INTERNATIONAL TELECOMMUNICATION UNION

# ITU-T H.450.9

(11/2000)

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Supplementary services for multimedia

# Call completion supplementary services for H.323

ITU-T Recommendation H.450.9

(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.450.9**

# Call completion supplementary services for H.323


## Summary

This new Recommendation describes the procedures and the signalling protocol for the Call Completion supplementary services in ITU-T H.323.

The Call Completion supplementary services are Completion of Calls to Busy Subscribers (SS-CCBS) and Completion of Calls on No Reply (SS-CCNR).

SS-CCBS enables a calling User A, encountering a busy destination User B, to have the call completed when User B becomes not busy, without having to make a new call attempt.

SS-CCNR enables a calling User A, encountering a destination User B that, though alerted, does not answer, to have the call completed when User B becomes not busy again after a period of activity, without having to make a new call attempt.

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 Completion supplementary service specified in ISO/IEC 13866 and ISO/IEC 13870.



## Source

ITU-T Recommendation H.450.9 was prepared by ITU-T Study Group 16 (2001-2004) and approved under the WTSA Resolution 1 procedure on 17 November 2000.

## 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.450.9**

# Call completion supplementary services for H.323

## 1 Scope

This Recommendation describes supplementary services Call Completion (SS-CC), which are applicable to various basic services supported by H.323 multimedia endpoints.

SS-CC consists of two supplementary services: the Completion of Calls to Busy Subscribers supplementary service (SS-CCBS) and the Completion of Calls on No Reply supplementary service (SS-CCNR). SS-CCBS allows completion of a call to a subscriber (i.e. user) that was unsuccessful because of a busy condition and SS-CCNR allows completion of a call to a user that was unsuccessful because the user, although alerted, did not answer.

## 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.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*.

## 3 Terms and Definitions

This Recommendation defines the following terms.

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

**3.2** **call completion**: The successful presentation of a previously unsuccessful call to a destination user (User B), which occurs when the call has entered an alerting phase or has been answered.

**3.3** **CC call**: The re-initiation, in the course of executing a CC Request, of the previously unsuccessful call from User A's endpoint to User B on behalf of User A.

**3.4** **CC recall**: An indication informing User A that User B is no longer busy (in the case of SS-CCBS) or has just completed a period of activity (in the case of SS-CCNR). Acceptance of this indication by User A will cause the call to be completed.

**3.5** **CC request**: An instance of SS-CCBS or SS-CCNR.

**3.6** **connection release/retain**: The capability to release/retain the call independent signalling connection during the various phases of a CC Request.

**3.7** **endpoint; gatekeeper; gateway; terminal; user**: See ITU-T H.323.

**3.8** **free**: A property of a user who can accept an attempt to present a call to that user (i.e. allow the call to reach the alerting or answered state).

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

**3.10** **recall timer**: This timer specifies the length of time that User A's endpoint shall wait for a response from User A to a CC Recall.

**3.11** **service duration timer**: This timer specifies the length of time that the SS-CC supplementary service shall be active within User A's endpoint.

**3.12** **service retention**: The optional capability to continue with a CC Request after the CC Call failed due to User B being busy again.

**3.13** **user A**: Originating user that originated the call and requested the supplementary service.

**3.14** **user B**: The remote user that was initially addressed in the original call set up.

## 4 Abbreviations and Acronyms

This Recommendation uses the following abbreviations:

| | |
|---|---|
| APDU | Application Protocol Data Unit |
| ASN.1 | Abstract Syntax Notation One |
| CC | Call Completion |
| CCBS | Completion of Calls to Busy Subscribers |
| CCNR | Completion of Calls on No Reply |
| CT | Call Transfer |
| EASE | Endpoint A Signalling Entity |
| EBSE | Endpoint B Signalling Entity |
| ID | Identification, Identifier |
| IE | Information Element |
| NFE | Network Facility Extension |
| SCN | Switched Circuit Network |
| SDL | Specification and Description Language |
| SS-CC | Supplementary Service Call Completion (This is a generic term, used to describe aspects common to both SS-CCBS and SS-CCNR.) |
| SS-CCBS | Supplementary Service Completion of Calls to Busy Subscribers |
| SS-CCNR | Supplementary Service Completion of Calls on No Reply |

## 5 SS-CCBS service description

Completion of Calls to Busy Subscribers (SS-CCBS) is a supplementary service that is offered to a calling User A. On encountering a busy called User B, it allows User A to request that User B's endpoint monitor User B and notify User A's endpoint when User B becomes free. On response by User A to that notification, User A's endpoint shall attempt to complete the call to User B.

## 5.1 Normal procedures

### 5.1.1 Activation/deactivation/registration/interrogation

SS-CCBS is permanently activated.

### 5.1.2 Invocation and operation

When a call from User A fails because the destination User, User B, is busy, User A shall be able to request SS-CCBS.

On receipt of a request for SS-CCBS, User B's endpoint shall check whether it is possible to initiate the service and if so, shall send an acknowledgment to User A's endpoint and start monitoring User B. On receipt of the acknowledgment, User A's endpoint shall start the SS-CCBS service duration timer. The acknowledgment means that User A can expect to receive a CC Recall if User B becomes free within the period of the SS-CCBS service duration timer.

NOTE 1 – The fact that User B is already being monitored, as a result of a call completion request from another user, need not cause rejection of the request from User A. The handling of multiple requests against the same User B is an implementation matter; typically involving some sort of queue arranged in chronological or priority order.

NOTE 2 – The fact that User A has already invoked call completion against another user need not cause rejection of any further requests for call completion from User A. Handling of multiple requests by the same User A is an implementation matter.

NOTE 3 – SS-CCBS can be invoked after notification that User B is busy. The recommended value of the SS-CCBS service duration timer is in the range of 1-60 minutes. A suitable default value could be 15 minutes.

NOTE 4 – The selective operation of SS-CCBS on calls associated with a specific basic service is for further study.

After the SS-CCBS request has been acknowledged, User A shall be able to receive and initiate other calls.

User A's endpoint may provide User A with the ability to request a list of outstanding CC Requests that User A has invoked. The list will be empty if there are no outstanding CC Requests.

When the monitoring of User B indicates that User B has become free, User B's endpoint shall provide a CC Recall. User A's endpoint shall notify User A of the CC Recall and start the Recall timer (the value of this timer is in the range of 10-30 seconds).

NOTE 5 – On receipt of such a notification, User A can:

- accept the CC Recall;
- ignore the notification, thereby causing the CC Recall to be cancelled by User A's endpoint when the CC Recall timer expires; or
- cancel the SS-CCBS request.

If User A accepts the CC Recall, User A's endpoint shall stop the Recall timer and attempt to complete a call between User A and User B. If the call is successfully presented to User B and enters an alerting phase or is answered, SS-CCBS shall be regarded as complete.

### 5.1.3 Cancellation

User A's endpoint shall provide User A with the ability to request cancellation of at least one of the following:

- all outstanding CC Requests for which a CC Recall is still expected;
- the most recent CC Request for which a CC Recall is still expected; and
- a specific CC Request for which a CC Recall is still expected.

User A shall be informed of a successful cancellation.

## 5.2 Exceptional procedures

### 5.2.1 Invocation and operation

#### 5.2.1.1 Rejection of SS-CCBS Service request

If User A is not permitted to request SS-CCBS, User A's endpoint shall reject the SS-CCBS request with an indication of whether denial is short or long term.

Short-term denial shall be used for temporary conditions where a later request for SS-CCBS might be successful. Examples of conditions that may result in a short-term denial are:

- limit of requests by User A already reached;
- limit of requests against User B already reached; or
- duplicate request (see 5.2.1.4).

Long term denial shall be used when later requests will also be rejected. Examples of conditions that may result in a long term denial are:

- SS-CCBS not provided to User A;
- interworking with a network that does not support SS-CCBS; or
- SS-CCBS not allowed against User B (Note).

NOTE – This is an implementation option that may apply to certain classes of users.

#### 5.2.1.2 User A is busy on CC Recall

If User A is found to be busy when User B becomes free, User A's endpoint shall suspend the CC Request either immediately or if User A is still busy after having provided User A notification of the CC Recall and having waited for a short time for User A to become not busy before acting on the CC Recall (i.e. before initiating a new call to User B).

NOTE – On receipt of such a notification, User A can:

- ignore the notification, thereby causing suspension of the CC Request;
- cancel the SS-CCBS request; or
- free resources (e.g. by disposing of an existing call), thereby allowing the CC Recall to proceed.

After having suspended SS-CC, when User A becomes free User A's endpoint shall either resume SS-CC, thereby causing User B to be monitored again, or attempt to complete a call as described in 5.1.2.

#### 5.2.1.3 User B becomes busy after successful CC Recall

If User B is busy for the call resulting from a successful CC Recall of User A, User B's endpoint shall either:

- abandon SS-CCBS, indicating the reason for the failure to User A and that CC Request has been cancelled. The endpoint A may allow, as an implementation option, the re-invocation of SS-CCBS if call completion to User B is still required;
- resume monitoring of User B, indicating the reason for the failure to User A and that the CC Request has been maintained. In such a case, User A may, as a user option, request cancellation of the CC Request, if call completion to User B is no longer required.

#### 5.2.1.4 Duplicate SS-CCBS requests

If User A has already requested SS-CCBS on User B and is awaiting a recall, any subsequent request from User A to invoke SS-CCBS on User B shall cause User B's endpoint to reject the request as a duplicate request.

#### 5.2.1.5 Other failure situations

A particular request for the service shall be automatically cancelled by User A's endpoint and User A shall be notified if:

- User B is still busy after the SS-CCBS service duration timer expires; or
- User A does not accept the CC Recall before the Recall timer expires.

### 5.2.2 Cancellation

A cancellation request shall be rejected if there are no CC Requests for User A or if the request is to cancel a specific CC Request that does not exist.

## 5.3 Interactions with other supplementary services

### 5.3.1 Completion of Calls on No Reply (SS-CCNR)

If User A has SS-CCNR activated on User B, and User A requests SS-CCBS on User B, this request shall be treated as a duplicate SS-CCBS request in accordance with 5.2.1.4.

NOTE – When User B is busy (the pre-requisite for invocation of SS-CCBS by User A) before a CC Recall has been started relating to a previous SS-CCNR request, the pending SS-CCNR request has effectively become an SS-CCBS request, as it is awaiting a free User B in order to recall User A. If an SS-CCBS request is then received from User A, relating to User B, this is therefore effectively a duplicate SS-CCBS request and is treated as such by User B's endpoint.

### 5.3.2 Call Transfer (SS-CT)

No interaction.

### 5.3.3 Call Forwarding Unconditional (SS-CFU)

a) SS-CFU activated by User B before User A requests SS-CCBS:

   If the call to User B is diverted to User C by SS-CFU and User C is busy, then an SS-CCBS request from User A shall be applied to the diverted-to User C.

b) SS-CFU activated by User B after User A requests SS-CCBS:

   If User B activates SS-CFU after User A has requested SS-CCBS and while the CC Recall has not yet been accepted by User A, the SS-CCBS request shall continue to be applied to User B.

c) SS-CFU activation by User A:

   If User A invokes SS-CCBS while SS-CFU is activated, or User A invokes SS-CCBS and subsequently activates SS-CFU, this shall not affect the provision of the CC Recall to User A.

### 5.3.4 Call Forwarding Busy (SS-CFB)

a) SS-CFB activated by User B before User A requests SS-CCBS:

   If the call from User A to User B is diverted to User C by SS-CFB and User C is busy, then an SS-CCBS request made by User A shall be applied to User B or to the diverted-to User C.

b) SS-CFB activated by User B after User A requests SS-CCBS:

   If User B activates SS-CFB after User A has requested SS-CCBS and while the CC Recall has not yet been accepted by User A, the SS-CCBS request shall continue to be applied to User B.

c) SS-CFB activation by User A:

If User A invokes SS-CCBS while SS-CFB is activated, or User A invokes SS-CCBS and subsequently activates SS-CFB, this shall not affect the provision of the CC Recall to User A.

### 5.3.5 Call Forwarding on No Reply (SS-CFNR)

a) SS-CFNR activated by User B before User A requests SS-CCBS:

If the call from User A to User B is diverted to User C by SS-CFNR and User C is busy, then an SS-CCBS request made by User A shall be applied to the diverted-to User C.

b) SS-CFNR activated by User B after User A requests SS-CCBS:

If User B activates SS-CFNR after User A has requested SS-CCBS and while the CC Recall has not yet been accepted by User A, then the SS-CCBS request shall continue to be applied to User B.

NOTE – In such a case, the call resulting from successful completion of SS-CCBS can be subject to SS-CFNR if not answered.

c) SS-CFNR activation by User A:

If User A invokes SS-CCBS while SS-CFNR is activated, or User A invokes SS-CCBS and subsequently activates SS-CFNR, this shall not affect the provision of the CC Recall to User A.

### 5.3.6 Call Hold

No interaction.

### 5.3.7 Call Park/Call Pickup

A CC Call cannot be picked-up by another user, even if in the same Pickup Group.

### 5.3.8 Call Waiting

SS-CCBS cannot occur, as a waiting call is not considered to be busy.

### 5.3.9 Message Waiting indication

No interaction.

### 5.3.10 Name Presentation

No interaction.

### 5.3.11 Call Linkage

If available, all calls and signalling connections related to a CC Request shall use the same Thread ID, but different Global Call IDs. Thus the Thread ID can be used to associate:

- the original call attempt which failed because of User B being busy;
- the SS-CCBS request;
- the CC Recall;
- the CC Call.

If the original call attempt did not contain a Thread ID, its Call Identifier may be used as Thread ID for the SS-CCBS request, CC Recall and CC Call.

# 6 SS-CCNR service description

Completion of Calls on No Reply (SS-CCNR) is a supplementary service that is offered to a calling User A. On encountering a called User B that does not answer, it allows User A to request that User B's endpoint monitors User B and notifies User A when User B becomes free after a subsequent period of activity. On response by User A to that notification, User A's endpoint shall attempt to complete the call to User B.

NOTE 1 – User Activities that constitute a subsequent period of activity are implementation specific and beyond the scope of this Recommendation.

NOTE 2 – SS-CCNR may also apply if the User B is not available (e.g. not registered). This requires a gatekeeper or a proxy that acts on behalf of the User B for SS-CCNR.

## 6.1 Normal procedures

### 6.1.1 Activation/deactivation/registration/interrogation

SS-CCNR is permanently activated.

### 6.1.2 Invocation and operation

When a call from User A fails because the destination user, User B, does not answer, User A shall be able to request SS-CCNR. User A shall also be able to request SS-CCNR while User B is alerting.

On receipt of a request for SS-CCNR, User B's endpoint shall check whether it is possible to initiate the service and if so, shall send an acknowledgment to User A and start monitoring User B. On receipt of the acknowledgment, User A's endpoint shall start the SS-CCNR service duration timer. The acknowledgment means that User A can expect to receive a CC Recall if User B becomes free, after a subsequent period of activity, within the period of the SS-CCNR service duration timer.

NOTE 1 – The fact that User B is already being monitored, as a result of a call completion request from another user, need not cause rejection of the request from User A. The handling of multiple requests against the same User B is an implementation matter; typically involving some sort of queue arranged in chronological or priority order.

NOTE 2 – The fact that User A has already invoked call completion against another user need not cause rejection of any further requests for call completion from User A. The handling of multiple requests by the same User A is an implementation matter.

NOTE 3 – SS-CCNR can be invoked after notification that User B is alerting. The recommended value of the SS-CCNR service duration timer is in the range of 1-1440 minutes (i.e. 24 hours). A suitable default value could be 60 minutes.

NOTE 4 – The selective operation of SS-CCNR on calls associated with a specific basic service is for further study.

After the SS-CCNR request has been acknowledged, User A shall be able to receive and initiate other calls.

User A's endpoint may provide User A with the ability to request a list of outstanding CC Requests that User A has invoked. The list shall be empty if there are no outstanding CC Requests.

When the monitoring of User B indicates that User B has become free after a period of activity, User B's endpoint shall provide a CC Recall. User A's endpoint shall notify User A of the CC Recall and start the Recall timer.

If User A accepts the CC Recall, User A's endpoint shall attempt to complete a call between User A and User B. If the call is successfully presented to User B and enters an alerting phase or is answered, SS-CCNR shall be regarded as complete.

### 6.1.3 Cancellation

Clause 5.1.3 shall apply.

## 6.2 Exceptional procedures

### 6.2.1 Invocation and operation

#### 6.2.1.1 Rejection of SS-CCNR service request

Clause 5.2.1.1 shall apply, with the exception that "SS-CCBS" shall be replaced by "SS-CCNR".

#### 6.2.1.2 User A is busy on CC Recall

If User A is found to be busy when User B becomes free after a period of activity, User A's endpoint shall suspend the CC Request either immediately or if User A is still busy after having provided User A notification of the CC Recall and having waited for a short time for User A to become not busy before acting on the CC Recall (i.e. before initiating a new call to User B).

NOTE – On receipt of such a notification, User A can:

- ignore the notification, thereby causing suspension of the CC Request;
- cancel the SS-CCBS request; or
- free resources (e.g. by disposing of an existing call), thereby allowing the CC Recall to proceed.

After having suspended SS-CC, when User A becomes free User A's endpoint shall either resume SS-CC, thereby causing User B to be monitored again, or attempt to complete a call as described in 6.1.2.

#### 6.2.1.3 User B becomes busy after successful CC Recall

If User B is busy for the call resulting from a successful CC Recall of User A, User B's endpoint shall either:

- abandon SS-CCNR indicating the reason for the failure to User A and that the CC Request has been cancelled. In such a case User A's endpoint shall allow User A the possibility of invoking SS-CCBS if call completion to User B is still required;
- automatically invoke SS-CCBS monitoring of User B, indicating the reason for the failure to User A and that SS-CCBS has been invoked. In such a case, User A may request cancellation of the SS-CCBS request if call completion to User B is no longer required.

#### 6.2.1.4 Duplicate SS-CCNR requests

Clause 5.2.1.4 shall apply, with the exception that "SS-CCBS" shall be replaced by "SS-CCNR".

#### 6.2.1.5 Other failure situations

A particular request for the service shall be automatically cancelled by User A's endpoint and User A shall be notified if:

- User B has still had no period of activity before the SS-CCNR service duration timer expires;
- User B is still busy (after a period of activity by User B) when the SS-CCNR service duration timer expires;
- User A does not accept the CC Recall before the Recall timer expires.

### 6.2.2 Cancellation

Clause 5.2.2 shall apply.

## 6.3 Interactions with other supplementary services

### 6.3.1 Completion of Calls to Busy Subscribers (SS-CCBS)

Clause 5.3.1 shall apply.

### 6.3.2 Call Transfer (SS-CT)

No interaction.

### 6.3.3 Call Forwarding Unconditional (SS-CFU)

a) SS-CFU activated by User B before User A requests SS-CCNR:

If the call to User B is diverted to User C by SS-CFU and User C does not answer, then an SS-CCNR request from User A shall be applied to the diverted-to User C.

b) SS-CFU activated by User B after User A requests SS-CCNR:

If User B activates SS-CFU after User A has requested SS-CCNR and while the CC Recall has not yet been accepted by User A, the SS-CCNR request shall continue to be applied to User B.

c) SS-CFU activation by User A:

If User A invokes SS-CCNR while SS-CFU is activated, or User A invokes SS-CCNR and subsequently activates SS-CFU, this shall not affect the provision of the CC Recall to User A.

### 6.3.4 Call Forwarding Busy (SS-CFB)

a) SS-CFB activated by User B before User A requests SS-CCNR:

If the call from User A to User B is diverted to User C by SS-CFB and User C does not answer, then an SS-CCNR request made by User A shall be applied to the diverted-to User C.

b) SS-CFB activated by User B after User A requests SS-CCNR:

If User B activates SS-CFB after User A has requested SS-CCNR and while the CC Recall has not yet been accepted by User A, the SS-CCNR request shall continue to be applied to User B.

c) SS-CFB activation by User A:

If User A invokes SS-CCNR while SS-CFB is activated, or User A invokes SS-CCNR and subsequently activates SS-CFB, this shall not affect the provision of the CC Recall to User A.

### 6.3.5 Call Forwarding on No Reply (SS-CFNR)

a) SS-CFNR activated by User B before User A requests SS-CCNR:

If the call from User A to User B is diverted to User C by SS-CFNR and User C does not answer, then an SS-CCNR request made by User A shall be applied to User B or to the diverted-to User C.

b) SS-CFNR activated by User B after User A requests SS-CCNR:

If User B activates SS-CFNR after User A has requested SS-CCNR and while the CC Recall has not yet been accepted by User A, then the SS-CCNR request shall continue to be applied to User B.

c) SS-CFNR activation by User A:

If User A invokes SS-CCNR while SS-CFNR is activated, or User A invokes SS-CCNR and subsequently activates SS-CFNR, this shall not affect the provision of the CC Recall to User A.

### 6.3.6 Call Hold

No interaction.

### 6.3.7 Call Park/Call Pickup

Clause 5.3.7 shall apply.

### 6.3.8 Call Waiting

Invocation of SS-CCNR shall terminate Call Waiting for the corresponding call.

### 6.3.9 Message Waiting Indication

No interaction.

### 6.3.10 Name Presentation

No interaction.

### 6.3.11 Call Linkage

If available, all calls and signalling connections related to a CC Request shall use the same Thread ID, but different Global Call IDs. Thus the Thread ID can be used to associate:

- the original call attempt which was not answered by User B;
- the SS-CCNR request;
- the CC Recall;
- the CC Call.

If the original call attempt did not contain a Thread ID, its Call Identifier may be used as Thread ID for the SS-CCNR request, CC Recall and CC Call.

## 7 Messages and information elements

The APDUs of the operations defined in clause 12 shall be conveyed within H.450.1 Supplementary Service APDUs included in User-user information elements, as specified in ITU-T H.450.1.

The operations defined in Abstract Syntax Notation One (ASN.1) in clause 12 shall apply.

When conveying the invoke APDU of the operations defined in clause 12, the *destinationEntity* data element of the NFE shall contain the value *endpoint*.

When conveying the invoke APDU of the operations defined in clause 12, the Interpretation APDU shall either be omitted or included with the value *rejectAnyUnrecognizedInvokePdu*.

## 8 Procedures

The following procedures are a combination of call related and call independent signalling. The latter uses the call-independent procedures as defined in 6.2/H.450.1, which provides call independent signalling connections for supplementary service control. For SS-CC, the call independent signalling connection shall be maintained until completion or cancellation of SS-CC unless the connection release option is applied, in which case the call independent signalling connection shall be released during idle phases of SS-CC signalling and re-established if further SS-CC control information is to be sent. Once chosen, the method (connection release or connection retain) shall apply throughout the lifetime of this CC Request, except where dictated by interworking considerations.

All SS-CC control information is exchanged between User A's endpoint and User B's endpoint.

There are two possible behaviours when User B is found to be busy again after User A responds to a CC Recall:

a) service retention method – the CC Request remains in force at User A's and User B's endpoints and User B's endpoint commences the monitoring of User B again;

b) service cancellation method – the CC Request is cancelled at User A's and User B's endpoints.

## 8.1 Actions at User A's endpoint

### 8.1.1 Normal procedures

#### 8.1.1.1 CCBS invocation

A "Busy" condition occurs when a *ReleaseCompleteReason* of *inConf* or a Cause IE with cause value #17, "User Busy" is received from the called endpoint in a Release Complete message.

If User A requests SS-CCBS and the request is acceptable to User A's endpoint, then User A's endpoint shall send a *ccbsRequest* invoke APDU to User B's endpoint according to 8.1.1.3.

NOTE – SS-CCBS can be requested after a call attempt has encountered a busy condition at User B. The release of the original call is beyond the scope of this Recommendation.

Upon receipt of a *ccbsRequest* return result APDU, User A's endpoint shall start the service duration timer T2 with a value appropriate for SS-CCBS (see 11.4.1), and wait for an indication that User B has become not busy.

#### 8.1.1.2 CCNR invocation

If User A requests SS-CCNR and the request is acceptable to User A's endpoint, then User A's endpoint shall send a *ccnrRequest* invoke APDU to User B's endpoint according to 8.1.1.3.

NOTE – SS-CCNR can be requested after a call to User B has remained unanswered. The release of the original call is beyond the scope of this Recommendation.

Upon receipt of a *ccnrRequest* return result APDU, User A's endpoint shall start the service duration timer T2 with a value appropriate for SS-CCNR (see 11.4.1), and wait for an indication that User B has become not busy after a subsequent period of activity.

#### 8.1.1.3 SS-CC invocation – detailed procedure

The *ccbsRequest* or *ccnrRequest* invoke APDU shall be sent to User B's endpoint in a Setup message of a new call independent signalling connection, established in accordance with the procedures as specified in ITU-T H.450.1.

The following information shall be included in the argument of the *ccbsRequest* or *ccnrRequest* invoke APDU:

- basic call information from the original call:
  - the address of User A, or an indication that it is not available or restricted, in element *numberA*;
  - the address of User B, in element *numberB*;
  - the intended basic service type, in element *service*;
  - element *ccIdentifier* (recommended; if present the value shall reflect the CallIdentifier value of the previous unsuccessful basic call attempt);
- element *can-retain-service*; its value shall be set to *TRUE* if endpoint A is able to use the service retention method, otherwise the value shall be set to *FALSE*;

- optionally element *retain-sig-connection*, with value *TRUE* if endpoint A requests to retain the signalling connection throughout the lifetime of this CC Request, and with value *FALSE* if endpoint A requests to release the signalling connection during idle phases of SS-CC signalling; if not included, endpoint A has no preference.

After sending the *ccbsRequest/ccnrRequest* invoke APDU, User A's endpoint shall start timer T1 and enter state CC-Wait-Ack.

Upon receipt of a *ccbsRequest* or *ccnrRequest* return result APDU in a Connect or Release Complete message, User A's endpoint shall stop timer T1, store the data related to the CC Request, and enter state CC-Invoked-User-A-RET (if the APDU was received in a Connect message) or CC-Invoked-User-A-RLS (if the APDU was received in a Release Complete message). If the result contains element *retain-service* with value TRUE and element *can-retain-service* was sent with value TRUE in the argument of the corresponding invoke APDU, User A's endpoint shall record the fact that the service retention method is to be used; otherwise, it shall record the fact that the service cancellation method is to be used.

#### 8.1.1.4 CC Call when User A not busy

If a *ccExecPossible* invoke APDU is received in a Facility message on the call independent signalling connection while in state CC-Invoked-User-A-RET and User A is not busy, User A's endpoint shall indicate the CC Recall to User A, start the recall timer T3 and enter state CC-Wait-User-A-Answer.

If a *ccExecPossible* invoke APDU is received in a Setup message for a call independent signalling connection and User A is not busy, User A's endpoint shall attempt to associate the APDU with a CC Request in state CC-Invoked-User-A-RLS, by comparing the information received within the invoke APDU and the information locally stored, and if successful, indicate the CC Recall to User A, start the recall timer T3 and enter state CC-Wait-User-A-Answer. The call independent signalling connection shall be cleared.

If the CC Recall is accepted by User A before timer T3 expires, User A's endpoint shall initiate a CC Call by sending a *ccRingout* invoke APDU in a Setup message towards User B's endpoint, stop timer T3 and enter state CC-Ringout. The Setup message shall contain information equivalent to the original unsuccessful call attempt, and the *ccRingout* invoke APDU should contain in its argument the element *ccIdentifier* indicating the value that was contained in the previous *ccbsRequest/ccnrRequest* invoke APDU.

If in state CC-Ringout an Alerting or a Connect message is received, User A's endpoint shall stop the service duration timer T2, delete the CC Request and return to state CC-Idle. The call independent signalling connection for SS-CC shall be cleared if it still exists, and the CC Call shall continue in accordance with basic call procedures.

#### 8.1.1.5 CC Call when User A busy

**Suspend procedure**

If a *ccExecPossible* invoke APDU is received in a Facility message on the call independent signalling connection while in state CC-Invoked-User-A-RET and User A is busy, User A's endpoint shall send a *ccSuspend* invoke APDU to User B's endpoint in a Facility message on the call independent signalling connection, start monitoring User A for not busy and enter state CC-Suspended-User-A.

If a *ccExecPossible* invoke APDU is received in a Setup message for a call independent signalling connection and User A is busy, User A's endpoint shall attempt to associate the APDU with a CC Request in state CC-Invoked-User-A-RLS, by comparing the information received within the invoke APDU and the information locally stored, and if successful, send a *ccSuspend* invoke APDU to User B's endpoint, start monitoring User A for not busy, and enter state CC-Suspended-User-A.

The *ccSuspend* invoke APDU shall be sent in a Release Complete message on the call independent signalling connection unless interworking requires specifically that the signalling connection is retained, in which case it may be sent in a Connect message.

**Resume procedure**

If User A, for whom a CC Request in state CC-Suspended-User-A exists, becomes not busy, User A's endpoint should send a *ccResume* invoke APDU – in a Facility message on the existing call independent signalling connection or in a Setup message for a new call independent signalling connection if no such connection exists – and enter state CC-Invoked-User-A-RET, waiting for another indication that User B is not busy. If sent in the Setup message, the invoke APDU shall contain all the information stored locally for this CC Request.

Alternatively User A's endpoint may continue with CC Call procedures as specified in 8.1.1.4.

#### 8.1.1.6 CCBS/CCNR cancellation

a) Cancellation initiated by User A's endpoint:

In order to cancel a CC Request, User A's endpoint shall send a *ccCancel* invoke APDU – in the Release Complete message that initiates clearing of the existing call independent signalling connection, in a Setup message for a new call independent signalling connection otherwise – to User B's endpoint. The Release Complete message shall contain a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason.* If sent in the Setup message, the invoke APDU shall contain all the information stored locally for this CC Request. User A's endpoint shall also release the CC Call (if already initiated) by sending a Release Complete message containing a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason*, delete all data stored for that CC Request, stop any timer running, and return to state CC-Idle. Unless cancellation was initiated by user request or by User A's endpoint automatically re-invoking SS-CC, User A shall, if the capability exists, be informed of failure of the service.

b) Cancellation initiated by User B's endpoint:

On receipt of a *ccCancel* invoke APDU from User B's endpoint in a Release Complete message for an existing call independent signalling connection, User A's endpoint shall delete all data stored for the associated CC Request, stop any timer still running, clear the CC Call (if already initiated), inform User A of the cancellation, if the capability exists, and return to state CC-Idle.

On receipt of a *ccCancel* invoke APDU from User B's endpoint in a Setup message for a new call independent signalling connection, User A's endpoint shall attempt to associate the APDU with an active CC Request, by comparing the information received within the invoke APDU and the information locally stored, and if successful, delete all data stored for the associated CC Request, stop any timer still running, clear the CC Call (if already initiated), inform User A of the cancellation if the capability exists, return a Release Complete message with a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason,* and return to state CC-Idle.

### 8.1.2 Exceptional procedures

#### 8.1.2.1 CCBS/CCNR invocation

If establishment of the call independent signalling connection fails, or if after sending a *ccbsRequest* or *ccnrRequest* invoke APDU no answer is received from User B's endpoint before timer T1 expires, or if the answer is a return error APDU or a reject APDU, a failure indication shall be given to User A, and User A's endpoint shall return to state CC-Idle. The call independent signalling connection shall be released, if it still exists. Timer T1 shall be stopped if still running.

#### 8.1.2.2 Service duration timer expiry

If the service duration timer T2 expires while in state CC-Invoked-User-A-RET, CC-Invoked-User-A-RLS, CC-Wait-User-A-Answer or CC-Suspended-User-A, User A's endpoint shall cancel the CC Request, using the procedure described in 8.1.1.6.

If the service duration timer T2 expires while in state CC-Ringout, User A's endpoint shall defer taking any action to allow reception of an Alerting or Connect message from User B's endpoint. Thus, the expiration of T2 shall be ignored and the procedures of 8.1.1.4 shall apply.

#### 8.1.2.3 Recall timer expiry

If User A does not accept the CC Recall before the recall timer T3 expires, User A's endpoint shall cancel the CC Request, using the procedure described in 8.1.1.6.

#### 8.1.2.4 Failure of CC Call presentation

If a Release Complete message with a *ccRingout* return error APDU is received with an error value *remoteUserBusyAgain*, User A's endpoint shall:

- if the service retention option is effective, return to state CC-Invoked-User-A-RET (in case of connection retain) or CC-Invoked-User-A-RLS (in case of connection release);
- if the service cancellation option is effective, stop timer T2 and release the call independent signalling connection if it still exists, by sending a Release Complete message containing a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason*. User A's endpoint shall then either:
  a) indicate the failure to User A, delete the CC Request and return to state CC-Idle; or
  b) automatically invoke SS-CCBS according to 8.1.1.1.

If the error indicated in the *ccRingout* return error APDU is *failureToMatch* or *unspecified*, User A's endpoint shall delete the CC Request, stop timer T2, indicate the failure to User A, and return to state CC-Idle. User A's endpoint shall also release the call independent signalling connection if it still exists, by sending a Release Complete message containing a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason*.

If the CC Call fails without a *ccRingout* return error APDU being received, User A's endpoint shall cancel the CC Request according to 8.1.1.6.

#### 8.1.2.5 Release of call independent signalling connection

If the call independent signalling connection associated with a CC Request in state CC-Wait-User-A-Answer, CC-Ringout or CC-Suspended-User-A is released for any reason, without sending or receiving a *ccCancel* invoke APDU, User A's endpoint shall remain in the current state.

If a Release Complete message without a *ccCancel* invoke APDU is received on the call independent signalling connection while in state CC-Invoked-User-A-RET, User A's endpoint shall enter state CC-Invoked-User-A-RLS.

#### 8.1.2.6 Unexpected APDU

If a Setup message for a call independent signalling connection is received containing a *ccExecPossible* invoke APDU which cannot be matched to any CC Request in state CC-Invoked-User-A-RLS, a Release Complete message shall be returned with a *ccCancel* invoke APDU and a *ReleaseCompleteReason* of *destinationReject.*

If a Setup message for a call independent signalling connection is received containing a *ccCancel* invoke APDU which cannot be matched to any outstanding CC Request, a Release Complete message shall be returned with a *ReleaseCompleteReason* of *destinationReject.*

## 8.2 Actions at User B's endpoint

### 8.2.1 Normal procedures

#### 8.2.1.1 CCBS invocation

If a *ccbsRequest* invoke APDU is received in a Setup message, User B's endpoint shall process the request according to 8.2.1.3 and start monitoring User B for becoming not busy, if the CC Request is accepted.

If User B is already not busy, this will result in immediate initiation of the procedures of 8.2.1.4 after the *ccbsRequest* return result APDU has been sent.

#### 8.2.1.2 CCNR invocation

If a *ccnrRequest* invoke APDU is received in a Setup message, User B's endpoint shall process the request according to 8.2.1.3 and start monitoring User B for becoming not busy after a subsequent period of activity, if the CC Request is accepted.

#### 8.2.1.3 SS-CC invocation – detailed procedure

If a CC Request can be accepted, User B's endpoint shall store the basic call information and *ccIdentifier* value (if present) contained in the argument of the *ccbsRequest/ccnrRequest* invoke APDU.

As an acknowledgment, User B's endpoint shall send a *ccbsRequest/ccnrRequest* return result APDU in a Connect or Release Complete message on the call independent signalling connection to User A's endpoint. Unless prohibited by a specific interworking requirement, the Connect message shall be chosen if the corresponding invoke APDU contained element *retain-sig-connection* with value TRUE, Release Complete shall be chosen if the corresponding invoke APDU contained element *retain-sig-connection* with value FALSE. In the absence of element *retain-sig-connection* User B's endpoint can choose either message.

If element *can-retain-service* was present in the corresponding invoke APDU with value TRUE and User B's endpoint chooses the service retention method, an element *retain-service* with value TRUE shall be included in the result. In all other cases, this element shall have the value FALSE.

User B's endpoint shall then enter state CC-Invoked-User-B and start to monitor User B for not busy (CCBS case) or not busy after a period of activity (CCNR case).

#### 8.2.1.4 Indication that User B is not busy

As soon as User B becomes not busy (in the case of SS-CCNR, after a period of activity subsequent to SS-CC invocation), User B's endpoint shall send a *ccExecPossible* invoke APDU in a Facility message on the existing call independent signalling connection or in a Setup message for a new call independent signalling connection if no such connection exists, enter state CC-Await-Call-Completion and wait for the incoming CC Call. If sent in the Setup message, the invoke APDU shall contain all the information stored locally for this CC Request.

NOTE – It is endpoint B's responsibility to protect against non-receipt of the CC Call. How this is achieved is outside the scope of this Recommendation.

#### 8.2.1.5 CC Call processing

If a Setup message is received with a *ccRingout* invoke APDU, User B's endpoint shall attempt to associate the incoming CC Call with a CC Request in state CC-Await-Call-Completion and, if successful and User B is still not busy, extend the CC Call to User B and enter state CC-Wait-User-B-Alert.

The association shall be achieved by comparing the basic call information and the *ccIdentifier* values locally stored with the basic call information and received in the Setup message and the *ccIdentifier* value contained in the *ccRingout* invoke APDU. Element *ccIdentifier* – if available – shall be the prime mechanism for this comparison; otherwise a match shall be deemed to occur if the elements stored locally match the corresponding information elements of the Setup message; any information element in the Setup message for which no corresponding element is stored (and vice versa) shall be ignored during the comparison.

If in state CC-Wait-User-B-Alert, an Alerting or a Connect message is sent to User A's endpoint, User B's endpoint shall delete the respective CC Request and enter state CC-Idle. The associated call independent signalling connection shall be released if it still exists, and the CC Call shall continue per basic call establishment.

#### 8.2.1.6 CCBS/CCNR suspension/resumption

If a *ccSuspend* invoke APDU is received in a Facility, Connect or Release Complete message on the existing call independent signalling connection while in state CC-Await-Call-Completion, User B's endpoint shall stay in state CC-Await-Call-Completion.

NOTE – Other actions are beyond the scope of this Recommendation.

If a *ccResume* invoke APDU is received in a Facility message on the existing call independent signalling connection while in state CC-Await-Call-Completion, User B's endpoint shall enter state CC-Invoked-User-B and resume monitoring of User B.

If a *ccResume* invoke APDU is received in a Setup message of a call independent signalling connection User B's endpoint shall attempt to associate it with a CC Request in state CC-Await-Call-Completion, by comparing the information received within the invoke APDU and the information stored locally, and if successful, enter state CC-Invoked-User-B and resume monitoring of User B.

The signalling connection shall be released if the connection release option applies.

If User B is already not busy this will result in immediate initiation of the procedures of 8.2.1.4.

#### 8.2.1.7 CCBS/CCNR cancellation

a) Cancellation initiated by User A's endpoint:

   If a *ccCancel* invoke APDU is received in a Release Complete message of a call independent signalling connection, User B's endpoint shall delete the information stored for the associated CC Request, release the signalling connection and enter state CC-Idle.

   If a *ccCancel* invoke APDU is received in a Setup message of a call independent signalling connection, User B's endpoint shall attempt to associate it with a CC Request and if successful, delete the information stored for the CC Request and enter state CC-Idle. The association shall be achieved by comparing the information received within the invoke APDU and the information stored locally (see 8.2.1.5). User B's endpoint shall return a Release Complete message with a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason*.

b) Cancellation initiated by User B's endpoint:

   In order to cancel a CC Request, User B's endpoint shall send a *ccCancel* invoke APDU either in a Setup message of a new call independent signalling connection (in the connection release case), or together with a Cause IE #16 "*normal call clearing*" or a *ReleaseCompleteReason* of *undefinedReason* in a Release Complete message on the existing call independent signalling connection, to User A's endpoint, delete the CC Request and enter state CC-Idle. If sent in the Setup message, the invoke APDU shall contain all the information stored locally for this CC Request.

### 8.2.2 Exceptional procedures

#### 8.2.2.1 CCBS/CCNR invocation

A CC Request may not be accepted if it is a duplicate of an already stored CC Request.

NOTE – Duplicate requests can also be caused by User B's endpoint discarding part of the received basic call information.

If a request for SS-CCBS or SS-CCNR cannot be accepted, User B's endpoint shall return a Release Complete message with a *ccbsRequest* return error APDU or *ccnrRequest* return error APDU, respectively, on the existing signalling connection. A *ReleaseCompleteReason* of *destinationReject* shall be indicated.

If the failure condition is persistent (e.g. service not provided for User B), error value *longTermRejection* shall be included.

If the failure condition is transient (e.g. duplicate CC Request, allowed number of active CC Requests exceeded), error value *shortTermRejection* shall be included.

#### 8.2.2.2 Unexpected APDUs

If a call related Setup message is received with a *ccRingout* invoke APDU that cannot be associated with a CC Request in state CC-Await-Call-Completion, User B's endpoint shall return a Release Complete message with a *ccRingout* return error APDU with error value *failureToMatch* and a *ReleaseCompleteReason* of *destinationReject*.

If a Setup message for a call independent signalling connection is received with a *ccResume* invoke APDU that cannot be associated with a CC Request in state CC-Await-Call-Completion, User B's endpoint shall return a Release Complete message with a *ccCancel* invoke APDU and a *ReleaseCompleteReason* of *destinationReject*.

If a Setup message for a call independent signalling connection is received with a *ccCancel* invoke APDU that cannot be associated with a CC Request, User B's endpoint shall return a Release Complete message with a *ReleaseCompleteReason* of *destinationReject*.

#### 8.2.2.3 User B busy again on CC Call presentation

If User B is busy again when receiving a *ccRingout* invoke APDU in a Setup message while in state CC-Await-Call-Completion, User B's endpoint shall return a Release Complete message with a *ccRingout* return error APDU with error value *remoteUserBusyAgain* and a *ReleaseCompleteReason* of *destinationReject*.

If the service retention option is in use, the corresponding CC Request shall be retained and monitoring of User B shall be resumed, returning to state CC-Invoked-User-B. The signalling connection shall be released if the connection release option applies.

If the service retention option is not in use, the CC Request shall be deleted and state CC-Idle shall be entered. If a signalling connection still exists, it shall be released.

#### 8.2.2.4 Interruption of CC Call

If a Release Complete message without any SS-CC invoke APDU is received for a CC Call in progress, or call establishment fails for a reason other than User B busy, the associated CC Request shall be deleted, state CC-Idle shall be entered and the call independent signalling connection shall be released if it still exists.

#### 8.2.2.5 Release of call independent signalling connection

If the call independent signalling connection associated with a CC Request in state CC-Await-Call-Completion, CC-Wait-User-B-Alert is released for any reason, without sending or receiving a *ccCancel* invoke APDU, User B's endpoint shall remain in the current state.

## 9 Interworking and interactions

### 9.1 Interworking with SCN

SS-CC may interwork with corresponding supplementary services for call completion as defined by other standards by means of gateway interworking functions.

SS-CC requires the availability of two functional entities EASE and EBSE as defined within this Recommendation. These two functional entities should be located as close as possible to the user's endpoint. Depending on the availability of a corresponding functional entity for call completion within the SCN, a gateway for SS-CC interworking shall either provide full EASE or EBSE functionality (see 9.1.1) or shall only provide limited EASE/EASE functions (see 9.1.2).

#### 9.1.1 No functional call completion entity exists in SCN

If a gateway has to provide interworking functions for SS-CC to an SCN with the SCN not providing any functional entity for call completion, a gateway may provide the procedures as defined within 8.1 and 8.2, including appropriate SCN signalling to convey the necessary information towards the SCN user. This functional entity may also be viewed as an SS-CC proxy function provided by the gateway on behalf of the remote SCN user.

NOTE – An SCN stimulus terminal device directly attached to a gateway may be viewed as an example scenario.

#### 9.1.2 Functional call completion entity exists in SCN

If a gateway has to provide interworking functions for SS-CC to an SCN with a functional entity for call completion (corresponding to EASE or EBSE) existing in the SCN, the gateway shall provide limited SS-CC functions as follows.

SS-CC related call states and timers as defined within 11.2.4, 11.3.4 and 11.4 do not apply to this type of gateway. Gateway functions are instead limited to the interworking of messages including the interworking of SS-CC related APDUs to the corresponding call completion signals at the SCN side and vice versa.

This includes the interworking of signalling related to:

– the CC Request (ccbsRequest and ccnrRequest APDUs);

– the progress of SS-CC (ccExecPossible, ccSuspend and ccResume APDUs);

– the CC Call (ccRingout APDUs); and

– the cancellation of SS-CC (ccCancel APDUs).

If SCN A's gateway received an indication that SCN A supports the service retention method, the SCN A's gateway shall indicate support for service retention method towards endpoint B.

SCN A's gateway shall indicate connection retain option to endpoint B if the SCN A indicated connection retain option and shall indicate connection release option to endpoint B if the SCN A indicated connection release option. If the SCN A does not indicate any preference on whether the signalling connection during User B monitoring phase is to be maintained or released, the SCN A's gateway may choose either one.

An SCN B gateway shall honour the connection retain or connection release option as indicated by endpoint A, unless prohibited by a specific interworking requirement.

The gateway that receives the *ccExecPossible* invoke APDU within a Setup message will not necessarily be the same gateway that handled SS-CC invocation. The same is true for the gateway that receives the *ccRingout* invoke APDU which does not necessarily have to be the same gateway that handled the SS-CC invocation or the indication that User B had become not busy. However, an endpoint may choose to make use of the capabilities to route all Setup messages belonging to one SS-CC procedure via one specific (i.e. the same) gateway.

NOTE – This can be achieved in most circumstances by addressing both the gateway and the SCN user using elements destinationAddress (= gateway address) and remoteExtensionAddress (= SCN User Address) within Setup-UUIE or elements destinationAddress (= gateway address) and remoteExtensionAddress (= SCN User Address) within EndpointAddress of H.450.9 APDUs contained within the Setup message. Similar methods – if available in the SCN – may be used for corresponding call establishment messages belonging to the same SS-CC procedure.

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

## 9.2 Protocol interaction between SS-CCBS and other supplementary services

This clause specifies protocol interactions of SS-CCBS with other supplementary services for which ITU-T H.450 series of Recommendations had been published at the time of publication of this Recommendation. For interactions with supplementary services for which ITU-T H.450 series of Recommendations are published subsequent to the publication of this Recommendation, see those other ITU-T H.450 Recommendations.

### 9.2.1 Call Transfer (ITU-T H.450.2)

No protocol interaction.

### 9.2.2 Call Diversion (ITU-T H.450.3)

#### 9.2.2.1 Call Forwarding Unconditional (SS-CFU)

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

##### 9.2.2.1.1 User A's endpoint procedures for invoking SS-CCBS at an SS-CFU diverted-to user

NOTE – In this case User A's endpoint with regard to SS-CCBS is also the calling endpoint with regard to SS-CFU.

If SS-CCBS is to be invoked at a busy diverted-to user, User A's endpoint shall store the content of element *nominatedNr*, if received in the argument of operation *divertingLegInformation1*, and use it as:

- element *numberB* in the argument of any SS-CCBS operation which requires this element;
- Called party number information element or *destinationAddress* element in the Setup message of the call independent signalling connection;
- Called party number information element or *destinationAddress* element in the Setup message of the CC Call.

The address of the originally called user shall not be used for SS-CCBS.

If element *nominatedNr* is not available, an SS-CCBS request received from User A shall be rejected.

##### 9.2.2.1.2 User A's endpoint procedures if SS-CFU is activated by User A

No protocol interaction.

#### 9.2.2.1.3 User B's endpoint procedures if SS-CFU is activated by User B after SS-CCBS has been invoked

No protocol interaction.

NOTE – An incoming CC Call is not subject to unconditional diversion.

### 9.2.2.2 Call Forwarding Busy (SS-CFB)

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

#### 9.2.2.2.1 User A's endpoint procedures for invoking SS-CCBS at an SS-CFB diverted-to user

NOTE – In this case User A's endpoint with regard to SS-CCBS is also the calling endpoint with regard to SS-CFB.

If SS-CCBS is to be requested against a busy diverted-to user (User C) rather than against the busy diverting User B then the procedures of 9.2.2.1.1 shall apply; otherwise no protocol interaction.

#### 9.2.2.2.2 User A's endpoint procedures if SS-CFB is activated by User A

No protocol interaction.

#### 9.2.2.2.3 User B's endpoint procedures if SS-CFB is activated by User B after SS-CCBS has been invoked

No protocol interaction.

NOTE – An incoming CC Call is not subject to diversion on busy.

### 9.2.2.3 Call Forwarding No Reply (SS-CFNR)

Clause 9.2.2.1 shall apply with "SS-CFU" replaced by "SS-CFNR".

NOTE – An incoming CC Call can be subject to diversion on no reply when not answered by User B.

### 9.2.2.4 Call Deflection (SS-CD)

Clause 9.2.2.1 shall apply with "SS-CFU" replaced by "SS-CD".

NOTE – An incoming CC Call can be subject to deflection by User B.

## 9.2.3 Call Hold (ITU-T H.450.4)

No protocol interaction.

## 9.2.4 Call Park and Call Pickup (ITU-T H.450.5)

No protocol interaction.

NOTE – An incoming CC Call to User B cannot be picked up by another user.

## 9.2.5 Call Waiting (ITU-T H.450.6)

No protocol interaction.

## 9.2.6 Message Waiting (ITU-T H.450.7)

No protocol interaction.

## 9.2.7 Name Presentation (ITU-T H.450.8)

No protocol interaction.

### 9.2.8 Call Linkage (ITU-T H.323, 1998)

A *callLinkage* element may be included in every message, encoded according to the following rules:

- If available, the Thread ID of the original call attempt, i.e. the call that encountered User B busy, shall also be used (element *callLinkage.threadId*):
  - for the call independent signalling connections that carry all SS-CC operations except ccRingout;
  - for the CC Call.
- If the original call attempt did not contain a Thread ID, its Call Identifier value (element *callIdentifier*) may be used as Thread ID (element *callLinkage.threadId*) in subsequent messages.
- Every call/signalling connection shall use a new Global Call ID (element *callLinkage.globalCallId*).

NOTE – Call Linkage is an optional feature that may be supported by H.323 version 3 (or later) endpoints and entities.

## 9.3 Protocol interaction between SS-CCNR and other supplementary services

This clause specifies protocol interactions of SS-CCNR with other supplementary services for which ITU-T H.450 series of Recommendations had been published at the time of publication of this Recommendation. For interactions with supplementary services for which ITU-T H.450 series of Recommendations are published subsequent to the publication of this Recommendation, see those other ITU-T H.450 Recommendations.

### 9.3.1 Call Transfer (ITU-T H.450.2)

No protocol interaction.

### 9.3.2 Call Diversion (ITU-T H.450.3)

#### 9.3.2.1 Call Forwarding Unconditional (SS-CFU)

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

##### 9.3.2.1.1 User A's endpoint procedures for invoking SS-CCNR at an SS-CFU diverted-to user

NOTE – In this case User A's endpoint with regard to SS-CCNR is also the calling endpoint with regard to SS-CFU.

If SS-CCNR is to be invoked at a diverted-to user who does not answer, User A's endpoint shall store the content of element *nominatedNr*, if received in the argument of operation *divertingLegInformation1*, and use it as:

- element *numberB* in the argument of any SS-CCNR operation which requires this element;
- Called party number information element or *destinationAddress* element in the SETUP message of the call independent signalling connection;
- Called party number information element or *destinationAddress* element in the SETUP message of the CC Call.

The address of the originally called user shall not be used for SS-CCNR.

If element *nominatedNr* is not available, an SS-CCNR request received from User A shall be rejected.

##### 9.3.2.1.2 User A's endpoint procedures if SS-CFU is activated by User A

No protocol interaction.

#### 9.3.2.1.3 User B's endpoint procedures if SS-CFU is activated by User B after SS-CCNR has been invoked

No protocol interaction.

NOTE – An incoming CC Call is not subject to unconditional diversion.

### 9.3.2.2 Call Forwarding Busy (SS-CFB)

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

#### 9.3.2.2.1 User A's endpoint procedures for invoking SS-CCNR at an SS-CFB diverted-to user

NOTE – In this case User A's endpoint with regard to SS-CCNR is also the calling endpoint with regard to SS-CFB.

The procedures of 9.3.2.1.1 shall apply.

#### 9.3.2.2.2 User A's endpoint procedures if SS-CFB is activated by User A

No protocol interaction.

#### 9.3.2.2.3 User B's endpoint procedures if SS-CFB is activated by User B after SS-CCNR has been invoked

No protocol interaction.

NOTE – An incoming CC Call is not subject to diversion on busy.

### 9.3.2.3 Call Forwarding No Reply (SS-CFNR)

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

#### 9.3.2.3.1 User A's endpoint procedures for invoking SS-CCNR at an SS-CFNR diverted-to user

If SS-CCNR is to be requested against a diverted-to user (User C) rather than against the diverting User B (with both users not answering) then the procedures of 9.3.2.1.1 shall apply; otherwise no protocol interaction.

#### 9.3.2.3.2 User A's endpoint procedures if SS-CFNR is activated by User A

No protocol interaction.

#### 9.3.2.3.3 User B's endpoint procedures if SS-CFNR is activated by User B after SS-CCNR has been invoked

No protocol interaction.

### 9.3.2.4 Call Deflection (SS-CD)

The protocol interactions with Call Deflection shall be as specified in 9.3.2.3 for interaction with SS-CFNR.

## 9.3.3 Call Hold (ITU-T H.450.4)

No protocol interaction.

## 9.3.4 Call Park and Call Pickup (ITU-T H.450.5)

No protocol interaction.

NOTE – An incoming CC Call to User B cannot be picked up by another user.

### 9.3.5 Call Waiting (ITU-T H.450.6)

No protocol interaction.

NOTE – Invocation of SS-CCNR terminates Call Waiting.

### 9.3.6 Message Waiting (ITU-T H.450.7)

No protocol interaction.

### 9.3.7 Name Presentation (ITU-T H.450.8)

No protocol interaction.

### 9.3.8 Call Linkage (ITU-T H.323, 1998)

A *callLinkage* element may be included in every message, encoded according to the following rules:

- If available, the Thread ID of the original call attempt, i.e. the call that was not answered by User B, shall also be used (element *callLinkage.threadId*):
  - for the call independent signalling connections that carry all SS-CC operations except ccRingout;
  - for the CC Call.
- If the original call attempt did not contain a Thread ID, its Call Identifier value (element *callIdentifier*) may be used as Thread ID (element *callLinkage.threadId*) in subsequent messages.
- Every call/signalling connection shall use a new Global Call ID (element *callLinkage.globalCallId*).

NOTE – Call Linkage is an optional feature that may be supported by H.323 version 3 (or later) endpoints and entities.

## 10 Gatekeeper actions

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

- gatekeeper passes on all received SS-CC operations for processing at the endpoint (10.1); or
- gatekeeper acts on behalf of endpoint A and/or endpoint B for SS-CC (10.2).

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

### 10.1 Gatekeeper passes on SS-CC operations to the endpoint

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

NOTE – A gatekeeper may modify the contents of SS-CC operations, if required (e.g. removal of an address information that is marked as presentation restricted and if the destination endpoint is not trusted in respect to honouring the presentation restriction indication).

### 10.2 Gatekeeper/proxy acts on behalf of an endpoint

#### 10.2.1 Gatekeeper/proxy acts on behalf of endpoint B

A gatekeeper/proxy (for the gatekeeper-routed model or, respectively, in the case of a call routed through a proxy) acting on behalf of endpoint B may decide to become the destination for all SS-CC operations destined for endpoint B and the source of all SS-CC operations sent to endpoint A. The gatekeeper/proxy shall then perform the actions as defined in 8.2 "Actions at User B's endpoint".

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.

A stimulus-based protocol may be used in this case between the gatekeeper/proxy and User B's endpoint for which the gatekeeper/proxy is acting as SS-CC control entity.

### 10.2.2 Gatekeeper/proxy acts on behalf of endpoint A

A gatekeeper/proxy (for the gatekeeper-routed model or, respectively, in the case of a call routed through a proxy) may act as the SS-CC control entity on behalf of endpoint A and thus become the source of all SS-CC operations sent to endpoint B and the destination for all SS-CC operations destined for endpoint A. The gatekeeper/proxy shall in this case provide the actions as defined in 8.1 "Actions at User A's endpoint".

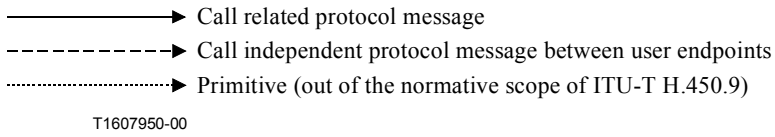
A stimulus-based protocol may be used by a User A's endpoint to initiate and control SS-CC within the gatekeeper/proxy. At the time of the CC Recall endpoint A must be notified and the CC Call must be established between endpoint A and endpoint B. How this is achieved is outside the scope of this Recommendation.

## 11 Dynamic description

### 11.1 Operational model and signal flows

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

1) The following notation is used:




SETUP Message name

Cx Number of connection x

xxx.inv Invoke APDU for operation xxx

xxx.rr Return result APDU for operation xxx

xxx.re Return error APDU for operation xxx

NOTE – The following scenarios are shown as separate signalling flow diagrams:

– direct routed call model (e.g. Figure 2);

– gatekeeper routed call model with gatekeeper passing on H.450.9 APDUs (e.g. Figure 3);

– gatekeeper routed call model with gatekeeper/proxy acting on behalf of endpoint A (e.g. Figure 4);

– gatekeeper routed call model with gatekeeper/proxy acting on behalf of endpoint B (e.g. Figure 5).

In the case where the gatekeeper passes on all SS-CC APDUs for endpoint processing, the endpoints support the procedures for SS-CC (see 10.1).

If the model as described in 10.2 applies, an origination gatekeeper/proxy A may perform the SS-CC actions as defined in 8.1 for User A's endpoint, and/or a destination gatekeeper/proxy B may perform the SS-CC actions as defined in 8.2 for User B's endpoint. In this case the interface between an endpoint and the gatekeeper/proxy is shown for informational purposes only and may be viewed as a stimulus feature protocol interface which is outside the scope of this Recommendation.

### 11.1.1 Successful CCBS

Figures 1 to 6 show successful CCBS request and recall scenarios for the direct routed call model, the gatekeeper routed call model (passing on SS-CCBS APDUs for endpoint processing), and for a GK or proxy acting on behalf of endpoint A or endpoint B, respectively, with regard to SS-CCBS.

Figure 3 shows a direct routed call with the connection release option where the signalling connection is released during monitoring of User B. This option applies also to gatekeeper routed cases (not shown in examples here).

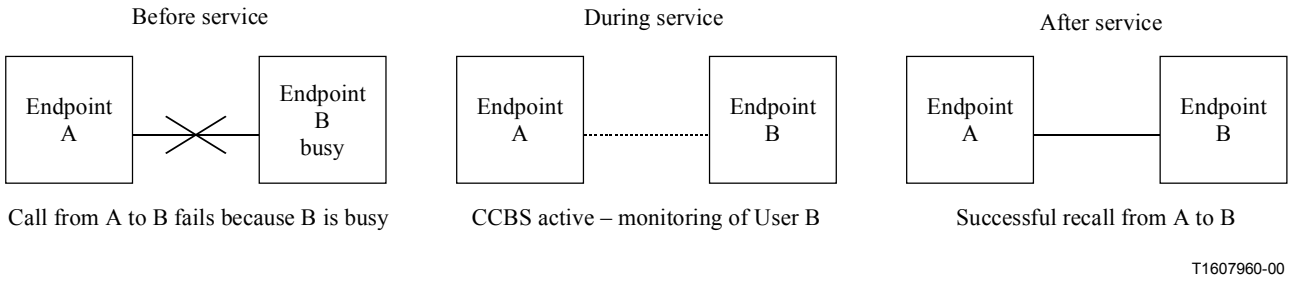



**Figure 1/H.450.9 – Operational model for SS-CCBS**

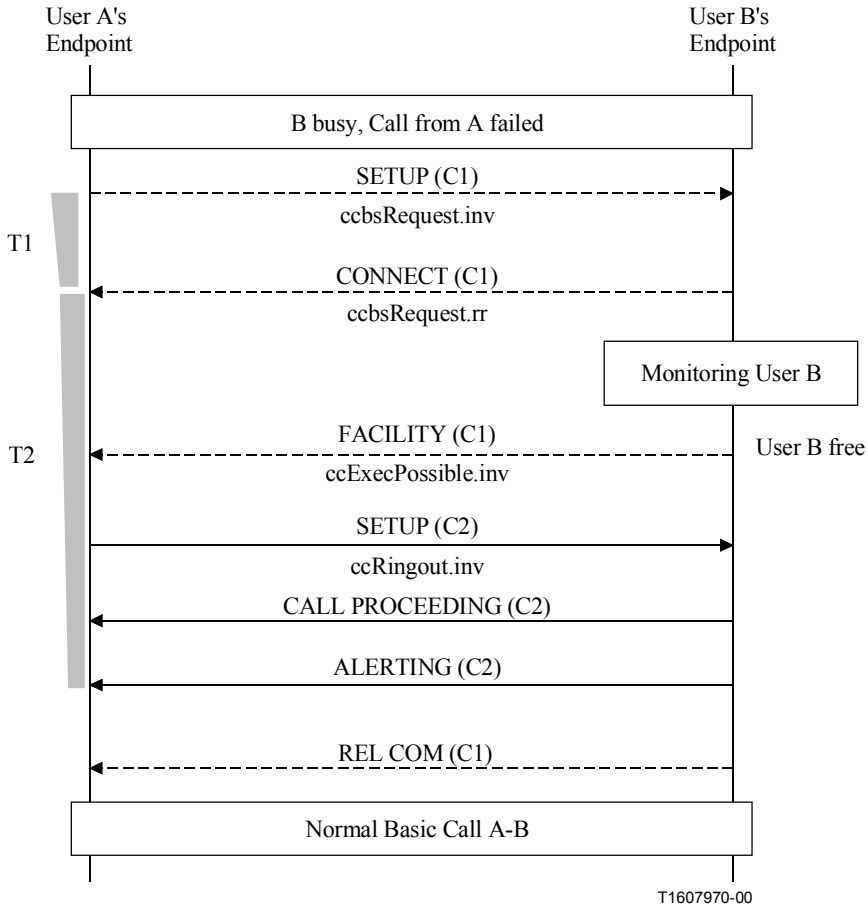



**Figure 2/H.450.9 – Example message flow for successful CCBS Direct routed call signalling – Signalling connection retained**

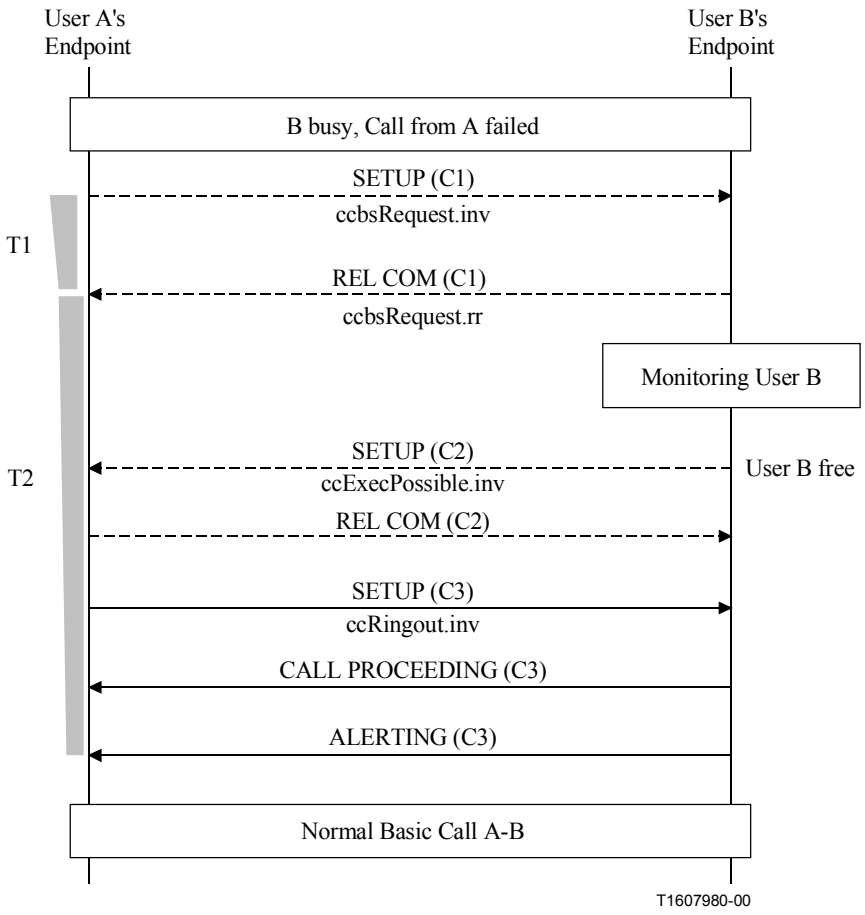



**Figure 3/H.450.9 – Example message flow for successful CCBS Direct routed call signalling – Connection release option**

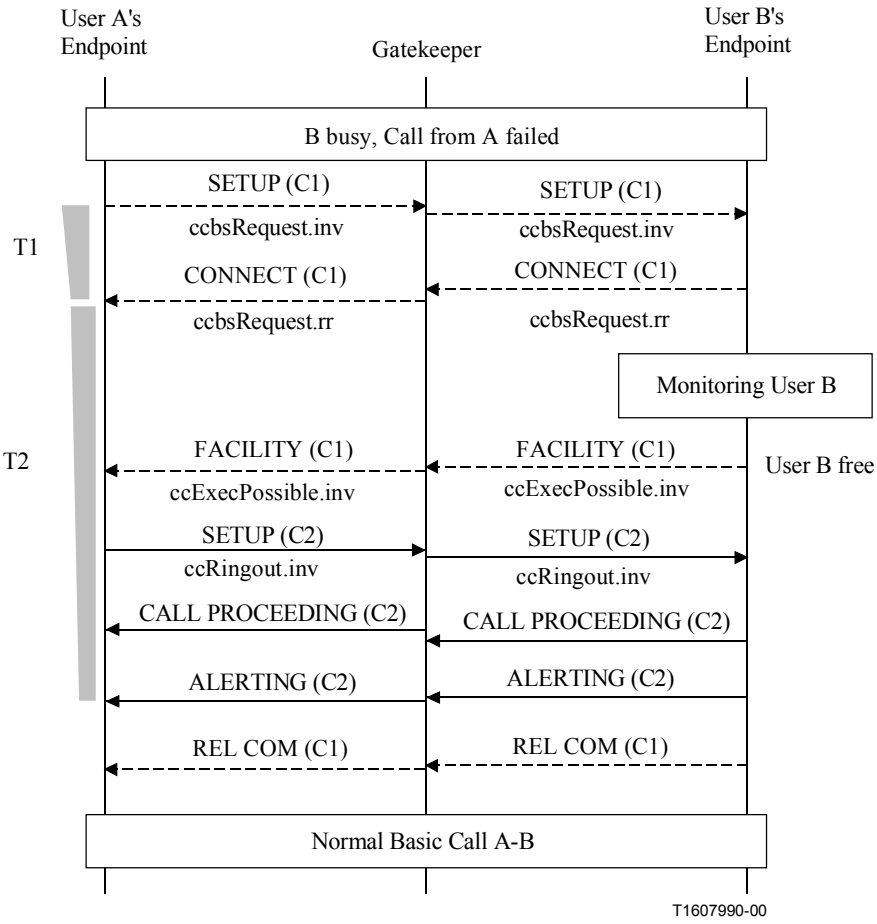
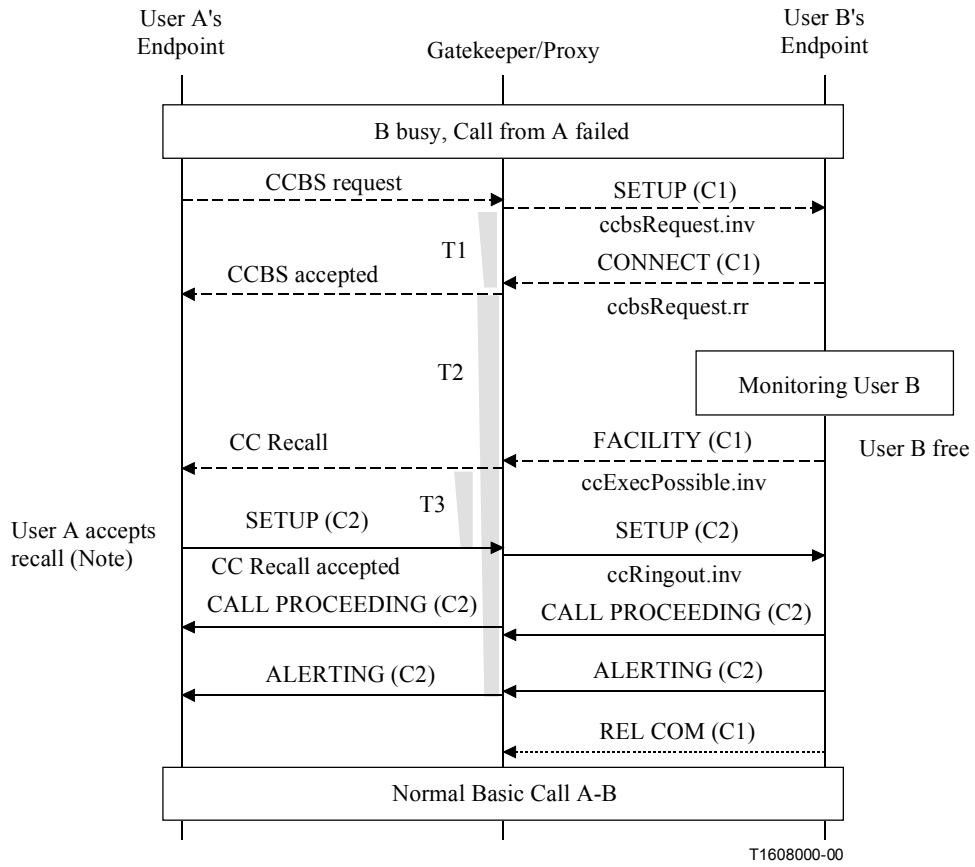



**Figure 4/H.450.9 – Example message flow for successful CCBS GK routed call signalling, GK not involved in SS-CCBS**




NOTE – SETUP sent from User A's endpoint is one possible way of establishing the CC Call. Other methods are possible. This is outside the scope of this Recommendation.

**Figure 5/H.450.9 – Example message flow for successful CCBS GK routed call signalling, GK handles SS-CCBS for User A**

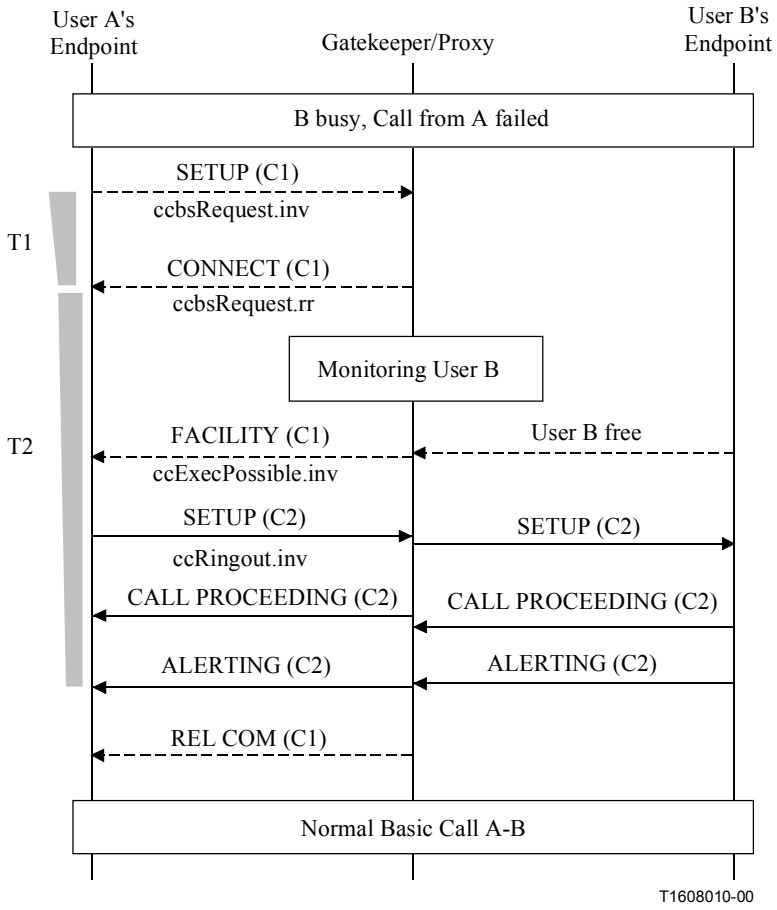



**Figure 6/H.450.9 – Example message flow for successful CCBS GK routed call signalling, GK handles SS-CCBS for User B**

### 11.1.2 Successful CCNR

Figures 7 and 8 show an example for a successful CCNR request and recall. In principle the same scenarios as for SS-CCBS are possible (see 11.1.1).

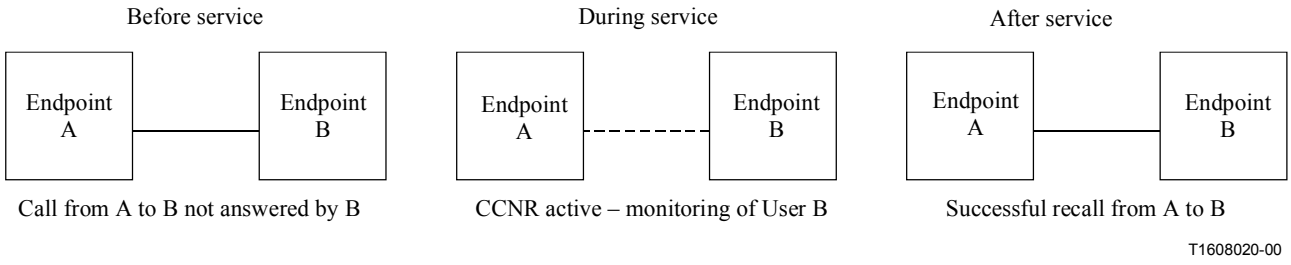



**Figure 7/H.450.9 – Operational model for SS-CCNR**

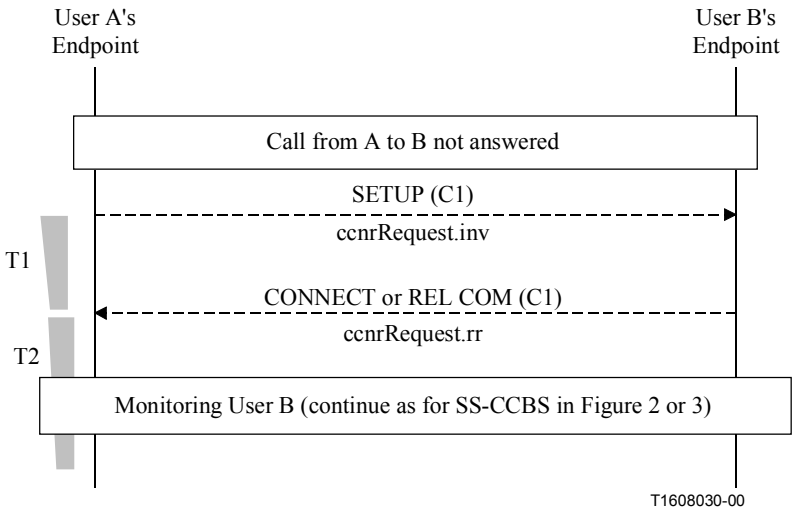



**Figure 8/H.450.9 – Example message flow for successful CCNR Direct routed call signalling**

### 11.1.3 User A busy

Figures 9 to 11 show a CC Request, which encounters User A busy when User B is not busy. For the direct routed call model, the cases retention of the signalling connection and connection release are shown. For other possible scenarios see 11.1.1.

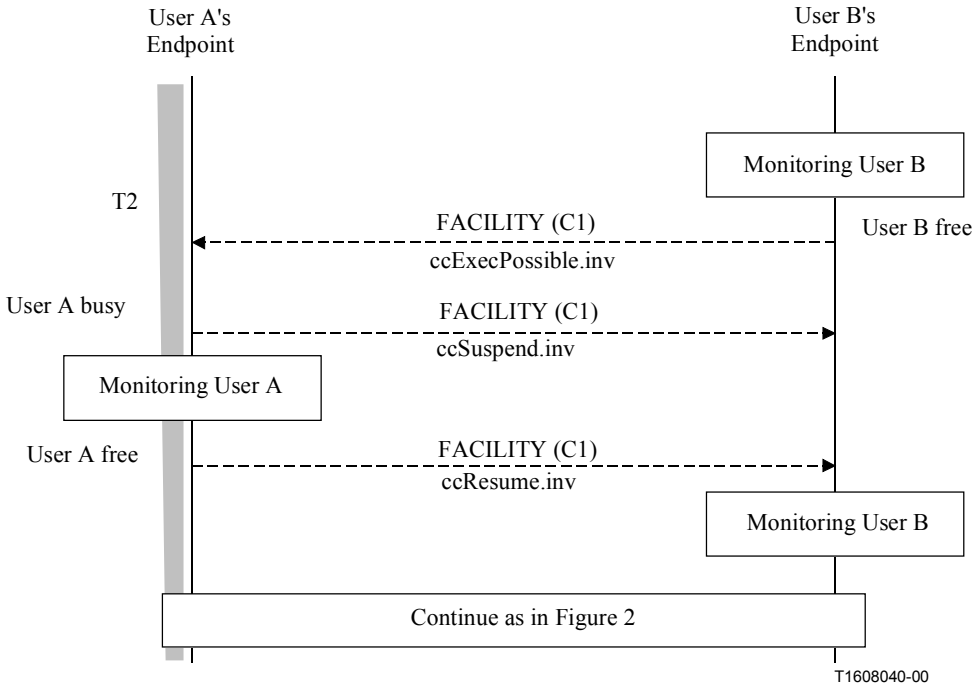



**Figure 9/H.450.9 – Example message flow for User A busy Direct routed call signalling – Signalling connection retained**

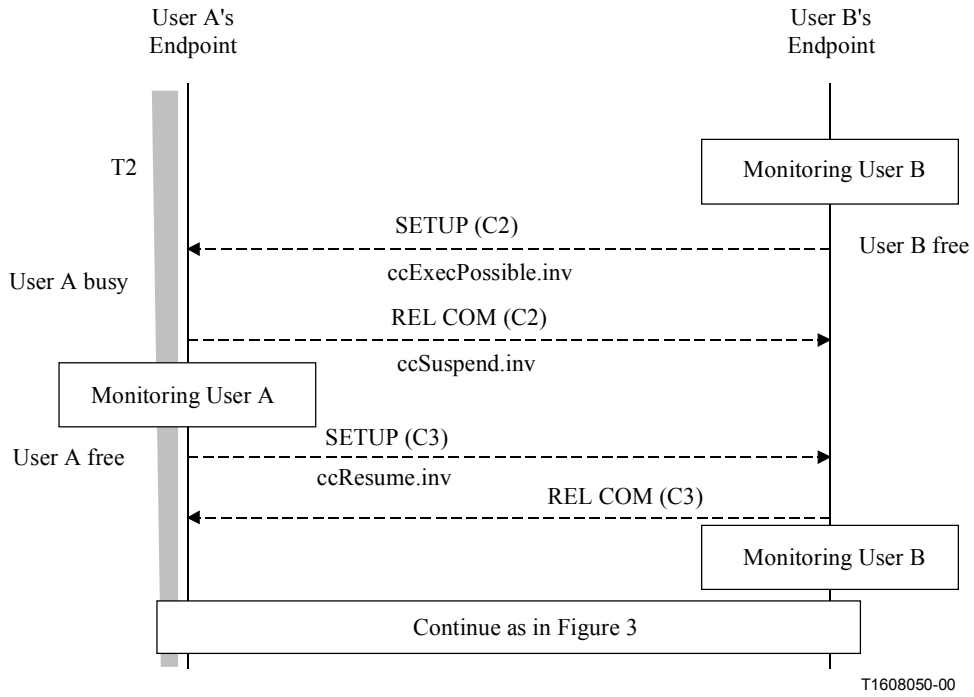



**Figure 10/H.450.9 – Example message flow for User A busy Direct routed call signalling – Connection release option**

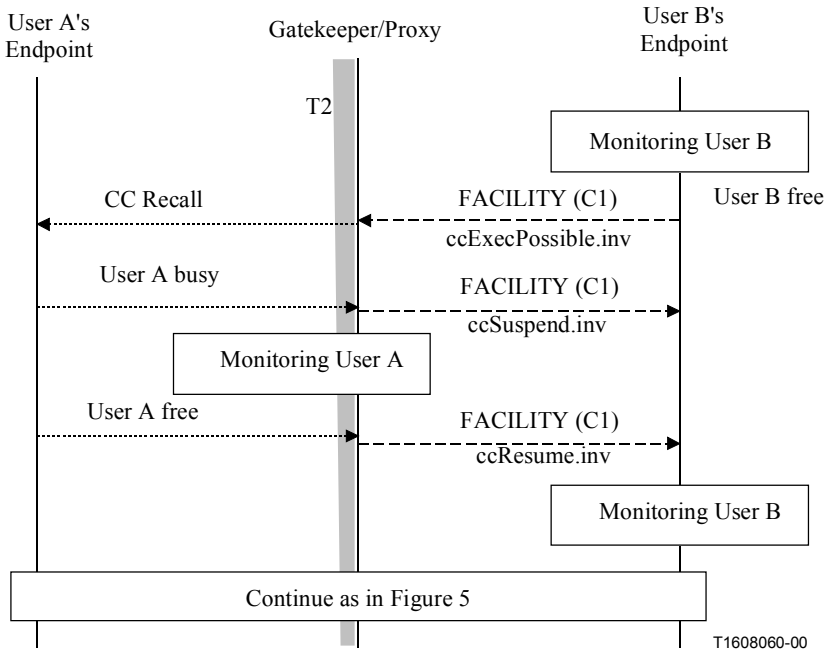



**Figure 11/H.450.9 – Example message flow for User A busy GK routed call signalling, GK handles SS-CCBS for User A**

### 11.1.4 User B busy again

#### 11.1.4.1 At CC Call presentation – without SS-CC service retention

Figures 12 to 14 show the case that a CC Call encounters User B busy again and the SS-CC service is not retained. For other possible scenarios see 11.1.1.

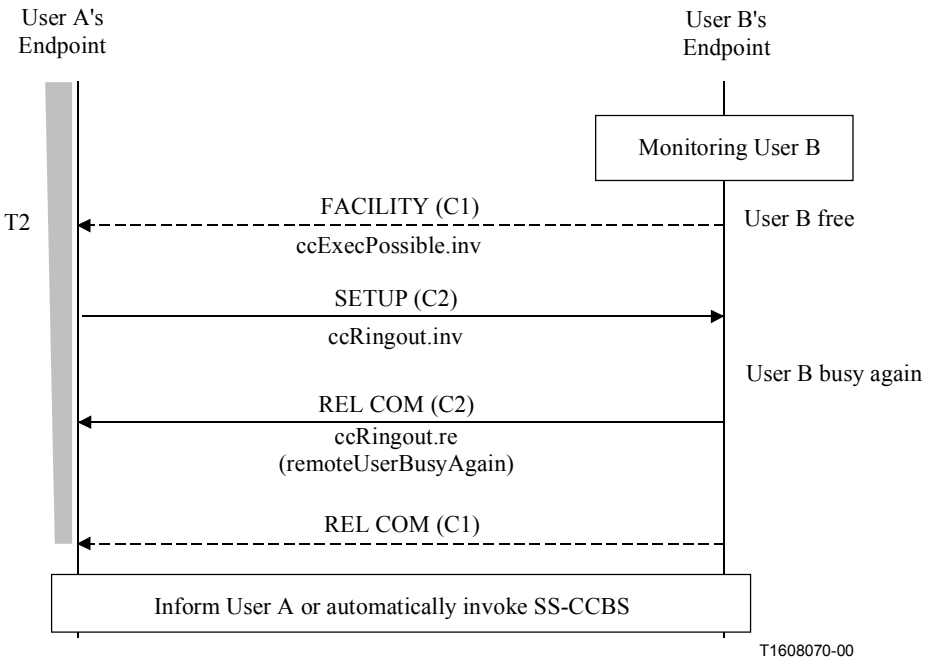



**Figure 12/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again Direct routed call signalling**

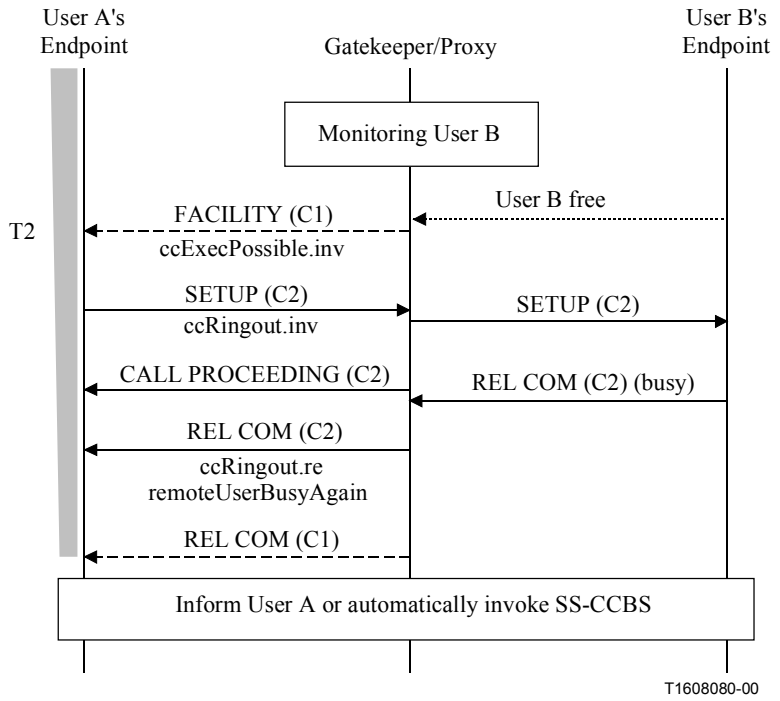



**Figure 13/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again GK routed call signalling, GK handles SS-CCBS for User B signalling connection retained**

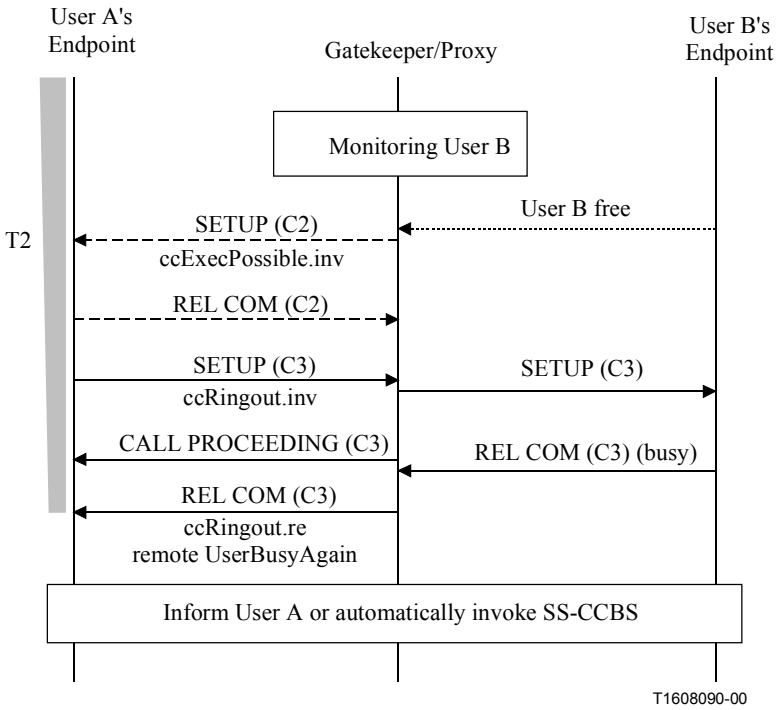



**Figure 14/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again GK routed call signalling, GK handles SS-CCBS for User B connection release option**

### 11.1.4.2 At CC Call presentation – with SS-CC service retention

Figures 15 to 17 show the case that a CC Call encounters User B busy again and the SS-CC service is retained. For other possible scenarios see 11.1.1.

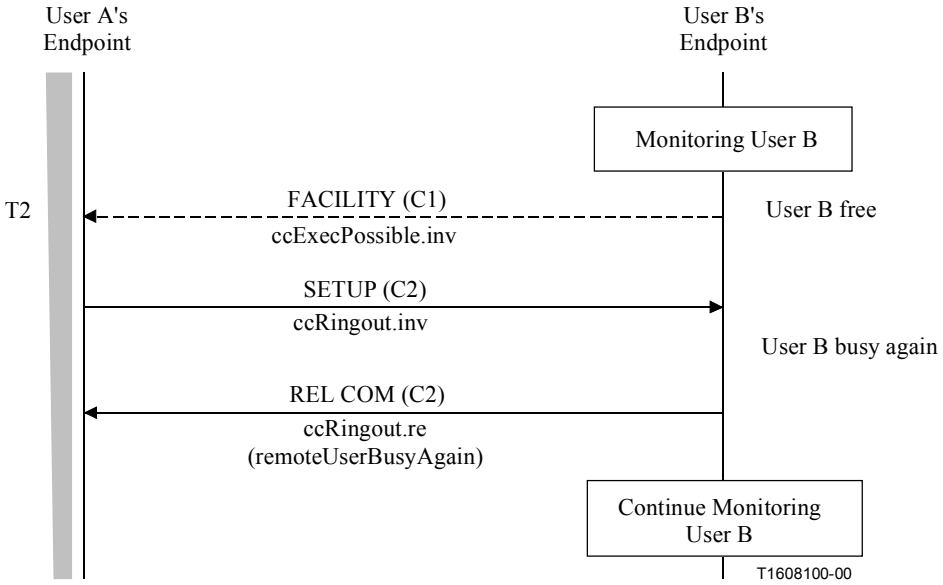



**Figure 15/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again Direct routed call signalling, signalling connection retained**

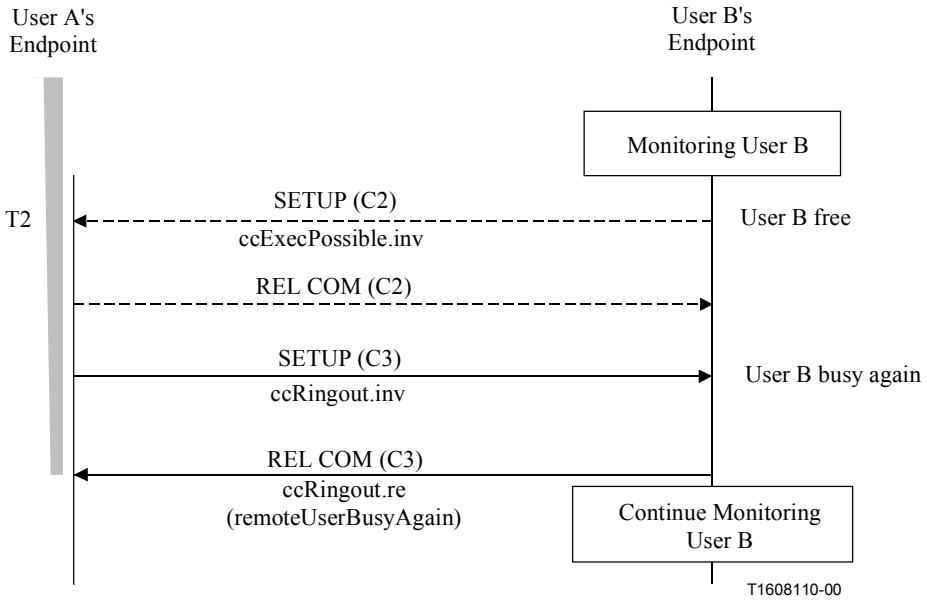



**Figure 16/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again Direct routed call signalling, connection release option**

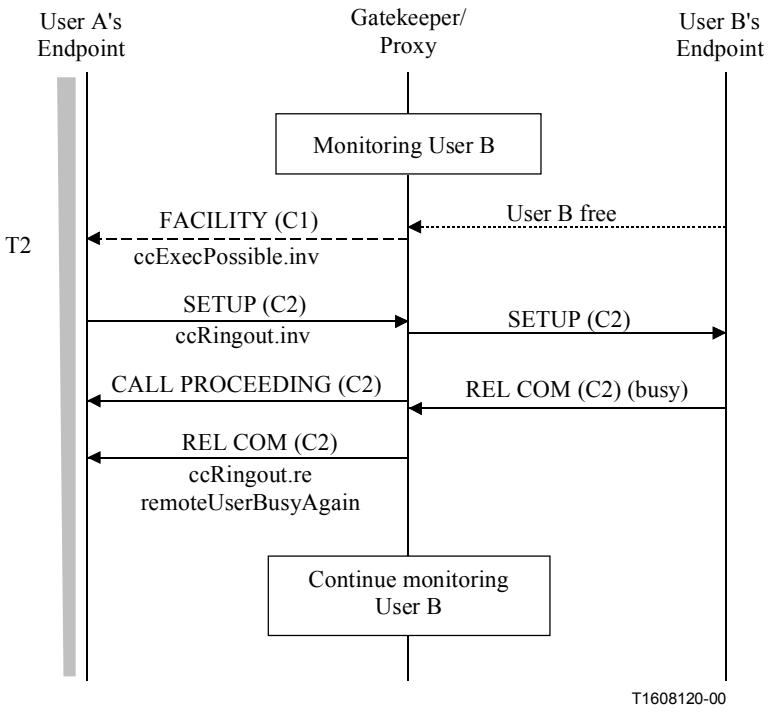



**Figure 17/H.450.9 – Example message flow for unsuccessful CCBS: User B busy again GK routed call signalling, GK handles SS-CCBS for User B**

## 11.2 Communication between an endpoint A (EASE) Signalling entity and its Signalling entity user (informative)

If a gatekeeper/proxy acts on behalf of an endpoint, the gatekeeper/proxy is considered as being the Signalling entity, whereas the endpoint that the gatekeeper/proxy is serving 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.

### 11.2.1 Table of primitives

See Table 1.

**Table 1/H.450.9 – Primitives at User A's endpoint**

| Generic name | Type | | | |
|---|---|---|---|---|
| | **Request (req)** | **Indication (ind)** | **Response (resp)** | **Confirm (conf)** |
| CcbsRequest | PARAMETERS | Not defined (Note) | Not defined | PARAMETERS |
| CcnrRequest | PARAMETERS | Not defined | Not defined | PARAMETERS |
| CcExecPossible | Not defined | PARAMETERS | Not defined | Not defined |
| CcRingout | PARAMETERS | Not defined | Not defined | PARAMETERS |
| CcCancel | PARAMETERS | PARAMETERS | Not defined | Not defined |
| CcSuspend | PARAMETERS | Not defined | Not defined | Not defined |
| CcResume | PARAMETERS | Not defined | Not defined | Not defined |
| NOTE – Means that this primitive is not defined. | | | | |

### 11.2.2 Primitive definition

The CcbsRequest.Request primitive is used to invoke SS-CCBS. The CcbsRequest.Confirm primitive is used to report the outcome of the invocation attempt.

The CcnrRequest.Request primitive is used to invoke SS-CCNR. The CcnrRequest.Confirm primitive is used to report the outcome of the invocation attempt.

The CcExecPossible.Indication primitive is used to report the possibility of re-calling User B.

The CcRingout.Request primitive is used to initiate the CC Call to User B. The CcRingout.Confirm primitive is used in error cases to report failure of the CC Call.

The CcCancel.Request primitive is used to cancel an SS-CC instance. The CcCancel.Indication primitive is used to report unsuccessful termination of an SS-CC instance.

The CcSuspend.Request primitive is used to suspend SS-CC while User A is busy and cannot proceed with a CC Recall.

The CcResume.Request primitive is used to resume SS-CC after having been suspended because of User A busy.

### 11.2.3 Parameter definition

**CcbsRequest.Request parameters**

sourceAddress: address of User A

destinationAddress: address of User B

ccIdentifier: ID of a previous (failed) call attempt to busy User B (optional)

service: the service type intended for the failed call attempt

canRetainService: indicator for service retention option

retainSigConn: indicator for handling of the signalling connection

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

**CcbsRequest.Confirm parameters**

Refer to 11.3.3 (parameters for CcbsRequest.Response primitive).

**CcnrRequest.Request parameters**

sourceAddress: address of User A

destinationAddress: address of User B

ccIdentifier: ID of a previous unanswered call attempt to User B (optional)

service: the service type intended for the unanswered call attempt

canRetainService: indicator for service retention option

retainSigConn: indicator for handling of the signalling connection

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

**CcnrRequest.Confirm parameters**

Refer to 11.3.3 (parameters for CcnrRequest.Response primitive).

**CcExecPossible.Indication parameters**

Refer to 11.3.3 (parameters for CcExecPossible.Request primitive).

**CcRingout.Request parameters**

ccIdentifier: association of the CC Call with an instance of SS-CC at EASE (optional)

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

**CcRingout.Confirm parameters**

Refer to 11.3.3 (parameters for CcRingout.Response primitive).

**CcCancel.Request and CcCancel.Indication parameters**

ccIdentifier: identifies the SS-CC instance to be cancelled (optional)

sourceAddress: the stored address of User A (optional)

destinationAddress: the stored address of User B (optional)

service: the stored service type (optional)

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

**CcSuspend.Request parameters**

ccIdentifier: identifies the SS-CC instance to be suspended (optional)

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

**CcResume.Request parameters**

ccIdentifier: identifies the SS-CC instance to be resumed (optional)

sourceAddress: the stored address of User A (optional)

destinationAddress: the stored address of User B (optional)

service: the stored service type (optional)

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

### 11.2.4 Call States

| CC State | Description |
|---|---|
| CC-Idle | This state exists if SS-CC is not active. |
| CC-Invoked-User-A-RET | This state exists for an active CC Request while waiting for the indication that User B is not busy, with the signalling connection retained. |
| CC-Invoked-User-A-RLS | This state exists for an active CC Request while waiting for the indication that User B is not busy, with the signalling connection released. |
| CC-Ringout | This state exists when User A has accepted the CC Recall, but completion of the call to User B is still pending. |
| CC-Suspended-User-A | This state exists when a CC Call has been postponed because User A is busy. |
| CC-Wait-Ack | This state exists during SS-CC invocation. |
| CC-Wait-User-A-Answer | This state exists while waiting for CC Recall acceptance from User A. |

## 11.3 Communication between an endpoint B (EBSE) signalling entity and its signalling entity user (informative)

If a gatekeeper/proxy acts on behalf of an endpoint, the gatekeeper/proxy is considered as being the Signalling entity, whereas the endpoint that the gatekeeper/proxy is serving 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.

### 11.3.1 Table of primitives

See Table 2.

**Table 2/H.450.9 – Primitives at User B's endpoint**

| Generic name | Type | | | |
|---|---|---|---|---|
| | **Request (req)** | **Indication (ind)** | **Response (resp)** | **Confirm (conf)** |
| CcbsRequest | Not defined (Note) | PARAMETERS | PARAMETERS | Not defined |
| CcnrRequest | Not defined | PARAMETERS | PARAMETERS | Not defined |
| CcExecPossible | PARAMETERS | Not defined | Not defined | Not defined |
| CcRingout | Not defined | PARAMETERS | PARAMETERS | Not defined |
| CcCancel | PARAMETERS | PARAMETERS | Not defined | Not defined |
| CcSuspend | Not defined | PARAMETERS | Not defined | Not defined |
| CcResume | Not defined | PARAMETERS | Not defined | Not defined |
| NOTE – Means that this primitive is not defined. | | | | |

### 11.3.2 Primitive definition

The CcbsRequest.Indication primitive is used to indicate an invocation attempt of SS-CCBS. The CcbsRequest.Response primitive is used to report the outcome of the invocation attempt.

The CcnrRequest.Indication primitive is used to indicate an invocation attempt of SS-CCNR. The CcnrRequest.Response primitive is used to report the outcome of the invocation attempt.

The CcExecPossible.Request primitive is used to report that User B has become free.

The CcRingout.Indication primitive is used to identify an incoming call as a CC Call. The CcRingout.Response primitive is used to report failure of a CC Call attempt.

The CcCancel.Request primitive is used to cancel an SS-CC instance. The CcCancel.Indication primitive is used to indicate cancellation of an SS-CC instance.

The CcSuspend.Indication primitive is used to indicate suspension of an SS-CC instance.

The CcResume.Indication primitive is used to indicate resumption of a suspended SS-CC instance.

### 11.3.3 Parameter definition

**CcbsRequest.Indication parameters**

Refer to 11.2.3 (parameters for CcbsRequest.Request primitive).

**CcbsRequest.Response parameters (ack and rej)**

(ack) retainService: indicator for service retention option if the invocation is accepted

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

(rej) rejectPerm: invocation rejected (long term)

(rej) rejectTemp: invocation rejected (short term)

**CcnrRequest.Indication parameters**

Refer to 11.2.3 (parameters for CcnrRequest.Request primitive).

**CcnrRequest.Response parameters (ack and rej)**

(ack) retainService: indicator for service retention option if the invocation is accepted

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

(rej) rejectPerm: invocation rejected (long term)

(rej) rejectTemp: invocation rejected (short term)

**CcExecPossible.Request parameters**

ccIdentifier: identifies an instance of SS-CC at EBSE (optional)

sourceAddress: the stored address of User A (optional)

destinationAddress: the stored address of User B (optional)

service: the stored service type (optional)

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

**CcRingout.Indication parameters**

Refer to 11.2.3 (parameters for CcRingout.Request primitive).

**CcRingout.Response parameters (rej)**

error: reason for failure of a CC Call

**CcCancel.Request and CcCancel.Indication parameters**

Refer to 11.2.3 (parameters for CcCancel.Request and CcCancel.indication primitive).

**CcSuspend.Indication parameters**

Refer to 11.2.3 (parameters for CcSuspend.Request primitive).

**CcResume.Indication parameters**

Refer to 11.2.3 (parameters for CcResume.Request primitive).

### 11.3.4 Call States

| CC State | Description |
|---|---|
| CC-Idle | This state exists if SS-CC is not active. |
| CC-Await-Call-Completion | This state exists while waiting for the incoming CC Call after having indicated that User B is not busy. |
| CC-Invoked-User-B | This state exists while User B is monitored as a result of a received CC Request. |
| CC-Wait-User-B-Alert | This state exists after a CC Call has been extended to User B, while waiting for acceptance (alerting or connect). |

## 11.4 Timers

### 11.4.1 Timers at User A's endpoint

User A's endpoint shall implement the following timers:

**Timer T1: SS-CC Request Protection**

This timer is started when sending a *ccbsRequest* or *ccnrRequest* invoke APDU and is stopped on receipt of a response.

If timer T1 expires, a failure indication is sent to User A.

Timer T1 shall have a duration in the range 10-30 seconds.

**Timer T2: SS-CC service duration**

This timer is started on receipt of a *ccbsRequest* or *ccnrRequest* return result APDU and stopped on completion of the CC Call or if the CC Request is cancelled.

If timer T2 expires, the CC Request is cancelled.

The duration of timer T2 is in the range of from 1 minute to 60 minutes, with a default of 15 minutes when started on receipt of a *ccbsRequest* return result APDU.

The duration of timer T2 is in the range of from 1 minute to 1440 minutes (i.e. 24 hours), with a default of 60 minutes when started on receipt of a *ccnrRequest* return result APDU.

**Timer T3: SS-CC Recall**

This timer is started when the CC Recall is indicated to User A and stopped when User A responds.

If timer T3 expires, the CC Request is cancelled.

Timer T3 shall have a duration in the range of 10-30 seconds.

### 11.4.2 Timers at User B's endpoint

None.

# 12 Operations in support of Call Completion supplementary service

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

```
Call-Completion-Operations
      { itu-t recommendation h 450 9 version1(0) call-completion-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)}
               EXTENSION, Extension{} FROM
                        Manufacturer-specific-service-extension-definition
                        { itu-t recommendation h 450 1 version1(0) msi-definition(18) }
               CallIdentifier FROM H323-MESSAGES -- see ITU-T H.225.0
               EndpointAddress FROM
                        Addressing-Data-Elements
               { itu-t recommendation h 450 1 version1(0) addressing-data-elements
               (9) }
                MixedExtension, undefined FROM Call-Hold-Operations
                        {itu-t recommendation h 450 4 version1(0) call-hold-operations(0)}
               supplementaryServiceInteractionNotAllowed FROM H4501-General-Error-List
```

```
                    { itu-t recommendation h 450 1 version1(0)
                    general-error-list (1) }
                BasicService FROM Message-Waiting-Indication-Operations
                    {itu-t recommendation h 450 7 version1(0)
                    message-waiting-operations(0)};


H323CallCompletionOperations  OPERATION ::=
{ccbsRequest | ccnrRequest | ccCancel | ccExecPossible | ccRingout | ccSuspend | ccResume }


ccbsRequest     OPERATION       ::=
    {
                ARGUMENT        CcRequestArg
                RESULT      CcRequestRes
                ERRORS      { shortTermRejection | longTermRejection | undefined |
supplementaryServiceInteractionNotAllowed
                    }
                CODE            local: 40
    }


ccnrRequest     OPERATION       ::=
    {
                ARGUMENT        CcRequestArg
                RESULT      CcRequestRes
                ERRORS      { shortTermRejection | longTermRejection | undefined |
                    supplementaryServiceInteractionNotAllowed
                    }
                CODE            local: 27
    }


ccCancel        OPERATION       ::=
    {
                ARGUMENT            CcArg OPTIONAL TRUE
                RETURN RESULTFALSE
                ALWAYS RESPONDS     FALSE
                CODE                local: 28
    }


ccExecPossible  OPERATION       ::=
    {
                ARGUMENT            CcArg OPTIONAL TRUE
                RETURN RESULTFALSE
                ALWAYS RESPONDS     FALSE
                CODE                local: 29
    }


ccRingout       OPERATION       ::=
    {
                ARGUMENT            CcShortArg OPTIONAL TRUE
                RETURN RESULTFALSE
                ERRORS      {
                    remoteUserBusyAgain | failureToMatch |
                    undefined
                    }
                CODE            local: 31
    }
```

```
ccSuspend       OPERATION       ::=
        {
                ARGUMENT                CcShortArg OPTIONAL TRUE
                RETURN RESULTFALSE
                ALWAYS RESPONDS FALSE
                CODE                    local: 32
        }


ccResume        OPERATION       ::=
        {
                ARGUMENT                CcArg OPTIONAL TRUE
                RETURN RESULTFALSE
                ALWAYS RESPONDS FALSE
                CODE                    local: 33
        }


CcRequestArg    ::=     SEQUENCE
        {
                numberA                 EndpointAddress,
                numberB                 EndpointAddress,
                ccIdentifier        CallIdentifier OPTIONAL,
                        -- If present, it shall be used as the prime mechanism for associating
                        -- the unsuccessful call, the CC Request and the CC Call.
                service                 BasicService,

                can-retain-service  BOOLEAN,
                retain-sig-connection   BOOLEAN OPTIONAL,
                extension               SEQUENCE SIZE (0..255) OF MixedExtension OPTIONAL,
                ...
        }


CcRequestRes    ::=     SEQUENCE
        {
                retain-service          BOOLEAN,
                extension               SEQUENCE SIZE (0..255) OF MixedExtension OPTIONAL,
                ...
        }


CcArg           ::=     CHOICE
        {
                shortArg    CcShortArg,     -- if signalling conection is retained
                longArg     CcLongArg,      -- if connection release option applies
                ...
        }


CcShortArg      ::=     SEQUENCE
        {
                ccIdentifier    CallIdentifier OPTIONAL,
                extension           SEQUENCE SIZE (0..255) OF MixedExtension OPTIONAL,
                ...
        }


CcLongArg       ::=     SEQUENCE
        {
                numberA             EndpointAddress OPTIONAL,
                numberB             EndpointAddress OPTIONAL,
                ccIdentifier    CallIdentifier OPTIONAL,
```

```
        service             BasicService OPTIONAL,
                -- these elements are used to identify the proper CC Request.
                -- The ccIdentifier if present shall be the prime mechanism for this purpose.
        extension           SEQUENCE SIZE (0..255) OF MixedExtension OPTIONAL,
        ...
    }


shortTermRejection      ERROR           ::=
    { CODE              local: 1010}

longTermRejection       ERROR           ::=
    { CODE              local: 1011}

remoteUserBusyAgain     ERROR           ::=
    { CODE              local: 1012}

failureToMatch          ERROR           ::=
    { CODE              local: 1013}


END -- of Call-Completion-Operations
```

## 13 Specification and Description Language (SDL) diagrams for SS-CCBS

The procedures for Call Completion signalling entities are described in SDL form in Figures 19 through 29. The SDLs only show SS-CC 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:

ack Acknowledgement

BC Basic Call

conf Confirmation

conn Connection

err Return error APDU

ind Indication

inv Invoke APDU

rej Reject APDU or Rejection

res Return result APDU

sc Signalling Connection

sig Signalling

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-CC on behalf of an endpoint are not provided.

The symbols used in the following SDLs are defined in Figure 18.

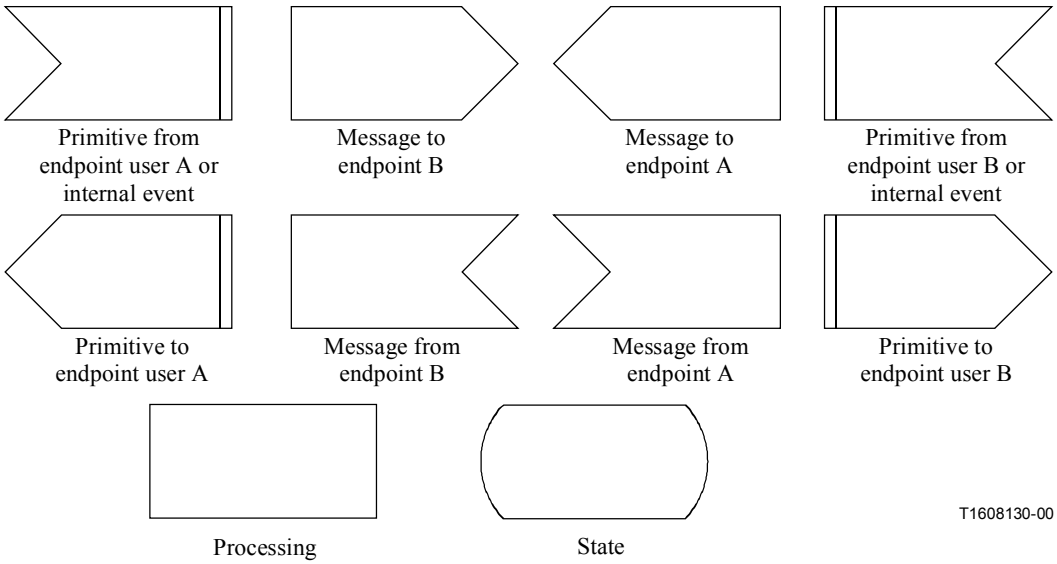



**Figure 18/H.450.9 – SDL symbols**

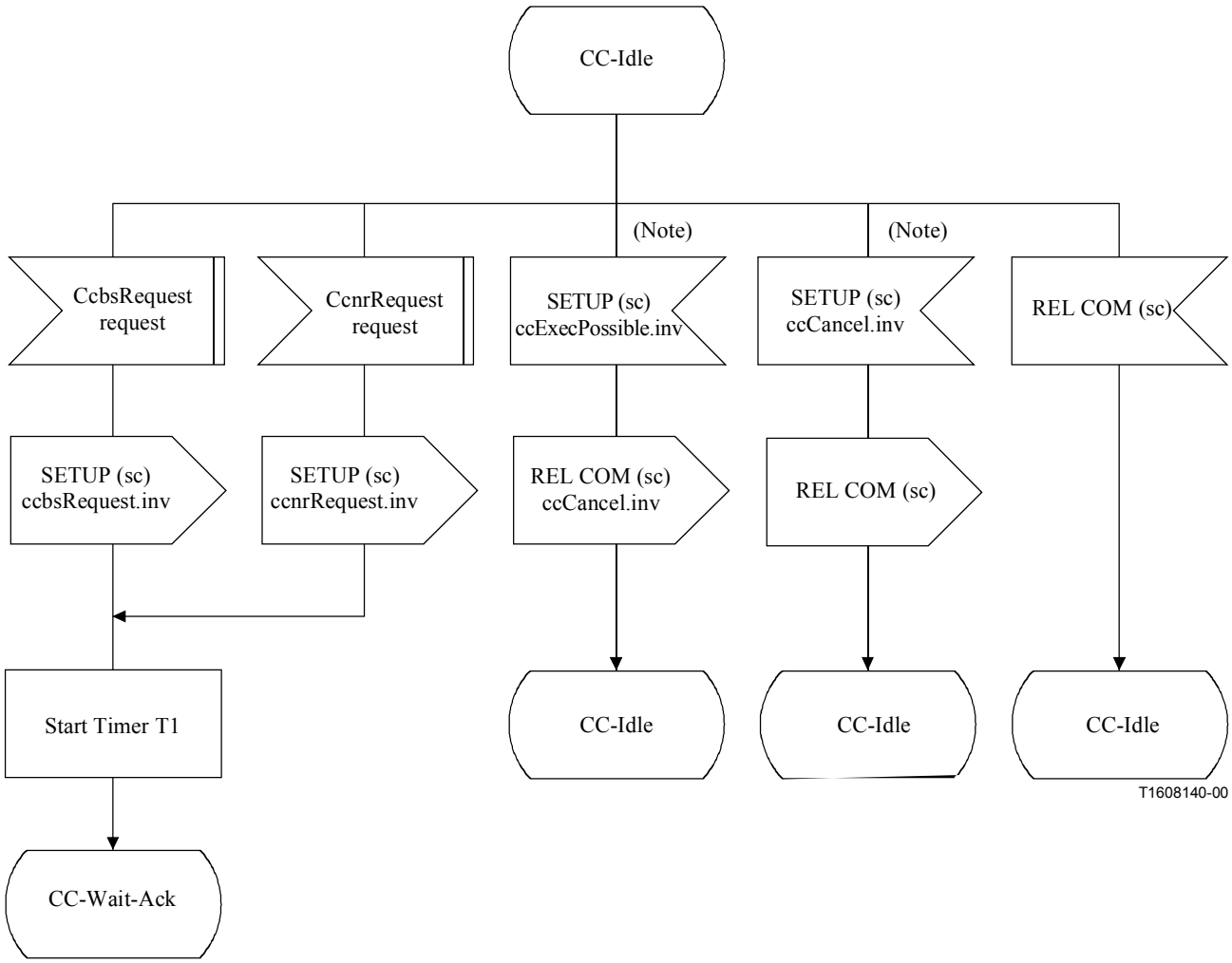
### 13.1 Behaviour of User A's endpoint

Figures 19 to 25 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 a prefix BC-;
- internal signal, e.g. timer expiry.

Input signals from the right and output signals to the right represent messages to the peer SS-Control entity (i.e. in User B's endpoint) which carry SS-CC control information. Call independent signalling connection messages are indicated by 'sc'. Messages not marked 'sc' are call related.




NOTE – This event can only occur in error situations.

**Figure 19/H.450.9 – User A's endpoint behaviour**

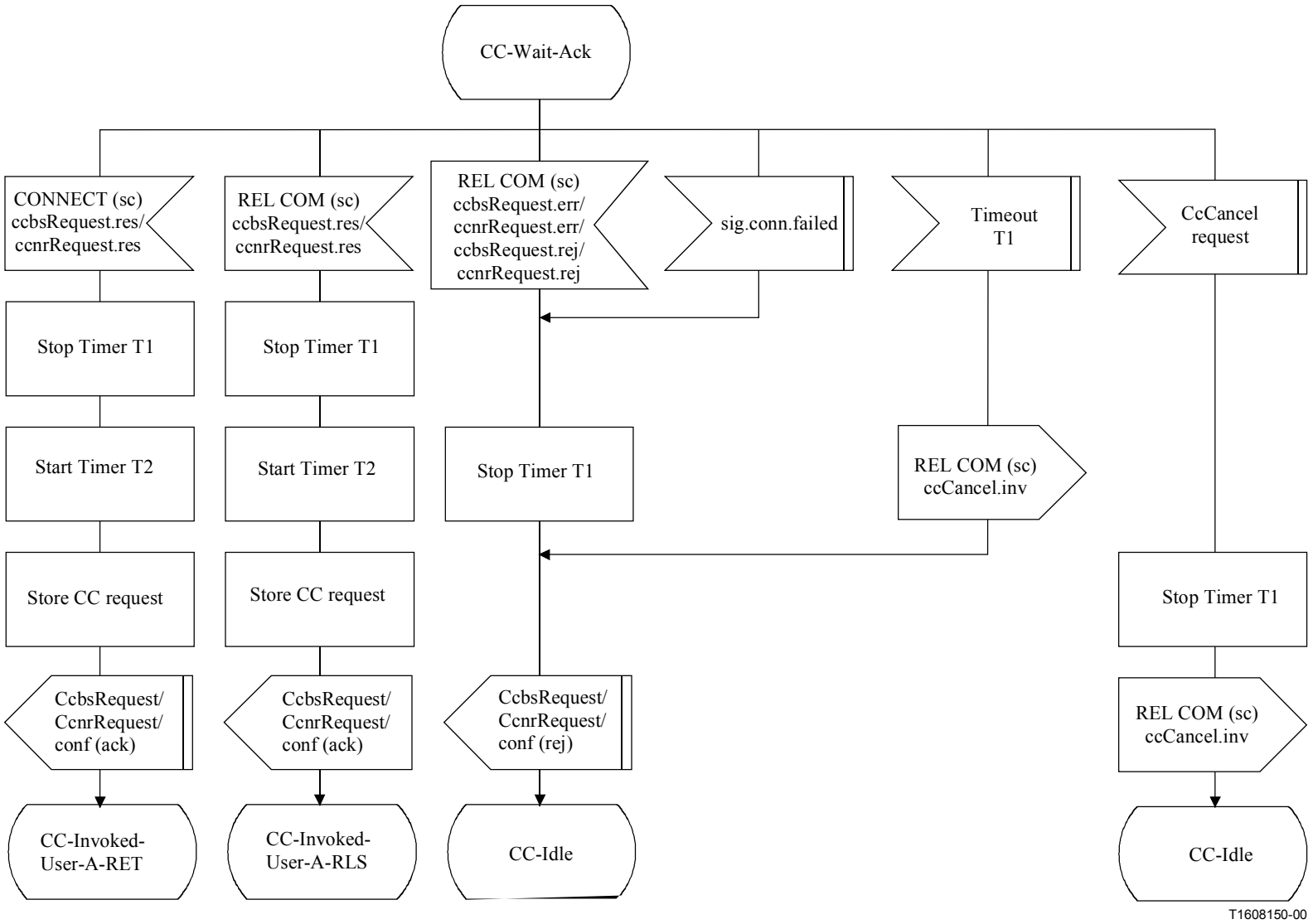



**Figure 20/H.450.9 – User A's endpoint behaviour**

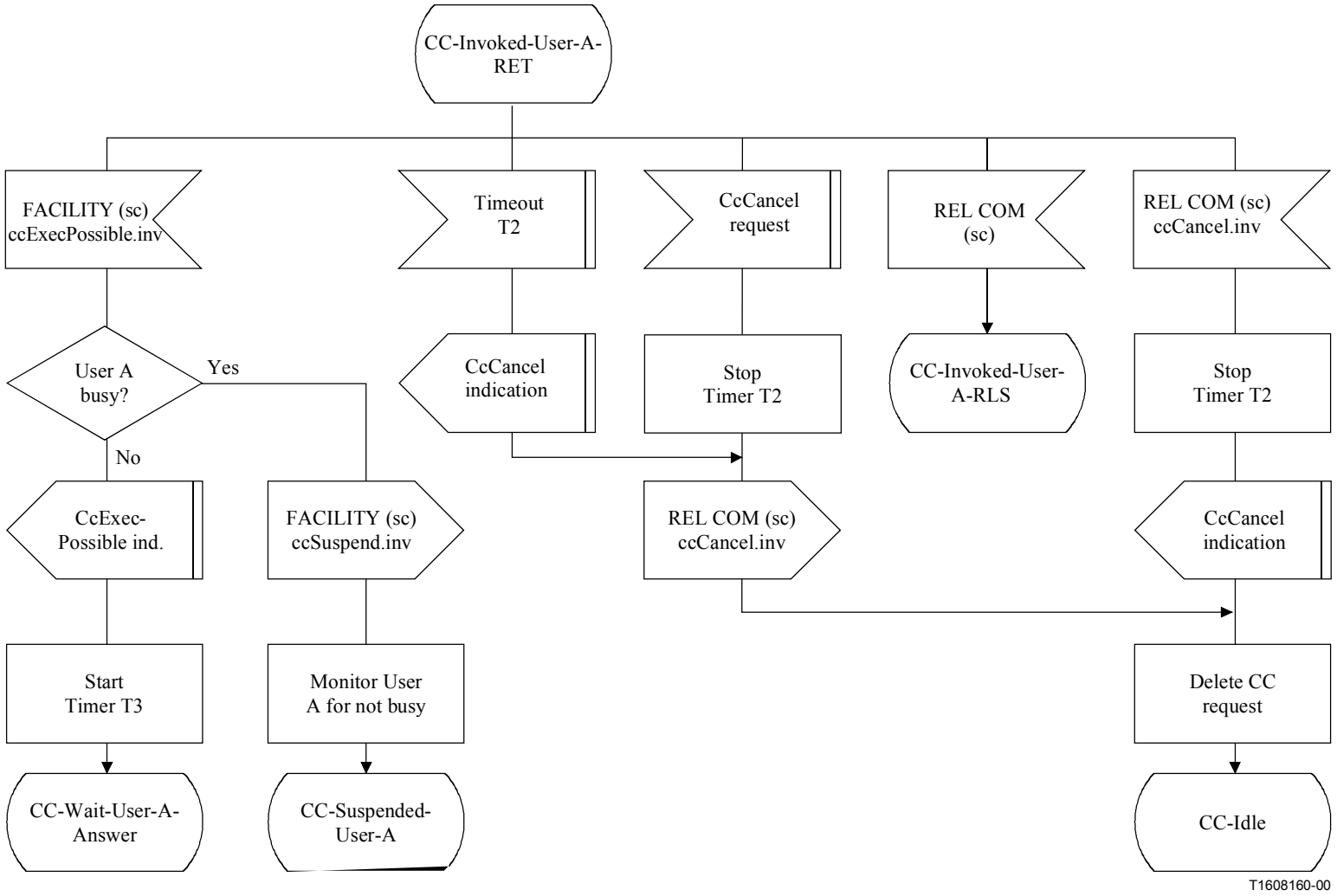



**Figure 21/H.450.9 – User A's endpoint behaviour**

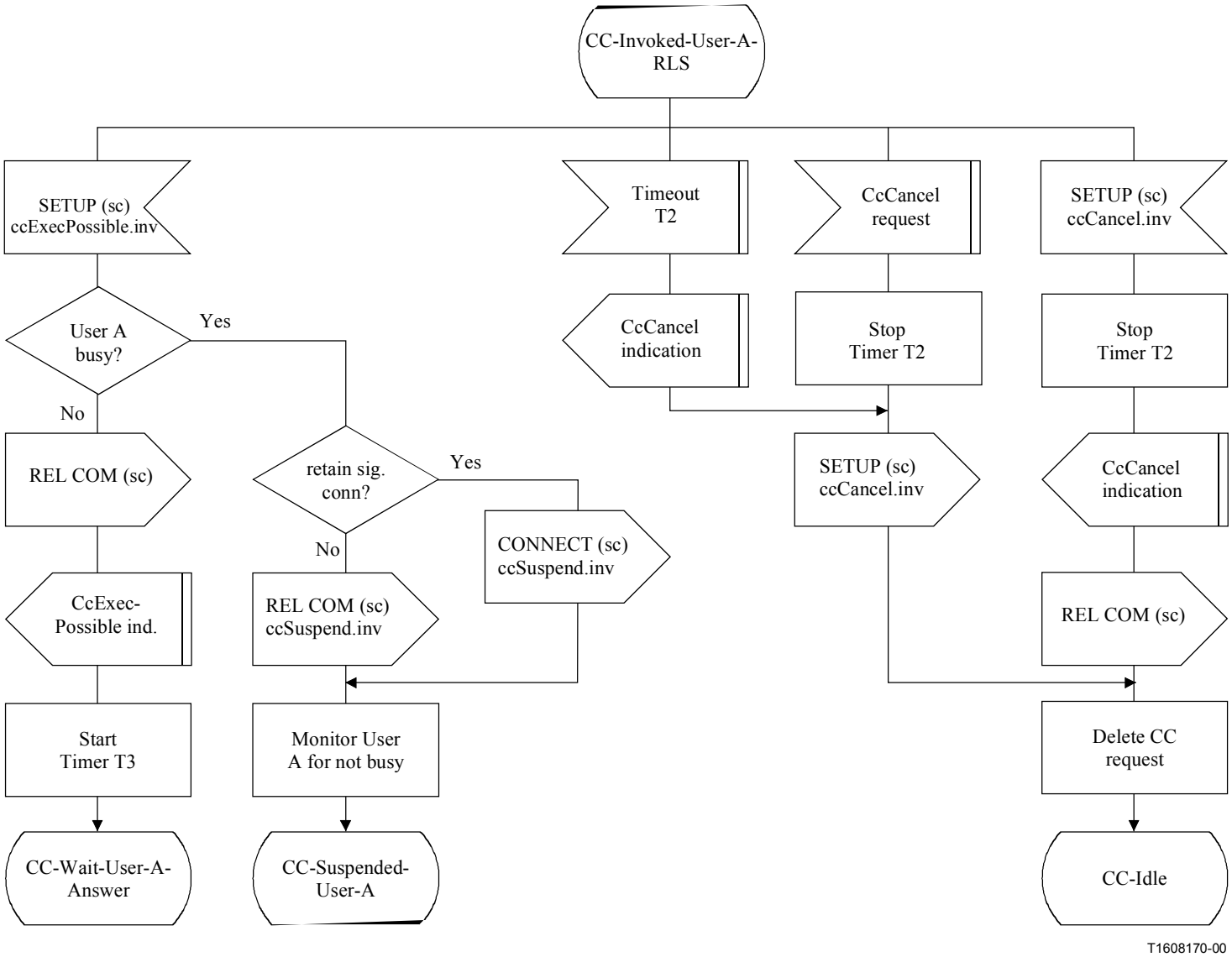



**Figure 22/H.450.9 – User A's endpoint behaviour**

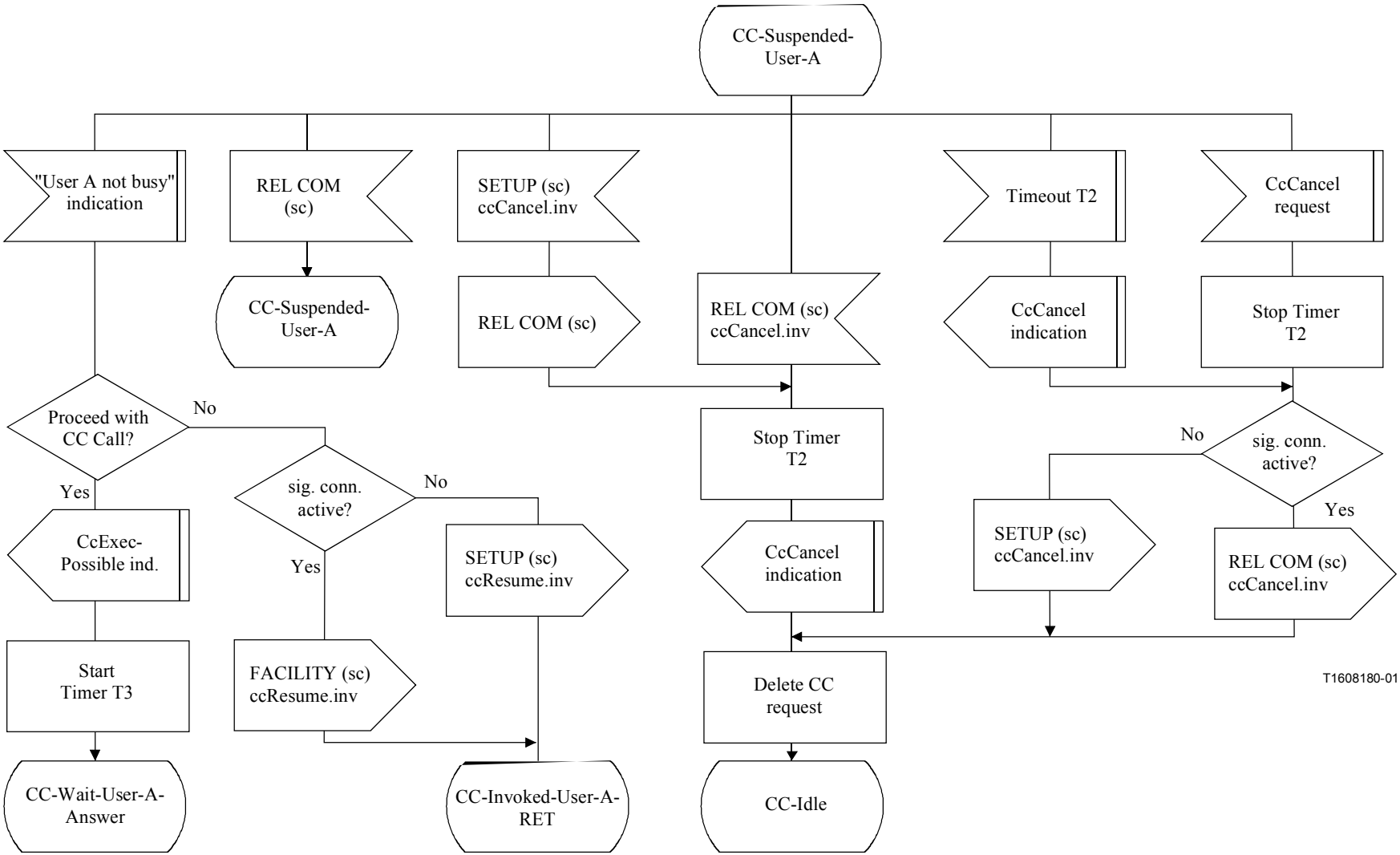



**Figure 23/H.450.9 – User A's endpoint behaviour**

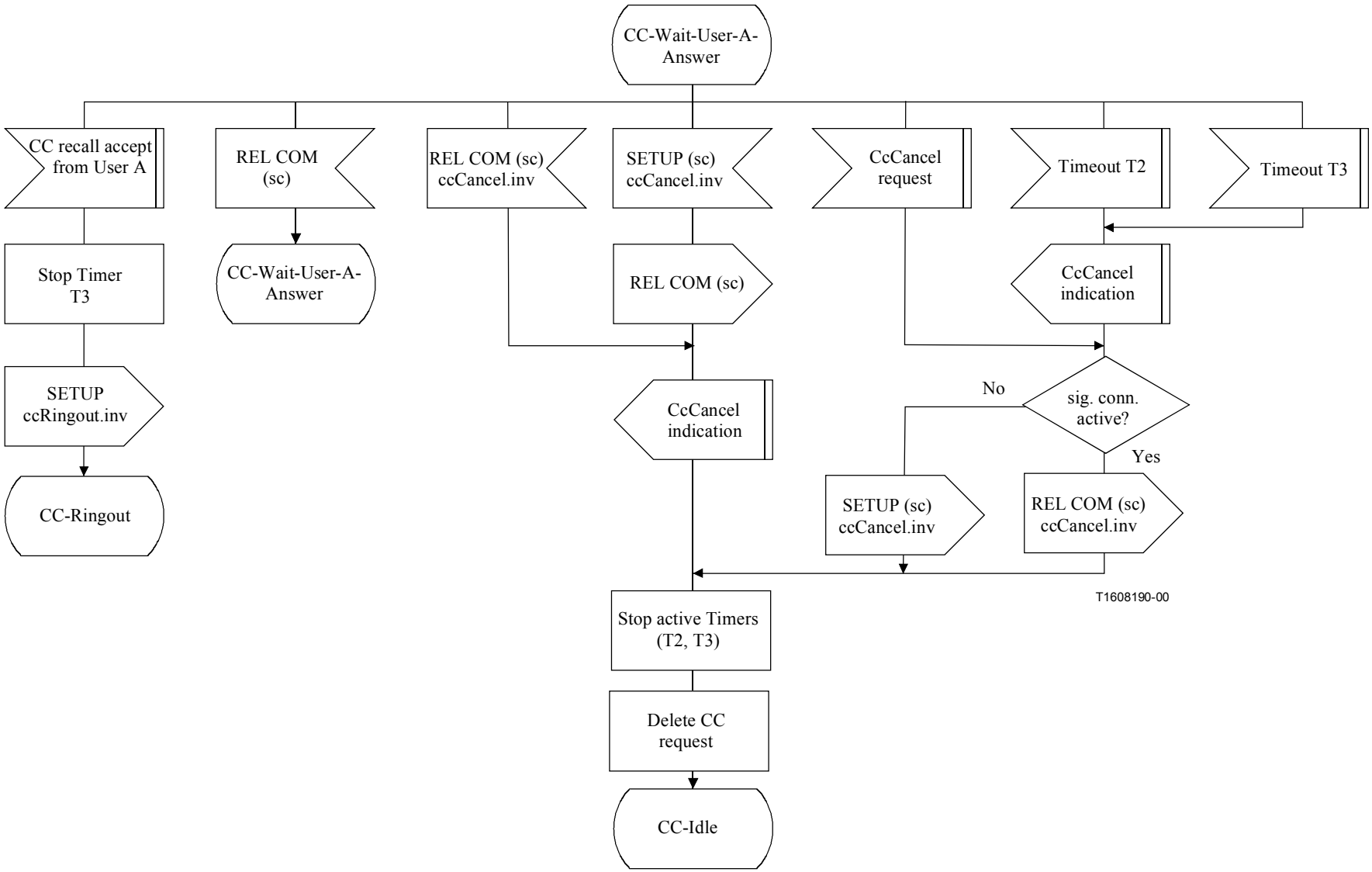



**Figure 24/H.450.9 – User A's endpoint behaviour**

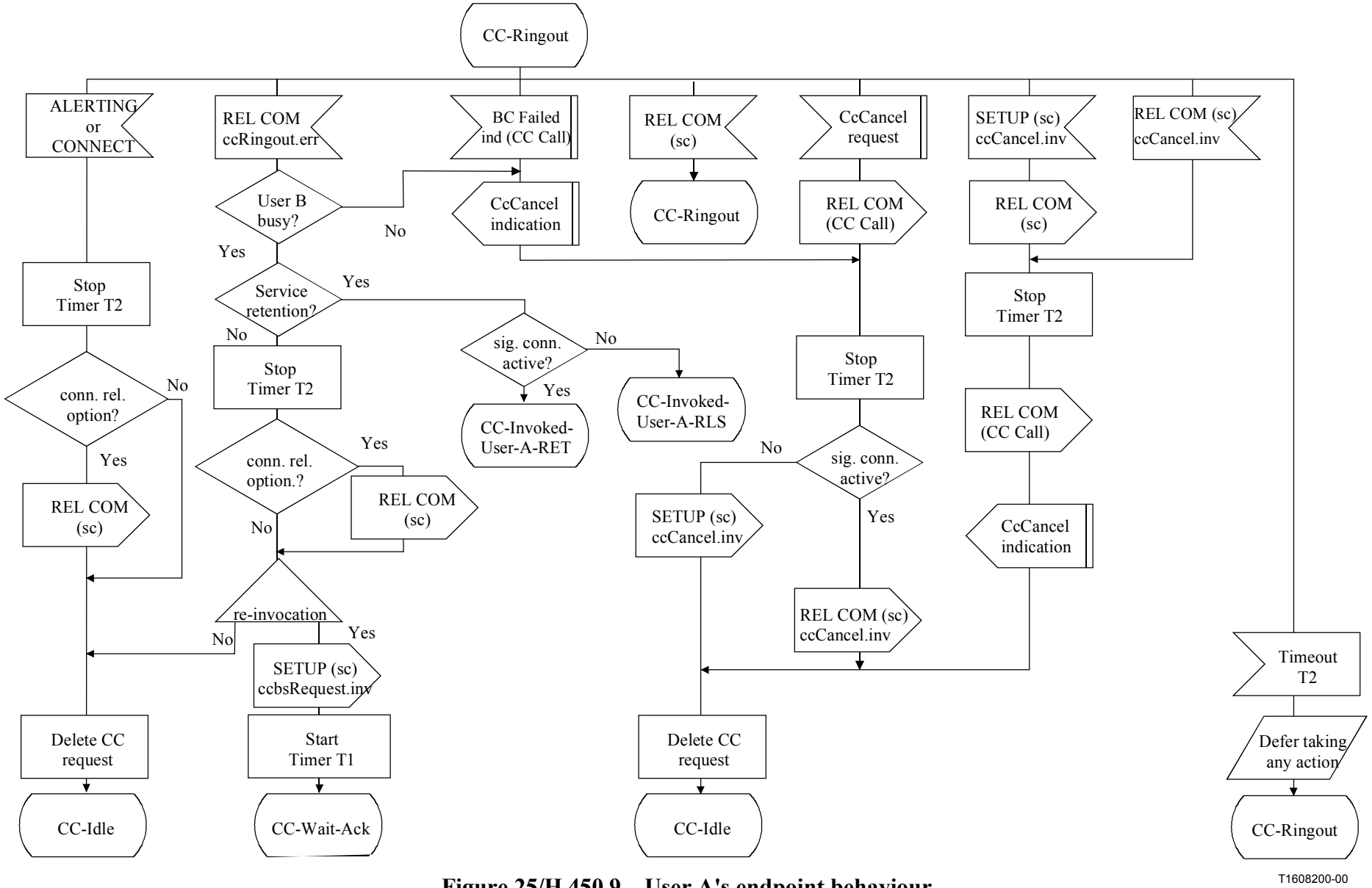



**Figure 25/H.450.9 – User A's endpoint behaviour**

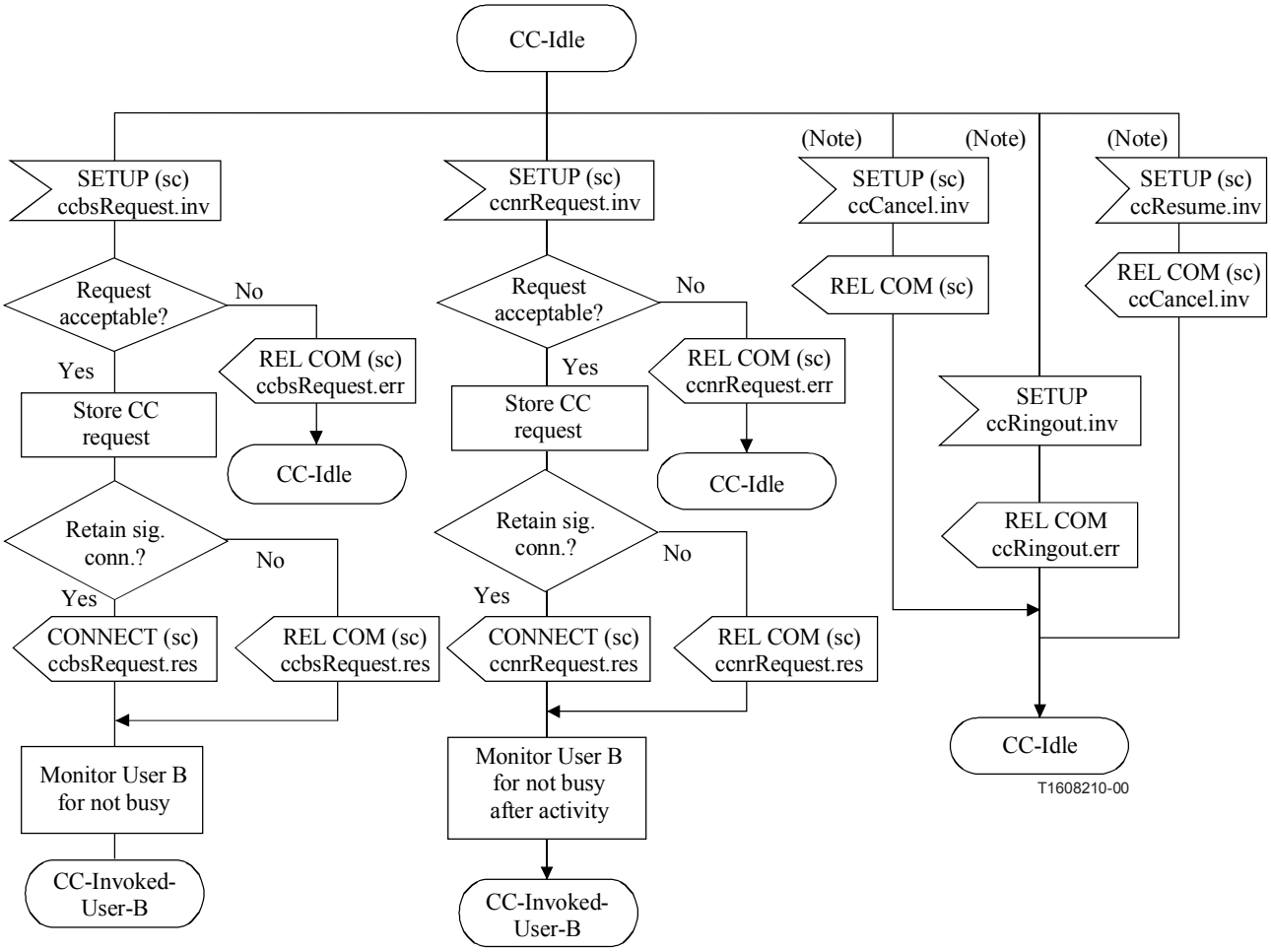
## 13.2 Behaviour of User B's endpoint

Figures 26 to 29 show the behaviour of User B's endpoint.

Input signals from the left and output signals to the left represent messages from and to the peer SS-Control entity (i.e. in User A's endpoint) which carry SS-CC control information. Call independent signalling connection messages are indicated by 'sc'. Messages not marked 'sc' are call related.

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

- from or to the called user (User B);
- from or to basic call control; these primitives are indicated by a prefix BC;
- internal signals.




NOTE – This event can only occur in error situations.

**Figure 26/H.450.9 – User B's endpoint behaviour**

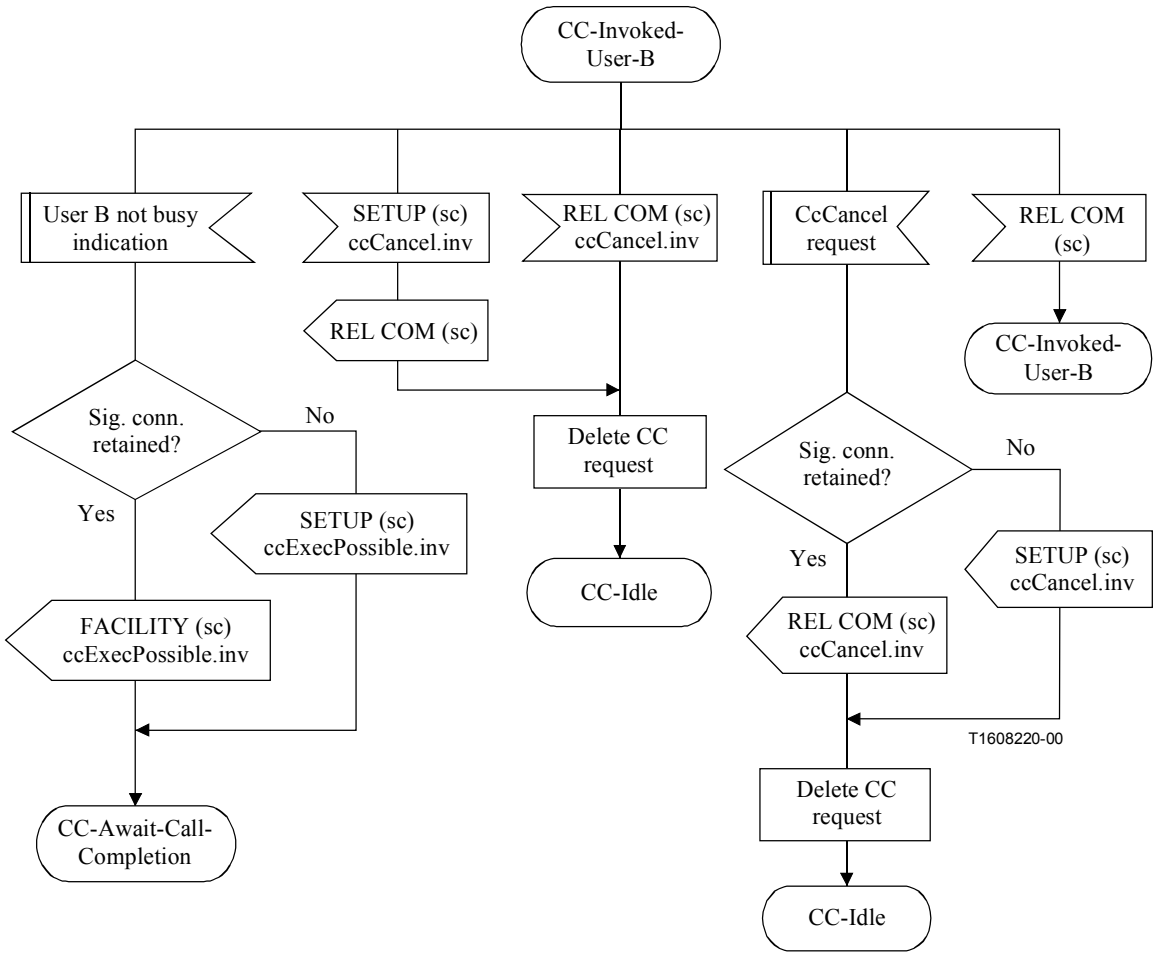



**Figure 27/H.450.9 – User B's endpoint behaviour**

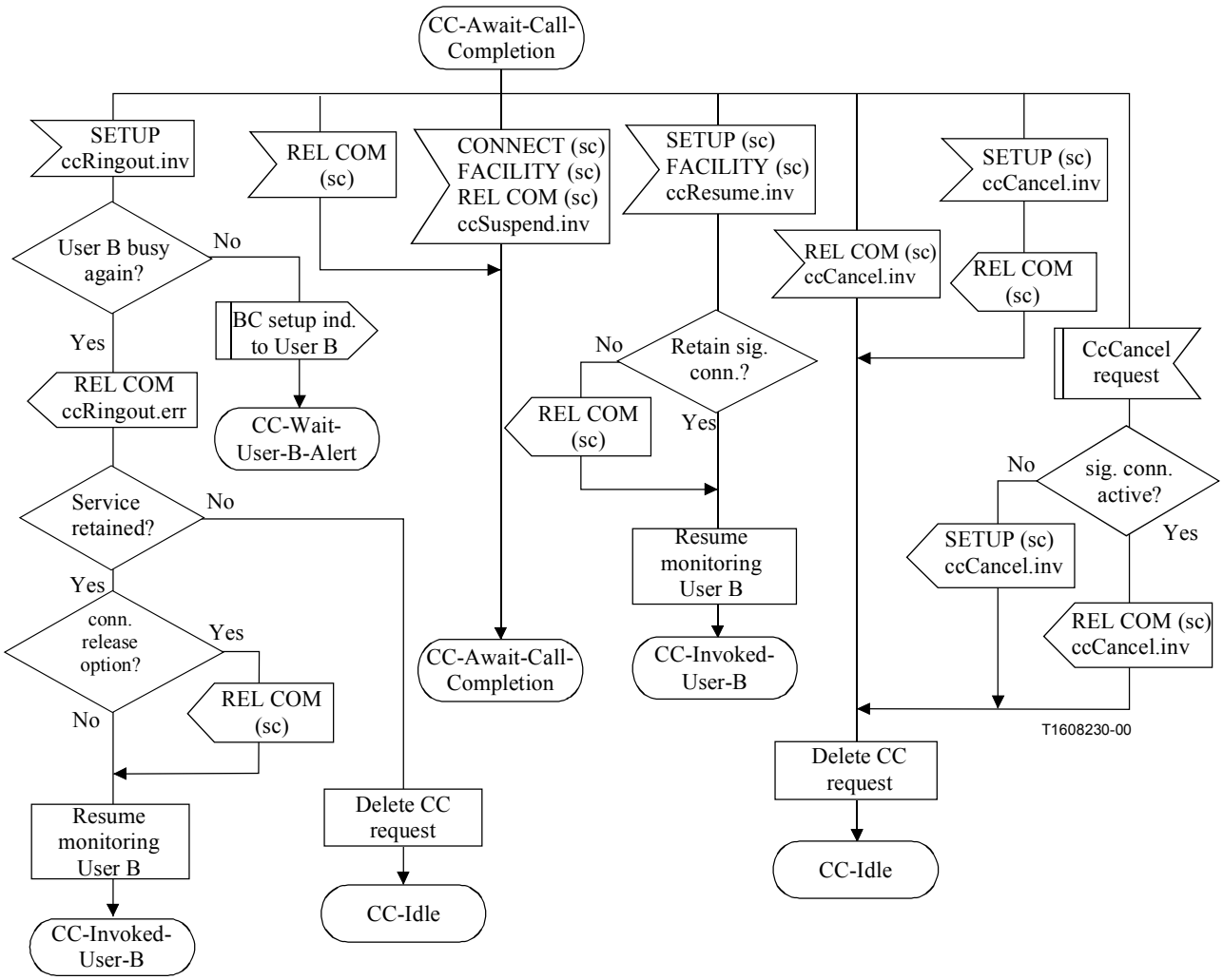



**Figure 28/H.450.9 – User B's endpoint behaviour**

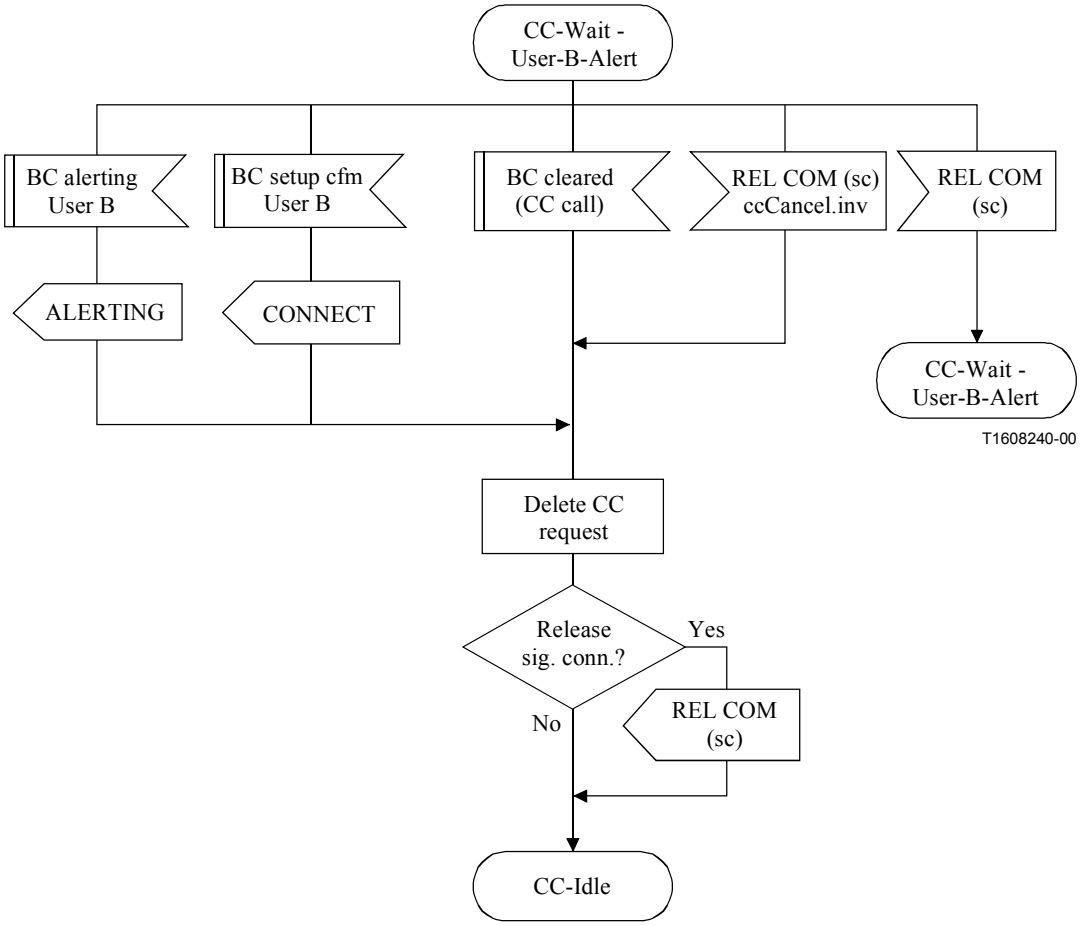



**Figure 29/H.450.9 – User B's endpoint behaviour**

## 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 | 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