Lucent Technologies
Bell Labs Innovations

DEFINITY®
Enterprise Communications Server
Release 5

CallVisor® ASAI Protocol Reference

555-230-221
Comcode 108001066
Issue 6
June 1997

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Lucent Technologies can assume no responsibility for any errors. Changes and corrections to the information contained in this document may be incorporated into future reissues.

Your Responsibility for Your System's Security

You are responsible for the security of your system. Lucent Technologies does not warrant that this product is immune from or will prevent unauthorized use of common-carrier telecommunication services or facilities accessed through or connected to it. Lucent Technologies will not be responsible for any charges that result from such unauthorized use. Product administration to prevent unauthorized use is your responsibility and your system administrator should read all documents provided with this product to fully understand the features available that may reduce your risk of incurring charges.

Federal Communications Commission Statements

Part 15: Class A Statement. This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio-frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his or her own expense.

Part 68: Network Registration Number. This equipment is registered with the FCC in accordance with Part 68 of the FCC Rules. It is identified by FCC registration number AS593M-13283-MF-E.

Part 68: Answer-Supervision Signaling. Allowing this equipment to be operated in a manner that does not provide proper answer-supervision signaling is in violation of Part 68 rules. This equipment returns answer-supervision signals to the public switched network when:

- Answered by the called station
- Answered by the attendant
- Routed to a recorded announcement that can be administered by the CPE user

This equipment returns answer-supervision signals on all DID calls forwarded back to the public switched telephone network. Permissible exceptions are:

- A call is unanswered
- A busy tone is received
- A reorder tone is received

Trademarks

CallVisor is a registered trademark of Lucent Technologies.
DEFINITY ECS is a registered trademark of Lucent Technologies in the US and throughout the world.
UNIX is a registered trademark in the USA and other countries, licensed exclusively through X/Open Company Limited.
UnixWare is a registered trademark of the Santa Cruz Operation, Inc. in the USA and other countries.

Ordering Information

The ordering number for this document is 555-230-221. To order this document, call the BCS Publications Center at 1 800 457-1235 (International callers use 1 765 361-5353). For more information about Lucent Technologies documents, refer to the *Business Communications Systems Publications Catalog,* 555-000-010.

You can be placed on a Standing Order list for this and other BCS documents you may need. Standing Order will enable you to automatically receive updated versions of individual documents or document sets, billed to account information that you provide. For more information on Standing Orders, or to be put on a list to receive future issues of this document, please contact the Lucent Technologies BCS Publications Fulfillment Center.

Lucent Technologies Fraud Intervention

If you suspect you are being victimized by toll fraud and you need technical support or assistance, call the BCS Technical Service Center Toll Fraud Intervention Hotline at 1 800 643-2353.

Comments

To comment on this document, return the comment card at the front of the document.

Acknowledgment

This document was prepared by the
BCS Product Documentation Development
Lucent Technologies
Middletown, NJ 07748-9972

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Contents

About This Document

This manual provides detailed protocol information for the CallVisor® Adjunct/Switch Application Interface for DEFINITY® Enterprise Communications Server Release 5. (In the interest of brevity, CallVisor ASAI is referred to as ASAI throughout the remainder of this manual.)

Reason for Reissue

This document has been updated with new features up to Release 5.5.

Terminology

See the glossary included in the *DEFINITY Enterprise Communications Server CallVisor ASAI Technical Reference*.

Intended Audience

This document is written for the library or driver programmer of an adjunct computer who is responsible for creating the library of commands for use by the applications programmer. However, this document will also be helpful to any individual who needs a protocol description of the ASAI.

ASAI provides its users with the capability to drive a variety of ECS features. It is essential, therefore, that readers of this document possess extensive knowledge of ECS features and their interactions in addition to ASAI functionality.

NOTE:
See "Related Documents" that follows for a list of documents that provide information on ECS features and ASAI functionality.

Related Documents

Adjunct/Switch Application Interface (ASAI) Specification, 555-025-203

The ASAI Specification document provides a detailed description of the ASAI Reference Model. The Reference Model contains all the capabilities available with ASAI.

DEFINITY Communications System CallVisor ASAI Planning Guide, 555-230-222

This manual provides planning and implementation information for CallVisor ASAI.

DEFINITY Enterprise Communications Server Release 5 CallVisor ASAI Technical Reference, 555-230-220

The Technical Reference is the “companion book” to the Protocol Reference. It provides a detailed functional description of CallVisor ASAI for Release 5.

DEFINITY Enterprise Communications Server Release 5 CallVisor ASAI Overview, 555-230-225

The Overview provides a general description of CallVisor Adjunct Switch Application Interface (ASAI) and applications. This document also describes the functions and services that can be provided using this interface.

DEFINITY Enterprise Communications Server Installation, Administration, and Maintenance of CallVisor ASAI over the DEFINITY LAN Gateway, 555-230-223

This document describes the installation, administration, and maintenance of the ASAI-Ethernet application, which provides ASAI functionality using 10Base-T Ethernet rather than BRI as a transport media.

DEFINITY Enterprise Communications Server Release 5 CallVisor PC ASAI Installation and Reference, 555-230-227

This document provides details on the installation, administration, programming, testing, troubleshooting and engineering of the CallVisor PC.

DEFINITY Enterprise Communications Server Release 5 Administration and Feature Descriptions, 555-230-522

The Administration and Feature Descriptions document serves as an overall reference for the planning, operation, and administration of Release 5 for DEFINITY ECS.

DEFINITY Enterprise Communications Server Release 5 Implementation, 555-230-302

This manual documents the implementation of the DEFINITY ECS.

ISDN Basic Rate Interface (BRI) Specification, 801-802-100

This technical reference provides detailed information about the physical interface structure and message-oriented signaling procedures of the ISDN Basic Rate Interface (BRI).

CallVisor ASAI CD Document Set, 585-246-801

This CD ROM contains four CallVisor ASAI release 5 documents. It includes CallVisor ASAI Technical reference, CallVisor ASAI Protocol Reference, CallVisor ASAI Overview, and CallVisor ASAI PC.

Introduction to Layer 3 Protocol

1

This chapter introduces the layer 3 protocol by explaining how the ISDN messages, Facility Information Elements (FIEs), and Information Elements (IEs) work together to implement ASAI. Message sequences and detailed encoding information are provided.

ASAI Association

An ASAI association is an exchange of one or more messages on the Integrated Services Digital Network Basic Rate Interface (ISDN BRI) signaling channel (D-channel) or on an Ethernet interface between the ECS and the adjunct.

To start an association, the initiating endpoint assigns an idle ISDN Call Reference Value (CRV) and places it into an initiating message that is sent to the serving endpoint. When the serving endpoint accepts this message, the association is initiated and the endpoints may exchange an unlimited number of intermediate messages. To close the association, one endpoint sends a terminating message that frees the CRV for later use.

The three ISDN messages and their functions used to control an association are as follows:

- A Q.931 **REGister** message initiates an ASAI association.
- A Q.931 **FACility** message is the intermediate message.
- A Q.931 **RELease COMplete** message ends an ASAI association.

While the ISDN message controls the association, the Q.932 Facility Information Element (FIE) within these messages carries the ASAI information across the interface. The Facility Information Element (FIE) provides the framework to invoke an ASAI capability and pass parameters associated with the capability.

In addition to the REGister, FACility, and RELease COMplete messages that provide ASAI application level functions across an interface, the ECS uses three other ISDN messages:

1. Management Information Messages (MIMs) (for link maintenance and management)
2. RESTart
3. RESTart ACKnowledge

These messages are discussed in greater detail in Chapter 4, ‘‘Information Elements’’.

ISDN Call Reference Values (CRVs)

ASAI endpoints use ISDN Call Reference Values (CRVs) to associate the sequence of messages carried on an ASAI association. CRVs map one-to-one to ASAI associations and each CRV carries information for only one ASAI association, regardless of how many capabilities are invoked during the association. CRVs are unique on an ASAI link and are present in every ASAI message sent across the interface.

The originating ASAI endpoint assigns an idle CRV for the ASAI association (one not in use for any other ASAI association) and sends the CRV in a REGister message to initiate the association. After the CRV is accepted, and while both endpoints exchange intermediate messages, the CRV contained in each message remains fixed for the duration of the association. When either endpoint passes that CRV in a RELease COMplete message, the association ends. The CRV becomes idle and the originating endpoint may assign the CRV to a later association. Use of a flag bit in the CRV encoding ensures that the two connected ASAI endpoints do not allocate the same CRV on an ASAI.[1] The link administration requirements for an ASAI specify that the length of the CRV value be set to one or two bytes.

The ECS uses CRVs for the following types of ASAI associations:

- **Call Control**

  When an adjunct has control of a call, it may invoke several Call Control capabilities. All call control invocations, together with call feedback, occur on one CRV for any given call.

- **Domain (Station) Control**

  When an adjunct controls calls for a specific station extension, Domain (Station) Control capabilities are invoked. Like Call Control, all domain control invocations, together with call feedback, occur on one CRV for any given call.

  Domain (ACD Split) Control is a subset of domain control and allows the adjunct to obtain agent-related information for agents in the specified ACD split.[2]

- **Notification**

  When an adjunct requests notification of certain events on a given CRV, the messages containing the event reports all use that CRV.

1. Since CRVs are local to an ASAI, endpoints on different interfaces may allocate the same CRVs on their respective interfaces.

2. In the formal ASAI model in 555-025-203, Domain (Station) Call Control and Domain (ACD Split) Control are subsets of Call Control, but for the purpose of this manual, they are discussed as separate capability groups.

- **Routing**

  When the ECS requests routing information for a call using a given CRV, the adjunct returns the route on that same CRV.

- **Request Features**

  The request and response for the ECS feature invocation use the same CRV.

- **Value Queries**

  The request and response for queries use the same CRV.

- **Set Value**

  Both the request and response for the Set Value capability use the same CRV.

- **Audits**

  The adjunct may request an audit operation over a CRV. The ECS uses the same CRV and responds with several messages containing audit information. When all the information has been sent, the ECS terminates the association.

Facility Information Element (FIE) General Description

The CCITT Q.932 Facility Information Element (FIE) identifies the capability being requested for or responded to within an association. The FIE carries ASAI capability information across the ASAI. At most, one FIE may be contained in a **REGister, FACility**, or **RELease COMplete** message. All REGister and FACility messages contain an FIE. All RELease COMplete messages used during normal ASAI operation also contain an FIE. The FIE carries information in a component that has one of four basic structures, explained as follows:

1. An **Invoke** component invokes an ASAI capability and contains:
   - An invoke-id used to identify this capability's invocation within the ASAI association, and used to associate any later result with the specific invocation
   - An Operation Value used to identify the capability
   - Any optional ASAI parameters
2. A **Return Result** component indicates that a previously invoked capability (within this association) has successfully completed, and contains:
   - The invoke-id of the FIE that carried the capability request
   - An optional Operation Value that identifies the completed capability[3]
   - Any optional ASAI parameters with a result
3. A **Return Error** component indicates that a previous ASAI request (within this association) is denied, and contains:
   - The invoke-id of the FIE that carried the capability request
   - An Operation Value that identifies the terminated capability
   - Any optional ASAI parameters with an error
4. A **Reject** component rejects a previous FIE that violates protocol, and contains:
   - The invoke-id of the rejected FIE (if it can be determined)
   - A problem code

3. The optional Operation Value and optional results parameters are both present in or absent from the Return Result component. (One is not present without the other.) This is derived from Q.932.

FIE Acknowledgements

ASAI Capability invocations may be:

acknowledged	The serving ASAI endpoint always responds with either a Return Result component or a Return Error component (an example is the Value Query capability).
unacknowledged	The serving ASAI endpoint does not send a response (an example is the Event Report capability).
acknowledged only on failure	The serving ASAI endpoint sends a Return Error component if it cannot process the request (or if the processing results in an error). The serving endpoint does not return a Return Result component in response to a successful request. Examples are the Routing and Third Party Make Call capabilities.
acknowledged only on success	None of the capabilities supported in the ECS ASAI are in this class.

The messaging procedures in Chapter 2, ‘‘Messaging Sequences and ASAI’’ explain when capability invocations require an acknowledgement. Within an ASAI association, an acknowledgement uses the same invoke-id and CRV in the Return Result or Return Error component as was present in the invoke request.

An endpoint need not wait for a capability to be acknowledged before invoking another capability within the same association. The messaging procedures in Chapter 2, ‘‘Messaging Sequences and ASAI’’ indicate when the requesting endpoint must wait for acknowledgements. For example, an adjunct may send a Third Party Clear Call request at any time during a Call Control association. Also, an ASAI endpoint may send an Abort request any time during any association.

FIE Protocol Errors

An ASAI endpoint may use the Reject component to reject a badly structured FIE or one that violates protocol. When FIE contents violate protocol, the ASAI endpoint may use the Reject component if it is able to determine the message type, CRV, and FIE within a message but not where the FIE contents violate the protocol. Or, the ASAI endpoint may also abort or return an empty RELease COMplete message.

ASAI permits the rejecting endpoint to send the Reject component in either:

- A FACility message if the rejecting endpoint permits the requesting endpoint to continue the association and retry
- A RELease COMplete message if the rejecting endpoint terminates the association when an FIE protocol error occurs

Of these options, the ECS always sends a Reject component in a RELease COMplete message to terminate any association where an FIE protocol violation occurs. The ECS does not permit the adjunct to retry within the same association after an FIE protocol violation.

The ECS does not attempt to retry during any association where an adjunct rejects an FIE sent by the ECS. If the ECS receives a Reject component in a FACility message, it immediately replies with a RELease COMplete message that terminates the association.

Operation Values

As previously noted, each FIE carries ASAI information for an ASAI capability. The Operation Value segment of the FIE component identifies the ASAI capability for which the FIE is carrying information. The Operation Value/Error Value Coding Table in Chapter 4 (Table 4-14 on page 4-48) lists the complete set of Operation Values and their encodings.

Invoke-id Values

Invoke-ids are identifiers that carry binary values within each association (CRV). To ensure orderly acknowledgements within an ASAI association, endpoints must use the following rules to select invoke-ids:

1. With any new request (whether it begins a new association or is one added onto an existing association), the requesting endpoint assigns an invoke-id value for the duration of that request. Because the invoke-id is a binary field, the requesting endpoint may use any binary value except all zeros. ASAI reserves the all zero value. In addition, the endpoint initiating the ASAI association must use invoke-ids with the low order bit set to one; the serving endpoint must use invoke-ids with the low order bit set to zero.

   An endpoint making a new request on an existing association need not be the endpoint that initially requested the association.

2. Invoke-ids for an association are in one of two states:
   - Available — Not assigned to an association
   - In-use — Assigned to an association
3. All invoke-ids are available when an association begins.

4. When an ASAI endpoint invokes an operation, it uses an available invoke-id in the FIE. If the capability is acknowledged, then the invoke-id state changes to “in-use.” If the operation is not acknowledged, then the invoke-id state remains “available.”
5. An endpoint may assign invoke-id values in any order; they do not have to be sequential. Therefore, an endpoint must be able to receive invoke-ids in any order.
6. When an ASAI endpoint receives a Return Result or Return Error component, the associated invoke-id becomes available.
7. If an adjunct re-uses invoke-ids within a single association, it is recommended that it select those ids that have been available for the longest period of time.
8. The initiating endpoint must not use (within a given association) the same invoke-id value for more than one acknowledged operation at a time. The receiving endpoint may reject subsequent requests using an “in-use” invoke-id. In other words, the initiating endpoint must ensure that it does not use an “in-use” invoke-id when invoking another capability.

Denying an ASAI Request

When an endpoint receives a capability request for a service that is permitted in the present ASAI context, but that it cannot provide (such as an invalid value for a request parameter), it responds with a message whose FIE contains a Return Error component and an optional reason for the denial. The return error response must be the first response to the request.

The denial may be carried in:

- A FACility message if the denying endpoint allows the association to continue
- A RELease COMplete message if the denying endpoint does not allow the association to continue

The ECS expects any adjunct denials to terminate the association. If the ECS receives a Return Error component in a FACility message, then the ECS sends a RELease COMplete and terminates the association.

Aborting an ASAI Association

Once an ASAI endpoint has started processing an ASAI request and finds, for some reason, that it cannot continue to process the request, the endpoint may abort the association. The abort mechanism may be used:

- When internal constraints within the ASAI endpoint terminate processing
- When a capability request is made on the wrong association
- When an error, unexpected, or abnormal condition occurs within the ASAI endpoint

Any ASAI endpoint may abort any ASAI association at any time. An ASAI endpoint must be prepared to receive an abort at any time.

To abort an ASAI association, the endpoint sends a RELease COMplete message containing an FIE with a special Abort Operation Value. The message may optionally include a cause.

Messaging Sequences and ASAI

2

This chapter describes the ASAI message sequences for the ASAI capabilities. These message descriptions include information necessary for understanding the procedures, such as the message direction, message type (REGister, FACility, or RELease COMplete), the FIE component type, Operation Value, and the parameters within the FIE.

The descriptions provided in this chapter focus on the information flowing across the ASAI. They are not bit-level descriptions of each message. The latter descriptions for each message are located in Chapter 5, ‘‘Byte Level Messages.’’

Message Conventions

In this chapter, optional parameters in the message will be enclosed by square brackets — [ ].

All procedures implicitly include the denial, association termination, protocol violation, reject, and abort messaging.

The message procedures are presented in this chapter in the following order:

- Common Capabilities (Event Reports)
- Call Control Association
- Domain (Station/ACD) Control Association
- Notification Association
- Routing Association
- Request Feature Association
- Value Query Association
- Set Value Association
- Ending an ASAI Association
- Link Management and Maintenance
- Application Timers

Conventions

Within certain messages, specific information elements are optional and may or may not be present in a specific message. These items are shown in square brackets [ ].

Common Capabilities

The Event Report capability is common to certain other capability groups that require message procedure instructions.

Event Reports

The ECS sends event reports to an adjunct for controlled calls (Third Party or Domain) and monitored calls. A call becomes either controlled or monitored in the following circumstances:

- The adjunct invoked a Third Party Make Call capability to set up the call (the call is controlled). Event reports are sent on the call control association.
- The adjunct invoked a Third Party Take Control capability to take control of the call (the call is controlled). In this case the event reports are sent on the call control association.
- The adjunct invoked the Request Notification capability on a domain and the event report pertains to a call that was offered as an incoming call to the domain (the call is monitored). The event reports are sent on the Request Notification association.
- The call is present at a station extension for which an adjunct has a Domain (Station) Control association. Event reporting for such a call ceases when the call leaves the controlled extension (though it may continue on another association because it enters a monitored domain or arrives at another controlled extension). Event reports are sent on the Domain Control Association.
- The adjunct invoked the Third Party Domain (Split) Control Request for a split domain. Event reports inform the adjunct when the agent(s) has logged into or logged out of the split domain. The event reports are sent on the Domain Control association.

The ECS also sends Charging Event Reports indicating charge advice received for ISDN-PRI calls. These are sent if the adjunct has invoked an Event Notification capability on the domain of all trunk groups. However these calls are not considered monitored by ASAI. No events other than the Charging Event are sent on the Event Notification association for all trunk groups.

Certain call-related event reports indicate that further control of a call is no longer possible. An adjunct might use this information to determine when it might terminate a control association. Event reports are of three types:

1. Those that inform the adjunct of some event; the control association continues and adjunct control of the call is still possible.
2. Those that may indicate that no further call control is possible; the only additional feedback about this call will be Third Party Call Ended Operation.

3. Call Ended Event Reports, which terminate a Call Control association. A RELease COMplete message carries the call ended operation and terminates the association. Within a notification association, a FACility message carries the Call Ended operation so that the association is not terminated and notification of any future calls will continue to occur.

Events may be sent to monitoring, call control, and domain control adjuncts as a result of either:

- A manual operation
- A request from another association controlling the call or endpoint

An endpoint making an ASAI call control request receives an acknowledgement, not an event report, such as when there is domain control on both stations of a call, and one of the associations is used to request a hold. The requesting association gets an acknowledgement, and the other association gets a Hold Event Report.

All call-related event reports (which *excludes* the Login and Logout Event Reports) contain the ASAI call identifier for the call. This identifier specifies the particular call the event report is for when the ECS sends event reports for multiple calls on a notification or extension control association.

Table 2-1. Use of Event Reports in Associations

Event Report	Domain (Station) Control	Call Control	Notification	Domain (Split) Control
Alerting	yes	yes	yes	no
Answered	yes	yes	no	no
Busy/Unavailable	yes	yes	yes	no
Call Conferenced	yes	yes	yes	no
Call Ended (FACility)	no	no	yes	no
Call Initiated	yes	no	no	no
Call Offered to Domain	no	no	yes	no
Call Originated	yes	no	no	no
Call Redirected	yes	no	yes	no
Call Transferred	yes	yes	yes	no
Charging	no	no	yes[1]	no
Connected	yes	yes	yes	no
Cut-Through	yes	yes	yes	no
Disconnect/Drop	yes	yes	yes	no
Entered Digits	no	yes	yes	no
Hold	yes	yes	yes	no
Login	no	no	no	yes
Logout	no	no	no	yes
Queued	yes	yes	yes	no
Reconnected	yes	yes	yes	no
Reorder/Denial	yes	yes	yes	no
Trunk Seized	yes	yes	yes	no

1. Charging Event Reports are only sent on the Notification association associated with all trunk groups. They are not sent on Split or VDN Notification associations.

Call-Related Event Reports

The event reports in this section show that further call control is possible within the Call Control and Domain (Station) Control associations. Also, the ECS may send additional event reports.

Alerting Event Report

The ECS sends the adjunct a FACility message. Note that within a Domain (Station) Control Association, this event is sent for both incoming calls alerting at the controlled extension and also for far-end alerting for calls originating at the controlled extension.

The message contains an invoke FIE with:

Operation Value = Event Report,
an “alerting” event (Specific Event IE),
a party_id (Party ID IE),
the calling number (Calling Number IE or Trunk Identifier IE),
the called number (Called Number IE),
[the number that is alerting if that party is local (Connected Number IE)],
[the reason for redirection] (Cause IE),
[the split that has distributed the call to an alerting agent, if any (Domain IE)],
a call_id (Call Identity IE),
[a cause value (Cause IE)],
[User-to-User Information] (User-User IE), and
[originating line information] (Originating Line IE).

The Alerting Event Report contains one of the following values for the Calling Number IE or Trunk identifier IE:

(For incoming calls:)

- An on-PBX extension passed in the Calling Number IE
- An off-PBX CPN/BN passed in the Calling Number IE
- The group number of the incoming trunk passed in the Trunk Identifier IE

(For outgoing calls:)

- An on-PBX extension originating the call

NOTE:
A Switch-Classified call (destination alerting first option) appears as an incoming call to the ACD agent. Thus, the Alerting Event Report for the agent and the Call Offered to Domain Event Report will contain the external number as the “caller” and the split or VDN extension as the “called” number.

For coding, see ‘‘Alerting Event Report’’ on page 5-3 of Chapter 5, “Byte Level Messages.”

Answered Event Report

The ECS sends this event report when either the destination answers or the call classifier detects SIT administrated for answer treatment. The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
an “answer” event (Specific Event IE),
a call_id (Call Identity IE),
[the answering number] (Connected Number IE),
the dialed number (Called Number IE),
a party_id (Party ID IE), and
the tone the classifier detected (Cause IE).

For coding, see ‘‘Answered Event Report’’ on page 5-6 of Chapter 5, “Byte Level Messages.”

Call Conferenced Event Report

When a local party on a monitored (or controlled) call uses a voice instrument (set) to conference another party onto the call, or when another association conferences two calls for a party, the ECS sends the Call Conferenced Event Report to the monitoring (controlling) association(s).

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “conferenced” event (Specific Event IE),
the call_id for the resulting conference call after
the conference operation (Call Identity IE)[1],
the call_id of the other call before the conference operation
(Call Identity IE),
the conferencing party’s number (Calling Number IE),
the conferenced party’s number (Called Number IE),
a list of up to six old party identifiers for the parties on the
call (Old Party ID IE)
a list of up to six party identifiers for the parties on the
call (Party ID IE), and
a list of up to six extensions of the parties on the call
(Connected Number IE).

For coding, see ‘‘Call Conferenced Event Report’’ on page 5-9 of Chapter 5, “Byte Level Messages.”

1. This is always one of the call_ids of the two calls being merged.

Call Initiated Event Report

The ECS sends the Call Initiated Event Report when a domain-controlled extension goes off-hook and receives dial tone.

This event is only sent within Domain (Station) Control associations.

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a call_id (Call Identity IE),
a party_id of the extension initiating the call (Party ID IE), and
a “call initiated” event (Specific Event).

For coding, see ‘‘Call Initiated Event Report’’ on page 5-11 of Chapter 5, “Byte Level Messages.”

Call Offered to Domain Event Report

The ECS sends a FACility message with an invoke FIE containing:

Operation Value = Event Report,
call offered event (Specific Event IE),
a call identifier that refers to this call
in any ASAI association (Call Identity IE),
the calling number (Calling Party Number IE or Trunk Identification IE)[2],
the called number (Called Party Number IE),
[Split or VDN number] (Domain IE),
[Lookahead Interflow information] (Lookahead Interflow IE),[3]
[digits collected by the ECS call prompting unit] (User Code IE),
[User-to-User Information] (User-User IE),
[Flexible Billing] (Feature IE), and
[originating line information] (Originating Line IE).

For coding, see ‘‘Call Offered to Domain Event Report’’ on page 5-12 of Chapter 5, “Byte Level Messages.”

NOTE:
A Switch-Classified call (destination alerting first option) appears as an incoming call to the ACD agent. Thus, the Alerting Event Report for the agent and the Call Offered to Domain Event Report will contain the external number as the “caller” and the split or VDN extension as the “called” number.

2. The ECS supplies the data that the network has passed to the ECS with the incoming call: CPN, or BN, but not both. The Calling Party IE is present when the calling number is known; it is mutually exclusive with the Trunk Identification IE, which is present when the number is not known. This is described in the Alerting Event Report section.

3. The ECS passes incoming Lookahead Interflow IE from incoming PRI calls with the first call offered event report for the incoming call.

Call Originated Event Report

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value=Event Report,
a “call originated” event (Specific Event IE),
a call_id (Call Identity IE),
a party_id (Party Identifier IE),
the connected number (Connected number IE)
the calling number (Calling number IE), and
a dialed number (Called Number IE), and
[User-to-User Information] (User-User IE).

For coding, see ‘‘Call Originated Event Report’’ on page 5-15 of Chapter 5, “Byte Level Messages.”

Call Redirected Event Report

The ECS sends this event report when a call leaves a monitored Automatic Call Distribution (ACD) split, monitored Vector Directory Number (VDN), or controlled extension. The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “call redirected” event (Specific Event IE), and
the call_id of the call (Call Identity IE).

For coding, see ‘‘Call Redirected Event Report’’ on page 5-17 of Chapter 5, “Byte Level Messages.”

Call Transferred Event Report

When a local party on an adjunct-monitored (or adjunct-controlled) call uses a voice instrument to transfer the call to another party, or when another association transfers a call on behalf of a party, the ECS sends the Call Transferred Event Report to the monitoring (controlling) adjunct(s).

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “transfer” event (Specific Event IE),
the call_id of the resulting transferred call (Call Identity IE),
the other call_id before the transfer (Call Identity IE),
the transferring party’s number (Calling Number IE),
the transferred party’s number (Called Number IE),
a list of up to six old party identifiers for the parties on the call (Old Party ID IE),
a list of up to six party identifiers for the parties on the call (Party ID IE), and

a list of up to six extensions of the parties on the call (Connected Number IE).

For coding, see ‘‘Call Transferred Event Report’’ on page 5-18 of Chapter 5, “Byte Level Messages.”

Charging Event Report

The ECS sends this event report when an ISDN-PRI trunk sends charge advice for an outbound call placed on that trunk. This event report is only sent over the Notification association for the domain of all trunk groups.

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “charging” event (Specific Event IE),
a call_id (Call Identity IE),
the called number (Called Number IE),
the charging number (Calling Number IE),
the trunk group and member (Trunk ID IE),
type of charge (Data Item IE),
charge amount (Data Item IE)
[party ID of trunk] (Party ID IE),
[reason for error] (Cause IE).

For coding, see ‘‘Charging Event Report’’ on page 5-20 of Chapter 5, “Byte Level Messages.”

Connected Event Report

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “connected” event (Specific Event IE),
a party_id (Party ID IE),
a call_id (Call Identity IE),
the calling number (Calling Number IE or Trunk ID IE),
the dialed number (Called Number IE),
cause giving type of answer supervision (Cause IE), and
[the number that actually answered the call if that party is local] (Connected Number IE)
[originating line information] (Originating Line IE).

For coding, see ‘‘Connected Event Report’’ on page 5-22 of Chapter 5, “Byte Level Messages.”

NOTE:
A Switch-Classified call (destination alerting first option) appears as an incoming call to the ACD agent. Thus, the Alerting and Connected Event Reports for the agent and the Call Offered to Domain Event Report will

contain the external number as the “caller” and the split or VDN extension as the “called” number.

Cut-Through Event Report

The ECS maps a Primary Rate Interface (PRI) PROGress message to an ASAI Cut-Through Event Report for nonswitch-classified calls. A PRI network may send the ECS a PROGress message for a variety of reasons contained in a Progress Indicator (within the PROGress message). The ECS forwards the Progress Indicator to the adjunct in the cut-through event.

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “cut through” event (Specific Event IE),
a party_id (Party ID IE),
the progress information from the ISDN network
(Progress Indicator IE), and
a call_id (Call Identity IE).

For coding, see ‘‘Cut-Through Event Report’’ on page 5-24 of Chapter 5, “Byte Level Messages.”

Disconnect/Drop Event Report

The ECS sends the Disconnect/Drop Event Report when a party drops from a multiparty call. The ECS does not send this event report for the last party on the call since doing so would be redundant with sending Call Ended. The ECS does send it for the last party dropped on domain-controlled associations.

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “party dropped from call” event (Specific Event IE),
a party_id (Party ID IE),
[the number of the party that dropped if local] (Connected Number IE),
a call_id (Call Identity IE),
[the tone a classifier detected if the classifier reports a drop] (Cause IE), and
[User-to-User Information (User-User IE)].

For coding, see ‘‘Disconnect/Drop Event Report’’ on page 5-25 of Chapter 5, “Byte Level Messages.”

Entered Digits Event Report

The ECS sends an Entered Digits Event Report to the adjunct when the ECS has collected the number of digits previously requested. The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
an “entered digits” event (Specific Event IE),

the call_id for the call (Call Identity IE), and
the digit(s) entered (User Entered Code IE).

For coding, see ‘‘Entered Digits Event Report’’ on page 5-27 of Chapter 5, “Byte Level Messages.”

Hold Event Report

The ECS sends the Hold Event Report when a local party on a monitored call puts the call on hold. Within a Domain (Station) Control association this means that the ECS sends the Hold Event Report for not only the controlled extension, but for all the local endpoints on calls that are present at the controlled extension.

The ECS sends the adjunct a FACility message with the invoke FIE containing:

Operation Value = Event Report,
a “party held call” event (Specific Event IE),
a party_id (Party ID IE),
the extension number of the party that held (Connected Number IE), and
a call_id (Call Identity IE).

For coding, see ‘‘Hold Event Report’’ on page 5-28 of Chapter 5, “Byte Level Messages.”

Queued Event Report

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “call queued” event (Specific Event IE),
a split identifier (Domain IE),
the number of calls now in the queue (Counter IE),
the dialed number (Called Party Number IE), and
a call_id (Call Identity IE).

For coding, see ‘‘Queued Event Report’’ on page 5-33 of Chapter 5, “Byte Level Messages.”

Reconnected Event Report

The ECS sends the Reconnect Event Report when a local party on a monitored call reconnects to a held call. Within a Domain (Station) Control association this means that the ECS sends the Reconnected Event Report for not only the controlled extension, but for all the local endpoints on calls that are present at the controlled extension.The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “party reconnected to call” event (Specific Event IE),
a party_id (Party ID IE),
the extension number of the party that reconnected

(Connected Number IE), and
a call_id (Call Identity IE).

For coding, see ‘‘Reconnected Event Report’’ on page 5-35 of Chapter 5, “Byte Level Messages.”

Trunk Seized Event Report

The ECS sends a Trunk Seized Event Report to the adjunct when a nonswitch-classified call leaves the ECS on a non-PRI facility. The application *may* only receive Connected and Dropped Event Reports for the far-end party, following the Trunk Seized Event Report. The ECS sends the Trunk Seized Event Report, for example, when the destination for a Third Party Make Call capability is off the ECS, the call uses a non-PRI trunk, and the call is manually classified; or when an incoming call is transferred or forwarded to a remote endpoint over a non-PRI facility.

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “trunk seized” event (Specific Event IE),
a party_id (Party ID IE),
the dialed number (Called Number IE), and
a call_id (Call Identity IE).

For coding, see ‘‘Trunk Seized Event Report’’ on page 5-37 of Chapter 5, “Byte Level Messages.”

Call-Related Event Reports When Ending Adjunct Control

The ECS sends the event reports described in this section within Call Control associations, extension control associations, and notification associations.

When sent for Call control, these event reports imply that limited further adjunct call control is possible. Subsequent to “busy” and “reorder,” the adjunct can still send a Third Party Drop or a Third Party Clear call. “Busy” and “reorder” events do not terminate the extension control association.

Busy and Reorder Event Reports are followed by a Call Ended Event Report for call control and event notification associations when the ECS frees the resources associated with the call.

Busy/Unavailable Event Report

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “busy” event (Specific Event IE),
the call_id for the call (Call Identity IE),

a cause (Cause IE), and
the called number (Called Number IE).

For coding, see ‘‘Busy/Unavailable Event Report’’ on page 5-8 of Chapter 5, “Byte Level Messages.”

Reorder/Denial Event Report

The ECS sends the adjunct a FACility message with an invoke FIE containing:

Operation Value = Event Report,
a “denial” event (Specific Event IE),
the call_id for the call (Call Identity IE),
the called number (Called Number IE), and
a cause (cause IE).

For coding, see ‘‘Reorder/Denial Event Report’’ on page 5-36 of Chapter 5, “Byte Level Messages.”

Example causes:	classifier not available	trunks not available
	split queue full	encountered vector busy step
	split has no logged-in agents	

Call Ended

The ECS sends the Call Ended Event Report when it frees the internal ECS resources associated with a controlled/monitored call (for example, the call has terminated). The ECS sends Call Ended in a RELease COMplete on a Call Control association so that the association is terminated. For coding, see ‘‘Third Party Call Ended — Association Terminates’’ on page 5-66 of Chapter 5, “Byte Level Messages.”

The ECS sends the Call Ended operation in a FACility message for a notification association so that the association continues. For coding, see ‘‘Notification: Call Ended — Association Continues’’ on page 5-102 of Chapter 5, “Byte Level Messages.” The ECS sends the adjunct a RELease COMplete or FACility message with an invoke FIE containing:

Operation Value = Third Party Call Ended,
the call_id of the call (Call Identity IE), and
a cause (Cause IE).

The cause generally indicates that the call terminated normally, or that the call terminated as a result of a transfer operation.

Non-Call Related Event Reports

Logout Event Report

The ECS sends the Logout Event Report on a Domain (Split) Control Association.

The ECS sends a FACility message with an invoke FIE containing:

Operation Value = Event Report
a logout event (Specific Event IE)
the split (Domain IE),
the agent's physical extension[4] (Domain IE),
[the agent's logical extension[4]] (Domain IE), and
[reason code[5]] (Domain IE).

For coding, see ''Logout Event Report — Domain (ACD Split/Skill) Control Association'' on page 5-31 of Chapter 5, "Byte Level Messages."

Login Event Report

The ECS sends the Login Event Report on a Domain (Split) Control Association.

The ECS sends a FACility message with an invoke FIE containing:

Operation Value = Event Report
a login event (Specific Event IE)
the split (Domain IE),
the agent's physical extension[4] (Domain IE),
[the agent's logical extension[4]] (Domain IE), and
work mode (Domain IE).

For coding, see ''Login Event Report — Domain (ACD Split/Skill) Control Association'' on page 5-29 of Chapter 5, "Byte Level Messages."

4. In an EAS environment, both the logical and physical extension are provided. In an ACD environment, only the physical extension is provided.

5. This IE is included only if the System-Parameters Feature field, Logout Reason Codes is "forced" or "requested" and the agent is logging out with a valid reason code (1 to 9).

Third Party Control Associations

The ECS provides three types of Third Party control associations:

1. Call Control, which monitors and controls all parties on a specified call
2. Third Party Domain (Station) Control, which monitors all calls at a specific station and allows control of the station only
3. Third Party Domain (ACD Split) Control, which monitors logout events for all agents in a given split

These control capability groups encompass call feedback event reports and call control operations (although, as Table 2-2 shows, there are different subsets).

Table 2-2. Use of Call Control Capabilities in Third Party Associations

Call Control Capability	Domain (Station) Control	Call Control	Domain (Split) Control
Third Party Make Call (I)	no	yes	no
Third Party Take Control (I)	no	yes	no
Domain Control Request (I)	yes (Extension)	no	yes (ACD split)
Third Party Auto Dial	yes	no	no
Third Party Drop	yes	yes	no
Third Party Hold	yes	yes	no
Third Party Merge	yes	yes	no
Third Party Reconnect	yes	yes	no
Third Party Answer	yes	no	no
Redirect Call	yes	yes	no
Send DTMF Digits	yes	yes	no
Third Party Call Ended/RELease COMplete (T)	no	yes	no
Third Party Clear Call (T)	no	yes	no
Third Party Relinquish Control (T)	yes	yes	yes
Domain Control Ended (T)	yes	no	yes
Third Party Selective Disconnect	no	yes	no
Third Party Selective Reconnect	no	yes	no

(I) is an initiating capability

(T) is a terminating capability

These procedures provide descriptions of the messaging procedures.

Call Control Association

A Call Control association allows an adjunct to control all the endpoints on a call using those Call Control capabilities shown in Table 2-2. Call control includes: establishing a call, taking control of an existing call, controlling a call, and the call feedback (event reports) that the ECS provides about a controlled call.

Initiating a Call Control Association

An adjunct begins a Call Control association and obtains control of a call when it:

1. Invokes the ASAI Third Party Make Call capability to set up a call
2. Invokes the ASAI Third Party Take Control capability to obtain control of an existing call

Call Control and Event Reporting on a Call Control Association

Once the association has been successfully established, the ECS designates the associated call as an adjunct-controlled call and thereby provides call feedback event reports. During the time the Call Control association exists, the adjunct can request Call Control operations.

The ECS terminates the association when the call terminates; the adjunct may use Third Party Relinquish Control to terminate the association when it no longer needs to control the call.

Termination of a Call Control Association

Either the adjunct or the ECS may terminate a Call Control association.

Three ways an adjunct can terminate such associations are as follows:

- Use the Third Party Clear Call procedure. This disconnects all parties from the call and terminates the association.
- Use the Third Party Relinquish Control procedure. This does not dismantle the call. The ECS continues normal processing of the call although adjunct control of the call (and call feedback) is terminated.
- Send RELease COMplete. For coding, see ‘‘Call Control: Normal Clearing Terminates Call Control Association’’ on page 5-67 of Chapter 5, “Byte Level Messages.”

The ECS terminates a Call Control association in two ways:

1. If the call terminates and the ECS frees call-associated resources, then the ECS invokes the Call Ended capability. For coding, see ‘‘Third Party Call Ended — Association Terminates’’ on page 5-66 of Chapter 5, “Byte Level Messages.”
2. An internal ECS audit detects that ECS resources are allocated for Call Control of a call that no longer exists. If the ECS detects that such an association exists, the ECS sends a RELease COMplete containing an invoke FIE with:

   an Operation Value = Abort and
   a cause indicating that an on-PBX switch audit terminated the association.

   For coding, see ‘‘Call Control: Internal switch Audit Finds Stale Call Control CRV’’ on page 5-65 of Chapter 5, “Byte Level Messages.”

If the adjunct uses RELease COMplete to terminate a Call Control association for an active, stable call, the ECS *does not* disconnect the call. Rather, the ECS terminates the ability of the adjunct to control that call (this is the same as relinquish control).

In addition, either the ECS or adjunct may send a RELease COMplete message with an abort operation value to terminate a Call Control association.

ASAI considers both the Third Party Relinquish Control and the more efficient RELease COMplete to be normal termination of the association. Both have the same effect within the ECS.

In general, if the ECS receives any RELease COMplete message for a Call Control association, the ECS continues to process the call normally. The exception to this occurs when the ECS receives any RELease COMplete message on a Call Control association for a switch-classified call while the call is in the classification stage (for example, has not yet been classified). In this case, the ECS dismantles the corresponding call on receipt of the RELease COMplete message.

Third Party Make Call — Initiating Procedure

The Third Party Make Call procedure includes the following sequence of messages:

1. The adjunct sends a REGister message to begin a Call Control association on a call reference value. The message contains:

   an invoke FIE,
   an invoke identifier,
   Operation Value = third party make call, and parameters for:
   originating address (Calling Party IE),

destination address (Called Party IE),
[Service Circuit = call classifier] (Service Circuit IE),
[number of rings before destination “no-answer” classification] (Call options IE),
[alerting order] (Call options IE),
[priority] (Call options IE),
[supervisor assist flag] (Call options IE),
[trunk access code or ARS/AAR digits] (Domain IE),
[trunk access code] (Domain IE),
[direct agent call flag] (Call Options IE),
[answer machine detection] (Call Options IE),
[ACD split extension for direct-agent call] (Domain IE),[6]
[*return_ack* flag if the optional “proceed” is desired] (Call Options IE), and [User-to-User Information] (User-User IE).

The Trunk Access Code in the Domain IE may contain either a TAC or ARS/AAR digits. TAC or ARS/AAR may optionally be included in the destination address. For coding, see ‘‘Third Party Make Call Request’’ on page 5-39 of Chapter 5, “Byte Level Messages.”

2. If ECS provisioning permits the ECS to accept the adjunct’s request and the adjunct has included the *return_ack* flag in the request, then once the ECS originates the call and assigns a call_id, the ECS sends a FACility message to acknowledge the request. The message contains an invoke FIE with Operation Value = Proceed, the extension of the phone originating the call (Connected Number IE), the party_id of the originator (Party ID IE), and the call_id of the call (Call ID IE). For coding, see ‘‘Acknowledgment of Third Party Make Call Request’’ on page 5-55 of Chapter 5, “Byte Level Messages.”
3. Various sequences of event reports and adjunct requests for call control may occur. In terms of the message procedure, the event reporting and call control are a sequence of FACility messages flowing across the interface. The call control procedures section details the messages for each call control procedure.
4. The ECS continues to send the adjunct events about the call. The adjunct may continue to request call control operations.
5. The association terminates when the ECS or adjunct takes any of the actions described in “Termination of a Call Control Association” earlier in this chapter.

6. The ACD split extension for a direct-agent call must be present when direct-agent flag is also present. When these two parameters are present, the destination address must not be a logical agent extension.

Third Party Take Control — Initiating Procedure

The adjunct uses this capability to take control of a call for future Call Control operations. The adjunct must have learned about the call, which could have been initiated manually, from an event report or query. The event reports and certain query responses include a call identifier that the adjunct may later use as a parameter in a Third Party Take Control request to create a new Call Control association.

When the adjunct uses Third Party Take Control to take control of a call that was once offered to an active notification split or vector domain, the ECS sends the event reports for the call over both the call control association and the request notification association. The adjunct receives duplicate event reports about a call unless it uses the Stop Call Notification capability to cease the event reporting for that call on the Notification Association.

1. The adjunct sends a REGister message to begin a Call Control association for the call on a new call reference value. The message contains an invoke FIE with:

   an invoke identifier,
   Operation Value = Third Party Take Control, and
   an argument with a call identifier (Call Identity IE).

   This REGister message allocates a CRV for a Call Control association over which the adjunct may send third party call control requests.

   For coding, see ‘‘Third Party Take Control Request’’ on page 5-42 of Chapter 5, “Byte Level Messages.”

2. If the request is successful, the ECS replies with a FACility message containing a return result FIE with:

   the invoke-id from the Take Control invocation,
   Operation Value = Take Control,
   a list of up to six party identifiers for the
   parties on the call (Party ID IE), and
   a list of up to six extensions of the parties
   on the call (Connected Number IE).

   The FACility message does not close the association. The invoke-id in the return result has the same value as the invoke-id in the Third Party Take Control request.

   For coding, see ‘‘Acknowledgment of Third Party Take Control Request’’ on page 5-57 of Chapter 5, “Byte Level Messages.”

   If the request is not successful, the ECS returns an error to terminate the new call control association. A RELease COMplete message carries this failure message.

Third Party Relinquish Control — Terminating Procedure

Third Party Relinquish Control terminates the association but does not disconnect the call. The association is terminated and the ECS stops sending the adjunct call feedback for the call. The ECS denies a relinquish control request for a switch-classified call in the process of being classified.

To relinquish control, the following messaging takes place:

1. The Adjunct sends a FACility message containing an FIE with an invoke component and Operation Value = Third Party Relinquish Control. For coding, see ‘‘Third Party Relinquish Control Request’’ on page 5-49 of Chapter 5, “Byte Level Messages.”
2. If the ECS accepts the relinquish control request, the ECS replies with a RELease COMplete message containing an FIE with a return result component. The invoke-id in the return result has the same value as the invoke-id in the Third Party Relinquish Control request. For coding, see ‘‘Call Control: Acknowledgment — Association Terminates’’ on page 5-63 of Chapter 5, “Byte Level Messages.”

Domain (Station) Control Procedure

The Domain (Station) Control allows an adjunct to:

1. Monitor call-related events for all calls present at a specific station extension
2. Perform call control activity for that station extension (and only that station extension)
3. Initiate calls outbound from the station extension (and only that station extension)

The adjunct uses the Domain (Station) Control Request capability to initiate the association. While the association exists, the ECS sends the adjunct event reports about any call at that station. The adjunct may use the Auto Dial capability to establish a call within the existing Domain (Station) Control association and the adjunct may use selected call control capabilities to control calls within the association.

Domain (ACD Split) Control

The Domain (ACD Split) Control allows an adjunct to receive agent-related event reports for agents in the specified ACD split.

The adjunct uses the Domain Control Request capability to initiate an agent control association. While the associations exists, the ECS sends the adjunct agent login and agent logout reports. Table 2-2 earlier in this chapter shows the subsets of the control capabilities that are used on a Domain (ACD Split) Control Association.

Domain Control Request — Initiating Procedure

The adjunct uses this capability to establish a domain (station or ACD split) control association. All call event reports on the domain (station) association include a call identifier that the adjunct may later use as a parameter in call control requests to specify the call being acted on at the controlled extension.

1. The adjunct sends a REGister message to begin a Domain (Station) Control association on a new call reference value. The message contains an invoke FIE with:

   an invoke identifier,
   Operation Value = Domain Control, and
   an argument with the number of the extension to be controlled or the extension number of the ACD split for agent related events (Domain IE).

This REGister message allocates a CRV for a Domain (Station) Control association. For coding, see ‘‘Domain Control (Station/ACD Split) Request’’ on page 5-70 of Chapter 5, “Byte Level Messages.”

2. If the ECS accepts the Domain Control request, the contents of the acknowledgement depend on whether the domain control association is for a station or split:

   - Station control acknowledgement:
     a FACility message containing a return result FIE with:

     the invoke-id in the control request,
     Operation Value = Third Party Domain (Station) Control, and
     parameters containing a list of:
     [call_ids (Call Identifier IE)],
     [party_id of the principal's extension on the call (Party ID IE)],
     and [the state of the principal's extension on
     the call (Specific Event IE)].

     The contents of the above FIE are present if and only if calls are present at the station. If no calls are present, the contents of the response are the same as the acknowledgement for a domain control request. For coding, see ''Acknowledgment of Domain (Station) Control Request'' on page 5-84 of Chapter 5, "Byte Level Messages."

   - Agent control acknowledgement:
     a FACility message containing a return result FIE with:

     the invoke-id in the control request

     For coding, see ''Domain Control: Acknowledgment (No Parameters) Association Continues'' on page 5-89 of Chapter 5, "Byte Level Messages."

3. If the request for Domain Control is unsuccessful, the ECS returns an error to terminate the new Domain Control association. A RELease COMplete message carries this failure message and the Domain Control association is not established.

Cancel Domain Control — Terminating Procedure

The adjunct terminates a Domain (Station) Control Association using the Third Party Relinquish Control capability in exactly the same way as it uses that capability to terminate a Call Control association.

Domain Control Ended — Terminating Procedure

The ECS uses this capability to terminate a Domain (Station/ACD Split) Control association. The ECS ends the Domain (Station/ACD Split) Control when, for example, the ECS administrator removes the controlled extension or ACD split domain from the ECS translation.

The ECS sends a RELease COMplete message to terminate the association. The message contains an invoke FIE with:

Operation Value = Domain Control Ended, and
a cause (Cause IE).

For coding, see ‘‘switch Ends Domain (Station) Control Association’’ on page 5-94 of Chapter 5, “Byte Level Messages.”

Auto Dial Procedure

The adjunct can use the Auto Dial procedure over an existing Domain (Station) Control association to begin an outbound call for the controlled extension. The ECS reports event reports for the call within the existing Domain (Station) Control association and the adjunct may invoke call control operations for the call also within the Domain (Station) Control association.

The ECS sends a Call Initiated Event Report when the user goes off-hook and the ECS allocates a call_id that is subsequently used for the call. The station user may go off-hook/idle before the adjunct sends the Auto Dial request. The Call Initiated Event Report contains the call_id for the resulting call.

The auto dial procedure includes the following sequence of messages:

1. The adjunct sends a FACility message on an existing extension control association. The message contains:

   an invoke FIE,
   an invoke identifier,
   Operation Value = AUTO DIAL,
   and parameters for:
   [trunk access code (Domain IE)],
   destination address (Called Party IE),
   [priority] (Call options IE),
   [*return_ack* flag if the optional “proceed” is desired] (Call Options IE),
   and [User-to-User Information] (User-User IE).

   For coding, see ‘‘Third Party Auto Dial Request for an Extension’’ on page 5-77 of Chapter 5, “Byte Level Messages.”

   The Trunk Access Code in the Domain IE may also optionally contain either TAC or ARS/AAR digits, or these can be in the called number. USE OF THE RETURN-ACK OPTION IS NOT RECOMMENDED.

2. If the ECS accepts the Auto Dial request and the adjunct has included the *return_ack* flag in the request, then once the ECS originates the call and assigns a call_id, the ECS sends a FACility message to acknowledge the request. The message contains an invoke FIE with:

   Operation Value = Proceed,
   the call_id (call Identity IE) of the resulting call, and
   the party_id of the originator (Party ID IE).

   For coding, see ‘‘Acknowledgment of Third Party Auto Dial Request’’ on page 5-86 of Chapter 5, “Byte Level Messages.”

If the ECS cannot accept the request, the ECS returns a denial. For coding, see ‘‘Domain Control: Request is Denied — Association Continues’’ on page 5-90 of Chapter 5, “Byte Level Messages.”

3. Various sequences of event reports and adjunct requests for call control may occur. In terms of the message procedure, the event reporting and call control are a sequence of FACility messages flowing across the interface.

Third Party Answer Procedure

An adjunct may use the Third Party Answer capability within an existing Domain (Station) Control association to answer a call at the controlled extension. Use of this capability with certain types of stations (including analog) may require user action (going off-hook before a timer expires) for station set types that the ECS cannot take off-hook remotely.

1. The adjunct sends a FACility message on an existing station control association. The message contains:

   an invoke FIE,
   an invoke identifier,
   Operation Value=Third Party Answer,
   and parameters for:
   the call identifier (Call Identifier IE).

2. If ECS provisioning permits the ECS to accept the Third Party Answer request and the ECS successfully completes the request, then the ECS sends a FACility message to return a successful result. The message contains:

   a return result FIE with:
   Operations Value=Third Party Answer, and
   the invoke-id from the request.

3. If the ECS cannot complete the request, the ECS returns a denial. For coding, see ‘‘Domain Control: Request is Denied — Association Continues’’ on page 5-90 of Chapter 5, “Byte Level Messages.”

Call Control Procedures

Once the adjunct has control of a third party call, within either a Call Control or Domain (Station) Control, it may invoke various call control capabilities.

The adjunct passes the CRV for the Call Control or Station Control association in the FACility message containing these capability requests.

If the ECS denies any of these requests, a FACility message carries the denial so that the call control association continues. Certain parameters are shown as optional; their use depends on whether the call is being controlled over a Call Control association or a Domain (Station) Control association:

- The call_id should not be included in requests on Call Control associations since such an association controls only one call. If an adjunct should include a call_id in such a request, the ECS will ignore the call_id. The call_id must always be included in requests on Domain (Station) Control associations since such an association may control more than one call at an extension.
- The party_id should not be included in requests on Domain (Station) Control associations since such an association controls only one endpoint on the call. If an adjunct should include a party_id in such a request, the ECS will ignore the party_id. The party_id must always be included in requests on a Call Control association since such an association may control more than one party on a call.

Third Party Drop Procedure

The adjunct requests that the ECS drop a party from a call that this ASAI association controls.

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Selective Drop and
   [a party to be dropped from the call] (Party ID IE), or
   [the call being controlled] (Call Identity IE),
   [User-to-User Information] (User-User IE), and
   [drop tones (Resource Identifier IE)[7]].

   For call control coding, see ‘‘Third Party Selective Drop Request’’ on page 5-45 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Third Party (Domain) Selective Drop Request’’ on page 5-72 of Chapter 5, “Byte Level Messages.”

7. Permitted on Call Control only. Not allowed on Domain Control.

2. The ECS drops the party and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the third party drop request. For call control coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-89 of Chapter 5, “Byte Level Messages.”

Third Party Hold Procedure

The adjunct requests that the ECS put a call on hold with respect to a given party.

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Hold and
   [a party for which the call will be placed
   on hold] (Party ID IE), or
   [the call being controlled] (Call Identity IE).

   For call control coding, see ‘‘Third Party Selective Hold Request’’ on page 5-46 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Third Party (Domain) Selective Hold Request’’ on page 5-73 of Chapter 5, “Byte Level Messages.”

2. The ECS puts the call on hold and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the Third Party Hold request. For call control coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-89 of Chapter 5, “Byte Level Messages.”

Third Party Reconnect Procedure

The adjunct requests that the ECS reconnect a party to a call that this ASAI association controls.

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Reconnect, and
   [a party for which the call will be reconnected] (Party ID IE), or
   [the call being controlled] (Call Identity IE).

   For call control coding, see ‘‘Third Party Reconnect Request’’ on page 5-47 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Third Party (Domain) Reconnect Request’’ on page 5-74 of Chapter 5, “Byte Level Messages.”

2. The ECS reconnects the party to the call and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the third party reconnect request. For call control coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-89 of Chapter 5, “Byte Level Messages.”

Third Party Merge Procedure

The adjunct requests that the ECS merge two calls controlled by an ASAI association or associations. Both calls have a party in common. Prerequisite to requesting a merge, the adjunct must place a call on hold with respect to the common party. After a successful merge, the ECS sends an acknowledgement on the call control association over which it received the merge request.

1. The adjunct, using the same association that was used to put the call on hold, sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Merge, and
   [the party that is common with respect to the held call (Party ID IE)], or
   [the held call (Call Identity IE)],
   the call_id that refers to the other call (Call identity IE), and
   an indication of whether the merge is a conference or transfer (Conference/Transfer IE).

   For call control coding, see ‘‘Third Party Merge Request’’ on page 5-48 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Third Party (Domain) Merge Request’’ on page 5-75 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a return result FACility message containing:

   Operation Value = Third Party Merge,
   a call identifier for the resulting call[8] (Call Identity IE),
   a list of up to six old party identifiers (Old Party-ID IE),
   a list of up to six party identifiers for the parties on the call (Party ID IE),
   and a list of up to six extensions of the parties on the call (Connected Number IE).

   The invoke-id in the return result has the same value as the invoke-id in the Third Party Merge request. For call control coding, see ‘‘Acknowledgment of Third Party Merge Request’’ on page 5-59 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Acknowledgment of Third Party Merge Request (Domain)’’ on page 5-87 of Chapter 5, “Byte Level Messages.”

8. The call identifier may be different from the call identifiers for the calls being merged. The ECS/adjunct uses this new call identifier for the merged call in all subsequent event reports and other interactions.

Adjuncts receiving event reports for the calls involved in the Third Party Merge receive an acknowledgement, Transfer Event Report, Conference Event Report, or Call Ended depending on the role they played in the merge, the type of merge, and what they are monitoring.

Third Party Clear Call Procedure

The Third Party Clear Call capability may be used only in Call Control associations and may not be used in Third Party Domain (Station) Control associations. The adjunct requests that the ECS disconnect all parties on a call.

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Clear Call.

   For call control coding, see ‘‘Third Party Clear Call Request’’ on page 5-44 of Chapter 5, “Byte Level Messages.”

2. The ECS sends a RELease COMplete acknowledgement containing a return result FIE with the invoke-id in the acknowledgement having the same value as the invoke-id in the Third Party Clear Call Request. For coding, see ‘‘Call Control: Acknowledgment — Association Terminates’’ on page 5-63 of Chapter 5, “Byte Level Messages.”

Send DTMF Signals Procedure

The adjunct requests that DTMF tones be sent on a call to selected parties.

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value=Send DTMF,
   call on which the tones are to be sent (Call Identity IE),
   [a list of up to five parties that will hear the DTMF tones] (Party ID IE),
   the DTMF tones (User Data IE),
   [tone duration] (Call Options IE), and
   [pause duration] (Call Options IE).

   For call control coding, see ‘‘Third Party Send DTMF Digits Request’’ on page 5-52 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Third Party (Domain) Send DTMF Digits Request’’ on page 5-80 of Chapter 5, “Byte Level Messages.”

2. The ECS sends the DTMF tone and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the send DTMF request. For call control coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-89 of Chapter 5, “Byte Level Messages.”

Redirect Call Procedure

Starting with G3V4, the Redirect Call procedure includes the following sequence of messages:

1. The ECS sends the adjunct a FACility message with an invoke FIE containing:

   Operation Value = Redirect Call,
   [the party the call is redirected from] (Party ID IE),
   the number that the call is redirected to (Redirection Number IE), and
   [the call_id of the call to be redirected] (Call Identity IE).

   For call control coding, see ‘‘Redirect Call’’ on page 5-54 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Redirect Call (Domain)’’ on page 5-82 of Chapter 5, “Byte Level Messages.”

2. The ECS redirects the call and replies with a return result FACility Message. The invoke-id in the return result has the same value as the invoke-id in the Redirect Call request.

   For call control coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-89 of Chapter 5, “Byte Level Messages.”

Third Party Listen Disconnect Procedure

NOTE:
Third Party Listen Disconnect capability may only be used in the Third Party Call Control Associations and may not be used in the Third Party Domain (Station) Control associations.

The adjunct requests that the ECS selectively disconnect a party (a listener) with respect to a given party or parties [a talker(s)].

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Listen Disconnect,
   a party which will be listen-disconnected (the listener)
   (Party ID IE),
   [a party from which the listener is disconnected (the talker)
   (party ID IE)].

   For coding, see ‘‘Third Party Listen Disconnect Request’’ on page 5-50 of Chapter 5, “Byte Level Messages.”

2. The ECS listen-disconnects the call and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the Listen Disconnect request. For coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.”

Third Party Listen Reconnect Procedure

NOTE:
Third Party Listen Reconnect capability may only be used in the Third Party Call Control Associations and may not be used in the Third Party Domain (Station) Control associations.

The adjunct requests that the ECS selectively reconnect a party (a listener) with respect to a given party or parties [talker(s)].

1. The adjunct sends the ECS a FACility message containing an invoke FIE with:

   Operation Value = Third Party Listen Reconnect,
   a party which will be listen-reconnected (the listener)
   (Party ID IE),
   [a party from which the listener is disconnected (the talker)
   (Party ID IE)].

   For coding, see ‘‘Third Party Listen Reconnect Request’’ on page 5-51 of Chapter 5, “Byte Level Messages.”

2. The ECS listen-reconnects the call and replies with a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the Listen Reconnect request. For coding, see ‘‘Call Control: Acknowledgment (No Parameters) Association Continues’’ on page 5-61 of Chapter 5, “Byte Level Messages.”

Notification Association Procedure

An adjunct can request that the ECS supply event reports over three types of domains: ACD splits, Vector Directory Numbers (VDNs) and trunk groups. The ACD split domains cannot be administered as controlled splits or vector-controlled splits. The event reports provide the adjunct with call related information.

The event reports for these domains all contain a call identifier that associates call-related event reports with specific calls.

1. The adjunct sends a REGister message to begin a notification association.

   The message contains an invoke FIE with:

   Operation Value = Request Notification, and
   the split or VDN domain (Domain IE).

   For coding, see ‘‘Event Notification Request’’ on page 5-98 of Chapter 5, “Byte Level Messages.”

2. The ECS sends a FACility message to acknowledge the request. The message contains a return result FIE. The invoke-id has the same value as the invoke-id in the notification request. For coding, see ‘‘Notification: Acknowledgement (No Parameters) Association Continues’’ on page 5-101 in Chapter 5, “Byte Level Messages.”

3. The ECS sends appropriate event reports to the adjunct via FACility message.

4. If, during the normal course of event reporting, the adjunct needs to terminate the event reporting for any given call within the notification association, the adjunct may send a FACility message. The message contains an invoke FIE with:

   Operation Value = Stop Call Notification and
   a call identifier for the call (Call Identity IE).

   For coding, see ‘‘Stop Notification on Call Request’’ on page 5-100 of Chapter 5, “Byte Level Messages.”

   a. If the call_id is valid, then the ECS stops notification for this call within the Notification Association and sends an ACK to the adjunct. The ACK is a return result FACility message. The invoke-id in the return result has the same value as the invoke-id in the Stop Notification request. For coding, see ‘‘Notification: Acknowledgement (No Parameters) Association Continues’’ on page 5-101 in Chapter 5, “Byte Level Messages.”

   b. If the call_id is not valid, the ECS returns a FACility containing a Return Error component and a cause.

Stop Call Notification is not supported on the Notification association for the domain of all trunk groups.

5. If, during the normal course of event reporting, the ECS has reason to terminate the event reporting (such as the domain being administered out of existence), the ECS sends a RELease COMplete message to terminate the association. The message contains an invoke FIE with:

   Operation Value = Notification Ended and
   a cause (Cause IE).

   For coding, see ‘‘switch Ends Notification Reporting Association’’ on page 5-105 of Chapter 5, “Byte Level Messages.”

6. To terminate the event reporting association, the adjunct sends a FACility message continuing an invoke FIE with:

   Operation Value = Cancel Notification.

   For coding, see ‘‘Cancel Event Notification Request’’ on page 5-99 of Chapter 5, “Byte Level Messages.”

   a. The ECS acknowledges the cancel with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the Cancel Notification Request.

      For coding, see ‘‘Notification: Acknowledgement (No Parameters) Association Terminated’’ on page 5-103 of Chapter 5, “Byte Level Messages.”

Routing Association Procedure

The ECS can ask the adjunct to supply a route for an incoming call.

1. The ECS sends a REGister message containing an invoke FIE with:

   Operation Value = Route, and
   parameters containing [CPN/BN] (Calling Party Number IE or Trunk Identification),[9]
   the dialed number (Called Party Number IE),
   the call identifier for the call to be routed (Call Identity IE),
   the VDN making the route request (Domain IE),[10]
   [PRI Lookahead Interflow information] (Lookahead Interflow IE),
   [digits collected by the ECS prompter] (User Code IE),
   [User-to-User Information] (User-User IE),
   [originating line information] (Originating Line IE), and
   [flexible billing] (Feature IE).

   For coding, see ‘‘Call Route Request’’ on page 5-109 of Chapter 5, “Byte Level Messages.”

   The ECS does not use the ASAI *return_ack* flag. The adjunct does not return an acknowledgement to the ECS on receipt of the routing request; it sends a route when one is available.

   If the vector step following the route step is not a WAIT or ANNOUNCEMENT or ADJUNCT ROUTE, the ECS goes to the next Step and sends a Route End to the adjunct to terminate the Routing association. See the *Adjunct/Switch Application Interface (ASAI) Specification*, 555-025-203, for more details.

2. The adjunct responds with one of the following:

   a. If the adjunct accepts the routing request and returns a route, it responds with a FACility message containing an invoke FIE with:

      Operation Value = Route Select,
      a route for the call (Called Party Number IE),
      [the origination number] (Calling Party Number IE),
      [option for a priority call] (Call Option IE),
      [option for a direct agent call] (Call Option IE),
      [Trunk Access Code necessary for external routing] (Domain IE),
      [ACD Split extension] (Domain IE)[11],

9. The ECS supplies the data that the network has passed to the ECS with the incoming call: CPN or BN, but not both. The Calling Party IE is present when the calling number is known; it is mutually exclusive with the Trunk Identification IE which is present when the number is not known.

10. This is typically the dialed number. However, if an initial VDN (with routing) directs the call to a second VDN (also with routing), then the dialed number is the first VDN and the Domain IE contains the number of the second VDN when it requests a route.

11. Present if and only if direct agent call option is also present. If the ACD split extension and the direct agent call option are present, the called number must contain a physical extension in the EAS environment.

[User-to-User Information] (User-User IE),
[ASAI-provided digits] (User Code IE),
[collect digits request] (User Code IE)[12], and
[event to terminate digit collection] (specific event IE).

For coding, see ‘‘Call Route Selection’’ on page 5-112 of Chapter 5, “Byte Level Messages.”

b. If the adjunct denies the request, it responds with a RELease COMplete message containing a return error FIE with Operation Value = Route, and a cause (Cause IE). In this case, the Step following the “adjunct route” step is skipped. For coding, see ‘‘Routing: Request is Denied — Association Terminated’’ on page 5-115 of Chapter 5, “Byte Level Messages.”

3. The ECS sends a Route End within a RELease COMplete message to terminate the routing association. The ECS sends this message in one of three ways:

a. The ECS responds to a Route Select. The RELease COMplete message contains an invoke FIE with:

Operation Value = Route End and
a cause (Cause IE).

The Cause is “normal termination” if the ECS accepted and used the adjunct-supplied route; the cause is “invalid number” if the ECS could not use the route.

For coding, see ‘‘End Adjunct Routing’’ on page 5-116 of Chapter 5, “Byte Level Messages.”

b. The adjunct supplied an invalid route in the route select message: The RELease COMplete message contains an invoke FIE with:

Operation Value = Route End and
a cause (Cause IE).

The Cause may be, for example, “bad permissions,” “agent not a member of split,” or “destination busy.”

For coding, see ‘‘End Adjunct Routing’’ on page 5-116 of Chapter 5, “Byte Level Messages.”

c. The adjunct has not supplied a route within the allotted time and the ECS terminates the routing association. The ECS sends a RELease COMplete message to terminate the association. The cause is “Recovery on Timer Expiry.” After the ECS sends the Route End message, any Route Select received from the adjunct for this association is ignored. (It is treated as a protocol error but the call is not affected.)

12. An application may either request digit collection or provide digits, but not both. The “event to release tone detection” is optional for digit collection and is not present for “provide digits.”

An adjunct may always abort a routing association to terminate that association.

Request Feature Association

All feature requests are invocations of the ASAI Feature Request Capability. An adjunct should be prepared to accept an ECS denial of any feature request.

Agent Login Procedure

The adjunct logs an agent into a given ACD split/skill.

1. To request an agent login, the adjunct sends a REGister message with an invoke FIE containing:

   Operation Value = feature request, and
   parameters for
   login feature (Feature IE),
   login identifier (User Code IE),
   ACD split (Domain IE),
   agent extension (Domain IE), and
   [initial work mode] (Domain IE).

   For coding, see ‘‘Agent Login Request’’ on page 5-120 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request. For coding, see ‘‘Request Feature: Acknowledgment — Association Terminates’’ on page 5-132 of Chapter 5, “Byte Level Messages.”

   When logging in EAS Agents, the following also apply:

   - The User Entered Code IE contains both the agent’s password and the login_id, separated by a “#.”
   - The split parameter must be present in the message as it is a mandatory ASAI parameter. (Since the EAS agents’ skill assignments are administered on the ECS, the system ignores the parameters in an EAS environment.)
   - The Domain IE must contain the agent’s physical (not logical) extension.

Agent Logout Procedure

The adjunct logs an agent out of a given ACD split/skill.

1. To request an agent logout, the adjunct sends a REGister message with an invoke FIE containing:

   Operation Value = feature request, and
   parameters for
   logout feature (Feature IE),
   ACD split (Domain IE),
   agent extension (Domain IE), and
   [reason code[13]] (Domain IE).

   For coding, see ''Agent Logout Request'' on page 5-122 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request. For coding, see ''Request Feature: Acknowledgment — Association Terminates'' on page 5-132 of Chapter 5, "Byte Level Messages."

   When logging out EAS agents, the agent extension (Domain IE) may contain either the agent's physical extension or logical extension.

Agent Work Mode Change Procedure

The adjunct changes an ACD agent work mode on a given split.

1. The adjunct sends a REGister message to request change of work mode. The message contains an invoke FIE with:

   Operation Value = feature request, and
   parameters for
   work mode change (Feature IE),
   ACD split (Domain IE),
   agent extension (Domain IE),
   new work mode (Domain IE), and
   [reason code[14]] (Domain IE).

   For coding, see ''Change Agent Work Mode Request'' on page 5-124 of Chapter 5, ''Byte Level Messages."

13. This IE is included only if the System-Parameters features field, Logout Reason Codes is "forced" or "requested," otherwise the ECS ignores the IE.

14. This IE is included only if the new work mode value is Auxiliary Work and the System-Parameters features field, AUX Reason Codes is "forced" or "requested," otherwise the ECS ignores the IE.

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request. For coding, see ‘‘Request Feature: Acknowledgment — Association Terminates’’ on page 5-132 of Chapter 5, “Byte Level Messages.”

   When changing work mode for EAS Agents, the following also applies:

   - The split parameter must be present as it is a mandatory ASAI parameter. (Since the EAS agents’ skill assignments are administered on the ECS, the system ignores the parameters in an EAS environment.)
   - The Domain IE may contain either the agent’s physical or logical extension.
   - The new work mode applies to all the agent skills.

Call Forwarding Activation Procedure

The adjunct requests that forwarding be activated from a given local station extension to another number (possibly external to the ECS).

1. The adjunct sends a REGister message to request forwarding activation. The message contains an invoke FIE with:

   Operation Value = feature request, and
   parameters for forwarding activation (Feature IE),
   forwarding extension with unconditional forwarding
   (Redirecting IE without optional byte 4), and
   destination for the calls (Redirection IE without optional byte 4).

   For coding, see ‘‘Call Forwarding Feature Activation’’ on page 5-128 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request.

   For coding, see ‘‘Request Feature: Acknowledgment — Association Terminates’’ on page 5-132 of Chapter 5, “Byte Level Messages.”

   If the feature cannot be activated, the ECS responds with a denial message containing a cause. For coding, see ‘‘Request is Denied — Association Terminated’’ on page 5-131 of Chapter 5, “Byte Level Messages.”

Call Forwarding Cancel Procedure

The adjunct requests that forwarding be canceled from a given local station extension.

1. The adjunct sends a REGister message to request forwarding cancel. The message contains an invoke FIE with:

   Operation Value = feature request, and
   parameters for forwarding cancel (Feature IE) and
   forwarding extension (Redirecting IE without optional byte 4).

   For coding, see ''Cancel Call Forwarding Feature Activation'' on page 5-129 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request.

   For coding, see ''Request Feature: Acknowledgment — Association Terminates'' on page 5-132 of Chapter 5, "Byte Level Messages."

   If the feature cannot be canceled, the ECS responds with a denial message containing a cause. For coding, see ''Request is Denied — Association Terminated'' on page 5-131 of Chapter 5, "Byte Level Messages."

Send All Calls Activation Procedure

The adjunct requests that Send All Calls be activated for a given local station extension.

1. The adjunct sends a REGister message to request Send All Calls. The message contains an invoke FIE with:

   Operation Value = feature request, and
   parameters for Send All Calls (Feature IE) and
   station extension (Redirecting IE without optional byte 4).

   For coding, see ''Send All Calls Feature Activation'' on page 5-126 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request.

   For coding, see ''Request Feature: Acknowledgment — Association Terminates'' on page 5-132 of Chapter 5, "Byte Level Messages."

   If the feature cannot be activated, the ECS responds with a denial message containing a cause. For coding, see ''Request is Denied — Association Terminated'' on page 5-131 of Chapter 5, "Byte Level Messages."

Cancel Send All Calls Procedure

The adjunct requests that Send All Calls be canceled for a given local station extension.

1. The adjunct sends a REGister message to request forwarding cancel. The message contains an invoke FIE with:

   Operation Value = feature request, and
   parameters for cancelling Send All Calls (Feature IE) and
   station extension (Redirecting IE without optional byte 4).

   For coding, see ‘‘Cancel Send All Calls Feature Activation’’ on page 5-127 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the feature request.

   For coding, see ‘‘Request Feature: Acknowledgment — Association Terminates’’ on page 5-132 of Chapter 5, “Byte Level Messages.”

   If the feature cannot be canceled, the ECS responds with a denial message containing a cause. For coding, see ‘‘Request is Denied — Association Terminated’’ on page 5-131 of Chapter 5, “Byte Level Messages.”

Value Query Association

All value queries are invocations of the ASAI Value Query capability. An adjunct should be prepared to accept the ECS's denial to a value query.

Date/Time Query Procedure

The adjunct queries the ECS for its current date and time.

1. The adjunct sends a REGister message to query for date/time. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   a parameter for date/time (Item IE).

   For coding, see ''Date/Time Query'' on page 5-145 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query and
   the date and time (Date/Time IE).

   For coding, see ''Response to Date/Time Query'' on page 5-168 of Chapter 5, "Byte Level Messages."

Split Status Query Procedure

The adjunct queries the ECS for status information on an ACD split.

1. The adjunct sends a REGister message to query for split status. The message contains an invoke IE with:

   Operation Value = Value Query, and
   a parameter for ACD split (Domain IE).

   For coding, see ''ACD Split Status Query'' on page 5-136 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query,
   the number of calls in the ACD queue (Counter IE),
   the number of agents logged into the split (Counter IE), and
   the number of available agents (Counter IE).

   For coding, see ''Response to ACD Split Status Query'' on page 5-153 of Chapter 5, "Byte Level Messages."

Agent Status Query Procedure

The adjunct queries the ECS for status information on an agent in an ACD split.

1. The adjunct sends a REGister message to query for agent status with an invoke FIE containing:

   Operation Value = Value Query, and
   parameters for ACD split (Domain IE) and
   agent extension (Domain IE).

   For coding, see ''Station Status Query'' on page 5-141 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query,
   the agent's work state (Domain IE), and
   the agent's talk state (Domain IE), and the
   [reason code] (Domain IE)[15].

   For coding, see ''Response to Agent Status Query'' on page 5-155 of Chapter 5, "Byte Level Messages."

   In a query for an EAS agent, the agent extension Domain IE may contain either the agent's physical or logical extension. The Split parameter (Domain IE) must be present since it is a mandatory ASAI IE. The ECS ignores the contents of the split parameter for EAS agents in Skills.

Station MWL Status Query Procedure

The adjunct can use the Message Waiting Lamp Status Query to obtain information about the ON/OFF status of the Message Waiting Lamp at a given extension. The query response breaks down which messaging services have an ON/OFF setting in the ECS.

1. The adjunct sends a REGister message to query for Message Waiting Lamp Status.

   The message contains an invoke FIE with:

   Operation Value=Value Query, and
   parameters for Message Waiting Lamp Status query (Item IE), and
   an extension (Calling Number IE).

   For coding, see ''Station Feature Query — Message Waiting Lamp'' on page 5-138 of Chapter 5, "Byte Level Messages."

15. This IE will be only included in the response if the System-Parameter Features field, AUX Reason Codes, is "forced" or "requested," and the Agent's Work State is Auxiliary.

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value=Value Query, and
   a parameter containing the MWL status for the messaging services (Status IE).

   For coding, see ‘‘Response to Station Message Waiting Lamp Status Query’’ on page 5-157 of Chapter 5, “Byte Level Messages.”

Station Send-All-Calls Query Procedure

The adjunct can use the Send All Calls Status Query to obtain information about the ON/OFF status of the Send All Calls feature at a given extension.

1. The adjunct sends a REGister message to query for Send All Calls Status. The message contains an invoke FIE with:

   Operation Value=Value Query, and
   parameters for the Send All Calls Status query (Feature IE) and an extension (Calling Number IE).

   For coding, see ‘‘Station Feature Query — Send All Calls’’ on page 5-139 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value=Value Query, and
   a parameter containing the Send All Calls status (Status IE).

   For coding, see ‘‘Response to Station Send-All-Calls Status Query’’ on page 5-158 of Chapter 5, “Byte Level Messages.

Station Call Forwarding Status Query Procedure

The adjunct can use the Call Forwarding Status Query to obtain information about the ON/OFF status of the Call Forwarding feature at a given extension. When the feature is active at the station, the response returns the number where the ECS will forward calls.

1. The adjunct sends a REGister message to query for Call Forwarding Status. The message contains an invoke FIE with:

   Operation Value=Value Query, and
   parameters for Call Forwarding Status query (Feature IE) and an extension (Calling Number IE).

   For coding, see ‘‘Station Feature Query — Call Forwarding’’ on page 5-140 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

Operation Value=Value Query, and
a parameter containing the Call Forwarding status (Status IE),
a parameter containing the forwarded to extension (Redirection Number IE).

For coding, see ‘‘Response to Station Call Forwarding Status Query’’ on page 5-159 of Chapter 5, “Byte Level Messages.”

Station Status Query Procedure

The adjunct queries the ECS for status information on an extension.

1. The adjunct sends a REGister message to query for extension status. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   a parameter for extension (Domain IE).

   For coding, see ‘‘Station Status Query’’ on page 5-141 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query, and
   a parameter for the extension’s talk state (Domain IE).

   For coding, see ‘‘Response to Station Status Query’’ on page 5-160 of Chapter 5, “Byte Level Messages.”

   The Domain IE in a Station Status Query for an EAS agent may contain either the agent’s physical or logical extension.

Trunk Group Query Procedure

The adjunct queries the ECS for the number of idle/in-use trunks in a trunk group.

1. The adjunct sends a REGister message to query for trunk information. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   a parameter indicating the trunk group (Domain IE).

   For coding, see ‘‘Trunk Group Status Query’’ on page 5-142 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query, and
   a parameter indicating the number of idle trunks and
   the number of trunks in use (Trunk Group Status IE).

   For coding, see ‘‘Response to Trunk Status Query’’ on page 5-161 of Chapter 5, “Byte Level Messages.”

Call Classifiers Query Procedure

The adjunct queries the ECS for the number of idle/in-use classifiers.

NOTE:
Starting with R5.5, if the **USA Default Algorithm?** field is set to *n* on the System Parameters OCM Call Classification form, the response to this query will provide the total number of ports for the TN744D and TN2182B boards (and any later versions) that are idle and in-use.This means that the response will not include any TN744C ports or earlier versions, nor will it include any TN2182 ports. If *y* is entered in this field, the query will provide the total number of idle and in-use ports for all the TN744 and TN2182 boards.

1. The adjunct sends a REGister message to query for classifier information. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   a parameter for classifiers (Resource IE).

   For coding, see ''Call Classifiers Status Query'' on page 5-143 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query, and
   a parameter indicating the number of classifiers in-use, and
   the number of classifiers that are idle (Resource Status IE).

   For coding, see ''Response to Number of Call Classifiers Query'' on page 5-162 of Chapter 5, "Byte Level Messages."

Calls Query Procedure

The adjunct can use the Active Call Query to obtain information about calls at a station extension.

1. The adjunct sends a REGister message to query for active call information. The message contains an invoke FIE with:

   Operation Value = Value Query,
   parameters for active calls (Item IE), and
   a station extension (Domain IE).

   For coding, see ''Call Information at Station Query'' on page 5-146 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

Operation Value = Value Query, and
parameters containing a list of call_ids (Call Identifier IE),
the party_id of the principal's extension on the call (Party ID IE), and
the state of the principal's extension on the call (Specific Event IE).

For coding, see ''Response to Call Query'' on page 5-163 of Chapter 5, "Byte Level Messages."

The contents of the Facility IE, ''Response to Call Query'' on page 5-163 in Chapter 5, after the invoke identifier are present if and only if calls are present at the station. If no calls are present, the ECS responds with a RELease COMplete message containing the information upto and including the invoke identifier value from the request.

The Domain IE in a Calls Query procedure for an EAS agent may contain the agent's physical or logical extension.

Party ID Query Procedure

The adjunct can use the Party ID Query to obtain information about the parties on a given call.

1. The adjunct sends a REGister message to query for party_id information. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   parameters for party_id query (Item IE) and
   a call_id (Call Identifier IE).

   For coding, see ''Party ID Information on Call Query'' on page 5-147 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query, and
   parameters containing a list of party ids (Party ID IE) and
   extensions (Connected Number IE).

   For coding, see ''Response to Party ID Query'' on page 5-165 of Chapter 5, "Byte Level Messages."

Extension Information Query Procedure

The adjunct can use the Extension Query to obtain information about a given extension.

1. The adjunct sends a REGister message to query for extension information. The message contains an invoke FIE with:

   Operation Value = Value Query, and
   parameters for extension query (Item IE), and
   an extension (Calling Number IE).

For coding, see ‘‘Extension Type/Class Information Query’’ on page 5-148 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE with:

   Operation Value = Value Query, and
   a parameter containing the extension type[16] [and the class of the station][17] (Domain IE).

   For coding, see ‘‘Response to Extension Information Query’’ on page 5-167 of Chapter 5, “Byte Level Messages.”

Agent Login Audit Procedure

The adjunct queries the ECS for information on the agents logged into a given ACD split.

1. The adjunct sends a REGister message with an invoke FIE containing:

   Operation Value = Value Query, and
   parameters for an ACD split (Domain IE), and
   an agent login audit (Item IE).

   For coding, see ‘‘ACD Agent Login Audit Query’’ on page 5-144 of Chapter 5, “Byte Level Messages.”

2. If the request is successful, the ECS replies with a number of FACility messages, each with an invoke FIE containing:

   Operation Value = Response Continued, and
   up to eight agent extensions (Domain IEs).

   For coding, see ‘‘Response to Agent Login Query (List of Logged-in Agent Extensions)’’ on page 5-151 of Chapter 5, “Byte Level Messages.”

3. After the ECS has sent as many FACility messages as necessary to supply the agent information, it sends a RELease COMPlete message to terminate the association. The RELease COMPlete message contains a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the original value query. For coding, see ‘‘Value Query: Acknowledgment — Association Terminates’’ on page 5-170 of Chapter 5, “Byte Level Messages.”

16. ACD split, VDN, announcement, data extension, (voice) station, ASAI, logical agent, and other.

17. If the type is station (voice), the address field indicates either “analog,” “BRI,” or “proprietary.” If the type is ASAI, the address field indicates “No Bearer Capabilities.” If the type is logical agent and if the agent is logged in, the address field indicates the physical extension of the agent

Integrated Directory Database Query Procedure

This query allows an application to pass the ECS an extension number and query for the corresponding name in the Integrated Directory database.

1. The adjunct sends a REGister message to query for an Integrated Directory name. The message contains an invoke FIE with:

   Operation Value=Value Query, and
   parameters for
   a name database query (Domain IE), and
   an extension (Domain IE).

   The domain type in the extension parameter must be for a type of extension that the Integrated Directory Database query feature supports (Hunt Group, Station, TAC, VDN, Announcement, Data Extension, Logical Agent). However, the extension domain type does not have to match the actual type of the queried number (it just has to be one of the permitted values).

   For coding, see ''Integrated Directory Database Query'' on page 5-149 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMPlete message containing a return result FIE with:

   Operation Value=Value Query; and
   parameters containing:
   the actual type and number of the queried extension (Domain IE), and
   the name (Data Item IE).

   If there is no administered name, the ECS sends this message without the actual type and name.

   For coding, see ''Response to Integrated Directory Query'' on page 5-169 of Chapter 5, "Byte Level Messages."

Set Value Association

All set value requests are invocations of the ASAI Set Value Capability.

Activate Message Waiting Lamp Procedure

The adjunct turns on a message waiting lamp (MWL) at a given extension.

1. The adjunct sends a REGister message to light the MWL.

   Operation Value = Set Value, and
   parameters for
   MWL Value (Item IE),
   Message waiting On (Status IE), and
   Extension (Called Party Number IE).

   For coding, see ''Activate Message Waiting Lamp Request'' on page 5-177 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the request.

   For coding, see ''Set Value: RELease COMplete Messages Sent by the switch'' on page 5-182 of Chapter 5, "Byte Level Messages."

Deactivate Message Waiting Lamp Procedure

The adjunct turns off a MWL at a given extension.

1. The adjunct sends a REGister message to extinguish a MWL. The message contains an invoke FIE with:

   Operation Value = Set Value,
   parameters for
   MWL Value (Item IE)
   Message waiting off (Status IE), and
   Extension (Called Party Number IE).

   For coding, see ''Deactivate Message Waiting Lamp Request'' on page 5-178 of Chapter 5, "Byte Level Messages."

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the request. For coding, see ''Set Value: RELease COMplete Messages Sent by the switch'' on page 5-182 of Chapter 5, "Byte Level Messages."

Billing Change Request Procedure

The adjunct requests a billing change for a given call.

1. The adjunct sends a REGister message to request a billing change. The message contains an invoke FIE with:

   Operation Value = Set Value, and
   parameters for
   Flexible Billing (Item IE)
   Call ID (Call Identity IE)
   Billing request type and amount (Generic Billing IE)

   For coding, see ‘‘Billing Change Request’’ on page 5-179 of Chapter 5, “Byte Level Messages.”

2. The ECS replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the request. For coding, see ‘‘Set Value: RELease COMplete Messages Sent by the switch’’ on page 5-182 of Chapter 5, “Byte Level Messages.”

Ending an ASAI Association

An ASAI endpoint terminates an association for the following reasons:

- The ASAI messaging procedure terminates normally. The endpoint sends the RELease COMplete message prescribed by the messaging procedure for the type of association that is in progress. ASAI procedures use RELease COMplete messages containing either (or both) an FIE and/or Cause IE. The FIE carries an Invoke, Return Result, or Return Error component. The messaging procedures for any given type of association determine the message used for termination.
- The endpoint must deny the requested operation. The endpoint understands the message, parses and understands the FIE, but the service request is invalid, the parameters are invalid, or the endpoint must deny service for some other reason. The ECS sends this message only as the first response to an ASAI request; otherwise it aborts the ongoing association.
- The endpoint must abort an ongoing association. The endpoint has begun to service an ASAI request, but cannot process it further.
- The endpoint cannot parse an FIE. The endpoint understands the message, detects an FIE within the message, but a protocol violation exists within the FIE.
- The endpoint cannot parse a layer 3 message. Message Parsing may fail in a number of ways.

If the adjunct passes Return Error or Reject components to the ECS in a FACility or REGister message, the ECS aborts the association on that CRV.

NOTE:
If a valid request is made by an adjunct over an incorrect association, the ECS terminates that association (sends a RELease COMplete).

Endpoint Denies a Request

When an endpoint receives a request for a service that is permitted in the present ASAI context, but cannot provide the service for some reason (such as an invalid value for a request parameter), it responds with a message containing an FIE with a Return Error component, Operation Value = the Operation Value from the FIE being denied, and an optional reason for the denial (Cause IE). The return error response must be the first response to the request. For coding, see ‘‘Call Control: Request is Denied — Association Terminated’’ on page 5-64 of Chapter 5, “Byte Level Messages.”

The denial may be carried in:

- A FACility message if the denying endpoint allows the association to continue. The ECS may use a FACility message for a denial of:

  - — The Call Control capabilities (Third Party Hold, Third Party Drop, Third Party Merge, Third Party Reconnect, and Third Party Relinquish control).
  - — The Stop Notification capability request within a Notification association.
  - — The Auto Dial capability in a Domain (Station) Control association.
- A RELease COMplete message if the denying endpoint does not allow the association to continue. The ECS always uses a RELease COMplete message for any denial of all other requests.

The ECS expects any adjunct denials to terminate the association. If the ECS receives a Return Error component in a FACility message, then the ECS sends a RELease COMplete message and aborts the association.

The ECS sends a denial if an Invoke FIE contains ASAI parameters that are incompatible with one another. Determination of incompatible parameters is capability-specific.

When the ECS is overloaded it may deny initiating requests (REGister messages) and return a Cause of CS0/42, “ECS equipment congested.” This cause is returned inside the Facility IE in the same way as any other denial.

An adjunct may also make a request for a service not provisioned on the ECS. The ECS responds with a denial containing a Cause of CS0/50, “requested facility not subscribed.” This cause is returned inside the Facility IE in the same way as any other denial.

If Flexible Billing is not allowed for a call, the ECS responds with a denial containing a cause of CS0/29 (Facility Reject) or CS0/98 (Message not Compatible with Call State).

The ECS may also return the Return Error component in a FACility message when it denies a request within a Call Control association, Domain (Station) Control association, or Notification association and the association is to continue.

For call control coding, see ‘‘Call Control: Request is Denied — Association Continues’’ on page 5-62 of Chapter 5, “Byte Level Messages.” For domain control coding, see ‘‘Domain Control: Request is Denied — Association Continues’’ on page 5-90 of Chapter 5, “Byte Level Messages.”

Endpoint Application Level Software Aborts ASAI Processing

Either the ECS or adjunct may abort an ASAI association at any time. An ASAI endpoint must be prepared to receive an abort at any time. To abort an association, the endpoint sends a RELease COMplete message. The message

contains: invoke FIE with Operation Value = Abort, and an optional cause (Cause IE). The ECS always supplies a cause when it aborts an association. The abort mechanism may be used:

- When internal constraints within the ASAI endpoint terminate processing;
- When a capability request is made on the wrong type of association;
- When an error, unexpected, or abnormal condition occurs within the ASAI endpoint that terminates processing.

Special cases include the following:

- The ECS always expects to receive Reject and Return Error components from the adjunct in RELease COMplete messages. If an adjunct sends one of the components in a FACility message, then the ECS aborts the association.
- The ECS aborts an association if it receives a message with an unexpected or unrecognized Operation Value. The ECS sends a RELease COMplete message containing an FIE with a Invoke component. The Invoke component contains a cause CS0/111 (protocol error). An example is the denial of a capability that is not provided within the context of the requesting association, such as a request for Third Party Clear Call capability in a Notification Association.

The ECS uses the abort mechanism to terminate Third Party Call Control associations that must halt because ECS processing encounters conditions such as invalid Class of Restriction.

Endpoint Rejects FIE

An ASAI endpoint may use the Reject component to reject an FIE that is badly structured or violates protocol. An ASAI endpoint may use[18] the Reject component when it is able to determine the message type, CRV, and the presence of an FIE within a message, but the contents of the FIE violate protocol for some reason. The Reject component contains a problem code that specifies the protocol violation within the FIE. ASAI permits the rejecting endpoint to send the Reject component in either:

- A FACility message if the rejecting endpoint permits the requesting endpoint to continue the association and retry
- A RELease COMplete message if the rejecting endpoint terminates the association when an FIE protocol error occurs

Of these options, the ECS always sends any Reject component in a RELease COMplete message and terminates any association where an FIE protocol violation occurs. The ECS does not permit the adjunct to retry within the same association after an FIE protocol violation. For coding, see ‘‘Call Control: Endpoint

18. It may also abort or return an empty RELease COMplete message.

Rejects an Invalid/Protocol Violation FIE — Terminates Association'' on page 5-68 of Chapter 5, "Byte Level Messages." For domain control coding, see ''Domain Control: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association'' on page 5-96 of Chapter 5, "Byte Level Messages."

The ECS does not attempt to retry during any association where an adjunct rejects an FIE that the ECS has sent. If the ECS receives a Reject component in a FACility message, the ECS immediately replies with a RELease COMplete that terminates that association.

ASAI and BRI Parser Interactions

Since ASAI is provided on a BRI interface, the ASAI adheres to certain protocol procedures for BRI. The BRI Parsing subsystem implements the following Q.931 procedures (shown in order of precedence):

- The ECS ignores any incoming message less than three bytes or greater than 260 bytes.
- The ECS checks for a valid protocol discriminator.
  - — The REGister, FACility, RELease COMplete, RESTart, RESTart ACKnowledge, and STATUS messages must carry the BRI protocol discriminator (0x08).

  The ECS ignores any message with an improper protocol discriminator.
- The ECS sends a RELease COMplete message with a Cause IE having value 97 (Message Type Invalid or Not Implemented) if a message other than REGister, FACility, RELease COMplete, STATUS, RESTart, or RESTart ACKnowledge is received.[19] For more information regarding BRI, see the *ISDN Basic Rate Interface (BRI) Specification*, 801-802-100.
- The ECS ignores STATUS messages.
- The ECS then checks to insure that the CRV is a permitted length. The CRV must be either the Global CRV, or its length must be the length permitted on the ASAI over which the message arrived. For each ASAI link, the customer administers the length of the ASAI CRV values to be either 1 or 2 bytes. The parameter settings may be different for different ASAI interfaces. When the CRV length is administered as 2 bytes:
  - — An adjunct is permitted to send messages containing a CRV with length 1 or 2.
  - — The ECS always sends 2-byte CRVs (even though the CRV value may fit into a single byte).

  The ECS ignores any message containing a CRV with a value that is either: 1) not permitted with the administered length, or 2) zero length (for example, the Global Call Reference Value).

19. BRI allows the network the option of sending STATUS or RELease COMplete in this situation. The ECS ASAI sends RELease COMplete.

- The ECS checks to make sure that in-use and available CRVs are used appropriately in ASAI messages. If the ECS receives:

  1. A REGister message containing a CRV that is already in use on the ASAI for another association, then the ECS responds with a RELease COMplete message containing Cause CS0/81, “Invalid CRV.” This message does not contain an FIE.
  2. A FACility message containing an idle CRV, then the ECS responds with a RELease COMplete message containing Cause CS0/81, “Invalid CRV.” This message does not contain an FIE.

- The ECS sends a RELease COMplete message with a Cause IE having value 96 (Mandatory IE missing) if a mandatory IE is omitted. For more information, see *ISDN Basic Rate Interface (BRI) Specification,* 801-802-100. This message may not contain an FIE. Specifically, this ECS takes this action if it receives a REGister or FACility message without the mandatory Facility Information Element. [20]
- If an incoming REGister message contains an FIE, but the FIE does not contain an Invoke component, the ECS responds with a RELease COMplete message with a Cause IE having value 100, “Invalid IE contents.” This message does not contain an FIE.
- If the ECS receives a REGister message (for example, a request to begin a new association) when it is in an overload state, the ECS denies the request as described in “Endpoint Denies A Request” earlier in this section.
- Any unrecognized or unexpected information element in a message is ignored; processing continues with recognized information elements. For more information, see the *ISDN Basic Rate Interface (BRI) Specification,* 801-802-100.
- Any unexpected or unrecognized information elements contained (as ASAI parameters) within an FIE are also ignored; processing continues with recognized information elements.
- When a mandatory parameter is not present within an FIE, the ECS sends a FACility or RELease COMplete message containing a Return Error component with a Cause IE having value 96 (Mandatory IE Missing). The ECS sends a FACility message in exactly the same situations as it would send a FACility message for denial; otherwise the ECS sends a RELease COMplete message.
- The ECS does not send any response to a RELease COMplete with an unrecognized (inactive) CRV.
- If the request contains a FACility IE with an operation value that is outside of the ASAI subset that the ECS supports, the ECS responds with CS0/111, “Protocol error.”

20. Note: BRI allows the network the option of sending STATUS or terminating the ASAI association in this situation. The ECS implementation of ASAI terminates the association.

- If a field within an IE contains an invalid, reserved, or unrecognized code point, the ECS returns CS0/100, “Invalid IE Contents.”

Link Management and Maintenance Procedures

Maintenance procedures use additional messages beyond the REGister, FACility, and RELease COMplete messages that carry capability invocations discussed previously. These ISDN messages (RESTart, RESTart ACKnowledge) provide additional procedures to keep the ECS and adjunct synchronized.

NOTE:
Adjunct support for heartbeat and restart procedures is mandatory for operation with DEFINITY ECS.

Maintenance Heartbeat Procedure

One ASAI endpoint queries the other to see if the ECS is processing layer 3 ASAI messages.

1. The initiating endpoint sends a REGister message containing an invoke FIE with:

   Operation Value = Heartbeat

   For coding, see ‘‘Heartbeat’’ on page 5-188 of Chapter 5, “Byte Level Messages.”

2. The receiving endpoint replies with a RELease COMplete message containing a return result FIE. The invoke-id in the return result has the same value as the invoke-id in the heartbeat.

   For coding, see ‘‘Response to Heartbeat’’ on page 5-189 of Chapter 5, “Byte Level Messages.”

Platform developers should be aware of another important use for Heartbeat messages. The layer 2 transport protocol underlying ASAI (ISDN LAPD) uses several timers to detect that a packet has not been received and to attempt retransmittal. The result is that the time it takes layer 2 to report a link failure to layer 3 may be too long for certain applications. In an environment where, for example, link failure needs to be detected in three seconds, the adjunct platform should send a Heartbeat message every three seconds to ensure that it can detect a link failure within the desired time.

ASAI Restart Procedure

Both the ECS and adjunct must adhere to the Restart Procedure on an ASAI link.[21] The Restart Procedure insures that if one ASAI endpoint detects a layer 2 drop (and therefore clears all its CRVs for the interface), ASAI messaging cannot continue on that interface without the other endpoint clearing its CRVs, also.[22] Both ECS and adjunct begin the Restart Procedure when:

- An ASAI has been established at layer 2.
- An ASAI layer 2 link has been re-established after a link failure. [23]
- An ECS or adjunct maintenance subsystem determines a need to restart (and resynchronize) the ASAI.

Platform developers must be aware that it is possible for an ASAI link to drop without the ECS going down (for example, the cable is unplugged). When the ECS detects a link drop, it clears all its ASAI data structures. Then, if the link has returned to service, the ECS begins the Restart Procedure. It is possible to unplug a link and return it to service before the ECS has cleared the ASAI data structures. This can occur, for example, if there are thousands of domain (station) associations on the link. The ECS divides the cleanup of the ASAI data structure into a number of subtasks so that critical call processing can continue as it cleans up the ASAI data structures. Thus, adjunct platforms are advised to wait for the initiating RESTart message from the ECS. The ECS ignores any RESTart messages that it receives from the adjunct during a cleanup period.

The Restart Procedure also incorporates a method for adjuncts to select a particular version of the ASAI protocol they wish to run on the link. Presently, there are three versions, V1, V2 and V3. If the adjunct does not include any version specification options, V1 is the default version that the ECS supplies.

NOTE:
Version selection is per link. Version selection facilitates the use of older ASAI applications when switches are upgraded with newer ASAI features.

Sending RESTart

An endpoint sends the RESTart message to return all Q.931/Q.932 resources (for example, Call Reference Values on an ASAI) to an idle state. The ECS encodes the RESTart message to restart the entire ASAI.

- The RESTART message must contain the Global CRV.

21. In *ISDN Basic Rate Interface (BRI) Specification*, the procedure is optional in the user-to-network direction.
22. It is possible, on a BRI interface, for one endpoint to detect an LAPD drop and re-establishment while the other endpoint does not detect the drop.
23. Note: T309 procedures are not provided for ASAI CRVs.

- The RESTART message MUST NOT contain the optional Channel Identifier IE. The absence of this IE indicates that the interface is to be restarted. Since a Channel Identification IE is present in a BRI RESTart message only to restart a specific B-channel [see the *ISDN Basic Rate Interface (BRI) Specification*], this information element is not applicable to the ASAI. The ECS ignores the Channel Identification IE if it is present.
- The RESTART message MUST contain the Restart Indicator IE with the class set to *all interfaces* [see the *ISDN Basic Rate Interface (BRI) Specification*].
- Starting with G3V3, switches include Version IEs for each supported ASAI Version in RESTART messages.

Upon transmitting RESTart, the sender initiates layer 3 Timer T316 (120 seconds) and waits for a RESTart ACKnowledge message. Receipt of REST ACK cancels timer T316. If a REST ACK is not received before the expiry of timer T316, the sending endpoint may retransmit the REST message *once*. If there is no response to a second transmission, the sending endpoint must take appropriate maintenance and recovery actions. The sending endpoint may not make or accept ASAI requests on the interface until recovery action is taken.

The originator of a REST message MAY NOT establish any ASAI associations over the interface while receipt of REST ACK is pending.

Receiving RESTart

An ASAI endpoint that receives a RESTart message for an ASAI frees all CRVs for that interface, terminates the corresponding ASAI associations, and then returns RESTart ACKnowledge.

- If the ECS receives a RESTART message containing a non-global CRV, the ECS responds with a RELease COMplete message containing the received CRV and a Cause IE with cause value 81, “Invalid CRV.”
- If the REST message does not contain a Restart Indicator IE, then the ECS ignores the RESTART message.
- If the Restart Indicator IE does not specify **all interfaces** for class, the ECS ignores the message.
- If the contents of the REST message are correct, the ECS terminates all ASAI associations on the interface and then sends a REST ACK containing the CRV (always the Global CRV) and Restart Indicator IE (always single interface class) that it received.

Starting with G3V3, the adjunct may select the ASAI version that is to be run on the link. The ECS RESTART message contains a Version IE for all versions that the ECS supports. The adjunct may include one of these IEs in the RESTart ACKnowledgement message to select a version. If no Version IE is included in the REST ACK, the ECS defaults to V1. If the adjunct responds with an invalid or unsupported version, the ECS again ignores the REST ACK and sends a RESTART upon expiry of Timer T316 with a list of available versions. The

expected response is a REST ACK either with no version IEs or with a version IE indicating one of the available versions. Any other response causes the ECS to log a maintenance error.

Suspend/Resume ECS Alarming on ASAI Link

These procedures let an adjunct:

- Suspend any ECS alarming in effect for an ASAI link when the adjunct takes the link out of service for scheduled maintenance or for graceful termination of the link for some other reason.
- Resume ECS alarms when the adjunct brings the link back into service.

Unnecessary alarms in these conditions may increase servicing and maintenance costs to the customer.

To suspend or resume alarms, the following message sequence is used:

1. The adjunct sends a Management Information Message (MIM) message containing a Management Information Element (MIE) with Link Alarm Status Change Request to Suspend or Resume Alarms. Upon transmission, the adjunct starts timer TM100 with a value of four seconds.
2. The ECS responds with a MIM containing a return result MIE.
3. If the timer TM100 expires without a response, the adjunct may retransmit the request and again set the TM100 timer. If there is no response to the transmission, the adjunct may continue to retry, but should use the timer TM200 with a value of at least 120 seconds.

   If the adjunct sent an incorrectly encoded MIM, the ECS responds with a REJECT MIM.

For coding, see ‘‘Suspend/Resume Alarming for ASAI Interface’’ on page 5-190 of Chapter 5, “Byte Level Messages.”

Application Timers

Timing of ASAI Responses (ACKs/NAKs)

The ASAI specification does not contain timers for situations where message loss results in an application waiting infinitely for the message from the other endpoint. Rather, ASAI applications are required to set timers when they make requests. If the timer expires and the application has not received a response, the application must abort the association, and it may retry if it desires.

Certain adjunct programming environments may take responsibility for such timers. The application programmers must ascertain whether they must explicitly include such timers in their application programs.

Any adjunct application level timers used in this fashion have a recommended minimum value of 10 seconds.

Initial Messages on an ASAI Link

Adjunct application developers must be aware that ASAI links have various flow control and hyperactivity thresholds. When an ASAI link drops and is re-established, the adjunct association may send several layer 3 queries (ASAI messages) to re-synchronize its internal data with current ECS data. When several queries are required to do this, the adjunct should pace the sending of the queries. To avoid an inadvertent triggering of any administered link alarms, the recommended maximum rate is five queries per second.

Message Descriptions

3

This chapter details the ASAI messages introduced in Chapter 1.

Message Overview

ASAI is based on standard protocols, including CCITT Recommendations Q.931 and Q.932, and the *ISDN Basic Rate (BRI) Interface Specification,* 803-802-100. These protocols contain such information as allowable message types and required format, which ASAI follows.

Table 3-1 lists the ASAI message types described in this chapter.

Table 3-1. Messages for ASAI

ASAI Message
FACility
Management Information Message (MIM)
REGister
RELease COMplete
RESTart
RESTart ACKnowledge
STATUS

Each explanation includes:

- A brief description of the message direction and use
- A table listing the information elements contained in the message. For each information element, the table indicates:
  - — The direction in which the information element may be sent; for example, adjunct-to-ECS, ECS-to-adjunct, or both. (All information elements for these messages are “both.”)
  - — Whether the information element is mandatory (M) or optional (O).
  - — The allowed length, in bytes. A question mark (?) identifies when a length is restricted only by the maximum length of an ASAI message.

The information elements are listed in their order of appearance in the message. The relative order of information elements is the same for all message types. See Chapter 4 for a full description of the information elements.

FACility Message

The FACility message is sent during an ASAI association to invoke an operation or convey information from one endpoint to another as part of the message exchange for that association.

Message type: FACility
Direction: both

Table 3-2. FACility Message Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Locking Shift to Code Set 6	both	M	1
Facility	both	M	8-?

Management Information Message (MIM)

This message is currently used in ASAI only to enable and disable the ECS alarming of an ASAI link. This document contains only the necessary subset of the MIM and its related procedures. See the *ISDN Basic Rate Interface (BRI) Specification* for a complete discussion. The MIM carries only the global CRV on an ASAI link.

Message type: Management Information Message (MIM)
Direction: both

Table 3-3. Management Information Message (MIM) Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	2
Locking Shift to Code Set 6	both	M	1
Management Information Element	both	M	6-?

REGister Message

The REGister message carries the initial capability invocation that initiates an ASAI association and assigns a call reference value that the endpoint uses for the duration of that ASAI association.

Message type: REGister
Direction: both

Table 3-4. REGister Message Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Locking Shift to Code Set 6	both	M	1
Facility	both	M	8-?

RELease COMplete Message

The RELease COMplete message ends an ASAI association. The sending and receiving endpoints release the CRV.

Message type: RELease COMplete
Direction: both

Table 3-5. RELease COMplete Message Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Cause	both	O	4-22
Locking Shift to Code Set 6	both	O	1
Facility	both	O	8-?

RESTart Message

The RESTart message may be sent from either side of the ASAI interface to request the recipient to restart the ASAI interface. The sending endpoint clears all ASAI associations in progress on the interface and expects the receiving endpoint to do the same and then acknowledge the request.

NOTE:
Clearing a Call Control association does not affect the call.

Although in the *ISDN Basic Rate Interface (BRI) Specification* RESTart is an optional message in the terminal-to-ECS direction, adjunct support of this message is mandatory in that direction on an ASAI.

NOTE:
The Call Reference Value is encoded as the Global CRV, and the Restart indicator is coded to “all interfaces.”

Message type: RESTart
Direction: both

Table 3-6. RESTart Message Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Restart Indicator	both	M	3
Locking Shift to Code Set 6	both	0	1
Version IE	both	O	3-7 (multiple IEs allowed)

RESTart ACKnowledge Message

The RESTart ACKnowledge message acknowledges the receipt of the RESTart message and signifies that the requested restart of the ASAI interface is complete.

Although in the *ISDN Basic Rate Interface (BRI) Specification* this is an optional message in the terminal-to-ECS direction, adjunct support of this message is mandatory in that direction on an ASAI.

NOTE:
The Call Reference Value is encoded as the Global CRV, and the Restart indicator is coded to “all interfaces.”

Message type: RESTart ACKnowledge
Direction: both

Table 3-7. RESTart ACKnowledge Message Content

Information Element	Direction	Type	Length
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Restart Indicator	both	M	3
Locking Shift to Code Set 6	both	0	1
Version IE	both	O	3-7

Status Message

The STATUS message is included here for completeness. Certain BRI (hence ASAI) implementations may send STATUS in response to protocol errors. Although the ECS does not transmit STATUS messages, the ECS recognizes incoming STATUS messages as valid messages and does not treat them as ASAI protocol errors. The ECS ignores incoming STATUS messages; there is no response.

Message type: STATUS
Direction: both

Table 3-8. STATUS Message Content

Information Element	Reference	Direction	Type
Protocol discriminator	both	M	1
Call reference	both	M	2-3
Message type	both	M	1
Call State	both	M	3

Information Elements

4

This chapter describes the element structure (information element) of the layer 3 protocol messages.

Every message contains a grouping of the following information elements (IEs):

- The BRI protocol discriminator
- Call Reference Value (CRV)
- Message type
- Mandatory information elements, as required
- Additional information elements, when required

The first three elements are common to all the messages and must always be present, while the last two elements are specific to each message type. Figure 4-1 illustrates this message structure.

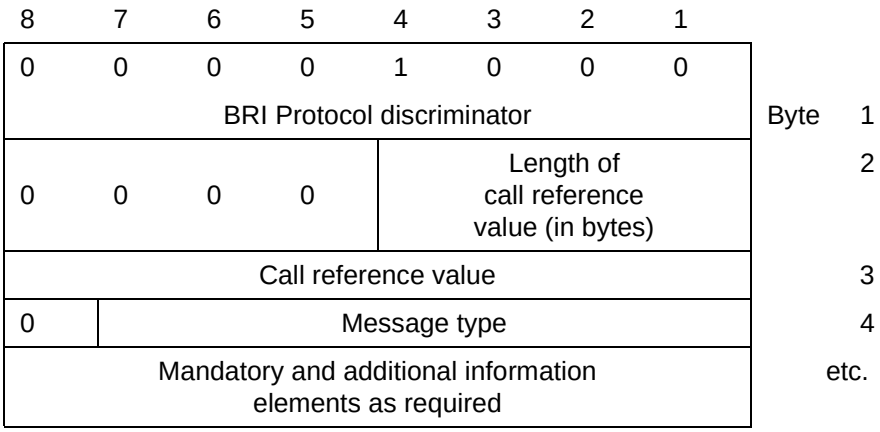


Figure 4-1. General Message Organization Example

Unless specified otherwise, a particular information element may be present only once in a given message.

The term **default** means the default value defined is used in the absence of any assignment or negotiation of alternate values.

Because an information element may occur several times in a message, the ordering of the instances of the information element is important. Chapter 5, ‘‘Byte Level Messages’’ explicitly notes where a particular ordering of a repeated information element is required. In all other cases, when an information element occurs more than once in a message, the receiving endpoint must be able to accept those instances in any order.

Every message carries the information elements in a byte structure. Within each byte, the bit appointed as bit 1 is transmitted first, followed by bits 2, 3, 4, and so on. These bits are assigned the information element values. Their order of appearance varies with each message group. Byte 1 is transmitted first also. When a field such as the call reference value extends over more than one byte, the order of the bit values decreases as the byte number increases. The least significant bit of the field is represented by the lowest-numbered bit of the highest-numbered byte of that field.

The ISDN Basic Rate Interface (BRI) Specification limits the maximum length of a BRI message, hence an ASAI message, to 260 bytes. In all information elements, including the Facility IE, the “length of IE contents” (which follows the IE identifier) is a single byte. Thus, the maximum binary value that the “length of IE contents” field may hold is 255. However, to limit the message to 260 bytes, the maximum value that the “length of IE contents” field can take on is less than the maximum binary value of 255. Consider a typical ASAI message containing a Facility IE. The

value of the “length of IE contents” for the Facility IE that gives a 260-byte message is shown in the following table:

Message Component	1-byte CRV	2-byte CRV
BRI Protocol Discriminator	1 byte	1 byte
length of CRV	1 byte	1byte
CRV	1 byte	2 bytes
message type	1 byte	1 byte
locking shift to code set 6	1 byte	1 byte
facility IE identifier	1 byte	1 byte
length of Facility IE	1 byte	1 byte
remaining bytes	253 bytes	252 bytes
Total	260 bytes	260 bytes

Protocol Discriminator

The BRI protocol discriminator is 0x08, “0 0 0 0 1 0 0 0.”

Call Reference Value (CRV)

ISDN CRVs are unique at an ASAI interface and are present in every ASAI message that passes across the interface. ASAI endpoints use the CRV to associate a sequence of messages that comprise an ASAI association.

The originating ASAI endpoint assigns a CRV for the ASAI association. These values are unique to the origination side within a specific D-channel layer 2 logical link connection. The CRV is fixed for the duration of the ASAI association. Once the association ends, the originating endpoint may assign the CRV to a later association.

Figure 4-2 shows the structure and encoding of the ISDN CRV.

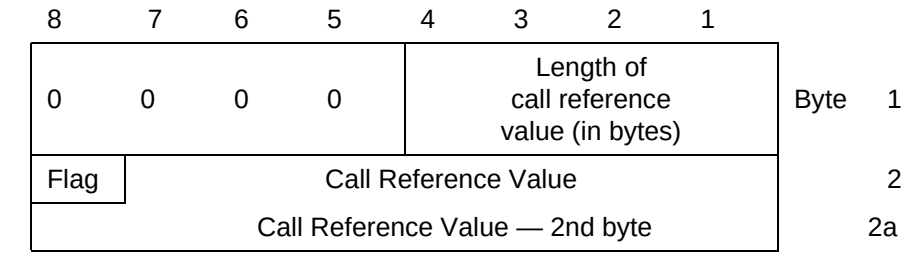


Figure 4-2. ISDN Call Reference Value

The CRV has three fields:

- The length of the CRV
- The call reference value
- The call reference flag

The CRV length is the number of bytes that **follow** the byte containing the length, so the length may take on the values “0 0 0 1” (one byte follows), or “0 0 1 0” (two bytes follow). Each ASAI interface is administered individually for one- or two-byte CRV lengths.

The Global Call Reference value is a CRV with length 1 or 2 (however administered for the ASAI link) and value zero.

If the ECS receives a CRV on an ASAI interface where the CRV is not the global CRV and has a length different from that permitted on the ASAI interface, the CRV is considered invalid and the ECS ignores the message. When the CRV length is administered as two bytes for an ASAI interface:

- An adjunct is permitted to send messages containing a CRV with length 1 or 2 on that interface.
- The ECS always sends two-byte CRVs (even though the CRV value may possibly fit into a single byte).

Certain ASAI messages (RESTart, RESTart Acknowledge, and MIM) always contain the Global Call Reference value.

The call reference flag accepts the values “0” or “1.” An originating ASAI endpoint always sets the flag to “0;” the destination ASAI endpoint always sets the flag to “1.” Furthermore, the interface receiving a CRV first complements the flag before using the CRV internally. For example:

> If the adjunct originates an ASAI association with a CRV of 64, the adjunct would code byte 2 of the CRV as “0 1 0 0 0 0 0 0.” Messages from the ECS for this ASAI association would have a CRV with byte 2 coded as “1 1 0 0 0 0 0 0.”

Thus, both endpoints may assign an identical value and the CRV flag prevents a collision in their use.

Although permitted, it is recommended that adjuncts **do not** initiate associations with CRVs in the range 1 to 32, inclusive. Rather, an adjunct might start assigning CRVs with the highest possible values and work downwards, or begin assigning CRVs at 33 and work upwards.

Message Type Information Element

The message type identifies the function of the message being sent. It is the third part of every message and may be one or two bytes long. The MIM is the only multi-byte message type used by the ASAI. Table 4-1 shows the single-byte message-type codings. Table 4-2 shows the two-byte message type codings.

Table 4-1. Single-Byte ASAI Message Types

8	7	6	5	4	3	2	1		
0		Message type						Byte 1	
0	1	0	1	1	0	1	0		RELease COMplete
0	1	1	0	0	0	1	0		FACility
0	1	1	0	0	1	0	0		REGister
0	1	0	0	0	1	1	0		RESTart
0	1	0	0	1	1	1	0		RESTart ACKnowledge
0	1	1	1	1	1	0	1		STATUS

Table 4-2. Two-Byte ASAI Message Types

8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	
0 0 0 0 0 0 0 0	1 - - - - - - -	Network Specific Message
	1 1 1 0 1 1 1	Management Information (MIM)

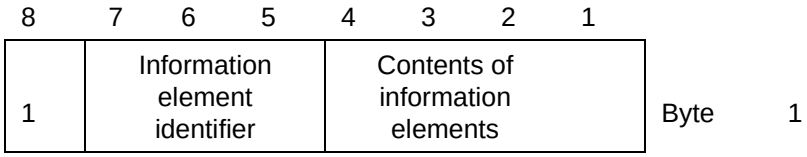
Codeset Information Elements

Coding Rules

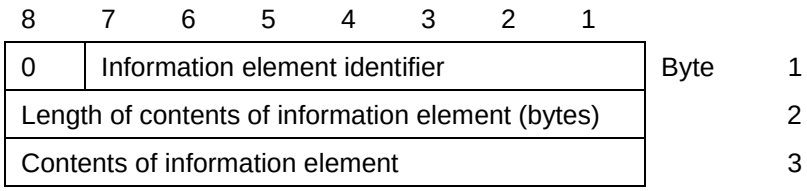
Information element coding follows the rules below.

Two categories of information elements are defined:

- Single-byte information elements (see Figure 4-3 [a] Single-byte information element format)
- Variable length information elements (see Figure 4-3 [b] Variable length information element format)



(a) Single-byte information element format



(b) Variable length information element format

Figure 4-3. Formats of Information Elements

Table 4-3 summarizes the coding of the information element identifier bits.

There is an order of appearance for information elements within a message or enveloping FIE. All IEs from any given code set are grouped together. Within the code set grouping, the code values of the information element identifier determine the order of appearance of the variable length information elements within a message. These IEs appear in ascending numerical order. Thus, within a message, the information elements from a given code set (for example, 0 and 6) must be presented in the order of increasing byte code identifier. This allows the receiving equipment to detect the presence or absence of a particular information element without scanning through an entire message.

Where the description of information elements in this specification contains spare bits, these bits are indicated as being set to “0.” To allow compatibility with future implementations, messages should not be rejected simply because a spare bit is set to “1.”

The second byte of a variable length information element indicates the total length of the contents of the remainder of that information element. It is the binary coding of the number of remaining bytes, with bit 1 as the least significant bit (2°×).

The following set of terms is used in the figures depicting the structure of variable length information elements:

a. The first digit in the byte number column to the right of the figure identifies one byte or a group of bytes.

b. Each byte group is a self-contained entity. The internal structure of a byte group may be defined in alternative ways.

c. A byte group is formed by using some extension mechanism. The preferred extension mechanism is to extend a byte (N) through the next byte(s) (Na, Nb, etc.) by using bit 8 in each byte as an extension bit. The bit value “0” indicates that the byte continues through the next byte. The bit value “1” indicates that this byte is the last byte. If byte (Nb) is present, the preceding bytes (N and Na) must also be present.

   In the format descriptions for the information elements, bit 8 is marked “0/1 ext” if another byte follows. Bit 8 is marked “1 ext” if this is the last byte in the extension domain.

d. In addition to the extension mechanism defined above, a byte (N) may be extended through the next byte(s) (N.1, N.2, and so on) by indications in bits 1 (of byte N).

e. The mechanisms in “c” and “d” may be combined.

f. Optional bytes are marked with asterisks (*).

Table 4-3. Information Element Identifier Coding

8	7	6	5	4	3	2	1	
1		:	:	-	-	-	-	**Single-byte information elements:** 0
0	:	:	:	:	:	:	:	**Variable length information elements:** **Codeset 0**
	0	0	0	1	0	0	0	Cause
	0	0	0	1	1	0	0	Connected number
	0	0	1	0	0	0	0	Call identity
	0	0	1	0	1	0	0	Call state
	0	0	1	1	1	1	0	Progress Indicator
	0	1	0	1	0	0	1	Date/time
	1	1	0	1	1	0	0	Calling party number
	1	1	1	0	0	0	0	Called party number
	1	1	1	0	1	0	0	Redirecting number
	1	1	1	0	1	1	0	Redirection number
	1	1	1	1	0	0	1	Restart indicator
	1	1	1	1	1	1	0	User-User Information
								Codeset 6
	0	0	0	0	0	0	1	Originating Line Information
	0	0	0	0	0	1	0	User Entered code
	0	0	0	0	0	1	1	Resource Status
	0	0	0	1	0	1	0	Trunk Identification
	0	0	0	1	0	1	1	Trunk group/trunk status
	0	0	1	0	1	1	1	Old Party Identifier
	0	0	1	1	0	1	1	Version
	0	0	1	1	1	0	0	Facility
	1	0	0	0	1	0	0	Party id
	1	0	0	0	1	1	0	Counter
	1	0	0	0	1	1	1	Specific event
	1	0	0	1	0	0	0	Feature
	1	0	0	1	0	0	1	Domain
	1	0	0	1	0	1	0	Conf/Trans options
	1	0	0	1	0	1	1	Call options
	1	0	0	1	1	0	1	Item
	1	0	0	1	1	1	0	Service Circuit
	1	0	0	1	1	1	1	Status
	1	0	1	0	0	0	1	Resource identifier
	1	0	1	0	0	1	0	Data Item
	1	0	1	0	0	1	1	Data Bit Map
	1	0	1	0	1	1	0	Generic Billing
	1	1	1	1	0	1	0	Management Information (MIE)
	1	1	1	1	0	1	1	Lookahead Interflow

One common value in the single-byte format is employed in each code set to shift from one code set to another. The contents of this shift item identify the code set to be used for the following information element(s). The code set in use at any given time is referred to as the **active code set**. By convention, code set 0 is the initially active code set. Codeset 6 is used for Lucent Technologies-specific supplementary service information elements, including ASAI information elements. The FIE transports ASAI information across the ECS/adjunct interface. The ECS ASAI does not use codesets 1, 2, 3, 4, 5, or 7.

The ASAI supports a locking shift to code set 6. The locking shift must follow the last information element in code set 0. One or more variable length information elements from code set 6 follow the locking shift.

Locking Shift Procedure

The locking shift procedure uses an information element to indicate the new active code set. For example:

Code set 0 is active at the start of message content analysis. If a locking shift to code set 6 is encountered, information elements in the message are interpreted according to the information element identifiers assigned in code set 6.

NOTE:
The FIE (which is a codeset 6 IE) contains a sequence of IEs within it. Interpretation of the IEs inside of the FIE begins in codeset 0 and shifts to codeset 6 if a lock shift is encountered within the FIE.

The locking shift to code set 6 **must** be present in a message when information elements from code set 6 are included in the message.

Figure 4-4 shows the single-byte information for the locking shift.

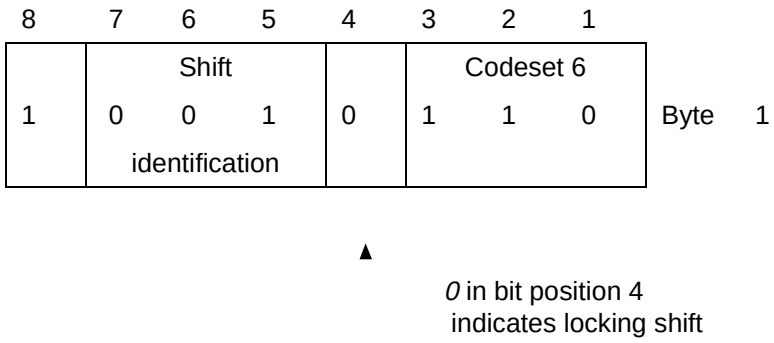


Figure 4-4. Locking Shift to Codeset 6

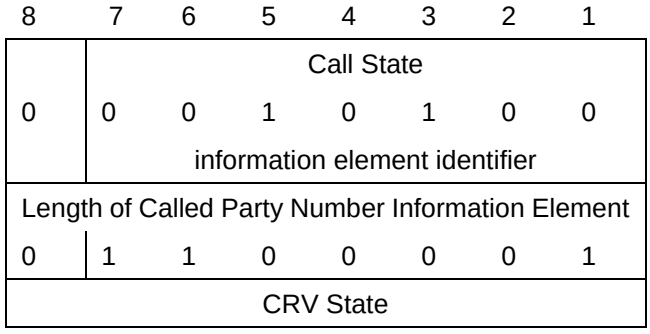
Codeset 0 Information Elements

The codeset 0 information elements are CCITT-approved and are incorporated in the CCITT specification. The ASAI codeset 0 IEs drawn from this specification are as follows:

Call State	Date/Time
Called Party Number	Progress Indicator
Calling Party Number	Redirecting Number
Cause	Redirection Number
Call Identity	Restart Indicator
Connected Number	User to User Information

Call State

The Call State information element shown in Figure 4-5 displays the state of a CRV and is present in the BRI STATUS message. Since certain ASAI endpoints might transmit a BRI STATUS message to the ECS when the endpoint encounters a protocol error, the Call State information element is included in this document. The ECS does not support its use in any ASAI capability messaging.



Value	Call State
0 0 0 0 0 0 0 0	Null — CRV not in use
0 0 0 0 1 0 1 0	Active — CRV in use

Figure 4-5. Call State Information Element

Called Party Number

The Called Party Number IE shown in Figure 4-6 identifies the destination of a call.

8	7	6	5	4	3	2	1		
	Called Party Number								
0	1	1	1	0	0	0	0	Byte	1
	information element identifier								
Length of Called Party Number Information Element									2
1									3
Ext	Type of Address			Numbering Plan					
0	Address Digits								4
Spare									etc.

Figure 4-6. Called Party Number Information Element

Extension Bit

0: description extends into next byte

1: last byte of the description element

Type of Address

Bits		
7	6	5
0	0	0 unknown
0	0	1 international
0	1	0 national
1	0	0 subscriber

Numbering Plan

Bits			
4	3	2	1
0	0	0	0 unknown
0	0	0	1 ISDN/telephony numbering plan
0	0	1	0 reserved
1	0	0	1 private numbering plan

Address Digits

Bits							Address Digit
7	6	5	4	3	2	1	value
0	1	1	0	0	0	0	0
0	1	1	0	0	0	1	1
0	1	1	0	0	1	0	2
0	1	1	0	0	1	1	3
0	1	1	0	1	0	0	4
0	1	1	0	1	0	1	5
0	1	1	0	1	1	0	6
0	1	1	0	1	1	1	7
0	1	1	1	0	0	0	8
0	1	1	1	0	0	1	9
0	1	0	0	0	1	1	#
0	1	0	1	0	1	0	*

The ECS permits a maximum of 31 address digits and sends/receives only those ASCII characters shown above. Any adjunct must:

a. Accept all characters shown.

b. Not send characters other than those shown. Doing so results in the ECS denying (return error) the request.

There may be instances of event reports where an address for the called party is not available to the ECS. In such cases, the Called Party Number IE is present in the event reports where ASAI requires it as a mandatory event report item. However, in such a case, the Address field of the IE is not present; the length of the resulting IE is one (1); the Type of Address and Numbering Plan fields are present. Any adjunct interfacing to the ECS should be prepared to receive the Called Party Number IE in this format.

The adjunct should also be prepared to accept the default called party number values “#####,” or “*****.”

Calling Party Number

The Calling Party Number IE shown in Figure 4-7 identifies the origin of a call.

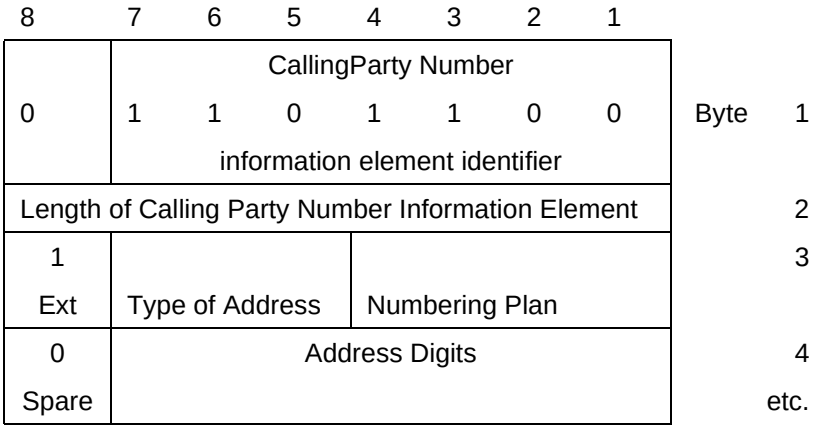


Figure 4-7. Calling Party Number

Extension Bit

0: description extends into next byte

1: last byte of the description element

Type of Address

Bits		
7	6	5
0	0	0 unknown
0	0	1 international
0	1	0 national
1	0	0 subscribe

The ECS does not include the optional PRI byte for the presentation indicator and screen indicator

Numbering Plan

Bits			
4	3	2	1
0	0	0	0 unknown
0	0	0	1 ISDN/telephony numbering plan
0	0	1	0 reserved
1	0	0	1 private numbering plan

Address Digits

Bits 7	6	5	4	3	2	1	Address Digit value
0	1	1	0	0	0	0	0
0	1	1	0	0	0	1	1
0	1	1	0	0	1	0	2
0	1	1	0	0	1	1	3
0	1	1	0	1	0	0	4
0	1	1	0	1	0	1	5
0	1	1	0	1	1	0	6
0	1	1	0	1	1	1	7
0	1	1	1	0	0	0	8
0	1	1	1	0	0	1	9
0	1	0	0	0	1	1	#
0	1	0	1	0	1	0	*

The ECS permits a maximum of 15 address digits and sends/receives only those ASCII characters shown above. Any adjunct must:

a. Accept all characters shown.

b. Not send characters other than those shown. Doing so results in the ECS denying (return error) the request.

There may be instances of event reports where an address for the calling party is not available to the ECS. In such cases, the Calling Party Number IE is present in the event reports where ASAI requires it as a mandatory event report item. However, in such a case, the Address field is not present; the length of the resulting IE is one (1); the Type of Address and Numbering Plan are present. Any adjunct interfacing to the ECS should be prepared to receive the Calling Party Number IE in this format.

The adjunct should also be prepared to accept the default calling party number values “#####,” or “*****.”

Cause

The Cause IE shown in Figure 4-8 gives the reason for generating certain messages to provide diagnostic information (see Table 4-4).

8	7	6	5	4	3	2	1		
	Cause								
0	0	0	0	1	0	0	0	Byte	1
	information element identifier								
Length of cause information element									2
1	Coding		0	Location					3
Ext	Standard		Spare						
1	Cause Value								4
Ext	(Class)			(value in class)					
Diagnostics omitted - Note 1									5, etc.

NOTE:
PRI has an optional diagnostic field; ASAI does not use this field.

Figure 4-8. Cause Information Element

Coding Standard

Bits 7	6	
0	0	CCITT standard cause value
1	1	ASAI specific cause value

ASAI Location Code Points

Bits 4	3	2	1	
0	0	0	0	user (for example, adjunct)
0	0	0	1	private network serving local user (PBX switch)

ASAI Cause Value Encodings

The cause value is divided into two fields, a class and a value within the class. Diagnostic information is optional.

ASAI CLASS Field Encodings

Class(0 0 0)	normal event
Class(0 0 1)	normal event
Class(0 1 0)	resource unavailable
Class(0 1 1)	service or option not available
Class(1 0 0)	service or option not implemented
Class(1 0 1)	invalid message (parameter out of range)
Class(1 1 0)	protocol error (unknown message)

Table 4-4. ASAI Coding Standard Zero Causes Returned by the ECS

Coding Standard Zero (0 0) Cause Values		
Class Value	**#**	**Cause**
7 6 5 4 3 2 1		
0 0 0 0 0 0 1	1	Unassigned Number
0 0 1 0 0 0 0	16	Normal Clearing
0 0 1 0 0 0 1	17	User Busy
0 0 1 0 0 1 0	18	No User Responding
0 0 1 0 1 0 1	21	Call Rejected
0 0 1 0 1 1 0	22	Number Changed/SIT — Vacant
0 0 1 1 1 0 0	28	Invalid Number/Domain/SIT — Ineffective other
0 0 1 1 1 0 1	29	Facility Rejected
0 0 1 1 1 1 1	31	Normal, Unspecified/SIT — unknown
0 1 0 0 0 1 0	34	No Circuit or Channel Available/SIT — No circuit
0 1 0 1 0 0 1	41	Temporary Failure
0 1 0 1 0 1 0	42	Switching Equipment Congestion/SIT — reorder
0 1 1 0 0 1 0	50	Requested Facility Not Subscribed/Provisioned
0 1 1 0 1 0 0	52	Outgoing Calls Barred
0 1 1 1 0 1 0	58	Bearer Capability not Currently Available
1 0 1 0 0 0 1	81	Invalid CRV
1 0 1 0 0 1 0	82	Identified Channel does not exist
1 0 1 1 0 0 0	88	Incompatible Destination
1 0 1 1 1 1 1	95	Invalid Message, Unspecified
1 1 0 0 0 0 0	96	Mandatory Information Element missing
1 1 0 0 0 0 1	97	Message Non-Existent/Not Implemented
1 1 0 0 0 1 0	98	Message Not Compatible with Call State
1 1 0 0 0 1 1	99	Information Element non-existent
1 1 0 0 1 0 0	100	Invalid Information Element Contents
1 1 0 0 1 1 0	102	Recovery on Timer Expiry
1 1 0 1 1 1 1	111	Protocol Error
1 1 1 1 1 1 1	127	Normal, Unspecified

Table 4-5. ASAI Coding Standard Three Causes Returned by the ECS

Coding Standard Three (1 1) Cause Values		
Class Value	**#**	**Cause**
7 6 5 4 3 2 1		
0 0 0 1 0 1 0	10	Agent on hook
0 0 0 1 0 1 1	11	Agent not member of split
0 0 0 1 1 0 0	12	Agent state inconsistent with request
0 0 0 1 1 0 1	13	Agent logged into maximum number of splits
0 0 0 1 1 1 0	14	Incorrect number of agent login digits
0 0 0 1 1 1 1	15	Agent not logged in
0 0 1 0 0 0 0	16	In Same State
0 0 1 0 0 0 1	17	Timed Answer
0 0 1 0 0 1 0	18	Voice Energy Answer
0 0 1 0 0 1 1	19	No Answer
0 0 1 0 1 0 0	20	Trunks not available
0 0 1 0 1 0 1	21	Classifiers not available
0 0 1 0 1 1 0	22	Queues full
0 0 1 0 1 1 1	23	Remains in Queue
0 0 1 1 0 0 0	24	Answering Machine
0 0 1 1 0 0 1	25	Call Forwarding (Redirection)
0 0 1 1 0 1 0	26	Cover, Principal Busy (Redirection)
0 0 1 1 0 1 1	27	Out of Service
0 0 1 1 1 0 0	28	Cover, Don't Answer (Redirection)
0 0 1 1 1 1 0	30	Redirected
0 0 1 1 1 1 1	31	Send All Calls (Redirection) Cover All Calls (Redirection) Go to Cover Active (Redirection) Adjunct Redirected Call (Redirection) Cover, All Call Appearances Used (Redirection)
0 1 0 0 1 1 0	38	Network out of order
0 1 0 1 0 0 0	40	Resources not available
0 1 0 1 0 0 1	41	Hunt group/split not administered correctly
0 1 0 1 0 1 0	42	Reorder/Denial
0 1 0 1 0 1 1	43	Permission Denied
0 1 0 1 1 1 0	46	Administration in progress
0 1 1 0 1 0 1	53	Feature Request Rejected
0 1 1 1 1 1 1	63	Service or Option Not available
1 0 0 1 1 1 1	79	Service or Option Not Implemented
1 0 1 0 0 0 0	80	Incompatible Options
1 0 1 0 1 1 0	86	Call with Requested Identity has been Terminated
1 0 1 0 1 1 1	87	Internal ECS Audit

In addition to the ASAI causes shown in Table 4-4 and Table 4-5, certain ASAI event reports may carry any cause that the ECS receives from a PRI network.

Call Identity

The ECS uses the Call Identity IE shown in Figure 4-9 to pass a call identifier to an adjunct that the adjunct might use later to start a new association for controlling the call. The ECS currently uses the root portion of an internal call identifier for the call identity; the current call identity is always 16 bits in length. The first byte is the high-order value and the second byte is the low-order value for the call_id (if the call_id is greater than 255). For call_ids less than 255, the first byte is 0, and the second byte contains the call_id value.

NOTE:
Future ASAI specifications with bridging may increase the size of the Call Identity IE beyond the present two bytes.

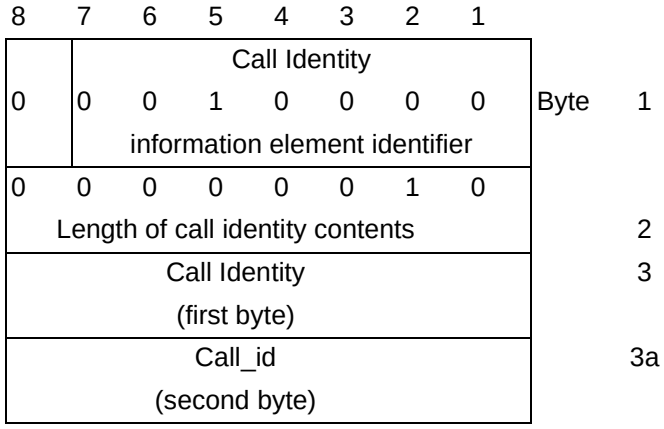


Figure 4-9. Call Identity Information Element

The ECS sends two-byte call identifiers. Since various DEFINITY ECS switches have different call capacities, host platforms should not make any assumptions about the range or sequencing of call identifiers. Hosts should be prepared to accept any values that can be encoded in the Call Identity bytes in any order.

Connected Number

The Connected Number IE shown in Figure 4-10 shows the address of the endpoint connected to a call. The connected number may be different from the calling party or called party number because the call may have interacted with a number of ECS features.

8	7	6	5	4	3	2	1		
	Connected Number								
0	0	0	0	1	1	0	0	Byte	1
	information element identifier								
Length of Connected Number Information Element									2
1									3
Ext	Type of Address			Numbering Plan					
0	Address								4
Spare									etc.

Figure 4-10. Connected Number Information Element

Extension Bit

0: description extends into next byte
1: last byte of the description element

Type of Address

Bits		
7	6	5
0	0	0 unknown
0	0	1 international
0	1	0 national
1	0	0 subscriber

Numbering Plan

Bits			
4	3	2	1
0	0	0	0 unknown
0	0	0	1 ISDN/telephony numbering plan
0	0	1	0 reserved
1	0	0	1 private numbering plan

Address Digits

Bits							Address Digit
7	6	5	4	3	2	1	value
0	1	1	0	0	0	0	0
0	1	1	0	0	0	1	1
0	1	1	0	0	1	0	2
0	1	1	0	0	1	1	3
0	1	1	0	1	0	0	4
0	1	1	0	1	0	1	5
0	1	1	0	1	1	0	6
0	1	1	0	1	1	1	7
0	1	1	1	0	0	0	8
0	1	1	1	0	0	1	9
0	1	0	0	0	1	1	#
0	1	0	1	0	1	0	*

The ECS permits a maximum of 15 address digits and sends/receives only those ASCII characters shown above. Any adjunct must:

a. Accept all characters shown.

b. Not send characters other than those shown. Doing so results in the ECS denying (return error) the request.

In certain ASAI messages (Transfer Event Report, for example) the Connected Number IE appears in a paired list of IEs. To keep one-to-one correspondence in such a list, the Connected Number IE appears even if an Address is not known. In such a case, the value of the length field is one, and the Address field is absent. Any adjunct interfacing to the ECS should be prepared to receive the Connected Number IE in this format.

The adjunct should be prepared to accept the Connected number default values "#####," or "*****."

Date/Time

The Date/Time IE shown in Figure 4-11 provides the date and/or time.

8	7	6	5	4	3	2	1		
	Date/Time								
0	0	1	0	1	0	0	1	Byte	1
	Information Element Identifier								
Length of Date/Time contents									2
Year									3
Month									4
Day									5
Hour									6
Minute									7
Second									8*

Figure 4-11. Date/Time Information Element

NOTE:
Bytes 3 through 8 are binary (bit 1 being the least significant bit).

Progress Indicator

The Progress Indicator IE shown in Figure 4-12 conveys information from the PRI network to an ASAI application. The ECS passes this information without altering it.

8	7	6	5	4	3	2	1		
0	Progress Indicator 0	0	1	1	1	1	0	Byte	1
	Information element identifier								
Length of Progress indicator contents									2
1 ext	Coding standard		0 spare		Location				3
1 ext	Progress description								4

Figure 4-12. Progress Indicator

Coding standard (byte 3)

Bits 7	6	
0	0	CCITT standard coding

Location (byte 3)

Bits 4	3	2	1	
0	0	0	0	User
0	0	0	1	public network serving local user
0	1	0	0	public network serving remote user
0	1	0	1	private network serving remote user

Progress Description (byte 4)

Bits							
7	6	5	4	3	2	1	
0	0	0	0	0	0	1	call is not end-to-end ISDN. Call prog. inband.
0	0	0	0	0	1	0	destination address is non-ISDN
0	0	0	0	0	1	1	origination address is non-ISDN
0	0	0	0	1	0	0	call has returned to the ISDN
0	0	0	1	0	0	0	in-band information now available

Redirecting Number

The Redirecting Number IE shown in Figure 4-13 identifies the called party number from which call redirection/diversion/transfer occurred.

NOTE:
The ASAI specification permits the Redirecting Number IE to be repeated within a message. The ECS uses only a single occurrence of this IE across the ASAI interface and does not repeat it.

8	7	6	5	4	3	2	1		
0	Redirecting number 1 1 1 0 1 0 0 Information element identifier							Byte	1
Length of redirecting number contents									2
0/1 ext	Type of number (Note 1)			Numbering plan identification (Note 1)					3
0/1 ext	Presentation indicator		0 Spare	0 0 Reserved		Screening indicator			3a*
1 ext	0 0 0 Spare			Reason for redirection					3b*
0 Spare	Number digits (Note 1)								4

The contents of this field are coded as shown in Called Party Number.

Figure 4-13. Redirecting Number

The ECS does not expect an ASAI adjunct to send the optional bytes 3a and 3b; these are shown for compatibility with the ASAI specification. If bytes 3a/3b are received, they are ignored.

The adjunct should be prepared to parse this IE both with and without optional bytes 3a and 3b.

Presentation Indicator and Screening Indicator (octet 3a, bits 6-7 and 1-2): Presently the ECS includes the optional octets 3a and 3b so that the Reason for Redirection in Octet 3b is available to adjunct applications. The presentation indicator is always coded as “0 0” (Presentation Allowed); the Screening Indicator is always coded as “0 0” (User provided — not screened). PRI uses these octets at a network-user interface.

NOTE:
Reason for redirection (byte 3b, bits 4-1):

These reasons are included for completeness. They are shown for compatibility with the ASAI specification. The ECS does not expect optional byte 3b in any feature invocations where the Redirecting IE is present.

Bits 4 3 2 1	
0 0 0 1	Call forwarding busy or called DTE busy
0 0 1 0	Call forwarding no reply
1 0 0 1	Called DTE out of order
1 0 1 0	Call forwarding by the called DTE
1 1 1 1	Call forwarding unconditional or systematic call redirection

Redirection Number

The Redirection Number IE shown in Figure 4-14 identifies the number toward which a call redirection/diversion/transfer occurred.

8	7	6	5	4	3	2	1		
0	Redirection number 1 1 1 0 1 1 0 Information element identifier							Byte	1
Length of redirection number contents									2
0/1 ext	Type of number (see Note)			Numbering plan identification (see Note)					3
0/1 ext	Presentation indicator		0 Spare	0 0 Reserved		Screening indicator			3a*
1 ext	0 0 Spare		0	Reason for redirection					3b*
0 Spare	Number digits (see Note)								4

NOTE:
The contents of this IE field are coded as shown in Called Party Number.

Figure 4-14. Redirection Number

The ECS does not expect an ASAI adjunct to send bytes 3a or 3b; these are shown for compatibility with the ASAI specification. If bytes 3a/3b are received, those bytes are ignored.

NOTE:
Reason for redirection (byte 3b, bits 4-1):

These reasons are included for completeness and are shown for compatibility with the ASAI specification. The ECS does not expect optional byte 3b in feature invocations where the Redirection IE is present.

Bits 4 3 2 1	
0 0 0 1	Call forwarding busy or called DTE busy
0 0 1 0	Call forwarding no reply

1 0 0 1	Called DTE out of order
1 0 1 0	Call forwarding by the called DTE
1 1 1 1	Call forwarding unconditional or systematic call redirection

Restart Indicator

The Restart Indicator IE shown in Figure 4-15 identifies the class of facility to be restarted. This information element is always encoded to restart the “interface.”

8	7	6	5	4	3	2	1		
	Restart Indicator								
0	1	1	1	1	0	0	1	Byte	1
	Information element identifier								
Length of Restart Indicator contents									2
1	0	0	0	0					
Ext	spare				Class				3

Figure 4-15. Restart Indicator Information Element

Class encodings (byte 3)

Bits	
3 2 1	
1 1 1	All Interfaces

User to User Information

The User to User Information IE shown in Figure 4-16 conveys information between ASAI applications. The ECS carries user to user information transparently; it does not interpret or act on user to user information. This IE can have a maximum of 32 bytes of user information.

The user to user information element is coded as shown below:

8	7	6	5	4	3	2	1		
0	User to User Information element Identifier							Octet	1
Length of User to User contents									2
Protocol discriminator									3
User information									4, etc.

Figure 4-16. User to User Information Element

Protocol Discriminator (Octet 3):

Bits								
8	7	6	5	4	3	2	1	
0	0	0	0	0	0	0	0	User-specific protocol
0	0	0	0	0	1	0	0	IA5 characters

Codeset 6 Information Elements

The codeset 6 information elements (IEs) are Lucent Technologies-specific and include the ASAI information elements.

These codeset 6 IEs are:

- Call Options
- Conference/Transfer Options
- Counter
- Data Item
- Data Bit Map
- Domain
- Facility Information Element (FIE)
- Feature
- Generic Billing Data Item
- Lookahead Interflow
- Management Information Element (MIE)
- Old Party Identifier
- Originating Line Information
- Party Identifier
- Resource Identifier
- Resource Status
- Service Circuit
- Specific Event
- Status
- Trunk Identification
- Trunk Group/Trunk Status
- User-Entered Code Version

Call Options

The Call Options IE shown in Figure 4-17 conveys information about options that may be used to establish a third party call.

8	7	6	5	4	3	2	1		
0	Call Options 1	0	0	1	0	1	1	Byte	1
	Information element identifier								
Length of Call Options contents									2
1 Ext	Option								3
1 Ext	Value								4*

Figure 4-17. Call Options Information Element

ASAI Option Encodings (Byte 3, bits 7 through 1)	Bits 7 6 5 4 3 2 1	
	0 0 0 0 0 0 0	Reserved
	0 0 0 0 0 0 1	Number of rings before No Answer Event Report
	0 0 0 0 0 1 0	Alerting order
	0 0 0 0 0 1 1	Priority calling
	0 0 0 0 1 0 0	Reserved
	0 0 0 0 1 0 1	Reserved
	0 0 0 0 1 1 0	Direct agent ACD call
	0 0 0 0 1 1 1	Supervisor assist call
	0 0 0 1 0 0 0	Reserved
	0 0 0 1 0 0 1	Reserved
	0 0 0 1 0 1 0	Tone Duration
	0 0 0 1 0 1 1	Pause Duration
	0 0 0 1 1 0 0	Return Acknowledgement Flag
	0 0 0 1 1 1 0	Answering Machine Treatment

Value (Byte 4, bits 7 through 1)

This field is coded as a binary number and is application-dependent. If the Option field = Alerting order, the Value field is coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Alert calling endpoint first
0 0 0 0 0 0 1	Alert called endpoint first

If the Option field = Answer Machine Treatment, then the Value field is coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Administered Treatment
0 0 0 0 0 0 1	Disconnect/Drop
0 0 0 0 0 1 0	Answer

If the Option field = Number of Rings before No Answer, then the Value field contains the binary encoded number of rings.

If the Option field = Tone Duration or Pause Duration, then the Value field contains the binary-encoded integer that is multiplied by 0.01 seconds to compute a duration time or pause time.

If the Option field = Priority, Direct-Agent Call, Return Acknowledgment Flag, or Supervisor-Assist, the Value field is omitted.

Conference/Transfer Options

The Conference/Transfer IE shown in Figure 4-18 specifies nondefault actions for the Third Party Merge capability.

8	7	6	5	4	3	2	1		
0	Conference/Transfer Options 1 0 0 1 0 1 0 Information element identifier							Byte	1
Length of Conference/Transfer Options contents									2
1 Ext	Option								3

Figure 4-18. Conference/Transfer Options Information Element

Option

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Controller remains on merged call (for example, conference)
0 0 0 0 0 1 0	Controller drops off merged call (for example, transfer)

Counter

The Counter IE shown in Figure 4-19 sends binary-encoded counts of various items in response to various query.

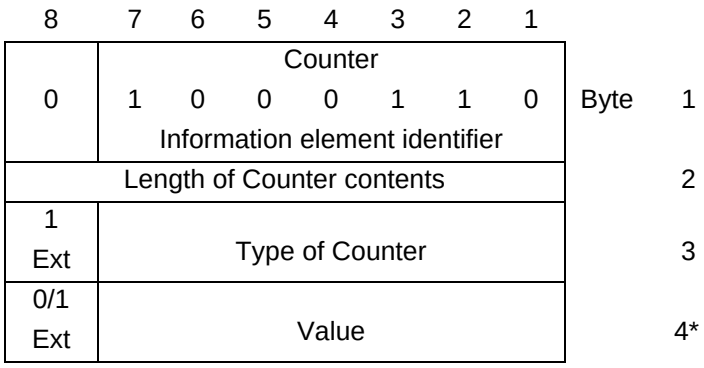


Figure 4-19. Counter Information Element

Type of Counter	Bits	
	7 6 5 4 3 2 1	
	0 0 0 0 0 0 0	Any
	0 0 0 0 0 0 1	Number of available trunks
	0 0 0 0 0 1 0	Number of available call classifiers
	0 0 0 0 0 1 1	Number of calls in queue
	0 0 0 0 1 0 0	Number of logged-in agents
	0 0 0 0 1 0 1	Number of available agents

Value This field is optional and represents the binary count associated with the Counter Type. The ECS currently sends one- or two-byte Value fields. Adjuncts should be prepared to receive two-byte values.

NOTE:
When a two-byte value is present, the high order bits are in byte 4 and the low order bits are in byte 4a, just as they are with two-byte CRVs. Refer to the *DEFINITY Enterprise Communications Server Administration and Features Description*, 555-230-522, for specifics on the ECS running ASAI to determine the maximum values for the parameters encoded in the Value field. This indicates the maximum number of bytes used to encode the information. This number varies from ECS to ECS. Hosts should operate with the maximum values for the largest ECS on which they will be running.

Data Item IE

The Data Item IE shown in Figure 4-20 returns information that an adjunct has queried for. The Data Item information element contains one information value. The data item type identifies what value is present in the IE. Presently, the Data Item IE is used in the response to the Integrated Directory Database query and in the reporting of ISDN Charge Advice.

8	7	6	5	4	3	2	1		
0	Data Items 1	0	1	0	0	1	0	Octet	1
	Information element identifier								
Length of Data Items contents									2
1 Ext	Data Item Type								3
1 Ext	Data Item Value								4*?

Figure 4-20. Data Item Information Element

Data Item Type (octet 3):

This byte may need to be interpreted differently in future messages. Currently this byte can take on the following values/meanings:

Bits							
7	6	5	4	3	2	1	
0	0	0	0	0	0	0	Reserved
0	0	0	0	0	0	1	Intermediate Charge
0	0	0	0	0	1	0	Final Charge
0	0	0	0	0	1	1	Split Charge
0	0	0	0	1	0	0	Name

Data Item Value

This value may be either an integer value, or an ASCII string. Charge amounts are encoded as integers and names are encoded as ASCII strings.

If an integer is encoded, then 7 bits of each octet are used. The high bit is an extension bit, and is set to 0 except in the last octet of the value. The most significant octets are ordered first. For example, 400 (decimal), which is 0x190 (hexadecimal) is encoded in two octets:

0	0	0	0	0	0	1	1	4a
EXT								
1	0	0	1	0	0	0	0	4b
EXT								

ASCII strings are encoded with the high bit set to 0 for every byte except the last byte.

Domain

The Domain IE shown in Figure 4-21 specifies the domain for which an event is reported.

8	7	6	5	4	3	2	1		
0	Domain Information 1 0 0 1 0 0 1 Information element identifier							Byte	1
Length of Domain Information contents									2
1 Ext.	Domain Type								3
0/1 Ext	Address								4*

Figure 4-21. Domain Information Element

Domain Type

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Adjunct-Monitored call audit
0 0 0 0 0 0 1	Group Extension (includes ACD split)
0 0 0 0 0 1 1	Extension
0 0 0 0 1 1 0	Work mode
0 0 0 0 1 1 1	Talk state
0 0 0 1 0 0 1	Trunk access code
0 0 0 1 1 0 0	Vector Directory Number
0 0 0 1 1 1 0	Announcement
0 0 0 1 1 1 1	Data Extension
0 0 1 0 0 0 0	ASAI
0 0 1 0 0 0 1	Station Type
0 0 1 0 0 1 0	Other
0 0 1 0 0 1 1	Logical Agent
0 0 1 0 1 0 1	Integrated Directory Database
0 0 1 1 0 0 0	Reason Codes

Address (byte 4, bits 7 through 1)

This optional field has a variable number of bytes and its format depends on the domain type.

- If the domain type is a group extension, an extension, logical agent, trunk access code, VDN, announcement, or data extension, the address field may contain the extension number (in ASCII).[1] In an incoming Domain IE the ECS will parse, the only ASCII characters the ECS permits are the digits 0 through 9.[2] There may be instances of event reports where an address should be included, but an address is not available to the ECS. In such cases, the address is present where ASAI requires it. However, in such a case, the ***Address*** field contains default values:
  - — ***** is the default value for a local ECS extension (might be an "interphantom" extension);
  - — ##### is the default value for an unknown external number.

  Any adjunct interfacing to the ECS should be prepared to receive the address with default values.
- If the domain type is "Work Mode," the address field contains a single-byte coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Aux mode
0 0 0 0 0 1 0	After Call Work Mode
0 0 0 0 0 1 1	Auto In Mode
0 0 0 0 1 0 0	Manual In Mode
0 0 0 0 1 1 1	Logged in
0 0 0 1 0 0 0	Not Logged in

1. The address (extension number) is not present when the Domain IE is responding to the extension information query, except for an extension information query containing a logical extension of a logged-in agent. In this case, the address contains the physical extension the agent is logged into. The address is present in all other cases.
2. The *Adjunct/Switch Application Interface (ASAI) Specification* also permits "*" and "#."

- If the domain type is “Talk State,” the address field contains a single byte coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 1 0 1	On Call
0 0 0 01 1 0	Available/Not on Call

- If the domain type is “Station Type,” the address field contains a single byte coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Analog
0 0 0 0 0 1 0	Proprietary
0 0 0 0 0 1 1	Basic Rate ISDN Interface

- If the domain type is “ASAI,” the address field contains a single byte coded as follows:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	No Bearer Channel Capabilities

- If the Domain type is Reason Codes, the address field contains a single byte with a binary value between 1-9.
- For other domain types, the address field is not used (call classifiers cannot be addressed individually).

Facility Information Element (FIE)

The Facility information element (FIE) identifies and contains an ASAI capability request or response. CCITT Recommendation Q.932 uses an X.209 encoding of X.229 Protocol Data Units.

The FIE only appears once in a message. The maximum length of the BRI message determines the number of bytes that the FIE may occupy; therefore, it imposes an upper bound on the length of an IE.

Table 4-6 to Table 4-9 show the FIE encoding.

8	7	6	5	4	3	2	1		
0	Facility 0	0	1	1	1	0	0	Byte	1
	information element identifier								
Length of facility contents									2
1	0	0	1	0	0	0	1		3
Ext	spare		Service Discriminator						
Component (Note 1)								4 etc.	

NOTE:
The FIE may contain only one component.

Figure 4-22. Facility Information Element

Service Discriminator

The ECS/ASAI recognizes only one service discriminator, Q.932 Supplementary Services, with the value shown: 1 0 0 0 1.

Components (Bytes 4, etc.)

A component is a sequence of data elements, each made up of a tag, a length, and contents. The first byte of the FIE component (byte 4) gives the component type. The component types for the Facility information element are:

- Invoke
- Return Result
- Return Error
- Reject

Table 4-6. Invoke Component

Invoke component	Section	Mandatory Indication	Byte Group
Component type tag Component length[1]	Component Type Tag "Length of Each Component or Contents"	Mandatory	4 5
Invoke identifier tag Invoke identifier length Invoke identifier	Invoke Identifier Tags "Length of Each Component or Contents" Invoke Identifier	Mandatory	6 7 8
Linked identifier tag[2] Linked identifier length Linked identifier	Not used Not used Not used	Optional	9 10 11
Operation value tag Operation value length Operation Value	Operation Value Tag "Length of Each Component or Contents" Operation Value	Mandatory	12 13 14
Argument[3]	Using Existing Q.931 Information Elements as Parameters	Optional	15, etc.

1. The component length is coded to indicate the number of bytes contained in the component (excluding the component type tag and the component length bytes).

2. This ASAI implementation never uses or expects linked identifiers. Refer to the *Adjunct/Switch Application Interface (ASAI) Specification* for a description.

3. Contains capability parameter(s) for the Invoke component type.

Table 4-7. Return Result Component

Invoke component	Section	Mandatory Indication	Byte Group
Component type tag Component length[1]	Component Type Tag "Length of Each Component or Contents"	Mandatory	4 5
Invoke identifier tag Invoke identifier length Invoke identifier	Invoke Identifier Tags "Length of Each Component or Contents" Invoke Identifier	Mandatory	6 7 8
Sequence tag[2, 3] Sequence length	Sequence Value Tag "Length of Each Component or Contents"	Optional	9 10
Operation value tag[3] Operation value length Operation Value	Operation Value Tag "Length of Each Component or Contents" Operation Value	Mandatory	11 12 13
Result[3, 4]	Using Existing Q.931 Information Elements as Parameters	Optional	14, etc.

1. The component length is coded to indicate the number of bytes contained in the component (excluding the component type tag and the component length bytes).

2. The sequence tag is an anomaly in the Return Result component. Although other components are similarly structured, the Return Result component is the only component with a sequence tag. This arises from a specific backward compatibility need for the Q.932 Return Result component with respect to other CCITT standards.

3. When the sequence tag is present, both the Operation Value and the Result must be present. All are present or none are.

4. Contains parameter(s) of the Return Result component type.

Table 4-8. Return Error Component

Invoke Component	Section	Mandatory Indication	Byte Group
Component type tag Component length[1]	Component Type Tag "Length of Each Component or Contents"	Mandatory	4 5
Invoke identifier tag Invoke identifier length Invoke identifier	Invoke Identifier Tags "Length of Each Component or Contents" Invoke Identifier	Mandatory	6 7 8
Error value tag Error value length Error value/Operation Value[2]	Error Value Tag "Length of Each Component or Contents" Error Value	Mandatory	9 10 11
Parameter[3]	Using Existing Q.931 Information Elements as Parameters	Optional	12, etc.

1. The component length is coded to indicate the number of bytes contained in the component (excluding the component type tag and the component length bytes).

2. The ASAI specification and the Q.932 specification have an "error value" here in the return error component. Both ASAI and Q.932 have the same encoding for the "error value tag" in the return error component and the "operation value tag" in the Invoke component. This means that what occupies the "error value" field is an Operation Value.

3. Contains parameter(s) of the return error component type

Table 4-9. Reject Component

Reject Component	Section	Mandatory Indication	Byte Group
Component type tag Component length[1]	Component Type Tag "Length of Each Component or Contents"	 Mandatory	4 5
Invoke identifier tag Invoke identifier length Invoke identifier	Invoke Identifier Tags "Length of Each Component or Contents" Invoke Identifier	 Mandatory	6 7 8
Problem tag Problem length[2] Problem	Problem Tag "Length of Each Component or Contents" Problem Tag	 Mandatory	6 7 8

1. The component length is coded to indicate the number of bytes contained in the component (excluding the component type tag and the component length bytes).

2. The problem length is always one byte.

Component/Content Length

Lengths up to 127 bytes are coded using the short form of Recommendation X.209: bit 8 is set to zero and the remaining seven bits are a binary encoding of the length with bit 1 the least significant bit.

This length encoding is identical to that of Recommendation Q.931 for lengths up to 127 bytes.

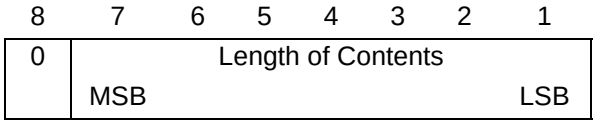


Figure 4-23. Format of the Length Field (Short Form)

If the length of the contents is greater than 127 bytes, then the long form of the length of the contents is used. The long form length, in the ECS/ASAI, is two bytes.[3] Bit 8 of the first byte is coded 1, and bits 1 to 7 of the first byte encode a number on less than the size of the length in bytes as an unsigned binary number whose MSB and LSB are bits 7 and 1, respectively. The length itself is encoded as an unsigned binary number whose MSB and LSB are bit 8 of the second byte and bit 1 of the last byte, respectively. This binary number shall be encoded in the fewest possible bytes, with no leading bytes having the value 0.

ASAI permits the length to be encoded in up to 127 bytes. However, two bytes suffice for encoding the maximum length of BRI messages.

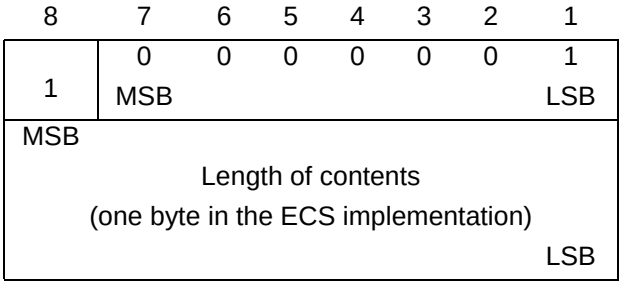


Figure 4-24. Long Form of the Length Field in the ECS Implementation (2 Bytes)

Component Type Tag

Table 4-10 shows the coding of the component type tag (control type).

Table 4-10. Component Type Tag

	8	7	6	5	4	3	2	1
Invoke	1	0	1	0	0	0	0	1
Return result	1	0	1	0	0	0	1	0
Return error	1	0	1	0	0	0	1	1
Reject	1	0	1	0	0	1	0	0

3. ASAI permits the length to be encoded in up to 127 bytes. However, two bytes suffice for encoding the maximum length of BRI messages.

Invoke Identifier Tags

An invoke identifier is used to identify an operation invocation and is present in the return result or return error response. When a protocol problem occurs with an FIE component, a Reject component contains the invoke-id of the problem component if it can be obtained. If the invoke-id cannot be ascertained, the Reject component contains a null invoke-id. Invoke identifiers are one byte long. The null invoke-id has zero length. Table 4-11 shows the coding of the component identifier tags.

Table 4-11. Invoke Identifier Tag Coding

	8	7	6	5	4	3	2	1
Invoke identifier	0	0	0	0	0	0	1	0
Null	0	0	0	0	0	1	0	1

Invoke Identifier

The invoke identifier is a binary field assigned by the sender of the Facility information element when the Component type tag in byte 4 indicates an invoke. For the remaining Component type tags (for example, return result, return error, and reject), the “invoke identifier” corresponds to one received in a Facility information element with an Invoke component type tag.

Invoke identifiers are coded in a single byte.

Operation Value Tag

Table 4-12 shows the coding for the operation value tag.

Table 4-12. Operation Value Tag Coding

	8	7	6	5	4	3	2	1
Operation value tag	0	0	0	0	0	0	1	0

Sequence Value Tag

The Return Result component contains a sequence value tag when an Operation Value and result are present. The sequence Value Tag, Operation Value Tag, and (non-null) result must all be present together, or absent from a Return Result component. Table 4-13 shows the encoding for the sequence tag.

Table 4-13. Sequence Tag Coding

Sequence Tag	0	0	1	1	0	0	0	0

Operation Value

The Operation Value specifies the ASAI capability being requested or responded to. Values are identified in bytes 12 and 11 of the Invoke and Return Result components, respectively. Table 4-14 shows the ASAI Operation Values; the ECS implementation currently does not use all Operation Values. Those that are not currently in use are marked **NOT USED** or **RESERVED.**

Table 4-14. Operation Value/Error Value Coding

ASAI Operation Value	8	7	6	5	4	3	2	1
Third Party Make Call	1	0	0	0	0	0	1	1
Third Party Answer	1	0	0	0	0	1	0	0
Third Party Selective Hold	1	0	0	0	0	1	0	1
Third Party Reconnect	1	0	0	0	0	1	1	0
Third Party Transfer **NOT USED**	1	0	0	0	0	1	1	1
Third Party Selective Drop	1	0	0	0	1	0	0	0
Third Party Merge	1	0	0	0	1	0	0	1
Event Notification Request	1	0	0	0	1	0	1	0
Reserved	1	0	0	0	1	0	1	1
Value Query	1	0	0	0	1	1	0	0
Set Value	1	0	0	0	1	1	0	1
Request Feature	1	0	0	0	1	1	1	1
RESERVED	1	0	0	1	0	0	0	0
RESERVED	1	0	0	1	0	0	0	1
RESERVED	1	0	0	1	0	0	1	0
Third Party Clear Call	1	0	0	1	0	0	1	1
Event Notification Cancel	1	0	0	1	0	1	0	0
Event Report	1	0	0	1	0	1	0	1
RESERVED	1	0	0	1	1	0	1	0
RESERVED	1	0	0	1	1	0	1	1
Route	1	0	1	1	0	0	0	0
RESERVED	1	0	1	1	0	0	0	1
Restart	1	0	1	1	0	0	1	0
Heartbeat	1	0	1	1	0	0	1	1
Third Party Conference **NOT USED**	1	0	1	1	0	1	0	0
Third Party Relinquish Control	1	0	1	1	0	1	0	1
Third Party Take Control	1	0	1	1	0	1	1	0
Route Select	1	0	1	1	0	1	1	1
RESERVED	1	0	1	1	1	0	0	0
Route End	1	0	1	1	1	0	0	1
Call Ended	1	0	1	1	1	0	1	1
Notification Ended	1	0	1	1	1	1	0	0
Third Party Make Call Proceed	1	0	1	1	1	1	0	1
Abort	1	0	1	1	1	1	1	0
Response Continued	1	0	1	1	1	1	1	1
RESERVED	1	1	0	0	0	0	0	0
Route Proceed **NOT USED**	1	1	0	0	0	0	0	1
Third Party Domain (Station) Control Request	1	1	0	0	0	1	0	0
Third Party Domain (Station) Control Ended	1	1	0	0	0	1	0	1
Third Party Auto Dial	1	1	0	0	0	1	1	0

Table 4-14. Operation Value/Error Value Coding — *Continued*

ASAI Operation Value	8	7	6	5	4	3	2	1
Stop Call Notification	1	1	0	0	0	1	1	1
Send DTMF	1	1	0	0	1	0	0	0
Redirect Call	1	1	0	0	1	0	0	1
Third Party Listen Disconnect	1	1	0	0	1	0	1	0
Third Party Listen Reconnect	1	1	0	0	1	0	1	1
RESERVED	1	1	0	0	1	1	0	0

Error Value Tag

Error reporting is specific to each capability. Table 4-15 shows the ASAI coding for the error value tag.

Table 4-15. Error Value Tag Coding

	8	7	6	5	4	3	2	1
Error value tag	0	0	0	0	0	0	1	0

Error Value

The error value is always encoded to be the Operation Value of the requested service that returned the error. Table 4-15 shows these encodings.

Problem Tag

The problem tag found in the Reject component supplies information about why an FIE was rejected. The possible protocol problems are divided into related groups. Table 4-16 indicates the tags for these groups. Table 4-17 to Table 4-20 give encodings for each set of problems within a problem tag. All ASAI problem codings are shown for completeness. Adjuncts should be prepared to accept any of them even though any given ECS release may only send a subset from the full set. If an adjunct receives an FIE that it cannot parse, it has the option of using a reject FIE (with problem tags), aborting the association, or sending a RELease COMplete message with a protocol error cause. These options all have the same effect from the ECS point of view.

Table 4-16. Problem Tag Coding

	8	7	6	5	4	3	2	1
General problem	1	0	0	0	0	0	0	0
Invoke problem	1	0	0	0	0	0	0	1
Return Result problem	1	0	0	0	0	0	1	0
Return Error problem	1	0	0	0	0	0	1	1

Table 4-17. General Problem Coding

	8	7	6	5	4	3	2	1
Unrecognized component	0	0	0	0	0	0	0	0
Mistyped component	0	0	0	0	0	0	0	1
Badly structured component	0	0	0	0	0	0	1	0

General Problem:

Unrecognized component	The component type tag is not a value in Table 4-9.
Mistyped component	A mandatory part of the component structure is missing.
Badly-structured component	The structure of the component cannot be parsed into one of the appropriate types of structures for the component type (Table 4-6, Table 4-7, Table 4-8, and Table 4-9); could be a length error.

Table 4-18. Return Result Problem Coding

	8	7	6	5	4	3	2	1
Unrecognized invocation	0	0	0	0	0	0	0	0
Result response unexpected	0	0	0	0	0	0	0	1
Mistyped result	0	0	0	0	0	0	1	0

Return Result problem:

Unrecognized invocation:	The Invoke-id in a Return Result component is not an active Invoke-id; for example, there is no such operation in progress.
Result response unexpected:	The Invoke-id in a Return Result component corresponds to an operation for which a Return result is not expected.
Mistyped result	Indicates the type of the result argument used is not the "Q.931 IE tag" shown in Figure 4-25.

Table 4-19. Invoke Problem Coding

	8	7	6	5	4	3	2	1
Duplicate invocation	0	0	0	0	0	0	0	0
Unrecognized operation	0	0	0	0	0	0	0	1
Mistyped argument	0	0	0	0	0	0	1	0
Resource limitation	0	0	0	0	0	0	1	1
Initiator releasing identifier	0	0	0	0	0	1	0	0
Unrecognized linked identifier	0	0	0	0	0	1	0	1
Linked response unexpected	0	0	0	0	0	1	1	0
Unexpected child operation	0	0	0	0	0	1	1	1

Invoke-problem:

Duplicate invocation:	An Invoke-id in an invoke component is already active.
Unrecognized operation:	The type of the operation argument supplied is not that shown in Table 4-12, or the Operation Value is not a supported Operation Value from Table 4-14.
Mistyped argument:	Indicates that the type of the operation argument is not the "Q.931 IE tag" shown in Figure 4-25.
Resource limitation:	The endpoint is not able to process the invoked capability due to a resource limitation.
Initiator releasing:	**NOT USED**

Unrecognized linked identifier	**DOES NOT APPLY**
Linked response unexpected:	A linked invoke-id was not expected.
Unexpected child operation:	**DOES NOT APPLY**

Table 4-20. Return Error Problem Coding

	8	7	6	5	4	3	2	1
Unrecognized invocation	0	0	0	0	0	0	0	0
Error response unexpected	0	0	0	0	0	0	0	1
Unrecognized error	0	0	0	0	0	0	1	0
Unexpected error	0	0	0	0	0	0	1	1
Mistyped parameter	0	0	0	0	0	1	0	0

Return Error problem:

Unrecognized invocation:	The Invoke-id in a return error component is not an active Invoke-id; for example, there is no such operation in progress.
Error response unexpected:	The Invoke-id corresponds to an operation for which a return result is not expected.
Unrecognized error:	The error value is not recognized. This means that the error value is not a recognized Operation Value.
Unexpected error:	The Error Value is recognized, but is not expected for the operation that the Invoke-id identifies.
Mistyped parameter:	The type of the result argument is not the Q.931 IE tag shown in Figure 4-25.

Using Existing Q.931 IEs as Parameters

All ASAI capabilities pass Q.931 IEs as parameters within the FIE parameters field. When Q.931 information elements (codesets 0 and 6) are parameters for a component, an X.209 data element encapsulates them, thereby retaining the Q.931 coding for these information elements. However, ASAI prohibits encapsulation of an FIE in this manner. The Q.931 information elements are grouped by codeset, and the IEs within each grouping are arranged in order of their byte identifiers (see "Coding Rules"). The “Q.931 information elements tag” precedes all Q.931 IEs and encapsulates them. See Figure 4-25.

8	7	6	5	4	3	2	1
0	1	0	0	0	0	0	0
Q.931 information elements tag							
Q.931 information elements length (X.209 encoded -- see section “Length of Each Component or Contents”)							
First Q.931 information element							
Second Q.931 information element							
. . . .							
Last Q.931 information element							

Figure 4-25. Encapsulation of Q.931 Information Elements

Feature

The Feature IE shown in Figure 4-26 specifies which feature is being requested. It is also used to indicate the availability of a feature in a Call Offered Event Report. (Currently, only Flexible Billing is supplied.)

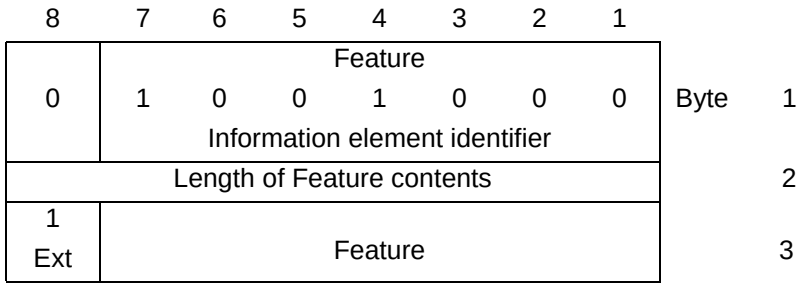


Figure 4-26. Feature Information Element

ASAI Feature Encodings:	Bits 7 6 5 4 3 2 1	
	0 0 0 0 0 0 0	Reserved
	0 0 0 0 0 0 1	Change agent work mode
	0 0 0 0 0 1 0	Log in agent
	0 0 0 0 0 1 1	Log out agent
	0 0 0 0 1 0 0	Reserved
	0 0 0 0 1 0 1	Reserved
	0 0 0 0 1 1 0	Flexible Billing
	0 0 0 0 1 1 1	LWC on **NOT USED**
	0 0 0 1 0 0 0	LWC off **NOT USED**
	0 0 0 1 0 0 1	Reserved
	0 0 0 1 0 1 0	Call Forward — Activate
	0 0 0 1 0 1 1	Call Forward — Cancel
	0 0 0 1 1 0 0	Send All Calls — Activate
	0 0 0 1 1 0 1	Send All Calls — Cancel

Generic Billing Data

The Generic Billing Data IE shown in Figure 4-27 specifies the billing for a specified call. It also provides billing information as an event field in the Offered Event and in route requests.

8	7	6	5	4	3	2	1		
0	Generic Billing Data								
	1	0	1	0	1	1	0	Byte	1
	Information element identifier								2
Length of Generic Billing Data contents									
1 Ext	Type of Billing Data								3
1 Ext	Encoding scheme								4*
	Billing data (hundreds)								5*
	Billing data (tens)								6*
	Billing data (units)								7*
	Billing data (tenths)								8*
	Billing data (hundredths)								9*

Figure 4-27. Generic Billing Data Information Element

Type of Billing Data	Bits 7 6 5 4 3 2 1	
	0 0 1 0 0 0 0	New rate
	0 0 1 0 0 0 1	Flat rate
	0 0 1 0 0 1 0	Premium Charge
	0 0 1 0 0 1 1	Premium Credit
	0 0 1 1 1 0 0	Free call

Encoding Scheme	Bits 7 6 5 4 3 2 1	
	0 0 0 0 0 1 0	IA5

Item

The Item IE shown in Figure 4-28 specifies the item being queried or set in the Value Query and Set Value capabilities.

8	7	6	5	4	3	2	1		
0	Item 1	0	0	1	1	0	1 Information element identifier	Byte	1
Length of Item contents									2
1 Ext	Item (Note)								3

NOTE:
The ECS permits only one item in a Value Query or Set Value capability invocation.

Figure 4-28. Item Information Element

ASAI ITEM Encodings:

Bits 7 6 5 4 3 2 1	Item
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Date/Time
0 0 0 0 0 1 1	MWL Status
0 0 0 1 1 0 1	Agent Login Audit
0 0 1 1 0 1 0	ACD Split **NOT USED**
0 0 1 1 0 1 1	Party Query Information
0 0 1 1 1 0 0	Call Query Information
0 0 1 1 1 0 1	Extension Query Information
0 0 1 1 1 1 1	Billing Change Request

Lookahead Interflow

The ECS passes the PRI Lookahead Interflow IE shown in Figure 4-29 with Call Offered Event Reports. This information element contains the calling information associated with the call at the previous ECS. Thus, when a call interflows from one ECS to another, the second ECS can pass information such as Dialed Number Identification Service (DNIS) to an adjunct on that ECS.

8	7	6	5	4	3	2	1		
0	1	1	1	1	0	1	1	Byte	1
Length of Lookahead Interflow Contents									2
1 ext	Priority Level				Type Interflow				3
1 ext	0	0	Time Stamp Hours						4
1 ext	0	Time Stamp Minutes							5
1 ext	0	Time Stamp Seconds							6
1/0 ext	DNIS Name Display ASCII Characters								

Figure 4-29. Lookahead Interflow Information Element

Type Interflow (Byte 3)

Bits 3	2	1	
0	0	0	All interflow (reserved)
0	0	1	Threshold interflow (reserved)
0	1	0	Vectoring interflow

Priority Level (Byte 3)

Bits 7	6	5	4	
0	0	0	0	Not in queue
0	0	0	1	Low
0	0	1	0	Medium
0	0	1	1	High
0	1	0	0	Top

Time Stamp (Bytes 4-6)	Hours: 5 bits: 0 through 23 Minutes: 6 bits: 0 through 59 Seconds: 6 bits: 0 through 59
DNIS NAME Display Characters (Bytes 7 through 21)	Variable: 0 through 15 ASCII characters. For interoperation with existing services, these should be printable ASCII characters.

Management Information Element (MIE)

The Management Information Element (MIE) shown in Figure 4-30 carries information to suspend and resume alarming on ASAI interfaces. The ASAI subset of the MIE structure and encoding shown here are part of a more general encoding from the *ISDN Basic Rate Interface (BRI) Specification*.

8	7	6	5	4	3	2	1		
	Management Information								
0	1	1	1	1	0	1	0	Byte	1
	Information element identifier								
Length of Contents									2
Management Protocol Discriminator									
0	0	0	0	0	0	1	1		3
0/1	Transaction Reference								4
ext	0	0	0	0	0	0	0		
0/1	OP Class			spare		Op Type			
ext				x	x	0	0		5
0/1	Operation Value								
ext									6
0/1	Parameter Identifier								
ext									*7a
Length of parameter									*7b
Parameter Value									*7c

NOTE:
BRI permits the parameter bytes (7a through 7c) to be repeated. The ASAI interface currently uses only one parameter in an MIE. With present ASAI use of the MIE, the parameter bytes are present only when the Operation Class is *Confirmed Operation*.

Figure 4-30. Management Information Element

Transaction Reference (byte 4):
ASAI supports only a synchronous use of the MIE. This requires that the Transaction Reference value be zero. Non-zero values are used only for asynchronous management operations. The ECS takes the transaction value from a request (in this case, zero) and returns it in the response to that request.

Operation Class (byte 5, bits 5 through 7): may contain the following encodings on the ASAI interface:

7 6 5	
0 0 1	Confirmed Operation
0 1 0	Return Result
1 0 1	Reject

Operation Type (byte 5, bits 1 through 2): has only one permitted encoding on an ASAI interface.

2 1	
0 0	Action

Operation Value (byte 6): has only one permitted encoding on an ASAI interface.

7 6 5 4 3 2 1	
0 0 0 0 0 1 0	Link Alarm Status Change

Parameter Identifier (byte 7a through 7c): only one parameter is permitted on an ASAI interface.

7 6 5 4 3 2 1	
1 0 1 0 0 1 1	Alarm Status Parameter

The Permitted values for the Alarm Status Parameter are:

8 7 6 5 4 3 2 1	
0 0 0 0 0 1 0 0	Suspend Alarming on link
0 0 0 0 0 0 1 1	Resume Alarming on Link

If the ECS rejects a MIM message, the **Operation Class** is set to *reject*, the **Operation Type** is irrelevant and is always encoded as “0 0”, and the **Operation Value** field contains a *Management Error Code*.

The possible values for the *Management Error Code* are:

7 6 5 4 3 2 1	
0 0 0 0 0 0 1	Protocol Violation
0 0 0 0 0 1 0	Unrecognized Operation

Old Party Identifier

The Old Party Identifier IE shown in Figure 4-31 associates a previous party-identifier with the current one in situations where a call event merges two calls and the ECS may reassign party-ids. The correspondence of Old Party Identifier information elements with Party Identifier information elements correlates the old and new party identifiers. The following figure provides the format of this information element.

8	7	6	5	4	3	2	1		
0	Old Party Identifier 0 0 1 0 1 1 1 Information element identifier							Byte	1
Length of Old Party Identifier contents									2
1 Ext	From Call	Spare 0	0	0	0	0	0		3
0 Ext	Party ID								4
1 Ext	Party ID								4a

NOTE:
The adjunct should accept both one- and two-byte party-id values.

Figure 4-31. Old Party Identifier

From Call (byte 3; bit 7)

Bit 7	
0	Other Call
1	Resulting Call

Party ID (bytes 4, 4a; bits 7-1)
Binary number assigned by the ECS

Originating Line Information

The Information Identifier (II) Digits IE shown in Figure 4-32 shows the II-Digits carried in the Originating Line Identifier IE.

8	7	6	5	4	3	2	1		
0	Orig Line Info 0	0	0	0	0	0	1 Information element identifier	Octet	1
	Length of OLI Contents								2
	II-Digits								3

Figure 4-32. Originating Line Information Data Information Element

NOTE:
See Bellcore's *Local Exchange Routing Guide*, Document Number TR-EOP-000085 for further information. This document is updated quarterly.

Party Identifier

The Party Identifier IE shown in Figure 4-33 identifies a party on a third party call for call feedback event reporting or call control.

8	7	6	5	4	3	2	1		
0	Party Identifier 1 0 0 0 1 0 0 Information element identifier							Byte	1
Length of Party Identifier contents									2
0 Ext	Party ID								3
1 Ext	Party ID								3a

NOTE:
The current implementation uses a single byte for the Party_id field. For forward compatibility, any adjunct implementation should be prepared to accommodate two-byte Party_ids.

Figure 4-33. Party Identifier Information Element

Party ID (bytes 3, 3a; bits 7 through 1)

Binary number assigned by the ECS. This value is between 1 and 6 for non-bridged parties and is greater than 6 for bridged parties.

Resource Identifier

The Resource Identifier IE shown in Figure 4-34 identifies the resource for which a query is being made.

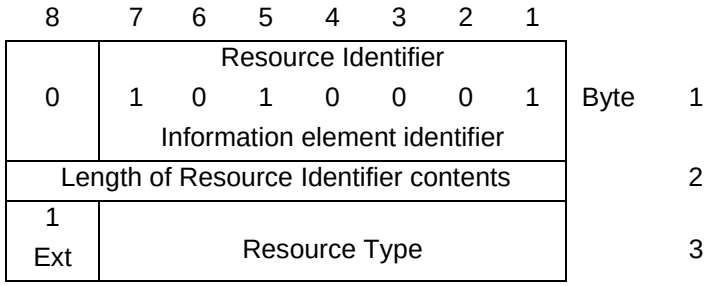


Figure 4-34. Resource Identifier Information Element

Resource Type:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Tone Detection/Call Classification
0 0 0 0 0 1 0	Tones

Resource Status

The Resource Status IE shown in Figure 4-35 gives status information about a resource.

8	7	6	5	4	3	2	1		
Resource status									
0	0	0	0	0	0	1	1	Byte	1
Information element identifier									
Length of resource status contents									2
1 Ext	Type of Resource								3
1 Ext	In Use	Idle	Maint Busy	Other Unavail	Spare				4
0/1 Ext	Count of Highest Order Status Shown								5
0/1 Ext	Count of Second Highest Order Status Shown								6
0/1 Ext	Count of Third Highest Order Status Shown								7
0/1 Ext	Count of Fourth Highest Order Status Shown								8

Figure 4-35. Resource Status Information Element

Type of resource:

Bit 7 6 5 4 3 2 1	
0 0 0 0 0 0	Reserved
0 0 0 0 0 1	Call classifier

Status Bits: When the In-use, Idle, Maintenance-busy, and Other-unavailable status bits are set to 1, a binary-encoded count corresponding to that status follows.

Counts: Bytes containing counts correlate with the status bits: The first count byte conveys the count for the highest order status bit set to 1. Typically, this is the “In-use” count, but it could be the idle count (for example). Later bytes convey the counts for the next highest order status bit set to 1.

NOTE:
Bytes 5 through 8 are extended to accommodate counts greater than 128.

The ECS typically returns counts for the resources in use and the idle resources. The adjunct may sum these two counts to calculate the resources in service. Refer to the *DEFINITY Enterprise Communications Server Administration and Feature Descriptions,* 555-230-522, for specifics on the ECS running ASAI to determine the maximum values for the parameters encoded in the Value field. This determines the maximum number of bytes used to encode the information. This number varies from ECS to ECS. Hosts should operate with the maximum values for the largest ECS on which they will be running.

Service Circuit

The Service Circuit IE shown in Figure 4-36 specifies service circuit(s) to be used for a given call.

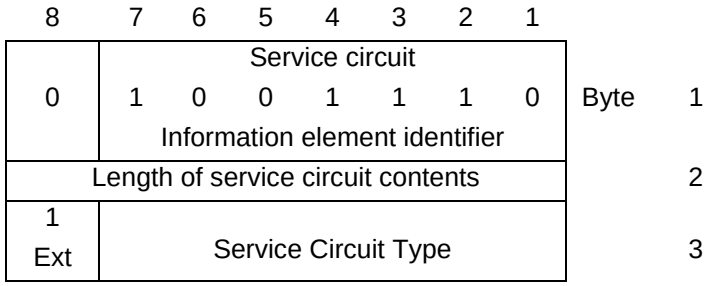


Figure 4-36. Service Circuit Information Element

Service Circuit Type:	Bit 7 6 5 4 3 2 1	
	0 0 0 0 0 0 0	Reserved
	0 0 0 0 0 0 1	Call Classifier

Specific Event

The Specific Event IE shown in Figure 4-37 specifies the call feedback event that has occurred. It also carries call state information when used in a query response.

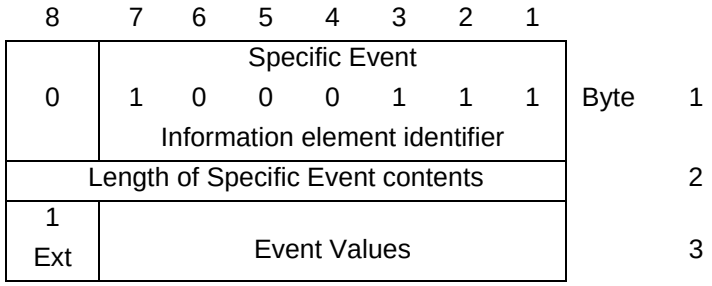


Figure 4-37. Specific Event Identifier Information Element

ASAI Event values:	Bits 7 6 5 4 3 2 1	
	0 0 0 0 0 0 0	Reserved
	0 0 0 0 0 0 1	Alerting Event/Alerting Call State
	0 0 0 0 0 1 0	Cut-through/Progress
	0 0 0 0 0 1 1	Reserved
	0 0 0 0 1 0 0	Disconnected/Dropped
	0 0 0 0 1 0 1	Reserved
	0 0 0 0 1 1 0	Hold Event/Held Call State
	0 0 0 0 1 1 1	Queued
	0 0 0 1 0 0 0	Busy/Unavailable
	0 0 0 1 0 0 1	Reserved
	0 0 0 1 0 1 0	Reserved
	0 0 0 1 0 1 1	Connected Event/Connected Call State
	0 0 0 1 1 0 0	Reorder/Denial
	0 0 0 1 1 0 1	Reserved
	0 0 0 1 1 1 0	Log In
	0 0 0 1 1 1 1	Log Out
	0 0 1 0 0 0 0	Reserved
	0 0 1 0 0 0 1	Call offered to domain/incoming call
	0 0 1 0 0 1 0	Trunk Seized
	0 0 1 0 0 1 1	Call Transferred
	0 0 1 0 1 0 0	Reserved
	0 0 1 0 1 0 1	Call Conferenced
	0 0 1 0 1 1 0	Classifier Detects Answer
	0 0 1 0 1 1 1	Call Initiated Event/Initiated Call State
	0 0 1 1 0 0 0	Reconnected
	0 0 1 1 0 0 1	Call Redirected
	0 0 1 1 0 1 0	Unknown Call State/Other
	0 0 1 1 0 1 1	Connected, Not Active (Bridged)/Bridged Call State
	0 0 1 1 1 0 0	Entered Digits
	0 0 1 1 1 0 1	Call Originated

Status

The Status IE:

- Supplies information for activating/deactivating Message Waiting Lamps
- Returns the status of the Message Waiting Lamp and messaging services supported by the ECS
- Returns the ON/OFF status of other ECS features such as Send-All-Calls and Call Forwarding

Figure 4-38 shows the format of the Status information element.

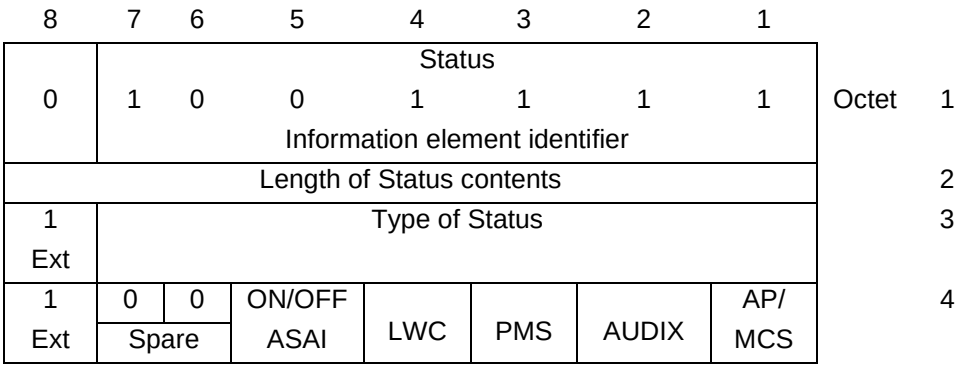


Figure 4-38. Status Information Element

Type of Status (byte 3, bit 7-1)

Bits
7 6 5 4 3 2 1

0 0 0 0 0 0 0	Reserved
0 0 0 0 0 0 1	Reserved
0 0 0 0 0 1 0	Message Waiting Indicator Status
0 0 0 0 1 0 0	Send-All-Calls Status
0 0 0 1 0 0 0	Call Forwarding Status

ON/OFF or ASAI (byte 4, bit 5)
In all Set Value and Feature Activation/De-activation requests, this bit contains an on/off indicator. In responses to feature status queries, this bit carries an ON/OFF indicator for Forwarding and Send-All-Calls responses. In the response to the MWL query, it indicates whether an ASAI adjunct has turned the MWL on. Thus, in the response to the MWL query, the adjunct must logically OR bits 1-5 in octet 4

to ascertain whether the MWL is on or off. If the result of the OR is a 1, then the MWL is on.

Bit 4	
0	Off (feature not activated or ASAI MWL off)
1	On (active feature or ASAI MWL on)

LWC (byte 4, bit 4) This bit is set only in the response for the Station Message Waiting Status Query.

Bit 4	
0	Off (no LWC messages)
1	On [LWC message(s)]

PMS (byte 4, bit 3) This bit is set only in the response for the Station Message Waiting Status Query.

Bit 4	
0	Off (no PMS messages)
1	On [PMS message(s)]

AUDIX (byte 4, bit 2) This bit is set only in the response for the Station Message Waiting Status Query.

Bit 4	
0	Off (no AUDIX messages)
1	On [AUDIX message(s)]

AP/MCS (byte 4, bit 1) This bit is set only in the response for the Station Message Waiting Status Query.

Bit 4	
0	Off (no AP/MCS messages)
1	On [AP/MCS message(s)]

Trunk Group Identification

The Trunk Group Identification IE shown in Figure 4-39 identifies a specific trunk group and trunk within the group.

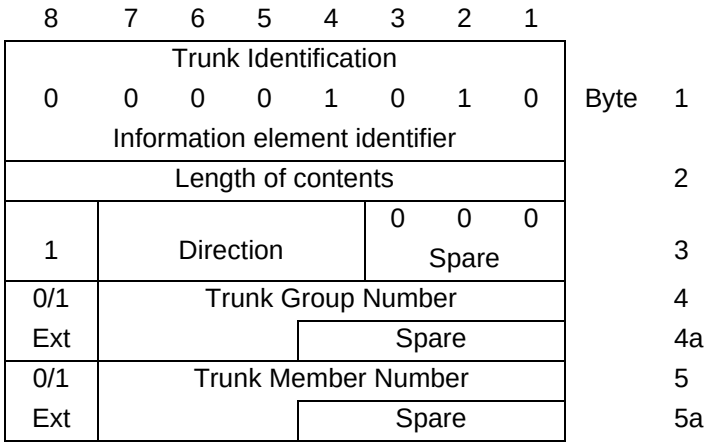


Figure 4-39. Trunk Identifier Information Elements

Direction (byte 3, bits 7 through 4)

Bits 7 6 5 4	
0 0 0 0	No direction
0 0 0 1	Incoming to ECS **NOT USED**
0 0 1 0	Outgoing from ECS **NOT USED**

Extension Bit (byte 4, 4a, 5, 5a, bit 8)

0: Description extends through next byte
1: Last byte of description

With a number greater than 127, the extension bits are set with bit 7 of byte 4/5 being the highest order bit of the extended byte. Bits 4 through 1 of bytes 4a and 5a are spare.

Trunk Group Number (bytes 4, 4a)

Contains a binary coding of the trunk group number with bit 7 of byte 4 being the highest order bit. Refer to the *DEFINITY Enterprise Communications Server Feature Description,* 555-230-301, for specifics on the ECS running ASAI to determine the maximum values for the parameters encoded in the Value field.

This determines the maximum number of bytes used to encode the information. This number varies from ECS to ECS. Hosts should operate with the maximum values for the largest ECS on which they will be running.

Trunk Member Number (bytes 5, 5a)

Contains a binary coding of the trunk member number with bit 7 of byte 5 being the highest order bit. Refer to the *DEFINITY Enterprise Communications Server Feature Description,* 555-230-301, for specifics on the ECS running ASAI to determine the maximum values for the parameters encoded in the Value field.

This determines the maximum number of bytes used to encode the information. This number varies from ECS to ECS. Hosts should operate with the maximum values for the largest ECS on which they will be running.

Trunk Group/Trunk Status

The Trunk Group/Trunk Status IE shown in Figure 4-40 conveys information about the busy/idle status of trunks at an ECS.

8	7	6	5	4	3	2	1		
Trunk Group/Trunk Status									
0	0	0	0	1	0	1	1	Byte	1
Information element identifier									
Length of contents									2
1 Ext	In Use	Idle	Maint Busy	Other Unavail	Trans	Spare			3
0/1 Ext	Count of Highest Order Status Shown								4
0/1 Ext	Count of Second Highest Order Status Shown								5
0/1 Ext	Count of Third Highest Order Status Shown								6
0/1 Ext	Count of Fourth Highest Order Status Shown								7

Figure 4-40. Trunk Group/Trunk Status Information Element

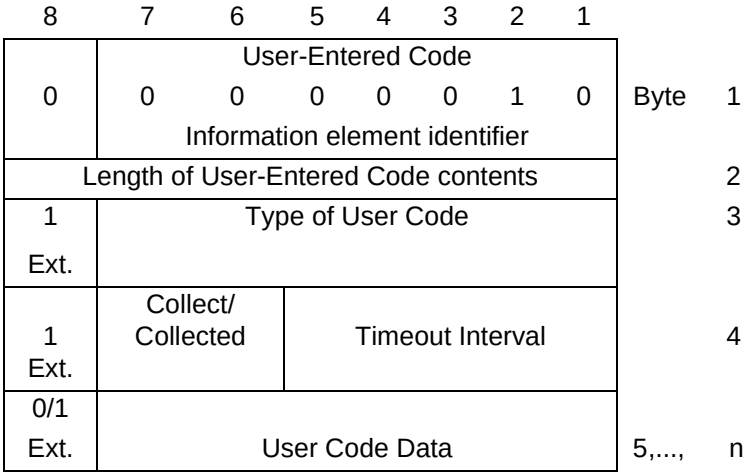
Status Bits When the In-use, Idle, Maintenance-busy, and Other-unavailable status bits are set to 1, a binary-encoded count corresponding to that status follows.

Counts Bytes containing counts correlate with the status bits: The first count byte conveys the count for the highest order status bit set to 1. Typically, this is the “In-use” count, but it could be the idle count (for example). Subsequent bytes convey the counts for the next highest order status bit set to 1. Bytes 4 through 7 are extended to accommodate counts greater than 128.

The ECS typically returns counts for the trunks in use and the idle trunks. The adjunct may sum these two counts to calculate the trunks in service in the trunk group. Refer to the *DEFINITY Enterprise Communications Server Feature Description* (555-230-301) for specifics on the ECS running ASAI to determine the maximum values for the parameters encoded in the Value field. This determines the maximum number of bytes used to encode the information. This number varies from ECS to ECS. Hosts should operate with the maximum values for the largest ECS on which they will be running.

User-Entered Code

The User-Entered Code IE shown in Figure 4-41 conveys user-entered information (for example, authorization codes or call prompter data).



8	7	6	5	4	3	2	1		
0	User-Entered Code 0	0	0	0	0	1	0 Information element identifier	Byte	1
Length of User-Entered Code contents									2
1 Ext.	Type of User Code								3
1 Ext.	Collect/ Collected	Timeout Interval							4
0/1 Ext.	User Code Data							5,...,	n

Figure 4-41. User-Entered Code Information Element

ASAI Encodings for Type of User Code:

Bits 7 6 5 4 3 2 1	
0 0 0 0 0 0 0	Any **NOT USED**
0 0 0 0 0 0 1	Reserved
0 0 0 0 0 1 0	Login Digits
0 0 0 0 0 1 1	Reserved
0 0 0 0 1 0 0	Reserved
0 0 0 0 1 0 1	Call Prompter
0 0 1 0 0 0 1	Customer-provided
0 0 1 0 0 1 0	Application
0 1 0 0 0 0 0	Tone Detector

Collect/Collected Indication

Bits 7 6	
0 0	Collect Digits. The following byte contains the number to be collected. (DEFINITY ECS maximum =24.)
1 0	The following bytes contain user-entered digits.

Timeout Interval

The interval for digit collection (in seconds) once DEFINITY ECS has received the first digit (0-63 sec). Encode as all zeros if no timeout is to be used.

User Code Data

The user-entered data is encoded in ASCII characters. The ECS sends up to 24 bytes of user code data (including * and #).

When the User-Entered Code carries information to log in an agent where EAS is activated, then the IE contains both the agent's EAS login ID and password. These are separated by a "#" character. If the password has fewer digits than what's administered on the ECS, you must provide a leading "0."

When the type of user code indicates tone detection and the collect/collected field indicates collect, then this field contains a binary value giving the number of digits to be collected. The current maximum is 24.

Version

The Version IE shown in Figure 4-42 conveys the ASAI protocol version used on a specified ASAI link. It may be repeated in the RESTART message to the adjunct in the case where the ECS supports multiple versions. The RESTart Acknowledgement message can also contain this IE.

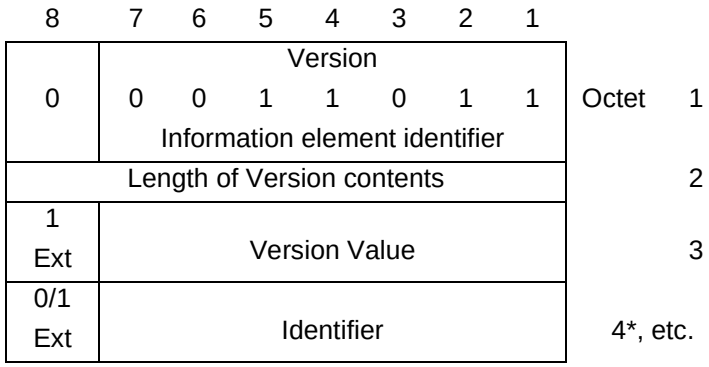


Figure 4-42. Version Information Element

Version Value (Octet 3)

Bits 7	6	5	4	3	2	1	
0	0	0	0	0	0	0	Reserved
0	0	0	0	0	0	1	ASAI Version 1
0	0	0	0	0	1	0	ASAI Version 2
0	0	0	0	0	1	1	ASAI Version 3

Identifier (Octet 4, etc.)

The identifier need not be present. For CallVisor for Unix System V (PC/ASAI), this is a 4-octet value.

Byte Level Messages

5

This chapter provides the encodings for each message that passes across the ASAI interface at the byte level for the following:

- Event Reports
- Call Control Capability Group
- Domain (Station/ACD Split/EAS Skill) Control Capability Group
- Notification Capability Group
- Routing Capability Group
- Request Feature Capability Group
- Value Query Capability Group
- Set Value Capability Group
- Maintenance Capability Group

Conventions

Within certain messages, specific information elements are optional and may or may not be present in a specific message. *Italics* denote these bytes in the **Byte Description** column of the tables in this chapter.

Some bytes may be repeated in a message; an asterisk ("*") at the beginning of the entry in the **Byte Description** column identifies such a byte.

In some messages, the ordering of instances of an information element that occurs multiple times is fixed. These situations are explicitly identified in notes with the message layouts. In all other cases, the ordering of the instances of an IE that appears multiple times is arbitrary.

The bit-wise OR operation is indicated by a vertical bar “|” and the resulting value is the bit-wise ORing of the values on either side of the bar. The word “or,” or a comma, indicates that the value is one of a list of possible values.

The occurrence of a question mark (?) in the Byte Value column means that a hex-byte value is present.

Event Reports

Event Reports: REGister Messages Sent by the Adjunct

None for Event Reports.

Event Reports: REGister Messages Sent by the switch

None for Event Reports.

Event Reports: FACility Messages Sent by the switch

A FACility message is used to initiate an action or pass information within an existing association on an active CRV. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Alerting Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
Length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81, 0xe1	
cause value	0x??	CS3/23 (Remains in Queue); CS3/31 (SAC, Cover All Calls, Go to Cover active, Adjunct Redirected-Alerting-Call); CS3/25 (Call Forwarding); CS3/26 (Cover, principal busy or all call appearances busy); CS3/28 (Cover Don't Answer)
The Connected Number IE	0x0c	
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x80	
** Address digits*	0x??	0x30 thru 0x39,0x23(#), 0x2a(*)

(Continued on next page)

Alerting Event Report — (*Continued*)		
Byte Description	**Byte Value**	**Comments**
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan Id	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
** Address digits*	0x??	0x30 thru 0x39,0x23(#), 0x2a(*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* ASCII Address digits	0x??	0x30 thru 0x39,0x23(#),0x2a(*)
User-User IE	0x7e	User-to-user information
The length of user-user IE	0x??	computed length
Protocol Discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
Originating Line Information IE	0x01	
Length of OLI	0x??	computed length
**II-Digits*	0x??	
The Trunk Identification IE	0x0a	
The length of IE	0x?	computed length
Direction	0x80	No Direction
** Trunk Group Number*	0x??	binary coded
** Trunk Group Member Number*	0x??	binary coded
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Alerting	0x81	
The Domain IE	0x49	present for ACD call
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split

(Continued on next page)

Alerting Event Report — (*Continued*)		
** The first to N-1 extension digits*	0x30 thru 0x39	
The last extension digit	0xbX	0xb0 thru 0xb9

NOTE:
The Calling Number IE and the Trunk Identification IE are mutually exclusive. One or the other will be present.

Answered Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
The Connected Number IE	0x0c	
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39,0x23(#), 0x2a(*)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

(Continued on next page)

Answered Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Answered	0x96	

Busy/Unavailable Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39,0x23(#), 0x2a(*)
Locking Shift to Code Set 6	0x96	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Busy	0x88	

Call Conferenced Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Call Identity IE	0x10	“other” call ID before conference
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Call Identity IE	0x10	“resulting” call ID after conference
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

(Continued on next page)

Call Conferenced Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan Id	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Locking Shift to Code Set 6	0x96	
* Old Party Identifier IE	0x17	Up to six old party IDs
* Length of the IE	0x02 or 0x03	
* From Call	0x??	0x80 or 0xc0
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
* The Party Identifier IE	0x44	Up to six party IDs
* Length of the IE	0x01 or 0x02	
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Conferenced	0x95	

NOTES:

The Call Identity IE containing the “other” call ID before the conference operation must precede the Call Identity IE that contains the “resulting” call ID after the conference operation.

There must be the same number of Party ID IEs, Old Party ID IEs, and Connected Number IEs.

The Nth Party ID IE and the Nth Old Party ID IE correspond to the Nth Connected Number IE (and thereby associate an extension with that party).

To preserve a 1:1 correspondence, if the type/address for a party is unknown, a Connected Number IE is present for that party, but the length byte is coded as zero (and no bytes follow).

Call Initiated Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Initiate	0x97	

Call Offered to Domain Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan Id	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
** Address digits*	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)

(Continued on next page)

Call Offered To Domain Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
User-User IE	*0x7e*	*User-to-User information*
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
Originating Line IE	0x01	
Length of OLI	0x??	computed length
***II-Digits*	0x??	
User Code IE	0x02	User Entered Data IE
length of user data	0x??	computed length
type of user data	0x85	Call Prompter Data
collected data indicator	0xc0	
** first n-1 user entered digits*	0x00 | 0x??	ASCII data
last user entered digit	0x80 | 0x??	ASCII data
The Trunk Identification IE	0x0a	
The length of IE	0x?	computed length
Direction	0x80	No Direction
** Trunk Group Number*	0x??	binary coded
** Trunk Group Member Number*	0x??	binary coded
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Incoming Call	0x91	
The Feature IE	0x48	
The Length of the Feature IE	0x01	
Feature	0x86	Flexible Billing
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type	0x8c or 0x81	VDN or split
* first to N-1 digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
The last digit	0xYY,	0xb0 thru 0x69, 0x23(#), 0xaa(*)
Lookahead Interflow IE	0x7b	PRI Interflow
length of lookahead interflow	0x??	computed length
Ext bit | Priority Level & Type	0x80 | 0x??	from PRI

(Continued on next page)

Call Offered To Domain Event Report — *(continued)*		
Ext bit | Time Stamp - hours	0x80 | 0x??	from PRI
Ext bit | Time Stamp - minutes	0x80 | 0x??	from PRI
Ext bit | Time Stamp - seconds	0x80 | 0x??	from PRI
**Ext bit | first n-1 ASCII characters*	0x00 | 0x??	ASCII data from PRI
Ext bit | last ASCII character	0x80 | 0x??	ASCII data from PRI

NOTE:
The Calling Number IE and the Trunk Identification IE are mutually exclusive. One or the other is present.

Call Originated Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	NOTE 1
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
The Connected Number IE	0x0c	The party extension
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x??	0x80
** ASCII Address digits*	0x??	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Call Identity IE	0x10	
Length of Call Identity IE	0x02	computed value

NOTE:
Not supported in G3V4 or Release 5.

(Continued on next page)

Call Originated Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
* Call identifier (2 octets)	0x??	switch-assigned value
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & Numbering Plan ID	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* ASCII Address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* ASCII Address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)
User-user IE	0x7e	NOTE 1
length of user-user IE	0x?	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second octet)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Call Originated	0x9d	

NOTE:
Not supported in G3V4 or Release 5.

Call Redirected Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Redirected	0x99	

Call Transferred Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Call Identity IE	0x10	“other” call ID before transfer
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Call Identity IE	0x10	“resulting” call ID after transfer
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

(Continued on next page)

Call Transferred Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan Id	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Locking Shift to Code Set 6	0x96	
* Old Party Identifier IE	0x17	Up to six old party IDs
* Length of the IE	0x02 or 0x03	
* From Call	0x??	0x80 or 0xc0
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
* The Party Identifier IE	0x44	Up to six party IDs
* Length of the IE	0x01or 0x02	
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Transferred	0x93	

NOTES:

The Call Identity IE containing the “other” call ID before the transfer operation must precede the Call Identity IE that contains the “resulting” call ID after the transfer operation.

There must be the same number of Party ID IEs, Old Party ID IEs, and Connected Number IEs.

The Nth Party ID IE and the Nth Old Party ID IE correspond to the Nth Connected Number IE (and thereby associate an extension with that party).

To preserve a 1:1 correspondence, if the type/address for a party is unknown, a Connected Number IE is present for that party, but the length byte is coded as zero (and no bytes follow).

Charging Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	*0x?*	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	*0x??*	*long form (length > 127)*
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	*0x??*	*long form (length > 127)*
Cause IE	*0x08*	
length of Cause IE	*0x?*	*computed length*
Ext bit | coding standard | location	*0xe1 or 0x81*	*CCITT or ASAI specific, private*
Cause Value	*0x??*	*cause-dependent*
Call Identity IE	0x10	
length of Call Identity IE	0x02	
*Call Identifier (2 octets)	0x??	switch-assigned value
Calling Party Number IE	0x6c	Charging Number
length of Calling Party Number IE	0x??	computed length
Address type & numbering plan	0x??	0x80
* Address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)

(Continued on next page)

Charging Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Called Party Number IE	0x70	
length of Called Party Number IE	0x??	computed length
Address type & numbering plan	0x??	0x80
* Address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code 6	0x96	
Trunk Identification IE	0x0a	Outgoing trunk
length of IE	0x??	computed length
Direction	0x80	No direction
* Trunk Group Number	0x??	binary-encoded value
* Trunk Group Member Number	0x??	binary-encoded value
Party Identifier IE	*0x44*	*party id of outgoing trunk*
Length of Party Identifier IE	*0x01 or 0x02*	
Ext bit | Party ID	*0x??*	
Ext bit | Party ID (second octet)	*0x??*	
Specific Event IE	0x47	
Length of Specific Event IE	0x01	
Event Value	0x1E	Charging Event
Data Item IE	0x52	Charge Information
Length of Data Item IE	0x??	computed length
Data Item type	0x??	type of charge=0x81(intermediate), 0x82 (final), or 0x83 (split)
* Data Item value	0x??	Charge value binary-encoded (0-536,870,880). Can be multiple bytes. Ext bit is set on last byte.

Connected Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
Length of Cause IE	0x02	
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	
The Connected Number IE	0x0c	
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
** Address digits*	0x??	0x30 thru 0x39, 0x23(#), 0x2a(*)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length

(Continued on next page)

Connected Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Address Type & numbering Plan Id	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
*Address digits	0x??	0x30 through 0x39, 0x23(#), 0x2a(*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
Originating Line IE	0x01	
Length of OLI	0x??	computed length
**II-Digits*	0x??	
Trunk Identification IE	0x0a	
Length of the IE	0x??	computed length
Direction	0x80	no direction
** Trunk Group Number*	0x??	binary coded
** Trunk Group Member Number*	0x??	binary coded
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Connected	0x8b	

NOTE:
The Calling Party Number IE and Trunk Identifier IE are mutually exclusive.

Cut-Through Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Progress Indicator	0x1e	
Length of Progress Indicator IE	0x02	
Coding Standard and Location	0x??	0x80 | data from PRI network
Progress description	0x??	0x80 | data from PRI network
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Cut Thru	0x82	

Disconnect/Drop Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
The Connected Number IE	0x0c	party dropped from call
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x80	
** Address digits*	0x??,	0x30 thru 0x39, 0x23(#), 0x2a(*)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	

(Continued on next page)

Disconnect/Drop Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Drop	0x84	

Entered Digits Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 octets)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
User Entered Code IE	0x02	User Entered Digit(s)
Length of IE	0x??	computed value
Type	0xa0	call prompter
collected data indicator	0x110XXXXX	Timeout 00000-11111
* first n-1 digits	0x??	ACSII 0x30-0x39, 0x23(#), 0x2a(*)
* last (nth) digit	0x??	ACSII 0xb0-0xb9, 0xa3(#), 0xaa(*)
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Entered Digits	0x9c	

Hold Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Connected Number IE	0x0c	
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x80	
* Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Hold	0x86	

Login Event Report — Domain (ACD Split/Skill) Control Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Login	0x8e	
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Work Mode	0x86	
The Work Mode	0x??	0x81, 08x2, 0x83, or 0x84

(Continued on next page)

Login Event Report — Domain (ACD Split/Skill) Control Association *(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain IE	0x49	Physical Extension
The length of the Domain IE	0x?	
The Domain type = Extension	0x83	
* The 1st to N-1 ASCII digits	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type=logical agent	0x93	Logical Extension
** first to N-1 digits*	0x??,	ACSII 0x30-0x39, 0x23(#), 0x2a(*)
The last digit	0x??,	ACSII 0xb0-0xb9, 0xa3(#), 0xaa(*)

NOTE:
The last Domain IE listed (Logical Extension) is present only if Expert Agent Selection is used.

Logout Event Report — Domain (ACD Split/Skill) Control Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Logout	0x8f	
The Domain IE	0x49	Physical Extension
The length of the Domain IE	0x?	
The Domain type = Extension	0x83	
* The first to N-1 extension digits	0x??	0x30 thru 0x39
The last extension digit	0x??	0xb0 thru 0xb9

(Continued on next page)

Logout Event Report — Domain (ACD Split/Skill) Control Association *(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain IE	0x49	Logical Extension Note[1]
The length of the Domain IE	0x?	
The Domain type = Logical Agent	0x93	
** The first to N-1 extension digits*	*0x??*	*0x30 thru 0x39*
The last extension digit	*0x??*	*0xb0 thru 0xb9*
The Domain IE	*0x49*	*The Reason Code*
Length of Domain IE	*0x?*	*Computed length*
Domain type=Reason code	*0x98*	*Reason Code* Note[2]
Reason Code	*0x8X*	*0x81 - 0x89 (Reason code 1-9)*

1. The Domain IE listed for Logical Extension is present only if Expert Agent Selection is used.
2. The reason codes Domain IE is present only if the System-Parameter Features field, Logout Reason Codes, is “forced” or “requested” and the agent logs out with a valid reason code (1-9).

Queued Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Counter IE	0x46	
Length of IE	0x02 or 0x03	Counter IE length
Type = Number of Calls in Queue	0x83	
* Value < 127 (NOTE 1)	0x??	XX = 0x80 | # of queued calls

(Continued on next page)

Queued Event Report — *(continued)*		
Byte Description	**Byte Value**	**Comments**
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Queued	0x87	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 extension digits	0x??	0x30 thru 0x39
The last extension digit	0x??	0xb0 thru 0xb9

NOTE:
The value takes up two bytes if the value is greater than 127. The high order bits are first followed by 0x80 | low order bits.

Reconnected Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Connected Number IE	0x0c	
Length of IE	0x?	
Ext bit | Type | Numbering plan	0x80	
* Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Reconnect	0x98	

Reorder/Denial Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Denial	0x8c	

Trunk Seized Event Report

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Report	0x95	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	0x80, 0x90, 0x81, 0x91, 0xa0, or 0xa1
* Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
The Specific Event IE	0x47	
The length of the Specific Event IE	0x01	
Event value = Seized	0x92	

Call Control Capability Group

Call Control: REGister Messages Sent by the Adjunct

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Third Party Make Call Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Make Call	0x83	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan Id	0x??	See Chapter 4
* ASCII Address digits	0xfs,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = Trunk Access Code	0x89	Trunk Access Code/ARS

(Continued on next page)

Third Party Make Call Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	present for direct agent call
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = Number of Rings	0x81	
Value = The Number of Rings	0x8X	0x82 through 0x8f
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = Alerting order	0x82	
Value = Destination first	0x81	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = supervisor assist	0x87	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = direct agent call	0x86	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = priority call	0x83	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = return ack flag	0x8c	
Call Option IE	0x4b	
Length of Call Option IE	0x02	
Option=Answering Machine Detection	0x8e	
Value	0x8x	0x80=Administered Treatment, 0x81=Drop, 0x82=Answer

(Continued on next page)

Third Party Make Call Request — *(continued)*		
Service Circuit IE	0x4e	
Length of Service Circuit IE	0x01	
Tone Detection/Call Classification	0x81	

The switch supports only certain combinations of call options. Not all optional bytes can be present. Appendix A describes how the switch validates the various optional Information Elements that may be present in this request and how the switch determines which of the internal call types it will attempt to establish.

Third Party Take Control Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Third party take control	0xb6	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

Call Control: REGister Messages Sent by the switch

None for Call Control.

Call Control: FACility Messages Sent by the Adjunct

A FACility message is used to initiate an action with an existing association. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

The Call Control message encodings located in this section are:

- Third Party Clear Call Request
- Third Party Selective Drop Request
- Third Party Selective Hold Request
- Third Party Listen-Disconnect Request
- Third Party Listen-Reconnect Request
- Third Party Reconnect Request
- Third Party Merge Request
- Third Party Relinquish Control Request
- Third Party Send DTMF Digits Request
- Redirect Call Request

Third Party Clear Call Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3P Clear Call	0x93	

Third Party Selective Drop Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3P Sel. Drop	0x88	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931.IEs length	0x??	long form (length > 127)
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
Resource Identifier IE	0x51	
Length of IE	0x01	
Resource type	0x82	Tone

Third Party Selective Hold Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Hold	0x85	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

Third Party Reconnect Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Reconnect	0x86	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

Third Party Merge Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Merge	0x89	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	active call
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	party common to both calls
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	
Conference/Transfer IE	0x4a	
Length of the IE	0x01	
Conference/Transfer Option	0x??	0x81=conference, 0x82=transfer

Third Party Relinquish Control Request

The adjunct invokes *Relinquish_control* to terminate a call control association.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Third Party Relinquish Control	0xb5	

Third Party Listen Disconnect Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Selective Disconnect	0xca	selective disconnect of listening path
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931.IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	*The Listener*
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
*Ext bit | Party ID (second octet)	0x??	
**The Party Identifier ID*	*0x44*	*The Talker*
**Length of IE*	*0x01 or 0x02*	
**Ext bit | Party ID*	*0x??*	
**Ext bit | Party ID (second octet)*	*0x??*	

Third Party Listen Reconnect Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Selective Reconnect	0xcb	selective reconnect of listening path
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931.IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	*The Listener*
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
*Ext bit | Party ID (second octet)	0x??	
**The Party Identifier ID*	*0x44*	*The Talker*
**Length of IE*	*0x01 or 0x02*	
**Ext bit | Party ID*	*0x??*	
**Ext bit | Party ID second octet)*	*0x??*	

Third Party Send DTMF Digits Request

The adjunct invokes *Send_DTMF_digits* to request that DTMF tones be sent on a call to selected parties.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Send DTMF Digits	0xc8	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length> 127)
Locking Shift to Code Set 6	0x96	
User Entered Code IE	0x02	User Entered Digit(s)
Length of IE	0x??	calculated length
Type	0x91	Customer-provided
collected data indicator	0xc0	Already Collected
* first n-1 digits	0x??	ACSII 0x30-0x39, 0x23(#), 0x2a(*)
* last (nth) digit	0x??	ACSII 0xb0-0xb9, 0xa3(#), 0xaa(*)
*The Party Identifier IE	0x44	
*Length of the IE	0x01 or 0x02	
*Ext bit | Party ID	0x??	
*Ext bit | Party ID (second octet)	0x??	

(Continued on next page)

Third Party Send DTMF Digits Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = tone duration	0x8a	
Value = tone duration	0x8X	seconds = value * 0.01
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = pause duration	0x8b	
Value = pause duration	0x8X	seconds = value * 0.01

NOTE:
The first Party ID on the list is the tone "sender." The Party ID list may optionally contain up to five additional members (who will hear the tones) on a Call Control Association Send DTMF request.

Redirect Call

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Redirect Call	0xc9	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Redirection Number IE	0x76	
Length of Redirection Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
Party Identifier IE	0x44	
Length of the IE	0x??	computed value
Ext Bit|Party ID	0x??	
Ext Bit|Party ID (second octet)	0x??	

Call Control: FACility Messages Sent by the switch

A FACility message is used to initiate an action within an existing association on an active CRV. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Acknowledgment of Third Party Make Call Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Proceed	0xbd	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Connected Number IE[1]	0x0c	
Length of IE	0x?	
Ext bit | type | Numbering Plan	0x??	0x80 or 0x90
* *ASCII Address digits*	0x??	0x30 through 0x39,0x23(#),0x2a(*)
Call identity IE	0x10	

1. The Connected Number IE is only sent in Third Party Make Call ACKnowledgements for EAS. It is not sent for non-EAS (physical extensions).

(Continued on next page)

Acknowledgment of Third Party Make Call Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Length of Call identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

Acknowledgment of Third Party Take Control Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Third Party Take Control	0xb6	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
* The Party Identifier IE	0x44	Up to six party IDs

(Continued on next page)

Acknowledgment of Third Party Take Control Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

NOTES:

There must be the same number of Party ID IEs as there are Connected Number IEs.

The Nth Party ID IE corresponds to the Nth Connected Number IE (and thereby associates an extension with that party).

To preserve a 1:1 correspondence, if the type/address for a party is unknown, a Connected Number IE is present for that party, but the length byte is coded as zero (and no bytes follow).

Acknowledgment of Third Party Merge Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Merge	0x89	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Call Identity IE	0x10	New Call Identifier for merged call
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	

(Continued on next page)

Acknowledgment of Third Party Merge Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
* Old Party Identifier IE	0x17	Up to six old party IDs
* Length of the IE	0x02 or 0x03	
* From Call	0x??	0x80 or 0xc0
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
* The Party Identifier IE	0x44	
* Length of the IE	0x02 or 0x03	
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	

NOTES:

There must be the same number of Party ID IEs, Old Party ID IEs, and Connected Number IEs.

The Nth Party ID IE and the Nth Old Party ID IE correspond to the Nth Connected Number IE (and thereby associate an extension with that party).

To preserve a 1:1 correspondence, if the type/address for a party is unknown, a Connected Number IE is present for that party, but the length byte is coded as zero (and no bytes follow).

Call Control: Acknowledgment (No Parameters) Association Continues

The switch uses the following message to acknowledge the following requests:

- Third Party Selective Drop Request
- Third Party Selective Hold Request
- Third Party Reconnect Request
- Third Party Listen Disconnect Request
- Third Party Listen Reconnect Request
- Send DTMF Request
- Redirect Call

.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Call Control: Request is Denied — Association Continues

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit |coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent

Call Control: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

See “RELease COMplete Messages Sent by the switch and the Adjunct.”

Call Control: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Call Control: Acknowledgment — Association Terminates

The switch uses the following message to acknowledge the following requests:

- Third Party Relinquish Control Request
- Third Party Clear Call Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Call Control: Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

Call Control: Internal switch Audit Finds Stale Call Control CRV

NOTE:
"Stale" in this context means that while conducting an internal switch audit, the switch has discovered a call control association with no corresponding call and terminates the association.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0xe1	switch Audit Cause
Ext bit | Class | Value in Class	0xd7	switch Audit Cause

Third Party Call Ended — Association Terminates

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3P Call Ended	0xbb	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

Call Control: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Call Control: Normal Clearing Terminates Call Control Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Call Control: A Message is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Call Control: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Call Control: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | Location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Domain Control Capability Group

Domain Control: REGister Messages Sent by the Adjunct

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Domain Control (Station/ACD Split) Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Domain Control	0xc4	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type	0x83 or 0x81	extension or split
* 1st to N-1 ASCII digits	0x??,	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9

Domain Control: REGister Messages Sent by the switch

None for Domain Control.

Domain Control: FACility Messages Sent by the Adjunct

A FACility message is used to initiate an action with an existing association. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

The message encodings located in this section are:

- Third Party Selective Drop Request
- Third Party Selective Hold Request
- Third Party Reconnect Request
- Third Party Merge Request
- Third Party Answer Request
- Third Party Auto Dial Request
- Third Party Cancel Domain (Station/ACD Split)
- Third Party Relinquish Control Request
- Third Party Send DTMF Digits Request
- Redirect Call Request

Third Party (Domain) Selective Drop Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3P Sel. Drop	0x88	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	

Third Party (Domain) Selective Hold Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Hold	0x85	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

Third Party (Domain) Reconnect Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Reconnect	0x86	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	

Third Party (Domain) Merge Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Merge	0x89	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	active call
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Call Identity IE	0x10	other (held) call
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
Conference/Transfer IE	0x4a	
Length of the IE	0x01	
Conference/Transfer Option	0x??	0x81 = conference, 0x82 = transfer

NOTE:
The call_id for the held call, when present, must follow the call_id for the active call.

Third Party (Domain) Answer Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Answer	0x84	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
** Call identifier (2 bytes)*	0x??	switch-assigned value

Third Party Auto Dial Request for an Extension

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Auto Dial	0xc6	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x80	unknown numbering plan
* ASCII Address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = Trunk Access Code	0x89	Trunk Access Code/ARS
*** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39

(Continued on next page)

Third Party Auto Dial Request for an Extension — *(continued)*		
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = priority call	0x83	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = return ack flag	0x8c	

NOTE:
Use of the “return-ack” option is not recommended.

Third Party (Domain) Relinquish Control Request

The adjunct invokes *Relinquish control* to terminate a Domain (Station/ACD Split) Control Request.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Third Party Relinquish Control	0xb5	

Third Party (Domain) Send DTMF Digits Request

The adjunct invokes *Send_DTMF_digits* to request that DTMF tones be sent on a call to selected parties.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Send DTMF Digits	0xc8	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length> 127)
Call Identity IE	0x10	
Length of Call Identity IE	0x02	computed value
*Call identifier (2 octets)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
**The Party Identifier IE*	0x44	Up to 5 receiving party IDs, all optional
**Length of the IE*	0x01 or 0x02	
**Ext bit | Party ID*	0x??	
**Ext bit | Party ID (second octet)*	0x??	

(Continued on next page)

Third Party (Domain) Send DTMF Digits Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
User Entered Code IE	0x02	User Entered Digit(s)
Length of IE	0x??	calculated length
Type	0x91	Customer-provided
collected data indicator	0xc0	Already Collected
* first n-1 digits	0x??	ACSII 0x30-0x39, 0x23(#), 0x2a(*)
* last (nth) digit	0x??	ACSII 0xb0-0xb9, 0xa3(#), 0xaa(*)
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = tone duration	0x8a	
Value = tone duration	0x8X	seconds = value * 0.01
Call Options IE	0x4b	
Length of Call Option IE	0x02	
Option = pause duration	0x8b	
Value = pause duration	0x8X	seconds = value * 0.01

NOTE:
The Party ID list can have up to five Party IDs.

Redirect Call (Domain)

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV Octet	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Redirect Call	0xc9	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity IE	0x02	computed value
* Call identifier (2 octets)	0x??	switch-assigned value
Redirection Number IE	0x76	
Length of Redirection Number IE	0x?	computed length
Address Type & Numbering Plan	0x80	unknown numbering plan
* ASCII address digits	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)

Domain Control: FACility Messages Sent by the switch

A FACility message is used to initiate an action within an existing association on an active CRV. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Acknowledgment of Domain (Station) Control Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	*0x??*	*long form (length > 127)*
operation value tag	0x02	
operation value length	0x01	
Operation Value = Domain Control	0xc4	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	*computed length*
Q.931 IEs length	0x??	*long form (length > 127)*
** Call Identity IE*	0x10	
** Length of Call Identity*	0x02	*computed value*
** * Call identifier (2 bytes)*	0x??	*switch-assigned value*
Locking Shift to Code Set 6	0x96	
** The Party Identifier IE*	0x44	
** Length of the IE*	0x01 or 0x02	
** Ext bit | Party ID*	0x??	
** Ext bit | Party ID (second byte)*	0x??	
* The Specific Event IE	0x47	
* The length of the Specific Event IE	0x01	
* Event/Call State value	0x??	Call state (initiate, alert, held, connected, other/unknown)

NOTES:

All bytes of this message from the sequence tag forward are present if, and only if, there are calls at the domain-controlled station. The presence of such calls results in parameters (call identity, party identity, and state) in this return result component. A return result component only contains a sequence tag and operation value when parameters are present.

There is an association of the Nth call ID, party ID, and specific event. These IEs are ordered accordingly.

Acknowledgment of Third Party Auto Dial Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Proceed	0xbd	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

Acknowledgment of Third Party Merge Request (Domain)

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3rd Party Merge	0x89	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??,	0x30 thru 0x39, 0x23 (#), 0x2a (*)
Call Identity IE	0x10	New Call Identifier for merged call
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
* Old Party Identifier IE	0x17	Up to six old party_id IEs
* Length of the IE	0x02 or 0x03	
* From Call	0x??	

(Continued on next page)

Acknowledgement of Third Party Merge Request — *(continued)*		
Byte Description	**Byte Value**	**Comments**
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	
* The Party Identifier IE	0x44	
* Length of the IE	0x01 or 0x02	
* Ext bit | Party ID	0x??	
** Ext bit | Party ID (second byte)*	0x??	

NOTES:

There must be the same number of Party ID IEs, Old Party ID IEs, and Connected Number IEs.

The Nth Party ID IE and the Nth Old Party ID IE correspond to the Nth Connected Number IE (and thereby associate an extension with that party).

To preserve a 1:1 correspondence, if the type/address for a party is unknown, a Connected Number IE is present for that party, but the address field may not be present.

Domain Control: Acknowledgment (No Parameters) Association Continues

The switch uses the following message to acknowledge the following requests:

- Third Party Selective Drop Request
- Third Party Selective Hold Request
- Third Party Reconnect Request
- Third Party Domain ACD Split Control Request
- Redirect Call
- Third Party Send DTMF

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Domain Control: Request is Denied — Association Continues

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent

Domain Control: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

See “RELease COMplete Messages Sent by Both the switch and the Adjunct.”

Domain Control: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Domain Control: Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	

(Continued on next page)

Byte Description	Byte Value	Comments
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

Domain Control: Acknowledgment — Association Terminates

The switch uses the following message to acknowledge Third Party Relinquish Control (cancel station control).

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

switch Ends Domain (Station) Control Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Domain (Extension) Control Ended	0xc5	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent

Domain Control: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Domain Control: Normal Clearing Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Domain Control: A Message is not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Domain Control: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Domain Control: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Notification Capability Group

Notification: REGister Messages Sent by the Adjunct

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Event Notification Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Request Notification	0x8a	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split, VDN or TAC	0x81 or 0x8c	0x81=split, 0x8c=VDN, 0x89=TAC
* The first to N-1 extension digits	0x3X	0x30 thru 0x39, 0x23 (#)
The last extension digit	0xbX	0xb0 thru 0xb9, 0xa3 (#)

Notification: REGister Messages Sent by the switch

None for this capability.

Notification: FACility Messages Sent by the Adjunct

A FACility message is used to initiate an action with an existing association. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Cancel Event Notification Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Event Notification Cancel	0x94	

Stop Notification on Call Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Stop Notification	0xc7	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

Notification: FACility Messages Sent by the switch

A FACility message is used to initiate an action within an existing association on an active CRV. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Notification: Acknowledgement (No Parameters) Association Continues

The switch uses the following message to acknowledge the following requests:

- Event Notification Request
- Stop Call Notification Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Notification: Call Ended — Association Continues

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = 3P Call Ended	0xbb	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

Notification: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

See “RELease COMplete Messages Sent by the switch and the Adjunct.”

Notification: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Notification: Acknowledgement (No Parameters) Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Notification: Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit| coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

switch Ends Notification Reporting Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Notification Ended	0xbc	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit| coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent

Notification: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Notification: Terminate ASAI Association with Normal Clearing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Notification: A Message is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Notification: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Notification: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Routing Capability Group

Routing: REGister Messages Sent by the switch

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Call Route Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Route	0xb0	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value

(Continued on next page)

switch Requests Call Route from Adjunct — *(continued)*		
Byte Description	**Byte Value**	**Comments**
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x??	See Chapter 4
** Address digits*	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a(*)
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	*0x??*	
Locking Shift to Code Set 6	0x96	
Originating Line IE	0x01	
Length of OLI	0x??	computed length
** II-Digits*	0x??	
User Code IE	0x02	User Entered Data IE
length of user data	0x??	computed length
type of user data	0x85	Call Prompter Data
collected data indicator	0xc0	
** first n-1 user entered digits*	0x00 | 0x??	ASCII data
last user entered digit	0x80 | 0x??	ASCII data
The Trunk Identification IE	0x0a	
The length of IE	0x	computed length
Direction	0x80	No Direction
**Trunk Group Number*	0x??	binary coded
**Trunk Group Member Number*	0x??	binary coded
The Feature IE	*0x48*	
The length of the feature IE	*0x01*	
Feature	*0x86*	Flexible billing

(Continued on next page)

switch Requests Call Route from Adjunct — *(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = VDN	0x8c	Vector Directory Number
*The 1st to N-1 ASCII digits	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
Lookahead Interflow IE	0x7b	PRI Interflow
length of lookahead interflow	0x??	computed length
Ext bit | Priority Level & Type	0x??	0x80 | data from PRI
Ext bit | Time Stamp - hours	0x??	0x80 | data from PRI
Ext bit | Time Stamp - minutes	0x??	0x80 | data from PRI
Ext bit | Time Stamp - seconds	0x??	0x80 | data from PRI
**Ext bit | first n-1 ASCII chars*	0x??	0x80 | ASCII data from PRI
Ext bit | last ASCII char	0x??	0x80 | ASCII data from PRI

NOTE:
The Calling Number IE and the Trunk Identification IE are mutually exclusive. One or the other will be present.

Routing: FACility Messages Sent by the Adjunct

A FACility message is used to initiate an action with an existing association. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Call Route Selection

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Route Select	0xb7	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x??	See Chapter 4
** Address digits*	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)

(Continued on next page)

Adjunct Call Routing Selection —*(continued)*		
Byte Description	**Byte Value**	**Comments**
User-User IE	0x7e	User-to-user information
length of user-user	0x??	computed length
protocol discriminator	0x??	
**user information*	0x??	
Locking Shift to Code Set 6 (Note 1)	0x96	
User Entered Code IE (Note 2)	*0x02*	*ASAI-provided digits*
length of User Entered IE	*0x??*	*computed length*
type of user data	*0x91*	*customer database provided*
collect data indicator	*0xc0*	*collected indication*
**first (n-1) digits*	*0x??*	*ASCII 0x30-0x39,0x23(#),0x2a(*)*
**last (nth) digit*	*0x??*	*ASCII 0xb0-0xb9,0xa3(#),0xaa(*)*
User Entered Code IE (Note 2)	*0x02*	*ASAI-requested digit collection*
length of user data	*0x??*	*computed length*
type of user data	*0xa0*	*Tone Detector*
collect data indicator	*bx100XXXXX*	*Timeout 00000 - 11111*
Number of digits	*0x??*	*0x81 -0x98*
The Party Identifier IE (Note 3)	*0x44*	*Party on the call the Tone Detector should listen to*
The length of the IE	*0x01 or 0x02*	
Ext bit | Party ID	*0x??*	
Ext bit | Party ID (second octet)	*0x??*	
The Specific Event IE (Note 4)	*0x47*	
The length of the Specific Event IE	*0x01*	
Event value = Connect or Drop	*0x??*	0x8b(Conn), 0x84(Drop)
The Domain IE	0x49	present for direct agent call
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	

(Continued on next page)

Adjunct Call Routing Selection —*(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain type = Trunk Access Code	0x89	Trunk Access Code/ARS
** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = direct agent call	0x86	
Call Options IE	0x4b	
Length of Call Option IE	0x01	
Option = priority call	0x83	

NOTES:

This locking shift must be present if any of the optional IEs following it are present; otherwise, it must be absent.

The ASAI application may either “collect digits” or “supply digits,” but not both. Thus, only one User Entered Code IE is permitted in the message.

Not currently used (ignored).

The Specific Event IE is present only when the application includes the User Code IE to “collect digits.”

Routing: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Routing: Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x8N or 0xeN	N = 0 or 1, direction-dependent
Cause Value	0x??	cause-dependent

Routing: Endpoint Aborts an Association

Routing: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

End Adjunct Routing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiator — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Route End	0xb9	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	switch-to-adjunct
Cause Value	0x??	cause-dependent

The optional cause, when present, is cause #102 (Recovery on Timer Expiry). This is sent when Vector Processing continues for a call without the switch having received a route.

Routing: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Routing: Terminate ASAI Association With Normal Clearing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Routing: A Message Is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe	
Cause Value	0x??	cause-dependent

Routing: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Routing: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Request Feature Capability Group

Request Feature: REGister Messages Sent by the Adjunct

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Agent Login Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Feature Request	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
User Code IE	0x02	
Length of User Code IE	0x?	computed length
Type of user Code = Login Digits	0x82	
collected data indicator	0xc0	

(Continued on next page)

Agent Login Request —*(continued)*		
Byte Description	**Byte Value**	**Comments**
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | The last ASCII digit	0xbX	0xb0 through 0xb9
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Login	0x82	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	computed length
The Domain type = Extension	0x83	Agent Extension
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | last ACSII digit	0xbX	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Work Mode	0x86	
The Work Mode	0x8X	0x81, 0x82, 0x83, or 0x84

Agent Logout Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Feature Request Operation Value	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Logout	0x83	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	computed length
The Domain type = Extension	0x83	Agent Extension
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | last ACSII digit	0xbX	0xb0 through 0xb9

(Continued on next page)

Agent Logout Request —*(continued)*		
Byte Description	**Byte Value**	**Comments**
The length of the Domain IE	0x?	computed length
The Domain type = Extension	0x83	Agent Extension
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | last ACSII digit	0xbX	0xb0 through 0xb9
Domain IE	*0x49*	*Specifies* *Reason Code for Logout*
Length of Domain IE	*0x?*	*Computed length*
Domain type=Reason code	*0x98*	*Reason Codes*
Reason Code	*0x8X*	*0x81 - 0x89* *(Reason codes 1-9)*

Change Agent Work Mode Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Feature Request Operation Value	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Change in Work Modes	0x81	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	computed length

(Continued on next page)

Change Agent Work Mode Request —*(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain type = Extension	0x83	Agent Extension
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | last ACSII digit	0xbX	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Work Mode	0x86	
The Work Mode	0x??	0x81, 08x2, 0x83, or 0x84
Domain IE	*0x49*	*Reason Code (if change to AUX)*
Length of Domain IE	*0x?*	*Computed length*
Domain type=Reason code	*0x98*	*Reason Codes*
Reason Code	*0x8X*	*0x81 - 0x89 (Reason codes 1-9)*

Send All Calls Feature Activation

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Feature Request	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Redirecting Number IE	0x74	
Length of Redirecting Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Send All Calls	0x8c	

Cancel Send All Calls Feature Activation

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Feature Request	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Redirecting Number IE	0x74	
Length of Redirecting Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Cancel Send All Calls	0x8d	

Call Forwarding Feature Activation

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Feature Request	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Redirecting Number IE	0x74	
Length of Redirecting Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Redirection Number IE	0x76	
Length of Redirection Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Forward Calls	0x8a	

Cancel Call Forwarding Feature Activation

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Feature Request	0x8f	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Redirecting Number IE	0x74	
Length of Redirecting Number IE	0x?	computed length
Address Type & Numbering Plan	0x??	See Chapter 4
* ASCII address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Cancel Forwarding	0x8b	

Request Feature: REGister Messages Sent by switch

None for this capability group.

Request Feature: FACility Messages Sent by the Adjunct

None for this capability group.

Request Feature: FACility Messages Sent by switch

None for this capability group.

Request Feature: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

See “RELease COMplete Messages Sent by the switch and the Adjunct.”

Request Feature: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

Request Feature: Acknowledgment — Association Terminates

The switch uses the following message to acknowledge Request Feature requests.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Request Feature: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Request Feature: Terminate ASAI Association With Normal Clearing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Request Feature: A Message is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Request Feature: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Request Feature: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV Byte	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Value Query Capability Group

Value Query: REGister Messages Sent by the Adjunct

The REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x6.

ACD Split Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9

Agent Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9
The Domain IE	0x49	
The length of the Domain IE	0x?	computed length
The Domain type = Extension	0x83	Agent Extension
* The 1st to N-1 ASCII digits	0x3X	0x30 through 0x39
Ext bit | last ASCII digit	0xbX	0xb0 through 0xb9

Station Feature Query — Message Waiting Lamp

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Item IE	0x4d	
Length of the IE	0x01	
Item = MWL Status	0x83	

Station Feature Query — Send All Calls

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV Byte	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IEs length	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Send-All-Calls	0x8c	

Station Feature Query — Call Forwarding

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Feature IE	0x48	
The length of the feature IE	0x01	
Feature = Forward Calls	0x8a	

Station Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	Extension
The length of the Domain IE	0x?	
The Domain type = Extension	0x83	
* The 1st to N-1 ASCII digits	0x??	0x30 through 0x39
Ext bit | last ASCII digit	0x??	0xb0 through 0xb9

Trunk Group Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = Trunk Access Code	0x89	Trunk Access Code/ARS
** The 1st to N-1 ASCII digits*	0x??	0x30 through 0x39
Ext bit | last ASCII *digit*	0x??	0xb0 through 0xb9

Call Classifiers Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Resource Identifier IE [1]	0x51	Call Classifiers
The length of the Resource IE	0x01	computed length
Resource type = Call Classifier	0x81	

1. If the **USA default algorithm** field is set to *n* on the System Parameters OCM Call Classification form, then the query will provide the total number of ports for TN744D and TN2182 boards (and later versions) that are idle and in-use.

ACD Agent Login Audit Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = ACD Split	0x81	ACD split
* The first to N-1 ASCII digits	0x??	0x30 through 0x39
Exit bit | last ASCII digit	0x??	0xb0 through 0xb9
The Item IE	0x4d	
Length of the IE	0x01	
Item = Login Audit	0x8d	

Date/Time Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Item IE	0x4d	
Length of the IE	0x01	
Item = Date/Time	0x81	

Call Information at Station Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
The Domain type = extension	0x83	Extension
* 1st to N-1 ASCII digits	0x??,	0x30 through 0x39
Ext bit | The last ASCII digit	0x??,	0xb0 through 0xb9
The Item IE	0x4d	
Length of the IE	0x01	
Item = Call Query	0x9c	

Party ID Information on Call Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call Identity	0x02	computed value
* Call identifier (2 bytes)	0x??	switch-assigned value
Locking Shift to Code Set 6	0x96	
The Item IE	0x4d	
Length of the IE	0x01	
Item = Party ID	0x9b	

Extension Type/Class Information Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Calling Party Number	0x6c	
Length of Calling Party IE	0x?	computed length
Address Type & numbering Plan ID	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Item IE	0x4d	
Length of the IE	0x01	
Item = Extension Information	0x9d	

Integrated Directory Database Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	assigned value
Second CRV *Octet*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Domain IE	0x49	Integrated Directory Query
Length of Domain IE	0x01	
Integrated Directory Domain	0x95	
Domain IE	0x49	Search Extension
length of Domain IE	0x??	
Domain Type	0x??	See Note
* 1st to (n-1)st digits	0x3?	0x30-0x39
Ext bit | last ASCII digit	0xb?	0xb0-0xb9

Note: Valid values are:
0x81 (group extension)
0x83 (station extension)
0x89 (trunk access code)
0x8c (vector directory number)
0x8e (announcement extension)
0x8f (data extension)
0x93 (split of skill hunt group extension

Value Query: REGister Messages Sent by the switch

None for this capability group.

Value Query: FACility Messages Sent by the switch

A FACility message is used to initiate an action within an existing association on an active CRV. All FACility messages have the Protocol Discriminator and the CRV followed by a FAC type = 0x62.

Response to Agent Login Query (List of Logged-in Agent Extensions)

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x62	FACility
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	even value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Response Continued	0xbf	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
* The Domain IE	0x49	Agent Ext. repeated up to eight times
* The length of the Domain IE	0x?	
* The Domain type = Extension	0x83	
* * 1st to N-1 ASCII digits	0x??,	0x30 through 0x39
* Ext bit | last extension digit	0x??,	0xb0 through 0xb9

Value Query: RELease COMplete Messages Sent by the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

See “RELease COMplete Messages Sent by the switch and the Adjunct.”

Value Query: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Response to ACD Split Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Counter IE	0x46	
Length of IE	0x02 or 0x03	
Type = Number of Calls in Queue	0x83	
* Ext bit | Value < 127 (NOTE 1)	0x??	0x80 | # of queued calls
The Counter IE	0x46	
Length of IE	0x02 or 0x03	
Type = Number of Logged in Agents	0x84	
* Ext bit | Value < 127 (NOTE 1)	0x??	0x80 | # logged in agents

(Continued on next page)

Response to ACD Split Status Query —*(continued)*		
Byte Description	**Byte Value**	**Comments**
The Counter IE	0x46	
Length of IE	0x02 or 0x03	
Type = Number of Available Agents	0x85	
* Ext bit | Value < 127 (NOTE 1)	0x??	0x80 | # avail. agents

The value will take up two bytes if the value is greater than 127. The high order bits will be first followed by 0x80 | low order bits.

Response to Agent Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Work Mode	0x86	
The Work Mode	0x??	0x81, 0x82, 0x83, or 0x84
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Talk State	0x87	
The Talk State	0x??	0x85 or 0x86

(Continued on next page)

Response to Agent Status Query —*(continued)*		
Byte Description	**Byte Value**	**Comments**
The Domain IE	*0x49*	Note
Length of Domain IE	*0x?*	*Computed length*
Domain type = Reason code	*0x98*	*Reason code*
Reason code	*0x8X*	*0x81- 0x89* *(Reason codes 1-9)*

NOTE:
This IE will only be present if the System-Parameters Feature field, AUX Reason Code, is “forced” or “requested,” and the agent’s work mode is AUX.

Response to Station Message Waiting Lamp Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Status IE	0x4f	
length of status IE	0x02	computed length
MWI Status	0x82	
Ext bit | Mesg Serv status	0x??	0x80 | services status bits

NOTE:
Bits 1 through 5 in byte 4 of the Status IE give the MWL on/off status of each switch messaging service. The adjunct should logically OR these bits. If the result is “1,” the lamp is on; otherwise, it is off. The adjunct must logically OR bits 1 through 5.

Response to Station Send-All-Calls Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Status IE	0x4f	
length of status IE	0x02	computed length
Send-All-Calls status type	0x84	
Send-All-Calls status	0x80 or 0x90	0x80=off, 0x90=on

Response to Station Call Forwarding Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Redirection Number IE	0x76	
Length of Redirection Number IE	0x?	computed length
Address Type & Numbering Plan	0x80	unknown numbering plan
* ASCII *address digits*	0x??	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
Status IE	0x4f	
length of status IE	0x02	computed length
Call Forward status type	0x88	
Call Forwarding status	0x80 or 0x90	0x80=off, 0x90=on

Response to Station Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x02	
The Domain type = Talk State	0x87	
The Talk State	0x??	0x85 or 0x86

Response to Trunk Status Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Trunk Group Status IE	0x0b	
Length of Trunk Group Status IE	0x??	computed length
Counts reported	0x??	
Trunks in use	0x??	present if corresponding count bit set
Trunks idle	0x??	present if corresponding count bit set

NOTE:
Trunks idle (available) + Trunks in use = trunks in service. The switch does not return counts for maintenance busy and otherwise unavailable.

Response to Number of Call Classifiers Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Resource Status IE	0x03	
Length of Status IE	0x??	computed length
Type of Resource[1]	0x81	call classifier resource
Counts reported	0X??	
Classifiers in use	0x??	present if corresponding count bit set
Classifiers idle	0x??	present if corresponding count bit set

1. If the **USA default algorithm** field is set to *n* on the System Parameters OCM Call Classfication form, then the query will provide the total number of ports for TN744D and TN2182 boards (and later versions) that are idle and in-use.

Response to Call Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	*0x??*	long form (length > 127)
operation value tag	*0x02*	
operation value length	*0x01*	
Operation Value = Value Query	*0x8c*	
Q.931 IEs tag	*0x40*	
Q.931 IEs length	*0x??*	computed length
Q.931 IEs length	*0x??*	long form (length > 127)
** Call Identity IE*	*0x10*	
** Length of Call Identity*	*0x02*	computed value
** * Call identifier (2 bytes)*	*0x??*	switch-assigned value
Locking Shift to Code Set 6	*0x96*	
** The Party Identifier IE*	*0x44*	
** Length of the IE*	*0x01 or 0x02*	
** Ext bit | Party ID*	*0x??*	
** Ext bit | Party ID (second byte)*	*0x??*	
** The Specific Event IE*	0x47	
** The length of the Specific Event IE*	0x01	
** Event / Call State value*	0x??	Call state (initiate, alert, held, connected, other)

NOTES:

All bytes of this message from the sequence tag forward are present if, and only if, there are calls at the station for which the query was made. The presence of such calls results in parameters (call identity, party identity, and state) in this return result component. A return result component only contains a sequence tag and operation value when parameters are present.

There is an association of the Nth call ID, party ID, and specific event. These IEs are ordered accordingly.

Bytes after the invoke-id are present only when calls are present at the station for which the query was made.

Response to Party ID Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
* The Connected Number IE	0x0c	
* Length of IE	0x?	
* Ext bit | Type | Numbering plan	0x80	
* * Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
The Party Identifier IE	0x44	
Length of the IE	0x01 or 0x02	

(Continued on next page)

Response to Party ID Query —*(continued)*		
Byte Description	**Byte Value**	**Comments**
Ext bit | Party ID	0x??	
Ext bit | Party ID (second byte)	0x??	

NOTE:
There is a correspondence between the Nth party ID IE and the Nth Connected Number IE.

Response to Extension Information Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
The Domain IE	0x49	
The length of the Domain IE	0x?	
Domain type (NOTE 1)	0x??	e.g., station type, ASAI, logical agent
Domain Address Field	0x??	domain-type dependent

NOTE:
If the domain type is logical agent and if the agent is logged in, there will be an Address Field containing the physical extension the agent is logged into. If the domain type is either ASAI or station type, there will be an Address Field containing additional information. There is no Address Field for other domain types.

Response to Date/Time Query

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
The Date/Time IE	0x29	
Length of the IE	0x06	
Year	X	X = Current year
Month	X	X = Current month
Day	X	X = Current day
Hour	X	X = Current hour
Minute	X	X = Current minute
Second	X	X = Current second

Response to Integrated Directory Query

Byte Description	Byte Value	Comments
Octet Description	Octet Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Octet	0x?	
Second CRV *Octet*	0x?	
Message Type	0x5a	RELease COMPlete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
Sequence Tag	0x30	Q.932 sequence tag
Sequence Length	0x??	computed length
Sequence Length	0x??	long form (length > 127)
operation value tag	0x02	
operation value length	0x01	
Operation Value = Value Query	0x8c	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Locking Shift to Code Set 6	0x96	
Domain IE	0x49	Actual type of Queried Device
length of Domain IE	0x??	computed value
Domain Type	0x??	0X81, 0x83, 0x89, 0x8C, 0x8F, 0x92, 0x93
* the 1st to N-1 ASCII digits	0x3x	0x30-0x39
Ext bit | last ASCII digit	0xbx	0xb0-0xb9
Data Item IE	0x52	Contains ASCII name
length of Data Item IE	0x??	computed value
Data Item Type	0x84	Directory Name
* 1st to (n-1)st ASCII characters	0x??	0x00 | ASCII character
last ASCII character	0x??	0x80 | ASCII character

NOTE:
If an application makes an integrated directory query for a number for which the switch does not have an administered name, the response will not contain the Sequence tag or any following octets.

Value Query: Acknowledgment — Association Terminates

The switch uses the following message to acknowledge Value Queries (final response to Login Audits).

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Value Query: RELease COMplete Messages Sent by the switch

Value Query: Request is Denied — Association Terminated

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

Value Query: Endpoint Aborts an Association

Value Query: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Value Query: Terminate ASAI Association With Normal Clearing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Value Query: A Message is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Value Query: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Value Query: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Set Value Capability Group

Set Value: REGister Messages Sent by the Adjunct

A REGister message initiates an association on a CRV. All REGister messages have the Protocol Discriminator and the CRV followed by a REG type = 0x64.

Activate Message Waiting Lamp Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Set Value Operation Value	0x8d	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
Item IE	0x4d	
Item length	0x01	computed length
MWL Item	0x83	MWL
Status IE	0x4f	
length of status IE	0x02	computed length
MWL Status	0x82	
MWL on	0x90	

Deactivate Message Waiting Lamp Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Set Value Operation Value	0x8d	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Called Party Number IE	0x70	
Length of Called Party Number	0x?	computed length
Address Type & Numbering Plan	0x80	unknown numbering plan
* ASCII Address digits	0x??,	0x30 through 0x39, 0x23 (#), 0x2a (*)
Locking Shift to Code Set 6	0x96	
Item IE	0x4d	
Item length	0x01	computed length
MWL Item	0x83	MWL
Status IE	0x4f	
length of status IE	0x02	computed length
MWL Status	0x82	
MWL off	0x80	

Billing Change Request

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Set Value operation value	0x8d	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Call Identity IE	0x10	
Length of Call identity	0x02	
* Call identifier (2 octets)	0x??	switch assigned value
Locking Shift to Code Set 6	0x96	
Item IE	0x4d	
Item length	0x01	computed length
Billing change request item	0x9f	Billing change request
Generic Billing Data IE	0x56	
Length of Generic Billing Data	0x??	computed length

(Continued on next page)

Billing Change Request —*(continued)*		
Byte Description	**Byte Value**	**Comments**
Type of Billing Data (NOTE 1)	0x??	0x90, 0x91, 0x92, 0x93 or 0x98
Encoding scheme	0x82	IA5 (optional for type 0x98)
Billing data - hundreds	0x00	$x00.00, not included for type 0x98
Billing data - tens	0x00	$0x0.00, not included for type 0x98
Billing data - units	0x00	$00x.00, not included for type 0x98
Billing data - tenths	0x00	$000.x0, not included for type 0x98
Billing data - hundredths	0x00	$000.0x, not included for type 0x98

NOTE:
0x90=new, 0x91=flat, 0x92=Premium charge, 0x93=Premium credit, 0x98=Free Call.

Set Value: REGister Messages Sent by the switch

None for this capability group.

Set Value: FACility Messages Sent by the Adjunct

None for this capability group.

Set Value: FACility Messages Sent by the switch

None for this capability group.

Set Value: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	return error
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request
error value tag	0x02	
error value length	0x01	
Operation Value	0x??	Operation Value of the denied request
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x81 or 0xe1	
Cause Value	0x??	cause-dependent

Set Value: RELease COMplete Messages Sent by the switch

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Set Value: Acknowledgment — Association Terminates

The switch uses the following message to acknowledge Set Value Requests (MWL On, MWL Off).

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Set Value: RELease COMplete Messages Sent by Both the switch and the Adjunct

A RELease COMplete message terminates an association on an existing CRV. All RELease COMplete messages have the Protocol Discriminator and the CRV followed by a REL COM type = 0x5a.

Set Value: Terminate ASAI Association With Normal Clearing

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
The Cause IE	0x08	
Length of IE	0x02	
Ext bit | coding standard | location	0x80 or 0x81	
Ext bit | Class | Value in Class	0x90	Normal Clearing

Set Value: A Message Is Not Understood — Association Aborted

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
Cause IE	0x08	
length of Cause IE	0x02	computed length
Ext bit | coding standard | location	0x80, 0x81, 0xe0, 0xe1	
Cause Value	0x??	cause-dependent

Set Value: Endpoint Rejects an Invalid/Protocol Violation FIE — Terminates Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa4	REJECT
component length	0x??	computed length
invoke identifier tag	0x02	
invoke identifier length	0x01	0x0 if null invoke-id
invoke identifier	0x??	active invoke-id; omitted if null
Problem tag	0x??	Problem-specific
Problem length	0x01	
Problem	0x??	Problem-specific

Set Value: Request is Denied — Association Terminated

Set Value: Endpoint Aborts an Association

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Bytes	0x?	
Second CRV *Bytes*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa3	Return error
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	
operation value tag	0x02	
operation value length	0x01	
Operation Value = Abort	0xbe	
Q.931 IEs tag	0x40	
Q.931 IEs length	0x??	computed length
Q.931 IE*s length*	0x??	long form (length > 127)
The Cause IE	0x08	NOTE 1
Length of IE	0x02	
Ext bit | coding standard | location	0x??	cause- and direction-specific
Ext bit | Class | Value in Class	0x??	cause-specific

NOTE:
The switch always supplies a cause value when it aborts an association; the adjunct may optionally supply a cause value.

Maintenance

Maintenance Messages Sent by Both the switch and the Adjunct

RESTart an ASAI Interface

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	global CRV
CRV Value	0x00	
Second CRV *Value Byte*	0x00	
Message Type	0x46	RESTart
Restart Indicator IE	0x79	
length of Restart Ind.	0x01	
all interfaces	0x87	
Locking Shift to Code Set 6[1]	0x96	
**Version* IE	0x1b	
**Length of Version* IE	0x01	
**ASAI Version*	0x??	

1. This locking shift must be present when the Version IE is present; otherwise, it is omitted.

Acknowledge Restart of an ASAI Interface

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	global CRV
CRV Value	0x80	flag bit set with zero value
Second CRV *Value Byte*	0x00	
Message Type	0x4e	RESTart ACKnowledge
Restart Indicator IE	0x79	
length of Restart Ind.	0x01	
all interfaces	0x87	
Locking Shift to Code Set 6[1]	0x96	
Version IE	0x1b	
Length of Version IE	0x01	
ASAI Version	0x??	

1. This locking shift must be present when the Version IE is present; otherwise, it is omitted.

Heartbeat

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	assigned value
Second CRV *Byte*	0x?	assigned value
Message Type	0x64	REGister
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag = invoke	0xa1	INVOKE component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	initiating — odd value
operation value tag	0x02	
operation value length	0x01	
Operation Value = Heartbeat	0xb3	

Response to Heartbeat

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	
First CRV Byte	0x?	
Second CRV *Byte*	0x?	
Message Type	0x5a	RELease COMplete
Locking Shift to Code Set 6	0x96	
Facility IE	0x1c	
Length of Facility IE	0x??	computed length
Q.932 Supplementary Service	0x91	
component type tag	0xa2	RETURN RESULT component
component length	0x??	computed length
component length	0x??	long form (length > 127)
invoke identifier tag	0x02	
invoke identifier length	0x01	
invoke identifier	0x??	value from request

Maintenance Messages Sent by the Adjunct

Suspend/Resume Alarming for ASAI Interface

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	global CRV
CRV Value	0x00	zero value
Second CRV *Value Byte*	0x00	
Message Type	0x00	MIM — first byte
Message Type	0xf7	MIM — second byte
Locking Shift to Code Set 6	0x96	
MIE	0x7a	
MIE length	0x07	
MIE Prot Disc	0x03	
MIE Trans Ref	0x80	synchronous MIM operation
MIE Op class & Type	0x90	class = confirmed, type = action
MIE Operation	0x82	Link Alarm Status Change
MIE Parameter	0xd3	Alarm Status Parameter
parameter length	0x01	
new service state	0x04 or 0x03	0x04 = suspend, 0x03 = resume

Maintenance Messages Sent by switch

Acknowledge Suspend/Resume Alarming for ASAI Interface

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	global CRV
CRV Value	0x80	flag bit set with zero value
Second CRV *Value Byte*	0x00	
Message Type	0x00	MIM — first byte
Message Type	0xf7	MIM — second byte
Locking Shift to Code Set 6	0x96	
MIE	0x7a	
MIE length	0x04	
MIE Prot Disc	0x03	
MIE Trans Ref	0x80	response to synchronous MIM operation
MIE Op class & Type	0xa0	class = ret. reslt., type = action
MIE Operation	0x82	Link Alarm Status Change

Reject MIM Message

Byte Description	Byte Value	Comments
Protocol Discriminator	0x08	BRI
CRV Length	0x01 or 0x02	global CRV
CRV Value	0x80	flag bit set with zero value
Second CRV *Value Byte*	0x00	
Message Type	0x00	MIM — first byte
Message Type	0xf7	MIM — second byte
Locking Shift to Code Set 6	0x96	
MIE	0x7a	
MIE length	0x04	
MIE Prot Disc	0x03	
MIE Trans Ref	0x??	taken from request
MIE Op class & Type	0xd0	class = reject
MIE Operation	0x81, 8x82	Management Error Code

Maintenance

6

The ECS Support for BRI Endpoints

The ECS requires all ASAI interfaces be administered as fixed or automatic Terminal Endpoint Identifier (TEI), point-to-point interfaces. Although the ECS also supports multipoint for BRI station sets, an ASAI interface is more restricted.

This restriction means an adjunct:

- Must support either fixed or automatic TEI, but does not have to support both
- Does not need to support layer 3 SPID initialization
- Only needs to support a restricted set of MIMs (used for Suspend/Resume alarms)

The ECS does not provide the following features for ASAI adjuncts:

- U-interface endpoints
- Short passive bus configuration
- Noninitializing endpoints
- The use of S and Q bits and their associated functions at layer 1

ASAI Endpoint Administration

To administer an ASAI interface, the system administrator enters the **add station** command. On the station form, the administrator first sets the `station type` to `ASAI`. The administrator must MANUALLY administer the ASAI interface as a fixed or automatic TEI, point-to-point link.

- `Fixed TEI` defaults to `NO` for ASAI station type. The administrator should leave this set to `NO` for an automatic TEI ASAI endpoint or change it to `YES` for an ASAI endpoint that supports fixed TEI assignment.
- `TEI` pops up when `Fixed TEI` is set to `YES`. A TEI in the range 0 to 63 must be entered. This must correspond to the TEI that the adjunct BRI interface uses.
- `MIM Support` defaults to `NO` for ASAI station type. This does not allow access to fields for `Endpoint Initialization, SPID,` and `MIMs for Maintenance/Management.`
- `XID Testing` may be enabled or disabled depending on the adjunct. Set the `XID Testing` field to `YES` when the ASAI host supports the LAPD XID inquiry message exchange. Lucent Technologies recommends including support of the XID test in any ASAI adjunct whenever possible. The `XID Testing` field defaults to `YES` for ASAI station type.

If an administrator attempts to create more than eight ASAI interfaces, the ECS responds with an error message and prevents the administration of more than eight ASAI links.

For more information, see the *DEFINITY Communications System Generic 1 and Generic 3 Feature Description,* 555-230-201, or the *DEFINITY Enterprise Communications Server Release 5 Feature Description,* 555-230-301.

ASAI Link Alarming

The ASAI Adjunct Alarm Administration Feature permits a customer to tune the level of alarm reporting for ASAI adjuncts. Using the feature, the customer can administer on-board and off-board alarms for ASAI ports and/or endpoints to raise warnings or minor or major alarms. Once an item is administered to raise a certain level of alarm, it alarms all ASAI links at that level.

The system default is to provide warnings. When an adjunct or adjunct link is being taken out of service for planned maintenance, it is not desirable for the ECS to generate alarms when the link drops. For this reason, the ECS supports a procedure where an adjunct should suspend and resume alarming on a specific link. Suspending alarms stops the ECS from generating alarms for that link. Resuming alarms causes the ECS to generate alarms at the administered level.

It is recommended that the Resume Alarms and Suspend Alarm capabilities be incorporated into any adjunct ‘‘startup’’ and ‘‘shutdown’’ operations for ASAI links.

ASAI Heartbeat

The ECS periodically sends ASAI Heartbeat messages to the adjunct. The adjunct must respond or the ECS maintenance subsystem takes action to tear down the link and re-establish it. It is recommended that any ASAI endpoint respond to an ASAI Heartbeat message as soon as possible after it is received.

The ECS similarly responds to any ASAI Heartbeat message that it receives from an adjunct so that an adjunct maintenance software subsystem can use the ASAI Heartbeat capability to monitor the health of the link.

ASAI Interactions with System Restarts

Reboot, COLD1, COLD2 System Restarts — Cause the BRI port board to be reset and thus cause the BRI signaling channels to be disconnected. Any ASAI associations are terminated.

System Warmstart, Hot Start — cause a layer 2 drop which in turn can lead to the adjunct clearing associations or restarting layer 3.

Management Information Messages

The ECS uses BRI Management Information Messages to suspend and/or resume alarms on an ASAI interface.

If the ECS receives any other message with a Management Protocol Discriminator on an ASAI interface, the message is ignored.

Temporary Layer 2 Drop

The BRI specification contains a procedure for a temporary loss of Link Access Protocol for D Channel (LAPD) connectivity. The procedure states that when connectivity is lost at layer 2, the user endpoint (for example, the adjunct) must clear all CRVs except those that are:

1. In the *active* state, AND
2. Connected to a B-channel

Since ASAI CRVs are never associated with a B-channel, the adjunct must immediately clear all CRVs associated with ASAI associations when layer 2 connectivity is lost.

The procedure further states that if layer 2 connectivity is reestablished before a specified timer expires, the user endpoint (for example, the adjunct) sends STATUS messages only for CRVs that were not cleared as stated above. Thus, neither the adjunct nor the ECS may send STATUS messages for ASAI CRVs.

Layer 3

Layer 3 RESTart AND RESTart ACKnowledgement Messages

Both the ECS and adjunct must adhere to the RESTart Procedure on an ECS ASAI interface. In the BRI specification, the procedure is optional in the user-to-network direction, but it is mandatory for an ASAI.

Layer 3 STATUS and STATUS ENQuiry Messages

Neither the ECS nor the adjunct sends the STATUS ENQuiry message on an ASAI interface.

BRI permits any endpoint to send a STATUS message whenever it receives an unexpected message. The ECS does not do so on a BRI ASAI interface. If the adjunct sends a STATUS message to the ECS, on a BRI ASAI interface, the ECS discards the STATUS message.

If the adjunct sends a STATUS ENQuiry message to the ECS, on a BRI ASAI interface, the ECS treats the STATUS ENQuiry as an unexpected message and responds as Chapter 5 describes.

Layer 3 Timers

The only layer 3 BRI Timer that the ECS ASAI uses is T316, the retry timer for RESTart. T316 has a value of 120 seconds. None of the BRI Management timers are used.

The ECS adjunct routing application also makes use of an application-level timer. Once this timer expires, a Cause with value ‘‘Timer Expiry’’ may be sent across the ASAI interface.

The ECS Congestion and Flow Control on ASAI Links

ECS Controls on Receive Traffic

This section describes the ECS congestion controls on incoming ASAI traffic (that is, from adjuncts to the ECS) in the following situations:

- Total incoming traffic causes ECS Central Processing Unit (CPU) congestion
- Layer 2 Processor congestion
- Single link congestion (hyperactivity)

ECS CPU Congestion on Received Data

The ECS automatically applies ECS congestion (overload) controls when the processor occupancy for call processing tasks exceeds a predefined threshold over a period of time. The measured call processing occupancy (which is compared to the threshold) is a collective measurement of the occupancy of all call processing-related ECS services, such as ASAI, telephone stations, and trunks. Therefore, while ASAI may not be a major contributor to the ECS congestion, the overload control affects ASAI interfaces.

The ECS congestion controls prevent new originations (that is, new calls and new ASAI associations). Established calls and ASAI associations are not affected by the overload controls.

During an overload control condition, the ECS denies requests for additional ASAI feature access as follows:

- When the ECS receives an ASAI message that begins a new association (for example, REGister message), it is discarded and the ECS responds with a RELease COMplete Message containing Q.931 Cause Value 42 (Coding Standard 0/Network Congestion).
- When the ECS receives a FACility message with a Domain Control Auto-Dial Request, it is discarded and the ECS responds with a FACility Message containing Q.931 Cause Value 42 (Coding Standard 0/Network Congestion).

ASAI associations opened prior to the congestion event were processed normally.

Adjuncts should provide complementary overload controls. Instead of immediately requesting the same service or other services (which may further aggravate the ECS congestion), an adjunct could refrain from sending such requests to the ECS for a short period of time after receiving a message containing Cause 42.

Layer 2 Processor Congestion on Received Data

The layer 2 (L2) processor uses a common buffer pool for receiving frames from all active links (including ASAI). The availability of buffers determines whether the L2 processor is congested. The L2 processor compares buffer usage against a threshold. When the number of buffers in use reaches the threshold, the ECS CPU is informed, and this causes the ECS processor to activate the congestion control described above. The congestion control is released when the buffer level returns to a normal operating range.

Exhaustion of all buffers in the L2 processor receive buffer pool causes the L2 processor to take additional flow control action on selected links. If total exhaustion occurs, the L2 processor sends a Receiver Not Ready (RNR) frame to each link over which it has received a frame. The L2 processor keeps track of the links to which to send an RNR. When the buffer level returns to a normal operating range, the L2 processor sends a Receiver Ready (RR) frame on these flow-controlled links.

Link Congestion (Hyperactivity) — Received Data

The ECS enables link congestion controls when traffic is greater than expected on an individual link. These controls prevent a single link from taking an inequitable share of the ECS resources (that is, processing power or buffers).

The L2 processor monitors the number of frames (Info, Supervisory, UI) received over each active link in a per unit time period. If the number of frames exceeds a specified threshold for the link, the L2 processor declares that link to be ‘‘hyperactive.’’

For hyperactive ASAI links, the L2 processor takes two actions:

1. It flow controls the link by sending an RNR to the link endpoint each time a frame is received from the endpoint.
2. It reports the hyperactivity event to the ECS CPU.

The L2 processor continues to flow control the link until the ECS processor notifies it to withdraw hyperactivity control on that link. The ECS waits a designated period of time (in seconds) after a hyperactivity event before it notifies the L2 processor to resume normal processing on the link. The L2 processor transmits an RR frame on the link to resume normal link activity.

ASAI adjuncts must recognize and respect this control by suspending info frame transmission until receiving an RR across the link.

The ECS maintenance software keeps track of the frequency of hyperactivity events on each BRI link. ASAI hyperactivity is most likely due to fluctuations in adjunct traffic or under-engineering of the link traffic parameters rather than faulty hardware. The current hyperactivity strategy for ASAI is to monitor the frequency of hyperactivity on the links, and alarm links with persistent hyperactivity, but keep such links in service. ASAI link alarms for hyperactivity should alert technicians and system administrators that traffic re-engineering for that link is required.

Parameters controlling the hyperactivity strategy for ASAI are:

- Prior to G3V4, the L2 processor uses the threshold level of 80 frames per five seconds to detect hyperactivity for G3i, and 50 frames per second for G3r. Starting with Release G3V4, the frames (messages per second) are higher: 160 frames per five seconds for G3i**,** and 200 frames per second for G3r. For Ethernet, these numbers are the same.
- The time for which hyperactivity controls (flow control of the link by sending RNRs) are applied by the L2 processor after detecting a hyperactive link is 20 seconds.
- The frequency (rate) of hyperactive events for a link which triggers maintenance to raise an alarm is five events over a 15-minute period.
- The length of time to retire a hyperactivity alarm on a link is approximately one hour.
- (R5 only) If an ASAI link is reestablished more than three times within a five second interval, then the link is considered hyperactive and is taken out of service.

NOTE:
These parameter values are not set on a per ASAI link basis; common values are defined for all ASAI links.

Controls on Send Traffic

This section discusses the ECS congestion controls for traffic from the ECS toward the adjuncts (the transmit direction):

1. Over all links (that is, the ECS congestion).
2. Over a single link (that is, link congestion).

Layer 2 Processor Congestion on Send Traffic

L2 processor congestion in the transmit direction (toward adjuncts) is measured by the number of available transmit buffers. A common buffer pool is used by the L2 processor for transmitting frames to all links (including ASAI). Exhausting all the buffers in the L2 processor transmit buffer pool results in the ASAI software receiving an indication of the condition. If ASAI discards a frame, the ECS maintenance is notified and takes corrective action.

Link Congestion on Send Traffic

Congestion on a single link occurs when the number of L2 buffers queued for transmission over a link exceeds a threshold. A thresholding mechanism is used such that when a low water-mark is reached, all new initiating associations on that link are denied. Further, when a high water-mark is reached, the ECS is no longer able to buffer messages and they are dropped.

The congestion of a single link is likely NOT due to traffic overflow by the ECS, but occurs when the L2 processor stops frame transmission in response to adjunct flow control. The adjunct can withhold an L2 acknowledgment or transmit an RNR frame to cause such a situation.

The threshold parameters for link transmit queues are as follows: The value of the low water-mark is 25 (R5i) and 75 (R5r). The value of the high water-mark is 75 (R5i) and 150 (R5r). These parameters are not individually set for each ASAI link; common values are defined for all ASAI links.

TCP Tunnel Protocol

7

Overview

This chapter describes version 1 of the TCP tunnel protocol.

CallVisor ASAI over the DEFINITY LAN Gateway is a communications interface that provides the functionality of the Adjunct/Switch Application Interface (ASAI) using an Ethernet transport instead of a Basic Rate Interface (BRI) transport.

DEFINITY LAN Gateway uses a TCP tunnel protocol in addition to the protocols defined by ASAI for layer 3 Q.931/2. This tunnel protocol works as follows: Before a client connects, the DEFINITY LAN Gateway application or brouter[1] issues ICMP Echo Request packets (that is, ''ping'') to each administered client to determine whether the client can be reached. As long as the ICMP Echo Request packets are being answered (by ICMP Echo Reply packets), the brouter reports to DEFINITY ECS that layer 1 is up for a particular virtual BRI port. If ICMP Echo Request packets are not being answered, then layer 1 is reported as down.

The software used by CallVisor ASAI over the DEFINITY LAN Gateway is shipped from the factory with a default IP address of 192.168.25.10 and a default hostname ''definity.'' It is also shipped with a default **client** IP address of 192.168.25.20 and hostname ''client.'' The brouter listens for connections from clients on TCP port number 5678. The client must establish a TCP connection to the brouter at this port and IP address. The customer may change the IP address and/or hostname, but the TCP port is fixed.

In addition to the normal TCP mechanism for establishing a connection, the brouter imposes its own protocol for all clients. Once the client has used the protocol to request service and the brouter has accepted the request, an

1. The term “brouter” is synonymous with DEFINITY LAN Gateway application. For brevity, “brouter” is used throughout this chapter.

ASAI-Ethernet connection is established, and pinging of the client ceases for that link. It resumes when the ASAI-Ethernet connection is closed. Since the ASAI-Ethernet connection rides on top of the TCP connection, closing the TCP connection terminates the ASAI-Ethernet connection.

TCP only insures the reliable delivery of a data stream, yet ASAI protocol expects to interact in complete messages. To overcome this mismatch, both the client and brouter must ensure that an entire message is sent or received before processing a subsequent message.

All network communication between clients and the brouter is message-oriented. A small number of messages are used in the TCP tunnel protocol, each of which is prefixed by a 4-octet header as shown in Table 7-1. All four octets of the header are always present on each message, even if they are not used.

NOTE:
''Brouter'' and ''server'' are used interchangeably in this chapter.

Table 7-1. TCP Tunnel Protocol Header Format

Octet 1	Octet 2	Octet 3	Octet 4	Add'l Data...
Message Type	Message Cause	Add'l Data Size High Octet	Add'l Data Size Low Octet	Add'l Data ... (defined by type)

These octets are explained as follows:

- Octet 1 contains the Message Type. Values are provided as an 8-bit unsigned integer.
- Octet 2 contains a Message Cause. Values are provided as an 8-bit unsigned integer.
- Octets 3 and 4 contain an Additional Data Size. This size represents the number of additional octets of data that are part of this Tunnel Protocol message. Octets 3 and 4 are used together to represent a 16-bit unsigned integer in network byte order.
- Additional Data may follow the above four octets. There must be exactly the number of octets of Additional Data indicated in the Additional Data Size. All values for additional data are provided as single octet unsigned integers unless explicitly stated otherwise.

Table 7-2 that follows lists the message types. A single asterisk (*) next to a message indicates that it is sent only by the server. A double asterisk (**) means that it is sent only by the client.

Table 7-2. TCP Tunnel Protocol Message Header Values

Octet 1 value Message Type Value and Description	**Octet 2 value Message Cause Value and Description**	**Octet 3 and 4 value Additional Octet Count Value and Description**
0 *Error Notification* The server will immediately close the TCP connection.	1 *Client too slow* The server established a TCP connection but did not receive the first TCP Tunnel Protocol message quickly enough.	0 No additional data.
	2 *Out of service* A connected ASAI-Ethernet connection was taken out of service on the brouter.	0 No additional data.
	3 *Invalid type* A message with an invalid type has arrived on this connection.	1 Octet 5 contains the offending message type.
	4 *Invalid cause* A message with an invalid cause has arrived on this connection.	2 Octet 5 contains the offending message type. Octet 6 contains the offending message cause.
	5 *No reply to heartbeat* A ‘‘heartbeat reply’’ was not received within the allotted time.	0 No additional data.

(Continued on next page)

Table 7-2. TCP Tunnel Protocol Message Header Values — *(Continued)*

Octet 1 value Message Type Value and Description	Octet 2 value Message Cause Value and Description	Octet 3 and 4 value Additional Octet Count Value and Description
	6 *Too much (or little) data* The octet count specified exceeds the maximum allowable for a particular message type/cause, or the size was not sufficient for the particular message type.	6 Octet 5 contains the offending message type. Octet 6 contains the offending message cause. Octets 7 and 8 contain the offending octet count as a 16-bit unsigned integer in network byte order. Octets 9 and 10 contain the maximum octet count as a 16-bit unsigned integer in network byte order.
	7 *Invalid client* The client is not administered on the brouter.	0 No additional data.
	8 *New connection made* A new connection has been accepted for the same host/link as an existing ASAI-Ethernet connection. This message will be sent on the old ASAI-Ethernet connection.	0 No additional data.
	9 *Invalid Context* A TCP tunnel protocol message was received by the server at an inopportune time.	2 Octet 5 contains the inopportune message type. Octet 6 contains the inopportune message cause.
		(Continued on next page)

Table 7-2. TCP Tunnel Protocol Message Header Values — *(Continued)*

Octet 1 value Message Type Value and Description	**Octet 2 value Message Cause Value and Description**	**Octet 3 and 4 value Additional Octet Count Value and Description**
	255 *Server error* The server experienced an internal error. Reconnecting may eliminate the error condition.	0 No additional data.
1 *Connection Request*** A client uses this message type to request an ASAI-Ethernet connection.	0 Not used.	2 Octet 5 contains the client's link number. Octet 6 contains the client's TCP Tunnel protocol version number.
2 *Connection Accepted** The server has accepted the connection request.	10 *Link up* ASAI data messages can now be exchanged.	0 No additional data.
	11 *Link Down* ASAI data messages cannot be exchanged.	1 Octet 5 contains the link down reason. Octet 5 Value and Description: 101 DEFINITY ECS is down. 102 Virtual BRI port administered. 103 DEFINITY ECS has taken layer 2 down. 104 Virtual BRI port busied-out on DEFINITY LAN Gateway system assembly.
		(Continued on next page)

Table 7-2. TCP Tunnel Protocol Message Header Values — *(Continued)*

Octet 1 value Message Type Value and Description	**Octet 2 value Message Cause Value and Description**	**Octet 3 and 4 value Additional Octet Count Value and Description**
3 *Connection Rejected** The server has rejected the connection request, and will close the TCP connection.	12 Invalid link The requesting client's link number is unknown to the server.	0 No additional data.
	2 *Out of service* The requested link number has been taken out of service on the brouter.	0 No additional data.
	13 *Unsupported TCP Tunnel Protocol version* The client's TCP Tunnel protocol version is not supported by the server.	1 Octet 5 contains the server's TCP Tunnel protocol version number.
4 *Disconnect Notification*** Used to inform the server that the client no longer needs an ASAI-Ethernet connection. The server will immediately close the TCP connection upon receipt.	0 Not used.	0 No additional data.
5 *Link Status** The server sends this message any time the status of a link changes. The ASAI-Ethernet connection remains up.	10 Link up ASAI data messages can now be exchanged.	0 No additional data.
		(Continued on next page)

Table 7-2. TCP Tunnel Protocol Message Header Values — *(Continued)*

Octet 1 value Message Type Value and Description	Octet 2 value Message Cause Value and Description	Octet 3 and 4 value Additional Octet Count Value and Description
	11 *Link Down* ASAI data messages cannot be exchanged.	1 Octet 5 contains the link down reason. Octet 5 Value and Description: 101 DEFINITY ECS is down. 102 Virtual BRI port busied-out or not administered. 103 DEFINITY ECS has taken layer 2 down. 104 Virtual BRI port busied-out on DEFINITY LAN Gateway system assembly.
		(Continued on next page)

Table 7-2. TCP Tunnel Protocol Message Header Values — *(Continued)*

Octet 1 value Message Type Value and Description	**Octet 2 value Message Cause Value and Description**	**Octet 3 and 4 value Additional Octet Count Value and Description**
6 *Heartbeat* This message can be used to determine the health of an ASAI-Ethernet connection. Upon receipt it must immediately be responded to with a ''heartbeat reply'' message.	0-255 *Invoke ID* This value is repeated by the *Heartbeat Reply* to allow correlating the response to the request.	0 No additional data.
7 *Heartbeat Reply*	0-255 *Invoke ID* This value should be set to the same value as the *Heartbeat Invoke ID* received in the *Heartbeat* message.	0 No additional data.
8 *ASAI Data* Used by both client and server to send ASAI data. The ASAI data must immediately follow this header. One and only one ASAI message should be sent with each *ASAI Data* message type.	0 Not used.	1-260 The actual ASAI message starts at Octet 5. One and only one ASAI message should be sent with each *ASAI Data* message header.

Tunnel Protocol Procedure

The method by which the messages in Table 7-2 are exchanged in order to establish an ASAI-Ethernet connection for ASAI messages is as follows:

1. The client opens a TCP connection to port 5678 on host *definity*.
2. The brouter accepts the connection, and either retains it or sends an *Error Notification* message and closes the connection. The normal case is to retain the connection. The *Error Notification* message may be sent at this point for any one of three cases: if the client does not send the *Connection Request* message within 30 seconds, if the client is invalid because it was not administered in the brouter, or if there is a server condition that prevents any connections from being made.
3. The client sends a *Connection Request* message type with octet 5 set to the client's link number and octet 6 set to 1, the client's TCP tunnel protocol version number.
4. The brouter responds with the *Connection Accepted, Connection Rejected*, or *Error Notification Message*. The normal case is to accept the connection. If the connection is accepted, then a link-up/down indication is provided. If the link is down, the reason for being down is provided, and exchange of ASAI data is deferred until a *Link Status - Link Up* message is received. The connection may be rejected because of an invalid client-link pair, because the assigned virtual BRI port is out of service (either on DEFINITY ECS or on the DEFINITY LAN Gateway system assembly), or because an unsupported version of the tunnel protocol was requested. The *Error Notification* message may also be sent at this point because the *Connection Request* was not received within 30 seconds after the client connected.
5. The client and server have now established an ASAI-Ethernet connection. If the connection was established with a *Link Up* cause, or if a subsequent *Link Status* message is received with a *Link Up* cause, then the link is up. When this is the case, either side may initiate the sending of *ASAI Data* messages and both sides should be prepared to receive them. Each piece of ASAI data that is sent must be prefixed by the *ASAI Data* message header with octets 3 and 4 set to the octet count of the ASAI data. The actual ASAI data must immediately follow the message header. One, and only one, ASAI message should be sent with each *ASAI Data* message header. The initial *ASAI Data* messages transport the Q.931 RESTart messages. For more information on the ASAI Q.931 protocol, refer to the earlier chapters in this guide.
6. While the ASAI-Ethernet connection is established, the client may be sent various messages. If an *Error Notification* message is sent, a cause is provided and the TCP connection is closed. If a *Link Status* message is sent indicating that the ASAI link is down, the client may either wait for another *Link Status* message that indicates that the ASAI link is up once again, or it may abort the connection. There is diagnostic information provided in octet 5 of the *Link Down* message that can help an application

decide whether to wait or to abort the connection. Finally, a *Heartbeat* message may be sent, and should be replied to immediately with a *Heartbeat Reply* message. Additional information about Heartbeat messages follows in this chapter.

7. When the client decides to terminate the connection, it should send a *Disconnect Notification* message. The client is free to immediately close its TCP connection after sending this message.
8. Upon receipt of the *Disconnect Notification* message, the server immediately closes the TCP connection and listens for a new one.

Sample Message Exchange Scenarios

This section provides examples of message exchange scenarios. For each scenario, the protocol interaction is presented, followed by a table showing the messages exchanged, their sequence, and their direction. Within each table, the TCP Tunnel Protocol Messages are provided both as text and as an octet sequence.

Successful Connection Scenario

An example of a typical successful connection scenario is as follows:

- The client opens a TCP connection to port 5678 on host *definity*.
- The client sends a *Connection Request* message with octet 5 set to the client's link number (3) and octet 6 set to the client's TCP tunnel protocol version number (1).
- The brouter responds with a *Connection Accepted* message with the cause set to *Link Up*.
- The client and server have now established an ASAI-Ethernet connection. Either side may initiate the sending of *ASAI Data* messages in order to restart Q.931 layer 3 and bring up ASAI, and both sides should be prepared to receive them. Each piece of ASAI data that is sent must be prefixed by the *ASAI Data* message header with octets 3 and 4 set to the octet count of the ASAI data. The actual ASAI data must immediately follow the message header. One and only one ASAI message should be sent with each *ASAI Data* message header.
- When the client decides to terminate the connection it should send a *Disconnect Notification* message. The client is free to immediately close its TCP connection after sending this message.
- Upon receipt of the *Disconnect Notification* message, the server immediately closes the TCP connection and listens for a new one.

Table 7-3. Typical Successful Connection Scenario

Client CTI-host, link 3, version 1	Direction	DEFINITY LAN Gateway Definity, version 1
TCP Open to definity	→	
Connection Request 1,0,0,2,3,1	→	
	←	*Connection Accepted, Link Up* *2,10,0,0*
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2...	→	
Disconnect Notification 4,0,0,0	→	
Close (TCP)	●	Close (TCP)

Client Connects While ASAI Link is Down

A typical scenario of a client connecting while its ASAI link is down follows:

- The client opens a TCP connection to port 5678 on host *definity*.
- Client sends server a *Connection Request* message with octet 5 set to the client's link number (3) and octet 6 set to the client's TCP Tunnel protocol version number (1).
- Server responds with a *Connection Accepted* message with the cause set to *Link Down*. Octet 5 indicates that the entire DEFINITY ECS switch is down (101). After the client receives this message it does not send any *ASAI Data* to the server until it receives a *Link Status* message with the cause set to *Link Up*.
- The client receives a *Link Status - Link Up* message.

- The client and server are now connected and bring up ASAI.
- Eventually the client decides to terminate the connection and sends a *Disconnect Notification* message.
- Both client and server close the TCP.

Table 7-4. ASAI-Link Down Scenario

Client **CTI-host, link 3, version 1**	**Direction**	**DEFINITY LAN Gateway** **definity, version 1**
TCP open to definity	→	
Connection Request 1,0,0,2,3,1	→	
	←	*Connection Accepted* *Link Down* 2,11,0,1,101
	←	*Link Status, Link Up* 5,10,0,0
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2....
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	
Disconnect Notification 4,0,0,0	→	
Close (TCP)	●	Close (TCP)

Invalid TCP Tunnel Protocol Version Scenario

A typical scenario where a client connect request is rejected because of an unsupported tunnel protocol version follows:

- The client opens a TCP connection to port 5678 on host *definity*.
- Client sends server a *Connection Request* message with octet 5 set to the client's link number (3) and octet 6 set to the client's TCP Tunnel protocol version number (2).
- Server responds with a *Connection Rejected* message with a cause of unsupported TCP Tunnel Protocol version, and closes the TCP connection. The client should also close the TCP connection. The cause indicates that the server does not support the client's version of the TCP Tunnel protocol.

  The server also returns the supported version of the TCP Tunnel protocol as octet 5 of the *Connection Rejected* message. The client should try reconnecting using this version of the protocol.
- The client and server are not connected, so they close TCP.

Table 7-5. Invalid TCP Tunnel Protocol Version Scenario

Client **CTI-host, link 3, version 2**	**Direction**	**DEFINITY LAN Gateway** **definity, version 1**
TCP Open to definity	→	
Connection Request 1,0,0,2,3,2	→	
	←	*Connection Rejected, unsupported TCP Tunnel Protocol version* (1 is Supported) 3,3,0,1,1
Close (TCP)	●	Close (TCP)

Error Notification Scenario

The server may send an *Error Notification* message and then close the connection at any time. The client should immediately close its TCP connection whenever it receives this message.

Table 7-6. Error Notification Scenario

Client CTI-host, link 3, version 1	Direction	DEFINITY LAN Gateway definity, version 1
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2 ...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	
	←	*Error Notification* 0,2,0,0
Close (TCP)	●	Close (TCP)

ASAI Link Status Scenario

A typical scenario depicting changes in link status follows. Links may be unadministered or busied out at any time on DEFINITY ECS. Layer 2 may also be brought down on a hyperactive link by DEFINITY ECS. To relay this change in link status, the server sends *Link Status* messages, which clients may receive at any time.

When a message arrives with the cause set to *Link Down*, the client should stop sending *ASAI Data* to the server. Any *ASAI Data* received by the server after it sends a *Link Status* message with the cause set to *Link Down* is discarded. When the link is brought back up, the server sends the client a *Link Status* message with the cause set to *Link Up.* At this time the client may again send/receive *ASAI Data*, and either the client or e DEFINITY ECS must restart the ASAI Q.931 protocol.

Table 7-7. ASAI Link Status Scenario

Client **CTI-host, link 3, version 1**	**Direction**	**DEFINITY LAN Gateway** **definity, version 1**
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2 ...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	
	←	*Link Status, Link Down,* L2 down 5,2,1,0,103
	←	*Link Status, Link Up* 5,1,0,0
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2 ...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	

Network Outages

It is possible for an established TCP connection to be unable to pass data from client to server (or vice versa). A number of problems, including the following, can cause this situation:

- A router failure between client and server
- A physical media failure between client an server
- A failure in either the client or server

TCP can detect such conditions and close down the connection; however, this may take up to 15 minutes.

CallVisor ASAI over the DEFINITY LAN Gateway is used to support real-time applications that cannot tolerate such delay in determining network outages. o compensate for TCP limitations, the TCP tunnel protocol uses a heartbeat procedure when message activity is low, and a reconnect procedure that favors new connections over existing ones.

Heartbeat Messages

The tunnel protocol defines a *Heartbeat* and a *Heartbeat Reply* message. These are in addition to the ASAI Q.931 *Heartbeat* messages, which are transported using an *ASAI Data Message* that contains the ASAI Q.931 *Heartbeat* in its Additional Data. Both client and brouter may issue a *Heartbeat* message. The peer responds with a *Heartbeat Reply* message. Each *Heartbeat* message contains an *Invoke ID*, which must be repeated in its *Heartbeat Reply* message.

Heartbeat Message Procedure

The brouter employs the following strategy on each active ASAI-Ethernet connection to detect network or client outages:

- If no messages are received on an ASAI-Ethernet connection in a 20-second period, the brouter sends a *Heartbeat* message.
- If the brouter does not receive a *Heartbeat Reply* (or other) message within 10 seconds, it sends an *Error Notification* message, closes the TCP connection, and begins listening for a new connection.

  NOTE:
  The brouter will not close a connection if the *Heartbeat Reply Invoke ID* does not match that sent in the *Heartbeat*, nor does it close a connection if a message of any type is sent from the client in the 10-second window. This behavior prevents ‘‘busy’’ clients from assuming additional overhead to support the heartbeat message procedure.

Table 7-8. Use of Heartbeat Messages

Client CTI-host, link 3, version 1	Direction	DEFINITY LAN Gateway definity, version 1
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2 ...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	
20 seconds elapse		
	←	*Heartbeat* 6,47,0,0
Heartbeat Reply 7,47,0,0	→	
20 seconds elapse		
	←	*Heartbeat* 6,48,0,0
10 seconds elapse		
	←	*Error Notification, no HB reply* 0,5,0,0
Close (TCP)	●	Close (TCP)

Reconnect Procedure

If a client believes it has lost connectivity to the server it may close the TCP connection and establish a new one. If the server receives a *Connection Request* from a client on a new connection for which it believes there is an existing connection, it closes the existing connection and accepts the new one. When this occurs, the server sends an *Error Notification* message on the old connection with the cause set to *New Connection Made* before closing the connection.

NOTE:
If a client wishes to support two ASAI-Ethernet links, it must have two entries in the brouter table that contains two different link values. In other

words, a client may open link 3 and link 4 simultaneously, but it may not open link 3 two times simultaneously.

Table 7-9. Connecting While a Connection Exists

Client **CTI-host, link 3, version 1**	**Direction**	**DEFINITY LAN Gateway** **definity, version 1**
	←	*ASAI Data* (Q.931 Msg.) 8,0,X,X,8,2 ...
ASAI Data (Q.931 Msg.) 8,0,X,X,8,2 ...	→	
		Connection Request was received for CTI-host, link3 on a new connection
	←	*Error Notification, New Connection Made* 0,8,0,0
Close (TCP)	●	Close (TCP)
		Messages are now exchanged on the new TCP connection

The ECS Mapping to Information Elements in Third Party Make Call

A

When an adjunct uses the Third Party Make Call capability to set up a call, the ECS maps the request into a request for one of four internal types of calls:

1. User-Classified call
2. Switch-Classified call
3. Direct-Agent call
4. Supervisor-Assist call

The *CallVisor ASAI Technical Reference* explains the attributes of these types of calls and their uses.

Adjunct applications are responsible for requesting call types appropriate for their needs. The Third Party Make Call message layout is a union of all possible IEs that may appear in the request; the rules detailed in this appendix show how the ECS maps the IEs present in a Third Party Make Call capability into a request for a specific call type. Table A-1 and Table A-2 also show how the ECS validates the IEs for each call type.

The numbers in the tables correspond to the order in which the ECS software looks for the presence of the information elements and/or validates them. The asterisks in the table indicate information about an IE's presence or absence. These numbers and asterisks are explained as follows:

(1) The presence of the Calling and Called Number IEs are checked for all Third Party Make Call requests. The contents of these IEs may be verified at a later time for some call types.

(2a through 2d) The presence of the Supervisor-Assist Call Option, Direct-Agent Call Option, Alert Order Call Options, and Service Circuit IEs determines the

call type and the validation done on IEs. The order in which the ECS checks (2a) through (2d) is implementation-specific and may change.

Adjunct applications should ensure that a legal combination of IEs is present and should not be written in such a way as to use this ordering.

(3) Once the call type is determined, these information elements are incompatible with that call type and the ECS will deny a request that contains them.

(4) The return acknowledgement option is checked for all call types.

(5) through (11) The order in which the ECS validates these IEs within the context of a specific call type. Thus, once a call type is determined, any boxes in the column for that call type, numbered from five upward, are validated in that order for the given call type.

* Not only must this IE be present, the ECS uses its presence to determine the call type. If the IE is not present, the ECS interprets the call type to be something different.

** Not only must this IE be absent, the ECS uses its presence to determine the call type. (The ECS uses the presence of this IE to determine if the request is for a type of call other than the one indicated in the column heading.)

Table A-1. IEs for User-Classified and Switch-Classified Third Party Make Call Requests

IE	Call Type	
	User-Classified	**Switch-Classified**
(1) Calling Number IE	(1) Must be present; denied if not. ..... (7) Must be station; denied if not.	(1) Must be present; denied if not. ..... (11) Must be Split, Hunt, Announce, VDN (only in an EAS environment). Denied if not.
(1) Called Number IE	(1) Must be present; denied if not. ..... Internal — Call Processing will validate. External dest — not validated	(1) Must be present; denied if not. ..... (10) Internal — denied if not station. External dest — not validated
(2a) Supv. Assist Call Option IE	Cannot be present**.	Cannot be present**.
(2b) Direct-Agent Call Option IE	Cannot be present**.	Cannot be present**.
(2c) Alert Order Call Options IE	Cannot be present**.	(2c) Must be present*. ..... (6) Denied if not “dest first.”
(2d) Service Circuit Service Circuit IE	(2d) Must not be present. Incompatible Option; deny if present.	(9) For external dest. Must be present, and must equal classifier. ..... Ignored for internal dest.
(4) Return ACK Call Option IE	(4) Presence is optional. ACK sent when present; ACK not sent when absent.	(4) Presence is optional. ACK sent when present; ACK not sent when absent.
		(Continued on next page)

Table A-1. IEs for User-Classified and Switch-Classified Third Party Make Call Requests — *(Continued)*

IE	Call Type	
	User-Classified	**Switch-Classified**
Domain IE	(6) If present, ECS processes dest. as external. Type must be Trunk, or denied. ..... Must omit for internal dest.	(8) If present, ECS processes dest. as external. Type must be Trunk, or denied. ..... Must omit for internal dest.
Number of Rings Call Option IE	Ignored	(7) Presence is optional. Default = 10 when not present. Deny if out of range (2-15).
Priority Call Call Option IE	(5) Presence is optional. Priority call when present. Nonpriority call when not present.	(5) Presence is optional. Priority call when present. Nonpriority call when not present.

Table A-2. IEs for Direct-Agent and Supervisor-Assist Third Party Make Call Requests

IE	Call Type	
	Direct-Agent	**Supervisor-Assist**
(1) Calling Number IE	(1) Must be present; denied if not. ..... (6) Must be local station; denied if not.	(1) Must be present; denied if not. ..... (6) Must be local station; denied if not.
(1) Called Number IE	(1) Must be present; denied if not. ..... (7) Must be local station; denied if not. Incompatible option; deny if present.	(1) Must be present; denied if not. ..... (7) Must be local station; denied if not.
(2a) Supv.-Assist Call Option IE	Must not be present**. Incompatible Option; deny if present.	(2a) Must be present*.
(2b) Direct Agent Call Option IE	(2b) Must be present*.	(3) Must not be present. Incompatible Option; deny if present.
(2c) Alert Order Call Options IE	(3) Incompatible Option; deny if present.	(3) Must not be present. Incompatible Option; deny if present.
(2d) Service Circuit Service Circuit IE	(3) Must not be present. Incompatible Option; deny if present.	(3) Must not be present. Incompatible Option; deny if present.
(4) Return ACK Call Option IE	(4) Presence is optional. ACK sent when present; ACK not sent when absent.	(4) Presence is optional. ACK sent when present; ACK not sent when absent.
Domain IE	(5) Must be present, type must be split or skill, and addr must be valid or denied.	(5) Must be present, type must be split or skill, and addr must be valid or denied.

(Continued on next page)

Table A-2. IEs for Direct-Agent and Supervisor-Assist Third Party Make Call Requests — *(Continued)*

IE	Call Type	
	Direct-Agent	**Supervisor-Assist**
Number of Rings Domain IE	Ignored if present.	Ignored if present.
Priority Call Call Option IE	(8) Presence is optional. Priority call when present. Non-priority call when not present.	(8) Presence is optional. Ignored when present. Supv. Assist always priority call.

Message Scenarios

B

This appendix presents sample scenarios showing the messages exchanged between the ECS and an ASAI adjunct processor (AP). A short description of the scenario and a figure showing the ECS elements (for example, VDNs, ACD splits, or stations) involved in the scenario are also provided with each messaging scenario.

Each messaging scenario separates into two columns the ASAI messages exchanged. The first column provides the messages originated or sent by the adjunct processor and the second column provides the messages sent by the ECS. The third column describes the action or event encountered by the scenario. Within each message, the CRV is used to distinguish among active associations maintained by the ECS and the adjunct processor.

Figure B-1 details the ASAI message components used in the scenarios presented in this appendix.

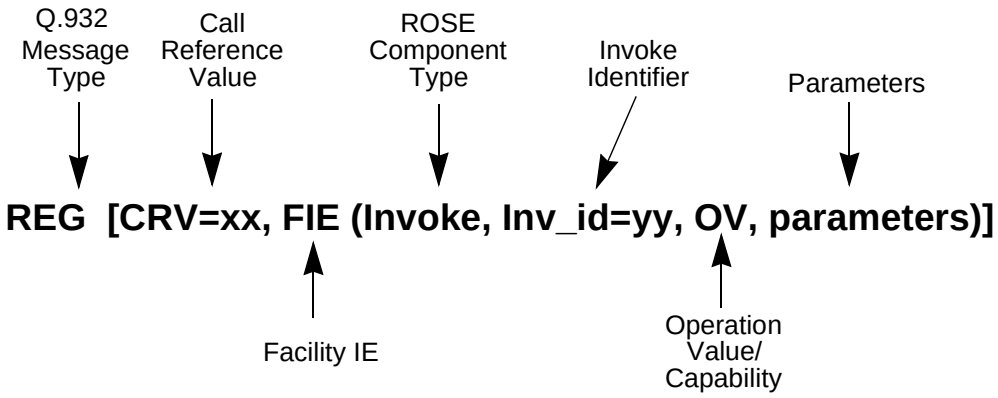


Figure B-1. Notation for the Messages in this Appendix

The values used for each item within an ASAI message (for example, CRVs, call_ids, party_ids, and extension numbers) are for demonstration purposes only and can change for different applications. However, the item values illustrate how the items are used within a scenario. For example, once a call_id has been assigned a value, the assigned value is used to identify the call until the call terminates or is merged with another call.

1. Initialization

This section presents a sample ASAI initialization procedure. The adjunct processor initializes the link and requests Event Notification for VDN 5678 and ACD split 3456. This sample initialization procedure is referred to by some of the scenarios presented in subsequent sections.

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	REST (CRV=global, Restart Indicator=all interfaces)	Restart ASAI Interface
REST ACK [CRV=global, Restart Indicator=all interfaces]		ASAI Interface Restarted
REG[CRV=7F FIE(Invoke, Inv_id=1 Heartbeat)]		
	RELCOMP[CRV=7f,FIE (Return Result, Inv_id=1)]	
REG [CRV=98, FIE(Invoke, Inv_id=1, Event Notification Request domain=VDN 5678)]		Notification Request for VDN 5678
	FAC [CRV=98, FIE(Return Result, Inv_id=1)]	Notification Accepted
REG [CRV=102, FIE(Invoke, Inv_id=3, Event Notification Request, domain=ACD split 3456)]		Notification Request for ACD split 3456
	FAC [CRV=102, FIE(Return Result, Inv_id=3)]	Notification Accepted

2. Notification Associations

This section illustrates the **Event Reports** provided by calls entering different notification domains. Figure B-2 shows the Vector Directory Numbers (VDNs), vectors, ACD splits, and agent extensions used for the message scenarios presented in this section. VDNs 5678, 5690, and 7656 point to vectors A, B, and C, respectively. Only the most relevant vector commands (that is, Adjunct Routing and Queue to Main) are shown. ACD splits 3459 and 3460 are vector-controlled. ACD split 3456 is not vector-controlled. (In reality, this is not possible on the same ECS.) The ACD agents at extensions 4534 and 3567 are logged into ACD split 3456. The ACD agent at extension 4566 is logged into ACD split 3460. There are no available agents for ACD split 3459.

VDN 5678 and ACD split 3456 are monitored by the adjunct processor as presented in the initialization scenario. VDN 7656 is monitored by another adjunct processor not shown in these scenarios. VDN 5690 is not monitored.

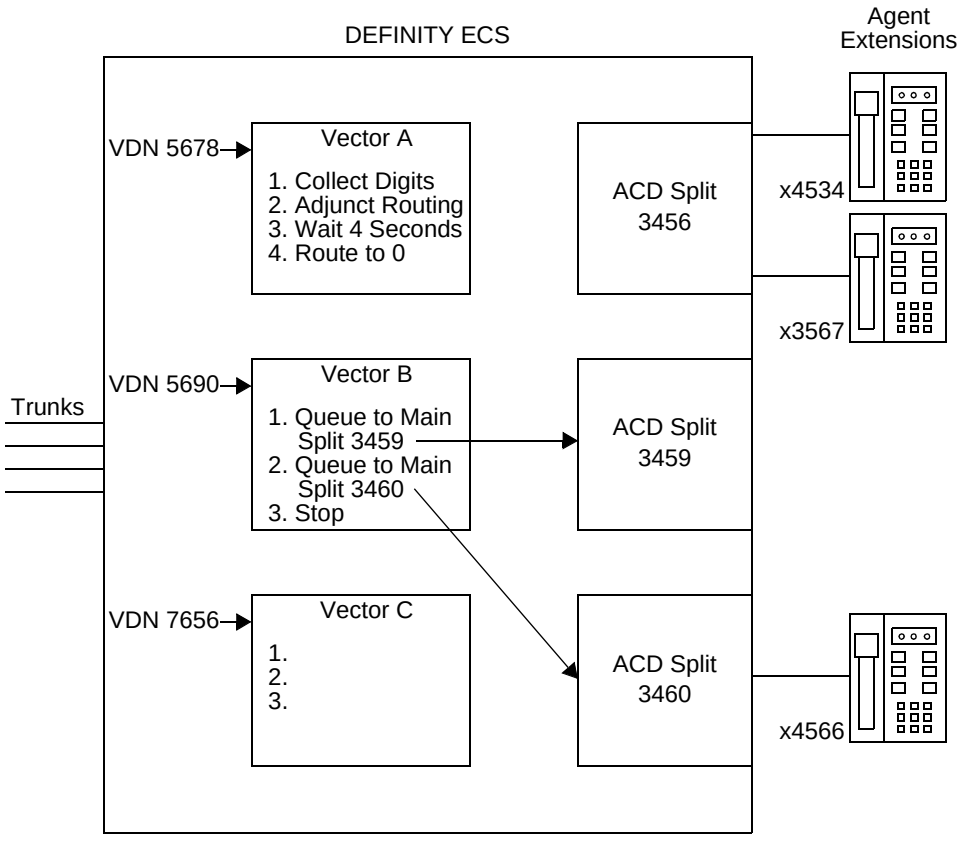


Figure B-2. Relationship Among VDNs, ACD Splits, and Stations

External Call to VDN Routed to ACD Split

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 that is routed via the Adjunct Routing vector command to ACD Split 3456 (Figure B-3). The agent at extension 4534 answers the call.

Event Notification for VDN 5678 and ACD Split 3456 is active as presented in the initialization scenario. The ASAI messages generated by the Adjunct Routing vector command are shown also.

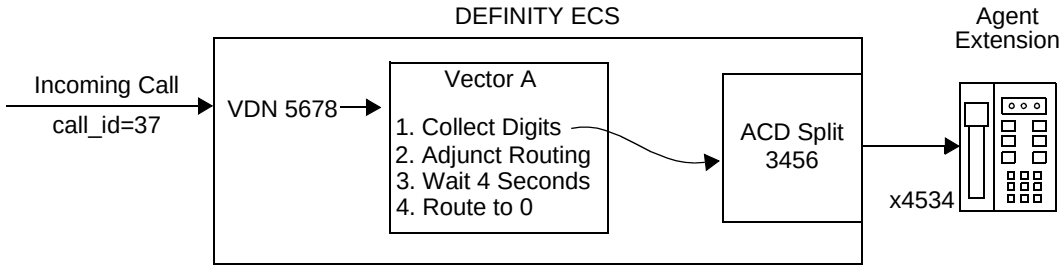


Figure B-3. Call Flow for Incoming Call to VDN Routed to ACD Split

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=37, event=call offered, calling number=2015666379, called number=9085765678, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, call_id=37, calling number=2015666379, called number=9085765678, user code=11569037, domain=VDN 5678)]	Route Request
FAC [CRV=93, FIE(Invoke, Inv_id=7, Route Select, called number=3456)]		Route to ACD Split 3456
	REL COMP [CRV=93, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=37, event=call redirected)]	Notification for Call Terminated over VDN Association
	FAC [CRV=102, FIE(Invoke, Inv_id=10, Event Report, call_id=37, event=call offered, calling number=2015666379, called number=9085765678, user code=11569037, domain=ACD Split 3456)]	Call Offered to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=12, Event Report, call_id=37, event=queued, called number=9085765678, calls in queue=2, domain=ACD Split 3456)]	Queued Event Report (Queued in ACD Split 3456)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=14, Event Report, call_id=37, party_id=2, event=alerting, calling number=2015666379, called number=9085765678, connected number=4534, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 4534)
	FAC [CRV=102, FIE(Invoke, Inv_id=16, Event Report, call_id=37, party_id=2, event=connected, calling number=2015666379, called number=9085765678, connected number=4534)]	Connected Event Report (Call Connected to Agent 4534)
	FAC [CRV=102, FIE(Invoke, Inv_id=18, Event Report, call_id=37, party_id=2, event=drop, connected number=4534, cause=normal clearing)]	Drop Event Report (Agent disconnects)
	FAC [CRV=102, FIE(Invoke, Inv_id=20, Call Ended, call_id=37, cause=normal clearing)]	Call Terminates

External Call to Monitored VDN Routed to Non-Monitored VDN

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 that is routed via the Adjunct Routing vector command to VDN 5690 (Figure B-4). The call queues to two ACD splits, ACD split 3459 and 3460. The agent at extension 4566 and logged into ACD split 3460 answers the call.

Event Notification for VDN 5678 is active as presented in the initialization scenario. The ASAI messages generated by the Adjunct Routing vector command are shown also.

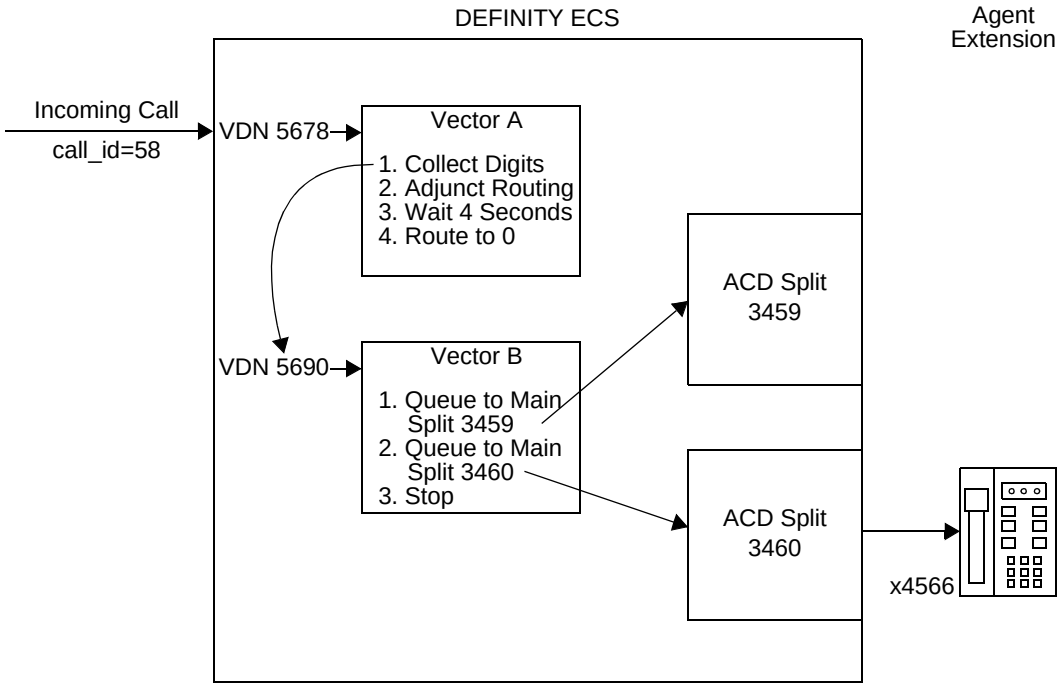


Figure B-4. Call Flow for Incoming Call to VDN Delivered to Agent

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98,FIE(Invoke, Inv_id=2, Event Report, call_id=58 event=call offered, calling number=2015663569, called number=9085765678 domain=VDN 5678)]	Call Offered to VDN Event Report
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, call_id=58, calling number=2015663569, called number=9085765678, user code=34569058, domain=VDN 5678)]	Route Request
FAC [CRV=93, FIE(Invoke, Inv_id=3, Route Select, called number=5690)]		Route to VDN 5690
	REL COMP [CRV=93, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=58, event=queued, called number=9085765678, calls in queue=2, domain=ACD Split 3459)]	Queued Event Report for ACD 3459
	FAC [CRV=98, FIE(Invoke, Inv_id=10, Event Report, call_id=58, event=queued, called number=9085765678, calls in queue=3, domain=ACD Split 3460)]	Queued Event Report for ACD 3460

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=12, Event Report, call_id=58, party_id=2, event=alerting, calling number=2015663569, called number=9085765678, connected number=4566, domain= ACD Split 3460)]	Alerting Event Report (Call Delivered to Agent 4566)
	FAC [CRV=98, FIE(Invoke, Inv_id=14, Event Report, call_id=58, party_id=2, event=connected, calling number=2015663569, called number=9085765678, connected number=4566)]	Connected Event Report (Call Connected to Agent)
	FAC [CRV=98, FIE(Invoke, Inv_id=16, Event Report, call_id=58, party_id=2, event=drop, connected number=4566, cause=normal clearing)]	Drop Event Report (Agent Disconnects)
	FAC [CRV=98, FIE (Invoke, Inv_id=18, Call Ended, call_id=58, cause=normal clearing)]	Call Terminates

External Call to VDN Routed to Monitored VDN

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 that is routed via the Adjunct Routing vector command to VDN 7656 (Figure B-5).

VDN 7656 is monitored by a different adjunct not shown in these scenarios. Event Notification from VDN 5678 is active as presented in the initialization scenario. The ASAI messages generated by the Adjunct Routing vector command are shown also.

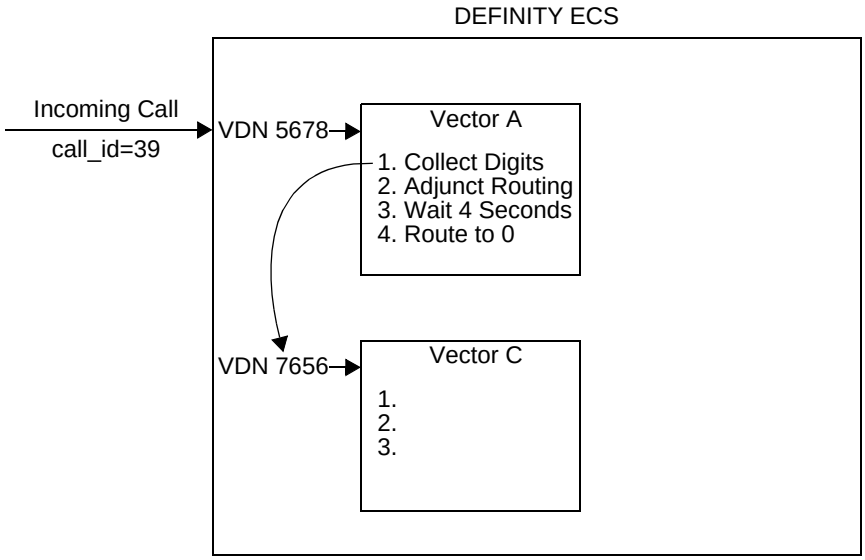


Figure B-5. Call Flow for Incoming Call to VDN Routed to Another VDN

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=39, event=call offered, calling number=6157856399, called number=9089575678, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=93, FIE(Invoke, Inv_id=2, Route, call_id=39, calling number=2015766399, called number=2015765678, user code=00569039, domain=VDN 5678)]	Route Request
FAC [CRV=93, FIE(Invoke, Inv_id=3, Route Select, called number=7656)]		Route Call to VDN 7656
	REL COMP [CRV=93, FIE(Invoke, Inv_id=4, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=4, Event Report, call_id=39, event=call redirected)]	Notification for Call terminated (VDN 7656 monitored by another AP Association)

External Call to VDN Routed to Station

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 that is routed via the Adjunct Route vector command to station 3001 (Figure B-6). Station 3001 does not answer the call and the call redirects to station 4003, maintaining a simulated bridged appearance at station 3001. Station 3001 answers the call causing the call to be disconnected from station 4003.

Event Notification for VDN 5678 is active as presented in the initialization scenario. The ASAI messages generated by the Adjunct Routing vector command are shown also.

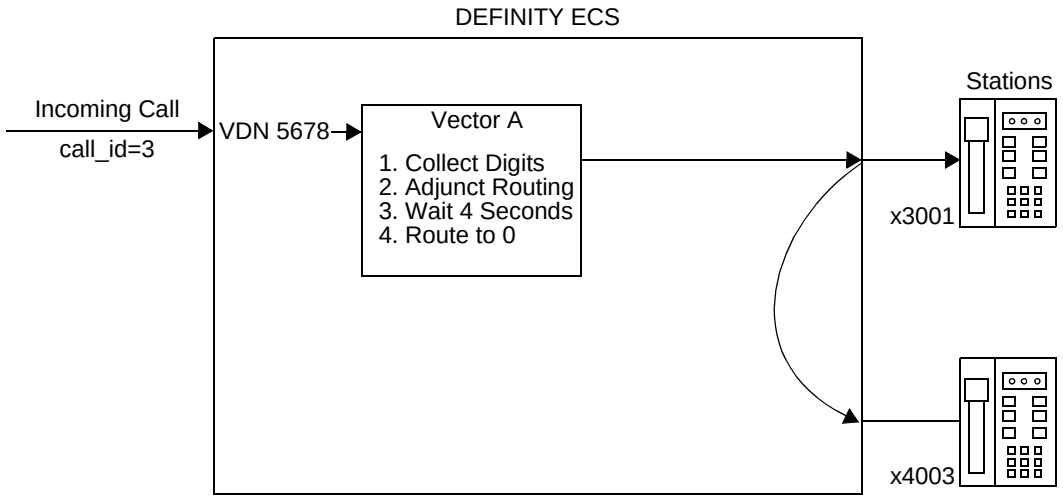


Figure B-6. Call Flow for Incoming Call to VDN Routed to Station

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=3, event=call offered, calling number=2015661234, called number=2015765678, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=13, FIE(Invoke, Inv_id=4, Route, call_id=3, calling number=2015661234, called number=2015765678, domain=VDN 5678, user code=322058)]	Route Request
FAC [CRV=13, FIE(Invoke, Inv_id=3, Route Select, called number=3001)]		Route to Station 3001
	REL COMP [CRV=13, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=4, Event Report, call_id=3, party_id=2, event=alerting, calling number=2015661234, called number=2015765678, connected number=3001)]	Alerting Event Report (Call Alerts Station 3001)
	FAC [CRV=98, FIE(Invoke, Inv_id=6, Event Report, call_id=3, party_id=8, event=alerting, calling number=2015661234, called number=2015765678, connected number=4003)]	Alerting Event Report (Call Alerts Station 4003)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=3, party_id=2, event=connected, calling number=2015661234, called number=2015765678, connected number=3001)]	Connected Event Report (Call Answered at Station 3001)
	FAC [CRV=98, FIE(Invoke, Inv_id=12, Event Report, call_id=3, party_id=1, event=drop, connected number=#####, cause=normal clearing)]	Drop Event Report (Calling Party Drops)
	FAC [CRV=98, FIE(Invoke, Inv_id=14, Call Ended, call_id=3, cause=normal clearing)]	Call Terminates

External Call to VDN Routed to Announcement Extension

This scenario shows the call flow for a non-ISDN incoming call to VDN 5678 routed via the Adjunct Route vector command to announcement extension 3002 (Figure B-7).

VDN 5678 has Event Notification active as presented in the initialization scenario. The ASAI messages generated by the Adjunct Routing vector command are also shown.

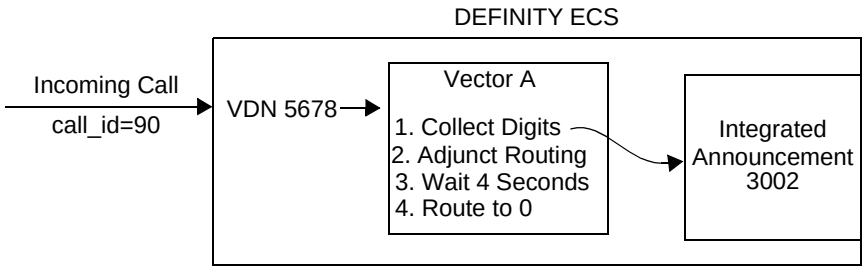


Figure B-7. Call Flow for Incoming Call to VDN Routed to Announcement

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=90, event=call offered, trunk group number=102 called number=5678, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=13, FIE(Invoke, Inv_id=4, Route, call_id=90, trunk group number=102 called number=5678, domain=VDN 5678, user code=322058)]	Route Request
FAC [CRV=13, FIE(Invoke, Inv_id=90, Route Select, called number=3002)]		Route to Announcement Extension 3002
	REL COMP [CRV=13, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=4, Event Report, call_id=90, party_id=2, event=alerting, Trunk Group number=102 called number=5678, connected number=3002)]	Alerting Event Report — Call Queues to Announcement 3002
	FAC [CRV=98, FIE(Invoke, Inv_id=6, Event Report, call_id=90, party_id=2, event=connected, Trunk Group number=102 called number=5678, connected number=3002)]	Connected Event Report — Call Connected to Announcement
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=90, party_id=2, event=drop, connected number=3002, cause=normal clearing)]	Drop Event Report — Announcement Disconnects
	FAC [CRV=98, FIE(Invoke, Inv_id=10, Call Ended, call_id=90, cause=normal clearing)]	Call Terminates

External Call to ACD Split

This scenario shows the call flow for an incoming ISDN PRI call to ACD split 3456 (Figure B-8). The agent at station 4534 answers the call. ACD Split 3456 has Event Notification active as presented in the initialization scenario.

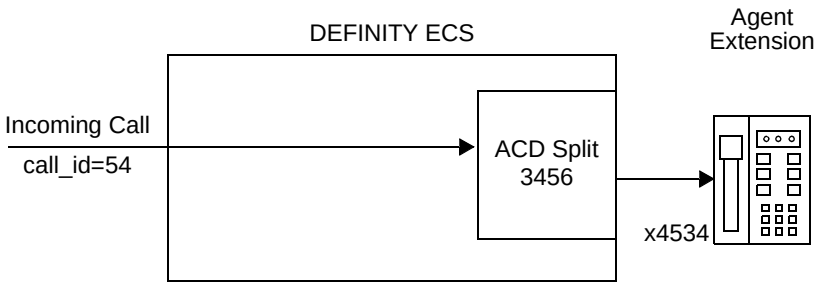


Figure B-8. Call Flow for Incoming Call to ACD Split

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=2, Event Report, call_id=54, event=call offered, calling number=2125437890, called number=9085763456, domain=ACD Split 3456)]	Call Offered to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=4, Event Report, call_id=54, event=queued called number=9085763456, calls in queue=2, domain=ACD Split 3456)]	Queued Event Report (Call Queues to ACD Split 3456)
	FAC [CRV=102, FIE(Invoke, Inv_id=6, Event Report, call_id=54, party_id=2, event=alerting, calling number=2125437890, called number=9085763456, connected number=4534, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 4534)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=8, Event Report, call_id=54, party_id=2, event=connected, calling number=2125437890, called number=9085763456, connected number=4534)]	Connected Event Report (Call Connected to Agent 4534)
	FAC [CRV=102, FIE(Invoke, Inv_id=10, Event Report, call_id=54, party_id=2, event=drop, connected number=4534, cause=normal clearing)]	Drop Event Report (Agent Disconnects)
	FAC [CRV=102, FIE(Invoke, Inv_id=12, Call Ended, call_id=54, cause=normal clearing)]	Call Terminates

3. Adjunct Routing Associations

This section presents sample Adjunct Routing scenarios. Additional Adjunct Routing scenarios are presented in the “Notification Associations” section earlier in this appendix.

Call to VDN Routed Directly to ACD Agent

This scenario shows the call flow for an incoming ISDN PRI call to VDN 8905 routed via the Adjunct Routing vector command to a specific ACD agent (station 3567) (see Figure B-9). The first attempt for a route fails because the specified ACD agent is not logged into the ACD Split.The adjunct processor reroutes the call to ACD agent 4534. The second route delivers the call to the ACD agent. The call queues for the agent and the caller drops before the agent answers the call.

The adjunct processor does not have Event Notification active for VDN 8905. Event Notification for ACD Split 3456 is active as presented in the initialization scenario. Note that Domain Control for the ACD station would have provided event reports for the call.

The adjunct processor receives the Adjunct Routing requests.

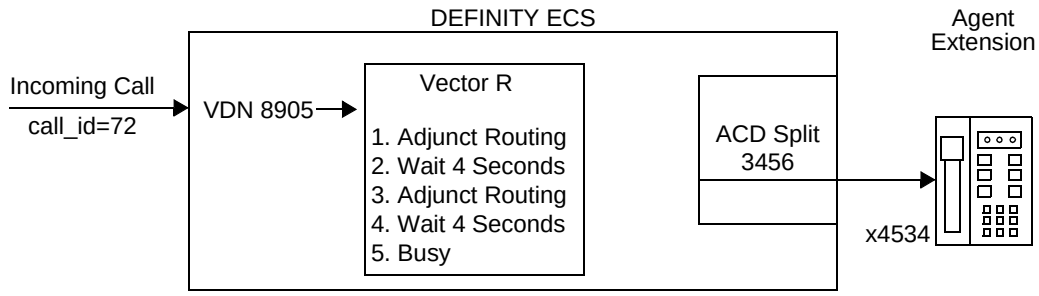


Figure B-9. Call Flow for Incoming Call to VDN Routed to Specific ACD Agent

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	REG [CRV=66, FIE(Invoke, Inv_id=2, Route, call_id=16, calling number=8097670313, called number=9085768905, domain=VDN 8905)]	Route Request
FAC [CRV=66, FIE(Invoke, Inv_id=3, Route Select, called number=3567, domain=ACD Split 3456, direct agent call=yes)]		Route Directly to ACD Agent 3567 on Split 3456
	REL COMP [CRV=66, FIE(Invoke, Inv_id=4, Route End, cause=agent not logged in)]	Route Failed
	REG [CRV=68, FIE(Invoke, Inv_id=2, Route, call_id=16, calling number=8097670313, called number=9085768905, domain=VDN 8905)]	Route Request
FAC [CRV=66, FIE(Invoke, Inv_id=3, Route Select, called number=4534, domain=ACD Split 3456, direct agent call=yes)]		Route Directly to ACD Agent 4534 on Split 3456
	REL COMP [CRV=66, FIE(Invoke, Inv_id=4, Route End, cause=normal)]	Call Routed
	FAC [CRV=102, FIE(Invoke, Inv_id=6, Event Report, call_id=16, party_id=2, event=alerting, calling number=8097670313, called number=9085768905, connected number=4534, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 4534)
	FAC [CRV=102, FIE(Invoke, Inv_id=8, Event Report, call_id=16, party_id=1, event=drop, connected number=#####, cause=normal clearing)]	Drop Event Report (Caller Disconnects)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC[CRV=102, FIE(Invoke, Inv_id=10, Call Ended, call_id=16, cause = normal clearing)]	Call Terminates

Call to VDN Routed to External Destination

This scenario shows the call flow for an incoming call to VDN 201 routed via the Adjunct Routing vector command to VDN 405 (see Figure B-10). At VDN 405, the call is routed via another Adjunct Routing vector command to an external destination. The adjunct processor does not have Event Notification active for VDNs 201 and 405. The adjunct processor receives the Adjunct Routing requests.

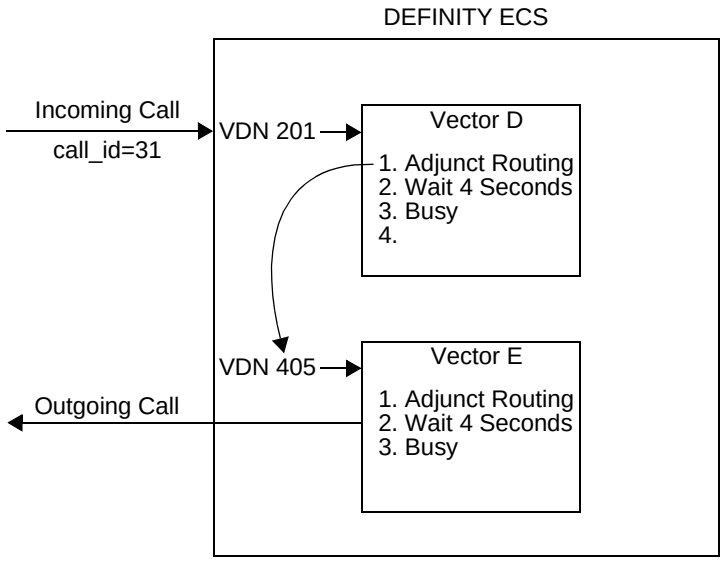


Figure B-10. Call Flow for Incoming Call to VDN Routed to External Destination

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	REG [CRV=23, FIE(Invoke Inv_id=4, Route call_id=31, trunk group number=560, called number=201, trunk group member=11, domain=VDN 201)]	Route Request
FAC [CRV=23, FIE(Invoke, Inv_id=3, Route Select, called number=405)]		Route to VDN 405
	REL COMP [CRV=23,FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	REG [CRV=56, FIE(Invoke, Inv_id=4, Route, call_id=31, trunk group number=560, called number=201, trunk group member=11, domain=VDN 405)]	Route Request
FAC [CRV=56, FIE(Invoke, Inv_id=3, Route Select, dest_route_select=9, called number=14157894567)]		Route to External Destination
	REL COMP [CRV=56, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed

Call Routed Based on II-Digits

This scenario shows how an incoming call is treated based on the value of the Information Indicator (II) Digits (see Figure B-11). These digits indicate the type of telephone that the call was placed from. The following example shows how a company offering emergency roadside assistance handles calls from cellular or pay telephones differently than calls from wired telephones. Known numbers of current customers may also be handled differently, at the adjunct's discretion.

Skill 1 (extension 70001) consists of people trained to deal with roadside problems (understand the problem, dispatch appropriate help, instruct the caller appropriately). Skill 2 (extension 70002) consists of agents with general training, who troubleshoot problems whenever they can, but who will also dispatch the call to an appropriate expert when necessary.

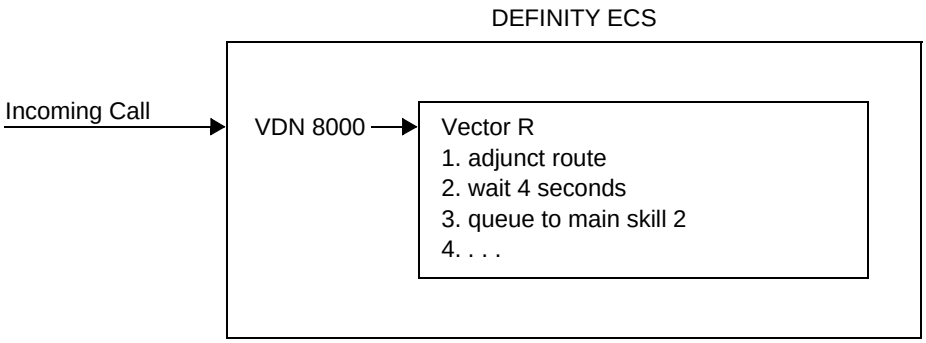


Figure B-11. Call Flow for Incoming Call Based on II-Digits

Call Sequence 1:

An incoming call on an ISDN-PRI trunk is delivered to VDN 8000. The ISDN trunk is configured to provide II-Digits along with CPN/BN. The II-Digits indicate that the call was placed from a cellular phone. The adjunct decides to route the call to skill 1, and to make it a priority call to speed its handling.

Call Sequence 2:

An incoming call on an ISDN-PRI trunk is delivered to VDN 8000. The ISDN trunk is configured to provide II-Digits along with CPN/BN. The II-Digits indicate that the call was placed from a normal wired phone. The adjunct decides to route the call to skill 2, and to make it a non-priority call.

Call Sequence 3:

An incoming call on and ISDN-PRI trunk is delivered to VDN 8000. The ISDN trunk is configured to provide II-Digits along with CPN/BN. The II-Digits indicate that the call was placed from a normal wired phone. The calling number indicates that this call is from a preferred customer. The adjunct decides to route the call to skill 2, but to make it a priority call.

Call Sequence 1:		
Messages Sent by the Adjunct Processor	**Messages Sent by the ECS**	**Comments**
	REGister [CRV=72, FIE(Invoke, Inv_id=2, Route, call_id=1111, domain=VDN 8000 Calling_Number=9085551234, II_Digits=61)]	Route Request
FAC [CRV=23, FIE(Invoke, Inv_id=3, Call_ID=1111, Called number=70001 call_option=priority)]		Route Select
	REL COMP [CRV=72, FIE(Invoke,Inv_id=4, Route End, cause=normal)]	Call Routed
Call Sequence 2:		
Messages Sent by the Adjunct Processor	**Messages Sent by the ECS**	**Comments**
	REG [CRV=74, FIE(Invoke,Inv_id=2, Route, call_id=1112, Call ID =1112 domain=VDN 8000, Calling_Number=9085551235, II_Digits=00)]	Route Request
FAC [CRV=74, FIE(Invoke, Inv_id=3, Call_ID=1112, Called number=70002 call_option=priority)]		Route Select
	REL COMP [CRV=74, FIE(Invoke,Inv_id=4, Route End, cause=normal)]	Call Routed

(*Continued on next page*)

Call Sequence 3:		
Messages Sent by the Adjunct Processor	**Messages Sent by the ECS**	**Comments**
	REGister [CRV=76, FIE(Invoke, Inv_id=2, Route, call_id=1113, domain=VDN 8000 Calling_Number=9085551780, II-Digits=00)]	Route Request
FAC [CRV=76, FIE(Invoke, Inv_id=3, Call_ID=1113, Called number=70002 call_option=priority)]		Route Select
	REL COMP [CRV=76, FIE(Invoke,Inv_id=4, Route End, cause=normal)]	Call Routed

ISDN Information Received with Incoming Call

This scenario describes how II-Digits, UUI and CINFO information are received with an incoming call over an ISDN trunk. It is assumed that the link version was negotiated at V2 or later.

Figure B-12 shows how an incoming call to VDN 5678 is routed to extension 3456 based on the information received with the call.

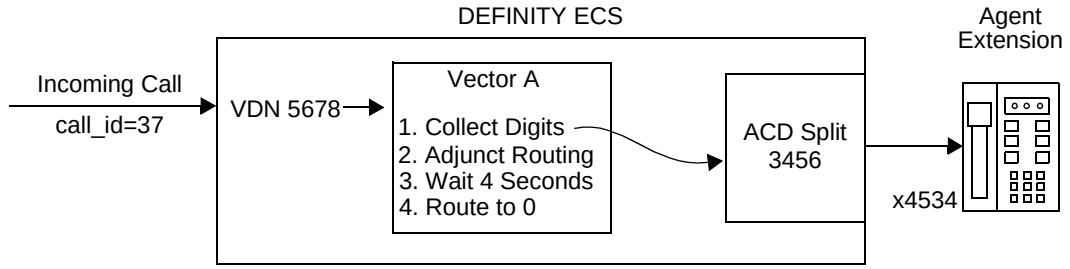


Figure B-12. Call Flow for Incoming Call to VDN Routed to ACD Split

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=26, Event Report, call_id=37, event=call offered, calling number=2015666379, called number=9085765678, II Digits=5, UUI=98765421, domain=VDN 5678)]	Call Offered event provided UUI and II Digits.
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, call_id=37, calling number=2015666379, called number=9085765678, user code (CINFO) =123, II Digits = 5, UUI=987654321)]	Route Request provides CINFO which was collected during the Collect step
FAC [CRV=93, FIE(Invoke, Inv_id=7, Route Select, called number=3456)]		
	REL COMP [CRV=93, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	

4. Call Control Associations

This section presents several scenarios for Call Control capabilities. It also provides the **Event Reports** received over Call Control associations.

Third Party Make Call to ACD Split 3456

This scenario shows the call flow for a **Third_Party_Make_Call** from extension 6097 to ACD split 3456 (see Figure B-13). The call is cleared by a **Third_Party_Clear_Call** request after it queues into ACD split 3456. Event Notification for ACD Split 3456 is active as presented in the initialization scenario.

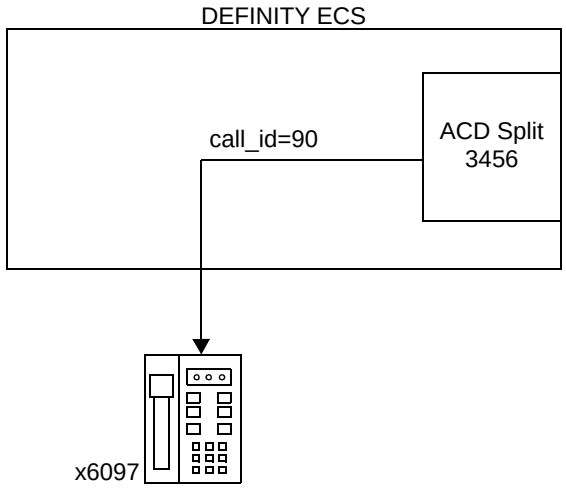


Figure B-13. Call Flow for Third_Party_Make_Call to ACD

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
REG [CRV=65, FIE(Invoke, Inv_id=1, 3P Make Call, called number-3456, calling number=6097)]		AP Requests Make Call from Station 6097 to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=2, Event Report, call_id=90, event=call offered, calling number=6097 called number=3456 domain=ACD Split 3456)]	Call Offered Event Report — Notification Association
	FAC [CRV=65, FIE(Invoke, Inv_id=2, Event Report, call_id=90, event=queued, called number=3456, calls in queue=1, domain=ACD Split 3456)]	Queued Event Report (Call Queues to ACD Split 3456) — 3P Make Call Association
	FAC [CRV=102, FIE(Invoke, Inv_id=4, Event Report, call_id=90, event=queued, called number=3456, calls in queue=1, domain=ACD Split 3456)]	Queued Event Report (Call Queues to ACD Split 3456) — Notification Association
FAC [CRV=65. FIE(Invoke Inv_ id=3, 3P Clear Call, call_id=90)]		AP Requests Clear Call — 3P Make Call Association
	REL COMP [CRV=65,FIE(Return Result, Inv_id=3)]	Call Cleared 3P Make Call Association Ended
	FAC [CRV=102, FIE(Invoke Inv_id=6,Call Ended, call_id=90, cause=normal clearing)]	Call Terminated (Because of Clear Request) — Notification Association

Third Party Merge

This scenario shows the call flow and event reports for a conference operation performed by the adjunct processor via Third Party Call Control capabilities (See Figure B-14). Station 6022 initiates two **Third_Party_Make_Calls** to external destinations and conferences the called destinations using **Third_Party_Merge**. Both ISDN and non-ISDN facilities (trunks) are used in this scenario.

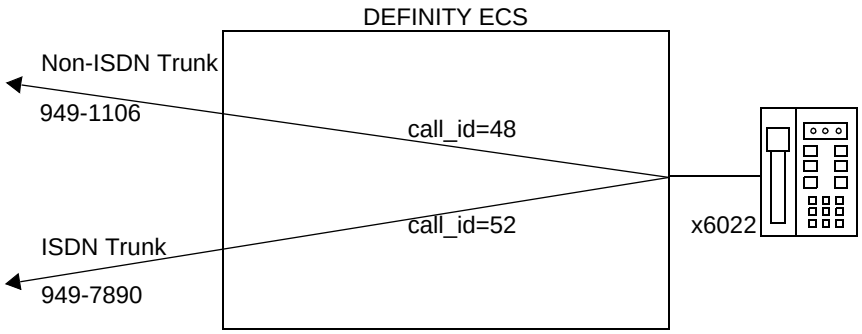


Figure B-14. Call Flow for Third_Party_Merge

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
REG [CRV=81, FIE(Invoke, Inv_id=1, 3P Make Call, called number=9491106, calling number=6022, dest_route_select=9)]		AP Requests Make Call from Station 6022 to External Destination
	FAC [CRV=81, FIE(Invoke, Inv_id=2, Event Report, call_id=48, party_id=2, event=trunk seized, called number=#####)]	Trunk Seized Event Report (Off-premises Call)
FAC [CRV=81, FIE(Invoke, Inv_id=3, 3P Selective Hold, party_id=1)]		AP Requests Hold
	FAC [CRV=81, FIE(Return Result, Inv_id=3)]	Call Placed On Hold
REG [CRV=80, FIE(Invoke, Inv_id=1, 3P Make Call, called number=99497890, calling number=6022)]		AP Requests Make Call from Station 6022 to External Destination
	FAC [CRV=80, FIE(Invoke, Inv_id=2, Event Report, call_id=52, party_id=2, event=alerting, calling number=9085766022, called number=2019497890, connected number=#####)]	Alerting Event Report (ISDN PRI Trunk Used)
	FAC [CRV=80, FIE(Invoke, Inv_id=4, Event Report, call_id=52, party_id=2, event=connected, calling number=9085766022 called number=2019497890, connected number=2019493456)]	Connected Event Report (ISDN Provides Connected Party)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
FAC [CRV=81, FIE(Invoke, Inv_id=5, 3P Merge, call_id=52, party_id=1 conf/transf flag=transfer)]		AP Requests Conference
	FAC [CRV=81, FIE(Return Result, Inv_id=5, 3P Merge, call_id=52, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, party_id=3, connected number=6022, connected number=#####, connected number=#####)]	Calls Conferenced
	FAC [CRV=80, FIE(Invoke, Inv_id=6, Event Report, other_call_id=48, resulting_call_id=52, event=call conferenced, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, party_id=3, connected number=6022, connected number=#####, connected number=#####, calling number=6022, called number=#####)]	Call Conferenced Event Report
	REL COMP [CRV=81, FIE(Invoke, Inv_id=4, Call Ended, call_id=48, cause=call cleared)]	3P Association Terminated (Because of Merge)

Third Party Send DTMF

Continuing from the Third Party Merge Scenario the previous page (refer to Figure B-14), the adjunct sends a string of DTMF digits to all parties connected on this call.

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
FAC [CRV=81, FIE(Invoke), Inv_id=5, 3P Send DTMF, party_id=1, digits=12345)]		
	FAC [CRV=81, FIE(Return Result), Inv_id=5]	
		Digits are now being sent to all parties on call_id 52.

User Scenario: Selective Listening

This scenario continues the The Third Party Merge Scenario (refer to Figure B-14) and shows the flow of messages for a call for which Selective Listening has been invoked for extension 6022.

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
FAC [CRV=81, FIE(Invoke), Inv_id=3, Selective Listening Disconnect, party_id=1]		
	FAC [CRV=81, FIE (Return Result,Inv_id=3)]	Extension 6022 can no longer listen to either party on the call.
FAC [CRV=81, FIE(Invoke), Inv_id=3, Selective Listening Reconnect, party_id=1]		
	FAC [CRV=81, FIE (Return Result,Inv_id=3)]	
		Extension 6022 can now listen to the other two parties on the call.

5. Switch-Classified Calls

This section presents several sample call flows for switch-classified calls (**Third_Party_Make_Calls** with the alert_dest_first and service_circuit options) originated from ACD Split 3456.

Switch-Classified Call Receives SIT Tone

This scenario shows a switch-classified call from ACD Split 3456 to an external destination (see Figure B-15). The call classifier detects a “Vacant Code” Special Information Tone (SIT) from the network and the ECS reports the outcome to the adjunct processor. The ECS has been administered to drop and/or disconnect calls that receive this SIT tone.

NOTE:
For all scenarios in this section, assume that the Feature-Related Parameter Call Classification after Answer Supervision is disabled.

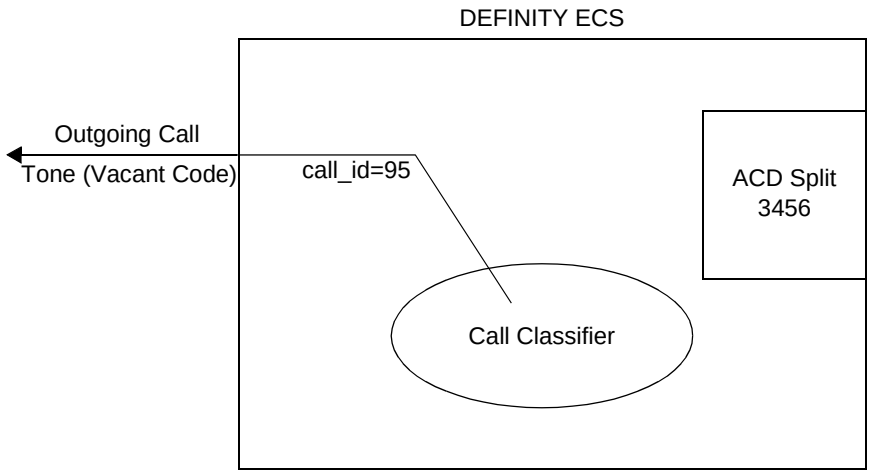


Figure B-15. Call Flow for Switch-Classified Call — SIT Tone Detected

Messages Sent by the Adjunct Processor	Messages Sent by the ECS	Comments
REG [CRV=92, FIE(Invoke, Inv_id=1, 3P Make Call, called number=8003705568, calling number=3456, dest route select=9, max_rings=7, service circuit=call classifier, alert_dest_first=yes)]		AP Requests Make Call (Switch-Classified) from ACD Split 3456 to External Destination
	REL COMP [CRV=92, FIE(Invoke, Inv_id=2, Call Ended, call_id=95, cause=unassigned number)]	Call Terminates (Vacant Code Detected) — Association Ends

Switch-Classified Call Delivered to Agent

This scenario shows a switch-classified call for ACD split 3456 to an external destination (see Figure B-16). The call, after being answered by the external destination, is delivered to the originating ACD split. The agent at extension 3567 and logged into ACD Split 3456 handles the call. ISDN PRI facilities are used for the call.

Event Notification for ACD Split 3456 is active as presented in the initialization scenario. In addition, this scenario shows **Stop_Call_Notification** and **Third_Party_Take_Control** capabilities.

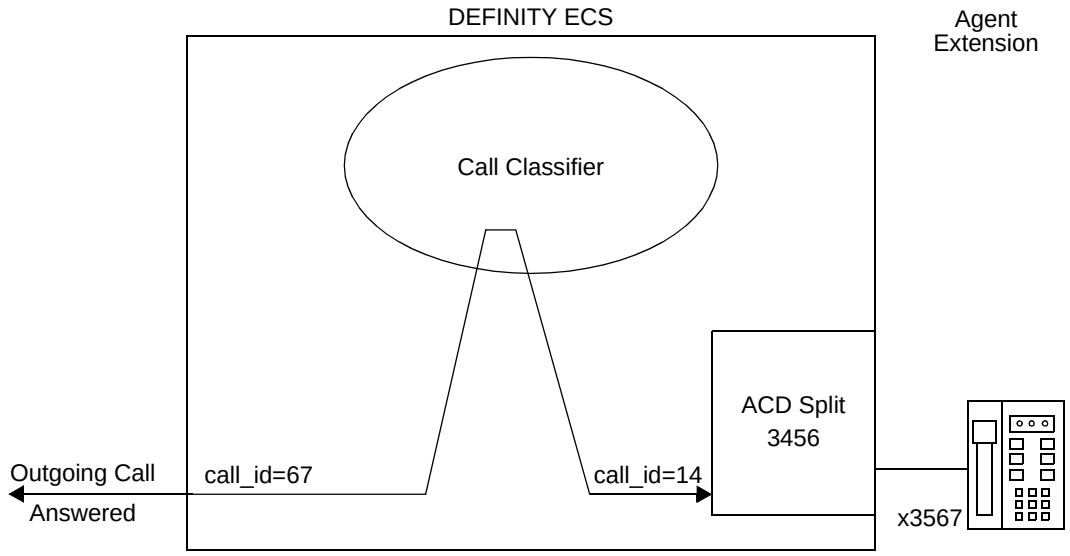


Figure B-16. Call Flow for Switch-Classified Call — Call Answered and Delivered to Agent

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=92, FIE(Invoke, Inv_id=3, 3P Make Call, called number=2018705118, calling number=3456, dest route select=9, max_rings=7, service circuit=call classifier, alert_dest_first=yes)]		AP Requests Make Call (Switch-Classified) from ACD Split 3456 to External Destination
	FAC [CRV=92, FIE(Invoke, Inv_id=2, Event Report, call_id=67, party_id=2, event=alerting, calling number=8095763456, called number=2018705118, connected number=#####)]	Alerting Event Report (Destination Party Alerts) — 3P Make Call Association
	FAC [CRV=92, FIE(Invoke, Inv_id=4, Event Report, call_id=67, party_id=2, event=answered, called number=2018705118, connected number=2018705118, cause=normal)]	Answered Event Report (Destination Answers Call) — 3P Make Call Association
	FAC [CRV=102, FIE(Invoke, Inv_id=2, Event Report, call_id=14, event=call offered, calling number=2018705118, called number=3456, domain=ACD Split 3456)]	Call Offered to ACD Split 3456 —
	FAC [CRV=102, FIE(Invoke, Inv_id=4, Event Report, call_id=14, event=queued, called number=3456, calls in queue=2, domain=ACD Split 3456)]	Queued Event Report (Call Queues to ACD Split 3456) — ACD Split Association
	FAC [CRV=92, FIE(Invoke, Inv_id=6, Event Report, call_id=14, event=queued, called number=3456, calls in queue=2, domain=ACD Split 3456)]	Queued Event Report (Call Queues to ACD Split 3456) — 3P Make Call Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=6, Event Report, call_id=14, party_id=3, event=alerting, calling number=2018705118, called number=3456, connected number=3567, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 3567) — ACD Split Association
	FAC [CRV=92, FIE(Invoke, Inv_id=6, Event Report, call_id=14, party_id=3, event=alerting, calling number=2018705118, called number-3456, connected number=3567, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 3567) — 3P Make Call Association
FAC [CRV=92, FIE(Invoke, Inv_id=5, 3P Relinquish Control)]		Terminate 3P Association
	REL COMP [CRV=92, FIE(Return Result, Inv_id=5)]	3P Association Terminated
	FAC [CRV=102, FIE(Invoke, Inv_id=8, Event Report, call_id=14, party_id=3, event=connected, calling number=2018705118, called number=3456, connected number=3567)]	Connected Event Report (Call connected to Agent 3567) — ACD Split Association
FAC [CRV=102, FIE(Invoke, Inv_id=3, Stop Call Notification, call_id=14)]		Stop Event Reports for Call over Notification Association
	FAC [CRV=102, FIE(Return Result, Inv_id=3)]	Event Reporting for Call Terminated
REG [CRV=91, FIE(Invoke, Inv_id=1, 3P Take Control, call_id=14)]		Request Control of Call

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=91, FIE(Return Result, Inv_id=1, 3P Take Control, party_id=1, party_id=3, connected number=#####, connected number=3567)]	Control Granted
	FAC [CRV=91, FIE(Invoke, Inv_id=2, Event Report, call_id=14, party_id=1, event=drop, connected number=#####, cause=normal clearing)]	Drop Event Report (Caller Disconnects) 3P Take Control Association
	REL COMP[CRV=91, FIE(Invoke, Inv_id=4, Call Ended, call_id=14, cause=normal clearing)]	Call Terminated (Agent Disconnects) — 3P Take Control Association Ends

Switch-Classified Call From ACD Split Forwarded to VDN

This scenario shows the call flow for a switch-classified call with ACD Split 3456 as the originator (see Figure B-17). The ACD split 3456 has call forwarding active to VDN 8901. VDN 8901 routes the call to extension 5001.

Event Notification for ACD Split 3456 is active as presented in the initialization scenario. VDN 8901 and ACD Split 2111 do not have Event Notification active.

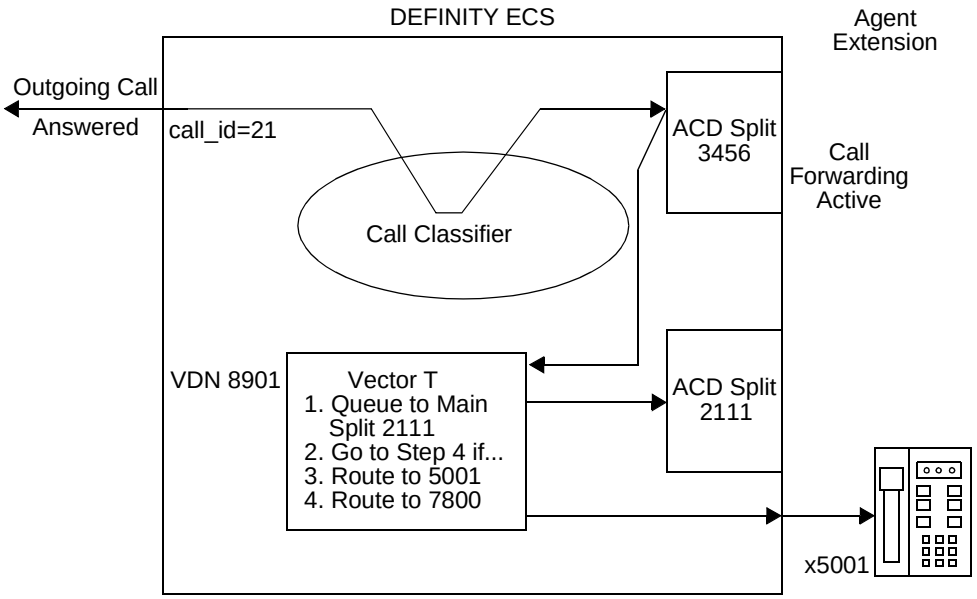


Figure B-17. Call Flow for Switch-Classified Call — Originator Forwards Call

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=92, FIE(Invoke, Inv_id=1, 3P Make Call, called number=19008705118, calling number=3456, dest route select=9, max_rings=3, service circuit=call classifier, alert_dest_first=yes)]		AP Requests Make Call (Switch-Classified) from ACD Split 3456 to External Destination
	FAC [CRV=92, FIE(Invoke, Inv_id=2, Event Report, call_id=21, party_id=2, event=alerting, calling number=3456, called number=9008705118, connected number=######)]	Alerting Event Report (Destination Alerts) — 3P Make Call Assoc. — Note that the ISDN Network returns a different alerting number than the one dialed
	FAC [CRV=92, FIE(Invoke, Inv_id=4, Event Report, call_id=21, party_id=2, event=answered, called number=9008705118, connec. number=2018793345, cause=normal)]	Answered Event Report (Destination Answers Call) — 3P Make Call Association — Note that the ISDN Network returns a different connected number than the one dialed
	FFAC [CRV=102, FIE(Invoke, Inv_id=2, Event Report, call_id=76, event=call offered, calling number=9008705118, called number=3456, domain=ACD split 3456)]	Call Offered Event Report ACD Split Association
	FAC [CRV=92, FIE(Invoke, Inv_id=6, Event Report, call_id=76, event=queued, called number=3456, calls in queue=2, domain= ACD Split 2111)]	Queued Event Report (Call Queues to ACD Split 2111 via VDN 8901) — 3P Make Call Association
	FAC [CRV=102, FIE(Invoke, Inv_id=4, Event Report, call_id=76, event=queued, called number=3456, calls in queue=2, domain= ACD Split 2111)]	Queued Event Report (Call Queues to ACD Split 2111 via VDN 8901) — ACD Split Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=92, FIE(Invoke, Inv_id=8, Event Report, call_id=76, party_id=2, event=alerting, calling number=9008705118, called number=3456, connected number=5001)]	Alerting Event Report (Call Delivered to Station 5001 via a Route to Number Vector Command) — 3P Make Call Association
	FAC [CRV=102, FIE(Invoke, Inv_id=6, Event Report, call_id=76, party_id=2, event=alerting, calling number=9008705118, called number=3456, connected number=5001)]	Alerting Event Report (Call Delivered to Station 5001 via a Route to Number Vector Command) — ACD Split Association
	FAC [CRV=92, FIE(Invoke, Inv_id=10, Event Report, call_id=76, party_id=2, event=connected, calling number=9008705118, called number=3456, connected number=5001)]	Connected Event Report (Call Connected to Station 5001) — 3P Make Call Association
	FAC [CRV=102, FIE(Invoke, Inv_id=8, Event Report, call_id=76, party_id=2, event=connected, calling number=9008705118, called number=3456, connected number=5001)]	Connected Event Report (Call Connected to Station 5001) — ACD Split Association
FAC [CRV=92, FIE(Invoke, Inv_id=5, 3P Relinquish Control)]		Terminate 3P Association
	REL COMP[CRV=92, FIE(Return Result, Inv_id=5)]	3P Association Terminated
FAC [CRV=102, FIE(Invoke, Inv_id=3, Stop Call Notification, call_id=76) Control)]		Stop Event Reports for Call— ACD Split Association
	FAC [CRV=102, FIE(Return Result, Inv_id=3)]	Event Reporting for Call Terminated

6. Multiple Monitors and Controllers

This section presents the **Event Reports** sent by the switch when two monitored and controlled calls are merged (call transferred to another call). The scenario shows an incoming ISDN PRI call to ACD Split 6000 that is transferred by ACD agent 6010 to ACD Split 5100 (see Figure B-18). Agent 5101 answers the transferred call.

The adjunct processor has Event Notification active for ACD Splits 6000 and 5100. In addition, the adjunct processor controls, via Call Control associations, the calls.

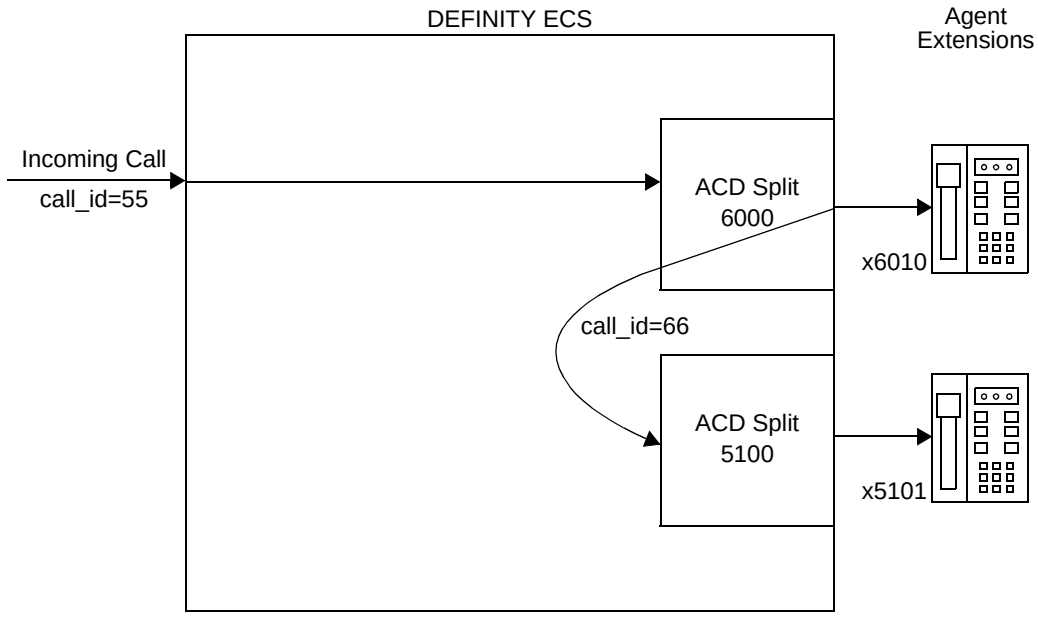


Figure B-18. Call Flow for Merging Two Calls with Event Notification and Call Control

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=10, FIE(Invoke, Inv_id=1, Event Notification Request, domain = ACD Split 6000)]		AP Requests Event Notification for ACD Split 6000
	FAC [CRV=10, FIE(Return Result, Inv_id=1)]	Event Notification Request Accepted
REG [CRV=20, FIE(Invoke, Inv_id=1, Event Notification Request, domain = ACD Split 5100)]		AP Requests Event Notification for ACD Split 5100
	FAC [CRV=20, FIE(Return Result, Inv_id=1)]	Event Notification Request Accepted
	FAC [CRV=10, FIE(Invoke, Inv_id=2, Event Report, call_id=55, event=call offered, calling number=8135308648, called number=8095766000, connected number=6010, domain=ACD Split 6000)]	Call Offered to ACD Split 6000 — ACD Split 6000 Event Notification Assoc.
	FAC [CRV=10, FIE(Invoke, Inv_id=4, Event Report, call_id=55, party_id=2, event=alerting, calling number=8135308648, called number=8095766000, connected number=6010 domain = ACD Split 6000)]	Alerting Event Report (Call Delivered to Agent 6010) — ACD Split 6000 Event Notification Association
	FAC [CRV=10, FIE(Invoke, Inv_id=6, Event Report, call_id=55, party_id=2, event=connected, calling number=8135308648, called number=8095766000, connected number=6010)]	Connected Event Report (Call Connected to Agent 6010) — ACD Split 6000 Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=11, FIE(Invoke, Inv_id=1, 3P Take Control, call_id=55)]		AP Request to Take Control of Call
	FAC [CRV=11, FIE(Return Result, Inv_id=1, 3P take Control, party_id=1, party_id=2, connected number=#####, connected number=6010)]	3P Take Control Request Accepted
FAC [CRV=11, FIE(Invoke, Inv_id=3, 3P Selective Hold, party_id=2)]		AP Requests to Place Call On Hold
	FAC [CRV=11, FIE(Return Result, Inv_id=3)]	Call Placed on Hold — 3P Take Control Assoc.
	FAC [CRV=10, FIE(Invoke, Inv_id=8, Event Report, call_id=55, event=hold party_id=2, connected number=6010)]	Hold Event Report (Call at Station 6010 Placed On Hold) — ACD Split 6000 Notification Association
REG [CRV=22, FIE(Invoke, Inv_id=1, 3P Make Call, calling number=6010, called number=5100, return ack=yes)]		AP Requests to Make Call from Station 6010 to ACD Split 5100
	FAC [CRV=22, FIE(Invoke, Inv_id=2, 3P Make Call Proceed, party_id=1)]	Third Party Make Call Acknowledgment
	FAC [CRV=20, FIE(Invoke, Inv_id=2, Event Report, call_id=66, event=call offered, calling number=5100, called number=6010, domain=ACD Split 5100)]	Call Offered to ACD Split 5100 — ACD Split 5100 Event Notification Association
	FAC [CRV=22, FIE(Invoke, Inv_id=4, Event Report, call_id=66, event=queued, called number=5100, calls in queue=1, domain=ACD split 5100)]	Queued Event Report (Call Queues to ACD Split 5100) — 3P Make Call Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=20, FIE(Invoke, Inv_id=4, Event Report, call_id=66, event=queued, called number=5100, calls in queue=1, domain=ACD split 5100)]	Queued Event Report (Call Queues to ACD Split 5100) — ACD Split 5100 Event Notification Association
REG [CRV=11, FIE(Invoke, Inv_id=5, 3P Merge, call_id=66, common_party_id=2, conf/transf flag=transfer)]		AP Requests Transfer of Held Call to ACD Split 5100
	FAC [CRV=11, FIE(Return Result, Inv_id=5, 3P Merge, call_id=66, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, connected number=#####, connected number=5100)]	Call Transferred — 3P Take Control Association
	FAC [CRV=20, FIE(Invoke, Inv_id=8, Event Report, other_call_id=55, resulting call_id=66, event=call transferred, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, connected number=#####, connected number=5100, calling number=6010, called number=5100)]	Call Transferred Event Report (Transfer to ACD Split 5100 Completed) — ACD Split 5100 Event Notification Association
	FAC [CRV=22, FIE(Invoke, Inv_id=8, Event Report, other_call_id=55, resulting call_id=66, event=call transferred, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, connected number=#####, connected number=5100, calling number=6010, called number=5100)]	Call Transferred Event Report (Transfer to ACD Split 5100 Completed) — 3P Make Call Association

(Continued on next page)

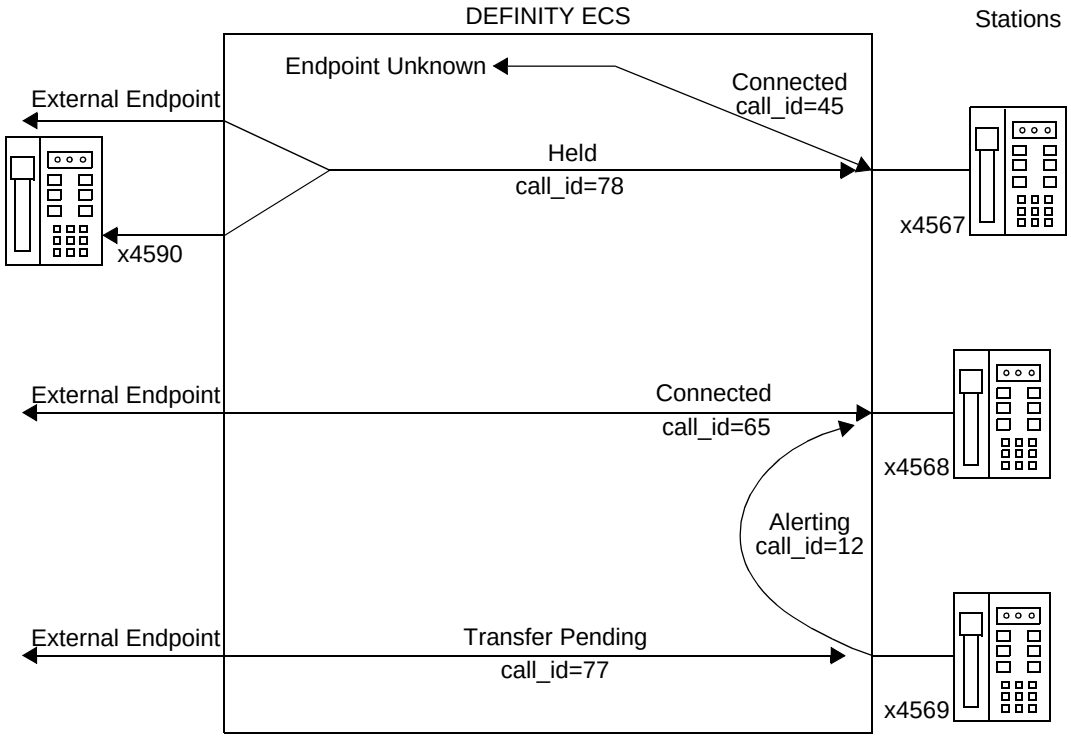
Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	REL COMP [CRV=11, FIE(Invoke, Inv_id=8, Call Ended, call_id=55, cause=call cleared)]	3P Take Control Association Ends — Call Merged with Another Call
	FAC [CRV=10, FIE(Invoke, Inv_id=8, Call Ended, call_id=55, cause=call cleared)]	Event Reports for Call Ended (Call Merged) — ACD Split 6000 Event Notification Association
	FAC [CRV=22, FIE(Invoke, Inv_id=4, Event Report, call_id=66, party_id=2, event=alerting, calling number=8135308648, called number=5100, connected number=5101)]	Alerting Event Report (Station 5101 Alerts) — 3P Make Call Association
	FAC [CRV=20, FIE(Invoke, Inv_id=10, Event Report, call_id=66, party_id=2, event=alerting, calling number=8135308648, called number=5100, connected number=5101)]	Alerting Event Report (Station 5101 Alerts) — ACD Split 5100 Event Notification Association
FAC [CRV=20, FIE(Invoke, Inv_id=5, Stop Event Notification call_id=78)]		AP Requests to Stop Receiving Event Reports for the Call
	FAC [CRV=20, FIE(Return Result, Inv_id=5)]	Event Notification for Call Ended
	FAC [CRV=22, FIE(Invoke, Inv_id=6, Event Report, call_id=66, party_id=2, event=connected, calling number=8135308648, called number=5100, connected number=5101)]	Connected Event Report (Station 5101 Answers) — 3P Make Call Association
	FAC [CRV=22, FIE(Invoke, Inv_id=12, Event Report, call_id=66, party_id=2, event=drop, connected number=5101, cause=normal clearing)]	Drop Event Report (Station 5101 Disconnects) — 3P Make Call Association
	REL COMP [CRV=22, FIE(Invoke, Inv_id=14, Call Ended, cause=normal clearing)]	Call Terminates — 3P Make Call Association Terminates

7. Domain (Station) Control Associations

This section presents call scenarios for Domain Control (Station Control) associations.

Domain Control Initiation

This scenario shows an adjunct processor requesting **Domain Control** of extensions 4567, 4568, and 4569. Figure B-19 shows the state of the calls present at the three domain-controlled stations. For instance, extension 4567 starts with two calls: one connected/active (call_id=45) and another held (call_ id=78).



Controlled Stations — Initial Call States

x4567	x4568	x4569
call_id=45 — Connected	call_id=12 — Alerting	call_id=12 — Connected
call_id=78 — Held	call_id=65 — Connected	call_id=77 — Other (Xfer Pending)

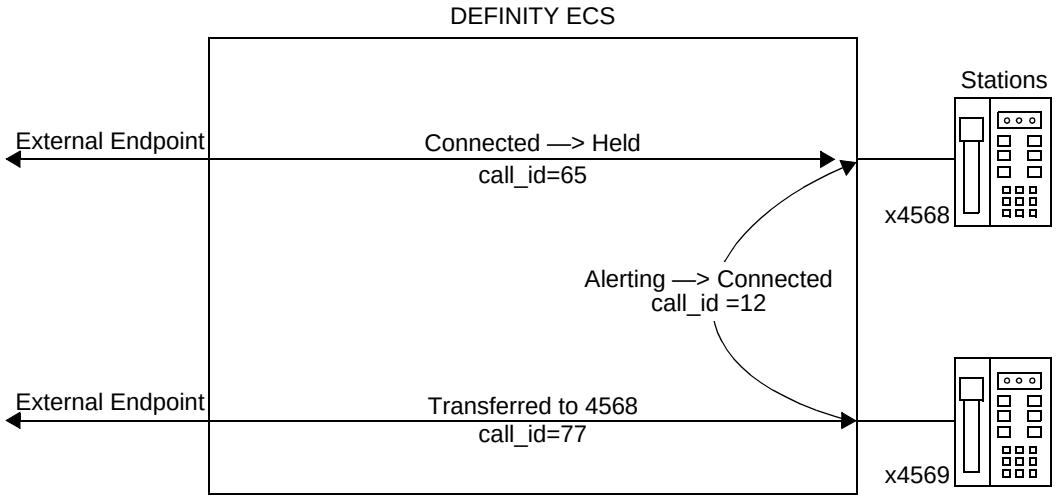
Figure B-19. Initial Call States for Domain (Station) Control

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=101, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 4567)]		AP Requests Domain Control of Station 4567
	FAC [CRV=101, FIE(Return Result, Inv_id=1, 3P Domain Control, call_id=45, call_id=78, party_id=1, party_id=3, call_state=connected, call_state=held)]	Domain Control Granted
REG [CRV=100, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 4568)]		AP Requests Domain Control of Station 4568
	FAC [CRV=100, FIE(Return Result, Inv_id=1, 3P Domain Control, call_id=12, call_id=65, party_id=2, party_id=1, call_state=alerting, call_state=connected)]	Domain Control Granted
REG [CRV=99, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 4569)]		AP Requests Domain Control of Station 4569
	FAC [CRV=99, FIE(Return Result, Inv_id=1, 3P Domain Control, call_id=12, call_id=77, party_id=1, party_id=1, call_state=connected, call_state=other)]	Domain Control Granted

Domain (Station) Control — Manual Transfer

This scenario shows a manual transfer operation for Domain Control stations (see Figure B-20). Station 4568 places on hold the active call (call_id=65) and answers the alerting call (call_id=12). Station 4569 completes the manual call transfer operation that had been initiated before Domain (Station) Control was requested.

Domain Control for stations 4568 and 4569 is active as presented in the Domain Control initiation scenario.



Controlled Stations — Call State Transitions

x 4568
call_id=12 — Alerting —> Connected
call_id=65 — Connected —> Held

x 4569
call_id=12 — Connected —> Transferred
call_id=77 — Other (Xfer Pending) —> Merged with call_id=12

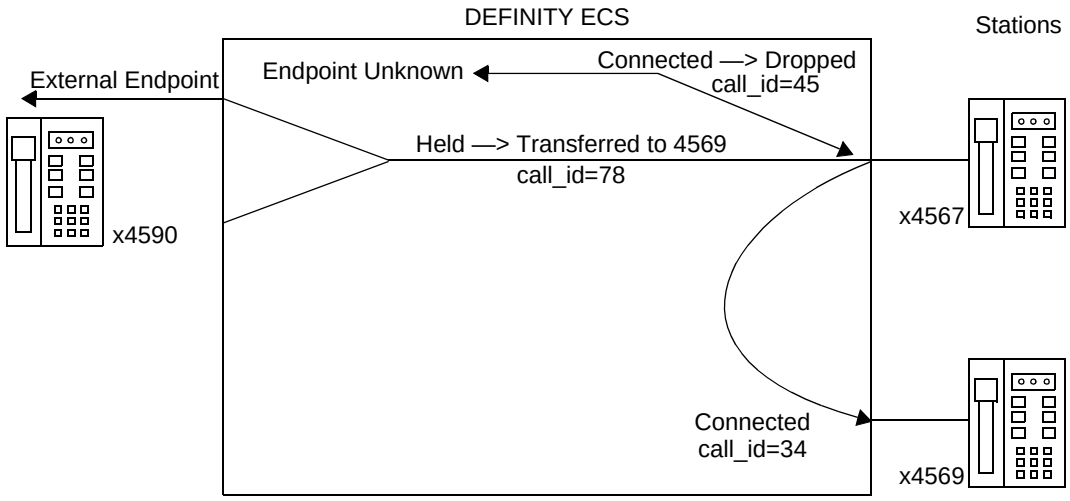
Figure B-20. Call Flow for Domain Control — Manual Transfer

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=100, FIE(Invoke, Inv_id=6, Event Report, call_id=65, party_id=1 event=hold, connected number=4568)]	Hold Event Report (Station 4568 Places Call on Hold) — x4568 Association
	FAC [CRV=100, FIE(Invoke, Inv_id=8, Event Report, call_id=12, party_id=2, event=connected, calling number=4569, called number=4568, connected number=4568)]	Connected Event Report (4568 Connects to Alerting Call) — x4568 Association
	FAC [CRV=99, FIE(Invoke, Inv_id=10, Event Report, call_id=12, party_id=2, event=connected, calling number=4569, called number=4568, connected number=4568)]	Connected Event Report (4568 Connects to Alerting Call) — x4569 Association
	FAC [CRV=100, FIE(Invoke, Inv_id=10, Event Report, other_call_id=77, resulting_call_id=12, event=call transferred, calling number=4569, called number=4568, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, connected number=4568, connected number=#####)]	Call Transferred Event Report (x4569 Transfers Call to x4568) — x4568 Association
	FAC [CRV=99, FIE(Invoke, Inv_id=4, Event Report, other_call_id=77, resulting_call_id=12, event=call transferred, calling number=4569, called number=4568, old party_id=resulting call 1, old party_id=resulting call 2, old party_id=oldcall1, party_id=1, party_id=2, connected number=4568, connected number=#####)]	Call Transferred Event Report (x4569 Transfers Call to x4568) — x4569 Association

Domain (Station) Control — Third Party Merge

This scenario shows the call flow for a Third Party Merge capability. Station 4567, after disconnecting from the active call (call_id=45), reconnects to the held call and transfers the call (call_id=78) to station 4569 (see Figure B-21).

This scenario is a continuation of the scenario shown in Figure B-19.



Controlled Stations — Call State Transitions

x 4567
call_id=45 — Connected —> Dropped
call_id=78 — Held —> Connected —> Held —> Merged with call_id=34
call_id=34 — New Call —> Transferred to x4569

x 4569
call_id=34 — Alerting —> Connected

Figure B-21. Call Flow for Third_Party_Merge over a Domain Control

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=101, FIE(Invoke, Inv_id=7, 3P Merge, call_id=34, call_id=78, conf/transf flag=transfer)]		AP Requests Transfer of Held Call to Station 4569
	FAC [CRV=101, FIE(Return Result, Inv_id=7, 3P Merge, call_id=34, party_id=1, party_id=2, party_id=3, old_party_id=other call 1 old_party_id=other call 2 old_party_id=reslt call 3 connected number=4590, connected number=#####, connected number=4569)]	Call Transferred — x4567 Association
	FAC [CRV=99, FIE(Invoke, Inv_id=8, Event Report, other_call_id=78, resulting_call_id=34, event=call transferred, party_id=1, party_id=2, party_id=3, old_party_id=other call 1 old_party_id=other call 2 old_party_id=reslt call 3 connected number=4590, connected number=#####, connected number=4569, calling number=4567, called number=4569)]	Call Transferred Event Report (Station 4567 Transfer Completed) — x4569 Association
	FAC [CRV=99, FIE(Invoke, Inv_id=10, Event Report, call_id=34, party_id=3, event=connected, calling number =4590, called number=4569, connected number=4569)]	Connected Event Report (Station 4569 Connects to Transferred Call) — x4569 Association

Value Queries for Calls at Domain Control Stations

This scenario shows the Call, Party_id, and Station Status **Value_Queries** for stations 4567, 4568, and 4569 (see Figure B-22).

This scenario is a continuation of the previous scenario.

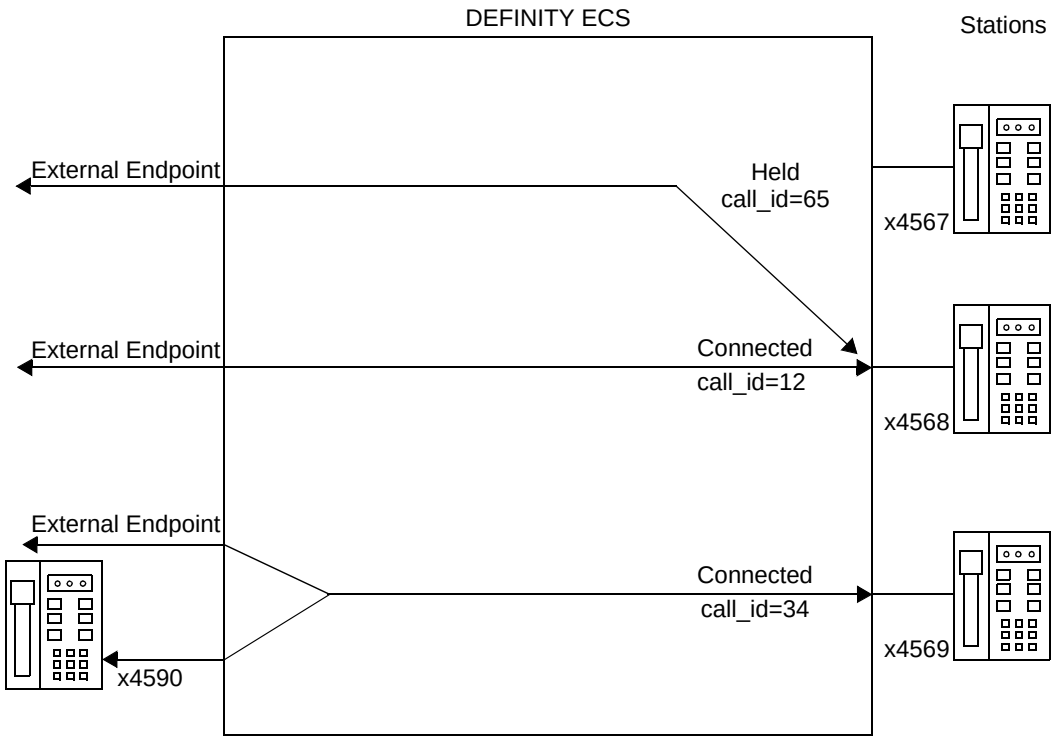


Figure B-22. Calls Present at Domain Controlled Stations 4567, 4568, and 4569

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=87, FIE(Invoke, Inv_id=1, Value Query, item=Calls Query, domain=extension 4567)]		AP Requests the Status of Calls at Station 4567
	REL COMP [CRV=87, FIE Return Result, Inv_id=1)]	No Calls Present at Station 4567
REG [CRV=86, FIE(Invoke, Inv_id=1, Value Query, item=Calls Query, domain= extension 4568)]		AP Requests the Status of Calls at Station 4568
	REL COMP [CRV=86, FIE Return Result, Inv_id=1, Value Query, call_id=12, call_id=65, party_id=1, party_id=1, call_state=connected, call_state=held)]	Two Calls Present at Station 4568
REG [CRV=86, FIE(Invoke, Inv_id=1, Value Query, item=Party_id, call_id=12)]		AP Requests the Parties Connected to Call 12
	REL COMP [CRV=86, FIE (Return Result, Inv_id=1, Value Query, party_id=1, party_id=2, connected number=4568, connected number=#####)]	Two Parties Connected to Call (Party 2 is a Trunk)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=54, FIE(Invoke, Inv_id=1, Value Query, item=Calls Query, domain= extension 4569)]		AP Requests the Status of Calls at Station 4569
	REL COMP [CRV=54, FIE (Return Result, Inv_id=1, Value Query, call_id=34, party_id=3, call_state=connected)]	One Call Present at Station 4569
REG [CRV=55, FIE(Invoke, Inv_id=1, Value Query, item=Party_id, call_id=34)]		AP Requests the Parties Connected to Call 34
	REL COMP [CRV=55, FIE (Return Result, Inv_id=1, Value Query, party_id=1, party_id=2, party_id=3, connected number=4590, connected number=#####, connected number=4569)]	Three Parties Connected to Call (Party 2 is a Trunk)
FAC [CRV=75, FIE(Invoke, Inv_id=1,Value Query, domain=Station 4569)]		AP Request Station Status for Station 4569
	REL COMP [CRV=75, (Return Result, Inv_id=1, Value Query, domain=talk state idle)]	Station Active on a Call

Domain Control — Call Disconnects

This scenario shows several call drops and/or disconnects for Domain Control stations (see Figure B-23). The call at station 4569 is dropped via a **Third_Party_Selective_Drop** request issued by the Domain Control association for station 4569. The held call at station 4568 is dropped by the held endpoint. The active call at station 4568 is dropped via a **Third_Party_Selective_Drop** request issued over a Call Control association.

This scenario is a continuation of the previous scenario.

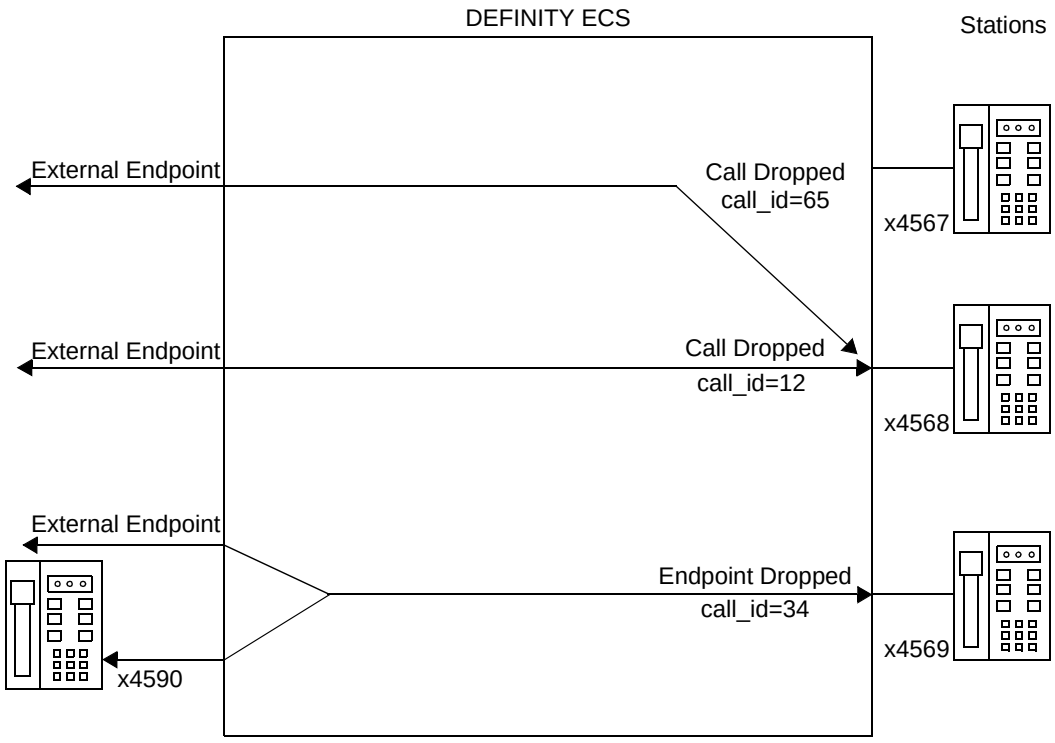


Figure B-23. Call Flow for Call Disconnects

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=99, FIE(Invoke, Inv_id=11, 3P Selective Drop, call_id=34)]		AP Requests to Drop Station 4569 from Call — x4569Association
	FAC [CRV=99, FIE(Return Result, Inv_id=11)]	Station 4569 Dropped from Call — x4569 Association
	FAC [CRV=100, FIE(Invoke, Inv_id=4, Event Report, call_id=65,party_id=2, event=drop, connected number=#####, cause=normal clearing)]	Drop Event Report (Caller Disconnects) — x4568 Association
	FAC [CRV=100, FIE(Invoke, Inv_id=6, Event Report, call_id=65, party_id=1 event=drop, connected number=4568, cause=normal clearing)]	Drop Event Report (Call Terminates) — x3468 Association
REG [CRV=85, FIE(Invoke, Inv_id=1, 3P Take Control, call_id=12)]		AP Requests to Take Control of Call 12
	FAC [CRV=85, FIE(Return Result, Inv_id=1, 3P Take Control, party_id=1, party_id=2, connected number=4568, connected number=#####)]	3P Take Control Granted
FAC [CRV=85, FIE(Invoke, Inv_id=3, 3P Selective Drop, party_id=1)]		AP Requests to Drop x4568
	FAC [CRV=85, FIE(Return Result, Inv_id=3)]	Station 4568 Dropped — 3P Association
	FAC [CRV=100, FIE(Invoke, Inv_id=12, Event Report, call_id=12, party_id=1, event=drop, connected number=4568, cause=normal clearing)]	Drop Event Report (Station 4568 Drops) — x4658 Association
	REL COMP [CRV=85, FIE(Invoke, Inv_id=2, Call Ended, call_id=12)]	Call Terminates — 3P Association

8. Call Redirection

This section presents several scenarios for call redirection (for example, Coverage, Call Forwarding, Send All Calls or ASAI Redirect Alerting Call).

Incoming Call to Station Forwards to Another Station

This scenario shows the call flow for a call from station 3568 to station 3569 (see Figure B-24). Station 3569 has call forwarding active to station 6055. Station 6055 answers the call.

Assume that Third Party Domain (Station) Control is active for stations 3568 and 3569 as shown in the Send All Calls scenario above.

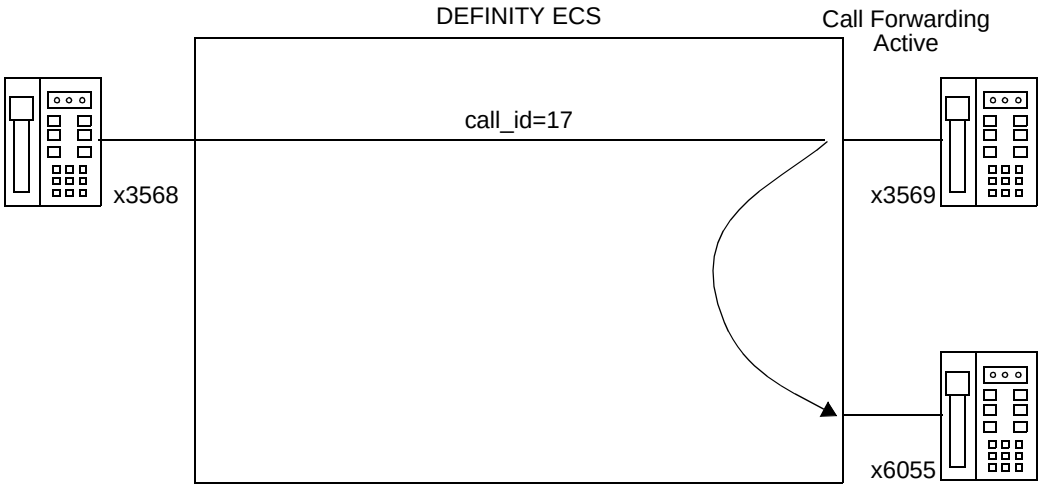


Figure B-24. Call Flow for Incoming Call to Station with Call Forwarding to Another Station

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=79, FIE(Invoke, Inv_id=1, Request Feature, feature=Activate Call Forwarding, redirecting number=3569, redirection number=6055)]		AP Requests to to Activate Call Forwarding for Station 3569
	REL COMP[CRV=79, FIE(Return Result, Inv_id=1)]	Call Forwarding Activated
FAC [CRV=86, FIE(Invoke, Inv_id=3, 3P Auto Dial, called number=3569)]		AP Requests Auto Dial Call to Station 3569 — x3568 Association
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=17, party_id=1, event=call initiated)]	Call Initiated Event Report (Station 3568 Forced Off-Hook) — x3568 Association
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=17, party_id=1, event=call originated, connected number=3568, calling number=3568, called number=3569)]	
	FAC [CRV=86, FIE(Invoke, Inv_id=4, Event Report, call_id=17, party_id=2, event=alerting, calling number=3568, called number=3569, connected number=6055 cause=325)]	Alerting Event Report (Call Delivered to Station 6055) — x3568 Association
	FAC [CRV=86, FIE(Invoke, Inv_id=6, Event Report, call_id=17, party_id=2, event=connected, calling number=3568, called number=3569, connected number=6055)]	Connected Event Report (Station 6055 Answers Forwarded Call) — x3568 Association
	FAC [CRV=86, FIE(Invoke, Inv_id=8, Event Report, call_id=17, party_id=1, event=drop, connected number=3568, cause=normal clearing)]	Drop Event Report (Station 3568 Disconnects from Call) — x3568 Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=79, FIE(Invoke, Inv_id=1, Request Feature, feature=Cancel Call Forwarding, redirecting number=3569)]		AP Cancels Call Forwarding for Station 3569
	REL COMP[CRV=79, FIE(Return Result, Inv_id=1)]	Call Forwarding Cancelled

Incoming Call to Station Forwards to VDN

This scenario shows the call flow for a call from station 3001 to station 2345 that has call forwarding active to VDN 405 (see Figure B-25). VDN 405 routes the call, via the Adjunct Routing vector command, to VDN 5678. VDN 5678 routes the call, via the Adjunct Routing vector command, to extension 9800.

VDN 405 is not monitored by the adjunct. VDN 5678 has Event Notification active as presented in the initialization scenario.

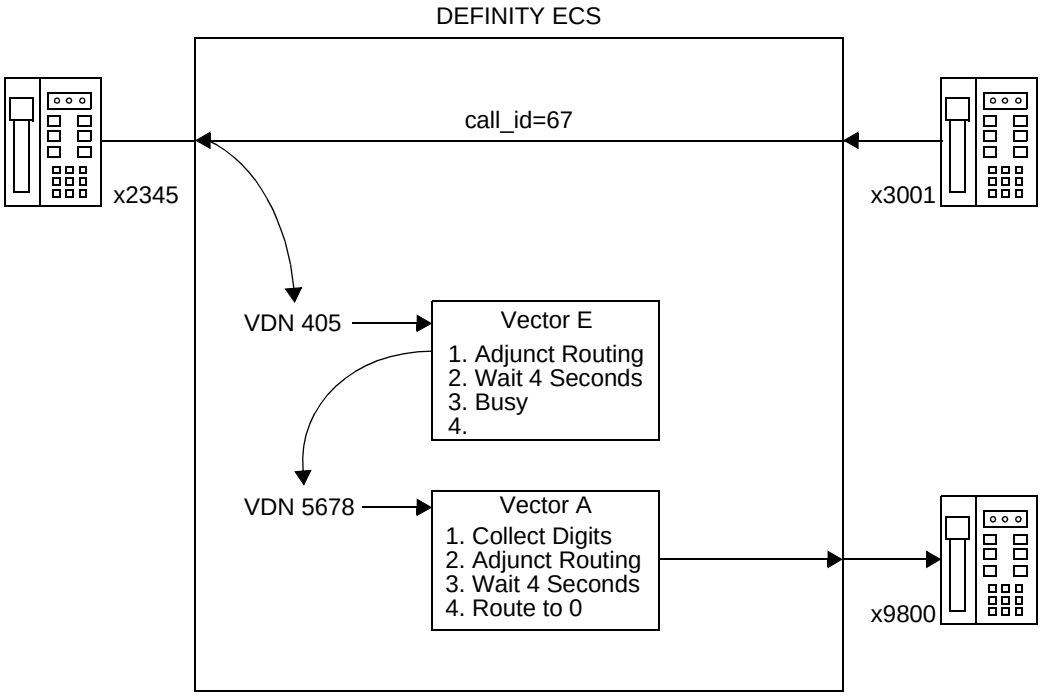


Figure B-25. Call Flow for Incoming Call to Station Forwarded to VDN

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	REG [CRV=32, FIE(Invoke, Inv_id=4, Route, call_id=67, calling number=3001, called number=2345, domain=VDN 405)]	Route Request
FAC [CRV=32, FIE(Invoke, Inv_id=3, Route Select, called number=5678)]		Route to VDN 5678
	REL COMP [CRV=32, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=24, Event Report, call_id=67, event=call offered, calling number=3001, called number=2345, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=90, FIE(Invoke, Inv_id=2, Route, call_id=67, calling number=3001, called number=2345, domain=VDN 5678, user code=1233)]	Route Request

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=90, FIE(Invoke, Inv_id=3, Route Select, called number=9800)]		Route to Station 9800
	REL COMP [CRV=90, FIE(Invoke, Inv_id=4, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=26, Event Report, call_id=67, party_id=2, event=alerting, calling number=3001, called number=2345, connected number=9800)]	Alerting Event Report (Call Alerts Station 9800)
	FAC [CRV=98, FIE(Invoke, Inv_id=28, Event Report, call_id=67, party_id=2, event=connected, calling number=3001, called number=2345, connected number=9800)]	Connected Event Report (Station 9800 Answers)
FAC [CRV=98, FIE(Invoke, Inv_id=9, Stop Call Notification, call_id=67)]		Stop Event Reports for Call over Notification Association
	FAC [CRV=98, FIE(Return Result, Inv_id=9)]	Event Reporting for Call Terminated

Incoming Call to ACD Split Forwards to Hunt Group

This scenario shows the call flow for an incoming ISDN PRI call to ACD split 3456 (see Figure B-26). ACD split 3456 has call forwarding active to hunt group 2000. Member 2005 of the hunt group 2000 receives the call.

ACD split 3456 has Event Notification active as presented in the initialization scenario. The adjunct processor does not have Event Notification active for Hunt Group 2000.

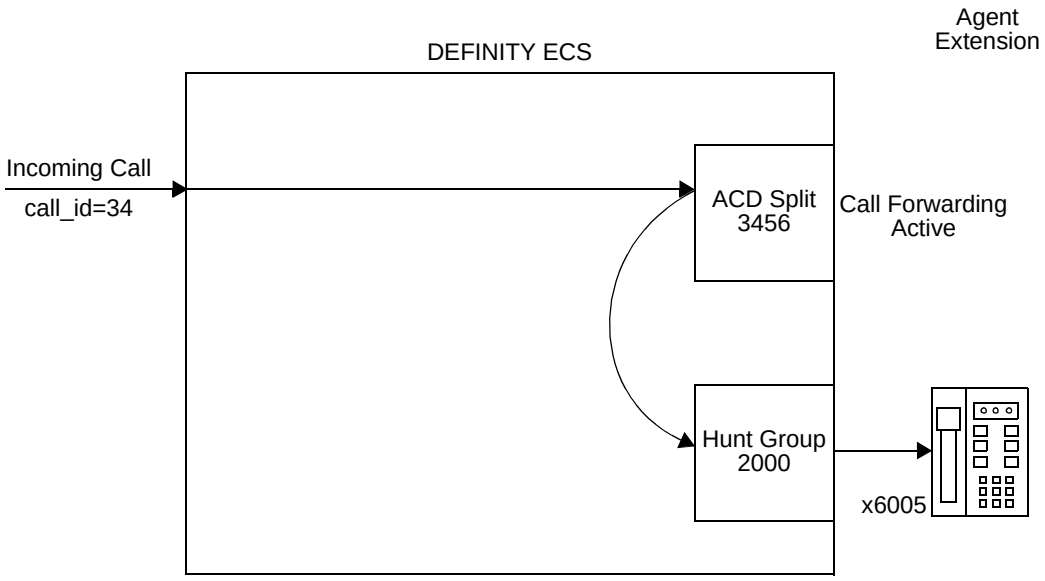


Figure B-26. Call Flow for Incoming Call to ACD Split Forwarded to Another ACD Split

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=24, Event Report, call_id=34, event=call offered, calling number=4159703478, called number=8095763456, domain=ACD Split 3456)]	Call Offered to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=26, Event Report, call_id=34, event=queued, called number=8095763456, calls in queue=1, domain=Hunt Group 2000)]	Call Queues to to Hunt Group 2000
	FAC [CRV=102, FIE(Invoke, Inv_id=28, Event Report, call_id=34, party_id=2, event=alerting, calling number=4159703478, called number=8095763456, connected number=2005, domain= Hunt Group 2000)]	Alerting Event Report (Call Alerts Station 2005)
	FAC [CRV=102, FIE(Invoke, Inv_id=28, Event Report, call_id=34, party_id=2, event=connected, calling number=4159703478, called number=8095763456, connected number=2005)]	Connected Event Report (Station 2005 Answers)
FAC [CRV=102, FIE(Invoke, Inv_id+5, Stop Call Notification, call_id=34)]		AP Requests to Stop Receiving Event Reports for the Call
	FAC [CRV=102, FIE(Return Result, Inv_id=5)]	Event Reporting for Call Terminated

Call to Station Goes to Coverage Station

This scenario shows the call flow for a call from station 6071 to station 3580 (see Figure B-27). After three rings, the call alerts the coverage station (extension 3569). The principal station (3580) answers the call (using its simulated bridged appearance) after it has alerted the coverage station.

Assume that Third Party Domain (Station) Control is active for station 3569 as shown in the Send All Calls scenario above.

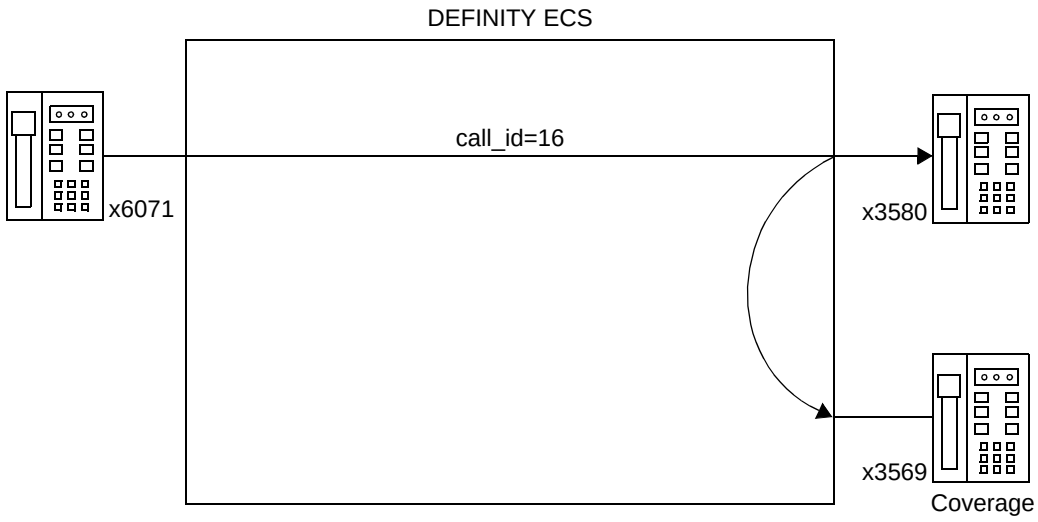


Figure B-27. Call Flow for Incoming Call to Station Redirected to Coverage

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=87, FIE(Invoke, Inv_id=1, 3P Make Call, called number=3580, calling number=6071)]		AP Requests Make Call from Station 6071 to Station 3580
	FAC [CRV=87, FIE(Invoke, Inv_id=2, Event Report, call_id=16, party_id=2, event=alerting, calling number=6071, called number=3580, connected number=3580)]	Alerting Event Report (Call Delivered to Station 3580) — 3P Make Call Association
	FAC [CRV=87, FIE(Invoke, Inv_id=4, Event Report, call_id=16, party_id=3, event=alerting, calling number=6071, called number=3580, connected number=3569 cause=3/28)]	Alerting Event Report (Call Goes to Coverage, Station 3569) — 3P Make Call Association
	FAC [CRV=88, FIE(Invoke, Inv_id=2, Event Report, call_id=16, party_id=3, event-alerting, calling number=6071, called number=3580, connected number=3569 cause=3/28)]	Alerting Event Report (Call Goes to Coverage, Station 3569) — x3569 Association
	FAC [CRV-87, FIE(Invoke, Inv_id=6, Event Report, call_id=16, party_id=2, calling number=6071, event=connected, called number=3580, connected number=3580)]	Connected Event Report (Station 3580 Answers Call)
	FAC [CRV=87, FIE(Invoke, Inv_id=8, Event Report, call_id=16, party_id=3, event=drop, connected number=3569, cause=normal clearing)]	Drop Event Report (Coverage 3569 Dropped)— 3P Make Call Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=88, FIE(Invoke, Inv_id=4, Event Report, call_id=16, event=call redirected)]	Call Redirected Event Report (Principal 3580 Picked Up the Call) — x3569 Association
FAC [CRV=87, FIE(Invoke, Inv_id=3, 3P Selective Drop, party_id=2)]		AP Requests to Drop Station 3580 from Call
	FAC [CRV=87, FIE(Return Result, Inv_id=3)]	Station 3580 Dropped
	REL COMP [CRV=87, FIE(Invoke, Inv_id=10, Call Ended, call_id=16, cause = normal)]	Call Terminated

Incoming Call to Station Sent to Coverage with Send All Calls (SAC)

This scenario shows the call flow for a call from station 3568 to station 3569 (see Figure B-28). After alerting at station 3569, the user at station 3569 activates Send All Calls (SAC) and redirects the call to coverage (station 6056). Station 3569 maintains a simulated bridged appearance for the call, but does not connect to the call.

Stations 3568 and 3569 have Domain (Station) Control active.

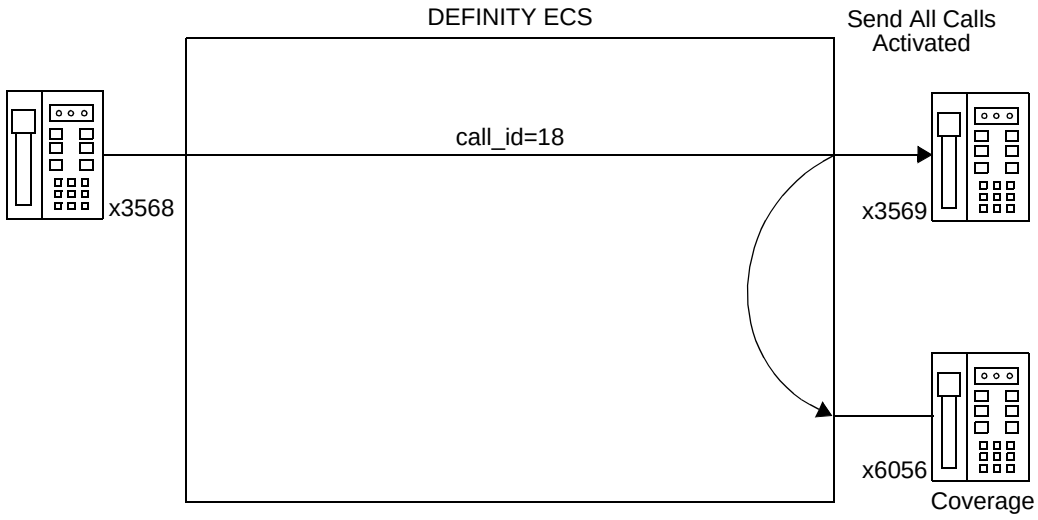


Figure B-28. Incoming Call to Station Sent to Coverage via Send All Calls (SAC)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=86, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 3568)]		AP Requests Domain Control of Station 3568
	FAC [CRV=86, FIE(Return Result, Inv_id=1)]	Domain Control Granted (No Calls Present)
REG [CRV=88, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 3569)]		AP Requests Domain Control of Station 3569
	FAC [CRV=88, FIE(Return Result, Inv_id=1)]	Control Granted (No Calls Present)
	FAC [CRV=86, FIE(Invoke, Inv_id=10, Event Report, call_id=18, party_id=1, event=call initiated)]	Call Initiated Event Report (Station 3568 Listens to Dial Tone) — x3568 Association
REG [CRV=87, FIE(Invoke, Inv_id=1, 3P Make Call, called number=3569, calling number=3568)]		AP Requests Make Call from Station 3568 to Station 3569
	FAC [CRV=87, FIE(Invoke, Inv_id=2, Event Report, call_id=18, party_id=2, event=alerting, calling number=3568, called number=3569, connected number=3569)]	Alerting Event Report (Call Delivered to Station 3569) — 3P Make Call Association
	FAC [CRV=86, FIE(Invoke, Inv_id=12, Event Report, call_id=18, party_id=2, event=alerting, calling number=3568, called number=3569, connected number=3569)]	Alerting Event Report (Call Delivered to Station 3569) — x3568 Association
	FAC [CRV=88, FIE(Invoke, Inv_id=6, Event Report, call_id=18, party_id=2, event=alerting, calling number=3568, called number=3569, connected number=3569)]	Alerting Event Report (Call Delivered to Station 3569) — x3569 Association
REG [CRV=81, FIE(Invoke, Inv_id=1, Request Feature, feature=Activate SAC, redirecting number=3569)]		AP Requests Send All Calls (SAC) for Station 3569

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	REL COMP [CRV=81, FIE(Return Result, Inv_id=1)]	SAC Activated
	FAC [CRV=87, FIE(Invoke, Inv_id=4, Event Report, call_id=18, party_id=3, event=alerting, calling number=3568, called number=3569, connected number=6056 cause=3/31)]	Alerting Event Report (Call Goes to Coverage, Station 6056) — 3P Make Call Association
	FAC [CRV=86, FIE(Invoke, Inv_id=14, Event Report, call_id=18, party_id=3, event=alerting, calling number=3568, called number=3569, connected number=6056 cause=3/31)]	Alerting Event Report (Call Goes to Coverage, Station 6056) — x3568 Association
	FAC [CRV=87, FIE(Invoke, Inv_id=20, Event Report, call_id=18, party_id=3, event=connected, calling number=3568, called number=3569, connected number=6056)]	Connected Event Report Station 6056 Answers — 3P Make Call Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=86, FIE(Invoke, Inv_id=16, Event Report, call_id=18, party_id=3, event=connected, calling number=3568, called number=3569, connected number=6056)]	Connected Event Report Station 6056 Answers — x3568 Association
REG [CRV=87, FIE(Invoke, Inv_id=3, 3P Clear Call)]		AP Requests Clear Call — 3P Make Call Assoc.
	REL COMP [CRV=87, FIE(Return Result, Inv_id=3)]	Call Cleared — 3P Make Call Assoc. Ends
	FAC [CRV=86, FIE(Invoke, Inv_id=18, Event Report, call_id=18, party_id=1, event=drop, connected number=3568, cause=normal clearing)]	Drop Event Report (Station 3568 Disconnects) — x3568 Association
	FAC [CRV=88, FIE(Invoke, Inv_id=8, Event Report, call_id=18, party_id=2 event=drop, connected number=3569, cause=normal clearing)]	Drop Event Report (Call Disconnects from Station 3569's Simulated Bridged Appearance) — x3569 Association

External Call to ACD Split Intraflows to Another ACD Split

This scenario shows a call flow for an incoming ISDN PRI call to ACD split 3456 that intraflows to ACD split 567 (see Figure B-29). Agent 6005, logged into ACD split 567, receives the call.

ACD split 3456 has Event Notification active as presented in the initialization scenario. The adjunct processor does not have Event Notification active for ACD split 567.

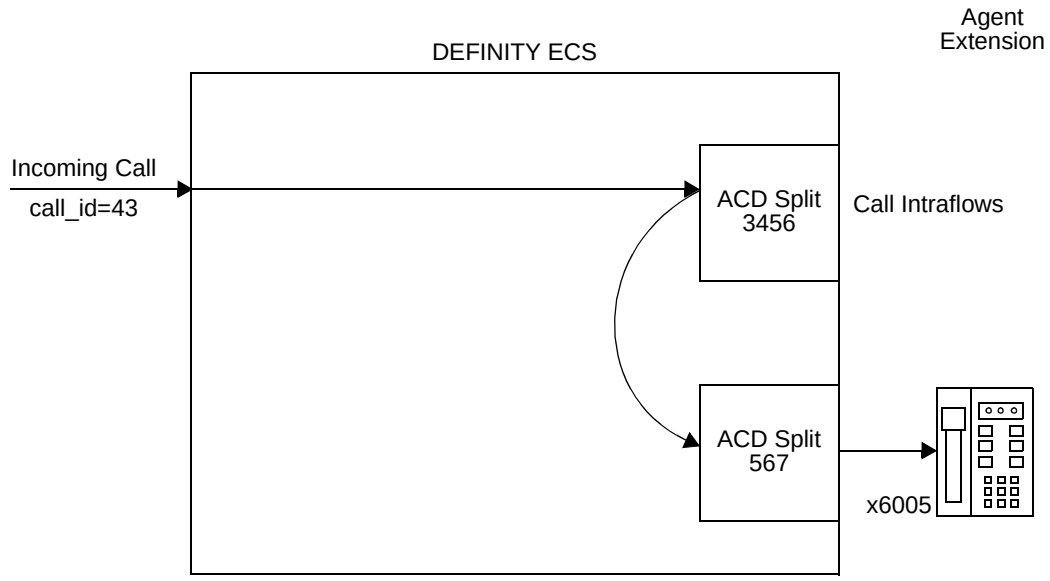


Figure B-29. Call Intraflow from ACD Split to ACD Split

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=102, FIE(Invoke, Inv_id=22, Event Report, call_id=43, event=call offered, calling number=4159703478, called number=8095763456, domain=ACD Split 3456)]	Call Offered to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=24, Event Report, call_id=43, event=queued, called number=8095763456, calls in queue=1, domain=ACD Split 3456)]	Call Queues to to ACD Split 3456
	FAC [CRV=102, FIE(Invoke, Inv_id=26, Event Report, call_id=43, event=queued, called number=8095763456, calls in queue=1, domain=ACD Split 567)]	Call Queues to to ACD Split 567
	FAC [CRV=102, FIE(Invoke, Inv_id=28, Event Report, call_id=43, party_id=2, event=alerting, calling number=4159703478, called number=8095763456, connected number=6005, domain= ACD Split 567)]	Alerting Event Report (Call Alerts Agent 6005)
	FAC [CRV=102, FIE(Invoke, Inv_id=28, Event Report, call_id=43, party_id=2, event=connected, calling number=4159703478, called number=8095763456, connected number=6005)]	Connected Event Report (Agent 6005 Answers)
FAC [CRV=102, FIE(Invoke, Inv_id=5, Stop Call Notification, call_id=34)]		AP Requests to Stop Receiving Event Reports for the Call
	FAC [CRV=102, FIE(Return Result, Inv_id=5)]	Event Reporting for Call Ended

Incoming Call is Redirected by Adjunct

Refer to Figure B-24 for this scenario in which an alerting call to Station 3569 is redirected via Redirect Alerting Call to Station 6055.

Assume that Third Party Domain Control is active for Station 3569, and that redirection is done on the call control.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=43, FIE(Invoke, Inv_id=4, Event Report, call_id=17, party_id=2, event=alerting, calling number=3568, called number=3569, connected number=3569)]	Call is alerting at ext. 3569
FAC [CRV=43, FIE(Invoke, Inv_id=5, Redirect call, party_id=2, redirection number=6055)]		
	FAC [CRV=43, FIE (Return Result, Inv_id=5)]	
	FAC [CRV=86 FIE(Invoke, Inv_id=4, event report, call_id=17, event=call redirected)]	Call redirected from ext. 3569 (x3569 domain control association)
	FAC [CRV=86 FIE(Invoke, Inv_id=4, event report, call_id=17, party_id=3, event=alerting, calling number=3568, called number=3569, connected number=6055, cause= normal clearing)]	Call alerting at redirect-to device (x6055)

9. AUDIX Interactions

This section presents call flows for calls transferred into and out of AUDIX ports.

Call Transferred to AUDIX via the AUDIX Transfer Feature Access Code

This scenario shows the call flow for a call originated via a **Third_Party_Make_Call** request from station 3459 to station 6001 (seeFigure B-30). Station 6001 has Send All Calls (SAC) active to coverage, Station 3333. Station 3333 manually transfers the call to AUDIX (hunt group 4000) via the AUDIX transfer feature access code.

The AUDIX port receiving the call (extension 4010) has Domain Control active.

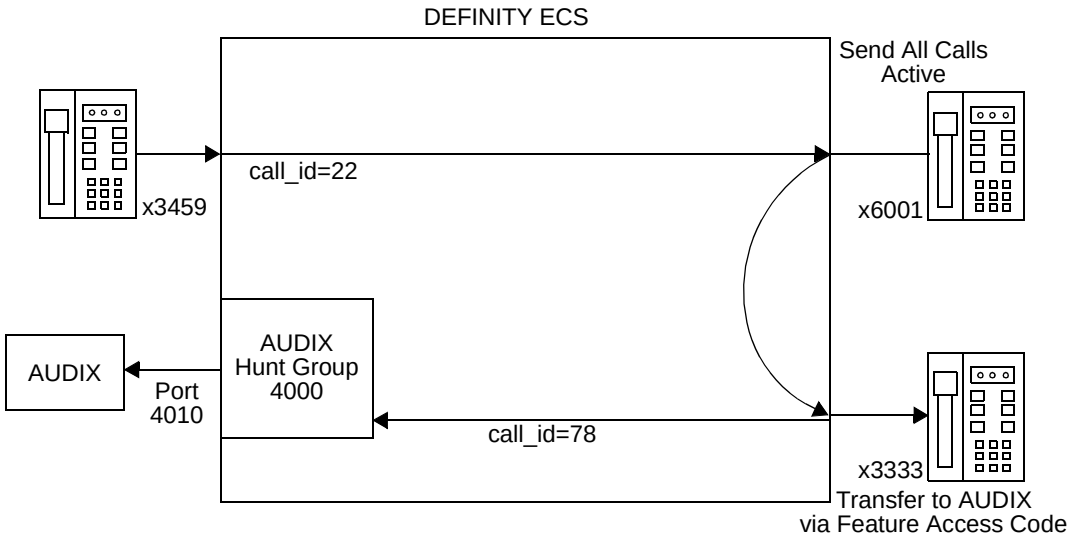


Figure B-30. Call Flow for Call Transfer into AUDIX

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=55, FIE(Invoke, Inv_id=1, 3P Domain Control, domain = extension 4010)]		AP Requests Domain Control for AUDIX Port 4010
	FAC [CRV=55, FIE(Return Result, Inv_id=1)]	Domain Control Granted (No Calls Present)
REG [CRV=43, FIE(Invoke, Inv_id=1, 3P Make Call, called number=6001, calling number=3459)]		AP Requests Make Call from Station 3459 to Station 6001
	FAC [CRV=43, FIE(Invoke, Inv_id=2, Event Report, call_id=22, party_id=2, event=alerting, calling number=3459, called number=6001, connected number=6001)]	Alerting Event Report (Station 6001 Alerts for Coverage Response Interval — Station 6001 has SAC Active) — 3P Make Call Association
	FAC [CRV=43, FIE(Invoke, Inv_id=2, Event Report, call_id=22, party_id=3 event=alerting, calling number=3459, called number=6001, connected number=3333)]	Alerting Event Report (Station 3333 Alerts — Station 6001 has SAC Active) — 3P Make Call Association
	FAC [CRV=43, FIE(Invoke, Inv_id=4, Event Report, call_id=22, party_id=3, event=connected, calling number=3459, called number=6001, connected number=3333)]	Connected Event Report (Station 3333 Answers) — 3P Make Call Association
	FAC [CRV=43, FIE(Invoke, Inv_id=6, Event Report, call_id=22, event=hold, party_id=3, connected number=3333)]	Hold Event Report (Transfer to AUDIX Initiated) — 3P Make Call Association
	FAC [CRV=55, FIE(Invoke, Inv_id=10, Event Report, call_id=78, party_id=2, event=alerting calling number=3459, called number=6001, connected number=4010, domain = Hunt Group 4000)]	Alerting Event Report (Call Delivered to AUDIX Port 4010) — Domain Control Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=43, FIE(Invoke, Inv_id=8, Event Report, other_call_id=22, resulting call_id=78, event=call transferred, party_id=1, party_id=3, connected number=3459, connected number=4010, calling number=3333, called number=4000)]	Call Transferred Event Report (Transfer to AUDIX Completed) — 3P Make Call Call Association
	FAC [CRV=55, FIE(Invoke, Inv_id=4, Event Report, other_call_id=22, resulting call_id=78, event=call transferred, party_id=1, party_id=3 connected number=3459, connected number=4010, calling number=3333, called number=4000)]	Call Transferred Event Report (Transfer to AUDIX Completed) — Domain Control Association
	FAC [CRV=43, FIE(Invoke, Inv_id=10, Event Report, call_id=78, party_id=2, event=connected, calling number=3459, called number=6001, connected number=4010)]	Connected Event Report (Call Connected to AUDIX Port 4010) — 3P Make Call Association
	FAC [CRV=55, FIE(Invoke, Inv_id=18, Event Report, call_id=78, party_id=2, event=connected, calling number=3459, called number=6001, connected number=4010)]	Connected Event Report (Call Connected to AUDIX Port 4010) — Domain Control Association
FAC [CRV=43, FIE(Invoke, Inv_id=5, 3P Clear Call, call_id=78)]		AP Requests to Terminate Call
	REL COMP[CRV=43, FIE(Return Result, Inv_id=5)]	3P Make Call Association Terminated — Call Cleared
	FAC [CRV=55, FIE(Invoke, Inv_id=20, Event Report, call_id=78, party_id=2, event=drop, connected number=4010, cause=normal clearing)]	Drop Event Report (Call Dropped from AUDIX Port 4010) — Domain Control Association

Call Transferred from AUDIX to Station via the AUDIX Enhanced Transfer Feature

This scenario shows the call flow for a call from station 3459 to station 6002 (see Figure B-31). The call goes to AUDIX coverage (ACD Split 4000) and is answered by AUDIX. AUDIX transfers the call to station 3333 via the enhanced AUDIX transfer feature.

Station 3333 has Domain Control active and ACD split 4000 has Event Notification active.

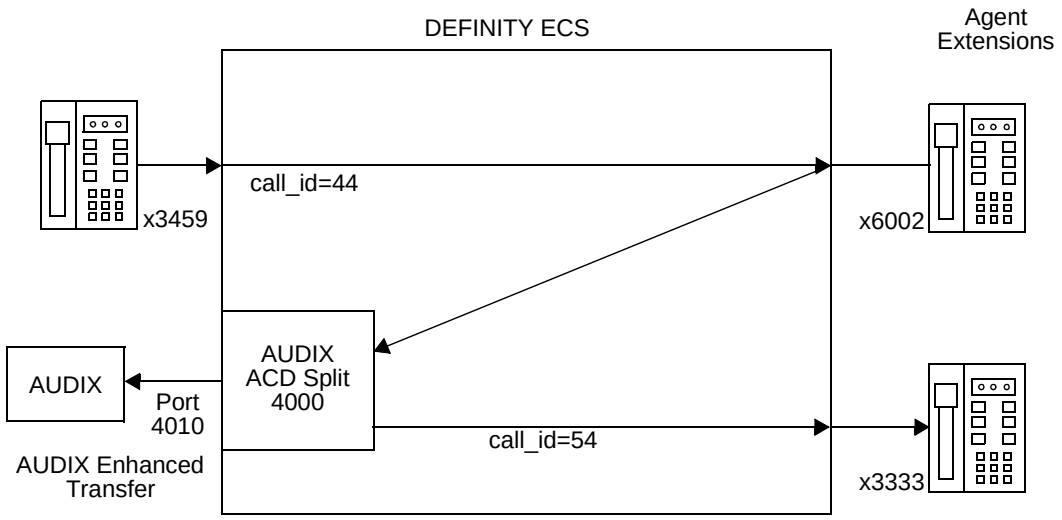


Figure B-31. Call Flow for Call Transferred out of AUDIX

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=54, FIE(Invoke, Inv_id=1, 3P Domain Control, domain = extension 3333)]		AP Requests Domain Control for Station 3333
	FAC [CRV=54, FIE(Return Result, Inv_id=1)]	Domain Control Granted (No Calls Present)
REG [CRV=120, FIE(Invoke, Inv_id=1, Event Notification Request, domain = ACD Split 4000)]		AP Requests Event Notification for ACD Split 4000
	FAC [CRV=120, FIE(Return Result, Inv_id=1)]	Event Notification Request Accepted
	FAC [CRV=120, FIE(Invoke, Inv_id=2, Event Report, call_id=44, event=call offered, calling number=3459, called number=6002, domain=ACD Split 4000)]	Call Offered to ACD Split 4000 — Event Notification Association
	FAC [CRV=120, FIE(Invoke, Inv_id=4, Event Report, call_id=44, party_id=2, event=alerting, calling number=3459, called number=6002, connected number=4010, domain = ACD Split 4000)]	Alerting Event Report (Call Delivered to AUDIX Port 4010) — Event Notification Association
	FAC [CRV=120, FIE(Invoke, Inv_id=6, Event Report, call_id=44, party_id=2, event=connected, calling number=3459, called number=6002, connected number=4010)]	Connected Event Report (Call Connected to AUDIX Port 4010) — Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=54, FIE(Invoke, Inv_id=2, Event Report, call_id=78, party_id=2, event=alerting, calling number=3459, called number=3333, connected number=3333)]	Alerting Event Report (Call Transferred to Station 3333 — Station 3333 Alerts) — Domain Control Association Note: called number=3333 for direct transfer (*T) or called number=6002 for redirect (*O)
	FAC [CRV=120, FIE(Invoke, Inv_id=8, Event Report, other_call_id=44, resulting call_id=78, event=call transferred, party_id=1, party_id=2, old_party_id=other call1, old_party_id-reslt call 2, connected number=3459, connected number=3333, calling number=4010, called number=3333)]	Call Transferred Event Report (Transfer to Station 3333 Completed) — Event Notification Association
	FAC [CRV=54, FIE(Invoke, Inv_id=4, Event Report, other_call_id=44, resulting call_id=78, event=call transferred, party_id=1, party_id=2, old_party_id=other call1, old_party_id-reslt call 2, connected number=3459, connected number=3333, calling number=4010, called number=3333)]	Call Transferred Event Report (Transfer to Station 3333 Completed) — Domain Control Association
	FAC [CRV=120, FIE(Invoke, Inv_id=10, Event Report, call_id=78, party_id=2, event=connected, calling number=3459, called number=3333, connected number=3333)]	Connected Event Report (Station 3333 Answers) — Event Notification Association Note: called number=3333 for direct transfer (*T) or called number=6002 for redirect (*O)
	FAC [CRV=54, FIE(Invoke, Inv_id=6, Event Report, call_id=78, party_id=2, event=connected, calling number=3459, called number=3333, connected number=3333)]	Connected Event Report (Station 3333 Answers) — Domain Control Association Note: called number=3333 for direct transfer (*T) or called number=6002 for redirect (*O)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=120, FIE(Invoke, Inv_id=5, Stop Event Notification call_id=78)]		AP Requests to Stop Receiving Event Reports for the Call
	FAC [CRV=120, FIE(Return Result, Inv_id=5)]	Event Notification for Call Ended
	FAC [CRV=54, FIE(Invoke, Inv_id=8, Event Report, call_id=78, party_id=2, event=drop, connected number=3333, cause=normal clearing)]	Drop Event Report (Station 3333 Disconnects) Domain Control Association

10. Calls Across Multiple Switches

This section presents several scenarios for calls routed, transferred, or conferenced across switches. Figure B-32 shows the VDNs, vectors, splits, and extensions for the following scenarios. Each switch has its own ASAI link, but ASAI links are not shown in the figure.

NOTE:
Section 15 gives examples of calls between switches using UUI.

The trunk groups that are connected between Switch A and Switch B in this scenario should be administered as non-DCS. See the DCS discussion in Chapter 12, “ASAI and Feature Interactions” in the *DEFINITY Enterprise Communications Server CallVisor ASAI Technical Reference.*

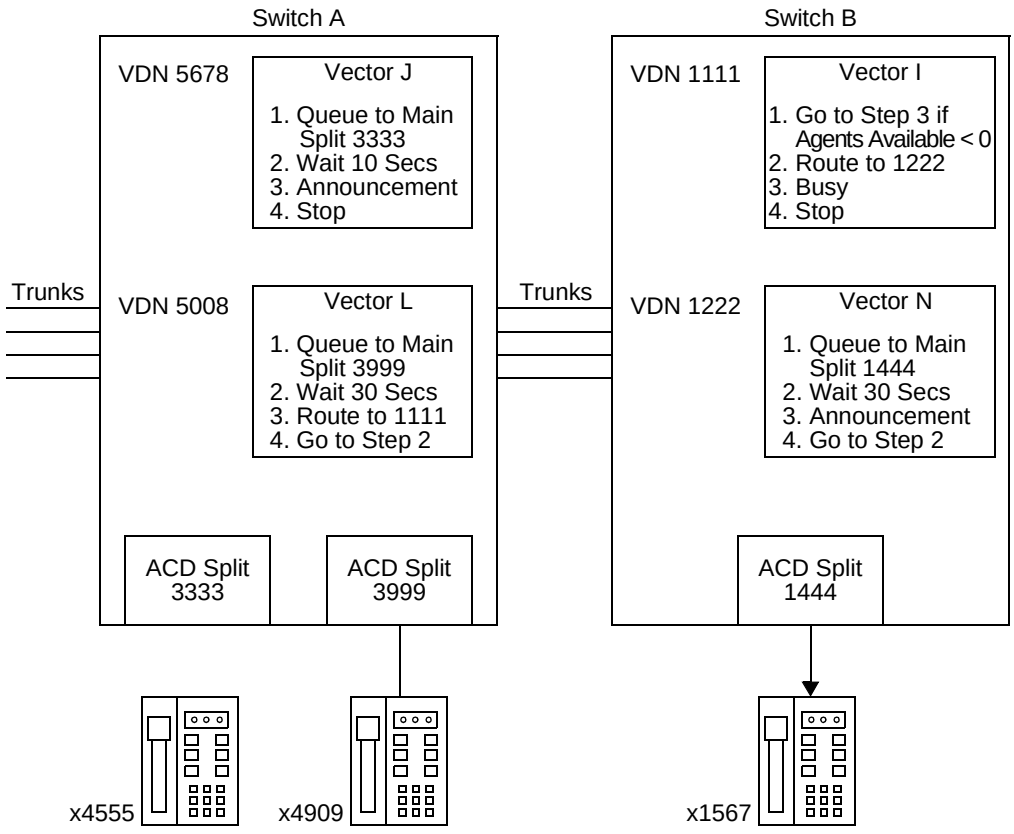


Figure B-32. Multiple Switch Configuration

External Call to VDN, Answered by Station and Transferred to a VDN on Another Switch

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 answered by extension 4555 in ACD split 3333 (see Figure B-33). The agent at extension 4555 manually transfers the call to VDN 1222 in Switch B. Extension 1567 in ACD split 1444 answers the call at Switch B.

The scenario shows the agent at extension 4555 completing the transfer operation while the call is in queue at ACD split 1444. Note that no Alerting or Connected Event Report is sent to Switch A, because the call to Switch B (call_id 45) is not monitored on Switch B until it is merged with the incoming call (call_id 37). If agent 4555 completed the transfer after talking to agent 1567, the Call Transferred Event Report would have occurred after the Connected Event Report was sent by Switch B. All other parameters would have remained the same. Similarly, if the operation is a conference instead of a transfer, the Call Transferred Event Report would have been replaced by a Call Conferenced Event Report.

Assume that VDN 5678 is monitored over CRV 98 by an ASAI Adjunct Processor connected to Switch A, and that VDN 1222 is monitored over CRV 26 by an ASAI Adjunct Processor connected to Switch B. **Messages in italics refer to messages exchanged by Switch B and the ASAI Adjunct Processor connected to Switch B.**

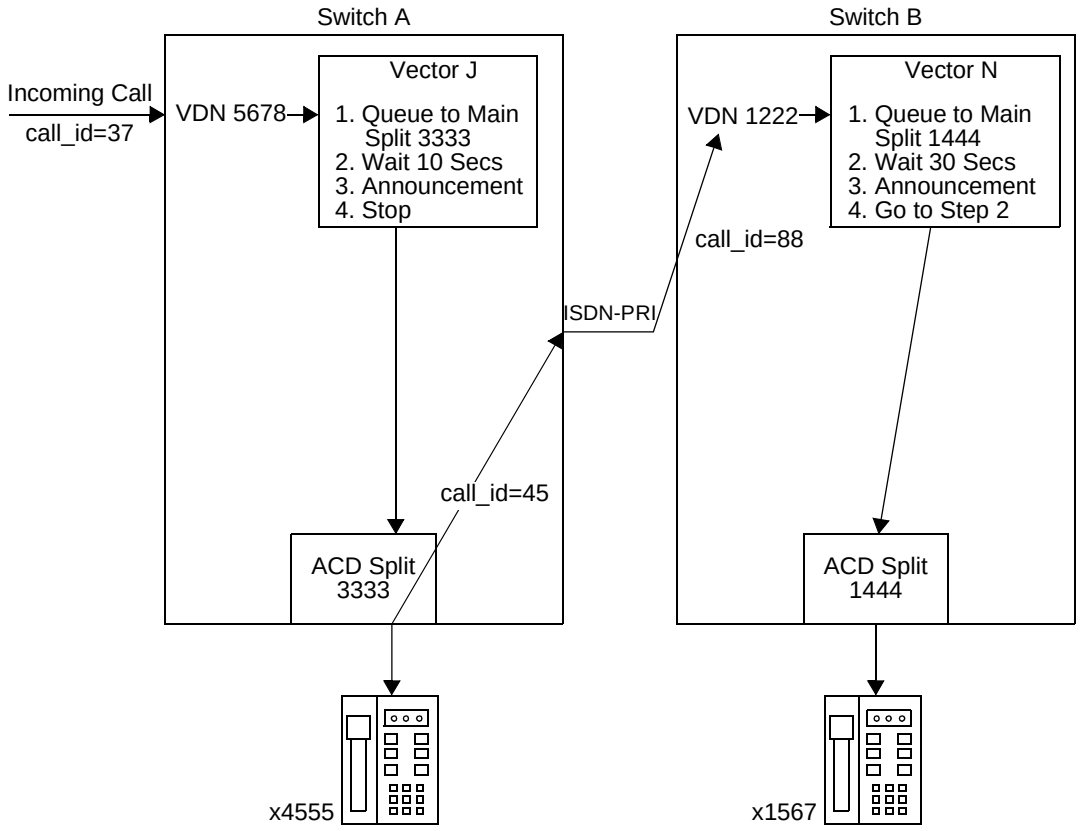


Figure B-33. Call Flow for Blind Transfer to Another Switch

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=37, event=call offered, calling number=4155766379, called number=2015765678, domain=VDN 5678)]	Call Offered to VDN 5678
	FAC [CRV=98, FIE(Invoke, Inv_id=4, Event Report, call_id=37, event=queued, called number=2015765678, calls in queue=2, domain= ACD Split 3333)]	Queued Event Report (Queued in ACD Split 3333)
	FAC [CRV=98, FIE(Invoke, Inv_id=6, Event Report, call_id=37, party_id=2, event=alerting, calling number=4155766379, called number=2015765678, connected number=4555, domain= ACD Split 3333)]	Alerting Event Report (Call Delivered to Agent 4555)
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=37, party_id=2, event=connected, calling number=4155766379, called number=2015765678, connected number=4555)]	Connected Event Report (Call Connected to Agent 4555)
	FAC [CRV=98, FIE(Invoke, Inv_id=10, Event Report, call_id=37, party_id=2, event=hold, connected number=4555)]	Hold Event Report (Agent 4555 Places Call on Transfer Hold)
	FAC [CRV=26, FIE(Invoke, Inv_id=2, Event Report, call_id=88, event=call offered, calling number=2015764555, called number=2015661222, domain= VDN 1222)]	*Call Offered to VDN 1222*
	FAC [CRV=26, FIE(Invoke, Inv_id=4, Event Report, call_id=88, event=queued, called number=2015661222, calls in queue=10, domain= ACD Split 1444)]	*Queued Event Report (Queued in ACD Split 1444)*

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=12, Event Report, other call_id=37, resulting call_id=45, event=call transferred, party_id=2, party_id=3, old_party_id=reslt call 2, old_party_id=other call 1 connected number=######, connected number=######, calling number=4555, called number=######)]	Transfer Event Report (Agent 4555 completes Transfer)
	FAC [CRV=26, FIE(Invoke, Inv_id=6, Event Report, call_id=88, party_id=2, event=alerting, calling number=2015764555, called number=2015661222, connected number=1567, domain= ACD Split 1444)]	*Alerting Event Report (Call Delivered to Station 1567)*
	FAC [CRV=26, FIE(Invoke, Inv_id=8, Event Report, call_id=88, party_id=2, event=connected, calling number=2015764555, called number=2015661222, connected number=1567)]	*Connected Event Report (Call Connected to Station 1567)*
	FAC [CRV=26, FIE(Invoke, Inv_id=10, Event Report, call_id=88, party_id=2, event=drop, cause=normal, connected number=1567)]	*Drop Event Report (Agent disconnects)*
	FAC [CRV=26, FIE(Invoke, Inv_id=12, Call Ended, call_id=88, cause=normal)]	*Call Terminates*
	FAC [CRV=98, FIE(Invoke, Inv_id=14, Event Report, call_id=45, party_id=3, event=drop, cause=normal, connected number=######)]	Drop Event Report (Trunk Disconnects) The Caller Disconnects Before the Trunk to Switch B Drops
	FAC [CRV=98, FIE(Invoke, Inv_id=16, Call Ended, call_id=45, cause=normal)]	Call Terminates

External Call to VDN, Answered by Station, and Transferred to a Station on Another Switch

This scenario shows the call flow for an incoming non-ISDN call to VDN 5678 answered by extension 4555 in ACD split 3333 (see Figure B-34). The agent at extension 4555 does a consultation transfer to extension 1567 on Switch B. That is, the transfer is completed after the agent on extension 4555 talks to the agent on extension 1567. The trunks between switches are ISDN PRI trunks.

Assume that VDN 5678 is monitored over CRV 98 by an ASAI Adjunct Processor connected to Switch A and that extension 1567 is domain-controlled over CRV 45 by an ASAI Adjunct Processor connected to Switch B. **Messages in italics refer to messages exchanged by Switch B and the ASAI Adjunct Processor connected to Switch B.**

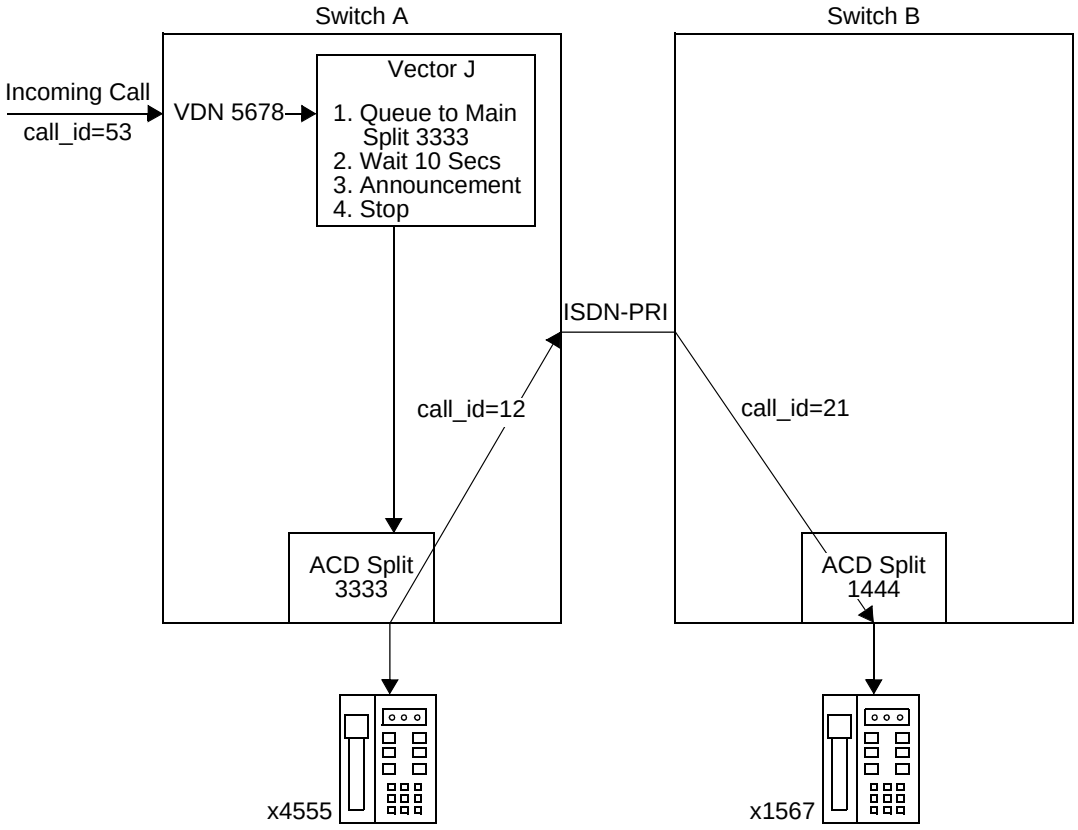


Figure B-34. Call Flow for Consultation Transfer to Another Switch

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=53, trunk group=102, event=call offered, called number=5678, domain=VDN 5678)]	Call Offered to VDN 5678
	FAC [CRV=98, FIE(Invoke, Inv_id=4, Event Report, call_id=53, event=queued, called number=5678, calls in queue=1, domain= ACD Split 3333)]	Queued Event Report (Queued in ACD Split 3333)
	FAC [CRV=98, FIE(Invoke, Inv_id=6, Event Report, call_id=53, party_id=2, event=alerting, trunk group=102, called number=5678, connected number=4555, domain= ACD Split 3333)]	Alerting Event Report (Call Delivered to Agent 4555)
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=53, party_id=2, event=connected, trunk group=102, called number=5678, connected number=4555)]	Connected Event Report (Call Connected to Agent 4555)
	FAC [CRV=98, FIE(Invoke, Inv_id=10, Event Report, call_id=53, party_id=2, event=hold, connected number=4555)]	Hold Event Report (Agent 4555 Places Call on Transfer Hold)
	FAC [CRV=45, FIE(Invoke, Inv_id=2, Event Report, call_id=21, party_id=2, event=alerting, calling number=2015764555, called number=2015661567, connected number=1567)]	*Alerting Event Report (Call Alerts Station 1567)*

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=45, FIE(Invoke, Inv_id=4, Event Report, call_id=21, party_id=2, event=connected, calling number=2015764555, called number=2015661567, connected number=1567)]	*Connected Event Report (Station 1567 Answers Call)*
		Extension 4555 Connected to Extension 1567
	FAC [CRV=98, FIE(Invoke, Inv_id=12, Event Report, other call_id=53, resulting call_id=12, event=call transferred, party_id=1, party_id=2, old_party_id=other call 1, old_party_id=reslt call 2, connected number=######, connected number=######, calling number=4555, called number=######)]	Transfer Event Report (Agent 4555 completes Transfer)
		Caller Connected to Extension 1567
	FAC [CRV=98, FIE(Invoke, Inv_id=14, Event Report, call_id=12, party_id=1, event=drop, cause=normal, connected number=######)]	Drop Event Report (Caller Disconnects)
	FAC [CRV=98, FIE(Invoke, Inv_id=16, Call Ended, call_id=12, cause=normal)]	Call Terminates
	FAC [CRV=45, FIE(Invoke, Inv_id=10, Event Report, call_id=21, party_id=1, event=drop, cause=normal, connected number=######)]	*Drop Event Report (Trunk Disconnects)*
	FAC [CRV=45, FIE(Invoke, Inv_id=12, Event Report, call_id=21, party_id=2, event=drop, cause=normal, connected number=1567)]	*Drop Event Report (Agent Drops)*

External Call to Lookahead Interflow VDN

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5008 that looks-ahead to VDN 1111 on Switch B (see Figure B-35). The first lookahead interflow attempt is denied by Switch B. The second lookahead interflow attempt is accepted by Switch B and the call is delivered to and answered by extension 1567 in ACD split 1444.

Note that Switch A is not guaranteed to receive an ISDN Alerting message from Switch B. In these cases, the ASAI adjunct connected to Switch A does not receive the Alerting Event Report. The lookahead display information is the VDN name provided in VDN administration form and is subject to the VDN display override rules.

Assume that VDN 5008 is monitored over CRV 80 by an ASAI Adjunct Processor connected to Switch A, and that VDNs 1111 and 1222 are monitored over CRVs 20 and 26, respectively, by an ASAI Adjunct Processor connected to Switch B. **Messages in italics refer to messages exchanged by the Switch B and the ASAI Adjunct Processor connected to Switch B.**

Note that the multiple monitors feature results in events on both the monitors for VDNs 1111 and 1222 on Switch B as the call flows from 1111 to 1222 and is answered.

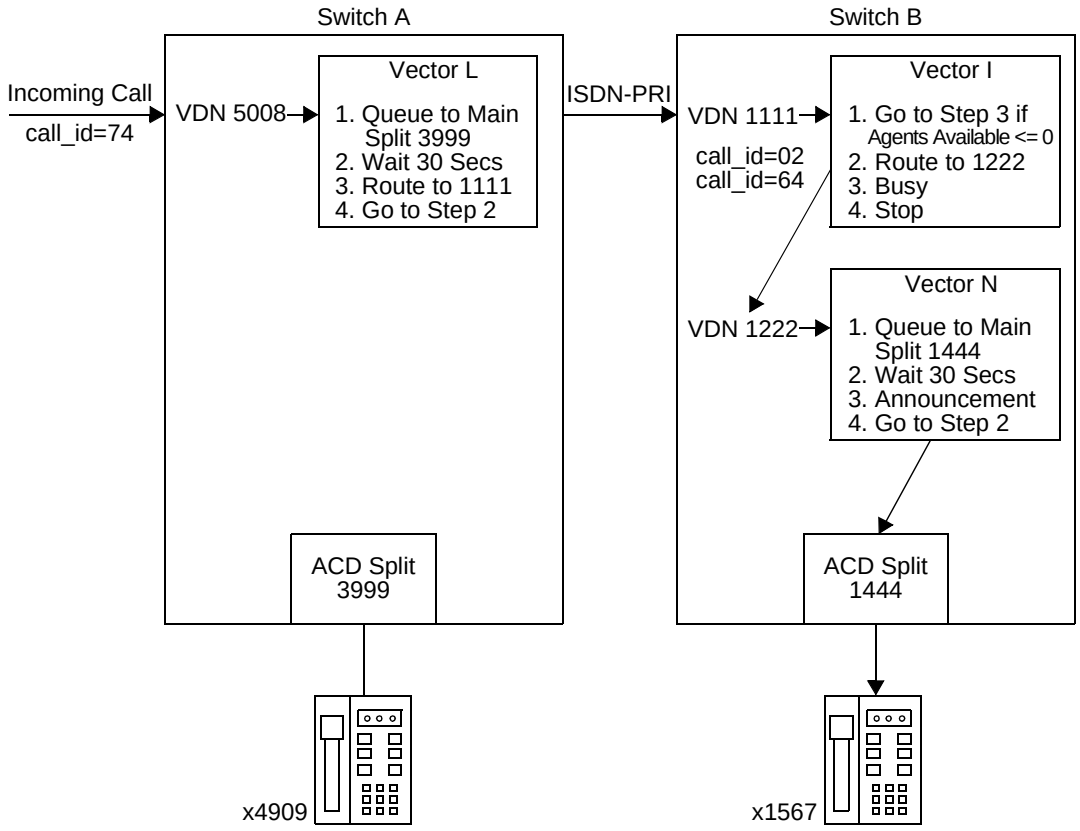


Figure B-35. Call Flow for Incoming Call to LookAhead Interflow Vector

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=80, FIE(Invoke, Inv_id=2, Event Report, call_id=74, event=call offered, calling number=4157566379, called number=2015765008, domain=VDN 5008)]	Call Offered to VDN 5008
	FAC [CRV=80, FIE(Invoke, Inv_id=4, Event Report, call_id=74, event=queued, called number=2015765008, calls in queue=20, domain= ACD Split 3999)]	Queued Event Report (Queued in ACD Split 3999)
	FAC [CRV=20, FIE(Invoke, Inv_id=22, Event Report, call_id=02, event=call offered, calling number=4157566379, called number=2015661111, domain=VDN 1111, LAI Display=VDN 008 Sales)]	*Call Offered to VDN 1111 — VDN 1111 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=24, Event Report, call_id=02, party_id=1, event=drop, cause=busy, connected number=######)]	*Drop Event Report (LAI Call Rejected by Busy Step)*
	FAC [CRV=20, FIE(Invoke, Inv_id=26, Call Ended, call_id=02, cause=busy)]	*Call Terminates VDN 1111 Event Notification Association*
	FAC [CRV=80, FIE(Invoke, Inv_id=10, Event Report, call_id=74, party_id=2, event=drop, cause=busy, connected number=######)]	Drop Event Report (LAI Fails)
	FAC [CRV=20, FIE(Invoke, Inv_id=36, Event Report, call_id=64, event=call offered, calling number=4157566379, called number=2015661111, domain=VDN 1111, LAI Display=VDN 008 Sales)]	*Call Offered to VDN 1111 — VDN 1111 Event Notification Association*

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=26, FIE(Invoke, Inv_id=66, Event Report, call_id=64, event=call offered, calling number=4157566379, called number=2015661111, domain=VDN 1222, LAI Display=VDN 008 Sales)]	*Call Offered to VDN 1222 — VDN 1222 Event Notification Association*
	FAC [CRV=26, FIE(Invoke, Inv_id=70, Event Report, call_id=64, party_id=2, event=alerting, calling number=4157566379, called number=2015661111, connected number=1567, domain= ACD Split 1444)]	*Alerting Event Report (Call Delivered to Station 1567) — VDN 1222 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=70, Event Report, call_id=64, party_id=2, event=alerting, calling number=4157566379, called number=2015661111, connected number=1567, domain= ACD Split 1444)]	*Alerting Event Report (Call Delivered to Station 1567) — VDN 1111 Event Notification Association*
	FAC [CRV=80, FIE(Invoke, Inv_id=6, Event Report, call_id=74, party_id=2, event=alerting, calling number=4157566379, called number=2015661111, connected number=######)]	Alerting Event Report (ISDN Alerting Message)
	FAC [CRV=26, FIE(Invoke, Inv_id=8, Event Report, call_id=64, party_id=2, event=connected, calling number=4157566379, called number=2015661111, connected number=1567)]	*Connected Event Report (Call Connected to Station 1567) — VDN 1222 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=8, Event Report, call_id=64, party_id=2, event=connected, calling number=4157566379, called number=2015661111, connected number=1567)]	*Connected Event Report (Call Connected to Station 1567) — VDN 1111 Event Notification Association*

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=80, FIE(Invoke, Inv_id=8, Event Report, call_id=74, party_id=2, event=connected, calling number=4155766379, called number=2015661111, connected number=######)]	Connected Event Report (ISDN PRI Connect)
	FAC [CRV=26, FIE(Invoke, Inv_id=10, Event Report, call_id=64, party_id=2, event=drop, cause=normal, connected number=1567)]	*Drop Event Report (Agent disconnects) — VDN 1222 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=10, Event Report, call_id=64, party_id=2, event=drop, cause=normal, connected number=1567)]	*Drop Event Report (Agent disconnects) — VDN 1222 Event Notification Association*
	FAC [CRV=26, FIE(Invoke, Inv_id=12, Call Ended, call_id=64, cause=normal)]	*Call Terminates — VDN 1222 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=12, Call Ended, call_id=64, cause=normal)]	*Call Terminates — VDN 1222 Event Notification Association*
	FAC [CRV=80, FIE(Invoke, Inv_id=14, Event Report, call_id=74, party_id=2, event=drop, cause=normal, connected number=######)]	Drop Event Report (ISDN Trunk Disconnects)
	FAC [CRV=80, FIE(Invoke, Inv_id=16, Call Ended, call_id=74, cause=normal)]	Call Terminates

External Call to VDN, Answered by a Local Station, and Transferred to a Lookahead Interflow VDN

This scenario shows the call flow for an incoming ISDN PRI call to VDN 5678 answered by extension 4555 in ACD split 3333 and subsequently transferred to lookahead interflow VDN 5008 (see Figure B-36). The call is accepted by the receiving switch (Switch B), waits in queue until the vector announcement (VDN 1222, vector N), and is abandoned by the caller while the call is alerting extension 1567. The lookahead display information is the VDN name provided in VDN administration form.

The transfer to VDN 5008 is completed before the call attempts the lookahead to Switch B. If the call had initiated the lookahead interflow before the transfer operation had been completed, the ASAI adjunct connected to Switch B would have received the extension 4555 as the calling party number instead of the original SID/ANI for the call.

Note that the Alerting Event Report received by the ASAI Adjunct Processor connected to Switch A is received only if the lookahead interflowed call receives alerting treatment (that is, wait hearing ringback/announcement or extension alerting) before the call is answered on Switch B. If the call waits with silence (for example, wait hearing silence) before Switch B applies ringback or answers the call, the ASAI Adjunct Processor connected to Switch A receives a Cut-Through/Progress Event Report. Subsequent Alerting and Connected Event Reports are provided depending on the call treatment provided by vector processing.

The Connected Event Report received by the ASAI Adjunct Processor connected to Switch A is triggered by the ISDN Connect message received from Switch B. Switch B provides a single ISDN Connect message with the first answer treatment provided. For example, listening to music or to an announcement while the call is in vector processing will trigger Switch B to send an ISDN Connect message to Switch A. Switch B will not send additional ISDN messages to Switch A for subsequent answers (for example, other announcements or answered by a station). Therefore, the ASAI Adjunct Processor connected to Switch A does not receive further answer/connect notifications for the call.

The ISDN called number received by the adjunct connected to Switch A is not the same as the VDN number, since ISDN digit manipulation has occurred.

Assume that VDNs 5678 and 5008 are monitored over CRVs 98 and 80, respectively, by an ASAI Adjunct Processor connected to Switch A, and that VDNs 1111 and 1222 are monitored over CRVs 20 and 26, respectively, by an ASAI Adjunct Processor connected to Switch B. **Messages in italics refer to messages exchanged by the Switch B and the ASAI Adjunct Processor connected to Switch B.**

Note that the multiple monitors feature results in copies of various events on both VDN monitors on Switches A and B.

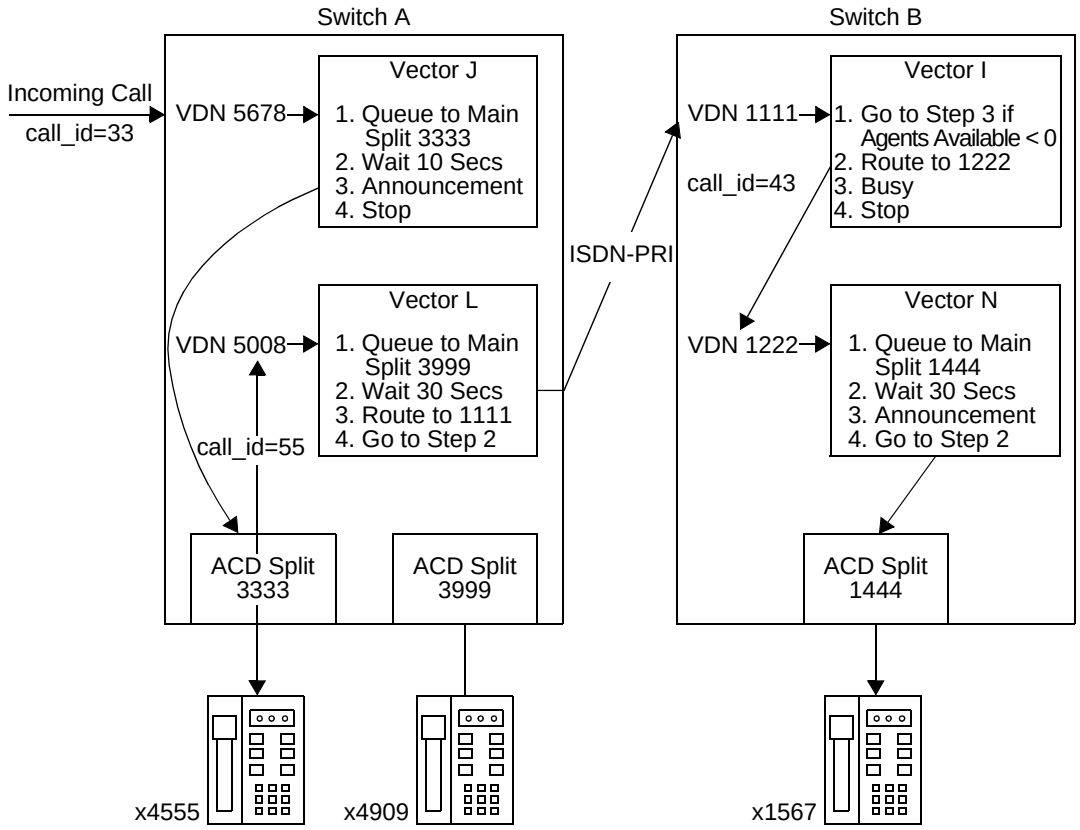


Figure B-36. Call Flow for a Transfer to a Lookahead Interflow Vector

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=20, Event Report, call_id=33, event=call offered, calling number=8135308484, called number=2015763448, domain=VDN 5678)]	Call Offered to VDN 5678 — VDN 5678 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=22, Event Report, call_id=33, event=queued, called number=2015763448, calls in queue=1, domain= ACD Split 3333)]	Queued Event Report (Queued in ACD Split 3333) — VDN 5678 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=24, Event Report, call_id=33, party_id=2, event=alerting, calling number=8135308484, called number=2015763448, connected number=4555, domain= ACD Split 333)]	Alerting Event Report (Call Delivered to Station 4555) — VDN 5678 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=26, Event Report, call_id=33, party_id=2, event=connected, calling number=8135308484, called number=2015763448, connected number=4555)]	Connected Event Report (Call Connected to Station 4555) — VDN 5678 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=28, Event Report call_id=33, party_id=2 event=hold, connected number=1567)]	Hold Event Report (Agent 455 Places Call on Transfer Hold) — VDN 5678 Event Notification Association
	FAC [CRV=80, FIE(Invoke, Inv_id=2, Event Report, call_id=55, event=call offered, calling number=4555, called number=5008, domain=VDN 5008)]	Call Offered to VDN 5008 — VDN 5008 Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=80, FIE(Invoke, Inv_id=4, Event Report, call_id=55, event=queued, called number=5008, calls in queue=40, domain= ACD Split 3999)]	Queued Event Report (Queued in ACD Split 3999) — VDN 5008 Event Notification Association
	FAC [CRV=80, FIE(Invoke Inv_id=12, Event Report, other call_id=33, resulting call_id=55, event=call transferred, party_id=1, old_party_id=other call 1, connected number=######, calling number=4555, called number=5008)]	Transfer Event Report (Agent 4555 completes Transfer) — VDN 5008 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=12, Event Report, other call_id=33, resulting call_id=55, event=call transferred, party_id=1, old_party_id=other call 1 connected number=######, calling number=4555, called number=5008)]	Transfer Event Report (Agent 4555 completes Transfer) — VDN 3698 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=64, Call Ended, call_id=33, cause=call cleared)]	Event Reports for Call Terminated (Call Merged) — VDN 5678 Event Notification Association
	FAC [CRV=20, FIE(Invoke, Inv_id=22, Event Report, call_id=43, event=call offered, calling number=8135308484, called number=5661111, domain=VDN 1111, LAI Display=Sales xfer 811)]	*Call Offered to VDN 1111 — VDN 1111 Event Notification Association*

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=26, FIE(Invoke, Inv_id=66, Event Report, call_id=43, event=call offered, calling number=8135308484, called number=5661111, domain=VDN 1222, LAI Display=Sales xfer 811)]	*Call Offered to VDN 1222 — VDN 1222 Event Notification Association*
	FAC [CRV=26, FIE(Invoke, Inv_id=64, Event Report, call_id=43, event=queued, called number=5661111, calls in queue=2, domain= ACD Split 1444)]	*Queued Event Report (Queued in ACD Split 1444) — VDN 1222 Event Notification Association*
	FAC [CRV=20, FIE(Invoke, Inv_id=64, Event Report, call_id=43, event=queued, called number=5661111, calls in queue=2, domain= ACD Split 1444)]	*Queued Event Report (Queued in ACD Split 1444) — VDN 1111 Event Notification Association*
	FAC [CRV=80, FIE(Invoke, Inv_id=6, Event Report, call_id=55, party_id=2, event=alerting, calling number=8135308484, called number=5661111, connected number=######)]	Alerting Event Report (ISDN PRI Alerting Message) — VDN 5008 Event Notification Association
	FAC [CRV=98, FIE(Invoke, Inv_id=6, Event Report, call_id=55, party_id=2, event=alerting, calling number=8135308484, called number=5661111, connected number=######)]	Alerting Event Report (ISDN PRI Alerting Message) — VDN 5678 Event Notification Association
	FAC [CRV=80, FIE(Invoke, Inv_id=8, Event Report, call_id=55, party_id=2, event=connected, calling number=8135308484, called number=5661111, connected number=######)]	Connected Event Report (Announcement Step Executed, ISDN PRI Connect) — VDN 5008 Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=8, Event Report, call_id=55, party_id=2, event=connected, calling number=8135308484, called number=5661111, connected number=######)]	Connected Event Report (Announcement Step Executed, ISDN PRI Connect) — VDN 5678 Event Notification Association
	FAC [CRV=26, FIE(Invoke, Inv_id=70, Event Report, call_id=43, party_id=2, event=alerting, calling number=8135308484, called number=5661111, connected number=1567, domain= ACD Split 1444)]	*Alerting Event Report (Call Delivered to Station 1567) — VDN 1222 Event Notification Association*
	FAC [CRV=80, FIE(Invoke, Inv_id=14, Event Report, call_id=55, party_id=1 event=drop, cause=normal, connected number=######)]	Drop Event Report (Caller Abandons) — VDN 5008 Event Notification Association
	FAC [CRV=80, FIE(Invoke, Inv_id=16, Call Ended, call_id=55, cause=normal)]	Call Terminates — VDN 5008 Event Notification Association
	FAC [CRV=26, FIE(Invoke, Inv_id=12, Call Ended, call_id=43, cause=normal)]	*Call Drops — VDN 1222 Event Notification Association*

11. Expert Agent Selection Interactions

This section presents call scenarios in the Expert Agent Selection Environment.

External Call to VDN, Answered by Logical Agent, and Conferenced with Another Logical Agent

This scenario shows an incoming non-ISDN call to VDN 5555 that queues to skills 3333 and 4444 (see Figure B-37). Logical agent 2345, logged in from station 6666, answers the call and conferences logical agent 8766 (logged in from station 9999). No queue event is provided for skill 4444 because logical agent 2345 with skill 4444 is available immediately to answer the call.

Note that the **called number** in the Call Conference Event Report provides the agent's physical extension, not the agent's login id extension.

Event Notification for VDN 5555 is active over CRV 78 as shown at the beginning of the call flow. The Adjunct Processor-initiated login for logical agent 2345 is also shown. Logical Agent 8766 is assumed to have logged in manually at the voice station. Also assume that the incoming trunk is an R2 MFC trunk which provides the CPN.

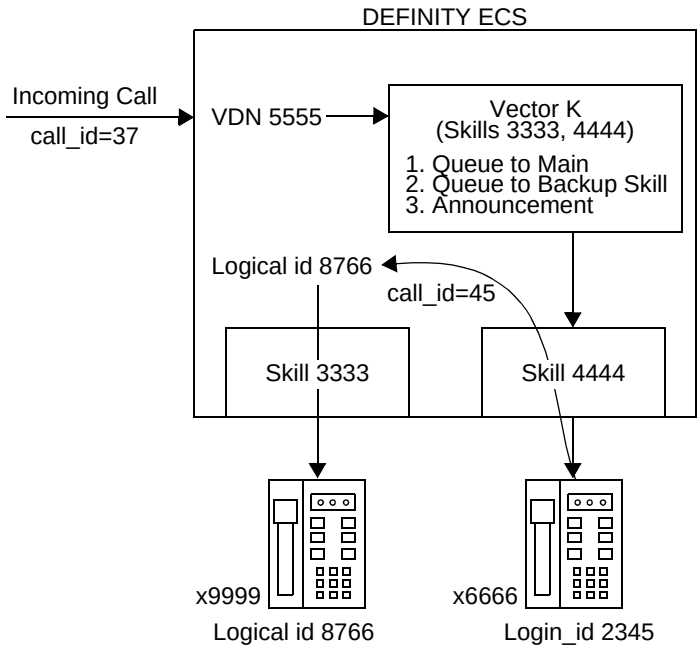


Figure B-37. Call Flow for Incoming Call to Skill VDN

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=78, FIE(Invoke, Inv_id=1, Event Notification Request, domain=VDN 5555)]		Notification Request for VDN 5555
	FAC [CRV=98, FIE(Return Result, Inv_id=1)]	Notification Accepted
REG [CRV=44, FIE(Invoke, Inv_id=1, Feature Request, feature=login, domain=ACD Split #####, domain=Extension 6666, domain=work mode-autoin, user code=2345#9000)]		Logical Agent 2345 Request Login from Physical Extension 6666 (##### = info. ignored)
	REL COMP [CRV=44, FIE(Return Result, Inv_id=1)]	Login Accepted
	FAC [CRV=78, FIE(Invoke, Inv_id=2, Event Report, call_id=37, event=call offered, calling number=8135308484, called number=5672035555, domain=VDN 5555)]	Call Offered to VDN 5555
	FAC [CRV=78, FIE(Invoke, Inv_id=4, Event Report, call_id=37, event=queued, called number=5672035555, calls in queue=2, domain= ACD Split 3333)]	Queued Event Report (Queued in Skill 3333)
	FAC [CRV=78, FIE(Invoke, Inv_id=6, Event Report, call_id=37, party_id=2, event=alerting, calling number=8135308484, called number=5672035555, connected number=6666, domain= ACD Split 4444)]	Alerting Event Report (Call Delivered to Logical Agent 2345)
	FAC [CRV=78, FIE(Invoke, Inv_id=8, Event Report, call_id=37, party_id=2, event=connected, calling number=8135308484, called number=5672035555, connected number=6666)]	Connected Event Report (Call Connected to Logical Agent 2345)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=78, FIE(Invoke, Inv_id=10, Event Report, call_id=37, party_id=2, event=hold, connected number=6666)]	Hold Event Report (Agent Places Call on Conference Hold)
	FAC [CRV=78, FIE(Invoke, Inv_id=12, Event Report, other call_id=37, resulting call_id=45, event=call conferenced, party_id=1, party_id=2, party_id=3, old party_id=reslt call 1 old party_id=reslt call 2 old party_id=other call 1 connected number=6666, connected number=9999, connected number=######, calling number=6666, called number=9999)]	Conference Event Report (Agent 2345 Completes Conference to Logical Agent 8766)
	FAC [CRV=78, FIE(Invoke, Inv_id=14, Event Report, call_id=45, party_id=1, event=drop, cause=normal, connected number=6666)]	Drop Event Report (Station 6666 Disconnects)
	FAC [CRV=78, FIE(Invoke, Inv_id=16, Event Report, call_id=45, party_id=2, event=drop, cause=normal, connected number=9999)]	Drop Event Report (Station 9999 Disconnects)
	FAC [CRV=78, FIE(Invoke, Inv_id=18, Call Ended, call_id=45, cause=normal)]	Call Terminates

External Call to a Logical Agent's Station Transferred to Another Logical Agent

This scenario shows an incoming ISDN PRI call to a domain-controlled station 6666 (see Figure B-38). Logical agent 2345 (logged in from station 6666) answers the call and transfers the call to logical agent 8766 logged in from station 9999. Logical agent 3456 completes the transfer operation while the call is queued for logical agent 8766.

Note that the **called number** in the Transfer Event Report contains the logical agent's login id extension. If the transferred operation had occurred after the call was delivered to an agent station, the **called party** would have contained the physical station's extension. A call is delivered to a station if the call is either alerting or connected to the station.

Domain Controls for stations 6666 and 9999 are active over CRV 56 and 34, respectively, as shown at the beginning of the call flow. Third Party Domain Control is only allowed on a physical extension number; it is not allowed on a login id extension.

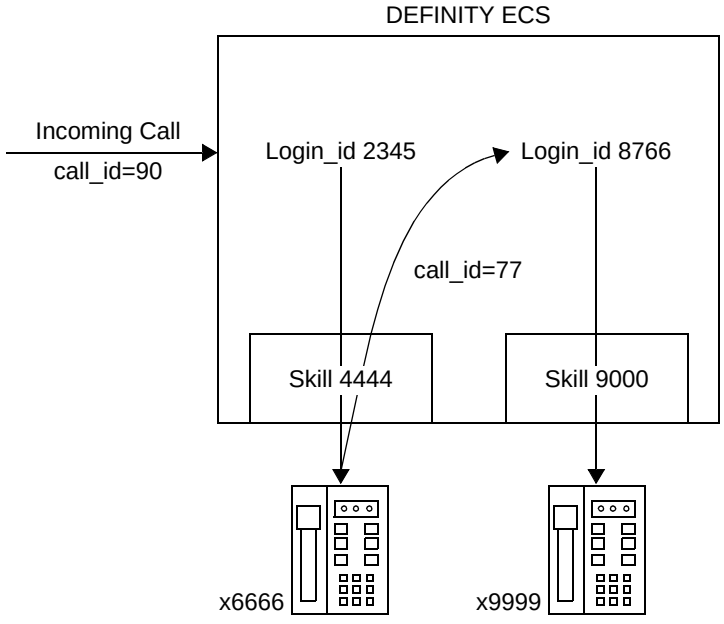


Figure B-38. Call Flow for Incoming Call to Logical Agent Transferred to Another Logical Agent

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=56, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 6666)]		AP Requests Domain Control of Station 6666
	FAC [CRV=56, FIE(Return Result, Inv_id=1)]	Domain Control Accepted
REG [CRV=34, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 9999)]		AP Requests Domain Control of Station 9999
	FAC [CRV=34, FIE(Return Result, Inv_id=1)]	Domain Control Accepted
	FAC [CRV=56, FIE(Invoke, Inv_id=6, Event Report, call_id=90, party_id=2, event=alerting, calling number=7890, called number=2345, connected number=6666)]	Alerting Event Report (Call Delivered to Logical Agent 2345) — Domain Control Association for Station 6666
	FAC [CRV=56, FIE(Invoke, Inv_id=8, Event Report, call_id=90, party_id=2, event=connected, calling number=7890, called number=2345, connected number=6666)]	Connected Event Report (Call Connected to Logical Agent 2345) — Domain Control Association for Station 6666
FAC [CRV=56, FIE(Invoke, Inv_id=3, 3P Selective Hold, call_id=90)]		AP Requests Hold for Station 6666
	FAC [CRV=56, FIE(Return Result, Inv_id=3)]	Call Placed On Hold
FAC [CRV=56, FIE(Invoke, Inv_id5, 3P Auto Dial, called number=8766, return_ack=yes)]		AP Requests Auto Dial Call from Station 6666 to Logical Agent 8766
	FAC [CRV=56, FIE(Invoke, Inv_id=8, 3P Proceed, call_id=77, party_id=12)]	3P Auto Dial ACK
	FAC [CRV=56, FIE(Invoke, Inv_id=10, Event Report, call_id=77, event=queued, called number=8766, calls in queue=7, domain=ACD split 3333)]	Queued Event Report (Queues to Logical Agent 8766) — Domain Control Association for Station 6666

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=56, FIE(Invoke, Inv_id=12, Event Report, other call_id=77, resulting call_id=90, event=call transferred, party_id=10, old party_id=reslt call 10, connected number=######, calling number=6666, called number=8766)]	Transfer Event Report (Agent 2345 Completes Transfer Manually) — Domain Control Association for Station 6666
	FAC [CRV=34, FIE(Invoke, Inv_id=2, Event Report, call_id=90, party_id=2, event=alerting, calling number=7890, called number=8766, connected number=9999, domain=ACD Split 3333)]	Alerting Event Report (Agent 8766 Becomes Available) — Domain Control Association for Station 9999
	FAC [CRV=34, FIE(Invoke, Inv_id=4, Event Report, call_id=90, party_id=2, event=connected, calling number=7890, called number=8766, connected number=9999)]	Connected Event Report (Call Connected to Station 9999) — Domain Control Association for Station 9999
	FAC [CRV=34, FIE(Invoke, Inv_id=10, Event Report, call_id=90, party_id=2, event=drop, cause=normal, connected number=9999)]	Drop Event Report (Station 9999 Disconnects) — Domain Control Association for Station 9999
	FAC [CRV=34, FIE(Invoke, Inv_id=12, Call Ended, call_id=90, cause=normal)]	Call Terminates — Domain Control Association for Station 9999

Direct Agent Call to Logical Agent — Make Call to Login ID

This scenario shows the call flow for a **Third Party Make Call** from logical agent 2345 to a logical agent 8766 (see Figure B-39). Logical agent 2345 is logged in from station 6666 and logical agent 8766 is logged in from station 9999. Logical agent 8766 is not available to receive the call and the call goes to the coverage destination for the login id 8766 (as opposed to following the coverage path associated with station 8900).

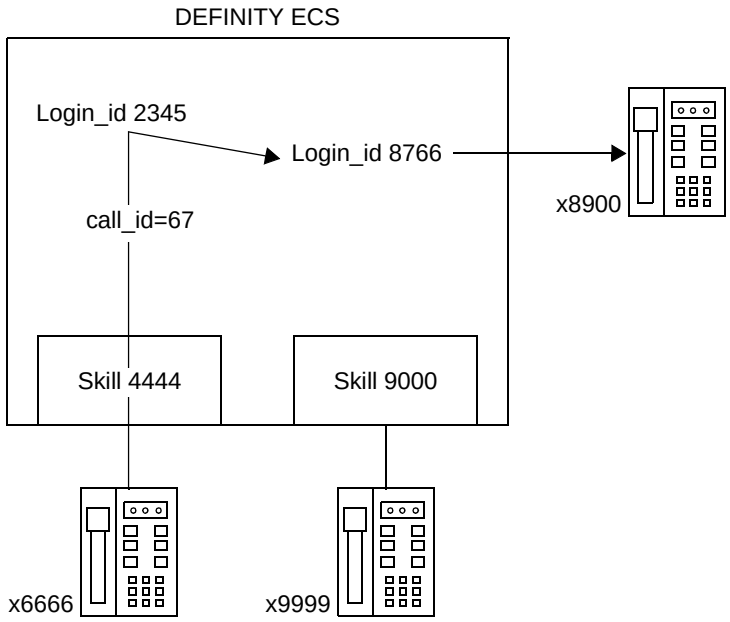


Figure B-39. Call Flow for Direct Agent Call to Logical Agent's Login ID

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=34, FIE(Invoke, Inv_id=1, 3P Make Call, called number=8766, calling number=2345, return_ack=yes)]		AP Requests Make Call from login id 2345 to login id 8766
	FAC [CRV=34, FIE(Invoke, Inv_id=2, 3P Proceed, call_id=67, party_id=11, connected number=6666)]	Make Call ACK (Call Originated from Station 6666)
	FAC [CRV=34, FIE(Invoke, Inv_id=4, Event Report, call_id=67, event=queued, called number=8766, calls in queue=2, domain=ACD Split 9000)]	Queued Event Report (Queued for Logical Agent 8766)
	FAC [CRV=34, FIE(Invoke, Inv_id=6, Event Report, call_id=67, party_id=21, event=alerting, calling number=6666, called number=8766, connected number=8900, domain=ACD Split 9000)]	Alerting Event Report (Call Delivered Coverage Station 8900)
	FAC [CRV=34, FIE(Invoke, Inv_id=8, Event Report, call_id=67, party_id=21, event=connected, calling number=6666, called number=8766, connected number=8900)]	Connected Event Report (Call Connected Station 8900)
	FAC [CRV=34, FIE(Invoke, Inv_id=10, Event Report, call_id=67, party_id=11, event=drop, cause=normal, connected number=6666)]	Drop Event Report (Station 6666 Disconnects)
	REL COMP [CRV=34, FIE (Invoke,Inv_id=12, Call Ended,call_id=67, cause=normal clearing)]	Call Terminates

Value Queries for Logical Agent and Skill Hunt Groups

This scenario shows the Login Audit Query, ACD Agent Status Query, and Extension Query for skill hunt group 4444 and logical agents 2345, 8766, and 6777 (see Figure B-40). Logical agents 2345 and 8766 are logged into skill 4444 from stations 6666 and 9999, respectively. Logical agent 6777 is not logged in.

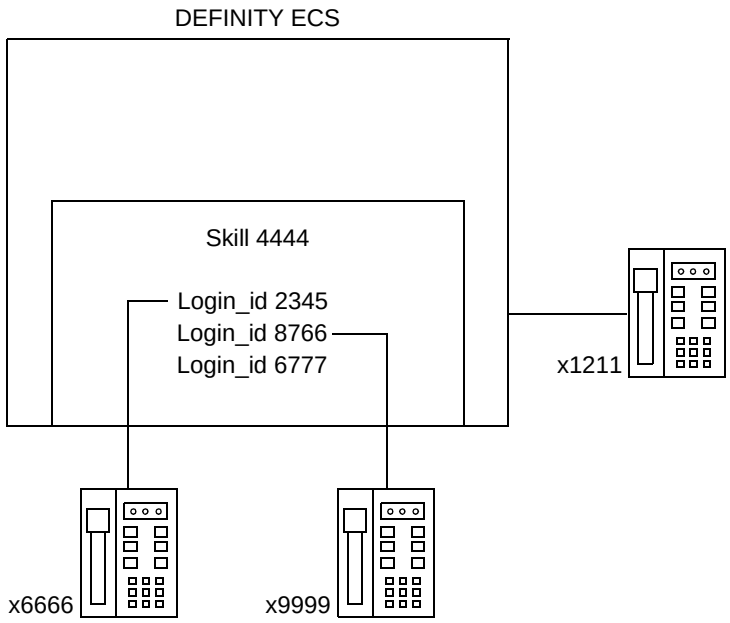


Figure B-40. Skill Hunt Groups and Logical Agents

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=78, FIE(Invoke, Inv_id=1, Value Query, item=Agent Login Audit, domain=ACD Split 4444)]		AP Requests Login Audit for Skill 4444
	REL COMP [CRV=78, FIE(Return Result, Inv_id=1, Value Query, domain=extension 6666, domain=extension 9999)]	Two Stations Logged In to Skill
REG [CRV=24, FIE(Invoke, Inv_id=1, Value Query, domain=extension 2345, domain=ACD split #####)]		AP Requests Status of Agent at Extension 2345 (##### = info ignored)
	REL COMP [CRV=24, FIE(Return Result, Inv_id=1, Value Query, domain=talk state idle, domain=work state auto in)]	Station Idle in Auto In Work Mode
REG [CRV=44, FIE(Invoke, Inv_id=1, Value Query, domain=extension 6666, domain=ACD split #####)]		AP Requests Status of Agent at Extension 6666 (##### = info ignored)
	REL COMP [CRV=44, FIE(Return Result, Inv_id=1, Value Query, domain=talk state idle, domain=work state auto in)]	Station Idle in Auto In Work Mode
REG [CRV=76, FIE(Invoke, Inv_id=1, Value Query, calling number=2345, item=extension)]		AP Requests Extension Query for 2345
	REL COMP [CRV=76, FIE(Return Result, Inv_id=1, Value Query, domain=login id 6666)]	Login Id Logged from station 6666
REG [CRV=76, FIE(Invoke, Inv_id=1, Value Query, calling number=6777, item=extension)]		AP Requests Extension Query for 6777
	REL COMP [CRV=76, FIE(Return Result, Inv_id=1, Value Query, domain=login id)]	Login Id not Logged in

12. Converse Vector Command Interactions

External Call to a VDN with a Converse Step that is Interrupted

This scenario presents the call flow for an incoming ISDN PRI call for VDN 7000 with a converse vector command that can be interrupted (see Figure B-41). The call comes into the VDN and gets queued to two splits, Split 6500 and Split 3400. The converse vector command then sends the call to the VRU (Split 1234) while maintaining the call's position in the other queues. When an agent in Split 6500 becomes available, the call leaves the VRU and is delivered to the agent. This "transfer" happens regardless of whether or not the caller has completed the VRU interaction.

Note that the Alerting Event Report sent when the call alerts the VRU port contains a cause value — CS3/23 (call remains in queue). This cause value informs the application that this is a converse split and that the call will not lose its place in any other splits that it has been queued to.

VDN 7000 has Event Notification active and each port on the VRU has Domain Control active.

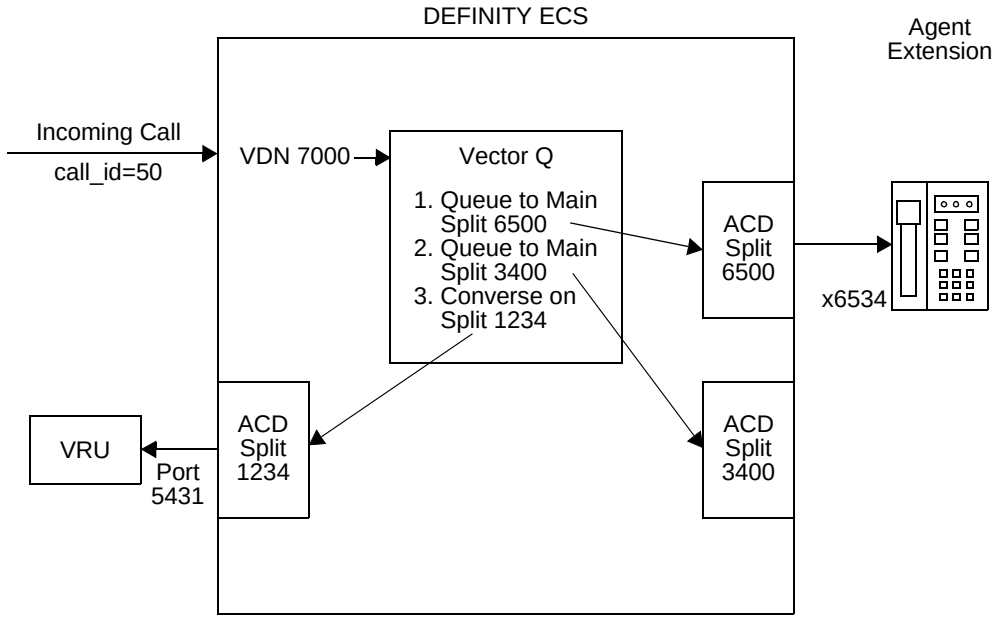


Figure B-41. Call Flow for a Converse Step that can be Interrupted

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=2, Event Report, call_id=50, event=call offered, calling number=9085766362, called number=9089577000, domain=VDN 7000)]	Call Offered to VDN 7000
	FAC [CRV=96, FIE(Invoke, Inv_id=6, Event Report, call_id=50, event=queued, called number=9089577000, calls in queue=3, domain=ACD Split 6500)]	Queued Event Report (Call Queues to ACD Split 6500) — Event Notification Association
	FAC [CRV=96, FIE(Invoke, Inv_id=8, Event Report, call_id=50, event=queued, called number=9089577000, calls in queue=8, domain=ACD Split 3400)]	Queued Event Report (Call Queues to ACD Split 3400) — Event Notification Association
	FAC [CRV=96, FIE(Invoke, Inv_id=14, Event Report, call_id=50, party_id=2, event=alerting, calling number=9085766362, called number=9089577000, connected number=5431, domain=ACD Split 1234, cause value=in queue)]	Alerting Event Report (Call Alerts VRU Port Extension 5431) — Event Notification Association
	FAC [CRV=74, FIE(Invoke, Inv_id=14, Event Report, call_id=50, party_id=2, event=alerting, calling number=9085766362, called number=9089577000, connected number=5431, domain=ACD Split 1234, cause value=in queue)]	Alerting Event Report (Call Alerts VRU Port Extension 5431) — Domain Control Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=12, Event Report, call_id=50, party_id=2, event=connected, calling number=9085766362, called number=9089577000, connected number=5431)]	Connected Event Report (Call Connects to VRU Port Extension 5431) — Event Notification Association
	FAC [CRV=74, FIE(Invoke, Inv_id=18, Event Report, call_id=50, party_id=2, event=connected, calling number=9085766362, called number=9089577000, connected number=5431)]	Connected Event Report (Call Connects to VRU Port Extension 5431) — Domain Control Association
	FAC [CRV=96, FIE(Invoke, Inv_id=10, Event Report, call_id=50, party_id=2, event=alerting, calling number=9085766362, called number=9089577000, connected num=6534, domain=ACD Split 6500)]	Alerting Event Report (Call Delivered to Agent 6534) — Event Notification Association
	FAC [CRV=96, FIE(Invoke, Inv_id=8, Event Report, call_id=50, party_id=2, event=drop, connected number=5431, cause=normal)]	Drop Event Report (VRU port 5431 Disconnected) — Event Notification Association
	FAC [CRV=74, FIE(Invoke, Inv_id=2, Event Report, call_id=50, party_id=2, event=drop, connected number=5431, cause=normal)]	Drop Event Report (VRU port 5431 Disconnected) — Domain Control Association
	FAC [CRV=96, FIE(Invoke, Inv_id=12, Event Report, call_id=50, party_id=2, event=connected, calling number=9085766362, called number=9089577000, connected number=6534)]	Connected Event Report (Agent 6534 Answers) — Event Notification Association

External Call to a VDN with a Converse Step that is not Interrupted

This scenario presents the call flow for an incoming ISDN PRI call for VDN 7001 with a converse vector command that will not be interrupted (see Figure B-42). The converse vector command passes both the ANI and the VDN number to the VRU. The VRU, after completing the session with the caller, sends the call back to vector processing. Along with sending the call back, the VRU also sends data back to the DEFINITY ECS. This data is collected in a collected digits step. An adjunct route is then done that sends these collected digits to the ASAI Adjunct processor. The ASAI adjunct processor then routes the call to ACD Split 3456.

Note that in this scenario, vector processing requires the caller to complete the interaction with the VRU before any additional processing is done to the call. Furthermore, the Alerting Event Report sent when the call alerts the VRU port contains a cause value — CS3/23 (remains in queue). This is to inform the ASAI adjunct processor that this is a converse split.

VDN 7001 has Event Notification active and each port on the VRU has Domain Control active.

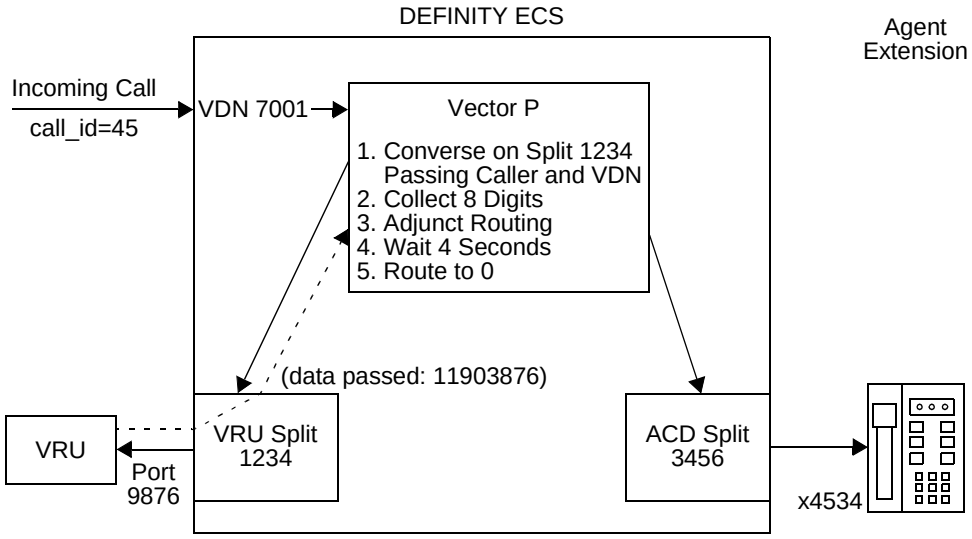


Figure B-42. Call Flow for a Converse Step that will not be Interrupted

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=45, event=call offered, calling number=9085766362, called number=9089577001, domain=VDN 7001)]	Call Offered to VDN 7001
	FAC [CRV=86, FIE(Invoke, Inv_id=6, Event Report, call_id=45, event=queued, called number=9089577001, calls in queue=4, domain=1234)]	Queued Event Report (Queued in Conversant Split 1234) — Event Notification Association
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=45, party_id=2, event=alerting, calling number=9085766362, called number =9089577001, connected number=9876, domain=ACD Split 1234 cause_value=in queue)]	Alerting Event Report (Alerting at VRU Port Extension 9876) — Event Notification Association
	FAC [CRV=97, FIE(Invoke, Inv_id=6, Event Report, call_id=45, party_id=2, event=alerting, calling number=9085766362, called number =9089577001, connected number=9876, domain=ACD Split 1234 cause_value=in queue)]	Alerting Event Report (Alerting at VRU Port Extension 9876) — Domain Control Association
	FAC [CRV=86, FIE(Invoke, Inv_id=8, Event Report, call_id=45, party_id=2, event=connected, calling number=9085766362, called number =9089577001, connected number=9876)]	Connected Event Report — Event Notification Association
	FAC [CRV=97, FIE(Invoke, Inv_id=4, Event Report, call_id=45, party_id=2, event=connected, calling number=9085766362, called number =9089577001, connected number=9876)]	Connected Event Report — Domain Control Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=97, FIE(Invoke, Inv_id=2, Event Report, call_id=45, party_id=2, event=dropped, cause value=normal)]	Drop Event Report (VRU drops off call) — Domain Control Association
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=45, party_id=2, event=dropped, cause value=normal)]	Drop Event Report (VRU drops off call) — Event Notification Association
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, call_id=45, calling number=9085766362, called number=9089577001, user code=11903876)]	Route Request
FAC [CRV=93, FIE(Invoke, Inv_id=7, Route Select, called number=3456)]		Route to ACD Split 3456
	REL COMP [CRV=93, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=86, FIE(Invoke, Inv_id=12, Event Report, call_id=45, event=queued, called number=9089577001, calls in queue=2, domain=ACD Split 3456)]	Queued Event Report (Queued in ACD Split 3456) — Event Notification Association
	FAC [CRV=86, FIE(Invoke, Inv_id=14, Event Report, call_id=45, party_id=2, event=alerting, calling number=9085766362, called number=9089577001, connected number=4534, domain=ACD Split 3456)]	Alerting Event Report (Call Delivered to Agent 4534) — Event Notification Association
	FAC [CRV=86, FIE(Invoke, Inv_id=16, Event Report, call_id=45, party_id=2, event=connected, calling number=9085766362, called number=9089577001, connected number=4534)]	Connected Event Report (Call Connected to Agent 4534) — Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=86, FIE(Invoke, Inv_id=18, Event Report, call_id=45, party_id=2, event=drop, cause=normal, connected number=4534)]	Drop Event Report (Agent disconnects) — Event Notification Association
	FAC [CRV=86, FIE(Invoke, Inv_id=20, Call Ended, call_id=45, cause=normal)]	Call Terminates — Event Notification Association

13. Redirection On No Answer (RONA) Interactions

Call to Agent with RONA

This scenario shows an incoming ISDN PRI call to VDN 7010 that is delivered to extension 6534 in split 6500 (see Figure B-43). The call is not answered by the agent at extension 6534 before the RONA timer expires. When the timer expires, the call is requeued to split 6500 and delivered to agent's station 6540.

In addition, extension 6534 is placed on AUX-work when the RONA timer expires so that no more ACD calls are delivered to the extension. If the call had been sent to an Auto-Available Split (AAS) and the AAS agent or port did not answer, RONA would have taken the agent's extension out of service by automatically logging out the extension that did not answer. If the AAS split has Domain Control active, the switch sends a Logout Event Report for the extension logged out.

VDN 7010 has Event Notification active over CRV 96.

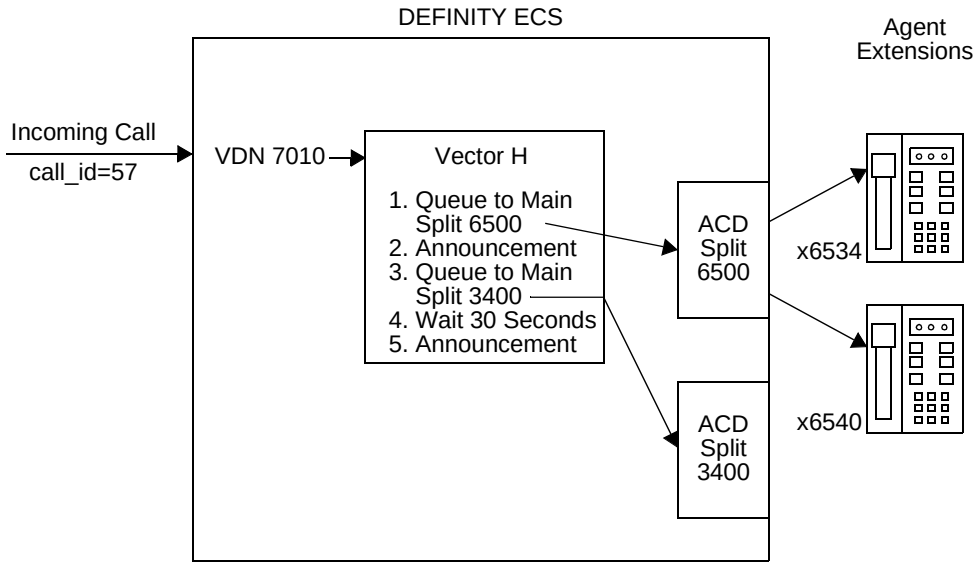


Figure B-43. Call Flow for a Call where RONA Timer Expires

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=2, Event Report, call_id=57, event=call offered, calling number=9085766362, called number=9089579001, domain=VDN 7010)]	Call Offered to VDN 7010
	FAC [CRV=96, FIE(Invoke, inv_id=4, Event Report, call_id=57, event=queued, called number=9089579001, calls in queue=2, domain=ACD Split 6500)]	Queued Event Report (Call Queues to Split 6500)
	FAC [CRV=96, FIE(Invoke, inv_id=6, Event Report, call_id=57, event=queued, called number=9089579001, calls in queue=6, domain=ACD Split 3400)]	Queued Event Report (Call Queues to Split 3400)
	FAC [CRV=96, FIE(Invoke, Inv_id=14, Event Report, call_id=57, party_id=2, event=alerting, calling number=9085766362, called number=9089579001, connected number=6534, domain=ACD Split 6500)]	Alerting Event Report (Call Delivered to Extension 6534)
	FAC [CRV=96, FIE(Invoke, inv_id=24, Event Report, call_id=57, event=queued, called number=9089579001, calls in queue=1, domain=ACD Split 6500)]	Queued Event Report (Call Re-Queues to Split 6500)
	FAC [CRV=96, FIE(Invoke, Inv_id=28, Event Report, call_id=57, party_id=2, event=alerting, calling number=9085766362, called number=9089579001, connected number=6540, domain=ACD Split 6500)]	Alerting Event Report (Call Delivered to Extension 6540)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=30, Event Report, call_id=57, party_id=2, event=connected, calling number=9085766362, called number=9089579001, connected number=6540)]	Connected Event Report (Extension 6540 Answers)
	FAC [CRV=96, FIE(Invoke, Inv_id=36, Event Report, call_id=57, party_id=1, event=drop, connected number=#####, cause=normal)]	Drop Event Report (Caller Disconnects)
	FAC [CRV=96, FIE(Invoke, Inv_id=38, Call Ended, call_id=57, cause=normal clearing)]	Call Terminates

Direct Agent Call with RONA

This scenario presents the call flow for an incoming ISDN PRI call to VDN 8905 that gets routed, via direct-agent call, to extension 1234. The call is not answered by the agent at extension 1234 before the RONA timer expires (see Figure B-44). Because this is a direct-agent call, RONA redirects the call to the agent's coverage path. Furthermore, the agent's extension is placed in the AUX-work mode so that no more ACD calls are delivered to the agent's extension.

If the incoming call had been sent to an Auto-Available Split (AAS) and the agent (or port) selected did not answer before the RONA timer expired, the call would have been redirected back to the split (and queued at the highest priority) for distribution.

Note that an Agent Status Value Query on Extension 1234 was done by the adjunct processor prior to selecting that agent to receive the call. At that point, extension 1234 was in the Auto-In mode and in the idle talk state. A second agent status Value Query was done after the call was redirected away from extension 1234. Then extension 1234 was in the AUX-work mode and in the idle talk state.

Extension 1234 has Domain Control active over CRV 102. VDN 8905 has Event Notification active over CRV 96, and VDN 9876 is not monitored. Extension 1234 is logged into ACD split 1200.

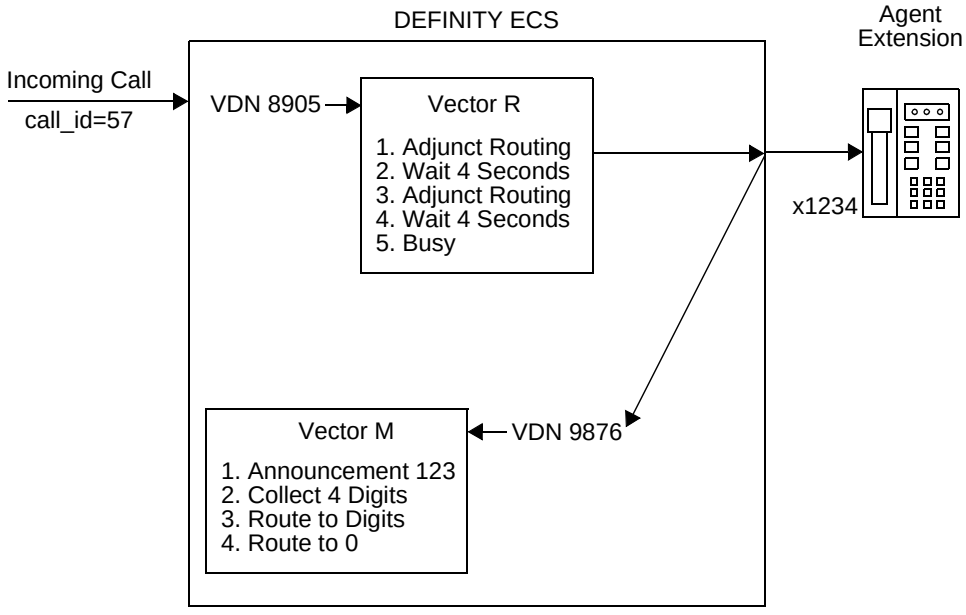


Figure B-44. Call Flow for a Direct Agent Call where RONA Timer Expires

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=2, Event Report, call_id=57, event=call offered, calling number=9085766362, called number=90895789051, domain=VDN 8905)]	Call Offered to VDN 8905
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, call_id=57, calling number=9085766362, called number=9089578905)]	Route Request
REG [CRV=30, FIE(Invoke, Inv_id=7, Value Query, domain=ACD Split 1200, domain=Extension 1234)]		Agent Status Query on Extension 1234
	REG [CRV=30, FIE(Return Result Inv_id=7, Value Query, work mode=auto-in, talk state=idle)]	Response to Agent Status Query
FAC [CRV=93, FIE(Invoke, Inv_id=7, Route Select, called number=1234, domain=ACD Split 1200, direct agent call=yes)]		Route to Agent (Extension 1234)
	REL COMP [CRV=93, FIE(Invoke, Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=96, FIE(Invoke, Inv_id=14, Event Report, call_id=57, party_id=2, event=alerting, calling number=9085766362, called number=9089578905, connected number=1234, domain=ACD Split 1200)]	Alerting Event Report (Call Delivered to Extension 1234)— Event Notification Association
	FAC [CRV=102, FIE(Invoke, inv_id=18, Event Report, call_id=57, party_id=2, event=alerting, calling number=90895766362, called number=9089578905, connected number=1234, domain=ACD Split 1200)]	Alerting Event Report (Call Delivered to Extension 1234) — Domain Control Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=102, FIE(Invoke, inv_id=8, Event Report, call_id=57, event=call redirected)]	Call Redirected Event RONA Redirect Call to Coverage Path
REG [CRV=45, FIE(Invoke, Inv_id=9, Value Query, domain=Split 1200, domain=Extension 1234)]		Agent Status Query
	REL COM [CRV=45, FIE(Return Result, Inv_id=9, Value Query, work mode=AUX-work, talk state=idle)]	Response to Agent Status Query

14. VDN in Coverage Path Interactions

Incoming Call routed to a Station that has a VDN in the Coverage Path

This scenario shows the call flow for an incoming non-ISDN call that gets routed to extension 1234 via the adjunct routing command. Extension 1234 does not answer the call and the call covers to extension 9876. Extension 9876 does not answer the call and the third coverage point is VDN 3634 (see Figure B-45).

VDN 8905 has Event Notification active over CRV 96. Extensions 1234 and 9876 have Domain Control active over CRV 80 and 95, respectively.

The ASAI messages generated by the Adjunct Routing vector command are also shown.

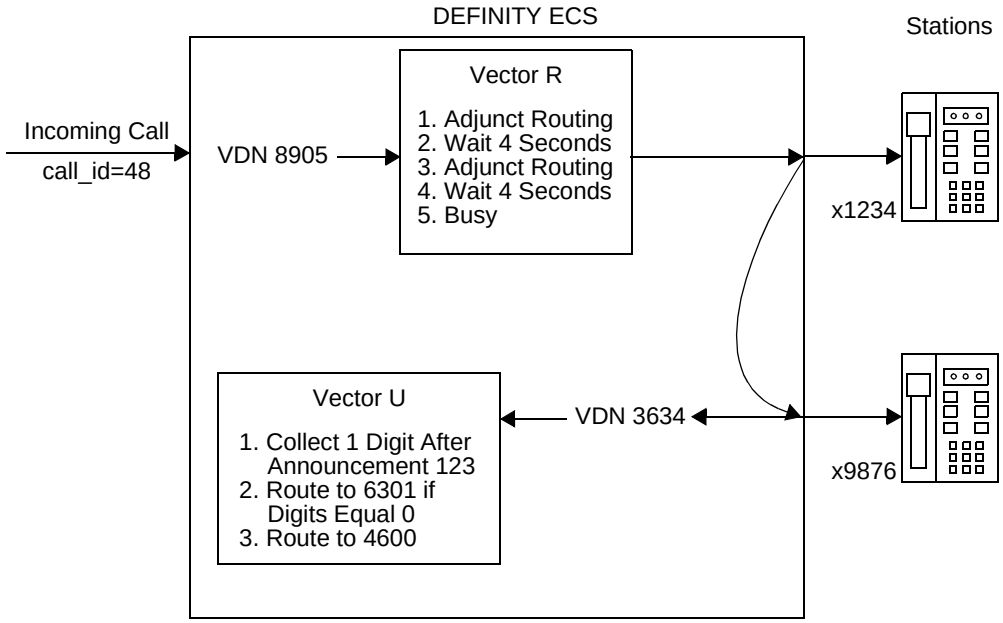


Figure B-45. Call Flow for an Agent who has a VDN in the Coverage Path

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=96, FIE(Invoke, Inv_id=4, Event Report, call_id=48, event=call offered, trunk group number=67, called=8905, domain=VDN 8905)]	Call Offered to VDN 8905
	REG [CRV=100, FIE(Invoke, Inv_id=6, Route, call_id=48, trunk group number=67, called=8905, domain=VDN 8905)]	Route Request
FAC [CRV=100, FIE(Invoke, Inv_id=5, Route Select, called number=1234)]		Route to Extension 1234
	REL COMP [CRV=100, FIE (Invoke,Inv_id=4, Route End, cause=normal)]	Call Routed
	FAC [CRV=96, FIE(Invoke, Inv_id=8,Event Report, call_id=48,party_id=2, event=alerting, trunk group number=67, called number=8905, connected number=1234)]	Alerting Event Report (Extension 1234 Alerts) — Event Notification Association
	FAC [CRV=80, FIE(Invoke, Inv_id=8, Event Report, call_id=48, party_id=2, event=alerting, trunk group number=67, called number=8905, connected number=1234)]	Alerting Event Report (Extension 1234 Alerts) — Domain Control Association for Extension 1234
	FAC [CRV=96, FIE(Invoke, Inv_id=8, Event Report, call_id=48,party_id=4, event=alerting, trunk group number=67, called number=8905, connected number=9876)]	Alerting Event Report (Extension 9876 Alerts) — Event Notification Association

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=95, FIE(Invoke, Inv_id=8, Event Report, call_id=48,party_id=4, event=alerting, trunk group number=67, called number=8905, connected number=9876)]	Alerting Event Report (Extension 9876 Alerts) — Domain Control Association for Extension 9876
	FAC [CRV=80,FIE(Invoke, Inv_id=8, Event Report, call_id=48, event=call redirected)]	Call Redirected Event Report (Call Enters Coverage VDN) — Domain Control Association for Extension 1234
	FAC [CRV=95,FIE(Invoke, Inv_id=8, Event Report, call_id=48, event=call redirected)]	Call Redirected Event Report (Call Enters Coverage VDN) — Domain Control Association for Extension 9876

External Call to a VDN with a Forced First Announcement that gets Routed to a Second VDN

This section presents the call flow for an incoming ISDN PRI call for VDN 5678 that hears a forced first announcement (see Figure B-46). After the announcement, the call gets routed via the Adjunct Route vector command to VDN 5700. The call eventually gets answered by Agent 4566 in Split 3460.

Note that no event reports are generated for the announcement. In general, ACD split forced first or second announcements and vector-controlled announcements do not generate events. However, event reports are generated for non-split announcements.

VDN 5678 has Event Notification active over CRV 98.

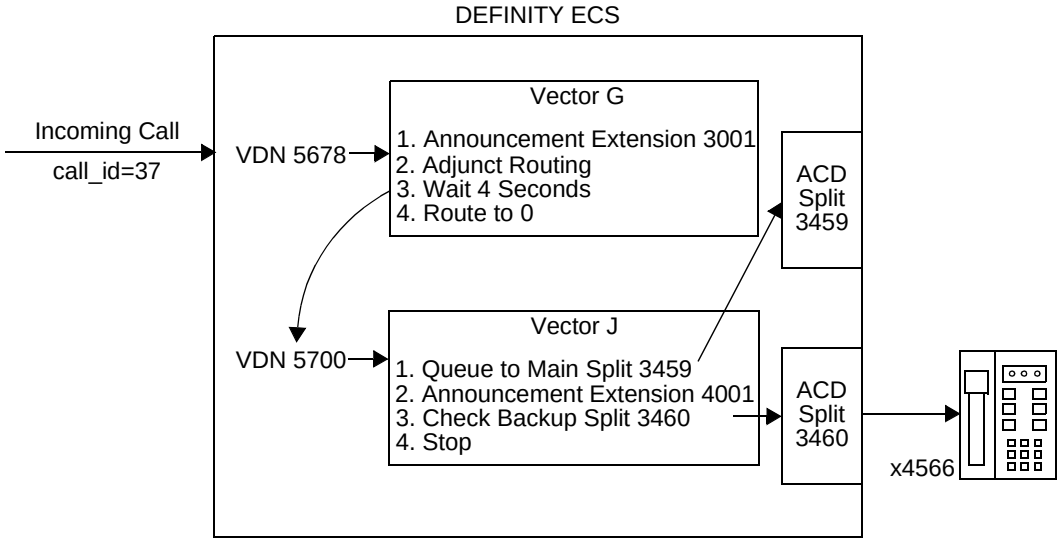


Figure B-46. Call Flow for Call to a VDN with Announcement and Routed to Another VDN

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=2, Event Report, call_id=37, event=call offered, calling number=9085669660, called number=5678, domain=VDN 5678)]	Call Offered to VDN 5678
	REG [CRV=93, FIE(Invoke, Inv_id=4, Route, calling number=9085669660, called number=5678)]	Route Request
FAC [CRV=93, FIE(Invoke, Inv_id=7, Route Select, called number=5700)]		Route to VDN 5700
	REL COMP [CRV=93, FIE (Invoke,Inv_id=6, Route End, cause=normal)]	Call Routed
	FAC [CRV=98, FIE(Invoke, Inv_id=98, Event Report, call_id=37, event=call offered, calling number=9085669660, called number=5678, domain=VDN 5700)]	Call Offered to VDN 5700
	FAC [CRV=98, FIE(Invoke, Queued Event Report Inv_id=12, Event Report, call_id=37, event=queued, calls in queue=21, domain=3459)]	Call Queues to ACD Split 3459
	FAC [CRV=98, FIE(Invoke, Queued Event Report Inv_id=12, Event Report, call_id=37, event=queued, calls in queue=2, domain=3460)]	Call Queues to ACD Split 3460

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=98, FIE(Invoke, Inv_id=14, Event Report, call_id=37, party_id=2, event=alerting, calling number=9085669660, called number=5678, connected number=4566, domain=3460)]	Alerting Event Report (Call Delivered to Agent 4566)
	FAC [CRV=98, FIE(Invoke, Inv_id=16, Event Report, call_id=37, party_id=2, event=connected, calling number=9085669660, called number=5678, connected number=4566)]	Connected Event Report (Call Connected to Agent 4566)
	FAC [CRV=98, FIE(Invoke, Inv_id=18, Event Report, call_id=37, party_id=2, event=drop, cause=normal, connected number=4566)]	Drop Event Report (Agent disconnects)
	FAC [CRV=98, FIE(Invoke, Inv_id=20, Call Ended, call_id=37, cause=normal)]	Call Terminates

Outgoing Call over Non-ISDN Trunk

This section presents the call flow for an outgoing call over a non-ISDN trunk. Station 1234 initiates this preview dialing call (see Figure B-47).

Note that a Trunk Seized Event Report is generated when the switch places the call over a non-ISDN trunk. Furthermore, no Alerting or Connected Event Reports follow a Trunk Seized Event Report. The only event report that may be generated for the destination is a Dropped Event Report.

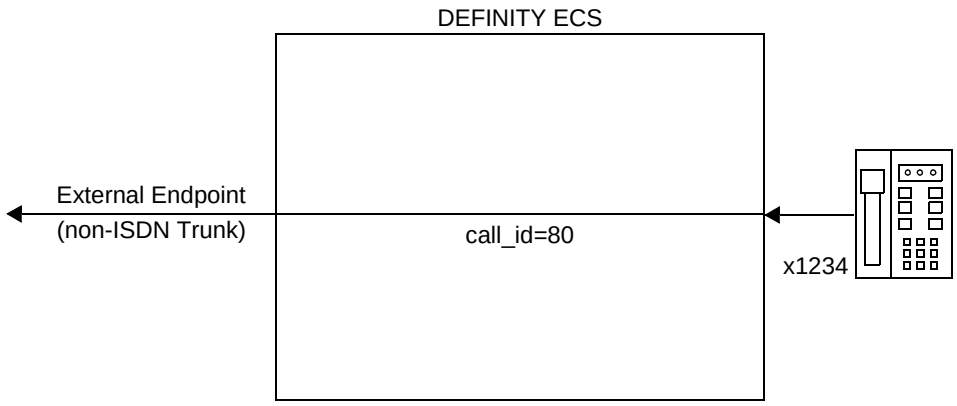


Figure B-47. Outgoing Call over Non-ISDN Trunk

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=96, FIE(Invoke, Inv_id=3, 3P Make Call, called number=99085766362 calling number=1234)]		AP Requests Make Call from Station 1234 to External Destination
	FAC [CRV=96, FIE(Invoke, Inv_id=4, Event Report, call_id=80, party_id=2, event=trunk seized, called number=######)]	Trunk Seized Event Report (Off-premises Call over non-ISDN trunk)
	FAC [CRV=96, FIE(Invoke, Inv_id=6, Event Report, call_id=80, party_id=1, event=hold, connected number=1234)]	Hold Event Report (Station 1234 goes on Hold)
FAC [CRV=96, FIE(Invoke, Inv_id=5, 3P Selective Reconnect, party_id=1)]		AP Requests Reconnect
	FAC [(CRV=96, FIE(Return Result, Inv_id=5)]	Call Reconnected
	FAC [(CRV=96, FIE(Invoke, Inv_id=8, Event Report, call_id=80, party_id=2, event=drop, connected number=######)]	Drop Event Report for External Destination
	REL COMP [(CRV=96, FIE (Invoke, Inv_id=4, Call Ended, call_id=80, cause=normal clearing)]	Call Terminated

Outgoing Call over ISDN Trunk that Results in an ISDN Progress Message

This section presents the call flow for an outgoing call over an ISDN trunk that traverses one or more non-ISDN networks before it reaches its destination. Station 4567 initiates an outgoing call to an external destination. Station 4567 has Domain Control active and uses the Auto Dial capability to initiate the call (see Figure B-48).

When a call leaves the ISDN network, an ISDN PROGress message is sent from the PRI network to the switch and subsequently to the ASAI Adjunct Processor. The switch sends the contents of the PROGress message in a Cut-Through Event Report. Multiple PROGress messages may be sent for a call; each one is mapped into a Cut-Through Event Report.

For a call that has resulted in a Cut-Through Event Report being generated, the Alerting Event Report is optional. A Connected and/or Drop Event Report is always sent as long as the call utilizes the ISDN facilities.

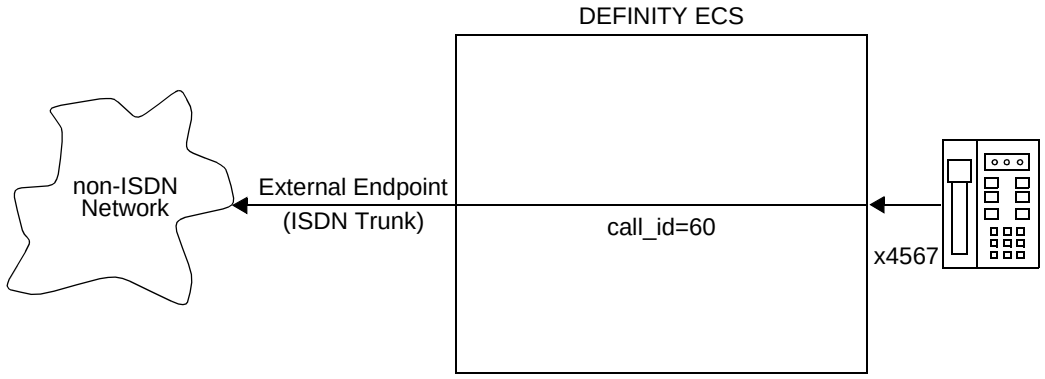


Figure B-48. Call Flow for Outgoing ISDN Call that Traverses a Non-ISDN Network

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
	FAC [CRV=30, FIE(Invoke, Inv_id=4, event report, call_id=60, party_id=1, event=call initiated)]	Call Initiated Event (Station 4567 goes off-hook)
FAC [CRV=30, FIE(Invoke, Inv_id=5, 3P Auto Dial, called number=99085766362)]		AP Requests Auto Dial for Station 4567
	FAC [CRV=30, FIE(Invoke, Inv_id=6, Event Report, call_id=60, party_id=1, calling number=4567, called number=99085766362, event=call originated)]	Call Originated
	FAC [CRV=30, FIE(Invoke, Inv_id=10, Event Report, call_id=60, party_id=2, event=cut-through Location=public network serving local user, progress=in-band info now available)]	Cut-Through Event Report (Contains the Progress Indicator IE)
	FAC [CRV=30, FIE(Invoke, Inv_id=8, Event Report, call_id=60, party_id=2, event=connected, calling number=4567, called number=99085766362, connected number=9085766362)]	ISDN Connect
	FAC [CRV=30, FIE(Invoke, Inv_id=6, Event Report, call_id=60, party_id=2, event=drop, connected=######, cause=normal clearing)]	(External Destination Drops)
	FAC [CRV=30, FIE(Invoke, Inv_id=4, Event Report, call_id=60, party_id=1, event=drop, connected=4567, cause=normal clearing)]	(Station 4567 Drops)

15. User to User Information (UUI)

User Scenarios — User to User Information

Figure B-49 shows a typical distributed call configuration. An ASAI host is connected to each of the switches and calls are delivered to either switch. The applications running on the ASAI hosts are not connected to each other.

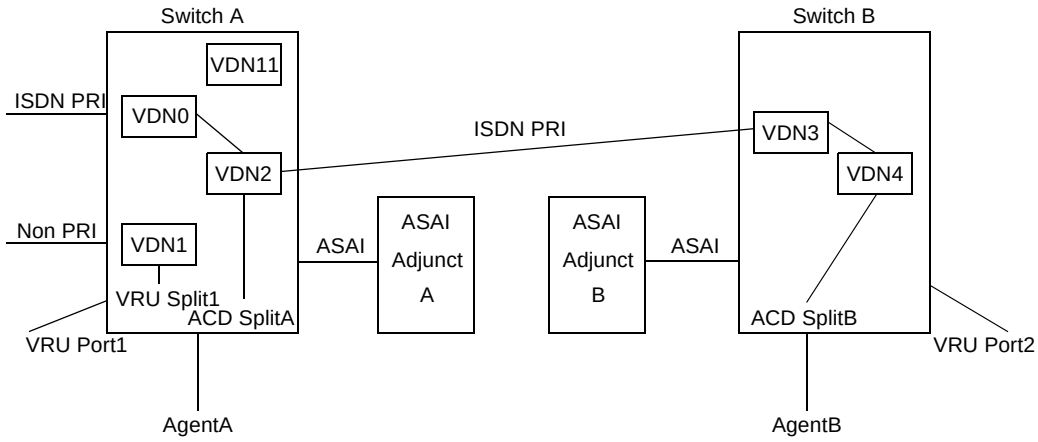


Figure B-49. Distributed ACD Configuration

The following call sequences show typical UUI scenarios. Call Sequence 1 shows an incoming non-ISDN PRI call to Switch A, delivered to VDN1. VDN1 delivers the call to VRU Port1 in VRU Split1. After the caller terminates the VRU session, the ASAI host transfers the call to VDN2 for further processing. The ASAI host includes UUI in the 3rd Party Make Call used to start the transfer. While the call is in VDN2, the call interflows to Switch B. The call is accepted at Switch B and an announcement is played while the call waits in queue for ACD Split B.

Call Sequence 2 shows an incoming ISDN PRI call to Switch A, delivered to VDN1. VDN1 contains a collect digits vector step followed by an adjunct routing vector step. The host routes the call, including UUI information, to VDN2 that tries to interflow the call to Switch B. Switch B does not accept the call and the call connects to Agent A in Switch A.

Call Sequence 3 shows an incoming ISDN PRI call to Switch A, delivered to VDN1. The incoming ISDN call contains UUI data. While in VDN1, the call is routed to VDN3 in Switch B including UUI information. Switch B connects the call to VRU Port2. After the VRU terminates the caller session, the ASAI host drops the call, including UUI information, back to Switch A. When the trunk to Switch B drops, the call is directed to VDN11. The host on Switch A drops the call including UUI information in the request.

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
Call Sequence 1: Incoming Call to Switch A	Call Offered call_id=1234 called number=VDN1 domain=VDN1 trunk group=AAA	
Call Delivered to VRU	Alerting call_id=1234 called number=VDN1 connected number=VRU PORT domain=VRU Split1 trunk group=AAA party_id=2	
Call Connected to VRU	Connected call_id=1234 called number=VDN1 connected number=VRU PORT1 trunk group=AAA party_id=2	
VRU Places Call on Hold	Hold call_id=1234 connected number=VRU PORT1 party_id=2	
VRU Requests new call	3rd Party Make Call calling number=VRU PORT1 called number=VDN2 UUI=info1	
VRU Transfers Call to VDN2	Call Transferred resulting call_id=5678 other call_id=1234 called number=VDN2 calling number=VRU PORT1 connected number=###### party_id=1 old party_id=1	
Call Queues	Queued call_id=5678 calls in queue=n called number=VDN2 domain=ACD SplitA	

(Continued on next page)

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
Call Intraflows to Switch B (VDN 3 is the LAI VDN; VDN 4 is the VDN used for accepted calls)		Call Offered call_id=9000 called number=VDN3 calling number domain=VDN4 LAI info=LAI info UUI=info1
Call Queues to Split		Queued call_id=9000 calls in queue=1 called number=VDN3 domain=ACD SplitB
Call Connects to Announcement	Connected call_id=5678 called number=VDN3 trunk group=AAA party_id=2	
Call Delivered to Agent		Alerting call_id=9000 called number=VDN3 calling number connected number=agentB domain=ACD SplitB party_id=2 UUI=info1
Call Connected to Agent		Connected call_id=9000 called number=VDN3 calling number connected number=agentB party_id=2
Caller Drops	Drop call_id=5678 connected number=##### party_id=1	
Agent B Disconnects		Drop call_id=9000 connected number=agentB party_id=2

(Continued on next page)

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
	Call Ended call_id=5678 cause=normal	
		Call Ended call_id=9000 cause=normal
Call Sequence 2: Incoming Call to Switch A	Call Offered call_id=1235 called number=1235678914 domain=VDN1 calling number=CPN/BN	
Digits Collected by Call Prompting		
Switch Request Route for Call	Route Request call_id=1235 called number=1235678914 calling number=CPN/BN domain=VDN0 collected digits=001	
Host Provides Route	Route Select called number=VDN2 UUI=info2	
Call Routed	Route End cause=normal	
Call Queues	Queued call_id=1235 calls in queue=y called number=VDN2 domain=ACD SplitA	
Call Intraflows to Switch B (VDN 3 is the LAI VDN)		Call Offered call_id=9000 called number=VDN3 calling number=CPN/BN domain=VDN3 LAI info=LAI info UUI=info2
Call Denied by the Switch B	Drop call_id=1235 connected number=###### party_id=2	Call Ended call_id=9001 cause=normal

(Continued on next page)

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
Call Delivered to Agent	Alerting call_id=1235 called number=1235678914 calling number=CPN/BN connected number=agentA domain=ACD SplitA party_id=2 UUI=info2	
Call Connected to Agent	Connected call_id=1235 called number=1235678914 calling number=CPN/BN connected number=agentA party_id=2	
Agent Drops	Drop call_id=1235 connected number=agentA party_id=2	
Call Terminates	Call Ended call_id=1235 cause=normal	

(Continued on next page)

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
Call Sequence 3: Incoming Call to Switch A	Call Offered call_id=1236 called number=1235678914 domain=VDN1 calling number=CPN/BN UUI=info0	
Switch Request Route for Call	Route Request call_id=1236 called number=1235678914 calling number=CPN/BN domain=VDN0 UUI=info0	
Host Provides Route	Route Select called number=VDN3 UUI=info3	
Call Routed	Route End cause=normal	
Call Enters Switch B		Call Offered call_id=9002 called number=VDN3 calling number=CPN/BN domain=VDN3 UUI=info3
Call Delivered to VRU	Alerting call_id=1236 called number=VDN3 calling number=CPN/BN connected number=##### UUI=info3	Alerting call_id=9002 called number=VDN3 calling number=CPN/BN connected number=VRU PORT2 party_id=2 domain=ACD SplitB UUI=info3
Call Connected to VRU Port	Connected call_id=1236 called number=VDN3 calling number=CPN/BN connected number=##### party_id=2	Connected call_id=9002 called number=VDN3 calling number=CPN/BN connected number=VRU PORT2 party_id=2

(Continued on next page)

Action/Operation	Host A — Event Notification VDN 1	Host B — Event Notification VDN 4
VRU is disconnected		3rd Party Drop call_id=9002 party_id=1 UUI=info4
Call Disconnected (Switch B)	Drop call_id=1236 connected number=##### party_id=2 UUI=info4	3rd Drop-ACK
		Call Ended call_id=9002 cause=normal
Call Delivered to Return VDN 11	Call Offered call_id=1236 called number=1235678914 domain=VDN11 calling number=CPN/BN UUI=info4	
Host Drops Call	3rd Party Drop call_id=1236 party_id=1 UUI=info5	
Caller Disconnected	3rd Party Drop-ACK Drop call_id=1236 connected number=##### party_id=1 UUI=info5	
Call Terminates	Call Ended call_id=1236 cause=normal	

16. User Scenarios — Connected IE for non-ISDN Trunks

Table B-1 shows the Event Reports provided for a monitored call that is routed over an outgoing non-ISDN trunk. The incoming call also uses a non-ISDN trunk and is directed to a VDN/vector that routes the call to an external number.

Table B-1. Incoming Call Routed to External Destination Example

Operation	Event Report
Incoming Call	Call Offered call_id=45 trunk group=102 called number=65678 domain=VDN 65678
Call Routed to External Destination	
Non-ISDN Trunk Seized	Trunk Seized call_id=45 party_id=2 called number=#####
Call Connected Answer Supervision Received from the Network	Connect call_id=45 party_id=2 trunk group=102 called number=##### connected number=##### cause=normal
Called Party Drops	Drop call_id=45 party_id=2 cause=normal connected number=#####
Call Terminates	Call Ended call_id=45 cause=normal

17. User Scenarios — ASAI-Provided Dial-Ahead Digits

This is a simple scenario in which the host provides dial-ahead digits via a Route Select. After the dial-ahead digits are stored by the switch, the digits are collected using call prompting vector commands. The scenario also shows the ASAI Event Reports sent to a monitoring host.

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Call Offered ER call_id=45, calling number=3156778888, called number=5678, domain=VDN 65678	Incoming ISDN Call to VDN Vector Executed (VDN 65678): 1. Collect 1 digit 2. Adjunct Routing 3. Wait 4 secs
	Route Request call_id=45, calling number=3156778888, called number=5678, domain=VDN 65678, user code=(call prompt, collected, digits=2)	Switch Requests Route
Route Select call_id=45, called number=null, user code=(cdp, collected, digits=123)		Host Provides 3 Digits & Invalid Route
	Route End cause=invalid number	Vector Execution Continues: 4. Collect 1 Digit 5. Goto step 8 if digits=1 6. Goto to step 9 if digits=2 7. Goto vector 111 if uncond 8. Route to 61123 if uncond 9. Route to 64444 if uncond

(Continued on next page)

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Route Request call_id=45, calling number=3156778888, called number=5678, domain=VDN 61123, user code=(call prompt, collected, digits=23)	Switch Requests Another Route Vector Executed (VDN 61123): 1. Collect 2 Digits 2. Adjunct Route 3. Wait 4 secs
Route Select call_id=45, called number=69990, user code=(cdp, collected,digits=6789)		Host Routes to VDN 69990
	Route End cause= normal	Call Routed VDN 69990 Executed: 1. Collect 4 Digits 2. Queue to 55555 3. Announcement
	Alerting ER call_id=45, party_id=2, calling number=3156778888, called number=5678, connected number=7777, domain=Split 55555	Call Delivered to Agent 7777
	Connect ER call_id=45 party_id=2, calling number=3156778888, called number=5678, connected number=7777	Agent 7777 Answers Call Agent Display for Call Prompting Digits Shows 6789
	Drop ER call_id=45, party_id=1, connected number=##### cause=normal	Calling Party Drops
	Call Ended call_id=45 cause=normal	Call Terminates

18. User Scenarios — ASAI-Requested Digit Collection

This is a sample scenario for an incoming ISDN call routed via Adjunct Routing to an external destination. The user has subscribed to receive 4-digit DNIS numbers. As part of the route, the host requests collecting 3 digits from the caller.

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Call Offered ER call_id=45, calling number=3156778888, called number=5678, domain=VDN 65678	Incoming ISDN Call to VDN Vector Executed: 1. Collect 1 digit 2. Adjunct Routing 3. Wait 4 secs
	Route Request call_id=45, calling number=3156778888, called number=5678, domain=VDN 65678, user code=(call prompt, collected, digits=2)	Switch Requests Route
Route Select call_id=45, called number=84154422800, party_id=1 user code=(tone detect, collect, timer=0, digits=3), event=connect		Host Routes to External Destination, Requests Tone Detector for Caller Party, Disconnect Upon Answer
	Route End cause=normal	Call Routed
	Trunk Seized ER call_id=45, party_id=2, connected number=######,	Route Call Over Non-ISDN Trunk

(Continued on next page)

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Entered Digits ER call_id=45, user code=(tone detect, collected, no timer, digits=4*#)	Digits Entered
	Connect ER call_id=45, party_id=2 calling number=3156778888, called number=5678, connected number=#####, cause=network connect	Call Answered by Destination
	Drop ER call_id=45, party_id=1 cause=normal connected number=#####	Calling Party Drops
	Call Ended call_id=45 cause=normal	Call Terminates

19. User Scenarios —VDN Return Destination

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 extension XXX1
3. Wait through 6 seconds of silence
4. Collect 16 digits after announcement 1002 (“Please enter the telephone number of your destination followed by # sign.”)
5. Adjunct Routing link extension XXX1
6. Wait through 6 seconds of silence
7. Disconnect after announcement 1003 (“We are sorry, but we are experiencing technical difficulties at this time, please try again later.”)

In this scenario, a remote caller calls into the switch by dialing the VDN administered with the Return Destination. The vector executed prompts the caller to enter an identification number and a password that is 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 collects digits for the destination 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 routes the call to an appropriate alternate destination (for example, announcement with security violation message) and logs the invalid call attempt. If the host is not available, the call is 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 extension XXX1
3. Wait through 6 seconds of silence
4. Disconnect after announcement 1003 (“We are sorry, but we are experiencing technical difficulties at this time, please try again later.”)

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 the call was routed to would ring, alerting the called party, but 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.

20. ASAI Messaging Scenarios — VDN Return Destination

This is a scenario where a call to a vector is routed using Adjunct Routing to an external destination. The host then drops the external destination and the call is delivered to the Return Destination for further vector processing. The scenario assumes that the call is being monitored by the ASAI host and that the Return Destination is VDN 77777.

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Call Offered ER call_id=45, calling number=3156778888, called number=9089575678, domain=VDN 65678	Incoming ISDN Call to VDN — Monitoring Association Return Destination=77777
	Route Request call_id=45, calling number=3156778888, called number=9089575678, domain=VDN 65678, user code=(call prompt, collected, digits=2)	Switch Requests Route
Route Select call_id=45, called number=8095766163, access code=9 user code=(tone detect, collect, timer=0, digits=6789)		Host Routes to External Destination, and Requests a Tone Detector
	Route End cause=normal	Call Routed
	Trunk Seized ER call_id=45, party_id=2, connected number=#####,	Call Seizes Outgoing Trunk — Monitoring and Controlling Association
	Entered Digits ER call_id=45, user code=(tone detect, collected, timer=0, digits=#)	Digits Entered Monitoring and Controlling Association

(Continued on next page)

Messages Sent by the Host	Messages Sent by the Switch	Comments
Take Control call_id=45	Take Control ACK	Host takes control of call so that it can drop trunk and have call return for VDN processing
3rd Party Drop party_id=2	3rd Party Drop-ACK	Host Requests Drop of Called Party Called Party Dropped
	Drop ER call_id=45, party_id=2 cause=normal connected number=#####	Called Party Dropped — Monitoring Association
		Call Delivered to Return Destination — VDN 77777 (No Digit Collection)
	Queued ER call_id=45, calls in queue=2, domain = ACD 4567, called number=9085766163	Call Queues
	Alert Event call_id=45, party_id=2 calling number=3156778888, called number=9085766163, connected number=12345, domain=Split 4567	
	Connect ER call_id=45, party_id=2 calling number=3156778888, called number=9085766163, connected number=12345, domain=Split 4567	Call Delivered to Agent 12345 — Agent Display for Call Shows Nothing

(Continued on next page)

Messages Sent by the Host	Messages Sent by the Switch	Comments
	Drop ER call_id=45, party_id=2 cause=normal connected number=12345	Called Party/Agent Drops
	Call Ended call_id=45 cause=normal	Call Terminates

21. Flexible Billing

User Scenarios — Flexible Billing

The following call sequences show typical Flexible Billing scenarios.

Call Sequence 1 shows an incoming call on an ISDN-PRI trunk delivered to Split A. The ISDN trunk is configured for MultiQuest service. The incoming call indicates in its SETUP message that Flexible Billing is supported. The call rings at agent 5001 and is answered. The agent requests a billing change on the call, setting the new rate to $10/minute. This billing change is sent over the ISDN-PRI trunk, and the 4ESS rejects the change. The switch sends the response to the ASAI adjunct. The agent requests a billing change on the call, setting the new rate to $5/minute. This billing change is sent over the ISDN-PRI trunk, and the 4ESS accepts the change. The switch sends the response to the ASAI adjunct.

Call Sequence 2 shows an incoming call on an ISDN-PRI trunk delivered to VDN A. The ISDN trunk is configured for MultiQuest service. The incoming call indicates in its SETUP message that Flexible Billing is supported. The call is adjunct-routed. The adjunct sees that Flexible Billing is enabled, and routes the call to split 1. The call rings at agent 5001 and is answered. The switch sends the response to the ASAI adjunct. The agent requests a billing change on the call, setting the new rate to $5/minute. This billing change is sent over the ISDN-PRI trunk, and the 4ESS accepts the change. The switch sends the response to the ASAI adjunct.

Call Sequence 3 shows an incoming call on an ISDN-PRI trunk delivered to VDN A. (This call sequence uses the same VDN as call sequence 2.) The ISDN trunk is configured for MultiQuest service. The incoming call indicates in its SETUP message that Flexible Billing is NOT supported. The call is adjunct-routed. The adjunct sees that Flexible Billing is NOT enabled and routes the call to split 2. The call rings at agent 5001 and is answered.

In Call Sequence 2, the agent knew that Flexible Billing was available on the call since the call routed to split 1. In Call Sequence 3, the agent knew to bill by a means other than Flexible Billing since the call routed to split 2.

Action/Operation	Event Notification Split 1	Set Value (billing Change)
Call Sequence 1:		
Incoming ISDN-PRI MultiQuest call with Flexible Billing	Call Offered call_id=1111 called number=9085551234 domain=Split1 calling number=CPN/BN Flexible Billing is enabled	
Call rings at agent	Alerting call_id=1111 called number=9085551234 calling number=CPN/BN	
Agent answers call	Connected call_id=1111 called number=9085551234 calling number=CPN/BN connected number=5001 party_id=2	
Agent requests billing change to $10/minute (via adjunct)		Set Value Flexible Billing type=new amount=1000
Billing change is sent to 4ESS via ISDN-PRI trunk		
4ESS rejects billing change. PBX notifies adjunct.		Release Complete Reject
Agent requests billing change to $5/minute (via adjunct)		Set Value Flexible Billing type=new rate amount=500
Billing change is sent to 4ESS via ISDN-PRI trunk		
4ESS accepts billing change. PBX notifies adjunct.		Release Complete Return Result

(Continued on next page)

Action/Operation	Route Request	Event Notification Split 1	Set Value (billing Change)
Call Sequence 2:			
Incoming ISDN-PRI MultiQuest call with Flexible Billing			
Adjunct Route Request	Route Request call_id=1112 called number=9085551234 domain=VDN A calling number=CPN/BN Flexible Billing is enabled		
Adjunct replies to Route Request	Route Select called number=Split 1		
Call is offered to split 1		Call Offered call_id=1112 called number=Split1 domain-=Split1 calling number=CPN/BN Flexible billing is enabled	
Call rings at agent		Alerting call_id=1112 called number=9085551234 calling number=CPN/BN	

(Continued on next page)

Action/Operation	Route Request	Event Notification Split 1	Set Value (billing Change)
Agent answers call		Connected call_id=1112 called number=9085551234 calling number=CPN/BN connected number=5001 party_id=2	
Agent requests billing change to $5/minute (via adjunct)			Set Value Flexible Billing type=new rate amount=500
Billing change is sent to 4ESS via ISDN-PRI trunk			
4ESS accepts billing change. PBX notifies adjunct			Release Complete Return Result
Call Sequence 3:			
Incoming ISDN-PRI MultiQuest call without Flexible Billing			
Adjunct Route Request	Route Request call_id=1112 called number=9085551234 domain=VDN A calling number=CPN/BN Flexible Billing is disabled		
Adjunct replies to Route Request	Route Select called number=Split2		

(Continued on next page)

Action/Operation	Route Request	Event Notification Split 1	Set Value (billing Change)
Call is offered to split 2		Call Offered call_id=1112 called number=Split1 domain=Split1 calling number=CPN/BN Flexible billing is disabled	
Call rings at agent		Alerting call_id=1112 called number=9085551234 calling number=CPN/BN	
Agent answers call		Connected call_id=1112 called number=9085551234 calling number=CPN/BN connected number=5001 party_id=2	

Advice of Charge

Switch-Classified Call Receives Charging Information During the Call

This scenario deals with a predictive call that is directed off DEFINITY ECS to a German 1TR6 network. The switch-classified call options, answering machine detection and call classification after answer, are not enabled while CDR call splitting is enabled. CDR is administered to report the group extension on calls to hunt groups. The predictive call is placed on behalf of a hunt group (A2).

Action/Operation	Notification Association (Trunk Group)	Make Call Association	Comments
Adjunct requests Event Notification on all trunk groups for ISDN AOC.	*Event Notification Request* Domain type=TAC Domain Value=#####		This request may be made once after link initialization.
ECS accepts request.	*Return Result=ACK*		
Adjunct initiates predictive call. (The network is German National ISDN 1TR6.)		*3P Make_Call* dest_addr A1 orig_addr A2 alert_dest_first	AMD is not enabled for this call.
Call alerts the far-end.		*Alerting Event* Call_id C1 Party_id 2 calling_number A2 called_number A1 connected_number #####[1]	The Alerting event is generated from the ISDN-PRI network and may not be provided over ASAI.
Far end answers. Answer supervision received.		*Answered Event* Call_id C2 Party_id 2 called_number A1 connected_number #####	
First AOC update received from ISDN INFORMATION message. *A charge of 23 is received.*			Note: The charging event report is not provided since the second leg of the call has not been merged with the first leg.
Second leg of the switch-classified call is set up to hunt group A2. Agent at ext. D1 is available. ECS merges first and second legs of the call.		*Alerting Event* Call_id C2 Party_id 2 calling_number A1 called_number A2 connected number D1	Agent extension D1 begins alerting. No split charge is output as the result of this merge.

1. In Germany, a 1TR6 network will provide connected number to the originator of the call.

(Continued on next page)

Action/Operation	Notification Association (Trunk Group)	Make Call Association	Comments
D1 answers.		*Connected Event* Call_id C2 Party_id 2 calling_number A1 called_number A2 connected_number D1	Agent D1 and far-end begin conversation.
Second AOC update arrives from the network. A charge of 46 is received.	*Charging Event* call_id C2 Party_id 2 Charging_number A2 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=intermediate charge value=46		ECS will send the charge value received in the second update from the network. Notes: 1. ECS may send a Charging Event Report before any other event report is sent that references the second leg of the call. 2. Called number is not consistent with the called number in call-control events.
Adjunct releases call.		*3P Selective Drop* Party_id 2	
DISCONNECT sent to network.		*Return Result=ACK*	
RELEASE received from the network with final charge. A charge of 69 is received.	*Charging Event* Call_id C2 Party_id 2 Charging_number A2 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=final charge value=69		Final charge is reported. Application may decide to distribute charges to D1's department.
ECS clears call record.		*3P Call Ended* Call_id C2 cause=normal clearing	

User-Classified Call, Charge Information

The following scenario involves a user-classified call placed over a trunk facility that connects to a European Telecommunications Standards Institute (ETSI) network. The ETSI network provides Advice of Charge (AOC-D) during the call.

Action/Operation	Notification Association (Trunk Group)	Make Call Association	Comments
Adjunct initiates user-classified call to German ETSI network.		*3P Make_Call* dest_addr A1 orig_addr 2 return_ack=yes	
ECS accepts and acknowledges.		*3P Make Call Proceed* Party id 1	
Station D1 goes off hook.			
ECS recognizes dialed number as an address.			Call is routed to an ISDN Trunk Group providing AOC-D.
Far end answers.		*Connected Event* Call_id C1 Party_id 2 trunk group_id T1 called_number ##### connected_number #####	From ISDN CONNECT ? <**Dan, see p22of RFS, anything missing?>** Note: Calling and called number information not available from network.
ECS receives first charge update from ISDN FACility message.	*Charging Event* call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=intermediate charge value=156		Application accumulates charge towards station D1's department.
ECS receives second charge update from ISDN FACility message.	*Charging Event* call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=intermediate charge value=179		Application accumulates charge and determines station D1's department is over-budget.

(Continued on next page)

Action/Operation	Notification Association (Trunk Group)	Make Call Association	Comments
Adjunct releases call.		*3P Clear Call*	Application takes action to correct over-budget problem.
DISCONNECT sent to network.			
RELEASE received from network *without* final charge.	*Charging Event* call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=final charge value=179 cause=CS3/38		ECS indicates that charge information is not received in first clearing message from the network and includes the last valid amount received as the final charge.
ECS acknowledges 3P Clear Call Request.		*Return Result=ACK*	

Conference Call in Progress, Multiple Outgoing Trunks

The following scenario represents a conference call involving multiple internal parties and multiple outgoing trunks receiving charge advice from the network. In this scenario, the CDR Call Splitting option is enabled, as well as the trunk-to-trunk transfer option. The adjunct application can use the charging number information provided within Charging Event Reports as a means of distributing the cost of the call. Alternatively, the application can use the Call_id reference provided with the Charging Event Report and match it with the Call_id of other ASAI associations in order to divide the charges according to the time spent on the conference by each participant. The network in this example is France VN4.

Before the events shown in the table, Station D1 added the two trunks to call C2, and called station D1 on call C1. Call C2 is on hold, about to be conferenced with call C1. The parties on call C2 are: Party ID 1=Station D1, Party ID 2 = called number A1 on trunk T1 (group 15, member 1), Party ID 3 = called number A2 on trunk T2 (group 600, member 31). The Party IDs for call C1 are Party ID 1 = Station D1, Party ID 2 = Station D2.

Action/ Operation	Event Notification Association (Trunk Group)	Domain Control D1	Domain Control D2	Comments
Adjunct requests Domain (station) control on D1 & D2.		*3P Domain Control Request* Domain D1	*3P Domain Control Request* Domain D2	Adjunct requests control over stations D1 and D2.
ECS accepts and acknowledges.		*Return Result = ACK* Call_id C1, Party ID 1 Call_ id C2, Party id 1 call_state=held call_state=connected	*Return Result = ACK* Call_id C1, Party ID 2 call_state=connected	The ACK notifiies the adjunct that multiple calls are already in progress.
ECS receives charge update from ISDN-PRI INFORMATION message. For Party_id 2, charge value 78659.	*Charging Event* call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=intermediate charge value=78659			Station D1 conferenced an outgoing trunk (15/1) earlier and is considered the controlling number.
Station D1 presses conference button.				

(Continued on next page)

Action/ Operation	Event Notification Association (Trunk Group)	Domain Control D1	Domain Control D2	Comments
ECS completes the conference.		*Call Conferenced Event* Other_call_id C1 Resulting _Call_id C2 Party_id 1 Party_id 2 Party_id 3 Party_id 4 old_party_id=reslt call 1 old_party_id=reslt call 2 old_party_id=reslt call 3 old_party_id=othercall 2 connected_number D1 connected _number ##### connected_number ##### connected_number D2 connected_number ##### calling_number D1 called_number A1	*Call Conferenced Event* Other_call_id C1 Resulting _Call_id C2 Party_id 1 Party_id 2 Party_id 3 Party_id 4 old_party_id=reslt call 1 old_party_id=reslt call 2 old_party_id=reslt call 3 old_party_id=othercall 2 connected_number D1 connected _number ##### connected_number ##### connected_number D2 connected_number ##### calling_number D1 called_number A1	4-party conference is established. Party ids 2 and 3 are trunks receiving charge advice that D1 conferenced earlier into Call_ id C2. Charging Events for call C2 have already been sent and logged by the application.
ECS receives charge update from ISDN-PRI INFORMATION message. For Party_id 2, charge value 80214.	*Charging Event* Call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Type of charge=intermediate charge value=80214			
ECS receives charge update from ISDN -PRI INFORMATION message. For Party_id 3, charge value 420.	*Charging Event* Call_id C2 Party_id 3 Charging_number D1 Called_number A2 Trunk Group=600 Trunk Member=31 Type of charge=intermediate charge value=420			Since D1 originally added this trunk (600/31), D1 is considered the charging number.
Station D1 drops.		*Drop Event* Call_id C2 Party_id 1	*Drop Event* Call_id C2 Party_id 1	Station D1 is no longer on the call. Even though CDR Call Splitting is enabled, D1 continues to receive charges. This is because the call still has three or more parties. No split charge is output at this time.

(Continued on next page)

Action/ Operation	Event Notification Association (Trunk Group)	Domain Control D1	Domain Control D2	Comments
ECS receives update from ISDN-PRI INFORMATION message, For Party_id 3, charge value 440.	*Charging Event* Call_id C2 Party_id 3 Charging_number D1 Called_number A2 Trunk Group=600 Trunk Member=31 Type of charge=intermediate charge value=440			D1 is charging number.
ISDN-PRI DISCONNECT received with charge value 84768. (Party_ id drops from call.)			*Drop Event* Call_id C2 Party_id 2	Party_id 2 drops. Now the call is split, since there are only two parties left.
	Charging Event Call_id C2 Party_id 2 Charging_number D1 Called_number A1 Trunk Group=15 Trunk Member=1 Type of charge=final charge value=84768			ECS sends final charge associated with party_id 2. This charge goes to station D1.
	Charging Event Call_id C2 Charging_number D1 Called_number A2 Trunk Group=600 Trunk Member=31 Type of charge=split charge value=440			ECS sends split charge. This charge goes to station D1. The party ID is not sent when CDR Call Splitting generates a split charge event.
ECS receives charge update from ISDN-PRI INFORMATION message for Party_id 3, charge value 460.	*Charging Event* Call_id C2 party _id 3 Charging_number D2 Called_number A2 Trunk Group=600 Trunk Member=31 Type of charge=intermediate charge value=20			Call splitting has left D2 as the charging party. Amount of charge is adjusted downward by the amount charged to D1 in the split charge event.

World-Class Routing (ARS/AAR), Incoming Call Routed over Outgoing ISDN Trunk Group, Charge Information Provided during the Call

The following scenario describes a single case in which event reports are received on the Charge of Advice Notification Association and not on any other notification, call-control, or domain-control association. In particular, this scenario involves an incoming call that is processed in the ISDN-PRI Trunk Group's Incoming Call Handling Treatment Table and is subsequently routed over another ISDN trunk group receiving Charge Advice.

Action/Operation	Notification Association (Trunk Group)	Comments
Incoming call on an ISDN trunk group is processed using the Incoming Call Handling Treatment Table.		
An entry in the incoming Call Handling Treatment Table matches the service of the incoming call NSF IE. The called number is deleted and a new number is inserted.		The new routing number for the call is processed using ARS analysis. ECS determines the routing pattern and selects the outgoing trunk group based on preference and trunk availability.
The call is routed as an outgoing call to an ISDN trunk group that is enabled for Charge Advice during the call.		
The far end answers.		*Note: Application will not receive notification of answer on far end.*
The network sends the first charge advice message.	*Charging Event* call_id C2 Party_id 2 Charging_number=T1 Called_number A1 Trunk Group=45 Trunk Member=5 Type of charge=intermediate charge value=407	The charging number, T1, is the Trunk Access Code (TAC) of the incoming trunk. This indicates that no local extension is on the call, and the ISDN-PRI calling number is unavailable.
The network sends the final charge advice message in RELEASE.	*Charging Event* call_id C2 Party_id 2 Charging_number=T1 Called_number A1 Trunk Group=45 Trunk Member=5 Type of charge=final charge value=679	*Note: Application will not receive notification that the call is complete, other than the final charge associated with the call.*

22. Miscellaneous Cases

This section presents unsuccessful ASAI capability requests including common error cases.

Unsuccessful Requests for Domain (Station) Control

This scenario shows an ISDN PRI incoming call to station 85046. The adjunct processor requests to drop an alerting call, reconnect to an alerting call, and answer a non-existent call. Station 85046 is domain-controlled by the adjunct processor

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=123, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 85046)]		AP Requests Domain Control of Station 85046
	FAC [CRV=123, FIE(Return Result, Inv_id=1, 3P Domain Control, call_id=45, party_id=2, call_state=alerting)]	Domain Control Granted for Station 85046 (One Call Alerting)
FAC [CRV=123, FIE(Invoke, Inv_id=3, 3P Selective Drop, call_id=45)]		AP Requests to Drop Station 85046
	FAC [CRV=123, FIE(Return Error, Inv_id=3, 3P Selective Drop, cause=request not compatible with call state)]	Request Denied (Call in the Alerting State)
FAC [CRV=123, FIE(Invoke, Inv_id=5, 3P Reconnect, call_id=45)]		AP Requests to Connect to Call
	FAC [CRV=123, FIE(Return Error, Inv_id=5, 3P Reconnect, cause=request not compatible with call state)]	Request Denied (Call in the Alerting State)
	FAC [CRV=123, FIE(Invoke, Inv_id=2, Event Report, call_id=45, event=call redirected)]	Call Redirection Event Report (Call Goes to Coverage)

(Continued on next page)

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
FAC [CRV=123, FIE(Invoke, Inv_id=7, 3P Answer, call_id=45)]		AP Requests to Connect to Call
	FAC [CRV=123, FIE(Return Error, Inv_id=7, 3P Answer, cause=invalid number)]	Request Denied (Call not Present)

ISDN Network Congestion

This scenario shows a call from station 67 to an external destination (75602) using ISDN PRI facilities. The network disconnects the call with a cause (network congestion). The adjunct processor requests **Domain_Control** for station 67.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=13, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 67)]		AP Requests Domain Control of Station 67
	FAC [CRV=13, FIE(Return Result, Inv_id=1)]	Domain Control Granted for Station 67 (No Calls Present)
FAC [CRV=13, FIE(Invoke, Inv_id=3, 3P Auto Dial, called number=75602)]		AP Requests Auto Dial Call to Station 75602
	FAC [CRV=13, FIE(Invoke, Inv_id=2, Event Report, call_id=76, party_id=1 event=call initiated)]	Call Initiated Event Report (Station 67 Goes Off-Hook)
	FAC [CRV=13, FIE(Invoke, Inv_id=4, Event Report, call_id=76, party_id=1, calling number=67, called number=75602, event=call originated)]	Call originated
	FAC [CRV=13, FIE(Invoke, Inv_id=6, Event Report, call_id=76, party_id=2, event=drop, connected number=#####, cause=network congestion)]	Drop Event Report (Network Disconnects)
	FAC [CRV=13, FIE(Invoke, Inv_id=8, Event Report, call_id=76, party_id=1, event=drop, connected number=67, cause=normal clearing)]	Drop Event Report Station 67 Disconnects (Call Terminates)

Invalid Call Destination— Reorder

This scenario shows a call initiated via the **Third_Party_Make_Call** capability from station 5067 to an invalid destination (46379). Station 5067 receives reorder tone. The adjunct processor requests **Domain_Control** for station 5067.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=86, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 5067)]		AP Requests Domain Control of Station 5067
	FAC [CRV=86, FIE(Return Result, Inv_id=1)]	Domain Control Granted for Station 5067 (No Calls Present)
REG [CRV=122, FIE(Invoke, Inv_id=1, 3P Make Call, called number=46379, calling number=5067)]		AP Requests Make Call from Station 5067 to Station 46379
	FAC [CRV=86, FIE(Invoke, Inv_id=2, Event Report, call_id=53, party_id=1, event=call initiated)]	Call Initiated Event Report (x5067 Goes Off-Hook) — x5067 Association
	FAC [CRV=86, FIE(Invoke, Inv_id=4, Event Report, call_id=53, event=reorder/denial, called number=*****, cause=invalid number)]	Reorder/Denial Event Report (Invalid Destination) — x5067 Association
	FAC [CRV=122, FIE(Invoke, Inv_id=2, Event Report, call_id=53, event=reorder/denial, called number=*****, cause=invalid number)]	Reorder/Denial Event Report (Invalid Destination) — 3P Make Call Association
	FAC [CRV=123, FIE(Invoke Inv_id=6, Event Report call_id=53, party_id=1, event=drop, connected number=5067, cause=normal clearing)]	Drop Event Report (Station 5067 Disconnects) — x5067 Association
	REL COMP [CRV=122, FIE (Invoke,Inv_id=4, Call Ended,call_id=53, cause=normal clearing)]	Call Terminates — 3P Make Call Association

User Does Not Go Off-hook

This scenario shows a **Third_Party_Make_Call** request for which the originating station (85046) does not go off-hook within five seconds.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=12, FIE(Invoke, Inv_id=3, 3P Make Call, calling number=85046, called number=75602)]		AP Requests Make Call from Station 85046 to Station 75602
	REL COMP [CRV=12, FIE(Return Error, Inv_id=3, 3P Make Call, cause=user not responding)]	Request Denied Station 85046 Does Not Go Off-Hook

Extension Removed

This scenario shows the message sent by the switch when a domain-control station (station 74567) is removed by the switch administrator and control is terminated.

Messages Sent by theAdjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=124, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=extension 74567)]		AP Requests Domain Control of Station 74567
	FAC [CRV=124, FIE(Return Result, Inv_id=1)]	Domain Control Granted for Station 74567 (No Calls Present)
	REL COMP [CRV=124, FIE(Invoke, Inv_id=20, 3P Domain Control Ended, cause=invalid number)]	Domain Control Terminated (x74567 Has Been Removed)

Invalid Association Requests

This scenario shows an adjunct processor requesting a **Third_Party_Auto_Dia**l and **Third_Party_Clear_Call** over an invalid association.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=124, FIE(Invoke, Inv_id=1, 3P Auto Dial, called number=35602)]		AP Requests Auto Dial Call to Station 35602
	REL COMP [CRV=124, FIE(Invoke, Inv_id=2, Abort, cause=protocol error)]	Association Aborted (Not an Initiating Capability)
REG [CRV=123, FIE(Invoke, Inv_id=1, 3P Domain Control, domain=Station 456)]		AP Requests Domain Control of Station 456
	FAC [CRV=123, FIE(Invoke, Inv_id=1)]	Domain Control Granted No Calls Present
	FAC [CRV=123, FIE(Invoke, Inv_id=2, Event Report, call_id=78, party_id=2, event=alerting, calling number=20145366397, called number=9085761456, connected number=456)]	Alerting Event Report (Another Call Delivered to Station 456)
FAC [CRV=123, FIE(Invoke, Inv_id=7, 3P Clear Call, call_id=78)]		AP Requests to Clear the Call
	REL COMP [CRV=123, FIE(Invoke, Inv_id=6, ABORT, cause=protocol error)]	Association Aborted (Capability not Supported)

Invalid Call_id Numbers

This section shows an adjunct processor requesting **Third_Party_Take_Control** and **Value_Query** (Party_id Query) for a non-existent call.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=122, FIE(Invoke, Inv_id=1, 3P Take Control, call_id=53)]		AP Requests Control of Call
	REL COMP [CRV=122, FIE(Return Error, Inv_id=1, 3P Take Control, cause=call cleared)]	Request Denied (Call 53 does not Exist)
REG [CRV=121, FIE(Invoke, Inv_id=1, Value Query, item=Party_id, call_id=53)]		AP Requests the Parties Connected to Call 53
	REL COMP [CRV=121, FIE(Return Error, Inv_id=1, Value Query, cause=call cleared)]	Request Denied (Call 53 does not Exist)

Invalid Station Numbers

This section shows an adjunct processor requesting **Domain_Control** and **Value_Query** (Calls Query) for a VDN extension.

Messages Sent by the Adjunct Processor	Messages Sent by the Switch	Comments
REG [CRV=121, FIE(Invoke, Inv_id=1, Value Query, item=Calls Query, domain=extension 88766)]		AP Requests Status of Calls at x85046
	REL COMP [CRV=121, FIE(Return Error, Inv_id=1, Value Query cause=invalid number)]	Request Denied x88766 not a Valid Extension
REG [CRV=122, FIE(invoke, Inv_id=1, 3P Domain Control, domain=extension 83567)]		AP Requests Domain Control of Station 83567
	REL COMP [CRV=122, FIE(Return Error, Inv_id=1, 3P Domain Control, cause=invalid number)]	Domain Control Denied (Invalid Extension)
REG [CRV=122, FIE(Invoke, Inv_id=1, Value Query, item=Extension, calling number=83567)]		AP Requests Extension Query
	REL COMP [CRV=122, FIE(Return Result, Inv_id=1, Value Query, domain=VDN)]	Extension 83567 is a VDN

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

V

W