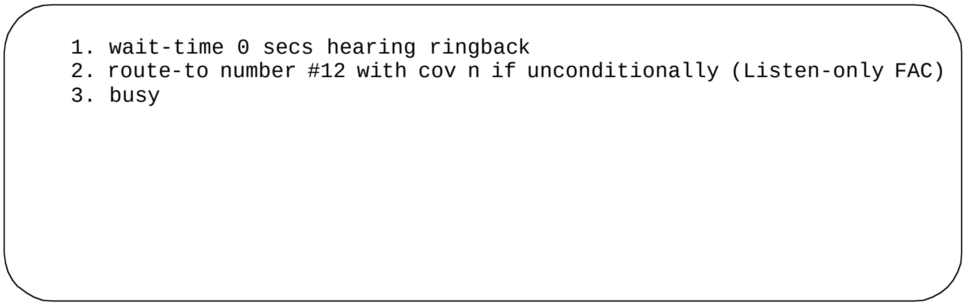
You can construct Service Observing vectors in one of four ways. Vectors can route calls to:

1. A Service Observing FAC
2. The Remote Access extension using Call Prompting to test against a user-entered security code.
3. A Service Observing FAC and extension entered by the user with Call Prompting enabled
4. One of several Service Observing FACs and extensions programmed into route-to number vector steps. In this case Call Prompting can be used to allow the observer to select the extension to be observed.

The first vector type is discussed below. See Chapter 5, "Call Prompting" for examples of Service Observing vectors that use Call Prompting.

Service Observing FAC Vector

The following vector connects the user to a Service Observing FAC. Be aware that this vector does not provide security checks and should be used with great care and only in situations where security is not a concern.



```
1. wait-time 0 secs hearing ringback
2. route-to number #12 with cov n if unconditionally (Listen-only FAC)
3. busy
```

Figure 4-14. Vector for Service Observing FAC

In this vector the caller is connected to a listen-only Service Observing FAC. Once connected, the user must dial the extension number to be observed. To observe in a listen/talk mode, the observer would dial a different VDN.

Branching/Programming

Basic Call Vectoring provides several programming methods that affect the processing flow within the vector. These methods, which are implemented via Call Vectoring commands, include the following:

- Unconditional branching
- Conditional branching
- Stopping vector processing

The following sections explain these programming methods.

Unconditional Branching

Unconditional branching is a method that always passes control from the current vector step to either a preceding or subsequent vector step or to another vector. This type of branching is enabled via the *goto step* and *goto vector* commands, each with a condition of *unconditionally* assigned.

Unconditional branching is illustrated in the following vector.

```
1. goto step 8 if calls-queued in split 3 pri m > 10
2. queue-to main split 3 pri m
3. wait-time 12 seconds hearing ringback
4. announcement 3001
5. wait-time 30 seconds hearing music
6. announcement 3002
7. goto step 5 if unconditionally
8. busy
```

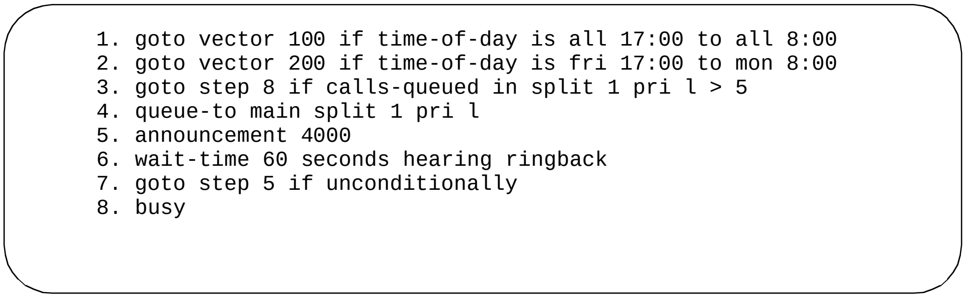
Figure 4-15. Unconditional Branching

The unconditional branch statement in Step 7 establishes an apparent ‘‘endless loop’’ involving Steps 5 through 7. The loop, however, really is not endless, since vector processing terminates if an agent answers the call. Vector processing also terminates when the system recognizes the caller has abandoned the call.

Conditional Branching

Conditional branching is a method that *conditionally* passes control from the current vector step to either a preceding or subsequent vector step or to a different vector. This type of branching is enabled via the *goto step* and *goto* *vector* commands, each with one of the following conditions assigned and tested: *available-agents, staffed-agents, calls-queued*, *oldest call-waiting*, or *time-of-day*. When Advanced Vector Routing is enabled, additional conditions can be tested: *rolling-asa*, *counted-calls*, *expected-wait*. See Chapter 6, "Advanced Vector Routing" for more information. When ANI and II-Digits Routing is enabled, the *ani* and *ii-digits* conditions can also be tested with a *goto* command. See, Chapter 7, "ANI and II-Digits Routing" for more information. If the command's condition is not met, control is passed to the step that follows.

Conditional branching is illustrated in the following vector.



```
1. goto vector 100 if time-of-day is all 17:00 to all 8:00
2. goto vector 200 if time-of-day is fri 17:00 to mon 8:00
3. goto step 8 if calls-queued in split 1 pri l > 5
4. queue-to main split 1 pri l
5. announcement 4000
6. wait-time 60 seconds hearing ringback
7. goto step 5 if unconditionally
8. busy
```

Figure 4-16. Conditional Branching

In this vector, a conditional branch test statement appears in steps 1, 2 and 3. If the call is placed during non-business hours (between 5:00 p.m. and 8:00 a.m.) on any day of the week, the *goto vector* command in Step 1 routes the call to vector 100. However, if the call is placed during business hours, control is passed to Step 2, where the *goto vector* command there checks whether the call is placed during the weekend. If this is the case, the call is routed to vector 200. If not, control is passed to Step 3, where the *goto step* command checks for the number of calls queued to the main split. If the number of calls is greater than 5, control is passed to *busy* in Step 8. If the number of calls is 5 or less, the call is queued (Step 4). Thereafter, an announcement-wait cycle (Steps 5 through 7) is implemented until an agent answers the call or the call is abandoned.

Stopping Vector Processing

Basic Call Vectoring provides a specific command that stops vector processing. The *stop* command halts the processing of any subsequent vector steps. If a call is not queued when vector processing stops, the call is dropped and tracked as an "abandon" by the Call Management System (CMS) and/or BCMS. After the *stop* command is processed, any calls that are already queued remain queued, and any wait treatment (silence, ringback, system music, or alternate audio/music source) is continued.

The following vector illustrates how vector processing is stopped via the *stop* command.

```
1. goto step 6 if calls-queued in split 21 pri m > 10
2. queue-to main split 21 pri m
3. announcement 4000
4. wait-time 30 seconds hearing ringback
5. stop
6. busy
```

Figure 4-17. Stopping Vector Processing

If the *stop* command is reached, the queued caller will continue to hear ringback. Also, if the *stop* command in Step 5 is executed, Step 6 is not executed immediately thereafter. The latter step can be executed only if the *goto* command in Step 1 succeeds.

Note that an *implied stop* follows the last step within a vector. In addition, a vector will stop processing whenever 1,000 vector steps have been processed.

Vector Chaining

Multiple vectors can be chained together to enhance processing capabilities. In this regard, the following points involving two Basic Call Vectoring commands should be noted:

- *Route-to number*. If this command is used to point to a VDN, the following happens:

  1. Vector processing continues at the first step in the vector assigned to the routed-to VDN.
  2. Call (if queued) is dequeued.
  3. Wait treatment (if any) is disabled.

  Processing then continues in the receiving vector at Step 1.

- *Goto vector*. If this command is used, the following happens:

  1. Vector processing continues at the first step in the branched-to vector.
  2. Call (if queued) remains in queue.
  3. Wait treatment (if any) is continued.

  Processing then continues in the receiving vector at Step 1.

Call Prompting

5

Introduction

Call Prompting provides flexible call handling based on information collected from a calling party. This information comes in the form of dialed digits originating from an internal or external touch-tone telephone, or from an internal rotary telephone. In effect, Call Prompting allows for the temporary transfer of call management control to the caller.

With Voice Response Integration (VRI), digits may be returned to the switch by a Voice Response Unit (VRU) script accessed via a *converse-on split* command. Such digits can also be used for call management.

Call Prompting may be used in various applications to achieve a better and more flexible handling of telephone calls.

Command Set

The following table illustrates the commands used for Call Prompting:

Table 5-1. Call Prompting Command Set

Command Category	Action Taken	Command
INFORMATION COLLECTION	Collect information from the calling party or from a Voice Response Unit (VRU).	*collect digits*
TREATMENT	Play an announcement.	*announcement*
	Delay with audible feedback of silence, ringback, system music, or an alternate audio/music source.	*wait-time*
ROUTING	Leave a message.	*messaging split*
	Route the call to a number programmed in the vector.	*route-to number*
	Route the call to digits supplied by the calling party.	*route-to digits*
BRANCHING/ PROGRAMMING	Go to a vector step.	*goto step*
	Go to another vector.	*goto vector*
	Stop vector processing.	*stop*

Touch-Tone Collection Requirements

Before the DEFINITY system can accept the touch-tone digits entered by a Call Prompting user, the switch must be equipped with a “collection resource.” The resource used for collecting and interpreting touch-tone digits is a unit of hardware called a Touch-Tone Receiver (TTR). These TTRs are provided on the TN744 call classifier and TN2182 tone detector (G3V4 and later releases), one of which is required for Call Prompting.

For new systems, the number of required TTRs is configured according to two sources, as follows:

- Customer input to the AT&T Account Team
- Account team input to the DOSS/ATTOMS configuration

For existing systems that are adding a Call Prompting application, the AT&T Account Team recommends the appropriate number of TTRs based on two factors, as follows:

- Account team input to the DOSS/ATTOMS configuration
- Application review by the AT&T Design Center

Outside callers must have a touch-tone phone to enter the digits requested via the *collect digits* command. For callers using rotary dialing, the Call Prompting timeout takes effect, the *collect digits* command times out, and vector processing continues at the next step. As a precaution, the customer should always provide a default treatment (for example, *route-to* attendant command, *queue-to main split* command) in the vector script unless the script is created exclusively for users of touch-tone telephones.

NOTE:
With G3V4 and later releases, the Call Prompting inter-digit timeout can be administered for any number of seconds from 4 to 10. This value is administered on the Feature-Related System Parameters form. See *DEFINITY Communications System Generic 3 Version 4 Implementation*, 555-230-655 or *DEFINITY Communications System Generic 3 V2/V3 Implementation*, 555-230-653, for instructions.

Provisions for users of rotary phones are illustrated in the vector scripts in this chapter.

Call Prompting Digit Entry

The touch-tone digits entered by a Call Prompting user are collected via the *collect digits* command. This command allows the system to collect up to 24 digits from a touch-tone phone. Sixteen of these digits may be collected immediately, while any remaining digits are stored as dial-ahead digits (explained later in this chapter).

Call Prompting allows some flexibility in entering digits. Specifically, the caller can do the following:

- Remove incorrect digits strings
- Enter variable-length digit strings
- Enter dial-ahead digits

The following sections explain these processes.

Removing Incorrect Digit Strings

The customer can (and probably should) include an announcement that requests the caller to enter digits. As an option, the announcement can instruct the caller to enter an asterisk (*) if he or she enters incorrect data. When the caller enters a ‘‘*,’’ the following happens:

1. Digits collected for the current *collect digits* command are deleted.

   NOTE:
   Also deleted are any dial-ahead digits that are entered and that do not exceed the maximum digit count of 24. (Dial-ahead digits are explained later in this chapter.)

2. Digit collection is restarted.
3. Announcement is not replayed.

Once the caller enters ‘‘*,’’ the caller can re-enter digits for processing.

Entering Variable-Length Digit Strings

The maximum number of digits requested from the caller must be specified in the administration of the *collect digits* command. In some cases, the caller might be permitted to enter fewer digits than the maximum specified. In fact, the number of digits entered by the caller can vary for several variations of one *collect digits* command. Each such grouping of digits is called a *variable-length digit string*.

Call Prompting allows for variable-length digit strings by providing an end-of-dialing indicator in the form of the pound sign (#). ‘‘#’’ is used to end any digit string entered by the caller, and it does the following:

- Tells the PBX that the caller has finished entering digits
- Causes the next vector step to be processed immediately

Whenever the caller is permitted to enter a variable-length digit string, the announcement portion of the *collect digits* command should specify the largest possible number of digits that can be entered. Accordingly, the customer should administer each *collect digits* command to collect no more than the intended maximum number of digits. The customer can have the caller enter ‘‘#’’ as part of a variable digit string entry either at the end of each variable digit string entered or at the end of each such string that, not counting ‘‘#,’’ contains *fewer* characters than the maximum number of allowable digits. In the first case, ‘‘#’’ should be included in the count of the number of maximum digits that can be entered; in the second case, ‘‘#’’ should *not* be included in this count.

If the caller enters more digits than the maximum number specified, the additional digits are saved as “dial-ahead” digits for subsequent *collect digits* commands. ("Dial-Ahead Digits" are explained later in this chapter.) If the vector, or vectors chained to it, do not contain another *collect digits* command, the extra digits are discarded.

If the caller enters fewer digits than the maximum number specified *and* does not complete the entry with “#,” a Call Prompting timeout occurs. The timeout terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

A common application involving the entering of variable-length digit strings allows the user to dial either the number for the attendant or an extension (to reach the desired destination.) Let’s say the maximum number of digits that can be entered is three. In such a case, if the user wishes to reach the attendant, the user should dial “0#.” However, if the user chooses to dial a 3-digit extension, the user should dial, for example, ‘‘748’’ and not ‘‘748#.’’ Since the maximum number of digits that can be dialed in this case is three, dialing ‘‘748#’’ would cause ‘‘#’’ to be saved as a dial-digit (explained later in this chapter). On the other hand, if the caller dials “748#,’’ and if the maximum number of digits that can be entered is *4*, ‘‘#’’ is not saved as a dial-ahead digit since it is the fourth of four digits that can be entered in this case.

Entering Dial-Ahead Digits

When digit collection for the current *collect digits* command completes, vector processing continues at the next vector step. However, the switch continues to collect any digits that the caller subsequently dials. These “dialed-ahead” digits are saved for processing by subsequent *collect digits* commands. Dial-Ahead Digits are explained fully on page 5-14.

Functions and Examples

Call Prompting uses some of the functions found in Basic Call Vectoring. This becomes evident when you compare the command set table for Basic Call Vectoring in Chapter 4 with the one for Call Prompting found at the beginning of this chapter.

Call Prompting also provides some additional functions that involve digit processing. These functions include the following:

- Treating digits as a destination
- Using digits to collect branching information (including Vector Routing Tables)
- Using digits to select options
- Displaying digits on the agent’s set
- Passing digits to an adjunct
- Creating Service Observing vectors

These functions are illustrated in the following sections.

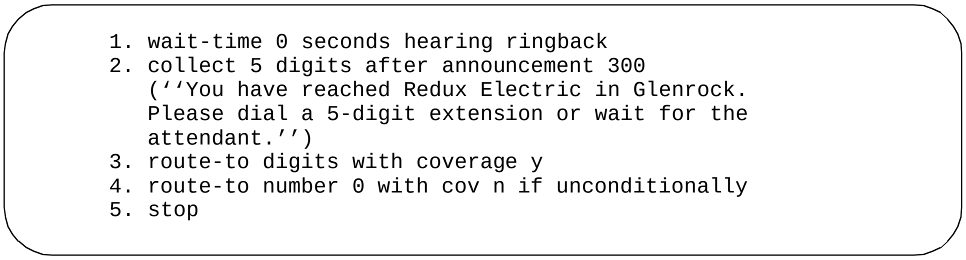
Treating Digits as a Destination

Call Prompting allows you to route calls according to the digits collected from the caller. Once the digits are collected via the *collect digits* command, the *route-to digits* command attempts to route the call to the destination that the digits represent. The command always routes the call to the destination that is indicated by the digits processed by the most recent collect digits command.

The digits can represent any of the following destinations:

- Internal (local) extension (for example, split/hunt group, station, announcement, etc.)
- VDN extension
- Attendant
- Remote access extension
- External number, such as a trunk access code (TAC) or an Automatic Alternate Route/Automatic Route Selection (AAR/ARS) feature access code (FAC) followed by a public network number (for example, 7 digit ETN, 10 digit DDD, etc.).

Let’s take a look at a vector that illustrates how a call is routed via digits that are collected from a caller:



```
1. wait-time 0 seconds hearing ringback
2. collect 5 digits after announcement 300
   (‘‘You have reached Redux Electric in Glenrock.
   Please dial a 5-digit extension or wait for the
   attendant.’’)
3. route-to digits with coverage y
4. route-to number 0 with cov n if unconditionally
5. stop
```

Figure 5-1.   Treating Digits as a Destination

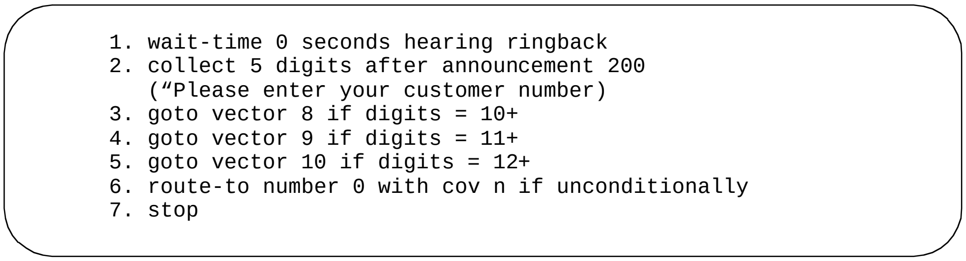
In this vector, the caller is prompted to enter the destination extension of the party he or she would like to reach (Step 2). (The extension in this vector may contain up to 5 digits.) The vector collects the digits, then routes to the destination via the *route to digits* command in Step 3.

If the *route-to digits* command fails (because the caller fails to enter any digits, or because the extension number entered is invalid), the *route-to number command* in Step 4 routes the call to the attendant (default). However, as long as the destination is a valid extension, the *route-to digits* command succeeds, coverage applies, and vector processing terminates. (Even if the destination is busy, vector processing terminates because coverage call processing takes effect.)

NOTE:
From time to time, all of the system’s TTRs might be in use. As a result, you should avoid starting your main vector with a *collect digits* command, since the caller in this case receives no audible feedback if he or she has to wait for a TTR to become available. Accordingly, it is a good practice to include some treatment (for example, *wait-time 0 seconds hearing ringback*) before the initial *collect digits* step.

Using Digits to Collect Branching Information

Call Prompting allows you to direct a call to another step or vector based on the digits entered by the caller. This branching is accomplished with a *goto* step. For example, in the following vector digits are used to route calls to different vectors based on an assigned customer number.



```
1. wait-time 0 seconds hearing ringback
2. collect 5 digits after announcement 200
   (“Please enter your customer number)
3. goto vector 8 if digits = 10+
4. goto vector 9 if digits = 11+
5. goto vector 10 if digits = 12+
6. route-to number 0 with cov n if unconditionally
7. stop
```

Figure 5-2. Using Digits to Collect Branching Information

The wildcard “+” indicates that the two digits can be followed by any number of additional digits (zero or more digits). So, customers with a number that begins with the digits 10 are routed to vector 8, customers with a number that begins with the digits 11 are routed to vector 9, and customers with a number that begins with the digits 12 are routed to vector 10.

Vector Routing Tables

When G3V4 Enhanced Vectoring is enabled, you can also test digits against entries in a Vector Routing Table.

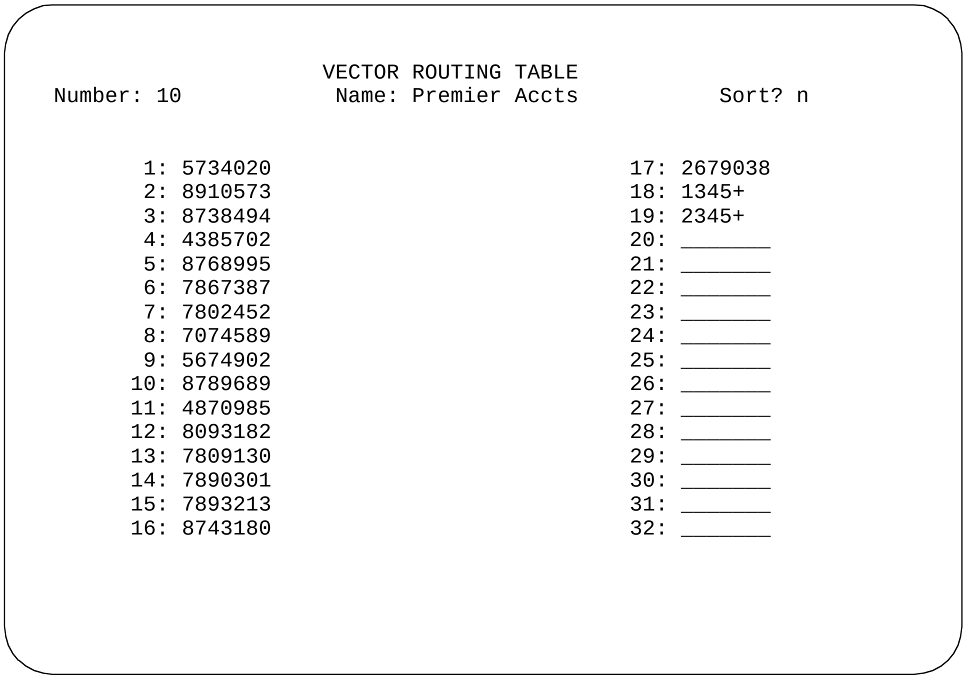
Vector Routing Tables contain lists of numbers that can be used to test a *goto...if digits* command. Digits collected with the collect digits step can be tested to see if they are either *in* or *not-in* the specified table. Entries in the tables can include either the “+” or “?” wildcard.

- The “+” represents a group of digits and can only be used as the first or last character of the string.

- The “?” represents a single digit. Any number of them can be used at any position in the digit string.

Tables are entered on the Vector Routing Table form. Each table can have up to 100 entries. See *DEFINITY Communications System Generic 3 Version 4 Implementation*, 555-230-655 or *DEFINITY Communications System Generic 3 V2/V3 Implementation*, 555-230-653 for complete instructions for creating Vector Routing Tables.

The following figure gives an example of a Vector Routing Table.



```
                     VECTOR ROUTING TABLE
Number: 10            Name: Premier Accts           Sort? n


       1: 5734020                            17: 2679038
       2: 8910573                            18: 1345+
       3: 8738494                            19: 2345+
       4: 4385702                            20: _______
       5: 8768995                            21: _______
       6: 7867387                            22: _______
       7: 7802452                            23: _______
       8: 7074589                            24: _______
       9: 5674902                            25: _______
      10: 8789689                            26: _______
      11: 4870985                            27: _______
      12: 8093182                            28: _______
      13: 7809130                            29: _______
      14: 7890301                            30: _______
      15: 7893213                            31: _______
      16: 8743180                            32: _______
```

Figure 5-3. Vector Routing Table

The following vector could be used to test against numbers in this table.

```
 1. wait-time 0 seconds hearing ringback
 2. collect 7 digits after announcement 200
    (“Please enter your account number)
 3. goto vector 8 if digits in table 10
 4. queue-to main split 5 pri l
 5. wait-time 10 seconds hearing ringback
 6. announcement 2771
 7. wait-time 10 seconds hearing music
 8. goto step 6 if unconditionally
```

Figure 5-4. Testing for Digits In Vector Routing Table

If the caller enters an account number that is listed in the Vector Routing Table the call is routed to vector 8. Likewise, if the caller enters an account number that matches the wildcard entry (for example 1345987) the call is routed to vector 8.

If the caller enters an account number that is not listed in the Vector Routing Table, or if the caller does not enter an account number, the call is queued to split 5.

Suppose that instead of containing a list of premier accounts the Vector Routing Table contained a list of accounts with a poor payment record. The following vector only queues calls with account numbers that are not in the table. Calls in the table route to the collection department.

```
 1. wait-time 0 seconds hearing ringback
 2. collect 7 digits after announcement 200
    (“Please enter your account number)
 3. goto step 11 if digits = none
 4. goto step 6 if digits not-in table 10
 5. route-to number 83456 with cov y if unconditionally (collections)
 6. queue-to main split 5 pri l
 7. wait-time 10 seconds hearing ringback
 8. announcement 2771
 9. wait-time 10 seconds hearing music
10. goto step 8 if unconditionally
11. route-to number 0 with cov n if unconditionally
12. stop
```

Figure 5-5. Testing for Digits Not In Vector Routing Table

If no digits are collected, the call routes to the operator.

Entries in Vector Routing Tables can also be tested against the telephone number of the caller (ANI). See "ANI and II-Digits Routing" on page 7-1 for more information.

Using Digits to Select Options

Call Prompting allows you to provide a menu of options that the caller can use to satisfy his or her information needs. The caller selects the desired option by entering the appropriate requested digit. Once the digit is entered, a conditional branch to the appropriate treatment is made. The treatment is usually provided via the *route-to number* command.

The following vector illustrates how digits are used to select options.

```
1.  wait-time 0 seconds hearing ringback
2.  collect 1 digits after announcement 3531
    (Thank you for calling Bug Out Exterminators.  If you
    wish to learn about the services we provide, please
    dial 1.  If you'd like to set up an appointment for
    one of our representatives to visit your home or
    place of business, please dial 2.)
3.  route-to number 4101 with cov y if digit = 1
4.  route-to number 4102 with cov y if digit = 2
5.  route-to number 0 with cov n if unconditionally
6.  disconnect after announcement none
```

Figure 5-6. Using Digits to Select Options

In Step 2 of this vector, the user is asked to enter either 1 or 2, depending upon the service he or she desires. If one of these digits is entered, the appropriate one of the next two steps (3 through 4) routes the call to the relevant extension (that is, either 4101 or 4102). If one of the digits is not entered, the call is routed to the attendant (Step 5).

Displaying Digits on the Agent's Set

The customer may choose to include the CALLR-INFO button at the agents' display stations to help process calls that are serviced by the Call Prompting feature. However, if the agent has a 2-line display set, such as a 7407 or Callmaster, the collected digits are automatically displayed on the second line. These digits remain on this line until they are overwritten, even after the call is released by the agent. On the other hand, for other display sets, the agent must press the CALLR-INFO button to display the collected digits.

The customer might find it beneficial to install this button if he or she wishes to expedite calls by reducing the amount of time agents spend on the telephone. For example, the button could be set up to collect specific information (such as a customer account number) before the caller is connected to the agent, thus eliminating the need for the agent to ask for this information.

The CALLR-INFO button displays information in the following format:

```
x=Info: 1234567890
```

where

- *x* is a call appearance letter (for example, *a, b, c,* etc.)
- *1234567890* represents the digits collected from the caller

The digits entered by the caller are collected by the most recent collect digits command. Any digits that were ‘‘dialed ahead’’ and not explicitly requested by the most recently executed *collect digits* command are not displayed.

Let’s assume that digits have been collected via Call Prompting. If the agent presses the CALLR-INFO button when the call is alerting at the agent’s station or when the station is active on a call appearance, the following events occur:

- Ten second timer for display interval is set.
- Status lamp (if available) associated with the button is lit.
- Display is updated. Specifically, the incoming call identification (calling party ICI) is replaced with the collected digits in the format presented earlier in this section. Only those digits collected for the last *collect digits* command are displayed.

If all the conditions to use the button (except for the collection of digits) are set, and the agent presses the button, the status lamp (if available) associated with the button flashes denial.

One or more events may occur during a successful execution after the button is pushed. These events include the following:

- Ten second timer times out
- Incoming call arrives (at any call appearance)
- Active call changes status (for example, another caller is added to the conference)

If any of these events occur, the following takes place:

- Status lamp (if available) associated with the button is turned off.
- Display is updated (as previously described).

NOTE:
If the agent needs to display the collected digits again, the CALLR-INFO button can be depressed again to repeat the operation described in this

section (provided the agent is active on the call or the call is still alerting). Also, the agent can flip between the collected digits and the ICI by alternately pressing the CALLR-INFO and NORMAL buttons.

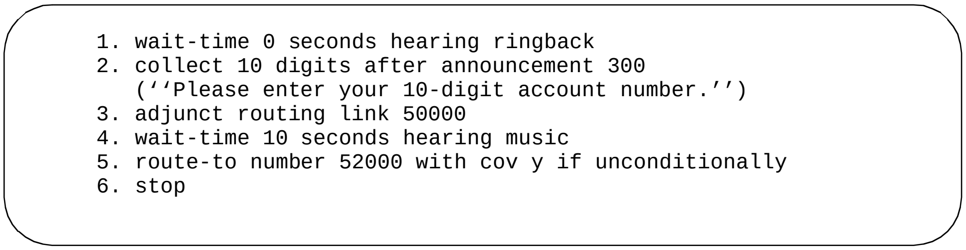
Passing Digits to an Adjunct

Call Prompting allows for the passing of information (in the form of collected digits) to an adjunct for further processing. Digits are passed to the adjunct via the ASAI Adjunct Routing capability.

An adjunct is any processor that is connected to a switch via the ASAI link. The adjunct makes a routing decision via the *adjunct routing* command according to caller information and/or agent availability, and it returns the routing response to the switch. For example, the adjunct can indicate the call be routed to a specific ACD agent (direct-agent option).

A maximum of 16 Call Prompting digits from the last *collect digits* command can be passed to the adjunct via use of the *adjunct routing* command.

The following vector illustrates how Call Prompting digits are passed to an adjunct:



```
1. wait-time 0 seconds hearing ringback
2. collect 10 digits after announcement 300
   (‘‘Please enter your 10-digit account number.’’)
3. adjunct routing link 50000
4. wait-time 10 seconds hearing music
5. route-to number 52000 with cov y if unconditionally
6. stop
```

Figure 5-7. Passing Digits to an Adjunct

In Step 2 of this vector, the caller is asked to enter a 10-digit account number. Once the account number is entered, the adjunct receives this information via the *adjunct routing* command in Step 3. This command then makes the appropriate routing decision if it is able to do so. If the command succeeds within the specified wait time, the command routes the call to the appropriate destination, and the call leaves vector processing. If the command fails, vector processing continues at the next step.

NOTE:
Adjunct Routing is fully discussed in Chapter 9.

Creating Service Observing Vectors

As mentioned in Chapter 4, Service Observing vectors can be constructed to allow users to observe calls from a remote location or local station. When combined with Call Prompting, Service Observing vectors can route calls to:

- A Remote Access extension
- A Service Observing Feature Access Code (FAC) and extension entered by the user
- A Preprogrammed FAC and extension

Remote Access Service Observing Vector

The following vector connects a user to Remote Access. Once connected, the user can dial either a listen-only or listen/talk Service Observing FAC followed by the extension number to be observed. Although not required, Call Prompting increases security by providing passcode protection.

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300
   ("Please enter your 5-digit security code.")
3. goto step 5 if digits = 12345 (security code)
4. disconnect after announcement 2000
5. route-to number 5000 with cov n if unconditionally
6. stop
```

Figure 5-8. Remote Access Service Observing Vector

User-Entered FAC and Extension

The following vector connects a user directly to the Service Observing FAC and extension based on digits collected by Call Prompting.

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300
   (‘‘Please enter your 5-digit security code.’’)
3. goto step 5 if digits = 12345 (security code)
4. disconnect after announcement 2000
5. wait-time 0 seconds hearing ringback
6. collect 6 digits after announcement 3245 (“Please enter the
   number 11 for listen-only observing or the number 12 for
   listen/talk observing followed by the number of the extension you
   would like to observe”)
7. route-to digits with coverage n
8. stop
```

Figure 5-9. Service Observing Vector with User-Entered FAC and Extension

Preprogrammed FAC and Extension

The following vector connects a user to a preprogrammed FAC and extension using Call Prompting to allow the observer to select the extension they would like to observe. In this example, the observer will be Service Observing a VDN.

```
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 2300
   (‘‘Please enter your 5-digit security code.’’)
3. goto step 5 if digits = 12345 (security code)
4. disconnect after announcement 2000
5. wait-time 0 seconds hearing ringback
6. collect 1 digits after announcement 2310 (“Enter 1 to observe
   sales, 2 to observe billing”)
7. route-to number 113001 with cov n if digit = 1 (11 = listen-only
   observe, 3001 = “Sales” VDN)
8. route-to number 113002 with cov n if digit = 2 (11 = listen-only
   observe, 3002 = “Billing” VDN)
9. goto step 6 if unconditionally
```

Figure 5-10. Service Observing Vector with Preprogrammed FAC and Extension

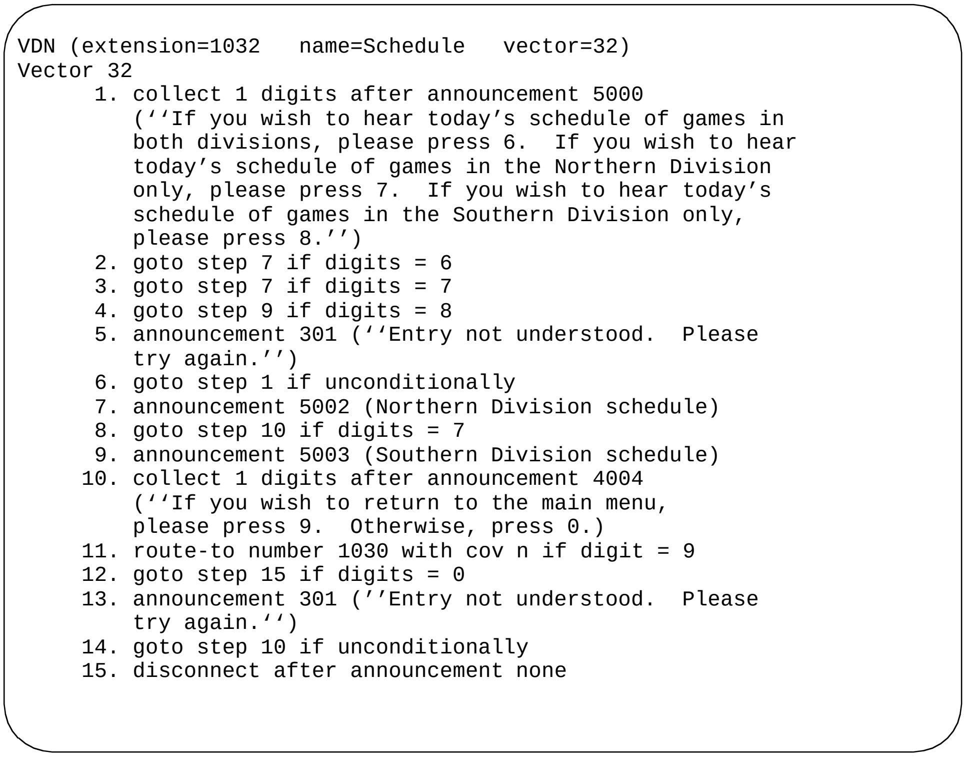
Dial-Ahead Digits

Dial-ahead digits provide the caller with a means of bypassing unwanted announcement prompts on the way to acquiring the information or servicing he or she desires. These digits are available for use only by subsequent *collect digits* commands. The digits are never used by other vector commands that operate on digits (for example, *route-to digits, goto...if digits,* etc.) until they are collected. In addition, these digits are not displayed as part of the CALLR-INFO button operation (see the next section) until they are collected by a *collect digits* command.

The vectors on the next several pages illustrate a situation where a caller can enter dial-ahead digits. Note that, in this case, we are requiring the caller to have a touch-tone telephone. Typically an alternative handling sequence should be programmed in case the caller does not dial a touch tone digit before the timeout period.

```
VDN (extension=1030   name=‘‘Coastal’’   vector=30)
Vector 30:
      1. wait-time 0 seconds hearing ringback
      2. collect 1 digits after announcement 3000
         (‘‘Thank you for calling Coastal League Baseball Hotline.
         You must have a touch-tone telephone to use this service.
         If you wish to hear the scores of yesterday’s games,
         please press 1.  If you wish to hear today’s schedule
         of games, please press 2.’’)
      3. route-to number 1031 with cov y if digit = 1
      4. route to number 1032 with cov y if digit = 2
      5. announcement 301 (‘‘Entry not understood.  Please
         try again.’’)
      6. goto step 2 if unconditionally
VDN (extension=1031   name=‘‘Scores’’   vector=31)
Vector 31:
      1. collect 1 digits after announcement 4000
         (‘‘If you wish to hear scores of games in both divisions,
         please press 3.  If you wish to hear scores for Northern
         Division games only, please press 4.  If you wish to hear
         scores for Southern Division games only, please press 5.’’)
      2. goto step 7 if digits = 3
      3. goto step 7 if digits = 4
      4. goto step 9 if digits = 5
      5. announcement 301 (‘‘Entry not understood.  Please
         try again.’’)
      6. goto step 1 if unconditionally
      7. announcement 4002 (Northern Division scores)
      8. goto step 10 if digits = 4
      9. announcement 4003 (Southern Division scores)
     10. collect 1 digits after announcement 4004
         (‘‘If you wish to return to the main menu,
         please press 9.  Otherwise, press 0.)
     11. route-to number 1030 with cov n if digit = 9
     12. goto step 15 if digit = 0
     13. announcement 301 (’’Entry not understood.  Please
         try again.‘‘)
     14. goto step 10 if unconditionally
     15. disconnect after announcement none
```

Figure 5-11. Dial-Ahead Digits



```
VDN (extension=1032   name=Schedule   vector=32)
Vector 32
      1. collect 1 digits after announcement 5000
         (‘‘If you wish to hear today’s schedule of games in
         both divisions, please press 6.  If you wish to hear
         today’s schedule of games in the Northern Division
         only, please press 7.  If you wish to hear today’s
         schedule of games in the Southern Division only,
         please press 8.’’)
      2. goto step 7 if digits = 6
      3. goto step 7 if digits = 7
      4. goto step 9 if digits = 8
      5. announcement 301 (‘‘Entry not understood.  Please
         try again.’’)
      6. goto step 1 if unconditionally
      7. announcement 5002 (Northern Division schedule)
      8. goto step 10 if digits = 7
      9. announcement 5003 (Southern Division schedule)
     10. collect 1 digits after announcement 4004
         (‘‘If you wish to return to the main menu,
         please press 9.  Otherwise, press 0.)
     11. route-to number 1030 with cov n if digit = 9
     12. goto step 15 if digits = 0
     13. announcement 301 (’’Entry not understood.  Please
         try again.‘‘)
     14. goto step 10 if unconditionally
     15. disconnect after announcement none
```

Figure 5-12. Dial-Ahead Digits

Step 2 in the first vector gives the caller two options, each of which provides different information. The caller is prompted to enter either 1 or 2, depending on what information he or she wishes to hear. Once the caller enters a digit, the digit is collected by the *collect digits* command. Thereafter, an attempt is made by the *route-to number* command to route the call to the appropriate vector (Step 3 or 4). If the caller enters a digit other than *1* or *2*, the appropriate announcement is provided (Step 5), and the digit entry cycle is repeated (Step 6).

Let’s suppose that the caller, when prompted, enters *1*. In such a case, the second vector is accessed.

In Step 1 of this vector, the caller is given three options that supplement the original option provided in the first vector. The caller is prompted to enter either *3*, *4*, or *5*, depending on what information he or she wishes to hear. If the caller enters an incorrect digit, the customary digit correction routine is implemented (Steps 5 and 6). Once an appropriate digit is entered, the call is routed—this time via use of a *goto step* command (Step 2, 3, or 4)—to the appropriate announcement (Step 7 or Step 9).

In Step 10 of the second vector, the caller is once again prompted. Specifically, the caller is given the choice of returning to the main menu provided in the first vector or of terminating the phone call. If the caller selects the former option (by entering *9*), the call is routed to the first vector, and the entire process is repeated.

Note the third vector is similar in design to the second vector. The major difference is the information provided and the requested digit entries.

In our example, we have just seen that the caller has to go through at least two sets of options to get the information he or she wants. Each option set is introduced by an announcement. However, because of the “dial-ahead” digit capability, the caller can bypass the announcements if he or she so chooses. Thus, in our example, the caller could enter *1* and *5* within a matter of seconds to hear yesterday’s Southern Division scores.

The caller may enter digits while he or she is being queued for an announcement or while the announcement is playing. If digits are entered during an announcement, the announcement is disconnected or removed from the queue.

Collection of dial-ahead digits continues until one of the following occurs:

- Vector processing stops or is terminated.
- Sum of the digits collected for the current *collect digits* command plus the dial-ahead digits exceeds the switch storage limit of 24. Any additional digits are discarded until storage is freed up by a subsequent *collect digits* command.

  NOTE:
  Any asterisk (*) and pound sign (#) digits dialed ahead count toward the 24 digit limit, as do any dial-ahead digits entered after the asterisk or pound sign digit.

- The TTR required by the user to collect digits has been disconnected. This happens whenever one of the following conditions is true:
  - — Successful or unsuccessful *route-to number* step is encountered during vector processing, *except* where the number routed to is a VDN extension.
  - — Successful or unsuccessful *route-to digits* step is encountered during vector processing, except where the number routed to is a VDN extension.
  - — Successful or unsuccessful *adjunct routing* step is encountered during vector processing.
  - — Successful or unsuccessful *converse-on* step is encountered during vector processing.
  - — Call Prompting timeout occurs, during which time the caller has not dialed any additional digits, asterisks (*) or pound signs (#).
  - — Vector processing stops or is terminated.

NOTE:
When the TTR is disconnected due to a *route-to number, route-to digits, converse-on* or an *adjunct routing* step, all dial-ahead digits will be discarded. This means that following a failed *route-to, converse* or *adjunct routing* step, a subsequent *collect digits* step always requires the user to enter digits.

The caller who enters dial-ahead digits no doubt knows which digits to enter ahead of time due to his or her familiarity with the service provided. Once the caller masters the digit sequence relevant to a particular service, the dial-ahead digit capability saves time and also eliminates much of the redundancy associated with automatic telephone servicing.

ASAI-Requested Digit Collection

The ASAI-requested digit collection feature gives an adjunct the ability to request that a DTMF tone detector (TN744 or TN 2182) be connected for the purpose of detecting user-entered digits. The digits collected as a result of this feature are passed to ASAI monitoring and/or controlling adjuncts for action. The switch handles these digits like dial-ahead digits. This feature allows the caller to request Sequence Dialing after the call has been routed to the final destination and has resulted in an unanswered call (busy, no answer, etc).

Note that these digits are not necessarily collected while the call is in vector processing. They are sent to an ASAI adjunct, and/or they may be used by Call Prompting features.

ASAI Adjunct Routing and Call Prompting features must be enabled on the switch for this feature to work.

ASAI-Provided Dial-Ahead Digits

The ASAI-provided digits feature allows an adjunct to include digits in a *Route Select* capability. These digits are treated as dial-ahead digits for the call. Dial-ahead digits are stored in a dial-ahead digit buffer and can be collected (one at a time or in groups) using the “collect digits” vector command(s). Although the adjunct may send more than 24 digits in a Route Select, only the first 24 (or 24-x) digits are retained as dial-ahead digits[*]. An application can use this capability to specify the digits that the switch should pass to the VRU as part of the *converse-on* vector step.

[*] The maximum number of dial-ahead digits that can be stored in the buffer is dependent on the number of digits already collected for the call by a previous “collect digits” vector command. If ’x’ digits were collected by vector processing prior to executing an “adjunct routing” vector command, the ’x’ digits collected reduces the maximum number of digits that can be stored as dial-ahead digits as a result of a Route Select. The rest is be discarded.

Advanced Vector Routing

6

Introduction

Advanced Vector Routing adds significantly to the conditional routing capabilities of Basic Call Vectoring. Specifically, it adds the following conditions for routing calls.

- Expected Wait Time (expected-wait)
- Rolling Average Speed of Answer (rolling-asa)
- VDN Calls (counted-calls)

Command Set

The following table illustrates the commands used in Advanced Vector Routing.

Table 6-1. Advanced Vector Routing Command Set

Command Category	Action Taken	Command
ROUTING	Queue the call to a backup ACD split.	*check-backup split*
BRANCHING/ PROGRAMMING	Go to a vector step.	*goto step*
	Go to another vector.	*goto vector*

Expected Wait Time (EWT)

EWT Routing allows you to make routing decisions based on the time that a caller can expect to wait in queue. This wait time can be predicted for a split or for a call. When predicted for a split, the wait time indicates the amount of time the caller can expect to wait if the call is queued to the specified split. When predicted for a call, the wait time indicates the time remaining that the caller can expect to wait in queue until the call is serviced from the queue. The expected wait time can also be passed to a VRU so that a caller can be notified of his or her expected time in queue. The *expected-wait* conditional can be used with either the *goto* or *check-backup* commands.

Call vectoring offers several conditionals that can be used to estimate the time a caller will be delayed waiting in queue, for example, EWT, rolling ASA and Oldest Call Waiting (OCW). EWT is the most accurate of these conditionals. It takes into account more real-time and historical information than the other predictors. For example, priority level, position in queue, number of working agents, etc.

EWT is very responsive to changing call center conditions. For example, it adjusts instantly to any staffing changes in the split; if an agent moves into or out of auxiliary work mode, the wait time predictions adjust immediately.

EWT does not include the time in a call vector before the call enters a queue. It also does not include the time the call rings at a voice terminal after it is removed from the queue.

See "When to Use Wait Time Predictions" later in this chapter for a description of when the predictions are most accurate and the circumstances that will limit their accuracy.

EWT for a Split

The EWT for a split is the time that a new call would be expected to remain in queue if it were queued to the split at the specified priority level. It is generally used to determine if a call should be queued to the split.

For example, the following vector uses EWT for a split to determine if a call should be queued to that split.

```
1. goto step 3 if expected-wait for split 1 pri l < 600
2. busy
3. queue-to main split 1 pri l
4. announcement 3001
5. wait-time 998 secs hearing music
```

Figure 6-1. EWT for a Split

If there are agents available, EWT is zero.

EWT is infinite if:

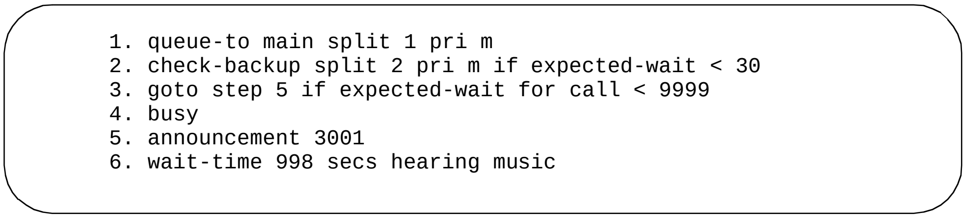
- There are no logged-in agents
- All logged-in agents are in AUX work mode
- The split queue is full
- There is no split queue and all agents are busy
- The split queue is locked

EWT for a Call

EWT for a call is the remaining time a caller can expect to wait before his or her call is serviced from queue. If the call is queued to multiple splits, the remaining queue time for each of the splits is calculated, and the shortest of these is taken as the call's EWT.

For a call to have an expected wait time it must be queued to at least one split. If it is not queued, or if it is queued to splits that are not staffed, the EWT value is infinite.

The following example uses EWT for a call to determine the treatment the call will receive.



```
1. queue-to main split 1 pri m
2. check-backup split 2 pri m if expected-wait < 30
3. goto step 5 if expected-wait for call < 9999
4. busy
5. announcement 3001
6. wait-time 998 secs hearing music
```

Figure 6-2. EWT for a Call

Passing EWT to a VRU

As stated, the Expected Wait Time for a call can be passed to a VRU so that a caller can be notified of his or her expected time in queue. EWT is passed to the VRU with the *converse-on* command as “wait” data. The value outpulsed to the VRU is the expected wait time of the call in seconds. The VRU can then convert the seconds to a spoken message probably rounding up to minutes or converting to minutes and seconds. The expected wait is calculated after the VRU port answers the call, so queuing to a converse split does not adversely impact the EWT value passed to the VRU.

The wait time passed to the VRU is the most accurate prediction possible. On the average 50% of the time the actual wait time will be shorter and 50% of the time it will be longer. It is recommended that VRU applications make an upwards adjustment of the prediction so that the majority of callers receive a predicted wait time that is equal to or greater than their actual wait time.

The VRU can also announce expected wait time to a caller periodically throughout the time that a call is in queue. In this way, the caller can observe his or her progress up the queue. However, this approach should be used with caution. Circumstances such as a reduction in the number of agents or a sudden influx of higher priority calls could cause the caller's expected wait time to increase from one announcement to the next.

If the call is not queued or if it is queued only to splits that are unstaffed or splits where all agents are in AUX work mode, the end-of-string character "#" is the only data item outpulsed.

The EWT Algorithm

EWT is calculated using an algorithm that is based on the number of calls in a queue at a particular priority level and the rate of service of calls from the queue at that priority level. It adjusts for many other factors such as multiple split queuing, call handling times, and the impact of direct agent calls on the wait time of other calls to the split. The algorithm adjusts EWT immediately for changes in staffing, such as agents logging in or taking breaks in AUX work mode.

Since changes occur constantly in a call center, and since EWT cannot predict the future, the accuracy of the EWT predictions will be in proportion to the rate at which call s are serviced from the queue and the level of stability achieved in the call center between the time that the prediction is made and the time that the call is serviced from queue.

When to Use Wait Time Predictions

Wait time predictions are best suited for medium or high volume call scenarios. In general, the potential accuracy of a wait time predictor increases as the rate of removal from queue increases. It is recommended that EWT be used when the rate of removal from queue at a given split priority level is at least one call every 30 seconds.

Predictions can be made for a split with multiple priority levels in use as long as the majority of calls are delivered to the lower priority levels. If the majority of calls are queued at the higher priority levels, any predictions made for the lower priority levels may not be accurate.

The following list describes circumstances that will limit the accuracy of the wait time predictions.

- Immediately after a system restart or when a new split is administered.

  The EWT algorithm uses a combination of historical and real-time information to make predictions. When no historical information exists, such as when a new split is added or a reset system 3 or 4 is completed, there is the potential for inaccuracies.

To prevent inaccurate predictions when there is no historical information, administer the “Expected Call Handling Time” field on the Hunt Group form. The value in this field is then used in place of the missing historical data.If the value of this field does not accurately reflect the call handling times of the split, EWT predictions may be inaccurate until some call history is generated. The algorithm normally requires about 30 queued calls to be answered from a split priority level before it reaches its maximum accuracy.

You can change the value in the “Expected Call Handling Time” field by executing a **change hunt group** command. Changing the value will not disrupt EWT predictions by overwriting EWT history. The value is stored and used the next time a reset system 3 or 4 is executed.

- Low call volume applications.

  Split priority levels where the rate of removal from queue is very low can only be predicted with limited accuracy.

- Sites with frequent staffing changes.

  Although EWT immediately adjusts for all types of staffing changes, since predictions may have already been made for calls waiting in queue, those past predictions will have been based on staffing information which is now out of date. Therefore, scenarios where large staffing changes are continually happening can only be predicted with limited accuracy.

- Staffed agents who rarely answer calls to a split.

  The EWT algorithm takes account of agents in multiple splits in its calculation. However, suppose there are many agents who are assigned to a split but spend most of their time answering calls in their other splits. If a large number of these agents are moved to or from the split, then EWT for this split may be temporarily inaccurate until it adjusts to those changes.

- Applications with widely varying call handling times.

  If the majority of calls to a split are handled within a narrow range of times the accuracy of any predictor will be much greater than that for a split where call handling times are widely different.

Examples

Example 1 — EWT Routing and Passing Wait to a VRU

The following vector illustrates routing based on the wait time of a split, as well as passing wait data to the VRU. Wait time is only given to the caller if the caller is expected to wait a total of more than 60 seconds in queue. Callers who would wait more than 10 minutes are told to call back later.

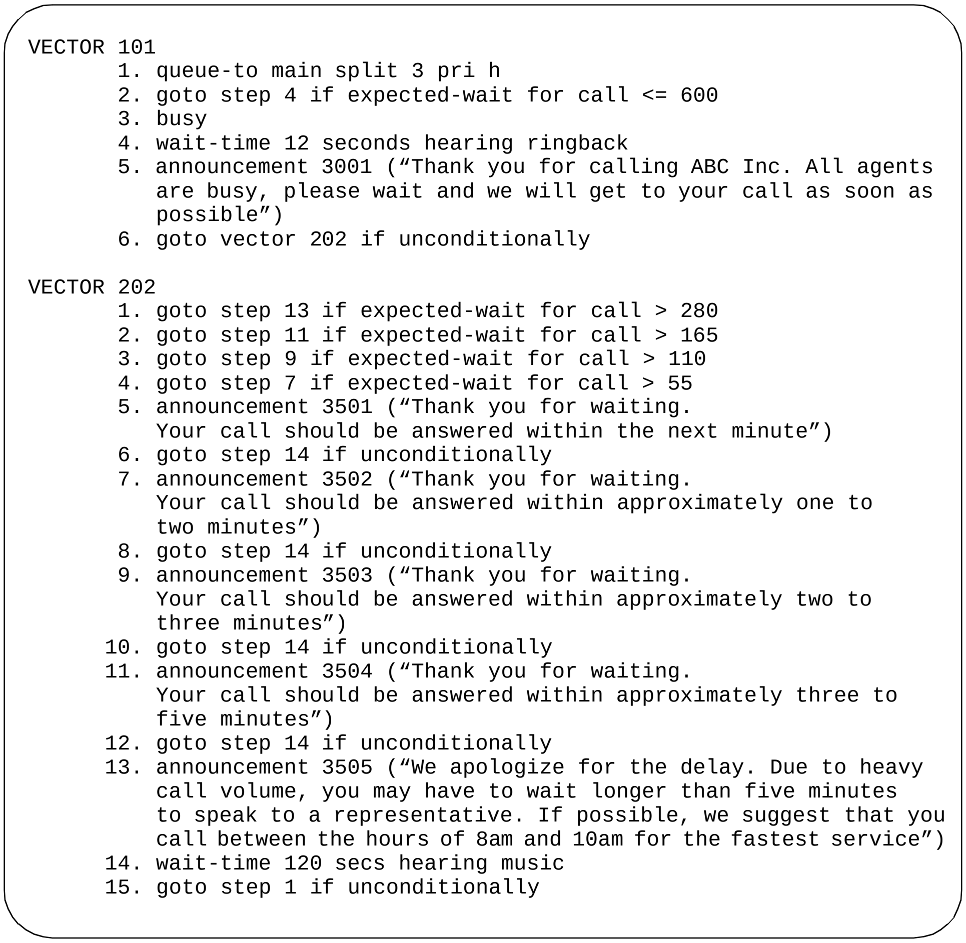
```
1. goto step 3 if expected-wait for split 32 pri l < 600
2. disconnect after announcement 13976
3. queue-to main split 32 pri l
4. wait-time 20 secs hearing ringback
5. goto step 7 if expected-wait for call < 40
6. converse-on split 80 pri l passing wait and none
7. announcement 11000
8. wait-time 60 secs hearing music
9. goto step 7 if unconditionally
```

Figure 6-3. EWT Routing and Passing VRU Wait

Calls with more than 10 minutes to wait fail step 1 and are disconnected after an announcement asking them to call back later. If the expected wait time is less than 10 minutes step one routes the call to step 3 where it is queued to split 32 and waits 20 seconds hearing ringback. After 20 seconds if the expected wait time for the call is less than 40 seconds, step 5 routes the call to an announcement followed by a wait with music. If the expected wait time for the call is equal to or greater than 40 seconds, step 6 informs the caller of the amount of time he or she can expect to wait before the call is answered.

Example 2 — Notifying Callers of Wait Time Without a VRU

You can still use EWT to notify calls of their expected wait time even without a VRU. This can be done using the DEFINITY system recorded announcements and by associating each recorded announcement with a time band.



```
VECTOR 101
       1. queue-to main split 3 pri h
       2. goto step 4 if expected-wait for call <= 600
       3. busy
       4. wait-time 12 seconds hearing ringback
       5. announcement 3001 (“Thank you for calling ABC Inc. All agents
          are busy, please wait and we will get to your call as soon as
          possible”)
       6. goto vector 202 if unconditionally

VECTOR 202
       1. goto step 13 if expected-wait for call > 280
       2. goto step 11 if expected-wait for call > 165
       3. goto step 9 if expected-wait for call > 110
       4. goto step 7 if expected-wait for call > 55
       5. announcement 3501 (“Thank you for waiting.
          Your call should be answered within the next minute”)
       6. goto step 14 if unconditionally
       7. announcement 3502 (“Thank you for waiting.
          Your call should be answered within approximately one to
          two minutes”)
       8. goto step 14 if unconditionally
       9. announcement 3503 (“Thank you for waiting.
          Your call should be answered within approximately two to
          three minutes”)
      10. goto step 14 if unconditionally
      11. announcement 3504 (“Thank you for waiting.
          Your call should be answered within approximately three to
          five minutes”)
      12. goto step 14 if unconditionally
      13. announcement 3505 (“We apologize for the delay. Due to heavy
          call volume, you may have to wait longer than five minutes
          to speak to a representative. If possible, we suggest that you
          call between the hours of 8am and 10am for the fastest service”)
      14. wait-time 120 secs hearing music
      15. goto step 1 if unconditionally
```

Figure 6-4. Notifying Callers of Wait-Time Without a VRU

In Step 1 the call is queued to split 3 at high priority. If the calls fails to get a queue slot in split 3, if split 3 has no working agents, or if the wait time in split 3 at high priority exceeds 10 minutes, step 2 fails and the caller receives busy tone. If step 2 succeeds, the caller hears ringback and an announcement and is then sent to vector 202. Steps 1 through 4 of vector 202 determine which of five time bands the caller’s remaining queuing time is estimated to be within. One of five recorded announcements is then played to the caller to inform him or her of the expected wait time in queue.

Notice that the EWT thresholds are set lower than the times quoted in the recorded announcements. Callers may become upset if their actual wait time exceeds the time stated in the announcement. Therefore, you may want to program your vectors such that few callers ever experience wait times that exceed the wait time of the announcement.

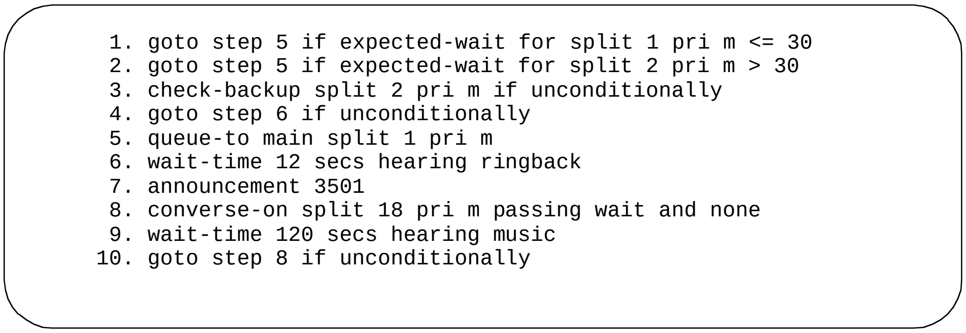
Notice also that vector 202 can be used for any application requiring that the caller be notified of their remaining time in queue.

Example 3 — Using EWT to Route to the Best Split

With EWT, you may wish to change your normal queuing strategy of queuing calls to multiple splits in order to insure the call is answered in the shortest possible time. This strategy uses additional system resources and can make it more difficult to read and analyze split reports.

Instead, you may wish to use EWT to determine up-front, which split is best for each call and avoid multiple split queuing.

In this example, there are two splits, a main split (1) and a backup split (2). Either split can service a particular type of call. It is preferable that an agent from the main split service the call. However, a 30-second maximum wait time is also desirable. The strategy in this vector is to use the backup split only if the backup split can answer the call within 30 seconds and the main split cannot.



```
 1. goto step 5 if expected-wait for split 1 pri m <= 30
 2. goto step 5 if expected-wait for split 2 pri m > 30
 3. check-backup split 2 pri m if unconditionally
 4. goto step 6 if unconditionally
 5. queue-to main split 1 pri m
 6. wait-time 12 secs hearing ringback
 7. announcement 3501
 8. converse-on split 18 pri m passing wait and none
 9. wait-time 120 secs hearing music
10. goto step 8 if unconditionally
```

Figure 6-5. EWT Routing—Routing to the Best Split

Step 1 branches to Step 5 to queue to the main split if the main split can answer the call within 30 seconds. If the main split cannot answer the call within 30 seconds, Step 2 checks to see if the backup split can answer the call within 30 seconds. If it cannot, the call branches to step 5 and is queued to the main split. If it can, the call is queued to the backup split in Step 3. At this point, the call is queued either to the main or the backup split but not to both.

Steps 6 through 10 provide audible feedback to the caller while the call is in queue. Note that in Step 8, which is executed every two minutes, a VRU is used to provide the caller with his or her remaining wait time in queue.

Factors that Effect the Value of EWT

Factors that Cause EWT for a Split Priority Level to Increase

Most common:

- Number of calls in queue increases
- Agents logout
- Agents go on break (AUX work mode)
- Agents are moved to another split
- Agents with multiple splits answer an increasing number of calls in other splits

Other possibilities:

- Average talk time increases
- Number of calls at higher priority increases
- Number of DAC calls increases
- Number of RONA calls increases
- Number of abandoned calls decreases
- Number of calls queued in this split but answered in another decreases

Factors that Cause EWT for a Split Priority Level to Decrease

Most common:

- Number of calls in queue decreases
- Agents login (and start answering calls)
- Agents return from break (leave AUX work mode)
- Agents are moved from another split
- Agents with multiple splits answer fewer calls in other splits

Other possibilities:

- Average talk time decreases
- Number of calls at higher priority decreases
- Number of DAC calls decreases
- Number of RONA calls decreases
- Number of abandoned calls increases
- Number of calls queued in this split but answered in another increases

Rolling Average Speed of Answer (ASA)

Rolling ASA Routing allows you to make routing decisions based on the current average time that it takes for a call to be answered in a split or VDN. In this way, a vector can route a call to the VDN or split where it is likely to be answered most quickly.

The Average Speed of Answer used for vector routing is called “rolling” ASA to differentiate it from the “interval” ASA that is recorded in BCMS and CMS reports. Rolling ASA is a running calculation that does not take into account the 15-minute, half-hour, or hour BCMS/CMS reporting intervals. It does not reflect interval boundaries. On the other hand, the “interval” ASA used for BCMS/CMS reporting is calculated on reporting interval boundaries and clears to zero at the start of each reporting interval.

The Rolling Average Speed of Answer for a split or VDN is calculated based on the speed of answer for all calls recorded since system start-up. When rolling ASA is calculated, each call is given a weighted value that is greater than the call that preceded it. In this way the most recent calls contribute the most to the average. Approximately 95% of the value of rolling ASA is obtained from the last ten calls.

The rolling ASA for a split or VDN is recalculated every time a call is answered so that it always reflects the most recently available data. Calls that are not answered, for example calls that receive a forced busy, are not considered for the rolling ASA calculation.

The rolling ASA is calculated for an entire split or VDN. The calculation does not consider the priority levels of answered calls.

The following sections explain what is included in the rolling ASA calculation for a split or VDN.

Rolling ASA Split Calculation

The rolling ASA for a split is the average time it takes for a call to be answered from the time the call attempts termination to the split until it is answered in that split. Rolling ASA includes the time the call is waiting in queue and the time it is ringing at a voice terminal.

If the call is answered in another split or the call is abandoned by the caller before it is answered, rolling ASA is not recorded for the call. If a call flows into a split from another split, the time queued and ring time for the previous split are not included. If a call is queued in multiple splits, only the rolling ASA for the split in which the call is answered is impacted.

Rolling ASA VDN Calculation

The rolling ASA for a VDN is the average time it takes for a call to be answered from the time it starts processing within the specified VDN until it is answered. It includes any time spent in vector processing including time spent in announcements administered as vector steps. If the call is answered by an agent, it includes the time the call is waiting in queue and the time it is ringing at the agent's voice terminal.

The rolling ASA for a VDN only includes data from calls answered in that VDN. If a call flows between VDNs, only the time spent within the answering VDN is used in the calculation. For example, if a call is placed to VDN1 and after ten seconds routes to VDN2 and is then answered in VDN2 after five seconds, the ASA for the call is recorded in VDN2 as five seconds. Nothing is recorded for VDN1 since the call was not answered there.

The VDN for a vector step can be specified in three ways: a VDN number, the value "latest," or the value "active." The "latest" VDN is the VDN that is currently processing the call. The value is not affected by VDN override. The "active" VDN is the VDN of record. That is, it is the called VDN as modified by override rules. For example, if a call routes from a VDN with override set to *yes* then the new VDN is the active VDN. If a call routes from a VDN with override set to *no* then the previous VDN is the active VDN.

Rolling ASA Considerations

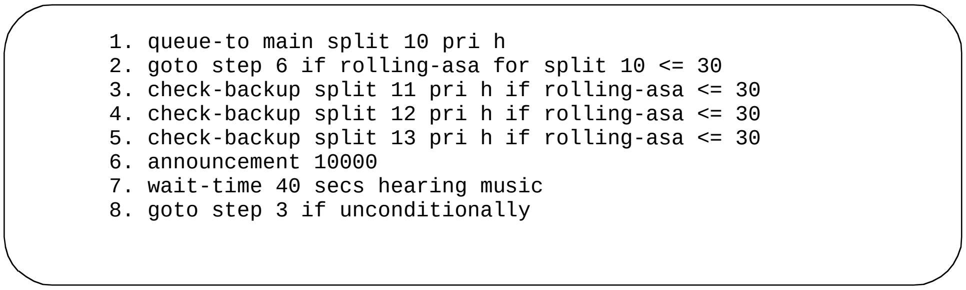
Because of its greater accuracy and greater flexibility, EWT is recommended over rolling ASA as a predictor of split/skill waiting time. However, rolling ASA is provided for those who may have a special requirement or wish to use the more traditional ASA measurement.

Normally rolling ASA conditionals should not be used to prevent calls queuing to the main split/skill or being answered in the principal VDN. Rather, rolling ASA should be used to see whether vector processing should attempt to queue the call to additional splits/skills if the main split/skill does not currently meet the targeted threshold. If no calls are being answered in the main split/skill or VDN, the value of rolling ASA will not change. This could result in all future calls being locked out of the main split/skill or VDN unless there are other call vectors in the system directing calls to them.

If you wish to implement a call flow that decides whether or not to queue a call to a main split/skill, use the EWT feature.

Example

The following example combines VDN and split ASA routing.



```
1. queue-to main split 10 pri h
2. goto step 6 if rolling-asa for split 10 <= 30
3. check-backup split 11 pri h if rolling-asa <= 30
4. check-backup split 12 pri h if rolling-asa <= 30
5. check-backup split 13 pri h if rolling-asa <= 30
6. announcement 10000
7. wait-time 40 secs hearing music
8. goto step 3 if unconditionally
```

Figure 6-6. Rolling ASA Routing

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds Step 2 bypasses all of the backup splits and goes directly to the announcement in Step 6. The assumption is that the call will be handled by split 10 within the time constraints. However, if the call is not answered by the time vector processing reaches Step 8, the backup splits are checked at that time.

If the rolling ASA for the main split is greater than 30 seconds, Steps 3, 4, and 5 check backup splits. The call is queued to any of these splits that have a rolling ASA of 30 seconds or less. If the call still is not answered by the time vector processing reaches Step 8, then the backup splits are checked again.

VDN Calls

VDN Calls routing allows you to make routing decisions based on the number of incoming trunk calls that are currently active in a VDN. With the VDN Calls conditional, a vector can be used to limit the number of simultaneous calls made to a particular VDN. For example, if a service agency is contracted to handle 100 simultaneous calls for a client, calls in excess of that number can be routed to a *busy* step.

When Advanced Vector Routing is enabled, a count of active incoming trunk calls is kept for each VDN. The VDN counter is incremented each time an incoming call is placed to the VDN. It is decremented each time an incoming call is released. A call is considered active in a VDN from the time the call routes to the VDN until all parties on the call have been dropped and the call is released.

NOTE:
The call is counted for the originally called VDN only. When a call is routed to another VDN, the call counter for the subsequent VDN is not incremented. And, the call counter for the original VDN is not decremented.

As with other Advanced Vector Routing conditionals, the VDN for a *goto* step can be specified in three ways: a VDN number, the value “latest,” or the value “active.”

The following section describes which calls are included in the VDN Calls counts and which are not.

Counted Calls

The VDN call count includes:

- Incoming trunk calls that route directly to the VDN.
- Incoming trunk night service calls where the VDN is the night service destination.
- Calls that cover or forward to the VDN if it is the first VDN routed to and the call is an incoming trunk call.
- Already counted calls that are conferenced with counted or not counted calls from the same VDN.

The VDN call count does not include:

- Internal calls to the VDN.
- Calls that are transferred to the VDN.
- Calls redirected to their VDN return destination.
- Conferenced calls previously counted on different VDNs.

Example

The following example shows how the *counted-calls* conditional can be used to route calls.

```
1. goto step 3 if counted-calls to vdn 1234 <= 100
2. busy
3. queue-to main split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 27000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally
```

Figure 6-7. VDN Calls Routing

If more than 100 calls are active in VDN 1234, the caller will hear busy tone and vector processing is terminated. If 100 or fewer calls are active, the call queues to split 60.

ANI and II-Digits Routing

7

Introduction

ANI and ii-digits allow you to make vector routing decisions based on the caller identity and the type of the originating line.

Command Set

ANI and ii-digits are both used for conditional branching with the *goto* step. The following table illustrates the commands used in ANI/II-Digits Routing .

Table 7-1. ANI/II-Digits Routing Command Set

Command Category	Action Taken	Command
BRANCHING/ PROGRAMMING	Go to a vector step.	*goto step*
	Go to another vector.	*goto vector*

ANI Routing

ANI routing allows you to make routing decisions based on incoming or internal caller identity. In this way, calls from a particular customer can receive unique routing, local calls can be routed differently from long distance calls, or calls from different geographical areas can receive different routing. See "ANI Routing Example" later in this section for more information. ANI also can be compared against entries in a Vector Routing Table. See "Vector Routing Tables with ANI" later in this section for more information.

ANI (Automatic Number Identification) is based on the Calling Party Number (CPN). It is not always identical to the Billing Number. For example, if the call is placed by a user from a Private Branch Exchange (PBX), the CPN can be either the PBX billing number or the Station Identification Number.

The ANI routing digit string can contain up to 16 digits. This supports international applications. However, ANI information in North America only contains 10 digits.

The following calls will have ANI values associated with them:

- Incoming ISDN-PRI calls that send ANI
- DCS calls
- Internal calls

If ANI is not provided by the network for a call, then it will not be available for vector processing on that call.

When an EAS agent makes a call to a VDN, the agent's login ID is used as the ANI, not the number of the physical terminal.

When a call is transferred to a VDN the following is true:

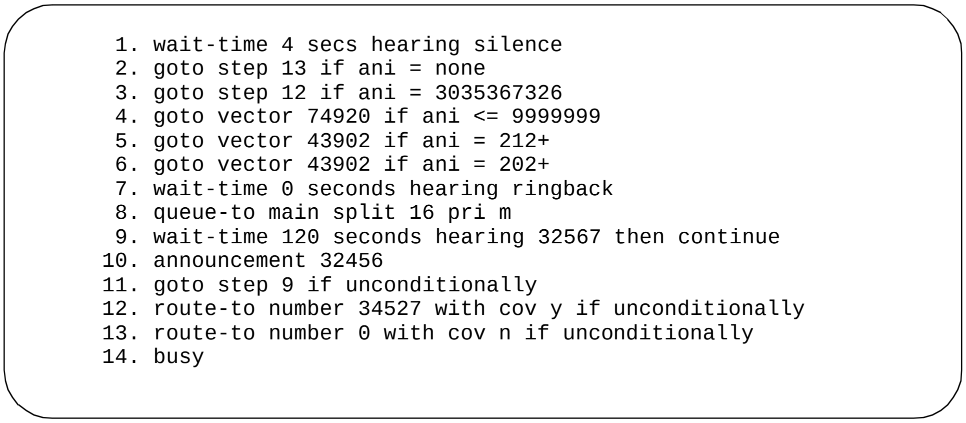
- If the transfer is completed before the call reaches the ANI conditional, the ANI value of the originator of the call is used.
- If the transfer is completed after the call reaches the ANI conditional, the ANI value of the terminal executing the transfer is used.

To ensure that the originator's ANI is preserved during a transfer, add a filler step (such as wait with silence) to the beginning of the vector. In this way a transfer can complete before the ANI conditional is encountered.

The ANI value specified for a goto step can include the "+" and/or "?" wildcard. The "+" represents a group of zero or more digits and can only be used as the first or last character of the string. The "?" represents a single digit. Any number of them can be used at any position in the digit string.

ANI Routing Example

The following example demonstrates several applications of ANI Routing.



```
 1. wait-time 4 secs hearing silence
 2. goto step 13 if ani = none
 3. goto step 12 if ani = 3035367326
 4. goto vector 74920 if ani <= 9999999
 5. goto vector 43902 if ani = 212+
 6. goto vector 43902 if ani = 202+
 7. wait-time 0 seconds hearing ringback
 8. queue-to main split 16 pri m
 9. wait-time 120 seconds hearing 32567 then continue
10. announcement 32456
11. goto step 9 if unconditionally
12. route-to number 34527 with cov y if unconditionally
13. route-to number 0 with cov n if unconditionally
14. busy
```

Figure 7-1. ANI Routing Example

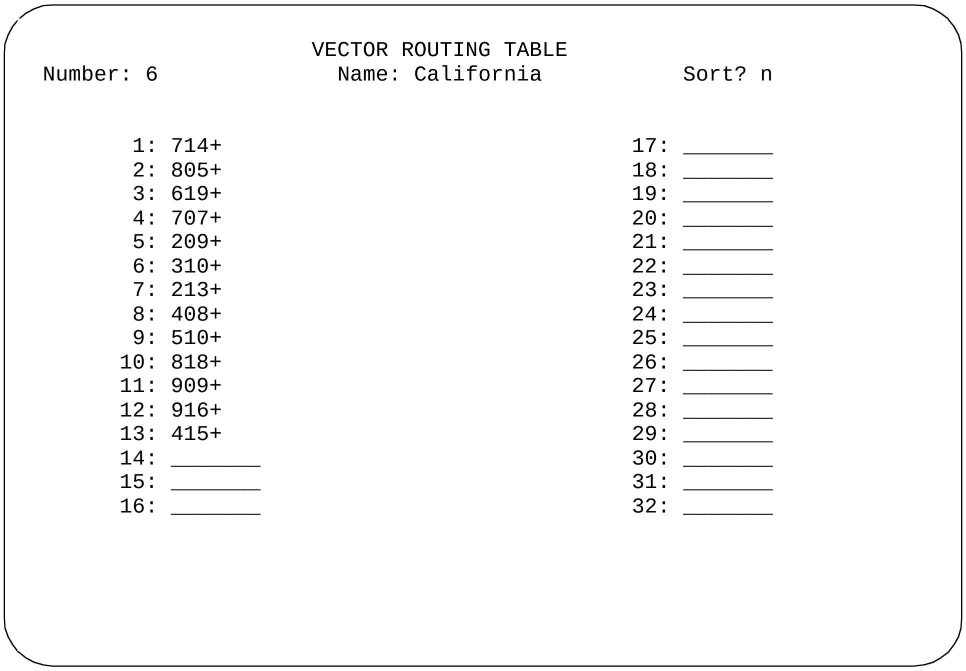
In Step 2 calls that do not have ANI associated with them are routed to an operator. Step 3 routes calls from a specific phone to a specified extension. Step 4 routes local calls (those with 7 or fewer digits) to a different vector. Steps 5 and 6 route calls from area codes 212 and 202 to a different vector. Calls that are not rerouted by the previous steps are then queued.

Vector Routing Tables with ANI

You can also test ANI against entries in a Vector Routing Table.

Vector Routing Tables contain a list of numbers that can be used to test a *goto...if ani* command. ANI can be tested to see if it is either *in* or *not-in* the specified table. Entries in the tables can also include the “+” and/or “?” wildcard.

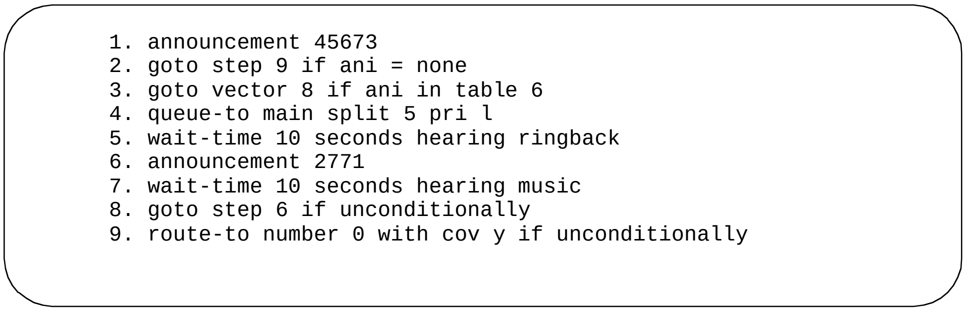
The following figure gives an example of a Vector Routing Table with ANI values. The table contains most of the area codes for the state of California.



```
                         VECTOR ROUTING TABLE
Number: 6               Name: California           Sort? n


       1: 714+                                17: _______
       2: 805+                                18: _______
       3: 619+                                19: _______
       4: 707+                                20: _______
       5: 209+                                21: _______
       6: 310+                                22: _______
       7: 213+                                23: _______
       8: 408+                                24: _______
       9: 510+                                25: _______
      10: 818+                                26: _______
      11: 909+                                27: _______
      12: 916+                                28: _______
      13: 415+                                29: _______
      14: _______                             30: _______
      15: _______                             31: _______
      16: _______                             32: _______
```

Figure 7-2. Vector Routing Table for ANI Routing

The following vector could be used to route these California area code calls to a separate vector.



```
1. announcement 45673
2. goto step 9 if ani = none
3. goto vector 8 if ani in table 6
4. queue-to main split 5 pri l
5. wait-time 10 seconds hearing ringback
6. announcement 2771
7. wait-time 10 seconds hearing music
8. goto step 6 if unconditionally
9. route-to number 0 with cov y if unconditionally
```

Figure 7-3. Testing for ANI In Vector Routing Table

In this example, if no ANI is available for the call, it is routed to an operator. If the first three numbers match an area code from table 6, the call is routed to vector 8. All other calls are queued.

II-Digits Routing

II-digits routing allows you to make routing decision based on the type of the originating line. In this way, calls from pay phones, cellular phones, or motel phones for example can receive unique routing. See "II-Digits Routing Example" later in this section for more information.

II-digits (Information Indicator digits) is a 2-digit string provided for an incoming call by ISDN PRI. II-digits delivery is a generally available ISDN PRI AT&T Network service. This service is bundled with ANI delivery and tariffed under the MEGACOM 800 ® and MultiQuest 800 ® INFO-2 feature to provide information about the call's origination. II-digits indicate the type of originating line.

II-digits routing can be used for example to:

- Help detect fraudulent orders for catalog sales, travel reservations, money transfers, traveler's checks, and so on.
- Assign priority or special treatment to calls placed from pay phones, cellular phones, or other types of lines. For example, special priority could be given by an automobile emergency road service to calls placed from pay phones.
- Detect calls placed from pay phones when it is the intention of the caller to avoid being tracked by collection agencies or dispatching services.
- Convey the type of originating line on the agent display by routing different type calls to different VDNs.

The ii-digits routing string can only contain 2 characters. The string can contain either the "+" or "?" wildcard. Leading zeros are significant. The ii-digits value "02" associated with a call will not match the digit string "2" in a vector step.

As with ANI routing and collected digit routing, ii-routing digits can be compared against entries in a Vector Routing Table.

The following calls will have ii-digits values associated with them.

- Incoming ISDN PRI calls that include ii-digits
- Incoming ISDN PRI Tie Trunk DCS or non-DCS calls that include ii-digits.

  Note that since tandeming of ii-digits is only supported if the trunk facilities used are ISDN PRI, traditional DCS will not support ii-digits transport but DCS Plus (DCS over PRI) will.

When a call is returned to vector processing as a result of the VDN Return Destination feature, the ii-digits are preserved.

When a call is transferred to a VDN the following is true:

- If the transfer is completed before the call reaches the ii-digits conditional, the ii-digits value of the originator of the call is used.

- If the transfer is completed after the call reaches the ii-digits conditional, the ii-digits value of the terminal executing the transfer is used. Under normal circumstances, there will be no ii-digits for a terminal executing a transfer.

To ensure that the originator's ii-digits is preserved, add a filler step (such as wait with silence) to the beginning of the vector. In this way a transfer can complete before the ii-digits conditional is encountered.

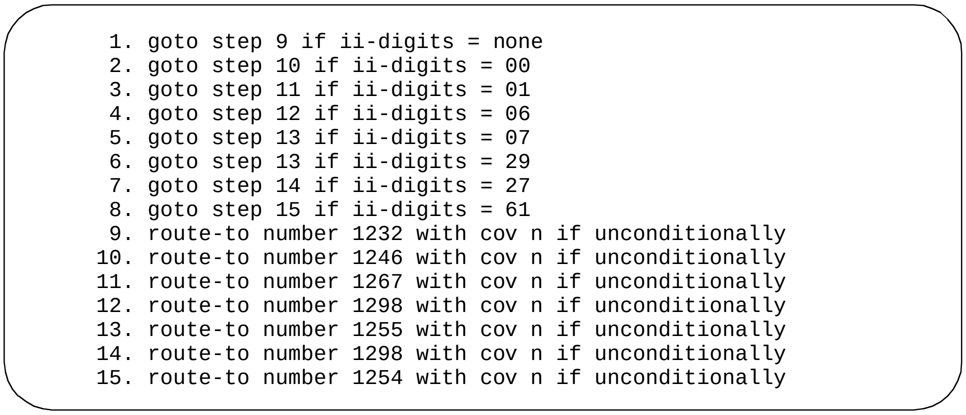
The following table is a paraphrased summary of currently available II-digits. A complete and more descriptive list of II-digits is published quarterly in Section 1 of the "Local Exchange Routing Guide" published by Bellcore.

Table 7-2. II-digits Summary

Code	Use
00	Identified line - no special treatment
01	Multiparty - ANI cannot be provided
02	ANI failure
06	Hotel/Motel - DN not accompanied by automatic room ID
07	Special operator handling required
20	AIOD - Listed DN of PBX sent
23	Coin or Non-Coin - line status unknown
24	800 Service
27	Coin Call
29	Prison/Inmate Service
30-32	Intercept
34	Telco Operator Handled Call
40-49	Locally determined by carrier
52	OutWATS
60	Telecommunication Relay Service (TRS) - Station Paid
61	Type 1 Cellular
62	Type 2 Cellular
63	Romer Cellular
66	TRS - From Hotel/Motel
67	TRS - From restricted line
70	Private paystation
93	Private Virtual Network call

II-Digits Routing Example

The following example demonstrates branching calls with different ii-digits to different VDNs. The VDN override is set to “yes” on the called VDN. In this way, the VDN name or VDN of Origin Announcement can be used to convey to the agent the type of ii-digits associated with the call.



```
 1. goto step 9 if ii-digits = none
 2. goto step 10 if ii-digits = 00
 3. goto step 11 if ii-digits = 01
 4. goto step 12 if ii-digits = 06
 5. goto step 13 if ii-digits = 07
 6. goto step 13 if ii-digits = 29
 7. goto step 14 if ii-digits = 27
 8. goto step 15 if ii-digits = 61
 9. route-to number 1232 with cov n if unconditionally
10. route-to number 1246 with cov n if unconditionally
11. route-to number 1267 with cov n if unconditionally
12. route-to number 1298 with cov n if unconditionally
13. route-to number 1255 with cov n if unconditionally
14. route-to number 1298 with cov n if unconditionally
15. route-to number 1254 with cov n if unconditionally
```

Figure 7-4. II-Digits Routing Example

In this vector, step 1 routes calls with no associated ii-digits to extension 1232. Steps 2 through 8 route calls with different ii-digits to different extensions.

Look-Ahead Interflow

8

Introduction

Look-Ahead Interflow enhances Call Vectoring for call centers with multiple ACD locations. Look-Ahead Interflow allows these centers to achieve optimal call-handling capability and agent productivity by intelligently routing calls among call centers to achieve an optimum ACD load balance. This service is provided via ISDN-PRI D-channel messaging. The receiving switch is able to accept or deny interflowed calls sent by the sending switch.

Command Set

There are no special commands for Look-Ahead Interflow. Instead, this feature uses the commands included within the Basic Call Vectoring and Call Prompting features. The following points concerning several of these commands should be noted:

- *Route-to number with coverage n* or *route-to digits with coverage n* command on a switch that has Look-Ahead Interflow optioned and that successfully seizes an ISDN-PRI trunk automatically results in a normal Look-Ahead Interflow call attempt being placed. The call attempt can be rejected or accepted by the remote end.
- *Route-to number with coverage y or route-to digits with coverage y* command never results in a Look-Ahead Interflow call attempt. The sending end assumes the call is always going to be accepted. This command always completes the call. Moreover, the command should not be used when the vector at the receiving location ends up denying the call, since the caller in this case would be given a busy signal, or the call

will be disconnected. This command with coverage y should only be used for those cases when an unconditional interflow is wanted (with Look-Ahead Interflow active) and the terminating switch is set up accordingly.

Functions and Examples

When a Look-Ahead Interflow call attempt is made, Call Vectoring at the sending location checks a potential receiving location to determine whether to hold or send the call. While this is done, the call remains in queue at the sending location. As such, the call can still be connected to the sending-location agent (if one becomes available) before the receiving location accepts the call.

Call Vectoring at the receiving location decides whether to accept the call from the sending location or to instruct the sending location to keep the call. In the latter case, the sending location can then either keep the call, check other locations, or provide some other treatment for the call. Conditions for sending, refusing, or receiving a Look-Ahead Interflow call attempt can include a combination of any of the following: number of calls in queue, average speed of answer or expected wait time for a split, number of staffed or available agents, the number of calls active in a VDN, time of day and day of week, or any other conditional.

If the call is accepted by the receiving switch, the call is removed from any queues at the sending switch, and call control is passed to the receiving switch. If the call is denied by the receiving switch, vector processing simply continues at the next step at the sending switch. Until the call is accepted by either switch, the caller continues to hear any tones applied by the sending switch. Should the call be denied, the call vector may then apply alternate treatment, such as placing another Look-Ahead Interflow call to an alternate backup switch.

NOTE:
It is important to note that the Look-Ahead Interflow operation is completely transparent to the caller. While a Look-Ahead Interflow call attempt is being made, the caller continues to hear any audible feedback provided by the sending switch vector. The caller also maintains his or her position in any split queues until the call is accepted at the receiving switch.

NOTE:
Look-Ahead interflow does not pass Call Prompting digits collected in the sending switch to the receiving switch. Therefore, when used with LAI, Call Prompting should be used to collect digits only after the last possible interflow. Alternately these digits can be passed to a CallVisor/ASAI application, which can include them in the UUI field of the call. UUI is passed to the receiving switch in the SETUP message.

Two Switch Configuration

Look-Ahead Interflow is enabled through the use of call vectors and their associated commands. For a two switch configuration, these vectors are included in both the sending switch, which processes vector outflow, and the receiving switch, which processes vector inflow. The Two Switch Configuration for LAI is illustrated as follows:

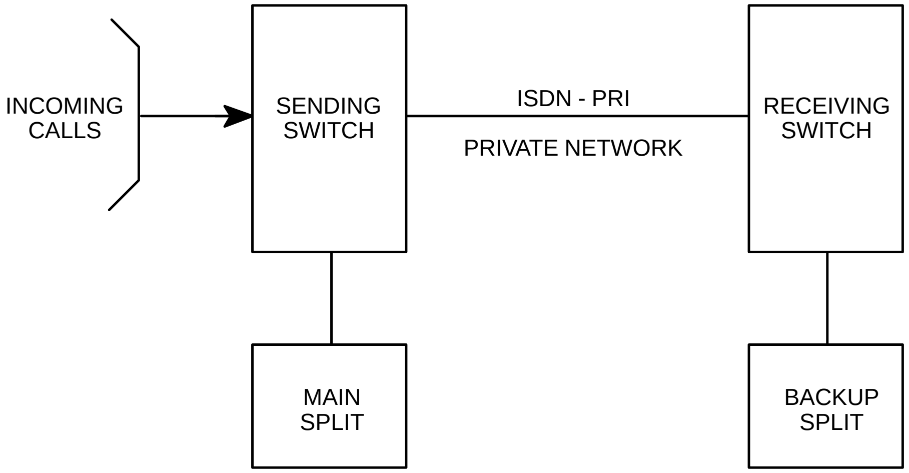
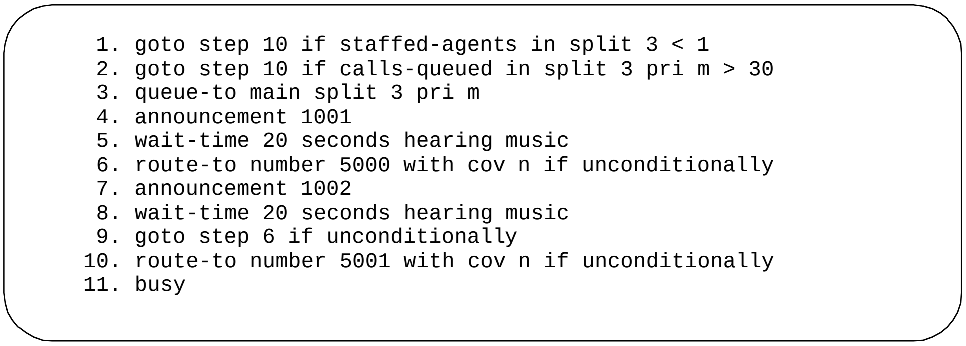


Figure 8-1. Two Switch LAI Connections

Sending Switch Operation

The vector(s) in the sending switch use the *goto* command to determine whether the call should be sent to the receiving switch. Recall that the *goto* command tests various outflow threshold conditions (such as number of calls queued). If the expressed condition is met, a branch is made to the appropriate *route to* command. This command sends the call to the receiving switch, which, as already noted, can accept or deny the call.

Here's an example of an outflow vector that might be included in a sending switch:



```
 1. goto step 10 if staffed-agents in split 3 < 1
 2. goto step 10 if calls-queued in split 3 pri m > 30
 3. queue-to main split 3 pri m
 4. announcement 1001
 5. wait-time 20 seconds hearing music
 6. route-to number 5000 with cov n if unconditionally
 7. announcement 1002
 8. wait-time 20 seconds hearing music
 9. goto step 6 if unconditionally
10. route-to number 5001 with cov n if unconditionally
11. busy
```

Figure 8-2. Sending Switch Outflow Vector

If split 3 has staffed agents, and if there are 30 or fewer calls queued to the split, Step 3 queues the call to the split's queue at a medium priority. In Step 4, an announcement that apologizes for the delay is provided. Vector processing then continues at Step 5, where music is provided for 20 seconds (Step 5). If the call is still not answered after 20 seconds, a Look-Ahead Interflow attempt is made in Step 6. If the call is accepted by the receiving switch, the call is removed from split 3's queue, and call control is passed to the receiving switch. If the receiving switch denies the call, the call remains in queue, and an announcement followed by music is provided. Step 9 sends control back to Step 6 unconditionally, thus allowing the LAI attempt in the latter step to be made as many times as necessary.

Returning to the first two vector steps, if split 3 does not have staffed agents, or if there are more than 30 calls queued to the split, control is passed to Step 10, which places an LAI call to a VDN (5001) at the receiving switch vector which, in this case, is designed to always accept calls during working hours. Step 11 provides a busy tone in case all trunks to the receiving location are busy, or the call is rejected because it is placed during nonworking hours.

Receiving Switch Operation

When the receiving switch receives the Look-Ahead Interflow request, the call first routes to a VDN. The VDN then maps the call to the receiving switch's inflow vector, and vector processing begins, starting with inflow checking. Inflow checking is enabled via conditional *goto* commands in the inflow vector. The decision to accept or deny a call can be based on checks such as any of the following:

- Number of staffed agents
- Number of available agents
- Time-of-day/day of the week

- Number of calls in split’s queue
- Average Speed of Answer
- Active VDN Calls
- Expected Wait Time
- ANI
- II-Digits

Once inflow checking is complete, acceptance of the Look-Ahead Interflow call is accomplished by executing any of the vector commands presented in the following table:

NOTE:
For each command in the next three tables, only one of the corresponding qualifications need be true for the command to effect the desired result (call acceptance, call denial, or no effect on such acceptance or denial).

Table 8-1. Call Acceptance Vector Commands

Call Acceptance Vector Command	Qualification
announcement	■ Announcement available ■ Queued for announcement ■ Retrying announcement
check-backup split	■ Call terminates to agent ■ Call queued to split
collect digits	■ Always
converse-on split	■ VRU answers the call ■ Call queued to converse split
disconnect	■ With *announcement* and announcement available ■ With *announcement* and queued for announcement ■ With *announcement* and retrying announcement
messaging split	■ Command successful ■ Call queued
queue-to main split	■ Call terminates to agent ■ Call queued to split
route-to	■ Terminates to valid local destination ■ Successfully seizes a non-PRI trunk ■ Results in a Look-Ahead Interflow call attempt, and the call is accepted by the far end switch
wait-time	■ Always (except *wait-time hearing i-silent* which is neutral)

If the receiving switch decides it is unable to accept the Look-Ahead Interflow call, call denial is accomplished by executing either of the vector commands listed in the following table. One recommendation: use *busy* instead of *disconnect* to allow for compatibility with similar network services such as ADR.

Table 8-2. Call Denial Vector Commands

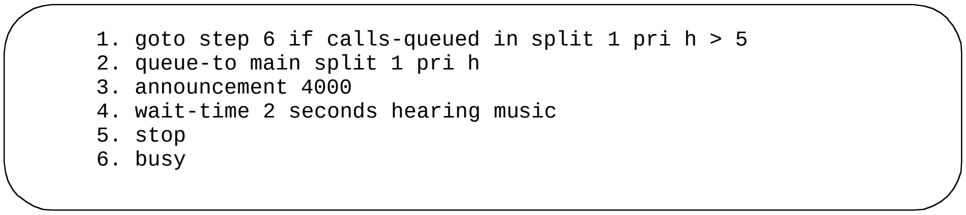
Call Denial Vector Command	**Qualification**
busy	■ Always
disconnect	■ With no announcement ■ With announcement but announcement unavailable

The vector commands presented in the next table are considered neutral because they do not generate either call acceptance or denial messages.

Table 8-3. Neutral Vector Commands

Neutral Vector Command	Qualification
adjunct routing	■ Always
announcement	■ Announcement unavailable
check-backup split	■ Call neither terminates nor queues **NOTE:** This occurs only if the command fails due either to a full split queue or to a split that is not vector-controlled.
converse-on split	■ Call neither terminates nor queues
goto step	■ Always
goto vector	■ Always
messaging split	■ Command failure
queue-to main split	■ Call neither terminates nor queues **NOTE:** This occurs only if the command fails due either to a full split queue or to a split that is not vector-controlled.
route-to	■ Unsuccessful termination ■ Trunk not seized ■ Look-Ahead Interflow call denied by far end switch
stop	■ Always
wait-time hearing i-silent	■ Always (New with G3V4) **NOTE:** This command is used following an *adjunct routing* command in applications where the adjunct decides whether to accept or reject the Look-Ahead calls.

The following is an example of an inflow vector that might be used by a receiving switch:



```
1. goto step 6 if calls-queued in split 1 pri h > 5
2. queue-to main split 1 pri h
3. announcement 4000
4. wait-time 2 seconds hearing music
5. stop
6. busy
```

Figure 8-3. Receiving Switch Inflow Vector

Step 1 of this inflow vector checks the inflow thresholds. The *goto step* command in Step 1 checks the number of queued calls in split 1. If the number of such calls is greater than 5, a branch is made to the *busy* command in Step 6. If executed, the *busy* command denies the call, and the receiving switch returns a call denial message to the sending switch. The sending switch, in turn, drops the Look-Ahead Interflow call attempt and then continues vector processing at the next vector step.

On the other hand, if the number of queued calls in split 1 is less than or equal to 5, the receiving switch returns a call acceptance message to the sending switch, and call control is passed to the receiving switch. Thereafter, the call is queued to split 1 in the receiving switch (Step 2). Once queued, the caller receives the appropriate announcement in Step 3 and is then provided with music until the call is answered by an agent or abandoned by the caller (Steps 4 and 5). (Remember, the *stop* command halts vector processing but does not drop the call.)

NOTE:

If the sending switch does not receive a call acceptance or call denial message within 120 seconds (G3 switches) or 10 seconds (G2 or network switches) after the Look-Ahead Interflow call request, the Look-Ahead Interflow attempt is dropped, and the sending switch continues vector processing at the next step.

Tandem Switch Configuration

Tandem Look-Ahead Interflow is implemented by using *route-to* commands that contain external destinations that route over ISDN-PRI facilities. This configuration is illustrated as follows:

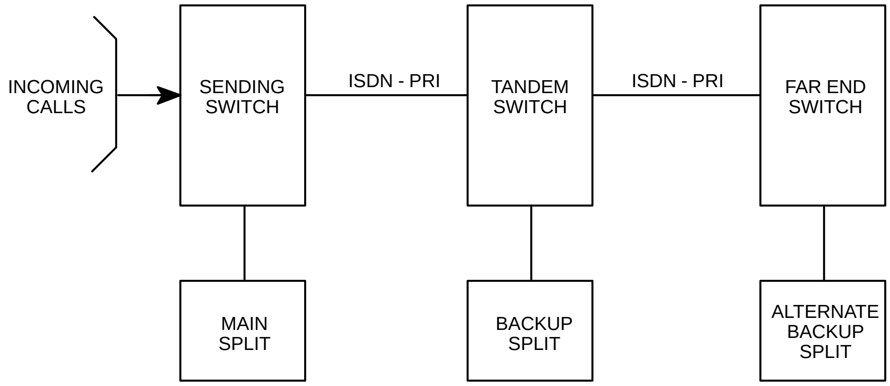


Figure 8-4. LAI Using a Tandem Switch

Sending Switch Operation

The sending switch is unaware that its Look-Ahead Interflow call is being tandemed to an alternate switch. The operation of the sending switch in the tandem switch configuration is the same as that in the two switch configuration.

Tandem Switch Operation

If the receiving switch executes a *route-to* command that routes the call over an ISDN facility before call acceptance, the *route-to* command is performed on a “look ahead” basis in the same manner as a sending switch. If the call is accepted at the far end switch, acceptance is passed to the sending switch, and call control is passed to the far end switch, along with tandeming of the original calling party information and the original DNIS name. If the call is denied, the next step of the tandem switch vector is executed.

Here’s an example of a tandem switch vector:

```
1. goto step 6 if calls-queued in split 30 pri h > 5
2. queue-to main split 30 pri h
3. announcement 200
4. wait-time 2 seconds hearing silence
5. stop
6. route-to number 4000 with cov n if unconditionally
7. busy
```

Figure 8-5. Tandem Switch Vector

Step 1 of this vector checks the inflow threshold. If the inflow criteria are acceptable, the vector flow drops to Step 2, where the *queue-to main split* command provides acceptance to the sending switch. Thereafter, Steps 3 through 5 provide a typical queuing-wait scheme.

If, however, the inflow criteria are not acceptable, a branch is made to Step 6. The *route-to* command in this step checks another switch enabled with Look-Ahead Interflow on a “look-ahead” basis. If this “far end” switch rejects the call, a denial message is relayed back to the sending switch, which then drops the Look-Ahead Interflow call attempt. On the other hand, if the far end switch accepts the call, an acceptance message is relayed back to the sending switch.

Note that no ringback is provided in this tandem switch vector. This is necessary so that an acceptance message is not returned to the sending switch. This operation is appropriate for the caller because the sending switch has already returned an announcement before a Look-Ahead Interflow attempt is made to the receiving switch.

Be sure the sending switch is not used as a backup location for the tandem switch or for any of the far end switches. Should the sending switch be administered in this manner, all trunk facilities could be tied up by a single call.

Far End Switch Operation

The far end switch is also unaware that tandeming has taken place. The far end switch functions in the same manner as the receiving switch within the two switch configuration.

DNIS and VDN Override in an LAI Environment

Look-Ahead Interflow handles Dialed Number Identification Service (DNIS) and VDN Override in various ways, depending on a number of different characteristics relevant to the call. Recall that DNIS (described in Chapter 3) allows any agent with a display-equipped voice terminal to receive visual displays that specify the name of the called VDN. Also recall that VDN Override in its basic form (as described in Chapter 3) allows the name of a subsequently routed to VDN to be displayed to the answering agent instead of the name of the originally called VDN.

The following sections discuss how Look-Ahead Interflow handles DNIS and VDN Override.

Answering Agent’s Display

For Look-Ahead Interflow, the DNIS name (the “called” VDN name from the sending switch) is presented on the answering agent’s display on the receiving switch if all of the following are true:

- Look-Ahead Interflow option is enabled

- Call routes to a VDN
- DNIS name field is not blank

The type of DNIS information that is displayed depends upon a number of different scenarios. This information is presented in the following table:

Table 8-4. DNIS Information Displayed for LAI Scenarios

Scenario	Information Displayed
Tandemed Look-Ahead Interflow call	■ Look-Ahead Interflow DNIS information from the original Look-Ahead Interflow call
No redirection at the sending switch	■ VDN name according to Override rules at the sending switch
Redirection at the sending switch	■ Original VDN name, or ■ If multiple VDNs are accessed, the name of the VDN last accessed via a *route-to* command
Sending switch sends a blank DNIS Name field (that is, a name is not assigned to the sending switch "called" VDN).	Name associated with the receiving VDN. (This name can be changed according to the rules of VDN Override at the receiving switch.)

NOTE:
VDNs that map to vectors that place Look-Ahead Interflow calls must have their ISDN Calling Party Number (CPN) prefixes administered. If an ISDN CPN prefix is not administered, the assigned VDN name is not sent, instead a DNIS of all blank space characters is sent and displayed on the answering agent's terminal.

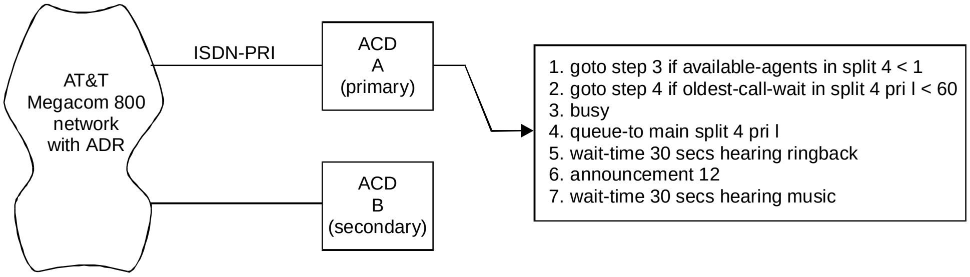
Originator's Display

For internal calls, the originator's display contains the same information as for Basic Call Vectoring. However, it is possible the originator might receive undesirable display updates during Look-Ahead Interflow call attempts. To avoid this scenario, ensure that the Look-Ahead Interflow calls are going out over trunk groups with the "Outgoing Display" field set to *no*. When the display field is set to no, internal callers calling that trunk group will see the digits that they dialed on their display.

Look-Ahead Interflow with AT&T Network Alternate Destination Redirection (ADR)

Call Vectoring and Look-Ahead Interflow are compatible with and supplement the AT&T network services Alternate Destination Redirection (ADR) rerouting feature. ADR uses ISDN-PRI connectivity with the customer-premises PBX/ACD in the same manner as Look-Ahead Interflow to allow the receiving PBX/ACD to indicate whether a call is to be accepted or rejected. The same type of vector used as a receiving ACD for Look-Ahead Interflow is used at the ADR-receiving ACD. If the call is accepted, it is connected through from the network to the PBX/ACD. If the call is rejected, the network routing number is translated to another number, which routes the call to the alternate location within dialing-plan constraints. ADR allows for only one alternate location. Look-Ahead Interflow can be used at the alternate location to test other locations for less busy conditions.

The following figure shows configuration for a multilocation application.



```
1. goto step 3 if available-agents in split 4 < 1
2. goto step 4 if oldest-call-wait in split 4 pri l < 60
3. busy
4. queue-to main split 4 pri l
5. wait-time 30 secs hearing ringback
6. announcement 12
7. wait-time 30 secs hearing music
```

Figure 8-6. ADR Example

The AT&T network requires ISDN-PRI connectivity to primary location A. Connection to secondary location B may or may not be ISDN-PRI. ADR attempts to route the call to location A over the ISDN-PRI link using a routing number that selects a VDN assigned to the receiving vector shown. When the routing attempt is made, Call Vectoring starts processing the vector. The example then proceeds at location A as follows:

1. Step 1 checks for staffing of the ACD split, and branches to step 3 if it is not staffed.
2. If the ACD split is staffed, step 2 checks the oldest call waiting time in the split, and branches to step 4 if it is less than 60 seconds.
3. If the ACD split is unstaffed or if the oldest call waiting time is 60 seconds or more, step 3 rejects the call (returns a busy indication to the network).

4. If the oldest call waiting time is less than 60 seconds, step 4 accepts the call and queues it. ADR then connects the call through to the receiving PBX/ACD.
5. Steps 5-7 provide ringback, announcement, and music to the caller.

If the vector at location A rejects the call by sending a busy indication back to the network over the ISDN-PRI link, ADR reroutes the call to location B, which must accept the call. If location B is closed or too busy to take the call, location B can use Call Vectoring and Look-Ahead Interflow to check other locations. If other locations exist and can take the call, location b can forward the call. If other locations do not exist or cannot take the call, location B can use Call Vectoring to route the call to location A. If location A is not open, location B can use Call Vectoring to provide an announcement or busy tone to the caller.

Adjunct Routing

9

Introduction

Adjunct Routing allows an associated adjunct to make a call routing decision when it encounters an *adjunct routing* vector command during vector processing.

An adjunct is any processor connected to a switch that can use ASAI features. The adjunct makes a routing decision according to caller information and/or agent availability, and it returns the routing response to the switch.

Adjunct Routing can be used in conjunction with Call Prompting and Look-Ahead Interflow. When coupled with Call Prompting, Adjunct Routing can pass up to 16 digits that have been collected from the last relevant *collect digits* vector command. When coupled with Look-Ahead Interflow, Adjunct Routing can pass the LAI information element that was passed from the originating switch in the ISDN message.

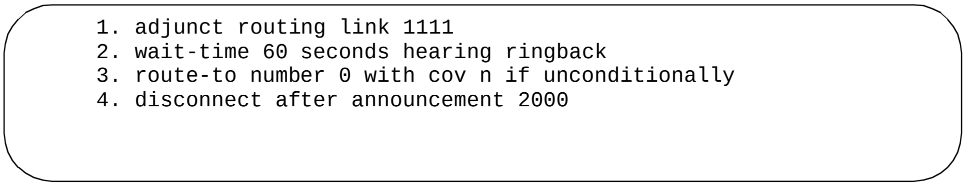
Functions and Examples

The *adjunct routing* command provides a means for an adjunct ASAI processor to specify the destination of a call. The switch provides information in an ASAI route request message that the ASAI adjunct can use to first access a database and then determine a route for the call. In a typical application, the ASAI adjunct might use the dialed number, the Calling Party Number (CPN/BN), or the digits collected via Call Prompting to access customer information and thereby determine the call route.

An adjunct specified in an *adjunct routing* command can route a call to an internal number, an external number, a split, a VDN, an announcement extension, or a particular agent. An adjunct can also provide priority ringing and priority queuing.

Sending the Call Route Request

Here's an example of a simple vector that uses adjunct routing:



```
1. adjunct routing link 1111
2. wait-time 60 seconds hearing ringback
3. route-to number 0 with cov n if unconditionally
4. disconnect after announcement 2000
```

Figure 9-1. Adjunct Routing Vector

In this vector, *1111* is the extension number of an ASAI link. Each ASAI link has a unique extension number, even in a configuration where there might be multiple ASAI links to the same adjunct.

When a call encounters an *adjunct routing* command, and if the call is not queued to a split, the switch sends an ASAI message requesting a call route over the specified adjunct link. The following list identifies the contents of the message, along with a comment or a brief explanation for each item:

- **Calling number information**. Calling party number or billing number (CPN/BN) provided by ISDN-PRI facilities, or by the administered destination for other trunk facilities. If the call originates from a local switch extension, this extension is the number dialed (after digit translation).
- **Called number**. Originally called extension (if a call is forwarded to a VDN), or the first VDN through which the call was routed (if the call was not forwarded to the VDN).
- **Routing VDN**. Last VDN that routed the call to the vector that contains the *adjunct routing* command.
- **Call identifier**. ASAI identifier that permits the ASAI adjunct to track multiple calls via either Event Notification or Third Party Call Control. (See *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204, or the *DEFINITY Generic 3 CallVisor ASAI Technical Reference*, 555-230-220, for more information on ASAI.)
- **Look-Ahead Interflow information** (if any). Includes the original VDN display information and the priority level of the call at the originating switch. (See Chapter 8.)
- **Digits collected via Call Prompting** (if any; maximum of 16 digits). Digits are collected by the most recent *collect digits* command. (See Chapter 5.)

If the call is queued, the *adjunct routing* step is ignored, and vector processing continues at the next vector step.

NOTE:
For reasons that we discuss later in this chapter, you should always include a *wait-time* step, *announcement* or another *adjunct routing* step (G3V3 and later releases) after an *adjunct routing* step.

Effects of ASAI Link Failure on Vector Processing

If the ASAI link specified in the *adjunct routing* step is down, the step is skipped.

An ASAI link failure can change the manner in which subsequent treatment (that is, *announcement* and/or *wait-time*) steps (if any) in the vector are processed. In some cases, such processing is influenced by the position the treatment steps occupy in the vector. In other cases, the positioning of these commands along with their relationship to specific *goto* commands come into play. For example, any *announcement* or *wait-time* step that immediately follows an *adjunct routing* step whose ASAI link is down is skipped.

NOTE:
In view of the previous thought, the second step after the *adjunct routing* step is often implemented as a default treatment. In Figure 9-1, for example, the default treatment in Step 3 is a route to an attendant. After the switch recognizes that the ASAI link is down, this step executes. immediately. (It can take up to 6 minutes for the switch to recognize that the link is down.) Otherwise, the step executes only if the application does not respond with a route within 60 seconds.

On the other hand, if a *goto* step follows such an *adjunct routing* step, the switch executes the *goto* step and then skips various treatment steps according to their position in the vector and based on the action of the *goto* step. Specifically, if the *goto* step *succeeds*, the switch skips any *announcement* or *wait-time* step that is the first non-*goto* step branched to by the *goto* step.

NOTE:
Actually, the first step to which a *goto* step is usually designed to branch is a nontreatment step (that is, a step containing a command other than a *wait-time* or an *announcement* command). Thus, the skipping of a treatment step according to the scenario described just before this note rarely occurs.

On the other hand, if the *goto* step fails, the switch skips any *announcement* or *wait-time* step that *immediately* follows the *goto* step.

NOTE:
The *goto* step that fails can be at the end of a sequence of *goto* steps that branch to each other.

The following vectors can be used to illustrate the processes just described.

```
VDN (extension=1040  name=‘‘Ad Route’’  vector=40)
Vector 40
      1. adjunct routing link 1000 (link is down)
      2. wait-time 10 seconds hearing ringback
      3. adjunct routing link 2000 (link is down)
      4. goto step 7 if available-agents in split 20 < 1
      5. wait-time 10 seconds hearing ringback
      6. goto vector 50 if unconditionally
      7. goto step 10 if calls-queued in split 20 pri l > 50
      8. announcement 4001
      9. goto vector 50 if unconditionally
     10. route-to number 6000 with cov n if unconditionally
VDN (extension=6000  name=‘‘Message’’  vector=60)
Vector 60
      1. announcement 4000 (‘‘We’re sorry.  We
         are still unable to connect you to an agent.
         If you’d like to leave a message, please do so
         after the tone.  Otherwise, please call back
         weekdays between 8:00 A.M. and 5:00 P.M.
         Thank you.’’)
      2. wait-time 6 seconds hearing silence
      3. messaging split 18 for extension 1500
      4. announcement 4010 (‘‘We’re sorry.  We
         were unable to connect you to our voice mail.
         If you’d like to try to leave a message again,
         please do so after the tone.  Otherwise, please
         call back weekdays between 8:00 A.M. and 5:00 P.M.
         Thank you.’’)
      5. goto step 2 if unconditionally
```

Figure 9-2. Skipping/Non-Skipping of Treatment Commands with ASAI Link Down

Because we are assuming the adjunct link is down, the *adjunct routing* command in Step 1 fails. Because the *wait-time* command in Step 2 immediately follows an *adjunct routing* command whose adjunct link is down, the *wait-time* step is skipped. Step 3 contains another *adjunct routing* command whose adjunct link is also down. As a result, the step fails, and control is passed to the *goto step* command in Step 4, which is automatically executed.

Now, let’s assume the *goto step* command in Step 4 is not successful (that is, no branch is made because there is at least one available agent in split 20). In such a case, the *wait-time* step (Step 5) following the unsuccessful *goto* step is skipped, and control is passed to the *goto vector* command in Step 6. This step then routes the call to vector 50 (not shown), which is designed to queue the call and provide standard call treatment.

On the other hand, let’s backtrack and assume the *goto step* command in Step 4 is successful. In such a case, control is passed to Step 7, where another *goto step* command determines whether there are more than 50 calls in split 20. If so, control is sent to Step 10, where the *route-to number* command sends the call to vector 60, which allows the caller to leave a recorded message. The first step of vector 60 contains an *announcement* command, which is not skipped, since the treatment step is *not* the first non-*goto* step branched to by a *goto* step that follows an adjunct routing command whose ASAI link is down [the *route-to number* step (Step 10) in vector 40 is the first such step]. Similarly, neither the *wait-time* step (Step 2) nor the second *announcement* step (Step 4) is skipped.

Returning to Step 7 in vector 40, let's assume that there are 50 or fewer calls in queue. In such a case, the *goto* step fails and, as a result, the *announcement* step (Step 8) that immediately follows this step is skipped. (Remember, even though this *goto* step does not immediately follow the *adjunct routing* step, the former step qualifies as a test case because it is branched to by another *goto* step that does immediately follow the adjunct ro*uting* step.) Thereafter, the *goto vector* step (Step 9) routes the call again to vector 50, which is designed to queue the call and provide standard call treatment.

The following table summarizes the procedures discussed in this section. In so doing, the table refers back to the vectors presented in the previous screen.

Table 9-1. Example of the Relationship Between Treatment Steps and Goto Steps that Follow

Goto Step #(s)	Disposition of Goto Step(s)	Treatment Step #(s)	Disposition of Treatment Step(s)	Reason for Disposition of Treatment Step(s)
4	Fails	5	Skipped	Immediately follows failed *goto* step.
4, 7	First step succeeds, second step fails.	8	Skipped	Immediately follows failed *goto* step.
	Both steps succeed.	1, 2 (both of vector 60)	Each step is executed.	Not the first non-*goto* steps accessed by a successful *goto* step. **NOTE:** Step 4 of vector 60 is also executed if the *messaging split* step (Step 3) fails.

Awaiting the Response to the Call Route Request

After the switch sends a route request to the ASAI adjunct, vector processing continues with the following vector steps.

The step that follows the *adjunct routing* step in effect determines the maximum length of time the switch will wait for the ASAI adjunct to reply with a call route.

Accordingly, you should almost always include either a *wait-time* step or an *announcement* step immediately after an *adjunct routing* step. Moreover, the switch cancels the route request if vector processing encounters a step containing one of the following commands:

- *adjunct routing (G3V2 and earlier releases.) G3V3 and later releases allow multiple outstanding adjunct route requests.*
- *busy*
- *check-backup split*
- *converse-on split*
- *queue-to main split*
- *collect digits*
- *disconnect*
- *messaging split*
- *route-to*

NOTE:
Actually, if another *adjunct routing* step is encountered, the route request information is not lost. Although the initial route request is cancelled, a second route request is sent, and this route request includes the same information included in the first route request.

If a valid call route is received by the switch before one of the vector commands in the previous list is executed, the switch routes the call to the destination specified by the adjunct route. Otherwise, the route request is terminated without affecting vector processing.

Let's return to our suggested strategy of including a treatment step after the *adjunct routing* step. Accordingly, here's an example presented earlier that illustrates this approach:

```
1.  adjunct routing link 1111
2.  wait-time 60 seconds hearing music
3.  route-to number 0 with cov n if unconditionally
4.  disconnect after announcement 2000
```

Figure 9-3.   Treatment Step Used as a Delay for Adjunct Routing

In Step 2 of this example, the *wait-time* command specifies a delay period of 60 seconds. As a result, the switch in this case will wait up to 60 seconds to receive a reply from the adjunct. On the other hand, replacing the *wait-time* command in Step 2 with an *announcement* command enables the switch to wait for no longer than the length of time it takes for the announcement to complete. Accordingly, judgement should be used in determining which of the treatment commands is best for the particular application.

Finally, note that the adjunct can also decide to not route a call by rejecting (that is, negatively acknowledging) the route request sent by the switch. Upon receiving a route request rejection, the switch terminates the *announcement* or *wait-time* step that is being executed for the call and then continues with the next vector step.

Receiving and Implementing the Call Route

When the switch receives a call route (destination) from the ASAI adjunct, the switch first validates the route as follows:

1. The switch verifies the VDN's COR permits the call to be terminated at the adjunct-supplied destination.
2. The switch verifies that the adjunct-supplied information (destination number, ACD split, TAC/AAR/ARS access code, etc.) for the route is valid. This includes checking that the destination is compatible with the dial plan, and that the options specified by the adjunct are correct.
3. If the ASAI adjunct specifies the Direct Agent Call (DAC) option, the destination number (agent) must be logged into the adjunct-specified ACD split.
4. If the destination for the call is external, the switch verifies the trunk is available for the call.

If any of these conditions are not met, the route validation fails, and the switch does the following:

1. Discards the route.
2. Notifies the ASAI adjunct that the route is invalid.
3. Continues with vector processing.

If the route is valid, the switch does the following:

1. Terminates vector processing immediately.
2. Notifies the ASAI adjunct that the route is accepted.
3. Routes the call to the destination specified by the ASAI adjunct.

When the call is routed, the caller hears normal call progress tones and feedback. However, if the call is routed to an extension with no available call appearances and no coverage path, the caller hears the busy tone. Any other features that may be in effect at the adjunct-supplied destination (such as Send-All-Calls or Call Forwarding) interact with the routed call.

NOTE:
The operation described in the previous paragraph is similar to that for the *route-to with coverage* command.

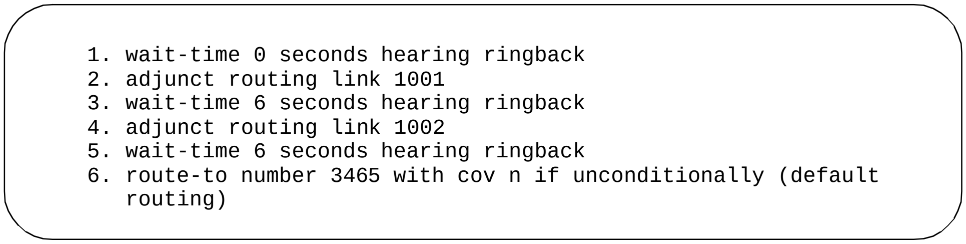
Multiple Outstanding Route Requests

This feature allows multiple ASAI Route Requests for the same call to be active at the same time (simultaneously). The Route Requests can be over the same or different ASAI links.

The requests are all made from the same vector. They must be specified back-to-back, without intermediate (*wait-time, announcement, goto,* or *stop*) steps. If the adjunct routing commands are not specified back-to-back, current adjunct routing functionality will apply (that is, previous outstanding route requests will be cancelled when an adjunct routing vector step is executed).

The first Route Select response received by the switch will be used as the route for the call, and all other outstanding Route Requests for the call will be cancelled.

This capability increases the redundancy options available with ASAI. Prior to G3V3, adjunct routing applications that wanted to have a backup link tested if the primary link was down and then executed the adjunct routing for the backup link with a vector as shown in Figure 9-4. Note that the route request over link 1001 is cancelled when step 4 is executed (because of the intermediate step 3).



```
1. wait-time 0 seconds hearing ringback
2. adjunct routing link 1001
3. wait-time 6 seconds hearing ringback
4. adjunct routing link 1002
5. wait-time 6 seconds hearing ringback
6. route-to number 3465 with cov n if unconditionally (default
   routing)
```

Figure 9-4. Sample Adjunct Routing Vector with Redundancy

With Multiple Outstanding Route Requests, multiple adjuncts can process the route call request without waiting for the first route attempt to fail. An application can make use of this feature to distribute the incoming call load evenly across adjuncts, based on the adjunct’s current CPU load.

User Scenarios

Figure 9-5 shows a typical vector where multiple adjunct route requests to multiple links will be active at the same time. The first adjunct to route the call is the active adjunct (that is, it specifies which VDN the call should be routed to at that point).

```
1. wait-time 0 seconds hearing ringback
2. adjunct routing link 1001
3. adjunct routing link 1002
4. adjunct routing link 1003
5. wait-time 6 seconds hearing ringback
6. route-to number 1847 with cov n if unconditionally (default
   routing)
```

Figure 9-5. Sample Adjunct Routing Vector with Redundancy in G3V3 and later releases

Expert Agent Selection

10

Introduction

Expert Agent Selection (EAS) allows Call Center managers to provide the best possible telephone service to the caller by matching the needs of the caller with the skills or talents of the agents. Caller needs and agent skills are matched via Call Vectoring. Skills can be thought of as needs or desires on the part of the caller and also the abilities or talents of the agent.

Matching the call to an agent with the appropriate skills reduces transfers and call-holding time. Accordingly, customer satisfaction is increased. Also, since an entire agent group need not be trained at the same time for the same skills, employee satisfaction is increased.

EAS requires ACD and Call Vectoring. All of the existing ACD features and Call Vectoring capabilities can be used within EAS applications.

As with Call Vectoring calls, EAS calls are directed to VDNs, which in turn point to vectors. However, unlike Basic Call Vectoring, skills can be assigned in EAS to VDNs, or they can be associated with vector steps to represent caller needs. As for Call Vectoring calls, EAS calls are queued to ACD hunt groups. However, with EAS enabled, ACD hunt groups are called ‘‘skill hunt groups’’ instead of ‘‘splits.’’

Skill hunt groups deliver calls to EAS agents. Agent skills are administered on the Agent LoginID form, usually by the switch administrator.

NOTE:
These are the same login IDs used by CMS (R3V2 and later releases) and BCMS (G3V3 and later releases).

Logical Agent implies that voice terminals are no longer preassigned to hunt groups; only when the agent logs in does the terminal become associated with all of the skill hunt groups assigned to the Agent Login ID.

With EAS optioned and enabled, ACD calls can also be directed to a particular agent, instead of to the skill hunt group, by using the Direct Agent Calling feature. The Direct Agent call is treated like an ACD call, but it waits in queue for a specific agent to become available. Direct Agent calls have a higher priority than skill hunt group calls.

This chapter describes EAS, and it explains, via a number of examples, how EAS is implemented. The chapter also discusses EAS upgrades. However, before you start with this chapter, you should take note of the following:

- With EAS, skill hunt groups replace splits. Skill hunt groups and splits cannot be administered simultaneously on a switch. This implies that all ACD hunt groups must be administered as either splits or skills. If EAS is optioned, all ACD hunt groups must be skill hunt groups.
- With EAS, all skill hunt groups must be vector-controlled.
- With EAS, non-ACD hunt groups are allowed, but they cannot be vector-controlled.
- Agent Login IDs are extensions in the dial plan, and they decrease the total number of stations that can be administered.
- With EAS, agents have a new login procedure and a single set of work mode buttons, regardless of the number of skills assigned to the agents.
- Skill hunt groups can distribute a call to the most-idle agent or to the most-idle-primary agent. Direct Department Call (DDC) distribution is not allowed for skill hunt groups.

For information on converting a Call Center to EAS, refer to Appendix K.

Identifying Caller Needs

Caller needs for a particular call can be identified by any of the following methods:

- Interpreting information (in the form of DNIS digits or ISDN messages) passed from the network
- Processing Call Prompting digits or digits entered at a Voice Response Unit (VRU)
- Using ASAI or a VRU (such as CONVERSANT) in a host database lookup

To illustrate how a Call Center manager might match caller needs and agent skills (which can be viewed as capabilities needed from the caller’s perspective), let’s assume that a Call Center receives inbound calls from auto club members who speak Spanish or English. The callers in this case either need to plan a vacation route or have car trouble and are calling for assistance.

Table 10-1. Example of Caller Need—Agent Skill Matching

Caller Need	Capability Needed
Tourist Information	Knowledge of the Region
To Speak Spanish	Bilingual
Emergency Assistance	Handle stressful callers
(Tow Truck)	(Access to Dispatch Systems)

Note in each case that the capability needed is such that it can accommodate the caller need. Let’s examine the strategy behind matching these caller needs to capabilities as deemed appropriate by the Call Center manager:

- **Tourist Information/Knowledge of the Region**

  Travelers may need information while traveling or regarding a future trip. All assigned agents can provide this information.

- **To Speak Spanish/Bilingual**

  Separate numbers are published and used as part of Spanish membership information, or Call Prompting is used after a general number is dialed.

- **Emergency Assistance/Handle Stressful Callers**

  Separate emergency Road Service numbers are published and used, or Call Prompting is used after a general number is dialed. (For example, a number is provided for tow trucking.)

Note that the Call Center chose to implement Call Prompting to identify Spanish-speaking callers and callers who require emergency assistance. This allows for quicker and more specialized treatment and therefore better satisfies the caller’s needs.

In addition, some customers might prefer to speak to the agent he or she spoke to on a previous call. To accommodate this request, a Call Center manager can implement Direct Inward Dialing (DID) at the Call Center. Also, direct agent calling can be used to direct a call to a specific agent.

The following sections explain further how caller needs are identified.

DNIS/ISDN Called Party

Recall from Chapter 3 that a set of DNIS digits can be interpreted as a VDN. The following table presents four services and their corresponding telephone number (including DNIS digits) that might be provided to the caller.

Table 10-2. Examples of Services and Corresponding DNIS Digits

Service	Telephone Number	Corresponding DNIS
Emergency Road Service (English)	800-765-1111	6001
Emergency Road Service (Spanish)	800-765-2222	6002
Route Planning (English)	800-765-3333	6003
Route Planning (Spanish)	800-765-4444	6004
General (Call Prompting)	800-765-5555	6005

NOTE:
DNIS digits must be extensions that are reflected in the dial plan.

Call Prompting/VRU Digits

The Call Prompting/VRU digits are entered by the caller in response to any recorded question about a caller’s needs. For example, a hotline for a product may request that a product code be entered, or a travel service may request a 2-digit state code to indicate the state to which the caller would like to travel. The following figure provides a prompt that encourages the caller to enter the appropriate Call Prompting digit for the needed service from the auto club.

Table 10-3. Example of a Prompt for Entering Call Prompting Digits

“For emergency road service, dial 1.

Para asistencia con su automovil, marque el dos.

For travel route directions, dial 3.

Para informacion sobre rutas, marque el cuatro.’’

In this example, the caller is requested to dial the appropriate number between ‘‘1’’ and ‘‘4 (cuatro),’’ inclusive.

Host Database Lookup

A host database lookup uses DNIS and ANI (calling party’s number) to determine what skills are required or even the agent desired. For example, the database may show that the caller speaks Spanish and has been working with Agent 1367. To access host information, either ASAI or a VRU in conjunction with a *converse-on skill* step is used.

Direct Agent Calling

Some callers might prefer to speak to the agent he or she spoke to on a previous call. To accommodate this request, a Call Center manager can implement Direct Agent Calling at the Call Center. The caller can dial the LoginID of the desired agent directly. As an alternative, the caller can dial an 800 number and then be prompted to enter the LoginID of the desired agent. Vectors can be designed to handle the Call Prompting function.

Functions and Examples

This section explains how EAS is implemented. To this purpose, skill administration, the delivering of calls to a skill queue, and the routing of calls to an agent are discussed.

Administering Skills

A skill is an attribute that is:

- Administered as a skill hunt group
- Administered to VDNs (VDN skill preference)
- Assigned to agents (agent skill)

A skill hunt group is administered for each skill. A skill hunt group is a set of agents trained to meet particular customer needs.

In G3s PBP and G3vs PBP, a skill is assigned a number from ‘‘1’’ through ‘‘24’’ inclusive. In G3i, a skill is assigned a number from ‘‘1’’ through ‘‘99’’ inclusive. In G3r, a skill is assigned a number from ‘‘1’’ through ‘‘255’’ inclusive. Generally, if the ability “Spanish speaking’’ is assigned to skill 127, for example, it follows that Agent skill 127 and VDN skill 127 both signify “Spanish speaking.’’ However, note that the agent skill might be assigned a skill term that is broader than that for the corresponding VDN skill. For example, Agent skill 127 might be labeled ‘‘bilingual.’’ The implication is that agents with skill 127 can handle calls from

Spanish callers as well as from callers who speak another language (probably English).

Skills for an application can be illustrated via a table. The following table presents a very abbreviated example of such a skill distribution for an auto club. We will refer back to this table several times in this chapter.

Table 10-4. Example of a Skill Table for an Auto Club

Supergroup-99	
Emergency Road Service-Bilingual-22	**Route Planning-Bilingual-44**
Emergency Road Service-English-11	Route Planning-English-33

In this table, five skills are defined. Each skill indicates knowledge or an ability (on the part of the agent) or a need for knowledge (on the part of the caller) vis-a-vis an auto club. One or more of these skills can be attributed to the agent according to the agent's expertise with the corresponding highway service(s) and his or her language-speaking ability. Similarly, one or more of these skills can be considered ''needs'' on the part of the caller.

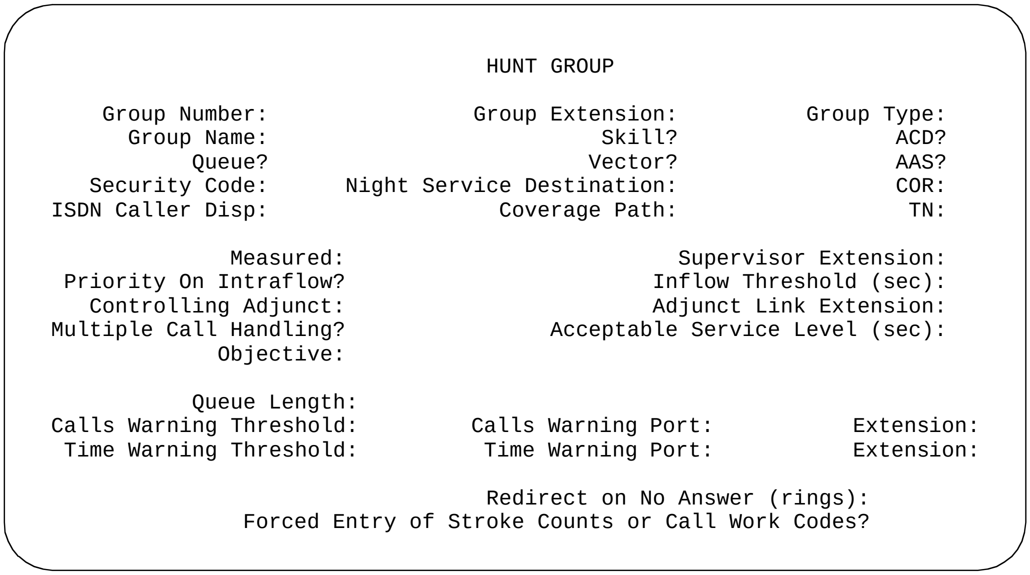
The previous table is arranged in such a manner that the agents at the top level have the broadest knowledge (that is, these agents can handle emergency road service and route planning calls and can speak Spanish). The top level (skill group) here is called ''Supergroup,'' and it contains agents who, as a group, can take any type of call regarding the auto club. Accordingly, this skill group serves as a ''backup'' skill group. As you descend through the table, each sublevel corresponds to a group of agents who have more specific skills and can therefore take more specialized calls.

Calls can be distributed to the most-idle agent by using either the Uniform Call Distribution (UCD) option or the Expert Agent Distribution (EAD) option. EAD distributes calls from a skill hunt group to agents to whom the relevant skill is assigned as a primary skill before distributing calls to agents to whom the skill is assigned as a secondary skill. Skills assigned to an agent as ''primary'' indicate a higher level of expertise or preference by the agent than any ''secondary'' skills assigned to that agent.

Agents are always given a preference for primary skill calls. With EAD optioned, callers are given a preference for primary skill agents. Such preferences provide the best caller—agent match. With UCD optioned, calls are sent to the most-idle agent with a primary or secondary skill. This scenario provides a more even distribution to calls and therefore keeps agents equally busy.

Multiple Call Handling on Request (G3V3 and later releases) and Forced Multiple Call Handling (G3V4 and later releases) allow an agent to receive additional ACD calls either after putting a call on hold, or when active on another ACD call. Forced Multiple Call Handling can be used to give priority to an ACD call over an in-progress non-ACD call, or to give priority to a call from one skill over an in-progress call from a different skill. See “Multiple Call Handling” in the *DEFINITY Communications System Generic 3 Feature Description,* 555-230-204 for more information.

To administer skills, the relevant Hunt Group form must be completed. The form appears as follows:



```
                                HUNT GROUP

    Group Number:             Group Extension:         Group Type:
      Group Name:                       Skill?                ACD?
          Queue?                       Vector?                AAS?
   Security Code:     Night Service Destination:              COR:
 ISDN Caller Disp:                Coverage Path:               TN:

             Measured:                         Supervisor Extension:
  Priority On Intraflow?                     Inflow Threshold (sec):
   Controlling Adjunct:                      Adjunct Link Extension:
 Multiple Call Handling?             Acceptable Service Level (sec):
            Objective:

           Queue Length:
 Calls Warning Threshold:         Calls Warning Port:          Extension:
  Time Warning Threshold:          Time Warning Port:          Extension:

                                Redirect on No Answer (rings):
                  Forced Entry of Stroke Counts or Call Work Codes?
```

Figure 10-1. Hunt Group Form with Expert Agent Selection Optioned

The skill, ACD, and Vector fields must all contain ‘‘y.’’ Instructions for completing this form are included in *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

VDN Skills

VDN skills (along with certain vector commands) define the requirements for routing calls to an ACD agent with a particular set of skills. A maximum of three different skills can be administered to a VDN. Skills administered to a VDN are commonly called VDN skill preferences. VDN skill preferences are labeled ‘‘1st,’’ ‘‘2nd,’’ and ‘‘3rd.’’

NOTE:
While skills can be optionally assigned to VDNs, the vector controls when and to what VDN skill the call queues. The application of VDN skills is described later in this chapter.

It is recommended that the 1st skill administered to a VDN be the skill required or desired to service a call to that VDN. The 2nd and 3rd skills administered to a VDN should represent other skills that are allowed to handle calls to that VDN. All skills to a VDN are optionally administered.

The following table illustrates how skill preferences can be assigned to the five VDNs used for the auto club that we discussed earlier. For each VDN, the corresponding call type and the number of the vector to which the VDN points are indicated. Be sure to refer back to Table 10-4 for a description of each skill.

Table 10-5. Example of VDN Skill Preferences Assignments

		Skill Preferences			
Call type	**VDN**	**1st**	**2nd**	**3rd**	**Vector**
General number	5555				1
Emergency Road Service (English)	1111	11	22	99	3
Emergency Road Service (Spanish)	2222	22		99	2
Route Planning (English)	3333	33	44	99	3
Route Planning (Spanish)	4444	44	99		2

From the table, note that two VDNs point to Vector 3, two VDNs point to Vector 2, and one VDN points to Vector 1. Note also that a 1st and 3rd VDN skill Preference, but no 2nd VDN skill Preference, are assigned to VDN 2222. Such a scenario implies that the call to this VDN (if not already answered) will wait longer before queuing to the backup skill (Supergroup-99, in our example), provided the vector is designed to execute accordingly.

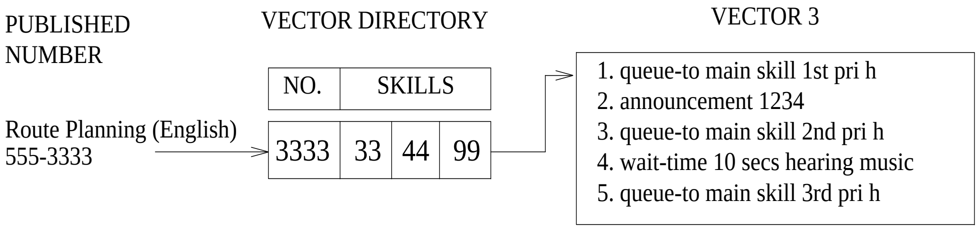
Now, let’s take a look at the following table, which illustrates the skill preferences assigned for one specific VDN (3333) that is used for the auto club:

Table 10-6. Skill Preferences Assignments for VDN 3333

		VDN 3333 - Skill Preferences
1st:	33	Directed to an agent who is knowledgeable about Route Planning and speaks English
2nd:	44	Directed to an agent who is knowledgeable about Route Planning and is bilingual
3rd:	99	Directed to an agent who can field all calls

In this table, note that the first VDN skill Preference corresponds to a knowledge area that could be considered a ‘‘subset’’ of the knowledge area represented by the second (and, taking it a step further, the third) Preference. Similarly, the second VDN skill Preference corresponds to a knowledge area that could be considered a "subset" of the knowledge area represented by the third Preference. Such an approach is commonly used to assign VDN skill preferences. The result of this approach is that the longer a call waits, the larger the pool of agents that the ACD considers for handling the call.

Now, recall that the vector number to which each VDN associated with the auto club has already been provided in Table 10-5 of this section. A quick glance at the table shows that VDN 3333 points to Vector 3. As such, the skill requirements associated with the VDN are ‘‘passed along’’ to the vector. This process can be illustrated as follows:



1. queue-to main skill 1st pri h
2. announcement 1234
3. queue-to main skill 2nd pri h
4. wait-time 10 secs hearing music
5. queue-to main skill 3rd pri h

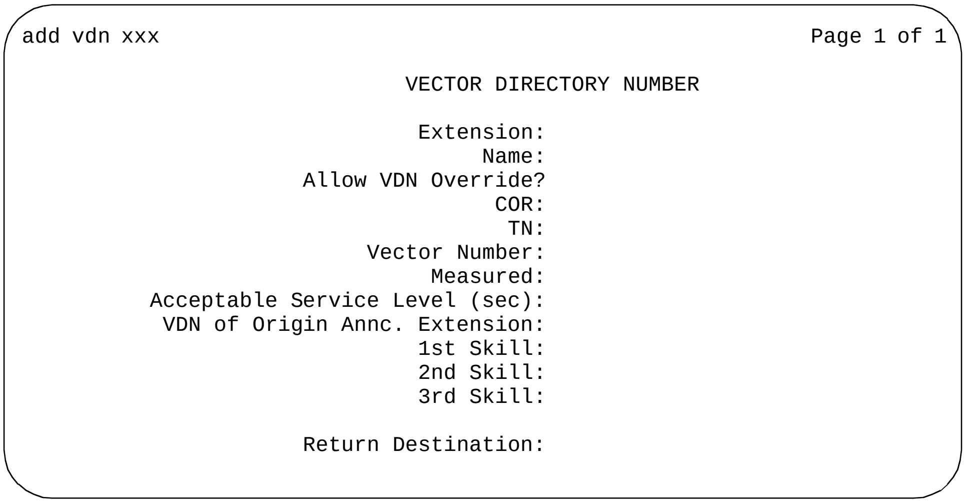
Figure 10-2. Example of VDN Skill Implementation

Let’s assume that the English-speaking caller needs information on Route Planning and dials the appropriate number (555-3333). In such a case, the call enters the switch and is directed to VDN 3333, which points to the appropriate vector. As illustrated earlier, VDN skill Preferences 33, 44, and 99 are administered as the 1st, 2nd, and 3rd skill preferences, respectively, for VDN 3333.

We will hold off explaining the vector processing of our application until the "Delivering the Call to the Skill Queue" section.

Vector Directory Number (VDN) Form

The Vector Directory Number (VDN) form is used to administer VDN skills. The form appears as follows:



```
add vdn xxx                                                         Page 1 of 1
                              VECTOR DIRECTORY NUMBER

                               Extension:
                                    Name:
                      Allow VDN Override?
                                     COR:
                                      TN:
                           Vector Number:
                                Measured:
          Acceptable Service Level (sec):
           VDN of Origin Annc. Extension:
                               1st Skill:
                               2nd Skill:
                               3rd Skill:

                      Return Destination:
```

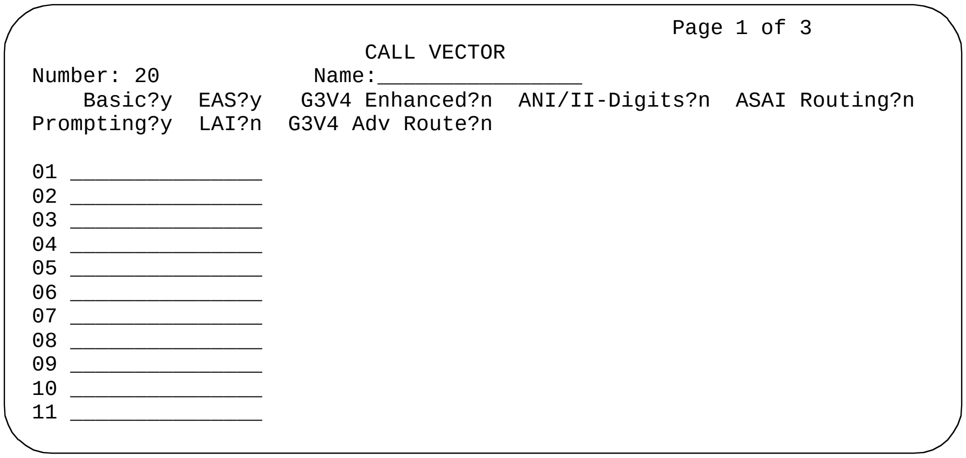
Figure 10-3. Vector Directory Number (VDN) Form

NOTE:
While skills can be optionally assigned to VDNs, the vector controls when and to what VDN skill the call queues.

Complete instructions for completing the form are included in *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

Call Vector Form

Completion of the Call Vector form is required for using vectors with EAS. The form contains three pages. However, if your vector contains 11 or fewer instructions, you need populate only the first page of the form. Page 1 of the Call Vector Form appears as follows:



```
                                                     Page 1 of 3
                                CALL VECTOR
Number: 20              Name:________________
    Basic?y  EAS?y   G3V4 Enhanced?n  ANI/II-Digits?n  ASAI Routing?n
Prompting?y  LAI?n  G3V4 Adv Route?n

01 _______________
02 _______________
03 _______________
04 _______________
05 _______________
06 _______________
07 _______________
08 _______________
09 _______________
10 _______________
11 _______________
```

Figure 10-4. Call Vector Form (Page 1 of 3)

NOTE:
While skills can be optionally assigned to VDNs, the vector controls when and to what VDN skill the call queues.

Complete instructions for completing the Call Vector form are provided in *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

Agent Skills

Agents are trained or hired to accommodate specific caller needs. Agent skills represent and define the ability of the agent to handle calls that require these skills. Agents are assigned skill numbers based on such characteristics as training or knowledge, access to systems or information, language ability, and interpersonal traits. Examples of agent skills include the following: speaks Spanish, knows about widget ‘‘X,’’ can handle complaint calls, has access to a database, etc.

A maximum of four agent skills can be assigned to any one agent. If an agent has multiple skills, a single skill group can be created for each set of skills. Agent skills are assigned to agents by completing the Agent Login ID form (see the "ACD Login ID Dialing" section later in this chapter).

It is recommended, though not mandatory, that you create a separate skill hunt group for direct agent calls. Direct agent calls are queued to the first primary skill hunt group that the agent is logged in to. If you wish to use a separate skill as a direct agent skill, you must ensure that this skill is the first skill administered for all agents who receive direct agent calls.

The following table illustrates the assignment of agent skills.

NOTE:
Refer to Table 10-4 for a description of the skills indicated in the following table.

Table 10-7. Example of Agent Skill Assignments

Agent	Skills Assigned			
Jan O'Hara	22 P	44 S		
Sam Thomas	99 P			
Sue Carlson	22 P	11 P	44 S	33 S
Mark Davis	44 P			
Amy Brown	44 P	22 S		

Table 10-7 indicates the skills assigned to five specific agents. Remember, a maximum of four agent skills may be assigned to any one agent. From the figure, we see that four agent skills (22, 11, 44, 33) are assigned to Sue Carlson. These assignments indicate that Sue is bilingual and can service callers who need emergency road service or information on route planning. On the other hand, we see that only one agent skill (99—Supergroup) is assigned to Sam Thomas. This means that Sam is serving only as a backup.

In the figure, a ''P'' or "S'' next to the skill number indicates whether the agent skill is a primary skill or a secondary skill. For example, Jan O'Hara has ''Emergency Road Service-Bilingual'' as a primary skill and "Route Planning-Bilingual'' as a secondary skill. This means that whenever Jan O'Hara becomes available for an ACD call, the ACD software first looks for English-speaking callers requesting information on ''Emergency Road Service'' from the agent. Only if there are no callers requesting "Emergency Road Service'' does the ACD software look for English-speaking callers requesting information on ''Route Planning.''

For any given application, EAS puts no restrictions on which agent skills can be assigned to an agent.

NOTE:
Agent skills are administered by completing the Agent Login ID form. This form is presented in the "ACD Login ID Dialing" section later in this chapter. Complete instructions for completing the form are provided in *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

Delivering the Call to the Skill Queue

Now that we have defined and illustrated VDN skills and agent skills, we are ready to discuss (with the help of an example) how a call is delivered to a skill hunt group queue via vector processing.

The skills assigned to a VDN define the requirements in the vector for routing calls to an ACD agent with a particular set of skills. These skills become active for an ACD call whenever a *queue-to main skill* command is executed. The skills also become active whenever a *check-backup skill* command is executed and the threshold condition is met. Once a skill is active for an ACD caller, the call cannot be delivered to an available ACD agent unless the agent also has one of the active VDN skills. Take a look at the following figure.

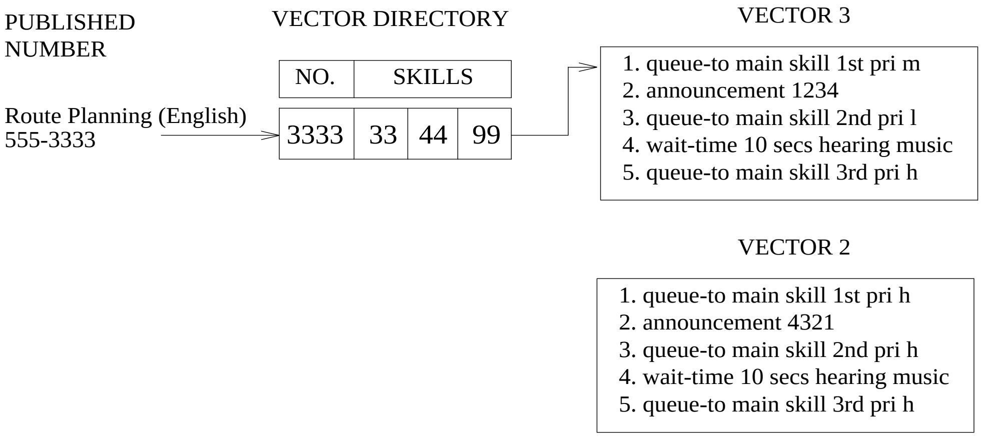

PUBLISHED NUMBER
VECTOR DIRECTORY
VECTOR 3
NO.
SKILLS
Route Planning (English)
555-3333
3333
33
44
99
1. queue-to main skill 1st pri m
2. announcement 1234
3. queue-to main skill 2nd pri l
4. wait-time 10 secs hearing music
5. queue-to main skill 3rd pri h
VECTOR 2
1. queue-to main skill 1st pri h
2. announcement 4321
3. queue-to main skill 2nd pri h
4. wait-time 10 secs hearing music
5. queue-to main skill 3rd pri h

Figure 10-5. Process for Delivery of a Call to a Skill Queue

Let's assume that an English-speaking caller needs information on "Route Planning'' and dials the appropriate number (555-3333). In such a case, the call enters the switch and is directed to VDN 3333, which points to Vector 3. Once vector processing starts, the *queue-to main skill* command in Step 1 queues the call to the skill hunt group corresponding to the 1st VDN skill (33-Route Planning-English). If an agent with skill 33 is available, this agent answers the call. If such an agent is not available, the call is eventually queued to the skill hunt group corresponding to the 2nd VDN skill (44-Route Planning-Bilingual) by the *queue-to main skill* command in Step 3. This time, if an agent with skill 44 is available, this agent answers the call. If the call is still not answered, the call is eventually queued to the skill hunt group corresponding to the 3rd VDN skill (99-Supergroup) by the *queue-to main skill* command in Step 5.

Note that the figure also shows Vector 2. This vector would have been executed if a Spanish-speaking caller had called into the switch. Accordingly, the announcement provided in Vector 2 is in Spanish, whereas the announcement in Vector 3, which is executed in our example, is in English.

Note also that each of the *queue-to main skill* commands in Vector 2 queues the call at a *high* priority, whereas only one of the *queue-to main skill* commands in Vector 3 queues the call at this high a priority level. The strategy presented here is very valuable in cases where there is a limited number of bilingual agents in that such agents would be available to service callers who speak only Spanish more quickly.

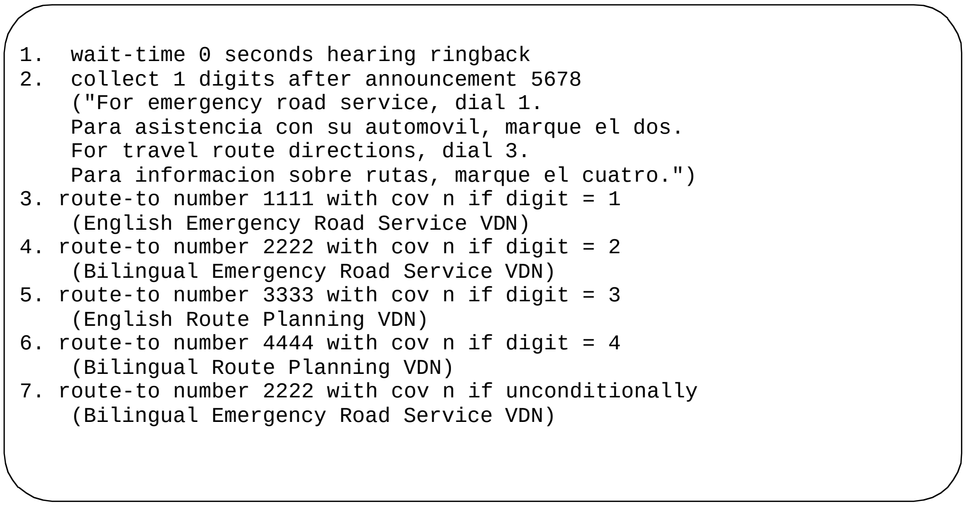
VDN skills can also be used in *check-backup skill, messaging skill,* and *converse-on skill* commands. Within any of these commands, a specific skill number can be used instead of a VDN skill Preference, provided the relevant skill hunt group is correctly administered. For example, Step 5 might have read ‘‘queue-to main skill 99 pri h.’’ We will discuss this concept further in the "Super Agent Pool" section.

Procedure Using Call Prompting

The procedure described in the previous section can be enhanced by using Call Prompting. For example, the user can dial a ‘‘general’’ telephone number whose VDN points to a Call Prompting vector.

Staying with our auto club example, recall that in Table 10-2 we define ‘‘555-5555’’ as the ‘‘general’’ telephone number for the service. Recall also that in Table 10-5 we identify ‘‘5555’’ as the VDN for this telephone number. Also, we indicate that VDN 5555 points to Vector 1.

Here’s an example of how Vector 1 might appear:



```
1.  wait-time 0 seconds hearing ringback
2.  collect 1 digits after announcement 5678
    ("For emergency road service, dial 1.
    Para asistencia con su automovil, marque el dos.
    For travel route directions, dial 3.
    Para informacion sobre rutas, marque el cuatro.")
3. route-to number 1111 with cov n if digit = 1
    (English Emergency Road Service VDN)
4. route-to number 2222 with cov n if digit = 2
    (Bilingual Emergency Road Service VDN)
5. route-to number 3333 with cov n if digit = 3
    (English Route Planning VDN)
6. route-to number 4444 with cov n if digit = 4
    (Bilingual Route Planning VDN)
7. route-to number 2222 with cov n if unconditionally
    (Bilingual Emergency Road Service VDN)
```

Figure 10-6. Call Prompting Vector for the Auto Club

Once the caller dials ‘‘555-5555,’’ the call enters the switch and is directed to VDN 5555, which points to our Call Prompting vector. At this point, vector processing begins. Step 1 provides ringback if the caller has to queue for the announcement in Step 2. The *collect digits* command in Step 2 first provides an announcement requesting the caller to dial ‘‘1,’’ ‘‘2,’’ ‘‘3,’’ or ‘‘4,’’ depending upon the caller need and the caller’s language speaking ability. If the caller dials a digit other than one of the four specified, each of the *route-to...if digits* commands in Steps 3 through 6 fails, and control is passed to the *route-to...if unconditionally* command in Step 7, which unconditionally routes the call to VDN 2222. This VDN is assigned the ‘‘Bilingual Emergency Road Service’’ skill and points to Vector 2, which is provided in the previous section.

On the other hand, returning to the *collect digits* step, let’s assume that the caller dials ‘‘4.’’ In such a case, Steps 3 through 5 fail because the required digit (‘‘1,’’ ‘‘2,’’ or ‘‘3,’’ respectively) has not been dialed. Thereafter, control is passed to Step 6, where the *route to...if digit* command finds a digit match and consequently routes the call to VDN 4444. This VDN is assigned the ‘‘Bilingual Route Planning’’ skill and also points to Vector 2, which is provided in the previous section.

NOTE:
VDN Override applies to the skills assigned to the VDN. See "VDN Override" on page 3-9 for more information.

Super Agent Pool

EAS allows a skill hunt group to function as a super agent pool. A super agent pool is a ‘‘backup’’ group of one or more agents that is able to handle many if not all types of calls coming into the application. In our auto club examples, Skill Hunt Group 99 (Supergroup) serves as a super agent pool. Also, you might recall that ‘‘99’’ appears as both a VDN skill and an Agent skill. However, a super agent pool can be assigned a skill hunt group number that is not assigned to a VDN skill. This can (and should) be done whenever the application requires four levels within the skill table distribution.

To illustrate this, let’s first modify and expand on the Auto Club Skill Table (Table 10-4) presented earlier in this chapter:

Table 10-8. Modified Skill Table for the Auto Club

Supergroup-99			
Emergency Road Service-Bilingual-88		Route Planning-Bilingual-77	
English-66	Spanish-55	English-44	Spanish-33
Bostonian-11	Castilian-13	Bostonian-15	Castilian-17
New Yorker-12	South American-14	New Yorker-16	South American-18

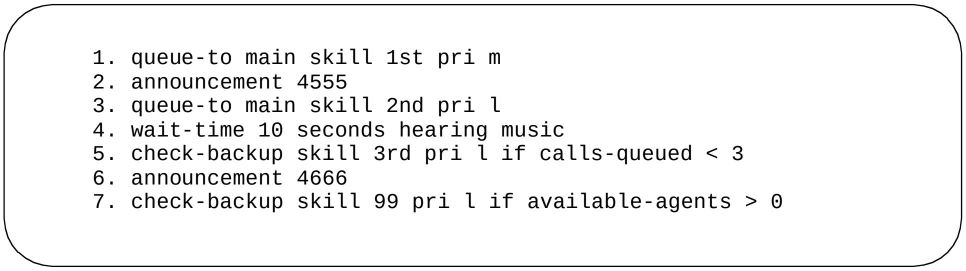
Besides a new skill numbering scheme, our modified skill table has four levels instead of the three levels provided in Table 10-4. Except for the skill numbering scheme, the top two levels (Supergroup-99 and Emergency Road Service-Bilingual-88/Route Planning-Bilingual-77) remain unchanged. However, note that the next level has been reorganized into segments to indicate the ability to speak English or Spanish. Finally, note that a new level has been added to denote particular types of accents or pronunciation in English and Spanish.

Now, let’s take a look at the following table, which illustrates how some of the skills in the preceding table are administered to one relevant VDN (1616):

Table 10-9. Skill Preferences Assignments for VDN 1616

VDN 1616 - Skill Preferences		
1st:	16	Knows about Route Planning, speaks English, has New Yorker accent
2nd:	44	Knows about Route Planning, speaks English
3rd:	77	Knows about Route Planning, is bilingual

Now we are ready to consider the following vector, which is a variation of one presented earlier in this chapter:



```
1. queue-to main skill 1st pri m
2. announcement 4555
3. queue-to main skill 2nd pri l
4. wait-time 10 seconds hearing music
5. check-backup skill 3rd pri l if calls-queued < 3
6. announcement 4666
7. check-backup skill 99 pri l if available-agents > 0
```

Figure 10-7. Modified Vector to Accommodate a Super Agent Pool

Let's assume an English-speaking caller needs information on ''Route Planning'' and would like to speak to an agent with a New Yorker accent. In such a case, the caller dials the appropriate number (555-1616, for example). Accordingly, the call enters the switch and is directed to VDN 1616, which points to the vector in the previous screen. Once vector processing starts, the *queue-to main skill* command in Step 1 queues the call to the skill group corresponding to the 1st VDN skill (New Yorker-16). If an agent with skill 16 is available, this agent answers the call. If such an agent is not available, the call is eventually queued to the skill group corresponding to the 2nd VDN skill (English-44) by the *queue to main skill* command in Step 3. This time, if an agent with skill 44 is available, this agent answers the call. If the call is still not answered, the *check-backup skill* command in Step 5 attempts to queue the call according to the parameter indicated (if calls-queued < 3) to the skill group corresponding to the 3rd VDN skill (Route Planning-Bilingual-77). If the call is queued, and if an agent with skill 77 is available, this agent answers the call. If the call is not queued, or if it is queued and an agent with skill 77 is not available, the *check-backup skill* command in Step 7 is executed.

Before we discuss the execution of Step 7, note that a specific skill hunt group number (99) and not a VDN skill Preference designation (1st, 2nd, or 3rd) is included within the *check-backup skill* command. Since the skill table for the application involves four levels of skills, and since there can be no more than three VDN skills, the specific skill group number (99) for the super agent pool must be included within the queuing command to allow caller access to the pool. Whereas a VDN skill is always represented in a vector by the term ''1st,'' ''2nd,'' or ''3rd,'' a super agent pool is always represented by a whole number according to the parameters of the relevant switch (see the manual pages for the queuing commands in Appendix A).

Returning to the vector execution, the *check-backup skill* command in Step 7 attempts to queue the call according to the parameter indicated (if available-agents > 0) to the super agent pool (Supergroup-99). If the call is queued, and if an agent in the super agent pool is available, this agent answers the call.

NOTE:
If the call has already queued to all three VDN skill hunt group preferences, it will not queue to the specific skill hunt group. This reflects the restriction that a call can only queue to a maximum of three splits or skills. The best approach is to test the splits/skills first to determine where to queue the call. Also, see "Expected Wait Time (EWT)" on page 6-2.

Routing the Call to an Agent

With EAS optioned, an agent becomes associated at login with one or more skill hunt groups. A single set of work mode buttons applies to all the skills assigned to a logged-in agent (for example, if the agent selects "Aux Work," the agent is in Aux Work for all the skills associated with the agent). Therefore, logged-in agents need only a single set of work-mode buttons for all relevant skill hunt groups.

Calls may be routed to the agent from a skill hunt group, by dialing an Agent Login ID, or by dialing an agent voice terminal extension directly. The following sections discuss these procedures.

Delivery from a Skill Hunt Group

An incoming call is matched to an agent who has at least one of the three skills required to handle the call. This matching is done via ACD queuing and via *queue-to main skill, check-backup skill, messaging skill,* or *converse-on skill* commands in the vector. If more than one agent is available for a call, the call is delivered according to whether EAD or UCD is administered for the skill hunt group.

For any one login session, an agent can have a maximum of four skills. Each agent skill is administered as either a primary skill or a secondary skill. The ACD software distributes any call that is waiting for one of the agent's primary skills whenever the agent becomes available. If no calls are waiting for a primary skill, the queued calls for the secondary skills are distributed to the agent.

Once an agent becomes available, he or she receives a waiting call in the following order:

1. Oldest Direct Agent call waiting for the agent
2. Oldest call waiting at the highest priority for all primary skills
3. Oldest call waiting at the highest priority for all secondary skills

For example, let's assume that Jill is the only agent with skills 22P, 13P, and 47S. Let's also assume that, while Jill is in AUX work mode, five calls are queued, as illustrated in the following table (which also provides the skill and priority level associated with each call):

Table 10-10. Example of Skill Call Queue Sequence

Call	Time Queued	Skill No.	Priority Level
A	8:00	13	Medium
B	8:01	47	Top
C	8:02	22	Direct Agent
D	8:03	22	Top
E	8:04	22	Medium

Given this scenario, the following table indicates and explains the order in which Jill would handle the five calls:

Table 10-11. Example of Skill Call Distribution for a Single Agent

Call Handled	Reason
C	Only Direct Agent call queued.
D	Oldest call waiting at the highest priority for all primary skills (Call B has the same priority level (Top), but it is assigned a secondary skill (47) and not a primary skill. Also, Call E has the same skill (22), but it has a lower priority level (Medium) and has not been waiting as long as Call D).
A	Oldest call waiting at the highest priority level for all primary skills (Call E also has a primary skill (22) and the same priority level as Call A, but Call A has been waiting four minutes longer than Call E).
E	Only remaining call with a primary skill (22) (Call B has a secondary skill (47)).
B	Last remaining call, and the only one that has a secondary skill (47).

If no calls are waiting when an agent becomes available, the agent is placed into the agent queue according to the call distribution method in effect. For UCD, the agent is placed at the bottom of the most-idle agent queue. For EAD, the agent is placed at the bottom of either the primary agents in queue or secondary agents in queue, depending on the agent’s skill type.

The following table illustrates a call scenario that is valid for either UCD or EAD:

Table 10-12. Example of UCD/EAD Call Scenario

Time	Event	Skills
9:00	Jill logs in	22P, 13P, 47S
9:01	Jill available	22P, 13P, 47S
9:02	Jack logs in	22P, 47P
9:03	Jack available	22P, 47P
9:04	Call A arrives	47
9:05	Call A drops	47
9:06	Call B arrives	13
9:07	Call B drops	13
9:08	Call C arrives	22

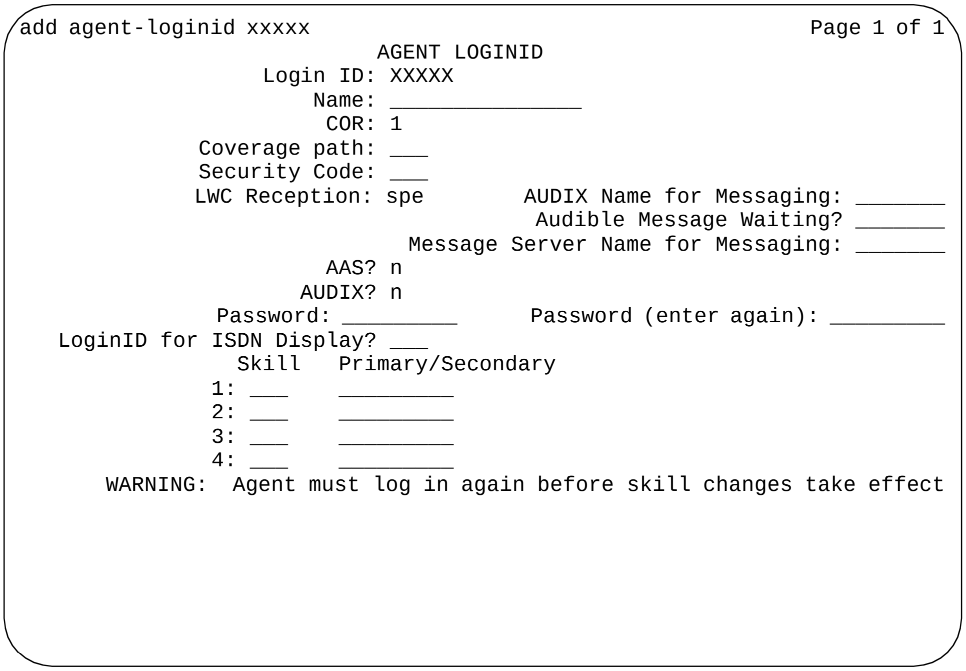
Given this scenario, the following table illustrates how Calls A, B, and C are distributed via UCD and EAD:

Table 10-13. Example of Call Distribution via UCD/EAD

Time	UCD or EAD?	Result	Reason
9:04	UCD	Jill receives Call A.	Jill is the most idle agent for skill 47.
	EAD	Jack receives Call A.	Jack is the ‘‘more expert’’ agent because he has skill 47 as a primary skill whereas Jill has skill 47 as a secondary skill.
9:06	UCD	Jill receives Call B.	Jill is the only agent logged into skill 13.
	EAD	Jill receives Call B.	Jill is the only agent with Primary skill 13.
9:08	UCD	Jill receives Call C.	Jill is the most idle agent for skill 22. She receives Call C even if she handled Call A.
	EAD	Jill receives Call C.	Both Jill and Jack have Primary skill 22, but Jill has been logged in two minutes longer than Jack (that is, she is the most idle agent).

ACD Login ID Dialing

The ACD Login IDs used in EAS are extension numbers included in a station numbering plan but not administered as stations. These IDs are administered by using the Agent Login ID form, which appears as follows:



```
add agent-loginid xxxxx                                              Page 1 of 1
                                AGENT LOGINID
                    Login ID: XXXXX
                        Name: _______________
                         COR: 1
               Coverage path: ___
               Security Code: ___
               LWC Reception: spe       AUDIX Name for Messaging: _______
                                         Audible Message Waiting? _______
                               Message Server Name for Messaging: _______
                        AAS? n
                      AUDIX? n
                 Password: _________      Password (enter again): _________
   LoginID for ISDN Display? ___
                   Skill   Primary/Secondary
                1: ___    _________
                2: ___    _________
                3: ___    _________
                4: ___    _________
       WARNING:  Agent must log in again before skill changes take effect
```

Figure 10-8. Agent Login ID Form

NOTE:
The AUDIX Name fields on the screen are display-only fields. These are only G3r features.

With EAS, an Agent's ACD Login ID is associated with a specific voice terminal only when the agent actually logs in at that terminal. When the agent logs off, the association of the agent's ACD Login ID with a specific voice terminal is removed. This capability is known as *Logical Agent*. If an agent does not answer a call, or if the agent is logged out, the call goes to the busy points on the coverage path.

When the agent logs in, the voice terminal display indicates the agent's skill assignments.

The agent logs in by doing the following:

- Going off-hook or selecting a line appearance

- Upon hearing the dial tone, entering the login Feature Access Code (FAC) or selecting the Login Abbreviated Dialing (AD) button
- Upon hearing the dial tone, entering the 1- to 5-digit Login ID

  NOTE:
  If someone is already logged in at that terminal, the agent will hear intercept.

- Upon hearing the dial tone, entering (optionally) the 0- to 9-digit password.

  NOTE:
  If the agent is using a DCP voice terminal such as CallMaster, the password digits will not be shown unless an abbreviated dial button is used. BRI voice terminals will show the password digits.

Once the login is accepted, confirmation tone is given. Also, the skills assigned, along with the corresponding ‘‘primary’’ or ‘‘secondary’’ indication, are displayed for five seconds on the voice terminal display. The previous login sequence allows an ACD call to be directed to a specific agent and to have that call tracked and treated as an ACD call.

When an EAS agent logs into a station with the station administered for audible message waiting, the agent will receive Audible Message Waiting tone only when calls are waiting for the agent login ID extension. When the agent logs out, Audible Message Waiting tone then applies again to messages waiting for the physical extension. This field does not impact whether an agent hears the EAS Login-ID Message Waiting tone during the login process.

With G3V4 and later releases, the Message Waiting Lamp by default tracks the status of messages waiting for the logged in EAS agent LoginID rather than messages for the physical terminal. The operation of the Message Waiting Lamp can be changed so that it tracks the status of messages waiting for the physical terminal where the agent is logged in. See the Feature-Related System-Parameters form in the *DEFINITY Communications System Generic 3 V4 Implementation manual*, 555-230-653 for more information.

In addition to skill assignments, the following capabilities are now associated with agents’ login IDs.

- **Call Routing**

  A call to the Login ID reaches the agent independent of the voice terminal the agent is currently using. In other words, such a call is sent to the voice terminal at which the agent is currently logged in.

  If the proper Class of Restrictions (COR) is set, callers can initiate a Direct Agent call either by dialing the Login ID extension directly or by calling a VDN that points to a vector containing first a prompt for the Login ID and then a *route-to digits* command. This allows external callbacks via Direct

Inward Dialing (DID) or an 800 number. Both the receiving agent's Login ID COR and the originator's (caller's) COR must have ‘‘Direct Agent Calling’’ set to ‘‘y.’’ The caller's COR is for the following:

— Voice terminal extension (for internal calls or transfers)

— Trunk group (for DID calls)

— VDN (for prompted calls)

- If the call covers or is forwarded, the COR of the originator (or VDN) and the final agent is used. All feature functionality for ACD calls, except Queue Status indications, is available for Direct Agent Calls.

  Internal and external users can originate Direct Agent calls by dialing the agent's login ID. Also, Direct Agent calling can be used to transfer ACD calls from one agent to another agent.

  If an agent receiving the Direct Agent Call is staffed but unavailable, the call waits in front of the ‘‘skill’’ calls in the agent's first primary skill until either the call is answered or a coverage timeout occurs. Also, the caller hears an optional direct-agent announcement followed by music or silence. There is one Direct Agent announcement per system. The agent, on the other hand, receives a ring-ping, and the current work mode button flashes. If the agent is available, the call is delivered to the agent according to the answering and alerting options. Calls are answered and handled in the same manner as ACD calls.

- **Login ID Name on the Voice Terminal Display**

  A call to a logged-in EAS LoginID by default displays the name associated with the LoginID and not the name associated with the voice terminal. This is also true on the receiving party's display for a call made from a voice terminal with an agent logged in. However, with G3V4 and later releases, the user can display the name of the physical terminal where the EAS agent is logged in. The user must be active on a call with the agent, and must have a terminal with an alphanumeric display and an inspect button. When the inspect button is pressed during a call to or from the EAS agent, the physical terminal name of the agent is displayed.

- **Coverage Path**

  Call coverage can occur whether or not the agent is logged in. If the agent is not logged in, the busy criteria is met and the call follows the points on the coverage path. If the agent is logged in but fails to answer, the don't answer criteria is met and the call follows the points on the coverage path. A call to the Login ID goes to the coverage path assigned to the Login ID rather than to the coverage path assigned to the voice terminal extension.

- **Agent Restrictions**

  A call to the Login ID or from the agent uses the restrictions associated with the agent and not the voice terminal.

Voice terminals are fully functional if an agent is not logged in. The restrictions, coverage, and name revert to the voice terminal administration when the agent logs out.

If a number of users are sharing one voice terminal (due to job sharing or shifts, for example), a unique Login ID extension is assigned to each user. Therefore, whenever a user is logged out, any calls to that user (login ID) are sent to his or her coverage path. As a result, Login IDs can be used to reach people independent of where they happen to be. Such people include those who use more than one phone because they have more than one office or (in the case of security guards, for example) sit at more than one desk.

Because AAS/AUDIX ports are not mobile, these ports are administered to Agent Login IDs. Whenever the AAS or AUDIX field is set to ‘‘y,’’ a field requesting the port number is brought up, and the password field disappears.

Interactions Involving EAS

This section discusses the feature and adjunct interactions involving EAS.

Feature Interactions

This section discusses the feature interactions involving EAS. Unless otherwise specified, the feature interactions for skill hunt groups are the same as for vector-controlled splits.

Abbreviated Dialing

Abbreviated dialing can be used to log in or log out EAS agents. Abbreviated dialing lists or buttons can be administered only for stations.

Administration Without Hardware

Although EAS Login IDs are extensions without hardware, they are not a part of the Administration without Hardware feature.

Agents in Multiple Splits Feature

With EAS, the ‘‘Agents in Multiple Splits’’ feature is called ‘‘Agents in Multiple Skills.’’ This feature allows an EAS agent to be logged into up to four skills.

Agent Work Modes

With EAS optioned, an agent can be in only a single work mode for all skills at any one time (for example, an agent cannot be in AUX work mode in one skill hunt group and also available in another skill hunt group). Also, if the After Call Work (ACW) mode button is selected, the agent is placed into ACW for the first skill administered and logged into.

Assist

This feature is used for skill hunt groups (that is, there is one supervisor per skill hunt group). A voice terminal can be administered with one or more Assist buttons for each skill that agents using the terminal might have. An Assist button can also be administered with no associated skill. In this case, the supervisor for the skill that the agent is currently active on is called. If the agent is not active on any skill, the supervisor for the agent's first skill is called.

Any assist button selected is tracked as an assist for the current call, regardless of the skill assigned (if any) to the button. The administered association of an Assist button with a particular skill and assigned supervisor is not affected when an EAS agent logs into that station.

Audible Message Waiting

If messages are waiting for an EAS agent login-ID extension, an agent will hear a special 5-burst EAS Login-ID Message Waiting tone (instead of confirmation tone) after successfully logging in. This does not require Audible Message Waiting to be assigned to the voice terminal or the system.

If Audible Message Waiting is optioned for the system and assigned to an agent's voice terminal, and messages are waiting for the agent login ID extension, the agent will hear the Audible Message Waiting tone whenever the agent goes off-hook, or selects a line appearance and hears dial tone. Messages waiting for the physical extension will not cause an Audible Message Waiting tone when an EAS agent is logged in.

Auto-Available Skills

If a skill hunt group is administered as an Auto-Available Skill (AAS), the EAS Login IDs assigned to this skill must also be administered as Auto-Available. When the switch reinitializes, these Login IDs are automatically logged in with the AUTO-IN work mode. If any switch features attempt to change the work mode to anything except AUTO-IN, this attempt is denied. Agents cannot have both Auto-Available and Non-Auto-Available Skills. This feature is not intended for human agents.

Automatic Answering with Zip Tone

This feature can be administered only for a physical extension. The feature is not associated with a LoginID.

Basic CMS

The Basic CMS user interface remains the same when EAS is optioned. The only change is that the labeling of the headings is changed from ‘‘split’’ to ‘‘skill.’’ With releases prior to G3V3, BCMS agent reports are based on the physical extension rather than on Login IDs. Accordingly, agents are tracked by physical extensions. With G3V3 and later releases, it is possible to administer BCMS/VuStats Login IDs. Agents can then be tracked based on these IDs.

BCMS tracks Direct Agent calls as skill calls. Direct Agent calls affect ACD talk time, ACW time, and Average Speed of Answer. Whenever Direct Agent calls are waiting, BCMS displays an asterisk immediately after the CALLS WAITING column.

Bridging

ACD calls do not alert on bridged appearances. However, bridged users can activate features on behalf of agents. Features that can be activated include: log in, log out, change work modes, and assist.

Call Coverage

Call coverage can occur whether or not the agent is logged in. If the agent is not logged in, the busy criteria is met and the call follows the points on the coverage path. If the agent is logged in but fails to answer, the don’t answer criteria is met and the call follows the points on the coverage path. A call to the Login ID goes to the coverage path assigned to the Login ID rather than to the coverage path assigned to the voice terminal extension.

Call Forwarding

Skill hunt groups (since they are vector-controlled) cannot be call forwarded. EAS agent Login IDs cannot be forwarded, but the physical extension where the EAS agent is logged in can be forwarded. If another station (with console permissions) tries to forward an EAS Login ID, an intercept is given.

Call Park

To retrieve a parked call via a Feature Access Code (FAC), the agent dials the Answer-Back FAC and the extension where the call is parked. If the person unparking the call dials the Answer-Back FAC and the physical extension of the station where the call is parked, they are connected to the parked call.

In some cases, the person un-parking the call may also be able to dial the Answer-Back FAC and the logical agent extension of the agent who parked the call. This operation is possible if the Class of Restriction (COR) of both the agent parking the call and the terminal or agent un-parking the call have a COR with the Direct Agent Calling flag set to “y.” If the terminal un-parking the call is not a logged in agent, then the terminal must have a COR with Direct Agent Calling set to “y.” If the station un-parking the call is a logged in agent, then the COR of the logical agent extension must have Direct Agent Calling set to “y.”

Call Pickup

Skill hunt group extensions and EAS Login ID extensions cannot be members of a Call Pickup group.

Class of Restriction

Skill hunt groups do have a Class of Restriction (COR). The COR is used if the skill hunt group extension is called directly. The COR for an EAS agent Login ID overrides the physical extension’s COR of the voice terminal that an agent logged into.

Class of Service

EAS agents do not have a COS associated with their Login ID.   Instead, the COS is associated with the physical extension. Therefore, the COS of the voice terminal is not affected when an EAS agent logs into that terminal.

Dial Plan

Agent Login IDs are part of the dial plan, and they reduce the total number of stations.

Direct Agent Calling

If a called EAS Agent Login ID and the call originator (extension, trunk, or VDN) both have a COR that allows Direct Agent calls, the call to the Login ID is treated as a Direct Agent call. A call to the voice terminal extension where an EAS agent is logged in, or a call to an EAS Agent LoginID where either the originator’s or the LoginID’s COR does not allow Direct Agent calls, is treated as a personal (non-ACD) call.

Displays—Voice Terminal

When an EAS agent is logged in, the display for originators who call the Login ID shows the Login ID and agent name (as administered via the Agent Login ID form). Calls originated by the agent show the Agent Login ID and agent name at the receiving terminal’s display. However, with G3V4 and later releases, the user can display the name of the physical terminal where the EAS agent is logged in. The user must be active on a call with the agent, and must have a terminal with an alphanumeric display and an inspect button. When the inspect button is pressed during a call to or from the EAS agent, the physical terminal name of the agent is displayed.

Calls to the physical extension show the physical extension’s number and name on the originator’s display.

Look-Ahead Interflow

Skills are not sent to another ACD/PBX when a call interflows using Look-Ahead Interflow. If skills have the same meaning on both ACDs, a Look-Ahead Interflow command to a VDN with the same skills assigned can provide a mapping of the skills.

Multiple Split Queuing

When EAS is enabled, the Multiple Split Queuing feature is called Multiple Skill Queuing, which has the same functionality. With Multiple Split/Skill Queuing a call can only queue to a maximum of 3 splits/skills.

OCM/EAS

If EAS is enabled on the DEFINITY switch, The Outbound Call Management (OCM)/Expert Agent Selection (EAS) feature is required for a CallVisor ASAI adjunct application to launch predictive Outbound Call Management (OCM) calls. Predictive Calling is an OCM feature that is often used in applications, such as sales or ‘‘cold-calling,’’ where it does not matter which agent is accessed by a caller and for which it is important to keep the agents utilized fully.

While OCM predictive calling is an outbound call management application, the EAS environment provides a number of desirable features for inbound call handling. The OCM/EAS feature allows the customer to enable both types of call handling on the switch. From a technical standpoint, if EAS is enabled, the feature is needed for the following reasons:

- All skill hunt groups are vector-controlled. (However, to launch a predictive OCM call in a traditional ACD environment, the ACD split cannot be vector-controlled.)
- The traditional ACD environment and EAS cannot be enabled on the switch at the same time.

The OCM/EAS feature extends the ASAI features to include launching predictive OCM calls from a VDN extension. Previously, ASAI hosts could launch predictive calls only from ACD split extensions. A limited number of Call Vectoring commands are supported in the VDNs used to launch or process OCM predictive calls. These commands are listed in the following section.

Commands for OCM Predictive Calls

The Call Vectoring commands indicated in the following table are available for processing OCM Predictive Calls. The customer must design vectors intended for processing predictive calls in such a manner that the vectors are limited to the supported steps.

The following table lists the supported commands and provides a brief comment for each command.

Table 10-14. Commands for OCM Predictive Calls

Command	Comment
queue-to main skill (single occurrence)	This command queues the call for handling by an agent in the skill pool.
announcement	This command plays an announcement if there are no agents available and if the queue (if any) is full.
stop	This command ends vector processing. The command also disconnects any call that is not queued.
adjunct routing	EAS supports adjunct routing to any of the following: skill extension, direct agent call, announcement, or local extension. The command does not involve routing to an external number.
wait-time	**NOTE:** This command is used with the *adjunct routing* step to determine how long the switch will wait for an adjunct route before continuing with vector processing.

Queue Status Indications

Physical extensions can be administered with Queue Status Indicator buttons and lamps for skill hunt groups that operate in the same manner as split Queue Status Indicators for traditional ACD splits. As long as enough buttons are available, Queue Status Indicators can be administered for all skills needed by agents using that physical extension. Also, any waiting Direct Agent calls are not reflected in the queue status indicators.

Service Observing

This feature is activated in the EAS environment by dialing the physical extension of the terminal where an EAS agent is logged in. With G3V3 and later releases it also can be activated by dialing the EAS agent loginID.

Remote Service Observing

Remote access to the Service Observing (SO) FACs can be provided via the Remote Access feature or through Service Observing vectors. See "Service Observing" on page 4-16 and "Creating Service Observing Vectors" on page 5-13 for additional information.

SMDR/CDR

For skill calls, the ‘‘called party’’ field can optionally be the Agent Login ID.

VDN Override

If VDN Override is set on the ‘‘previous’’ VDN, the VDN skills of the ‘‘current’’ VDN are used. If VDN Override is not set on the ‘‘previous’’ VDN, the VDN skills of the ‘‘previous’’ VDN are used.

Work Mode Buttons

Only a single set of agent work mode buttons is needed. If multiple buttons are assigned, all lamps for that work mode (for example, manual-in) light whenever any one button is pushed.

Adjunct Interactions

This sections discusses the adjunct interactions involving EAS.

ASAI

ASAI support for EAS may be organized into the following categories: call control, feature requests, value queries, event notification, and adjunct-controlled skills. This section provides a high-level overview of the behavior of ASAI in the EAS environment.

Call Control

Call control capabilities work exactly the same in the EAS environment as in the traditional ACD environment except for the following:

- User-classified third party make calls (calls classified by originator) may originate from an EAS Login ID and terminate to a Login ID. User-classified calls terminating to a Login ID are given the same Direct Agent treatment provided for such calls that are dialed from a station extension.
- Switch-classified third party make calls (which are classified by a call classifier board and delivered (when answered) to the originating hunt group) may originate from or terminate to EAS Login IDs.

- Direct Agent third-party make calls (ACD calls terminated to a selected member of an ACD skill group) may be requested by including a Direct Agent option, an agent's physical extension and a skill group extension (compatibility mode), or by requesting a user-classified third-party make call with a Login ID destination. The primary differences between the two methods of requesting Direct Agent calls are that the compatibility mode allows the adjunct to specify the skill hunt group to which a given Direct Agent call is queued and that the non-compatibility mode allows the adjunct to direct the call to a Login ID, regardless of which station an agent is logged into. Direct Agent third-party make calls may not originate from an EAS Login ID.
- Supervisor assist third party make calls (supervisor assist calls originated by a selected member of an ACD split) may originate from an EAS Login ID, and they may terminate to an EAS Login ID. Unlike dialed Direct Agent calls, supervisor assist calls terminated to a Login ID behave as though they have been previously directed to the requested Login ID's physical extension (for example, they do not cover if the requested agent is not logged in and if the originator's display shows the agent's physical extension and not the agent's Login ID).
- Extension (Domain) control may not be requested for an EAS Login ID, but it may be requested on behalf of a Logical Agent's physical extension. Auto-dial calls (calls initiated by an extension-controlled station) may be terminated to an EAS Login ID, in which case the call is given Direct Agent treatment.
- Adjunct routing calls (vector calls routed by an ASAI adjunct via the *adjunct routing* Call Vectoring command) are similar to third party make calls. Such calls may include a Direct Agent option, an ACD agent's physical extension, and a skill extension. If this is true, these calls are given compatibility mode Direct Agent treatment and may be terminated to an EAS Login ID (in which case they behave like dialed Direct Agent calls).
- If EAS is optioned, ASAI launches OCM switch-classified or predictive calls from a VDN extension via the OCM/EAS feature. On the other hand, to launch a predictive call in a traditional ACD environment, an adjunct OCM application sends to the switch an ASAI request with an ACD split number as the ‘‘originating number.’’ The application also sends flags identifying the call as a switch-classified call. In the traditional ACD environment, the ACD split cannot be vector-controlled.

Feature Requests

In the EAS environment, agent login, logout and change work-mode requests are fully supported. Agent login requests must contain an EAS Agent Login ID and optional password (delimited by ‘#’) in the login request's user code IE. Agent logout requests and change work-mode requests may contain the desired agent's physical extension or Login ID. Call Forwarding and Send all Calls feature requests are denied for EAS Login IDs but may be requested for EAS physical extensions where an EAS agent is logged in.

Multiple Monitors

Multiple Monitors provides the ability for up to three ASAI applications to monitor the same ACD Split or VDN domain.

This is not only helpful in environments were ICM is primary, it can also be used to add an OCM application to launch calls at off-peak times without disrupting the primary application in any way. Multiple Monitors can also be used to monitor an ACD split over 2 links in call environments where ASAI link failure recovery is important.

Value Queries

Value queries function identically in the EAS and traditional environments, except that the Extension Type/Class Information Query returns a new indication that a requested extension is an EAS Login ID along with an indication of whether the Login ID is currently logged in and where (in other words, at which physical extension).

Event Notification

Because all skill hunt groups are vector-controlled, event notification may not be requested on the basis of a skill hunt group extension. Event notification may, however, be requested on the basis of a controlling VDN extension. Generally, all event reports involving EAS agents contain the agent's physical extension rather than the agent's Login ID.

Adjunct-Controlled Skills

Agents with adjunct-controlled skills are considered to be adjunct-controlled agents. Adjunct-controlled agents exhibit the same behavior as agents within adjunct-controlled splits in the traditional ACD environment. The following list provides more details:

- Stations are locked for all logged-in adjunct-controlled agents. The only action an agent can take from the station is to go onhook (or unplug the headset) from an auto-answer station, which causes the agent to be logged out.
- Stations are unlocked whenever the controlling adjunct's ASAI link goes down. Stations are locked again when the adjunct's link is reestablished.
- The adjunct controls all skill/agent activities such as login, logout and change work-mode (with the exception of agent logout via onhook).
- Only adjunct-controlled calls can terminate to the extension of an adjunct-controlled agent.
- Only adjunct-controlled calls can terminate to an adjunct-controlled skill hunt group extension.

- Adjunct-controlled EAS Agents can be administered with only one skill. Accordingly, EAS agents may not mix adjunct-controlled and non-adjunct-controlled skills.

AUDIX

Calls to the EAS Agent Login ID can cover to AUDIX. Each agent must enter his or her Agent Login ID when calling AUDIX to obtain messages.

AUDIX agents are assigned to EAS agent extensions. These Login IDs are used for CMS and BCMS (G3V3 and later releases) tracking if the associated AUDIX skill hunt group is externally measured. The ‘‘aut-msg-wt’’ button can be used to indicate that the Login ID has a message.

An agent cannot have both AUDIX and non-AUDIX skills.

CMS (R3V2 and later releases)

The following is true for R3V2 and later release CMS Agent Tables:

- Separate Direct Agent database items starting with ‘‘DA_’’ are tracked.
- Standard reports combine statistics for Direct Agent calls and skill calls. However, reports can be customized to separate these statistical groupings.

The following is true for R3V2 CMS Skill Tables:

- Skill queues can be monitored for Direct Agent calls on the ‘‘Queue/Agent Summary’’ report.
- Direct Agent calls are not tracked.
- Agent time while on a Direct Agent call is tracked as ‘‘other’’ time.
- Non-ACD calls while in Direct Agent ACW are tracked.

The following is true for R3V2 CMS VDN/Vector Tables:

- Direct Agent calls and skill calls are combined as ACD calls.

Speech-Processing Adjuncts

Speech-processing adjuncts which have a ‘‘line’’ interface to the switch are able to initiate Direct Agent calls by dialing the Login ID for an agent.

EAS Agent LoginID Table

The following table indicates which DEFINITY features can be administered with an EAS Agent loginID.

Table 10-15. EAS LoginID Table

Feature	Administered LoginID?
Abbreviated Dialing Buttons	
7103A	Yes
Enhanced	Yes
Group	Yes
Personal	Yes
System	Yes
Agent-LoginID	
Port Extension	No
Announcements	No
Buttons	
abrdg_app	No
aut-msg-wt	Yes
brdg_app	No
busy-ind	Yes
data_ext	No
man_msg_wt	No
q-calls	No
q-time	No
signal	No
Call Processing	
Auto-Callback	No
Call Forward from Agent Login ID	No
Call Forward to Agent Login ID	Yes
Call Park	Yes
Hundreds group	No
LWC Retriever gets lagt msgs	Yes
Service observ Agent Login ID	Yes (G3V4 and later releases)
CDR Parameters	
Primary Extension	No

Table 10-15. EAS LoginID Table

Feature	Administered LoginID?
Secondary Extension	No
Code-Calling	Yes
Communication Link Form	
Communication Link Digits	No
Console Parameters	
CAS-backup ext	No
IAS Att Access Code	No
Coverage Groups	
Answer Group Member	No
Path	Yes
Measured Principals	
Coverage Measurement	No
Feature-Related Parameters	
ACA-referral dest.	No
ACA - long holding	No
ACA - short holding	No
Controlled out restriction	No
Controlled Terminal	No
Controlled Stn-to-Stn	No
DAA Extension	No
DID/Tie/ISDN announcement	No
Emergency Access Redirection	No
CDR output extension	No
SVN referral destination (announcement)	Yes
System LWC retriever	No
System Printer	No
Hospitality Parameters	
Journal Printer	No
LWC wakeup	No
PMS ext	No
PMS log	No
Routing on Voice Synthesis	No
Hunt Group Form	
Announcement extension	No

Table 10-15. EAS LoginID Table

Feature	Administered LoginID?
ASAI link	No
AUDIX extension	No
Calls Warning extension	No
Member	No
Night Service	No
Supervisor	Yes
Time Warning extension	No
Intercom Group Member	No
Intra-switch CDR	Yes
Listed Directory Number	
Member	No
Night Destination	Yes
Malicious Call Trace	
MCT Member	No
Permanent Switched Calls	No
Personal CO Line	No
Pickup Group Member	No
Remote Access Extension	No
Term Extension Group Member	No
Trunk Group	
Night Service	Yes
Incoming Destination	Yes
Member Night Service	Yes
Vector Administration	
adjunct extension	No
announcement	No
messaging	Yes
route-to	Yes

Upgrading to a G3 EAS Environment

For information about how to create a Call Center that uses EAS see "Call Vectoring/G3 EAS Option" on page J-10. For information on converting a Call Center to EAS, refer to Appendix K.

Call Vectoring Applications

11

Introduction

This chapter is intended to present several generic Call Vectoring applications a customer might use. Each application is based on one or more of the Call Vectoring features discussed in this guide. Vector Exercises are provided at the end of the chapter.

The following table identifies the feature(s) used within each example in this chapter. The examples are numbered according to the order in which they appear within the chapter. The name of the section in which each example appears is listed first.

Table 11-1. Applications and Corresponding Feature(s)

Section Title	Example No.	Feature(s) Used
Customer Service Center	1	Basic Call Vectoring
Automated Attendant	2	Call Prompting
DIVA and Data/Message Collection	3	Call Prompting, Basic Call Vectoring
Distributed Call Centers	4	Look-Ahead Interflow, Basic Call Vectoring
Help Desk	5	Adjunct Routing, Call Prompting, Basic Call Vectoring
Insurance Agency/Service Agency	6	Basic Call Vectoring, Call Prompting, Rolling ASA, EWT, VDN Calls, and ANI Routing
Warranty Service	7	Basic Call Vectoring, EAS
Resort Reservation Service	8	Basic Call Vectoring, Adjunct Routing, Call Prompting, EAS

Customer Service Center

Example 1 presents a scenario where a customer service center is open weekdays from 8 a.m. until 5 p.m. The center provides two separate telephone numbers. One number is for ordinary customers, while the other number is for priority customers. The following three vectors illustrate how calls to the customer service center are handled.

```
VDN (extension=1021  name=‘‘Customer Serv’’  vector=21)
Vector 21:
      1. goto vector 29 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 10 if calls-queued in split 1 pri l > 10
      4. queue-to main split 1 pri m
      5. wait-time 10 seconds hearing ringback
      6. announcement 3521
      7. wait-time 50 seconds hearing music
      8. announcement 3522
      9. goto step 7 if unconditionally
     10. busy
VDN (extension=1022  name=‘‘Priority Cust’’  vector=22)
Vector 22:
      1. goto vector 29 if time-of-day is all 17:00 to all 08:00
      2. goto vector 29 if time-of-day is fri 17:00 to mon 08:00
      3. goto step 12 if calls-queued in split 1 pri h > 10
      4. queue-to main split 1 pri h
      5. announcement 3521
      6. wait-time 10 seconds hearing music
      7. check-backup split 2 pri h if oldest-call-wait < 20
      8. check-backup split 3 pri h if oldest-call-wait < 20
      9. announcement 3522
     10. wait-time 60 seconds hearing music
     11. goto step 7 if unconditionally
     12. route-to number 0 with cov n if unconditionally
No VDN
Vector 29:
      1. announcement extension 3529
      2. wait-time 10 seconds hearing silence
      3. disconnect after announcement 3529
```

Figure 11-1. Example 1: Customer Service Center

First, let’s assume that a priority customer places a call. In such a case, if the correct number is dialed, vector 22 is accessed. The first two steps of this vector determine if the call arrives during nonbusiness hours. If the call arrives between 5:00 p.m. and 8:00 a.m. on any given day, Step 1 routes the call to Vector 29. Step 2 does the same if the call arrives during the weekend (that is, between 5:00 p.m. Friday and 8:00 a.m. Monday). If vector 29 is accessed, the caller is given the appropriate announcement twice (Steps 1 and 3) and is then disconnected (Step 3).

If the call is placed during business hours, Step 3 of vector 22 determines if the number of calls queued in the main split exceeds 10. If so, control is sent to Step 12, which routes the call to the attendant. If not, the call is queued to the main split (Step 4). Thereafter, if necessary, the appropriate announcement is provided (Step 5), followed by a wait period (Step 6).

If the call is not answered by this time, Steps 7 and 8 attempt to queue the call to a backup split (*2* and *3*, respectively). The call is queued to either split if the oldest call therein has been waiting fewer than 20 seconds. Whether or not the call is queued, Steps 9 through 11 implement an announcement-wait cycle that continues until an agent answers the call, or until the caller abandons the call.

A call placed by a nonpriority customer is processed by vector 21. Vector 21 provides a treatment similar to that provided by vector 22. The three differences are that: the nonpriority customer’s call is not given the chance to be queued to more than one split; the priority customer’s call is given a higher priority in the queue; and the priority customer’s call routes to an operator when there are too many calls queued whereas the nonpriority customer routes to a busy signal.

Automated Attendant

Example 2 illustrates Automated Attendant, which is one of the applications that can be supported by the Call Prompting feature. Automated Attendant allows the caller to enter the extension of the party the caller would like to reach. Depending upon the parameters established, the user can enter up to 16 digits from a touch-tone phone.

Automated Attendant is usually used for customers without DID trunks whose callers know the extension of the people they are calling. Because it reduces the need for ‘‘live attendants,’’ Automated Attendant allows the customer to reduce costs.

Here’s an example of a vector that implements Automated Attendant:

```
1. wait-time 0 seconds hearing ringback
2. collect 5 digits after announcement 30001
   (‘‘You have reached Ridel Publications in Greenbrook.
   Please dial a 5-digit extension or wait for the
   attendant.’’)
3. route-to digits with coverage y
4. route-to number 0 with cov n if unconditionally
5. stop
```

Figure 11-2. Example 2: Automated Attendant

Step 1 of this vector contains the *wait-time* command, which is placed before the *collect digits* command in Step 2 to provide the caller with ringback in the event that a TTR is not immediately available. (Recall that a TTR must be connected in order for the *collect digits* command to take effect.) Once a TTR is connected, the caller is prompted to enter the destination extension of the party he or she would like to reach (Step 2). The *collect digits* command in Step 2 collects the digits. Thereafter, the *route-to digits* command in Step 3 attempts to route the call to the destination.

If the *route-to digits* command fails (because the caller fails to enter any digits, or because the digits entered do not comprise a valid extension), the *route-to number* command in Step 4 routes the call to the attendant. However, as long as the destination is a valid extension, the *route-to digits* command succeeds, coverage applies, and vector processing terminates. (Even if the destination is busy, vector processing terminates because coverage call processing takes effect.)

DIVA and Data/Message Collection

Example 3 involves a mutual fund company that is open 24 hours a day, seven days a week. All incoming calls are directed to a single VDN extension that maps to a main vector. The main vector presents a menu of options to the calling party, and it uses Call Prompting to determine the desired service. Three services are offered, and they are identified and described as follows:

- *New accounts* enables the customer to open a new account.
- *Account inquiries* enables the customer to make inquiries concerning his or her account.
- *Net asset values* enables the customer to hear information concerning the net asset values of company's funds.

If the caller selects ‘‘account inquiries,’’ he or she is prompted to input his or her account number before being answered by an agent. The agent can display this number via use of the CALLR-INFO button, if the button is available and needed.

NOTE:
If the agent has two-line display supported by the system, the account number is automatically displayed on the second line. The supported display terminals include: 7404, 7407, 7444, 8434 and CALLMASTER.

NOTE:
CALLMASTER sets are unable to display CALLR-INFO information for a ringing call. However, this information is displayed once the call is answered.

This example uses three other applications that can be supported by the Call Prompting feature. These applications are described as follows:

- **Data In/Voice Answer (DIVA)** allows a caller to receive information on a topic selected at the prompt. The caller selects the desired topic by entering the appropriate digit(s).
- **Data Collection** provides a method of collecting digits from a caller. The digits requested comprise an official number of some sort (for example, Social Security Number), and they help the Call Center process the call more efficiently.

- **Message Collection** allows the caller to leave a recorded message for the customer in lieu of waiting for the customer to answer his or her call.

The following four vectors illustrate how the mutual fund company handles telephone calls.

```
VDN (extension=1030  name=‘‘ABC Inv’’  vector=30  display override=‘‘y’’)
Vector 30
      1. wait-time 0 secs hearing ringback
      2. collect 1 digits after announcement 3531
         (‘‘Thank you for calling ABC Investments. If
         you wish to open a new account, please dial 1.  If
         you wish to make an account inquiry, please dial 2.
         If you wish to know the current net asset values of
         our funds, please dial 3.’’)
      3. route-to number 1031 with cov y if digit = 1
      4. route-to number 1032 with cov y if digit = 2
      5. route-to number 1033 with cov y if digit = 3
      6. route-to number 0 with cov n if unconditionally
      7. disconnect after announcement none
VDN (extension=1031  name=‘‘New Account’’  vector=31)
Vector 31
      1. queue-to main split 1 pri t
      2. announcement 3535
      3. wait-time 10 secs hearing music
      4. collect 1 digits after announcement 4020
         (‘‘We’re sorry.  All of our operators are busy at
         the moment.  If you’d like to leave your name and
         telephone number so that we can get back to you,
         dial 1.’’)
      5. goto step 9 if digit = 1
      6. announcement 3537
      7. wait time 50 secs hearing music
      8. goto step 6 if unconditionally
      9. messaging split 5 for extension 4000
     10. announcement 3538 (‘‘We’re sorry, we cannot take
         your message at this time.  You may continue to hold, or
         you can call back later.’’)
     11. goto step 6 if unconditionally
VDN (extension=1032  name=‘‘Account Enq’’  vector=32)
Vector 32:
      1. wait-time 0 secs hearing ringback
      2. collect 6 digits after announcement 3533
         (‘‘Please enter your 6-digit account number.’’)
      3. queue-to main split 1 pri m
      4. announcement 3535
      5. wait-time 60 secs hearing music
      6. goto step 4 if unconditionally
VDN (extension=1033  Name=‘‘Net Asset Val’’  Vector=33)
Vector 33:
      1. disconnect after announcement 3534
         (‘‘The net asset values of our funds at the close
         of the market on Wednesday, May 15 were as follows:
         ABC Growth.....33.21.....up 33 cents; ABC
         High Yield.....11.48.....down 3 cents.’’)
```

Figure 11-3. Example 3: DIVA and Data/Message Collection

When the call is placed, vector processing begins in vector 30, which is the main vector. Step 1 of the vector contains the *wait-time* command, which is placed before the *collect digits* command in Step 2 to provide the caller with feedback in the event a TTR is not immediately available. Once a TTR is connected, the *collect digits* command provides an announcement requesting the caller to enter *1, 2,* or *3*, depending upon the service desired. If the caller enters a digit other than one of the three mentioned, or if the caller fails to enter any digits within 10 seconds, the command fails, and the call is routed to the attendant (Step 6). On the other hand, if the caller enters *1, 2,* or *3* within 10 seconds, the call is routed to the vector specified in the appropriate *route-to number* command, which appears in Steps 3, 4, and 5.

Let’s say that, when prompted, the caller enters *3* because he or she wants to learn about the net asset values of the company’s funds. In such a case, the *route-to number* commands in Step 3 and in Step 4 fail because, in each case, the digit tested for in the condition portion of the command is not *3*. However, the *route-to number* command in Step 5 succeeds because the digit tested for matches the one entered by the caller. Accordingly, the call is routed to VDN extension 1033, and vector processing continues in vector 33.

The *announcement* command in Step 1 of vector 33 provides the caller with the information on net asset values and then disconnects the call.

The process just described, whereby the caller receives information as a result of making a request at the prompt, is an example of the *Data In/Voice Answer (DIVA)* application.

Returning to the main vector, suppose another caller wants to make an enquiry into his or her account and the caller enters *2* when prompted. In such a case, Step 3 fails, but Step 4 succeeds. Accordingly, the call is routed to VDN extension 1032, and vector processing continues in vector 32.

The *collect digits* command in Step 1 of vector 32 first requests the caller to enter his or her 6-digit account number. The command then collects the digits entered by the caller. Whether or not the caller correctly enters the digits, the *queue-to main split* command in Step 2 queues the call. If an agent does not immediately answer the call, the standard announcement is provided in Step 3 and, if necessary, a delay is provided in Step 4. The *goto step* command in Step 5 returns call control back to Step 3, thus ensuring that the announcement-wait cycle will continue until the agent answers the call, or until the caller abandons the call.

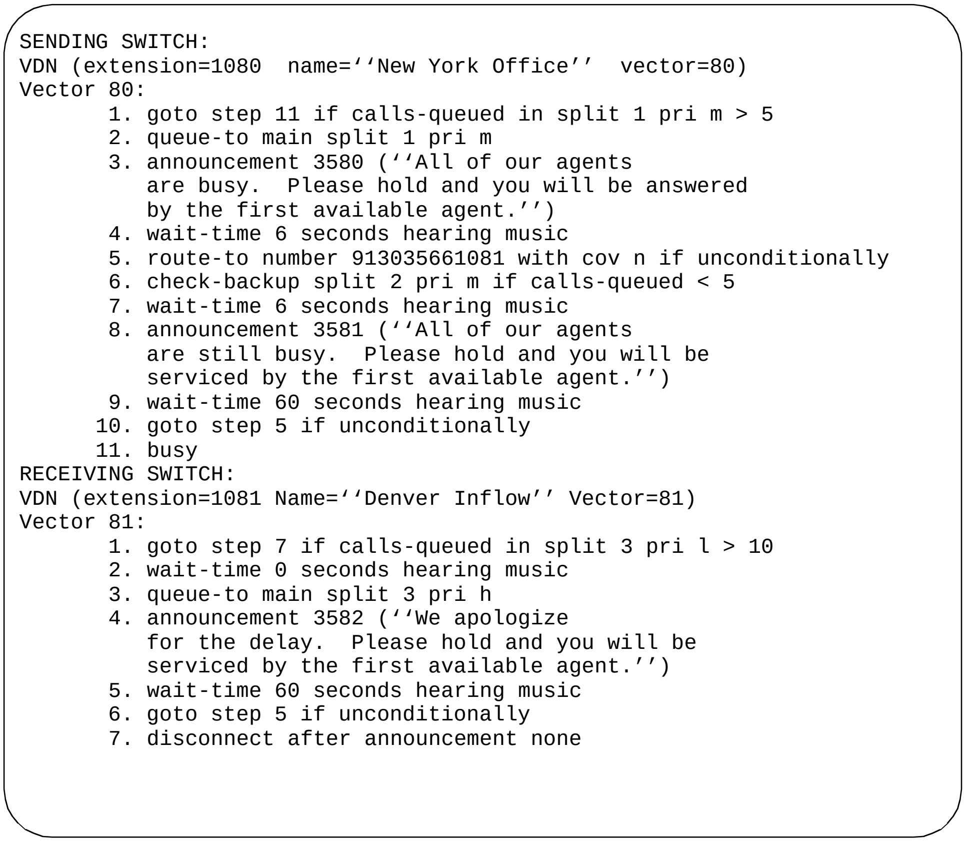
The process just described, whereby the caller, when prompted, enters digits that comprise an official number (an account number, in this case), is an example of the Data Collection application. If the agent has a CALLR-INFO button or a two-line display, the agent can see the digits entered by the caller. As a result, the agent need not request the account number from the caller.

Finally, suppose a third caller wants to open an account and that he or she enters *1* when prompted in the main vector. In such a case, Step 3 of the main vector is successful. Accordingly, the call is routed to VDN extension 1031, and vector processing continues in vector 31.

In Step 1 of vector 31, the call is queued to the main split. Thereafter, if necessary, Step 2 provides the appropriate announcement, and Step 3 provides a delay period. The announcement in Step 4 provides the caller with the option of leaving a recorded message for the mutual fund company instead of having his or her call wait in queue. (This is an example of the Message Collection application.) The caller is instructed to enter *1* if he or she wishes to leave a recorded message. If the caller does not enter *1*, the *goto step* command in Step 5 fails, and an announcement-wait cycle is implemented by Steps 6, 7, and 8 until the call is answered or abandoned. If the caller does enter *1* within 10 seconds, Step 5 passes control to Step 9. The *messaging split* command in Step 9 attempts to connect the caller to an AUDIX or Message Center split so that the caller can leave a message. If the connection is made, the caller first hears ringback and can then leave a message. If the connection is not made, the step is unsuccessful, and Step 10 provides an announcement indicating that a connection could not be made. Thereafter, the *goto step* command in Step 11 sends call control back to Step 6, which is the first step of the aforementioned announcement-wait cycle.

Distributed Call Centers

Example 4 involves two distributed call centers, one in New York and the other in Denver. Calls to the New York call center are queued to up to two splits. If calls remain unanswered for a period of time, a Look-Ahead Interflow call attempt is made to the Denver call center. If there are 10 or fewer queued calls in Denver, the Look-Ahead call attempt is accepted and serviced there. Otherwise, the call is denied and remains in queue in New York until an agent becomes available. The following two vectors illustrate this procedure.



```
SENDING SWITCH:
VDN (extension=1080  name=‘‘New York Office’’  vector=80)
Vector 80:
       1. goto step 11 if calls-queued in split 1 pri m > 5
       2. queue-to main split 1 pri m
       3. announcement 3580 (‘‘All of our agents
          are busy.  Please hold and you will be answered
          by the first available agent.’’)
       4. wait-time 6 seconds hearing music
       5. route-to number 913035661081 with cov n if unconditionally
       6. check-backup split 2 pri m if calls-queued < 5
       7. wait-time 6 seconds hearing music
       8. announcement 3581 (‘‘All of our agents
          are still busy.  Please hold and you will be
          serviced by the first available agent.’’)
       9. wait-time 60 seconds hearing music
      10. goto step 5 if unconditionally
      11. busy
RECEIVING SWITCH:
VDN (extension=1081 Name=‘‘Denver Inflow’’ Vector=81)
Vector 81:
       1. goto step 7 if calls-queued in split 3 pri l > 10
       2. wait-time 0 seconds hearing music
       3. queue-to main split 3 pri h
       4. announcement 3582 (‘‘We apologize
          for the delay.  Please hold and you will be
          serviced by the first available agent.’’)
       5. wait-time 60 seconds hearing music
       6. goto step 5 if unconditionally
       7. disconnect after announcement none
```

Figure 11-4. Example 4: Distributed Call Centers

In this example, vector 80 is on the sending switch from a call center in New York, while vector 81 is on the receiving switch at a call center in Denver.

In the sending switch, the call is queued to split 1 at a medium priority (Step 2) if the condition in Step 1 is met. If the condition is not met, the call is routed to busy in Step 11.

If the call is queued but not immediately answered, an announcement (Step 3) and music (Step 4) are provided. If the call is still not answered at this point, Step 5 places a Look-Ahead Interflow call attempt to the receiving switch, on which vector 81 resides.

Step 1 in the receiving switch determines whether the call can be serviced in Denver. If the number of calls queued at any priority in split 3 is greater than 10, vector 81 cannot service the call. In such a case, control is passed to Step 7, which rejects the Lookahead Interflow call attempt. However, if the test in Step 1 succeeds, the call is queued by the receiving switch in split 3 at a high priority (Step 3). This results in the Lookahead Interflow call attempt being accepted.

Accordingly, the call is removed from the main split queue in New York, and control is passed to the Denver switch, where vector processing continues at Step 4.

If the receiving switch does not accept the Look-Ahead Interflow call attempt, control is passed to Step 6 of the sending vector. This step then queues the call to split 2 at a medium priority, provided that there are fewer than five calls queued in that split. Thereafter, the customary announcement-wait sequence is implemented (Steps 7, 8, and 9). Finally, if necessary, Step 10 sends control back to Step 5, which makes another Look-Ahead Interflow attempt, and the cycle is repeated.

NOTE:
To avoid confusing the caller, the treatment provided at the receiving switch should be consistent with the treatment provided at the sending switch. In Example 4, note that the caller hears music (and never ringback or silence) at the sending switch. Accordingly, music should be (and, in our example, is) featured at the receiving switch.

Help Desk

Example 5 involves a help desk at a computer firm. The help desk is configured into three groups. One group handles hardware problems, the second group handles software problems, and the third group handles general problems. For this application, the information provided in the ASAI Route request (that is, calling party number, called number, collected digits) is used to route the call to the most appropriate agent. Such an agent might be the one that last serviced the caller, or it might be the next available agent for the specific caller. Also, based on switch traffic conditions and the caller-entered digit, the call can be diverted to other destinations, such as other ACD splits, announcements, or switches/PBXs.

The following vector illustrates the help desk application:

```
1. collect 1 digits after announcement 4704
   (‘‘Welcome to the TidyBits Computer Corporation help desk.
   If you have a question about hardware, please dial 1.
   If you have a question about software, please dial 2.
   If you have a general question, please dial 3.’’)
2. adjunct routing link 2400
3. wait-time 4 seconds hearing ringback
4. route-to number 3710 with cov y if digit = 1
5. route-to number 3720 with cov y if digit = 2
6. route-to number 3730 with cov y if digit = 3
7. route-to number 0 with cov n if unconditionally
8. stop
```

Figure 11-5. Example 5: Help Desk

In Step 1 of this vector, the caller is instructed to enter *1, 2,* or *3*, depending upon the service (hardware, software, general) he or she desires. Thereafter, the *adjunct routing* command in Step 2 instructs the switch to send a Route request to the adjunct processor, which is connected to extension 2400. The Route request contains the called party number, the calling party number, and the digit collected in Step 1, along with the other pertinent information for adjunct routing (see Chapter 9). If one of these digits is not entered, and if the adjunct does not return a route, the call is eventually routed to the attendant (Step 7).

If the *adjunct routing* command in Step 2 succeeds, the adjunct uses the information included in the Route request to select the appropriate route for the call. Let's assume the caller enters *1* and the *adjunct routing* command *succeeds*. In such a case, if the caller is judged to be a ''prime'' hardware customer, the call might be routed to one of a handful of specific agents who are assigned to handle such customers. On the other hand, if the caller is judged to be a ''casual'' hardware customer, the call might be routed to a larger group of ACD agents before being queued, or to an appropriate announcement.

Finally, let's assume that the caller enters *1* and that the *adjunct routing* command fails. In such a case, the call is routed by the *route-to number* command in Step 4, probably to a vector that queues the call or provides an appropriate announcement.

Insurance Agency/Service Agency

Example 6 is an insurance company Call Center. It handles calls from: independent field agents; policy holders with claims; policy holders needing customer service; and several general service agency type 800 number client accounts. Each of the different types of calls has its own 800 number that routes the calls to associated VDNs. The following list describes the Call Center requirements.

- The independent field agents require fast service. They call the company to find out the latest rates for specific clients, to setup policies, to make adjustments, and so on. Often their clients are waiting as they call. Therefore the insurance company wants to maintain an Average Speed of Answer (rolling-ASA) of 30 seconds or less for field agent calls. These are the most important calls and are given high priority in queues.
- The calls to claims must be separated by area code. The claims agents receive different training based on the area of the country for the claim. A particular group of agents can be given training for more than one area code. Therefore, area codes do not need to be tested individually and can be grouped in Vector Routing Tables.
- The insurance company wants to give customer service callers an announcement indicating how long they can expect to wait for service.

- The insurance agency is also selling spare call center capacity to client accounts. The account contracts are provided on the basis that only so many calls to a particular account will be accepted at any given time.

In this example, rolling ASA Routing is used to maintain the rolling ASA objective of 30 seconds or less for field agent calls. ANI Routing is used to partition calls based on area code and route the call to the appropriate claims agents. EWT Routing is used to notify customer service callers of their expected wait time if it is longer than 60 seconds. VDN Calls Routing is used to regulate the number of calls to service agency clients.

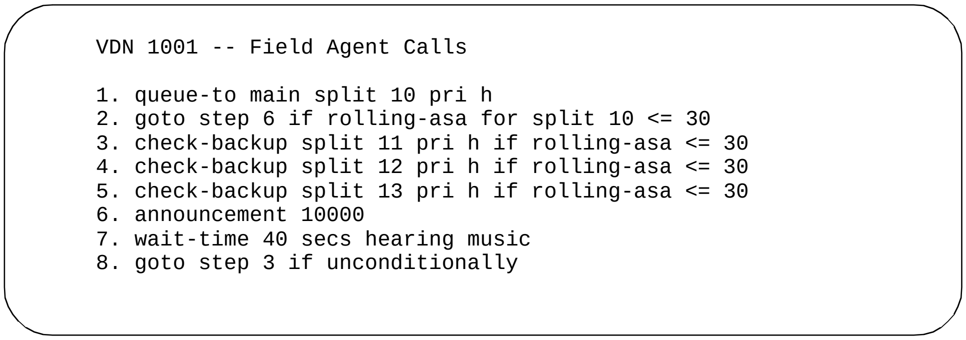
The following table shows the VDNs and vectors associated with each type of call.

Table 11-2. VDN Table for Insurance/Service Agency

Type of Service	VDN Number	Vector Number
Field Agents	1001	1
Claims	1002	2
Customer Service	1003	3
Client 1	1004	4
Client 2	1005	5

NOTE:
To clearly demonstrate the features described in this example, the sample vectors do not include tests for unstaffed or full queues, out of hours operation and so on.

The following vector could be used to maintain a rolling ASA of 30 seconds for field agent calls.



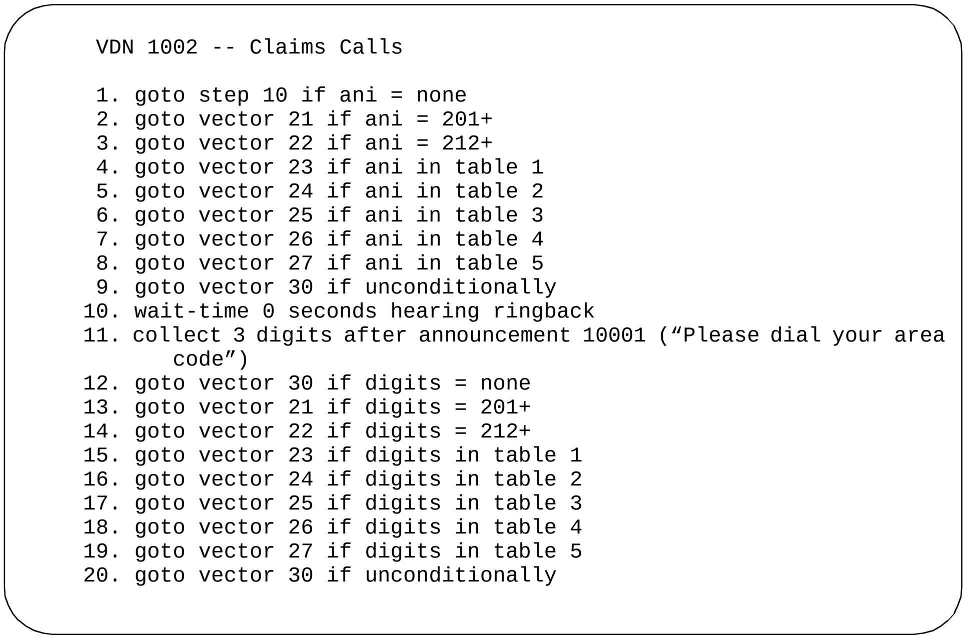
```
VDN 1001 -- Field Agent Calls

1. queue-to main split 10 pri h
2. goto step 6 if rolling-asa for split 10 <= 30
3. check-backup split 11 pri h if rolling-asa <= 30
4. check-backup split 12 pri h if rolling-asa <= 30
5. check-backup split 13 pri h if rolling-asa <= 30
6. announcement 10000
7. wait-time 40 secs hearing music
8. goto step 3 if unconditionally
```

Figure 11-6. Example 6: Field Agent Vector

Step 1 queues the call to the main split. If the main split is currently answering calls within the target time of 30 seconds Step 2 bypasses all of the backup splits and goes directly to the announcement in Step 6. The assumption is that the call will be handled by split 10 within the time constraints. However, if the call is not answered by the time vector processing reaches Step 8, the backup splits are checked at that time.

If the rolling ASA for the main split is greater than 30 seconds, Steps 3, 4, and 5 check backup splits. The call is queued to any of these splits that have a rolling ASA of 30 seconds or less. If the call still is not answered by the time vector processing reaches Step 8, then the backup splits are checked again.

The following vector could be used to route claims calls by area code.

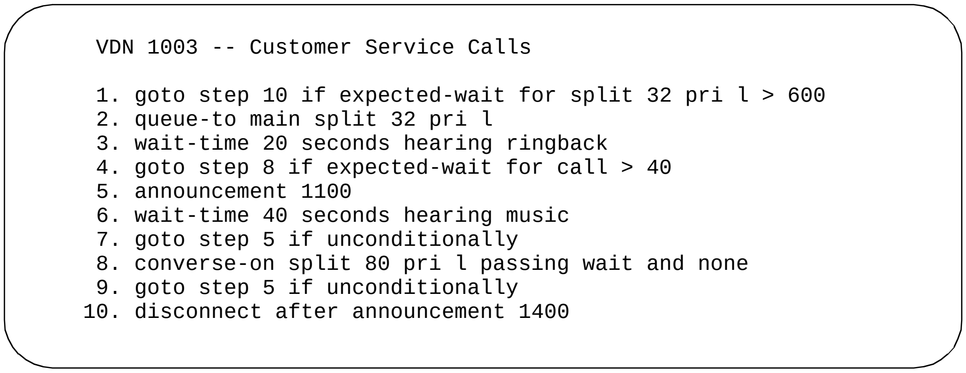


```
 VDN 1002 -- Claims Calls

 1. goto step 10 if ani = none
 2. goto vector 21 if ani = 201+
 3. goto vector 22 if ani = 212+
 4. goto vector 23 if ani in table 1
 5. goto vector 24 if ani in table 2
 6. goto vector 25 if ani in table 3
 7. goto vector 26 if ani in table 4
 8. goto vector 27 if ani in table 5
 9. goto vector 30 if unconditionally
10. wait-time 0 seconds hearing ringback
11. collect 3 digits after announcement 10001 (“Please dial your area
       code”)
12. goto vector 30 if digits = none
13. goto vector 21 if digits = 201+
14. goto vector 22 if digits = 212+
15. goto vector 23 if digits in table 1
16. goto vector 24 if digits in table 2
17. goto vector 25 if digits in table 3
18. goto vector 26 if digits in table 4
19. goto vector 27 if digits in table 5
20. goto vector 30 if unconditionally
```

Figure 11-7. Example 6: Claims Vector

Each Vector Routing Table referred to in Figure 11-7 contains a list of area codes with the “+” wildcard. Each list of area codes is handled by a specific group of agents. Vectors 21 through 27 queue calls to the appropriate group of agents. Vector 30 provides a live agent to screen calls that have area codes not listed in any table or vector step. It also provides access to an agent when ANI is not available and the caller has not entered an area code when prompted.

The following vector will notify customer service callers of their expected wait time unless they will not have long to wait

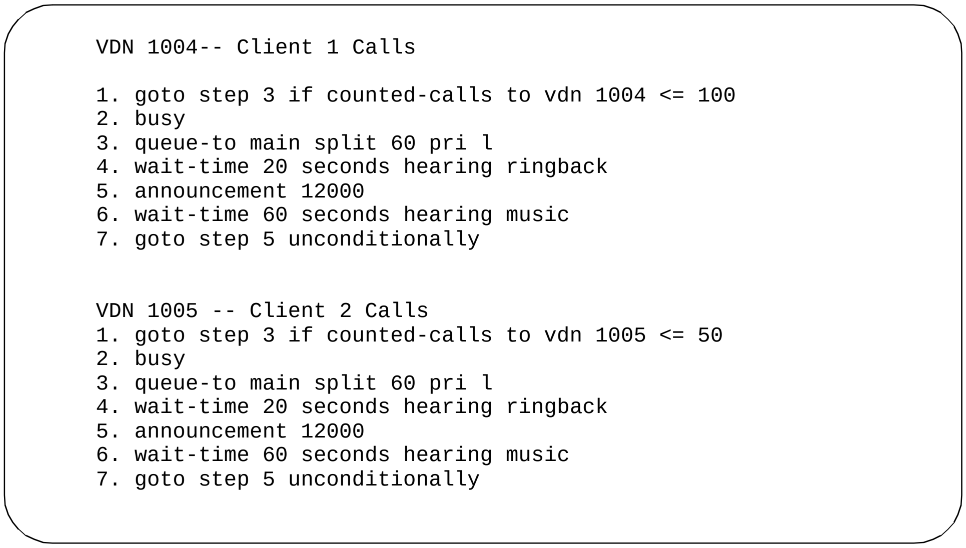


```
VDN 1003 -- Customer Service Calls

 1. goto step 10 if expected-wait for split 32 pri l > 600
 2. queue-to main split 32 pri l
 3. wait-time 20 seconds hearing ringback
 4. goto step 8 if expected-wait for call > 40
 5. announcement 1100
 6. wait-time 40 seconds hearing music
 7. goto step 5 if unconditionally
 8. converse-on split 80 pri l passing wait and none
 9. goto step 5 if unconditionally
10. disconnect after announcement 1400
```

Figure 11-8. Example 6: Customer Service Vector

In step 1, callers who would wait more than 10 minutes are routed to a “call back later announcement.” Step 4 routes callers to a Conversant VRU to be given the expected wait time announcement while they hold their place in queue.

The following vectors can be used to regulate the number of calls to service agency clients. In this example, Client 1 has contracted for 100 simultaneous calls while client 2 has contracted for only 50 simultaneous calls.



```
VDN 1004-- Client 1 Calls

1. goto step 3 if counted-calls to vdn 1004 <= 100
2. busy
3. queue-to main split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 12000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally


VDN 1005 -- Client 2 Calls
1. goto step 3 if counted-calls to vdn 1005 <= 50
2. busy
3. queue-to main split 60 pri l
4. wait-time 20 seconds hearing ringback
5. announcement 12000
6. wait-time 60 seconds hearing music
7. goto step 5 unconditionally
```

Figure 11-9. Example 6: Service Agency Clients Vectors

In both vectors the first step routes calls to queue if the number of contracted calls is not exceeded. Otherwise callers receive busy signal.

Warranty Service (with EAS)

Example 7 deals with a major appliance company that offers one year warranties and extended warranties on its major appliances (dishwashers, refrigerators, washers, and dryers). The warranties are printed in English and Spanish to accommodate customers who speak and understand these languages. Naturally, callers need to speak with someone who is familiar with the appliances they have bought and who speaks the appropriate language. Accordingly, 800 numbers are provided for calling English-speaking agents as well as Spanish-speaking agents. Bilingual agents with Spanish-speaking Skills are hired so that they can back up the groups of English-speaking agents. Agents are trained first on all appliance models of a certain type and then on all appliance models for a room (such as the kitchen, the laundry room, etc).

The following Skills are needed for the warranty service call center:

Table 11-3. Skill Table for the Warranty Service Call Center

Appliance Type	English Skill #	Spanish Skill #
Kitchen Appliances	10	20
Dishwashers	11	21
Refrigerators	12	22
Laundry Appliances	30	40
Washers	31	41
Dryers	32	42
Supervisors		100

The VDN Skill Preferences are set up as follows:

Table 11-4. VDN Skill Table for the Warranty Service Call Center

VDN - Skill Preferences							
English				**Spanish**			
Dishwasher	Refrig	Washer	Dryer	Dishwasher	Refrig	Washer	Dryer
VDN: 1100	VDN: 1101	VDN: 1102	VDN: 1103	VDN: 1200	VDN: 1201	VDN: 1202	VDN: 1203
1st:11	1st:12	1st:31	1st:32	1st:21	1st:22	1st:41	1st:42
2nd:10	2nd:10	2nd:30	2nd:30	2nd:20	2nd:20	2nd:40	2nd:40
3rd:20	3rd:20	3rd:40	3rd:40	3rd:	3rd:	3rd:	3rd:

The agent Skills are set up as follows:

Table 11-5. Agent Skills for the Warranty Service Call Center

Agent	Agent Skills			
	Primary		Secondary	
Jan	42	40	41	30
Judy	100	--	--	--
Sam	31	--	--	--
Sue	32	--	30	--

Once Skills are assigned to VDNs and to agents, calls are directed to the appropriate vector.

The goal of the warranty service call center is to answer 80 percent of the incoming calls within 20 seconds. Accordingly, if a call directed to a vector is not answered by the time the announcement finishes, a second group of agents is viewed, thus enlarging the agent pool. If the call is not answered within the following 10 seconds, a third group of agents is viewed.

Since the call center has only a few bilingual agents, the center's management wants to reserve these agents for Spanish-speaking callers. This can be done by giving Spanish-speaking callers a higher priority in the vector or by assigning a higher priority to Spanish Skills. Also, if a Spanish-speaking caller waits more than 30 seconds for service, a supervisor of the Spanish-speaking Skills takes the calls.

The following two figures illustrate the setup for the warranty service call service. Specifically, the figures show the vectors and call flows for callers with a broken washer or dryer. Separate vectors are used to provide an announcement in Spanish and in English (see Step 2). The same two vectors can be used for callers with broken dishwashers and refrigerators.

The first figure shows how the call comes into the network and is then directed to the appropriate VDN, which in turn points to the appropriate vector. For each VDN, the corresponding VDN Skills are indicated.

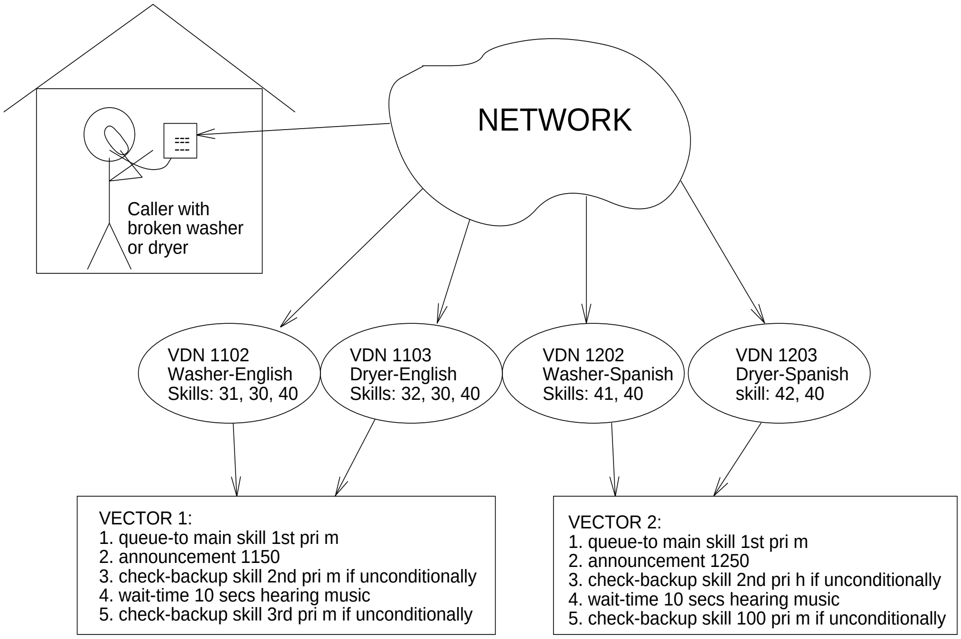

NETWORK
Caller with
broken washer
or dryer
VDN 1102
Washer-English
Skills: 31, 30, 40
VDN 1103
Dryer-English
Skills: 32, 30, 40
VDN 1202
Washer-Spanish
Skills: 41, 40
VDN 1203
Dryer-Spanish
skill: 42, 40
VECTOR 1:
1. queue-to main skill 1st pri m
2. announcement 1150
3. check-backup skill 2nd pri m if unconditionally
4. wait-time 10 secs hearing music
5. check-backup skill 3rd pri m if unconditionally
VECTOR 2:
1. queue-to main skill 1st pri m
2. announcement 1250
3. check-backup skill 2nd pri h if unconditionally
4. wait-time 10 secs hearing music
5. check-backup skill 100 pri m if unconditionally

Figure 11-10. Example 6: Warranty Service Call Center (Part 1)

The second figure shows how the vector-processed call is directed to the appropriate call queue. The figure also shows how the call is directed to the appropriate agent(s). The agent Skills are indicated below each agent’s name. Dashed lines indicate backup or secondary Skills.

NOTE:
Only a small sample of agents is shown in this figure.

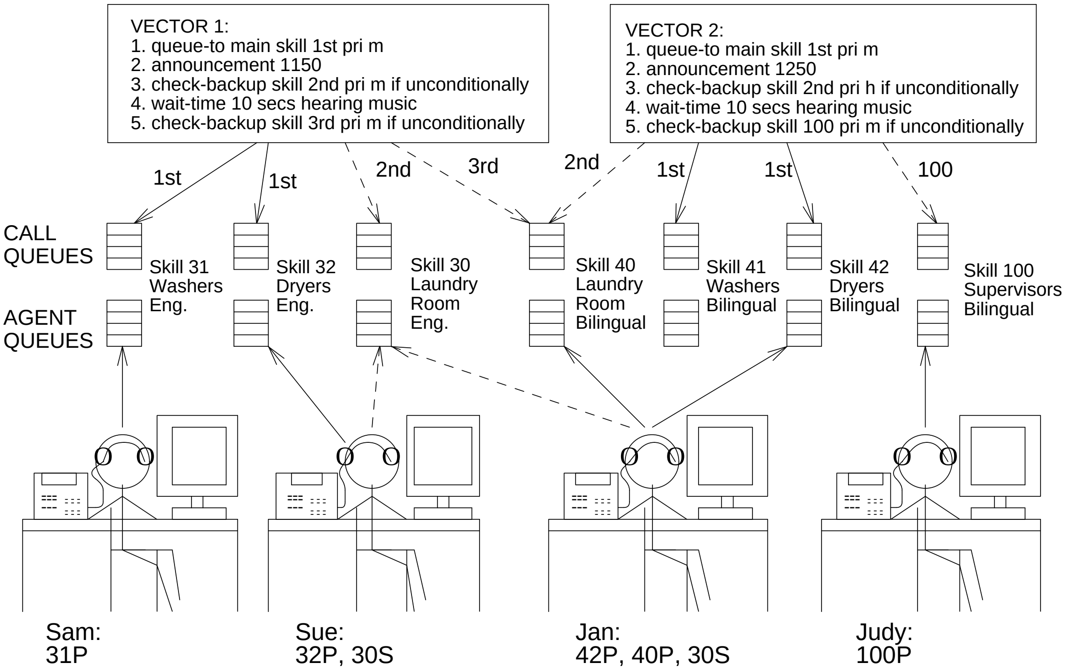

VECTOR 1:
1. queue-to main skill 1st pri m
2. announcement 1150
3. check-backup skill 2nd pri m if unconditionally
4. wait-time 10 secs hearing music
5. check-backup skill 3rd pri m if unconditionally
VECTOR 2:
1. queue-to main skill 1st pri m
2. announcement 1250
3. check-backup skill 2nd pri h if unconditionally
4. wait-time 10 secs hearing music
5. check-backup skill 100 pri m if unconditionally
1st
1st
2nd
3rd
2nd
1st
1st
100
CALL QUEUES
AGENT QUEUES
Skill 31 Washers Eng.
Skill 32 Dryers Eng.
Skill 30 Laundry Room Eng.
Skill 40 Laundry Room Bilingual
Skill 41 Washers Bilingual
Skill 42 Dryers Bilingual
Skill 100 Supervisors Bilingual
Sam: 31P
Sue: 32P, 30S
Jan: 42P, 40P, 30S
Judy: 100P

Figure 11-11. Example 6: Warranty Service Call Center (Part 2)

Let's assume that a Spanish-speaking caller has a broken dryer and decides to call the warranty service call center. In such a case, the caller dials the appropriate number. The call then enters the network (switch) and is directed to VDN 1203, which points to Vector 2. As illustrated earlier, VDN Skill preferences 42 (dryers) and 40 (laundry appliances) are administered as the 1st and 2nd Skill preferences, respectively, for VDN 1203. (Recall also that these preferences are actually subpreferences under the Skill "Spanish.'')

Once vector processing starts, the *queue-to main skill* command in Step 1 of Vector 2 queues the call to the Skill group corresponding to the first VDN Skill (42-Dryers Bilingual). If an agent with Skill 42 (Jan, for example) is available, this agent answers the call. If such an agent is not available, the appropriate delay announcement in Step 2 is played. Thereafter, the *check backup skill* command in Step 3 attempts to queue the call to the Skill group corresponding to the 2nd VDN Skill (40-Laundry Appliances Bilingual). Accordingly, if an agent with Skill 40 is available (Jan, for example), that particular agent answers the call. Otherwise, as a last resort, a wait period is provided in Step 4, and the *check backup skill* command in Step 5 checks the "specific" Skill (100-Supervisors Bilingual) for available agents.

Resort Reservation Service (with EAS)

Example 8 deals with a resort company that places a variety of ads in magazines for information on a particular resort or state. Callers responding to these ads can dial one of several numbers provided in the ad. A call center makes the reservations for the resort company. To satisfy one request voiced by many callers to the service, an effort is made to have callers connected to an agent who has visited the resort they are interested in visiting. Also, the resort company has determined it is easier to sell additional sightseeing packages if the agent has a regional accent.

Placing the Reservation

To respond to an ad, the caller can dial a number that directly routes him or her to a VDN for that state's resorts. As an alternative, the caller can dial the general number for the resort chain and be serviced via Call Prompting. The following sections discuss these methods.

Specific Number Dialing

The call center is set up in such a way that a VDN with an accompanying set of VDN Skill Preferences is assigned to each state that has a resort. For example, the following Skill Preferences are assigned to Texas VDN 3222:

Table 11-6. VDN 3222 Skill Preferences Assignments for the Resort Reservation Service

Texas VDN 3222 - Skill Preferences		
1st:	30	Agent who has a Texas accent and has visited resorts in Texas
2nd:	31	Agent who has visited resorts in Texas
3rd:	130	Any agent who can take a reservation

The following figure shows how a call to VDN 3222 can be processed vis-a-vis Call Vectoring:

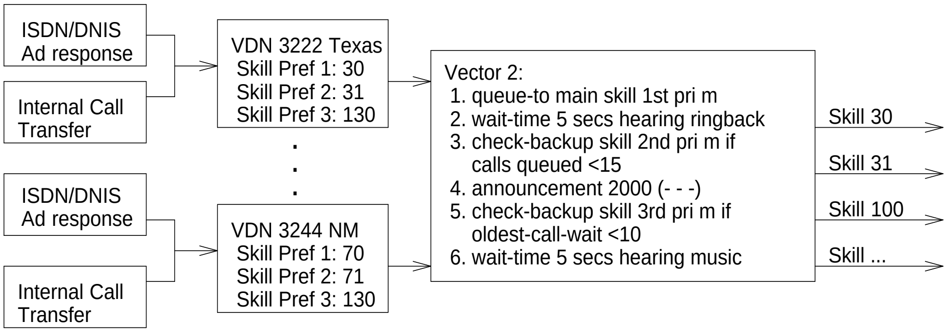


Figure 11-12. Example 7A: Process Involving Specific Number Dialing

For this process, a single VDN for each state is assigned to Vector 2. Accordingly, our diagram shows the VDN and the associated VDN Skills for two states: Texas and New Mexico.

Let’s assume that a caller would like information on resorts in Texas and dials the appropriate number (for example, 615-3222). In such a case, the call enters the switch and is directed to VDN 3222, which points to Vector 2.

Once vector processing starts, the *queue-to main skill* command in Step 1 queues the call to the Skill group corresponding to the 1st VDN Skill (30-Agent with Texas accent who has visited resorts in Texas). If an agent with Skill 30 is available, this agent answers the call. If such an agent is not available, the *check-backup skill* command in Step 3 attempts to queue the call vis-a-vis the stated conditions (if calls-queued < 15) to the Skill group corresponding to the 2nd VDN Skill (31-Agent who has visited resorts in Texas). If Step 3 fails, the *check-backup skill* command in Step 5 attempts to queue the call vis-a-vis the stated conditions (if the oldest-call waiting < 10) to the Skill group corresponding to the 3rd VDN Skill (100-Any agent who can take a reservation).

General Number Dialing

This option allows the caller to dial the general number provided (for example, 615-3111). The caller is then serviced in part via Call Prompting.

The following figure shows how a call to VDN 3111 can be processed vis-a-vis Call Vectoring:

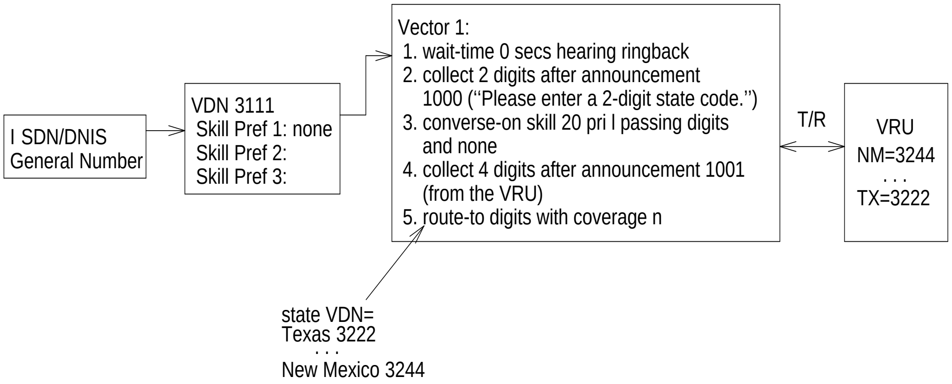

I SDN/DNIS
General Number
VDN 3111
Skill Pref 1: none
Skill Pref 2:
Skill Pref 3:
Vector 1:
1. wait-time 0 secs hearing ringback
2. collect 2 digits after announcement 1000 (“Please enter a 2-digit state code.”)
3. converse-on skill 20 pri l passing digits and none
4. collect 4 digits after announcement 1001 (from the VRU)
5. route-to digits with coverage n
T/R
VRU
NM=3244
. . .
TX=3222
state VDN=
Texas 3222
. . .
New Mexico 3244

Figure 11-13. Example 7B: Process Involving General Number Dialing

After the number is dialed, the call is directed to VDN 3111, which points to Vector 1. Note there are no Skill Preferences assigned to VDN 3111. Also, VDN 3111 is the only VDN administered to point to Vector 1. Therefore, this VDN is used for calls from all states.

The *collect digits* command in Step 2 of the previous vector first requests the caller to enter the appropriate 2-digit state code and then collects the digits. Let’s assume that the caller enters the correct code for Texas, which is ‘‘05.’’ In such a case, the *converse-on skill* command in Step 3 delivers the call to the converse Skill if there is a queue for the Skill and the queue is not full, or if a VRU port is available.

NOTE:
The *converse-on* command is discussed in Chapter 4.

When the VRU port responds, the step then outpulses the state code ‘‘05’’ to the VRU via the *passing digits* parameter included in the command. Once the VRU receives this state code, the VRU in turn outpulses the Texas VDN (3222) to the switch. Thereafter, the *collect digits* command in Step 4 collects the digits comprising this VDN. Finally, the *route-to digits* command in Step 5 routes the call to Texas VDN 3222, which points to Vector 2 (illustrated in the previous section).

Call-Back Provisions

After a caller makes a reservation for a resort site, the caller is given a call-back number. Such a number is helpful if the caller needs more information or would like to check on some arrangement that was previously made. The following figure illustrates one approach for enabling call-back provisions:

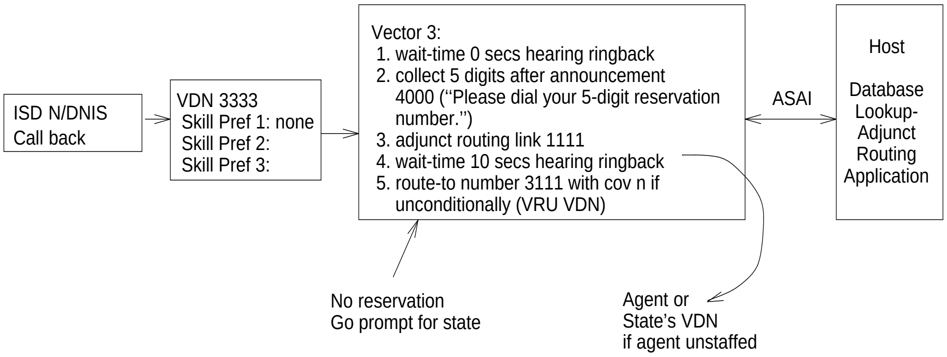

ISD N/DNIS
Call back
VDN 3333
Skill Pref 1: none
Skill Pref 2:
Skill Pref 3:
Vector 3:
1. wait-time 0 secs hearing ringback
2. collect 5 digits after announcement 4000 (“Please dial your 5-digit reservation number.”)
3. adjunct routing link 1111
4. wait-time 10 secs hearing ringback
5. route-to number 3111 with cov n if unconditionally (VRU VDN)
ASAI
Host
Database Lookup-Adjunct Routing Application
No reservation
Go prompt for state
Agent or
State’s VDN
if agent unstaffed

Figure 11-14. Example 7C: Call-Back Provisions

After the number is dialed, the call is directed to VDN 3333, which points to Vector 3. Note there are no Skill Preferences assigned to VDN 3333. Also, VDN 3333 is the only VDN administered to point to Vector 3. Therefore, this VDN is used for calls from all states.

The *collect digits* command in Step 2 of the previous vector first requests the caller to enter his or her 5-digit reservation number and then collects the digits. Once the digits are collected, the *adjunct routing* command (if successful) in Step 3 causes the switch to send the collected digits (along with other information) to the host in the ASAI adjunct routing request. The host then uses these digits to perform a database lookup for the agent who made the reservation and the resort corresponding to the reservation. If the agent is currently logged in, the call is automatically routed to the agent. Once this happens, information on the relevant reservation is displayed at the agent’s data terminal, thus providing quicker and more personal service. On the other hand, if the agent is not logged in, the call is routed to Step 5, where the *route to* command unconditionally routes the call to the VRU VDN 3111 (discussed in the "General Number Dialing" section’).

Vector Exercises

This section presents several typical business world scenarios involving telephone usage, and it shows how to write one or more vectors to handle each of these scenarios.

Note that the vectors presented here are intended to be ‘‘suggested solutions.’’ The customer should take into account his or her requirements and budget in selecting and/or writing vectors.

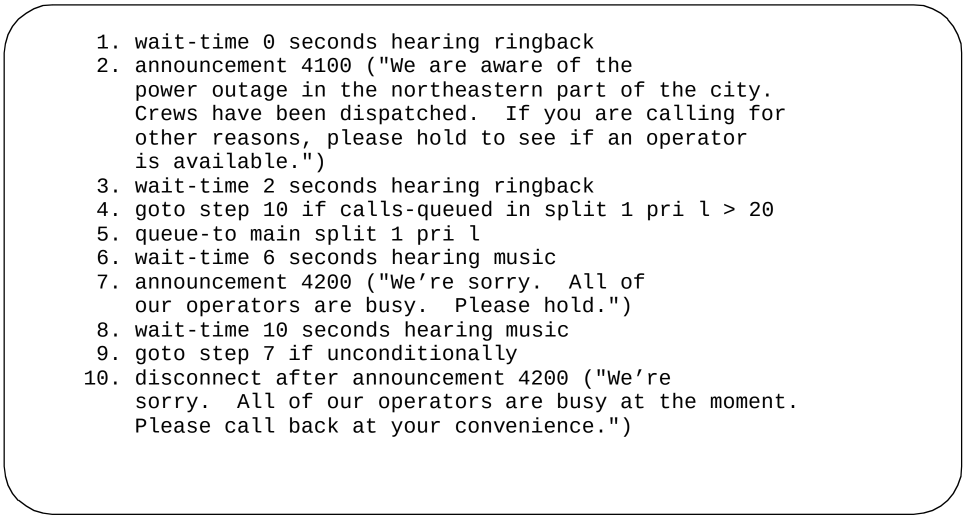
NOTE:
Exercise 1 in this section presents two solutions, one of which involves Call Prompting, which is discussed in Chapter 5.

Exercise 1: Emergency and Routine Service

Write a vector that does the following:

- Delivers the following message to handle emergency calls: "We are aware of the power outage in the northeastern part of the city. Crews have been dispatched. If you are calling for other reasons, please hold to see if an operator is available."
- Enables the caller to speak with an agent (if available) concerning a nonemergency matter.

Suggested Solution 1:



```
 1. wait-time 0 seconds hearing ringback
 2. announcement 4100 ("We are aware of the
    power outage in the northeastern part of the city.
    Crews have been dispatched.  If you are calling for
    other reasons, please hold to see if an operator
    is available.")
 3. wait-time 2 seconds hearing ringback
 4. goto step 10 if calls-queued in split 1 pri l > 20
 5. queue-to main split 1 pri l
 6. wait-time 6 seconds hearing music
 7. announcement 4200 ("We’re sorry.  All of
    our operators are busy.  Please hold.")
 8. wait-time 10 seconds hearing music
 9. goto step 7 if unconditionally
10. disconnect after announcement 4200 ("We’re
    sorry.  All of our operators are busy at the moment.
    Please call back at your convenience.")
```

Figure 11-15. Emergency and Routine Service (Call Vectoring Option)

In Step 2 of this vector, the *announcement* command provides the caller with the appropriate emergency information, and it invites the caller to hold if he or she wishes to speak with an operator on another matter. If the caller holds, the caller hears several seconds of ringback provided by the *wait-time* command in Step 3. Thereafter, the *goto step* command in Step 4 checks whether there are more than 20 calls queued in split 1. If so, a branch is made to Step 10, where the *disconnect after announcement* command first informs the caller that the call cannot be serviced at this time and then drops the call. On the other hand, if 20 or fewer calls are queued to split 1, the call is queued to the split by the *queue-to main split* command in Step 5. Thereafter, unless the call is answered, feedback in the form of music is provided by Step 6 and an announcement urging the caller to hold is provided by Step 7. After another wait with music period (if necessary) provided by Step 8, the *goto step* command in Step 9 branches back to the aforementioned “please hold” announcement in Step 7. The resulting “announcement-wait” loop (Steps 7 through 9) is then repeated until either an agent answers the call or the caller hangs up.

Suggested Solution 2:

```
VDN (extension=1030   name="Hub"   vector=30)
Vector 30:
      1. wait-time 0 seconds hearing ringback
      2. collect 1 digits after announcement 3000
         ("We are aware of the power outage in the northeastern
         part of the city.  Crews have been dispatched.  If
         you are calling for other reasons, please press 1.
         Otherwise, please hang up now.")
      3. route-to number 1031 with cov y if digit = 1
      4. announcement 3100 ("Entry not understood.  Please
         try again.")
      5. goto step 2 if unconditionally
VDN (extension=1031   name="Service"   vector=31)
Vector 31:
      1. announcement 4000 ("Please hold.  We will
         try to connect you to an operator.")
      2. wait-time 2 seconds hearing ringback
      3. goto step 9 if calls-queued in split 1 pri l > 20
      4. queue-to main split 1 pri l
      5. wait-time 6 seconds hearing music
      6. announcement 4200 ("We’re sorry.  All of
         our operators are busy.  Please hold.")
      7. wait-time 10 seconds hearing music
      8. goto step 6 if unconditionally
      9. disconnect after announcement 4200 ("We’re
         sorry.  All of our operators are busy at the moment.
         Please call back at your convenience.")
```

Figure 11-16. Emergency and Routine Service (Call Vectoring and Call Prompting Option)

Suggested Solution 2 involves both Call Vectoring and Call Prompting. Also, it involves two vectors instead of just one vector, and it assumes the caller has a touch-tone telephone.

The announcement portion of the *collect digits after announcement* command in Step 2 of Vector 30 first provides the caller with the appropriate emergency information. It then invites the caller to press ‘‘1’’ if the caller is calling for some other reason. If this is not the case, it finally suggests that the caller hang up.

First, let’s assume the caller wants to hold the line but enters the incorrect touch-tone digit (‘‘2,’’ for example). In such a case, the *route-to number* command in Step 3 attempts to route the call to VDN extension 1031 according to the entered digit. However, because a number other than ‘‘1’’ has been entered, the call is not routed to the VDN extension. Instead, control is passed to Step 4, where the *announcement* command first informs the caller of the input error and then invites the caller to try again. Thereafter, the *goto step* command in Step 5 unconditionally sends control back to Step 2, where the *collect digits* command ultimately collects the digit entered by the caller. The digit-input loop (Steps 2 through 5) continues for as long as the caller enters an incorrect digit.

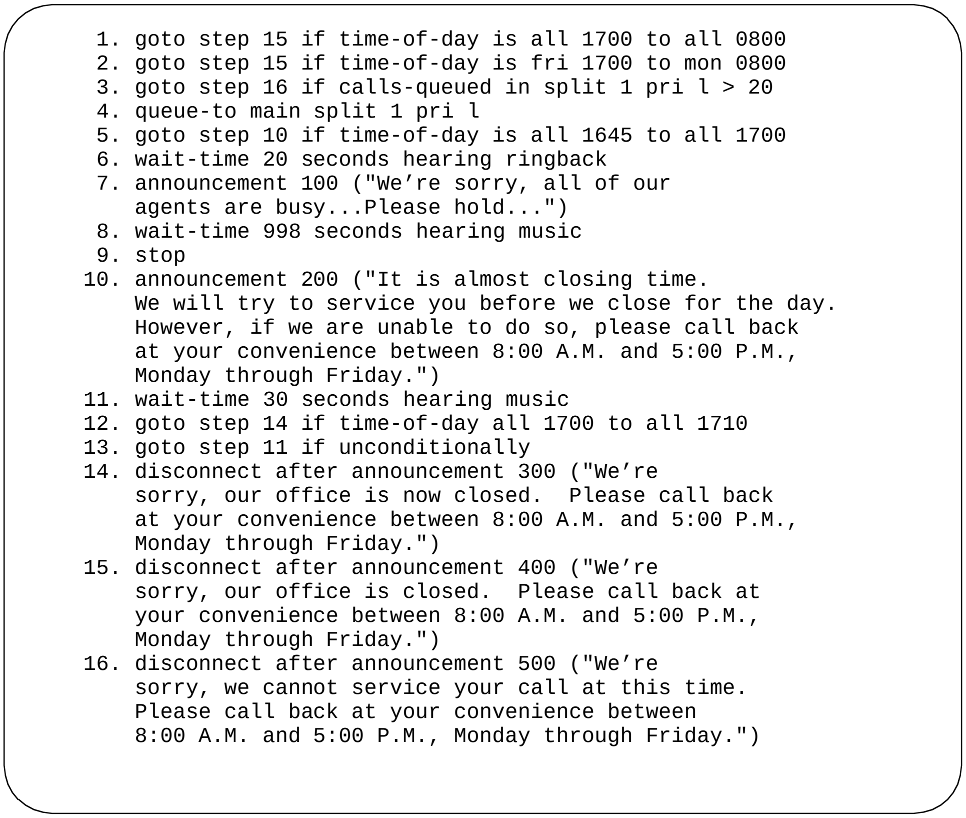
On the other hand, let's assume that the caller correctly enters the digit ''1'' as requested by the *collect digits* command in Step 2. In such a case, the *route-to number* command in Step 3 sends control to the vector whose VDN extension is ''1031'' (that is, to Vector 31). Thereafter, the call is processed almost identically to the procedure explained in Suggested Solution 1.

Exercise 2: Late Caller Treatment

Your customer's ACD is staffed by union agents. The latest union agreement stipulates that these agents are free to leave promptly at 5:00 P.M. However, you are concerned about the callers who will call shortly before 5:00 P.M. on any given day and find themselves waiting in queue (and, in effect, ignored) after the top of the hour.

Write a vector that warns late callers that their call may not be serviced. (Business hours are from 8:00 A.M. to 5:00 P.M., Monday through Friday.)

Suggested Solution:



```
 1. goto step 15 if time-of-day is all 1700 to all 0800
 2. goto step 15 if time-of-day is fri 1700 to mon 0800
 3. goto step 16 if calls-queued in split 1 pri l > 20
 4. queue-to main split 1 pri l
 5. goto step 10 if time-of-day is all 1645 to all 1700
 6. wait-time 20 seconds hearing ringback
 7. announcement 100 ("We’re sorry, all of our
    agents are busy...Please hold...")
 8. wait-time 998 seconds hearing music
 9. stop
10. announcement 200 ("It is almost closing time.
    We will try to service you before we close for the day.
    However, if we are unable to do so, please call back
    at your convenience between 8:00 A.M. and 5:00 P.M.,
    Monday through Friday.")
11. wait-time 30 seconds hearing music
12. goto step 14 if time-of-day all 1700 to all 1710
13. goto step 11 if unconditionally
14. disconnect after announcement 300 ("We’re
    sorry, our office is now closed.  Please call back
    at your convenience between 8:00 A.M. and 5:00 P.M.,
    Monday through Friday.")
15. disconnect after announcement 400 ("We’re
    sorry, our office is closed.  Please call back at
    your convenience between 8:00 A.M. and 5:00 P.M.,
    Monday through Friday.")
16. disconnect after announcement 500 ("We’re
    sorry, we cannot service your call at this time.
    Please call back at your convenience between
    8:00 A.M. and 5:00 P.M., Monday through Friday.")
```

Figure 11-17. Late Caller Treatment

This vector provides specific treatment for calls coming into the switch after working hours, during the weekend, or as the working day comes to a close.

The *goto step* command in Step 1 checks whether the call is being placed during nonworking hours during the week (that is, between 5:00 P.M. and 8:00 A.M. on any day of the week). If the call is being placed at this time, a branch is made to Step 15, where the *disconnect after announcement* command first informs the caller that the office is closed and then drops the call. On the other hand, if the call is not being placed at this time, control is passed to Step 2, where another *goto step* command checks whether the call is being placed during ‘‘weekend’’ hours (that is, between 5:00 P.M. Friday and 8:00 A.M. Monday). If so, a branch is made to Step 15, as is the case for a failure of the *goto step* command in Step 1. On the other hand, if the call is not being placed at this time, control is passed to Step 3.

The *goto step* command in Step 3 checks for the number of calls in split 1. If more than 20 calls are queued to split 1, control is passed to Step 16, where the *disconnect after announcement* command first informs the caller that the call cannot be serviced at this time and then disconnects the call. On the other hand, if 20 or fewer calls are queued to split 1, control is passed to Step 4, where the *queue-to main split command* queues the call to split 1.

Control is then passed to Step 5, where the *goto step* command checks whether the current time is any time between 4:45 P.M. and 5:00 P.M. inclusive (or, in other words, very close to [if not] closing time). If the current time does not fall within this clock range, the *wait-time* command in Step 6 provides the caller with 20 seconds of ringback. Thereafter, the *announcement* command in Step 7 plays the appropriate ‘‘hold’’ message, and the *wait* command in Step 8 provides the caller with 998 seconds of music. Finally, the *stop* command in Step 9 halts vector processing, and the call remains in queue until either the agent answers the call or the caller hangs up.

On the other hand, if the current time is any time between 4:45 P.M. and 5:00 P.M., inclusive when Step 5 is executed, a branch is made to Step 10, where the appropriate ‘‘late caller’’ announcement is provided to the caller. Thereafter, the *wait* command in Step 11 provides the caller with 30 seconds of music. Control is then passed to Step 12, where the *goto step* command checks whether the time is currently any time between 5:00 P.M. and 5:10 P.M., inclusive. If so, control is passed to Step 14, where the *disconnect after announcement* command first informs the caller that the office is now closed and then invites the caller to call back at the appropriate time before finally disconnecting the call. On the other hand, if the time is currently not between 5:00 P.M. and 5:10 P.M. inclusive, control is passed to Step 13, where the *goto step* command branches back to the *wait-time* command in Step 11. The resulting loop consisting of Steps 11 through 13 is repeated for as long as the time is between 5:00 P.M. and 5:10 P.M. inclusive, or until the caller hangs up. Once Step 12 is executed at least a second after 5:10 P.M., control is passed to Step 14 as described previously.

Exercise 3: Messaging Option

Write a vector that:

- Does the following if the oldest call waiting has been in queue longer than 75 seconds:
  - Sends the call to AUDIX (if possible)
  - Delivers to the caller the following personalized AUDIX message: "All of our MegaSports agents are busy...Please leave you name and telephone number."
- Plays for the caller 30 seconds of ringback
- Plays for the caller (after the ringback) an announcement followed by music

Suggested Solution:

```
 1. goto step 8 if oldest-call-wait in split 50 pri l > 74
 2. goto step 8 if calls-queued in split 50 pri l > 20
 3. queue-to main split 50 pri l
 4. wait-time 30 seconds hearing ringback
 5. announcement 1000 ("All of our MegaSports
    agents are busy...Please wait...")
 6. wait-time 998 seconds hearing music
 7. stop
 8. announcement 2000 ("We’re sorry, all of our
    MegaSports agents are busy.  If you’d like to leave a
    message, please do so after the tone.  Otherwise, please
    call back between 8:00 A.M. and 5:00 P.M, Monday through
    Friday. Thank you.")
 9. messaging split 20 for extension 4000
10. disconnect after announcement 2050 ("We’re sorry, we are unable
    to take your message at this time.  Please call back
    between 8:00 A.M. and 5:00 P.M., Monday through Friday.
    Thank you.")
```

Figure 11-18. Messaging Option

The *goto step* command in Step 1 of this vector checks whether the oldest call waiting in split 50 has been waiting for 75 seconds or more. If so, control is passed to Step 8, where the *announcement* command first informs the caller that all the agents are busy and then invites the caller to either call back at the appropriate time or leave a recorded message for the agent. If the caller elects to leave a message, the *messaging split* command in Step 9 is executed. Upon execution of the *messaging split* command, an attempt is made to connect the caller to AUDIX so that he or she can leave a recorded message. If the split queue is full, or if the AUDIX link is down, termination to AUDIX is unsuccessful, and vector processing continues at the next vector step, which (as is the case here) usually contains an announcement that provides the caller with the appropriate apology and subsequent directives. If the caller is successfully connected to AUDIX, vector processing terminates, and a message may be left for the specified mailbox (4000, in this case.)

Returning to Step 1, if on the other hand the oldest call waiting in split 50 has been waiting fewer that 75 seconds, control is passed to Step 2, where another *goto step* command checks for the number of calls in split 50. If more than 20 calls are queued to split 50, control is passed to Step 8. Thereafter, the procedure for the messaging option provided in the previous paragraph is implemented. On the other hand, if there are 20 or fewer calls waiting in split 50, control is passed to Step 3, where the *queue-to main split* command queues the call to the split. Thereafter, the obligatory *wait-time* and *announcement* steps (Steps 4 through 6) are executed, followed by the *stop* step (Step 7).

Call Vectoring Commands

A

Introduction

This appendix is intended to provide various information about the commands used within Call Vectoring. Specifically, the following information is presented:

- Table that contains a brief description of each command’s function and also the appendix page where the command can be referenced.
- Table that identifies the commands available in Basic Call Vectoring and/or Call Prompting.
- Job aid tables that graphically illustrate how to use the Call Vectoring commands.
- Manual page directory that details the purpose and function of the Call Vectoring commands and also any relevant interactions involving the commands.
- Table that summarizes the criteria for the success/failure of the Call Vectoring commands.

Command Description/Reference

The following table provides a brief description of the function of each of the Call Vectoring command. Also included is the page where the command is described in full.

Table A-1. Command Description/Reference Table

Command	Function	Page
adjunct routing	To request adjunct to route call.	page A-15
announcement	To connect caller to delay recording.	page A-22
busy	To connect caller to busy tone.	page A-24
check-backup split (skill)	To connect/queue call on a conditional basis.	page A-26
collect digits	To prompt caller for digits.	page A-31
converse-on split (skill)	To deliver a call to a converse split (skill) and to activate a Voice Response Unit (VRU).	page A-35
disconnect	To force disconnect of call with optional announcement.	page A-46
goto step	To cause unconditional/conditional branch to another step in the vector.	page A-48
goto vector	To cause unconditional/conditional branch to another vector.	page A-54
messaging split (skill)	To allow caller to leave message for callback.	page A-61
queue-to main split (skill)	To connect/queue call to the primary split (skill).	page A-65
route-to	To connect call to destination entered via *collect digits* command, or to connect call to internal/external destination.	page A-69
stop	To stop further vector processing.	page A-77
wait-time	To initiate feedback to caller and delay processing of the next step.	page A-79

Command/Option Summary

The following table indicates which Call Vectoring commands can be used within Basic Call Vectoring and/or Call Prompting. Other options or later releases may be required for certain commands or functions as noted. See Appendix L for more details about feature availability. Note, if EAS is enabled, “skill” replaces “split.”

Table A-2. Command/Option Summary Table

Command	Basic	Prompting	Other Options Required
adjunct routing	x		ASAI
announcement	x	x	
busy	x		
check-backup split/skill if <condition>	x		ACD
check-backup split/skill if rolling-asa	x	x	ACD; G3V4 Enhanced; G3V4 Advanced Routing
check-backup split/skill if expected-wait	x	x	ACD; G3V4 Enhanced; G3V4 Advanced Routing
check-backup split/skill if oldest-call-wait pri	x		ACD; G3V4 Enhanced
collect digits		x	
converse-on split/skill	x		
converse-on split/skill passing wait	x	x	ACD; G3V4 Enhanced; G3V4 Advanced Routing
disconnect	x		
goto step/vector if unconditionally	x	x	
goto step/vector if <condition> in split/skill	x		ACD
goto step/vector if digits		x	
goto step/vector if time-of-day	x		
goto step/vector if oldest-call-wait pri	x		ACD; G3V4 Enhanced
goto step/vector if rolling-asa	x	x	ACD; G3V4 Enhanced; G3V4 Advanced Routing
goto step/vector if expected-wait	x	x	ACD; G3V4 Enhanced; G3V4 Advanced Routing
goto step/vector if counted-calls	x	x	G3V4 Enhanced; G3V4 Advanced Routing

Continued on next page

Table A-2. Command/Option Summary Table

Command	Basic	Prompting	Other Options Required
goto step/vector if ani	x	x	G3V4 Enhanced; G3V4 ANI/II-Digits Routing
goto step/vector if ii-digits	x	x	G3V4 Enhanced; G3V4 ANI/II-Digits Routing
messaging split/skill	x	x	
messaging split/skill active/latest[1]	x	x	
queue-to main split/skill	x		ACD
route-to digits with cov y (n)		x	
route-to number if digit		x	
route-to number if unconditionally with cov y (n)[1]	x	x	
route-to number if digit with cov y (n)[1]		x	
route-to number if unconditionally	x	x	
stop	x	x	
wait-time	x	x	
wait-time secs hearing i-silent[1]	x	x	
wait-time secs hearing <extension> then <treatment>	x	x	G3V4 Enhanced

[1] If G3V4 software has not been purchased, these commands require the G3V4 maintenance load.

Command Job Aid

Table A-3. Vectoring Commands — Generic 3 Version 4

adjunct routing link ______ (extension of adjunct port)
announcement[1] ______ (1- to 5-digit extension)
busy
collect[1] ____ digits after announcement_____ (1-16) (Optional: 1- to 5-digit extension or none)
disconnect after announcement ______ (1- to 5-digit extension or none [default])
messaging split[1] _____ for extension _____ (1-24 [G3vs/s]) (1- to 5-digit extension, active [default], latest)[2] (1-99 [G3i]) (1-255 [G3r])
EAS only messaging skill[1] ______ for extension_____ (1-24 [specific: G3vs/s](1- to 5-digit extension, active [default], latest)[2]) (1-99 [specific: G3i]) (1-255 [specific: G3r]) (1st, 2nd, 3rd [VDN])

[1] This command is also available with Call Prompting.

[2] Active means the called VDN as changed by VDN override. Latest means the VDN assigned to the vector in which the call is currently being processed.

Table A-4. Vectoring Commands — Generic 3 Version 4 (continued)

queue-to main split	______ (1-24 [G3vs/s]) (1-99 {G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)					
EAS only queue-to main skill	______ (1-24 [specific: G3vs/s]) (1-99 {specific: G3i]) (1-255 [specific:G3r]) (1st, 3nd, 3rd]VDN])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)					
route-to	digits[1]	with coverage__ y or n [default]					
route-to	number	____________ (1- to 16-digit number, which may include ~p, ~s, ~w, ~W, ~m, *, or #)	with cov __ (y or n [default]	if	digit[1]	____ (<, >, <=, >=) (<>, =)	__ (0-9) (0-9, #)
					unconditionally		
stop[2]							
wait-time[3]	_____ ((0-998)[3]	secs hearing	____ (music, ringback, silence, i-silent) ____ (1- to 5-digit extension)	then	____ (continue, music, ringback,silence)		

[1] The Call Prompting feature must be enabled.

[2] This command is also available with Call Prompting.

[3] Even numbers only.

Table A-5. Vectoring Commands — Generic 3 Version 4 (continued)

check-backup split	______ (1-24 [G3vs/s]) (1-99 {G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	if	rolling-asa	<	______ (2-998)[1]
				available-agents	>	______ (0-149 [G3vs/s]) (0-199 [G3i]) (0-998 [G3r])
				calls-queued	<	______ (1-200 [G3ivs/s]) (1-999 [G3r])
				expected-wait	<	______ (1-9999)
				oldest-call-wait	<	______ (2-998)[1]
				staffed-agents	>	______ (0-149 [G3vs/s]) (0-199 [G3i]) (0-998 [G3r])
				unconditionally		

Table A-5. Vectoring Commands — Generic 3 Version 4 (continued)

EAS only						
check-backup skill	______ (1-24 [specific G3vs/s]) (1-99 {specific G3i]) (1-255 [specific G3r]) (1st, 2nd, 3rd [VDN])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	if	rolling-asa	<	______ (2-998)[2]
				available-agents	>	______ (0-149 [G3vs/s]) (0-199 [G3i]) (0-998 [G3r])
				calls-queued	<	______ (1-200 [G3ivs/s]) (1-999 [G3r])
				expected-wait	<	______ (1-9999)
				oldest-call-wait	<	______ (2-998)[1]
				staffed-agents	>	______ (0-149 [G3vs/s]) (0-199 [G3i]) (0-998 [G3r])
				unconditionally		

[1] Even numbers only.

Table A-6. Vectoring Commands — Generic 3 Version 4 (continued)

converse-on split	______ (1-24 [G3vs/s]) (1-99 {G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	passing	DATA 1[1] and (string of up to six digits or asterisks, “vdn,” “ani,” “digits,”[2] “qpos,” “wait,”[3] “#,” or “none”)	DATA 2 (string of up to six digits or asterisks, “vdn,” “ani,” “digits,”[2] “qpos,” “wait,”[3] “#,” or “none”)
EAS only converse-on skill	______ (1-24 [specific G3vs/s]) (1-99 [specific G3i]) (1-255 [specific G3r]) (1st, 2nd, 3rd [VDN])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	passing	DATA 1[1] and (string of up to six digits or asterisks, “vdn,” “ani,” “digits,”[2] “qpos,” “wait,”[3] “#,” or “none”)	DATA 2 (string of up to six digits or asterisks, “vdn,” “ani,” “digits,”[2] “qpos,” “wait,”[3] “#,” or “none”

[1] If DATA 1 is administered as “none,” DATA 2 must also be administered as “none.”

[2] The Call Prompting feature must be enabled.

[3] The Vectoring (G3V4 Advanced Routing) option must be enabled.

Table A-7. Vectoring Commands — Generic 3 Version 4 (continued)

goto step[1]	____ if (1-32)	ani[2]	_____ (<, >, <=, >=)[3] (<>, =)	_____ (1-16 characters: 0-9) (1-16 characters, 0-9, ?, +, none)[4]			
		ani[2]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])		
		rolling-asa for[5]	vdn	_____ (VDN, active, latest)[6]	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[7] (0-998)[7]	
		rolling-asa for[5]	split	_____ (1-24 [G3vs/s])[8] 1-99 [G3i]) (1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[7] (0-998)[7]	
		available- agents in split	_____ (1-24 [G3vs/s]) (1-99 [G3i]) 1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (1-150 [G3vs/s], 1-200 [G3i],, 1-999 [G3r]) (0-149 [G3vs/s],0-199 [G3i], 0-998 [G3r])		
		calls-queued in split	_____ (1-24 [G3vs/s]) (1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	_____ (<, <=) (>, >=, <>, =)	_____ (1-200 [G3vs/s/i], 1-999 {G3r]) (0-199 [G3vs/s/i], 0-998 [G3r])	
		counted-calls to vdn[5]	_____ VDN, active, latest)[6]	_____ (<, <=) (>, >=, <>, =)	_____ (1-999) (0-999)		
		digits[9]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])		
		digits[9]	_____ (<, >, <=, >=) (<>, =)	___________ (1-16 characters: 0-9) (1-16 characters: 0-9, #, ?, +, none)[4]			
		expected-wait for[5]	split	_____ (1-24 [G3vs/s]) (1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	______ (<, <=) (>, >=, <>, =)	______ (1-9999) (0-9999)

[1] The "goto vector" command is identical to the "goto step" command, except the word step is replaced by the word vector. The valid values for vector are: 1-48 [G3vs/s], 1-256 [G3i], 1-512 [G3r].

[2] The Vectoring (ANI/II-Digits Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[3] The <=, >=, <>, and = comparators are new for G3V4. <= means less than or equal to, >= means greater than or equal to, <> means not equal to, and = means equal to.

[4] A ? can be entered in any character position and matches any character in that single character position. A + must be entered as either the first or last character of the string and matches any or no characters. If + is used the maximum length of the field is 15.

[5] The Vectoring (G3V4 Advanced Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[6] Active means the called VDN as changed by VDN override. Latest means the VDN assigned to the vector in which the call is currently being processed.

7. Even numbers only.
8. Active means the called VDN as changed by VDN override. Latest means the VDN assigned to the vector in which the call is currently being processed.
9. The Call Prompting feature must be enabled.

Table A-8. Vectoring Commands — Generic 3 Version 4 (continued)

goto step[1]	____ if (1-32)	expected-wait[2] call for		_____ (<, <=)[3] (>, >=, <>, =)	_____ (1-9999) (0-9999)	
		ii-digits[4]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])	
		ii-digits[4]	_____ (<, >, <=, >=) (<>, =)	_____ (00-99) [00-99, +, +?, ?+, ??, ?x (where x = 0-9), x?, none][5]		
		oldest-call-wait in split	______ (1-24 [G3vs/s]) 1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[6] (0-998)[6]
		staffed-agents in split	_____ (1-24 [G3vs/s]) (1-99 [G3i]) 1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (1-150 [G3vs/s], 1-200 [G3i],, 1-999 [G3r]) (0-149 [G3vs/s], 0-199 [G3i], 0-998 [G3r])	
		time-of-day is	_____ (mon, tue, wed, thu, fri, sat, sun, all)	_____ to (hh:mm in military time)	_____ (mon, tue, wed, thu, fri, sat, sun, all)	_____ (hh:mm in military time)
		unconditionally[7]				

[1] The “goto vector” command is identical to the “goto step” command, except the word step is replaced by the word vector. The valid values for vector are: 1-48 [G3vs/s], 1-256 [G3i], 1-512 {G3r].

[2] The Vectoring (G3V4 Advanced Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[3] The <=, >=, <>, and = comparators are new for G3V4. <= means less than or equal to, >= means greater than or equal to, <> means not equal to, and = means equal to.

[4] The Vectoring (ANI/II-Digits Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[5] A ? matches any digit 0-9 in that single digit position. A + matches any or no characters.

[6] Even numbers only.

[7] This command is also available with Call Prompting.

Table A-9. Vectoring Commands — Generic 3 Version 4 (continued)

goto step[1]	____ if (1-32)	ani[2]	_____ (<, >, <=, >=)[3] (<>, =)	_____ (1-16 characters: 0-9) (1-16 characters, 0-9, ?, +, none)[4]		
		ani[2]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])	
		rolling-asa for[5]	vdn	_____ (VDN, active, latest)[6]	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[7] (0-998)[7]
		rolling-asa for[5]	skill	_____ (1-24 [G3vs/s]) 1-99 [G3i]) (1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[7] (0-998)[7]
		available- agents in skill	_____ (1-24 [G3vs/s]) (1-99 [G3i]) 1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (1-150 [G3vs/s], 1-200 [G3i],, 1-999 [G3r]) (0-149 [G3vs/s],0-199 [G3i], 0-998 [G3r])	
		calls-queued in skill	_____ (1-24 [G3vs/s]) (1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	_____ (<, <=) (>, >=, <>, =)	_____ (1-200 [G3vs/s/i], 1-999 {G3r]) (0-199 [G3vs/s/i], 0-998 [G3r])
		counted-calls to vdn[5]	_____ VDN, active, latest)[6]	_____ (<, <=) (>, >=, <>, =)	_____ (1-999) (0-999)	
		digits[8]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])	
		digits[9]	_____ (<, >, <=, >=) (<>, =)	___________ (1-16 characters: 0-9) (1-16 characters: 0-9, #, ?, +, none)[4]		

[1] The “goto vector” command is identical to the “goto step” command, except the word step is replaced by the word vector. The valid values for vector are: 1-48 [G3vs/s], 1-256 [G3i], 1-512 {G3r].

[2] The Vectoring (ANI/II-Digits Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[3] The <=, >=, <>, and = comparators are new for G3V4. <= means less than or equal to, >= means greater than or equal to, <> means not equal to, and = means equal to.

[4] A ? can be entered in any character position and matches any character in that single character position. A + must be entered as either the first or last character of the string and matches any or no characters. If + is used the maximum length of the field is 15.

[5] The Vectoring (G3V4 Advanced Routing) and Vectoring (G3V4 Enhanced) options must be enabled.

[6] Active means the called VDN as changed by VDN override. Latest means the VDN assigned to the vector in which the call is currently being processed.

[7] Even numbers only.

[8] The Call Prompting feature must be enabled.

Table A-10. Vectoring Commands — Generic 3 Version 4 (continued)

goto step[1]	____ if (1-32)	expected-wait for[5]	skill	_____ (1-24 [G3vs/s]) (1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	______ (<, <=) (<, <=, <>, =)	______ (1-9999) (0-9999)
		expected-wait[2] for	call	_____ (<, <=)[3] (>, >=, <>, =)	_____ (1-9999) (0-9999)		
		ii-digits[4]	_____ (in, not-in)	table	_____ (1-5 [G3vs/s]) (1-10 [G3i]) (1-100 [G3r])		
		ii-digits[4]	_____ (<, >, <=, >=) (<>, =)	_____ (00-99) [00-99, +, +?, ?+, ??, ?x (where x = 0-9), x?, none][5]			
		oldest-call-wait in skill	_____ (1-24 [G3vs/s]) 1-99 [G3i]) (1-255 [G3r])	pri __ (**l**ow, **m**ed, **h**igh, **t**op)	_____ (<, <=) (>, >=, <>, =)	_____ (2-998)[6] (0-998)[6]	
		staffed-agents in skill	_____ (1-24 [G3vs/s]) (1-99 [G3i]) 1-255 [G3r])	_____ (<, <=) (>, >=, <>, =)	_____ (1-150 [G3vs/s], 1-200 [G3i],, 1-999 [G3r]) (0-149 [G3vs/s],0-199 [G3i], 0-998 [G3r])		
		time-of-day is	_____ (mon, tue, wed, thu, fri, sat, sun, all)	_____ to (hh:mm in military time)	_____ (mon, tue, wed, thu, fri, sat, sun, all)	_____ (hh:mm in military time)	
		unconditionally[7]					

1. The “goto vector” command is identical to the “goto step” command, except the word step is replaced by the word vector. The valid values for vector are: 1-48 [G3vs/s], 1-256 [G3i], 1-512 {G3r].
2. The Vectoring (G3V4 Advanced Routing) and Vectoring (G3V4 Enhanced) options must be enabled.
3. The <=, >=, <>, and = comparators are new for G3V4. <= means less than or equal to, >= means greater than or equal to, <> means not equal to, and = means equal to.
4. The Vectoring (ANI/II-Digits Routing) and Vectoring (G3V4 Enhanced) options must be enabled.
5. A ? matches any digit 0-9 in that single digit position. A + matches any or no characters.
6. Even numbers only.
7. This command is also available with Call Prompting.

Command Directory

The manual page directory in this section lists and discusses all of the commands used within Call Vectoring. For each command presented, the following is provided: purpose, syntax, valid entries, requirements, an example, description of the command's operation, answer supervision considerations, feature interactions, and CMS/BCMS interactions. The following points concerning the appearance of the command line are in effect:

- Data that must be entered as part of the command line is shown in **bold**.
- Variable fields that (in most cases) must be completed are enclosed in < >.
- Optional fields are enclosed in [ ].

NOTE:
If a variable field appears within an optional field, an entry for the variable field appears only if the optional field is included during command execution.

NOTE:
If EAS is enabled, ''skill'' replaces "split."

Adjunct Routing Command

```
adjunct routing
```

Purpose

Causes a message to be sent to an adjunct requesting routing instructions.

Syntax

adjunct routing link <extension>

Valid Entries

Valid ASAI-BRI link extension number

Requirements

Basic Call Vectoring and ASAI software must be installed. Also, an ISDN-BRI port is required, and the port must be connected to an ASAI host.

Example

adjunct routing link *765*

Operation

The *adjunct routing* command provides a means for an adjunct ASAI processor to specify the destination of a call. The switch provides information in an ASAI route request message that the ASAI adjunct can use to first access a data base and then determine a route for the call. In a typical application, the ASAI adjunct might use the dialed number, the calling party number (CPN/BN), or the digits collected via Call Prompting to access customer information and thereby determine the call route. A maximum of 16 digits collected from the last *collect digits* command can be passed.

An adjunct specified in an *adjunct routing* command can route a call to an internal number, an external number, a split, a VDN, an announcement extension, or a particular agent. An adjunct can also provide priority ringing and priority queuing.

When a call encounters an *adjunct routing* command, the switch sends to the specified adjunct an ASAI message requesting a call route. The following list identifies the contents of the message, along with a comment or a brief explanation for each item:

- **Calling number information**. DNIS number provided by ISDN-PRI facilities, or by the administered destination for other trunk facilities. If the call originates from a local switch extension, this extension is the number dialed (after digit translation).
- **Called number**. Originally called extension (if a call is forwarded to a VDN), or the first VDN through which the call was routed (if the call was not forwarded to the VDN).
- **Routing VDN**. Last VDN that routed the call to the vector that contains the *adjunct routing* command.
- **Call identifier**. ASAI identifier that permits the ASAI adjunct to track multiple calls via either Event Notification or Third Party Call Control. (See *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204 or *Generic 3 CallVisor ASAI Technical Reference* for more information on ASAI.)
- **Look-Ahead Interflow information** (if any). Includes the original VDN display information and the priority level of the call at the originating switch. (See Chapter 8.)
- **Digits collected via Call Prompting** (if any). Digits are collected by the most recent *collect digits* command. (See Chapter 5.)

If the call is queued, the *adjunct routing* step is ignored, and vector processing continues at the next vector step.

If the ASAI link specified in the *adjunct routing* step is down, the step is skipped.

An ASAI link failure can change the manner in which subsequent treatment (that is, *announcement* and/or *wait-time*) steps (if any) in the vector are usually processed. In some cases, such processing is influenced by the position that the treatment steps occupy in the vector. In other cases, the positioning of these commands along with their relationship to specific *goto* commands come into play. For example, any *announcement* or *wait-time* step that immediately follows an *adjunct routing* step whose ASAI link is down is skipped.

NOTE:

The second step after the *adjunct routing* step is often implemented as a default treatment (for example, a route-to an attendant). If the ASAI link is down, the default step executes immediately. Otherwise, the step executes only if the application does not respond with a route within the time period specified by the *wait-time* step.

On the other hand, if a *goto* step follows such an *adjunct routing* step, the switch executes the *goto* step and then skips various treatment steps according to their position in the vector and based on the performance of the *goto* step. Specifically, if the *goto* step *succeeds*, the switch skips any *announcement* or *wait-time* step that is the first non-*goto* step branched to by the *goto* step.

NOTE:
Actually, other than to another *goto* step, the first step to which a *goto* step is usually designed to branch is a nontreatment step (that is, a step containing a command other than a *wait-time* or an *announcement* command). Thus, the skipping of a treatment step according to the scenario described just before this note rarely occurs.

On the other hand, if the *goto* step fails, the switch skips any *announcement* or *wait-time* step that *immediately* follows the *goto* step.

NOTE:
The *goto* step that fails can be at the end of a sequence of *goto* steps that branch to each other.

After the switch sends a route request to the ASAI adjunct, vector processing continues with the vector steps that follow.

The step that follows the *adjunct routing* step, in effect, determines the *maximum* length of time the switch will wait for the ASAI adjunct to reply with a call route. Accordingly, you should always include either a *wait-time* step or an *announcement* step immediately after an *adjunct routing* step. Moreover, the switch cancels the route request if vector processing encounters a step containing one of the commands that follow:

- *adjunct routing*
- *busy*
- *check-backup split*
- *collect digits*
- *converse-on split*
- *disconnect*
- *messaging split*
- *queue-to main split*
- *route-to*

NOTE:
Actually, if another *adjunct routing* step is encountered, the route request information is not lost. Although the initial route request is cancelled, a second route request is sent, and this route request includes the same information included in the first route request.

If a valid call route is received by the switch before one of the vector commands in the previous list is executed, the switch routes the call to the destination specified by the adjunct route. Otherwise, the route request is terminated without affecting vector processing.

Finally, note that the adjunct can also decide to not route a call by rejecting (that is, negatively acknowledging) the route request sent by the switch. Upon receiving a route request rejection, the switch terminates the *announcement* or *wait-time* step that is being executed for the call and then continues with the next vector step.

When the switch receives a call route (destination) from the ASAI adjunct, the switch first validates the route as follows:

1. The switch verifies that the VDN's COR permits the call to be terminated at the adjunct-supplied destination.
2. The switch verifies that the adjunct-supplied information (destination number, ACD split, TAC/AAR/ARS access code, etc.) for the route is valid. This includes checking that the destination is compatible with the dial plan, and that the options specified by the adjunct are correct.
3. If the ASAI adjunct specifies the Direct Agent Call (DAC) option, the destination number (agent) must be logged into the adjunct-specified ACD split.
4. If the destination for the call is external, the switch verifies the trunk is available for the call.

If any of these conditions are not met, the route validation fails, and the switch does the following:

1. Discards the route.
2. Notifies the ASAI adjunct that the route is invalid.
3. Continues with vector processing.

If the route is valid, the switch does the following:

1. Terminates vector processing immediately.
2. Notifies the ASAI adjunct that the route is accepted.
3. Routes the call to the destination specified by the ASAI adjunct.

When the call is routed, the caller hears normal call progress tones and feedback. However, if the call is routed to an extension with no available call appearances and no coverage path, the caller hears the busy tone. Any other features that may be in effect at the adjunct-supplied destination (such as Send-All-Calls or Call Forwarding) interact with the routed call.

NOTE:
The operation described in the previous paragraph is similar to that for the *route-to with coverage* commands.

Answer Supervision Considerations

The command has no bearing on answer supervision.

Feature Interactions

For a call coming in directly to a VDN, the command is treated like a *route-to with coverage=y* command. However, for a call that is covered to a VDN, the command is treated like a *route-to with coverage=n* command. A covered call that is routed by an *adjunct routing* command to a destination that has Call Forwarding activated is not further redirected (since the call has already been redirected by coverage).

For Look-Ahead Interflow, the command is considered a neutral vector command in all cases. However, the command is usually followed by an *announcement* or *wait-time* command, each of which is a call acceptance command. The G3V4 *wait-time hearing i-silent* command can be used when a neutral *wait-time* command is required to allow the adjunct to accept or reject the call.

If an *announcement* command follows a failed *adjunct routing* command, the announcement is interrupted. If the *adjunct routing* command succeeds (that is, the switch receives a destination from the ASAI adjunct), the announcement terminates immediately.

CMS Interactions

R3 CMS: Adjunct routing attempts are stored in the ADJATTEMPTS database item and reported as ‘‘Adjunct Routing Attempts’’ in standard reports. If the call is queued to a split when the *adjunct routing* command is encountered, the step is skipped, and no messages are sent to CMS. Accordingly, ‘‘Adjunct Routing Attempts’’ is not reported for this call.

When a routing response from the adjunct is successfully executed by the switch, this action is tracked in the ADJROUTED and ADJROUTTIME database items and shown as ‘‘Adjunct Routing Completions’’ in standard reports.

Additional tracking of the ‘‘Adjunct Routing’’ command varies based on the destination successfully routed to as follows:

Routed to Station or to Attendant

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out	
INTIME	Avg Time In Vector	
CONNECTCALLS/ CONNECTTIME	Other Calls Connect	answered calls on G3

Routed to Trunk

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out VDN Flow Out	
INTERFLOWCALLS / INTERFLOWTIME	VDN Flow-Interflow	
INTIME	Avg Time In Vector	

Routed to VDN

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Vector Flow Out VDN Flow Out	
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In VDN Flow In	new vector new VDN

Routed to Split or to Hunt Group

Database Item	Report Heading	Notes
CALLSOFFERRED		new split
LOWCALLS/MEDCALLS		no priority/priority

Split calls are also shown in the standard reports based on the final disposition of the call.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ‘‘ACD Calls,’’ ‘‘Split ACD Calls,’’ and ‘‘Avg Speed Ans.’’

A call abandoned after the command routes the call to a station or an attendant is tracked in the VDN tables as ABNCALLS/ABNTIME.

R2 CMS: Adjunct routing attempts are not stored in R2 CMS. Successful adjunct-routed calls are stored in OUTFLOW and FLOWTIME in the Vector and VDN tables. These calls are not shown on standard reports. VECTIME is recorded for the vector and reported as ‘‘Avg Time in Vector.’’

Calls that route to a split are tracked as INFLOW and are reported as ‘‘Flow In.’’ Calls that route to a trunk are tracked as ROUTEDCALLS/ROUTEDTIME. However, these calls are not shown on standard reports.

BCMS Interactions

If the command advances a call to another position (that is, ASAI routing is successful), the call is tracked as “outflow” in the VDN Report.

Announcement Command

announcement

Purpose

Provides the caller with a recorded announcement.

Syntax

announcement <extension>

Valid Entries

Valid announcement extension number

Requirements

Basic Call Vectoring or Call Prompting software must be installed. Also, integrated board, aux trunk or analog (T&R or Lineside DS1) announcement equipment must be installed. Finally, the announcements themselves need to be administered and recorded. See “Recorded Announcements” in the *DEFINITY Communications System Generic 3 Feature Description,* 555-230-204, for more information.

Example

announcement *2982*

Operation

The announcement is played from beginning to end unless an agent becomes available. In such a case, the announcement is interrupted and (if manual answering operation is assigned to the agent, or if calls are delivered to the agent on a manual answering basis) ringback is provided. If the call is queued, the call remains as such while the announcement is played. Any feedback provided before an announcement is continued until the announcement is played.

If an *announcement* command follows a failed *adjunct routing* command, the announcement is interrupted. If the *adjunct routing* command succeeds (that is, the switch receives a destination from the ASAI adjunct), the announcement terminates immediately.

If the announcement's queue is currently full, the call retries the announcement step every five seconds and for an indefinite period of time before any new vector steps are processed.

The *announcement* command step is skipped, and vector processing continues at the next vector step, whenever any of the following conditions exist:

- Requested announcement is busied out, not available, or not administered.
- Integrated board is not installed.
- External aux trunk or analog equipment is not attached.

For a complete description of the types and operation of announcements see "Recorded Announcements" in the *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204.

Answer Supervision Considerations

Unless answer supervision has already been sent, it is sent as soon as the command starts to process the call (even before the announcement starts).

Feature Interactions

For Look-Ahead Interflow, the command may be considered a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a neutral vector command whenever the announcement is unavailable.

BCMS/CMS Interactions

The command is not tracked on CMS or BCMS.

Busy Command

```
busy
```

Purpose

Gives the caller a busy signal and causes termination of vector processing.

Syntax

busy

Requirements

Basic Call Vectoring software must be installed.

Operation

The command takes effect on non-CO trunk calls whether or not answer supervision has been sent. However, if the call is on a CO trunk and answer supervision has not been sent, the busy is not passed back by the CO, and the caller continues to hear ringback from the CO. Calls are dropped approximately 45 seconds after the busy tone is applied.

If ISDN-PRI is involved, the application of the busy tone is enabled via D-channel messaging. The network switching office returns the busy tone to the caller. The facility to the PBX/ACD is dropped, thus making it immediately available for another call.

Answer Supervision Considerations

After the 45 second timeout, an unanswered CO trunk call is answered and then dropped. All other unanswered calls after this timeout are dropped without being answered. For an ISDN call that has not yet queued or been answered, no timeout occurs, and answer supervision is not sent. Instead, a message requesting a busy tone is sent to the network and, subsequently, the trunk is released.

Feature Interactions

For Look-Ahead Interflow, the command is considered a call denial vector command in all cases.

CMS Interactions

R3 CMS:

Busy Command

Database Item	Report Heading
BUSYCALLS/BUSYTIME	Calls Forced Busy Calls Busy/Disc
OTHERCALLS/OTHERTIME	Inbound Other Calls
INTIME	Avg Time In Vector

BUSYTIME, OTHERTIME, and INTIME for splits and vectors are tracked according to when the busy tone starts. BUSYTIME, OTHERTIME and INTIME for VDNs are tracked according to when the trunk idles.

R2 CMS: Calls given forced busy are tracked in the FBUSYCALLS and FBUSYTIME database items. These calls are not shown in standard reports. FBUSYTIME is tracked according to when the busy tone starts. VECTIME is recorded for the vector, and it is reported as ‘‘Avg Time in Vector.’’

BCMS Interactions

A call that is forced busy due to the command is tracked as ‘‘OTHER’’ in the VDN Report.

Check-Backup Command

check-backup split (skill)

Purpose

Checks the status of a split (skill) for possible termination of the call to that split (skill).

Syntax

check-backup split <split #> pri <priority level> if <condition> [<comparator> <threshold>]

check-backup skill <skill #> pri <priority level> if <condition> [<comparator> <threshold>]

Valid Entries

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs); *1st, 2nd, 3rd* (VDN)

priority level: *l* (low), *m* (medium), *h* (high), *t* (top).

condition:	comparator:	threshold:
unconditionally	N/A	N/A
rolling-asa	<	2-998 (even numbers only)
available-agents	>	*0-199* (G3i), *0-998* (G3r) *0-149* (G3s, G3vs)
calls-queued	<	*1-200* (G3i, G3s, G3vs), *1-999* (G3r)
expected-wait	<	1-9999
oldest-call-wait	<	*2-998* (even numbers only)
staffed-agents	>	*0-199* (G3i), *0-998* (G3r) *0-149* (G3s, G3vs)

Requirements

Basic Call Vectoring software must be installed, and the split (skill) involved must be vector-controlled. Also, skills replace splits only if Expert Agent Selection (EAS) is optioned. The conditions “rolling-asa” and “expected-wait” are only available if the “Vectoring (G3V4 Advanced Routing)” customer option is enabled.

Examples

check-backup split *22* pri *h* if **unconditionally**

check-backup split *11* pri *l* if **available-agents** > *5*

check-backup split *11* pri *t* if **calls-queued** < *5*

check-backup skill *2nd* pri *m* if **staffed-agents** > *5*

check-backup skill *25* pri *l* if **oldest-call-wait** < *60*

check-backup skill *12* pri *l* if **rolling-asa** < *50*

check-backup skill *10* pri *m* if **expected-wait** < *20*

Operation

The *check-backup split (skill)* command checks the status of a split (skill) for possible termination of the call to that split (skill). When termination is not possible, queuing at the specified priority is attempted. Termination and/or queuing are attempted if the split (skill) meets certain conditions specified as part of the command.

The command is customized to check for and/or respond to specific conditions. For example, the command can queue/terminate unconditionally. The command can also queue/terminate if any of the following is true:

- Number of available agents is *greater than* the threshold value.
- Number of staffed agents is *greater than* the threshold value.
- Number of calls queued for a specified priority level or higher is *less than* the threshold value.
- Oldest call waiting in queue at the specified priority level or higher has been waiting *less than* the threshold value, which is expressed in seconds.
- Rolling average speed of answer is *less than* the threshold value, which is expressed in seconds.

- Expected wait time is *less than* the threshold value, which is expressed in seconds.

A call may be queued to up to three splits (skills) simultaneously. A call remains queued either until vector processing terminates (via a successful *disconnect, busy*, or *route-to* command, or via an abandoned call) or until the call reaches an agent. When an agent becomes available in any split (skill) to which the call is queued, the following actions take place:

- Call begins alerting the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the desired backup split (skills) is one of the splits (skills) to which the call is already queued, the call is requeued at the new priority level, provided that the command conditions are met. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Command conditions are not met.
- Desired split's (skill's) queue is full.
- Desired split (skill) has no queue and also no available agents.
- Desired split (skill) is not vector-controlled.
- Call is already queued to this split (skill) at the specified priority level.
- Call has been previously queued to three different splits (skills).

NOTE:
A *route-to* to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

Answer Supervision Considerations

No answer supervision is returned.

Feature Interactions

The *check-backup split (skill)* command can access a AUDIX/Message Center/Server split (skill) in cases where a VDN is assigned as a coverage point. To enable this function, the split (skill) must be assigned as a vector-controlled hunt group.

For Look-Ahead Interflow, the command can be considered either a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent.
- Call queues to a split (skill).

The call is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a *check-backup* step places a call to a split or skill.

CMS Interactions

Calls answered via the check-backup command are indicated as answered by backup in CMS.

R3 CMS: Calls queued via a *check-backup split (skill)* command are tracked as CALLSOFFERRED and LOWCALLS/MEDCALLS/HIGHCALLS/TOPCALLS.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as “ACD Calls,” “Split ACD Calls,” and “Avg Speed Ans.” If the call is also queued to other splits (skills), OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split (skill) to which the call queues, and “Flow Out” is reported (unless the split (skill) turns out to be the answering split (skill)). DEQUECALLS/DEQUETIME is tracked in the second and third splits (skills) if these splits (skills) are not the answering split (skill), and the call is reported as “Dequeued Calls” and ‘‘Dequeued Avg Queue Time.’’ However, if the second or third split (skill) is the answering split (skill), INFLOWCALLS is tracked in the split (skill), and the call is reported as “Flow In.”

Whenever the call is answered in a split (skill) accessed by the *check-backup split* command, the BACKUPCALLS data base item is incremented, and the call is reported as ‘‘Calls Ans in Backup’’ and ‘‘Calls Handled/Backup.’’ The ‘‘Calls Ans in Main’’ report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

If the call abandons after the command queues the call to a split (skill), ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split (skill) to which the call is queued. The call is reported as ‘‘Aban Call’’ and ‘‘Avg Aban Time.’’ If the call is also queued to other splits (skills), DEQUECALLS/DEQUETIME is tracked in these splits (skills), and the call is reported as ‘‘Dequeued Calls’’ and ‘‘Dequeued Avg Queue Time.’’ **R2 CMS**: An ACD call that is directed via the command and that is subsequently answered is tracked as ANSWERED/ANSDELAY and ACDCALLS. The call is reported as ‘‘No. ACD Calls,’’ ‘‘No. Calls Ans,’’ and ‘‘Avg Speed Ans.’’ If the call is also queued to other splits (skills), OUTFLOW is tracked in these splits (skills), and the call is reported as ‘‘Flow Out.’’ Calls directed via the command are tracked as ANSBACK, and they are reported as ‘‘No. Calls Ans in Backup.’’

If the call abandons after the command queues the call to a split (skill), ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split (skill) to which the call is currently queued. The call is reported as ‘‘No. Aban Calls’’ and ‘‘Avg Aban Time.’’ If the call is also queued to other splits (skills), OUTFLOWCALLS/OUTFLOWTIME is tracked in these splits (skills), and the call is reported as ‘‘Flow Out.’’

BCMS Interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ‘‘NUM ANS’’ in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as ‘‘AVG TIME TO CONNECT’’ in the same report.

Collect Digits Command

```
collect digits
```

Purpose

Allows the user to enter up to 16 digits from a touch-tone phone.

Syntax

collect <# of digits> digits after announcement <extension>

Valid Entries

# of digits: 1 through 16.

extension: *none* or valid announcement extension.

Requirements

Call Prompting software must be installed. Also, at least one TN744 Call Classifier circuit pack must be in the system unless the command is used only to collect digits returned by a VRU that is accessed by the command and never to collect digits from a caller.

Example

collect *12* digits after announcement *2982*

Operation

This command, which is part of the Call Prompting feature, allows the caller to enter digits from a touch-tone or an internal rotary phone. An optional announcement may be used to request the caller to enter these digits. The announcement can instruct the user to enter an asterisk (*) if incorrect data is entered. When the caller enters an asterisk, the digits collected for the current *collect digits* command are deleted, digit collection is restarted, and the announcement is not replayed.

In using this command, the *maximum* number of digits requested of the caller must be specified in the administration of the command. If the caller can enter fewer digits than the maximum specified, the announcement should instruct the caller to terminate the entry with a pound sign (#) digit as an end-of-dialing indicator. If all the digits strings for all the variations of a specific *collect digits* command are terminated with "#," the "#" must be counted as one of the digits. Therefore, the number of digits collected should include any "#" that needs to be collected. Otherwise, the terminating ‘‘#’’ is kept as a dial-ahead digit and is processed by a subsequent *collect digits* command. If fewer digits than the maximum specified are entered, and if the caller does not complete the entry with a pound sign, an interdigit timeout occurs. The timeout terminates the command, and any digits collected prior to the timeout are available for subsequent vector processing.

Generally, processing of the command requires that a TTR be connected. (If the call originates from an internal rotary phone, no TTR is needed.) TTRs accept the touch-tone digits that are entered by Call Prompting users. TTRs are automatically connected as needed by the system.

The connection of the announcement prompt is skipped and the digit collection phase begins whenever one of the following conditions is true:

- Dial-ahead digits exist.
- No announcement is administered for the *collect digits* step.
- Announcement administered for the *collect digits* step does not exist.

Otherwise, an attempt is made to connect the administered announcement. If the announcement to be connected is busy, and if the queue for the announcement is full, or if there is no queue, the calling party continues to hear the current feedback. The system waits five seconds and then tries again to connect the call to the announcement. This process continues until the call is successfully queued or connected to the announcement, or until the calling party disconnects from the call. If the queue for the announcement is not full, the call is queued for the announcement.

If the announcement to be connected is available (either initially or after queuing, or after system retry), any previous feedback is disconnected, and the calling party is connected to the announcement.

While the announcement is playing, or while the call is being queued for an announcement, the caller may enter digits at any time. This causes the announcement to be disconnected or removed from the queue, as appropriate, and the digit collection phase to begin. If the caller does not enter any digits during the announcement phases, the digit collection phase begins when the announcement completes.

As soon as the digit collection phase begins, interdigit timing is started, unless the TTR is already in timing mode (that is, the dial-ahead capability is active and the TTR is not disconnected).

Digits are *collected* either as digits dialed during the *collect digits* command or as dial-ahead digits dialed since a previous *collect digits* command but prior to the current appearance of the command. Digit collection continues for the current command until one of the following conditions exists:

- Number of digits specified is collected.

- Pound sign (#) digit is collected (signifying end of dialing).
- Inter-digit timer expires.

If, during the digit collection phase, a ‘‘*’’ is encountered within a stream of dialed or dial-ahead digits, all digits that are collected for the current *collect digits* step are discarded. If additional dial-ahead digits occur after the asterisk, these digits continue to be processed. If there are no such digits, and if no TTR is connected, vectoring continues at the next vector step. If a TTR is connected, the caller can start entering digits again. In such a case, the announcement is not replayed, and the interdigit timer is restarted.

NOTE:
If an asterisk is entered after the requested number of digits are entered, the asterisk has no effect on the previously entered digits. However, in such a case, the asterisk is treated as a dial-ahead digit for the next *collect digits* command.

When digit collection is completed, and if a TTR is connected (for a touch-tone phone), the interdigit timer is restarted to detect a timeout for releasing the TTR. Vector processing then continues at the next vector step. However, the switch continues to collect any subsequent dialed digits [(including the pound sign (#) and asterisk (*) digits)] to allow for the dial-ahead capability. These additional ‘‘dialed ahead’’ digits are saved for use by subsequent *collect digits* commands, and they provide the caller with a means to bypass subsequent unwanted announcement prompts. A single "#" digit can be collected and tested by subsequent *route-to...if digits* or *goto...if digits* commands. Alternately, the digit can be passed to a host via ASAI. Collection of dial-ahead digits continues until one of the following occurs:

- Vector processing stops or is terminated.
- The sum of the digits collected for the current *collect digits* command and the dial-ahead digits exceeds the switch storage limit of 24. Any additional dialed digits are discarded until storage is freed up by a subsequent *collect digits* command.

  NOTE:
  Any asterisk (*) or pound sign (#) digits count towards the 24-digit limit, as do any dial-ahead digits entered after the asterisk or pound sign digit.

- The TTR required by the touch-tone phone user to collect digits is disconnected. This occurs under the following conditions:
  - Successful or unsuccessful *route-to number* step is encountered during vector processing *except* where the number routed to is a VDN extension.
  - Successful or unsuccessful *route-to digits* step is encountered during vector processing *except* where the number routed to is a VDN extension.

— Successful or unsuccessful *adjunct routing* step is encountered during vector processing.

— Successful or unsuccessful *converse-on* step is encountered during vector processing.

— 10 second timeout occurs, during which time the caller does not dial any digits, asterisks (*) or pound signs (#).

NOTE:
When the TTR is disconnected due to a *route-to number, route-to digits, converse-on*, or an *adjunct routing* step, *all dial-ahead digits are discarded*. This means that, following a failed *route-to, converse-on* or *adjunct routing* step, a subsequent *collect digits* step always requires the caller to enter digits.

NOTE:
Dial-ahead digits are available for use *only* by subsequent *collect digits* commands. The digits are *never* used by other vector commands that operate on digits (for example, *route-to digits*, *goto...if digits*, etc). In addition, these digits are not displayed as part of the CALLR-INFO button operation until they are collected via a *collect digits* command.

Answer Supervision Considerations

Answer supervision is provided as soon as a TTR is connected and processing of the command starts. The command always provides answer supervision to an incoming trunk if supervision has not been previously provided.

Feature Interactions

For Look-Ahead Interflow, the command is considered a call acceptance vector command in all cases.

CMS Interactions

The command is not tracked on CMS.

Converse-on Command

```
converse-on split (skill)
```

Purpose

Delivers a call to a converse split (skill) and activates a voice response script that is housed within a Voice Response Unit (VRU).

Syntax

converse-on split <split #> pri <priority level> passing <data_1> and <data_2>

converse-on skill <skill #> pri <priority level> passing <data_1> and <data_2>

Valid Entries

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs); *1st, 2nd, 3rd* (VDN)

priority level: *l* (low), *m* (medium), *h* (high), *t* (top).

data_1, data_2: String consisting of any digits *0-9* and/or of one or more asterisks *(*)* and not exceeding a total of six such digits and/or asterisks; the pound sign *(#)*; any of the following keywords: *vdn, ani, digits, qpos, wait, none* (with the exception that *none* cannot be included as an entry for data_1 if data_2 has an entry other than *none*).

Requirements

Basic Call Vectoring software must be installed, and the converse split must be vector-controlled. If the keyword *digits* is included within the command syntax, Call Prompting software must be installed. Moreover, Call Prompting software is necessary to allow for the full functionality of Voice Response Integration (VRI). Also, skills replace splits only if Expert Agent Selection (EAS) is optioned. To include the keyword *wait* in the command syntax, the “Vectoring (G3V4 Advanced Routing” customer option must be enabled.

Examples

converse-on split *1* pri *h* passing *none* and *none*

converse-on split *20* pri *m* passing *123456* and *none*

converse-on skill *57* pri *h* passing *vdn* and *ani*

converse-on skill *3rd* pri *t* passing *digits* and *qpos*

converse-on skill *23* pri *h* passing *wait* and *none*

Operation

NOTE:
Refer to Appendix H for details regarding call flows involving the *converse-on* command.

The *converse-on* command is designed primarily to integrate Voice Response Units (VRUs), principally the CONVERSANT Voice Information System (VIS), with the DEFINITY system. The command effects data passing between DEFINITY and the VRU, and it enables the caller to hear the appropriate voice response script housed in the VRU.

If the command is successful, it delivers the call to a predetermined split (skill), which is referred to as the converse split (skill). Once the call is answered by the VRU, the command may or may not pass data to the VRU (depending upon the parameters of the command). Regardless of whether or not data is passed, the caller is then connected to the VRU, which in turn executes the voice response script. If by this time the call has already queued to a nonconverse split (skill), the call retains its position in the nonconverse split (skill) queue. If an agent from the nonconverse split (skill) becomes available to service the call while the voice response script is being executed, DEFINITY drops the line to the voice information system and connects the caller to the available agent. The voice information system, in turn, detects the disconnect and terminates the voice response script. Whenever a voice response script is executed, any audible feedback provided by the vector is disconnected, and no further vector steps are executed until the voice response script is executed.

The VRU may or may not eventually return data to the switch. If, once the voice response script is completed, there is no data to be returned from the voice information system to DEFINITY, the VRU drops the line to DEFINITY, and vector processing is reactivated on the switch.

If there is data to be returned to the switch, the ‘‘Converse data return code’’ is outpulsed before the data to be passed is outpulsed. Once all VRU data is received, it is stored in the Call Prompting digits buffer as dial-ahead digits, and vector processing is reactivated. Digits returned by the voice information system are not heard by the caller.

Digits returned from the VRU can be:

- Displayed on the answering agent’s display set (automatically for 2-line displays, or by using the ‘‘CALLR-INFO’’ button for 1-line displays)
- Treated as an extension in a *route-to digits* step

- Used for vector conditional branching in a step containing a command with the *if digits* parameter
- Tandemed to an ASAI host

DEFINITY can be set up to pass information in-band to the voice information system. In such a case, the *converse-on* command can outpulse up to two groups of digits to the voice information system. The digits may serve two major purposes: the digits may notify the voice information system of the application to be executed, and they may share call related data, such as ANI (BN) or caller digits collected by DEFINITY. (In many applications, both application selection and data sharing are required.)

Since in many cases the digit strings are of variable length, DEFINITY always appends a pound sign (#) character to the end of each digit string. The *Prompt and collect* steps in the voice response script must therefore always be administered to expect ‘‘#’’ as the end-of-string symbol and to include ‘‘#’’ in the digit count.

The sending of ‘‘#’’ prevents excessive delays caused by digit timeouts, and it prevents other problems caused by timeouts. It also ensures that each data field is used to satisfy a single *prompt and collect* step.

Any data passed from DEFINITY to a VRU is outpulsed in-band. The user can administer two time delays on the System Parameter Features form: ‘‘converse first data delay’’ and ‘‘converse second data delay.’’ These delays may range from 0 to 9 seconds with a default of zero seconds for the converse first data delay and a default of two seconds for the converse second data delay. The delays are needed to give the VRU time to invoke an application and to allocate a touch-tone receiver to receive the passed digits.

NOTE:
No time delays are invoked when the keyword ‘‘none’’ is administered.

If <data_1> is not ‘‘none,’’ the converse first data delay timer starts when the call is answered by the VRU. When the timer expires, the <data_1> digits are outpulsed in-band to the VRU. The end-of-string character (#) is then outpulsed.

If <data_2> is not ‘‘none,’’ the converse second data delay timer starts when the end-of-string character (#) from the first digit string is outpulsed. When the timer expires, the <data_2> digits are outpulsed in-band to the VRU. The end-of-string character (#) for the second digit string is then outpulsed. The following values may be administered for <data_1> and <data_2> within the *converse-on* command:

- **Administered digit string**: This string can contain up to six characters consisting of one or more digits (0 through 9) or asterisks (*). The pound sign (#) may not be included in a digit string because it is reserved as the end-of-string character. However, a single ‘‘#’’ may be administered.

- **ani**: If the call is an internal call or an incoming DCS call, this data type causes the extension of the calling party to be outpulsed. If the call is an incoming ISDN-PRI call with ANI (BN) provided to DEFINITY, the calling party number/billing number (CPN/BN) of the calling party is outpulsed to the voice information system. If there is no ANI (BN) to send, the end-of-string pound sign (#) is the only character outpulsed. Any other type of incoming call results in ‘‘#’’ being outpulsed.
- **digits**: This data type can be used only if Call Prompting is optioned, and it causes the most recent set of digits collected in vector processing to be outpulsed. If no digits are available, the end-of-string pound sign (#) is the only character outpulsed.
- **none**: This data type causes no characters to be outpulsed. Also, no end-of-string pound character (#) is outpulsed, and no time delays are invoked.
- **qpos**: This data type causes the value of the queue position of a call in a nonconverse split to be outpulsed. This value is a variable length data item from which between one and three digits can be outpulsed. If the call is not queued, the end-of-string pound sign (#) is the only character that is outpulsed.

  NOTE:
  The use of this keyword is not recommended with multiple split (skill) queuing because any queue position value that is sent may not be meaningful. However, if the call is queued to multiple nonconverse splits (skills), the value of the caller’s queue position in the first nonconverse split (skill) is sent.

  This data may be used by the voice information system to inform callers of their position in queue or to decide whether to execute a long or short version of a voice response script.
- **vdn**: This data type causes the VDN extension to be outpulsed. In cases where multiple VDNs are accessed, normal VDN override rules determine which VDN extension is outpulsed.
- **wait:** This data type can be used only if the “Vectoring (G3V4 Advanced Routing)” customer option is enabled. It causes the expected wait time of the call to be outpulsed. See "VDN Calls" on page 6-12 for a detailed description of expected wait time. If the call is not queued or if it is queued only to splits that are unstaffed or splits where all agents are in AUX work mode, the end-of-string character “#” is the only character outpulsed.
- **#**: This is the only character outpulsed. Outpulsing this character causes the corresponding *prompt and collect* command in the voice response script to be skipped.

DEFINITY always outpulses a pound character (#) at the end of each digit string. Where ‘‘#’’ is administered, or where the ‘‘digits’’ keyword is administered and the last digit collected from the caller is ‘‘#,’’ only one ‘‘#’’ is outpulsed. No ‘‘#’’ is outpulsed when the keyword ‘‘none’’ is administered.

If *data_1* is administered as ‘‘none,’’ *data_2* must also be ‘‘none.’’

Answer Supervision Considerations

Answer supervision is returned only once during the life of a call. If a call is answered as a result of a *converse-on* step, answer supervision is sent only if it has not been sent previously. If digits are passed to the VRU, answer supervision is not sent until after the digits are outpulsed.

Feature Interactions

- Abandon Call Search

  If the *converse-on* step places a call to a hunt group, and if the incoming call was placed via a trunk group with Abandon Call Search activated, the system checks that the calling party has not abandoned the call (that is, hung up) before terminating to an agent.

- Adjunct Switch Applications Interface (ASAI)

  Since vector-controlled splits (skills) cannot be ASAI-monitored domains, ASAI cannot be used to supplement the operation of the *converse-on* step.

  If a *converse-on* step places a call to an ASAI-monitored domain, ASAI event messages are sent over the ASAI link.

  Whenever a *converse-on* step places an ASAI-monitored call, the ALERTing message sent to the ASAI host includes a Cause IE, Coding Standard 3 value 23 (CS3/23). This informs the ASAI host that the call has not been de-queued from any nonconverse splits (skills).

  If a *converse-on* step is executed while an adjunct routing request is outstanding, the route request is canceled.

- Audio Information Exchange (AUDIX)

  If a *converse-on* step calls AUDIX, the call is treated as a direct call to AUDIX. The caller hears the ‘‘welcome to AUDIX’’ message and may retrieve his or her messages in the usual manner.

  If a call is forwarded to or covers to a VDN and is then delivered to an AUDIX hunt group by a *converse-on* step, the call to AUDIX is treated as a redirected call, and the caller may leave a message for the principal.

- Auto-Available Splits (Skills)

  A *converse-on* step may place a call to an auto-available split (skill). Except in cases where the converse split (skill) is ASAI-controlled, auto-available converse splits (skills) are recommended for Voice Response Integration (VRI).

- Basic Call Management System (BCMS)

  BCMS tracks calls placed by a *converse-on* step to a BCMS-measured hunt group. Since with the *converse-on* step it is now possible for a call to be ‘‘answered’’ in more than one split (skill), trunk totals may no longer match split totals. However, VDN totals and trunk totals will match.

- BCMS VDN Reports

  For call tracking in BCMS VDN reports, a *converse-on* step is treated like an *announcement* step. A call is considered ‘‘answered’’ when it is answered by a nonconverse split (skill) but never when it is answered by a converse split (skill).

- Call Coverage

  Call Coverage does not apply because the *converse-on* step may deliver calls only to vector-controlled splits (skills), which do not have coverage paths.

- Call Detail Recording

  For incoming calls to a VDN, the duration of the call is recorded from the time answer supervision is returned. Answer supervision is returned for a successful *converse-on* step. No ineffective call attempt records are generated for *converse-on* steps that fail. Also, no outgoing calls can be placed by a *converse-on* step.

- Call Management System (CMS)

  CMS tracks calls placed by a *converse-on* step to a CMS-measured hunt group or split. *converse-on* vector steps may be administered from CMS 3.1 and later releases. Since with the *converse-on* step it is now possible for a call to be ‘‘answered’’ in more than one split (skill), trunk totals no longer match split totals. However, VDN totals and trunk totals will match.

- Call Park

  Calls placed by a *converse-on* step may not be parked.

- Call Pickup

  Calls placed by a *converse-on* step ringing at an agent station may be picked up if that agent is part of a pickup group. Subsequent transfers are denied.

- Call Prompting

  The Call Prompting customer option must also be enabled to gain full VRI functionality. Without Call Prompting, any data returned by the voice information system cannot be collected and processed by the switch.

  If the *converse-on* step places a call to a split (skill) of live agents, any digits collected previously may be displayed by agents using the callr-info button.

- Call Vectoring—Basic

  The *converse-on* step is an enhancement to the Basic Call Vectoring customer option. This option must be enabled in order to invoke the VRI feature.

- Class of Restriction (COR)

  As is the case for the *queue-to main split (skill)* and *check-backup split (skill)* vector steps, no COR checking is carried out when a *converse-on* step places a call to a split (skill).

- Conference

  Any attempt to conference a call placed by a *converse-on* step is denied.

- Coverage Callback

  A call placed by a *converse-on* step does not follow any coverage paths. Therefore, Coverage Callback is not available. Also, if a call reaches a *converse-on* step via a VDN in a coverage path, coverage callback cannot be used.

- Direct Department Calling (DDC)

  A converse split may be administered as a direct department calling split.

- Distributed Communications System (DCS)

  If an incoming DCS call is placed to a vector with a *converse-on split (skill) x pri y passing ani ...* step, the DCS extension of the calling party is outpulsed.

- Priority Levels

  A call placed by a *converse-on* step may be queued at one of four priority levels: low, medium, high or top.

- Hunt Groups

  The *converse-on* step may deliver a call to a vector-controlled hunt group, ACD split (skill), Message Center or AUDIX hunt group.

- Integrated Services Digital Network (ISDN)

  The *converse-on* step may be administered to outpulse to the voice information system the ANI (calling party number/billing number CPN/BN) of the calling party via use of the ‘‘ani’’ keyword.

- Intercept Treatment

  A caller is never given intercept treatment upon execution of a *converse-on* step. Failing to place a converse call successfully results in the failure of the *converse-on* step. Vector processing continues at the next vector step.

- Interflow

  Since a *converse-on* step can place calls only to hunt groups that are vector-controlled, and since the activation of Call Forwarding for a vector-controlled hunt group is blocked, calls placed by a *converse-on* step to a hunt group cannot interflow.

- Intraflow

  Since a *converse-on* step can place calls only to hunt groups that are vector-controlled (that is, without coverage paths), intraflow is not possible.

- Live Agents

  Although not recommended, the switch does not prevent a *converse-on* step from delivering a call to a group of live agents. To the agent, the call looks like any other ACD call. However, certain features, such as call transfer, conference, and supervisor assist are denied.

  The answering agent can display any digits collected prior to executing the *converse-on* step by using the ‘‘callr-info’’ button.

- Look-Ahead Interflow

  If a call placed by a *converse-on* vector step is answered by a VRU, or if such a call queues to a split (skill) on the receiving switch while a Look-Ahead Interflow call attempt is outstanding, the Look-Ahead Interflow call attempt is accepted.

  A *converse-on* step that fails is neutral.

- Message Center

  The *converse-on* step may deliver calls to message hunt groups. Such calls are treated as direct calls to the message.

  If a call is forwarded to a VDN and then delivered to a message split by a *converse-on* step, the call is treated as a redirected call.

- Multiple Split (Skill) Queuing

  A call can be queued to three different splits (skills) and then to a converse split (skill) as a result of a *converse-on* step.

- Music on Hold

  During the data return phase of a *converse-on* step, the caller is temporarily placed on hold. Music on hold, if administered, is suppressed.

- Non-Vector Controlled Splits (Skills)

  A *converse-on* step may not place a call to a nonvector-controlled split (skill).

- Priority Queuing

  The queue priority of a call placed by a *converse-on* step is administrable on the vector step.

- Queue Status

  All queue status display, queue status indication and queue warning wall lamp feature capabilities also apply to calls queued by the *converse-on* command.

- Queuing

  Calls handled by the *converse-on* step queue when they are delivered to busy hunt groups. Call Vectoring audible feedback is not disconnected while a converse call is in queue.

  If a *converse-on* step is executed while a call is queued to a nonconverse split (skill), the call remains in queue for the nonconverse split (skill).

  The queue priority of the call is administrable on the vector step.

- Recorded Announcement

  VRI may be used to increase the system's recorded announcement capacity by offloading some recorded announcements to the VRU. Callers can be redirected by the *converse-on* step to a group of VRU ports and use data passing to specify the correct announcement to play.

- Redirection on No Answer (RONA)

  If a *converse-on* step places a call to a hunt group with a ''no answer timeout'' administered, and if the call rings at an agent terminal/port for longer than the administered timeout, the call is redirected, and the agent/port is put into the AUX work state (or logged out if the agent is a member of an auto-available split (skill)).

  Thereafter, under RONA, the call is requeued to the split (skill) unless there is no room in the queue or unless this is an auto-available split (skill) whose agents are all logged out. If the call cannot be requeued, the *converse-on* step fails, a vector event is logged, and vector processing is restarted at the next vector step.

- Service Observing

  Calls placed by a *converse-on* step may be service observed. To prevent the observer from hearing tones being outpulsed to the VRU, the observer is not connected to the call until the data passing phase is complete. If data is returned by the VRU, the observer is put in service observing pending mode, and the calling party is temporarily put on hold while the VRU digits are outpulsed. Upon completion of the converse session, and once the VRU hangs up the line, the observer remains in service observing pending mode.

  It is not recommended that a service observing warning tone be administered since the warning tone may interfere with the interaction between the voice information system and the calling party.

- System Access Terminal (SAT)

  converse-on steps may be administered from the SAT terminal.

- System Measurements

  System measurements track converse calls to hunt groups and attendant groups.

- Touch-Tone Dialing

  Any touch-tone dialing by the calling party during the digit passing phases of a session involving a *converse-on* step does not result in corruption of data or in the collection of this data in the form of dial-ahead digits by DEFINITY.

  Only after the digit passing phase from DEFINITY to the voice information system is completed can the calling party enter touch-tone digits in response to a voice information system prompt. Only after the voice information system to DEFINITY data return phase is completed and an additional *collect digits* vector step is executed can the calling party enter a touch-tone response to a DEFINITY prompt.

- Transfer

  A call placed by a *converse-on* step may not be transferred. The only form of transfer allowed is the data passing operation during the data return phase at the end of a voice response script.

  If an illegal attempt to transfer a converse call is made, a vector event is logged, the line to the voice information system is dropped, and vector processing is reactivated at the next vector step.

  If an illegal transfer is attempted by a live agent with a multifunction set, the transfer is denied and the agent may reconnect to the call.

- Transfer out of AUDIX

  If a *converse-on* step delivers a call to an AUDIX hunt group, and if the calling party then attempts to transfer out of AUDIX, the transfer fails, and vector processing is reactivated at the next vector step.

- Uniform Call Distribution (UCD)

  A converse split (skill) may be administered as a Uniform Call Distribution split (skill).

- VDN as a Coverage Point

  If a call covering to a VDN is processed by the *converse-on* command and subsequently reaches a station user (that is, a member of a ‘‘converse split (skill)’’), and if the ‘‘converse split (skill)’’ agent attempts to activate Consult (coverage), or Coverage Leave Word Calling, any of these coverage attempts is denied because the call is still in vector processing. If the ‘‘converse split (skill)’’ is an AUDIX/Message Center split (skill), the call covered to the VDN is treated like a redirected call to AUDIX/MCS; the original principal and reason for redirection is used in the same manner as a Call Forwarded call to a VDN.

- VDN Override

  If a call that accesses multiple VDNs encounters a *converse-on* step passing ‘‘vdn,’’ normal override rules determine which VDN number is outpulsed to the VRU.

- VDN Reports

  For call tracking in CMS and BCMS VDN reports, a *converse-on* step is treated like an *announcement* step. A call is considered ‘‘answered’’ when it is answered by a nonconverse split (skill) but never when it is answered by a converse split (skill).

- Vector-controlled Splits (Skills)

  A *converse-on* step may place a call to a split (skill) only if that split (skill) is administered as a vector-controlled split (skill).

CMS Interactions

CMS tracks calls placed by a *converse-on* step to a CMS-measured split (skill). *converse-on* vector steps may be administered from R3 CMS and later releases. Since with the *converse-on* step it is now possible for a call to be ‘‘answered’’ in more than one split (skill), trunk totals no longer match split (skill) totals. However, VDN totals and trunk totals will match.

For call tracking in CMS VDN reports, a *converse-on* step is treated like an *announcement* step. A call is considered ‘‘answered’’ when it is answered by a nonconverse split (skill) but never when it is answered by a converse split (skill).

BCMS Interactions

BCMS tracks calls placed by a *converse-on* step to a BCMS-measured split (skill). Since with the *converse-on* step it is now possible for a call to be ‘‘answered’’ in more than one split (skill), trunk totals may no longer match split totals. However, VDN totals and trunk totals will match.

For call tracking in BCMS VDN reports, a *converse-on* step is treated like an *announcement* step. A call is considered ‘‘answered’’ when it is answered by a nonconverse split (skill) but never when it is answered by a converse split (skill).

Disconnect Command

```
disconnect
```

Purpose

Ends treatment of a call and removes the call from the switch. Also allows the optional assignment of an announcement that will play immediately before the disconnect.

Syntax

disconnect after announcement <extension>

Valid Entries

extension: *none* or valid announcement extension

Requirements

Basic Call Vectoring software must be installed. Also, the relevant announcements must be administered and recorded.

Example

disconnect after announcement 2556

Operation

While the command's optional announcement is playing, the call remains in queue and can be connected to an agent. When the announcement completes (or is not specified), the command forces a disconnect, ends the treatment of the call, and removes the call from the switch.

Answer Supervision Considerations

If the switch has not yet sent answer supervision, the switch does so immediately before disconnecting the call, whether an announcement is specified or not. If an announcement is specified, answer supervision is given before an attempt is made to connect the announcement. The exception is for ISDN calls, where the disconnect can occur without answer supervision being sent when an announcement is not played.

Feature Interactions

For Look-Ahead Interflow, the command can be considered either a call acceptance vector command or a call denial vector command.

The command is considered a call acceptance vector command whenever an announcement is included within the command and one of the following is true:

- Announcement is available.
- Call is queued for an announcement.
- Announcement is retried.

The command is considered a call denial vector command whenever one of the following is true:

- No announcement is included within the command.
- Announcement is included within the command, but the announcement is unavailable.

CMS Interactions

R3 CMS:

Disconnect Command

Database Item	Report Heading
DISCCALLS/DISCTIME	Calls Forced Disc
	Calls Busy/Disc
OTHERCALLS/OTHERTIME	Inbound Other Calls
INTIME	Avg Time In Vector

DISCTIME, OTHERTIME, and INTIME for splits and vectors are tracked according to when the announcement starts. DISCTIME, OTHERTIME and INTIME for VDNs are tracked according to when the trunk idles.

R2 CMS: Calls given forced disconnect are tracked in the FDISCCALLS and FDISCTIME database items. These calls are not shown in standard reports. FDISCTIME is tracked according to when the announcement starts. VECTIME is recorded for the vector, and it is reported as ‘‘Avg Time in Vector.’’

BCMS Interactions

A call that is disconnected via the command is tracked as “OTHER” in the VDN Report.

Goto Step Command

```
goto step
```

Purpose

Allows conditional or unconditional movement (branching) to a preceding or subsequent step in the vector.

Syntax

goto step <step #> if **unconditionally**

goto step <step #> if **digits** <comparator> <digits>

goto step <step #> if **digits** <option> table <table>

goto step <step #> if **ani** <comparator> <digits>

goto step <step #> if **ani** <option> table <table>

goto step <step #> if **ii-digits** <comparator> <digits>

goto step <step #> if **ii-digits** <option> table <table>

goto step <step #> if **time-of-day** is <day> <hour>: <minute> to <day> <hour>: <minute>

Conditions = available-agent, staffed-agents:

goto step <step #> if <condition> in split <split #> <comparator> <threshold>

goto step <step #> if <condition> in skill <skill #> <comparator> <threshold>

Conditions = calls-queued, oldest call-wait:

goto step <step #> if <condition> in split <split #> pri <priority level> <comparator> <threshold>

goto step <step #> if <condition> in skill <skill #> pri <priority level> <comparator> <threshold>

Condition = rolling-asa:

goto step <step #> if <condition> for split <split #> <comparator> <threshold>

goto step <step #> if <condition> for skill <skill #> <comparator> <threshold>

goto step <step #> if <condition> for vdn <vdn> <comparator> <threshold>

Condition = counted-calls:

goto step <step #> if <condition> to vdn <vdn> <comparator> <threshold>

Condition = expected-wait:

goto step <step #> if <condition> for split <split #> pri <priority level> <comparator> <threshold>

goto step <step #> if <condition> for skill <skill #> pri <priority level> <comparator> <threshold>

goto step <step #> if <condition> for call <comparator> <threshold>

Valid Entries

step #: *1-32*.

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs); *1st, 2nd, 3rd* (VDN)

condition:	comparator:[1]	threshold:
unconditionally	N/A	N/A
rolling-asa	>, >=, =, <>	*0-998* seconds (even numbers only)
	<, <=	*2-998* seconds (even numbers only)
available-agents	>, >=, =, <>	*0-199* (G3i), *0-998* (G3r)
		0-149 (G3s, G3vs)
	<, <=	*1-200* (G3i), *1-999* (G3r)
		1-150 (G3s, G3vs)
calls-queued	>, >=, =, <>	*0-199* (G3i, G3s, G3r), *0-998* (G3r)
	<, <=	*1-200* (G3i, G3s, G3vs), *1-999* (G3r)
counted-calls	>, >=, =, <>	*0-999*
	<, <=	*1-999*
expected-wait	>, >=, =, <>	*0-9999* seconds
	<, <=	*1-9999* seconds
oldest call-wait	>, >=, =, <>	*0-998* seconds (even numbers only)
	<, <=	*2-998* seconds (even numbers only)

condition:	comparator:[1]	threshold:
staffed-agents	>, >=, =, <>	*0-199* (G3i), *0-998* (G3r)
		0-149 (G3s, G3vs)
	<, <=	*1-200* (G3i), *1-999* (G3r)
		1-150 (G3s, G3vs)

[1] Comparators =, >=, <=, <> are only available for these conditions with "Vectoring (G3V4 Enhanced)."

digits: the following values are accepted:

command	comparator[1]	value[2]
goto step <step#> if digits	<, <=, >, >=	String of 0-9
	=, <>	String of 0-9, #, +, ?, none
goto step <step#> if ani	<, <=, >, >=	String of 0-9
	=, <>	String of 0-9, #, +, ?, none
goto step <step#> if ii-digits	<, <=, >, >=	Two consecutive digits from string of 0-9
	=, <>	Two consecutive digits from string of 0-9, +, ?, none

[1] Comparators >, >=, <, <=, <> are only available for the *goto step if digits* command with Vectoring (G3V4 Enhanced)

[2] Wildcards (+, ?) can only be used with Vectoring (G3V4 Enhanced). The + represents a group of digits (0 or more) and can only be used as the first or last character of the string. Only one + can be used in any digit string. The ? represents a single digit. Any number of them can be used at any position in the digit string. *None* is only a valid entry with G3V4 and later releases.

priority level: *l* (low), *m* (medium), *h* (high), *t* (top).

day: *mon, tue, wed, thu, fri, sat, sun, all (that is, "on any day of the week").*

hour: *00* to *23* (military format).

minute: *00* to *59* (military format).

option: *in, not-in (Vectoring (G3V4 Enhanced) only)*

table: 1 through 5 (G3vs/s); 1 through 10 (G3i); 1 through 100 (G3r). The number of the Vector Routing Table containing the numbers to be used to compare against ani, ii-digits or digits collected with the *collect digits* command.

vdn: assigned vdn extension, *active*, *latest*. Active is the active called VDN as modified by VDN override rules. Latest is the VDN assigned to the vector in which the call is currently being processed.

Requirements

Basic Call Vectoring software must be installed for all the options. Call Prompting software is required for the digits option. Also, skills replace splits only if Expert Agent Selection (EAS) is optioned. Wildcard matching and the use of Vector Routing Tables require Vectoring (G3V4 Enhanced). In addition, to use the *rolling-asa*, *expected-wait*, and *counted-calls* conditions, the “Vectoring (G3V4 Advanced Routing” customer option must be enabled. To use *ani* and *ii-digits* routing, G3V4 or a later release is required and the “Vectoring (ANI/II-Digits Routing)” customer option must be enabled.

Examples

goto step *8* if **available-agents** in split *67 < 5*

goto step *12* if **calls-queued** in split *51* pri *t < 17*

goto step *7* if **time-of-day** is *mon 16:30* to *tue 7:30*

goto step *12* if **calls-queued** in skill *3rd* pri *t > 50*

goto step *8* if **available-agents** in skill *1st < 10*

goto step *11* if **oldest-call-wait** in split *26* pri *t >= 20*

goto step *6* if **rolling-asa** for split *9 > 30*

goto step *10* if **counted-calls** to vdn *5372 >= 50*

goto step *4* if **ani** *in* table *10*

goto step *9* if **ii-digits** *= 06*

goto step *11* if **expected-wait** for call *> 30*

Operation

If the command syntax includes **unconditionally**, the command always branches. Otherwise, branching takes place according to one of the conditions that follow:

- The average speed of answer for the indicated split (skill) or VDN meets the constraints defined by the comparator and threshold value.
- Number of available agents in the indicated split (skill) meets the constraints defined by the comparator and the threshold value.
- Number of queued calls in the indicated split (skill) and at the specified priority level (or higher) meets the constraints defined by the comparator and the threshold value.
- The number of active calls in the indicated VDN meets the constraints defined by the comparator and the threshold value.
- The expected wait time at the specified priority level for the indicated split (skill), or for the call meets the constraints defined by the comparator and the threshold value.
- Oldest call-waiting in the indicated split (skill) at the specified priority level (or higher) has been waiting for a period of time within the constraints defined by the comparator and the threshold value, which is expressed in seconds.
- Number of staffed agents in the indicated split (skill) meets the constraints defined by the comparator and the threshold value.
- Digits collected via the *collect digits* command match the criteria defined by the comparator for the specified digit string. Or, the digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table. The "#" digit can be tested against as a single digit.
- The ani digits match the criteria defined by the comparator for the specified digit string. Or, the ani digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- The ii-digits match the criteria defined by the comparator for the specified digit string. Or, the ii-digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- Time-of-day criteria are met.

  NOTE:
  The syntax for this condition can be illustrated by a couple of examples, as follows: *mon 8:01 to fri 17:00* means ‘‘anytime between 8:01 A.M. Monday through 5:00 P.M. Friday,’’ and *all 17:00 to all 8:00* means ‘‘between 5:00 P.M. and 8:00 A.M. on any day of the week.’’)

The unconditional form of the command is commonly used for skipping vector commands as well as for looping through vector commands.

Answer Supervision Considerations

The call answer is not affected by the command.

Feature Interactions

For Look-Ahead Interflow, the command is considered a neutral vector command in all cases. When a call experiences Look Ahead interflow, the ANI value is sent along with the call only for ISDN PRI calls. ANI is not sent for internal or DCS calls.

CMS Interactions

The command is not tracked on CMS or BCMS.

The ANI value is passed to CMS when the call first starts vector processing if the following is true:

- Basic Call Vectoring and/or Call Prompting is optioned
- ANI is available from the network, the call is internal, or is received over DCS
- The CMS is R3 or a later version
- The G3V4 customer option is set to “y.”

The ii-digits value for a call is not passed to CMS. ANI and ii-digits are no passed to BCMS.

Goto Vector Command

```
goto vector
```

Purpose

Allows conditional or unconditional movement (branching) to another vector. The goto vector step does not remove a call from queues in which it is already placed.

Syntax

goto vector <vector #> if **unconditionally**

goto vector <vector #> if **digits** <comparator> <digits>

goto vector <vector #> if **digits** <option> table <table>

goto vector <vector #> if **ani** <comparator> <digits>

goto vector <vector #> if **ani** <option> table <table>

goto vector <vector #> if **ii-digits** <comparator> <digits>

goto vector <vector #> if **ii-digits** <option> table <table>

goto vector <vector #> if **time-of-day** is <day> <hour> : <minute> to <day> <hour> : <minute>

Conditions = available-agent, staffed-agents:

goto vector <vector #> if <condition> in split <split #> <comparator> <threshold>

goto vector <vector #> if <condition> in skill <skill #> <comparator> <threshold>

Conditions = calls-queued, oldest call-wait:

goto vector <vector #> if <condition> in split <split #> pri <priority level> <comparator> <threshold>

goto vector <vector #> if <condition> in skill <skill #> pri <priority level> <comparator> <threshold>

Condition = rolling-asa:

goto vector <vector #> if <condition> for split <split #> <comparator> <threshold>

goto vector <vector #> if <condition> for skill <skill #> <comparator> <threshold>

goto vector <vector #> if <condition> for vdn <vdn> <comparator> <threshold>

Condition = counted-calls:

goto vector <vector #> if <condition> to vdn <vdn> <comparator> <threshold>

Condition = expected-wait:

goto vector <vector #> if <condition> for split <split #> pri <priority level> <comparator> <threshold>

goto vector <vector #> if <condition> for skill <skill #> pri <priority level> <comparator> <threshold>

goto vector <vector #> if <condition> for call <comparator> <threshold>

Valid Entries

vector #: *1 through 256* (G3i), *1 through 512* (G3r), *1 through 48* (G3s, G3vs).

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (specific: G3i), *1 through 255* (specific: G3r), *1 through 24* (specific: G3s, G3vs); *1st, 2nd, 3rd* (VDN)

condition:	comparator: 1	threshold:
unconditionally	N/A	N/A
rolling-asa	>, >=, =, <>	*0-998* seconds (even numbers only)
	<, <=	*2-998* seconds (even numbers only)
available-agents	>, >=, =, <>	*0-199* (G3i), *0-998* (G3r)
		0-149 (G3s, G3vs)
	<, <=	*1-200* (G3i), *1-999* (G3r)
		1-150 (G3s, G3vs)

condition:	comparator: [1]	threshold:
calls-queued	>, >=, =, <>	*0-199* (G3i, G3s, G3r), *0-998* (G3r)
	<, <=	*1-200* (G3i, G3s, G3vs), *1-999* (G3r)
counted-calls	>, >=, =, <>	*0-999*
	<, <=	*1-999*
expected-wait	>, >=, =, <>	*0-9999* seconds
	<, <=	*1-9999* seconds
oldest call-wait	>, >=, =, <>	*0-998* seconds (even numbers only)
	<, <=	*2-998* seconds (even numbers only)
staffed-agents	>, >=, =, <>	*0-199* (G3i), *0-998* (G3r)
		0-149 (G3s, G3vs)
	<, <=	*1-200* (G3i), *1-999* (G3r)
		1-150 (G3s, G3vs)

[1] Comparators =, >=, <=, <> (not equals) are only available for these conditions with Vectoring (G3V4 Enhanced).

digits: the following values are accepted:

command	comparator[1]	value[2]
goto step <step#> if digits	<, <=, >, >=	String of 0-9
	=, <>	String of 0-9, #, +, ?, none
goto step <step#> if ani	<, <=, >, >=	String of 0-9
	=, <>	String of 0-9, #, +, ?, none
goto step <step#> if ii-digits	<, <=, >, >=	Two consecutive digits from string of 0-9
	=, <>	Two consecutive digits from string of 0-9, +, ?, none

[1] Comparators >, >=, <, <=, <> (not equals) are only available for the *goto step if digits* command with Vectoring (G3V4 Enhanced)

2.Wildcards (+, ?) can only be used with Vectoring (G3V4 Enhanced). The + represents a group of digits (0 or more) and can only be used as the first or last character of the string. Only one + can be used in any digit string. The ? represents a single digit. Any number of them can be used at any position in the digit string. *None* is only a valid entry with G3V4 and later releases.

priority level: *l* (low), *m* (medium), *h* (high), *t* (top).

day: *mon, tue, wed, thu, fri, sat, sun, all*.

hour: *00* to *23* (military format).

minute: *00* to *59* (military format).

option: *in, not-in* (Vectoring (G3V4 Enhanced) only)

table: 1 through 5 (G3vs/s); 1 through 10 (G3i); 1 through 100 (G3r). The number of the Vector Routing Table containing the numbers to be used to compare against ani, ii-digits or digits collected with the *collect-digits* command.

vdn: assigned vdn extension, active, latest. Active is the active called VDN as modified by VDN override rules. Latest is the VDN assigned to the vector in which the call is currently being processed.

Requirements

Basic Call Vectoring software must be installed for all the options. Call Prompting software is required for the digits option. Also, skills replace splits only if Expert Agent Selection (EAS) is optioned. Wildcard matching and the use of Vector Routing Tables require Vectoring (G3V4 Enhanced). In addition, to use the *rolling-asa*, *expected-wait*, and *counted-calls* conditions, the “Vectoring (G3V4 Advanced Routing)” customer option must be enabled. To use *ani* and *ii-digits* routing, G3V4 or a later release is required and the “Vectoring (ANI/II-Digits Routing)” customer option must be enabled.

Examples

goto vector *107* **if unconditionally**

goto vector *8* if **available-agents** in split *67 < 5*

goto vector *21* if **digits** *>=14*

goto vector 8 if **available-agents** in skill *1st > 10*

goto vector *32* if **digits** *in* table *12*

goto vector 9 if **expected-wait** in skill *3* pri *t > 10*

goto vector 99 if **calls-queued** in skill *2nd* pri *t > 5*

goto vector 8 if **rolling-asa** in split *14 > 25*

Operation

If the command syntax includes **unconditionally**, the command always branches. Otherwise, branching takes place according to one of the conditions that follow:

- The rolling average speed of answer for the indicated split (skill) or VDN meets the constraints defined by the comparator and threshold value.
- Number of available agents in the indicated split (skill) meets the constraints defined by the comparator and the threshold value.
- Number of queued calls in the indicated split (skill) and at the specified priority level (or higher) meets the constraints defined by the comparator and the threshold value.
- The number of active calls in the indicated VDN meets the constraints defined by the comparator and the threshold value.
- The expected wait time at the specified priority level for the indicated split (skill), or the expected wait time for the call meets the constraints defined by the comparator and the threshold value.
- Oldest call-waiting in the indicated split (skill) at the specified priority level has been waiting for a period of time within the boundaries defined by the comparator and the threshold value, which is expressed in seconds.
- Number of staffed agents in the indicated split (skill) meets the constraints defined by the comparator and the threshold value.
- Digits collected via the *collect digits* command match the criteria defined by the comparator for the specified digit string. Or, the digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- The ani digits match the criteria defined by the comparator for the specified digit string. Or, the ani digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- The ii-digits match the criteria defined by the comparator for the specified digit string. Or, the ani digits are found or not found, depending upon the option chosen, in the specified Vector Routing Table.
- Time-of-day criteria are met.

NOTE:
The syntax for this condition can be illustrated by a couple of examples, as follows: *mon 8:01 to fri 17:00* means ‘‘anytime between 8:01 A.M. Monday through 5:00 P.M. Friday,’’ and *all 17:00 to all 8:00* means ‘‘between 5:00 P.M. and 8:00 A.M. on any day of the week.’’)

The unconditional form of the command is useful for applications that require the processing of more than 32 commands.

Answer Supervision Considerations

Call answer is not affected by the command.

Feature Interactions

For Look-Ahead Interflow, the command is considered a neutral vector command in all cases. When a call experiences Look Ahead interflow, the ANI value is sent along with the call only for ISDN PRI calls. ANI is not sent for internal or DCS calls

CMS Interactions

R3 CMS:

Goto Vector Command

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME GOTOCALLS/ GOTOTIME	Vector Flow Out	
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In	new vector

R2 CMS: Calls that go to another vector are tracked as OUTFLOW and INFLOW (for the new vector) in the vector tables. These calls are not shown in the standard reports. VECTIME is recorded for the vector and reported as ‘‘Avg Time in Vector.’’

The ANI value is passed to CMS when the call first starts vector processing if the following is true:

- Basic Call Vectoring and/or Call Prompting is optioned
- ANI is available from the network, the call is internal, or is received over DCS
- The CMS is R3 or a later version

- The G3V4 customer option is set to “y.”

The ii-digits value for a call is not passed to CMS.

Messaging Command

```
messaging split (skill)
```

Purpose

Allows the caller to leave a message for the specified extension or the active or latest VDN extension (default).

Syntax

messaging split <split #> for extension <extension>

messaging skill <skill #> for extension <extension>

Valid Entries

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs); *1st, 2nd, 3rd* (VDN)

extension: extension number, “active,” “latest.” Active is the active called VDN as modified by VDN override rules. Latest is the VDN assigned to the vector in which the call is currently being processed. Active is the default for this field. (Prior to G3V4, “none” was used to specify “active” VDN.)

Requirements

Basic Call Vectoring software must be installed. Also, the split (skill) involved must be an AUDIX split (skill), a remote AUDIX split or skill (DCS-AUDIX), or a Message Server Adjunct (MSA) split (skill). Also, skills replace splits only if Expert Agent Selection (EAS) is optioned.

Example

messaging split *18* for extension *2000*

messaging skill *45* for extension *active*

Operation

This command causes the caller to be connected to the AUDIX or Message Center split (skill) so that the caller may leave a message for the specified extension (call answering service or ‘‘mail’’).

If the split (skill) number specified in the command is a valid message service split or skill (such as an AUDIX or a Message Server Adjunct), and if the extension is either a valid assigned extension or is administered as *none* (pre G3V4), or active or latest (G3V4 and later releases) the system attempts to terminate the call to the message service split (skill) for call answering service.

If the call is queued to the message service split (skill), or if the call terminates to an available message service agent or AUDIX voice port, the caller is connected to ringback (signifying successful termination), and vector processing terminates. Termination is unsuccessful, and vector processing continues at the next vector step if any one of the following is true:

- Split (Skill) queue is full.
- AUDIX link is down.
- All AUDIX voice ports are out of service.
- Message service split (skill) is DCS-AUDIX and all DCS trunks are busy.

If call termination is successful, and if the administered extension (or default VDN) is a message service subscriber, the caller can leave a message for the specified extension.

NOTE:
Agent and/or supervisor stations may be equipped with Automatic Message Wait (AMW) lamps to accommodate the ‘‘mail’’ specified in the *messaging split (skill)* command. The lamps can be assigned for VDNs or extensions used to access the messaging split (skill) and for which messages are to be left. When messages are left for these VDNs or extensions, the assigned AMW lamps light.

If the extension or VDN is not a subscriber of the message service, one of the following may occur:

- If the message service split (skill) is AUDIX, the caller receives ringback until he or she disconnects.
- If the message service split is a MSA, the caller may be answered by a message service agent, but no message is taken since the specified extension (default VDN) is not a MSA subscriber.

Answer Supervision Considerations

If answer supervision has not already been returned, it is returned when the messaging service port or station is connected to the call (that is, when the call is answered by the port or station).

Feature Interactions

The command can use an AUDIX or MSA hunt group in its operation.

If the command specifies a specific ‘‘mailbox’’ extension, the original principal for a call covered by a VDN is not passed to the adjunct, and it does not appear in the display to the answering agent. The specified extension appears in the display.

If the command is accessed via a direct call to the VDN, and if the mailbox is administered as ‘‘active’’ or “latest,” the corresponding active or latest VDN extension mailbox is sent to the messaging adjunct. Additionally, if the call is sent to a DEFINITY Message Service split (skill), the associated VDN name is sent to the messaging adjunct.

If the command specifies ‘‘active’’ or “latest” as the mailbox extension, the original principal for a call covered to or forwarded to a VDN is used as the default mailbox for the call instead of the ‘‘active’’ or “latest” VDN. Accordingly, the original principal extension and the reason for redirection are passed to the messaging adjunct, and they subsequently appear in the display to the answering agent.

AUDIX does not support mixed length numbering plans.

If the command leaves a message for a VDN or for another messaging service extension, the Automatic Message Waiting Lamp (AMWL) associated with the VDN or extension lights steady.

For Look-Ahead Interflow, the command can be considered as either a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent or to an AUDIX port.
- Call queues to a messaging split (skill).

The command is considered a neutral vector command whenever the command fails.

CMS Interactions

R3 CMS: When a queued call successfully goes to the messaging split, OUTFLOWCALLS/OUTFLOWTIME (1st split (skill)) and DEQUECALLS/DEQUETIME (2nd/3rd splits (skills)) are tracked in the split (skill) tables. These calls are reported as split (skill) “Flow Out,’’ ‘‘Dequeued Calls,’’ and ‘‘Dequeued Avg Queue Time.’’

Calls that queue via a *messaging split (skill)* command are tracked as CALLSOFFERRED and LOWCALLS (no priority) or MEDCALLS (priority). These calls are shown in the standard reports according to the final disposition of the call.

The presence of the command in a vector enables the calls serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ‘‘ACD Calls,’’ ‘‘Split ACD Calls,’’ and ‘‘Avg Speed Ans.’’

Finally, if the command directs a call to a split (skill), the BACKUPCALLS database item is incremented, and the call is reported as ‘‘Calls Ans in Backup’’ and ‘‘Calls Handled/Backup.’’ The ‘‘Calls Ans in Main’’ report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

A call abandoned after the command routes the call to a station or to an attendant is tracked as ABNCALLS/ABNTIME for the messaging split (skill) and in the VDN/vector tables.

R2 CMS: Calls that go to a messaging split (skill) are tracked as OUTFLOW in the split (skill), vector, and VDN tables. The messaging split (skill) also receives an INFLOW. The calls are shown as ‘‘Flow Out’’ and ‘‘Flow In’’ in standard split (skill) reports. VECTIME is recorded for the vector and reported as ‘‘Avg Time in Vector.’’

BCMS Interactions

A call advanced to another position via the command is tracked as an ‘‘outflow’’ in the VDN Report.

Queue-to Main Command

```
queue-to main split (skill)
```

Purpose

Unconditionally queues a call to a split (skill) and assigns a queuing priority level to the call in case all agents are busy.

Syntax

queue-to main split <split #> pri <priority level>

queue-to main skill <skill #> pri <priority level>

Valid Entries

split #: *1 through 99* (G3i), *1 through 255* (G3r), *1 through 24* (G3s, G3vs)

skill #: *1 through 99* (specific: G3i), *1 through 255* (specific: G3r), *1 through 24* (specific: G3s, G3vs); *1st, 2nd, 3rd* (VDN)

priority level: *l* (low), *m* (medium), *h* (high), *t* (top).

Requirements

Basic Call Vectoring software must be installed. The split (skill) involved must be vector-controlled. Also, skills replace splits only if Expert Agent Selection (EAS) is optioned.

Example

queue-to main split *53* pri *t*

queue-to main skill 1st pri *t*

Operation

A call sent with this command either connects to an available agent in the split (skill) or enters the split's (skill's) queue.

A call may be queued to up to three splits (skills) simultaneously. A call remains queued either until vector processing terminates (via a *disconnect, busy*, or *route-to* command, or via a dropped or abandoned call) or until the call reaches an agent. When an agent becomes available in any split (skill) to which the call is queued, the following actions take place:

- Call begins alerting the agent.
- Call is removed from any other queues.
- Vector processing terminates.

If the entered split (skill) is one of the splits (skills) to which the call is already queued, the call is requeued at the new priority level. If the priority level specified is the same as the priority level at which the call is queued, the call remains in the same position in queue. The step is skipped, and vector processing continues at the next step if any of the following conditions are true:

- Desired split's (skill's) queue is full.
- Desired split (skill) is not vector-controlled.
- Desired split (skill) has no queue and also no available agents.
- Call has been previously queued to three different splits (skills).

NOTE:
A *route-to* to another VDN can be used to remove the call from the splits it is queued to if necessary. The steps in the routed-to vector then can be used to queue to other splits.

Answer Supervision Considerations

Answer supervision is returned (if not already returned) when the call is connected to an answering agent.

Feature Interactions

The *queue-to main split (skill)* command can access a AUDIX/Message Center/Server split (skill) in cases where a VDN is assigned as a coverage point. To enable this function, the split (skill) must be assigned as a vector-controlled hunt group.

For Look-Ahead Interflow, the command can be considered either a call acceptance vector command or a neutral vector command.

The command is considered a call acceptance vector command whenever one of the following is true:

- Call terminates to an agent
- Call queues to a split (skill)

The command is considered a neutral vector command when the call neither terminates nor queues.

No COR checking is carried out when a queue-to main step places a call to a split (skill).

CMS Interactions

R3 CMS: Calls queued via a *queue-to main split (skill)* command are tracked as CALLSOFFERRED and LOWCALLS/MEDCALLS/HIGHCALLS/TOPCALLS.

Split (skill) calls are reported in the standard reports according to the final disposition of the call.

The presence of the command in a vector enables the calls that are serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ‘‘ACD Calls,’’ ‘‘Split (skill) ACD Calls,’’ and ‘‘Avg Speed Ans.’’ If the call is also queued to other splits (skills), OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split (skill) to which the call queues, and “Flow Out’’ is reported (unless the split (skill) turns out to be the answering split (skill)). DEQUECALLS/DEQUETIME is tracked in the second and third splits (skills) if these splits (skills) are not the answering split (skill), and the call is reported as ‘‘Dequeued Calls’’ and ‘‘Dequeued Avg Queue Time.’’ However, if the second or third split (skill) is the answering split (skill), INFLOWCALLS is tracked in the split (skill), and the call is reported as “Flow In.’’

If the call abandons after the command queues the call to a split (skill), ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split (skill) to which the call is queued. The call is reported as ‘‘Aban Call’’ and ‘‘Avg Aban Time.’’ If the call is also queued to other splits (skills), DEQUECALLS/DEQUETIME is tracked in these splits (skills), and the call is reported as ‘‘Dequeued Calls’’ and ‘‘Dequeued Avg Queue Time.’’

R2 CMS: *queue-to main split (skill)* calls are shown in the standard reports according to the final disposition of the call.

An ACD call directed via the command and subsequently answered is tracked as ANSWERED/ANSDELAY and ACDCALLS. The call is reported as ‘‘No. ACD Calls,’’ ‘‘No. Calls Ans,’’ and ‘‘Avg Speed Ans.’’ If the call is also queued to other splits (skills), OUTFLOW is tracked in these splits (skills), and the call is reported as ‘‘Flow Out.’’ Calls directed via the command are tracked as ANSMAIN, and they are reported as ‘‘No. Calls Ans in Main.’’

If the call abandons after the command queues the call to a split (skill), ABNCALLS/ABNTIME is tracked for the vector, the VDN, and the first split (skill) to which the call is currently queued. The call is reported as ‘‘No. Aban Calls’’ and ‘‘Avg Aban Time.’’ If the call is also queued to other splits (skills), OUTFLOWCALLS/OUTFLOWTIME is tracked in these splits (skills), and the call is reported as “Flow Out.’’

BCMS Interactions

The total number of calls to the VDN that are queued via the command and then answered by an agent within a specified time period is tracked as ‘‘NUM ANS’’ in the VDN Report. The average time that calls spend in a vector before being connected via the command as an ACD call to an agent is tracked as ‘‘AVG TIME TO CONNECT’’ in the same report.

Route-to Command

```
route-to
```

Purpose

Routes calls either to a destination that is specified by digits collected from the caller or an adjunct (*route-to digits*), or routes calls to the destination specified by the administered digit string (*route-to number*).

Syntax

route-to digits with coverage <option>

route-to number <number> with cov <option> if **unconditionally**

route-to number <number> with cov <option> if **digit** <comparator> <digit>

Valid Entries

number: 1 to 16 digits (includes the Abbreviated Dialing (AD) special characters (*~p, ~w, ~m, ~s, ~W), *, #.*)

option: *n* (no), *y* (yes)

comparator: *=, <, <=, >, >=, <>*

digit: *0 through 9 or a single "#"*

Requirements

Route-to digits requires Call Prompting software.

Route-to number requires Basic Call Vectoring software. For the digit option, Call Prompting software must be installed. Comparators other than = (equals) are only available with Vectoring (G3V4 Enhanced).

Example

route-to digits with coverage *y*

route-to number *3300* with cov *n* if **unconditionally**

route-to number *473957* with cov *y* if **digit** >= *8*

Operation

The *route-to* command attempts to route a call to a set of digits collected from the caller or from an adjunct, or to the destination specified by the administered digit string.

For the route-to number ... if digit command, the call is conditionally routed to a specified destination according to a single digit entered by the caller. If the digit collected in the last *collect digits* command matches the specified comparison in relation to the administered digit, the command attempts to route the call to the specified destination.

The destination for a route-to command can be any of the following:

- Internal extension (for example, split/hunt group, station, etc.)
- VDN extension
- Attendant or Attendant Queue
- Remote extension (UDP/DCS)
- External number, such as a TAC or AAR/ARS FAC followed by a public or private network number (for example, 7-digit ETN, 10-digit DDD, etc.)
- Remote Access Extension
- Service Observing FAC

NOTE:
The VDN's Class of Restriction (COR) is used for calling permissions.

The *route-to digits* command fails if no digits are collected, and vector processing continues at the next vector step.

The *route-to number ... if digit* command fails if more than 1 digit is collected or if the digit comparison fails. Vector processing continues at the next command.

If the *route-to* command is successful, vector processing terminates. Otherwise, vector processing continues at the next vector command.

If the number expressed in the command is a system extension or an attendant group (and not a VDN), the system considers the step successful if one of the following conditions occurs:

- The endpoint is alerted.
- The endpoint has Call Forwarding or night service (hunt group) enabled, and the (night service) destination forwarded to is alerted; or, if off-premises Call Forwarding (UDP hunt night service), a trunk is seized.

The system then provides ringback to the caller, and vector processing terminates. However, if the call cannot complete successfully (for example, no idle appearance is available), vector processing continues at the next vector command.

If the number is a VDN extension, the following events occur:

- Vector processing terminates within the current vector.
- If the current VDN is administered with override, the new VDN overrides current VDN information.
- Processing of the vector associated with the VDN extension begins.

If the number is an AAR/ARS FAC plus digits, or if it is a remote UDP extension, standard AAR/ARS processing is performed to select the trunk group and outpulse the digits. If a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the FRL/COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is a TAC plus digits, and a trunk is seized, vector processing terminates, and the calling party hears feedback provided by the far end. Otherwise, the call cannot complete successfully (because no trunks are available, the COR is restricted, etc.), and vector processing continues at the next vector command.

If the number is any other number (such as an FAC other than an AAR/ARS or Service Observing), the command is unsuccessful, and vector processing continues at the next vector command.

Abbreviated Dialing special characters can also be used in the number field. Each of these characters instructs the system to take a different action when dialing reaches the point where the character is stored. The characters are as follows:

- ~*p* (pause)
- ~*w* (wait)
- ~*m* (mark)
- ~*s* (suppress)
- ~W (indefinite wait)

Each special character counts as two digits towards the maximum. The maximum number of digits for the command is 16.

The *route-to digits* command can be used to implement an automated attendant function.

Coverage

The optional coverage parameter determines whether coverage should apply during routing. If coverage applies, and if the digits entered are valid, the following occurs:

- Ringback is provided.
- Vector processing terminates.
- Normal termination and coverage are implemented.

NOTE:
For detailed information about the operation of the route-to command with or without coverage for the different destinations see "Operation Details for the Route-to Command" on page G-1.

Answer Supervision Considerations

Generally, answer supervision is provided when the destination answers the call. The exception to this involves incoming trunk calls routed to another non-ISDN-PRI trunk. Such calls provide answer supervision when the outgoing trunk is seized.

Feature Interactions

The *route-to* command may specify the AAR or ARS access codes. The COR associated with the VDN is used to determine the Partitioned Group Number (PGN) time-of-day routing chart. The PGN determines the choice or route tables used on a particular call.

The command may call the AUDIX extension. If this happens, the call is treated as a direct call to AUDIX, and the calling party may retrieve his or her messages.

If the call covers to a VDN, the command supports a remote AUDIX interface to a local hunt group extension that is assigned as a remote AUDIX hunt group. The ‘‘remote AUDIX hunt group’’ (which has no members and cannot be vector-controlled) forwards the call to the remote AUDIX destination in the same manner as when the hunt group is assigned as a point in the coverage path. A DCS link down condition for a call that covers to a VDN is treated as a direct call to AUDIX.

If the command is directed to a station with bridged appearances, the bridged appearance button lamps are updated.

The following destinations always result in a failure, and vector processing continues at the next step:

- Controlled trunk group
- Code calling FAC

- Facility test call
- TAAS access code
- Priority access code
- Loudspeaker paging access code
- Station Message Detail Recording (SMDR) account code
- Voice message retrieval access code

If the command is executed and Direct Outward Dialing (DOD) is in effect, the COR of the VDN is compared with the COR of the called facility to determine if the call is permitted. If access is not permitted, the command fails and vector processing continues. In the case where a COR requiring the entry of account codes is assigned to a VDN, and the command is executed by the associated vector, the command is unsuccessful, and vector processing continues at the next step.

The individual extension number assigned to an attendant console can be used as the command's argument.

A call processed by the command can wait in the individual attendant queue and is subsequently removed from vector processing.

The command can access both public and private networks.

If the command dials the attendant, and if the system is in night service, the call routes to the DID Listed Directory Number (LDN) night destination.

The command can place AAR/ARS calls that implement subnet trunking, which is the routing of calls over trunk groups that terminate in switches with different dial plans.

Authorization codes are disabled with respect to routing via VDNs. In other words, if authorization codes are enabled, and a *route-to* command in a prompting vector accesses AAR or ARS, and the VDN's FRL does not have the permission to utilize the chosen routing preference, no authorization code is prompted for, and the *route-to* command fails.

If the command routes the call without coverage to a display station, the station displays the following: ‘‘a = Originator Name to VDN Name.’’

If the command calls a station that is a member of a pickup group, the call can be picked up by another pickup group member.

Anytime a *route-to with cov n* command initiates a call over ISDN-PRI facilities and Look-Ahead Interflow is optioned, the call will be treated on a Look-Ahead basis. However, if the command is used with the *coverage yes* option in effect, unconditional interflow results.

For Look-Ahead Interflow, the *route-to* command can be considered either a call acceptance vector command or a neutral vector command. The command is considered a call acceptance vector command whenever one of the following is true:

- Command terminates to a valid local destination.
- Command successfully seizes a non-PRI trunk.
- Command execution results in a Look-Ahead Interflow call attempt, and the call is accepted by the far end switch.

The command is considered a neutral vector command whenever one of the following is true:

- Termination is unsuccessful.
- Trunk is not seized.
- Look-Ahead interflow call attempt is denied by the far end switch.

For a call that covers or forwards to a VDN, the *route-to with coverage y* command functions the same way as the *route-to with coverage n* command. For a covered or forwarded call, the coverage option for the command is disabled since such a call should not be further redirected.

A *route-to with cov y* to a station that has call forwarding activated is forwarded.

Service Observing can be initiated with Call Vectoring using the route-to command. See "Service Observing" on page 4-16 for detailed instructions.

NOTE:
Appendix G gives a detailed description of the feature interactions for the *route-to* number with and without coverage command.

CMS Interactions

R3 CMS: Tracking of the *route-to digits* command varies according to the destination successfully routed to, as follows:

Routed to Station or to Attendant

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
CONNECTCALLS/ CONNECTTIME	Other Calls Connect	answered calls on G3

Routed to Trunk

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out VDN Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTERFLOWCALLS / INTERFLOWTIME	VDN Flow-Interflow	
INTIME	Avg Time In Vector	

Routed to VDN

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out Vector Flow Out VDN Flow Out	1st split
DEQUECALLS/DE QUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
INFLOWCALLS	Vector Flow In VDN Flow In	new vector new VDN

Routed to Split or Hunt Group

Database Item	Report Heading	Notes
OUTFLOWCALLS/ OUTFLOWTIME	Flow Out	1st split
DEQUECALLS/ DEQUETIME	Dequeued Calls Dequeued Avg Queue Time	2nd/3rd splits
INTIME	Avg Time In Vector	
CALLSOFFERRED		new split
MEDCALLS/ HIGHCALLS		no priority/priority

NOTE:
For calls that ‘‘route to’’ a split or a hunt group and later intraflow to a station or to an attendant, OTHERCALLS/OTHERTIME are tracked in the vector and in the VDN tables.

Split calls are also shown in the standard reports according to the final disposition of the call.

Calls that route over an ISDN trunk are Look-Ahead Interflow calls. When a call attempts to ‘‘route to’’ an ISDN trunk (Look-Ahead Interflow), the LOOKATTEMPTS database item is tracked and reported as ‘‘Look-Ahead Interflow Attempts.’’ If the call successfully routes, LOOKFLOWCALLS/LOOKFLOWTIME are tracked and reported as ‘‘Look-Ahead Interflow Completions.’’ Interflow always occurs whenever the *with coverage yes* option is in effect.

The presence of the command in a vector enables the calls that are serviced by the vector to be vector-directed. When such a call is answered by an agent, the call is tracked as ACDCALLS/ANSTIME, and it is reported as ‘‘ACD Calls,’’ ‘‘Split ACD Calls,’’ and ‘‘Avg Speed Ans.’’ If the call is also queued to other splits, OUTFLOWCALLS/OUTFLOWTIME is tracked in the first split to which the call queues, and ‘‘Flow Out’’ is reported (unless the split turns out to be the answering split). DEQUECALLS/DEQUETIME is tracked in the second and third splits if these splits are not the answering split, and the call is reported as ‘‘Dequeued Calls’’ and ‘‘Dequeued Avg Queue Time.’’ However, if the second or third split is the answering split, INFLOWCALLS is tracked in the split, and the call is reported as ‘‘Flow In.’’

If the command directs a call to a destination, the BACKUPCALLS data base item is incremented, and the call is reported as ‘‘Calls Ans in Backup’’ and ‘‘Calls Handled/Backup.’’ The ‘‘Calls Ans in Main’’ report item is calculated by using the algorithm ACDCALLS - BACKUPCALLS.

A call abandoned after the command routes the call to a station or an attendant is tracked in the VDN tables as ABNCALLS/ABNTIME.

R2 CMS: Calls that ‘‘route to’’ successfully are tracked as OUTFLOW and reported as “Flow Out.” The vector and VDN tables are not credited with an answer or abandon for calls that route to a split. VECTIME is recorded for the vector, and it is reported as “Avg Time in Vector.”

Calls that ‘‘route to’’ a split are tracked as INFLOW and reported as ‘‘Flow In’’. Calls that ‘‘route to’’ a trunk are tracked as ROUTEDCALLS/ROUTEDTIME, but they are not shown on standard reports.

BCMS Interactions

A call advanced to another position via the command is tracked as ‘‘outflow’’ in the VDN Report. A call answered by an attendant via the command is also tracked as ‘‘outflow.’’

Stop Command

```
stop
```

Purpose

Halts the processing of any subsequent vector steps.

Syntax

stop

Requirements

Basic Call Vectoring or Call Prompting software must be installed.

Operation

After the *stop* command is processed, any calls already queued remain queued, and any wait treatment (for example, silence, ringback, music) is continued. On the other hand, any calls not queued are dropped under the same scenario.

If a TTR is allocated to the call, and if the *stop* command is encountered, the TTR is disconnected. However, current call processing continues (that is, the call is not dropped). The caller continues to hear the feedback that was provided before the *stop* command was encountered.

NOTE:
An implicit stop is processed following the last administered command in a vector.

Answer Supervision Considerations

The command has no effect on answer supervision.

Feature Interactions

For Look-Ahead Interflow, the command is considered a neutral vector command in all cases except when a call is dropped, then it is considered a denial.

CMS Interactions

R3 CMS: When the command or the end of the vector is encountered, vector INTIME is recorded. This is reported as ‘‘Avg Time in Vector.’’

R3V4 CMS: VDISCCALLS database item in the VDN tables pegs call that pass all the way through a vector without ever having been queued.

R2 CMS: Nothing is recorded when the command is encountered. When the disposition (that is, ‘‘answered’’ or ‘‘abandoned’’) of the call is known, VECTIME is recorded and reported as “Total Time in Vector.’’

Wait-time Command

```
wait-time
```

Purpose

Delays the processing of the next vector step if a specified delay time is included in the command's syntax. Also provides feedback (in the form of silence, ringback, or music) to the caller while the call advances in queue. The G3V4 Multiple Audio/Music Sources for Vector Delay feature allows a specified audio or music source to be selected when a call encounters a *wait-time* command. See "Multiple Audio or Music Sources" on page 4-6 for more information.

Syntax

wait-time <seconds> secs hearing <treatment>

Multiple Audio/Music Sources Syntax (G3V4 and later releases):

wait-time <seconds> secs hearing <extension> then <treatment>

Valid Entries

seconds: *0 through 998* (even numbers only); *0 through 8* when using the i-silent treatment.

treatment: silence, ringback, music, i-silent, continue.

With Multiple Audio/Music Sources, treatment refers to what the caller hears after the wait-time command times out, if the announcement finishes before time out, or if the call cannot be connected to the audio/music source extension. Continue is only a valid treatment with Multiple Audio/Music Sources. It indicates that the caller will continue to hear the alternate audio/music source until another vector command takes effect.

When music is indicated as a treatment, it refers to the system music, not an alternate music source.

In G3V4 and later releases, the tenant number of the active VDN determines the system music that will be heard by the caller.

The “i-silent” keyword is for use with adjunct routing-ADR/Lookahead Interflow applications. I-silent provides silence for the specified time, but it is neutral to Look-Ahead Interflow while all other wait treatments (even with 0 secs settings) provide acceptance.

extension: The valid extension number of an alternate audio/music source

Requirements

Basic Call Vectoring or Call Prompting software must be installed. Also, a music-on-hold port must be provided for the music treatment. Multiple Audio/Music Sources for Vector Delay requires that the “Vectoring (G3V4 Enhanced)” customer option be enabled.

Example

wait-time *224* secs hearing *music*

wait-time *120* secs hearing *54795* then *continue*

Operation

The specified feedback is given to the caller, and vector processing waits the specified time before going on to the next step. If the time specified is *0*, feedback is provided without any delay in the processing of the next vector step. The feedback given to the caller continues until any one of the following occurs:

- Subsequent vector step (containing *wait-time* or *announcement*) changes the treatment.
- Vector processing encounters a *disconnect* or *busy* command.
- Call is routed to another location or to a step that includes an announcement (for example, *collect digits*).
- Call is routed to another VDN.
- Call is delivered to a destination (starts ringing at an agent’s terminal).
- Switch receives a destination from the ASAI adjunct.

Answer Supervision Considerations

If the *music* or audio source treatment is included in the command, answer supervision is triggered. If the command is encountered and answer supervision was sent previously, the caller hears the treatment specified in the current command. If, for a CO trunk user, the command with *silence, ringback,* or *i-silent* treatment is encountered prior to answer supervision, the caller continues to hear ringback from the CO.

Feature Interactions

When the command is implemented with music as the treatment, the system-wide music-on-hold feature must be administered. Otherwise, the caller hears silence. When Tenant Partitioning is in use the tenant number of the active VDN determines the system music that is heard.

Feedback continues while a subsequent vector step queues for an announcement or for a TTR.

For Look-Ahead Interflow, the command is considered a call acceptance vector command in all cases, except i-silent, which is considered a neutral vector command.

NOTE:
An implicit wait of one second (with no change in the feedback to the caller) is provided after every seven vector steps if one of these steps does not suspend vector processing. (The following steps, if successful, do not suspend vector processing: *queue-to main split, check-backup split, goto step, goto vector* and *wait-time 0 seconds*. The following steps, if unsuccessful, also do not suspend vector processing: *check-backup split, route-to, adjunct routing*, and *messaging split*. The only commands that suspend vector processing are the following: *announcement, wait-time > 0 seconds, collect digits*, and *converse-on split*.

CMS Interactions

The command is not tracked on CMS or BCMS.

Criteria for Success/Failure of Call Vectoring Commands

The table appearing on the next several pages summarizes the success and failure criteria for various vector commands. Before you write or evaluate vectors, it is important to understand the information in this table.

NOTE:
If EAS is enabled, ‘‘skill’’ replaces ‘‘split.’’

Table A-11. Call Vectoring Command Success/Failure Criteria

Command	Success/Failure Criteria	Vector Processing Disposition
adjunct routing	Fails if any of the following are true: ■ VDN's COR does not permit routing to the adjunct-supplied destination. ■ TAC/ARS/AAR code is invalid. ■ Specified agent is not logged into the specified split for a direct agent call. ■ Local extension is not in the dialplan. ■ Invalid number was dialed.	Stop *wait-time* or *announcement* step (if present). Then continue vector processing with the next sequential step.
	Otherwise, succeeds.	Route the call and provide feedback.
announcement	Fails if specified announcement is unadministered, not recorded, or busied out.	Continue vector processing with the next sequential step.
	Otherwise, succeeds.	Play the announcement, then continue at the next sequential step.
busy	Always succeeds. (CO without answer supervision trunk callers will not hear the busy tone.)	Exit vector processing, then play the busy tone for 45 seconds before dropping the call. (Unanswered CO trunk calls receive 45 seconds of ringback.)

Table A-11. Call Vectoring Command Success/Failure Criteria

Command	Success/Failure Criteria	Vector Processing Disposition
check-backup split	Fails if any of the following are true: ■ Vector conditional is false. ■ Split's queue is full. ■ Split is not vector-controlled. ■ Call is already queued at the specified priority to the specified split. ■ Call is already queued to three different splits.	Continue vector processing with the next sequential step.
	Otherwise:	
	Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
	Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.
collect digits	Fails if any of the following are true:	
	■ Call originates from an outside caller who is not using a touch-tone telephone.	Call Prompting timer takes effect, command times out, and vector processing continues at the next vector step.
	■ No TTR is in the system, or the TTR queue is full.	Continue vector processing at the next step.
	■ Caller enters fewer digits than the maximum specified.	Call Prompting timer takes effect, command is terminated, and any digits collected prior to the timeout are available for subsequent processing.
	Otherwise, succeeds.	Continue vector processing at the next step.

Table A-11. Call Vectoring Command Success/Failure Criteria

Command	Success/Failure Criteria	Vector Processing Disposition
converse-on split	Fails if any of the following are true: ■ Converse split queue is full. ■ Converse split is not vector-controlled. ■ Auto-available split is in effect, and all agents are logged out by Redirection on No Answer (RONA).	Continue vector processing with the next sequential step.
	Otherwise: Succeeds, call is delivered to the converse split, and (if administered) digits are outpulsed to the VRU. The caller is connected to the VRU, the voice response script is executed, and (if necessary) digits are outpulsed to the switch.	Continue vector processing with the next sequential step.
disconnect	Always succeeds.	Play the announcement (if specified). Then drop the call.
goto step	Fails if the step condition is not met.	Continue vector processing with the next sequential step.
	Succeeds if the step condition is met.	Continue vector processing with the destination step.
goto vector	Fails if the step condition is not met.	Continue vector processing with the next sequential step.
	Succeeds if the step condition is met.	Continue vector processing with the first nonblank step of the destination vector.

Table A-11. Call Vectoring Command Success/Failure Criteria

Command	Success/Failure Criteria	Vector Processing Disposition
messaging split	Fails if any of the following are true: ■ Specified split is not an AUDIX split or a Message Server Adjunct (MSA) split. ■ Specified extension is invalid. ■ Messaging split queue is full. ■ Messaging split is not vector controlled and has no working agents (none logged in or all in AUX work mode). ■ Communications link with the messaging adjunct is inaccessible.	Continue vector processing with the next sequential step.
	Otherwise, succeeds.	Terminate vector processing.

Table A-11. Call Vectoring Command Success/Failure Criteria

Command	Success/Failure Criteria	Vector Processing Disposition
queue-to main split	Fails if any of the following are true: ■ Split's queue is full. ■ Split is not vector-controlled. ■ Call is already queued at the specified priority to the specified split. ■ Call is already queued to three different splits.	Continue vector processing with the next sequential step.
	Otherwise:	
	Succeeds, and the call is terminated to an agent.	Exit vector processing, and pass control to call processing.
	Succeeds, and the call is queued or requeued in the specified split at the specified priority.	Continue vector processing with the next sequential step.
stop	Always succeeds.	Exit vector processing. Control is passed to normal call processing. Any queuing or treatment in effect remains in effect. Call is dropped if not queued.
wait-time	Always succeeds.	Connect the specified treatment and pass control to the delay timer. Any feedback is continued until other feedback is provided.

NOTE:
Complete operational details for the *route-to* commands are provided in Appendix G.

Call Vectoring Management

B

Introduction

Call Vectoring management involves a number of different considerations and tasks. This appendix describes these considerations/tasks. Specifically, the following topics are discussed:

- Call Vectoring feature requirements
- Upgrading to a Call Vectoring environment
- Changing the vector
- Testing the vector

Implementation Requirements for the Call Vectoring Features

The tables appearing on the next several pages indicate the forms and the hardware required for implementing each of the Call Vectoring features. Details on completing the forms can be found in the *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

Table B-1. Basic Call Vectoring Requirements

Feature	Form(s)	Hardware
Basic Call Vectoring	■ Vector Directory Number Form ■ Hunt Group Form ■ Call Vector Form ■ Feature Related System Parameters Form	Announcement capabilities require either: ■ TN750 Integrated Announcement circuit pack(s), **or** ■ External announcement facility (analog announcements). Also, each analog announcement requires a port on a TN742, TN746B, or TN769 Analog Line circuit pack. **NOTE:** International equivalents for these circuit packs include the following: ■ TN467, TN468B (Australia and United Kingdom) ■ TN479 (Japan) ■ TN2135 (Italy) ■ TN2144 (Netherlands) ■ TN2149 (Belgium)

NOTE:
The Basic Call Vectoring and/or Call Prompting option(s) must be enabled on the System-Parameters Customer-Options form before the associated forms and the fields on the forms can be administered.

NOTE:
The TN750 Integrated Announcement circuit pack provides 16 ports for listening to announcements. On each port in G3i, there can be five listeners, while in G3r, there can be 255 listeners. Theoretically, then, 80 callers in G3i and 4080 callers in G3r can be listening to announcements on an integrated board at one time. The recording time provided is either 4:16 or 8:32, depending on the sampling rate employed (16K or 32K). An external announcement off an analog port can have only five simultaneous listeners in G3i and 128 simultaneous listeners in G3r. G3V4 and later releases provide for the installation of multiple TN750C Integrated Announcement circuit packs. Up to 5 boards can be installed in the G3i and 10 boards in the G3r. See “Recorded Announcements” in the *DEFINITY Communications System Generic 3 Feature Description*, 555-230-204 for more details.

Table B-2. Call Prompting Requirements

Feature	Form(s)	Hardware
Call Prompting	■ Vector Directory Number Form ■ Hunt Group Form ■ Call Vector Form ■ Vector Routing Tables Forms (G3V4 Enhanced)	Announcement capabilities require either: ■ TN750 Integrated Announcement circuit pack(s), **or** ■ External announcement facility (analog announcements). Each analog announcement requires a port on a TN742, TN746B, or TN769 Analog Line circuit pack. **NOTE:** International equivalents for these circuit packs include the following: ■ TN467, TN468B (Australia and United Kingdom) ■ TN479 (Japan) ■ TN2135 (Italy) ■ TN2144 (Netherlands) ■ TN2149 (Belgium)

NOTE:
The Basic Call Vectoring and/or Call Prompting option(s) must be enabled on the System-Parameters Customer-Options form before the associated forms and the fields on the forms can be administered.

Table B-3. G3V4 Enhanced Vectoring Requirements

Feature	Form(s)	Hardware
G3V4 Enhanced Vectoring	■ Vector Directory Number Form ■ Hunt Group Form ■ Call Vector Form	Requires no hardware in addition to that required for Basic Call Vectoring.

NOTE:
G3V4 or a later release is required for G3V4 Enhanced Vectoring. The following options must be enabled on the System-Parameters Customer-Options form before G3V4 Enhanced Vectoring capabilities can be used: Basic Call Vectoring and/or Call Prompting option(s); and "Vectoring (G3V4 Enhanced)."

Table B-4. Advanced Vector Routing Requirements

Feature	Form(s)	Hardware
Advanced Vector Routing	■ Vector Directory Number Form ■ Hunt Group Form ■ Call Vector Form	Requires no hardware in addition to that required for Basic Call Vectoring.

NOTE:
G3V4 or a later release is required for Advanced Vector Routing. The following options must be enabled on the System-Parameters Customer-Options form before Advanced Vector Routing conditionals can be programmed in a vector step: Basic Call Vectoring and/or Call Prompting option(s); "Vectoring (G3V4 Enhanced)"; and "Vectoring (G3V4 Advanced Routing)."

Table B-5. ANI/II-Digits Requirements

Feature	Form(s)	Hardware
ANI/II-Digits Routing	■ Vector Directory Number Form ■ Hunt Group Form ■ Call Vector Form ■ Trunk Group Forms ■ Vector Routing Tables Forms	Requires no hardware in addition to that required for Basic Call Vectoring.

NOTE:
G3V4 or a later release is required for ANI/II-Digits Routing. The following options must be enabled on the System-Parameters Customer-Options form: Basic Call Vectoring and/or Call Prompting option(s); “Vectoring (G3V4 Enhanced)”; and “Vectoring (ANI/II-Digits Routing).”

Table B-6. Look-Ahead Interflow Requirements

Feature	Form(s)	Hardware
Look-Ahead Interflow	■ Trunk Group Form (ISDN-PRI) ■ CPN Prefix Table Form	Existing ISDN-PRI hardware can be used for Look-Ahead Interflow ISDN-PRI connectivity to the receiving switch. Interconnecting facilities must be ISDN-PRI with no interworking (that is, call connections that use both ISDN-PRI and non-ISDN-PRI facilities to complete) for the full capabilities of the feature to be operational. Look-Ahead Interflow calls that interwork may interflow successfully, but the ability to do so on an intelligent basis will be lost as will the Look-ahead DNIS information.

NOTE:
The Look-Ahead Interflow, ISDN-PRI, Basic Call Vectoring and (usually) private network access options must be enabled on the System-Parameters Customer-Options form.

NOTE:
Both the sending switch and the receiving switch must have the Basic Call Vectoring and the Look-Ahead Interflow features active. In addition, Look-Ahead Interflow calls can connect ISDN-PRI switch-to-switch using private, public, or SDN facilities.

Table B-7. Adjunct Routing Requirements

Feature	Form(s)	Hardware
Adjunct Routing	■ Hunt Groups ■ Class of Restriction (for Direct Agent Calls) ■ Call Vector ■ Station ■ Station (ISDN-BRI-ASAI)	A TN556 ISDN-BRI circuit pack and a TN778 packet control must be in place. The latter provides packet bus control. Also, an adjunct/host processor must be in place to receive the request and select the route. A TN2198 two-wire BRI port circuit pack can be used in place of the TN556. In this case, an NT1 is also required.

NOTE:
The Basic Call Vectoring options must be enabled on the System-Parameters Customer-Options form.

NOTE:
The ACD, ASAI Interface, and ASAI Capability Groups must be enabled on the system.

NOTE:
The Packet Bus option (G3r only) must be enabled on the Maintenance-Related System Parameters form before associated ISDN-BRI forms and fields can be administered.

NOTE:
Direct Agent Calls (DACs) remain in queue for the agent if the agent does not have call coverage. Otherwise, DACs follow the agent’s coverage path after the no answer time interval expires.

Enabling the Vector Disconnect Timer

Call Vectoring makes available a Vector Disconnect Timer, which can be set for any amount of time between 1 and 240 minutes inclusive. The timer is enabled by selecting the timer field in the Feature-Related System-Parameters form. The timer is started when vector processing is started. Once the timer runs out, the call is dropped. The timer is canceled when vector processing terminates.

Enabling the timer allows queued calls that have not been answered within a determined amount of time to be dropped. For more information, refer to *DEFINITY Communications System Generic 3 Implementation*, 555-230-653.

Upgrading to a Call Vectoring Environment

If you are already equipped with ACD and want to use Call Vectoring, the ACD environment must be upgraded to a Call Vectoring environment. This involves installing VDNs, vectors and hunt groups for the desired Call Vectoring feature(s).

The set of guidelines that follows is intended to serve as a general procedure for upgrading to a Call Vectoring environment. For complete details of this process, refer to *DEFINITY Communications System Generic 3 Implementation*, 555-230-655.

1. Verify the vector options on the Customer Option Form.

   NOTE:
   This is always done by AT&T personnel.

2. Add the VDNs.
3. Evaluate the number of queue slots assigned to each split. Usually, you want to assign enough queue slots to allow all calls processed by Call Vectoring to be queued. (See the considerations for Basic Call Vectoring in Appendix C for more details.)
4. Change hunt-groups to be vector-controlled.
5. Administer the vectors and at least one test hunt group.
6. Test all of the vectors to be installed.
7. Change the trunk groups, night destinations, etc., to use the VDNs.

Changing and Testing the Vector

Vectors currently being used to process calls should not be changed because changes would have an immediate and uncertain effect on the treatment that the calls are receiving. Instead, a new vector should always be written.

In testing the vector, you should not consider the entire vector at once. Rather, you should first figuratively divide the vector into portions, then test each of these portions until the entire vector is tested.

After the new vector is thoroughly tested, the vector should be brought into service by changing the VDN to point to the new vector.

The set of following guidelines is intended to serve as a general procedure for changing and testing vectors. For complete details of this process, refer to *DEFINITY Communications System Generic 3 Implementation*, 555-230-655.

1. Check that a current version of the translation data is available.
2. Create a new VDN that points to the new vector. This VDN, which is temporary, is necessary to test the new vector.
3. Administer the new vector. Vector commands should be added and tested, one command at a time, starting with the first command. Be sure that each line is correct before proceeding to the next one.
4. Test the new vector with the new VDN. This ensures the new vector will function correctly when the vector is installed.
5. Install the new vector by changing the old VDN's vector assignment so that the VDNs now point to the new vector. Calls that are already being processed by the old vector will continue to be handled by that vector until the vector terminates vector processing.
6. Once all the calls are handled, remove the old vector and the VDN that was used for testing.

Considerations for the Call Vectoring Features

C

Introduction

This appendix contains several lists of considerations you should bear in mind when using the Call Vectoring features. These considerations are intended to help you get the highest degree of productivity from Call Vectoring.

NOTE:
If EAS is optioned, ''skill'' replaces ''split.''

Basic Call Vectoring Considerations

The following are considerations you should keep in mind when working with Basic Call Vectoring:

- Make the split queues large enough so that all incoming calls queue and are not dropped. If a queue is too small, a *queue-to main split* or a *check-backup split* command might fail to queue a call due to a lack of available queue slots. Accordingly, it is also always a good practice to include in the vector a step that checks a split's queue before queuing occurs and a corresponding step that provides alternate treatment if the queue is full. To check the queue size, you can use a *goto* command (for example, *goto Step 5 if calls-queued in split 20 pri l > 30*). The alternate treatment, which, if needed, is usually accessed by the *goto* command that checks the queue size, can queue the call to a backup split, make an unconditional Look-Ahead Interflow attempt, provide a busy signal, etc.
- A default treatment or a *route-to* destination step should be supplied after a *route-to* command in case the first destination is unavailable.

- Calls should not be queued to an unstaffed split (unless this is intended by the customer) without some alternate treatment.
- Interflow calls should not be permitted to interflow back and forth between a remote switch vector and a local switch. This process could cause a single call to use up all available trunks.
- After an announcement is provided, the audible feedback (such as music) should be re-attached.
- For ease-of-use purposes, each specific vector function or operation should be included in a separate vector and linked via one or more *goto vector* commands.
- In creating a vector, commands can be chosen and arranged in a manner such that answer supervision is delayed as long as possible. This should be done to keep down the service cost.
- The caller should always be provided with initial feedback (usually ringback).
- Direct agent calls merit special attention because such calls can affect call queuing. Although direct agent calls take up a queue slot, they are not always reported as using such a slot on CMS/BCMS reports (discussed in Appendix F). For example, a direct agent call is never counted toward the total of queued calls within a split (that is, the *calls-queued* test condition has no effect on this type of call).
- If it is necessary for a caller to hear an entire CONVERSANT script before talking to an agent, the caller should not be queued until after the *converse-on* step is executed.
- Audible feedback should be provided prior to a *converse-on step* whenever a large number of digits are to be outpulsed to the VRU.

Call Prompting Considerations

The following list includes considerations you should keep in mind when working with Call Prompting:

- To enter the digits requested via the *collect digits* command, outside callers must have a touch-tone telephone. For such callers using rotary dialing, a 10 second inter-digit timeout takes effect, and the *collect digits* command is omitted. As a precaution, a default treatment (for example, *route-to* attendant command, *queue-to main split* command) should always be provided in the vector script unless the script is created exclusively for users of touch-tone telephones.
- If a caller does not enter the full number of digits specified in the *collect digits* step, a 10~second timeout occurs. Thereafter, vector processing continues with subsequent vector steps, and an attempt is made to process the call using the digits that have been collected. If the digits entered do not represent a valid destination, and if Automated Attendant

is being implemented via a *route-to digits* command, the *route-to digits* command fails, and vector processing continues at the next step, which should be a default treatment.

- It may be prudent to take steps in case a *route-to* attendant command fails, such as providing a disconnect announcement.
- From time to time, all of the system's touch-tone receivers might be in use. As a result, you should avoid starting your main vector with a *collect digits* command, since the caller on a DID or tie trunk in this case receives no audible feedback if he or she has to wait for a receiver to become available. Accordingly, it is a good practice to include some treatment (for example, a *wait-time 0 seconds hearing ringback* step) before the initial *collect digits* step.

Look-Ahead Interflow Considerations

The following are considerations you should keep in mind when working with Look-Ahead Interflow:

- Never interflow to a remote vector that in turn might interflow back to the same local vector. This could cause a single call to use up *all* available trunks.
- The *oldest-call-wait* test condition should not be used in LAI vectors. OCW corresponds to the very next call to be answered and, as such, this test condition gives no information on the current state of call overload (for example, if OCW = 30 seconds, all we know from this is that the queue was overloaded 30 seconds ago). In place of *oldest-call-wait*, use the EWT conditional. See "Expected Wait Time (EWT)" on page 6-2.
- If an LAI call attempt is accepted by a step that contains a *queue-to main, check-backup split,* or *route-to* command, there is a small but finite interval during which the call could be answered by an agent at the sending switch before notification of ‘‘acceptance’’ is received by the sending switch. In this case, the caller would be connected to the agent at the sending switch, while the agent at the receiving switch might receive a ‘‘phantom’’ call. For this reason, you should consider using a short *wait-time* or *announcement* step at the receiving switch to allow the call to be accepted and taken out of queue at the sending switch. If call acceptance is to be based on available agents, use of a *wait-time* > 0 seconds or an *announcement* is not recommended. A *wait-time* with 0 seconds of silence might be useful in this case.
- When an LAI call attempt is made, the TTR (if attached) is disconnected, and any dial-ahead digits are discarded. This implies that a subsequent *collect digits* command would require that the TTR be connected.
- Be sure the feedback provided by the receiving switch after a successful LAI attempt is consistent with what the caller has already received.

- It is perfectly acceptable for a vector to route a call over an ISDN-PRI facility to a destination that is not a VDN. In such a case, the sending switch treats the call like a Look-Ahead Interflow call. Generic ISDN processing at the receiving switch causes the call to be accepted. The DNIS name is ignored.
- If a Look-Ahead Interflow call terminates to a VDN on a receiving switch where the Look-Ahead Interflow option is *not* enabled, intelligent interflow still results. However, any relevant DNIS information is ignored, and intelligent interflow to far-end switches is not possible.
- The LAI timeout in the sending switch occurs after two minutes.
- T-1 equipment might modify the ISDN D-channel that is used for Look-Ahead Interflow. If multiplexors are introduced into the ISDN-PRI circuit, bit compression and echo cancellation must be turned off for the D-channel.

Adjunct Routing Considerations

The following are considerations you should keep in mind when working with Adjunct Routing:

- Depending upon your application, you may want to include a second *adjunct routing* step in your vector in case the first such step fails.
- If you include an *announcement* step immediately after an *adjunct routing* step, be sure the announcement does not contain any information essential to the caller (such as further instructions) since the step following the *adjunct routing* step immediately terminates the moment the switch receives a destination from the ASAI adjunct.
- If you include a *wait-time* step after an *adjunct routing* step, it is a good idea to specify either *ringback* or *music* (and not *silence*) as the feedback. If the caller does not hear any feedback, he or she might give up on the call and hang up.
- The second step after the *adjunct routing* step could (and, in many cases, should) be implemented as a default treatment in case the host application or ASAI link is down. The step containing this default treatment (for example, *route-to number 0 if unconditionally*) executes immediately if the ASAI link is down and if the step is preceded by either a *wait-time* or an *announcement* step. On the other hand, if the host application is down, the default step executes only if the application does not respond with a route within 20 seconds.

VDN Return Destination Considerations

The VDN Return Destination feature allows an incoming trunk call to be placed back in vector processing after all parties, except the originator, drop. This feature is activated through switch administration of the VDN form. It is an optional system feature, and as such, it must be optioned on the System-Parameters/Customer-Options form.

A new field added to the VDN form will allow the user to enter a VDN extension as a Return Destination. In this section, the VDN which has the Return Destination field administered will be called the "VDN with this feature active." The Return Destination VDN (the one specified in the new field) will be referred to as the "Return Destination."

Every incoming trunk call which is processed through a VDN with this feature active will be placed back in vector processing when all parties on the call, except the originator, drop. For this feature, the "originator" is the incoming party which originated the call at the time the call entered the VDN with this feature active.

The VDN that the call will be placed in (when the originator is the only remaining party) is determined by the "Return Destination". This VDN may be the same or different than the original VDN.

This feature is used to keep the call active and give the caller the opportunity to signal the need for sequence dialing (by entering a "#"). There are two ways this can happen:

1. When the destination drops on its own (after having answered), the call will go to the Return Destination which will have a "collect digits" vector step. This step will try to collect the "#" sign entered by the caller.
2. When the call is not answered, the caller enters the "#" to request sequence calling (this "#" will be collected by the ASAI-Requested Digit Collection feature). This "#" is reported to the adjunct. The adjunct requests the third_party_drop (or third_party_end_call) for the destination, and at that point the call goes to the Return Destination.

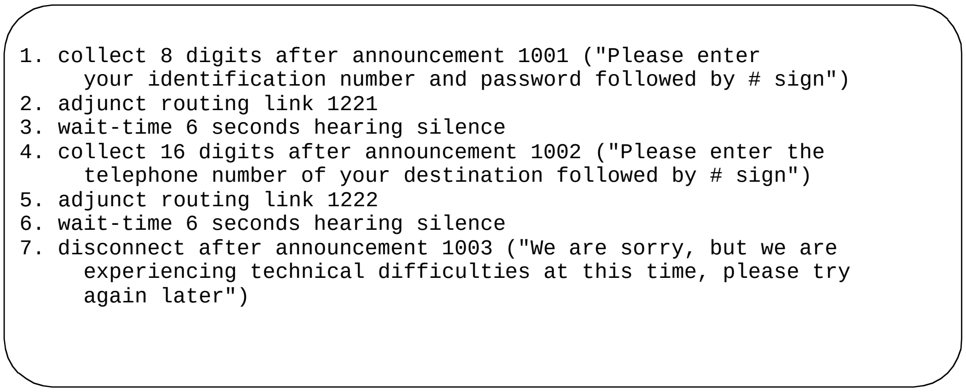
The VDN Return Destination and ASAI-Requested Digit Collection features may be used independently, with the following rules:

1. If there is no ASAI request to collect digits, but a Return Destination is provided: when all parties, except the originator, drop, the switch will route the call with only one party active (the caller) to the Return Destination. At this point, the call enters vector processing for the VDN specified by the Return Destination.
2. If a request is made to collect digits but there is no Return Destination provided: the switch will collect the digits and pass them on to the ASAI adjunct. It will be up to the adjunct to take action. However, if the action

taken by the adjunct is to drop one party on the call, the switch will drop the other party as well and clear the call (it cannot retain a call with only one party, if there is no Return Destination for further processing).*

User Scenario — Remote Access with Host Provided Security

A customer may use the VDN Return Destination feature to provide a more flexible remote access feature together with host based call security. The remote user/caller does not have to call back into the switch when multiple destinations need to be reached nor does the caller have to enter his/her identification every time a new destination is desired. For example, a customer can program the following vector that is accessed by dialing a VDN that has a Return Destination administered.



```
1. collect 8 digits after announcement 1001 ("Please enter
     your identification number and password followed by # sign")
2. adjunct routing link 1221
3. wait-time 6 seconds hearing silence
4. collect 16 digits after announcement 1002 ("Please enter the
     telephone number of your destination followed by # sign")
5. adjunct routing link 1222
6. wait-time 6 seconds hearing silence
7. disconnect after announcement 1003 ("We are sorry, but we are
     experiencing technical difficulties at this time, please try
     again later")
```

Figure C-1. Sample Return Destination Vector with Remote Access

In this scenario, a remote caller will call into the switch by dialing the VDN administered with the Return Destination. The vector executed will prompt the caller to enter an identification number and a password that will be passed, via the adjunct routing vector command, to the host for validation. The host can keep track of invalid attempts or decide to de-activate or activate certain identification numbers based on customer set criteria.

After the host based security is passed (the host sends an Abort to cancel the switch Route request; otherwise, the host routes the call to an exception destination/VDN), the switch will collect digits for the destination that the caller wants to reach (vector step 4 above). The host receives the number entered by the caller (vector step 5 above) and validates the entered number to check if the caller is allowed to reach the specified destination. If so, the host routes the call to the desired (dialed) destination.

If the host security is not passed, the host will route the call to an appropriate alternate destination (e.g., announcement with security violation message) and log the invalid call attempt.

If the host is not available, the call will be disconnected after an announcement (vector step 7 above).

After the called destination disconnects from the call, the caller can remain on the line to be connected to the Return Destination. A sample Return Destination vector is as follows:

```
1. collect 16 digits after announcement 1002 ("Please enter the
     telephone number of your destination followed by # sign")
2. adjunct routing link 1221
3. wait-time 6 seconds hearing silence
4. disconnect after announcement 1003 ("We are sorry, but we are
     experiencing technical difficulties at this time, please
     try again later")
```

Figure C-2. Sample Return Destination Vector with Disconnect

The caller, once connected to the Return Destination, can enter a second destination/phone number to connect to. The host performs the same validation on the destination number as in the first destination and routes the call as appropriate (destination entered by caller or alternate destination). Note that the host can also provide reports on all the destinations and times reached by each remote user.

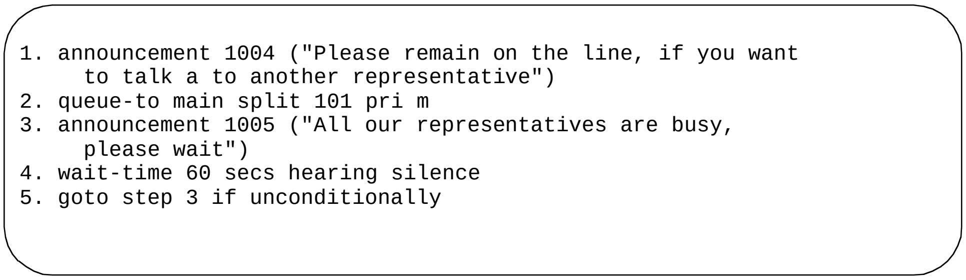
In the Return Destination vector, it is recommended that the first vector command give the caller the opportunity to disconnect from the call rather than immediately routing the call to some destination. If the call was immediately routed and then the caller decided to hang-up, the destination that the call was routed to would ring, alerting the called party, but then no one would be on the line at the other end (this could be confusing to customers, and could be misinterpreted as a problem with the feature). Vector commands such as "wait", "collect after announcement", and "announcement" can provide the caller with the opportunity to disconnect before the call is routed. As an example, an "announcement" command with the recording "Please hang-up to end your call, or remain on the line if you wish to place another call" instructs the caller to disconnect, before the call is routed.

User Scenario — Saving in Trunk Facilities Between Call Centers

A customer can also use VDN Return Destination to return a call to a local agent after the call is transferred to a remote destination (call). This will eliminate the need for the remote agent to transfer the caller back to a local agent and will save in switch trunk facilities, since each time the call is transferred back to a local agent an additional trunk is being used by the call.

For example, calls can be received at the local call through a VDN that has the return destination administered. These calls will be delivered to an agent on the local switch. If the local agent transfers the call to a remote destination (because the caller needed to talk to an agent on the remote switch), the call will return to the Return Destination after the remote switch drops the call. The remote switch agent must inform the caller to remain on the line after they are finished and the remote agent just needs to disconnect from the call (hang up).

The Return Destination for this scenario should include an "announcement" vector command at the beginning to inform the caller to disconnect from the call, if they do not want to be reconnected to an agent on the local switch. A sample Return Destination vector will be as follows:



```
1. announcement 1004 ("Please remain on the line, if you want
     to talk a to another representative")
2. queue-to main split 101 pri m
3. announcement 1005 ("All our representatives are busy,
     please wait")
4. wait-time 60 secs hearing silence
5. goto step 3 if unconditionally
```

Figure C-3. Sample Return Destination Vector with Announcement

Troubleshooting Vectors

D

Introduction

This chapter is intended to serve as a troubleshooting guide for Call Vectoring. The first part of the chapter includes two tables that indicate and explain unexpected operations within Call Vectoring that the customer may encounter. The first table focuses on the Call Vectoring features, while the second table focuses on the Call Vectoring commands. The second part of the chapter contains a table that focuses on *converse-on* command debugging. Finally, the third part of the chapter contains procedures for tracking many of the unexpected operations within Call Vectoring that are discussed in the two tables.

NOTE:
If EAS is optioned, "skill" replaces "split."

Unexpected Feature Operations

The table in this section indicates and explains unexpected operations within the Call Vectoring features the customer may encounter.

NOTE:
For solutions to these unexpected operations, refer to Chapters 4 through 9 and to Appendix A, Appendix C, and Appendix G in this guide.

Table D-1. Unexpected Feature Operations

Feature/Area	Customer Observation(s)	Cause(s)
General Vector Processing	Vector stuck.	1,000 steps executed. No default treatment in the vector.
	Audible feedback lasts longer than the delay interval.	Last vector step. Queuing for an announcement. Queuing for a touch-tone receiver for a *collect digits* step.
Look-Ahead Interflow	Agent receiving phantom call.	Agents on both switches become available simultaneously. (Avoid by including at the beginning of the receiving switch vector a short *wait-time* or *announcement* step.)
	No Look-Ahead Interflow attempts accepted.	No trunks. PRI network failure. Insufficient FRL.
	All Look-Ahead Interflow attempts accepted.	Look-Ahead Interflow attempts are interworking off of one of the following: ■ Interworking off of the PRI network ■ Receiving vector not designed for conditional acceptance ■ *route-to with coverage yes* command was used to interflow
	Look-Ahead DNIS name not displayed.	Look-Ahead Interflow not optioned at the receiving switch.

Unexpected Command Operations

The following table indicates and explains the unexpected operations the customer may encounter in using the Call Vectoring commands.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
adjunct routing	Step skipped (that is, default treatment).	Invalid link extension. No trunks available. COR/FRL restricted. Timeout. (Application did not respond within the time specified in the *wait-time* command and/or within the time length of the recorded announcement.) Digit string inconsistent with networking translation. ASAI link down. Invalid route destination returned from adjunct.
	Busy tone.	Busy local destination has no available coverage points.
	Network reorder or intercept.	Digit string supplied by adjunct inconsistent with public network translation. Digit string inconsistent with networking translation.
	Intercept or reorder tone heard.	Vector processing succeeded routing off switch, but a problem has occurred before routing to its final destination.
	All trunks busy on a quiet system.	Two switches treating each other as backup switch.
	Step skipped.	Port Network (PN) link down.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
announcement	Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Link to TN750 down.
	Extra delay before hearing announcement.	Announcement queue full. All integrated announcement ports busy. Analog announcement busy.
	Vector processing stops.	Analog announcement does not answer.
	Listening to silence after announcement.	Announcement is the last step.
	Incomplete announcement.	Agent becomes available. Previous *adjunct routing* step succeeds.
busy	Ringback heard instead of busy tone.	Unanswered CO trunk.
check-backup split	Call does not enter queue or terminate to agent.	Step condition not met.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
check-backup split and *queue-to main split*	Call does not enter queue or terminate to agent.	Queue length specified on the hunt group screen has been exceeded. Invalid split. Split not vector-controlled. Already queued to three different splits. No queue. Queue or check backup status indicates space when queue is full due to direct agent calls.
	Call apparently answered in wrong order.	Call being requeued at different priority. Call superseded by higher priority call, including direct agent call.
collect digits	Announcement not heard while waiting for digits, but network billing indicates that the call was answered.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress. Dial ahead digit exists.
	Collect step and announcement skipped.	TTR TN744 not in system. Link to PN that has TN744 is down. TTR queue full for TN744.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
collect digits (Continued)	Delay before hearing announcement.	All TTR TN744 ports busy, but space in queue.
		Announcement queue full.
		All integrated announcement ports busy.
		Analog announcement busy.
	Vector stuck.	Analog announcement does not answer.
	Dial-ahead digits not recognized.	Dial-ahead digits entered prior to first collection step.
		Call has been transferred.
		LAI attempt has been made.
		TTR has been released.
		24 digits have already been provided.
		Call Prompting timeout since the last digit was entered.
	Vector processing halted at collect step; announcement heard again upon return.	Call put on hold, transferred, or conferenced.
	Insufficient digits collected; call routed to intercept.	Caller dialed *#* too soon.
		Caller dialed *** without reentering correct digits.
		Call Prompting interdigit time-out.
	Caller information button denied.	No digits were collected.
		Display not in Normal mode.
	Collect announcement not heard and first collected digit incorrect.	System does not contain all TN748C Vintage 5 (or later) boards.
	Incomplete announcement.	Agent becomes available.
		First digit dialed.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
converse-on split	VRU script not executed.	Queue full. No queue. Invalid split. Split not vector-controlled. VRU down.
	"Ani" digits not passed.	ANI not available.
	"Qpos" digits not passed.	Call not queued to a nonconverse split.
	No data returned from VRU.	No TTRs available on DEFINITY.
	VRU script terminated prematurely.	Agent becomes available. VRU script attempted to transfer the call.
	Wait digits not passed	Call not queued or no working agents in splits where call is queued.
disconnect	Announcement not heard.	Announcement board not present. Announcement not administered. Announcement not recorded. Announcement being rerecorded. All ports busied out. Announcement restore in progress.
	Extra delay.	Announcement queue full. All integrated announcement ports busy. All analog announcements busy.
	Vector stuck.	Analog announcement does not answer.
goto step	Branch is not made to the specified step.	Step condition not met. System time not set.
goto vector	Branch is not made to the specified vector.	Step condition not met.
	Vector stuck.	Goto vector with no steps or with all failed steps.

NOTE:
Refer to the "Converse Command Debugging" section later in this appendix for more details on *converse-on* command debugging.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
messaging split	Vector stuck (with ringback).	Extension unknown to AUDIX.
	Step skipped, no message left.	AUDIX link down.
		DCS link to remote AUDIX down.
		All DCS trunks busy.
		Queue for AUDIX voice ports is full.
	Vector stuck (with busy).	Remote AUDIX link down.
	Messages not found.	Message extension is *none* (message is left for VDN that accessed the vector).
	Delay before AUDIX answers.	All AUDIX ports busy, but space in queue.
	Busy tone.	Queue for AUDIX voice ports is full.
	Step skipped.	Split not AUDIX split anymore.
route-to	Step skipped (that is, default treatment).	Invalid local extension.
		No trunks available.
		COR/FRL restricted.
		Digit string inconsistent with networking translation.
		Busy local destination (route to digits without coverage and route to number).
		No digits collected.
	Network reorder.	Digit string inconsistent with public network translation.
	Intercept or reorder tone heard.	Vector processing succeeded routing off switch, but a problem has occurred before routing to its final destination.
	All trunks busy on a quiet system.	Two switches treating each other as a backup switch.
stop	Call dropped.	Call not queued when vector processing stops.

NOTE:
Complete operation details for the *route to* commands are presented in Appendix G.

Table D-2. Unexpected Command Operations

Command Step	Customer Observation(s)	Cause(s)
wait-time	Audible feedback longer than delay interval.	Queuing for an announcement or for a TTR.
		Stop command executed.
	Audible feedback shorter than delay interval.	Previous *adjunct routing* step succeeds.
		Agent becomes available.
	Music not heard.	No music port administered.
		Music source disconnected or turned off.
	Alternate audio/music source not heard	Announcement board not present.
		Audio/Music source not administered.
		Audio/Music source not recorded.
		Audio/Music source being rerecorded.
		All ports busied out.
		Announcement restore in progress.

Converse Command Debugging

The following table is intended to help your troubleshooting efforts with the *converse-on* command.

NOTE:
Refer to Appendix H for details on the call flow for converse-VRI calls.

Table D-3. Converse Command Debugging

SYMPTOM	CAUSES	EVIDENCE
PLACING A CALL:		
Converse step skipped.	VRU down (RONA).	Vector event.
	Split queue full	Vector event.
Call stuck in converse.	VRU port doesn't answer, RONA not used.	Check split administration.
	VRU down, RONA leaves call in queue.	Check split status.
DATA PASSING:		
First set of digits not collected.	Converse first delay too short.	Check administration.
	No ANI available.	Vector event.
	No digits collected.	Vector event.
	Call not queued (qpos).	Vector event.
	Expected wait time not available	Vector event.
	VRU timed out awaiting first digit.	VRU error log/trace.
	VRU first digit timeout too short.	Check VRU script.
		Check converse first data delay.
	Faulty hardware.	Diagnostics
Second set of digits not collected.	VRU digit count on first prompt in VRU script does not include "#."	Check VRU script.
	Converse second delay too short.	Check administration.
	No ANI available.	Vector event.
	No digits collected.	Vector event.
	Call not queued (qpos).	Vector event.
DATA PASSING: (Continued)	Expected wait time not available because call is not queued or the splits/skills that the call is queued to are not staffed	Vector Event

Table D-3. Converse Command Debugging

SYMPTOM	CAUSES	EVIDENCE
	VRU timed out awaiting first digit.	
	VRU error log/trace.	
	VRU first digit timeout too short.	Check VRU script. Check converse second data delay.
	Inter-digit timeout too short on first prompt and collect.	Check VRU script.
	Faulty hardware.	Diagnostics.
Digits incomplete.	Converse data delay too short.	Check administration.
	Faulty hardware.	Diagnostics.
Second set of digits is the same as the first digits passed.	VRUs first prompt timed out.	Check administration.
	Faulty hardware.	Diagnostics.
DATA RETURN:		
No digits returned to the DEFINITY switch.	Flash not recognized by switch.	VRU error log/trace.
		Check flash timing on VRU.
	Converse data return FAC not administered.	Check administration.
	VRU does not return FAC.	VRU script. Transfer attempt vector event.
	VRU returns incorrect FAC.	VRU script. Transfer attempt vector event.
	Digit timeout during FAC.	Transfer attempt event.
	Converse data return FAC overlaps with other entries in the dialplan	Check dialplan.
	Faulty hardware.	Diagnostics.
Not all digits returned to the DEFINITY switch.	Digit timeout after FAC.	None unless VRU logs being dropped by DEFINITY.
	Overflow of Call Prompting buffer	Vector Event.
	Faulty hardware.	Diagnostics.
Collect announcement not heard.	Too many digits returned by VRU.	Check VRU script.
	Faulty hardware.	Diagnostics.

Tracking Unexpected Vector Events

If you have an SAT terminal, you can display unexpected vector events. A vector event is an error that results from resource exhaustion or from faulty vector programming, rather than from a DEFINITY software error. For example, failures involving the *route-to* command are usually due to an invalid extension entered by the user.

By displaying vector events, you can do the following:

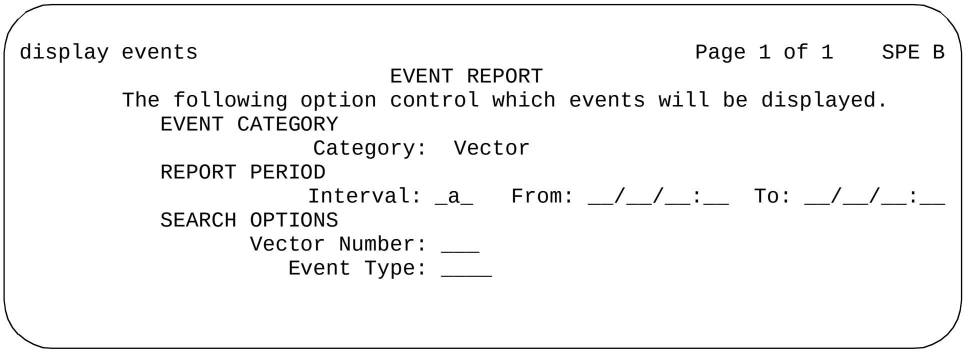
- Diagnose and correct each Call Vectoring problem, as indicated by its corresponding vector event, and thereby
- Eliminate the need for a technician to make on-site visits to do the same.

The following sections explain how you can troubleshoot by tracking unexpected vector events.

Display Events Form

The first step is to initiate the display of vector events. You do this by entering the *display events* command at the *enter command* prompt.

Once the command is entered, the *Display Events Form* appears on the screen. The form appears as follows:



```
display events                                       Page 1 of 1    SPE B
                             EVENT REPORT
        The following option control which events will be displayed.
           EVENT CATEGORY
                       Category:  Vector
           REPORT PERIOD
                      Interval: _a_   From: __/__/__:__  To: __/__/__:__
           SEARCH OPTIONS
                  Vector Number: ___
                     Event Type: ____
```

Figure D-1. Layout of Display Events Form

The following list indicates the options on the form, comments on these options, and also discusses the field(s) within each option.

- **EVENT CATEGORY**. This option is intended to indicate the class of logged events to be displayed. For our purposes, the default value *Vector* automatically appears in this display-only *Category* field. The value *Vector* indicates that only vector events will be displayed.

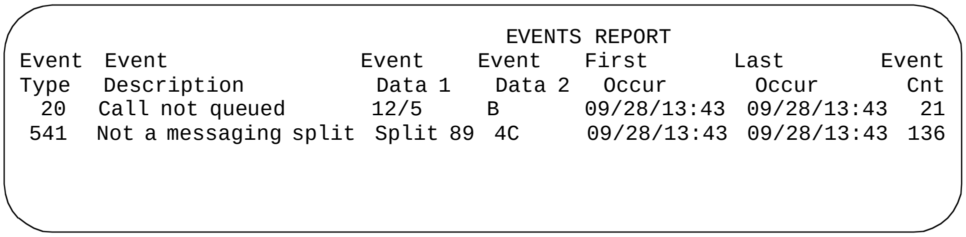
- **REPORT PERIOD**. This option allows you to specify a report period. This period consists of an *Interval* field, a *From* date/time stamp, and a *To* date/time stamp. Valid entries for the *Interval* field include *(h)our*, *(d)ay*, *(w)eek*, and *(a)ll*. Both stamps consist of a series of numbers that represent a period of time, as follows: *1 through 12* (month), *1 through 31* (day), *0 through 23* (hour), *0 through 59* (minutes). If the field and stamps are populated, only the vector events that occurred within report period specified are displayed. Otherwise, all vector events are displayed regardless of when they occurred.
- **SEARCH OPTIONS**. This option contains two fields, *Vector Number* and *Event Type*.

  Vector Number allows you to specify a vector number. If this field is populated, only vector events that are associated with this vector number are displayed. Otherwise, all vector events are displayed regardless of the vector number with which they are associated.

  Event Type allows you to specify the number associated with a particular type of vector event. This number may range from *0* to *999*. If the *Event Type* field is populated, only vector events of the type indicated are displayed. Otherwise, all vector events are displayed regardless of type.

Display Events Report

After you complete the *Display Events Form*, you can generate the *Display Events Report* by submitting the display request and hitting the Enter key a second time. A sample report appears as follows:



```
                                                EVENTS REPORT
Event  Event               Event    Event   First        Last        Event
Type   Description          Data 1   Data 2  Occur        Occur        Cnt
  20   Call not queued      12/5     B       09/28/13:43  09/28/13:43   21
 541   Not a messaging split Split 89 4C      09/28/13:43  09/28/13:43  136
```

Figure D-2. Display Events Report

The *Display Events Report* provides details of all the logged vector events that meet the selection criteria supplied by the user. The following list identifies and discusses the fields in the report.

- **Event Type** contains a unique number between *0* and *999* that identifies the type of vector event that occurred.
- **Event Description** contains text that describes the vector event.

- **Event Data 1** is a 9-character fields that contains data in one of two formats:
  - *<number1>/<number2>* (for example, *12/5*), where *<number1>* is the vector number associated with the vector event, and where *<number2>* is the step number associated with the vector event. This format is used for events to which an event type in the range of 0 through 499 is assigned.
  - *Split<number>* (for example, *Split 89*), where *<number>* is the split associated number associated with the vector event. This format is used for events to which an event type in the range of 500 through 999 is assigned.
- **Event Data 2** is an 8-character field that contains additional data encoded as a hex number (for example, *4C*). This number serves as a call identifier. If two or more events with an identical identifier occur at about the same time, it can be concluded that the events were caused by the same call.
- **First Occur** is an 11-character field that contains the date and time when the vector event first occurred (for example, *09/28/13:43*).
- **Last Occur** is an 11-character field that contains the date and time when the vector event last occurred (for example, *09/28/13:48*).
- **Evnt Cnt** (Event Count) contains a number ranging from *1* to *255* that indicates the total number of vector events of this type that have occurred.

Summary of Vector Events

This section contains a table that does the following:

- Lists the number of each vector event
- Provides a description and an explanation (and sometimes possible causes and solutions) for each event type

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
1	Call dropped; call not queued at *stop* step.	Vector processing ended without the call being queued to a split and, as a result, the call cannot be answered. This implies that some default condition was not programmed or that the vector was designed to not always answer the call. Also, call was subsequently dropped.
2	Vector with no steps	The call encountered a vector with no steps administered.
3	1,000 step executed	This can occur due to the following: ■ Incorrect vector programming (for example, including a series of *goto* steps that point to one another) ■ Excessive repetition of a programmed loop during a single call (for example, recurring announcement-wait loop)
4	Administration change	The administration of this step occurred while the step was being executed. The call flow for this call is unpredictable. Vectors should not be changed while calls are active.
5	Call dropped by vector disconnect timer	The call was still in vector processing when the vector disconnect timer expired. The call dropped.
10	Retrying announcement	During an *announcement* step, a *collect digits* step that contains an announcement, or a *disconnect* step, the announcement was not available, and the announcement queue (if specified) was full. The step is retried at regular intervals.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
11	No announcement available	During an *announcement* step, a *collect digits* step that contains an announcement, or a *disconnect* step, the announcement was not available for one of the following reasons: ■ Announcement was not recorded ■ Analog announcement was busied out ■ Integrated announcement board was not installed ■ Integrated announcement ports were busied out ■ Integrated announcement was being recorded or restored
20	Call cannot be queued	A *queue-to main split, messaging split*, or *check-backup split* command failed to queue the call. **NOTE:** Event types *520, 521*, *522* and *541* may be observed for the same call at the same time.
21	Queued to three splits	The call attempted to queue to four splits. Multiple split queuing allows the call to queue to a maximum of three splits simultaneously. If the call queued to one or more splits, and if it should now be dequeued from those splits and then queued elsewhere, one solution is to route the call to a station (which may be administered without hardware). Once this happens, the call is forwarded to the VDN that controls the next stage of the call.
30	No TTR available	A *collect digits* command failed because ■ TN744 port was not available ■ All queue slots were occupied
31	Dial-ahead discarded	Previously entered dial-ahead digits have been discarded via access of a(n) *adjunct routing, converse-on, route-to number*, or *messaging split* step.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
32	Prompting buffer overflow	The prompting digit buffer already contained the maximum of 24 digits when additional dial-ahead digits were entered by the caller. These additional digits are not stored.
40	Messaging step failed	A messaging step failed because the Messaging Adjunct was not available. **NOTE:** Event types 540 and 541 may be observed for the same call at the same time.
50	Route -to step failed	A *route-to* step failed to reach the intended destination. **NOTE:** Event types 51 and 52 may provide more specific information regarding the reason for the failure. See Appendix G, "Operation Details for the Route-to Command".
51	No digits to route-to	The *route-to digits* step was unable to route the call because the previous *collect digits* step failed to collect any digits. This could result from an error in vector programming (for example, a *route-to digits* step appears without a preceding *collect digits* step). More often, however, this results because the caller was unable to enter the required digits (that is, the caller was using a rotary telephone), or because the caller was not provided with enough information to do so (as can be the case for auto-attendant applications).
52	No available trunks	A *route-to* command was unable to reach the specified off-switch destination due to a lack of available trunks.
53	Route-to step failed	The step was unable to seize a trunk because of a hardware problem or glare.
54	LAI retry	Look Ahead Interflow *route-to* step failed because of glare. The route will be retried once.
55	Double coverage attempt	Coverage option on route-to step was ignored because double coverage is not allowed. This may happen when the call has covered to a VDN.
60	Adjunct route failed	An adjunct route failed for one of reasons indicated in event types 61 through 66.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
61	Invalid destination	The *adjunct routing* command returned digits that did not represent a valid destination.
62	Adjunct route cancelled	The *adjunct routing step* was cancelled because another "routing" step (such as a *queue-to main split* step) was encountered in the vector.
63	Queue before route	The *adjunct routing* command was skipped because the call had already been queued via a *queue-to main split* or a *check-backup split* command.
64	Adjunct link error	The *adjunct routing* command was cancelled for one of the following reasons: ■ Link to the adjunct was down ■ ASAI protocol violation prevented the call from completing ■ Software resources to complete the call were unavailable
65	Agent not logged in	A Direct Agent Call was made to an agent who was not logged into the relevant split. Used for adjunct routing request only.
66	Agent not member of split	A Direct Agent Call was made to an agent who is not a member of the relevant split. Used for adjunct routing request only.
67	Invalid direct agent	A Direct Agent Call was made to an agent extension that is not valid. Used for adjunct routing request only.
70	Busy step for CO trunk	A CO trunk call reached a *busy* step in a vector without having previously received answer supervision. As a result, the caller continues to hear ringback rather than the busy tone.
80	Time not set	A *goto* step with a *time-of-day* conditional was processed, but the switch time was not set.
81	No digits collected	No digits were collected and a comparison was requested against a digit string or “in-table”. The comparison test was considered false and the next step in the vector was executed.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
90	Wait step music failed	A *wait-time* step with music was accessed, but the music was not connected. Music may not be administered correctly.
91	Wait step ringback failed	A *wait-time* step with ringback was accessed, but the ringback was not connected.
100	Redirect unanswered call	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent.
101	Redirect of call failed	The call was sent to an agent via a vector, but, due to the Redirection on No Answer (RONA) feature, the call was redirected from the ringing agent. The call could not be redirected.
110	Converse no ANI digits	On a *converse-on* step with passing type *ani*, no information was available to populate the field.
111	Converse no qpos digits	On a *converse-on* step with passing type *qpos*, no information was available to populate the field.
112	Converse no prompt digits	On a *converse-on* step with passing type *digits*, no information was available to populate the field.
113	Converse drop during data	On a *converse-on* step, the converse agent hung up while data was being passed. This may indicate a port failure.
115	ASAI transfer converse	ASAI attempted a transfer of a call that was active at a *converse* step. The transfer failed, and vector processing continued at the next vector step.
116	Converse transfer denied	A transfer of a call that was active at a *converse-on* step was attempted. The transfer either failed or was denied, and vector processing continued at the next vector step.
117	Agent drops converse	While active on a *converse-on* step, an agent became available in a split associated with a *queue-to main split* or *check-backup split* step. The call was delivered to the nonconverse agent, and the converse agent was dropped.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
125	Data return no digits	On a *converse-on* step, the converse agent activated data return but did not return any digits.
126	Data return timeout	On a *converse-on* step, the converse agent activated data return but timed out while waiting to return digits. Vector processing continued at the next vector step.
140	Coverage conference denied	Coverage to a VDN in a coverage path was denied because more than one party was active on the call.
150	Invalid EAS hunt group used in the vector step	Either the skill hunt group was removed or the skill hunt group became a non-ACD hunt group.
151	Skill indirection used improperly	Either no VDN skills are administered or the vector command has skill indirection and EAS is not enabled.
160	No vector steps, ANI sent	ANI was sent to CMS for a call that reached a VDN that accessed a vector with no steps defined.
170	ASA - invalid VDN	A *check-backup* or *goto* test requested a comparison of ASA for a VDN that had been removed since the vector was programmed. The comparison test was considered false and the next step in the vector was executed.
200	ANI not avail - digits	A *goto* test requested a comparison of ANI against a digit string and ANI was not available for the call. The comparison test was considered false and the next step in the vector was executed.
210	Routing table not assigned	A *goto* test requested a comparison with a vector routing table that is not assigned or had been removed since the vector was programmed. The comparison test was considered false and the next step in the vector was executed.
211	No entries in routing table	A *goto* test requested a comparison with a vector routing table that has no entries. This is considered as a non-match.
212	ANI not avail - table	A *goto* test requested a comparison of ANI against “in-table” and ANI was not available for the call. The comparison test was considered false and the next step in the vector was executed.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
220	EWT call not queued	A *goto* test for a call or converse data passing requested EWT for a call not in queue. In this case, the wait time was assumed to be infinite and the comparison was based on EWT > largest possible threshold.
221	EWT not sent to VRU	The EWT “wait” time for the call was not sent to the VRU for a *converse-on passing wait* vector step because the call was not queued or the splits/skills that the call was queued to were unstaffed.
222	System clock change	The system clock was changed, therefore any calculations involving time (i.e., ASA and EWT) will be inaccurate.
230	II-digits not avail - digits	A *goto* test requested a comparison of II-digits against a digit string and II-digits were not available for the call. The comparison test was considered false and the next step in the vector was executed.
231	II-digits not avail - table	A goto test requested a comparison if II-digits against “in-table” and II-digits were not available for the call. The comparison test was considered false and the next step in the vector was executed.
520	Split queue is full	A *queue-to main split, check-backup split*, or *messaging split* command was executed, but the call did not queue to the split because the queue (if administered) was full. To prevent this condition, use a *goto step...if calls queued in split...>...* before each *queue-to main split* or *check-backup split* step so that an alternative treatment may be provided for these cases.
521	Not vector-controlled	The split accessed by a *queue-to main split* or *check-backup split* command is not vector-controlled. As a result, the step is skipped.
522	AAS split cannot queue	A *queue to main split, check backup split,* or *messaging split* command was executed on an auto-available split (AAS), but the call did not queue to the split because all the agents were logged out by Redirection on No Answer (RONA).

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
540	AUDIX link down	AUDIX could not be accessed via a *messaging split* command, because the AUDIX link was down. As a result, the step is skipped.
541	Not a messaging split	The split administered for the *messaging split* command is not a messaging split (that is, it does not have a messaging type administered). As a result, the step is skipped.
550	ASA - No staffed agents	A *check-backup* or *goto* test requested a comparison of ASA for a split/skill that has no staffed agents. The comparison was based on ASA > largest possible threshold.
560	EWT no history for split	A *goto* test requested EWT for a split/skill that has not yet acquired history. The wait time in this case is assumed to be the default value.
561	EWT no split queue	A *goto* test requested EWT for a split/skill that has no queue. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.
562	EWT split queue full	A *goto* test requested EWT for a split/skill whose queue is currently full. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.

Table D-4. Summary of Vector Events

Event Type	Event Description	Event Explanation
563	EWT split no working agents	A *goto* test requested EWT for a split/skill that has no agents logged in or all logged in agents are in the AUX work mode. The wait time in this case is assumed to be infinite and the comparison was based on EWT > largest possible threshold.
564	EWT split locked	A goto test requested EWT for a split/skill that is currently locked. The wait time is assumed to be infinite. The comparison was based on EWT > largest possible threshold.
565	EWT call no working agents	A goto test for a call or converse data passing “wait” requested EWT for a call that is queued only to splits/skills that have no agents logged in or that have all logged in agents in AUX work mode. In this case, the wait time was assumed to be infinite and the comparison was based on EWT > largest possible threshold.

Functional Differences for G2 and G3 Call Vectoring and EAS

E

Introduction

This appendix provides the Call Vectoring functional differences between the Generic 2 and Generic 3 switches. This information should prove helpful to system administrators who administer networks that utilize both switches.

The appendix is organized into three sections. The first section indicates differences between the Call Vectoring commands common to both the Generic 2 and Generic 3 switches. The second section consists of a table that indicates general Generic 2 and Generic 3 differences within Call Vectoring. Finally, the third section indicates differences between the two switches in defining and/or interpreting split flows.

NOTE:
Call Prompting is not supported on G2.

Differences in Command Function

The following sections indicate the differences between the Generic 2 and Generic 3 switch versions of the Call Vectoring commands. The commands discussed include the following:

- *queue-to main split*
- *check-backup split*
- *goto step*
- *goto vector*
- *route-to number*

- *announcement*
- *wait-time*
- *busy*

Queue-to Main Split and Check-Backup Split

The *queue-to main split* command queues the call to the specified main split and assigns a queuing priority level.

The *check-backup split* command checks the status of a split for possible termination of the call to that split. When termination is not possible, queuing at the specified priority is attempted. Termination and/or queuing is attempted if the split meets certain conditions that are specified as part of the command.

Table E-1. G2/G3 Differences for Queuing Commands

GENERIC 3	GENERIC 2
The call is simultaneously queued to a maximum of three different splits. The indicated split is checked only once, and if the specified condition is met, an attempt to terminate or queue the call is made. Multiple checking of a backup split requires repeating the *check-backup split* command multiple times and/or unconditional *goto step* looping. After the call is queued to three splits, subsequent queue commands in the vector for additional splits fail and are skipped (unless these commands specify a different priority).	The call is queued to one split at a time. Successful queue commands that occur after the call is already queued cause the call to be dequeued from the first split and queued to a new split. Each *check backup* step executed by vector processing is rechecked in the background every two seconds while the steps that follow are processed. This process continues until the specified conditions are met. The periodic threshold checking of the *check backup split* commands is implemented to simulate multiple split queuing.
Calls can be queued to vector-controlled splits via Call Vectoring or to ACD splits/hunt groups directly via hunt group/split extensions when vectoring/prompting is active. Vector-controlled splits can be directly accessed via split extensions or via *route-to* commands to the extension ACD splits/hunt groups can also be accessed via *route-to* commands to the extension.	When Call Vectoring is active, calls can be queued to ACD splits **only** via the *queue to main split* and *check backup split* Call Vectoring commands.
Calls cannot be queued to splits that already hold the number of queued calls defined by the split queue size on the hunt group screen. Therefore, every queuing command should be preceded by a check step to determine if the queue is full. Also, queue limits should be set as high as possible to ensure the call queues. In G3i, a maximum of 1,000 queue slots can be preassigned to all the splits and hunt groups in a system; in G3r, a maximum of 10,500 such slots can be preassigned; in G3s PBP and G3vs PBP, a maximum of 200 such slots can be preassigned. In G3i, G3s PBP, or G3vs PBP, a maximum of 200 queue slots can be assigned to any one split or hunt group; in G3r, a maximum of 999 of such slots can be assigned.	No split queuing capacity limits are in effect, and the commands are never skipped.
The *check-backup split* command can test a maximum threshold of 199 calls in G3i, G3s PBP, or G3vs PBP, and 999 calls in G3r.	The command can test a maximum threshold of *only* 99 calls.

Table E-1. G2/G3 Differences for Queuing Commands

GENERIC 3	GENERIC 2
The oldest-call-waiting test condition within the *check-backup split* command has a range of 0 through 998 seconds in two-second increments.	The oldest-call-waiting test condition within the command has a range of 0 through 999 seconds in one-second increments.
An unconditional *check-backup split* command is allowed.	The *check backup split* command is conditional only.
The rolling-asa and expected-wait conditions are available with the *check-backup* split command.	These capabilities are not provided.

Goto Step and Goto Vector

The *goto step* command allows conditional or unconditional movement (branching) to a preceding or subsequent step in the vector.

The *goto vector* command allows conditional or unconditional movement (branching) to another vector.

Table E-2. G2/G3 Differences for Goto Commands

GENERIC 3	GENERIC 2
The commands can test a maximum threshold of 200 calls in G3i, G3s PBP, or G3vs PBP. In G3r, the maximum threshold is 999.	The commands can test a maximum threshold of *only* 99 calls.
The oldest-call-waiting test condition within the commands contains a range of 0 through 998 seconds and is checked according to a 2-second increment.	The oldest-call-waiting test condition within the commands contains a range of 0 through 999 seconds and is checked according to a 1-second increment.
The rolling-asa, expected-wait, counted-calls, ani, and ii-digits conditions are available with the *goto* commands. Vector routing tables can be checked for the digits, ani and ii-digits conditions.	These capabilities are not provided.

Route-to Number

This command routes the call to a specific number.

Table E-3. G2/G3 Differences for Route-to Number Command

GENERIC 3	GENERIC 2
The actual digit string is used as the destination. The string can contain special characters that may be stored in an AD string, including *~p, ~w, ~W, ~m,* and *~s* (but not *** or *#*). (See the *route-to number* command in the manual pages of Appendix A.) Feature access codes (AAR/ARS) or trunk access codes may be used to route calls externally.	The AD member number is used as the destination. None of the special characters may be used. The special functions are handled by the AAR/ARS pattern routing.
The trunk may be accessed via ARS/AAR, TAC, or UDP.	AAR/ARS is required for non-DCS trunk calls.
Routing to individual attendant extensions is permitted.	The individual attendant extension feature is not available.
Routing to announcement extensions is permitted.	The announcement command is required for all announcement access.
If the command fails, and if the command is the last step in the vector, the command is not retried. If retrying is required, an unconditional *goto step* can be used to loop back to the *route to* step.	If the command is the last step in the vector, a busy destination targeted by the command is retried every two seconds.
Routing to an ACD split extension is allowed even if Call Vectoring is operational.	This capability is not provided.
Routing to a Service Observing FAC is allowed	This capability is not provided.

Announcement

This command indicates that the caller should expect to hear an announcement. Although the announcement strategy G2 differs from that in G3, each strategy assures that, theoretically, the entire announcement is played from the beginning.

Table E-4. G2/G3 Differences for Announcement Command

GENERIC 3	GENERIC 2
Announcement extensions are used.	Announcement numbers are used.
Provides integrated board internal announcements.	Integrated announcement board is not supported.
G3i (prior to G3V2) supports only analog line port-connected external announcement devices. All G3V2 and later releases support auxiliary trunk-connected external announcement devices.	Supports only auxiliary trunk-connected announcement devices.
The maximum number of calls that can be queued and connected to an announcement is limited by preassigned queue slots. G3i has 150 such slots for analog ports and a common pool of 50 slots for an integrated board. G3i allows only 25 calls to be connected to an announcement. G3r has 300 slots for analog and auxiliary trunk ports and 300 slots for an integrated board. For an integrated or auxiliary trunk, G3r allows 255 calls to be connected to an announcement. For an analog trunk, G3r allows 128 calls to be connected to an announcement. G3V4 and later releases allow for multiple integrated announcement boards. G3i can have up to 5 boards, G3r can have up to 10. The queue slot and maximum connected calls capacities have been increased. See the *DEFINITY Communications System Generic 3 Feature Description,* 555- 230-204, for details	Limited only by the number of time slots available on the module to which the announcement channel is connected. The maximum number of time slots is 256.

Wait-time

This command sets a length of time for a call to wait in the queue. The command also specifies one of the following treatments while the call advances in the queue(s):

- Silence
- Ringing
- Music
- I-silent (G3V4 and later releases)
- Alternate Audio/Music Source (G3V4 and later releases)

Table E-5. G2/G3 Differences for Wait Command

GENERIC 3	GENERIC 2
The system-wide music-on-hold feature must be active for music treatment on the command. With G3V4 and later releases, an alternate audio/music source can be administered for a *wait-time* step.	A separate music option is available for Call Vectoring.

Busy

This command terminates vector processing and gives the caller a busy signal.

Table E-6. G2/G3 Differences for Busy Command

GENERIC 3	GENERIC 2
A timeout after 45 seconds is provided.	A 20 second timeout is provided for both CO and non-CO trunks.

General Call Vectoring Functional Differences

This table provides an overview of general differences for Call Vectoring operations between the Generic 2 and Generic 3 switches.

Table E-7. General Call Vectoring Functional Differences

TOPIC	GENERIC 3	GENERIC 2
General ACD	Split queue size is administered on a per split basis with a system-wide maximum of calls. In G3i, this maximum is 1,000 calls; in G3s PBP and G3vs PBP, this maximum is 200 calls; in G3r, the maximum is 10,500 calls. Call queue space for the appropriate maximum number of calls must be distributed on a preassigned basis over all assigned hunt groups and (vector-controlled or nonvector-controlled) ACD splits. In G3i, G3s PBP, or G3vs PBP, the maximum queue space that can be allocated for any one split and/or hunt group is 200; in G3r, it is 999.	There is no limit to the size of individual split queues.
	An agent may be concurrently logged into three splits at a time.	An agent may be logged into only one split at a time.
	The agent hears the same zip tone signal for calls that are queued to the main split as well as for intraflowed/interflowed calls.	One burst zip tone is provided for calls that are queued to the main split. Two burst zip tones are provided for intraflowed calls (via the *check backup split* command), and three burst zip tones are provided for interflowed calls (via Look-Ahead Interflow).
ACD Split Strategy	A split or a hunt group can be accessed by either a call vector or a group extension. This allows for both vector calls and nonvector calls in a single split's queue.	When Call Vectoring is optioned, splits do not have extensions. All access to splits **must** go through a Call Vector via *queue to main split* or *check backup split* commands.

Table E-7. General Call Vectoring Functional Differences

TOPIC	GENERIC 3	GENERIC 2
	Non-vector-controlled splits can specify redirection treatment (such as Call Coverage, Call Forwarding, etc.) and announcement treatment.	Only vector-controlled splits are available when Call Vectoring is active.
VDN Access/Capacity	COR checking is used for access to a VDN and for routing to a station.	No restriction checking is used to access a VDN. **NOTE:** Both G2 and G3 use the Facility Restriction Level (FRL) associated with the VDN for outgoing trunk calls.
	COR checking is used when routing locally from a vector.	No restriction check is implemented for local routing.
	A maximum of 500 VDNs [G3i (R3 CMS)], 100 VDNs [G3s PBP (R3 CMS), G3vs PBP (R3 CMS)], or 20000 VDNs [G3r (R3 CMS)] can be used.	The maximum number of VDNs is limited only by the number of extensions capacity (32K).
Voice Mailbox	*messaging split* command is used.	Calls are routed to a messaging split via a route to another VDN assigned to a vector with a queue to AUDIX.
Miscellaneous	Changes made to vector administration take effect upon submission. These changes can affect current calls.	A ‘‘scratch’’ pad is used for vector changes. Consequently, only new calls that enter the vector receive the treatment specified in the corrected vector. Vector processing for existing calls is completed in the old vector.

Table E-7. General Call Vectoring Functional Differences

TOPIC	GENERIC 3	GENERIC 2
	An existing vector can not be copied to another blank vector. (This capability, however, is available via CMS administration.)	These capabilities are provided by the switch administration.
	Either the VDN or the final destination (but not both) is provided in the CDR record.	Variable format CDR (formerly SMDR) records can be used. Consequently, both the VDN and the final destination can be provided. **NOTE:** CDR records allow the VDN to be specified in the calling party field.
	Blank steps are allowed in vectors, and blank vectors (with no steps defined) may exist.	Blank steps or blank vectors are not allowed (CMS also does not support this).
	Trunk groups can be assigned to VDNs *only* via switch administration.	Trunks groups can be assigned to VDNs via CMS administration.
	Vector processing is limited to a maximum of 1,000 step executions for a call. Once this maximum is reached, processing stops. There is an implied wait of one second for every seven executed steps.	Separate 1,000 step counters are provided for execution of *goto step* commands and *check backup split* retries. If either counter exceeds 1,000, the call is forced disconnected. Only *check backup split* retries are counted on internal calls.

Differences in Defining/Interpreting Split Flows

Split flows are defined and/or interpreted according to the switch version and the management system involved. The following sections illustrate how split flow interpretation differs within the G1/G3 and G2 switch versions and according to two management systems, including R3 CMS and R2 CMS.

NOTE:
BCMS is not available on G2 (with or without vectoring).

R3 CMS Standards

The following tables illustrate how split flows that occur in the G1/G3 and G2 versions of the switch are interpreted vis-a-vis R3 CMS:

Table E-8. R3 CMS Standards for Interpreting Split Flows

Flow Type	Switch Version	Interpretation
Inflow	G1/G3 with vectoring	Calls answered by a split other than a primary split. **NOTE:** A primary split is the first split to which a call queues.
	G2/traditional ACD	Calls that intraflow from one split's queue to another split's queue (that is, calls that queue to a split after having been previously queued to another split).
Outflow	G1/G3 with vectoring	Calls that are dequeued from a primary split via a *route to* or *messaging split* command, or by being answered by an agent in another split to which the call is also queued.
	G2/traditional ACD	Calls that are taken out of a split's queue and then sent to another destination.
Dequeue	G1/G3 with vectoring	Calls that are dequeued from any split other than the primary split in a VDN.
	G2/traditional ACD	(Not used.)

When a call is not answered [due to a(n) outflow, abandon, busy, or disconnect], the call's disposition is tracked for the primary split. On R3 CMS, the other splits to which the call is queued tracks a dequeue when the call outflows, abandons, is given busy treatment, or is disconnected.

If the primary split in a VDN is unmeasured, a(n) outflow, abandon, busy, or disconnect is not tracked for the call. Also, an answer is not tracked if the call is answered by an agent in the primary split.

R2 CMS Standards

For single split queuing, R2 CMS tracks split inflows and outflows according to the definitions provided in the previous section for ‘‘G2/traditional ACD.’’

However, when multiple split queuing is involved, a call can look like two or three separate calls to R2 CMS. As a result, if a call is queued to multiple splits and is then answered by an agent in one of these splits, an *inflow* is not tracked in R2 CMS. However, if a call is *requeued* to one or more splits (via a *route to* command, for example), an *inflow* is tracked *only in the first split to which the call requeues*.

Also, when multiple split queuing is involved, R2 CMS tracks an *outflow* in those splits to which the call queues and from which it eventually dequeues without being answered there. In effect, then, R2 CMS tracks an outflow in the same situations where R3 CMS tracks a dequeue.

Differences Between G2 and G3r EAS

This section lists the differences between release G2 and G3r for EAS.

- Capacities:

	G2.2	G3r
Measured Agents	1023	5200
Total Agents	2048	5200 (each agent in one skill)
Skills/agent	5 (1 default + 4 additional)	4
Skill Groups	600 (numbered 10-609)	255

- G2.2 does not have logical agent capabilities.
  - — Voice terminals are preassigned to default skill groups (groups ending in zero).
  - — Agents sharing voice terminals must have the same default skill group.
  - — The voice terminal extension is used to provide a name, COR, and coverage path.
- G3 logical agent provides the following:
  - — Any voice terminal can be used as an ACD terminal for any skills.
  - — Agents can be reached by dialing their login IDs.
  - — Name, COR, and coverage path follow the agent to the voice terminal currently logged into.
- G2.2 does not support Direct Agent Calling.
- G2.2 does not support Call Prompting.
- G2.2 login procedure is: dial feature access code, dial login ID twice. G3 login procedure is: dial feature access code, dial login ID, dial optional password.
- G2.2 restricts agents with multiple skills to skills in the same skill tens group (for example, skill 20-29). G3 allows agent to be in any combination of skills.

- G2.2 restricts calls queuing to multiple skills simultaneously to skills in the same skill tens group. This also applies to VDN skills. G3 allows calls to queue to any three skills simultaneously.
- G2.2 administers agents to a default skill and the agents enter their other skills after logging in. G3 administers all of the agents' skills, and the agents are logged into all of their assigned skills during login. G3 agents cannot change their skills.
- CMS can only change an agent's default skill on G2.2 (when the agent is unstaffed). CMS can change all skills for an agent on G3 (change affected the next time the agent logs in).
- G2.2 does not support primary/secondary skills for agents. This also implies G2.2 does not support expert agent distribution (EAD). G3 does support primary/secondary agent skill assignments and EAD.
- On G2.2, when a change is made to a VDN skill preference, only new calls to the VDN will be impacted by the change. On G3 when a change is made to a VDN preference, existing calls will be impacted as they encounter a vector step that references the VDN skill preference.

Interactions Between Call Vectoring/EAS and BCMS/CMS

F

Introduction

Call Vectoring and EAS interact with a management information system that helps to monitor and report on the activity within Call Vectoring and EAS. In most cases, the management system is either the Call Management System (CMS) or the Basic Call Management System (BCMS).

CMS, which resides on an adjunct processor, collects and processes ACD information to generate various reports. BCMS performs the same duties. The main difference between CMS and BCMS is that the latter resides on the customer switch. Also, it should be noted that CMS reporting capabilities are much more extensive than those of BCMS.

This chapter is intended to illustrate how these management systems interpret and report on activity within Call Vectoring and EAS. Special emphasis is placed on interpreting and reporting on this activity as it occurs within splits during a series of Call Vectoring or EAS events.

NOTE:
The manual pages in Appendix A provide a summary of the CMS/BCMS interactions with each Call Vectoring command (where applicable).

BCMS/CMS Tracking in a Call Vectoring Environment

Tracking is the identifying of various call flows and other actions relevant to call handling. For our purposes, there are three classes of call flows: split flows, VDN flows, and vector flows. Also, we are most concerned with tracking in the Call Vectoring environment. The specific types of call flows and actions in this environment that are tracked by BCMS/CMS include the following:

- Inflows (flow ins)
- Outflows (flow outs)
- Dequeues
- Abandons
- Answers
- Busies
- Disconnects

The split supervisor can use VDN and vector flows to evaluate how effective vector programming is at the site in question. The supervisor can use split flows to determine the manner in which the splits at the site are handling incoming telephone calls.

Defining and Interpreting Call Flows

The manner in which specific call flows are defined and interpreted depends upon the call flow class in question, the management system in effect, and the version of the DEFINITY switch being used. Management systems include *R3 CMS, R2 CMS*, and *BCMS*.

The following sections define and interpret specific call flows according to these parameters.

Answered and Abandons

The most important tracking items for most VDNs and vectors are the number of calls answered and the number of calls abandoned. R3 CMS provides VDN profiles that show when calls are answered and abandoned. Ten service level intervals are administered for these profiles. These intervals can have smaller time intervals around the time most calls are answered and when most call abandon to get more detailed information.

This data can be used to determine what an acceptable service level is for most callers. The percentage answered within the administered acceptable service level is also shown on the Call Profile reports. For VDNs, the calculation is ACD calls answered and nonACD calls connected within the service level divided by calls offered to the VDN (including calls that inflow to the VDN).

For split/skill statistics, the calculation is ACD calls answered within the service level divided by calls queued to the split/skill (answered calls, abandoned calls, calls that flow out, calls that dequeue). In most cases the VDN percentage will be higher then the split percentage since calls dequeued from a split/skill are counted as answered, abandoned, or outflows for the VDN.

Changes made to a vector or to staffing will typically impact the VDN call profile. Even the wording of an announcement can impact the abandon profile. It is worthwhile to review the VDN's call profile before and after any change to determine if the change had a positive impact.

Busies and Disconnects

Busy calls and forced disconnects reported on CMS indicate how many calls this VDN/vector turned away. If forced disconnect is used out of business hours, this item would indicate how many customers expected you to be operating during a specific time interval. If busies are given when the queues are full or waiting times are long, the number of busies in an interval might suggest a staffing change is needed. If disconnect is used to deny a lookahead interflow attempt, a large number of denials would indicate a busy time at multiple sites.

VDN Inflows and Outflows

The following section discusses the specific VDN flows vis-a-vis R3 CMS and BCMS.

R3 CMS and BCMS Standards

The following table illustrates how R3 CMS and BCMS interpret specific VDN flows for the G1/G3 versions of the DEFINITY switch:

Table F-1. R3 CMS and BCMS Standards for Interpreting VDN Flows (in G1/G3)

Flow Type	Management System	Interpretation
VDN flow in	R3 CMS	Calls that flow into the vector from another vector via a *route-to* command.
	BCMS	(Not tracked.)
VDN flow out	R3 CMS	Calls that successfully flow out of a vector to another VDN or external location via a *route-to* command.
	BCMS	Calls that are advanced to another position via a successful *route-to* or *messaging split* command. This can involve adjunct routing, calls forwarded, calls routed to a VDN, and calls picked up by an agent who is not in the split for which the call is queued by the VDN. Calls that are answered by an attendant (via a *route-to* command).

NOTE:
(R3 CMS only): If a call that covers to a VDN is originally a call to a measured (nonvector-controlled) VDN, R3 CMS records a VDN flow in for the coverage to the second VDN and a VDN flow out for the first VDN.

Vector Inflows and Outflows

The following section discusses the specific vector flows vis-a-vis R3 CMS.

R3 CMS Standards

Vector flow in pertains to calls that flow into a vector from another vector via a *route to* or a *goto vector* command. *Vector flow out* pertains to calls that successfully flow out of a vector via a *route to* or a *goto vector* command.

Split Inflows, Outflows, and Dequeues

The following sections discuss the various split flow types vis-a-vis R3 CMS, R2 CMS, and BCMS.

R3 CMS and BCMS Standards

R3 CMS and BCMS are grouped together because both of these systems interpret two split flow types identically. These flows include *inflow* and *outflow*. However, whereas R3 CMS interprets another split flow type, namely *dequeue*, BCMS does not do so because this system does not have a dequeue tracking item. This means that in a situation where R3 CMS tracks a dequeue, BCMS does not because it is unable to do so.

Before we detail how R3 CMS and BCMS interpret split flows, we should discuss the term *primary split*, since this concept plays a significant role in tracking. *Primary split* is defined as the *first split in a VDN to which a call actually queues or at which the call is connected to an agent*. Therefore, this split is *not necessarily* the first split referenced in the vector.

Another split becomes the primary split if either of the following events occur:

- Call cannot queue to the originally-targeted split because the split has no queue slots available.
- Call leaves the VDN (via a *route to* VDN command, for example) and is queued to another split as a result.

If the call leaves vector processing and does not queue to another split (as a result of a *route-to* extension command, for example), there is no new primary split.

With this discussion in mind, let's take a look at the following table to see how R3 CMS and BCMS interpret split flows for the G1/G3 versions of the DEFINITY switch:

Table F-2. R3 CMS and BCMS Standards for Interpreting Split Flows (in G1/G3)

Flow Type	Management System	Interpretation
Inflow	R3 CMS	Calls answered by a split other than a primary split.
	BCMS	(Same as for R3 CMS.)
Outflow	R3 CMS	Calls that are dequeued from a primary split via a *route-to* or *messaging split* command, or by being answered by an agent in another split to which the call is also queued.
	BCMS	(Same as for R3 CMS.)
Dequeue	R3 CMS	Calls that are dequeued from and not answered by any split other than the primary split in a VDN.
	BCMS	(Not tracked.)

When a call is not answered [due to a(n) outflow, abandon, busy, or disconnect], the call's disposition is tracked for the primary split. On R3 CMS, the other splits to which the call is queued track a dequeue when the call outflows, abandons, is given busy treatment, or is disconnected.

If the primary split in a VDN is unmeasured, a(n) outflow, abandon, busy, or disconnect is not tracked for the call. Also, an answer is not tracked if the call is answered by an agent in the primary split.

R2 CMS Standards

When multiple split queuing is involved, a call can look like two or three separate calls to R2 CMS. As a result, if a call is queued to multiple splits and is then answered by an agent in one of these splits, an *inflow* is not tracked in R2 CMS. However, if a call is *requeued* to one or more splits (via a *route to* command, for example), an *inflow* is tracked *only in the first split to which the call requeues*.

Also, when multiple split queuing is involved, R2 CMS tracks an *outflow* in those splits to which the call queues and from which it eventually dequeues without being answered there. In effect, then, R2 CMS tracks an outflow in the same situations where R3 CMS tracks a dequeue.

Examples of Split Flow Tracking

The following sections provide some examples of tracking in R3 CMS, R2 CMS, and BCMS. Each section first presents a scenario of Call Vectoring events. The scenario is then followed by a table in which the tracking for the various splits involved is recorded. Following each "tracking table,'' an explanation of the tracking procedure is provided.

The scenarios presented include the following:

- Call answered by a primary split
- Call answered by a nonprimary split
- Call abandoned
- Call answered by a primary split after a route to VDN
- Call answered by a nonprimary split after a route to VDN
- Call answered after a route to split

NOTE:
Inflows, outflows, and dequeues are not tracked for splits administered by the *converse on split* command. However, if a call is answered both by a converse split and (subsequently) by a nonconverse split, an ''answer'' is tracked for each split. However, a call is really considered ''answered'' only when it is answered by a nonconverse split. Therefore, traffic measurements for converse splits should be used only to measure converse split traffic and not to calculate the total number of calls.

Call Answered by a Primary Split. The following scenario involves a call answered by the primary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 1.

Here's the tracking table for this scenario:

Table F-3. Tracking for Call Answered by Primary Split

	Split Tracking		
	1	**2**	**3**
R3 CMS	answer	dequeue	dequeue
BCMS	answer		
R2 CMS	answer	outflow	outflow

Comments:

- **R3 CMS**: *Dequeue* is tracked in split 2 as well as in split 3 because the call is answered by the primary split (split 1) and is thus dequeued from splits 2 and 3 without being answered in these splits.
- **BCMS**: No dequeue tracking item is available.
- **R2 CMS**: *Outflow* is tracked in the same situations where R3 CMS tracks a *dequeue*. Accordingly, *outflow* is tracked in splits 2 and 3 because the call is dequeued from these splits without being answered in either one of the splits.

Call Answered by a Non-Primary Split. The following scenario involves a call answered by a nonprimary split. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is answered in split 2.

Here's the tracking table for this scenario:

Table F-4. Tracking for Call Answered by Non-Primary Split

	Split Tracking		
	1	**2**	**3**
R3 CMS	outflow	inflow answer	dequeue
BCMS	outflow	inflow answer	
R2 CMS	outflow	answer	outflow

Comments:

- **R3 CMS**: *Outflow* is tracked in split 1 because the call is answered by an agent in another split to which the call is queued (that is, split 2). Although the call is obviously removed from split 1 after it is answered in split 2, *dequeue* is **not** tracked in split 1 because split 1 is the primary split. *Inflow* is tracked in split 2 because the call is answered in this split and the split is not the primary split. *Dequeue* is tracked in split 3 because the call is removed from the split without being answered there. When the call is removed from split 3, *outflow* is **not** tracked in split 3 because this split is not the primary split.
- **BCMS**: Follows the same scheme as R3 CMS except for the *dequeue* tracking.
- **R2 CMS**: *Outflow* is tracked in splits 1 and 3 because the call is dequeued from these splits without being answered there.

Call Abandoned. The following scenario involves a call abandoned by the caller. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Call is abandoned.

Here's the tracking table for this scenario:

Table F-5. Tracking for Abandoned Calls

	Split Tracking		
	1	**2**	**3**
R3 CMS	abandon	dequeue	dequeue
BCMS	abandon		
R2 CMS	abandon	outflow	outflow

Comments:

- **R3 CMS**: *Abandon* is tracked in split 1 because this split is the primary split. *Dequeue* is tracked in splits 2 and 3 because the call is dequeued from these splits without being answered in either split.
- **BCMS**: *Abandon* is tracked in split 1 because this split is the primary split. Tracking is not recorded in splits 2 and 3 because no dequeue tracking item is available.
- **R2 CMS**: *Abandon* is tracked in split 1 because this split is the primary split. *Outflow* is tracked in splits 2 and 3 because the call is dequeued from these splits without being answered in either one of the splits.

Call Answered by a Primary Split after a Route To VDN. The following scenario involves a call answered by the primary split after a *route-to VDN* command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a *route-to VDN* step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 4.

Here's the tracking table for this scenario:

Table F-6. Tracking for Call Answered by Primary Split after Route to VDN

	Split Tracking					
	1	**2**	**3**	**4**	**5**	**6**
R3 CMS	outflow	dequeue	dequeue	answer	dequeue	dequeue
BCMS	outflow			answer		
R2 CMS	outflow	outflow	outflow	inflow answer	outflow	outflow

Comments:

Split 1 is the *original* primary split, because this is the first split to which the call actually queues. However, split 4 becomes the *new* primary split because:

- Call leaves the *original* VDN upon execution of the *route-to VDN* step.
- Split 4 is the first split to which the call queues upon execution of this step.
- **R3 CMS**: *Outflow* is tracked in split 1 because this split is the *original* primary split, and the call is dequeued from this split via a *route-to VDN* step. *Dequeue* is tracked in splits 2, 3, 5, and 6 because the call is dequeued from each of these splits without being answered in any one of them.
- **BCMS**: Follows the same scheme as R3 CMS except for the *dequeue* tracking.
- **R2 CMS**: *Outflow* is tracked in splits 1, 2, 3, 5 and 6 because the call is dequeued from these splits without being answered in any one of them. *Inflow* is tracked in split 4 because split 4 is the first split to which the call requeues after the *route to* command is executed.

Call Answered by the Non-Primary Split after a Route To VDN. The following scenario involves a call answered by the nonprimary split after a *route to VDN* command is executed. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a *route-to VDN* step.
3. Call is then queued to splits 4, 5 and 6.
4. Call is answered in split 5.

Here's the tracking table for this scenario:

Table F-7. Tracking for Call Answered by Non-Primary Split after Route to VDN

	Split Tracking					
	1	**2**	**3**	**4**	**5**	**6**
R3 CMS	outflow	dequeue	dequeue	outflow	inflow answer	dequeue
BCMS	outflow			outflow	inflow answer	
R2 CMS	outflow	outflow	outflow	inflow outflow	answer	outflow

Comments:

- **R3 CMS**: *Outflow* is tracked in split 1 because this split is the *original* primary split, and the call is dequeued from this split via a *route-to VDN* step. *Dequeue* is tracked in splits 2, 3, and 6 because the call is dequeued from each of these splits without being answered in any one of them. *Outflow* is tracked in split 4 because this split becomes the *new* primary split after the *route-to VDN* step is executed, and the call is subsequently dequeued from this split by being answered in another split (split 5) to which the call is also queued. Finally, *inflow* is tracked in split 5 because the call is answered in this split, and the split is not the primary split.
- **BCMS**: Follows the same scheme as R3 CMS except for the *dequeue* tracking.
- **R2 CMS**: *Outflow* is tracked in splits 1, 2, 3, 4, and 6 because the call is dequeued from these splits without being answered in any one of them. *Inflow* is tracked in split 4 because this split is the first one to which the call is requeued after the *route to* command is executed.

Call Answered after a Route To Split. The following scenario involves a call answered after it is routed to a split via a *route-to digits* or *messaging split* command. The scenario is as follows:

1. Call comes into a VDN whose vector queues the call to splits 1, 2 and 3.
2. Vector executes a *route-to digits* (or *messaging split*) step.

3. Call is queued to split 4.

Here's the tracking table for this scenario:

Table F-8. Tracking for Call Answered after Route to Split

	Split Tracking			
	1	**2**	**3**	**4**
R3 CMS	outflow	dequeue	dequeue	answer
BCMS	outflow			answer
R2 CMS	outflow	outflow	outflow	inflow answer

Comments:

- **R3 CMS**: *Outflow* is tracked in split 1 because this split is the *original* primary split, the call is dequeued from this split via a *route-to digits* (or *messaging split*) step, and the call is answered in split 4, which becomes the *new* primary split. *Dequeue* is tracked in splits 2 and 3 because the call is dequeued from each of these splits without being answered in any one of them.
- **BCMS**: Follows the same scheme as R3 CMS except for the *dequeue* tracking.
- **R2 CMS**: *Outflow* is tracked in splits 1, 2, and 3 because the call is dequeued from these splits without being answered in any of them. *Inflow* is tracked in split 4 because this split is the first one to which the call is requeued after the *route-to digits* (or *messaging split*) command is executed.

Evaluating Split Performance

By using the information presented to this point, along with the information from various reports (as discussed in the next section), the split supervisor can answer one or more questions concerning split performance and then make adjustments, if necessary. Here are some of the questions the supervisor can answer:

1. How many ACD calls offered to my split were ''mine'' (that is, were offered to this split as the primary split)?

NOTE:
Split ''ACD calls'' include Direct Agent Calls for BCMS and for R2 CMS, but not for R3 CMS, which tracks Direct Agent Calls separately.

2. How many ACD calls did ‘‘my’’ split answer that were “mine?” (And, by implication, how many did I answer that were not “mine?”)
3. How many of ‘‘my’’ ACD calls did “my” split not answer?
4. How many ACD calls that I didn’t answer weren’t “mine?”

The following sections present the answers to these questions from the perspective of R3 CMS, BCMS, and R2 CMS.

R3 CMS Standard. The following answers reflect the use of R3 CMS:

1. The number of calls offered to “my” (primary) split that were ‘‘mine’’ can be determined via examination of the CMS Split Summary Report. The algorithm is as follows: CALLSOFFERRED - INFLOWCALLS - DEQUEUECALLS (that is, the total number of calls offered *minus* the number of calls not “mine’’ that I answered *minus* the number of calls not ‘‘mine’’ that I didn’t answer.)
2. The number of calls that my split answered that were “mine’’ can be determined via examination of the CMS Split Summary Report. The algorithm is as follows: ACDCALLS - INFLOWCALLS (that is, the total number of calls I answered *minus* the number of calls not “mine” that I answered.)
3. The number of ‘‘my” calls that ‘‘my’’ split didn’t answer can be determined via examination of the CMS VDN Report. The algorithm is as follows: ABNCALLS + BUSYCALLS + DISCCALLS + OUTFLOWCALLS (that is, the number of abandoned calls *plus* the number of busy calls *plus* the number of disconnected calls *plus* the number of calls outflowed from “my’’ split tagged as a primary split).
4. The number of calls not “mine” that “my” split didn’t answer is DEQUEUECALLS, which is indicated in the CMS Split Summary Report.

BCMS Standard. The following answers reflect the use of BCMS:

1. The number of calls offered to “my” split that were “mine” can be determined via examination of the BCMS Split Report. The algorithm is as follows: ACDCALLS + ABNCALLS + OUTFLOWCALLS - INFLOWCALLS (that is, the total number of calls answered *plus* the total number of calls abandoned from ‘‘my’’ split tagged as a primary split *plus* the number of calls that outflowed “my” split tagged as a primary split *minus* the number of calls answered that were not directed to “my’’ split tagged as a primary split).
2. The number of calls that ‘‘my’’ split answered that were “mine’’ can be determined via examination of the BCMS Split Report. The algorithm is as follows: ACDCALLS - INFLOWCALLS (that is, the total number of calls I answered *minus* the number of calls not ‘‘mine” that I answered).

The other two questions cannot be answered because BCMS does not have a *dequeue* tracking item.

R2 CMS Standard. Customers using R2 CMS connected to G1/G3 with vectoring enabled cannot *necessarily* answer any of the questions. If multiple-split queuing is involved, the OUTFLOWCALLS track contains both ''my'' calls and other splits' calls that outflowed. As a result, the answers to questions 1, 3 and 4 cannot be calculated. Also, question 2 cannot be answered because there is no track for the number of calls coming from elsewhere that ''my'' split actually answered.

Using BCMS/CMS Reports to Evaluate Call Vectoring Activity

There exists a number of CMS and BCMS reports that allow the customer to evaluate various facets of Call Vectoring activity. Some of these facets include the call flows present within Call Vectoring as well as the speeds at which calls are answered. The sections that follow identify and discuss the CMS and BCMS reports that indicate this activity.

CMS Reports

CMS has real-time and historical reports. Most CMS historical reports are available in four versions: intra-hour, daily, weekday, and monthly. The following list identifies and describes several CMS reports that summarize Call Vectoring activity. For further details on these and other related reports, refer to the *3B Call Management System Administration* 585-215-511.

NOTE:
The reports described in this section are generated in R3 CMS. Corresponding R2 CMS reports may not provide information that reflects capabilities that are new to the DEFINITY Switch (for example, internal/external call tracking).

- **Split Summary Report** summarizes the call activity for an entire split. Among other information, the report provides the total number of flow ins (inflows), flow outs (outflows), dequeues, and abandoned calls.

  The report also indicates the *average speed of answer* (interval ASA) for calls. This refers to the sum of the queue time and ring time for a call within the answering split *only*. Finally, the report indicates the *dequeued average queue time*, which is the average time a call waits until it is answered by another split to which the call is also queued.

- **VDN Report** summarizes VDN activity for specific vectors. Among other information, the report provides the number of VDN Flow Ins/Outs, calls forced busy, and calls forced disconnect. *VDN Flow In* pertains to calls that flow into a vector from another vector via a *route to* command. *VDN Flow Out* pertains to calls that successfully flow out of vector to another VDN or external location via a *route to* command.

- **Vector Report** summarizes vector activities. Among other information, the report provides the number of Vector Flow Ins/Outs, calls forced busy, and calls forced disconnect. *Vector Flow In* pertains to calls that flow into a vector from another vector via a *route-to* or *goto vector* command. *Vector Flow Out* pertains to calls that successfully flow out of a vector via a *route-to* or *goto vector* command.

BCMS Reports

BCMS has a real-time split report, split historical reports, real-time VDN reports, and VDN historical reports. The following list identifies and describes several BCMS reports that summarize Call Vectoring activity. For more information on these and other related reports, refer to *DEFINITY Communications System Generic 3 Basic Call Management System (BCMS) Operations* manual 555-230-704.

- **BCMS Split Report** summarizes the call activity for an entire split. The information can be requested either daily or by the administered time period. Among other information, the report provides the total number of flow ins (inflows) and flow outs (outflows). The report also provides the average speed of answer time for calls handled by the split during the indicated time period.

- **VDN Summary Report** summarizes statistical information for all internally-measured VDNs. The information can be requested by the administered time interval or daily. The "list bcms vdn" report gives multiple time periods or days for a single VDN. The "list bcms summary vdn" report gives a one-line summary per vdn (with data from the specified times or days), but can give the data for numerous vdns.

  The report also indicates the total number of flow outs, specifically, the number of calls that are advanced to another position via a successful *route-to* or *messaging split* command. This includes adjunct routing, calls forwarded, calls routed to a VDN, and calls picked up by an agent who is not in the split to which the call is queued by the VDN. Calls that are answered by an attendant (via a *route-to* command) are also tracked as outflow. However, calls that encounter a *goto vector* command are not shown as outflows. No further measurements are taken on the calls once the calls have outflowed. If an outflowed call later abandons, this is not indicated in the report.

  Among other information, the VDN report provides a total for offered calls, answered calls, abandoned calls, and also one for calls that were either ‘‘forced busy’’ or ‘‘forced disconnect.’’

- **VDN Real-Time Report** provides statistical information including the number of calls currently waiting and the oldest call waiting. The VDN real-time report has the same characteristics as other real-time BCMS reports.

Using CMS in an EAS Environment

CMS R3V2 or a later release is required to support Expert Agent Selection (EAS). The same tracking and database items used within a traditional Call Vectoring environment are used within an EAS environment. In addition, CMS split tables are used to track the equivalent skill hunt group information. However, for EAS, these tables are expanded to handle four skills per agent. Finally, all existing custom reports should work when you are upgrading to EAS.

Tracking Entities

The following sections explain how the following entities are tracked in an environment with EAS optioned:

- Agents and their skills
- Direct Agent Calls
- Non-ACD Calls
- VDN Skill Preferences

Agents and their Skills

The fields under the ‘‘Extn’’ column in the CMS Real-Time Agent Report show the voice terminal extension that the agent is logged into. These fields can be used to locate the agent or to service observe the agent.

With EAS optioned, the Skill Status Report replaces the Split Status Report. This report indicates the skills logged into and whether each skill is a primary or secondary skill. If too many calls are waiting, or if calls are waiting too long (also shown on the Skill Status report), it is possible that not enough agents have the skill shown as a primary skill. Agents with multiple skills always handle waiting primary skill calls first. If Expert Agent Distribution (EAD) is assigned to the primary skill, secondary agents may have much longer available times than primary agents because calls always go to primary agents first if any such agents are available.

If an agent logs into some but not all of the skills assigned to the agent, the Skill Status report shows only the skills logged into. An agent may be denied login to some skills if the maximum agents/skill number is met or if the CMS limit on agent/skill pairs logged in has been reached.

The Login/Logout Historical Report also lists the agent’s skills and whether the skills are primary or secondary.

Direct Agent Calls

Waiting Direct Agent calls are not included in the ‘‘Calls waiting’’ and ‘‘Oldest Call Waiting’’ report fields for skills because such calls are not skill calls. However, Direct Agent Calls are included in these two report fields for VDNs.

The Queue/Agent Summary Real-Time Report lists separately the Direct Agent calls waiting in a skill queue. Any waiting Direct Agent calls are queued to the first primary skill that the agent is assigned and logged into. To manage the skill's queue slots effectively, it is recommended that a skill be dedicated for Direct Agent calls.

Since Direct Agent calls are not skill calls, the skill tables do not track Direct Agent calls; however, the tables do monitor skill queue slots. The agent's time is tracked as ''OTHER'' in the skill tables. In the agent tables, there are separate Direct Agent call items. The standard CMS agent reports add the Direct Agent calls and the skill ACD calls and report these calls as ''ACD Calls.'' The VDN tables track Direct Agent calls as ACD calls.

Non-ACD Calls

The first measured (primary or secondary) skill that an EAS agent is logged into is used by CMS to track non-ACD calls unless the agent has an ACD call on hold. If an ACD call is on hold, the non-ACD call is counted for the skill of the held ACD call.

VDN Skill Preferences

VDN skill preference data is collected to provide information on what groups of agents (skills) are handling calls and on how effectively each skill group handles a particular VDN.

Real-time and historical VDN Skill Preference reports can be used to compare the percentage of calls being answered by the 1st, 2nd, and 3rd VDN preferences against an objective. If too few calls are being answered by the 1st skill preference, the vector can be adjusted to allow more time for the 1st skill preference group to answer calls; another alternative is to train or hire more agents with the 1st skill preference.

You can use VDN skill preference data to compare the average talk time and average ACW time for agents in the 1st, 2nd, and 3rd skill groups. If these times vary too much across groups, more training may be needed for the backup groups (that is, the 2nd and 3rd skill groups).

VDN skill preference data is tracked according to the skill preferences (1st, 2nd, 3rd) assigned to the VDN. Whenever a vector step either references a 1st, 2nd, or 3rd skill or specifies a skill number that matches the 1st, 2nd, or 3rd skill administered, the new database items are tracked. For example, if VDN 1000 has Skills 21, 22, and 23 administered as the 1st, 2nd, and 3rd skills, respectively, and if the vector associated with VDN 1000 has a ''queue to main skill 22'' step, tracking occurs for the 2nd VDN skill preference if the call is answered by an agent in Skill 22. Skill preference tracking also occurs for Skills

21 and 23. This allows users who prefer to specify the actual skill number in the vector to take advantage of the tracking for VDN skill preferences.

"G3V2 EAS Administration from CMS" R3V2 CMS can be used to administer vectors as well as skills for agents and VDNs. The ACD Administration: Change Agent Skills CMS screen is used to request that the current set of skills assigned to an agent be sent from G3V2. Both the skill number and skill type (primary or secondary) is displayed and can be modified.

NOTE:

Any changes made are not put into effect until the next time the agent logs in.

The ACD Administration: Change VDN Skill Preferences screen is used to request a VDN's skill preferences and to modify the VDN's skills.

The CMS Vector Contents screen is used to create and modify vectors. CMS supports the Call Vectoring commands that queue calls to the 1st, 2nd, or 3rd VDN skill.

Operation Details for the Route-to Command

G

Introduction

The *route-to* command can be programmed with or without coverage. The table in this appendix summarizes the operation of the *route-to* command for each of the destination types and conditions associated with the commands.

Table G-1. DEFINITY G3 Route-To Command Operation

CONDITION	INTERACTION	
	cov = n	**cov = y**
	ANY STEP	**ANY STEP[2]**
Invalid Destination[3]	Goes to next step, else stop	Goes to next step, else stop
VDN Extension		
- Vector Assigned	Goes to new vector	Goes to new vector
- Vector Has No Steps	Stop[1]	Stop[1]
Station Extension Idle (all appearances idle)		
- CF-ALL Active or -CF-DA Applies	Forwards if possible, else next step, else stop[1]	Forwards if possible, else coverage, else busy
- Coverage		
- DA Applies	Rings idle app.	Coverage on DA
- All Applies	Goes to next step, else stop[1]	Coverage
- SAC Applies	Rings idle appearance	Coverage
- None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover
Station Extension Active (with idle 2-way app)		
- CF-ALL Active	Forwards if possible, else next step, else stop[1]	Forwards if possible, else coverage, else busy
- Coverage		
- DA Applies	Rings idle app (no DA timing)	Coverage on DA
- Ext Act Applies	Rings idle appearance	Coverage
- All Applies	Goes to next step, else stop[1]	Coverage
- SAC Applies	Rings idle appearance	Coverage
- None of Above Applies	Rings idle appearance	Call delivered and is allowed to cover
Station Extension Busy (no idle 2-way app)		
- Extension in Hunt Grp (also see ACD Hunt Grp)	Queues if possible, else next step, else stop[1]	Queues if possible,else coverage, else busy
- CF-ALL Active or -CF-DA Applies	Forwards if possible, else next step, else stop[1]	Forwards if possible, else coverage, else busy
- Call Waiting to Analog Sta Would Apply	Goes to next step, else stop[1]	Call waits
- Coverage		
- Ext Act Applies	Goes to next step, else stop[1]	Coverage
- Ext Bsy Applies	Goes to next step, else stop[1]	Coverage
- All Applies	Goes to next step, else stop[1]	Coverage
- SAC Applies	Goes to next step, else stop[1]	Coverage
- None of Above Applies (or hunt, fwd, or cov dest is unavailable)	Goes to next step, else stop[1]	Busy tone given

Table G-1. DEFINITY G3 Route-To Command Operation

CONDITION	INTERACTION	
	cov = n	cov = y
	ANY STEP	ANY STEP[2]
Extension with Incompatible COR	Goes to next step, else stop.	Goes to next step, else stop.
Terminating Extension Group		
- All Members Idle	Rings idle appearance	Call delivered and is allowed to cover
- A Member Active on TEG	Goes to next step, else stop[1]	Coverage, else busy
- No Idle App on Any Member	Goes to next step, else stop[1]	Coverage, else busy
Hunt Group Extension		
- Idle Agent	Rings idle appearance	Call delivered and is allowed to cover
- No Idle Agent		
- Call can't queue	Goes to next step, else stop[1]	Busy tone given
- Call can queue	Call is queued	Call is queued
Extension on Another Node (Uniform Dialing Plan - UDP DCS or non-DCS)		
- Trunk Available	Call delivered	Call delivered
- Trunk Not Available	Goes to next step, else stop[1]	Queues if possible, else reorder
- No DCS Buffer for Routing	Call delivered w/o DCS msg	Call delivered w/o DCS msg
Trunk Access Code (TAC) Destination		
- Trk Grp No Dial Access	Goes to next step, else stop[1]	Routes to local and
- Trunk Available	Call delivered	Call delivered
- Trunk Not Available	Goes to next step, else stop[1]	Queues if possible, else reorder
AAR/ARS FAC Dest. (including Subnet Trkng)		
- Trk Grp No Dial Access	Tries next route	Routes to local attendant
- Trunk Available	Call delivered	Call delivered
- Other Routes Avail	Tries next route	Tries next route
- All Routes Busy		
- No Pattern Queuing	Goes to next step, else stop[1]	Reorder tone given
- Queuing Assigned	Goes to next step, else stop[1]	Queues to pattern
Attendant Queue (dial 0)		
- Idle Atnd	Rings idle appearance	Call delivered and is allowed to cover
- No Idle Atnd		
- Not In Night Svc	Call is queued	Call is queued
- In Night Svc		
- Nite Dest. Assigned	Delivered to night svc.	Delivered to night svc.
- Not Assigned	Call is queued	Call is queued

Table G-1. DEFINITY G3 Route-To Command Operation

CONDITION	INTERACTION	
	cov = n	cov = y
	ANY STEP	ANY STEP[2]
Individual Attendant Access		
- Atnd Idle	Rings idle appearance	Call delivered and is allowed to cover
- Atnd Busy	Queues if possible else Goes to next step, else stop[1]	Queues if possible, else Busy tone given
CAS Attendant With Caller on Branch		
- RLT Available	Rings idle appearance	Call delivered and is allowed to cover
- All RLTs Busy	Queues if possible, else next step, else stop[1]	Queues if possible, else busy tone
Inter-PBX Atnd Calling		
- Trk Grp Controlled	Routes to local atnd	Routes to local atnd
- Trk Available	Call delivered	Call delivered
- Trk Not Available	Goes to next step, else stop[1]	Reorder tone given
Look Ahead Interflow (LAI) (feat. active & routes over ISDN-PRI facility)		
- B-Channel Not Avail	Goes to next step, else stop[1]	Queues if possible, else reorder
- B-Channel Avail and		
- Receiving Switch		
- Accepts	Interflow succeeds[4]	Call cut-through
- Rejects	Goes to next step, else stop[1]	Call gets busy/disconnect
- Receiving Switch w LAI Acting as Tandem Sees from Remote Receiving Sw.		
- Call Accepted	Interflow succeeds[4]	Call cut-through
- Call Rejected	Goes to next step at receiving switch, else sending switch considers call rejected after 2-minute timeout	Call gets busy/disconnect

Notes:

[1] The interaction “Stop’’ means the following: vector processing is stopped, the call remains queued to a split, and the caller continues to hear feedback initiated by a previous step. In the case where the *route-to* command fails and processing stops (due to a busy station or trunk group destination), retry can be implemented in the vector. Retrying is accomplished by including an unconditional *goto* step as the last step to allow for a loop back to the *route to* command. Use of an intermediate *wait-time* command step with appropriate feedback and delay interval is strongly recommended in order to reduce processor occupancy.

[2] The call is removed from vector processing (that is, the call is taken out of any split queue, and any feedback, such as music or ringback, is removed) for *with coverage y* interactions, even if the destination is not available. Any *route-to with coverage y* step that routes over ISDN-PRI facilities cancels the Look-Ahead Interflow feature. Specifically, the call is taken out of vector processing when the step is reached, regardless of facility or remote switch availability. If the call is subsequently ‘‘rejected’’ by the receiving switch vector, the call is given the treatment defined by the ‘‘rejection’’ command (either busy or forced disconnect). Finally, note that a call routed via an *adjunct routing* command is treated the same way as a call that is routed via a *route-to with coverage y* command.

[3] Invalid destinations include the following: empty (for example, zero collected digits) or invalid *route-to* destination number, unassigned extension number, incomplete number of digits for AAR/ARS pattern, non-AAR/ARS feature access code (FAC), maintenance busy station extension, COR of the VDN that prevents access (for example, origination restricted), FRL of a VDN that is lower than required for the AAR/ARS pattern access, no routes assigned to the AAR/ARS pattern, incompatible calling and destination partitions, ACTGA trunk group destination, or an off-net forwarding destination. If a TAC (trunk access code) destination is involved, and if the TAC is for a CO/FX trunk with a *route-to with coverage n* step, the digits entered must match a valid ARS analysis string. If not, the destination is considered invalid. For other trunk types with a *route-to number* or *route-to digits with coverage n* step, the step succeeds when the trunk is seized (that is, vector processing stops). For a *route-to with coverage y* step, the step succeeds if the TAC is assigned.

[4] On the sending switch, the call is removed from vector processing (that is, the call is taken out of any split queue and any feedback, such as music or ringback, is removed).

Detailed Call Flow for Converse—VRI Calls

H

Introduction

This appendix discusses the detailed call flow for calls involving a *converse-on* vector step and Voice Response Integration (VRI). This call flow is segmented into the following phases:

1. Converse call placement
2. Data passing (optional)
3. VRU data collection (optional)
4. Script execution
5. Data return (optional)
6. Script completion
7. DEFINITY switch data collection (optional)

NOTE:
If, during any phase of this call flow, a *converse-on* step is executed while the caller is in the split queue and an agent becomes available to service the caller, the VRU port is dropped, vector processing is terminated, and the calling party is immediately connected to the available agent.

NOTE:
If EAS is optioned, ‘‘s(S)kill’’ replaces ‘‘split.’’

Converse Call Placement

The first action taken by the *converse-on* step is to deliver the call to the converse split. Ringback tone is not heard by the caller. Any audible feedback supplied by vector processing remains until the VRU answers the call and all digits (if administered) have been outpulsed to the VRU. Vector processing is suspended. Callers remain in any nonconverse split queues, and they retain their position in queue while the converse session is active.

If a Call Prompting TN744 TTR is allocated to the call, the TTR is released. Any dial-ahead digits are discarded. However, any digits collected prior to the *converse-on* step are kept.

Calls to busy converse splits are allowed to queue. The priority of the call in queue is administrable within the *converse-on* step. Again, any audible feedback supplied by vector processing continues until the call is answered by the VRU and any data is outpulsed. Calls to busy converse splits have either no queue or a full queue fail. For this scenario, a vector event is logged, and vector processing continues at the next vector step.

Whenever a *converse-on* step places a call to an auto-available split whose agents are all logged out, the call is not queued. Instead, the *converse-on* step fails, a vector event is logged, and vector processing continues at the next vector step.

NOTE:
Usually, this scenario occurs whenever the Voice Response Unit (VRU) goes down, the ports are members of an Auto-Available Split (AAS) and the Redirection on No Answer (RONA) feature has taken all the ports out of service.

The originator's display is not changed by the terminating or answering of a converse call. Also, whenever a call is delivered to a display station via a *converse-on* step, the station displays the following information: ‘‘Originator Name to VDN Name.’’ Conventional Call Vectoring rules for Override are in effect.

Valid destinations for converse calls must be vector-controlled and include the following:

- Hunt groups
- ACD (including Auto-Available) splits
- Agent (including Auto-Available) skill groups
- AUDIX hunt groups.

Undefined and nonvector-controlled hunt group, split or skill numbers are rejected at administration time.

Any attempt to remove a hunt group, split or skill administered within a *converse-on* vector step is denied until the vector has been changed. Also, any attempt to make a hunt group, split, or skill nonvector-controlled is denied if the hunt group, split, or skill is called by a *converse-on* step.

Data Passing

NOTE:
This phase is optional and is in effect only if the application calls for DEFINITY to pass information in-band to the VRU.

The *converse-on* step may outpulse up to two groups of digits to the VRU. Two sets of digits are available for outpulsing. The digits can serve two major purposes, as follows:

- Notify the VRU of the application to be executed
- Share call-related data, such as ANI (BN) or caller digits collected by DEFINITY

In many applications both application selection and data sharing are required.

Since in many cases the digit strings are of variable length, DEFINITY always appends a pound sign (#) to the end of each digit string. *Prompt and collect* steps in the VRU script must therefore always be administered to expect the pound sign (#) as the end-of-string symbol and to include the pound sign in the digit count.

Sending the pound sign (#) prevents excessive delays and other problems caused by digit timeouts.

The complete outpulse sequence is summarized as follows:

1. VRU answers the call
2. Delay for the time administered in the ‘‘Converse first data delay’’ field in the System Parameters-Features form occurs
3. <data_1> is outpulsed
4. ‘‘#’’ is outpulsed
5. Delay for the time administered in the ‘‘Converse second data delay’’ field in the System Parameters-Features form occurs
6. <data_2> is outpulsed
7. ‘‘#’’ is outpulsed

NOTE:
DTMF tones (digits) with a length of 350_msec and with an interdigit pause of 100_msec are currently outpulsed by DEFINITY. This results in a rate of approximately 2.2 digits per second.

Any audible feedback supplied by DEFINITY is disconnected only after the outpulse sequence is completed. Also, any touch-tone dialing by the calling party during the data passing phase does not result in data corruption.

The following values may be administered for <data_1> and <data_2> within the *converse-on* command:

- **Administered digit string**: This string can contain up to six characters consisting of one or more digits (0 through 9) or asterisks (*). The pound sign (#) may not be included in a digit string because it is reserved as the end-of-string character. However, a single "#" may be administered.
- **ani**: If the call is a local call or an incoming DCS call, this data type causes the extension of the calling party to be outpulsed. If the call is an incoming ISDN PRI call with ANI (BN) provided to DEFINITY, the calling party number/billing number (CPN/BN) of the calling party is outpulsed to the voice information system. If there is no ANI (BN) to send, the end-of-string pound sign (#) is the only character outpulsed. Any other type of incoming call results in "#" being outpulsed.
- **vdn**: This data type causes the VDN extension to be outpulsed. In cases where multiple VDNs are accessed, normal VDN override rules determine which VDN extension is outpulsed.
- **digits**: This data type can be used only if Call Prompting is optioned, and it causes the most recent set of digits collected in vector processing to be outpulsed. If no digits are available, the end-of-string pound sign (#) is the only character outpulsed.
- **qpos**: This data type causes the value of the queue position of a call in a nonconverse split to be outpulsed. This value is a variable length data item from which between one and three digits can be outpulsed. Valid ranges for the value are 1 through 200 in G3iV2, G3sV2, and G3vsV2 and 1 through 999 in G3rV2. If the call is not queued, the end-of-string pound sign (#) is the only character outpulsed.

  NOTE:
  The use of this keyword is not recommended with multiple split queuing because any queue position value sent may not be meaningful. However, if the call is queued to multiple nonconverse splits, the value of the caller's queue position in the first nonconverse split is sent.

  This data may be used by the voice information system to inform callers of their position in queue or to decide whether to execute a long or short version of a voice response script.

- **wait**: This data type sends the expected wait time for a call in vector processing that is queued to at least one split. It is a value from 0 to 9999 seconds. If the call is not queued, or is queued only to splits with no working agents, only the # is outpulsed.
- **‘‘#’’**: This is the only character outpulsed. Outpulsing this character causes the corresponding prompt and collect command in the voice response script to be skipped.
- **‘‘none’’**: This data type causes no characters to be outpulsed. Also, no end-of-string pound character (#) is outpulsed, and no time delays are invoked.

DEFINITY always outpulses a pound character (#) at the end of each digit string. Where ‘‘#’’ is administered, or where the ‘‘digits’’ keyword is administered and the last digit collected from the caller is ‘‘#,’’ only one ‘‘#’’ is outpulsed. No ‘‘#’’ is outpulsed when the keyword ‘‘none’’ is administered.

If <data_1> is administered as ‘‘none,’’ <data_2> must also be ‘‘none.’’

Any data to be passed to the VRU from DEFINITY is outpulsed in-band. Two time delays on the System Parameter-Features form (‘‘Converse first data delay’’ and ‘‘Converse second data delay’’) are administrable by customers. These delays may range from 0 through 9 seconds, with a default of zero seconds for the converse first data delay and a default of two seconds for the converse second data delay. The delays may be needed to give the VRU time to invoke an application and allocate a touch-tone receiver to receive the passed digits.

If <data_1> is not ‘‘none,’’ the converse first data delay timer starts when the call is answered by the VRU. Once the timer expires, the data_1 digits are outpulsed in-band to the VRU, followed by the end-of-string pound sign (#).

If <data_2> is not ‘‘none,’’ the converse second data delay timer starts when the end-of-string pound sign (#) from the first digit string is outpulsed. Once the timer expires, the data_2 digits are outpulsed in-band to the VRU, followed by the end-of-string pound sign (#).

No time delays are invoked when the keyword ‘‘none’’ is administered.

NOTE:
The outpulsing of digits is not heard by the caller.

If the VRU hangs up during the data passing phase, DEFINITY will log a vector event, reactivate vector processing at the next vector step, and ensure the VRU port is accessible for future calls.

Once all digits have been passed to the VRU, any audible feedback is disconnected.

NOTE:
At this point, control has effectively been passed to the VRU.

To ensure the robust operation of the VRU data passing operation, be sure to implement the following recommendations:

- Include the *prompt and collect* command in the VRU script for each data field passed in the *converse-on* step.
- Administer each *prompt and collect* command to recognize the ‘‘#’’ character as the end-of-string character.
- Ensure the number of digits expected is one greater than the number of digits passed to allow for the ‘‘#’’ character, which terminates every converse data field.

  Also, ensure no announcement is played in these *prompt and collect* steps.
- Ensure the first digit timeout in the *prompt and collect* steps is five seconds greater than the corresponding converse data delay. (For example, if the *converse-on* step passes two data fields, and if the converse first data delay is 0 secs and the converse second data delay is 4 secs, the first digit timeouts for the two *prompt and collect* commands should be at least 5 and 9 seconds, respectively.)
- Ensure the interdigit timeout in the *prompt and collect* steps is at least five seconds.
- Administer the converse first data delay to give a VRU under a heavy load sufficient time to allocate a DTMF touch-tone receiver after answering the call.
- Administer the converse second data delay to give a VRU under a heavy load sufficient time to complete any tasks between the first and second *prompt and collect* command. (For example, the VRU can invoke a new application if the first data field passed is used to identify the application script to be executed.)
- In general, for *converse-on* steps pass data to the VRU, ensure the VRU script does not execute any commands between the time the call is answered and the time when the first *prompt and collect* command is executed.

VRU Data Collection

When digits are passed from DEFINITY to the VRU, the first VRU script commands executed are *answer phone* and *prompt and collect*. No announcement is programmed for the *prompt and collect* command, and the pound sign (#) is programmed as the end-of-string sign. If two sets of digits (that is, <data_1> and <data_2>) are passed by DEFINITY, there will be two *prompt and collect* commands on the VRU to receive them.

If the first digit string (<data_1>) passed to the VRU is for application selection, the CONVERSANT Script Builder *exec* command invokes the appropriate script. If a second digit string (<data_2>) is also used to pass an argument to this selected application, the first command in the *exec*'ed script is a *prompt and collect* command with no announcement prompt programmed and with the pound sign (#) programmed as the end-of-string character.

The ‘‘Converse second data delay’’ is used to give the VRU time to invoke the selected application before the <data_2> digit string is outpulsed.

The application developer should ensure the administered *converse first data delay* and *converse second data delay* timers allow sufficient time for the VRU to successfully collect all outpulsed digits, even during periods of heavy call volume. Loss of digits from <data_2> is an indication the converse second data delay timer needs to be increased.

Script Execution

During script execution, digits input by the calling party in response to *prompt and collect* commands are collected by the VRU but are not collected by DEFINITY as dial-ahead digits. Also, audible feedback is determined by the VRU.

If an agent from a nonconverse split becomes available to service the call while the VRU script is being executed, the VRU port is dropped from the call, and the caller is immediately connected to the agent. Any digits collected prior to executing the *converse-on* step are still available and may be displayed using the CALLR-INFO button.

The entire call is dropped if the caller abandons during the execution of a *converse-on* step.

Data Return

This phase is optional and is in effect only if the application calls for the VRU to return information to DEFINITY before returning control to vector processing.

Digits returned by the VRU are treated as dial-ahead digits. The rules for collecting and processing VRU-returned digits are identical to those for collecting and processing Call Prompting digits (see Chapter 5, "Call Prompting").

VRU data return is done in a manner similar to an analog transfer. Specifically, the VRU does an analog switchhook flash, outpulses DTMF digits, and then hangs up. If converse data is returned, the DTMF digits comprise two parts. The first sequence of digits is the converse data return feature access code administered on the DEFINITY Feature-Access-Codes form. The second sequence of digits is the sequence to be passed by the VRU. These digits are collected later during vector processing.

The CONVERSANT VRU offers a built-in external function called ‘‘converse_data.’’ This function allows applications developers to perform this operation in a convenient and robust fashion.

To ensure the robust operation of the VRU data return operation, be sure to follow these recommendations:

- Set the analog flash timing to 600 msecs
- Ensure DTMF tones last at least 70 msec
- Ensure interdigit pauses last at least 50 msec
- (CONVERSANT only) Use the ‘‘converse_data’’ external function to return data to DEFINITY

For applications involving VRUs other than CONVERSANT VRUs, be sure to follow these recommendations:

- After the flash, ensure the VRU performs dialtone detection (stutter dialtone) for five seconds before outpulsing the converse data return feature access code.
- If no dialtone is received before the timeout, ensure the VRU does two more retries of the analog flash. Also, if no dialtone is detected after two retries, ensure the VRU logs an error.
- Whenever dialtone is detected, ensure the digits of the converse data return feature access code are outpulsed.
- After the converse data return feature access code is outpulsed, ensure the VRU performs dialtone detection (steady dialtone) again for five seconds before outpulsing the returned digits. If no dialtone is detected, ensure an error is logged by the VRU.
- Whenever dialtone is detected, ensure the VRU digits are outpulsed and the line to DEFINITY is dropped.

The TN744 Call Classifiers required by the Call Prompting feature are not required for returning digits in-band from the VRU to DEFINITY. Instead, general purpose TTR boards (TN748s) are used. As long as dial-ahead digits are available, any *collect digits* steps following a *converse-on* step do not require a TN744 to be allocated to the call.

If no general purpose TTRs are immediately available, and if the call queues for a TTR, no dial tone is provided. For this scenario, the VRU does not outpulse any digits until a TTR is available and dial tone is provided.

If there are no general purpose TTRs available on DEFINITY, and if there is no space in the TTR queue, the operation fails. Usually, the VRU logs an error and then quits, and vector processing continues at the next vector step. Existing system measurements reports inform customers when the system is configured with an insufficient number of TTRs.

The ‘‘Converse Data Return Code’’ can be followed by a maximum of 24 digits. The VRU touch-tones the code and the digits in-band. However, the code and the digits are not heard by the caller. The digits are stored in DEFINITY as Call Prompting dial-ahead digits. If ‘‘x’’ digits are collected by vector processing before the *converse-on* step is executed, the maximum number of digits that can be returned is reduced to ‘‘24-x.’’ Any additional digits returned by the VRU are discarded. The data return is completed once the VRU hangs up.

The digit string returned by the VRU can consist of the digits (0 through 9) and pound signs (#). The pound sign (#) is interpreted by the *collect digits* step as an end-of-string character. If the digit string being returned is of variable length, the VRU can terminate the string with a pound sign (#) to avoid the ten second timeout delay that occurs when the digits are collected. If the digit string being returned is ‘‘multi-part’’ (that is, to be collected by multiple *collect digits* steps), and if some of the parts are of variable length, the pound sign (#) can be used to terminate each of the variable length parts.

NOTE:
An asterisk (*) may be included as part of the converse data return code. However, since the asterisk is interpreted as a ‘‘delete’’ character by DEFINITY, it makes little sense to use it as a returned digit. If it is used as such, all characters returned prior to the asterisk are discarded.

During the data return phase, the caller is temporarily put on hold. Music-on-hold, if administered, is suppressed. Since the caller hears silence during this phase, feedback should be provided to the caller as soon as possible after the *converse-on* step is executed.

Any touch-tone digits dialed by the calling party during the data return phase are discarded. These digits do not cause data corruption, and they are not collected as dial-ahead digits by DEFINITY.

If an interdigit timeout occurs during the data return phase, the switch logs a vector event, keeps the digits already returned, drops the VRU, and reactivates vector processing at the next vector step.

If the timeout occurs before the converse data return code is returned, the operation is the same except that no discarded digits will be available.

Script Completion

The VRU script returns control to vector processing on DEFINITY by simply hanging up the line. In cases where no data is returned to DEFINITY, this is done usually by executing the *quit* command. In cases where data is returned, this occurs whenever the VRU hangs up on completion of the VRU data return operation.

The last set of digits collected before the *converse-on split* step is executed is still available and may be displayed by an answering agent on the nonconverse split by using the CALLR-INFO button.

A VRU script can be programmed to continue running after hanging up the voice line. This after-call work is usually very short, and it may involve either a final message to a host or a final update to a local database. For this scenario, the VRU port (channel) is still associated with the running script even though there is no longer a voice connection.

From DEFINITY's point of view, the agent (port) is available for the next call. If a call is delivered to this port, the VRU does not answer the call until the previous script has completed. As long as the VRU script's after call work is short in duration, this poses no significant problem for the VRI feature. However, high volume VRI applications with lengthy after call work periods should be avoided, especially if such periods are so lengthy they approach the administered timeout period on DEFINITY for the Redirection on No Answer (RONA) feature. In such a case, RONA might think the VRU ports are faulty and might therefore start to take these ports out of service.

DEFINITY Switch Data Collection

NOTE:
This phase is in effect only if the VRU returns information to DEFINITY.

Once the VRU script has completed and vector processing is reactivated, the returned digits are collected and processed by vector commands in the usual manner. Since the digits must be collected by a *collect digits* command, data may be returned and processed only if the Call Prompting option is enabled.

The data returned can consist of multiple parts. For example, the VRU could return a stream of seven digits in which a single digit success/fail code is followed by a six-digit account code. For this scenario, the *converse-on* step would be followed by a sequence of vector steps including two *collect digits* steps. The first *collect digits* step would collect one digit and then check the result code; the second *collect digits* step would collect the six-digit account code.

Any touch-tone digits dialed by the calling party during the data collection phase are discarded, do not cause data corruption, and are not collected as dial-ahead digits by DEFINITY.

If VRU data is returned, the calling party is able to touch-tone a response to a DEFINITY prompt only after the data collection phase is completed and another *collect digits* step is executed. This is true because each executed *collect digits* step does not allocate a TTR when dial-ahead digits are present. Since VRU-returned digits are treated as dial-ahead digits, a TTR is attached to the call only after all returned digits are collected and another *collect digits* step is encountered. Only at this point can the caller hear an announcement for the *collect digits* command and successfully enter digits.

Security Issues

I

Introduction

There are a number of security issues vis-a-vis Call Vectoring that should be noted. For example, Call Vectoring and Call Prompting can be used to help prevent unauthorized users from gaining access to the switch via the Remote Access feature. This appendix explains how this is done.

NOTE:
For more information on security issues, refer to the *GBCS Products Security Handbook*, 555-025-600.

Remote Access

Abuse of remote access on the switch is one of the main methods by which unauthorized users obtain telephone services illegally. This section explains how a number of Call Vectoring features can be used to prevent unauthorized use of the DEFINITY remote access feature. No new development is required for any of these services.

Two methods are available, as follows:

- Front-ending remote access (that is, reaching the remote access extension via Call Vectoring)
- Replacing the function of the remote access extension by one or more call vectors.

Front-Ending Remote Access

Via this method, authorized external callers are given a VDN extension to call instead of the remote access extension, which is kept private. The corresponding call vector can then implement a number of security checks before routing callers to the remote access extension. Routing can be done via a *route-to number* or *route-to digits* step.

The following advantages are possible via this method.

- Call Vectoring can introduce a delay before the dial-tone is provided to the caller. Immediate dial-tone is often one criterion searched for by a hacker's programs when the hacker is trying to break into a system.
- A recorded announcement declaring that the use of the switch services by unauthorized callers is illegal and that the call is subject to monitoring and/or recording can be played for the caller.
- Call Prompting can be used to prompt for a password. In such a case, the call is routed only if there is a match on the password.
- Use of the remote access extension can be limited to certain times of the day or certain days of the week.
- Real-time and historical reports on the use of the remote access feature can be accessed from BCMS and/or CMS.
- Different passwords can be used on different days of the week or at different times during the day.
- Many VDNs that call the remote access extension can be identified. Accordingly, individuals or groups can be given their own VDN with unique passwords, permissions and reports. Any abuse of the system or security leak can then be attributed to an individual or a group.
- The caller can be routed to a VRU using the *converse-on* step where more sophisticated security checking, such as speaker recognition, can take place.
- Anyone failing any of the security checks can be routed to a ''security" VDN that routes the caller to security personnel with a display set or to a VRU. Such a call would show ''security'' and possibly also the attempted password on the display. If the call is passed to a VRU, the VDN, the ANI and/or the prompted digits can be captured. BCMS/CMS reports on this security violation VDN will give information on how often and when security violations occur.

Replacing Remote Access

For this method, the remote access extension is not used. One or more VDNs are designed to access call vectors that can employ all of the security checks described in the previous section. The same reports and monitoring/recording capabilities described in the previous section can also be used. Instead of routing to the remote access extension, the vector collects digits from the caller and then routes to the given destination if there is a match on the password.

Again, multiple VDNs can be created for individuals or groups with different security checks and different permissions and/or restrictions. Destination numbers provided by callers can be screened by the vectors and denied if the user does not have permission to access that destination. For example, an individual user could be restricted to placing calls to numbers beginning with area codes “303” and “908.”

EAS

With EAS, agents’ voice terminals can be locked when the terminal is not staffed. This is accomplished by assigning the voice terminals a Class of Restriction that does not allow outbound calls or it could be restricted from toll calls.

EAS agents have an optional password of up to nine digits to log in. This password is not displayed on DCP terminals when the agent is entering the password on the dial pad.

Vector Initiated Service Observing Security

The following restrictions can be used with vector initiated Service Observing (G3V4 and later releases) to guard against unauthorized use.

- Call prompting commands can be used in Service Observing vectors to provide passcode protection, and to limit access to observing specific destinations or verified caller entered digits.
- Time of Day/Day of Week checks can be incorporated in Service Observing vectors.
- A vector can be created to be used exclusively for Service Observing.
- For a VDN to be observed as the result of a route-to command, the VDN must have a COR that allows it to be observed.
- The calling permissions of the COR assigned to the Service Observing VDN in conjunction with the “can be observed” settings of the COR assigned to the destination determine what agents, stations, or VDNS can be observed.

Setting Up a Call Center

J

Introduction

Call center managers need some key indicators to measure ACD performance at their site. Usually, in setting up a call center, several factors involving call management are considered. The following list identifies and defines the most common of these factors, and it provides a typical question that might be asked. In addition, an insurance company example will be used to discuss the different options in this chapter.

- **Volume**

  Number of calls going in or out of the ACD. (How many calls did Split 1 answer?)

- **Productivity**

  Call volume per unit of time. (How many calls did Split 1 answer between 8 a.m. and 9 a.m.?)

- **Utilization**

  Overall use of the phone center. (What was my agent occupancy?)

- **Accessibility**:

  Availability of lines and agents when customers call the ACD (this is an area that CMS can probably most clearly define and help improve). (Were lines busy when customers called or did they have to wait too long?)

- **Quality of Service**:

  Accuracy of information, a pleasant manner, responsiveness to caller concerns, successful completion of business, and efficient time utilization (not all measured directly by CMS). (Was the caller given good service?)

This chapter explains how to set up a call center for customers with Call Vectoring and/or Expert Agent Selection (EAS).

Call Vectoring/Non-EAS Option

To set up a call center that has Call Vectoring but not EAS, do the following:

1. Determine your call center’s objectives. Think about how you want your call center to handle calls and also about what you want your call center to achieve. See "Non-EAS Worksheet #1: Call Center Objectives Worksheet".

   A company’s basic goals are to increase profits and market share and to decrease costs. The purpose of setting up a call center is to monitor these goals using CMS reports. It is best to have more than one objective. (Some customers set and then live by only one objective.) Call center objectives must then be created to meet the goals. These objectives must be communicated to the Split Supervisor or to the Administrator managing the call center.

   The following list provides an example set of call center objectives:

   - Establish the following measured entities:
     - Average Speed of Answer = 15 seconds
     - Abandon Rate <= 3%
     - Average Talk Time = 2 1/2 minutes
     - ACD calls per agent = 80 to 90 per day
     - Number of calls in queue = 6
     - Percentage of calls answered within the service level = 95%
     - Agent occupancy > 90%
     - Percentage of trunks busy < 3%
   - Generate revenue through the call center.
   - Train agents to back up each other.
   - Adequately train agents to provide service that meets customer expectations.
2. Review your existing operation and determine your customer/call center needs (see "Non-EAS Worksheet #2: Current Split Operation Worksheet" and the following table).
3. On the switch, assign a unique Hunt Group number and Call Distribution method to each caller need. This number will be your split number (see "Non-EAS Worksheet #3: Customer Needs Worksheet" and the following table).
4. Assign DNIS (Dialed Number Identification Service) (that is, the number dialed) as a Vector Directory Number (VDN) (see the following table).

As an option, you can assign one VDN for a main number and use Call Prompting to route the call to the proper split.

The following table illustrates the guidelines given up to this point.

Customer/Call Center Needs	Split Number (Hunt Group)	Call Distribution*	VDN
New policy	1	UCD	555-6543
Questions about policy, Rate Quotes, Billing	2	UCD	555-6432
Spanish speaking for policy, service, and claims	3	DDC	555-6321
Claims	4	UCD	555-6210

* Options include Direct Department Calling (DDC) and Uniform Call Distribution (UCD).

Notice that this call center has only one split for all Spanish calls. However, resources permitting, you could create a New Policy split, a Service split, and a Claims split, each containing agents who speak Spanish. As an alternative, you could use one main VDN to point to a Call Prompting vector designed to route the calls to the splits.

5. On the switch, assign extensions to the agents’ physical terminal locations (see the following table).
6. In CMS: Dictionary: Login Identifications, assign each agent a unique loginID (see the following table).

   Agents are known to CMS by the loginID. If assigned, reports refer to an agent by name, not by loginID.

   The following table illustrates the assignments described in the previous two items:

Agent Name*	Extension	LoginID*
Randy Tyler	1231	2000
Cathy Smith	1232	2001
Carla Silva	1238	2002

* = assigned in CMS Dictionary

NOTE:
When you are adding names to extensions on the switch, the agent name should be the same name as the loginID assigned to CMS.

7. On the switch, assign agent extensions to splits (see the following table).

On a G3 system, more than four splits can be assigned to an agent; however, the agent can log into a maximum of four splits. An agent assignment to splits can be changed in CMS: ACD Administration: Move Extensions Between Splits if the agent is logged off.

The following table illustrates the assignment of agent extensions to splits:

Split (Hunt Group)	Agent Extensions
1 - Sales	1231, 1232, 1233, 1234, 1235, 1236, 1237, 1238, 1239
2 - Service	1231, 1232, 1234, 1238, 1239, 1240
3 - Spanish	1238, 1240, 1245
4 - Claims	1238, 1239, 1240, 1241, 1242

8. On the switch or in CMS: ACD Administration: VDN Assignments, assign a vector to each VDN (see the following table).

   A VDN can point to only one vector. However, a vector can have more than one VDN pointing to it.

   The following table illustrates VDN/vector assignments.

VDN	Vector
6543	1 (Sales)
6432	2 (Service)
6321	3 (Spanish)
6210	4 (Claims)

9. On the switch or in CMS: ACD Administration: Vector Contents, write your vectors. See "Non-EAS Worksheet #4: Vector Design Worksheet".

   Your vectors should match your call center objectives. To meet these objectives, you must make a number of relevant decisions (for example, you may decide how soon you want to enlarge an agent pool or what kind of treatment the caller should receive). If your VDN and vector reports do not satisfy your call center objectives, you must consider your alternatives (for example, you may deem it necessary to train agents or to increase the amount of time elapsed from when a call queues to one split and then to another split).

   The following lists indicate the actions produced by two different vectors:

   Actions Produced by Vector #1:

   1. Tell the caller to select one of the following prompts:
      - 1 = Sales

- 2 = Service
- 3 = Spanish
- 4 = Claims
- Nothing or 0 = Service

2. Queue the call.
3. Provide an announcement to the caller.

10. **Actions Produced by Vector #2:**
    1. Queue the call to the correct service at a medium priority.
    2. If no agents are available, provide a message and then play music.
    3. If the call is not answered within 10 seconds, provide a second message and then play music.
    4. If the call is not answered within 7 more seconds, queue the call to the Service split.
    5. If the call is not answered within 7 more seconds, queue the call to the Spanish split at a high priority.

    NOTE:
    In G3, a *check-backup split* command queues the call to up to three splits if the conditions are met. If the conditions are not met, the *check-backup split* command may not get read again (if the vector step in which it appears is not executed again).

11. In CMS: Dictionary, assign names to the splits, VDNs, and vectors.
12. Once your system is up and operational, you will need to monitor it to ensure you are meeting your call center objectives. CMS can be used to monitor many of your objectives. See for more details. Some objectives will need to be monitored and have adjustments made in real time. For example, if the number of calls waiting, average speed of answer, or percent answered within a service level is not meeting your objectives, you might want to immediately move some agents, direct calls to another vector, or look-ahead interflow some calls. Other items such as agent occupancy and percent all trunks busy may only need to be monitored daily to look for trends.

Non-EAS Worksheet #1
Call Center Objectives Worksheet

What Are My Call Center Objectives?

Figure J-1. Non-EAS Worksheet #1: Call Center Objectives Worksheet

Non-EAS Worksheet #2
Current Split Operation Worksheet

Split ______________

Primary Backup ______________ **Secondary Backup** ______________ **Tertiary Backup** ______________

List Individually You Customer/ Caller Needs and Your Agent Skill Sets in this Split	**Do You Have Agent Expertise? (Yes/No)**	**Do You Want to Separate Skill Set with EAS? (Yes/No)**
1.		
2.		
3.		
4.		
5.		
6.		

Split ______________

Primary Backup ______________ **Secondary Backup** ______________ **Tertiary Backup** ______________

List Individually You Customer/ Caller Needs and Your Agent Skill Sets in this Split	**Do You Have Agent Expertise? (Yes/No)**	**Do You Want to Separate Skill Set with EAS? (Yes/No)**
1.		
2.		
3.		
4.		
5.		
6.		

Figure J-2. Non-EAS Worksheet #2: Current Split Operation Worksheet

Non-EAS Worksheet #3
Customer Needs Worksheet

Customer/Call Center Needs	Split Number (Hunt Group)	Call Distribution	VDN

Figure J-3. Non-EAS Worksheet #3: Customer Needs Worksheet

Vector # ____________ Name ____________________ Description ________________________________

Assigned VDNs ________________________________ Assigned Trunk Groups __________________________

1. ____
2. ____
3. ____
4. ____
5. ____
6. ____
7. ____
8. ____
9. ____
10. ____
11. ____
12. ____
13. ____
14. ____
15. ____
16. ____
17. ____
18. ____
19. ____
20. ____
21. ____
22. ____
23. ____
24. ____
25. ____
26. ____
27. ____
28. ____
29. ____
30. ____
31. ____
32. ____

Figure J-4. Non-EAS Worksheet #4: Vector Design Worksheet

Call Vectoring/G3 EAS Option

To set up a call center that has both Call Vectoring and G3V2 EAS, do the following:

1. Determine your call center's objectives. Think about how you want your call center to handle calls and also about what you want your call center to achieve. See "EAS Worksheet #1: Call Center Objectives Worksheet".

   A company's basic goals are to increase profits and market share and to decrease costs. The purpose of setting up a call center is to monitor these goals using CMS reports. It is best to have more than one objective. (Some customers set and then live by only one objective.) Call center objectives must then be created to meet the goals. These objectives must be communicated to the Split Supervisor or to the Administrator managing the call center.

   The following list provides an example set of call center objectives:

   - Establish the following measured entities:
     - Average Speed of Answer = 15 seconds
     - Abandon Rate <= 3%
     - Average Talk Time = 2 1/2 minutes
     - Expected Wait Time < 30 seconds
     - ACD calls per agent = 80 to 90 per day
     - Number of calls in queue = 6
     - Percentage of calls answered within the service level = 95%
     - Agent occupancy > 90%
     - Percentage of trunks busy < 3%
   - Generate revenue through the call center.
   - Train agents to back up each other.
   - Adequately train agents to provide service that meets customer expectations.

2. Review your existing operation and determine your customer/call center needs and your business needs. Also, determine if all of these needs require skills. See "EAS Worksheet #2: Current Split Operation Worksheet".

   The call center could have many more skills than those indicated in the following table. However, we'll assume that the insurance company represented in the table services only certain states on the West Coast. The point is that EAS allows you to expand on your customer/call center needs. An example of a business need is a temporary promotion.

3. Assign a skill hunt group and call distribution method to each set of needs. See "EAS Worksheet #3: Customer needs Worksheet".

With EAS, most hunt groups tend to use EAD to allow callers to reach the most expert agent. However, UCD might be used if all agents assigned a skill are equally trained and if you want equal call distribution to the agents.

4. In CMS: Dictionary: Splits/Skills, assign names to the skills.

   The following table illustrates the guidelines provided up to this point.

Customer/Call Center Needs	Skill Name	Skill Number	UCD/ EAD	COR
New policy — West	Sales West	1	EAD	
New policy — West/Coast	Sales Coast	10	EAD	
New policy — West/OR	Sales OR	11	EAD	
New policy — West/CA	Sales CA	12	EAD	
New policy — West/AZ	Sales AZ	13	EAD	
New policy — West/Mountains	Sales Mountains	20	EAD	
New policy — West/CO	Sales CO	21	EAD	
New policy — West/TX	Sales TX	22	EAD	
New policy — West/NM	Sales NM	23	EAD	
Questions (Rate Quotes, Billing)	Service	30	EAD	
Questions (Rate Quotes, Billing)	Service Coast	31	EAD	
Questions (Rate Quotes, Billing)	Service Mountains	32	EAD	
Claims	Claims	2	EAD	
Spanish Speaking	Spanish	50	UCD	
Spanish Speaking Sales	Spanish Sales	51	EAD	
Spanish Speaking Service	Spanish Service	52	EAD	
Spanish Speaking Claims	Spanish Claims	53	EAD	

Customer/Call Center Needs	Skill Name	Skill Number	UCD/ EAD	COR
Sales/Service	Sales/Service	3	EAD	
Service/Claims	Service/Claims	4	EAD	
Super Group/All Skills	Super	5	EAD	

5. On the switch, administer the VDNs. On the switch or in CMS: ACD Administration, change the VDN Skill Preferences and assign up to three skills to each VDN. See "EAS Worksheet #6: Skill Preferences Worksheet". Then assign a VDN Skill Preference (either 1st, 2nd, or 3rd) to each VDN to establish which skills are ‘‘primary,’’ ‘‘secondary,’’ or ‘‘tertiary.’’

6. On the switch or in CMS, assign a vector to each VDN.

   The following tables illustrate a sample of the VDNs for the two previous numbered items:

Main VDNs	1st Skill	2nd Skill	3rd Skill	Vector
6543	Sales AZ	Sales Coast	Sales West	1
New policy AZ	13	10	1	
6432	Sales CA	Sales Coast	Sales West	1
New policy CA	12	10	1	
6321	Sales OR	Sales Coast	Sales West	1
New policy OR	11	10	1	
6210	Service Coast	Service/ Claims	Service/Claim	2
Questions Coast	31	30	4	
6123	Claims	Service/ Claims	Spanish Claims	2
Claims	2	4	53	

VDN 6234

The caller can enter a generic VDN and be prompted. The following table illustrates prompting for Spanish callers, where callers would be prompted for type of service they require.

Prompting Digit	VDN Accessed From Vector	1st Skill	2nd Skill	3rd Skill	Vector
1	6651	Spanish Sales	Spanish		3
		51	50		
2	6652	Spanish Service	Spanish		3

Prompting Digit	VDN Accessed From Vector	1st Skill	2nd Skill	3rd Skill	Vector
		52	50		
3	6653	Spanish Claims	Spanish		3
		53	50		

The previous tables contain only a few examples of the VDNs (and, accordingly, the VDN Skill Preferences assigned to the VDNs) that the call center could assign. Be sure to determine the VDNs and VDN Skill Preferences (primary and backup skills) that you require. Also, be sure to determine if the VDNs can share a vector (as is illustrated by the second table) or if some VDNs require a separate vector (as illustrated by the first table). The second table demonstrates that individual VDNs can be accessed from within a single vector. This is accomplished via Call Prompting digits.

Note that the switch link does not have to be taken down to assign skills and Skill Preferences. Also, if you change these assignments once they are administered, the changes take effect immediately, even if there are calls in queue. As a result, calls in queue can be assigned another VDN skill, and they can be queued to another skill hunt group.

7. On the switch, administer a dial plan that differentiates between physical extensions and loginIDs. Assign a loginID to each agent. Also, determine if the agent needs the Direct Agent feature. (Will agents be taking callbacks or transferring to each other, and do you want those to be treated as ACD calls?) If so, determine the login and logout

   coverage for each agent (which can be a VDN, a skill hunt group, or AUDIX). In CMS, assign to each agent a loginID (to enable the printing of the agent names on the reports). See "EAS Worksheet #4: Individual Agent Skill Worksheet Generic 3".

   The following table illustrates dial plan administration:

Agent Name	LoginID	Direct Agent	Login Coverage Pt1/Pt2/Pt3	Logout Coverage Pt1/Pt2/Pt3
Randy Tyler	2000	Yes	2/6543/AUDIX	AUDIX
Cathy Smith	2001	Yes	4/6012/AUDIX	AUDIX
Carla Silva	2002	No	--------	-------
Trish Carara	2003	No	--------	--------

The loginID assigned on the switch and in CMS to the agent is the Logical Agent ID. The ID name assigned in the switch should match the ID name assigned in the CMS Dictionary.

It is strongly recommended that an agent with Direct Agent status be assigned a ‘‘Direct Agent skill’’ as the primary skill (see the following table). This way, Direct Agent calls will not be sharing queue slots with other skill calls.

8. Determine which agents you want to answer calls in each skill hunt group. On the switch, assign each agent up to four skills, and assign each skill a primary or secondary status.

   The agent always answers primary calls before secondary. If you want to give preference to some call types, assign them primary status. Primary can be thought of as expertise level, too, so route to these agents first when multiple agents are available. See "EAS Worksheet #5: Agent Skills Worksheet Generic 3".

   The following table illustrates skill status assignments:

Agent Name	Skill	Skill Assigned
Randy Tyler	12 — Sales CA	Primary
	10 — Sales Coast	Secondary
	22 — Sales TX	Secondary
Cathy Smith	21 — Sales CO	Primary
	23 — Sales NM	Primary
	1 — Sales West	Secondary
Carla Silva	30 — Service	Primary
	4 — Claims	Secondary
	40 — Spanish	Secondary
Trish Carara	53 — Spanish Claims	Primary
	4 — Claims	Primary
	8 — Service/Claims	Secondary

NOTE:
This table indicates a small sample of agents in the call center.

Agent skill assignments can be modified from within CMS: ACD Administration: Change Agent Skills. The agent must log out (if he or she is already logged in) and then log back in for the changes to take effect.

9. On the switch or in CMS: ACD Administration: Vector Contents, write your vectors. See "EAS Worksheet #7: Vector Design Worksheet".

   Your vectors should match your call center objectives. To meet these objectives, you must make a number of relevant decisions (for example, you must decide how soon you want to enlarge an agent pool or what kind of treatment the caller should receive). If your VDN and vector reports do not satisfy your call center objectives, you must consider your alternatives (for example, you may deem it necessary to train additional agents or to increase the amount of time elapsed from when a call queues to one skill hunt group and then to another skill hunt group).

The following list indicates the actions produced by a vector:

1. Queue the call to the 1st main skill hunt group (Sales).
2. If no agents are available, provide a message and then play music.
3. If the call is not answered within 10 seconds, provide a second message and then play music.
4. If the call is not answered within 7 more seconds, queue the call to the 2nd main skill hunt group (Service).
5. If the call is not answered within 7 more seconds, queue the call to the 3rd main skill hunt group (Claims).

10. In CMS: Dictionary, assign names to the skills, VDNs, vectors, and loginIDs.

11. Once your system is up and operational, you will need to monitor it to ensure you are meeting your call center objectives. CMS can be used to monitor many of your objectives. See for more details. Some objectives will need to be monitored and have adjustments made in real time. For example, if the number of calls waiting, average speed of answer, or percent answered within a service level is not meeting your objectives, you might want to immediately move some agents, direct calls to another vector, or look-ahead interflow some calls. Other items such as agent occupancy and percent all trunks busy may only need to be monitored daily to look for trends.

EAS Worksheet #1
Call Center Objectives Worksheet

What Are My Call Center Objectives?

Figure J-5. EAS Worksheet #1: Call Center Objectives Worksheet

EAS Worksheet #2
Current Split Operation Worksheet

Split

Primary Backup **Secondary Backup** **Tertiary Backup**

List Individually You Customer/ Caller Needs and Your Agent Skill Sets in this Split	**Do You Have Agent Expertise? (Yes/No)**	**Do You Want to Separate Skill Set with EAS? (Yes/No)**
1.		
2.		
3.		
4.		
5.		
6.		

Split

Primary Backup **Secondary Backup** **Tertiary Backup**

List Individually You Customer/ Caller Needs and Your Agent Skill Sets in this Split	**Do You Have Agent Expertise? (Yes/No)**	**Do You Want to Separate Skill Set with EAS? (Yes/No)**
1.		
2.		
3.		
4.		
5.		
6.		

Figure J-6. EAS Worksheet #2: Current Split Operation Worksheet

EAS Worksheet #3
Customer Needs Worksheet Generic 3

Customer/Call Center Needs	Split Name	Split Number	UCD/EAD	Class of Restriction

When defining your skills, be aware of how you can combine skills for backup purposes.

Figure J-7. EAS Worksheet #3: Customer needs Worksheet

EAS Worksheet #4
Individual Agent Skill Worksheet Generic 3

Agent Name	Login ID	Direct Agent?	Class of Restriction	Login Coverage			Login Coverage		
				Point 1	Point 2	Point 3	Point 1	Point 2	Point 3

Figure J-8. EAS Worksheet #4: Individual Agent Skill Worksheet Generic 3

EAS Worksheet #5
Agent Skills Worksheet Generic 3

	Agent name	Login ID	First* Skill (P/S)	Second Skill (P/S)	Third Skill (P/S)	Fourth Skill (P/S)	Class of Restriction
1.							
2.							
3.							
4.							
5.							
6.							
7.							
8.							
9.							
10.							
11.							
12.							
13.							
14.							
15.							
16.							
17.							
18.							
19.							
20.							
21.							
22.							
23.							
24.							
25.							
26.							
27.							
28.							
29.							
30.							
31.							
32.							

* Direct agent calls go to this skill first.

Figure J-9. EAS Worksheet #5: Agent Skills Worksheet Generic 3

EAS Worksheet #6
VDN Skill Preferences Worksheet

VDN	VDN Name	Class of Restriction	Skill Preferences			Vector
			1st Skill Number	2nd Skill Number	3rd Skill Number	

Figure J-10. EAS Worksheet #6: Skill Preferences Worksheet

EAS Worksheet #7
Vector Design Worksheet

Vector # ____________ **Name** ____________________ **Description** ________________________________

Assigned VDNs ______________________________ **Assigned Trunk Groups** ____________________

1. ______________________________
2. ______________________________
3. ______________________________
4. ______________________________
5. ______________________________
6. ______________________________
7. ______________________________
8. ______________________________
9. ______________________________
10. ______________________________
11. ______________________________
12. ______________________________
13. ______________________________
14. ______________________________
15. ______________________________
16. ______________________________
17. ______________________________
18. ______________________________
19. ______________________________
20. ______________________________
21. ______________________________
22. ______________________________
23. ______________________________
24. ______________________________
25. ______________________________
26. ______________________________
27. ______________________________
28. ______________________________
29. ______________________________
30. ______________________________
31. ______________________________
32. ______________________________

Figure J-11. EAS Worksheet #7: Vector Design Worksheet

Converting a Call Center to EAS

K

Introduction

The procedures in this appendix provide guidelines for upgrading a call center from a non-EAS ACD environment to an EAS ACD environment. The primary activities involved in this conversion are:

- Step 1: Pre-EAS Cutover Administration for the G3V2 (or later) Switch
- Step 2: Pre-EAS Cutover Administration for CMS
- Step 3: Pre-EAS Cutover Administration for AUDIX
- Step 4: Pre-EAS Cutover Administration for Messaging Server
- Step 5: Pre-EAS Cutover Administration for ASAI
- Step 6: EAS Cutover

For more information on various approaches that may be used to cut over to EAS, refer to the CMS Administration documents.

Before the transition to EAS takes place, decisions must be made concerning:

- Which area of the current dial plan is to be used for EAS agent loginIDs. EAS agent loginIDs cannot conflict with already defined extension numbers (for example, an EAS agent loginID cannot be the same as a station extension number).
- Whether the current incoming call routing through VDNs and vectors will remain the same after the EAS upgrade, or whether new VDNs and/or vectors are required.
- How incoming call traffic is to be handled during EAS cutover.

Once these decisions are made, the pre-EAS cutover administration activities can be started in preparation for the conversion of the call center to EAS.

NOTE:
Even though EAS administration changes are being made, non-EAS ACD call handling and agent operations are unaffected. When the cutover to EAS is completed, all non-EAS ACD call handling and agent operations will cease.

Step 1: Pre-EAS Cutover Administration for the G3V2 (or later) Switch

Perform the following activities to prepare the G3V2 (or later) switch for the cutover to EAS:

1. At administration terminal display the System-Parameters Customer-Options form and verify that the G3 version is V2 or later and that ACD, Expert Agent Selection?, and Vectoring (Basic) fields are set to y.
2. If you haven't already done so, display the Feature Access Code form and administer the ACD Agent Feature Access Codes (for example, ''Login,'' ''Logout,'' and ''Auto-In'') as required for call center agent operations.
3. Using the CDR System Parameters form, administer whether the EAS loginID, or the terminal extension where the EAS agent is logged in, should appear on CDR reports by setting the Agent Login ID - Record? field to y or n, respectively. This field affects the CDR tracking for incoming calls only; outgoing calls made by a logged-in EAS agent are always recorded by CDR using the agent's loginID.
4. If new VDNs are desired for the EAS environment, using the VDN administration form, administer the VDN Skills and other VDN information for the VDNs used to route calls to EAS agents. If the ''1st,'' ''2nd,'' and/or ''3rd'' skill options are to be used in the vectors or for CMS tracking associated with these VDNs, then administer the 1st Skill, 2nd Skill, and 3rd Skill fields as required.
5. If new vectors are desired for the EAS environment, using the Vector administration form, administer the vectors associated with the VDNs added in the previous step. As part of the EAS feature, the ''1st,'' ''2nd,'' or ''3rd'' skill options may be used in the vector step fields where a skill hunt group is entered (rather than entering an absolute skill hunt group number). Refer to Chapter 10, "Expert Agent Selection" for more information concerning vector programming for the EAS feature.
6. If new skill hunt groups are required, using the Hunt Group administration form, administer the desired skill hunt groups.

NOTE:
Entering a y in the Skills? field automatically causes the ACD? and Vector fields to be set to y. With EAS optioned, it is not possible to administer members for a skill hunt group.

7. If coverage paths are to be administered for EAS agents, using the Coverage Path administration form, set up the coverage paths to be assigned to EAS agent loginIDs.

   NOTE:
   There is a difference between coverage treatment for an EAS ‘‘Direct Agent’’ call (where both the calling party and called loginID have the Direct Agent Calling COR option set to y), and an EAS ‘‘personal’’ call (where either the calling party or called loginID does not have the Direct Agent Calling COR option set to y).

   NOTE:
   A Direct Agent call is routed to an EAS agent as an ACD-type call, and therefore its coverage behavior is considerably different from the coverage for a normal station call. For example, if an EAS agent is not available for an ACD call when a Direct Agent call is made to that agent, the Direct Agent call will be queued in the agent’s first skill hunt group (after initiating a ring-ping and then fluttering the active work-mode button at the agent’s terminal). On the other hand, a personal call to an EAS agent is not an ACD-type call, and its coverage behavior is similar to the coverage treatment for a call to a station extension. For example, a personal call to an EAS agent who is busy on any call appearance will result in the call being sent to an idle call appearance at that agent’s terminal rather than being queued in the agent’s first skill hunt group.

   Depending on the type of coverage criteria desired for Direct Agent and personal calls to EAS loginIDs, administer the desired coverage path criteria as follows:

   - To provide coverage for a non-ACD ‘‘personal’’ call to an EAS loginID when the agent is logged in and active on any call appearance, set the Active? coverage criteria to y. The Active? coverage criteria does not apply for a Direct Agent call to an EAS loginID.
   - To provide coverage for calls to an EAS loginID when the agent is logged out, set the Busy? coverage criteria to y. Busy coverage will also be applied to a logged-in EAS agent when:
     - — A Direct Agent call is made to the EAS agent and there are no available queue slots in the agent’s first skill hunt group;

       or

  - — A personal call is made to an EAS agent and the agent's station has no idle call appearances.

- To provide coverage for calls to an EAS loginID when the agent is logged in but does not answer after a certain number of ring cycles, set the Don't Answer? coverage criteria to y, and enter a number for the desired ring time-out in the Number of Rings field.
- To provide immediate coverage for calls to an EAS loginID whether the agent is logged in or logged out, set the All? coverage criteria to y.
- To provide coverage for calls to EAS loginIDs when the call is to a logged-in agent who has activated the Send All Calls or Go To Cover features, set the DND/SAC/Goto Cover? coverage criteria to y.

8. Up to three coverage paths for different types of call coverage criteria may be linked together by administering the Next Path Number field on the Coverage Path form. If the criteria for the first coverage path are not met, then the criteria for the second linked coverage path are checked by the system, and so on. This can be used to provide different coverage paths for calls to an EAS loginID when the associated agent is logged in or logged out.

NOTE:
If a call to a logged-in EAS loginID is a ''personal'' call and coverage goes into effect, the redirected call maintains a ''simulated bridged appearance'' at that agent's terminal. The agent may still answer the call after redirection takes place by going off-hook on this line appearance. However, if a call to a logged-in EAS loginID is a Direct Agent call, the redirected call does not maintain a simulated bridged appearance at the agent's terminal. The agent may not then answer the call after redirection takes place.

NOTE:
If the Redirection on No Answer (RONA) feature is enabled for skill hunt groups, set the ring time-out interval for the RONA feature such that it does not conflict with the coverage ring time-out criteria.

9. If coverage paths are administered for EAS loginIDs, using the Feature-Related System Parameters form, set the Coverage - Subsequent Redirection No Answer Interval field to the desired ring time-out interval for calls routed to administered coverage points.

NOTE:
EAS loginIDs may be administered as coverage points for a coverage path, and this administered coverage no-answer interval applies to Direct Agent or ''personal'' calls made to these coverage points as well.

10. Using the COR administration form, set the Direct Agent Calling field to y for any COR to be assigned to a trunk or station user who may initiate a Direct Agent call to an EAS agent, or to be assigned to an EAS loginID that may receive Direct Agent calls.

11. If EAS agent loginID passwords are to be administered, using the Feature-Related System Parameters form, set the Minimum Agent-LoginID Password Length field to the desired number of minimum password digits (0 to 9) which must be specified when agent passwords are administered via the Agent LoginID form. The total number of digits which may be assigned to a password is between the value of the Minimum Agent-LoginID Password Length field and 9 digits. If a password is administered for an agent, this password must be entered in addition to the agent's loginID to log in.

12. Using the Agent LoginID form, add the desired EAS loginIDs to be associated with human agents, AUDIX ports, and/or AAS (Auto-Available Split) VRU ports. For human agents, the following fields are administered:

    - Name
    - COR
    - Coverage Path (optional)
    - Security Code (optional for Demand Print feature)
    - LWC Reception (optional)
    - AUDIX Name for Messaging (for MIPS only, if the LWC Reception field is set to audix, or if administered coverage path for the agent has an AUDIX coverage point)
    - Messaging Server Name for Messaging (for MIPS only, if the LWC Reception field is set to msa-spe, or if administered coverage path for the agent has a Messaging Server coverage point)
    - Password (optional)
    - Skills - Primary/Secondary (for at least one skill)

13. For AUDIX and AAS VRU port extensions, when these ports are associated with ACD-type hunt groups, these extensions must be associated with skill hunt groups as part of the cutover to EAS. Additionally, for skill hunt groups used for AAS ports, the AAS? field must be set to y for these hunt groups before any EAS AAS agents can be administered.

    If AUDIX port extensions (such as for the Embedded AUDIX product) are not associated with an ACD hunt group, no administration is required for these ports as part of the cutover to EAS. For the AUDIX and/or AAS ports that are associated with ACD hunt groups, add EAS agent loginIDs for these ports, where only the following fields need to be administered:

    - Name
    - COR

- Coverage path (optional)
- AUDIX? (set to y for AUDIX ports)
- AAS? (set to y for AAS VRU ports)
- Port Extension (set to the AUDIX or AAS port extension administered in the non-EAS environment)
- Skills - Primary/Secondary (where a single skill is entered for the skill hunt group associated with the AUDIX or AAS station ports)

14. Using the Station Administration form, administer any stations to be used by EAS agents and the desired work-mode buttons for each station (if not already administered).

NOTE:
If stations are already administered with work-mode buttons associated with splits, it is NOT necessary to readminister these buttons for EAS. If new work-mode buttons are added to a station, it is not possible to enter data in the Grp field after EAS is enabled except for the AUX work-mode button (which may be administered with a hunt group number if the entered hunt group is a non-ACD hunt group).

NOTE:
Also, if more than one set of work-mode buttons is administered on a station set, these buttons may be left as is until after the cutover to EAS. After the cutover, it is desirable to remove the extra sets of work-mode buttons since EAS requires only one set of work-mode buttons for agent operations.

Step 2: Pre-EAS Cutover Administration for CMS

If R3V2 or later release CMS is to be used at the EAS customer site, refer to the appropriate CMS Administration document for the procedures used to configure CMS for the EAS feature. This document is also helpful in providing overall planning strategies for implementing call center operations.

Step 3: Pre-EAS Cutover Administration for AUDIX

If EAS agents' loginIDs are administered with coverage paths that route to an AUDIX coverage point, the loginIDs for these agents must be administered via the AUDIX console so that the caller will hear the appropriate AUDIX voice responses for calls made to EAS loginIDs.

NOTE:
On the MIPS, the AUDIX Name for Messaging field on the Agent LoginID form must be set to the correct AUDIX name to provide proper AUDIX coverage of calls made to EAS agents, or to leave LWC messages for EAS agents if LWC reception to AUDIX is set up for the agents' loginIDs.

Refer to "Step 1: Pre-EAS Cutover Administration for the G3V2 (or later) Switch" for information on how to administer EAS loginIDs for AUDIX port extensions on the switch.

Step 4: Pre-EAS Cutover Administration for Messaging Server

If EAS agents are administered with coverage paths that route to a Messaging Server coverage point, the Messaging Server adjunct must be administered with extension information that correlates to these EAS loginIDs.

NOTE:
The Messaging Server Name for Messaging field on the Agent LoginID form must be administered to provide proper Messaging Server coverage of calls made to EAS agents, or to leave LWC messages for EAS agents if LWC reception to AUDIX is set up for the agents' loginIDs.

Step 5: Pre-EAS Cutover Administration for ASAI

For customers who utilize ASAI-based applications for call center operations, the cutover to EAS may necessitate an upgrade of the ASAI-related application software on the adjunct. If the customer uses OCM (Outgoing Call Management), the upgrade to EAS requires that specialized vectors be administered to handle the launching of calls from VDNs (as opposed to the non-EAS environment where OCM calls are launched from splits). For more information on the procedures to convert an ASAI application for EAS, refer to *DEFINITY Communications System CallVisor ASAI Planning*, 555-230-222.

Step 6: EAS Cutover

After all pre-EAS activities have been completed, the EAS feature may be activated. Just prior to the EAS cutover, a tape backup of the current switch translations should be made for possible recovery purposes in case some difficulty is encountered during cutover. In particular, since the transition to EAS results in the removal of all ACD hunt group members, the pre-EAS tape backup could save a considerable amount of time in restoring non-EAS hunt group translations if the cutover to EAS is not completed.

It is recommended that incoming ACD call traffic be blocked to prevent the queuing of new ACD calls to existing splits during the cutover from the non-EAS to EAS environment. Blocking of new incoming calls can be accomplished by:

- Busying out the appropriate trunk groups

  and/or

- Using the Vectoring form and setting the first vector step for actively used incoming call vectors to the ‘‘busy’’ step.

Once this is accomplished, perform these activities:

1. Make sure all EAS agents are logged out of all splits. If CMS or BCMS is operational, the CMS real-time reports for splits or the G3-MA/Manager 1 *mon bcms split* command can be used to identify the terminals where agents may still be logged in.
2. Issue the *busy mis* command at the administration terminal to busy-out the CMS link.
3. Issue the *busy link n* command at the administration terminal to busy-out any AUDIX or Messaging Server switch-to-adjunct links.
4. Issue the *busy station x* command at the administration terminal to busy-out any AAS ports.
5. Using the Hunt Group form, convert any ACD splits to skill hunt groups by setting the Skilled? field to y for these hunt groups.
6. Using the Feature-Related System Parameters form, set the Expert Agent Selection (EAS) Enabled? field to y, and if a R3V2 or later release CMS is installed, set the Adjunct CMS Release field.

   NOTE:
   The EAS feature is not compatible with the R2 and R3V1 CMS releases.

7. Release the link to CMS (if installed) by entering the *release mis* command at the administration terminal.
8. Inform the on-site agents that they can log into their terminals using the EAS login procedure and become available to receive ACD calls using the AUTO-IN or MANUAL-IN work-mode operations.
9. Using the Vectoring form, restore any vector steps temporarily changed to ‘‘busy’’ (to block incoming calls) to their previous vector step format.
10. Using the Trunk Group Administration form, if the routing for incoming trunks is to be changed to EAS-related VDNs, administer the Incoming Destination field for any trunk groups to the appropriate VDN extension number.
11. Issue the *release station x* command at the administration terminal to release any AAS ports (where the EAS loginID associated with each AAS port will be automatically logged in).

12. Issue the *release link n* command at the administration terminal to release any adjunct AUDIX links (where the adjunct will cause the associated ports to be logged in).

At this point, the cutover to EAS is complete. It is recommended that a backup of the switch translations be performed as soon as possible after the cutover to preserve the EAS-related administration changes. Also, if agent stations are administered with multiple sets of work-mode buttons, it is recommended that all but one set of work-mode buttons be removed from these stations. Also, multiple queue lights are required for EAS.

V4 Feature Availability

L

The following lists describe vectoring enhancements that are available with G3V4 and later releases. For a detailed description of any item see the referenced section of this guide.

When V4 has been installed on your switch as a bugfix (that is, a maintenance load), but you have not purchased V4 software capabilities you receive:

- Enhanced vector editing capabilities. See "Enhanced Vector Editing (G3V4 and later releases)" on page 2-5.
- The ability to *route-to number* with coverage. See "Route-to Command" on page A-69.
- The ability to include “*” and “#” in the *route-to number* field. "Route-to Command" on page A-69.
- The ability to use “active” instead of “none” keyword for the messaging split/skill command. See "Messaging Command" on page A-61.
- Addition of the i-silent keyword to the *wait-time* command for use with the adjunct routing ADR/Lookahead Interflow applications. See "Wait-time Command" on page A-79.
- Vector initiated Service Observing. See "Service Observing" on page 4-16.

When you have purchased V4 but have not purchased additional V4 enhancements, in addition to the bugfix enhancements you receive:

- The ability to pass ANI to CMS for inclusion in the CMS call record
- The ability to use “latest” keyword for the messaging split/skill command. See "Messaging Command" on page A-61.

Vectoring (G3V4 Enhanced) provides the following additional capabilities:

- The ability to specify a priority level with the oldest-call-wait conditional on the *check-backup* and *goto* commands. See these commands in Appendix A.
- The use of enhanced comparators (<>, >=, and <=) with the *goto* and *route-to* commands as well as use of “none” as an entry for digits checking, and “active” or “latest” VDN thresholds for indirect VDN references. See these commands in Appendix A.
- The use of wildcards in digit strings for matching on collected digits and ANI or II-digits. See Appendix A.
- The use of Vector Routing Tables for matching on collected digits and ANI or II-digits. See "Vector Routing Tables" on page 5-7 or "ANI and II-Digits Routing" on page 7-1.
- Multiple Audio/Music Sources for use with the *wait-time* command. "Multiple Audio or Music Sources" on page 4-6.

Vectoring (G3V4 Advanced Routing) provides the following additional capabilities (Vectoring (G3V4 Enhanced) must also be enabled.):

- Rolling Average Speed of Answer (ASA) Routing. See "Rolling Average Speed of Answer (ASA)" on page 6-10.
- Expected Wait Time (EWT) Routing. See "Expected Wait Time (EWT)" on page 6-2.
- VDN Calls Routing. See "VDN Calls" on page 6-12.

Vectoring (ANI/II-Digits Routing) provides the following additional capabilities (Vectoring (G3V4 Enhanced) must also be enabled.):

- ANI Routing. See "ANI Routing" on page 7-1.
- II-Digits Routing. See "II-Digits Routing" on page 7-5.

Improving Performance

M

Introduction

This appendix provides recommendations on how to write vectors that promote favorable performance practices. Two basic principles to follow are:

1. Minimize the amount of call processing
   - — Minimize the number of vector steps to process a call
   - — Use the lower cost steps when possible (refer to Table M-4 and Table M-5)
2. Avoid vector steps which have a substantial probability of failure
   - — Calls made outside of business hours
   - — Queues to groups with less than desirable resources or characteristics

The most wasteful use of processing resources is frequently caused by inefficient looping. For example, performance could be compromised when a vector loops through steps too often. This is especially true with long queue times.

Some examples with looping are discussed and recommendations are given on how to maximize performance. They are:

- Audible Feedback
- Lookahead Interflow
- Check Backup

Examples other than looping are also discussed. They are:

- After Business Hours

- Lookahead Interflow

All looping examples in this appendix use only loops within a single vector. It is important to also be aware of looping to other vectors through the use of vector chaining. The same principles can be extrapolated from the looping examples. Creating a flow diagram is often helpful for identifying looping errors.

In addition to the example vectors, tables rating the relative performance costs of specific vector commands are also included.

NOTE:
Remember to test vectors for performance in addition to call flow.

Looping Examples

Audible Feedback

Recommendation: Evaluate the length of the wait period between repetitions of an announcement and increase the length, if possible. For optimum performance, add a second announcement after the initial announcement and repeat the second announcement less often.

The example in Figure M-1 repeats the “All representative are busy. Please hold.” announcement every 10 seconds as long as the call is in queue.

```
1. queue-to main split 1
2. announcement 2770     (“All representatives are busy. Please hold.”)
3. wait-time 10 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

Figure M-1. Example Vector

The example in Figure M-2 repeats the announcement only every 60 seconds, thus improving performance.

```
1. queue-to main split 1
2. announcement 2770     (“All representatives are busy. Please hold.”)
3. wait-time 60 seconds hearing music
4. goto step 2 if unconditionally
5. stop
```

Figure M-2. Example Vector with Improved Performance

The example in Figure M-3 adds a second announcement, “All representatives are still busy. Please hold.” in addition to the initial announcement and repeats the second announcement less often (every 120 seconds), thus improving performance again.

```
1. queue-to main split 1
2. announcement 2770     (“All representatives are busy. Please hold.”)
3. wait-time 120 seconds hearing music
4. announcement 2771     (“All representatives are still busy. Please
                         continue to hold.”)
5. goto step 3 if unconditionally
6. stop
```

Figure M-3. Another Example Vector with Improved Performance

Table M-2 compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the first announcement is 3 seconds long and the second announcement is 4 seconds long.

Table M-1. Approximate Number of Vector Steps Executed for the Audible Feedback Examples

	Example in Figure M-1	Example in Figure M-2	Example in Figure M-3
when a agent is available in split 1	1	1	1
queueing time of 5 minutes	70	15	9

When a call is queued for 5 minutes, the amount of vector steps drops dramatically when the amount of time between announcements is increased (Figure M-2), and drops even more when a second announcement is added, and the amount of time between announcements is increased again (Figure M-3). When an agent in split 1 is immediately available to answer the call, there is no difference in the amount of vector steps for the three examples.

Lookahead Interflow

Recommendation: Add a wait period between successive lookahead interflow attempts and make the waiting period as long as feasible.

The example in Figure M-4 continuously attempts a lookahead interflow as long as the call is in queue or until a lookahead attempt succeeds.

```
1. queue-to main split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. goto step 4 if unconditionally
```

Figure M-4. Example Vector

The example in Figure M-5 adds a delay so that the lookahead interflow attempt occurs only every 10 seconds.

```
1. queue-to main split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 10 seconds hearing music
6. goto step 4 if unconditionally
```

Figure M-5. Example Vector with Improved Performance

The example in Figure M-6 increases performance even more by increasing the delay between lookahead interflow attempts to 30 seconds.

```
1. queue-to main split 1 pri l
2. announcement 3000
3. wait-time 20 seconds hearing music
4. route-to number 93035555555 cov n if unconditionally
5. wait-time 30 seconds hearing music
6. goto step 4 if unconditionally
```

Figure M-6. Another Example Vector with Improved Performance

Table M-2 compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the announcement is 5 seconds long.

Table M-2. Approximate Number of Vector Steps Executed for Lookahead Interflow Examples

	Example in Figure M-4	**Example in Figure M-5**	**Example in Figure M-6**
when a agent is available in split 1	1	1	1
queueing time of 5 minutes	up to 1,000	85	30

When a call is queued for 5 minutes, the amount of vector steps drops dramatically when a delay is added (Figure M-5), and drops even more when the length of the delay is increased (Figure M-6). When an agent in split 1 is immediately available to answer the call, there is no difference in the amount of vector steps for the three examples.

Check Backup

Recommendation: When using check-backup commands to queue a call to backup splits, ensure that an adequate amount of time has elasped before checking the backup splits again.

NOTE:
With the introduction of the ‘Expected Time Wait Time’ feature in Generic 3 Version 4, the style of programming used in this example is no longer relevant. The best approach is to use the ‘Expected Time Wait’ feature to locate the most appropriate split for the call and queue it there.

The example in Figure M-7 checks backup splits continuously as long as the call is in queue.

```
1. queue-to main split 1 pri h
2. announcement 3000
3. wait-time 10 seconds hearing music
4. check-backup split 21 pri m if available-agents > 0
5. check-backup split 22 pri m if available-agents > 0
6. check-backup split 23 pri m if available-agents > 0
7. check-backup split 24 pri m if available-agents > 0
8. check-backup split 25 pri m if available-agents > 0
9. goto step 4 if unconditionally
```

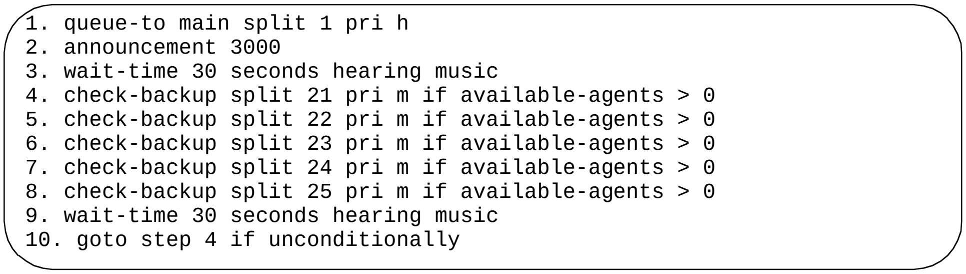
Figure M-7. Example Vector

The example in Figure M-8 adds a delay of 10 seconds to ensure that some time has elasped before checking the backup splits again.

```
1. queue-to main split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check-backup split 21 pri m if available-agents > 0
5. check-backup split 22 pri m if available-agents > 0
6. check-backup split 23 pri m if available-agents > 0
7. check-backup split 24 pri m if available-agents > 0
8. check-backup split 25 pri m if available-agents > 0
9. wait-time 10 seconds hearing music
10. goto step 4 if unconditionally
```

Figure M-8. Example Vector with Improved Performance

Since the agent availability status may not be likely to change every 10 seconds, it may make sense to increase the wait time to 30 seconds, as shown in the example in Figure M-9.



```
1. queue-to main split 1 pri h
2. announcement 3000
3. wait-time 30 seconds hearing music
4. check-backup split 21 pri m if available-agents > 0
5. check-backup split 22 pri m if available-agents > 0
6. check-backup split 23 pri m if available-agents > 0
7. check-backup split 24 pri m if available-agents > 0
8. check-backup split 25 pri m if available-agents > 0
9. wait-time 30 seconds hearing music
10. goto step 4 if unconditionally
```

Figure M-9. Another Example Vector with Improved Performance

Table M-2 compares the relative processing cost of the three examples by looking at the approximate number of vector steps executed while processing the call. Assumption is that the announcement is 5 seconds long.

Table M-3. Approximate Number of Vector Steps Executed for Check Backup Examples

	Example in Figure M-7	Example in Figure M-8	Example in Figure M-9
when a agent is available in split 1	1	1	1
queueing time of 5 minutes	up to 1,000	190	65

When a call is queued for 5 minutes, the amount of vector steps drops dramatically when a delay is added before checking the backup splits again (Figure M-8), and drops even more when the length of the delay is increased again (Figure M-9). When an agent in split 1 is immediately available to answer the call, there is no difference in the amount of vector steps for the three examples.

Other Examples

After Business Hours

Recommendation: Test to see if the destination resources are available (such as during business hours) before queuing.

The example in Figure M-10 queues calls to a hunt group regardless of the time of the call. When the call is made after business hours, the announcement is repeated until the caller hangs up.

```
1. queue-to main split 1
2. announcement 5000
                        (“All agents are busy. Please hold.”)
3. wait-time 120 seconds hearing music
4. announcement 5001
                        (“All agents are still busy. Please continue to
                        hold.”)
5. goto step 3 if unconditionally
```

Figure M-10. Example Vector

The example in Figure M-11 tests for business hours before queuing the call. If the call is made after business hours, an announcement informs the caller of the business hours and the call is terminated.

```
1. goto step 7 if time-of-day is all 17:00 to all 8:00
2. queue-to main split 1
3. announcement 5000
                              (“All agents are busy. Please hold.”)
4. wait-time 120 seconds hearing music
5. announcement 5001
                              (“All agents are still busy. Please
                              continue to hold.”)
6. goto step 4 if unconditionally
7. disconnect after announcement 5001
                              (“Business hours are 8:00 AM to 5:00 PM,
                              Please call back then.”)
```

Figure M-11. Example Vector with Improved Performance

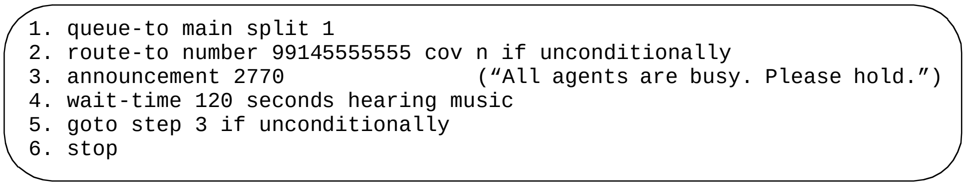
In the first example, unnecessary processing occurs when a call is queued after business hours and the call is terminated only when the caller hangs up. As shown in the second example, it is more economical to test for business hours before queuing a call.

Lookahead Interflows

Recommendation: When using a lookahead interflow, first test to see if the receiving office is open for business.

The scenario is a sending switch in Los Angeles, with office hours from 8:00 AM to 5:00PM (8:00-17:00) PST and the receiving switch is in New York, with office hours from 8:00 AM to 5:00PM EST (5:00-14:00 PST). There is a 3 hour difference between the two switches.

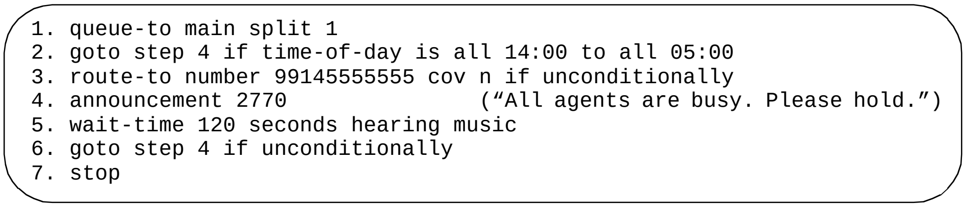
The example in Figure M-12 routes calls to the New York switch. If there are no agents available at the Los Angeles switch. It is possible for calls to be interflowed during hours that the agents in New York are not available, thus doing unnecessary processing.



```
1. queue-to main split 1
2. route-to number 99145555555 cov n if unconditionally
3. announcement 2770               (“All agents are busy. Please hold.”)
4. wait-time 120 seconds hearing music
5. goto step 3 if unconditionally
6. stop
```

Figure M-12. Example Vector

The example in Figure M-13 tests first to see if the New York switch is open before requesting a queue to the New York switch, thus avoiding unnecessary processing.



```
1. queue-to main split 1
2. goto step 4 if time-of-day is all 14:00 to all 05:00
3. route-to number 99145555555 cov n if unconditionally
4. announcement 2770               (“All agents are busy. Please hold.”)
5. wait-time 120 seconds hearing music
6. goto step 4 if unconditionally
7. stop
```

Figure M-13. Example Vector with Improved Performance

The example in Figure M-14 can be used if you have Generic 3 Version 4 Advanced Routing optioned. In this case, the 'Expected Wait Time' feature may be used to determine whether it is worthwhile placing a lookahead interflow call attempt.

```
1. queue-to main split 1
2. goto step 5 if expected-wait for call < 30
3. goto step 5 if time-of-day is all 14:00 to all 05:00
4. route-to number 99145555555 cov n if unconditionally
5. announcement 2770              (“All agents are busy. Please hold.”)
6. wait-time 120 seconds hearing music
7. goto step 5 if unconditionally
8. stop
```

Figure M-14. Another Example Vector with Improved Performance

There is little reason to attempt an interflow if the call will be answered quickly at the main switch. For the examples in Figure M-13 and Figure M-14, vector steps are avoided which do not aid in the call being answered sooner.

Relative Processing Cost of Vector Commands

Some vector commands use more processing resources than others. Table M-4 and Table M-5 show the relative processing costs of specific vector commands for Generic 3 Version 4i/v/vs and Generic 3 Version 4r respectively. Whenever possible, use the lower cost vector commands. This will minimize your performance costs and upgrade your performance.

Table M-4. Relative Processing Cost of Vector Commands for Generic 3 Version 4i/v/vs

relative performance cost	vector command
high	adjunct routing
high	check-backup
high	collect <2-6> digits
high	queue-to
high	route-to
high	collect <1> digits
medium	converse
medium	announcement
medium	goto step
medium	goto vector
medium	messaging
low	busy
low	disconnect
low	stop
low	wait-time

Table M-5. Relative Processing Cost of Vector Commands for Generic 3 Version 4r

relative performance cost	vector command
medium	adjunct routing
medium	check-backup
medium	collect digits
medium	converse
medium	goto vector (table comparison)
medium	messaging
medium	queue-to
medium	route-to
low	announcement
low	busy
low	disconnect
low	goto step
low	goto vector
low	stop
low	wait-time

Call Vectoring System Parameters

N

G3iV1.1 and G3V2 System Parameters

Table N-1. Maximum System Parameters for Call Vectoring/EAS for G3V2

ITEM	G3iV1.1-286	G3vsV2 ABP/PBP	G3sV2 ABP/PBP	G3iV2-386	G3rV2
Call Vectoring					
Multiple Splits per Call	3	NA/3	NA/3	3	3
Priority Levels	4	NA/4	NA/4	4	4
Recorded Announcement	128	NA/128	NA/128	128	256
Steps per Vector	32	NA/32	NA/32	32	32
Vector Directory Numbers[1]	500	NA/100	NA/100	512	20,000
Measured VDNs	500	NA/100	NA/100	512	2,000
Vectors per System	256	NA/48	NA/48	256	512
Expert Agent Selection (EAS)					
Skill Groups	NA	NA/24	NA/24	99	255
VDN Skill Preferences	NA	NA/3	NA/3	3	3
Multiple Skills per Call	NA	NA/3	NA/3	3	3
Multiple Skills per Agent	4	NA/4	NA/4	4	4
Agent Login IDs	NA	NA/450	NA/450	1,500	10,000

[1] The total of VDN, Station, and Login ID extensions cannot exceed 25,000.
2,000 for G3rV3
7,084 for G3rV3 for "Simultaneous 3-way Conf. Calls"

G3V3 System Parameters

Table N-2. Maximum System Parameters for Call Vectoring/EAS for G3V3

ITEM	G3vsV3 ABP/PBP	G3sV3 ABP/PBP	G3iV3	G3rV3
Call Vectoring				
Max. Skills a Call Can Simultaneously Queue to	NA/3	NA/3	3	3
Priority Levels	NA/4	NA/4	4	4
Recorded Announcement	NA/128	NA/128	128	256
Steps per Vector	NA/32	NA/32	32	32
Vector Directory Numbers[1]	NA/100	NA/100	512	20,000
CMS Measured VDNs[2]	NA/100	NA/100	512	2,000
Vectors per System	NA/48	NA/48	256	512
No. of Collected Digits for Call Prompting	NA/16	NA/16	16	16
No. of Dial-Ahead Digits for Call Prompting	NA/24	NA/24	24	24
Vector Routing Tables	NA	NA	NA	NA
Expert Agent Selection (EAS)				
Skill Groups	NA/24	NA/24	99	255
VDN Skill Preferences	NA/3	NA/3	3	3
Max. Skills a Call Can Simultaneously Queue to	NA/3	NA/3	3	3
Agent Login IDs	NA/450	NA/450	1,500	10,000
Max. Skills per Agent	NA/4	NA/4	4	4
Max. Agents that can be Logged-In				
When Each Has 4 Skills Assigned	NA/37	NA/37	125	1,300
When Each Has 1 Skill Assigned	NA/150	NA/150	500	5,200

[1] The total of VDN, Station, and Login ID extensions cannot exceed 25,000.
2,000 for G3rV3
7,084 for G3rV3 for "Simultaneous 3-way Conf. Calls"

[2] Measured limits depend on the CMS release used.

G3V4 System Parameters

Table N-3. Maximum System Parameters for Call Vectoring/EAS for G3V4

ITEM	G3vsV4 ABP/PBP	G3sV4 ABP/PBP	G3iV4	G3rV4
Call Vectoring				
Max. Skills a Call Can Simultaneously Queue to	NA/3	NA/3	3	3
Priority Levels	NA/4	NA/4	4	4
Recorded Announcement	NA/128	NA/128	128	256
Steps per Vector	NA/32	NA/32	32	32
Vector Directory Numbers[1]	NA/100	NA/100	512	20,000
CMS Measured VDNs[2]	NA/100	NA/100	512	2,000
Vectors per System	NA/48	NA/48	256	512
No. of Collected Digits for Call Prompting	NA/16	NA/16	16	16
No. of Dial-Ahead Digits for Call Prompting	NA/24	NA/24	24	24
Vector Routing Tables	NA/5	NA/5	10	100
Expert Agent Selection (EAS)				
Skill Groups	NA/24	NA/24	99	255
VDN Skill Preferences	NA/3	NA/3	3	3
Max. Skills a Call Can Simultaneously Queue to	NA/3	NA/3	3	3
Agent Login IDs	NA/450	NA/450	1,500	10,000
Max. Skills per Agent	NA/4	NA/4	4	4
Max. Agents that can be Logged-In				
When Each Has 4 Skills Assigned	NA/37	NA/37	125	1,300
When Each Has 1 Skill Assigned	NA/150	NA/150	500	5,200

[1] The total of VDN, Station, and Login ID extensions cannot exceed 25,000.
2,000 for G3rV3
7,084 for G3rV3 for "Simultaneous 3-way Conf. Calls"

[2] Measured limits depend on the CMS release used.

Glossary

A

ACD
See **Automatic Call Distribution**.

ACD split
A MIA/DDC hunt group in a system where ACD has been optioned by the customer and where the hunt group has been administered as used for ACD.

active call
A call to a VDN that is still active in the switch or tandemed through the switch. The call remains active until the call is released.

active VDN
The active VDN is the called VDN as determined by VDN Override. The active VDN would be displayed to an answering agent if delivered at this time. If the previous VDN has override set to y, the routed to VDN becomes the active VDN. The active VDN is also called the VDN of record.

adjunct processor
A processor that performs one or more tasks for another processor (for example, a switch).

After-Call work mode
An ACD agent work mode that makes the agent unavailable to receive any ACD calls for any split. This mode is used when the agent is doing ACD call-related work.

agent
A member of a hunt group or split.

Agent LoginID
The digits dialed in the agent login procedure to identify the agent. With the logical agent capability, these digits must be part of the extension numbering plan and they are administered using the ‘Agent LoginID’ form.

agent skill
An attribute that is assigned to an ACD Agent LoginID. Agents can be assigned up to four skills. The skills indicate the type of calls an agent can handle.

AMWL
See **Automatic Message Waiting Lamp (AMWL)**.

ANI
See **Automatic Number Identification (ANI)**.

announcement
A pre-recorded message that may give a caller information about the status of the call or other important items.

answer supervision
A signal set by a terminating communication system to an originating communications system — or immediate charging point, such as a central office (CO) switch — indicating that an incoming call has been answered. On receiving this signal, the originating system or other charging point begins tracking charges for the call, if charges apply.

In terms of network services, answer supervision is a feature offered by the network provider on certain types of trunks. When the network returns answer supervision for a particular trunk group, the ‘Answer Supervision’ field on the ‘Trunk Group’ form should be set to a *yes*. The ‘Answer Supervision Timeout’ field on the ‘Trunk Group’ form should be set to a large value in this case.

ASA
See **Average Speed of Answer (ASA)**.

ASAI route request message
The information provided by the switch for the adjunct processor that the latter can use to first access a database and then determine a route for the call. The message contains the calling number information, the called number from DNIS, the called extension forwarded to the VDN, the routing VDN, the call identifier, look-ahead interflow information (if any), and the digits collected via Call Prompting (if any).

Audio Information Exchange (AUDIX)
A voice mail service that allows users to record, edit, store, forward, and retrieve voice messages to and/or from other users.

AUDIX
See **Audio Information Exchange (AUDIX)**.

Auto-In work mode
An ACD agent work mode that makes the agent available to receive calls and allows the agent to receive a new ACD call immediately after disconnecting from the previous call.

automated attendant
Call prompting application that allows the caller to route to the extension of the party that the caller would like to reach. The caller may also be prompted to enter digits that are not an actual extension, but allow the call to be routed to the appropriate destination.

Automatic Call Distribution
A PBX feature that distributes incoming calls to hunt groups (called splits or skills) of available agents. ACD hunt groups may be assigned with a queue for when all agents are busy as a call arrives, the call will be queued and wait for an available position. Once an agent becomes available, the call will be routed to the position.

Automatic Number Identification (ANI)
A general industry term referring to knowledge of the calling party number (CPN). When the calling party is behind a PBX, the number provided can be either a billing number (BN) for the PBX or the Calling Party Number (CPN). See also **Billing Number (BN)**, **Calling Party Number (CPN)**, and **Integrated Services Digital Network (ISDN)**.

Automatic Message Waiting Lamp
A device that, when lit, notifies an agent that a caller has left an AUDIX message.

AUX work mode
See **Auxiliary Work (AUX) mode**.

Auxiliary work (AUX) mode
An ACD agent work mode that indicates the agent is unavailable to receive any ACD calls for the specified split/skill. This work mode is used for non-ACD related activities, such as breaks.

available agent
An agent that is available to receive a call through an ACD split/skill (ACD call). The agent is available if logged in and in either the auto-in or manual-in work mode, and is not on a call.

Average Speed of Answer (ASA)
The average amount of time it takes before calls have been answered.

The ASA for a split/skill includes the time spent in queue and the time ringing an agent.

The ASA for a VDN includes the time spent in vector processing for the VDN that the call was answered in.

Interval ASA is used for BCMS and CMS reporting where the ASA is calculated on reporting interval boundaries and the ASA is cleared to zero at the start of each reporting interval.

Rolling ASA is used for vector routing which is a running weighted average calculation without regard to any interval boundaries.

B

backup split
In a multiple-split environment, the secondary or tertiary split to which the call is queued.

Basic Call Management System (BCMS)
An application running on the customer switch that collects information from an ACD unit. This application enables customers to monitor and manage telemarketing centers by generating reports on the status of agents, splits, etc., and enables customers to administer the ACD feature for a communications system.

Basic Call Vectoring
A call vectoring feature that allows the user to program the type of processing that an incoming telephone call receives. The user programs accordingly by arranging a set of vector commands in the proper sequence. This feature provides call queuing and call management capabilities.

BCMS
See **Basic Call Management System (BCMS)**.

Billing Number (BN)
The 10-digit number (for North America) that is billed when the calling party makes a toll call. The BN is not always identical to the Calling Party Number (CPN). For example, a company may have a BN of 555-7000, yet an individual at the company may have a CPN of 555-7335. Taken together the CPN/BN information allows the called party to identify the calling party and to determine call volumes from particular geographic areas. See also **Calling Party Number (CPN)**.

BN
See **billing number (BN)**.

branching
Sending of vector control from one vector step to either another vector step or another vector. Branching can be done unconditionally or conditionally (that is, according to a stated condition).

call coverage
A function that provides the automatic redirection of certain calls to alternate answering positions in a call coverage path.

call coverage path
A list of one, two, or three alternate answering positions (covering users) that are accessed in sequence when the called individual or group (principal) is not available to answer the call.

call identifier
An ASAI identifier that permits the ASAI adjunct to first perform simultaneous routing for multiple calls and then to track the progress of the call. This information is included in the ASAI route request message that is sent by the switch.

Call Prompting

A feature that uses vector commands to collect and test digits, and/or display digits to the agent or pass them to an adjunct processor. Based on the digits dialed, the call is routed to a desired destination or receives other treatment. External callers must use touch-tone dialing for entering digits. The Call Prompting feature can be used with other call vectoring capabilities.

Caller Information (CALLR-INFO) button

A device that displays the digits that have been collected when a Call Prompting user enters digits from a touch-tone telephone or when a VRU has returned digits to the switch.

Call Management System (CMS)

An application running on an adjunct processor that collects, stores, analyzes, displays, and reports ACD information provided by the switch. CMS enables customers to monitor and manage telemarketing centers by generating reports on the status of agents, splits, trunks, trunk groups, vectors, vector directory numbers. CMS also enables customers to partially administer the ACD features on the DEFINITY switch.

'Call Vector' form

An on-line screen used for entering a vector.

call vectoring

A method that manages inbound calls by using routing tables called vectors to uniquely define treatments for each call type. The call type is based on the dialed number or trunk group termination to a vector via VDNs. The vectors are customer programmable using commands that resemble a high-level programming language to specify what treatments the call should be given. See also **vectoring**.

Calling Party Number (CPN)

The 10-digit number (for North America) of the station that is calling. The CPN is not always identical to the billing number (BN). For example, a company may have a BN of 555-7000, yet an individual at that company may have a CPN of 555-7335. Taken together, CPN/BN information allows the called party to identify the calling party and to determine call volumes from particular geographic areas. Formerly called station identification (SID). See also **Billing Number (BN)**.

CCITT

See **Consultative Committee on International Telegraphy and Telephony (CCITT)**.

Class of Restriction (COR)

A VDN property consisting of a 1- or 2-digit number that is used to determine any calling party restrictions (for example, lack of calling permission).

CMS

See **Call Management System (CMS**).

Consultative Committee on International Telegraphy and Telephony (CCITT)

One of the four permanent parts of the International Telecommunications Union, based in Geneva, Switzerland.

collected digits

The touch tone digits entered by a caller or returned from a VRU and collected by a *collect digits* vector step. See also **dial-ahead digits**.

conditional branching

The type of vector control flow that is passed from the current vector step to the specified vector step, provided that the condition in the first step is met.

CONVERSANT

A Voice Response Unit (VRU) produced by AT&T that is prominently used with the Voice Response Integration (VRI) feature.

Converse Data Return code

A feature access code administered on the DEFINITY switch. Whenever data is to be returned from CONVERSANT upon completion of a script invoked by a *converse* vector step, CONVERSANT must outpulse the converse data return code immediately prior to outpulsing the returned digits.

converse first data delay

An administrable short delay to prevent the DEFINITY system from outpulsing digits before CONVERSANT is ready. (This outpulsing is enabled via the *converse* vector step.) The delay is administered on the 'System Parameters' form, and it starts when the CONVERSANT port answers the call.

converse second data delay

An administrable short delay to prevent the DEFINITY system from outpulsing the second group of digits before CONVERSANT is ready. (This outpulsing is enabled via the *converse* vector step.) The delay is administered on the 'System Parameters' form, and it starts once the first group of digits has been outpulsed.

converse split

A split or hunt group that is accessed by a *converse* vector step.

COR

See **Class of Restriction (COR)**.

CPE

See **Customer Premises Equipment (CPE)**.

Customer Premises Equipment (CPE)

The terminal equipment supplied by either the telephone common carrier or by a competitive supplier, which is connected to the nationwide telephone network and resides on the customer's premises.

data collection

A call prompting application that provides the switch with a method of collecting a calling party digits that can be displayed and used by an adjunct or an agent to assist in processing the call.

Data In/Voice Answer (DIVA)

A call prompting application that allows a caller to hear an announcement based on the digits that the caller enters.

DCS

See **Distributed Communications System (DCS)**.

DDC

See **Direct Department Calling (DCC)**.

dequeue

Call flow or action that is defined according to the management system and the switch version involved. For R3 CMS with G3 Call Vectoring enabled, the term refers to a call that is not answered by any split other than the primary split in a VDN. This call flow is not pegged for BCMS and G3 with Call Vectoring. R2 CMS pegs this call flow in the same situations where R3 CMS pegs a dequeue.

dequeued average queue time

The average time a call waits until it is answered by another split to which the call is also queued. This time is indicated in the CMS Split Summary report.

dial-ahead digits

The touch tone digits that are entered by the caller or returned by a VRU, but not yet collected by a *collect digits* vector step. Dial-ahead digits are stored in the call prompting buffer until collected by a *collect digits* vector step. See also **collected digits**.

Dialed Number Identification Service (DNIS)

DNIS provides a display to the answering agent of the service or project, or of the number called (or associated routing number) so that agents grouped in one split/skill can answer appropriately for many different services. DNIS can also be sent to a host computer or other adjunct device.

direct agent call

A call directed to an adjunct (via ASAI) or by a voice terminal user (via EAS Logical Agent ID) to a specific agent where the call is treated as an ACD call with zip tone answer, queuing, after-call-work, and CMS measurement as an ACD call. These calls are queued and delivered before any normal split/skill ACD calls

Direct Department Calling (DDC)

A hunting algorithm which delivers calls to available agents within a hunt group in a pre-defined order.

Distributed Communications System (DCS)

A network configuration linking two or more switches in such a way that selected features appear to operate as if the network were one system/switch.

DIVA

See **Data In/Voice Answer (DIVA)**.

DNIS

See **Dialed Number Identification Service (DNIS)**.

EAD

See **Expert Agent Distribution (EAD)**.

EAS

See **Expert Agent Selection (EAS)**.

End-of-Dialing Indicator

A call prompting symbol, specifically #, which allows the user to enter fewer touch-tone digits than the maximum specified. The user does this by entering # at the end of the digit string.

Expected Wait Time (EWT)

An estimate of how long a caller will have to wait to be served by a split/skill while in queue considering the current and past traffic, handling time, and staffing conditions. Time spent in vector processing before being queued and the time spent ringing an agent with manual answering operation is not included in the EWT.

Expert Agent Distribution (EAD)

Call distribution method that allows a new call to the hunt group to first look for the most-idle primary agent. The call only looks for the most-idle secondary agent if no primary agents are available. Therefore, this method allows a call to be distributed to an agent who can best handle the call in cases where multiple agents (primary and secondary) are available.

Expert Agent Selection (EAS)

An optional feature that provides a group of capabilities including: assigning skills to VDNs and agents, defining hunt groups as skills, providing call distribution via matching skills, logical agents and direct agent calling from voice terminals.

EWT

See **Expected Wait Time (EWT)**.

far-end switch

A switch within the tandem switch configuration of an LAI call setup that decides whether to accept or deny an LAI call according to a number of conditions.

forced first announcement

A pre-recorded message that is given to a caller before the call is queued to a split.

E

EAD

See **Expert Agent Distribution**.

Expert Agent Distribution

Uses the Most Idle Agent algorithm to route calls and is only available with Expert Agent Selection.

F

G

H

hunt group

An ordered group of stations.

I

II-Digits

See **Information Indicator-Digits (II-Digits)**.

Information Indicator-Digits (II-Digits)

A 2-digit string provided via ISDN PRI with an incoming call to indicate the type of originating line, such as a hotel, prison, or pay phone, of the caller. As an option, subscribers to the CPN/BN network service (ANI Delivery Service option) will receive the II-Digits for the caller when provided by the LEC at no additional charge.

inflow

A call flow or action that is defined according to the management system and the switch version involved.

For R3 CMS/BCMS and G3 with call vectoring enabled, the term refers to a call that is answered by a split other than a primary split.

For R2 CMS, when multiple-split queuing is involved, the term refers to a call that requeues to the first of multiple splits.

Integrated Services Digital Network (ISDN)

A public or private network that provides end-to-end digital connectivity for all services to which users have access by a limited set of standard multi-purpose user-network interfaces defined by the CCITT. Through internationally accepted standard interfaces, ISDN provides digital circuit-switched or packet-switched connectivity within the network and links to other ISDNs to provide national and international digital connectivity.

intelligent interflow

A method of interflowing calls via Look-Ahead Interflow whereby the receiving switch can decide to accept or deny calls on a call-by-call basis.

inter-digit timer

A device that is restarted once the call prompting digit collection is completed. The device is restarted to detect a timeout for releasing the TTR.

interflow

A process that allows calls that are directed or redirected to one split to be redirected to an external destination.

intraflow

A process that allows calls that are unanswered at a split within a predefined time frame to be redirected to other splits on the same switch.

ISDN

See **Integrated Services Digital Network (ISDN)**.

ISDN PRI link

A standard ISDN frame format that specifies the protocol used between two or more communications systems. PRI runs at 1.544 Mbps and, as used in North America, provides 23 64-Kbps B-channels (voice or data) and one 64-Kbps D-channel (signaling). The D-channel is the 24th channel of the interface and contains multiplexed signaling information for the other 23 channels.

J

K

L

LAI
See **Look-Ahead Interflow**.

latest VDN
The VDN to which the caller was most recently routed. The latest VDN is not affected by VDN Override.

LEC
See **Local Exchange Carrier (LEC)**.

Local Exchange Carrier (LEC)
A local phone company.

logical agent - logical agent ID
This capability provides special extensions without hardware called Agent Login IDs to which the characteristics of the agent (including skills, coverage path, COR, etc.) are assigned. With this capability, the agent can login using any physical terminal and that terminal then takes on the characteristics of the agent. When the agent login ID is called, the call is directed to the agent at the terminal the agent had logged into on a direct agent call basis if the CORs are properly set. Logical agent also provides a single set of work-mode buttons for all skills.

Look-Ahead Interflow (LAI)
A feature that enhances call vectoring interflow so that calls will interflow to those remote locations that can accept the calls. The call vectoring capabilities at both the sending and receiving communications systems along with the ISDN-PRI private and public network connectivity between the locations are utilized to route a call only when accepted by the receiving communications system.

M

manual-answer
With manual-answering operation, the agent is on-hook while available for an ACD call and the call is delivered via ringing (alerting) the station set and the agent goes off-hook on the alerted appearance to answer the call.

Manual-In work mode
An ACD agent work mode that avails the agent to receive calls and automatically puts the agent into After Call Work mode after disconnecting from an ACD call.

Message Collection

A call prompting application that gives the caller the option of not waiting (in queue, if vectoring is enabled) to be serviced by an agent, but rather of leaving a message for the agent or the agent's associated split.

Message Server Adjunct

A message service split that is used in conjunction with the *messaging split* command.

MIA

See **Most Idle Agent (MIA)**.

Most Idle Agent (MAI)

An algorithm which delivers the next call to the agent within the hunt group, who has been idle for the longest period of time.

MSA

See **Message Server Adjunct (MSA)**.

MSQ

See **Multiple Split Queuing (MSQ)**.

multiple split (skill) queuing

A process that allows a call to be sent to more than one split (skill) at the same time. This serves to minimize the waiting time before the call is answered. The maximum number of splits (skills) to which a call can be queued is three. Skills replace splits if EAS is optioned.

N

nonconverse split (skill)

A split (skill) or hunt group that is not being accessed by a converse vector step.

O

outflow

A call flow or action that is defined according to the management system and the switch version involved.

For R3 CMS/BCMS and G3 with call vectoring enabled, the term refers to a call that is dequeued from a primary split via a *route-to* or *messaging split* command, or by being answered by an agent in another split to which the call is also queued.

For R2 CMS, when multiple-split queuing is involved, the term refers to a call that first queues to multiple splits and then eventually dequeues from these splits without being answered in any of them.

P

primary split

The first split in a VDN to which a call actually queues. Another split becomes the primary split if the call leaves the VDN (via a *route-to VDN* command, for example), and the call is then queued to another split; or, if the call leaves vector processing (via a *route-to <extension>* command, for example), and the call queues to another split.

priority

The queuing level to which an incoming call is assigned and at which the call is processed. Priorities are top, high, medium, and low.

Q

R

receiving switch

A switch within an LAI call setup that processes vector inflow. The switch makes the decision to accept or deny an LAI call according to a number of conditions.

routing

A call vectoring method that allows an incoming call to be sent to a specific destination. For example, a call can be queued to a split.

S

sending switch

A switch within an LAI call setup that processes vector outflow. The vector(s) in the sending switch use the *goto* command to determine whether the call should be sent to the receiving switch.

Sequential Flow

The type of vector control flow that is automatically passed from the current vector step to the step that follows.

skill

An attribute that is assigned to an ACD Agent LoginID (agent skill) and is administered to calls on a VDN basis (VDN skill).

skill hunt group

A hunt group for EAS which has the same attributes as a vector controlled split. EAS must be enabled.

split/skill

A group of agents organized to receive calls in an efficient and cost effective manner. Refers to both ACD splits and hunt groups without ACD optioned.

Split Summary report
A report that summarizes the call activity for an entire split. It can be generated by CMS as well as in BCMS.

split supervisor
An ACD split manager who monitors split flows.

staffed
For ACD splits, staffed connotes an agent being logged in. Therefore, when an agent is staffed in an ACD split, this means that the agent is present. However, the term does not indicate if the agent is available to receive calls. When an agent logs in initially, the agent is in Auxiliary Work mode and does not receive calls until the agent becomes available by entering either Auto-in or Manual-in mode.

For non-ACD hunt groups, staffed connotes administered.

super agent pool
A backup skill hunt group of agents that is able to handle any type of call coming into the switch. EAS must be enabled for this pool to be in effect.

T

tandem switch configuration
A LAI call setup that consists of three switches, including the sending switch, the tandem switch, and the far end switch.

The sending switch determines whether the call should be sent to the tandem switch, which, in essence, is the first receiving switch. The tandem switch contains a *route-to* command that is executed on a look ahead basis in the same manner as the sending switch.

If the far end switch (in essence, the second receiving switch) accepts the call, the sending switch is notified, and call control is passed to the far end switch. If the call is denied, the next step of the tandem switch vector is executed.

touch-tone digit code
An alphabetic or numeric code entered by pressing one or more selected buttons on a touch-tone dialing pad. It can be entered from within the customer's network or via a touch-tone telephone external to the network.

touch-tone receiver
A device required to accept the touch-tone digits that are entered by users of the Call Prompting feature.

tracking
Identifies various split flows and other actions relevant to call handling.

treatment
A service provided to the customer or other action taken by call vectoring in response to the customer's placing a call. Examples are: playing an announcement, providing a delay with background music, playing a busy tone, disconnecting a call.

TTR
See **Touch-Tone Receiver (TTR)**.

two switch configuration

A LAI call setup that consists of two switches: sending switch and receiving switch. The vector(s) in the sending switch determine whether the call should be sent to the receiving switch. The vectors in the receiving switch determine whether to accept or deny the call according to a number of conditions.

U

UCD

See **uniform call distribution**.

unconditional branching

The type of vector control flow that is automatically passed from the current vector step to the specified vector or vector step.

uniform call distribution (UCD)

Uses the Most Idle Agent algorithm to route calls.

V

vector

A programmed routing table for processing incoming calls that provide various responses to the caller before the call is answered or receives other treatment. See also **call vectoring**, **vector command**, and **vector step**.

vector-controlled split/skill

A split or skill that is accessed from a vector step that results in queuing to the split/sill while the call is still under vector-processing control.

vector command

A command used in call vectoring to specify the treatment that a call will receive. See also **vector** and **vector step**.

Vector Directory Number (VDN)

An extension number that terminates to a vector. Calls to the VDN are processed by the vector to which the VDN points. When used with call vectoring, the dialed number received by the communications system to provide DNIS is the VDN assigned for that service.

vector number

The vector-assigned value that determines which vector is activated when a call comes into a VDN. Several VDNs may send calls to the same vector.

vector step

A single step of a programmed routing table for processing incoming calls that provide various responses to the caller before the call is answered or receives other treatment. See also **vector**, **call vectoring**, and **vector command**.

VDN

See **Vector Directory Number (VDN)**.

VDN name
The name associated with the VDN. It contains up to 15 characters and is optional. It appears on the agent's display.

VDN of record
See **active VDN**.

VDN override
A VDN Override is an option that allows information about a subsequently-routed-to VDN (if any) to be used instead of the information about the current VDN.

VDN report
A report that reports on VDN activity. The report can be generated by CMS as well as in BCMS. The CMS version of the report summarizes the VDN activity for a specific vector. The BCMS version of the report provides statistical information for all internally-measured VDNs.

VDN skill
An attribute administered to calls on a VDN basis. This skill establishes the requirement for routing a call to an ACD agent with a particular set of abilities (skills).

Voice Response Integration (VRI)
A call vectoring feature which allows a VRU script to be executed while a call is in vector processing and allows data to be exchanged between the PBX and VRU.

Voice Response Unit (VRU)
A computer that uses voice as the media to provide callers with specific information in response to unique requests and commands input to the system by the caller using his/her phone.

VRI
See **Voice Response Integration (VRI)**.

VRU
See **Voice Response Unit (VRU)**.

W

working agent
An agent who is logged into an split or skill and who is not in AUX-work mode.

X

Y

Z

Index