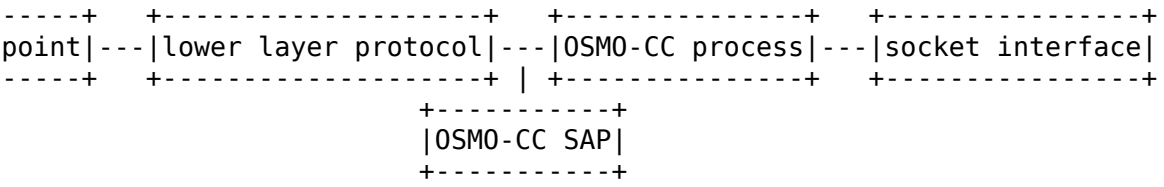
OSMO-CC

Doc version 0.9 (Dec 21 2020)

Table of Contents

1 Overview



OSMO-CC is an unified interface (**OSMO-CC SAP**) between a **lower layer protocol** and a call control process. This call control process can be handled by an application or provides a socket interface to interconnect to another OSMO-CC interface. The lower layer protocol may serve a physical or virtual **endpoint**, like SIP, ISDN, GSM, DECT, FXS, FXO or even bluetooth phones and headsets. The interface structure is always the same, no matter what lower layer protocol, nor what modes (NT-Mode, TE-Mode, MS, BS, PtP, PtMP) are used. Special protocol requirements must be handled by the lower layer protocol itself.

The interface is almost symmetric, so that two lower layer protocols can be connected with minor **call processing**. This interconnection can be provided by a **socket interface**. The remote endpoint may also be of any type or mode.

Unspecified messages may be received from lower layer protocol. They may be forwarded via socket interface. The lower layer protocol may ignore unsupported messages. Messages that change the call state are mandatory for all lower layer protocols, even if not supported (at certain states). In this case the lower layer protocol must convert these messages to get into the right call state.

There may be a message queue between lower layer protocol and call processing, so race conditions may occur. These race conditions must be handled by the call processing and lower layer interface.

SDP information element is used to negotiate codecs between two lower layer protocols. The first message (setup) is used to offer codec information. The originating interface must now be able to receive RTP data as being offered. The first reply with an SDP information will complete the negotiation and allows the originating lower layer protocol to transmit RTP data as negotiated.

At least a-law **and** µ-law codec shall be offered by the originating endpoint. The terminating endpoint shall always be able transcode any local codec from and to a-law and µ-law. For special setups this rule must not apply. In this case, at least one offered codec must be supported by both endpoints. If the negotiation fails, the terminating endpoint shall reject the call and notification shall be included in log file.

2 Messages

Message	Description
CC-SETUP-REQ	Initiates a call towards lower layer.(Note 3 for outgoing calls)
CC-SETUP-IND	The lower layer initiates a call. (Note 3 for incoming calls)
CC-REJ-REQ	Reject the call towards lower layer. (Note 3 for incoming calls)
CC-REJ-IND	The lower layer rejects the call. (Note 3 for outgoing calls)
CC-SETUP-ACK-REQ	Request more digits from lower layer. (overlap dialing)
CC-SETUP-ACK-IND	The lower layer requests more digits. (overlap dialing)
CC-PROC-REQ	Tell the lower layer that the call is proceeding.
CC-PROC-IND	The lower layer tells that the call is proceeding.
CC-ALERT-REQ	Tell the lower layer that the call is ringing.
CC-ALERT-IND	The lower layer tells that the call is ringing.
CC-SETUP-RSP	Tell the lower layer that the call has been answered. (Note 3 for incoming calls)
CC-SETUP-CNF	The lower layer tells that the call has been answered. (Note 3 for outgoing calls)
CC-SETUP_COMP-REQ	Acknowledge the answer towards lower layer.
CC-SETUP_COMP-IND	The lower layer acknowledges the answer. (Note 3 for incoming calls)
CC-DISC-REQ	Disconnect the call towards lower layer.
CC-DISC-IND	The lower layer disconnects the call.
CC-REL-REQ	Release the call towards lower layer. (Note 3)
CC-REL-CNF	The lower layer confirms the release. (Note 3)
CC-REL-IND	The lower layer releases the call. (Note 3)
CC-PROGRESS-REQ	Tell the lower layer about change in progress of the call.
CC-PROGRESS-IND	The lower layer tells about change in progress of the call.
CC-NOTIFY-REQ	Notify the lower layer about events during call.
CC-NOTIFY-IND	The lower layer notifies about events during call. (Note 1)
CC-INFO-REQ	Send additional digits towards lower layer. (Note 2)
CC-INFO-IND	Receive additional digits from lower layer. (Note 2)
CC-DUMMY-REQ	Dummy message to check link. (Note 4)
CC-MODIFY_REQ/ IND	Updating SDP media desctiption (Note 5)
CC-MODIFY_CNF/ RSP	Reply with SDP to modification request

Note 1: There are no messages to control features like SUSPEND/RESUME or HOLD/RETRIEVE or call forwarding directly. It is the task of the lower layer. The lower layer may send a notification about what is going on.

Note 2: There are no messages to control DTMF generation/detection in detail. It is the task of the lower layer. The tones info is sent/received via INFO message.

Note 3: Mandatory to be handled by the lower layer protocol.

Note 4: The dummy message is sent on socket to check if the link is disconnected. This is required, because link failure must be detected in reasonable time.

Note 5: This is currently optional. It may be possible that endpoints supporting updating SDP (like SIP re-invite) will loose connections with endpoints that does not support updating SDP. It is for further study how to deal with this problem.

3 States

State	Description
IDLE	No call, this state may exist right after a call process has been created or right before it is destroyed.
INIT-OUT	A call towards lower layer was initiated via CC-SETUP-REQ message. In this state the process waits for acknowledgment or reject from lower layer.
INIT-IN	A call from lower layer was initiated via CC-SETUP-IND message. In this state the process needs to acknowledge or reject towards lower layer.
OVERLAP-OUT	The CC-SETUP-ACK-IND message was received from lower layer. In this state the process can send further digits via CC-INFO-REQ message towards lower layer. (overlap dialing)
OVERLAP-IN	The CC-SETUP-ACK-REQ message was sent towards lower layer. In this state the process waits for further digits via CC-INFO-IND message from lower layer. (overlap dialing)
PROCEEDING-OUT	The CC-PROC-IND message was received from lower layer. The address is complete, the call is proceeding. In this state the process can receive CC-PROGRESS-IND message from lower layer.
PROCEEDING-IN	The CC-PROC-REQ message was sent towards lower layer. The address is complete, the call is proceeding. In this state the process send CC-PROGRESS-REQ message towards lower layer.
ALERTING-OUT	The CC-ALERT-IND message was received from lower layer. The call is ringing.
ALERTING-IN	The CC-ALERT-REQ message was sent towards lower layer. The call is ringing.
CONNECTING-OUT	The CC-SETUP-CNF message was received from lower layer. The call has been answered. It must be completed by sending CC-SETUP-COMP-REQ message towards lower layer.
CONNECTING-IN	The SETUP_RSP message was sent towards lower layer. The call has been answered. It must be completed by receiving CC-SETUP-COMP-IND message from lower layer.
ACTIVE	The call has been acknowledged by CC-SETUP-COMP-REQ or CC-SETUP-COMPL-IND. The caller and the callee talk to each other.
DISCONNECTING-OUT	The call was disconnected towards lower layer via CC-DISC-REQ message. A disconnect tone may be available. In this state the process waits for CC-REL-IND from lower layer. This will terminate the process. (Note)
DISCONNECTING-IN	The call was disconnected from lower layer via CC-DISC-IND message. A disconnect tone may be available. In this state the process waits for sending CC-REL-REQ towards lower layer.
RELEASING-OUT	The REL_REQ message was sent towards lower layer. In this state the process waits for CC-REL-CNF from lower layer. This will terminate the process.

Note: There is no RELEASING-IN state, because a CC-REL-IND message will directly bring the process into IDLE state without any further message.

4 State Transitions

4.1 Call setup to lower layer protocol

Old State	Event	New State	MS	BS	TE	NT	SIP	Note
IDLE	CC-SETUP-REQ	INIT-OUT	*	*	*	*	*	
INIT-OUT	CC-SETUP-ACK-IND	OVERLAP-OUT			*	*		
	CC-PROC-IND	PROCEEDING-OUT	*	*	*	*	*	
	CC-ALERT-IND	ALERTING-OUT				*		
	CC-SETUP-CNF	CONNECTING-OUT				*		1
OVERLAP-OUT	CC-PROGRESS-IND				*	*		
	CC-INFO-REQ				*	*		
	CC-PROC-IND	PROCEEDING-OUT			*	*		
	CC-ALERT-IND	ALERTING-OUT			*	*		
	CC-SETUP-CNF	CONNECTING-OUT			*	*		1
PROCEEDING-OUT	CC-PROGRESS-IND		*		*	*	*	
	CC-NOTIFY-IND				*	*	*	
	CC-ALERT-IND	ALERTING-OUT	*	*	*	*	*	
	CC-SETUP-CNF	CONNECTING-OUT	*	*	*	*	*	1
ALERTING-OUT	CC-PROGRESS-IND				*		*	
	CC-NOTIFY-IND		*	*	*	*	*	
	CC-SETUP-CNF	CONNECTING-OUT	*	*	*	*	*	1
CONNECTING-OUT	CC-NOTIFY-IND		*	*	*	*	*	
	CC-SETUP-COMP-REQ	ACTIVE			*			

The listed protocols are MS (GSM mobile station mode), BS (GSM base station mode), TE (ISDN terminal equipment mode), NT (ISDN network termination mode) and SIP. The asterisks indicate which messages are supported by that protocol in each state. This is just an example, other protocols may use OSMO-CC too.

Note 1: If lower layer protocol does not support this transition, it shall ignore it and enter ACTIVE state. If it receives a CC-SETUP-COMPL-REQ afterwards, it may forward included information elements using CC-NOTIFY-REQ or appropriate message.

4.2 Call setup from lower layer protocol

Old State	Event	New State	MS	BS	TE	NT	SIP	Note
IDLE	CC-SETUP-IND	INIT-IN	*	*	*	*	*	
INIT-IN	CC-SETUP-ACK-REQ	OVERLAP-IN			*	*		2
	CC-PROC-REQ	PROCEEDING-IN	*	*	*	*	*	
	CC-ALERT-REQ	ALERTING-IN			*			3
	CC-SETUP-RSP	CONNECTING-IN			*			3;1
OVERLAP-IN	CC-PROGRESS-REQ				*	*		
	CC-INFO-IND				*	*		
	CC-PROC-REQ	PROCEEDING-IN			*	*		
	CC-ALERT-REQ	ALERTING-IN			*	*		
	CC-SETUP-RSP	CONNECTING-IN			*	*		1
PROCEEDING-IN	CC-PROGRESS-REQ			*	*	*	*	
	CC-NOTIFY-REQ				*	*	*	
	CC-ALERT-REQ	ALERTING-IN	*	*	*	*	*	
	CC-SETUP-RSP	CONNECTING-IN	*	*	*	*	*	1
ALERTING-IN	CC-PROGRESS-REQ					*	*	
	CC-NOTIFY-REQ		*	*	*	*	*	
	CC-SETUP-RSP	CONNECTING-IN	*	*	*	*	*	1
CONNECTING-IN	CC-NOTIFY-REQ		*	*	*	*	*	
	CC-SETUP-COMPL-IND	ACTIVE				*		

Note 1: If lower layer protocol does not support this transition, it shall send a CC-SETUP-COMPL-IND when changing to ACTIVE state.

Note 2: If overlap dialing is not supported by lower layer protocol, the lower layer shall treat CC-SETUP-ACK-REQ as it would have received a CC-PROC-REQ message. If it receives a CC-PROC-REQ afterward, it may forward included information elements using CC-PROGRESS-REQ or appropriate message. Also the lower layer protocol should add the complete information element (IE_COMPLETE) to the CC-SETUP-IND message, which tells the other side that the number is complete and overlap dialing is not required. This does not guaranty that the remote side does not send CC-SETUP-ACK-REQ.

Note 3: If lower layer protocol does not support this transition, it shall assume a CC-PROC-REQ was received before.

4.3 Call in active state

Old State	Event	New State	MS	BS	TE	NT	SIP	Note
ACTIVE	CC-NOTIFY-IND		*	*	*	*		1
	CC-NOTIFY-REQ		*	*	*	*		1
	CC-INFO-IND		*	*	*	*	*	1
	CC-INFO-REQ		*	*	*	*	*	1
	CC-MODIFY-REQ/ IND		*	*			*	2
	CC-MODIFY-CNF/ RSP		*	*			*	2

Note 1: Special lower layer infos, such as DTMF info / display info or hold/retrieve are sent via CC-INFO-* or CC-NOTIFY-* message.

Note 2: The lower layer sends an IND and the upper layer sends a RSP. The upper layer sends a REQ and the lower layer sends a CNF.

4.4 Call release from / towards lower layer

Old State	Event	New State	MS	BS	TE	NT	SIP	Note
INIT-OUT	CC-DISC-REQ	DISCONNECTING-OUT	*	*	*	*		
	CC-DISC-IND	DISCONNECTING-IN						
	CC-REJ-IND	IDLE	*	*	*	*	*	
INIT-IN	CC-DISC-IND	DISCONNECTING-IN	*	*	*	*		
	CC-DISC-REQ	DISCONNECTING-OUT						1
	CC-REJ-REQ	IDLE	*	*	*	*	*	
OVERLAP-OUT OVERLAP-IN PROCEEDING-OUT PROCEEDING-IN ALERTING-OUT ALERTING-IN CONNECTING-OUT CONNECTING-IN ACTIVE	CC-DISC-REQ	DISCONNECTING-OUT	*	*	*	*	*	
	CC-DISC-IND	DISCONNECTING-IN	*	*	*	*	*	
	CC-REL-IND	IDLE						
DISCONNECTING-OUT	CC-REL-IND	IDLE	*	*	*	*	*	2
	CC-DISC-IND	DISCONNECTING-COLL	*	*	*	*	*	4
	CC-REL-REQ	DISCONNECTING-OUT						7
DISCONNECTING-IN	CC-REL-REQ	RELEASING-OUT	*	*	*	*	*	
	CC-DISC-REQ	DISCONNECTING-COLL	*	*	*	*	*	4
	CC-REL-IND	DISCONNECTING-IN						7
DISCONNECTING-COLL	CC-REL-IND	IDLE	*	*	*	*		
	CC-REL-REQ	RELEASING-OUT	*	*	*	*		
RELEASING-OUT	CC-REL-CNF	IDLE	*	*	*	*		3
	CC-REL-IND	IDLE	*	*	*	*		5
IDLE	CC-REL-REQ	IDLE	*	*	*	*		5
other states	CC-REL-REQ	DISCONNECTING-OUT						6
	CC-REL-IND	DISCONNECTING-IN						6

Note 1: If the lower layer protocol does not allow a CC-DISC-REQ during INIT-IN state, it shall insert a CC-PROC-REQ message itself, to handle the CC-DISC-REQ correctly. IEs like codec informations need to be taken from CC-DISC-REQ and inserted into the CC-PROC-REQ.

Note 2: A “disconnect collision” at lower layer protocol may cause it to send a CC-REL-CNF instead of a CC-REL-IND, caused by an internal transition to RELEASING-OUT state. The lower layer protocol must track this condition (compare state with own states) and send CC-REL-IND when remote side has confirmed the release.

Note 3: A CC-REL-CNF is not transferred via socket interface. It confirms the CC-REL-REQ after the socket connection has been closed.

Note 4: When a disconnect is received from lower layer protocol and upper layer protocol, is is a disconnect collision. A special state is entered. A release in either direction changes that state.

Note 5: When a release request is sent towards lower layer protocol, a release indication may be received from lower layer protocol. Also when a release indication is received from lower layer protocol, a release request may be received from upper layer protocol. These are release collisions.

Note 6: To help lower and upper layer protocol to terminate a call without waiting for a reply, a release message may be sent at any state. This will terminate the relation to the layer it was received from. The Osmo-CC process will turn that message into a disconnect message, if required. Further messages are handled by OsmoCC process until the call is released by the other end. This is useful, if a layer fails and must terminate calls instantly. Also it can be used to end the call without further processing, just for convenience.

Note 7: A REL-* may be sent by upper or lower layer after sending a DISC-*. It is turned into a DISC-* by the Osmo-CC process. This may happen, if one layer finished sending disconnect tones, so that is sends a REL-* message that is then turned into a DISC-* message without progress indicator. The destination layer will then receive a second DISC-*. It may also happen, if the socket connection fails. This message can be safely ignored or be used to release the link or be used to provide locally generated tones from that time on. In any case, RTP connection should be released.

5. Information elements

IE	Description
IE_CALLED type plan digits	Called number
IE_CALLED_SUB subtype subdigits	Called number sub address
IE_COMPLETE	All called number digits are complete, no digits follow.
IE_CALLING type plan present screen digits	Calling or connected number
IE_CALLING_SUB subtype subdigits	Calling number sub address
IE_COMPLETE	Sending complete
IE_BEARER coding capability mode	ISDN/GSM Bearer capability
IE_NOTIFY notify	Notification Indicator (according to Q.931 and ff.)
IE_REDIR digits type plan present screen reason	Redirection/Redirecting number (in SETUP for a redirected call, in NOTIFY when a call gets redirected with Notification Indicator “CALL_IS_DIVERTING”).
IE_KEYPAD digits	Keypad
IE_DTMF digit(s) duration pause start stop	DTMF tones
IE_PROGRESS coding location progress	Call progress

IE_CAUSE socket q850 sip location	Cause for a message
IE_DISPLAY text	Display text on terminal
IE_INTERFACE name	Name of interface
IE_NETWORK type id	Network type of calling or connected endpoint.
IE_SDP sdp	SDP to describe and negotiate media stream
IE_COMPLETE	The called number, that is included in this message is complete. Further digits are not required to complete the call. The receiver may use this information to complete the without waiting for additional digits. If not enough digits are received for a call, the receiver may reject the call, because there are no more digits expected to be dialed.
IE_PRIVATE unique data	Private information element for vendor specific controls. The ‘unique’ field shall contain a random number chosen by the vendor. This information element shall be ignored, if the unique id does not match. The ‘data’ field is vendor specific.

6. Coding and Rules

6.1 Coding

All messages consist of a one byte message type followed by two bytes of message length in network order (without type and length) followed by 0..n information elements. Each information element consists of a one byte information element type followed by a two byte length in network order (without type and length) followed by a data structure.

All numerical values are stored in network order. This is not only true for the socket interface, but also for messages from and to lower layer protocol.

New messages and information elements can be added without changing the protocol version. If messages/elements are unknown to an endpoint, they shall be ignored.

Information elements may be repeated multiple times, so each information element can be an array of that elements. (E.g. calling number on ISDN line)

Information elements may be stored in any order. Repeated information elements may be separated by other information elements. The lower layer protocol has to sort the elements, if required.

All types and data structures are defined in “include/osmocom/cc/message.h” of libosmo-cc.

6.2 SDP content and Progress Indicator Rule

The offer/answer model for codec negotiation is adopted from SIP. See RFC 3264 for more details.

The offer of a codec shall be performed in the CC-SETUP-REQ/IND message, which includes SDP information element if a codec must be used.

The first response to that CC-SETUP-REQ/IND message that includes SDP information completes the codec negotiation. A response shall only be CC-SETUP-ACK-REQ/IND, CC-PROC-REQ/IND, CC-ALERT-REQ/IND, CC-SETUP-RSP/CNF, CC-DISC-REQ/IND and CC-PROGRESS-REQ/IND. Also the SDP information element shall be included in one message only and only once in that message.

If the lower layer protocol does not support the response message that includes the SDP information element, it shall store its SDP information and use it (later) in a message that completes codec negotiation. Alternatively the lower layer protocol may send a different response message that includes the SDP information, but must ignore if that message is later received from Osmo-CC.

If SDP negotiation is completed, the Progress Indicator 8 should be added by the called endpoint to cause the caller to through-connect media. The Progress Indicator 8 may be added in a later message to cause the caller to through-connect media later after SDP negotiation. If the called party answers, there is no Progress Indicator required, the media is through-connected when the SDP negotiation is completed. The calling party may through-connect media, if Progress Indicator 1 or 8 is included in a response message, but at least when the called party answers. When a CC-DISC-IND message is received: If there is no Progress Indicator 1 or 8 included in the message, the lower layer protocol shall assumes that there is no in-band information available and release the call as soon as possible. If it is included, the lower layer protocol may listen to the tones/announcements until it releases the call.

All endpoints shall support at least a-law and µ-law codec and announce both in order of preference (optionally with other codecs). This ensures that every endpoint can communicate with every other endpoint. Because SIP protocol shares the same rule about supported codecs, no trans-coding is required. For special setups the a-law and µ-law codec may not be supported, if other supported codecs are shared with the remote endpoint.

The bearer capability IE is optional, but it can represent properties of the codec that is specified with SDP. (e.g. if a regular audio codec is used, bearer capability can be ‘audio’ or ‘speech’; if ‘clearmode’ is specified with SDP, bearer capability can be ‘data’). If a receiving endpoint gets no bearer capability IE, it must use information from SDP offer. If a receiving endpoint gets bearer capability IE (and if it is capable of using that, like GSM or ISDN), it must override that information, whatever defined in SDP.

6.3 Overlap dialing

The called endpoint does not need to support overlap receiving. The calling endpoint must have the capability of sending a complete number. If the calling endpoint connects to devices that do not support overlap dialing (e.g. PSTN), it should provide a feature to collect the digits and send the CC-SETUP-IND message when the dialed number is complete (en-block dialing). It then depends on the operator, if en-block dialing or overlap dialing is used.

Even if the calling endpoint does not support overlap sending, it must support CC-SETUP-ACK-REQ which changes the state to overlap sending. It must change the message into a supported message (e.g. CC-SETUP-PROC-REQ) or wait for another message that changes the state. It must store SDP and progress information until forwarding/processing it with a

supported message. The reason is that the remote endpoint may not yet know if the number is complete and therefore send CC-SETUP-ACK-REQ. Later it will send other message, if the number was complete.

If the called endpoint does not support overlap receiving, it should send IE_COMPLETE as soon as it knows the number is complete. This is optional.

7. OSMO-CC Process

7.1 Socket Process

The OSMO-CC protocol was designed to have a unified interface to lower layer signaling protocols. Towards the upper layer it is absolutely symmetric, so that two endpoints can be connected just by forwarding the upper layer messages. This means that no complex gateway application (e.g. PBX) is required. The only thing that must be done is converting the direction information in the message type. (See below.) Additionally some essential information like the caller ID or dialed number might need to be changed. This can be implemented by lower layer protocol or upper layer protocol or OSMO-CC process.

To interconnect interfaces on different hosts, TCP socket is used. Each endpoint shall bind a TCP socket for incoming connections to a given port or 4200, if undefined. If this port is already in use and undefined, the next higher unused port is chosen.

When a connection is established, both ends send a version string of eight bytes: “OSMOCCv1” first. (no \0 termination, no line feed, case sensitive) The Version number is defined by the eighth byte. In this case the protocol version is ‘1’. Both ends shall send and receive version information. The client side detects, if the server’s version is different, so this is reported back to the endpoint. The server side shall detect if the client’s version is different before trying to parse invalid data, so it disconnects the socket. If a client does not receive the string, it shall send a reject message (CC-REJ-REQ) message towards the lower layer with the socket cause set to “FAILED”. If a client receives the string with a different version, the socket cause shall be set to “VERSION_MISMATCH”.

Messages between lower layer protocol and OSMO-CC process shall be forwarded via socket without or with minimal modifications. Messages sent via socket interface are buffered and more or less delayed. This may result in race conditions, where both ends have different call states, while a message is on the way through the socket. The following rules apply:

1. All indication messages from lower layer protocol (ending with -IND) shall be changed to request messages (ending with -REQ), before they are transmitted to the socket. All response messages from lower layer protocol (ending with -RSP) shall be changed to confirm messages (ending with -CNF) respectively.

2. If a reject request message (CC-REJ-REQ) is received from the socket (in response to CC-SETUP-REQ), it shall be changed to a release request message (CC-REL-REQ), if the call state is not INIT-IN. This is the case where the lower layer protocol sends a disconnect indication message (CC-DISC-IND) while not yet received the reject request message (CC-REJ-REQ). (race condition)

Note: The reject request message CC-REJ-REQ may be converted by the lower layer protocol to an equivalent of a disconnect request message, so that the lower layer interface may generate tones and/or announcements that describe the reject cause. It must then handle further call states itself.

3. If both ends send disconnect request messages (CC-DISC-REQ) simultaneously, the situation is called “disconnect collision”. Both ends receive a disconnect message while being

in DISCONNECTING-IN state. Then both ends silently close their socket connections and send a release request message (CC-REL-REQ) towards lower layer. The process shall wait until a release confirm message (CC-REL-CNF) was received.

4. If a release request message (CC-REL-REQ) is received from socket while being in DISCONNECTING-IN state, the socket connection shall be closed, the message shall be forwarded and the process shall wait until a release confirm message (CC-REL-CNF) was received. The release confirm message (CC-REL-CNF) is never transferred via socket interface.

If a release request message (CC-REL-REQ) is received from socket while being in DISCONNECTING-OUT state, the socket connection shall be closed, but the message shall not be forwarded and the process shall wait until a release indication message (CC-REL-IND) was received.

If a release request message (CC-REL-REQ) is received from socket while not being in DISCONNECTING-IN or DISCONNECTING-OUT state, the socket shall be closed, the message shall be forwarded as disconnect request message (CC-DISC-REQ), the state shall be changed to DISCONNECTING-OUT state and and the process shall wait until a release indication message (CC-REL-IND) was received.

Note: While being in DISCONNECTING_OUT state, lower layer interface may generate tones and/or announcements that describe the release cause. It must then handle further call states itself.

5. Whenever a socket fails, before a release / reject message was received from this socket, it is treated as CC-REJ-REQ in INIT-IN state or CC-REL-REQ in other states. The cause for the socket failure is applied as IE_CAUSE to the message. Then the rule 4 shall apply.

6. When a message is received from or sent to the socket and there is no reference to a call, a CC-REL-REQ/CC-REJ-REQ shall be sent to the other layer with the cause that the callref is invalid (cause 81). If the message is CC-REL-REQ or CC-REJ-REQ, it shall be silently discarded.

7.2 Cause conversion Process (optional, but highly recommended)

The task is to convert a cause between SIP and Q.850, so that the lower layer protocol must not provide this conversion itself.

When a lower layer protocol receives a message without cause value that is mandatory for this protocol, it assumes ‘normal call clearing’ (caller hangs up). If cause conversion process is provided, ‘normal call clearing’ is set when no cause is given in a message.

If Q.850 cause is set, but SIP cause not, or SIP cause is set, but Q.850 cause not, conversion according to RFC4497 section 8.4.4 is applied by the OSMO-CC process.

If socket cause is set, but SIP and Q.850 not, the most applicable cause value shall be added by the OSMO-CC process. Note: Most socket failures are ‘temporary failure’. Mismatching version is a ‘permanent failure’.

7.3 Attachment Process (optional)

The attachment is useful to implement a router using OSMO-CC. Each interface can attach to the router and tell it’s name and socket address. OSMO-CC process of the router keeps track

of all registered interfaces. If the router forwards a call back to OSMO-CC, OSMO-CC will select the correct remote address from the called interface name.

Every OSMO-CC process that connects to a router must send a CC-ATTACH-REQ over the socket interface, including the source address and interface name. The receiving OSMO-CC process will then add the given address and interface name to a list. Then it responds with CC-ATTACH-RSP message over the socket interface. The sending OSMO-CC process will then enter the attached state. If the socket fails, the sending OSMO-CC process will start the attachment timer and restarts the attachment after timeout.

8. Timers

8.1 Keep-alive Timer

The TX keep-alive timer shall be started when the socket connection has been established and for every message transmitted (including CC-DUMMY-REQ). If the timer times out, a CC-DUMMY-REQ message shall be sent. The timer duration shall 10 seconds by default.

The RX keep-alive timer shall be started when the socket connection has been established and for every message received (including CC-DUMMY-REQ). If the timer times out, the socket connection shall be closed and a CC-REL-REQ shall be sent to lower layer protocol including the socket timeout cause. The timer duration shall be 20 seconds by default.

8.2 Attachment Timer

The attachment timer shall be started when outgoing attachment fails. If the timer times out, the attachment shall be restarted. The timer duration shall be 2 seconds by default.

Annex

A.1 Check your Implementation **TBD**

- setup->more, proceeding, alerting
- connect
- disconnect (with and without tones)
- disconnect collisions
- release collision
- release any time
- sdp negotiation

B.1 Notes on inter-operation with ISDN **TBD**

ISDN phone offer one or more codecs in the bearer capability of the setup message. As Osmo-CC endpoints requires to offer a-law and µ-law codecs, They must be appended to the setup message, if not already included. Transcoding must be performed by endpoint. (a-law to µ-law and vice versa).

B.2 Notes on inter-operation with SIP **TBD**

There is no re-invite message with Osmo-CC to negotiate the codec again during call. For this case the SIP endpoint has to store the SDP list that has been received from socket and use that to complete the re-invite. This will ensure that the remote endpoint (connected via socket interface) must not change codec.

If a disconnect request message (DISC_REQ) was received by SIP endpoint, it may (as an option) to continue the active call, so that the user can receive tones/announcements. If the SIP user later disconnects the call, a release indication message (REL_IND) shall be sent instead of a disconnect indication message.

TBD: disconnect collision (ind nach req). Was macht das lower layer protocoll?

B.3 Notes on inter-operation with GSM **TBD**

A special message is used with (current) MNCC interface between OSMO-MSC and LCR to modify the channel after codec negotiation is complete. OSMO-MSC must perform the channel modification itself by using the first reply to the setup message that includes the SDP response.