„Digitalradio-Gateway-Interface“
(DF-Stecker)

Technical Description

Publication of AK BOS-Leitstellen
(Proposed Baseline DR-GW-Interface V0.3 Rel1.2)

Berlin, 28.02.2014

Document History

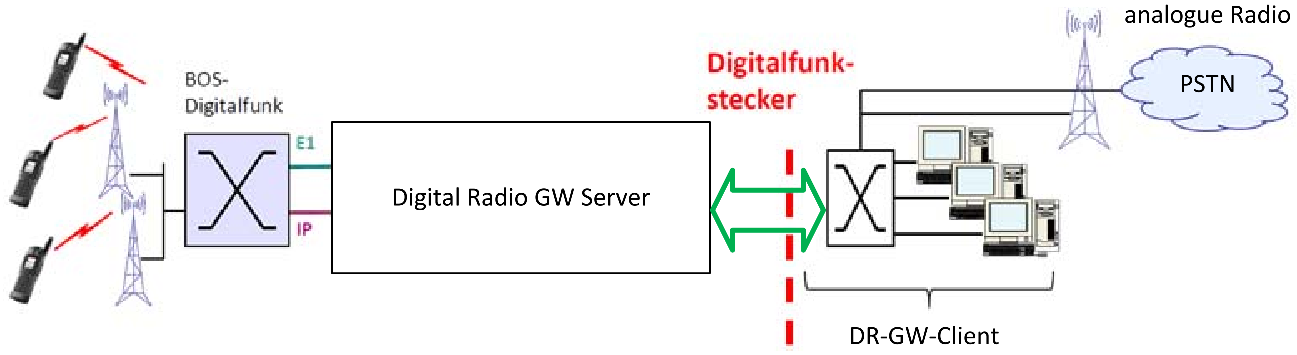
Version	Reference	Date
Version 0.3 Rel 0.9	Output and review version (Expertenforum)	31.01.2014
Version 0.3 Rel 0.91	Input version „Expertenforum 11.02.14”	11.02.2014
Version 0.3 Rel 0.93	Internal review	18.02.2014
Version 0.3 Rel 1.0	Output version „Expertenforum”	20.02.2014
Version 0.3 Rel 1.1	Output version „Expertenforum”	21.02.2014
Version 0.3 Rel 1.2	editorial input of responses	28.02.2014

Table of Contents

Table of Figures

1 INTRODUCTION

This document describes the Digitalradio-Gateway-Interface (DF-Stecker) which is used to interface third-party applications (Digitalradio-Gateway-Clients) to the Digitalradio-Gateway (DR-GW).



This interface provides access to the TETRA System for third-party applications without being familiar with TETRA Connectivity Server API or Microsoft Component Object Model (COM) or Distributed Component Object Model (DCOM).
This interface is using SIP and RTP for audio signaling and audio real time transfer, and SOAP for data manipulation. One of the goals of this interface is to be network administrator friendly, so that configuration of network elements in different network environments would be easy and almost entirely automatic using the latest network technologies.

This document serves as a reference manual for third parties which develop applications to access the Digitalradio-Gateway. The present version 0.3 of the interface description shall be used as the common basis for the development of DR-Gateways as well as DR-GW-Clients.

Once the first interoperable realizations from different producers are available and positively tested this document will be transferred into a finalized version 1.0.

This document is not a description of a DR-GW server product and it may not be considered as a document of vendor specific products and functionalities. For further information about Concentrator products, please refer to appropriate and specific product documentation.

How to use this document

This document comprises of following chapters:

- Chapter 1: *Introduction:* introduce the purpose of this document
- Chapter 2: *Glossary:* list of used abbreviations
- Chapter 3: *Applicable documents*
- Chapter 4: *Overview*
- Chapter 5: *Use Case Description*
- Chapter 6: *Security Aspects*
- Chapter 7: *Profile Standard for the use of SIP*
- Chapter 8: *Profile Standard for the use of SOAP*
- Chapter 9: *Request, Response and Event system*
- Chapter 10: *Interface Definition:* complete DR-GW-API description

For qualification and differenciation of the need of realization of functionalities this document uses the terms MUST, SHALL and MAY.

2 GLOSSARY

Term	Definition
ACELP	Algebraic Code Excited Linear Prediction
API	Application Programming Interface
BDBOS	Bundesanstalt für den Digitalfunk der Behörden und Organisationen mit Sicherheitsaufgaben
BOS	Behörden und Organisationen mit Sicherheitsaufgaben
BSI	Bundesamt für Sicherheit in der Informationstechnik
COM	Component Object Model
Cseq	Command Sequence
DCOM	Distributed Component Object Model
DR-GW-Client (DF-Stecker / Clientapplikation)	Client application of a Digital Radio-Gateway-Server
DR-GW (DF-Stecker Server)	Digital Radio-Gateway (from an API perspective)
E2E encrypted	End-to-end-encrypted (specific BSI)
FSTE	First Speech Transport Encoding Format
G.711	G.711 is an ITU-T standard for audio companding
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IP	Internet Protocol
ISSI	Individual Short Subscriber ID
ITSI	Internet Engineering Task Force
LAN	Local Area Network

Term	Definition
LIP	Location Information Protocol
MSC	Message Sequence Chart
NAT	Network Address Translation
OSTE	Optimized Speech Transport Encoding Format
PCM	Pulse Code Modulation.
PCMA	Pulse Code Modulation A-Law format
PCMU	Pulse Code Modulation µ-Law format
PSTN	Public Switched Telephone Network
RTCP	Real-Time Control Protocol
PTT	Push To Talk
RTP	Real Time Transport Protocol
Rx	Receive/Receiving
SDP	Session Description Protocol
SDS	Short Data Services
SIP	Session Initiation Protocol
SIPS	Session Initiation Protocol Security
SOAP	Simple Object Access Protocol
SRTP	Secure Real-Time Transport Protocol
SSI	Short Subscriber ID
TCP	Transmission Control Protocol
TCS	TETRA Connectivity Server
TCS-Client	Client of the TETRA system from a TCS perspective
TETRA	Terrestrial Trunked Radio
TLS	Transport Layer Security
Tx	Transmit/Transmission/Transmitting
UAC	User Account Control

Term	Definition
UDP	User Datagram Protocol
VoIP	Voice Over IP
WAN	Wide Area Network
WSDL	Web Service Definition Language
WWW	World Wide Web
XML	Extensible Markup Language
XSD	XML Schema Definition

3 APPLICABLE DOCUMENTS AND STANDARDS

3.1 REF. TO RFC OR ETSI

RFC	Content
RFC 3261	SIP: Session Initiation Protocol
RFC 3550	RTP: Real-Time Transport Protocol
RFC 3326	SIP: The Reason Header Field for the Session Initiation Protocol
RFC 2976	SIP: SIP INFO Method
RFC 3262	SIP: Reliability of Provisional Responses in the Session Initiation Protocol
RFC 3311	SIP: UPDATE Method
RFC 3515	SIP: REFER Method
RFC 4028	SIP: Session Timer
RFC 3265	SIP: Specific Event Notification
RFC 3911	SIP: Join Header
RFC 3891	SIP: Replaces Header
RFC 4566	SDP: Session Description Protocol
ETSI EN 300 392	ETSI EN 300 392-2 V2.3.2 (2001) Terrestrial Trunked Radio

3.2 REF. DOCUMENTS

Document	Content	Date
DR-GW-Schemas_v1.zip	Container of xsd schemas	31.01.2014
DR_GW_payload_ref_vxx.pdf	Payload Format for voip transport	
EADS TETRA System Release 6.0–6.5	TCS API Description	1/2013

4 OVERVIEW

4.1 CONTRACTS OVERVIEW

Description

This document describes two different layers of services interacting with tetra systems.

- application layer control protocol for all services concerning speech services. The referred protocol is described in RFC 3261 SIP. For different call types and scenarios refer to the use cases in this document. The login method is via SIP typical invite / register.
- web service description for the non speech services (e.g. SDS) . The login method is a typical Web Service Authorization.

Basics about BOS TETRA Systems and TCS API are helpful to understand the functionality of the DR-GW interface. (Ref: TCS-Description - EADS TETRA System Release 6.0–6.5)

4.2 API WORKFLOW

Depending on the used service a DR-GW-Client establishes a connection either with a login, or with a SIP invite/register.

Login and invite / register logic refer respectively to the SIP and the SOAP part of the interface, these two parts work independently and can also be used separately.

In case of SIP services we use synchronous events in a session. In case of web services, requests and events are fired in asynchronous manner.

In both cases, the resource management is fully under control of the DR-GW and its performance depends on the manufacturer of the DR-GW.

4.2.1 R-GW BUSINESS LOGIC

The DR-GW manages states for resource management (ISSIs, B channels and M channels). The DR-GW is responsible for managing sessions, DR-GW- Clients and resource distribution and the way it is done is DR-GW product specific.

4.2.2 DR-GW-CLIENT SESSION ESTABLISHMENT AND MANAGEMENT

The DR-GW-Client has to be provided with IP Addresses in order to establish a connection and a session with the DR-GW. At least, the DR-GW-Client shall be provisioned with a single IP address.

Sessions are not disconnected by the DR-GW if the “real tetra call” on the air is disconnected. Normally a session lasts until the DR-GW-Client does finish the call.

Failures of the session could be the result of multiple types of failures. Failures could be the result of network equipment failure or network equipment misconfiguration. They can also be related to firewall failures, cabling failures or disconnection. There is also the possibility of uncontrolled delays or congestions on the network level within the LAN or at the WAN (problem with the network provider).

In case when a DR-GW-Client is losing its session with the DR-GW, for example by having a failure at the HTTP or TCP/UDP layer or by not receiving Keep Alive Event regularly, the DF Client has to resend a login request to that IP Address periodically in order to restore the session, until it receives a valid answer. It is then the responsibility of the DR-GW-Client to restore the session by requesting resources and group monitoring.

4.2.3 B and M Channel

The concept of B and M Channels in the Tetra system and their handling is explained in the document "Hinweise und Handreichungen zur Schnittstelle Digitalfunkstecker (DF-Stecker) und ihrer Verwendung" published by PMeV.

B and M Channels are useful to support resource management by the DR-GW. Technically the DR-GW Client needs no further information about the channels and resource management in the tetra network and this document will not make use of them. Yet knowing the concept may be a helpful background information for the programmer of a control room application.

4.2.4 API Versioning

There are 3 major interfaces contained in the "DR-GW-Interface":

- RTP for Audio Transport
- SIP for Speech Communication Control
- SOAP for any other Services like data, group management, …

The explicit version announcement shall prepare the interface for future version management and even version negotiation in case of an incompatible version change is needed for any reason. As long as no different versions are deployed no other handling than the according version announcements are needed.

4.2.4.1 RTP for Audio Transport Versioning

There is no other versioning besides 2 Bit field in the RTP Header defined in RFC 3550. Currently the version is set to 2.

4.2.4.2 SIP Interface Versioning

In order to give incompatible changes of the interface characteristics a chance in the future, DR-GW-Interface introduces a specific SIP Header field for versioning purposes. The present document SHALL be referred to as "radio.01" in the DR-GW-Version SIP Header Field as described in 7.6.1.

4.2.4.3 SOAP INTERFACE VERSIONING

In order to give incompatible changes of the interface characteristics a chance in the future, DR-GW- Soap Interface introduces a specific a version argument for sessionlogin in SOAP requests (see chapter 10.2.1 and 10.3.2).

4.3 NETWORKING ASPECTS

4.3.1 FIREWALL

A Firewall should restrict reception of traffic coming from known and expected sources (Concentrator/DR-GW and Control rooms/DR-GW-Client) and steps should be taken to prevent IP address spoofing while designing a DR-GW-Server/DR-GW-Client network.

Due to the application of standard SIP with standard SDP each voice-over-IP can be introduced with SIP aware stateful firewall including Session Border Controllers to ensure the highest level of security separation between different network domains. No static setting for RTP Ports is required as these devices keep track of sessions and explicitly and dynamically enable peer connectivity on a peer source IP Address, source port, destination IP Address, destination port.

4.3.2 NAT

For support of Network Address Translation in the network between the DR-GW-Client and the DR-GW a “SIP aware” application layer gateway has to be introduced such as the NATting device.

4.3.3 TRAFFIC TAGGING

Traffic tagging is not covered by this DR-GW-Interface (API definition) but it is considered an important point while deploying a Concentrator/Control room combination. To be able to configure those parameters on both sides this feature should be present and each application should support a predefined tagging.

4.3.4 MULTICAST IGMP

In case when a DR-GW and/or Control room is implementing audio distribution using multicast group, IGMP V2.0 or 3.0 shall be supported on the network.

4.4 Mandatory and Optional features

4.4.1 Audio Compression

This section is listing which codecs a DR-GW Server side implementation may contain to be compliant with the standard.

Codec	DR-GW	DR-GW-Client
G.711	Mandatory	Optional
ACELP	Mandatory	Optional
FSTE / OSTE	Optional	Optional
Any other codecs	Optional	Optional

4.4.2 Audio encryption

Encryption state	DR-GW	DR-GW-Client
Unencrypted	Mandatory	Mandatory
End-to-end-encrypted	Optional	Optional

4.4.3 Audio IP distribution

Audio IP Distribution	DR-GW	DR-GW-Client
Unicast	Mandatory	Mandatory
Multicast	Optional	Optional

4.4.4 Control Method SOAP-Request

Method	Mandatory Client	Mandatory DR-GW	Optional DR-GW
DR-GW-Session			
Session_Login	X	X	
Session_Logout	X	X	
Session_Supervise	X	x	

Method	Mandatory Client	Mandatory DR-GW	Optional DR-GW
DR-GW-Session.Events			
Session_Response		X	
Session_LoginEvent		X	
Session_LogoutEvent		X	
Session_SuperviseEvent		X	
DR-GW-SDS			
SDS_Send		X	
SDS_SendReport		X	
DR-GW-SDS.Events			
SDS_Response		X	
SDS_SendEvent		X	
SDS_ReceiveEvent		X	
SDS_ReportEvent		X	
DR-GW-Status			
Status_Send		X	
DR-GW-Status.Events			
Status_Response		X	
Status_SendEvent		X	
Status_ReceiveEvent		X	
DR-GW-OrganisationBlock			
Org_Get			X
Org_GetList			X
DR-GW-OrganisationBlock.Events			X
Org_Response			X
Org_GetEvent			X
Org_GetListEvent			X
Org_Event			X
DR-GW-Group			
Group_Get		X	
Group_GetList		X	
Group_GetRadioMembers		X	
Group_GetAppMembers			X
Group_Track			X

Method	Mandatory Client	Mandatory DR-GW	Optional DR-GW
Group_AddRadioMember			X
Group_RemoveRadioMember			X
Group_GetCombinations			X
Group_AddCombination			X
Group_RemoveCombination			X
Group_SubscribeData		X	
DR-GW-Group.Events			
Group_Response		X	
Group_GetEvent		X	
Group_GetListEvent		X	
Group_GetRadioMembersEvent			X
Group_GetAppMembersEvent			X
Group_Event			X
Group_RadioMemberEvent			X
Group_AppMemberEvent			X
Group_GetCombinationsEvent			X
Group_CombinationEvent			X
DR-GW-Application			
App_Get			X
App_GetList			X
DR-GW-Application.Events			
App_Response			X
App_GetEvent			X
App_GetListEvent			X
DR-GW-System.Events			
Sys_TETRAStatesEvent			X
Sys_LogEvent			x
DR-GW-Radio			
Radio_Get			X
Radio_GetList			X
Radio_GetGroups			X
Radio_Track			X
DR-GW-Radio.Events			

Method	Mandatory Client	Mandatory DR-GW	Optional DR-GW
Radio_Response			X
Radio_GetEvent			X
Radio_GetListEvent			X
Radio_GetGroupsEvent			X
Radio_Event			X
Radio_TrackEvent			X

4.4.5 Load And Resource Sharing

This DR-GW switchover functionality is optional.

Based on the assumption that multiple servers could exist in the DR-GW handling requests from control rooms, the following scenario shall be supported by the API. The exact details of the implementation on the server are out of scope of this API document.

If a DR-GW-Client requests a resource on a server which does not have enough available resources to accommodate the DR-GW-Client, the server may request other physical server of the DR-GW for resource availability and return the IP address of the server which could provide the service requested in the resource request event.

It is up to the server implementation (out of scope of this document) to determine how many sessions are allowed on a DR-GW using the same credentials on the DR-GW-Client side.

4.4.6 DR-GW Failover functionality

In a regular/simple deployment, it is a responsibility of the DR-GW-Client to establish or re-establish a valid session with the DR-GW and request resources. Resources are automatically released upon a session failure. It is a responsibility of the DR-GW-Client to know (via initial provisioning/configuration) the available IP addresses which it can connect to.

The following DR-GW failover functionality is optional.

Based on the assumption that a DR-GW server could exist in a redundant deployment model (Active-Standby or Active-Active), the following scenario shall be supported by the API. The exact details of the implementation on the server are out of scope of this API document.

If a server fails while in service, the peer server (redundancy partner) shall be able to take over the role and resources (including IP Address) of the failed server. In such case the failover would be entirely transparent for the DR-GW-Client.

It is then the responsibility of the DR-GW-Client to establish a new session with the new server by re-authenticating and by re-requesting the resource using the same messages workflow.

4.4.7 DR-GW-CLIENT REDUNDANCY

The interface described in here does not explicitly foresee any redundant client implementation. In scenarios where the DR-GW-Client is realized as a centralized element such a redundancy makes sense to achieve a service with high reliability. As an implementation hint these redundant elements shall perform a local communication in a way that only one DF-GW-Client service is active at one moment. The DF-GW server strictly follows a first come first serve strategy considering the specified priorities for arbitration.

5 USE CASE DESCRIPTION

This document describes use cases for using TETRA with a gateway to TETRA net. The digital radio gateway was named DigitalRadio-Gateway (or short form DR-GW).

The context of the use cases is always the DR-GW, focused on the interface to the User. The user could be a more or less complex system or a dispatcher including some systems.

Lean diagram conditions of extensions were not drawn in a diagram, but described in corresponding texts above diagrams.

The user was connected to the DR-GW. The DR-GW was connected to the radio exchange system. The TETRA subscriber subscribes over a base station of the radio exchange system data and call services.

Utilization of the SIP Message Header Fields as well as SIP Message Bodies is explained in the specific Message Sequence Charts. Chapter 7 “Profile Standard for the use of SIP” normatively describes the use of the respective fields.

Use Cases in this chapter focus on the more complex part of the interface which is the handling of audio and hence SIP-related parts. Exchange of pure data makes use of SOAP and is more straightforward. Here specific Use Cases for the SOAP-related parts seem not to be necessary.

5.1 RESOURCE ALLOCATION

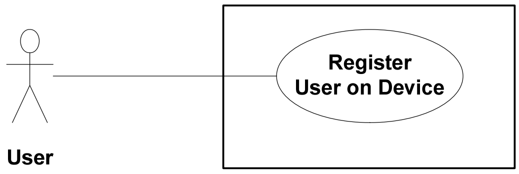


Figure 1 – Use Case Resource Allocation

If a user wants to get individual calls from the DR-GW or the TETRA Subscriber he has to register himself on the device inside the DR-GW. After registration the user is able to receive calls from the TETRA Subscriber.

Depending on the implementation of the DR-GW this registration may be used to implicitly allocate the according resources for that specific client either in an exclusive or a shared manner. Especially for the allocation of the so called B-Channel with its associated ISSI this way of resource reservation shall be treated as a strong recommendation for implementation in the DR-GW.

Before expiration of the registration Timer, the user has to refresh his registration. The user finishes his resource allocation by registering a timeout value of 0 which is the SIP “UNREGISTER” method.

5.1.1 Message Sequence Chart Description For Resource Allocation

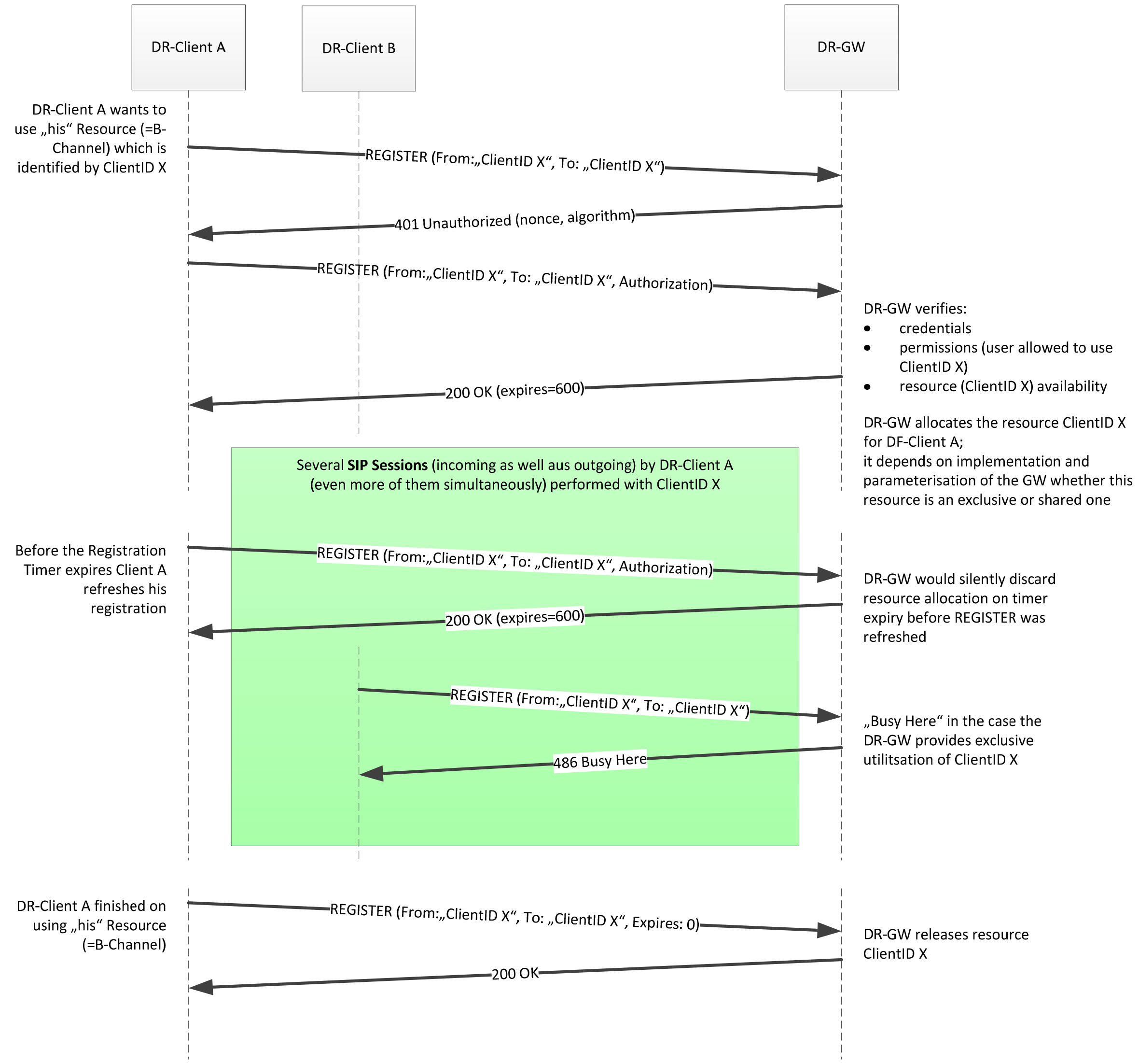


Figure 2 – MSC for Resource Allocation

5.2 INCOMING INDIVIDUAL CALLS

Incoming call in this context means a call originated somewhere within the "Digitalfunk" propagated via the "DR-GW-Server" to the "DR-GW-Client". As a precondition before being able to receive any incoming call the DR-GW-Client has to allocate the called to resource via the SIP REGISTER method as described in chapter 5.1 "Resource allocation"

5.2.1 ESTABLISH AND TERMINATE INCOMING FULL DUPLEX INDIVIDUAL CALL

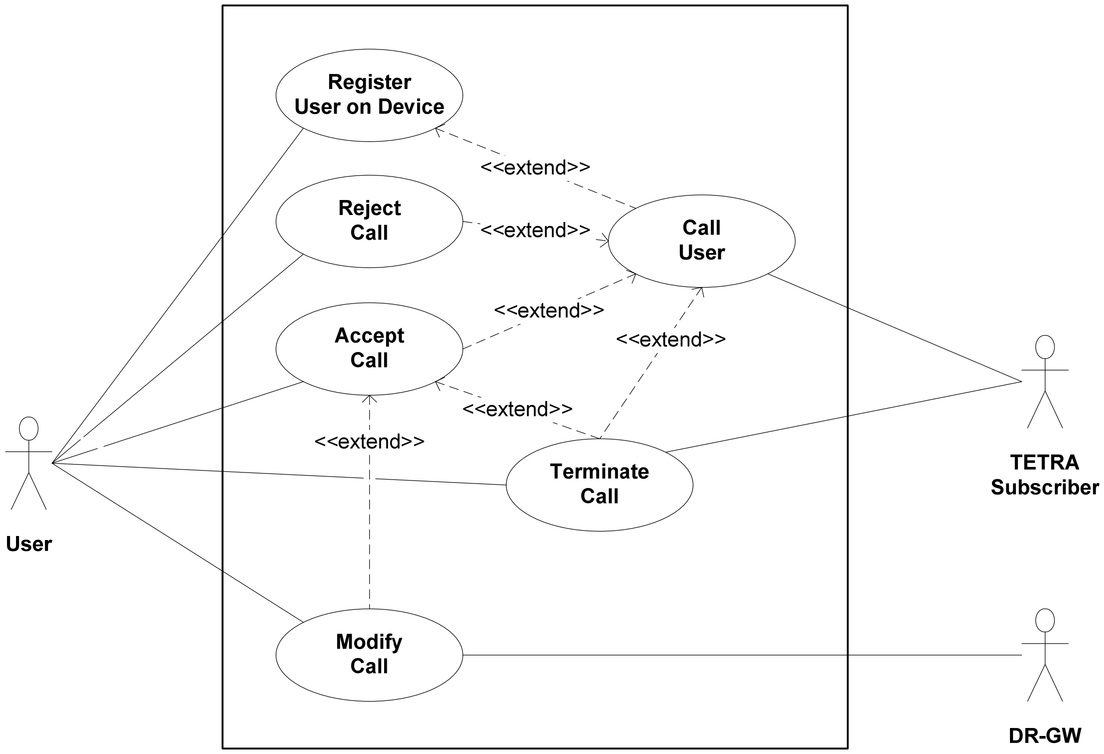


Figure 3 – Use Case establish and terminate Incoming Full Duplex Individual Call

To get signaling of calls from the TETRA subscriber a user has to register himself for a device. A device could be an identifier, as well as a calling line or number. If the device is an identifier, the DR-GW maps the identifier to a calling line or number.

If the user has registered himself for a device, the TETRA subscriber could call the registered user on the device. The DR-GW alerts.

The user could either reject or accept a call alerted by the DR-GW. If the user has not rejected or accepted the call, the TETRA-Subscriber could terminate the alerted call.

If the user has accepted the call, both parties, the caller DR-GW and the callee User, had a dialog and could talk to each other in the established call by using it. The user or the DR-GW could modify the accepted call.

The user and the TETRA subscriber represented by the DR-GW could terminate the established dialog.

In case of non hook calls acceptance of the call was done automatically by the radio exchange system.

5.2.2 ESTABLISH AND TERMINATE INCOMING HALF DUPLEX INDIVIDUAL CALL

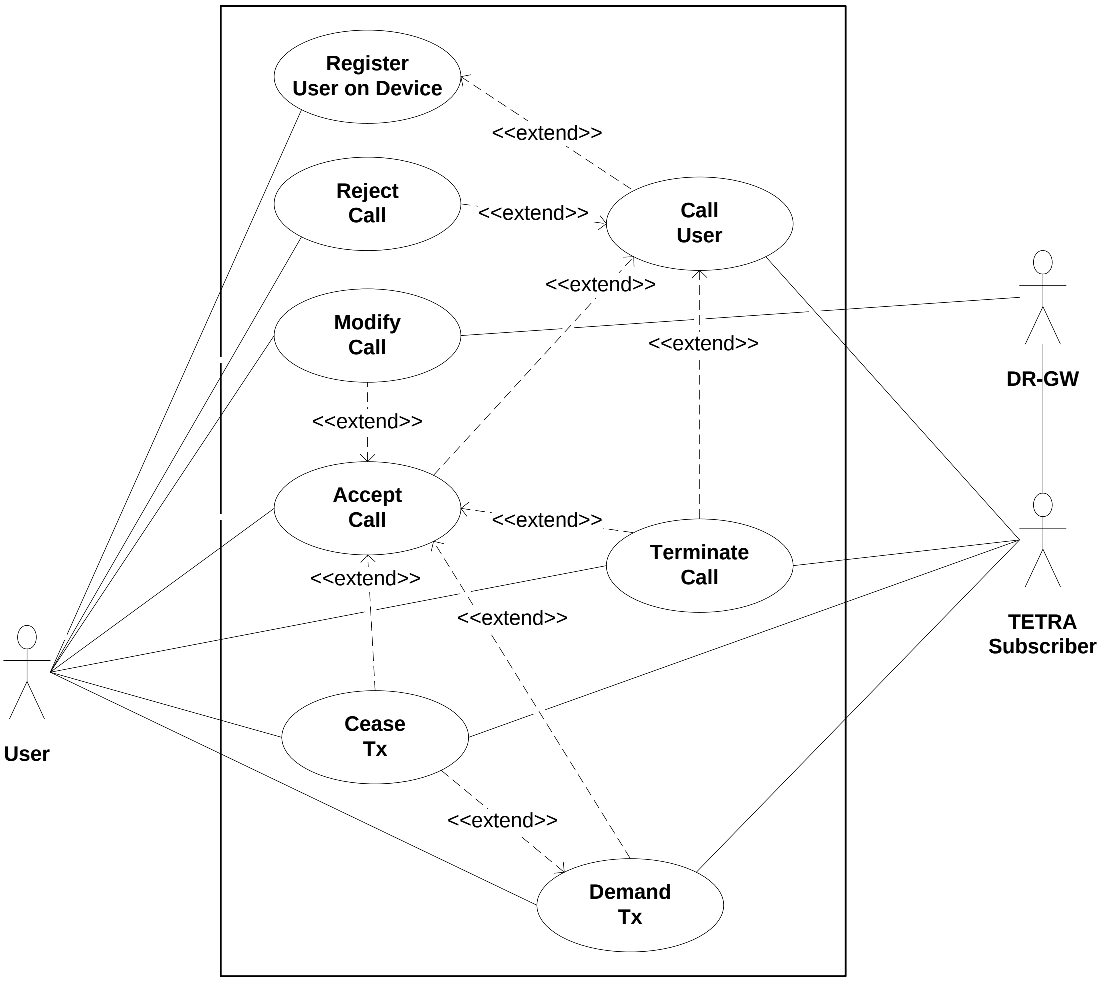


Figure 4 – Use Case establish and terminate Incoming Half Duplex Individual Call

To get signaling of calls from the TETRA subscriber a user has to register himself for a device. A device could be an identifier, as well as a calling line or number. If the device is an identifier, the DR-GW maps the identifier to a calling line or number.

If the user has registered himself for a device, the TETRA subscriber could call the registered user on the device. The DR-GW alerts.

The user could either reject or accept a call alerted by the DR-GW. If the user has not rejected or accepted the call, the TETRA-Subscriber could terminate the alerted call.

After call is accepted by the User, he and the TETRA subscriber could demand Tx – radio exchange system queue or grant Tx after demanding. After Tx has been granted to one call party the subscriber with the speech item could talk. The party with the speech item could cease Tx. Accepted calls could be modified by the user as well as by the DR-GW.

The user and the TETRA subscriber represented by the DR-GW could terminate the established dialog.

In case of non hook calls acceptance of the call was done automatically by the radio exchange system.

"Modify" in this chapter's sense is related to the SIP Session only. It is used to negotiate new IP Address/Port pairs and it will be performed via SIP UPDATE or Re-INVITE method.

5.2.3 Message Sequence Chart Description for Incoming Individual Calls

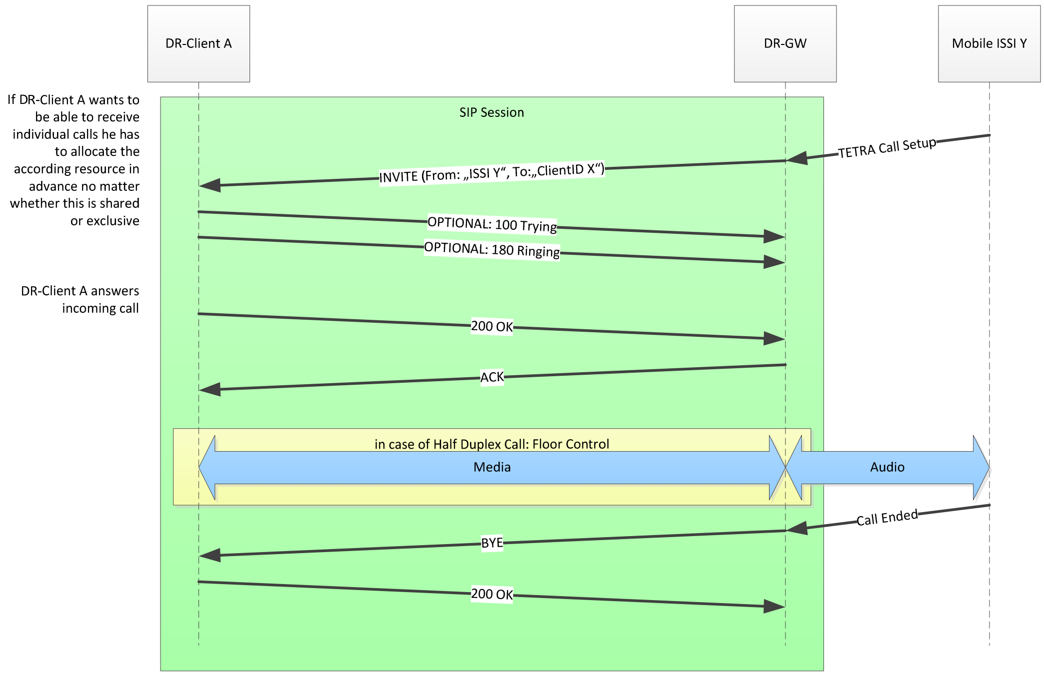


Figure 5 – MSC for Incoming Individual Call

Depending on which party releases the call first, the BYE may be either sent by the DR-GW (as shown in Figure 6) or by the DR-GW-Client.

5.3 USE CASES FOR OUTGOING INDIVIDUAL CALLS

5.3.1 ESTABLISH AND TERMINATE OUTGOING FULL DUPLEX INDIVIDUAL CALL

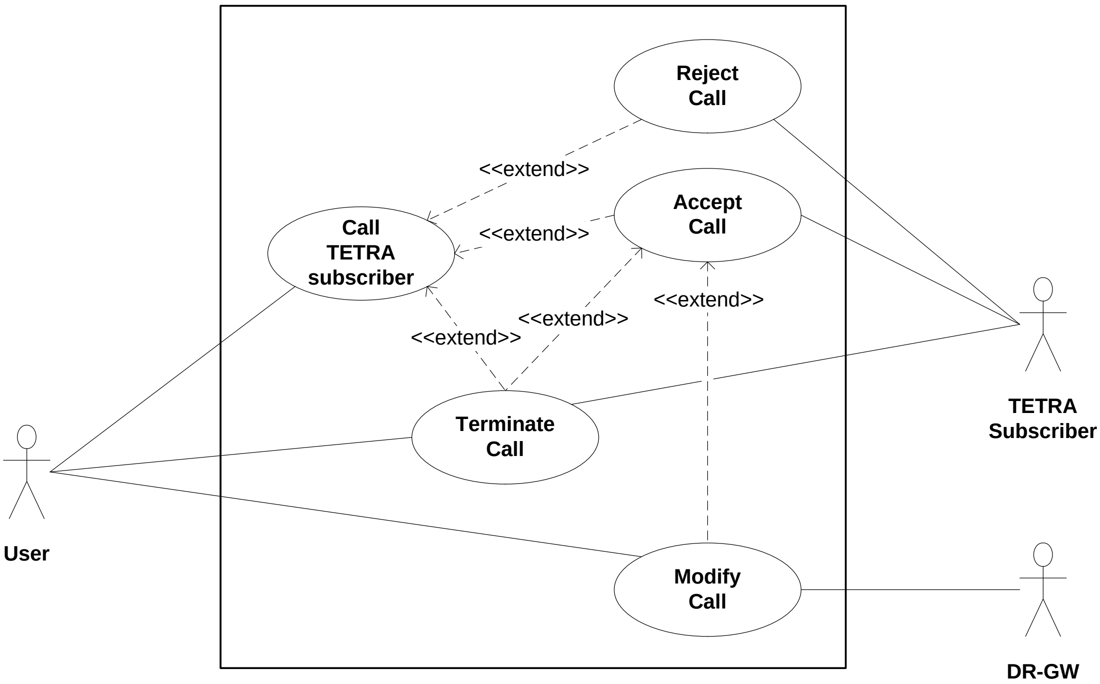


Figure 6 – Use Case establish and terminate Outgoing Full Duplex Individual Call

The user could call the TETRA subscriber.

The TETRA subscriber could either reject or accept the call alerted by the DR-GW. When the TETRA subscriber has not rejected or accepted the call, the user could terminate the alerted call.

If the TETRA subscriber has accepted the call both parties, the caller user and the callee TETRA subscriber represented by the DR-GW, had a dialog and could talk to each other in the established call by using it. The user or the DR-GW itself could modify the accepted call – the TETRA subscriber couldn't modify the call.

The user and the TETRA subscriber represented by the DR-GW could terminate the established dialog.

For an outgoing individual call registration of user for a device is not necessary.

In case of non hook calls acceptance of the call was done automatically by the radio exchange system.

5.3.2 Establish and Terminate Outgoing Half Duplex Individual Call

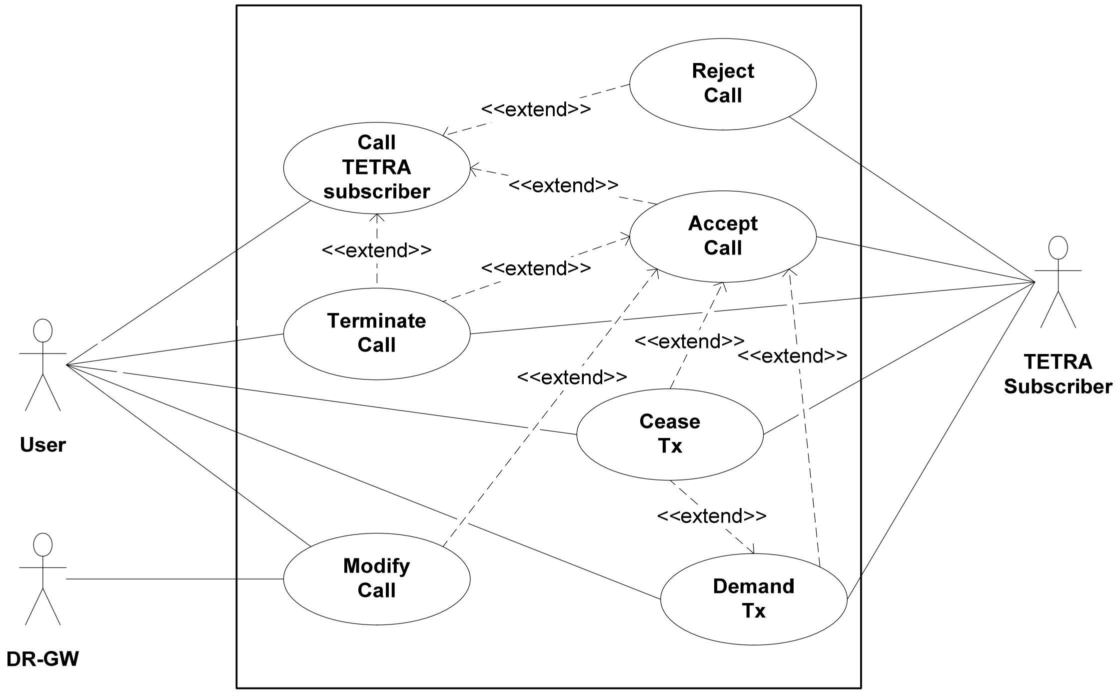


Figure 7 – Use Case establish and terminate Outgoing Half Duplex Individual Call

User could call the TETRA subscriber.

The TETRA subscriber could either reject or accept the call alerted by the DR-GW. When the TETRA subscriber has not rejected or accepted the call, the user could terminate the alerted call.

After call is accepted by the TETRA subscriber, he and the user could demand Tx – radio exchange system queue or grant Tx after demanding. After Tx has been granted to one call party the subscriber with the speech item could talk. The party with the speech item could cease Tx. Accepted calls could be modified by the user or the DR-GW.

The user and the TETRA subscriber represented by the DR-GW could terminate the established dialog.

For an outgoing individual call registration of user for a device is not necessary.

In case of non hook calls acceptance of the call was done automatically by the radio exchange system.

5.3.3 Message Sequence Chart for Outgoing Individual Call

The example message sequence chart below shows an individual call initiated by the user. It depicts the establishment of the SIP session including the digest authentication (“407

Proxy Authentication Required"). If the call is a half-duplex call, floor control is used for arbitration of the media. There is neither negotiation nor signaling about half or full-duplex call but this property is know-how that has to be exchanged preliminary via configuration.

The example shows the call being teared down by the TETRA subscriber.

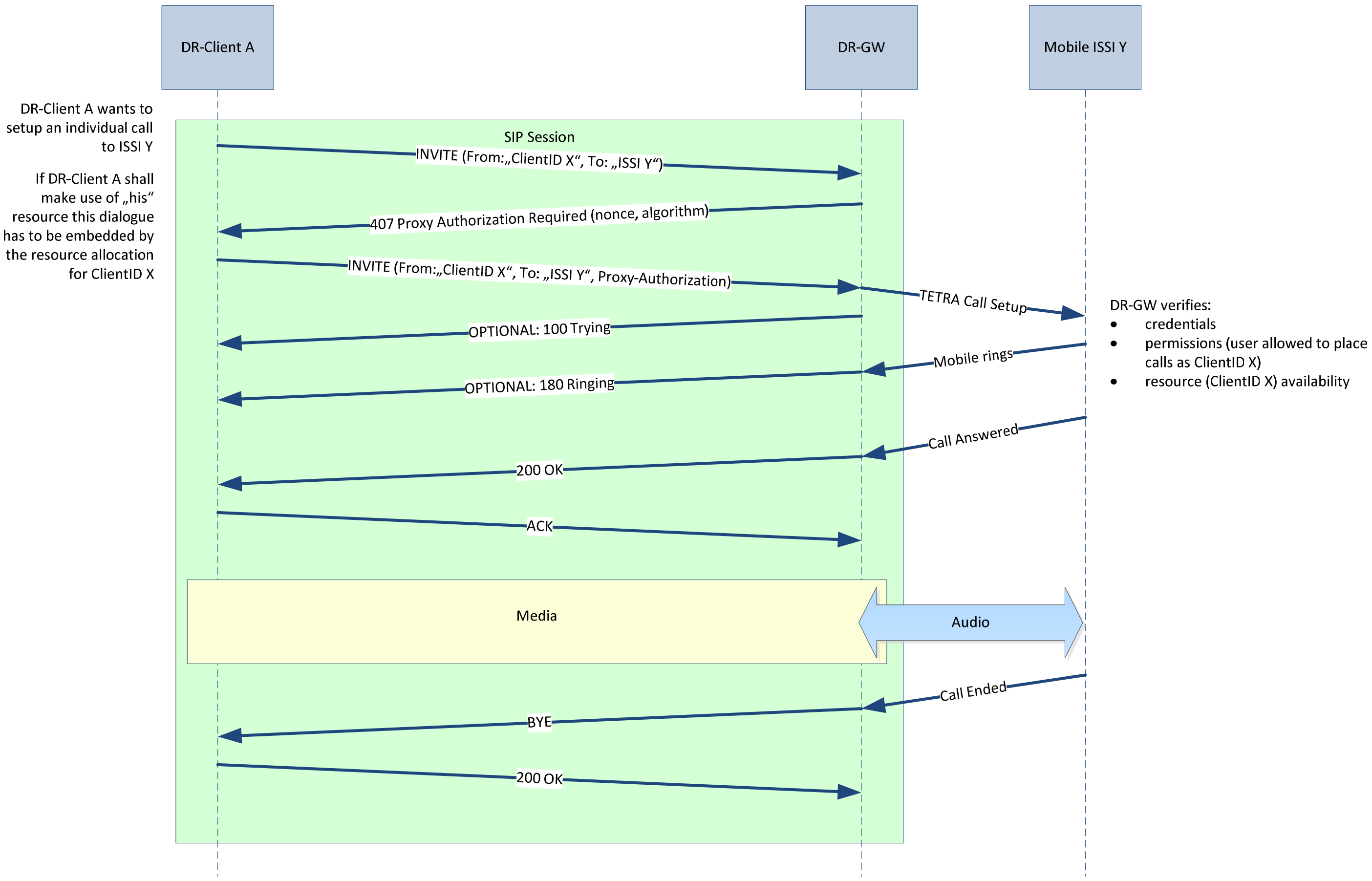


Figure 8 – MSC for Outgoing Individual Call

Before answering the TETRA subscriber may reject the offered call which is shown in the following message sequence chart.

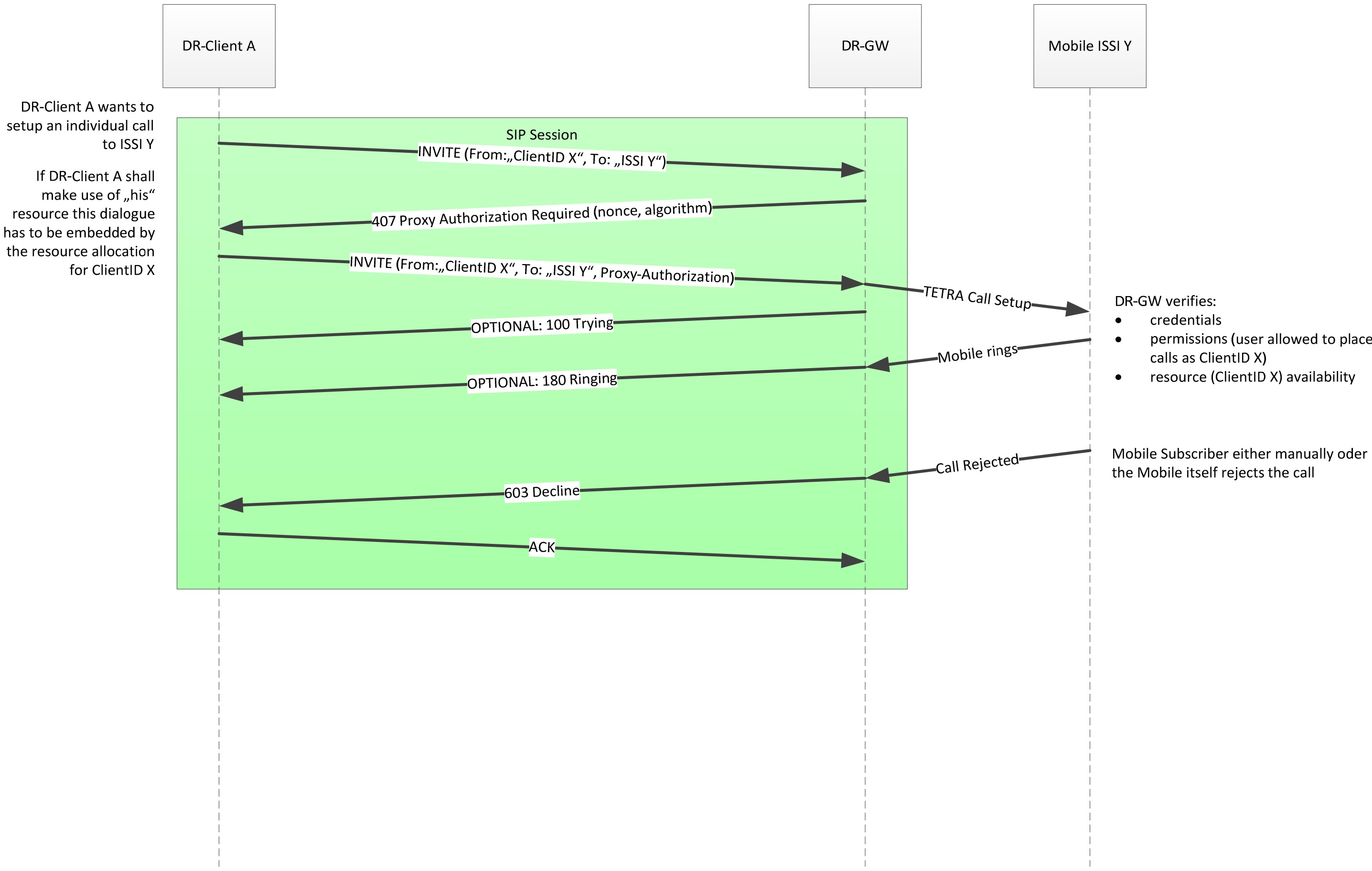


Figure 9 – MSC for GW Rejected Outgoing Individual Call

5.4 USE CASES FOR GROUP CALLS

5.4.1 EVENT MONITORING

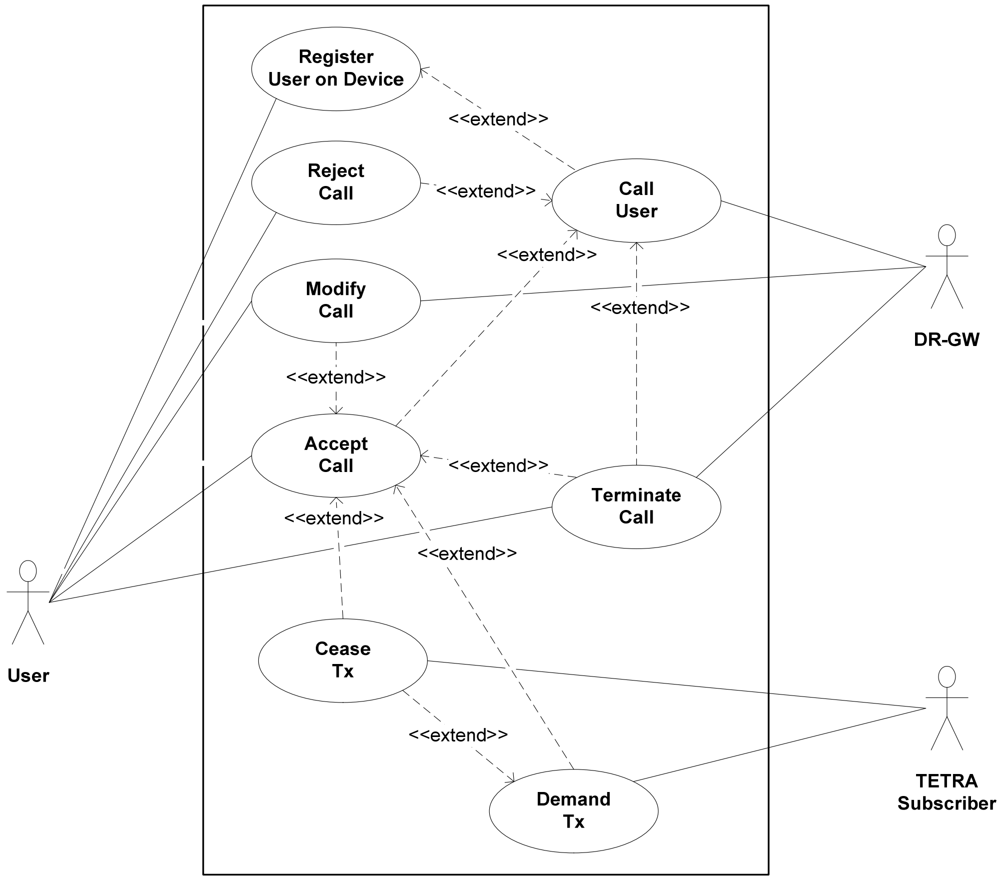


Figure 10 – Use Case Event Monitoring of TETRA Group

To get events of a TETRA group the user has to invite the group to a non-audio SIP Session. The session has to be established to the group in terms of a device. A device could be an identifier, as well as a calling line or number. If the device is an identifier, the DR-GW maps the identifier to a calling line or number. By calling the group the user defines the Selection Mode for group usage: in this case event monitoring so that no audio has been transported between DR-GW and the user.

If the user has established a session for a device, the DR-GW could notify in case of events.

The user could either reject or accept a call alerted by the DR-GW.

After the session is established, the DR-GW signals the User, if any TETRA-Subscriber on that particular group has demanded Tx–radio exchange system queue or grant Tx after demanding. After Tx has been granted to one call party the subscriber with the speech item could talk – only signaling would be transported in case of event monitoring. The party with the speech item could cease Tx. Accepted calls could be modified by the user or by the DR-GW.

Regularly the user terminates the established dialog while the DR-GW tears down the session only in cases where an error occurs.

5.4.2 AUDIO MONITORING

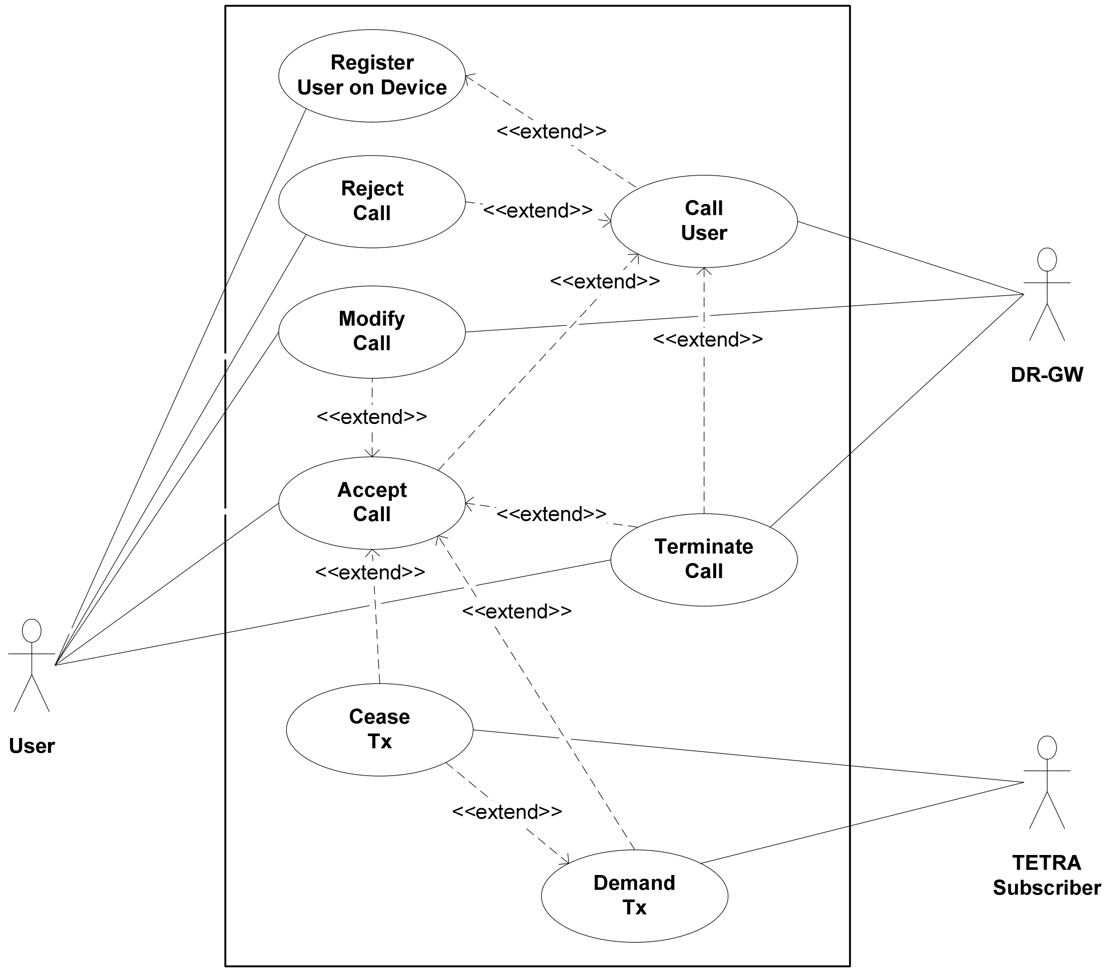


Figure 11 – Use Case Audio Monitoring of TETRA Group

To monitor audio of a TETRA group the user has to call the group. By calling the group the user defines the Selection Mode for group usage: in this case audio monitoring – so that signaling and audio has been transported between the DR-GW and the User. The DR-GW has to accept the call in order to be able to propagate the TETRA groups audio to the user.

After the call is accepted, the DR-GW signals the User, if the TETRA-Subscriber has demanded Tx – radio exchange system queue or grant Tx after demanding. After Tx has been granted to one call party the subscriber with the speech item could talk–signaling and audio would be transported in case of audio monitoring. The party with the speech item could cease Tx. Accepted calls could be modified by the user or by the DR-GW.

The user and the DR-GW could terminate the established dialog. Regularly the user terminates the dialog if he does not want to monitor the talk group anymore. In case of error occurrence the DR-GW tears down the SIP session which is equal to terminate the dialog.

5.4.3 USE TETRA GROUP

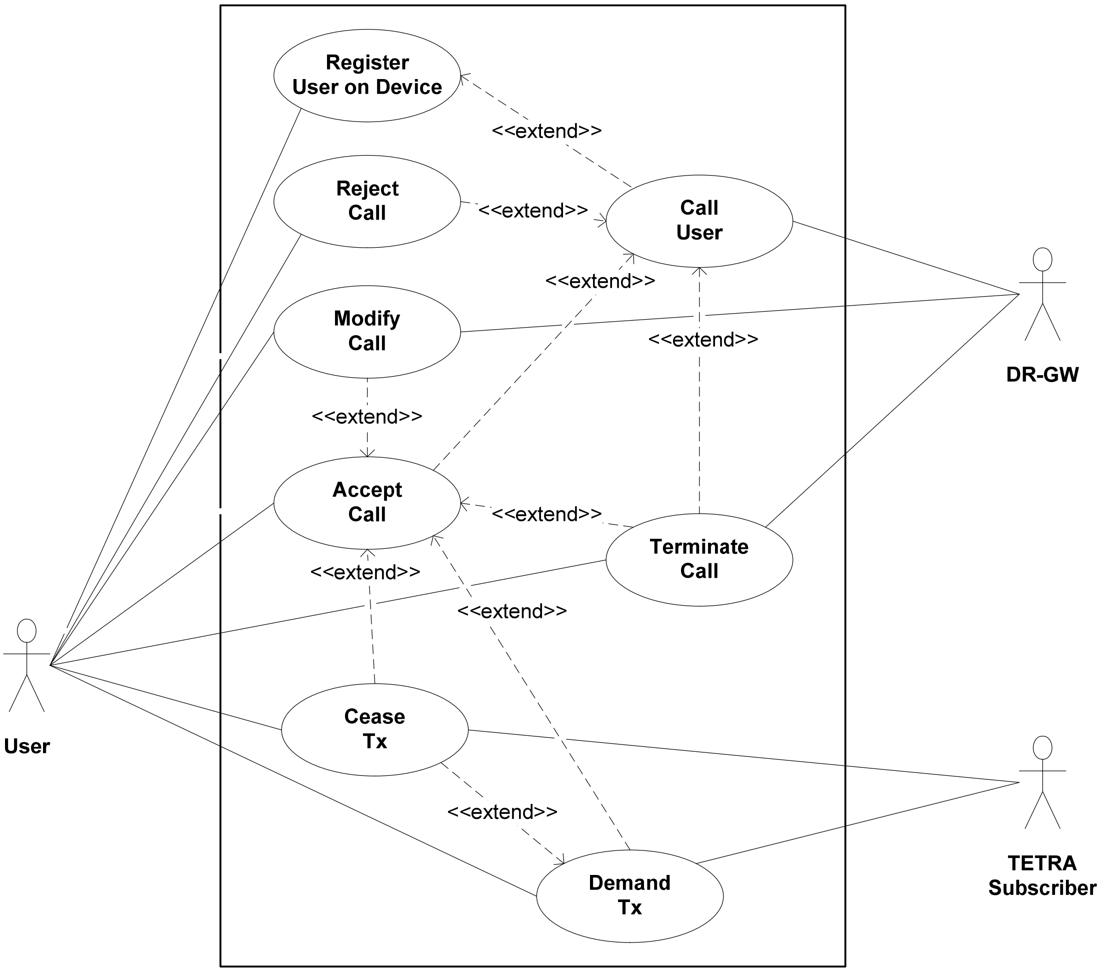


Figure 12 – Use Case Use TETRA Group

For using a TETRA group the use cases are similar to the use case of monitoring audio of a TETRA group. When using the TETRA not only the TETRA subscriber could demand and cease Tx, but the user as well.

To use a TETRA group the user has to invite this group to a SIP session. If the user wants to use his exclusive B-Channel for this purpose, he has to register himself in terms of a device. A device could be an identifier, as well as a calling line or number. If the device is an identifier, the DR-GW maps the identifier to a calling line or number. Without any registration it is a matter of DR-GW implementation which identification within the digital radio network will be used, when the user demands (and is granted for) Tx.

By calling the group the user defines the Selection Mode for group usage: in this case use of the talk group– so that signaling and audio can be transported between the DR-GW and the User. The DR-GW has to accept the call in order to initiate audio path from the user to the TETRA talk group and vice versa.

After the call is established, the DR-GW signals the User, if the TETRA-Subscriber has demanded Tx – radio exchange system queue or grant Tx after demanding. After Tx has been granted to one call party the subscriber with the speech item could talk–signaling and audio would be transported in this case. The party with the speech item could cease Tx. Accepted calls could be modified by the user or by the DR-GW.

The user and the DR-GW could terminate the established dialog. Regularly the user terminates the dialog if he does not want to use the talk group anymore. In case of error occurrence the DR-GW tears down the SIP session which is equal to terminate the dialog.

5.4.4 CHANGE SELECTION MODE FOR GROUP CALLS

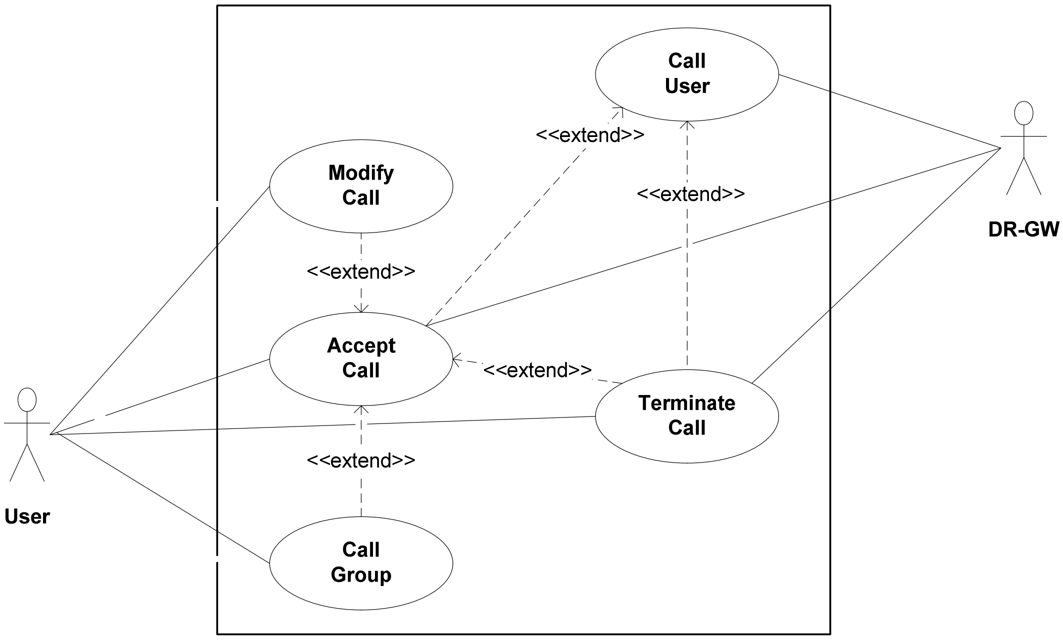


Figure 13 – Use Case SELECTION MODE FOR GROUP CALLS

To change the Selection Mode for group calls the user could modify the call. Modification of the call will be done by the user with a re-invitation.

5.4.5 DISCONNECT GROUP CALL

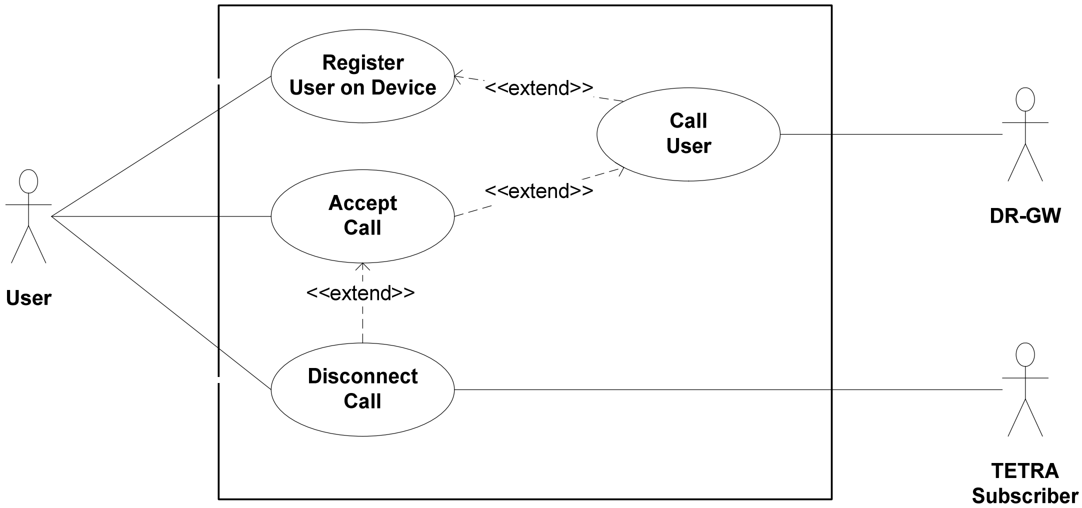


Figure 14 – Use Case Disconnect GROUP CALLS

Once the SIP session for the group call is established the user could disconnect the call. The TETRA subscriber or the DR-GW (including exchange) could disconnect the call as well.

5.4.6 MESSAGE SEQUENCE CHART DESCRIPTION FOR GROUP CALL

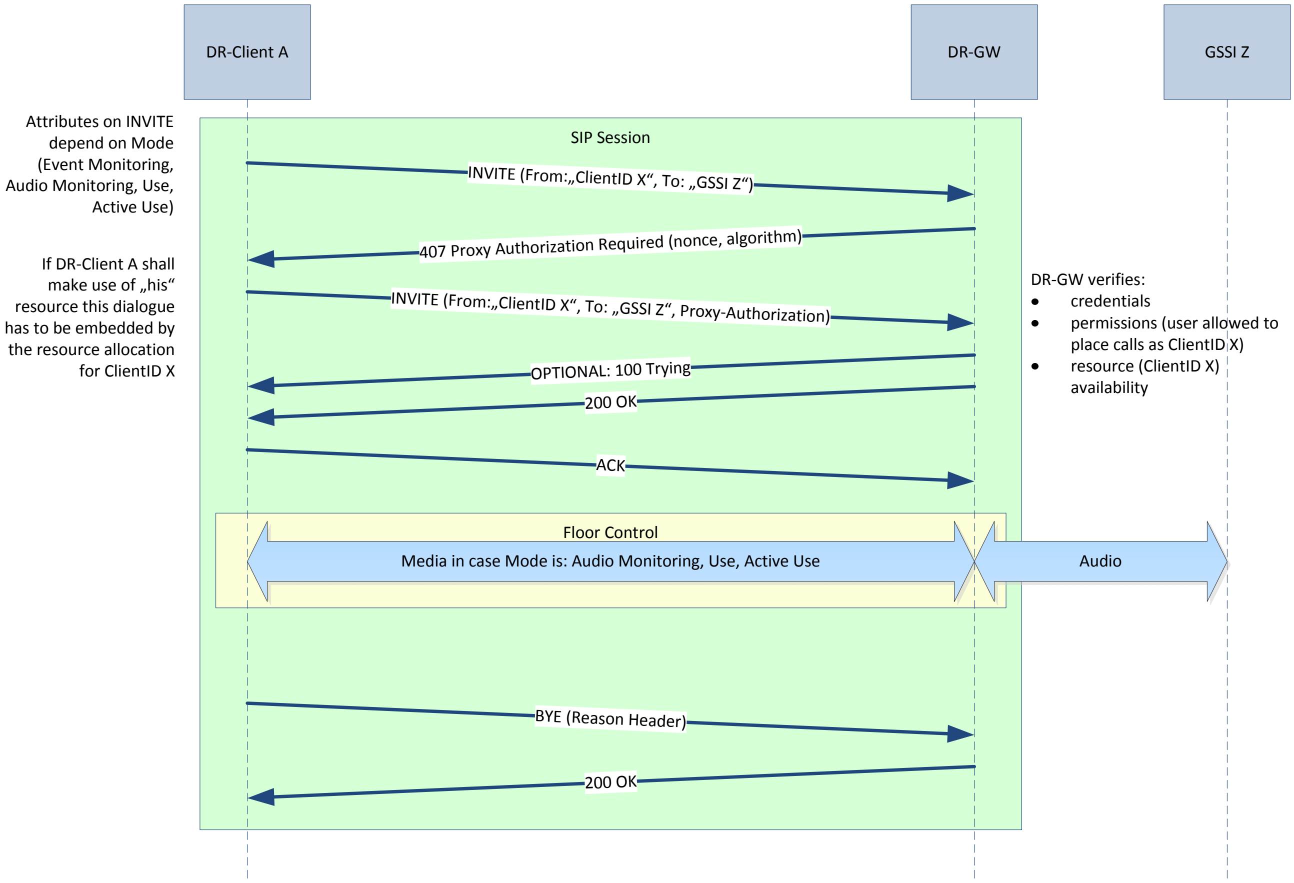


Figure 15 – MSC for successful Group Call

In standard behavior the call is cleared by the DR-GW-Client with the SIP BYE message as shown in Figure 15. The DR-GW shall tear down the SIP Session only in case of an error. In such case the DR-GW Reason Header has to be provided to indicate the specific failure according to 7.6.2.

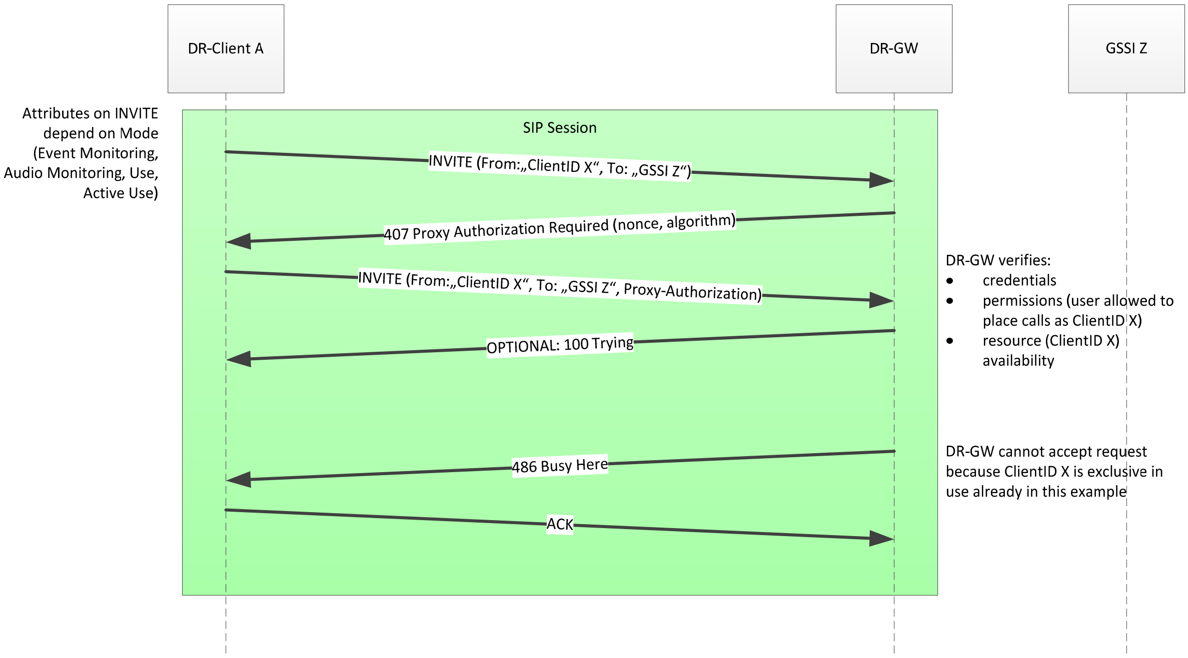


Figure 16 – MSC for GW rejected Group Call

5.5 Floor Control

5.5.1 Floor Life Cycle

Once a Talkgroup is selected the DR-GW-Client can start opening the floor. Explicitly (as shown in Figure 17) or implicitly (on first Tx Demand) the TETRA call SHALL be established by the DR-GW. After granting Tx by the DR-GW the floor is opened – the DR-GW-Client is able to use the audio Connection. This audible connection is closed when the Tx is ceased. After the TETRA timeout or initiated by the DR-GW-Client the DR-GW optionally tears down the TETRA call.

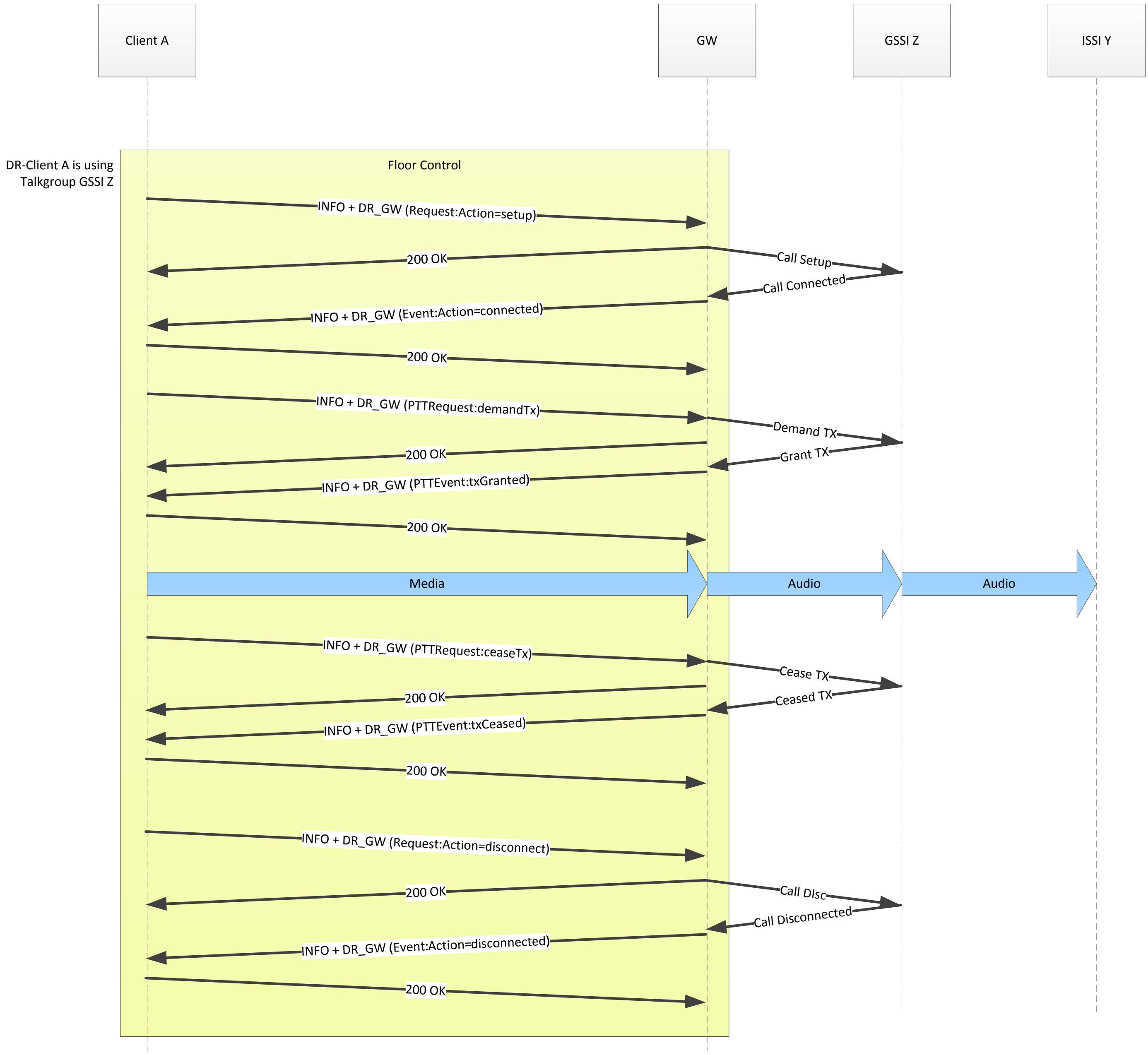


Figure 17 – MSC Floor Life Cycle

5.5.2 Floor Control Outgoing

Before the DR-GW-Client is able to transmit audio the Client has to demand the floor which has to be granted by the DR-GW. There is no need for the client to explicitly set up the TETRA call even though there is a possibility to do so. If the TETRA call is not yet established and the DR-GW-Client does not explicitly set it up, the DR-GW will set up the TETRA call implicitly.

To allow the DR-GW a safety feature such as “stuck-PTT detection” (in case of simple DR-GW-Clients similar to a plain SIP Phone with latching PTT activation) the DR-GW-Client SHOULD announce a refresh timeout in each of his “DEMAND Tx” messages. If the DR-GW-Client does not refresh his Tx demand within the announced timeout the DR-GW should revoke the Tx demand automatically.

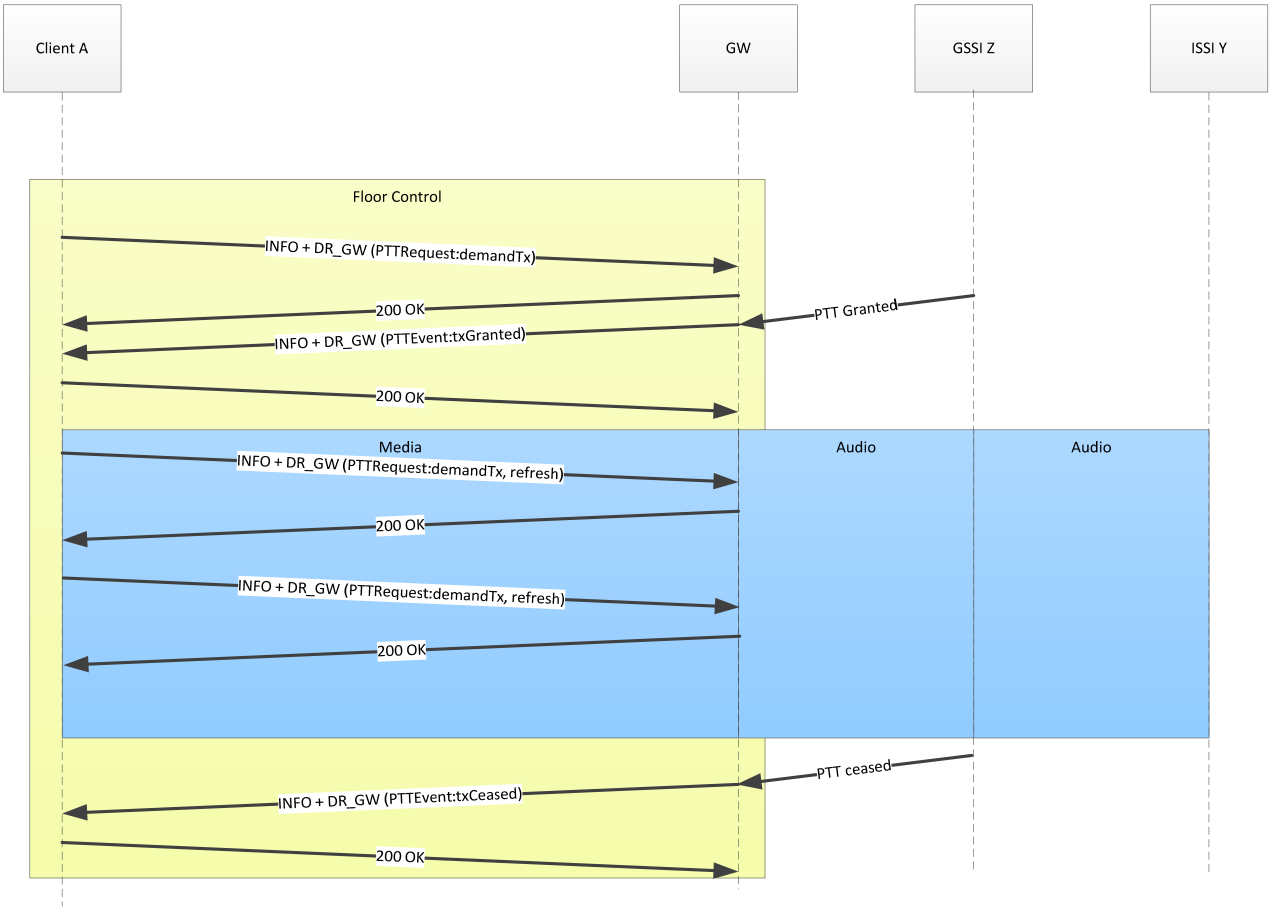


Figure 18 – MSC Floor Control Out

5.5.3 Floor Control Incoming

When the DR-GW-Client at least monitors events of a talkgroup each change of audio arbitration on that talkgroup is indicated via SIP INFO.

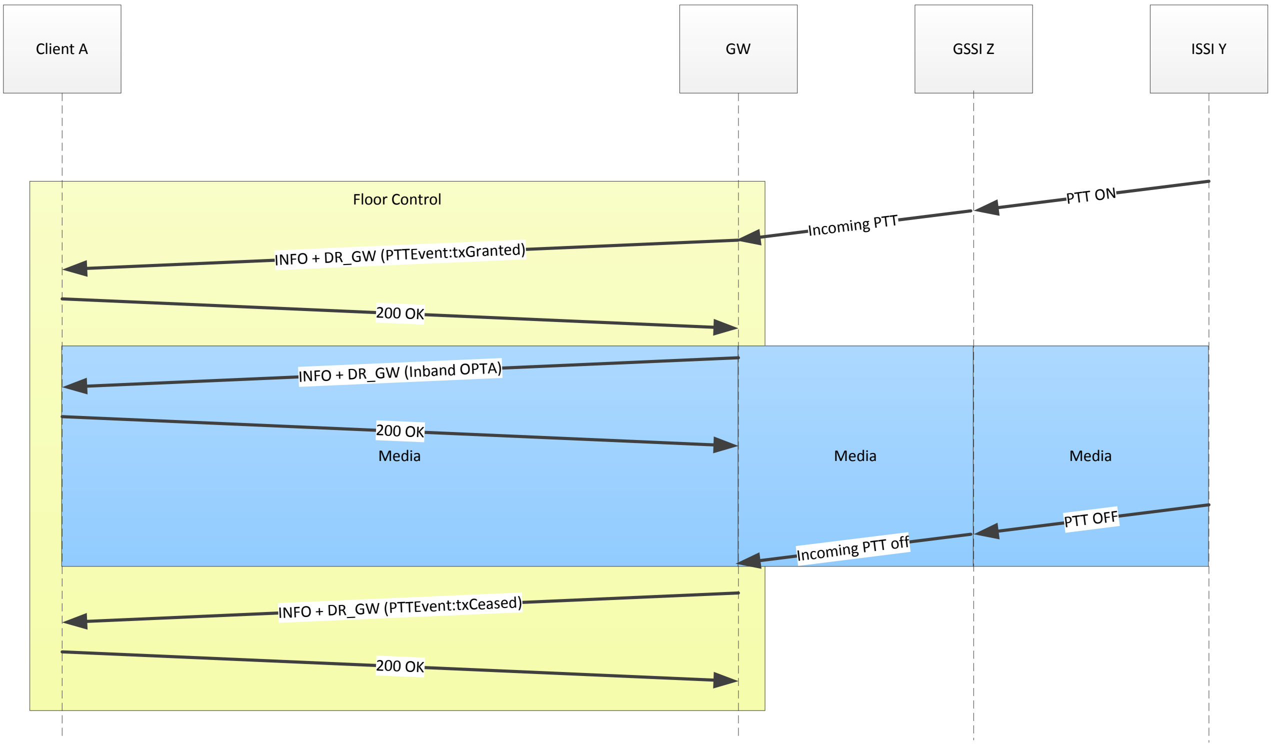


Figure 19 – MSC Floor Control In

5.6 Triggering of Key Exchange

5.6.1 Demanding new key for a TETRA Group

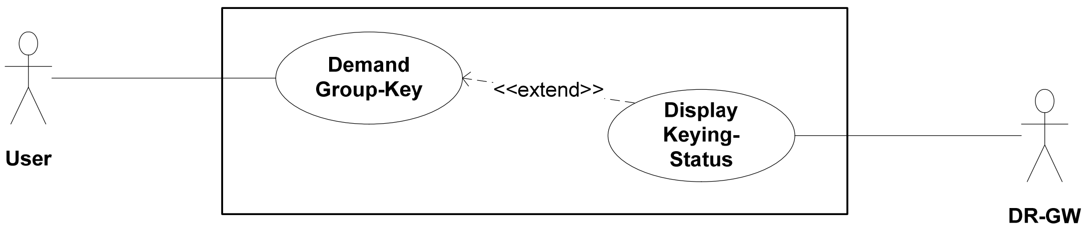


Figure 20 – Use Case Demanding a new key for a TETRA Group

The user may demand a new Key for a TETRA group. In case of rekeying the DR-GW informs the user about the keying status.

5.6.2 PROVIDING A NEW KEY FOR A TETRA GROUP

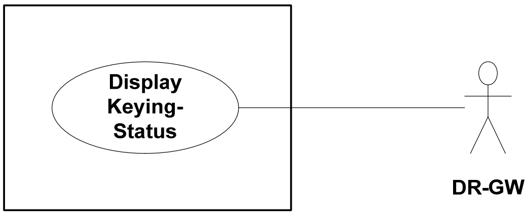


Figure 21 – Use Case Providing a new key for a TETRA Group

In the use case description, the presentation of the key status of a group or its change is shown from the direction of DR-GW.

5.6.3 MESSAGE SEQUENCE CHARTS

When one client demands a new key for a specific TETRA Group, each client which at least monitors events of that TETRA Group will be notified about the key exchange. To prevent the TETRA infrastructure from a denial-of-service attack (even if it is performed by mistake) the DR-GW shall accept and forward Key Exchange Request with a limited maximum rate.

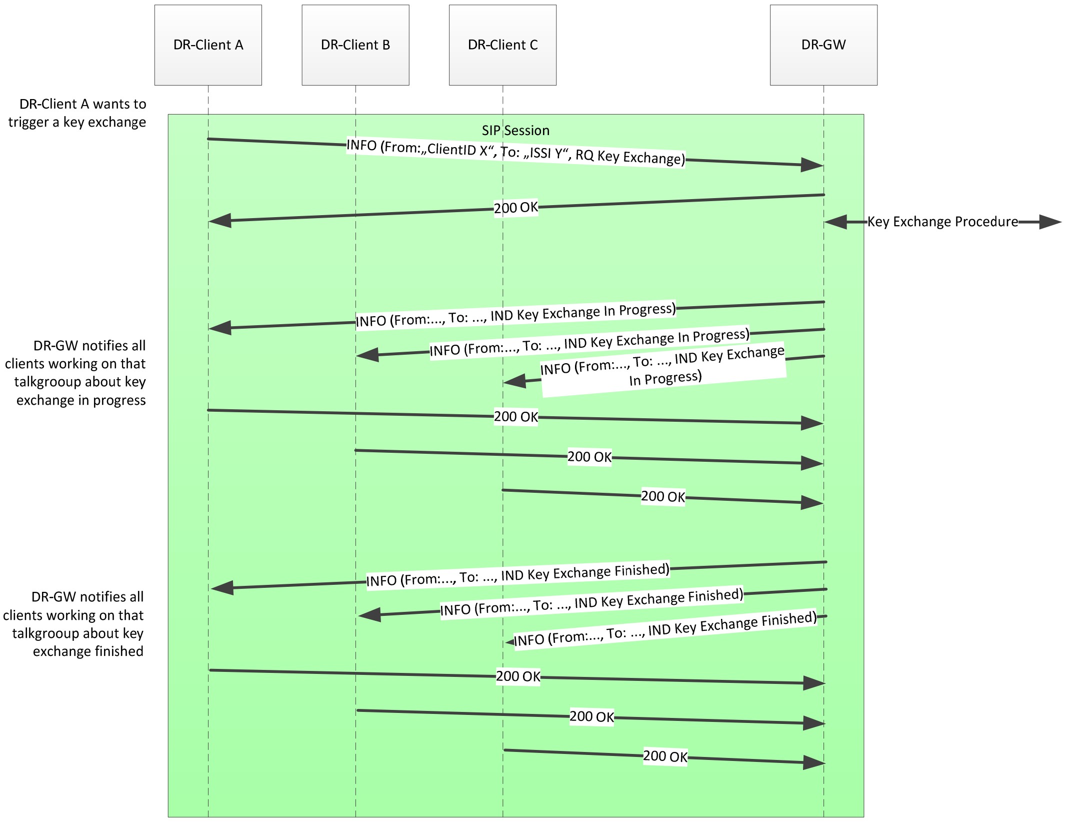


Figure 22 – MSC for DR-GW-Client demanding a new key

While Figure 22 shows successful demand for a new key for a certain group by one DR-GW-Client, Figure 23 shows demand which does not lead to a successful result.

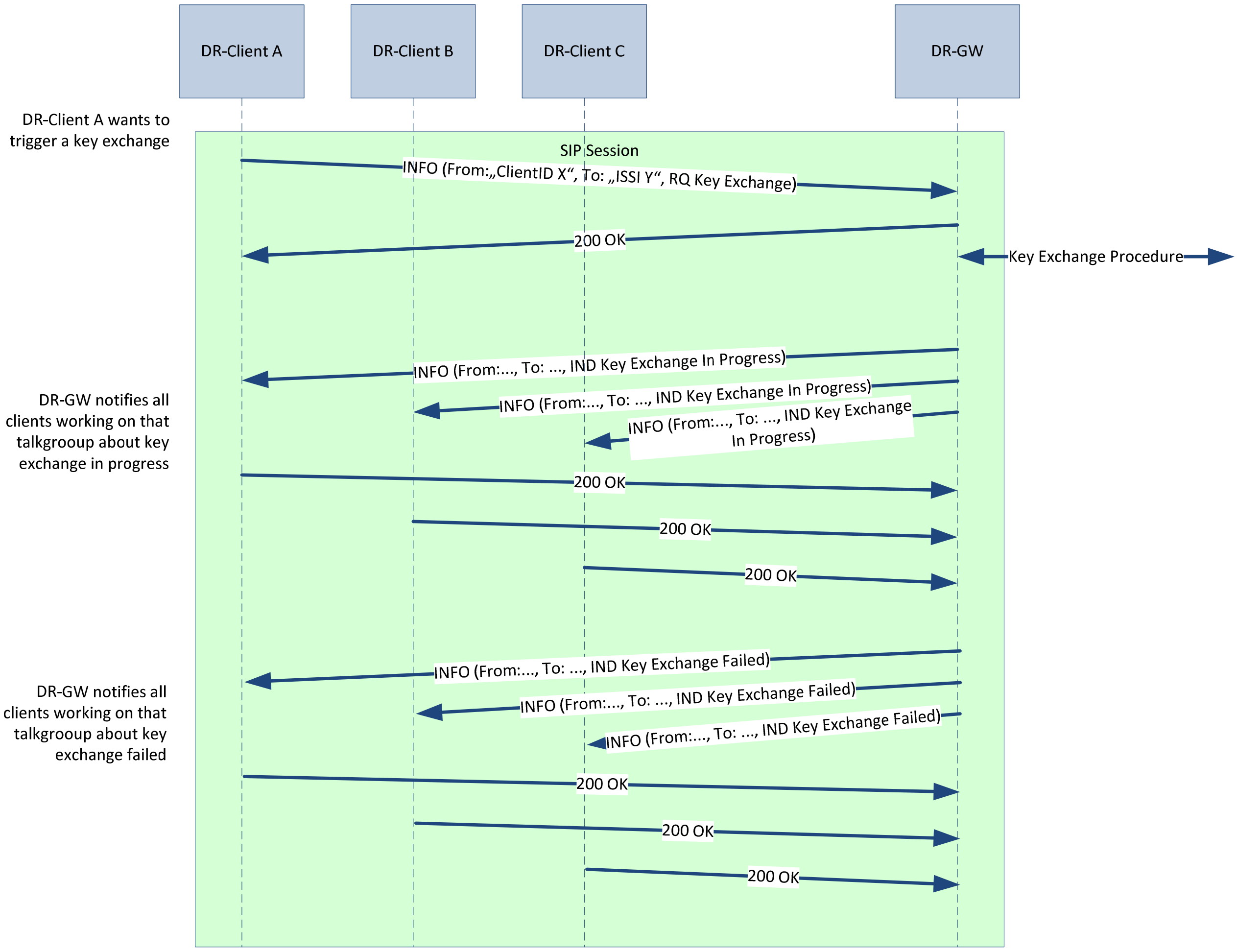


Figure 23 – MSC for unsuccessful new key demand

As the progress of key exchange is reported to each client which is at least event monitoring the specific talkgroup, such client can abort a key exchange during the processing. Such scenario is shown in Figure 24.

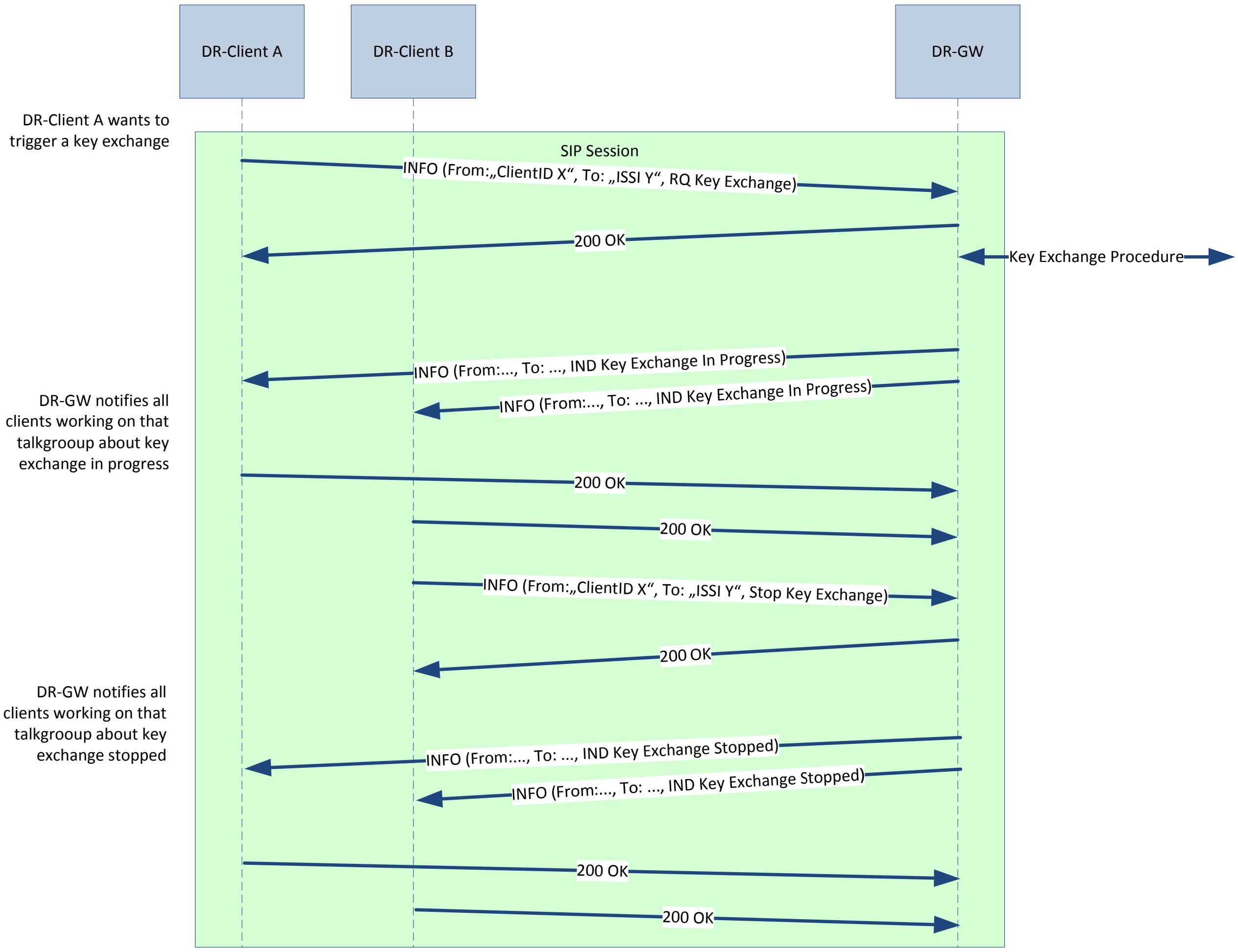


Figure 24 – MSC for client aborting new key demand

It is a matter of the rights management and implementation specifics of the DR-GW whether another client than the one who triggered the key exchange can abort the key exchange or not.

5.7 CALL HOLD

Even though it is obvious how the best practice for putting a SIP call on hold would look like, this version of the DR-GW Interface Specification does not yet include this feature. The detailed specification of this feature will be provided in the later version of this interface specification.

5.8 CALL TRANSFER

Even though there are some existing practices for transferring a call in the SIP world, for detailed selection of the model (attended call transfer, unattended transfer, blind transfer and supervised transfer) additional clarification is needed. For this reason, the detailed specification of this feature will be provided in the later version of this interface specification.

5.9 ERROR CONGESTION

Within the SIP Protocol part of this interface, errors are handled by rejecting requests, tearing down SIP Sessions and closing other dialogues (e.g. REGISTER) mainly by the DR-GW. There is no need to re-invent the wheel and to make the root cause on how to analyze failures and miss-behaviors clear, there shall be a way to propagate TCS error without any need for remapping to the DR-Client.

This aim is achieved by using the SIP Reason Header in the respective requests or responses. The same way as the SIP community propagates errors originated within a PSTN (public switched telephone network) the SIP Gateway is selected. Therefore the RFC3326. The Reason Header Field will be extended according to the chapter "7.6.2 DR-GW-Reason Header".

In case that the error condition may not be handled transparently for the client by the DR-GW, the DR-GW shall tear down the affected SIP sessions and close all other affected dialogues. For redundancy switchover the DR-Client has to perform the necessary action at the backup DR-GW as described in the chapter "4.4.6 DR-GW Failover functionality".

The SOAP Part will handle failures in a very similar way. Established connections shall be torn down by the DR-GW. The DR-Client has to switch to the backup DR-GW.

6 SECURITY ASPECTS

The DR-GW-Interface as interface based on IP will be run on infrastructure which is treated as “secure”. Therefore, neither Parkerian’s Hexad Model[1] nor the basic concept of CIA triad (confidentiality, integrity and availability) on application level has to be applied to provide required security aspects. The reason for security on the DR-GW-Interface’s application level, are needs of:

- Resource Management
- Audit Trail
- Accountability

While the first two items are mandatory and SHALL be implemented within the very first implementation, the latter is an optional feature which could be required by a specific customer. The basis for all these three features is authentication in a state of the art way where no credentials are propagated in a plain readable form over the network.

SIP protocol suite as being used by public telephone operators provides means for:

- Confidentiality (SIPS → SIP over TLS, SRTP with payload encryption)
- Authentication (SIP Authentication, SRTP with payload authentication)

SOAP over http protocol provides means for:

- Confidentiality (https → http over TLS)
- Authentication (digest authentication)

For the context of the DR-GW-Interface the authentication part of the control protocols SHALL be used which means:

- Each SIP dialogue established by the DR-GW-Client (REGISTER, INVITE) SHALL be authorized by the DR-GW upon user name password base using digest authentication
- Each SIP dialogue established to the DR-GW-Client SHALL be accepted only when being originated from the DR-GW
- Each SOAP Request originated by the DR-GW-Client SHALL be accepted by the DR-GW if a valid security token is presented. Security token exchange will be performed during SOAP login procedure.
- No confidentiality measures will be applied – no SIPS, no https, no SRTP

---

[1] Parker, Donn B. (1998). Fighting Computer Crime. New York, NY: John Wiley & Sons. ISBN 0-471-16378-3. The work in which Donn B. Parker introduced this model.

7 PROFILE STANDARD FOR THE USE OF SIP

7.1 SESSION INITIATION PROTOCOL

DR-GW and DR-GW-Client SHALL support SIP version 2 as specified in RFC 3261.

The SIP protocol is an application-layer control protocol which has been developed and designed within the IETF and is defined by RFC 3261. With respect to TETRA Radio applications the SIP protocol SHALL be used by each Digitalradio-Gateway (DR-GW)-Client to establish, modify and terminate a SIP session with a Digitalradio-Gateway (DR-GW).

Once a communication session between the DR-GW-Client and the DR-GW has been established using the SIP protocol, the two endpoints SHALL then employ the Real time Transport Protocol (RTP) (RFC 3550) communication. Once the RTP is active this communication SHALL be used for the transport of audio in RTP packets between the endpoints (as defined in RFC 3550).

The audio transport MAY be augmented by its associated control protocol (RTCP) (RFC 3550 [21]) to allow monitoring of voice packet delivery.

7.2 LOGICAL SIP ENTITIES

The DR-GW-Client with respect to TETRA Radio applications is acting as SIP User Agent. The DR-GW in the same context is acting as SIP Registrar, SIP Proxy Server and SIP User Agent. Both entities (DR-GW, DR-GW-Client) acting as SIP User Agent have to implement both roles the UAC (User Agent Client) as well as UAS (User Agent Server).

7.2.1 USER AGENTS

User Agents in a TETRA Radio environment SHALL support the following services and procedures:

7.2.1.1 REGISTRATION

Registration Discovery in the one and only variant “configured registrar address” according to RFC 3261, section 10.2.6

Adding Bindings according to RFC 3261, section 10.2.1

Removing Bindings according to RFC 3261, section 10.2.2

Refreshing Bindings according to RFC 3261, section 10.2.4

Ordering Contacts according to RFC 3261, section 10.2.1.2

7.2.1.2 CALL CONTROL

7.2.1.2.1 Establishing a session

UAC procedures according to RFC 3261, section 13.2

UAS procedures according to RFC 3261, section 13.3

Based on Location server messages procedures according to RFC 3261, section 8.1.3.4

7.2.1.3 Terminating a session with BYE

UAC procedures according to RFC 3261, section 15.1.1

UAS procedures according to RFC 3261, section 15.1.2

7.2.1.4 Cancelling a session

UAC procedures according to RFC 3261, section 9.1

UAS procedures according to RFC 3261, section 9.2

7.2.2 Registrar

DR-GW acting as SIP Registrar in a TETRA Radio environment SHALL support the following services and procedures:

7.2.2.1 Registration

Maintaining Bindings according to RFC 3261, section 10.3

Ordering contacts according to RFC 3261, section 10.2.1.2

Unicast Registration according to RFC 3261, section 10.3

7.2.3 Proxy Server

DR-GW acting as SIP Proxy Server in a TETRA Radio environment SHALL support the following services and procedures:

7.2.3.1 Call Control

7.2.3.1.1 Establishment a session

Stateful procedures according to RFC 3261, sections 16 and 8.2

Based on Location server messages according to RFC 3261, sections 16 and 8.2

7.2.3.1.2 Terminating a session with BYE

Stateful procedures according to RFC 3261, sections 16 and 8.2

7.2.3.1.3 Cancelling a session

Stateful procedures according to RFC 3261, sections 16 and 8.2

7.3 Supported Requests

Method	DR-GW-Client		DR-GW-Server	
	Sending	Receiving	Sending	Receiving
INVITE	m	m	m	m
ACK	m	m	m	m
CANCEL	m	m	m	m
BYE	m	m	m	m
REGISTER	o	-	x	m
INFO	o	o	m	m
SUBSCRIBE	o	o	o	o
NOTIFY	o	o	o	o
UPDATE	o	o	o	m
OPTIONS	o	o	o	o
REFER	o	o	x	o
MESSAGE	o	o	o	o
PUBLISH	o	o	o	o
PRACK	o	m	o	m

"m": mandatory; "o": optional; "x": prohibited; "-": not applicable

The requirements of RFC 2976 (SIP INFO), RFC 3261 (SIP 2), RFC 3262 (Provisional Responses), RFC 3311 (SIP UPDATE) and RFC 3515 (SIP REFER) and RFC 4028 (SIP Session Timers) SHALL apply.

7.3.1 BYE

In case of group Communication the DR-GW SHALL disconnect a SIP session with BYE only if error occurs.

7.3.2 INFO

SIP INFO is used for following reasons:

- arbitration of the floor in case of group communication and half duplex individual call
- indicating and control of TETRA Call which has been established and torn down
- trigger and notification of new TETRA end 2 end encryption key exchange

7.4 Supported Responses

Response	DR-GW-Client		DR-GW-Server	
	Sending	**Receiving**	**Sending**	**Receiving**
100 – Trying	o	m	o	m
180 – Ringing	o	m	o	m
181 – Call Is Being Forwarded	-	m	o	-
182– Queued	-	m	x	-
183 – Session Progress	o	m	o	m
200 – Ok	m	m	m	m
202 – Accepted	o	o	o	o
300 – Multiple Choices	x	-	x	-
301 – Moved Permanently	o	m	o	m
302 – Moved Temporarely	o	m	o	m
400 – Bad Request	-	m	m	-
401 – Unauthorized	x	m	o	-
404 – Not Found	o	m	o	m
405, 406	-	m	m	-
407 – Proxy Authentication Required	x	m	o	-
408, 410,413, 414	-	m	m	-
415 – Unsupported Media Type	-	m	m	-

Response	DR-GW-Client		DR-GW-Server	
	Sending	Receiving	Sending	Receiving
416 – Unsupported URI Scheme	-	m	m	-
420,421, 423		m	o	m
422 – Session Interval To Small	m	m	m	m
480 – Temporarily Unavailable	-	m	m	-
481 -485, 487, 489	-	m	o	-
486 – Busy Here	o	m	m	m
488 – Not Acceptable Here	m	m	m	m
491 – Request Pending	-	m	m	-
493 – Undecipherable	-	m	m	-
501 – Request Not Supported	m	m	m	m
502 -505, 513	-	m	o	-
603 – Decline	m	m	m	m
604, 606	o	m	o	m

“m”: mandatory; “o”: optional; “x”: prohibited; “-“: not applicable

7.5 Supported SIP Header Fields

Complete and sufficient description of mandatory and optional Header Fields

7.5.1 User Agent Request Headers

UA Request Header Field	Requests										
	ACK	BYE	CAN	INV	MES	NOT	OPT	REF	REG	SUB	UPD
Allow	---	o	---	o	o	o	o	o	o	o	o
Allow-Events (RFC 3265)	o	o	---	o	---	o	o	---	o	o	---
Authorization	o	o	o	o	o	o	o	o	o	o	o
Call-ID	m	m	m	m	m	m	m	m	m	m	m
Contact	o	---	---	m	---	m	o	m	o	m	m
Content-Length	m	m	m	m	m	m	m	o	m	m	m
Content-Type	*	*	---	*	*	*	*	*	*	*	*
Cseq	m	m	m	m	m	m	m	m	m	m	m
Date	o	o	o	o	o	o	o	o	o	o	o
Event (RFC 3265)	---	---	---	---	---	m	---	---	---	m	---
Expires	---	---	---	o	o	---	---	o	o	o	---
From	m	m	m	m	m	m	m	m	m	m	m
In-Reply-to	---	---	---	o	o	---	---	---	---	---	---
Join (RFC 3911)	---	---	---	o	---	---	---	---	---	---	---
Max-Forwards	m	m	m	m	m	m	m	m	m	m	m
MIME-Version	o	o	---	o	---	o	o	o	o	o	o
Priority	---	---	---	m	o	---	---	---	---	o	---
Proxy-Authorization	o	o	---	o	o	o	o	o	o	o	o

UA Request Header Field	Requests										
	ACK	BYE	CAN	INV	MES	NOT	OPT	REF	REG	SUB	UPD
Proxy-Require	---	o	---	o	o	o	o	o	o	o	o
Record-Route	o	o	o	o	---	o	o	o	---	o	o
Reason	---	o	o	---	---	---	o	---	---	--	---
Refer-To (RFC 3515)	---	---	---	---	---	---	---	o	---	---	---
Replaces (RFC 3891)	---	---	---	o	---	---	---	---	---	---	---
Reply-to	---	---	---	o	o	---	---	---	---	---	---
Require	---	c	---	c	c	o	c	c	c	o	c
Route	c	c	c	c	o	c	c	c	c	c	c
Subject	---	---	---	m	o	---	---	---	---	---	---
Subscription-State (RFC 3265)	---	---	---	---	---	m	---	---	---	---	---
Supported	---	o	o	m*	---	o	o	o	o	o	o
To	m	m	m	m	m	m	m	m	m	m	m
Via	m	m	m	m	m	m	m	m	m	m	m
DR-GW-Version	m	m	m	m	m	m	m	m	m	m	m

7.5.2 USER AGENT RESPONSE HEADERS

UA Response Header Field	Status Code	Requests										
		ACK	BYE	CAN	INV	MES	NOT	OPT	REF	REG	SUB	UPD
Allow	2xx	---	o	---	m	o	o	m*	---	o	o	o
Allow	405	---	m	---	m	m	m	m*	m	m	m	m

UA Response Header Field	Status Code	Requests										
		ACK	**BYE**	**CAN**	**INV**	**MES**	**NOT**	**OPT**	**REF**	**REG**	**SUB**	**UPD**
Allow	All except 2xx, 415	---	o	---	o	o	o	o	o	o	o	o
Allow-Events (RFC 3265)	2xx	o	o	---	o	---	o	o	---	o	o	---
Allow-Events (RFC 3265)	489	---	---	---	---	---	m	---	---	---	m	---
Authentication-Info	2xx	---	o	---	o	o	o	o	o	o	o	o
Call-ID	All	m	m	m	m	m	m	m	m	m	m	m
Contact	1xx	---	---	---	o	---	o	---	---	---	o	o
Contact	2xx	---	---	---	m	---	o	o	m	o	m	m
Contact	3xx	---	o	---	o	o	m	o	---	o	m	o
Contact	485	---	o	---	o	o	o	o	o	o	o	o
Content-Length	All	m	m	m	m	m	m	m	o	m	m	m
Content-Type	All	*	*	---	*	*	*	*	*	*	*	*
Cseq	All	m	m	m	m	m	m	m	m	m	m	m
Date	All	o	o	o	o	o	o	o	o	o	o	o
Expires	2xx	---	---	---	o	---	---	---	---	o	---	---
From	All	m	m	m	m	m	m	m	m	m	m	m
MIME-Version	All	o	o	---	o	---	o	o	o	o	o	o
Min-Expires	423	---	---	---	---	---	---	---	---	m	m	---
Proxy-Authenticate	407	---	m	---	m	m	m	m	m	m	m	m
Proxy-Authenticate	401	---	o	o	o	o	---	o	o	o	---	o

UA Response Header Field	Status Code	Requests										
		ACK	**BYE**	**CAN**	**INV**	**MES**	**NOT**	**OPT**	**REF**	**REG**	**SUB**	**UPD**
Reason	3xx, 4xx, 6xx	---	---	---	o	---	--	o			o	
Record-Route	2xx, 18x	o	o	o	o	---	---	o	o	---	---	o
Record-Route	401, 484	---	---	---	---	---	o	---	---	---	o	---
Reply-To	All	---	---	---	o	o	---	---	---	---	---	---
Require	All	---	c	---	c	c	o	c	c	c	o	c
Supported	2xx	---	o	o	m*	---	o	m*	o	o	o	o
To	All	m	m	m	m	m	m	m	m	m	m	m
Unsupported	420	---	m	---	m	o	o	m	o	m	o	m
Via	All	m	m	m	m	m	m	m	m	m	m	m
Warning	All	---	o	o	o	o	o	o	o	o	o	o
WWW-Authenticate	401	---	m	---	m	m	m	m	m	m	m	m
WWW-Authenticate	407	---	o	---	o	o	---	o	o	o	---	o
DR-GW-Version	All	m	m	m	m	m	m	m	m	o	m	m

7.6 Definition of SIP Header Fields

7.6.1 DR-GW-Version

The DR-GW -Version header field SHALL appear in any request and in any response, but SIP elements need to be prepared to receive messages without that header field.

The syntax of the header field follows the standard SIP parameter syntax as defined in RFC 3261 and SHALL have the following content:

```
DR-GW-Version = “DR_GW-Version” HCOLON version-value *(COMMA version-value)
version-value = field-value *(SEMI version-params)
field-value = type “.” number
type = “radio” / “phone”
number = 2*DIGIT
```

The field-value SHALL contain the latest document version that reflects the implemented version of the corresponding interface specification as listed in the section ”4.2.4.2. SIP Interface”. Implementations MUST be able to process multiple header field rows with the same name in any combination of the single-value-per-line or comma-separated value forms. Respective actions may be specified where applicable.

Example:

```
DR-GW-Version: radio.01
```

NOTE:Version-params may be used for future extensions and are not described further in this document.

7.6.2 DR-GW-Reason Header

The DR-GW Reason Header may appear in any 3xx, 4xx, 5xx 6xx response answering a SIP INVITE. Additionally the DR-GW Reason Header may be sent in any request except for a SIP INFO request.

RFC 3326 will be extended in the way (see RFC 3326 “2. The Reason Header Field”):

```
protocol =  "SIP" / "Q.850" / “DR-GW ”
```

Definition of Error Values missing:

Cause	Text
0	Not defined or unknown
1	User requested disconnect
2	Called party busy
3	Called party not reachable
4	Called party does not support encryption
5	Congestion in infrastructure
6	Not allowed traffic case
7	Incompatible traffic case

Cause	Text
8	Requested service not available
9	Pre-emptive use of resource
10	Invalid call identifier
11	Call rejected by the called party
12	No idle cc entity
13	Expiry of timer
14	SwMI requested disconnection
15	Acknowledged service not completed
16	Unknown tetra identity
17	SS specific disconnection
18	Unknown external subscriber identity
19	Call restoration of the other user failed
20	Called party requires encryption
21	Concurrent setup not supported
22	Called party is under the same DM Gate of the calling party

(Ref: ETSI EN 300 392-2, chapt. 14.8.18)

Examples at the end should look like:

```
Reason: DR-GW ;cause=3 ;text="Called party not reachable"
```

7.7 MESSAGE BODY

7.7.1 SDP MESSAGE BODY

The SDP Message body is encoded according to RFC4566. Deviations, interpretations and extensions to that standard are stated within these chapters.

7.7.1.1 SELECTION LEVEL

In order to separate technical and operational connection demand the selection level is propagated either via the DR-GW-Interface xml Body or via a SDP attribute as described herein. In this particular case it is possible to establish a silence RTP connection, even in case of audio Monitoring only. The purpose is to „technically monitor" the other party which is still alive (as an option) in order to immediately release resources in case of crashed peer.

Attributes ("a=")	type:<call type>	Radio-Idle Radio-Rxonly Radio-TxRx or Radio (default value) Coupling

7.7.1.2 CODEC PRIMITIVES

In general the DR-GW-Interface supports 3 different Codec types. The G.711, ACELP and the FSTE-OSTE Codec.

G.711 is well defined in the RFC4566. The utilization of ACELP in the RTP Payload and the respective SDP parameters are defined in the document "Real-Time Transport Protocol (RTP) Payload Format for the TETRA Audio Codec". It has to be mentioned that the RTP packets contain exactly one ACELP packet – the packetization on the UDP layer matches with the one on the codec layer. This fact is well described in the document above.

When encrypted payload is in use the entire double frame of FSTE or OSTE is transported within one RTP packet. In such case this RTP packet contains 60ms of voice.

7.7.1.3 PTIME PARAMETER

The ptime parameter allows to negotiate the packet size. Depending on the codec, following packet sizes SHALL be supported by the DR-GW:

Codec	ptime	20 ms	30 ms	60 ms
G.711		X	X	X
ACELP			X	X
FSTE / OSTE				X
Any other codecs			X	X

7.7.1.4 MONITORING SESSION WITH ACELP ENCODING

Monitoring Session with ACELP encoding – Example:

```
v=0
o=yourClient 0 2 IN IP4 172.31.60.191
s=subject
t=0 0
m=audio 8196 RTP/AVP 99
c=IN IP4 172.31.60.191
```

```
a=rtpmap:99 TETRA/8000
a=recvonly
a=type:Radio-Rxonly
```

7.7.1.5 MONITORING SESSION WITH G.711 A-LAW ENCODING

The speciality of this SDP is the SIP session negotiation in full duplex while the selection mode is Rx only. This means the client expects RTP packets even if he does not make use of it for audio propagation. An application for such a sdp is negotiation to provide heartbeat supervision based on RTP which is very quick and highly reliable in detection of failures.

Monitoring Session with G.711 A-Law encoding – Example:

```
v=0
o=myclient 0 0 IN IP4 192.168.12.97
s=conversation
c=IN IP4 192.168.12.97
t=0 0
m=audio 10020 RTP/AVP 8
a=rtpmap:8 pcma/8000
a=sendrecv
a=type:Radio-Rxonly
```

7.7.1.6 SELECTED TALKGROUP WITH ACELP ENCODING

Selected Talkgroup with ACELP encoding – Examble:

```
v=0
o=hisClient 0 2 IN IP4 10.180.22.93
s=subject
t=0 0
m=audio 8196 RTP/AVP 99
c=IN IP4 10.180.22.93
a=rtpmap:99 TETRA/8000
a=sendrecv
a=type:Radio-TxRx
```

7.7.2 FLOOR CONTROL MESSAGE BODY

The following example illustrates the structure of FLOOR CONTROL MESSAGE BODY:

```
<?xml version="1.0" encoding="utf-8"?>
<Call_PTTEvent>
  <action>txGranted</action>
```

```
    <txGrant>granted</txGrant>
    <talkingParty>
      <subscriber>
        <tsisimple>248.162.70000001</tsisimple>
      </subscriber>
      <opta>OPTA123789</opta>
    </talkingParty>
    <attributes>
      <demandPriority>normal</demandPriority>
    </attributes>
    <txrepeat>30</txrepeat>
    <workstationId>Place 231</workstationId>
  </Call_PTTEvent>
```

7.7.3 Key Exchange Triggering Message Body

Any kind of key management is performed based on in-dialogue SIP INFO messages with xml structures in the SIP message body. The particular xml schema with exemplary xml content will be specified in a subsequent revision of this document.

8 PROFILE STANDARD FOR THE USE OF SOAP

DR-GW and DR-GW-Client SHALL support SOAP in version 1.2.

SOAP provides a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized, distributed environment using XML.

In this document the SOAP is used in combination with HTTP.

Complete definition and description of XML data, used in this interface, is defined in the Chapter 10: *Interface Definition*.

9 REQUEST, RESPONSE AND EVENT SYSTEM

SOAP 1.2 is the chosen encapsulation layer for propagating REQUESTS and the corresponding RESPONSES. For this purpose the standard transport layer of http 1.1 is in use. To be able to pass firewalls as well as NAT routers EVENTS from the DR-GW to the DR-Client are propagated via websocket technology. The websocket itself is opened by the client so no server port at the DR-Client side is needed for this kind of communication relation. To simplify and harmonize implementation with the REQ/RESP part of the protocol SOAP 1.2 encapsulation is used to propagate and serialize EVENTS from the DR-GW to the DR-Client.

9.1 RESPONSE OR EVENT DECISION DIAGRAM

This diagram shows whether the result of TCS Method is in DR-GW-Interface represented as a Response or as an Event.

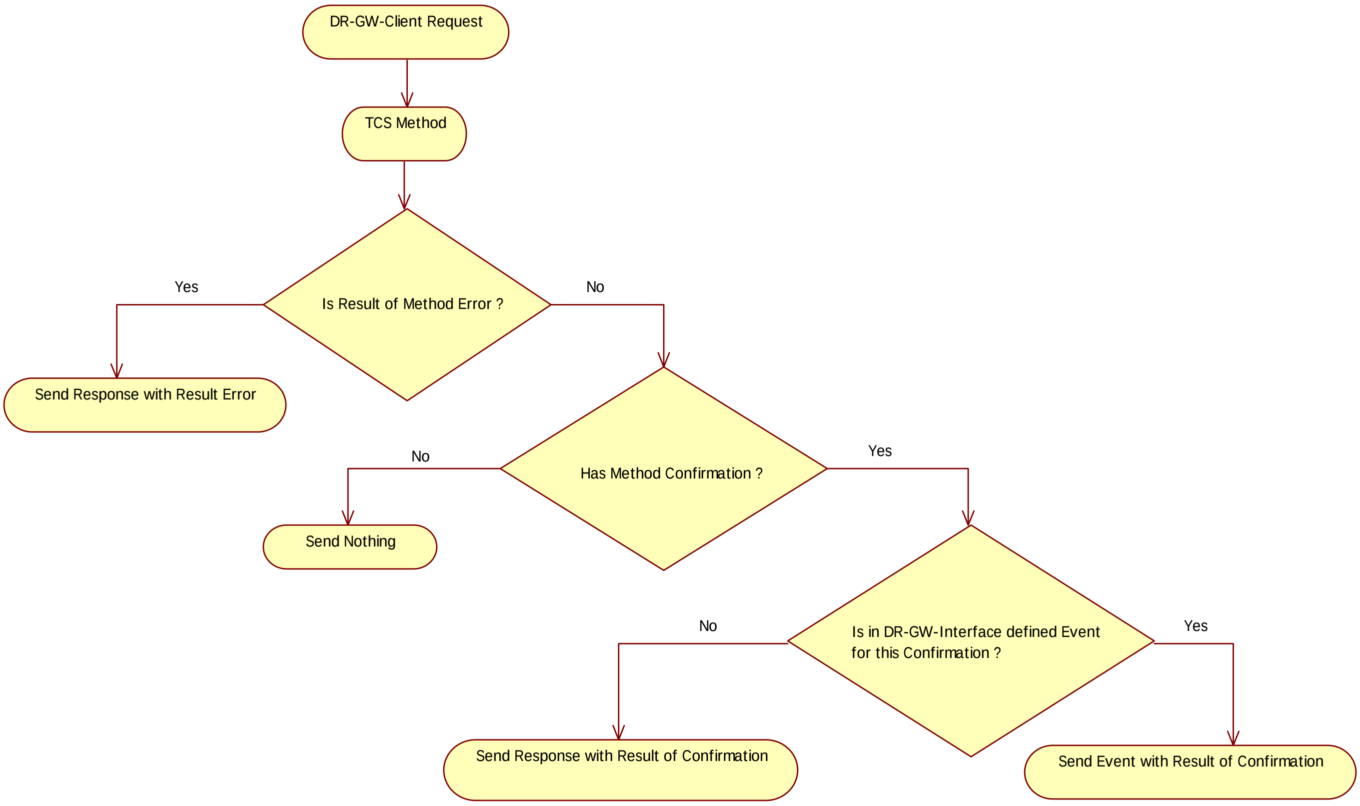


Figure 25 – EVENT DECISION DIAGRAM

9.2 GATEWAY HAS DATA

This diagram shows situation when DR-GW-Client is making Request, which result is already known in DR-GW, so there's no need to ask TCS, the result can be directly sent from DR-GW.

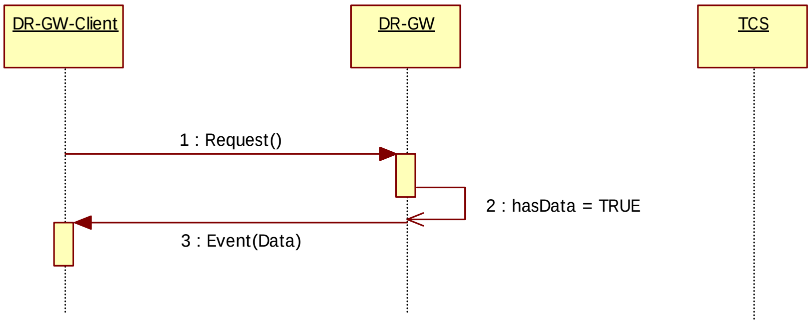


Figure 26 – DATA EVENT

9.3 GETTING DATA USING EVENT

This diagram shows the most common Request-Response scenario. The DR-GW-Client's request is resulting into DR-GW's call of TCS's Method. TCS's Method confirmation is then represented as DR-GW's Response with Result and TCS's Indication with Data is represented as DR-GW's Event with Data.

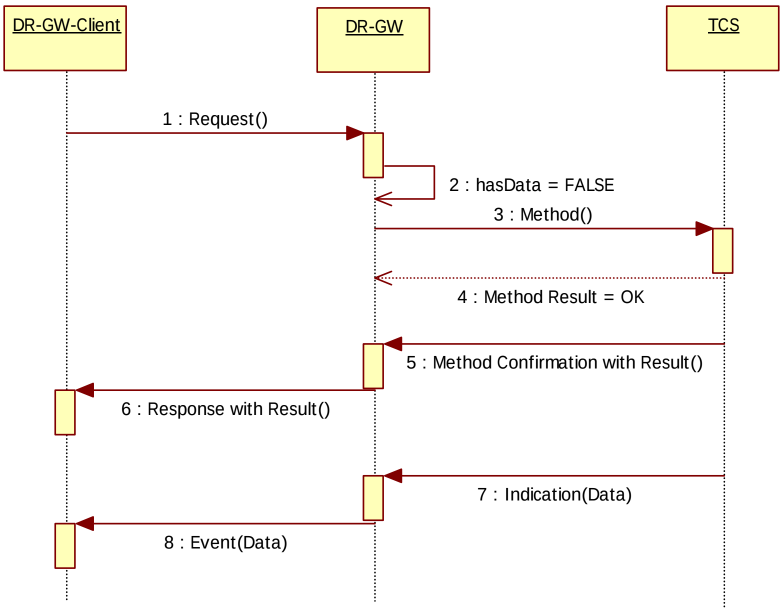


Figure 27 – DATA USING EVENT

9.4 RESULT IS ERROR

This diagram shows scenario, when the DR-GW-Client's Request is resulting in TCS's error response, which is then represented as DR-GW's Response with Error.

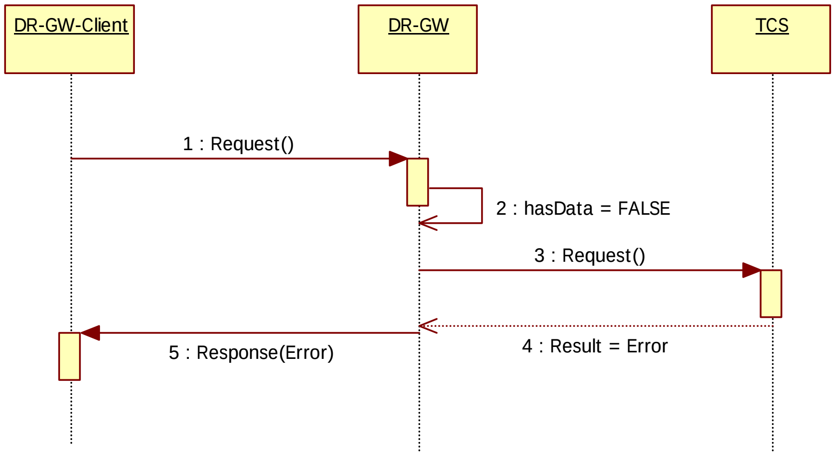


Figure 28 – ERROR RESULT

10 INTERFACE DEFINITION

10.1 INTERFACE DIAGRAM

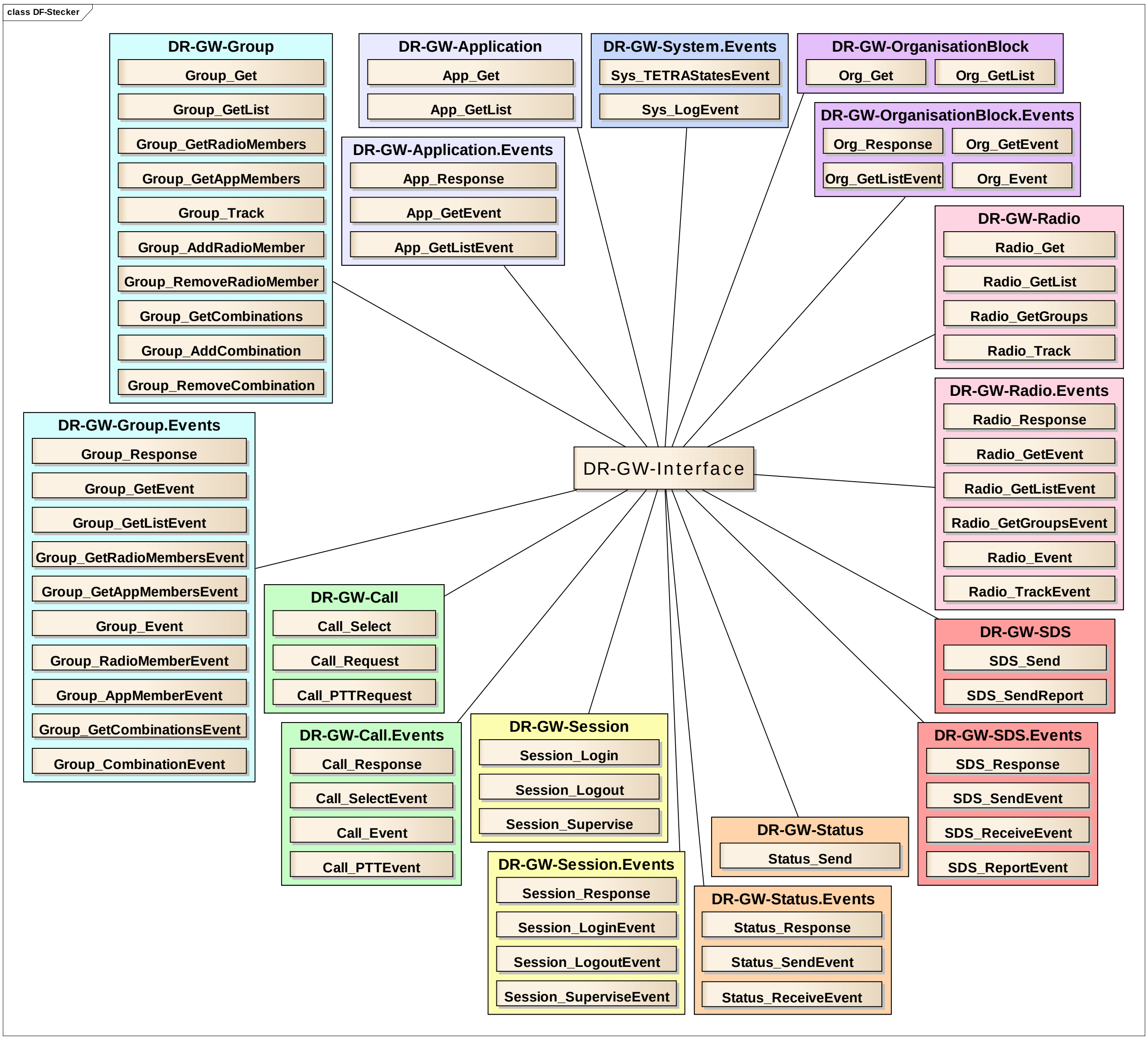


Figure 29 – DR-GW Graph

The Interface Diagram represents all method calls the DR-GW interface for Tetra call handling (in SIP) and Tetra messaging/control (in Soap) . See for details also the schema definitions in the appendix "DR-GW-Schemas".

10.2 REQUEST INTERFACE: DR-GW-SESSION

10.2.1 REQUEST: SESSION_LOGIN

This method is used to perform login procedure.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
clientId	xs:string	1..1	DR-GW-Client identifier
supervise	ctS:typeSuperviseTimeout	0..1	Supervise timeout in seconds (20, 30, 60)
version	xs:string	0..1	Client version

10.2.2 REQUEST: SESSION_LOGOUT

This method is used to perform logout procedure.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
token	xs:string	1..1	Session token

10.2.3 Request: Session_Supervise

This method is used to perform supervised action.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
token	xs:string	1..1	Session token

10.3 Response and Event Interface: DR-GW-Session.Events

10.3.1 Response: Session_Response

This is a general response for DR-GW-Session.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.3.2 Event: Session_LoginEvent

This event is generated as a result of DR-GW-Client Session_Login.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	0..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
token	xs:string	1..1	Session token
version	xs:string	0..1	Server version

10.3.3 EVENT: SESSION_LOGOUTEVENT

This event is generated as a result of DR-GW-Client Session_Logout.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	0..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
token	xs:string	1..1	Session token
reason	xs:unsignedLong	0..1	Reason of logout event

10.3.4 EVENT: SESSION_SUPERVISEEVENT

This event is generated as a result of DR-GW-Client Session_Supervise.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	0..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
token	xs:string	1..1	Session token

10.4 REQUEST INTERFACE: DR-GW-SDS

10.4.1 REQUEST: SDS_SEND

This method is used to send SDS.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
sds	ctS:typeSDS	1..1	All SDS attributes

10.4.2 Request: SDS_SendReport

This method is used to send SDS report on received SDS.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
msgRef	xs:unsignedByte	0..1	Message reference
deliveryStatus	xs:unsignedByte	1..1	Delivery status

10.5 Response and Event Interface: DR-GW-SDS.Events

10.5.1 Response: SDS_Response

This is a general response for DR-GW-SDS.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.5.2 Event: SDS_SendEvent

This event is generated as a result of DR-GW-Client SDS_Send.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
msgRef	xs:unsignedByte	1..1	Message reference

10.5.3 EVENT: SDS_RECEIVEEVENT

This event is generated upon receiving SDS.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
sds	ctS:typeSDS	1..1	All SDS attributes

10.5.4 EVENT: SDS_REPORTEVENT

This event is generated upon receiving SDS report.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
msgRef	xs:unsignedByte	0..1	Message reference
deliveryStatus	xs:unsignedByte	1..1	Delivery status

10.6 REQUEST INTERFACE: DR-GW-STATUS

10.6.1 REQUEST: STATUS_SEND

This method is used to send status.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
status	ctS:typeStatus	1..1	All Status attributes

10.7 Response and Event Interface: DR-GW-Status.Events

10.7.1 Response: Status_Response

This is a general response for DR-GW-Status.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.7.2 Event: Status_SendEvent

This event is generated as a result of DR-GW-Client Status_Send.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description

10.7.3 Event: Status_ReceiveEvent

This event is generated upon receiving status.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
status	ctS:typeStatus	1..1	All Status attributes

10.8 Request interface: DR-GW-OrganisationBlock

10.8.1 Request: Org_Get

This method is used to get organization block attributes.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
orgblockId	ctO:typeOrganisationBlockId	1..1	Organization block id

10.8.2 Request: Org_GetList

This method is used to get organization block subtree attributes.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
orgblockId	ctO:typeOrganisationBlockId	0..1	Organization block id

10.9 RESPONSE AND EVENT INTERFACE: DR-GW-ORGANISATION-BLOCK.EVENTS

10.9.1 RESPONSE: ORG_RESPONSE

This is a general response for DR-GW-OrganisationBlock. Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.9.2 EVENT: ORG_GETEVENT

This event is generated as a result of DR-GW-Client Org_Get.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
orgblock	ctO:typeOrganisationBlock	1..1	Organization block attributes

10.9.3 EVENT: ORG_GETLISTEVENT

This event is generated as a result of DR-GW-Client Org_GetList.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description

orgblock	ctO:typeOrganisationBlock	0..N	Organization block attributes
listEnd	xs:boolean	0..1	End of list indicator

10.9.4 EVENT: ORG_EVENT

This event is generated upon creation, modification or deletion of organization block.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
orgblock	ctO:typeOrganisationBlock	1..1	Organization block attributes
delete	xs:boolean	0..1	Deletion indicator

10.10 REQUEST INTERFACE: DR-GW-GROUP

10.10.1 REQUEST: GROUP_GET

This method is used to get group attributes.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address

10.10.2 REQUEST: GROUP_GETLIST

This method is used to get groups and their attributes belonging to a given organization block.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
orgblockId	ctO:typeOrganisationBlockId	0..1	Organization block id

10.10.3 Request: Group_GetRadioMembers

This method is used to get radio members of group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address

10.10.4 Request: Group_GetAppMembers

This method is used to get application members of the group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address

10.10.5 Request: Group_Track

This method is used to start or stop tracking of the group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address
Mask	ctG:typeGroupTrackingMask	0..1	Tracking mask
Stop	xs:boolean	0..1	Stop indicator

10.10.6 REQUEST: GROUP_ADDRADIOMEMBER

This method is used to add radio member to the group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
member	ct:typeSubscriberAddress	1..1	Member address
group	ct:typeSubscriberAddress	1..1	group address

10.10.7 REQUEST: GROUP_REMOVERADIOMEMBER

This method is used to remove a radio member from the group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
member	ct:typeSubscriberAddress	1..1	Member address
group	ct:typeSubscriberAddress	1..1	group address
membership	ctG:typeMembershipType	0..1	Membership type

10.10.8 REQUEST: GROUP_GETCOMBINATIONS

This method is used to get groups belonging to the same combined group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address

10.10.9 Request: Group_AddCombination

This method is used to add a group to the combined group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address
baseGroup	ct:typeSubscriberAddress	1..1	Base group address
Force	xs:boolean	0..1	Force indicator

10.10.10 Request: Group_RemoveCombination

This method is used to remove a group from the combined group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ct:typeSubscriberAddress	1..1	group address
baseGroup	ct:typeSubscriberAddress	1..1	Base group address

10.10.11 Request: Group_SubscribeData

This method is used to start or stop receiving of SDS and Status for the group.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
group	ctG:typeGroupSubscribeData	1..N	Group subscribe data attributes

10.11 Response and Event Interface: DR-GW-Group. Events

10.11.1 Response: Group_Response

This is a general response for the DR-GW-Group.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.11.2 Event: Group_GetEvent

This event is generated as a result of DR-GW-Client Group_Get.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ctG:typeGroup	1..1	Group attributes

10.11.3 Event: Group_GetListEvent

This event is generated as a result of DR-GW-Client Group_GetList.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ctG:typeGroup	0..N	Group attributes
listEnd	xs:boolean	0..1	End of list indicator

10.11.4 Event: Group_GetRadioMembersEvent

This event is generated as a result of DR-GW-Client Group_GetRadioMembers.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
radio	ct:typeSubscriberAddress	0..N	Radio address
listEnd	xs:boolean	0..1	End of list indicator

10.11.5 Event: Group_GetAppMembersEvent

This event is generated as a result of DR-GW-Client Group_GetAppMembers.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
app	ct:typeSubscriberAddress	0..N	Application address
listEnd	xs:boolean	0..1	End of list indicator

10.11.6 Event: Group_Event

This event is generated upon creation, modification or deletion of group.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ctG:typeGroup	1..1	Group attributes
delete	xs:boolean	0..1	Deletion indicator

10.11.7 Event: Group_RadioMemberEvent

This event is generated upon addition or deletion of a radio member from the group.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ct:typeSubscriberAddress	1..1	Group address
radio	ct:typeSubscriberAddress	1..1	Radio address
delete	xs:boolean	0..1	Deletion indicator

10.11.8 Event: Group_AppMemberEvent

This event is generated upon addition or deletion of the application member from the group.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ctG:typeGroup	1..1	Group attributes

app	ct:typeSubscriberAddress	1..1	Application address
delete	xs:boolean	0..1	Deletion indicator

10.11.9 Event: Group_GetCombinationsEvent

This event is generated as a result of DR-GW-Client Group_GetCombinations

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
baseGroup	ct:typeSubscriberAddress	0..1	Base group address
constitGroup	ct:typeSubscriberAddress	0..7	Constituent groups addresses

10.11.10 Event: Group_CombinationEvent

This event is generated upon addition or deletion of a group from the combined group.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ct:typeSubscriberAddress	1..1	Group address
baseGroup	ct:typeSubscriberAddress	1..1	Base group address
constitGroup	ct:typeSubscriberAddress	0..7	Constituent groups addresses

10.12 Request Interface: DR-GW-Application

10.12.1 Request: App_Get

This method is used to get application attributes.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
app	ct:typeSubscriberAddress	1..1	Application address

10.12.2 Request: App_GetList

This method is used to get applications and their attributes belonging to a given organization block.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
orgblockId	ctO:typeOrganisationBlockId	1..1	Organization block id

10.13 Response and Event Interface: DR-GW-Application.Events

10.13.1 Response: App_Response

This is a general response for DR-GW-Application. Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.13.2 Event: App_GetEvent

This event is generated as a result of DR-GW-Client App_Get.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
app	ctA:typeApplication	1..1	Application attributes

10.13.3 Event: App_GetListEvent

This event is generated as a result of DR-GW-Client App_GetList.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
app	ctA:typeApplication	1..1	Application attributes
listEnd	xs:boolean	0..1	End of list indicator

10.14 Event Interface: DR-GW-System.Events

10.14.1 Event: Sys_TETRAStatesEvent

This event is generated upon change of a DR-GW system state.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
tcsState	ctS:typeSystemElementState	0..1	TCS state
dxtState	ctS:typeSystemElementState	0..1	DxT state
cddconnectionState	ctS:typeSystemElementState	0..1	CDD connection state
cddserverState	ctS:typeSystemElementState	0..1	CDD server state

10.14.2 Event: Sys_LogEvent

This event is generated upon DR-GW sending additional textual information.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
value	xs:hexBinary	0..1	Value
text	xs:normalizedString	0..1	Text

10.15 Request Interface: DR-GW-Radio

10.15.1 Request: Radio_Get

This method is used to get radio attributes.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
radio	ct:typeSubscriberAddress	1..1	Radio address

10.15.2 Request: Radio_GetList

This method is used to get radios and their attributes belonging to a given organization block.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
orgblockId	ctO:typeOrganisationBlockId	1..1	Organization block id

10.15.3 Request: Radio_GetGroups

This method is used to get groups, to which a given radio belongs.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
radio	ctR:typeRadio	1..1	Radio attributes

10.15.4 Request: Radio_Track

This method is used to start or stop tracking of radio.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
radio	ct:typeSubscriberAddress	1..1	Radio address
stop	xs:boolean	0..1	Stop indicator

10.16 Response and Event Interface: DR-GW-Radio.Events

10.16.1 Response: Radio_Response

This is a general response for DR-GW-Radio.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.16.2 Event: Radio_GetEvent

This event is generated as a result of DR-GW-Client Radio_Get.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
radio	ctR:typeRadio	1..1	Radio attributes

10.16.3 Event: Radio_GetListEvent

This event is generated as a result of DR-GW-Client Radio_GetList.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
radio	ctR:typeRadio	0..N	Radio attributes
listEnd	xs:boolean	0..1	End of list indicator

10.16.4 Event: Radio_GetGroupsEvent

This event is generated as a result of DR-GW-Client Radio_GetGroups.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
group	ct:typeSubscriberAddress	0..N	Groups addresses
listEnd	xs:boolean	0..1	End of list indicator

10.16.5 Event: Radio_Event

This event is generated upon creation, modification or deletion of radio.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
radio	ctR:typeRadio	0..N	Radio attributes
delete	xs:boolean	0..1	End of list indicator

10.16.6 Event: Radio_TrackEvent

This event is generated upon change of radio tracking attributes.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
radio	ctR:typeRadio	0..N	Radio attributes
delete	xs:boolean	0..1	End of list indicator

10.17 Request Interface: DR-GW-Call

10.17.1 Request: Call_Select

This method reserves speech line for the targets of selection operation.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
sel	ctC:typeSelection	1..1	Selection attributes

10.17.2 Request: Call_Request

This method is used to accomplish all call related operations.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
tetraCallId	xs:unsignedLong	0..1	TETRA Call Id
action	ctC:typeActionRequest	1..1	Action type
attributes	ctC:typeCallAttributes	0..1	Call attributes
callingParty	ct:typeAddress	0..1	Calling party address
calledParty	ct:typeAddress	0..1	Called party address
workstationId	ctC:typeWorkstationId	0..1	Workstation Id

10.17.3 Request: Call_PTTRequest

This method is used for "DemandTx" and "CeaseTx" actions.

Input parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
tetraCallId	xs:unsignedLong	0..1	TETRA call Id
action	ctC:typePTTRequest	1..1	Action type (demandTx, ceaseTx)
attributes	ctC:typeCallAttributes	0..1	Call attributes
talkingParty	ct:typeAddress	0..1	Talking party address
workstationId	ctC:typeWorkstationId	0..1	Workstation Id

10.18 Response and event Interface: DR-GW-Call.Events

10.18.1 Response: Call_Response

This is a general response for DR-GW-Call.Events interface.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	1..1	Result code as in TCS API description

10.18.2 Event: Call_SelectEvent

This event is generated as a result of DR-GW-Client Call_Select.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
sel	ctC:typeSelection	1..1	Selection attributes

10.18.3 EVENT: CALL_EVENT

This event is generated upon all call related operations.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
result	xs:unsignedLong	0..1	Result code as in TCS API description
tetraCallId	xs:unsignedLong	0..1	TETRA call Id
action	ctC:typeActionEvent	1..1	Action type
attributes	ctC:typeCallAttributes	0..1	Call attributes
callingParty	ct:typeAddress	0..1	Calling party address
calledParty	ct:typeAddress	0..1	Called party address
keymngstate	xs:hexBinary	0..1	State of key management

10.18.4 EVENT: CALL_PTTEVENT

This event is generated upon "txGranted" and "txCeased" events.

Output parameters:

Argument	Type	Occurs	Description
requestId	xs:unsignedLong	1..1	Request identifier
Result	xs:unsignedLong	0..1	Result code as in TCS API description
tetraCallId	xs:unsignedLong	0..1	TETRA call Id
Action	ctC:typePTTEvent	1..1	Action type (txGranted, txCeased)
txGrant	ctC:typeTxGrant	0..1	TxGrant flag
txInterrupt	xs:boolean	0..1	TxInterrupt flag
talkingParty	ct:typeAddress	0..1	Talking party address
Attributes	ctC:typeCallAttributes	0..1	Call attributes
Txrepeat	xs:unsignedLong	0..1	PTT repeat timer suggested by server
workstationId	ctC:typeWorkstationId	0..1	Workstation Id