INTERNATIONAL TELECOMMUNICATION UNION

ITU-T H.450.11

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(03/2001)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Supplementary services for multimedia

Call intrusion supplementary service for H.323

ITU-T Recommendation H.450.11

(Formerly CCITT Recommendation)

ITU-T H-SERIES RECOMMENDATIONS

AUDIOVISUAL AND MULTIMEDIA SYSTEMS

CHARACTERISTICS OF VISUAL TELEPHONE SYSTEMS	H.100–H.199
INFRASTRUCTURE OF AUDIOVISUAL SERVICES	
General	H.200–H.219
Transmission multiplexing and synchronization	H.220–H.229
Systems aspects	H.230–H.239
Communication procedures	H.240–H.259
Coding of moving video	H.260–H.279
Related systems aspects	H.280–H.299
SYSTEMS AND TERMINAL EQUIPMENT FOR AUDIOVISUAL SERVICES	H.300–H.399
SUPPLEMENTARY SERVICES FOR MULTIMEDIA	**H.450–H.499**

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation H.450.11

Call intrusion supplementary service for H.323

Summary

This Supplementary Service describes the procedures and the signalling protocol for the Call Intrusion supplementary service in H.323 (Packet Based Multimedia Communications Systems) networks.

The Call Intrusion supplementary service (SS-CI) enables a calling user A, encountering a busy destination user B, to establish communication with user B by breaking into an established call between user B and a third user C.

This Recommendation makes use of the "Generic functional protocol for the support of supplementary services in H.323" as defined in ITU-T H.450.1.

This Recommendation requires H.323 version 2 (1998) or later. Version 2 products can be identified by H.225.0 messages containing a **protocolIdentifier** = {itu-t (0) recommendation (0) h (8) 2250 version (0) 2} and H.245 messages containing a **protocolIdentifier** = {itu-t (0) recommendation (0) h (8) 245 version (0) x}, where "x" is 3 or higher.

The procedures and the signalling protocol of this Recommendation are derived from the Call Intrusion supplementary service specified in ISO/IEC 14845 and ISO/IEC 14846.

Source

ITU-T Recommendation H.450.11 was prepared by ITU-T Study Group 16 (2001-2004) and approved under the WTSA Resolution 1 procedure on 1 March 2001.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

ITU-T Recommendation H.450.11

Call intrusion supplementary service for H.323

1 Scope

This Recommendation specifies the Call Intrusion supplementary service (SS-CI), which is applicable to various basic services supported by H.323 multimedia endpoints.

Call Intrusion (SS-CI) is a supplementary service which, on request from the served user, enables the served user to establish communication with a busy called user (user B) by breaking into an established call between user B and a third user (user C).

The service description, the procedures and the signalling protocol of this Recommendation are derived from the Call Intrusion supplementary service as specified in ISO/IEC 14845 and 14846.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

- ITU-T H.225.0 (2000), *Call signalling protocols and media stream packetization for packet-based multimedia communication systems*.
- ITU-T H.245 (2000), *Control protocol for multimedia communication*.
- ITU-T H.248 (2000), *Gateway control protocol*.
- ITU-T H.323 (2000), *Packet-based multimedia communications systems*.
- ITU-T H.450.1 (1998), *Generic functional protocol for the support of supplementary services in H.323*.
- ITU-T H.450.2 (1998), *Call transfer supplementary service for H.323*.
- ITU-T H.450.3 (1998), *Call diversion supplementary service for H.323*.
- ITU-T H.450.4 (1999), *Call hold supplementary service for H.323*.
- ITU-T H.450.6 (1999), *Call waiting supplementary service for H.323*.
- ITU-T H.450.10 (2001), *Call offer supplementary service for H.323*.

3 Terms and definitions

This Recommendation defines the following terms:

3.1 busy, busy condition: A condition where a destination endpoint engaged in one or more calls cannot accept another incoming call due to resource limitations.

NOTE – In the absence of any supplementary services that might modify the behaviour, the endpoint will in this situation send a Release Complete message containing a ReleaseCompleteReason of inConf or a Cause IE with cause value #17, "user busy"; an H.323 endpoint may be busy with one call or may be busy with more than one call depending on implementation.

3.2 **call**: Refer to ITU-T H.323.

3.3 **common information**: Supplementary service capability information which may be exchanged between H.323 endpoints/entities at call establishment time or during a call.

3.4 **conference type connection**: A connection between the served user, user B and user C, where all users have user information connection with each other.

3.5 **delayed invocation**: Invocation of SS-CI after the calling user has been informed that a call has failed because of busy at the destination.

3.6 **endpoint, gatekeeper, gateway, terminal, user**: See ITU-T H.323.

3.7 **established call**: The active call that is selected for intruding on.

3.8 **forced release**: The release of the established call on request from the served user.

3.9 **immediate invocation**: Invocation of SS-CI as part of the initial call setup.

3.10 **intruding call**: A call in which the served user requests SS-CI.

3.11 **intrusion state**: The condition after establishment of communication between the served user and user B using SS-CI and prior to termination of SS-CI.

3.12 **isolation, Held type of connection**: The breaking of the user information connection to and from user C during the intrusion state by means of isolating user C.

NOTE – For example, user C may be held using the procedures of ITU-T H.450.4.

3.13 **proxy**: An entity that acts on behalf of an endpoint for the SS-CI procedures. The proxy may or may not be co-located with the gatekeeper.

3.14 **served user, user A**: The user who requests SS-CI (**calling user**).

3.15 **silent monitoring type of connection**: A connection between the served user, user B and user C, similar to a conference type of connection but with the served user monitoring the established call without users B and C being informed about that fact.

3.16 **user B**: The wanted user that is subject to the call intrusion (**called user**).

3.17 **user C**: The other user in the established call, also referred to as the **unwanted user**.

3.18 **wait-on-busy**: A condition in which the intruding call is disconnected from user B and is waiting for user B to become not busy.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations:

APDU Application Protocol Data Unit

ASN.1 Abstract Syntax Notation One

CICL Call Intrusion Capability Level

CIPL Call Intrusion Protection Level

EASE Endpoint A Signalling Entity

EBSE Endpoint B Signalling Entity

ECSE Endpoint C Signalling Entity

GK Gatekeeper

IE Information Element

MC Multipoint Control

MP Multipoint Processor

NFE Network Facility Extension

SCN Switched Circuit Network

SDL Specification and Description Language

SS-CI Supplementary Service Call Intrusion

SS-CO Supplementary Service Call Offer

WOB Wait On Busy

5 SS-CI service description

Call Intrusion (SS-CI) is a supplementary service which, on request from the served user, enables the served user to establish communication with a busy called user (user B) by breaking into an established call between user B and a third user (user C).

On successful intrusion, user C is either connected in a held type of connection, connected in a conference type connection, connected in a silent monitoring type of connection, or user C is force-released.

Upon SS-CI request, if no specific option is requested, either the held type or the conference type of SS-CI is invoked depending on the implementation options supported within user B's endpoint.

As an option, the forced release type of SS-CI may be requested by a served user that is authorized appropriately, either within the initial call setup or after the held type or conference type of SS-CI have been invoked successfully.

As an option, the silent monitoring type of SS-CI may be requested by a served user that is authorized appropriately.

5.1 Implementation options

- *Conference type of connection*

  Upon successful invocation of SS-CI, the served user (user A), user B and user C are merged into a conference type of connection.

  As an option, users B and C may be provided with a call intrusion warning notification and/or a warning tone for a short period of time before the conference type of connection is established. The notification may also be sent to user A.

  NOTE 1 – An MC (and occasionally also MP) type of functionality is required within endpoint B or within a proxy acting on behalf of endpoint B.

- *Held type of connection*

  Upon successful invocation of SS-CI the unwanted user C is split from user B. For this reason, user B's endpoint shall automatically invoke a suitable service – e.g. the Call Hold Supplementary Service – against user C prior to establishment of communication between user B and the served user.

  In addition, user C may be provided with an indication that SS-CI has been invoked prior to or together with the call hold notification.

  As an option a served user may be able to request transition from the conference type to the held type of SS-CI during the intrusion state.

- *Silent monitoring type of connection*

  As an option – if the served user is provided with the capability – a served user may request the silent monitoring type of SS-CI against a busy user B.

After successful invocation of silent monitoring, the served user can listen to (i.e. monitor) the established call. Users B and C are not informed about SS-CI taking place.

NOTE 2 – An MC (and probably also MP) type of functionality is required within endpoint B or within a proxy acting on behalf of endpoint B.

NOTE 3 – This feature is intended for business applications like call centres. It is not intended as a tool for lawful interception.

- *Forced release*

  As an option – if the served user is provided with the capability – a served user may request forced release of the established call at the busy called user B.

  Forced release may either be invoked initially or may be invoked after the conference type or held type of SS-CI has been invoked successfully.

- *Wait on busy*

  As an option a served user may be able to request transition from the intrusion state (held type or conference type of SS-CI) to a wait-on-busy state, and vice versa.

Endpoint B shall support at least one of the options conference type or held type of SS-CI and may additionally support the options silent monitoring, forced release and wait on busy.

5.2 Provision of capability and protection levels

A Call Intrusion Capability Level (CICL) shall be allocated to the served user. CICL shall have a value in the range 1 (lowest capability) to 3 (highest capability). At least one of the CICL values shall be supported.

Call Intrusion Protection Levels (CIPLs) shall be allocated to potential individual users B and C. CIPL shall have a value in the range 0 (no protection) to 3 (total protection). CIPL values 0 and 3 shall be supported; values 1 and 2 may additionally be supported.

NOTE – Variable CIPL values are not precluded, e.g. a user may have the possibility to change the CIPL value. CIPL values assigned to gateways may also be variable, e.g. depending on whether the gateway is used for an incoming or outgoing call.

If endpoint C does not support SS-CI (and user C has therefore no explicit CIPL allocated) a default CIPL shall be assumed for user C, e.g. 0 (no protection).

The procedure by which CICL and CIPL are allocated is outside the scope of this Recommendation.

5.3 Normal procedures

5.3.1 Activation/deactivation/registration/interrogation

SS-CI is permanently activated.

Registration and interrogation are not applicable.

5.3.2 Invocation and operation

There are two different ways to invoke SS-CI. At least one of the following methods shall be supported:

– Delayed invocation: the served user, on being informed that a call has failed because of busy at the destination, shall be able, within a defined period, to request SS-CI.

– Immediate invocation: the served user shall be able to request SS-CI as part of the initial call setup.

Regardless of the type of SS-CI invocation, an intrusion request is only accepted if the served user has a higher Call Intrusion Capability Level (CICL) than the Call Intrusion Protection Level (CIPL)

of both user B and user C. If user B has several calls in the active state, one of them (with sufficiently low CIPL) shall be selected as the established call for intrusion.

An Impending intrusion warning (e.g. notification, in-band tone or announcement) may be provided to the users in the established call (and optionally also to the served user), and a short delay (not exceeding 10 s) may apply before the connection between the served user and user B is formed.

Depending on which implementation option is chosen, the served user is connected to user B:

- by forming a conference between user B, user C and the served user; in this case the users in the established call shall be provided with a notification when the served user is connected, and the served user shall receive confirmation that the intrusion request has been accepted and that a conference type connection has been formed;

  NOTE 1 – The three users can also receive a superimposed in-band indication (e.g. a repeated tone) while the conference type connection exists.

- by isolating (e.g. putting on hold) user C and connecting the served user only to user B; in this case user C shall be given a notification that isolation has occurred, user B shall be informed that user C has been isolated and that an intrusion has occurred, and the served user shall receive confirmation that the intrusion request has been accepted and that isolation has occurred;

  NOTE 2 – User C can also receive an in-band tone or announcement while isolated.

- by forming a "silent monitoring" conference between user B, user C and the served user; in this case the users in the established call shall not be provided with any intrusion notification, but the served user shall receive confirmation that the intrusion request has been accepted and that a silent monitoring conference type connection has been formed;

- by forced release of the established call; in this case, the established call shall be released with a notification given to user C, user B shall be informed that user C has been released and that an intrusion has occurred, the served user shall receive confirmation that the intrusion request has been accepted, and the intruding call shall continue as an ordinary call between the served user and user B.

If the relevant options are supported, the served user may request isolation or forced release of user C also out of a conference type intrusion, and the served user may request forced release also after user C has been isolated. For the provision of notifications in these cases the statements above shall apply accordingly.

If the wait-on-busy option is supported, the served user may request wait on busy out of a conference type intrusion or after user C has been isolated. On acceptance of the request, the served user shall receive a confirmation, the established call shall revert back to the state that existed before intrusion, user B shall be reconnected to user C if isolated, the intruding call shall be disconnected from user B and the intrusion state shall be terminated. The intruding call shall not be released but shall be treated as a waiting call. User B and user C shall be notified that intrusion has terminated.

If the served user releases the intruding call, SS-CI shall be terminated, and:

- in the case of a conference type connection, the established call shall continue between user B and user C, and both users shall be notified that intrusion has terminated;

- in the case of isolation of user C, users B and C shall be reconnected, and both users shall be notified that intrusion has terminated;

- in the silent monitoring case, the established call shall continue between user B and user C without any notifications given.

If either user B or user C releases the established call and silent monitoring does not apply, the served user – and also user B if user C initiated the release – shall be notified that intrusion is no longer applicable. SS-CI shall be terminated and the call from the served user to user B shall continue according to basic call procedures.

If either user B or user C releases the established call and silent monitoring applies, the intruding call shall also be released.

If during the intrusion state user B releases the intruding call, the established call shall revert to the state that existed before intrusion, and user C shall be notified that the intrusion state has terminated. In case of silent monitoring, user B cannot release the intruding call.

5.3.3 Exceptional procedures

5.3.3.1 Activation/deactivation/registration/interrogation

Not applicable.

5.3.3.2 Invocation and operation

If the served user requests invocation of SS-CI as part of the initial call request, and immediate invocation is not provided to the served user, then the request shall be ignored and the call shall proceed as if the request had not been made.

If an SS-CI request is rejected, the served user shall be informed, and may be given an indication of the reason for the rejection. Possible reasons to reject an SS-CI request are for instance:

- served user has a lower or equal CICL compared with user B's and/or user C's CIPL value;
- user B is busy but not involved in a compatible call in the active state;
- temporary lack of resources;
- the established call is already being intruded upon;
- the established call is intruding on another call.

If SS-CI is requested and user B is found to be not busy, the call shall be treated as a normal incoming call to user B.

If a forced release, isolate or wait on busy request fails while intrusion is in progress, the served user shall be notified and may be given an indication of the reason for failure, and intrusion shall continue.

Delayed invocation is not possible if the defined time period has expired or if the called user is busy but intrusion is not allowed. Basic call procedures shall apply.

5.4 Interactions with other supplementary services

5.4.1 Call Transfer (SS-CT)

The served user shall not be able to invoke SS-CT while intrusion is in progress but may transfer the intruding call during wait-on-busy. Transfer during wait-on-busy shall operate similarly to Call Transfer during the alerting state, except that the call shall continue as a waiting call. The transferred user may be notified that the call is in a waiting condition against user B, and also that the waiting condition has ceased if user B subsequently enters an alerting phase. The transferred user may subsequently invoke SS-CI if authorized by a sufficiently high CICL.

During the intrusion state, user B shall not be able to transfer an established or intruding call, except for silent monitoring, where user B may transfer the established call. This shall result in termination of silent monitoring and the release of the intruding call.

User C may be able to transfer an established call during intrusion. If transfer occurs during silent monitoring, SS-CI shall be terminated and the intruding call shall be released. In all other cases the transferred-to user that becomes connected to user B shall become the new user C if the CIPL of this user allows intrusion by user A, and shall receive all notifications accordingly. If the new user's CIPL is too high SS-CI shall be terminated and the intruding call shall be released. In case of call transfer without secondary call, the transferred user may initiate SS-CI invocation if the transferred-to user is found to be busy.

5.4.2 Call Forwarding Unconditional (SS-CFU)

SS-CI, if invoked, shall operate on a busy user that has been reached as a result of one or more invocations of SS-CFU, provided neither SS-CFNR nor Call Deflection has taken place.

5.4.3 Call Forwarding Busy (SS-CFB)

If the called user is busy and has SS-CFB active, by default SS-CFB shall take precedence over an SS-CI request. In this case SS-CI shall operate on the final diverted-to user if that user is busy, provided neither SS-CFNR nor Call Deflection has taken place.

Alternatively, on explicit request, SS-CI may operate on the first SS-CFB forwarding user.

An implementation may permit the calling user to make the choice between these two alternatives.

5.4.4 Call Forwarding on No Reply (SS-CFNR)/Call Deflection (SS-CD)

SS-CI if invoked, shall not operate on a busy user arrived at as a result of one or more diversions, at least one of which is SS-CFNR or SS-CD. The procedures of SS-CFNR/SS-CD shall apply.

5.4.5 Call Hold

The intruding call shall not be put on hold.

NOTE – Call Hold may be invoked in the course of intrusion to achieve isolation of user C.

5.4.6 Call Park/Call Pickup

It shall not be possible to park the intruding call.

It shall not be possible to pick up the intruding call while intrusion is impending.

5.4.7 Call Waiting

If SS-CI is invoked for a call SS-CW does not apply (i.e. intrusion takes precedence over call waiting).

The served user may be able to invoke SS-CI for a waiting call.

5.4.8 Message Waiting Indication

No interaction.

5.4.9 Name Presentation

No interaction.

5.4.10 Completion of Calls to Busy Subscriber (SS-CCBS)

During intrusion, the served user may be able to invoke SS-CCBS against user B.

5.4.11 Completion of Calls on No Reply (SS-CCNR)

No interaction.

5.4.12 Call Offer (SS-CO)

An SS-CI request made after an SS-CO request has been accepted shall be allowed. If the request is rejected due to intrusion not allowed, SS-CO shall remain in progress. If SS-CI is accepted, the SS-CO request shall be cancelled.

If both SS-CI immediate invocation and SS-CO immediate invocation are requested at call setup, the action is an implementation option.

NOTE – Possibilities include for example:

– unconditionally accept one service request and ignore or reject the other;

– accept one service request and ignore or reject the other based on certain conditions (e.g. call type, destination endpoint type);

– try one service first and if not successful, the other;

– reject both service requests and proceed with the call as if neither of the services had been requested.

If the served user is provided with SS-CO network invocation (immediate) and the served user requests SS-CI immediate invocation, in the absence of other implementation specific rules the network shall not invoke SS-CO. (For some possible other actions see Note above.)

5.4.13 Common information

By exchange of Common Information data the served user may have *a priori* knowledge of the SS-CI capabilities at the called endpoint, e.g. the CIPL of user B.

By exchange of Common Information data the CIPL of potential unwanted users may be available at the called endpoint prior to invocation of SS-CI.

5.4.14 Call Linkage

In case of delayed invocation of SS-CI, the Thread ID of the original call attempt (i.e. the call that failed due to user B busy), if available, shall also be used for the intruding call.

6 Messages and information elements

The operations defined in Abstract Syntax Notation One (ASN.1) in clause 11 shall apply.

The APDUs of these operations shall be conveyed within H.450.1 Supplementary Service APDUs included in User-user information elements, as specified in ITU-T H.450.1.

When conveying the invoke APDU of the operations defined in clause 11, the *destinationEntity* data element of the NFE shall contain the value *endpoint*.

When conveying the invoke APDU of operations *remoteUserAlerting* or *callIntrusionNotification* the Interpretation APDU should be included with value *discardAnyUnrecognizedInvokePdu.* When conveying the invoke APDU of the other operations defined in clause 11, the Interpretation APDU should be omitted or included with value *rejectAnyUnrecognizedInvokePdu*.

NOTE – If considered more appropriate by an implementation, the Interpretation APDU may instead be set to rejectAnyUnrecognizedInvokePdu in the case of a callIntrusionNotification invoke APDU with status "callIntrusionComplete", or to clearCallIfAnyInvokePduNotRecognized in the case of a callIntrusionSilentMonitor invoke APDU.

7 Signalling procedures

7.1 Actions at user A's endpoint

7.1.1 Procedure for initial invocation of SS-CI

NOTE – Clause 7.1.5 describes the invocation procedure for a waiting call.

7.1.1.1 Normal procedure

To invoke call intrusion, endpoint A shall perform one of the following actions:

- If normal intrusion is requested, send a *callIntrusionRequest* invoke APDU in the Setup message that establishes the call, start timer T1 and enter state CI-Wait-Ack. The argument shall convey the CICL of the calling user.
- If the silent monitoring option is supported and if an authorized calling user requests it, send a *callIntrusionSilentMonitor* invoke APDU in the Setup message that establishes the call, start timer T1 and enter state CI-Wait-Ack. The argument shall convey the CICL of the calling user and optionally the *call identifier* of an existing call at endpoint B if a specific call is to be monitored.

  NOTE – How call identifiers of active calls at B are obtained is outside the scope of this Recommendation.
- If the forced release option is supported and if an authorized calling user requests it, send a *callIntrusionForcedRelease* invoke APDU in the Setup message that establishes the call, start timer T1 and enter state CI-Wait-Ack. The argument shall convey the CICL of the calling user.

In state CI-Wait-Ack, on receipt of a Connect message including a *callIntrusionRequest, callIntrusionSilentMonitor* or *callIntrusionForcedRelease* return result APDU, endpoint A shall stop timer T1 and shall enter state CI-Orig-Invoked if the result contains status value "callIntruded", or state CI-Orig-Isolated if the result contains status value "callIsolated", or state CI-Idle otherwise. Establishment of media channels shall follow normal H.323 procedures, except that in the case of silent monitoring no logical channels shall be opened for transmitting media from endpoint A.

7.1.1.2 Exceptional procedure

In state CI-Wait-Ack, on receipt of:

- any message containing a *callIntrusionRequest*, *callIntrusionSilentMonitor* or *callIntrusionForcedRelease* return error or reject APDU; or
- an Alerting, Connect or Release Complete message without an SS-CI specific return result, return error or reject APDU,

endpoint A shall stop timer T1 and enter state CI-Idle. Failure of SS-CI may be indicated to the calling user and the call shall continue in accordance with basic call procedures.

On expiry of timer T1, endpoint A shall enter state CI-Idle. Failure of SS-CI may be indicated to the calling user and the call shall continue in accordance with basic call procedures.

7.1.2 Optional procedure for invocation of isolation

7.1.2.1 Normal procedure

In state CI-Orig-Invoked, if isolation of the unwanted user is requested and supported, endpoint A shall send a *callIntrusionIsolate* invoke APDU in a Facility message, start timer T2 and enter state CI-Isolation-Request.

In state CI-Isolation-Request, on receipt of a *callIntrusionIsolate* return result APDU in a Facility message, endpoint A may indicate the result of the isolation request to the calling user, shall stop timer T2 and shall enter the CI-Orig-Isolated state.

7.1.2.2 Exceptional procedure

In state CI-Isolation-Request, on receipt of a Facility message containing a *callIntrusionIsolate* return error or reject APDU, endpoint A may indicate failure of the isolation request to the calling user, shall stop timer T2 and return to state CI-Orig-Invoked.

Upon expiry of timer T2, endpoint A may indicate the rejection of the isolation request to the calling user and shall return to state CI-Orig-Invoked.

7.1.3 Optional procedure for invocation of forced release

7.1.3.1 Normal procedure

In state CI-Orig-Invoked or state CI-Orig-Isolated, if forced release of the unwanted user is requested and supported, endpoint A shall send a *callIntrusionForcedRelease* invoke APDU in a Facility message, start timer T3 and enter state CI-ForcedRelease-Request.

In state CI-ForcedRelease-Request, on receipt of a *callIntrusionForcedRelease* return result APDU in a Facility message, endpoint A may indicate the result of the forced release request to the calling user, shall stop timer T3 and shall enter state CI-Idle.

7.1.3.2 Exceptional procedure

In state CI-ForcedRelease-Request, on receipt of a Facility message containing a *callIntrusionForcedRelease* return error or reject APDU, endpoint A may indicate failure of the forced release request to the calling user, shall stop timer T3 and return to the previous state, CI-Orig-Invoked or CI-Orig-Isolated.

Upon expiry of timer T3, endpoint A may indicate the rejection of the forced release request to the calling user and shall return to the previous state, CI-Orig-Invoked or CI-Orig-Isolated.

7.1.4 Optional procedure for invocation of wait-on-busy

7.1.4.1 Normal procedure

In state CI-Orig-Invoked or state CI-Orig-Isolated, if wait-on-busy is requested and supported, endpoint A shall send a *callIntrusionWOBRequest* invoke APDU in a Facility message, start timer T4 and enter state CI-WOB-Request.

In state CI-WOB-Request, on receipt of a *callIntrusionWOBRequest* return result APDU in a Facility message, endpoint A may indicate acceptance of the wait-on-busy request to the calling user and shall enter state CI-Idle.

NOTE – Call intrusion may be re-invoked subsequently, using the procedures of clause 7.1.5.

7.1.4.2 Exceptional procedure

In state CI-WOB-Request, on receipt of a Facility message containing a *callIntrusionWOBRequest* return error or reject APDU, endpoint A may indicate failure of the wait-on-busy request to the calling user, shall stop timer T4 and return to the previous state, CI-Orig-Invoked or CI-Orig-Isolated.

Upon expiry of timer T4, endpoint A may indicate the rejection of the wait-on-busy request to the calling user and shall return to the previous state, CI-Orig-Invoked or CI-Orig-Isolated.

7.1.5 Optional procedure for (re-)invocation of SS-CI on a waiting call

If for a waiting call SS-CI is requested, endpoint A shall send a *callIntrusionRequest* invoke APDU in a Facility message, including the CICL of the calling user, and follow the procedures of 7.1.1, except that the result should also be expected in a Facility message. The outcome of the SS-CI request may be indicated to the calling user.

If for a waiting call the forced release option of SS-CI is requested and supported, endpoint A shall send a *callIntrusionForcedRelease* invoke APDU in a Facility message, including the CICL of the calling user, and follow the procedures of 7.1.1, except that the result should also be expected in a Facility message. The outcome of the forced release request may be indicated to the calling user.

If for a waiting call the silent monitoring option of SS-CI is requested, endpoint A shall send a *callIntrusionSilentMonitor* invoke APDU in a Facility message, including the CICL of the calling user and optionally the *call identifier* of an established call at endpoint B, and follow the procedures of 7.1.1, except that the result should also be expected in a Facility message. The outcome of the silent monitoring request may be indicated to the calling user.

NOTE – If the invocation of SS-CI is successful the waiting call becomes an intruding call; if the invocation of SS-CI fails the call remains a waiting call.

7.1.6 Procedure for completion of SS-CI

In any state except CI-Idle and CI-Wait-Ack, on receipt of a *callIntrusionNotification* invoke APDU with status information "callIntrusionComplete" in a Facility message, endpoint A may indicate completion of SS-CI to the calling user, shall stop any SS-CI timer and shall enter state CI-Idle.

7.2 Actions at user B's endpoint

7.2.1 Procedure for invocation of SS-CI

7.2.1.1 Normal procedure

If, while processing an incoming Setup or Facility message containing a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU, the called user is found to be busy, endpoint B shall check whether the called user is involved in a compatible active call (in the following called "established call"), that the CIPL of the called user is lower than the received CICL of the calling user, and that there are no other reasons for denying intrusion (e.g. if the established call is already being intruded on or a requested option – forced release or silent monitoring – cannot be supported). If in the case of a silent monitoring request a *call identifier* is present in the argument of the invoke APDU, if it exists the indicated call shall be chosen as the established call, but no other call shall be chosen if no call exists with the indicated *call identifier*.

NOTE 1 – The method by which endpoint B checks whether an active call is compatible with the intruding call is outside the scope of this Recommendation.

If as far as endpoint B is concerned, SS-CI is possible, endpoint B shall check if the CIPL of the remote user of the established call (unwanted user, user C) is known and is lower than the CICL of the calling user and if so, continue with SS-CI as described below.

NOTE 2 – User C's CIPL may be known, for example, through a previous exchange of Common Information.

If user C's CIPL is not known endpoint B shall send a *callIntrusionGetCIPL* invoke APDU in a Facility message to endpoint C, start timer T5 and enter state CI-Get-CIPL. In state CI-Get-CIPL, on receipt of a *callIntrusionGetCIPL* return result APDU in a Facility message, endpoint B shall stop timer T5 and check that the CIPL of user C is lower than the CICL of the calling user.

If all conditions are met, endpoint B may provide a notification of impending intrusion to the involved users. If no such notification is to be given, as for instance in the case of silent monitoring, execution of intrusion shall take place immediately. If notification of impending intrusion is to be

given, endpoint B shall send on the established call (in a Facility message) and optionally also on the intruding call (in an Alerting message if possible, otherwise in a Progress or Facility message) a *callIntrusionNotification* invoke APDU with status information "callIntrusionImpending", start timer T6 and enter state CI-Dest-Notify. If endpoint B also provides an intrusion warning tone, a Progress indicator information element with progress descriptor #8, *In-band information or an appropriate pattern is now available,* should be included in the Alerting or Progress message; the Facility message should not be used in this case; a Facility message should only be chosen if no tones are provided by endpoint B. Execution of intrusion shall commence on expiry of timer T6 in state CI-Dest-Notify.

In the absence of requested options (silent monitoring or forced release), execution of intrusion shall result either in a conference type connection involving all three users (the calling user, the called user and the unwanted user) or in isolation of the unwanted user (disconnection of the unwanted user and connection of the calling user and the called user). The choice between these variants is up to endpoint B.

If a conference type connection is established, endpoint B shall send on the intruding call a Connect message if possible, otherwise (i.e. in the case of re-invocation) a Facility message, containing a *callIntrusionRequest* return result APDU with value "callIntruded". Endpoint B shall also send a Facility message containing a *callIntrusionNotification* invoke APDU with status information "callIntruded" on the established call, join the two calls – e.g. according to H.323 procedures for conference out of consultation – and enter state CI-Dest-Invoked.

If the unwanted user is isolated, endpoint B shall send on the intruding call a Connect message if possible, otherwise (i.e. in the case of re-invocation) a Facility message, containing a *callIntrusionRequest* return result APDU with value "callIsolated". Endpoint B shall also send a Facility message containing a *callIntrusionNotification* invoke APDU with status information "callIsolated" on the established call, put the established call on hold (e.g. by means of H.450.4 procedures), and enter state CI-Dest-Isolated.

If the option forced release is requested and can be supported, endpoint B shall send on the intruding call a Connect message if possible, otherwise (i.e. in the case of re-invocation) a Facility message, containing a *callIntrusionForcedRelease* return result APDU, release the established call and enter (or remain in) state CI-Idle. A *callIntrusionNotification* invoke APDU with status information "callForceReleased" shall be included in the Release Complete message to the unwanted user.

If the option silent monitoring is requested and can be supported, and if the CIPL of users B and C allow silent monitoring (by values *ciProtectionLevel* and *silentMonitoringPermitted*), endpoint B shall send on the intruding call a Connect message if possible, otherwise (i.e. in the case of re-invocation) a Facility message, containing a *callIntrusionSilentMonitor* return result APDU, and enter (or remain in) state CI-Idle. Endpoint B shall open the logical channel(s) necessary for one-way media transmission to user A and join the intruding call into a conference with the established call between users B and C, e.g. by means of H.323 procedures for conference out of consultation. No notification shall be given to users B and C.

7.2.1.2 Exceptional procedure

On receipt of a Setup message containing a *callIntrusionRequest* or *callIntrusionForcedRelease* invoke APDU, if the called user is not busy the call shall continue according to basic call procedures. Endpoint B shall return a *callIntrusionRequest* or *callIntrusionForcedRelease* return error APDU containing error "notBusy" in the resulting Alerting or Connect message to endpoint A and shall remain in state CI-Idle. If the called user is busy but invocation of SS-CI is not possible (including the case where the CIPL of user B or user C is too high) the intruding call shall be released. Endpoint B shall include a *ReleaseCompleteReason* of *destinationReject* and a *callIntrusionRequest* or *callIntrusionForcedRelease* return error APDU containing an error other than "notBusy" in the

Release Complete message and shall remain in state CI-Idle. If a requested option is not supported or not allowed error values "notAvailable" or "notAuthorized" shall be chosen.

On receipt of a Setup message containing a *callIntrusionSilentMonitor* invoke APDU, if the called user is not busy – here in the sense of engaged in a compatible call – or the called user is busy but silent monitoring is not possible for any reason, endpoint B shall remain in state CI-Idle and shall release the call, including in the Release Complete message a *ReleaseCompleteReason* of *destinationReject* and a *callIntrusionSilentMonitor* return error APDU with a suitable error value, e.g. "notBusy" if user B is not engaged in a compatible call, "temporarilyUnavailable" if the argument of the invoke APDU contained a *call identifier* which does not belong to an existing call, or "notAuthorized" if the CIPL of user B or user C does not allow silent monitoring.

On receipt of a Facility message containing a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU, if the called user is not busy, or the called user is busy but invocation of SS-CI is not possible (including the case where the CIPL of user B or user C is too high) endpoint B shall return a Facility message containing a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* return error APDU with a suitable error value and shall remain in the current state. If a requested option is not supported or not allowed error values "notAvailable" or "notAuthorized" shall be chosen.

In state CI-Get-CIPL, on receipt of a *callIntrusionGetCIPL* reject APDU in a Facility message from endpoint C, endpoint B shall stop timer T5 and shall act as if timer T5 had expired.

On expiry of timer T5, endpoint B shall choose a default CIPL for user C and:

- either apply the normal procedures specified above for invocation of intrusion, if user A's CICL is sufficiently high; or
- if user A's CICL is too low, return a Release Complete message with a *ReleaseCompleteReason* of *destinationReject* and with a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* return error APDU containing error "notAuthorized" to endpoint A.

NOTE – The default CIPL value chosen by endpoint B is an implementation or configuration matter. It may depend, for example, on security policies applied within the involved administrative domain(s).

If in the case of a normal intrusion or forced release request, during state CI-Dest-Notify or state CI-Get-CIPL the called user becomes not busy and presentation of the intruding call becomes possible, a *callIntrusionRequest* or *callIntrusionForcedRelease* return error APDU containing error "notBusy" shall be sent in the resulting Alerting, Connect or Facility message to endpoint A, timer T6 or T5 shall be stopped and state CI-Idle shall be entered.

If in the case of a normal intrusion or forced release request, during state CI-Dest-Notify or state CI-Get-CIPL the established call is released but the called user remains busy, a *callIntrusionRequest* or *callIntrusionForcedRelease* return error APDU containing error "temporarilyUnavailable" shall be sent in the resulting Release Complete message to endpoint A, timer T6 or T5 shall be stopped and state CI-Idle shall be entered.

If in the case of a silent monitoring request, during state CI-Get-CIPL the established call is released, a *callIntrusionSilentMonitor* return error APDU containing error "temporarilyUnavailable" and a *ReleaseCompleteReason* of *destinationReject* shall be sent in the resulting Release Complete message to endpoint A, timer T5 shall be stopped and state CI-Idle shall be entered.

7.2.2 Optional procedures for invocation of isolation

7.2.2.1 Normal procedure

In state CI-Dest-Invoked, on receipt of a *callIntrusionIsolate* invoke APDU in a Facility message from endpoint A, if isolation is possible, endpoint B shall disconnect the unwanted user C from the conference type connection by putting user C on hold (e.g. by means of H.450.4 procedures) and

leave the calling and called users connected together. Endpoint B shall also send Facility messages to endpoint A – with a *callIntrusionIsolate* return result APDU – and to endpoint C – with a *callIntrusionNotification* invoke APDU with status information "callIsolated" – and enter state CI-Dest-Isolated.

7.2.2.2 Exceptional procedure

In state CI-Dest-Invoked, on receipt of a *callIntrusionIsolate* invoke APDU in a Facility message from endpoint A, if isolation is not possible, endpoint B shall send a *callIntrusionIsolate* return error APDU in a Facility message to endpoint A and remain in state CI-Dest-Invoked.

7.2.3 Optional procedures for invocation of forced release

7.2.3.1 Normal procedure

In state CI-Dest-Invoked or CI-Dest-Isolated, on receipt of a *callIntrusionForcedRelease* invoke APDU in a Facility message from endpoint A, if forced release is possible, endpoint B shall initiate release of the established call. From state CI-Dest-Invoked, endpoint B shall disconnect the unwanted user from the conference type connection and leave the calling and called users connected together. Endpoint B shall also send a *callIntrusionForcedRelease* return result APDU in a Facility message to endpoint A, and a *callIntrusionNotification* invoke APDU with status information "callForceReleased" in the Release Complete message to endpoint C, and enter state CI-Idle.

7.2.3.2 Exceptional procedure

In state CI-Dest-Invoked or CI-Dest-Isolated, on receipt of a *callIntrusionForcedRelease* invoke APDU in a Facility message from endpoint A, if forced release is not possible, endpoint B shall send a *callIntrusionForcedRelease* return error APDU in a Facility message to endpoint A and remain in state CI-Dest-Invoked or CI-Dest-Isolated, respectively.

7.2.4 Optional procedures for wait-on-busy (WOB)

7.2.4.1 Normal procedures

In state CI-Dest-Invoked or CI-Dest-Isolated, upon receipt of a *callIntrusionWOBRequest* invoke APDU in a Facility message from endpoint A, if WOB is possible endpoint B shall disconnect the calling user from the conference type connection or from the called user, and shall reconnect the unwanted user to the called user. Endpoint B shall also send Facility messages to endpoint A, containing a *callIntrusionWOBRequest* return result APDU, and to endpoint C, containing a *callIntrusionNotification* invoke APDU with status information "callIntrusionEnd", and enter state CI-Dest-WOB. The established call shall no longer be associated with the waiting intruding call and shall continue as if SS-CI had not occurred.

If in state CI-Dest-WOB user B becomes free and the waiting call starts alerting, endpoint B shall send a *remoteUserAlerting* invoke APDU in a Facility message to endpoint A and remain in state CI-Dest-WOB.

7.2.4.2 Exceptional procedure

In state CI-Dest-Invoked or CI-Dest-Isolated, upon receipt of a *callIntrusionWOBRequest* invoke APDU in a Facility message from endpoint A, if WOB is not possible, endpoint B shall send a *callIntrusionWOBRequest* return error APDU in a Facility message to endpoint A and remain in state CI-Dest-Invoked or CI-Dest-Isolated, respectively.

7.2.4.3 Re-invocation of SS-CI

In state CI-Dest-WOB, on receipt of a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU the procedures of 7.2.1 shall apply.

NOTE – This means that endpoint B remains in state CI-Dest-WOB if re-invocation of SS-CI fails.

7.2.5 Procedures for completion of SS-CI

In state CI-Dest-Invoked or CI-Dest-Isolated, if the established call is released, endpoint B shall send a *callIntrusionNotification* invoke APDU with status information "callIntrusionComplete" in a Facility message to endpoint A and enter state CI-Idle. The intruding call shall continue as an active two-party call between users A and B.

In state CI-Dest-WOB, if the waiting call is answered, endpoint B shall send a *callIntrusionNotification* invoke APDU with status information "callIntrusionComplete" in a Facility message to endpoint A and enter state CI-Idle. The call shall continue as an active two-party call between users A and B.

If in the case of silent monitoring (state CI-Idle) the established call is released, the intruding call shall also be released with a *ReleaseCompleteReason* of *destinationReject*.

If the intruding call is released in any state, endpoint B shall enter state CI-Idle and stop any SS-CI timer. If release occurs during state CI-Dest-Notify, CI-Dest-Invoked or CI-Dest-Isolated, the established call shall be restored to the state that existed prior to intrusion and a Facility message containing a *callIntrusionNotification* invoke APDU with status information "callIntrusionEnd" shall be sent on the established call.

7.3 Actions at user C's endpoint

On receipt of a *callIntrusionGetCIPL* invoke APDU in a Facility message on the established call, endpoint C shall send a Facility message to endpoint B, including a *callIntrusionGetCIPL* return result APDU with the CIPL of the unwanted user C and element *silentMonitoringPermitted* if silent monitoring of user C is permitted. If silent monitoring of user C is not permitted element *silentMonitoringPermitted* shall not be included.

On receipt of a *callIntrusionNotification* invoke APDU in a Facility message on the established call, endpoint C may indicate the intrusion status information to user C.

8 Interworking and interactions

8.1 Interworking with SCN

SS-CI may interwork with corresponding supplementary services as defined by other standards by means of gateway interworking functions.

The specification of detailed gateway interworking procedures for SS-CI is beyond the scope of this Recommendation and may be defined for various SCNs by other Recommendations.

8.2 Protocol interaction between SS-CI and other supplementary services

The following subclauses describe protocol interactions of SS-CI with other standardized supplementary services. Further interactions without effects on the protocol may apply – see 5.4.

8.2.1 Call Transfer (SS-CT)

The following protocol interactions shall apply if SS-CT is supported in accordance with ITU-T H.450.2.

If the served user A requests call transfer for two calls and SS-CI option wait-on-busy has been successfully invoked for the secondary call, the actions of SS-CT for transfer during alerting shall apply. The transferred-to endpoint (being in state CI-Dest-WOB) may include a *callWaiting* invoke APDU (see ITU-T H.450.6) when sending the *callTransferSetup* return result APDU in an Alerting

message to the transferred endpoint. The transferred-to endpoint may then also send a *remoteUserAlerting* invoke APDU in a Facility message to the transferred endpoint when the transferred-to user becomes not busy. If no *callWaiting* invoke APDU was sent, then also no *remoteUserAlerting* invoke APDU shall be sent. If the transferred-to user answers, a Connect message shall be sent to the transferred endpoint, but no *callIntrusionNotification* invoke APDU shall be sent.

If call transfer occurs on the established call during silent monitoring, endpoint B shall release the intruding call with a *ReleaseCompleteReason* of *destinationReject*, regardless of which user – B or C – requested the transfer.

If user C transfers the established call while call intrusion is in progress, and the CIPL of the transferred-to user is unknown, endpoint B shall include a *callIntrusionGetCIPL* invoke APDU together with the *callTransferSetup* invoke APDU in the Setup message to the transferred-to endpoint, and the transferred-to endpoint shall return a *callIntrusionGetCIPL* return result APDU together with the *callTransferSetup* return result APDU. If no *callIntrusionGetCIPL* return result APDU is received endpoint B shall assume the default CIPL. If the CICL of the calling user is higher than the transferred-to user's CIPL then the transferred-to user shall become the new user C; otherwise, endpoint B shall release the intruding call with a *ReleaseCompleteReason* of *destinationReject* and enter state CI-Idle.

If the secondary call does not exist, the transferred endpoint may request SS-CI against a (busy) transferred-to user by including in the Setup message a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU together with the *callTransferSetup* invoke APDU. The argument shall contain the CICL of the transferred endpoint. The transferred-to endpoint shall then follow the procedures of 7.2.

8.2.2 Call Forwarding Unconditional (SS-CFU)

The following protocol interactions shall apply if SS-CFU is supported in accordance with ITU-T H.450.3.

When executing call diversion (unconditional), the re-routing endpoint shall include in the Setup message to the diverted-to endpoint any SS-CI invoke APDUs that were present in the Setup message to the (most recent) diverting endpoint, in addition to the *divertingLegInformation2* invoke APDU.

8.2.3 Call Forwarding Busy (SS-CFB)

The following protocol interactions shall apply if SS-CFB is supported in accordance with ITU-T H.450.3.

If endpoint A wishes that a call intrusion request is applied to a busy user even if this user has activated SS-CFB then endpoint A shall include in the Setup message a *cfbOverride* invoke APDU in addition to the *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU.

When executing call diversion (after busy), the re-routing endpoint shall include in the Setup message to the diverted-to endpoint any *SS-CI* invoke APDU that was present in the Setup message to the (most recent) diverting endpoint, in addition to the *divertingLegInformation2* invoke APDU.

If a call including a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU arrives at a busy user who has activated SS-CFB then SS-CFB shall be invoked, unless a *cfbOverride* invoke APDU is also present in the Setup message. If *cfbOverride* is also present then SS-CFB shall be overridden and SS-CI shall apply as described in 7.2.

8.2.4 Call Forwarding on No Reply (SS-CFNR)/Call Deflection (SS-CD)

No protocol interaction.

NOTE – This means that the rerouting endpoint does not include any SS-CI invoke APDU in the new Setup message when executing call diversion (no reply/call deflection).

8.2.5 Call Hold

No protocol interaction.

8.2.6 Call Park/Call Pickup

No protocol interaction.

8.2.7 Call Waiting

The following protocol interactions may apply if SS-CW is supported in accordance with ITU-T H.450.6.

If SS-CI is requested for a waiting call, endpoint A shall follow the procedures of 7.1.5.

In state Call_Waiting_Invoked, on receipt of a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU in a Facility message, endpoint B shall follow the procedures of 7.2.1 and, if successful, enter state Call_Waiting_Idle. If invocation of SS-CI fails, endpoint B shall remain in state Call_Waiting_Invoked.

8.2.8 Message Waiting Indication

No protocol interaction.

8.2.9 Name Presentation

No protocol interaction.

8.2.10 Completion of Calls to Busy Subscriber (SS-CCBS)/on No Reply (SS-CCNR)

No protocol interaction.

8.2.11 Call Offer (SS-CO)

The following protocol interaction may apply if SS-CO is supported in accordance with ITU-T H.450.10.

If for an offered call SS-CI is requested, endpoint A shall follow the procedures of 7.1.5.

In state CO-Dest-Invoked, on receipt of a *callIntrusionRequest, callIntrusionForcedRelease* or *callIntrusionSilentMonitor* invoke APDU in a Facility message, endpoint B shall follow the procedures of 7.2.1 and, if successful, enter state CO-Idle. If invocation of SS-CI fails, endpoint B shall remain in state CO-Dest-Invoked.

8.2.12 Common Information

No protocol interaction.

8.2.13 Call Linkage

No protocol interaction.

9 Gatekeeper/Proxy actions

In the case of a gatekeeper-routed model, two modes are possible:

- the gatekeeper passes on all received SS-CI operations for processing at the endpoint (clause 9.1); or
- the gatekeeper acts on behalf of endpoints A and/or B and/or C for SS-CI (clause 9.2).

NOTE – Besides a gatekeeper, other "transit" entities may act on behalf of an endpoint for SS-CI. A "transit" entity in this sense is referred to as a "proxy" in the following subclauses.

9.1 Gatekeeper passes on SS-CI operations to the endpoint

In this mode, a gatekeeper shall pass on SS-CI operations to the endpoint for appropriate endpoint processing.

NOTE – A gatekeeper may modify the contents of SS-CI operations, if required.

9.2 Gatekeeper/proxy acts on behalf of an endpoint

9.2.1 Gatekeeper/proxy acts on behalf of endpoint A

A gatekeeper/proxy (for the gatekeeper-routed model or in case a call is routed through a proxy) may act as the SS-CI control entity on behalf of endpoint A, and thus become the source of all SS-CI operations sent to endpoint B and the destination for all SS-CI operations destined for endpoint A. The gatekeeper/proxy shall in this case provide the actions as defined in 7.1.

A stimulus-based protocol may be used between the gatekeeper/proxy and user A's endpoint.

9.2.2 Gatekeeper/proxy acts on behalf of endpoint B

A gatekeeper/proxy (for the gatekeeper-routed model or in case a call is routed through a proxy) acting on behalf of endpoint B may decide to become the destination for all SS-CI operations destined for endpoint B and the source of all SS-CI operations sent to endpoint A or endpoint C. The gatekeeper/proxy shall then perform the actions as defined in 7.2.

For this purpose the gatekeeper/proxy shall monitor the busy/free status of endpoint B. How this is achieved is outside the scope of this Recommendation.

The gatekeeper/proxy shall act as MC for the conference type of intrusion and for silent monitoring and shall, in the case of a centralized conference, also provide MP functionality. Alternatively a decentralized conference may be established (see ITU-T H.323 for details) between endpoints A, B and C under the control of the gatekeeper/proxy acting on behalf of endpoint B.

For the held type of intrusion the gatekeeper/proxy may send Hold APDUs (see ITU-T H.450.4) to endpoints B and C and/or initiate the necessary channel reconfiguration using for instance "third party initiated pause and re-routing" procedures as described in ITU-T H.323.

A stimulus-based protocol may be used between the gatekeeper/proxy and user B's endpoint.

9.2.3 Gatekeeper/proxy acts on behalf of endpoint C

A gatekeeper/proxy (for the gatekeeper-routed model or in case a call is routed through a proxy) acting on behalf of endpoint C may decide to become the destination for all SS-CI operations destined for endpoint C and the source of all SS-CI operations sent to endpoint B. The gatekeeper/proxy shall then perform the actions as defined in 7.3.

A stimulus-based protocol may be used between the gatekeeper/proxy and user C's endpoint.

10 Dynamic description

10.1 Operational model

Figure 1 shows the functional model for successful SS-CI before and after SS-CI invocation.

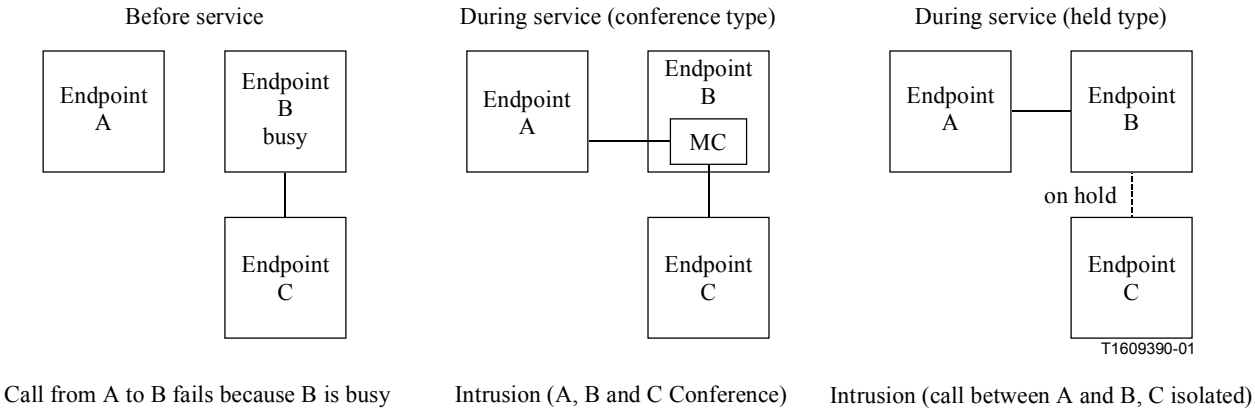
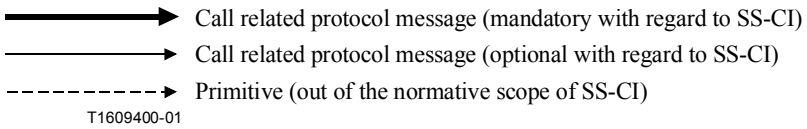


Figure 1/H.450.11 – Operational model for SS-CI

10.2 Signalling flows

This clause describes some typical message flows for SS-CI. The following conventions are used in the figures of this clause.

The following notation is used:

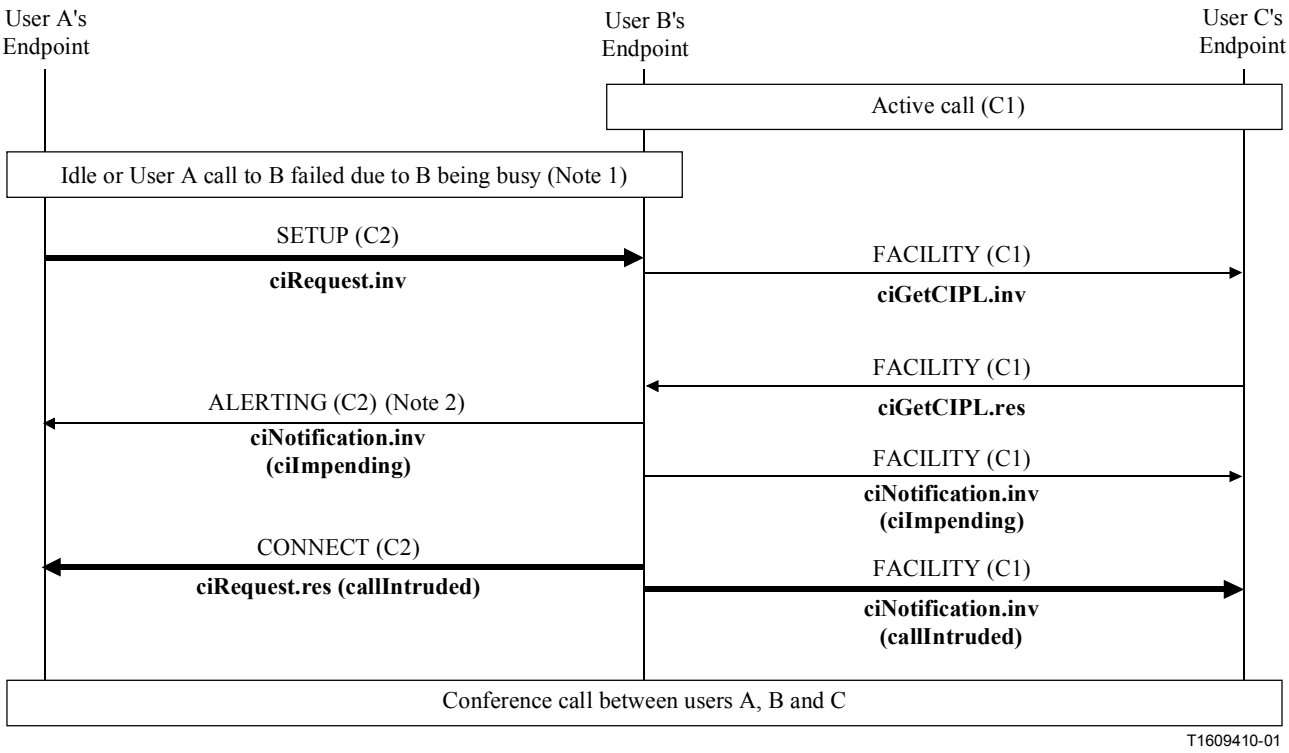


SETUP H.225.0 message name
Cx Number of connection x
xxx.inv Invoke APDU for operation xxx
xxx.res Return result APDU for operation xxx
xxx.err Return error APDU for operation xxx

10.2.1 Successful SS-CI – direct call signalling

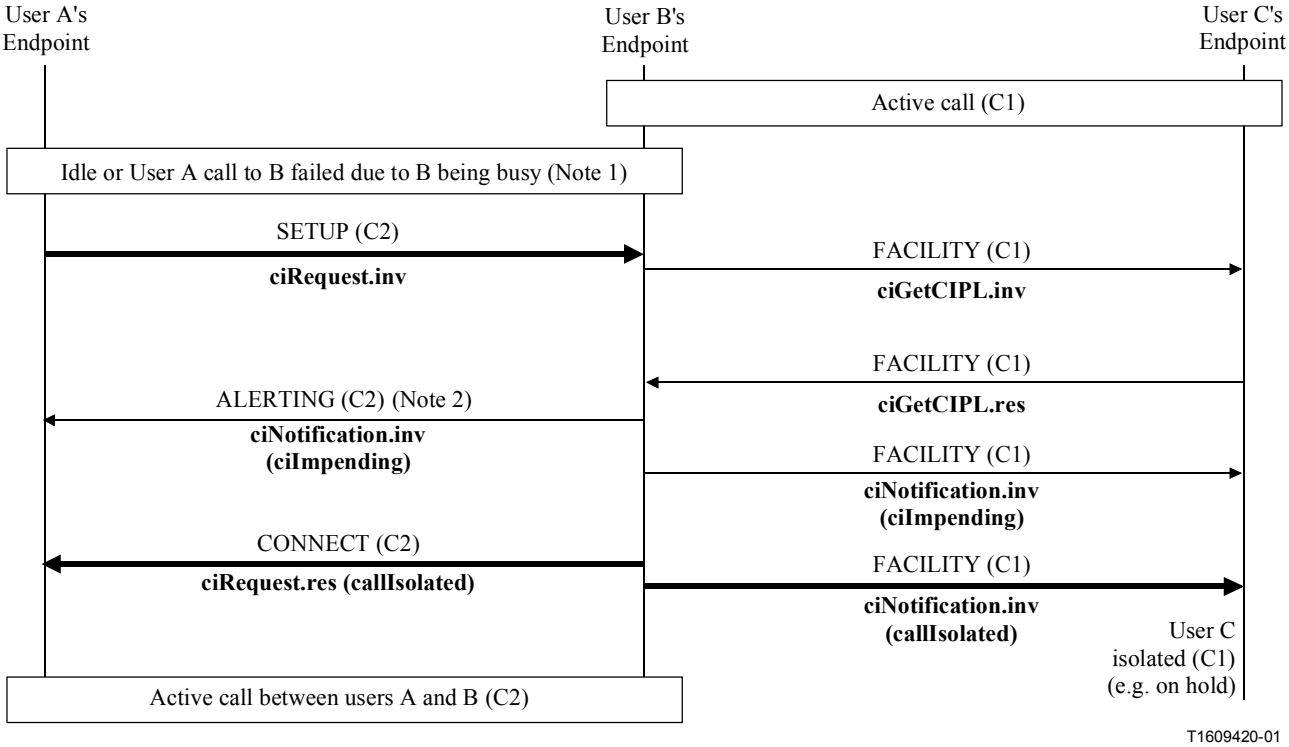
Figures 2 to 11 show example signalling flows for a successful SS-CI invocation and operation.

NOTE – The figures as shown are applicable for both invocation methods of SS-CI, delayed or immediate.



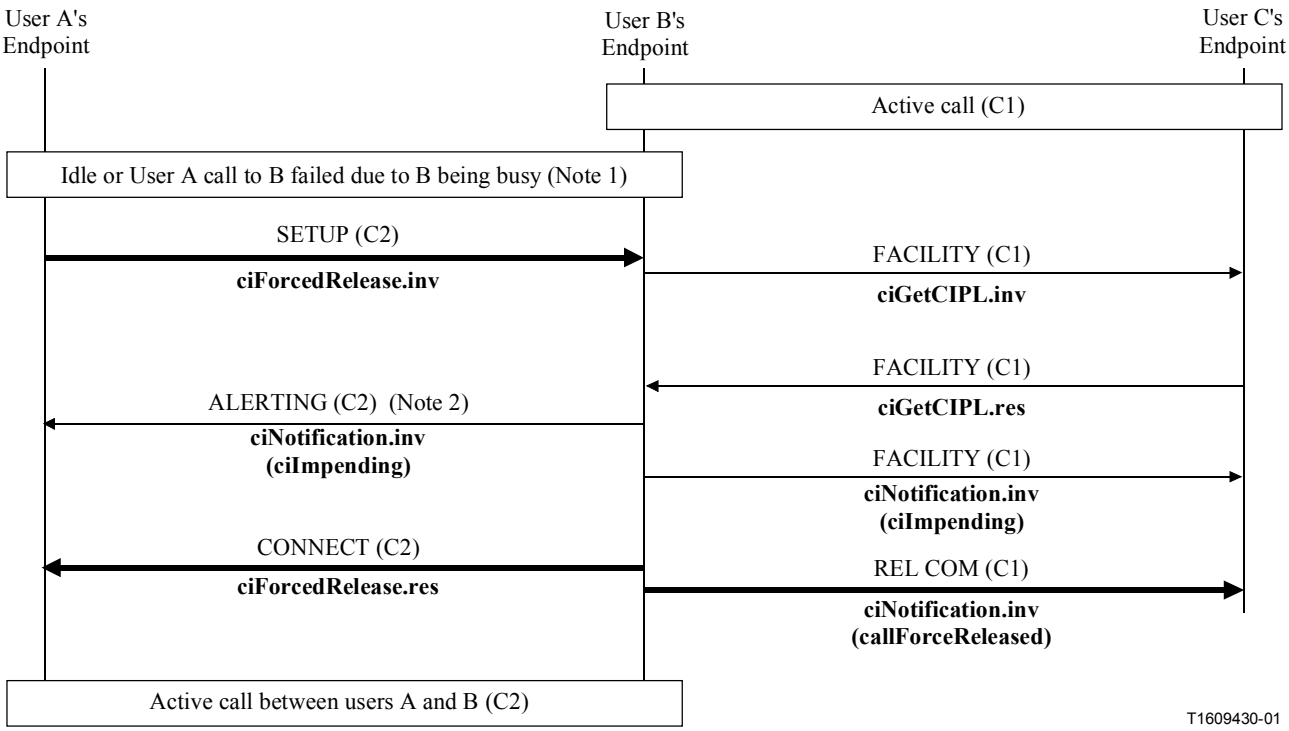
NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf*.
NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

Figure 2/H.450.11 – Example message flow for successful SS-CI – direct routed call signalling, conference type



NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf*.
NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

Figure 3/H.450.11 – Example message flow for successful SS-CI – direct routed call signalling, held type



NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf*.
NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

Figure 4/H.450.11 – Example message flow for successful SS-CI – direct routed call signalling, forced release

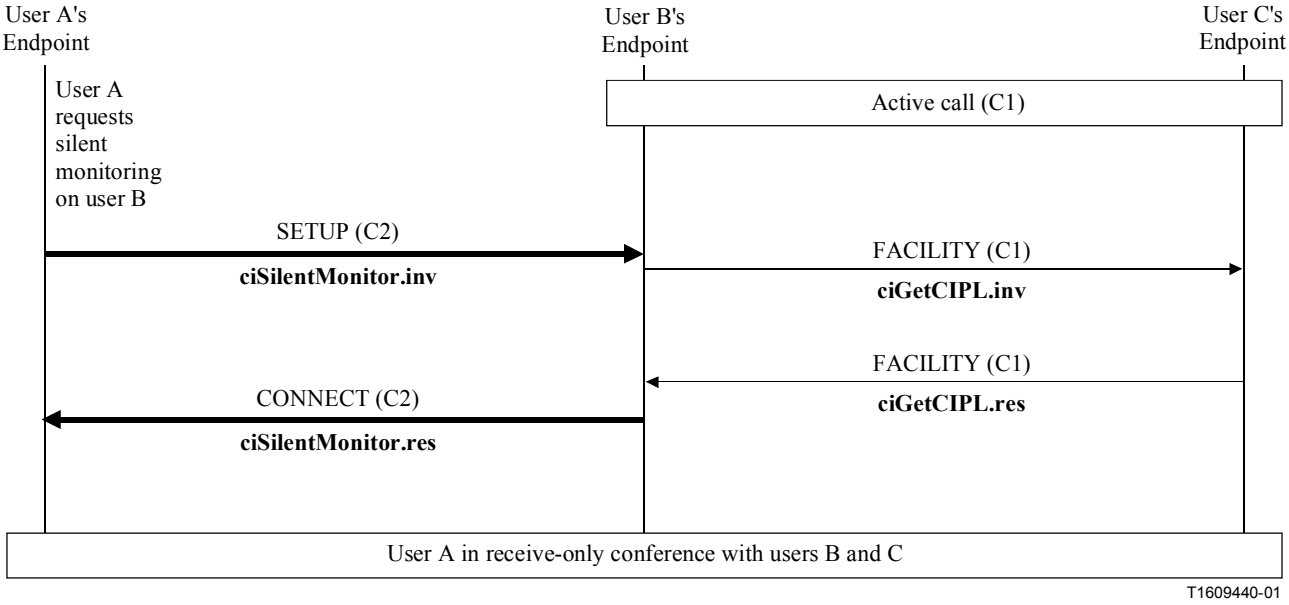


Figure 5/H.450.11 – Example message flow for successful SS-CI – direct routed call signalling, silent monitoring

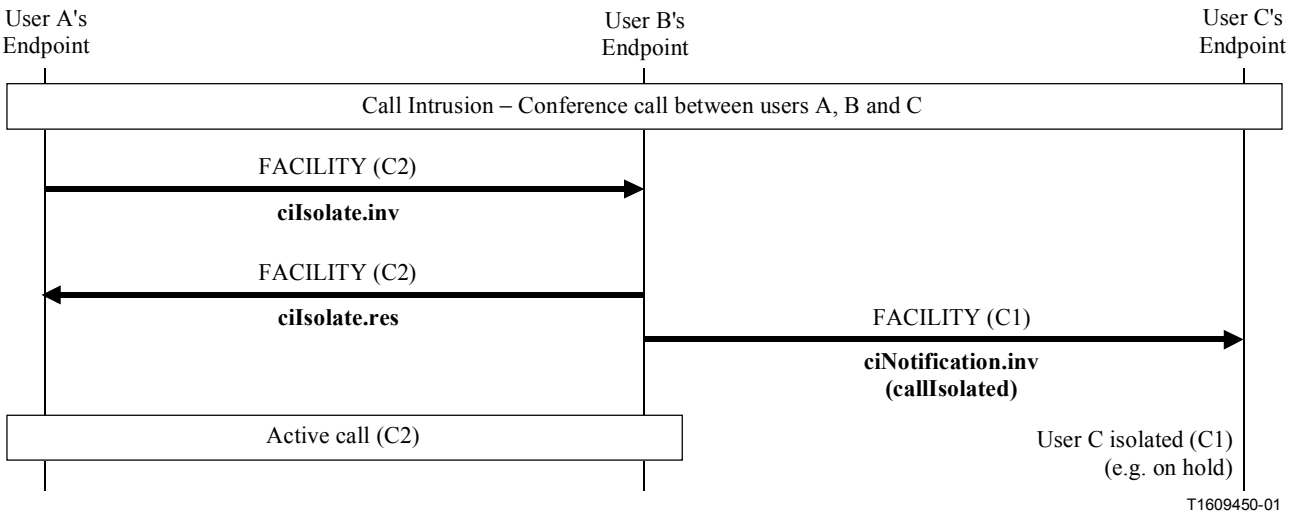


Figure 6/H.450.11 – Example message flow for successful SS-CI – isolation after conference type intrusion

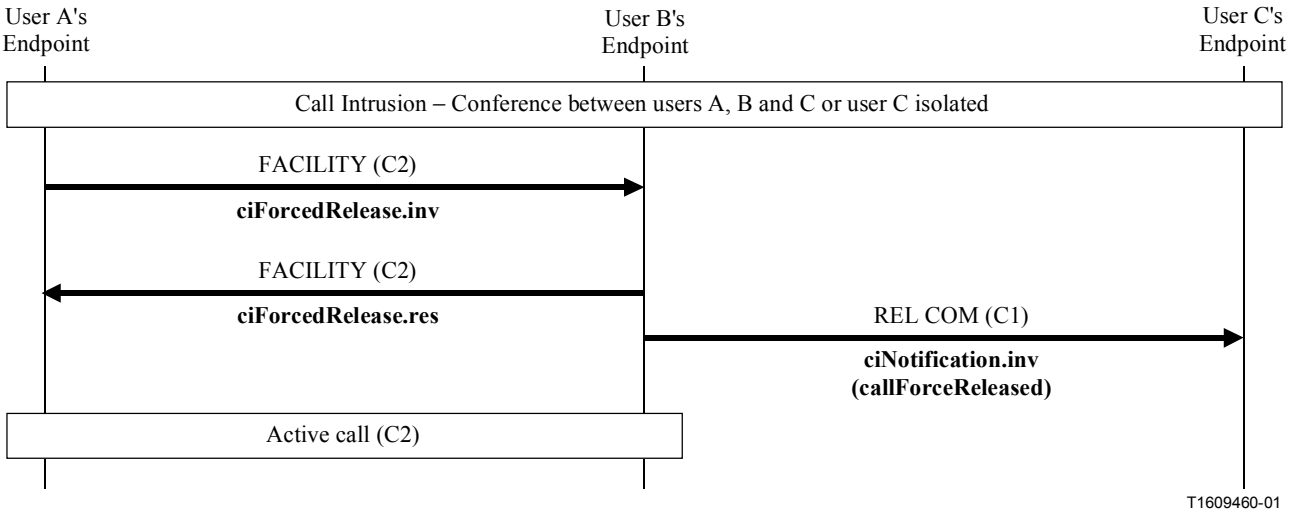


Figure 7/H.450.11 – Example message flow for successful SS-CI – forced release after intrusion

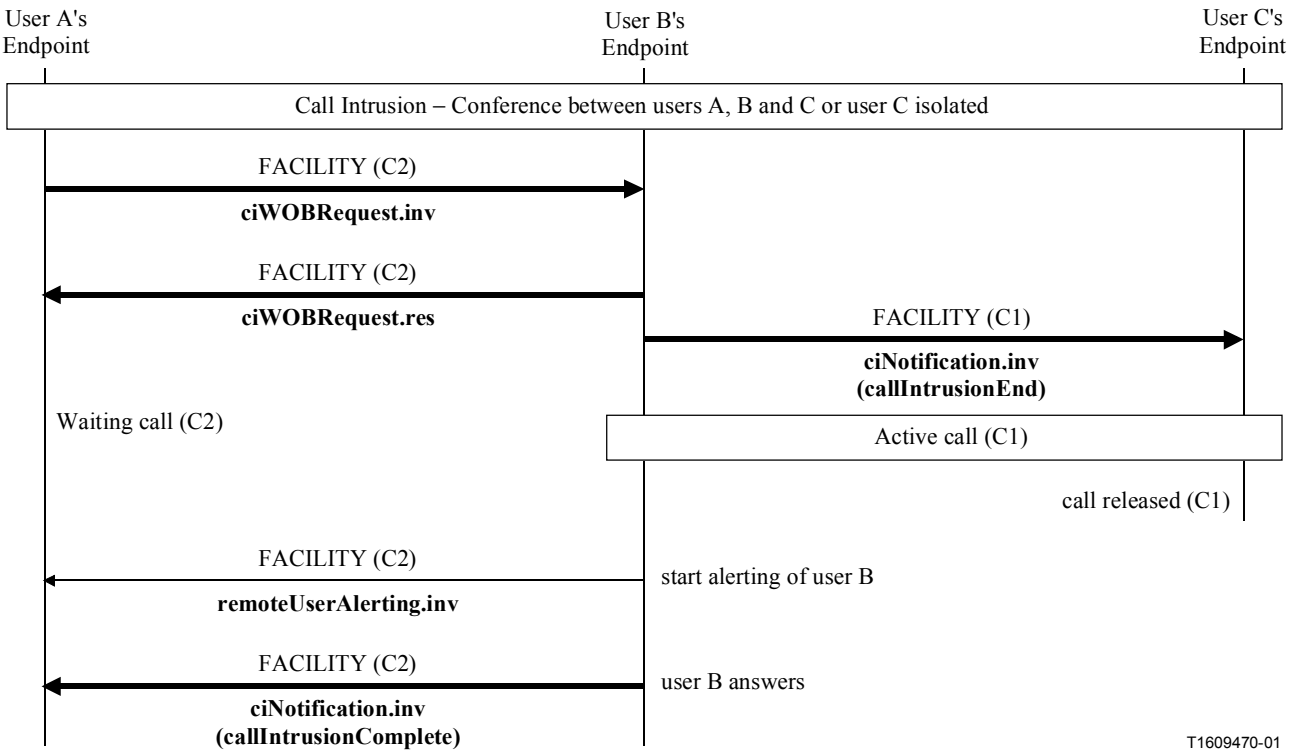


Figure 8/H.450.11 – Example message flow for successful SS-CI – wait-on-busy request after intrusion

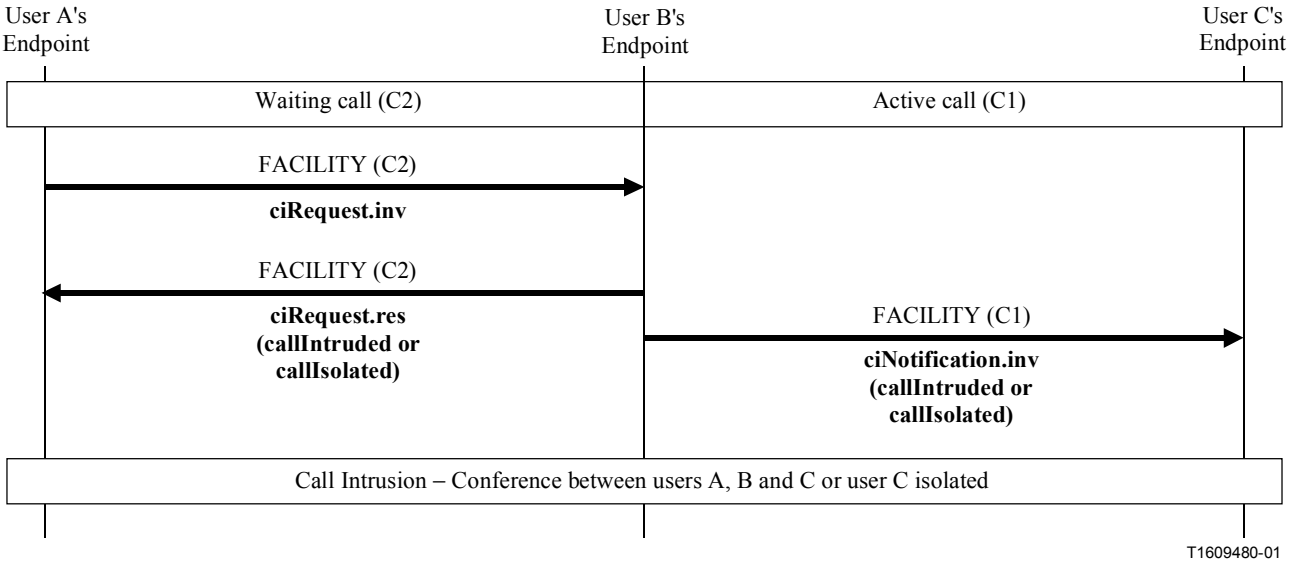


Figure 9/H.450.11 – Example message flow for successful SS-CI – new intrusion request after WOB

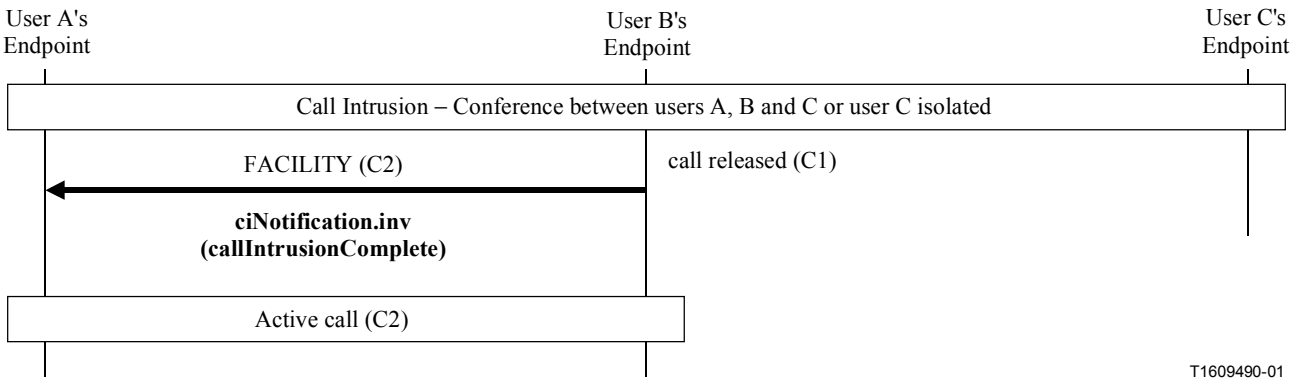


Figure 10/H.450.11 – Example message flow for successful SS-CI – end of intrusion – established call released

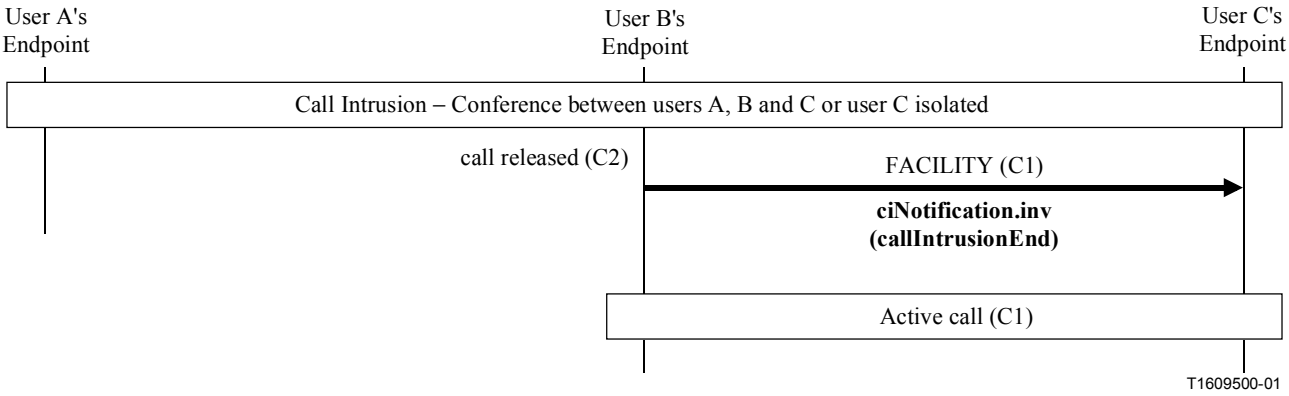
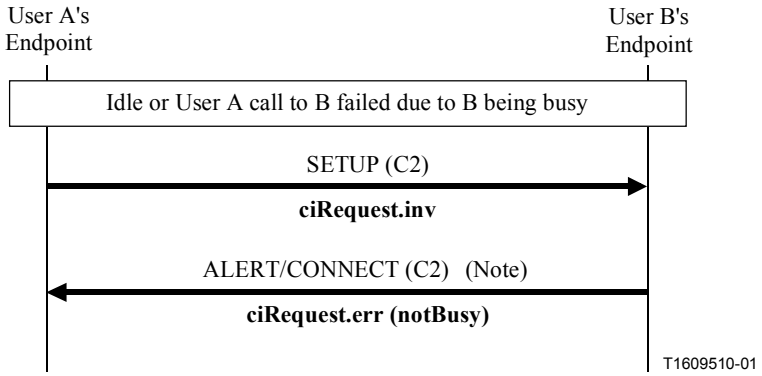


Figure 11/H.450.11 – Example message flow for successful SS-CI – end of intrusion – intruding call released

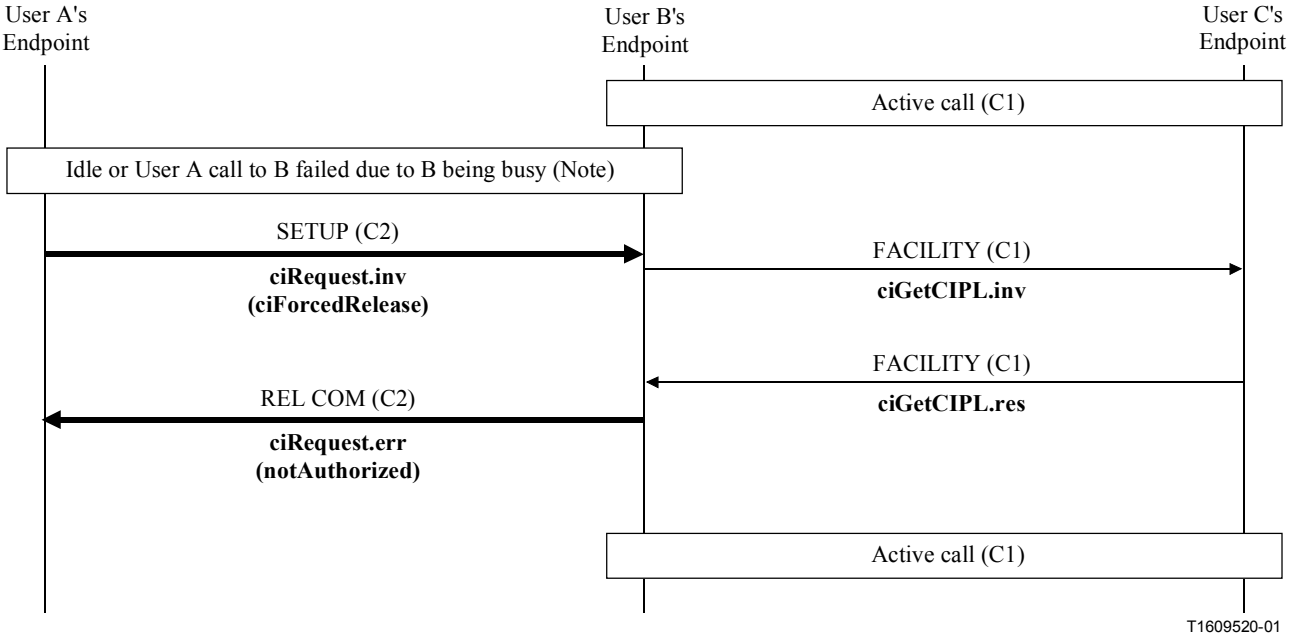
10.2.2 Unsuccessful SS-CI

Figures 12 and 13 show example signalling flows for unsuccessful SS-CI invocation in the direct call signalling case.



NOTE – The first backward message from endpoint B to endpoint A will carry the return error.

Figure 12/H.450.11 – Example message flow for unsuccessful SS-CI – user B not busy



NOTE – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf.*

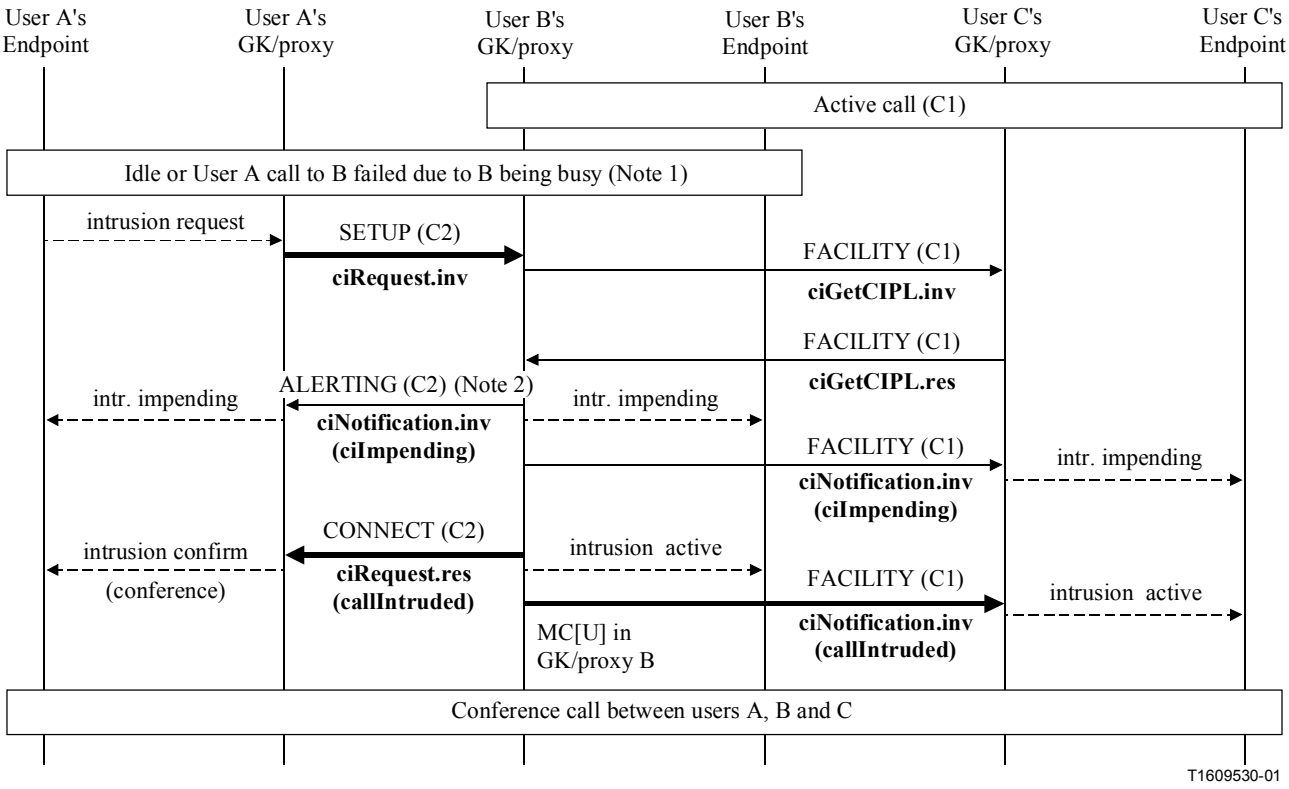
Figure 13/H.450.11 – Example message flow for unsuccessful SS-CI – insufficient capability level

10.2.3 Successful SS-CI – GK routed call signalling

Figures 14 to 17 show examples of the signalling flows for a successful SS-CI invocation and operation with terminal endpoints A, B and C not being capable of SS-CI according to H.450.11 (e.g. H.323 terminals with stimulus feature control). In this example, a gatekeeper A or a proxy A acts on behalf of endpoint A for SS-CI. A gatekeeper B or a proxy B acts on behalf of endpoint B for SS-CI. A gatekeeper C or a proxy C acts on behalf of endpoint C for SS-CI.

As an alternative, endpoints A, B and C may be ITU-T H.248 and H.248 Annex G terminals. In this case, an MGC that terminates H.248 and interworks to H.323/H.450.11 is required within the network (e.g. co-located with the proxy).

The terminal interfaces at endpoints A, B and C as shown illustrate examples only. These interfaces are out of the normative scope of this Recommendation. Only the interfaces between gatekeeper/proxy A and gatekeeper/proxy B and between gatekeeper/proxy B and gatekeeper/proxy C are part of the normative scope of this Recommendation.

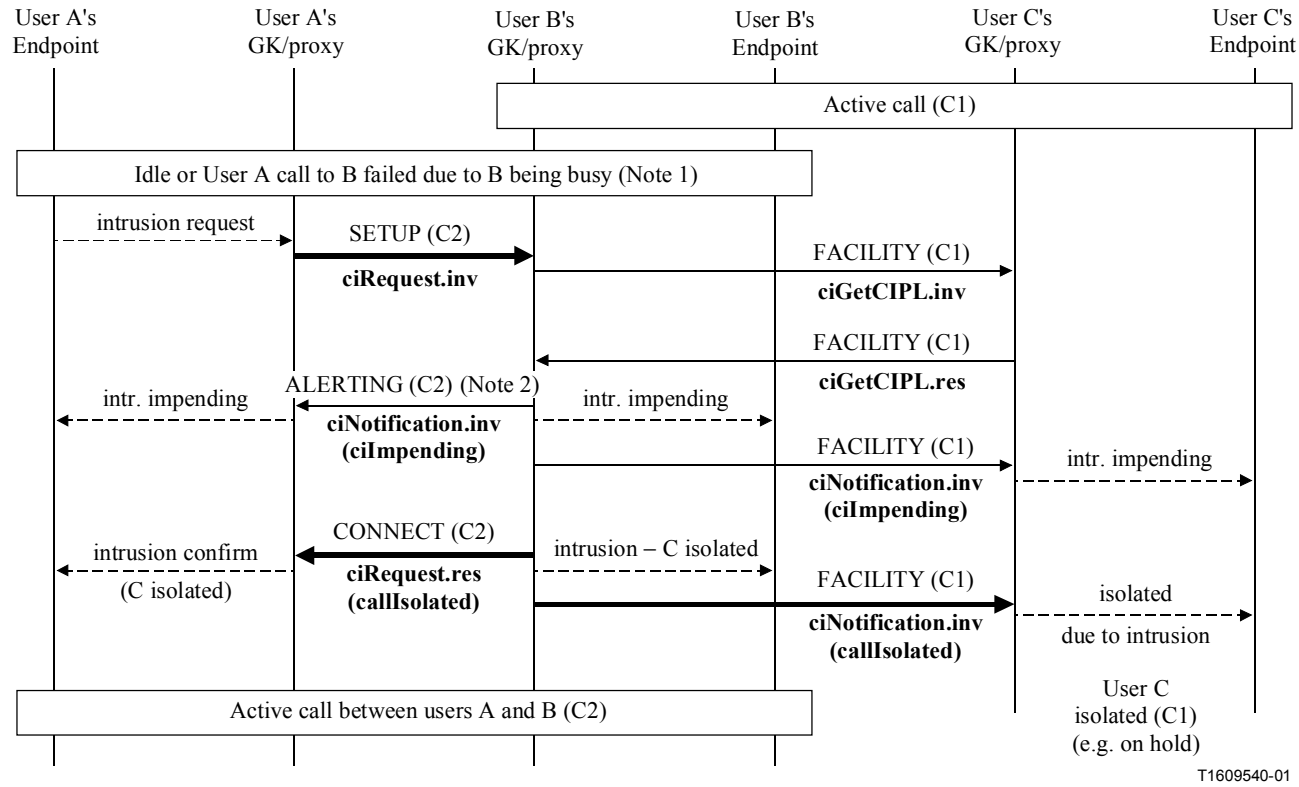


NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf.*

.

NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

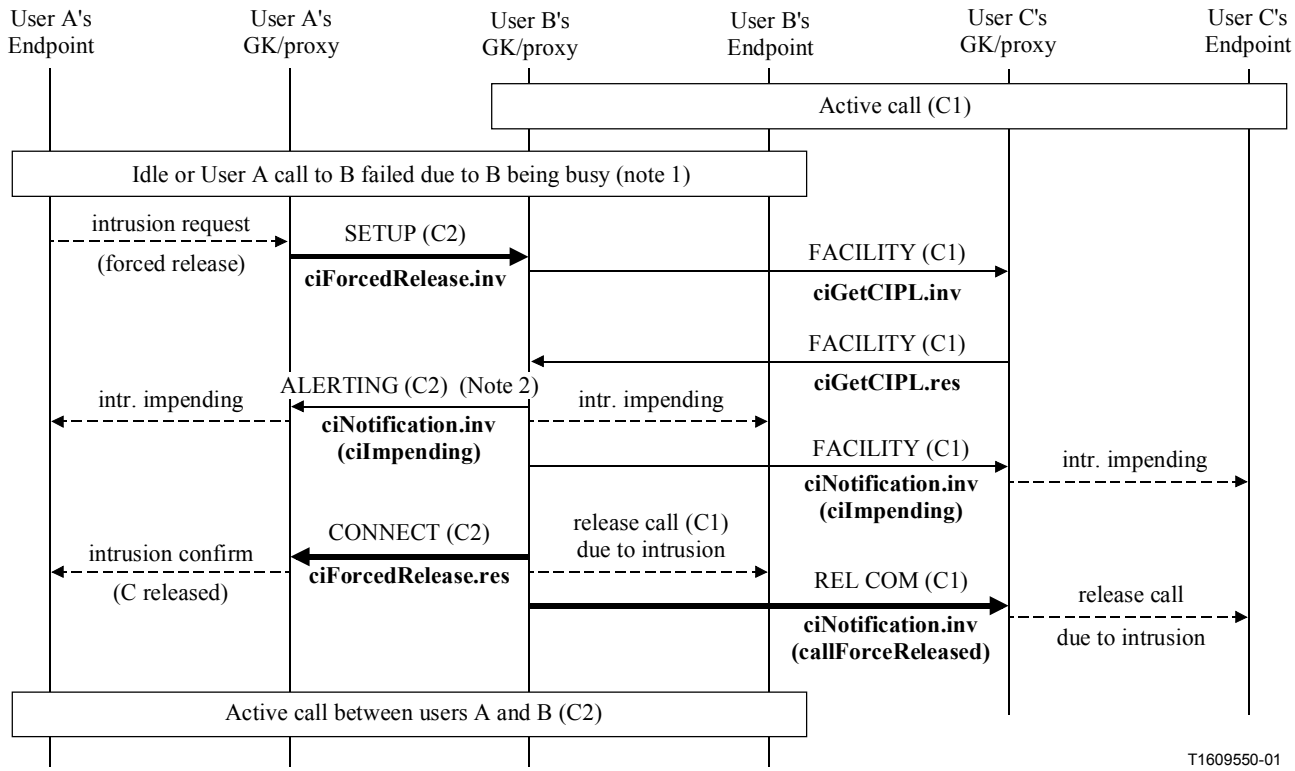
Figure 14/H.450.11 – Example message flow for successful SS-CI – GK routed call signalling, conference type



NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf*.

NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

Figure 15/H.450.11 – Example message flow for successful SS-CI – GK routed call signalling, held type



NOTE 1 – RELEASE COMPLETE received with Cause #17 *user busy* or with releaseCompleteReason *inConf*.

NOTE 2 – PROGRESS or FACILITY are possible alternative messages.

Figure 16/H.450.11 – Example message flow for successful SS-CI – GK routed call signalling, forced release

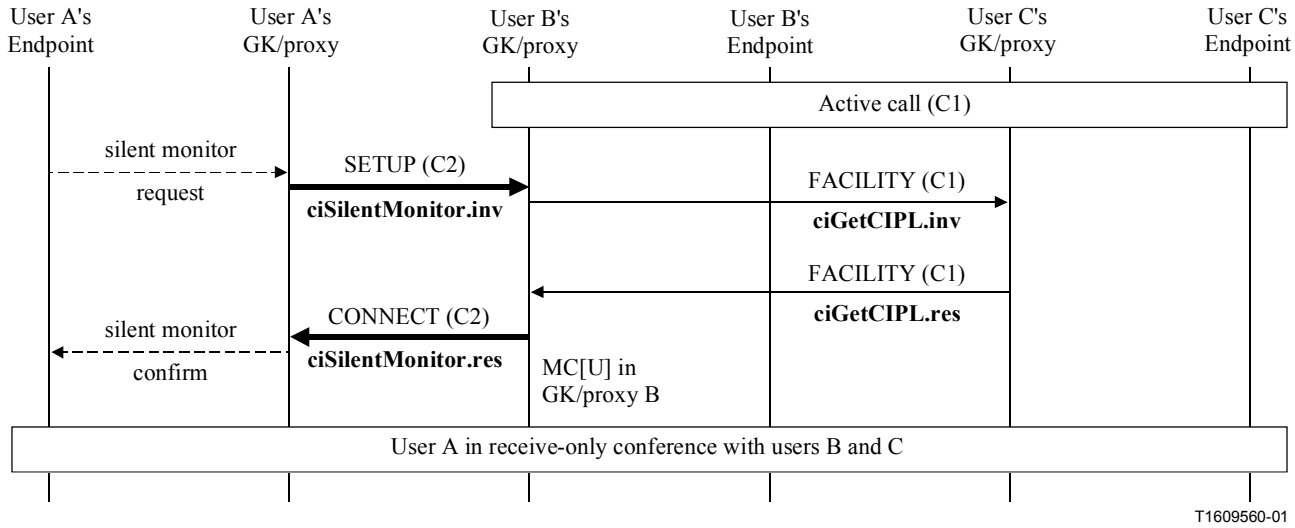


Figure 17/H.450.11 – Example message flow for successful SS-CI – GK routed call signalling, silent monitoring

10.3 Communication between an endpoint A signalling entity (EASE) and its Signalling entity user (for information purposes)

If a gatekeeper/proxy acts on behalf of an endpoint, the gatekeeper/proxy is considered as being the signalling entity, whereas the endpoint served by the gatekeeper/proxy is to be viewed as the signalling entity user. In this case, the local primitive procedures are to be replaced, e.g. by appropriate stimulus feature signalling procedures.

10.3.1 Table of primitives

See Table 1.

Table 1/H.450.11 – Primitives at user A's endpoint

Generic name	Type			
	Request (req)	**Indication (ind)**	**Response (resp)**	**Confirm (conf)**
CiRequest	PARAMETERS	Not defined (Note 1)	Not defined	PARAMETERS
CiIsolate	– (Note 2)	Not defined	Not defined	[PARAMETERS] (Note 3)
CiForcedRelease	[PARAMETERS]	Not defined	Not defined	[PARAMETERS]
CiSilentMonitor	PARAMETERS	Not defined	Not defined	[PARAMETERS]
CiWOBRequest	–	Not defined	Not defined	[PARAMETERS]
CiNotification	Not defined	PARAMETERS	Not defined	Not defined
CfbOverride	–	Not defined	Not defined	Not defined

NOTE 1 – Means that this primitive is not defined.

NOTE 2 – Means no parameters are defined in this Recommendation. Non-standard parameters may be present.

NOTE 3 – Square brackets indicate optionality. Parameters are present only in certain (e.g. error) cases.

10.3.2 Primitive definition

The *CiRequest.Request* primitive is used to invoke SS-CI. The *CiRequest.Confirm* primitive is used to report the outcome of the invocation attempt.

The *CiIsolate.Request* primitive is used to request isolation of user C. The *CiIsolate.Confirm* primitive is used to report the outcome of the isolation attempt.

The *CiForcedRelease.Request* primitive is used to enforce the release of user C. The *CiForcedRelease.Confirm* primitive is used to report the outcome of the forced-release attempt.

The *CiSilentMonitor.Request* primitive is used to invoke silent monitoring. The *CiSilentMonitor.Confirm* primitive is used to report the outcome of the silent monitoring attempt.

The *CiWOBRequest.Request* primitive is used to change from intrusion to wait-on-busy. The *CiWOBRequest.Confirm* primitive is used to report the outcome of the change attempt.

The *CiNotification.Indication* primitive is used to indicate a specific status of intrusion.

The *CfbOverride.Request* primitive is used to request SS-CI against the first busy user B even if this user has activated call forward on busy (SS-CFB).

10.3.3 Parameter definition

CiRequest.Request parameters

cicl: the call intrusion capability level of user A.

extension: non-standard (e.g. manufacturer specific) information (optional).

CiRequest.Confirm parameters

Refer to 10.4.3 (parameters for CiRequest.Response primitive).

CiIsolate.Request parameters

extension: non-standard (e.g. manufacturer specific) information (optional).

CiIsolate.Confirm parameters

Refer to 10.4.3 (parameters for CiIsolate.Response primitive).

CiForcedRelease.Request parameters

cicl: the call intrusion capability level of user A (optional).

extension: non-standard (e.g. manufacturer specific) information (optional).

CiForcedRelease.Confirm parameters

Refer to 10.4.3 (parameters for CiForcedRelease.Response primitive).

CiSilentMonitor.Request parameters

cicl: the call intrusion capability level of user A.

callID: the identifier for an established call of user B (optional).

extension: non-standard (e.g. manufacturer specific) information (optional).

CiSilentMonitor.Confirm parameters

Refer to 10.4.3 (parameters for CiSilentMonitor.Response primitive).

CiWOBRequest.Request parameters

extension: non-standard (e.g. manufacturer specific) information (optional).

CiWOBRequest.Confirm parameters

Refer to 10.4.3 (parameters for CiWOBRequest.Response primitive).

CiNotification.Indication parameters

Refer to 10.4.3 (parameters for CiNotification.Request primitive).

CfbOverride.Request parameters

extension: non-standard (e.g. manufacturer specific) information (optional).

10.4 Communication between an endpoint B signalling entity (EBSE) and its signalling entity user (for information purposes)

If a gatekeeper/proxy acts on behalf of an endpoint, the gatekeeper/proxy is considered as being the signalling entity, whereas the endpoint served by the gatekeeper/proxy is to be viewed as the signalling entity user. In this case, the local primitive procedures are to be replaced, e.g. by appropriate stimulus feature signalling procedures.

10.4.1 Table of primitives

See Table 2.

Table 2/H.450.11 – Primitives at user B's endpoint

Generic name	Type			
	Request (req)	**Indication (ind)**	**Response (resp)**	**Confirm (conf)**
CiRequest	Not defined (Note 1)	PARAMETERS	PARAMETERS	Not defined
CiGetCIPL	– (Note 2)	Not defined	Not defined	PARAMETERS
CiIsolate	Not defined	–	[PARAMETERS] (Note 3)	Not defined
CiForcedRelease	Not defined	[PARAMETERS]	[PARAMETERS]	Not defined
CiSilentMonitor	Not defined	PARAMETERS	[PARAMETERS]	Not defined
CiWOBRequest	Not defined	–	[PARAMETERS]	Not defined
CiNotification	PARAMETERS	Not defined	Not defined	Not defined
CfbOverride	Not defined	–	Not defined	Not defined
NOTE 1 – Means that this primitive is not defined. NOTE 2 – Means no parameters are defined in this Recommendation. Non-standard parameters may be present. NOTE 3 – Square brackets indicate optionality. Parameters are present only in certain (e.g. error) cases.				

10.4.2 Primitive definition

The *CiRequest.Indication* primitive is used to invoke SS-CI. The *CiRequest.Response* primitive is used to report the outcome of the invocation attempt.

The *CiGetCIPL.Request* primitive is used to obtain the protection level of user C. The *CiGetCIPL.Confirm* primitive is used to deliver the protection level of user C.

The *CiIsolate.Indication* primitive is used to request isolation of user C. The *CiIsolate.Response* primitive is used to report the outcome of the isolation attempt.

The *CiForcedRelease.Indication* primitive is used to enforce the release of user C. The *CiForcedRelease.Response* primitive is used to report the outcome of the forced-release attempt.

The *CiSilentMonitor.Indication* primitive is used to request silent monitoring. The *CiSilentMonitor.Response* primitive is used to report the outcome of the silent monitoring attempt.

The *CiWOBRequest.Indication* primitive is used to change from intrusion to wait-on-busy. The *CiWOBRequest.Response* primitive is used to report the outcome of the change attempt.

The *CiNotification.Request* primitive is used to indicate a specific status of intrusion.

The *CfbOverride.Indication* primitive is used to request SS-CI against busy user B even if this user has activated call forward on busy (SS-CFB).

10.4.3 Parameter definition

CiRequest.Indication parameters

Refer to 10.3.3 (parameters for CiRequest.Request primitive).

CiRequest.Response parameters (ack and rej)

(ack) status: status of user C; indicates which variant of intrusion has been applied.

(ack) extension: non-standard (e.g. manufacturer specific) information (optional).

(rej) reason: indication of the reason for failure.

CiGetCIPL.Request parameters

extension: non-standard (e.g. manufacturer specific) information (optional).

CiGetCIPL.Confirm parameters

Refer to 10.5.3 (parameters for CiGetCIPL.Response primitive).

CiIsolate.Indication parameters

Refer to 10.3.3 (parameters for CiIsolate.Request primitive).

CiIsolate.Response parameters (ack and rej)

(ack) extension: non-standard (e.g. manufacturer specific) information (optional).

(rej) reason: indication of the reason for failure.

CiForcedRelease.Indication parameters

Refer to 10.3.3 (parameters for CiForcedRelease.Request primitive).

CiForcedRelease.Response parameters (ack and rej)

(ack) extension: non-standard (e.g. manufacturer specific) information (optional).

(rej) reason: indication of the reason for failure.

CiSilentMonitor.Indication parameters

Refer to 10.3.3 (parameters for CiSilentMonitor.Request primitive).

CiSilentMonitor.Response parameters (ack and rej)

(ack) extension: non-standard (e.g. manufacturer specific) information (optional).

(rej) reason: indication of the reason for failure.

CiWOBRequest.Indication parameters

Refer to 10.3.3 (parameters for CiWOBRequest.Request primitive).

CiWOBRequest.Response parameters (ack and rej)

(ack) extension: non-standard (e.g. manufacturer specific) information (optional).

(rej) reason: indication of the reason for failure.

CiNotification.Request parameters

status: status information with regard to SS-CI.

extension: non-standard (e.g. manufacturer specific) information (optional).

CfbOverride.Request parameters

Refer to 10.3.3 (parameters for CfbOverride.Request primitive).

10.5 Communication between an endpoint C signalling entity (ECSE) and its signalling entity user (for information purposes)

If a gatekeeper/proxy acts on behalf of an endpoint, the gatekeeper/proxy is considered as being the signalling entity, whereas the endpoint served by the gatekeeper/proxy is to be viewed as the signalling entity user. In this case, the local primitive procedures are to be replaced, e.g. by appropriate stimulus feature signalling procedures.

10.5.1 Table of primitives

See Table 3.

Table 3/H.450.11 – Primitives at user C's endpoint

Generic name	Type			
	Request (req)	Indication (ind)	Response (resp)	Confirm (conf)
CiGetCIPL	Not defined (Note 1)	– (Note 2)	PARAMETERS	Not defined
CiNotification	Not defined	PARAMETERS	Not defined	Not defined
NOTE 1 – Means that this primitive is not defined. NOTE 2 – Means no parameters are defined in this Recommendation. Non-standard parameters may be present.				

10.5.2 Primitive definition

The *CiGetCIPL.Indication* primitive is used to obtain the protection level of user C. The *CiGetCIPL.Response* primitive is used to deliver the protection level of user C.

The *CiNotification.Indication* primitive is used to indicate a specific status of intrusion.

10.5.3 Parameter definition

CiGetCIPL.Indication parameters

Refer to 10.4.3 (parameters for CiGetCIPL.Request primitive).

CiGetCIPL.Response parameters (ack)

cipl: call intrusion protection level of user C.

silentMonitoring: silent monitoring permitted (optional).

extension: non-standard (e.g. manufacturer specific) information (optional).

CiNotification.Indication parameters

Refer to 10.4.3 (parameters for CiNotification.Request primitive).

10.6 Call states

10.6.1 Call states at endpoint A

The procedures for endpoint A are written in terms of the following conceptual states existing within the SS-CI signalling entity EASE in association with a particular call.

CI State	Description
CI-Idle	This state exists if SS-CI is not active.
CI-Wait-Ack	This state exists after an SS-CI request while waiting for a response.
CI-Orig-Invoked	This state exists while intrusion is active in a conference type connection.
CI-Orig-Isolated	This state exists while intrusion is active with user C isolated.
CI-Isolation-Request	This state exists for an active intrusion while waiting for the response to an isolation request.
CI-ForcedRelease-Request	This state exists for an active intrusion while waiting for the response to a forced-release request.
CI-WOB-Request	This state exists for an active intrusion while waiting for the response to a wait-on-busy request.

10.6.2 Call states at endpoint B

The procedures for endpoint B are written in terms of the following conceptual states existing within the SS-CI signalling entity EBSE in association with a particular call.

CI State	Description
CI-Idle	This state exists if SS-CI is not active.
CI-Get-CIPL	This state exists after sending a request for CIPL to endpoint C while waiting for the response.
CI-Dest-Notify	This state exists while an impending intrusion warning is given.
CI-Dest-Invoked	This state exists while intrusion is active in a conference type connection.
CI-Dest-Isolated	This state exists while intrusion is active with user C isolated.
CI-Dest-WOB	This state exists during wait-on-busy.

10.6.3 Call states at endpoint C

The procedures for endpoint C are written in terms of the following conceptual states existing within the SS-CI signalling entity ECSE in association with a particular call.

CI State	Description
CI-Idle	This state exists if SS-CI is not active.

10.7 Timers

10.7.1 Timers at endpoint A

Timer T1

Timer T1 operates during state CI-Wait-Ack. Its purpose is to protect against an absence of response to a request for invocation or re-invocation of intrusion.

Timer T1 should have a value not less than 30 s.

Timer T2

Timer T2 operates during state CI-Isolation-Request. Its purpose is to protect against an absence of response to a request for isolation.

Timer T2 should have a value not less than 30 s.

Timer T3

Timer T3 operates during state CI-ForcedRelease-Request. Its purpose is to protect against an absence of response to a request for forced release.

Timer T3 should have a value not less than 30 s.

Timer T4

Timer T4 operates during state CI-WOB-Request. Its purpose is to protect against an absence of response to a request for wait on busy.

Timer T4 should have a value not less than 30 s.

10.7.2 Timers at endpoint B

Timer T5

Timer T5 operates during state CI-Get-CIPL. Its purpose is to protect against an absence of response to a request for the CIPL of the unwanted user.

Timer T5 should have a value not less than 10 s.

Timer T6

Timer T6 operates during state CI-Dest-Notify. Its purpose is to control the delay between the impending intrusion warning notification and the execution of intrusion.

Timer T6 should have a value not higher than 10 s.

11 Operations in support of Call Intrusion supplementary service

The operations defined in Abstract Syntax Notation One (ASN.1) below shall apply.

```
Call-Intrusion-Operations
     { itu-t recommendation h 450 11 version1(0) call-intrusion-operations(0)}

DEFINITIONS AUTOMATIC TAGS  ::=

BEGIN
IMPORTS       OPERATION, ERROR FROM Remote-Operations-Information-Objects
                         {joint-iso-itu-t remote-operations(4)
                         informationObjects(5) version1(0)}
              CallIdentifier FROM H323-MESSAGES -- see H.225.0
              MixedExtension, undefined FROM Call-Hold-Operations
                         {itu-t recommendation h 450 4 version1(0)
                         call-hold-operations(0)}
```

```
            notAvailable, supplementaryServiceInteractionNotAllowed FROM
                     H4501-General-Error-List
                     { itu-t recommendation h 450 1 version1(0)
                     general-error-list (1) }
            callWaiting FROM Call-Waiting-Operations
                     {itu-t recommendation h 450 6 version1(0)
                     call-waiting-operations(0)}
            cfbOverride, remoteUserAlerting FROM Call-Offer-Operations
                     {itu-t recommendation h 450 10 version1(0)
                     call-offer-operations(0)};

H323CallIntrusionOperations  OPERATION ::=
{callIntrusionRequest | callIntrusionGetCIPL | callIntrusionIsolate |
callIntrusionForcedRelease |
callIntrusionWOBRequest | callIntrusionSilentMonitor | callIntrusionNotification
|  cfbOverride | remoteUserAlerting | callWaiting }
          -- callWaiting is only used for interaction with Call Transfer --

callIntrusionRequest         OPERATION    ::=
    {
            ARGUMENT           CIRequestArg
            RESULT             CIRequestRes
            ERRORS             { notAvailable |
                                 notBusy |
                                 temporarilyUnavailable |
                                 notAuthorized |
                                 undefined |
                                 supplementaryServiceInteractionNotAllowed }
            CODE               local: 43
    }

callIntrusionGetCIPL         OPERATION    ::=
    {
            ARGUMENT           CIGetCIPLOptArg OPTIONAL TRUE
            RESULT             CIGetCIPLRes
            ALWAYS RESPONDS    FALSE
            CODE               local: 44
    }

callIntrusionIsolate         OPERATION    ::=
    {
            ARGUMENT           CIIsOptArg OPTIONAL TRUE
            RESULT             CIIsOptRes OPTIONAL TRUE
            ERRORS             { notAvailable |
                                 undefined |
                                 supplementaryServiceInteractionNotAllowed }
            CODE               local: 45
    }

callIntrusionForcedRelease   OPERATION    ::=
    {
            ARGUMENT           CIFrcRelArg OPTIONAL TRUE
            RESULT             CIFrcRelOptRes OPTIONAL TRUE
            ERRORS             { notAvailable |
                                 notBusy |
                                 temporarilyUnavailable |
                                 notAuthorized |
                                 undefined|
                                 supplementaryServiceInteractionNotAllowed }
            CODE               local: 46
    }
```

```
callIntrusionWOBRequest     OPERATION ::=
     {
               ARGUMENT           CIWobOptArg OPTIONAL TRUE
               RESULT             CIWobOptRes OPTIONAL TRUE
               ERRORS             { notAvailable |
                                    undefined |
                                    supplementaryServiceInteractionNotAllowed }
               CODE               local: 47
     }

callIntrusionSilentMonitor    OPERATION    ::=
     {
               ARGUMENT           CISilentArg
               RESULT             CISilentOptRes OPTIONAL TRUE
               ERRORS             { notAvailable |
                                    notBusy |
                                    temporarilyUnavailable |
                                    notAuthorized |
                                    undefined |
                                    supplementaryServiceInteractionNotAllowed }
               CODE               local: 116
     }

callIntrusionNotification     OPERATION    ::=
     {
               ARGUMENT           CINotificationArg
               RETURN RESULT      FALSE
               ALWAYS RESPONDS    FALSE
               CODE               local: 117
     }

CIRequestArg       ::=  SEQUENCE
     {
          ciCapabilityLevel       CICapabilityLevel,
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIRequestRes       ::=  SEQUENCE
     {
          ciStatusInformation     CIStatusInformation,
          resultExtension         SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIGetCIPLOptArg    ::=  SEQUENCE
     {
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIGetCIPLRes       ::=  SEQUENCE
     {
          ciProtectionLevel            CIProtectionLevel,
          silentMonitoringPermitted    NULL OPTIONAL,
          resultExtension              SEQUENCE SIZE (0..255) OF MixedExtension
                                       OPTIONAL,
          ...
     }
```

```
CIIsOptArg          ::=  SEQUENCE
     {
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIIsOptRes          ::=  SEQUENCE
     {
          resultExtension         SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIFrcRelArg         ::=  SEQUENCE
     {
          ciCapabilityLevel       CICapabilityLevel,
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIFrcRelOptRes      ::=  SEQUENCE
     {
          resultExtension         SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIWobOptArg         ::=  SEQUENCE
     {
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CIWobOptRes         ::=  SEQUENCE
     {
          resultExtension         SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CISilentArg         ::=  SEQUENCE
     {
          ciCapabilityLevel       CICapabilityLevel,
          specificCall            CallIdentifier  OPTIONAL,
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CISilentOptRes      ::=  SEQUENCE
     {
          resultExtension         SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
          ...
     }

CINotificationArg   ::=  SEQUENCE
     {
          ciStatusInformation     CIStatusInformation,
          argumentExtension       SEQUENCE SIZE (0..255) OF MixedExtension
                                  OPTIONAL,
```

```
        ...
    }

CICapabilityLevel  ::=  INTEGER (1..3)
                   {          intrusionLowCap(1),
                              intrusionMediumCap(2),
                              intrusionHighCap(3)
                   }

CIProtectionLevel  ::=  INTEGER (0..3)
                   {          lowProtection(0),
                              mediumProtection(1),
                              highProtection(2),
                              fullProtection(3)
                   }

CIStatusInformation    ::= CHOICE
    {
                   callIntrusionImpending      NULL,
                   callIntruded                NULL,
                   callIsolated                NULL,
                   callForceReleased           NULL,
                   callIntrusionComplete       NULL,
                   callIntrusionEnd            NULL,
                   ...
    }

notBusy  ERROR    ::=
{   code     local:1009     }   -- used when the called user is not busy

temporarilyUnavailable  ERROR ::=
{   code     local:1000     }   -- used when conditions for invocation of SS-CI
                                -- are momentarily not met

notAuthorized ERROR    ::=
{   code     local:1007     }   -- used when a SS-CI request is rejected
                                -- because of insufficient CICL or if silent
                                -- monitoring is not permitted

END  -- of Call-Intrusion-Operations
```

12 Specification and description language (SDL) diagrams for SS-CI

The procedures for Call Intrusion signalling entities are described in SDL form in Figures 19 through 29. The SDLs only show SS-CI specific information transported on an H.225.0 connection. H.245 procedures (e.g. terminal capability exchange, master/slave determination, opening and closing of logical channels, etc.) are not shown. The following abbreviations are used:

BC Basic Call

err Return error APDU

inv Invoke APDU

rej Reject APDU or Rejection

res Return result APDU

In case of a conflict between SDLs and the text within the previous clauses, the text shall take precedence.

Specific gatekeeper/proxy SDLs for the model where a gatekeeper/proxy acts on SS-CI on behalf of an endpoint are not provided.

The symbols used in the following SDLs, irrespective of the direction of input and output signals, are defined in Figure 18.

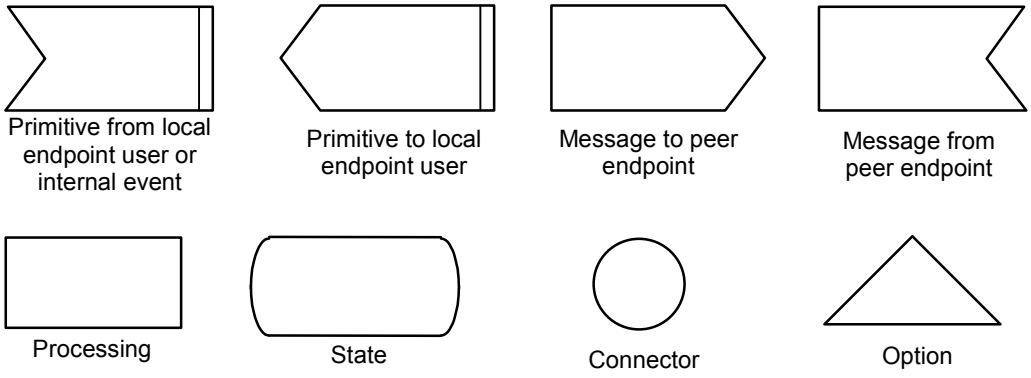


Figure 18/H.450.11 – SDL symbols

12.1 Behaviour of user A's endpoint

Figures 19 through 23 show the behaviour of user A's endpoint.

Input signals from the left and output signals to the left represent primitives:

- from or to the served user (user A);
- from or to basic call control; these primitives are indicated by "BC";
- internal signals, e.g. timer expiry.

Input signals from the right and output signals to the right represent messages from or to the called peer SS-Control entity (i.e. in user B's endpoint) which carry SS-CI control information.

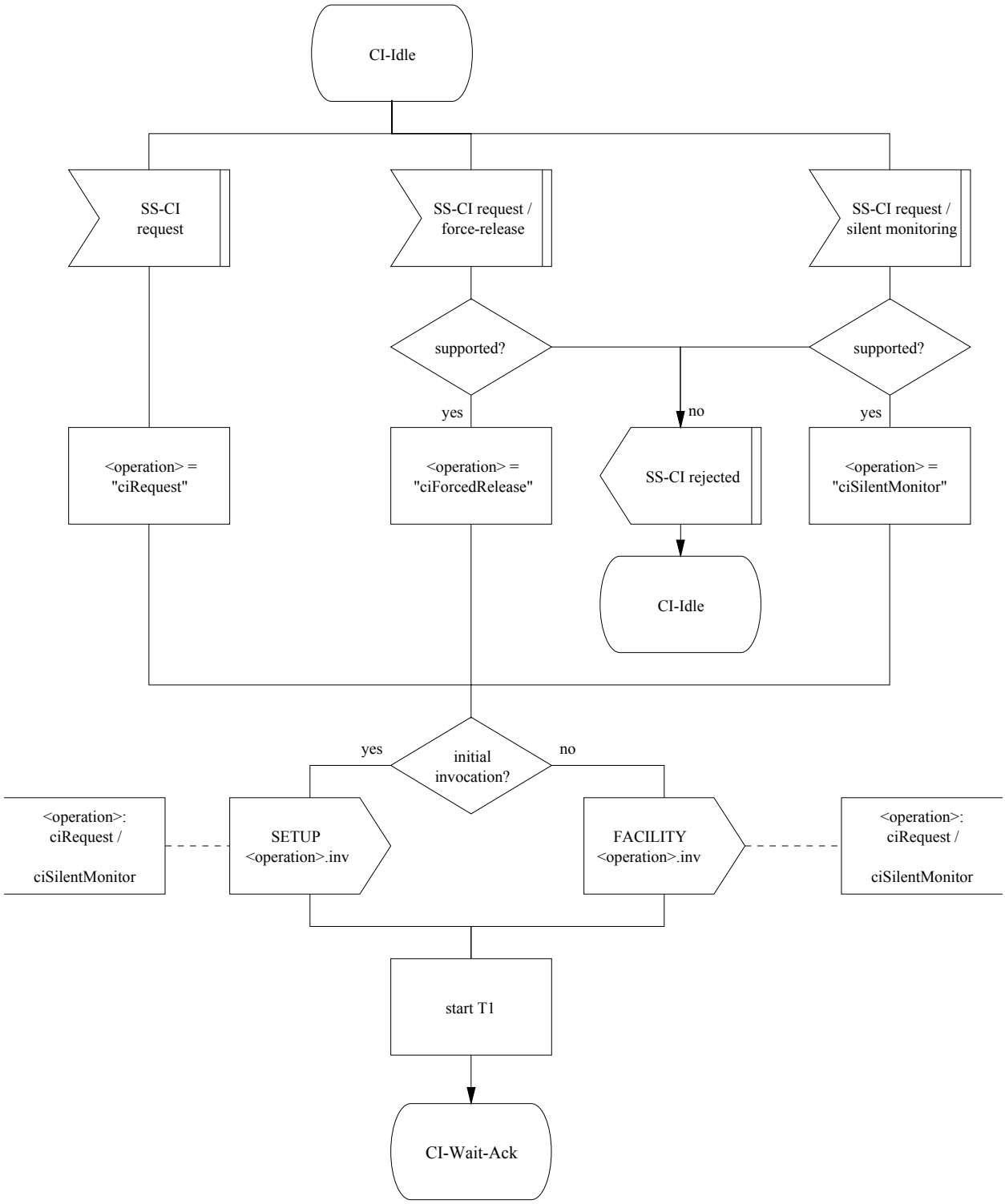


Figure 19/H.450.11 – Endpoint A SDL *(sheet 1 of 5)*

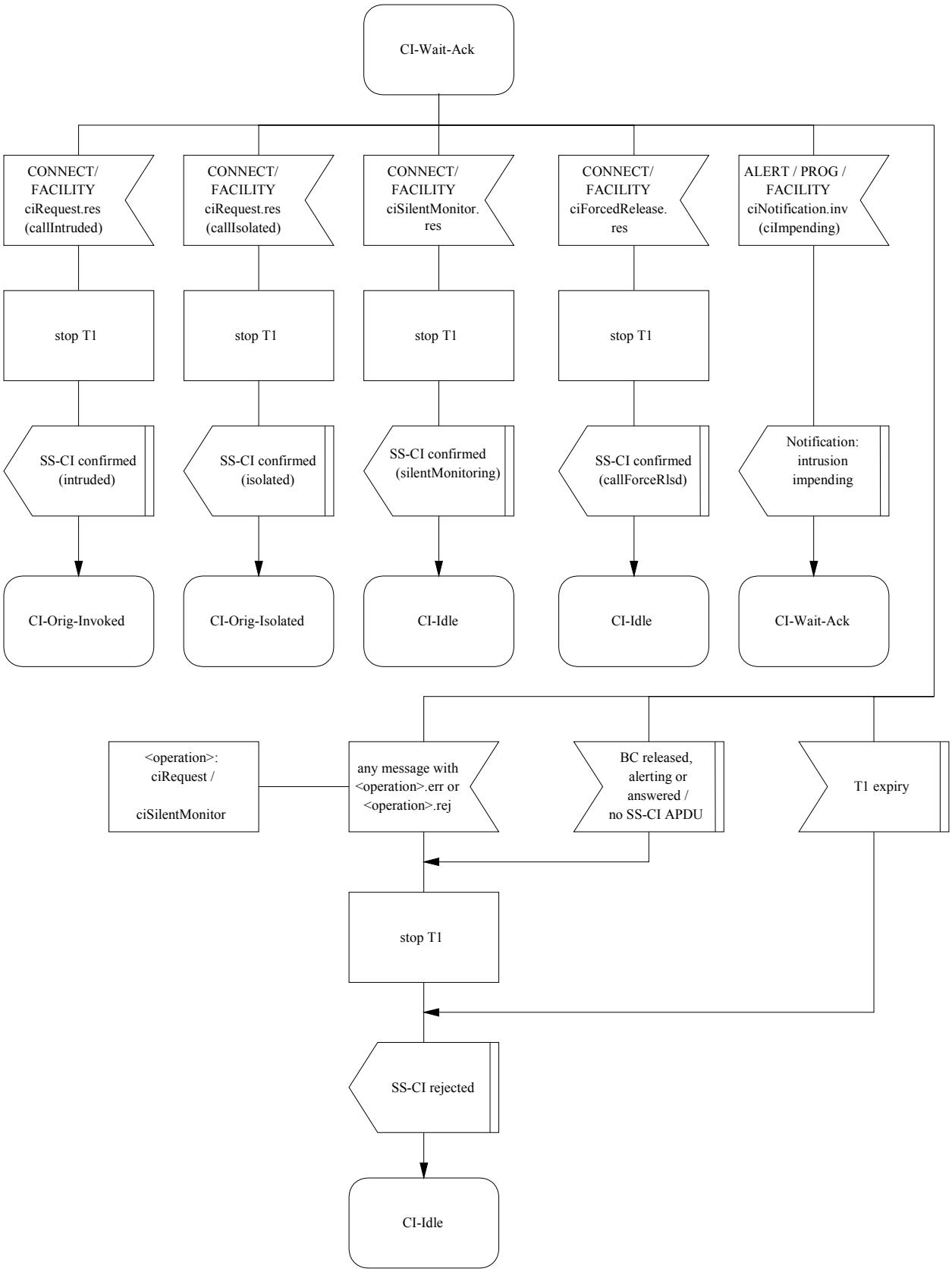


Figure 20/H.450.11 – Endpoint A SDL *(sheet 2 of 5)*

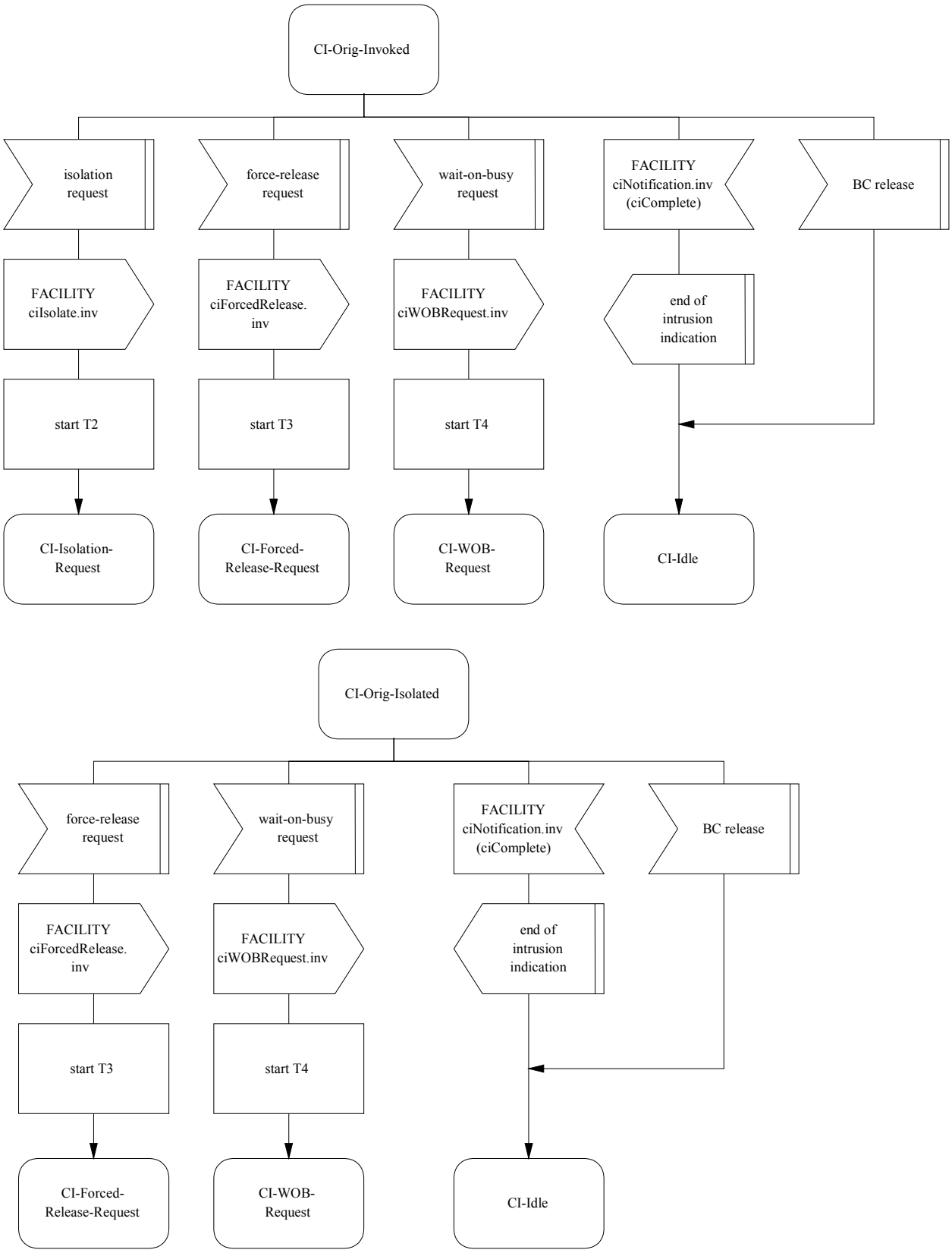


Figure 21/H.450.11 – Endpoint A SDL *(sheet 3 of 5)*

Figure 22/H.450.11 – Endpoint A SDL *(sheet 4 of 5)*

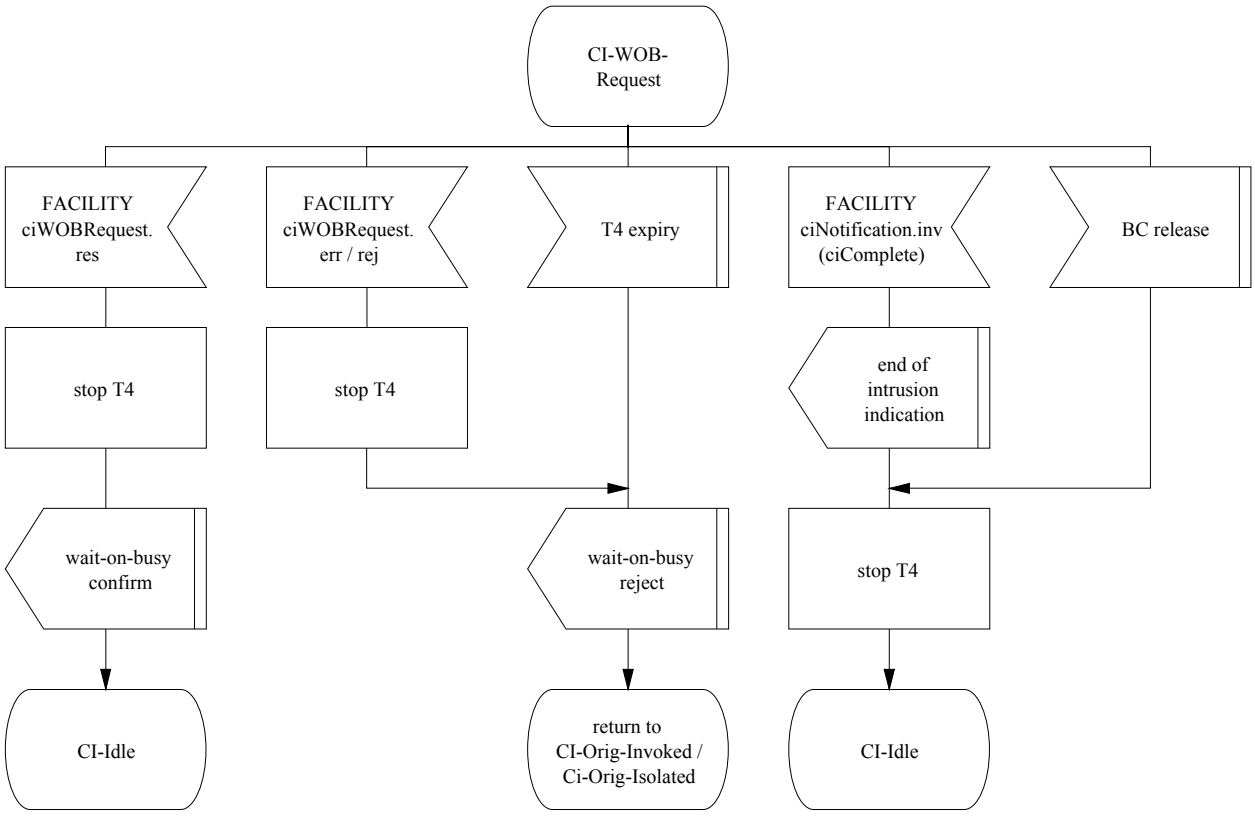


Figure 23/H.450.11 – Endpoint A SDL *(sheet 5 of 5)*

12.2 Behaviour of user B's endpoint

Figures 24 through 28 show the behaviour of user B's endpoint.

Input signals from the right and output signals to the right represent:

- messages from or to the unwanted user's peer SS-Control entity (i.e. in user C's endpoint) which carry SS-CI control information;
- primitives from or to basic call control with relation to the established call; these primitives are indicated by "BC".

Input signals from the left and output signals to the left represent:

- messages from or to the calling peer SS-Control entity (i.e. in user A's endpoint) which carry SS-CI control information;
- primitives from or to basic call control with relation to the intruding call; these primitives are indicated by "BC";
- internal signals, e.g. timer expiry.

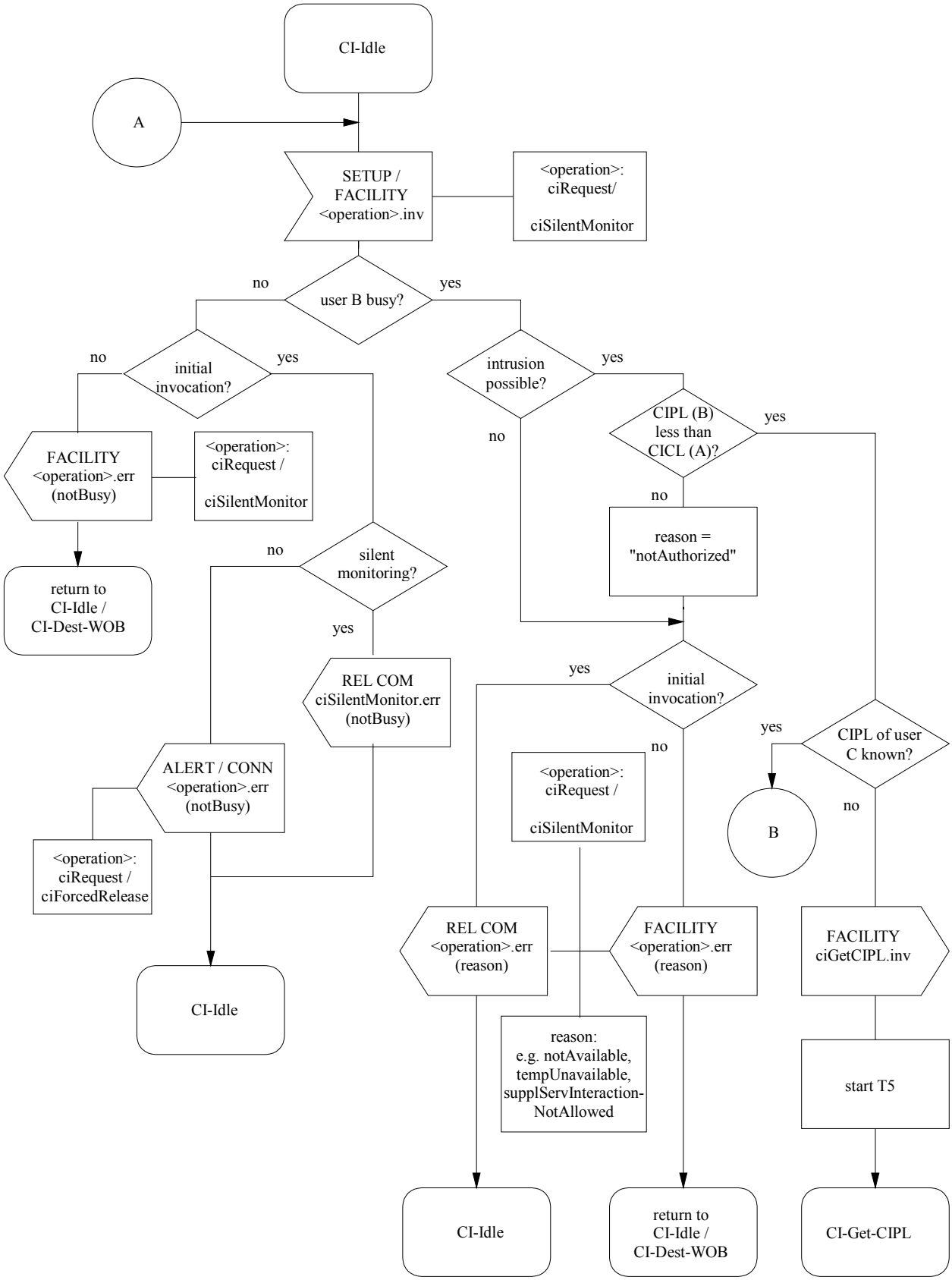


Figure 24/H.450.11 – Endpoint B SDL *(sheet 1 of 5)*

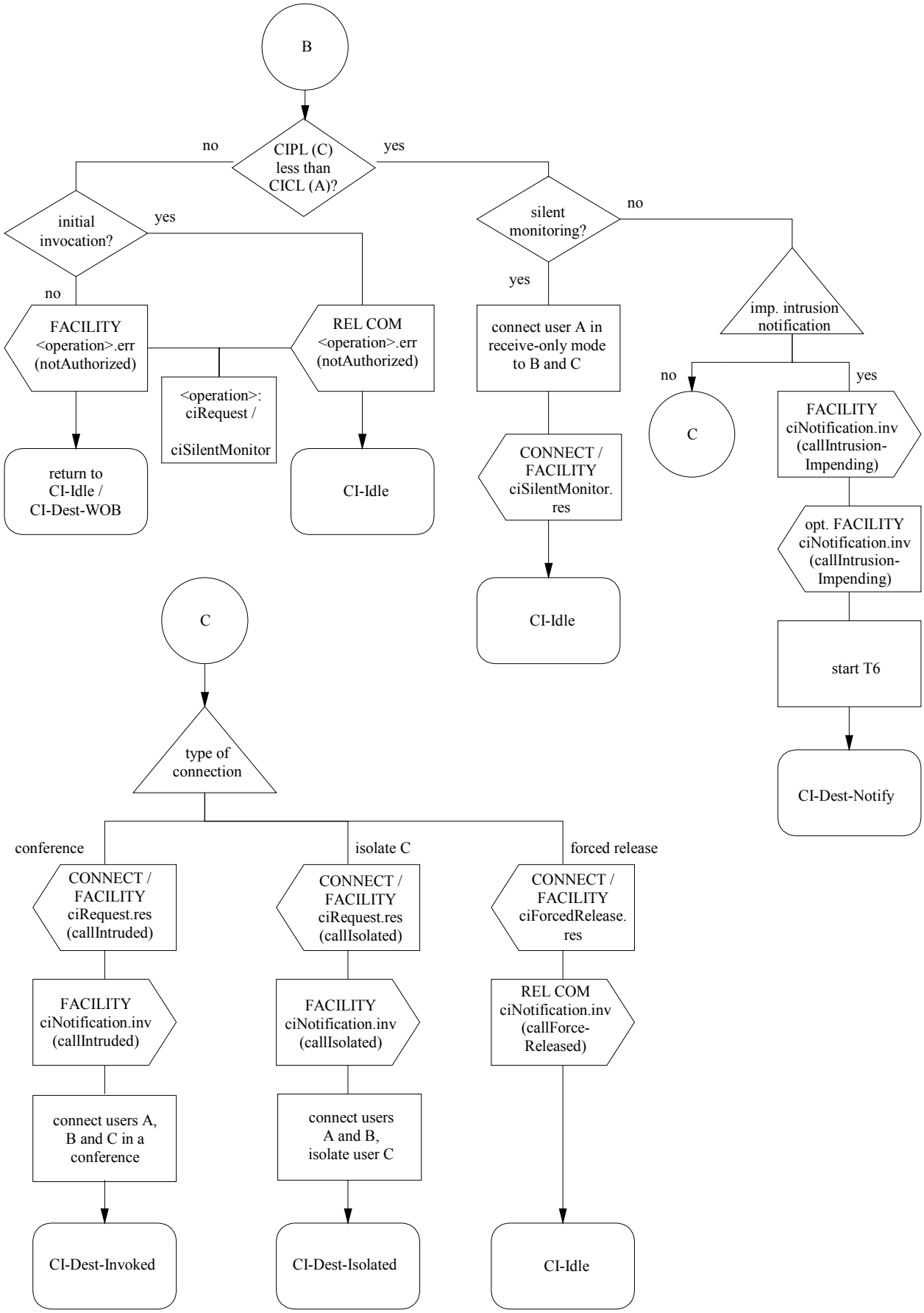


Figure 25/H.450.11 – Endpoint B SDL *(sheet 2 of 5)*

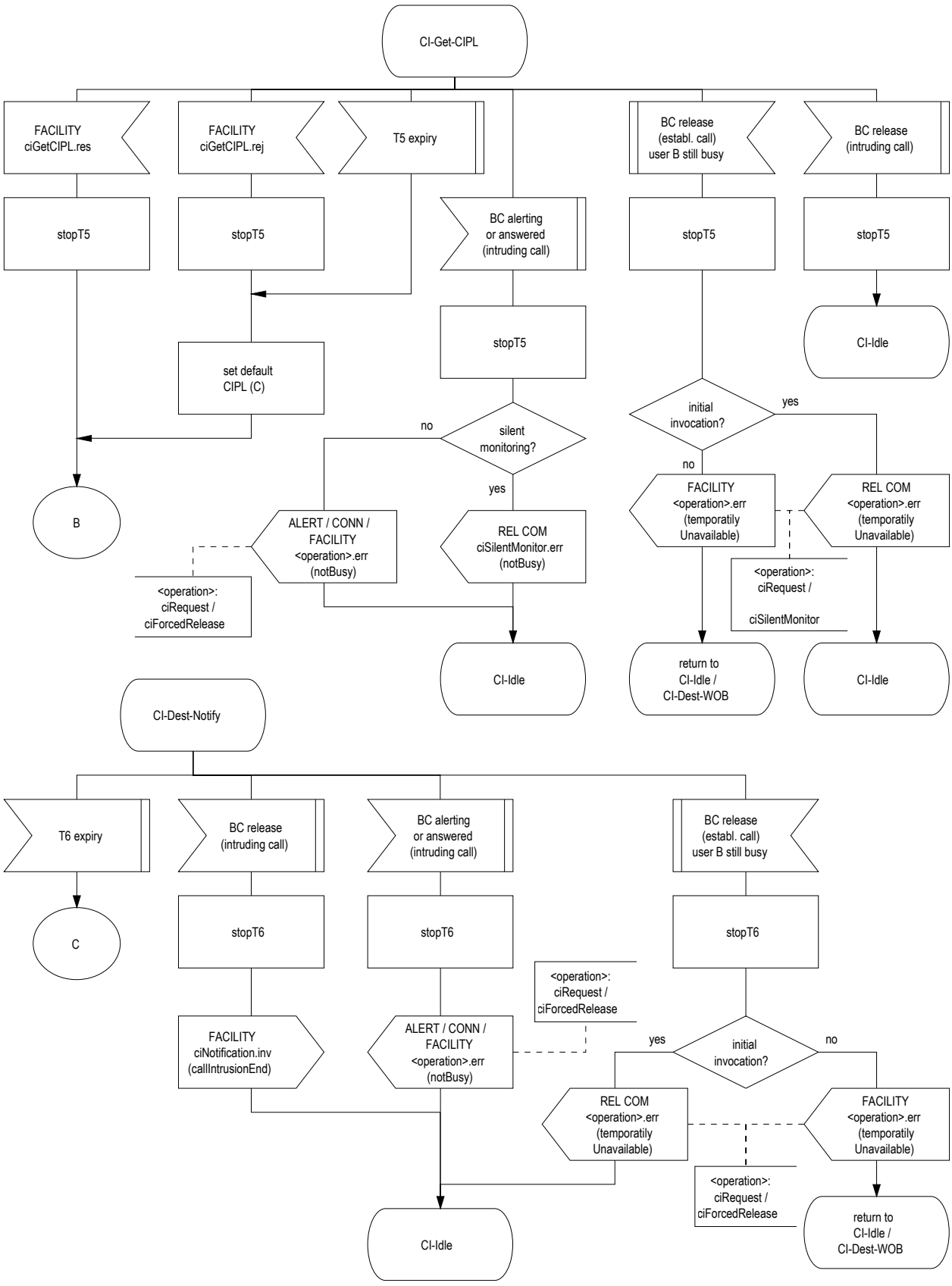


Figure 26/H.450.11 – Endpoint B SDL *(sheet 3 of 5)*

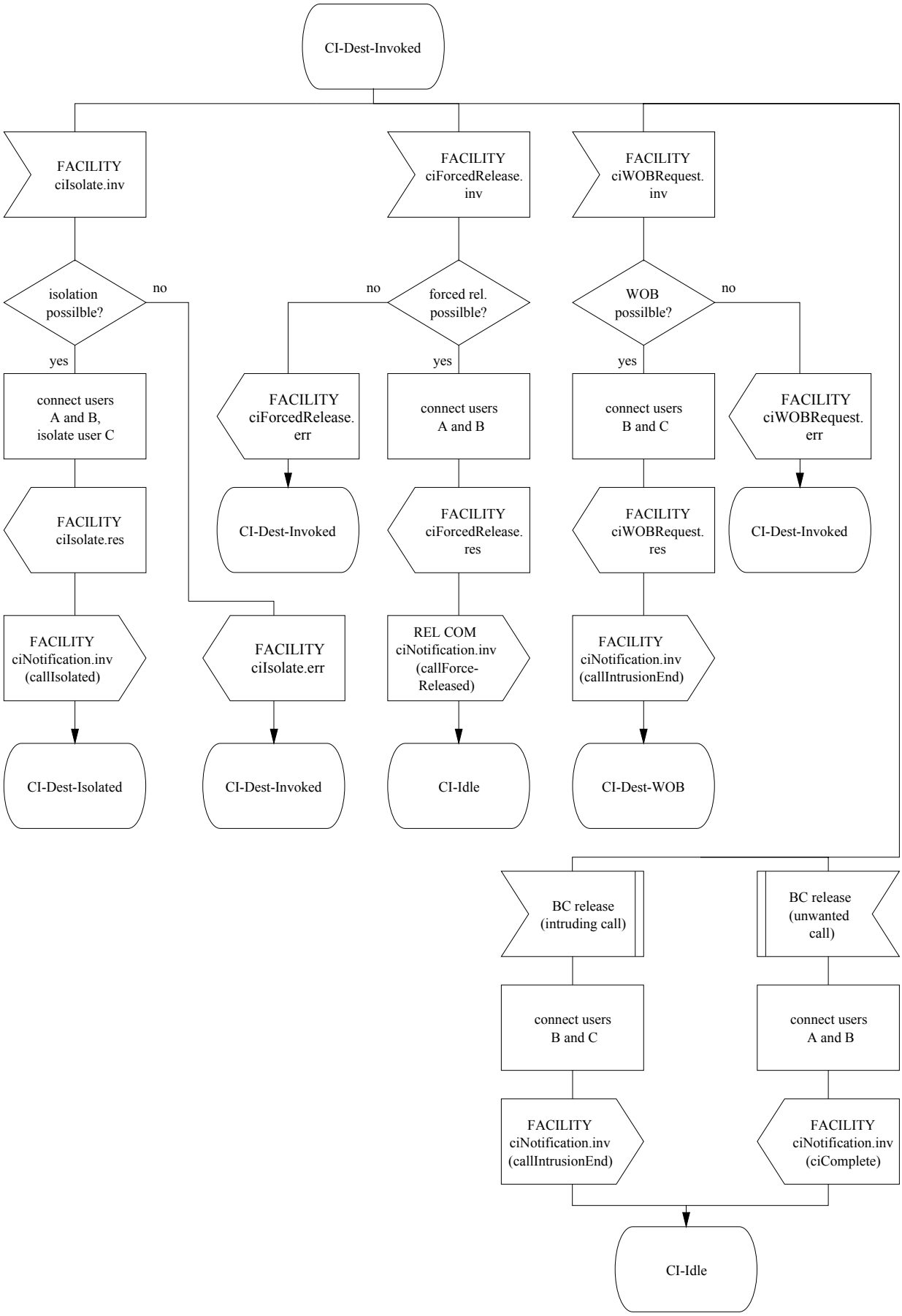


Figure 27/H.450.11 – Endpoint B SDL *(sheet 4 of 5)*

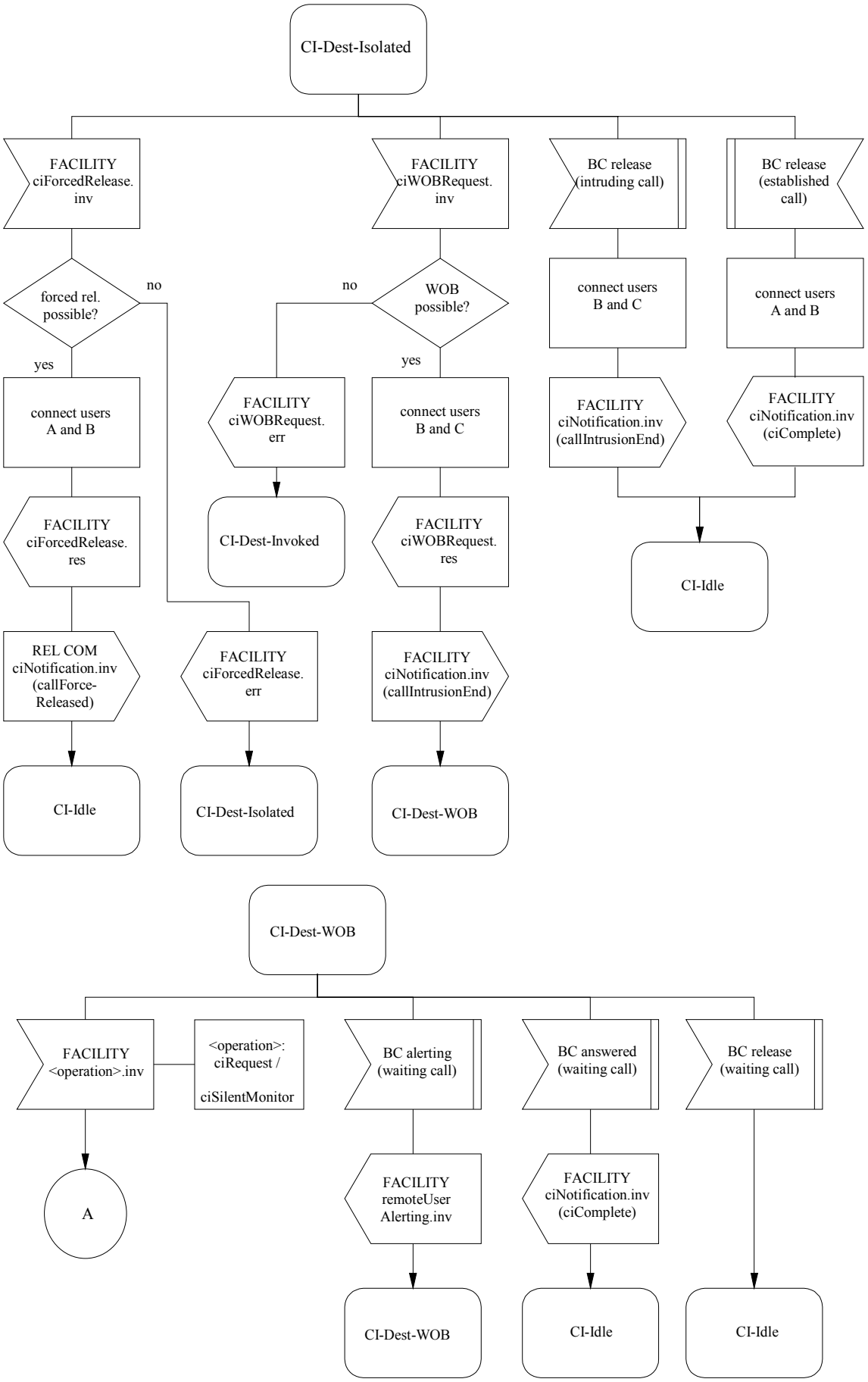


Figure 28/H.450.11 – Endpoint B SDL *(sheet 5 of 5)*

12.3 Behaviour of user C's endpoint

Figure 29 shows the behaviour of user C's endpoint.

Input signals from the left and output signals to the left represent messages from or to the peer SS-Control entity (i.e. in user B's endpoint) which carry SS-CI control information.

Output signals to the right represent primitives to the unwanted user (user C).

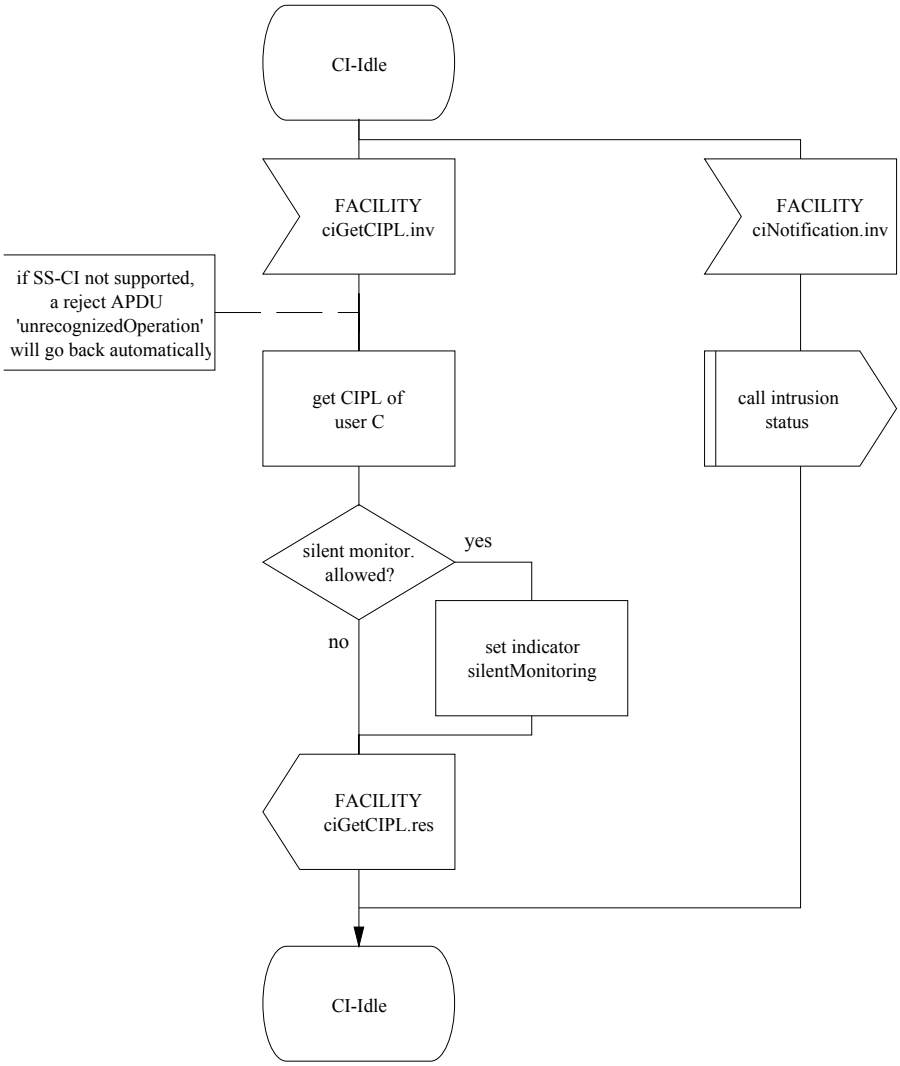


Figure 29/H.450.11 – Endpoint C SDL

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	**Audiovisual and multimedia systems**
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure and Internet protocol aspects
Series Z	Languages and general software aspects for telecommunication systems

Geneva, 2001