Die approbierte Originalversion dieser Dissertation ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

DISSERTATION

Event-Based Radio Signaling using the Session Initiation Protocol

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften
unter der Leitung von

EM.O.UNIV.PROF. DR.TECHN. RICHARD EIER und
UNIV.LEKTOR DR.TECHN. KARL MICHAEL GÖSCHKA
Institut für Computertechnik, E384

und

O.UNIV.PROF. DR.TECHN. HARMEN R. VAN AS
Institut für Breitbandkommunikation, E388

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik

von

KLAUS DARILION
Matrikelnummer: 9426373
1090 Wien, Währinger Gürtel 156/4

Wien, Juni 2004

Abstract

The long-term growth of air traffic requires integrated audio and data services to improve the support of air traffic controllers in maintaining safety despite increased capacity. The use of an IP (Internet Protocol) based network for voice and data communication will ease this integration but on the other hand causes new challenges for voice communication when facing the characteristics of packet based networks.

In this thesis, the requirements for voice communication systems for air traffic control (ATC) are obtained. This leads to the specification of a service environment which supports three types of services: data services, phone services, and radio services. The services are logically separated to enable disjoined development and integration of 3rd party products. As phone services based on VoIP technology and data services already exist, the focus and main contribution of this thesis is on the radio part. The requirements of the radio service lead to a client/server based architecture consisting of operator positions, radio servers and radio gateways.

As the radio communication between operators and pilots is half duplex, they have to obey the listen-before-talk procedure and request radio transmissions by pressing the push-to-talk (PTT) button. The radio service requires the transmission of the PTT signals from the operator position to the radio gateway throughout the whole system. Therefore, existing VoIP related protocols are extended to transport the PTT signals. Different solutions are compared amongst each other and the most promising, the Session Initiation Protocol (SIP) based approach, is chosen for the prototype implementation as it allows radio signaling without the need to setup a voice conversation. Furthermore, using a dedicated event package for the SIP-based event notification allows ongoing extensions while still being standard conform.

As proof of concept, an operator position with basic radio and phone capabilities, and a radio server with PTT-lockout functionality are implemented. Various delay measurements are performed to verify the implementation against the delay requirements. The concept is proven in general, but implementation details have to be improved to reach the performance requirements for large systems. For future work, the RTCP based signaling as used in the PTT-over-Celluar service should be verified whether it is capable of overcoming the performance issues of the SIP based approach. Furthermore, the presented redundancy concepts have to be evaluated against the high availability requirements.

The presented architecture is suitable for IP based ATC centers and for their interconnection. IP technology is used in the ground network only, but not for the radio link, which still uses the existing radio technology. However, if IP will only be used to interconnect existing circuit switched ATC centers, tunneling methods like time division multiplexing over IP (TDMoIP) will be a better solution.

Kurzfassung

Durch den ständig ansteigenden Flugverkehr ist es notwendig, Fluglotsen bei ihren Aufgaben besser zu unterstützen. Dies kann durch Integration von Datendiensten in bestehende Sprachkommunikationssysteme (VCS) erreicht werden. Die Einführung eines auf dem Internet Protokoll (IP) basierten Netzwerkes für Sprach- und Datendienste erleichtert diese Integration. Durch die Verwendung von paketvermittelnder Technologie ergeben sich jedoch auch neue Herausforderungen für die Sprachdienste.

In dieser Dissertation werden die Anforderungen an ein VCS analysiert und drei Dienste identifiziert: Telefonie, Funk und Daten. Diese werden in weiterer Folge als voneinander unabhängige Dienste betrachtet. Dadurch ist es möglich, diese Dienste von einander getrennt zu entwickeln und Produkte anderer Hersteller zu integrieren. Diese Arbeit widmet sich hauptsächlich dem Funkdienst. Dabei ergibt sich durch Analyse der Randbedingungen eine Client/Server-basierte Architektur mit folgenden Komponenten: Lotsenarbeitsplatz, Funk-Server und Funk-Gateway.

Da der Funkkanal half-duplex ist, muss der Fluglotse den „Push-To-Talk"-Knopf (PTT) betätigen um einen Funkspruch abzusetzen. Dieses PTT-Signal muss vom Arbeitsplatz bis zum Gateway übertragen werden. Dazu werden die bestehenden VoIP-Protokolle erweitert, um die PTT-Information zu übertragen. Die verschiedenen Lösungen werden untereinander verglichen und der SIP-basierte Ansatz zur Implementierung ausgewählt. Dieser ermöglicht PTT-Signalisierung ohne Aufbau eines Sprachkanals, und zusätzliche Erweiterungen innerhalb des Standards.

Es wird ein Arbeitsplatz mit elementarer Telefonie- und Funkfunktionalität, sowie ein Funk-Server, welcher den gleichzeitigen Zugriff mehrer Lotsen auf einen Kanal verwaltet, implementiert. Die Implementierung zeigt, dass sich SIP gut eignet um PTT- und dazugehörige zusätzliche Information zu übertragen. Messungen der Signalisierungsverzögerungen zeigen jedoch, dass innerhalb der Grenzwerte nur eine geringe Anzahl von Arbeitsplätzen über Kanalaktivitäten benachrichtig werden kann. Die Limitierung entsteht hierbei durch den SIP-Stack, welcher die SIP-Transaktion zu langsam generiert. Als weiterführender Schritt sollte überprüft werden, ob mit dem RTCP-basierten Ansatz die Signalisierungsverzögerungen verkleinert werden können. Des Weiteren müssen die vorgestellten Redundanzkonzepte implementiert und bzgl. der Verfügbarkeitsanforderungen überprüft werden.

Das vorgestellte System eignet sich für rein IP-basierte Flugsicherungslösungen sowie für deren Vernetzung. Falls jedoch IP-Netze nur verwendet werden sollen, um bestehende Flugsicherungssysteme zu vernetzen, dann sollten „Tunneling"-Ansätze wie „Time Division Multiplexing over IP" (TDMoIP) verwendet werden. Des Weiteren führt das vorgestellte System keine Änderungen an der Funkübertragung durch - es wird weiterhin amplitudenmodulierter, analoger Funk verwendet.

Danksagung

Großer Dank gebührt Prof. Richard Eier für die Betreuung meiner Dissertation sowie für die zahlreichen Diskussionen, die mir immer wieder neue Aspekte meiner Arbeit aufzeigten.

Großer Dank gebührt auch Prof. Harmen van As für die Begutachtung und unkomplizierte Abwicklung der Dissertation.

Ganz besonders danke ich Karl Michael Göschka, der mir diese Dissertation ermöglicht hat. Er vermittelte mir den Forschungsposten am Institut für Computertechnik, sorgte für meine Finanzierung und gab mir immer den notwendigen Druck, ohne den meine Dissertation noch nicht fertig wäre. Besonders möchte ich mich auch für die immer konstruktive Kritik zu meiner Arbeit und den Publikationen bedanken.

Wolfgang Kampichler danke ich für die kritischen und konstruktiven Beiträge zu meinen Arbeiten und für die Bewirtung bei meinen Frequentis-Besuchen (Pizza und Mozartkugeln). Weiters danke ich Wolfgang, Gio und Frequentis für die Finanzierung meiner Arbeit und für die Freiheit meiner Forschung.

Christoph Kurth danke ich für die kritische Begutachtung meiner Ideen, für die zahlreichen Diskussionen und für seine Liebe zu 80er Musik.

Meinen Arbeitskollegen Alex, Edin, Fritz, Jürgen, Martin, Michi, Paul, Robert, Roman und Wolfgang danke ich für die heißen Diskussionen und belanglosen Gespräche über meine Arbeit und alles andere.

Meinem Bruder, der immer dann anruft, wenn man keine Zeit zum Telefonieren hat und selber keine Zeit zum Telefonieren hat, wenn ich ihn anrufe, danke ich ebenfalls. ;-)

Meiner Familie danke ich dafür, dass sie mich immer unterstützt hat und regelmäßig Kontakt zu mir hielt, da ich teilweise vor Forscherdrang auf die Außenwelt vergaß.

Großer Dank gebührt auch meinen Freunden. Düdl danke ich für den Espresso und für den Lärm am Morgen, ohne den ich nicht so bald aufstehen würde und somit weniger Zeit für meine Arbeit hätte. Heike danke ich für das Begutachten dieser Arbeit. Allen meinen anderen Freunden danke ich dafür, dass sie immer so spät fortgehen und ich mich somit direkt von der Arbeit ins Vergnügen stürzen konnte.

Contents

1 Air Traffic Control and Voice over IP 1

1.1 Air Traffic Control 1

1.2 Voice Communication Systems 5

1.3 Voice over IP 6

1.3.1 Session Signaling using the Session Initiation Protocol 7

1.3.2 Media Transport using the Real-time Transport Protocol 9

2 System Architecture 10

2.1 Motivation 10

2.2 Requirements 11

2.3 Use Case Modeling 12

2.4 Radio Service Model 14

2.5 Service Environment 14

2.6 Radio Architecture 16

2.6.1 Operator Position 19

2.6.2 Radio Gateway 21

2.6.3 Radio Server 22

2.7 Remote Sidetone Problem 23

3 Radio Signaling 26

3.1 Event Based Signaling versus Continuous Signaling 27

3.2 SIP-based Radio Signaling 30

3.2.1 SIP-Specific Event Notification 30

3.2.2 PTT/SQU Event Package 31
3.3 RTP/RTCP-based Radio Signaling 37
3.3.1 Implicit RTP Radio Signaling 37
3.3.2 Explicit RTP Radio Signaling 38
3.3.3 Explicit RTCP Radio Signaling 40
3.4 Proprietary Signaling 42
3.5 Signaling Summary 43
3.6 Putting It Together: The Radio Service 44

4 Radio Server Implementation and Verification 49
4.1 Addressing Schema 49
4.1.1 SIP-URI 50
4.1.2 TEL-URI 50
4.1.3 ENUM 50
4.1.4 Numbering Plan for ATC centers 51
4.2 Dissipate SIP stack 53
4.3 Radio Server Implementation 55
4.4 Radio Gateway Implementation 57
4.5 Operator Position Implementation 59
4.6 Signaling Delay Measurements 61
4.6.1 PTT Signaling: Results and Analysis 63
4.6.2 SQU Signaling: Results and Analysis 65
4.6.3 RTP Forwarding: Results and Analysis 66
4.6.4 SIP proxy forking performance 67
4.6.5 Discussion of the results 67

5 A Dependable Radio Service 69
5.1 Dependable SIP Proxies 71
5.2 Dependable Operator Position 72
5.3 Dependable Radio Gateways 73
5.4 Dependable Radio Servers 75
5.5 Failure Detection 79
5.5.1 Failure Types 79
5.5.2 Server Monitoring 80

6 Summary and Conclusion **85**

6.1 Summary 85

6.2 Related Work 86

6.2.1 AudioLAN 86

6.2.2 Internet Radio Linking Project 87

6.2.3 Push-to-Talk over Cellular 89

6.2.4 RTCP extension 93

6.2.5 Proprietary Solutions 93

6.3 Future Work 94

6.4 Conclusion 95

Abbreviations

3GPP	3rd Generation Partnership Project
AOR	Address-Of-Record
API	Application Programming Interface
ARP	Address Resolution Protocol
ATA	Air Traffic Authority
ATC	Air Traffic Control
ATM	Air Traffic Management
B2BUA	Back-to-Back User Agent
CGI	Common Gateway Interface
COS	Carrier Operated Squelch
COTS	Commercial Off The Shelf
CPL	Call Processing Language
CSRC	Contributing Source Identifier
CTA	Control Area
CTCSS	Continuous Tone Coded Subaudible Squelch Signals
CTR	Controller or Control Zone
DA	Direct Access
DAI	Direct Access Intercom
DNS	Domain Name System
DTMF	Dual-Tone Multi Frequency
E.164	International Public Telecommunication Telephony Numbering Plan.
ENUM	E.164 Number Mapping
GCP	Gateway Control Protocol
GLMS	Group and List Management Server

GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
GW	Gateway
HMI	Human Machine Interface
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IAX	Inter-Asterisk Exchange
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IRLP	Internet Radio Linking Project
ITU	International Telecommunication Union
ITU-T	ITU Telecommunication Standardization Sector
LAN	Local Area Network
LNZ	Linz Airport
MAC	Media Access Control
MGCP	Media Gateway Control Protocol
NAPTR	Naming Authority Pointer
NDB	Non-Directional Beacon
OP	Operator Position
PBX	Private Branch Exchange
PC	Personal Computer
PoC	Push-to-talk over Cellular
PTT	Push-to-Talk
RGW	Radio Gateway
RS	Radio Server
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
Rx	Receive

RxTx	Receive and Transmit
SDP	Session Description Protocol
SIMPLE	SIP for Instant Messaging and Presence Leveraging Extensions
SIP	Session Initiation Protocol
SLP	Service Location Protocol
SOAP	Simple Object Access Protocol
SRA	Surveillance Radar Approach
SRV	Server Selection records in the Domain Name System
SSRC	Synchronization Source Identifier
TCP	Transmission Control Protocol
TDM	Time Division Multiplex
TMA	Terminal Manoeuvring Area
Tx	Transmit
UA	User Agent
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
UE	User Equipment
UML	Unified Modeling Language
URI	Uniform Resource Identifier
VCS	Voice Communication System
VoIP	Voice over IP
VOR	VHF Omni-directional Radio Range
VOX	Voice Operated Transmission
WAN	Wide Area Network
WPos	Working Position
XML	eXtensible Markup Language

Chapter 1

Air Traffic Control and Voice over IP

In air traffic control (ATC) and public safety is a trend to integrate data services into the voice communication systems. To use a single network for voice and data communication, voice over Internet protocol (VoIP) will be used. In order to better understand the requirements and boundary conditions, this chapter first provides an overview about ATC and VoIP. Air traffic control will be used representative for all applications, which require radio communication like public safety and public transports. The following section explains the air traffic control domain and tasks the air traffic controllers have to perform. Furthermore, the features of voice communication systems (VCSs) required to support operators in their tasks and current implementations of voice communication systems will be explained. The last section explains the difference between session signaling and media transport, presents the related protocols and states why the session initiation protocol was preferred over other signaling protocols.

1.1 Air Traffic Control

> *"Air traffic control has the task of monitoring the environment of every aircraft in controlled airspace. The air traffic controllers give the pilots instructions so that they can preserve the internationally prescribed minimum separations between aircraft in the airspace.*
>
> *From the moment that the doors are closed until the aircraft arrives at the gate at the destination airport, the pilots require to secure authorisation from air traffic control for every manoeuvre, every climb or descent by the aircraft, and every alteration to the flight route.*

Because the air traffic controllers always know the air situation and know which planes are flying where and when, they can instruct the pilots as to how they must slot themselves into this uninterrupted flow of traffic. In this way, air traffic control increases the efficiency of aviation. The instructions from air traffic control are binding upon all pilots in the controlled airspace and failure to comply with them is strictly penalised.

Another important function of air traffic control is the supporting of pilots in flight planning. Via a global network, the ATC's 'Aeronautical Information Service' brings together all the information necessary for the planning of flights. Accordingly, the pilots, when they come to file their flight plan with the air traffic controllers, can pick up information from this unit on the weather conditions, the traffic flows and the runway and airport conditions relevant to their own flight.

Air traffic control also has an important role to play in compliance with noise provisions, when issuing pilots with the approach and start routes established." [Sky]

In air traffic control, many different terms are used which have the same meaning. Therefore, this section will start with a short explanation of the terms. The alias terms in table 1.1 are identical to their counterparts and will be used alternately.

term	**alias terms**
air traffic control (ATC)	air traffic management (ATM)
air traffic operator	air traffic manager air traffic controller flight operator, operator flight controller, controller

Table 1.1: Air traffic control terms

Usually, every state has an air traffic authority[1] (ATA) which monitors and instructs aircrafts within their sovereign territory. The borders of the airspace an ATA is responsibly for, are typically identical to the borders of the state. To ease air traffic control, the airspace is divided into sectors [Nol94][DKGK04]. The size of the sectors depends on the amount of traffic in the certain sectors and on geographical properties. The number of aircrafts an operator can manage concurrently is limited, therefore sectors with heavy traffic are smaller than sectors with low traffic. To form sectors, the airspace is divided in a 3-dimensional way. Figure 1.1 shows for example the sectors above the airport Linz

[1]Other terms are: air traffic service provider, air traffic services authority, civil aviation authority, civil aviation administration, air navigation services department, aviation authority et al.

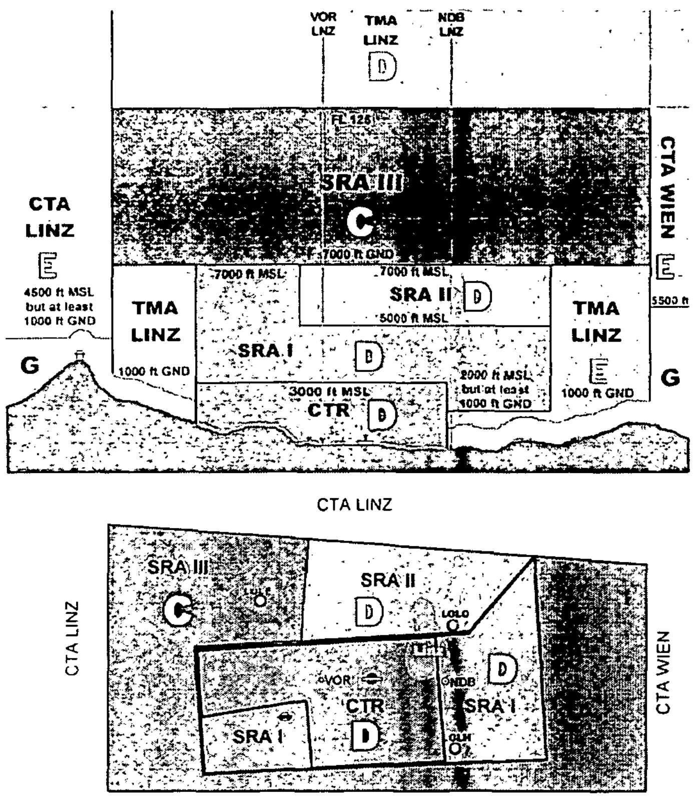


Figure 1.1: Airspace sectors in the area of airport Linz [Aus]

(Austria). Also, the properties of the landscape may affect the size of the sectors, e.g. mountains may require additional radio stations to achieve complete coverage which may affect the partitioning of the airspace.

Each of these sectors is usually managed by one operator. The communication between operators and pilots is based on amplitude modulated radio communication [Hay94]. To differentiate the several sectors, every sector uses its own frequency. The operator monitors all aircrafts in his sector by watching the aircrafts on the radar screen and by listening for incoming radio messages on the frequency allocated to this sector. The operator and the pilots in the corresponding sector will talk and listen on the same frequency, so this communication is only half-duplex. If two or more persons talk at the same time, the radio messages will interfere—called *collision*. On collisions, typically the original radio messages are lost and the talkers have to repeat their radio messages. Therefore, operators and pilots have to obey the ***listen before talk*** procedure to verify that the radio channel is free before they transmit a radio message. To transmit a radio message, the operator has to press the push-to-talk (PTT) button. This activates the radio transmitter and is necessary as the transmitter may only be active during sending of the radio message or it will interfere with all the other radio transmitters operating at the same frequency.

During the transit through a sector, the pilot is instructed by the operator regarding speed, course and altitude of the aircraft. When an aircraft leaves a sector, the operator hands the aircraft over to the new sector. The operator instructs the pilot about the frequency of the next sector and then the pilot signs off from the current sector, changes the frequency, and logs on to the new sector by contacting the operator of the new sector.

In the above scenario, the operator listens and talks to the pilots. Sometimes, an operator uses a sector (frequency) in a *receive only* mode. This is useful in several situations, for example an operator may configure the voice communication system to receive also radio messages from neighboring sectors to get a more detailed view onto the traffic situation. Additionally this will be used to train new operators by listening to real voice traffic.

Although radio communication is the most important part of air traffic control, operators also need other communication services to fulfill their tasks. Phone communication is necessary for operators to communicate among themselves. To initiate a phone communication, the operator has to dial the phone number of the other participant or he uses a dedicated *direct access* key, which is associated with the phone number of the other participant. The phone service usually supports enhanced features like call forwarding, call transfer, group calls, conferences and priority calls. Priority calls have a higher priority than normal phone calls and therefore may disrupt existing phone calls in case of limited phone resources. Another important service is *intercom*. This service allows an operator to talk directly to another operator without waiting for the called operator to accept the call. This is useful in emergency situations to avoid loss of time due to call acceptance delay. Furthermore, an intercom call will not be queued and will be established also, if the called operator is busy.

Another important instrument to maintain air traffic control is the radar service. At the working position, the operator uses a radar screen, which displays all flying objects within the sector controlled by the operator and in neighboring sectors. Although the radar service is essential for trouble-free air traffic management, it is not as important as the radio communication service—if the radar service fails, the operator may still be able to instruct the pilots, but if the radio communication fails, the operator may see upcoming accidents on the radar screen, but is not able to avoid them.

Recapitulating, the voice communication service and especially the radio communication service is the most important service for operators to control aircrafts. The following section discusses the current implementations of voice communication systems for ATC and several important technical details.

1.2 Voice Communication Systems

Today's installed and available VCSs for ATC rely on circuit switched technologies and are fully digital [Mar01]. As one of their tasks is to ensure interoperability with the local existing voice communication networks like the public switched telephony network (PSTN), dedicated voice communication networks for ATC, and legacy voice infrastructure, the VCSs can be equipped with interface cards to the respective networks. Generally, voice communication is divided into phone and radio communication.

Although the phone part of the VCS equals a private branch exchange (PBX), the end devices differ. Typically, standard phones with a dial pad are connected to the PBX, whereas in ATC, the end device is a touch pad display with a graphical user interface (GUI). The GUI presents speed dial buttons, a dial pad, call queues and access to data services like phone books. The speed dial buttons can be differentiated into direct access (DA) and direct access intercom (DAI) buttons. By pressing the DA button a call will be established to the respective party whereas DAI keys establish an intercom session. Nevertheless, VCSs also support standard phones as end devices.

The radio part deals with the radio communication between operators and the pilots. The VCS has interfaces to radio transmitters and receivers and supports several PTT/SQU[2] signaling techniques like out-band signaling with radio control lines, in-band signaling or VOX-PTT[3]. The radio is a shared resource and only one operator at a time can transmit a radio message. Therefore, the VCS manages the exclusive access to the radio sector. Although the radio communication itself is only half-duplex, the radio part of the VCS is full-duplex. This enables the operator to receive his own radio transmission (remote sidetone) and implicitly acknowledges the radio transmission (refer to section 2.7 for the remote sidetone problem in packet switched VCSs).

The VCS also offers services which combine both parts, e.g. phone calls can be interconnected with radio sectors to enable the participant of the phone call to listen and talk on the radio sector.

International flights require hand over of airplanes between sectors that belong to different authorities. This often requires communication between operators which work for different ATAs. Therefore, VCSs support interconnection amongst them via PSTN or dedicated ATC voice communication networks.

Another important part of VCSs is configuration and management, including not only configuration of the voice switch and its interfaces, but also the configuration of the GUI at the operator position. This will be combined with a "role and mission" concept: the administrator selects missions for the VCS and roles will be assigned to the operators.

[2]SQU means "squelch" and signals incoming radio messages.

[3]Using the VOX-PTT, there is no explicit PTT/SQU signaling, but the signals are derived from the audio signal. If the signal level is above a certain threshold, the VCS generates the radio signals itself.

The "mission" can be seen as the configuration of the VCS and the "role" can be seen as the configuration of the operator position within this mission. Roles enable the operator to immediately access all resources needed to execute the respective role.

Finally, ATC environments need a recording of all radio messages. Thus, the VCS offers interfaces to connect voice recorders directly to the switch and to the operator position.

1.3 Voice over IP

"Voice over IP" is a technology to transport audio data over packet switched networks using the Internet Protocol (IP) [Pos81] as network protocol. In circuit switched networks, the network's main purpose is to deliver audio signals—formerly analog, nowadays mostly digital. In packet switched networks, the audio signal is just another form of data like emails or homepages in the World Wide Web. Figure 1.2 shows the typical VoIP infrastructure. A host, which offers VoIP services—this can be a standard PC or a dedicated IP phone—needs an audio interface and a network interface. The VoIP application consists of a user interface, an audio processing part, and a signaling part. The signaling part handles the call signaling like call setup and teardown. The audio part records the callers voice, fragments it into small packets and sends packet after packet over the network to the other call participant. At the other party, the audio processing part decodes the received audio packets and sends them to the audio interface for play-back. Typically, a voice packet contains 10–60 ms of voice.

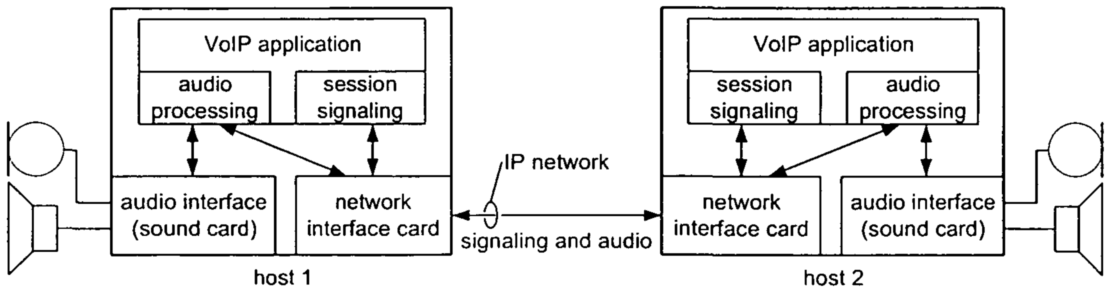


Figure 1.2: A typical voice over IP architecture

Although the signaling data will be sent over the same network as the audio data, it uses a different protocol. For audio signaling, the real-time transport protocol is the de facto standard, whereas there are several competing signaling protocols:

H.323: H.323 [H32] is an umbrella recommendation and was defined by the telecommunication standardization sector of the International Telecommunication Union (ITU-T). It is similar to the ISDN signaling and uses binary message encoding.

Megaco, GCP, MGCP: The gateway control protocol (GCP) [GPA+03] is based on the Megaco protocol [CGR+00]. Like the media gateway control protocol (MGCP) [AF03], it is intended for the communication between a media gateway controller and multimedia gateways. These protocols are used only in gateway applications.

SIP: The session initiation protocol (SIP) [RSC+02] is standardized by the Internet Engineering Task Force (IETF). The protocol is text based and its syntax is similar to the syntax of the hypertext transfer protocol (HTTP) [FGM+99].

Skinny: Skinny is a proprietary signaling protocol from Cisco Systems which is used in their IP Telephony systems.

IAX: The inter-asterisk exchange protocol (IAX) is part of the Asterisk software [AST], an open source IP based PBX. IAX integrates media transport and signaling into a single stream.

As SIP is developed by the IETF it is an open standard and several open source SIP implementations are available. Furthermore, SIP already supports the required phone services including call transfers, 3rd party call control, priority signaling and intercom. SIP also offers instant messaging and presence services, which will be interesting for future ATC applications. Furthermore, SIP will be the signaling protocol in the 3rd Generation Partnership Project (3GPP) [3GP] and therefore undergoes further development. These are the reasons why SIP was chosen over other signaling protocols.

The following sections will give a very short overview about SIP and the real-time transport protocol (RTP).

1.3.1 Session Signaling using the Session Initiation Protocol

SIP is a text based application layer protocol whose syntax and behavior is similar to the HTTP: Like HTTP, SIP uses a request/response transaction model. Every SIP user agent (UA) consists of a UA client (UAC) and a UA server (UAS). The UAC generates requests and processes their responses whereas the UAS processes requests and generates responses. The most important requests are[4]:

`REGISTER` A UA registers its current location at the registrar service.

`INVITE` An invitation for a multimedia session (e.g. phone call).

`ACK` Acknowledges an accepted invitation to a multimedia session.

`BYE` Quits an existing multimedia session.

[4]For a full reference of the SIP requests refer to RFC 3261 [RSC+02] or [SJ01].

Typically, when a SIP UA is activated, it registers is current location with its address-of-record at the appropriate SIP proxy, as shown in figure 1.3. The current location is a SIP-URI[5] which points to the socket (IP address and port) the UA is bind to. The address-of-record is the public SIP address of the user. In the shown example, the address-of-record is `sip:bob@biloxy.com`. Through the registration, this address will be linked to the current location of Bob, in this case `sip:bob@192.0.2.4`. From now on, every request for `sip:bob@biloxy.com` will be resolved by biloxy's SIP proxy to `sip:bob@192.0.2.4` as shown in figure 1.4.

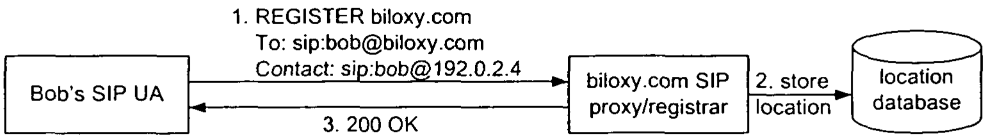


Figure 1.3: The SIP UA registers at the SIP proxy

The `INVITE` request includes a session description based on the session description protocol (SDP) [HJ98], e.g. for a phone session the session description will include the IP address and the port where the caller wants to receive the RTP stream and the acceptable codecs[6].

A typical SIP environment consists of SIP UAs and SIP proxies. The SIP proxy usually includes the registrar and the location service. Figure 1.4 shows the *SIP trapezoid*—two user agents, which belong to different domains, establish a multimedia session. Every domain is served by a SIP proxy, which also accepts registrations from local UAs. The atlanta.com proxy is the outgoing proxy for Alice's UA, therefore, outgoing requests from Alice will be sent to this proxy first. The atlanta.com proxy is not responsible for requests to biloxy.com users and therefore forwards the request to the biloxy.com proxy. This proxy queries to location service for the user Bob, replaces the request-URI with the current location of Bob and forwards the request. Responses to SIP requests always use the same route as the requests, back to the UAC. The accepted invitation will be confirmed using the `ACK` method. Now, the session is established and the UAs send their voice packets using RTP directly to each other.

Once the session is established, further SIP messages within this session—so called *in-dialog* requests and responses—are usually exchanged directly between the UAs[7].

[5]Uniform Resource Identifier: For a detailed description of the SIP addressing schema refer to section 4.1.

[6]The codec is the compression algorithm used to compress the media data at the sender and to decompress the RTP payload at the receiver.

[7]A proxy may force all requests through the proxy by using the *record-route* feature [RSC+02].

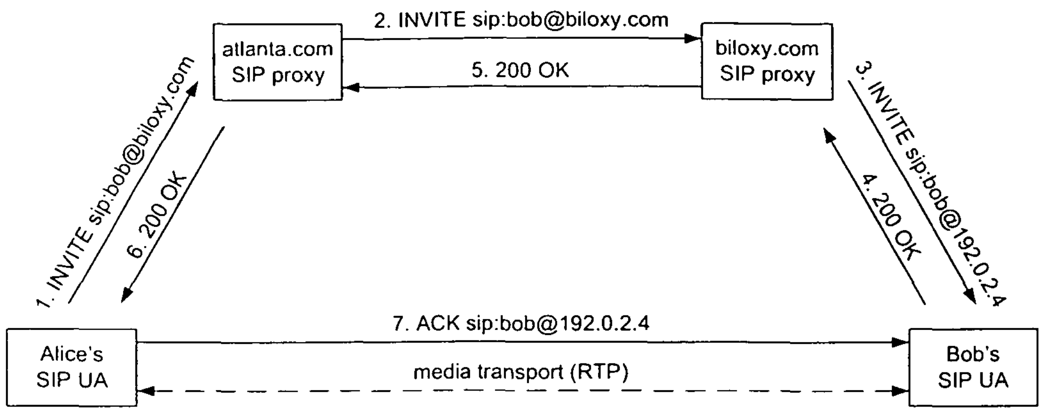


Figure 1.4: The SIP trapezoid—a typical call setup

1.3.2 Media Transport using the Real-time Transport Protocol

The RTP [SCFJ03] is used to transport the audio data between the call participants. By means of the offer/answer mechanism within the SDP [RS02a] during the `INVITE` transaction, both VoIP applications know the IP address and port where they should send their RTP packets to. The audio processing module (figure 1.2) processes the recorded audio samples, compresses them using a certain codec, adds the RTP header (see section 3.3.2) and sends the packet to the other participant. The RTP header contains the payload type, a sequence number, a timestamp, and a source identifier. The receiver uses the payload type to choose the proper codec for payload de-compression, the sequence number for packet loss detection, and the timestamp to synchronize the playback of several incoming RTP streams. By means of the source identifier, the receiver can differentiate incoming RTP streams, if it receives several streams on the same port.

Related with RTP is the RTP control protocol (RTCP) [SCFJ03], which can be used to exchange statistical information about the RTP streams. A more detailed explanation of RTP/RTCP can be found in section 3.3.

Chapter 2

System Architecture

2.1 Motivation

The currently deployed and available voice communication systems (VCS) for air traffic control and public safety are built on circuit switched technologies. They are highly reliable but suffer from the following limitations:

- The time division multiplex (TDM) technology is used in the core switch of current systems. The number of time slots in the switch is limited. Thus, in a non-blocking environment, the number of connected operator working positions and voice communication interfaces is also limited.
- The current systems have a very poor integration with data services in the field of ATC e.g. radar systems, weather information systems, or tactical mission planning systems.
- The systems are built upon proprietary software running on a proprietary operating system on proprietary hardware. Therefore, the system is expensive to maintain, expansions of the installed systems are very complicated, and commercial off the shelf (COTS) components can not be integrated easily.
- Current systems are not designed for wide area integration of ATC centers. This leads to problems, if several ATC centers should be interconnected.
- Current systems lack of scalability. Even small ATC centers with only a few working positions use the high-end core switch, which will be utilized poorly.

The aim of this research work is to develop a new service oriented system architecture based on an IP network infrastructure, which is scalable, modular and does not have the limitations of current circuit switched voice communication systems, but still fulfills the requirements for reliability and availability. Another objective is to enable the easy integration of COTS components. This will reduce hardware costs and development time, and makes the system more open for future enhancements. Thus, both, the consumer and the VCS supplier, will be independent from the hardware manufacturer.

After presentation of the requirements, the results of the use case modeling followed by an in-depth discussion of the whole service environment and the radio service are shown.

2.2 Requirements

The architecture shall avoid the limitations of current systems and shall be prepared for future trends. This leads to a number of additional requirements for ATC systems:

- The architecture shall be modular and flexible resulting in a scalable, distributable, and reliable environment for voice and data communication services[1]. There shall not be any architectural limitation concerning the number of connected working positions or offered services. There must be a mechanism to promote new or added services. The working positions must be able to retrieve a list of all available services and to use them immediately.
- The architecture shall enable every part of the system to transmit data to and receive data from data services (e.g. weather data and flight plan data).
- The architecture shall separate phone, radio, and data services (server). This enables the manufacturer to use 3rd party products in the field of phone and data services independent from the development of the radio services. This will also enhance reliability, because a failure of e.g. the phone service does not impair the radio or data services.
- The radios[2] shall be connectible to the system via low bandwidth connections. This allows the use of remote radio sites.
- COTS components shall be integrated except in areas where they cannot fulfill the strong requirements of ATC for reliability and timing. Standard PCs with standard interface cards and standard operating systems shall be used as working positions. Also, open and established standard protocols shall be used.

[1]A service is a collection of functionality which is offered to the system by service providers (e.g. a server offering this service) and consumed by customers (e.g. the working position).

[2]In this context the term *radio* means a radio receiver, a transmitter or a transceiver.

- ATC centers at different locations shall be interconnectable by means of WAN technologies.
- The new architecture shall be modular and scalable to be suitable for ATC centers consisting of a few up to hundreds working positions.

As ATC and public safety are mission critical domains, high availability up to 0.99999 is required for the communication system.

The design of the underlying network is out of the scope of this work. The local network can be seen as managed LAN with high bandwidth and therefore can be designed to fulfill the requirements of voice communication systems. This is a legitimate assumption, because e.g. Fast Ethernet with prioritization by means of IEEE 802.1p [I80] or similar technologies are available at low cost and fulfill the throughput and delay requirements for voice and data communications, when network design rules are applied accordingly [Cisa]. Nevertheless, the applications must be designed to request prioritized traffic classes for the audio transmission (RTP packets) and the signaling (SIP, DNS). We also consider the network as highly available. Nevertheless, the applications shall be prepared for network outages in the dimension of several seconds.

2.3 Use Case Modeling

During the development process, software engineering methods as described in [JCJO92] are applied. Therefore, the first step is to analyze the problem domain using the use case driven approach. The use case modeling is defined as part of the Unified Modeling Language (UML) [Obj]. Modeling the use cases is an important step to become familiar with the problem domain and to understand the needs of a communication system for air traffic control. Later, the use case model is used to find the basic needs of ATC systems for the development of a prototype.

For clearness, similar use cases were grouped together into packages. The *Radio Use Cases* package includes all use cases the operator needs to communicate with the pilots and the other operators working in the same sectors, and to get the status information for the sectors he is working on. Furthermore, we found packages for the *Phone Use Cases*, the *Data Services Use Cases*, the *Configuration Use Cases*, and the *Remote Use Cases*.

The use case modeling leads us to the following important insights:

- A seamless integration of data services is necessary.

- Interconnected systems should appear as one single system to the users, so the usage of remote resources[3] must be transparent to them. This also leads to the demand of a security policy for services and access control for users within their local system and in the global view of interconnected systems.
- All current and upcoming features of connected nodes, e.g. the monitoring and configuration of radios, must be supported.
- The system must cooperate with external resources but must not rely on them to ensure high availability.
- The transmission and reception of radio messages is the most important service of an ATC system. Hence, even in partly degraded systems these use cases must be available as long as no total system failure occurs.
- Complex services like the coupling of frequencies[4], coverage, and PTT lockout[5] should be easily configured and set up.

Representative for all use cases, figure 2.1 shows the use case diagram of the primary use case scenario—the transmission of radio messages and the monitoring of the radios including the reception of radio messages. The following architecture has been designed towards the results of the use case analysis.

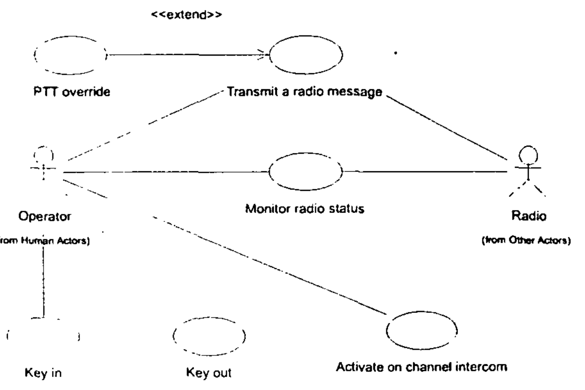


Figure 2.1: *Transmit a Radio Message* Use Case

[3]A remote resource (e.g. a radio transmitter) is located at another ATC center, but interconnected with the local system.

[4]During times of less air traffic, sectors (with different frequencies of their radios) will be coupled. So the ATC system reacts as a repeater and relays radio messages received in one sector to the other coupled sectors.

[5]If more than one operator requests sector access at the same time, all except the first one will be locked out.

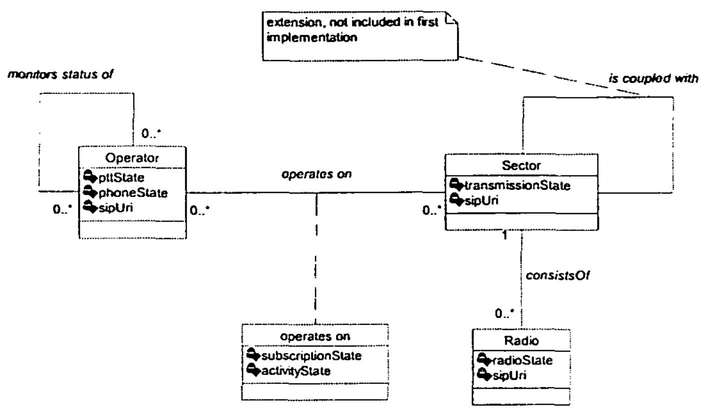


Figure 2.2: System analysis model of the radio system

2.4 Radio Service Model

Analyzing the ATC domain and creating a static view on the system leads to the class diagram shown in figure 2.2. A radio sector consists of several radios. Operators operate on these sectors. The current status of the operation can be represented by the kind of the subscription (subscriptionState) and the activity (activityState) within this subscription. Furthermore, a sector may be coupled with other sectors.

As shown in the following sections, the system will be composed of several types of applications running on several nodes. Each of these applications has a specific view onto the system, and therefore, the models for the respective applications will differ from the previously described domain model.

2.5 Service Environment

Figure 2.3 shows the IP based ATC service environment. Every host in the network can offer and consume services. For example, a radio service offers access to radios, whereas the gateway service offers connectivity to legacy systems or to the PSTN. All services can be consumed by every host in the ATC network, regardless if the consumer is located in the same LAN or connected via WANs, as long as it is not prohibited by domain policies. Even participants outside the ATC network (e.g. in the PSTN) may consume services using the gateways. The separation of the offered services allows to add new services to the system without conflicting with existing services.

Every host that wants to consume a service has to implement a client for the respective service. In an ATC system, the main consumer of the offered services is the operator position – the working place of the operator. Therefore, the working position must implement clients for all the needed services as shown in figure 2.4. The operator position includes an additional convergence layer to combine the miscellaneous service clients to high level services, for example the weather forecast will be synthesized to speech and transmitted using the radio service. The service APIs ease the integration with other systems in ATC, e.g. the radar system can use the ATC radio service to initiate radio transmissions. Currently, the radio communication between aircrafts and ground stations supports only voice, but data links are already discussed within the ATC community [DoT]. These data links for example enable aircrafts to send their identity within radio messages. This identification can be utilized by the radar application to highlight the aircraft of the currently talking pilot at the radar screen, and will ease the work of the operator. Additional data services will use the simple object access protocol (SOAP) [W3C] or similar protocols to access the data resources.

This work has its focus on the voice communication services for ATC, especially radio services for air/ground communication. The voice services in this architecture are already combined services as shown in figure 2.5. For standard phone calls, SIP is used for session signaling and RTP/RTCP is used for the voice transmission. In ATC, the operator requires a more sophisticated phone service, where he could see the status of other operators before calling them. To enable such a service, the standard phone call will be combined with the SIP presence service[6].

The radio service is also close to the standard phone service, except that it requires means to signal radio events (PTT and SQU) and possible additional data. Figure 2.5 already

[6]The use of SIP for presence is currently under development by the SIMPLE IETF working group.

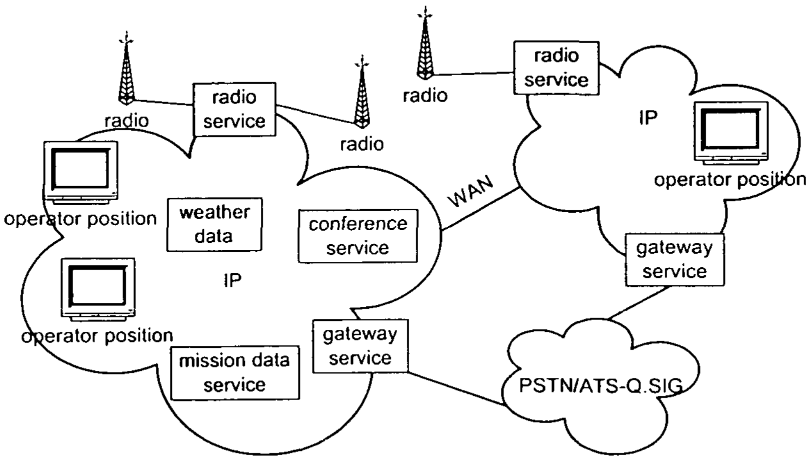


Figure 2.3: Service environment for air traffic control

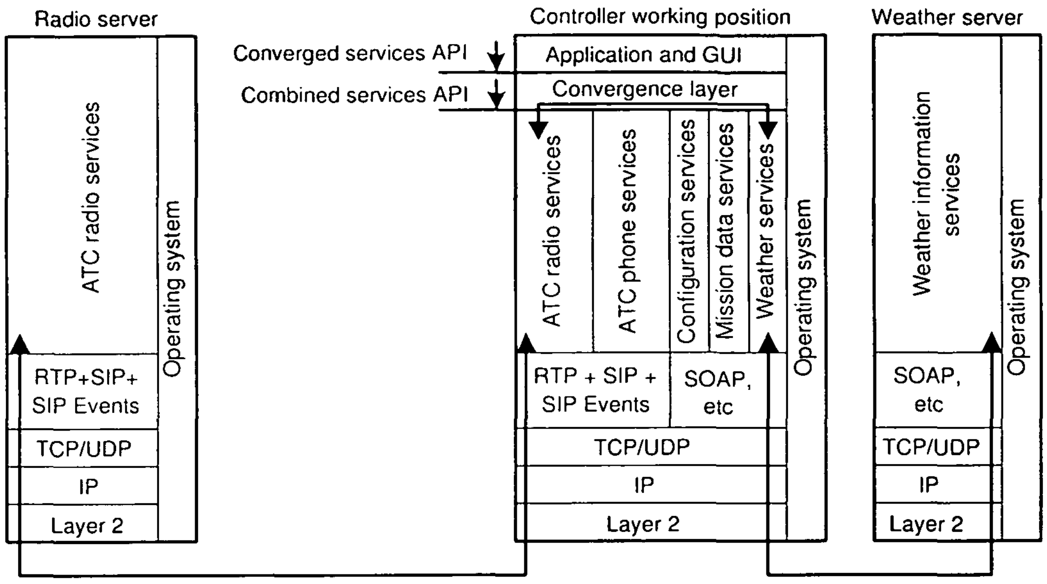


Figure 2.4: Layered service integration

shows the proposed solution—the SIP specific event notification with a PTT/SQU event package—which is the result of the analysis in chapter 3.

All SIP based communications will be set up using SIP elements like registrars and proxies and the media will be sent directly between the two participants.

2.6 Radio Architecture

As described in section 1.1, the airspace is divided into sectors. Every sector has a certain frequency, on which the radio transmitters and receivers operate, and one air traffic controller who manages the traffic in this sector. The introduction of an IP based ground communication network should not require to change the existing radio equipment. Therefore, a radio gateway is necessary which translates the SIP signaling to radio control lines and the audio packets to analog audio and vice versa.

The radio resources like senders, receivers, and their antennas are typically on remote sites. As the radios only support one voice channel, the bandwidth of the connection between the remote site and the ATC center is typically dimensioned for one voice stream per radio. To allow several operators to listen to the same sector concurrently, the voice stream coming from the radio has to be forwarded to all operator positions. This leads

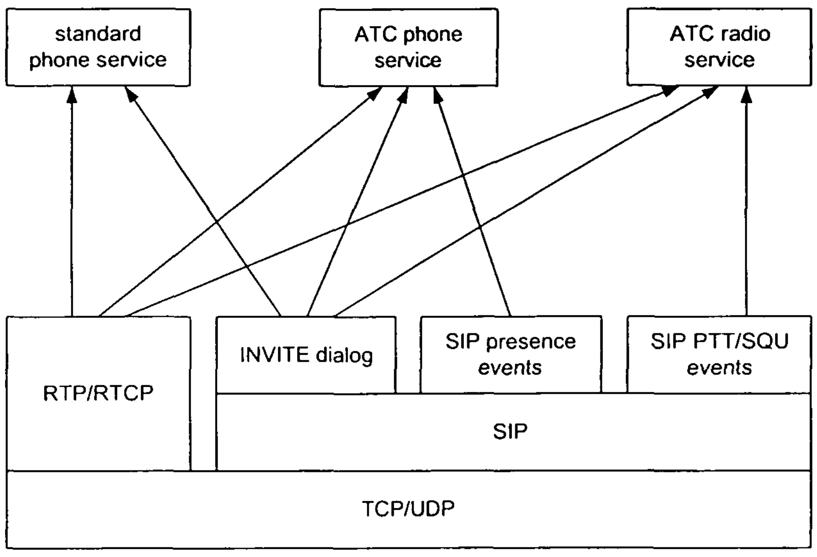


Figure 2.5: Enhanced voice services

to a radio server[7] in the LAN of the ATC center, which receives the single RTP stream from the radio gateway and forwards the stream to each interested operator. In many cases, ATC requires more advanced configurations where several controllers operate on the same sector and sectors are covered by several radios to achieve better coverage. These advanced services are also provided by the radio server. It arbitrates the access to the radio sector and aggregates streams from multiple radio gateways to a single stream, which will be forwarded to the operators.

Figure 2.6 shows the architecture of the radio service. The architecture is a hierarchical client-server solution. The radio gateway acts as server and offers a basic radio gateway service to the radio server. The radio server in turn acts as server for the working position. It offers the ATC radio service, manages access to radios and distributes incoming radio messages. The signaling and voice transmission protocols between operator position and radio server are identical to the protocols between radio server and radio gateway; only the transmitted information differs. This allows the operator position to connect directly to the radio gateway. This is only useful as fallback scenario where all radio servers have failed, as it allows only one operator to connect to a certain radio resource[8]

[7]The radio server is not a physical device, but a server application. Therefore, several radio servers, i.e. several radio sectors can be hosted on the same host. This is similar to web servers, where multiple web server instances can run on the same host.

[8]Multiple connections may be possible if the bandwidth to the radio gateway is sufficient, but this also requires basic lockout functionality in the radio gateway to handle concurrent radio access.

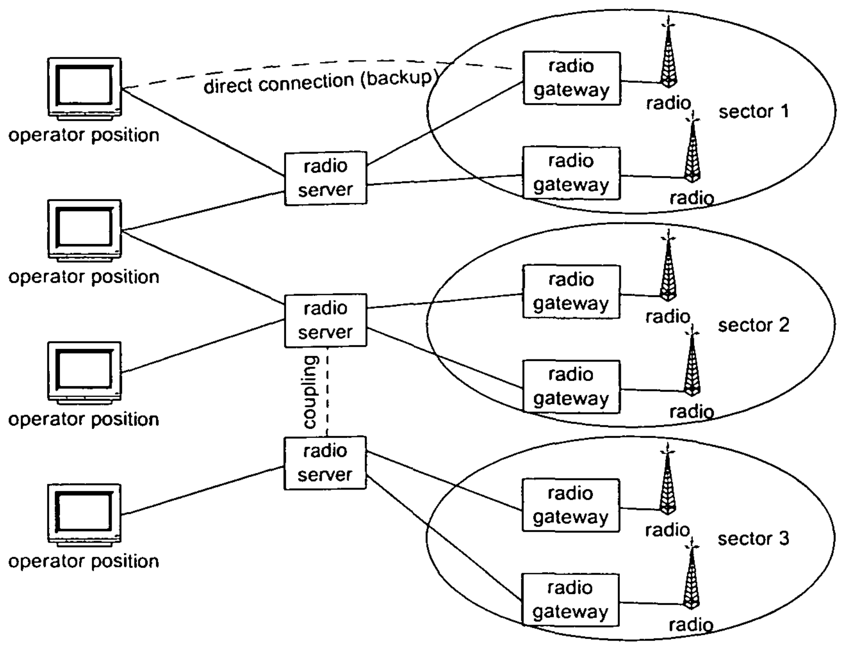


Figure 2.6: Radio architecture

In periods with low air traffic, sectors are grouped together to form a new sector, called coupling. As the initial sectors operate on different frequencies, the voice communication system has to forward incoming radio messages from one sector to the other sectors. Therefore, the voice stream has to be forwarded between the radio servers. This forwarding can be done at the operator position which is connected to both radio servers or directly between the two radio servers. The coupling is a global configuration, i.e. if an operator configures coupling for certain sectors, this effects all operators. If more than 2 sectors are coupled, the coupling should be done centrally as shown in figure 2.7. One of the radio servers acts as the master server, which relays the RTP streams to the slave radio servers and arbitrates the sector access. This setup minimizes the voice delay and avoids loops, which can occur in a meshed setup.

To keep the overall voice delay in the system low, it should be avoided to process audio packets, but simply relay them. Therefore, the radio gateway and the operator position use the same codec to avoid transcoding at the radio server. Furthermore, all the nodes use the same RTP packet size to avoid an additional delay because of buffering of smaller packets to build larger packets.

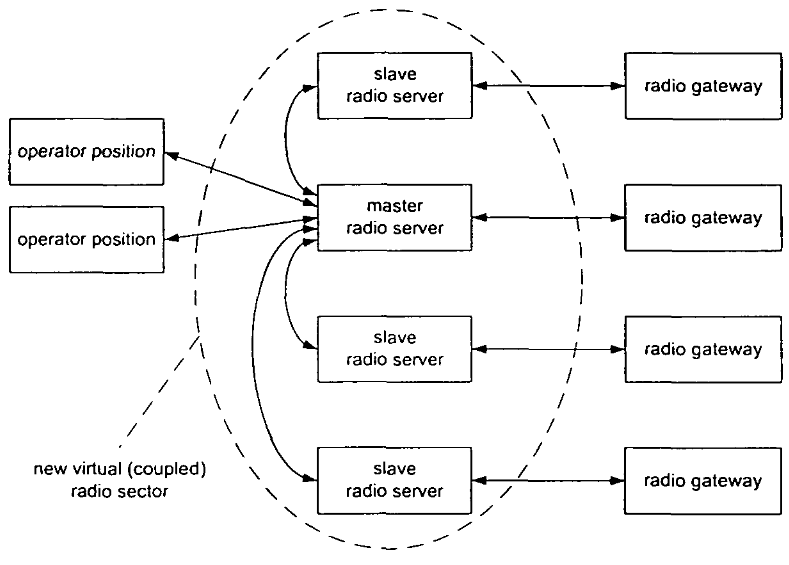


Figure 2.7: Coupling of radio sectors

2.6.1 Operator Position

The operator position is the working position of the operator. In air traffic control, this working position basically consists of a radar screen and the voice communication system client. The voice part consists of an input device, i.e. a keyboard or a touch screen device, a PTT button, and one ore more audio in- and output channels. Figure 2.2 showed the class diagram of the radio system from an outside point of view. From the operator's point of view, the class diagram changes slightly. The operator does not care on which particular radio a message is sent or received. The operator is only interested in the sector. Therefore, the radio class needs no representation in the operator position application, as shown in figure 2.8.

The Sector class represents the radio sector, which is originally handled by the radio server, within in the operator position application. It will be used by the Operator class, the main controller class within the operator position application, for radio communication. As the operator also needs to communicate with other operators, they will be represented by the otherOperator class.

The Sector class needs to communicate with the radio server and the otherOperator class needs to communicate with other operator positions. This will be handled by the Rtp-Connection class, which manages the transmission and reception of the audio stream, and by the SignalingConnection class, which handles the session setup and PTT/SQU signaling. As the "connection" objects may be associated with a Sector or an otherOperator object, but only one at a time, there is an *xor* constraint on the association.

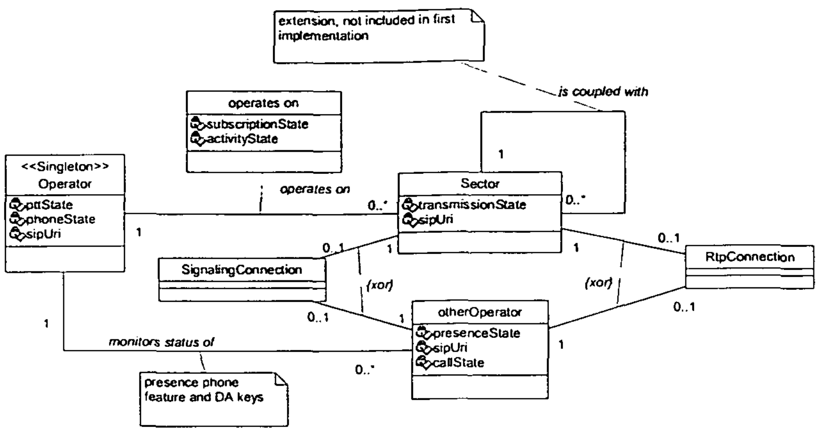


Figure 2.8: High level operator position design model

An operator's subscription to a sector may have several stages. The stages of this subscription and the activity within this subscription are represented by the status of the association class operates on. The possible subscription modes and the transition from one mode to another are shown in figure 2.9. In the disconnected state, the operator position is aware of the sector and the possibility to use it, but does not request any service. In the status state, the operator position may retrieve plain status information from the Sector class, like ongoing sector activity without any voice information. In the Rx state, the operator also receives the voice stream from the radio sector and in the RxTx state, the operator is allowed to transmit radio messages, too.

If the operator wants to transmit a radio message, the operator position requests access

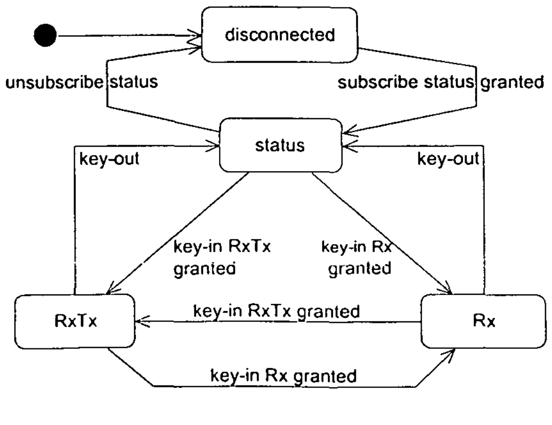


Figure 2.9: Subscription state diagram

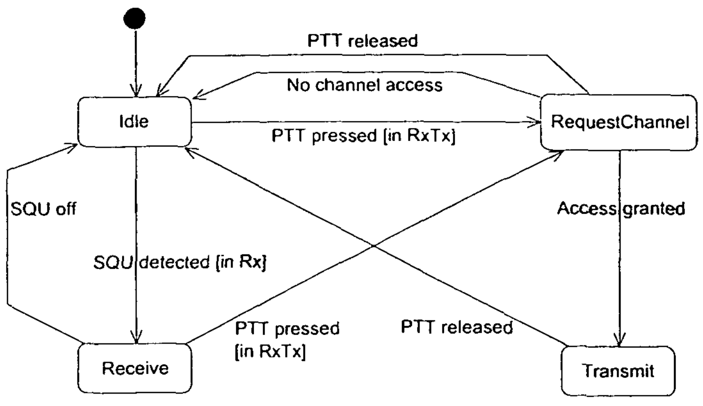


Figure 2.10: Sector activity state diagram at operator position

to the sector and waits for a positive (channel access granted) or negative response (channel access prohibited). This activity is modeled in the sector activity state diagram shown in figure 2.10. As the sector arbitration is done by the radio servers, the requests for channel access and responses will be sent via the IP network. For a detailed discussion about this signaling refer to chapter 3.

2.6.2 Radio Gateway

The new IP based radio architecture supports the existing radio equipment. Thus, the IP based system needs interfaces to the used radio interfaces. Most interfaces are analog with dedicated audio inputs and outputs along with signaling lines for PTT and SQU. The radio gateway translates the radio control signals to the corresponding SIP messages and vice versa. Furthermore, the gateway digitizes incoming analog audio and sends the audio as RTP stream to the subscriber of this radio channel. Contrary, the received RTP stream is played back as analog audio whereby jitter buffers and playout algorithms have to be used to achieve uninterruptible audio playback. The radio resources usually reside on remote sites and depending on connectivity of the remote site two configurations are possible.

Figure 2.11 shows the possible locations of the radio gateway. If IP is used in the last mile, the radio gateway will reside at the radio location and communicates directly with the radio. When existing connections (4-wire, ISDN) are used, the radio gateway resides in the ATC center, possibly on the same host as the radio server.

The radio gateway is not a physical device, but a logical entity. Therefore, multiple radio gateways can be hosted on a host with multiple radio interfaces.

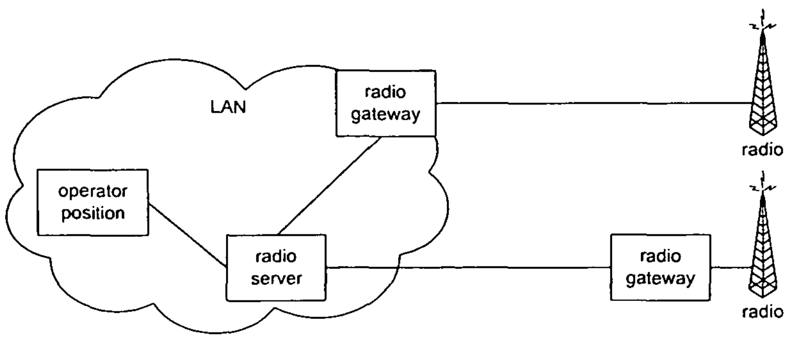


Figure 2.11: Radio gateway positioning

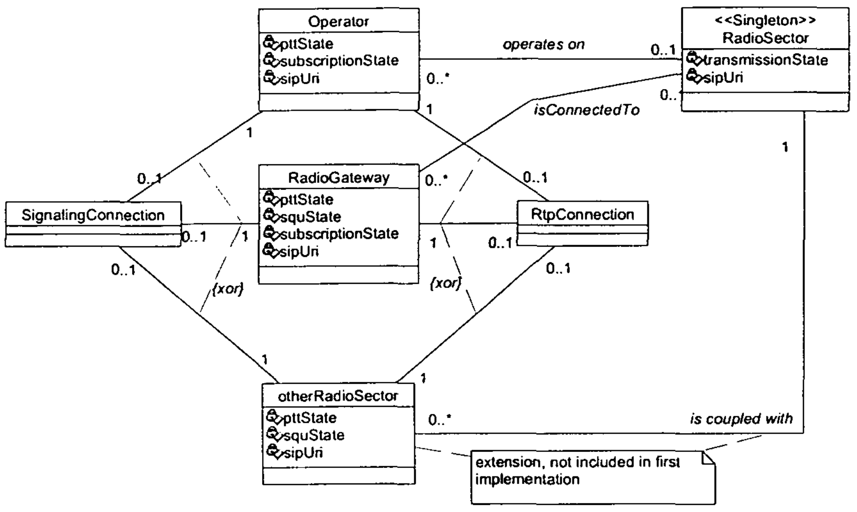


Figure 2.12: High level design model of the radio server

2.6.3 Radio Server

The radio server is the only component of the radio service which deals with all other components of the radio service, in case of coupled radio sectors even with other radio servers. Figure 2.12 shows the class diagram of the radio server. The Operator, RadioGateway and otherRadioSector classes represent the remote parties in the local radio server application. The RtpConnection class implements the audio communication tasks, whereas the SignalingConnection class implements the session setup and radio signaling tasks. The basic radio server functionality is independent of the signaling protocol used by the SignalingConnection class, i.e. any of the signaling methods described in chapter 3 can be used.

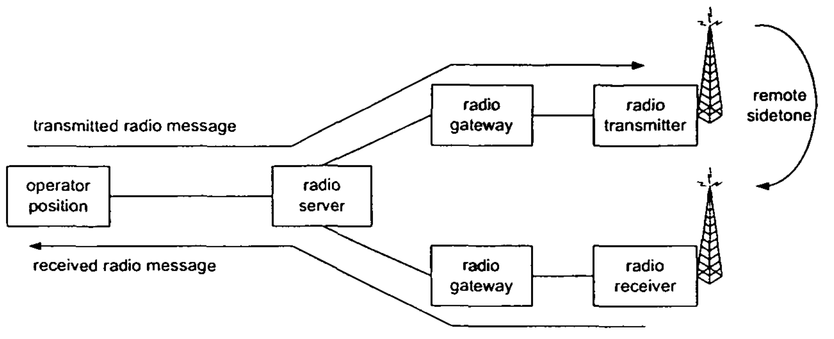


Figure 2.13: Remote sidetone

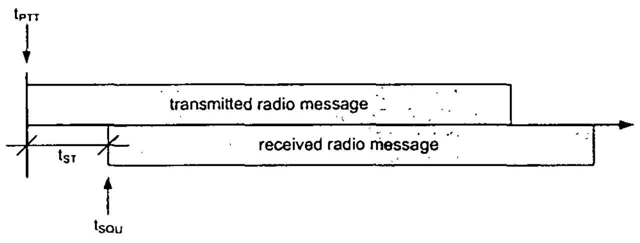


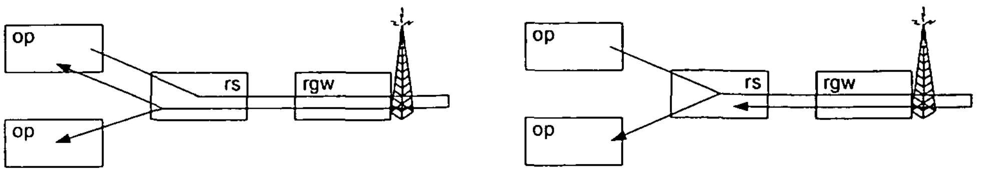
Figure 2.14: Remote sidetone delay

2.7 Remote Sidetone Problem

Remote sidetone means, that an operator will hear its own, currently transmitted radio message, because the radio messages will be received by a radio receiver and sent back to the operator as shown in figure 2.13. In current circuit switched systems, the remote sidetone is a feature which enables the operator to verify the successful transmission of a radio message. As each node in the system delays the radio message, the incoming radio message arrives at the operator position with delay in respect to the sent radio message as shown in figure 2.14. If the sidetone delay $t_{ST} = t_{SQU} - t_{PTT}$ exceeds a certain limit, the feature turns into a nasty echo which disturbs the operator. Current circuit switched systems are known for low voice delays, in the range of several milliseconds.

The delay characteristic of VoIP based communication systems causes problems when the remote sidetone feature is enabled. The remote sidetone can be seen as echo which disturbs the operator, if the delay exceeds 25 ms [JVP00]. The delay arises from the framing delay, the transmission delay[9], the jitter buffer and the analog/digital conversion and vice versa.

[9]The transmission delay is negligible in LANs, but considerable in WANs.



(a) Blocking at the operator position (b) Blocking at the radio server

Figure 2.15: Blocking of the remote sidetone

As the sum of these delays is likely more than 25 ms[10], the remote sidetone feature is not possible in VoIP based systems and the remote sidetone must be disabled. If the operator still needs to hear its own voice during a radio transmission, the sidetone has to be produced locally at the operator position. Therefore, an instance, which monitors the radio transmission and informs the operator about failed transmission, is necessary. The notification can be done e.g. by generating an audio warning at the operator position.

There are two options to disable the remote sidetone:

Blocking: During transmission phases, the received radio message will be blocked and ignored. The reception can only be blocked at the radio server or at the operator position as the radio gateway does not know if the reception is a remote sidetone or an incoming radio message from a pilot because the transmitted radio message may have been sent via another radio gateway.

Echo cancellation: An echo canceler will be used to clean the received signal from the outgoing voice signal. Echo cancellation is a complex task, especially when the echo delay varies like in packet switched systems, and requires additional hardware or software algorithms. [MUA90][RZR02].

Blocking does not only block the remote sidetone but also blocks incoming radio messages from pilots when they talk at the same time as the operator. Usually the received sidetone is so much stronger that a received pilot's message is not audible at all. Furthermore, the listen-before-talk procedure mostly avoids these situations, and, if not, the radio messages have to be repeated by the participants. The blocking of the remote sidetone can be done at the operator position or at the radio server as shown in figure 2.15.

[10]Typical values for one-way are: framing delay 10 ms, jitter buffer 20 ms, recording and playback delay 10 ms.

If the blocking is done only at the operator position, the radio server does not distribute the outgoing message to the operator position, but rather waits for the remote sidetone, which will be forwarded to all the operator positions. Subsequently, the operator positions have to decide, whether the message should be blocked or not. If the blocking is done by the radio server, incoming messages will be blocked during PTT phases. Furthermore, the radio server waits for the remote sidetone and triggers some errors, if there is no sidetone within a certain period (see section 4.3). Unlike the human operator, the radio server can hardly detect if the received radio message is the same as the outgoing radio message and therefore, the SQU-on signal is no 100% confirmation that a radio message was sent. This problem also exists in current implementations with disabled remote sidetone and can only be solved by the operator by restransmitting the radio message in case of a missing answer from the pilot. If an outgoing radio message ends (PTT-off) and no SQU-off will be received before a timeout, the radio server will unblock the incoming voice and forwards a SQU event. The timeout can be estimated as the time period between sending the PTT signal to the radio gateway and receiving the SQU event from the radio gateway.

The blocking at the radio server is the most efficient option and keeps the operator position simple. It is used in this work and was implemented in the radio server.

Chapter 3

Radio Signaling

The most challenging part in voice communication systems for air traffic control is the radio signaling. Radio signaling is necessary to inform operators about ongoing activities at a sector and to control radio devices. There are two signals associated with radio devices: PTT and SQU. The PTT signal triggers the radio device to activate the transmitter to transmit a radio message. When the operator presses the push-to-talk button, a PTT signal will be produced at the operator position and sent to the radio server as shown in figure 3.1. The radio server analyzes the incoming PTT request, usually grants sector access for the requesting operator and forwards the PTT signal to the radio gateway. Furthermore, the PTT signal has to be forwarded to all operators working on the same sector to inform them about the sent radio message. The radio server may also decline access to the sector, for example if sector access is already granted to another operator. Regardless if sector access is granted or declined, the radio server has to inform the PTT requesting operator about the result of the PTT request.

In the opposite direction, if there is an incoming radio message, the radio gateway signals SQU to the radio server, which in turn forwards the SQU signal to all connected operator positions (figure 3.2).

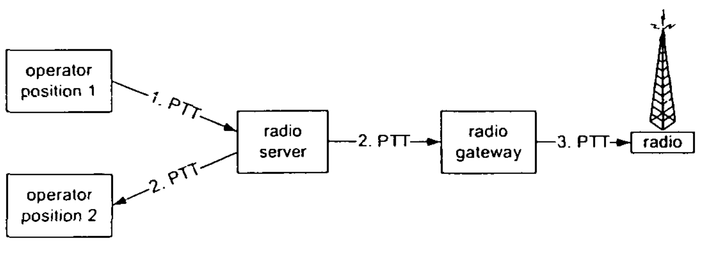


Figure 3.1: PTT signaling scenario

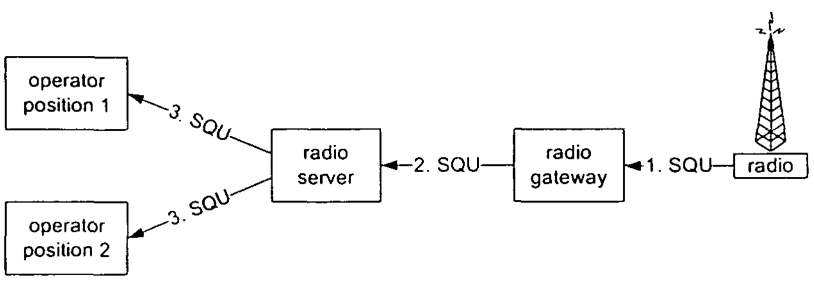


Figure 3.2: SQU signaling scenario

Additionally, the PTT and SQU signals will contain supplementary information like the name of the receiving radio or the name of the currently talking operator. The kind of supplementary data depends on the signal type (PTT or SQU), and on the destination of the message.

There are two fundamental methods to signal the PTT and SQU activities: event based and continuous signaling. In both cases, a radio gateway is necessary to transform the signaling mechanism from the IP based network onto the signaling mechanism used by the radio. Radios typically use separate radio control lines or in-band signaling for receiving PTT and sending SQU. Depending on the amount of supplementary data that needs to be signaled, the rate of status changes and the required reliability of status signaling, one of the following techniques or a combination of them will be the optimal solution.

In the following, the entity which produces the radio signals and sends them will be called *sender* and the entity which receives the radio signals will be called *receiver*. The next sections will discuss the theory of status signaling and the usage of SIP, RTP/RTCP and a proprietary protocol as status signaling protocol. Furthermore, the combination of SIP based signaling and RTP based audio streams to build up radio services will be discussed.

3.1 Event Based Signaling versus Continuous Signaling

The principle of event based signaling is to transmit status information only if the status changes, whereas in continuous signaling the current status is transmitted permanently. In a packet switched network, real continuous signaling is not possible, therefore, continuous signaling means that the current status will be signaled in short, periodic intervals. As event based signaling transmits only status changes, it is possible to have huge time intervals without status signaling. Therefore, it is very important that the signals will be transmitted in a reliable manner. This in turn allows to transmit only the update to the previous state without the necessity to transmit the complete status information. When using continuous signaling, a reliable delivery is not necessary, as the status will be transmitted in certain, usually short intervals. As lost packets won't be retransmitted, every packet has to include the complete state information.

Choosing the proper signaling method is a trade-off between bandwidth consumption, error detection, performance, implementation costs, interoperability, and dependability:

Bandwidth Consumption: If the status changes rarely, continuous signaling causes much more traffic overhead than event based signaling, whereas at high update rates the event based approach may cause more overhead due to additional traffic for confirmation of the received status updates. Bandwidth consumption can be optimized by using multicast or broadcast for status delivery, if there are multiple receivers. As event based signaling requires reliable delivery, i.e. by using TCP or by introducing a retransmission mechanism at application level, it is more complicated to use multicast with event based signaling than with continuous signaling.

Error Detection: In highly available environments it is very important to detect failed senders and broken network links. This is very easy using continuous signaling, as the receiver knows the time interval between the notifications and therefore, missing notifications will indicate failures. If the system uses event based signaling, during long periods without status signaling messages, additional *keep alive* signaling can be used to verify that the sender and the network link is still alive[1].

Performance: If the operator wants to transmit a radio message, he presses PTT and usually starts talking immediately. The PTT request has to be sent to the radio server and if the radio server declines PTT, the operator should be informed at best before he started talking. Therefore, fast PTT signaling is necessary. Furthermore, slow signaling will raise the probability of collisions of radio messages.

In [Eurb], Eurocontrol suggests a maximum PTT signaling delay of 25 ms. One reason for the required fast PTT signaling is the characteristics of the current circuit switched systems and the radio transmitters. The PTT signaling delay plus the transmitter attack-time delay[2] must be less than the audio delay plus the

[1] A missing response to a keep alive request does not necessarily mean a failure of the remote node, i.e. the response might have been lost in the network. Nevertheless, a received response is an indicator that the network link and the remote service were alive at the time of sending the request.

[2] The duration from the instant PTT is signaled to the radio transmitter to the instant the transmitted radio frequency signal amplitude has increased to a specified level, e.g. 90% of its key-on steady-state value.

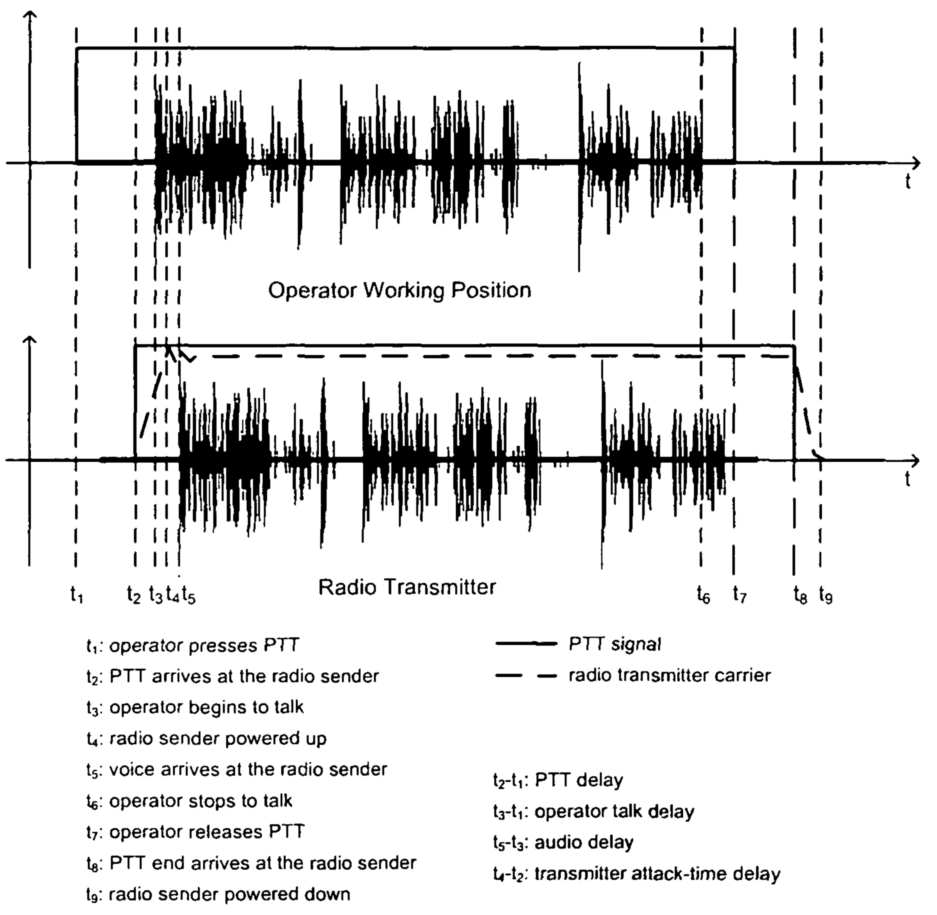

Operator Working Position
Radio Transmitter
t
t
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9
t_1: operator presses PTT
t_2: PTT arrives at the radio sender
t_3: operator begins to talk
t_4: radio sender powered up
t_5: voice arrives at the radio sender
t_6: operator stops to talk
t_7: operator releases PTT
t_8: PTT end arrives at the radio sender
t_9: radio sender powered down
PTT signal
radio transmitter carrier
t_2-t_1: PTT delay
t_3-t_1: operator talk delay
t_5-t_3: audio delay
t_4-t_2: transmitter attack-time delay

Figure 3.3: PTT delay and voice delay

gap between pressing PTT and start talking, to avoid loss of syllables (figure 3.3). As the voice delay in current systems is very small, the PTT signaling has to be very fast. In an IP based system the characteristics will change. Due to the packetizing delay and a necessary jitter buffer in the radio gateway, e.g. 20 ms frames and 40 ms jitter buffer result in 60 ms voice delay, the timing requirements for the PTT signaling in packet switched networks are not that stringent as in circuit switched systems.

Implementation Costs: The radio signaling can be done using existing protocols or by defining a new protocol. To keep implementation costs low, the reuse of existing protocols is preferred. As described in section 1.3, typical VoIP applications use one protocol for session signaling and another for the media transport. In principle, both of them can be used to transport the radio signaling messages as well.

Interoperability: The usage of open standardized protocols for session setup, audio transmission, and radio signaling will ease integration of 3rd party products and will ease the interconnection of several air traffic control centers.

In circuit switched systems, the signaling methods can be distinguished into *in-band* and *out-band* signaling methods. In-band signaling will be transmitted in the same logical channel as the voice data it belongs to, e.g. in the same time slot as the voice. Therefore, the signaling and the voice data belong to each other and will be transmitted through the system together until they reach their destination, where they will be separated again. Out-band signaling methods transport the radio status information in a separate channel. Hence, the radio signaling may take different routes than the voice data and the signaling data has to include some information to which voice channel it belongs.

As the proposed radio architecture uses SIP and RTP/RTCP for enabling voice services over IP, the next sections deal with status signaling on top of these protocols. Additionally, a proprietary signaling protocol will be discussed to show the assets and drawbacks of the mentioned protocols.

3.2 SIP-based Radio Signaling

SIP offers several extensions which can be used for status information delivery. The "Session Initiation Protocol (SIP) Extension for Instant Messaging" [CER+02] defines the MESSAGE method, which transports instant messages in- and outside of existing SIP dialogs. "The SIP INFO Method" [Don00] defines the INFO method, which is intended to transport mid-session information like DTMF digits generated during a SIP session. The INFO message must be sent within an existing dialog.

Although these extensions can be used for status signaling, there is an extension which is exactly designed for status signaling: the "Session Initiation Protocol (SIP)-Specific Event Notification" [Roa02]. It defines a framework by which SIP nodes can request notifications from other SIP nodes. These notifications indicate that certain events have occurred.

3.2.1 SIP-Specific Event Notification

The SIP-specific event notification allows to subscribe to asynchronous notification of events to build up various SIP services, for example a buddy list can be implemented by subscribing the presence events of a buddy and receiving notifications every time the presence status of the buddy changes. This extension is no "ready to use" extension, as it only defines how to subscribe and receive certain events. The events and some special behavior related to the respective events have to be defined in a separate event package.

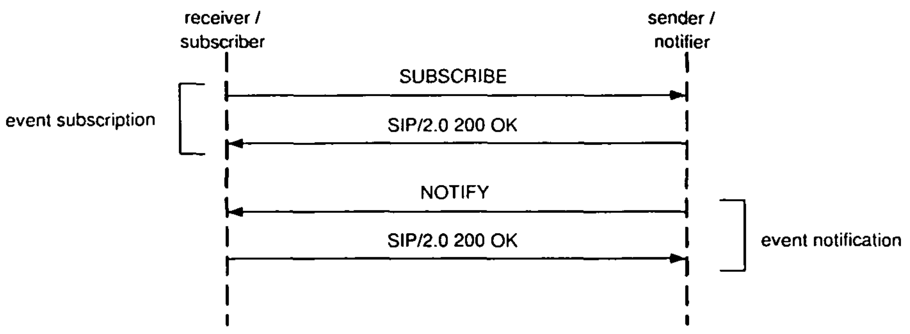


Figure 3.4: Event subscription and notification

Figure 3.4 shows the general concept of the event notification. The receiver, in SIP terms called *subscriber*, subscribes to events using the SUBSCRIBE message. The requested event must be stated in the Event: header. The sender, in SIP terms called *notifier*, has to verify the event header before accepting the request. Subscriptions to not supported events must be declined using the 489 Bad Event response. A 200 OK response indicates that the request was understood and the subscription was accepted, whereas a 202 OK indicates that the subscription was understood, but authorization may or may not have been granted. That means, after a 202 OK response the subscriber has to wait for an incoming NOTIFY message with a Subscription-State: active header to be sure that the subscription was accepted. The Expires: header indicates the duration of the subscription in seconds. If the subscription expires, the notifier cancels the subscription. Therefore, the subscriber has to renew the subscription regularly.

If the SUBSCRIBE request was accepted, the notifier immediately has to send a notification which describes the current state of the requested resource. As the 200 OK and the NOTIFY message may take different routes, the subscriber must be prepared to receive the first notification prior to the 200 OK response of the subscription. The subscription can be canceled by the subscriber and the notifier. The subscriber cancels by sending a SUBSCRIBE request with an Expires: 0 header and the notifier may cancel a subscription by sending a NOTIFY message with a Subscription-State: terminated header.

3.2.2 PTT/SQU Event Package

To use the SIP-specific event notification for radio signaling, a dedicated PTT/SQU event package is necessary. The event package defines the following types of events:

PTT-on: Indicates the beginning of a radio message sent by an operator.

PTT-off: Indicates the end of a radio message sent by an operator.

PTT-lockout: Indicates that an operator was locked out of a sector, e.g. if the sector is already assigned to another operator or if an operator was overridden by another operator.

SQU-on: Indicates the beginning of an incoming radio message.

SQU-off: Indicates the end of an incoming radio message.

There is no explicit notification to the PTT requesting operator when sector access is granted. The operator position requests access to a sector by sending a PTT-on notification to the radio server. If the radio server grants access, it has to inform all subscribers by sending a PTT-on notification to all of them. As a Tx subscribed operator is always also Rx subscribed, the PTT requesting operator also receives the PTT-on notification from the radio server. This notification indicates the PTT requesting operator as current PTT holder and therefore it is an implicit acknowledgment that sector access was granted.

All notifications have to contain the SIP address of the originating node inside the notification body, as the SIP `From:` header might indicate intermediate nodes like the radio server. Furthermore, the notifications may include additional PTT/SQU information like the signal level at the receiver, the location of the receiver or the flight number of the sender of the radio message. The events will be transmitted in the body of the `NOTIFY` message. The syntax of the message body is defined in the PTT/SQU event package and is `type=value` based. Figure 3.5 shows a notification sent from the radio server to a subscribed operator, indicating an incoming radio message on sector f100.

The several nodes of the radio architecture generate and process various events. Table 3.1[3] shows the signaling dependencies between the nodes, e.g. the operator position receives the PTT, PTT-lockout and SQU information from the radio server. Therefore, the operator subscribes the events of the sector at the appropriate radio server. After a subscription, all events related to this sector, e.g. SQU-on and PTT-on, will be signaled to the subscriber. These events correlate with the state transition in figure 2.10, e.g. when the operator is in the Idle state and receives a SQU-on event, the state will change to Receive.

The `SUBSCRIBE` messages usually take the same route as the `INVITE`s, including proxies as shown in figure 3.6. Figure 3.7 shows notifications, which are directly sent to each subscriber using unicast. Notifications will be retransmitted until a final response or a timeout. On success, the subscriber responds with a `200 OK` message. If a `NOTIFY` request fails, i.e. the response times out, or a non-2xx class final response code is received, the notifier will remove the subscription for this subscriber.

[3]If the PTT and SQU events are mentioned without exact definition of the event, the statement implies both, *-on* and *-off* events.

```
NOTIFY sip:operator12@pc44.atc.org SIP/2.0
Via: SIP/2.0/UDP 128.131.80.6:5062
From: "Sector Austria West f100" <sip:f100@atc.org>;tag=F68
To: "Operator 12" <sip:op12@atc.org>;tag=1ec23337
Contact: "Sector Austria West f100" <sip:f100@128.131.80.6:5062>
Call-ID: 3c9196445d5240bfb7b660a16ee6d390@128.131.80.223
CSeq: 522 NOTIFY
Event: PTT/SQU
Content-Type: application/text
Content-Length: 113

sector=sip:f100@atc.org
status=SQU-on
radio=sip:radio15@atc.org
signal-level=-30dB
flight-number=1234567890
```

Figure 3.5: SIP notification of an incoming radio message event

to / from	radio gateway	radio server	operator position
radio gateway		SQU	
radio server	PTT	PTT/SQU[a]	PTT/PTT-lockout/SQU
operator position		PTT/SQU	

[a]Only necessary for coupling

Table 3.1: PTT/SQU events sending and receive matrix

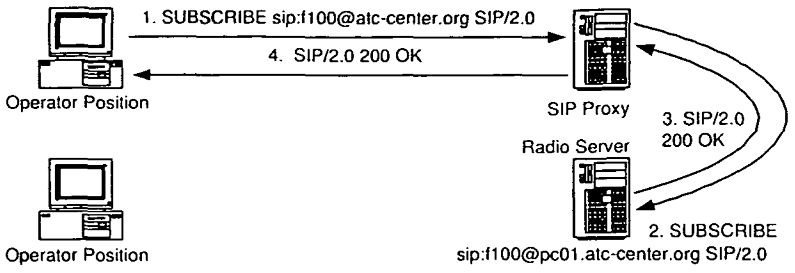

1. SUBSCRIBE sip:f100@atc-center.org SIP/2.0
4. SIP/2.0 200 OK
Operator Position
SIP Proxy
Radio Server
3. SIP/2.0 200 OK
Operator Position
2. SUBSCRIBE sip:f100@pc01.atc-center.org SIP/2.0

Figure 3.6: Event subscription

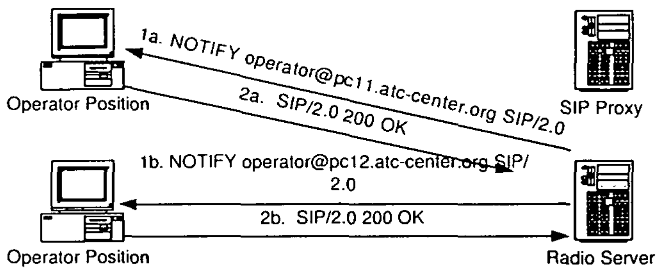
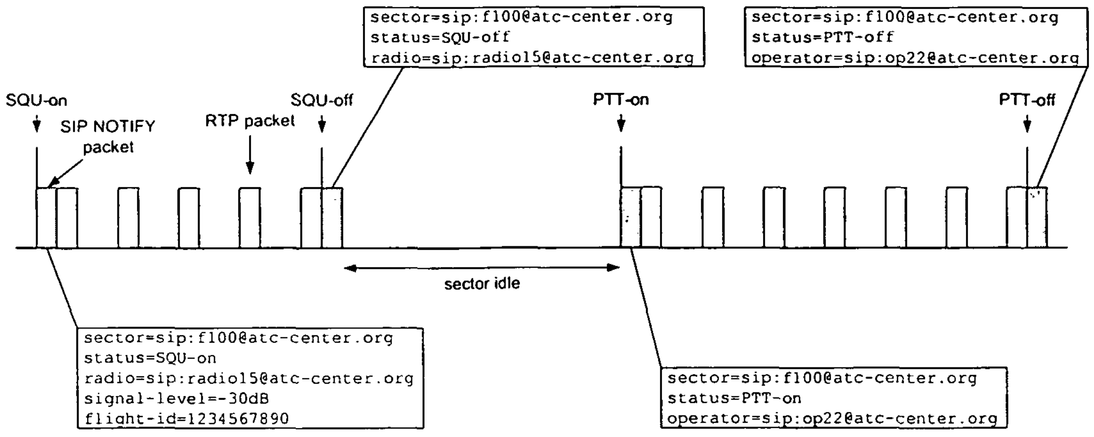

1a. NOTIFY operator@pc11.atc-center.org SIP/2.0
2a. SIP/2.0 200 OK
Operator Position
SIP Proxy
1b. NOTIFY operator@pc12.atc-center.org SIP/2.0
2b. SIP/2.0 200 OK
Operator Position
Radio Server

Figure 3.7: Event notification

A typical signaling scenario is shown in figure 3.8. This scenario shows the SIP and RTP packets flow between the radio server of sector f100 and an operator position during a SQU and a PTT event. The begin of the incoming radio message, received by radio15, is signaled by a SIP `NOTIFY` message. Immediately afterwards, the radio server starts forwarding the RTP packets, which it has received from the radio gateway. When the incoming radio message is over, the RTP forwarding will be stopped and the radio server signals SQU-off to the operator position. During idle times, when there is no voice activity at the certain sector, no RTP packets are sent. After the idle time, operator op22 transmits a radio message at sector f100.

Although the previous scenario shows SIP and RTP packets, the SIP based radio signaling is independent from the audio session. This allows every SIP client to subscribe the PTT/SQU events to monitor the sector activity without the need of setting up an audio session. Furthermore, sector arbitration is easier if the signaling is event based. If the PTT/SQU information is contained in the RTP stream, the radio server has to analyze the PTT/SQU information in every incoming RTP packet, apply the arbitration logic, and put the result in every outgoing RTP packet. Using the SIP based technique, the radio server waits for an incoming event, applies the arbitration logic, configures the RTP forwarding unit, and sends the result to all subscribers. Therefore, the arbitration



```
sector=sip:f100@atc-center.org
status=SQU-on
radio=sip:radio15@atc-center.org
signal-level=-30dB
flight-id=1234567890
```

```
sector=sip:f100@atc-center.org
status=SQU-off
radio=sip:radio15@atc-center.org
```

```
sector=sip:f100@atc-center.org
status=PTT-on
operator=sip:op22@atc-center.org
```

```
sector=sip:f100@atc-center.org
status=PTT-off
operator=sip:op22@atc-center.org
```

Figure 3.8: SIP based radio signaling scenario

logic has to be applied only once per radio message.

As the SIP based signaling is an out-band signaling method, the characteristic of IP networks and SIP has to be kept in mind. Due to different possible routes, various prioritization mechanisms and possible packet loss, it is possible that the SIP packet that marks the beginning of a radio message arrives later at the destination than the first RTP packet. Vice versa, the SIP NOTIFY message, which indicates the end of a radio message, might overtake the last RTP packets. In the worst case, the receiver will ignore the RTP packets which arrive late[4]. This situation is unlike to happen in LANs, but not in WANs with several simultaneous routes for load balancing. Additionally, the SIP messages might traverse several proxies on their way, whereas RTP packets are usually sent directly between the participants. Therefore, and for general performance issues, it is suggested to keep the number of intermediate SIP proxies low and prefer direct end-to-end SIP signaling.

To deal with jitter, the received audio data has to be buffered[5] before playback. Therefore, as shown in figure 3.9, the receiver must not turn off playback after the reception of the PTT-off signal, but has to empty the jitter buffer and accept late RTP packets. Aborting the playback immediately after the PTT-off reception would probably cut off some syllables at the end of the radio message.

SIP is based on a request-response model—every time a SIP client sends out a SIP request, it awaits a response. In SIP terms, a request and its corresponding responses

[4]A solution for this problem can be found in section 6.3.

[5]A typical jitter buffer size for Internet applications is 2 times the frame size (e.g. 40 ms for 20 ms frame packets) or more, depending on the quality of the Internet connection. In LANs, the jitter buffer can be reduced.

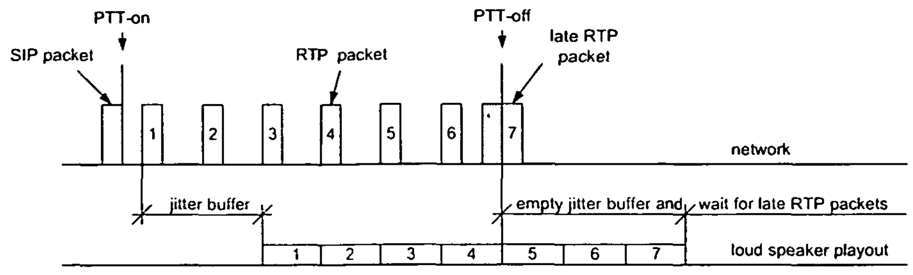


Figure 3.9: Audio playback despite PTT-off signal

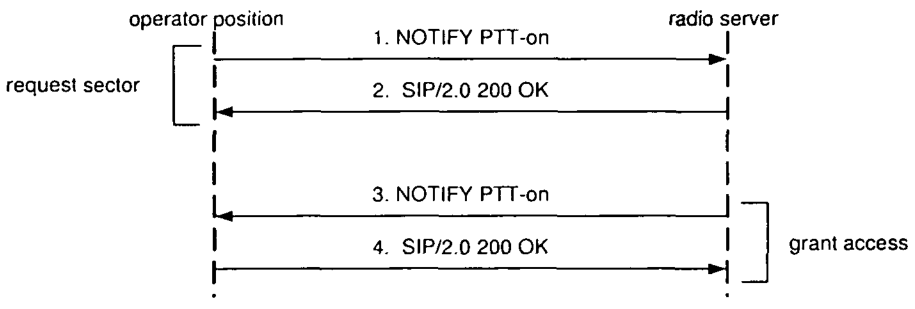


Figure 3.10: 4-way PTT handshake

are called *transaction*. The sector access signaling consists of 2 transactions (figure 3.10), one transaction signals the radio server that the operator requests access to the sector, whereas the second transaction signals the operator the result of the sector access: granted or lockout. Therefore, sending of 3 SIP messages is necessary until the operator receives the result of its request because of the behavior of SIP. Another possibility would be to integrate the result into the 200 OK response, e.g. in the body of the message. The possible use of this method depends on the used SIP stack, e.g. *dissipate2* [Bila] automatically answers 200 OK to incoming NOTIFY messages before relaying the body of the notification to the application layer. Therefore, this SIP stack does not allow to integrate signaling data into the response. Other signaling methods like the explicit RTCP signaling (section 3.3.3) or a proprietary signaling (section 3.4) do not have this characteristic.

As discussed in section 3.1, event based signaling requires a reliable transport of the signaling messages. SIP fulfills the requirement by using TCP as transport protocol or by using UDP and a retransmission mechanism.

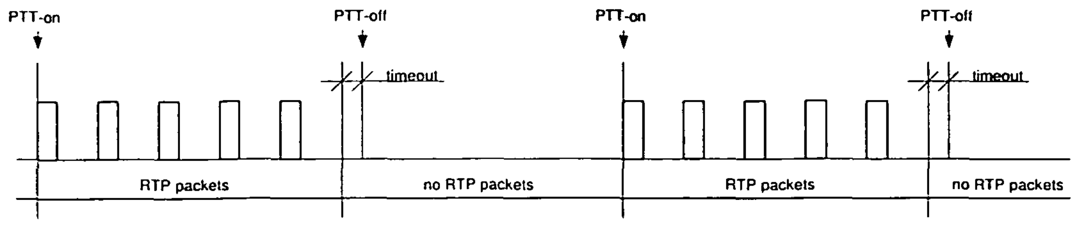


Figure 3.11: Implicit PTT signaling

3.3 RTP/RTCP-based Radio Signaling

The real-time transport protocol (RTP) [SCFJ03] provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio or video over multicast or unicast network services. The data transport is augmented by the RTP control protocol (RTCP) to allow monitoring of the data delivery in a manner scalable to large multicast networks, and to provide minimal control and identification functionality. RTCP is not necessary to transmit audio data, but it might be useful to report and collect statistics about the audio transmission. These data can be used to adapt the voice processing, for example to change the jitter buffer size dynamically or to switch to another codec.

RTP and RTCP offer various ways for signaling radio events. The simplest way is to use the presence of RTP packets as PTT/SQU indicator. More advanced methods integrate the PTT/SQU information into the RTP stream using the RTP extension header or embed it into the voice payload. This allows to transport more information than just PTT on and off. Furthermore, the RTCP protocol can be extended by self-defined, application specific messages to signal PTT/SQU.

3.3.1 Implicit RTP Radio Signaling

Implicit radio signaling means, that there is no explicit PTT/SQU signaling in the system and the PTT/SQU information will be derived from some other information. In this case, the reception of RTP packets will be used to derive the PTT/SQU information. As shown in figure 3.11, the first incoming RTP packet signals PTT/SQU-on and after a certain timeout, the absence of RTP packets will be treated as PTT/SQU-off. This method does not distinguish between PTT and SQU signals, nevertheless in most cases it is unambiguous, e.g. the PTT/SQU information derived from an RTP stream sent by the radio gateway always indicates SQU, and never PTT.

This method requires that the nodes do not send RTP packets if there is no radio message. This is similar to *silence suppression*[6], which is also offered by several Internet telephony applications [Mic][Gra], except that the voice activity phases are not derived from the recorded signal level but from the current PTT/SQU status.

This method of signaling is not suitable for typical PTT/SQU signaling in ATC, as ATC requires additional PTT/SQU related data to be signaled. Nevertheless, this method can be very useful as an extension to explicit signaling methods for detecting failures in the system. Receiving RTP packets without an explicit PTT/SQU signal or receiving PTT/SQU-on signals without receiving RTP packets will indicate a failure in the PTT/SQU signaling or in the RTP processing unit. Explicit signaling methods which can be combined with implicit RTP signaling are the SIP based radio signaling or the explicit RTP and RTCP based methods.

3.3.2 Explicit RTP Radio Signaling

Signaling radio events using an existing RTP stream falls into the category of continuous signaling (refer to section 3.1). Therefore, the current status has to be transmitted in every RTP packet.

RTP [SCFJ03] offers two possibilities to integrate explicit radio signaling: the extension header and the payload field[7]. Integrating the signaling into the payload, that means into voice data, might only be useful, if the source and the destination of the voice and signaling data already combine them. In any other case it will be better to put the signaling information into the extension header. Both methods can be compared to the in-band signaling method used in circuit switched communication systems, because the signaling is always bound to the corresponding voice data while traversing the network. Figure 3.12 shows the RTP header. The X field represents the extension bit. If this bit is set, the fixed header must be followed by exactly one header extension, with a format defined in figure 3.13. The length field counts the number of 32-bit words in the extension, excluding the four-octet extension header, whereas the first 16 bits of the header extension are left open for distinguishing identifiers or parameters. The header extension can be used to transmit any kind of data.

As the RTP packets are sent regularly in small intervals, typically 20 ms, the additional traffic will be considerable, if a textual representation is used. The size of the SQU signal in figure 3.5 is 113 Bytes. This must be round up to a multiple of 32 Bits which will lead to an extension header of 116 Bytes (including the profile and length field). Compared to the payload of the RTP packet which is 160 Bytes when using the G.711

[6]Silence suppression means, that a UA won't send RTP packets if the user does not speak.

[7]Although the extension header is defined since the first RTP RFC (RFC 1889), there exist SIP phones which do not support the extension header. These phones will handle the header extension as audio payload and therefore produce incorrect audio output

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3.12: RTP header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      defined by profile       |           length              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        header extension                       |
|                             ....                              |
```

Figure 3.13: RTP extension header

codec, this will nearly double the packet size. Furthermore, as every extension has to be processed, this will also increase the processing needs of all nodes which will lead to a reduced number of streams a node can handle.

The explicit RTP signaling requires continuous sending of RTP packets, also during idle time where the RTP payload contains no information. This will also lead to more bandwidth consumption compared to a SIP based signaling method. Nevertheless, this method is perfect to detect failures as a certain amount of time without received RTP packet immediately will indicate a failure of the sending node or of the network. A scenario showing a SQU event, a PTT event and a failure situation is shown in figure 3.14.

Another drawback of this solution is the fixed period for sending RTP packets, e.g. every 20 ms. If a PTT/SQU event occurs 1 ms after the last sent RTP packet, it will take 19 ms before sending the next RTP packet which signals the event. This problem does not occur with event based signaling techniques.

As RTP is also suitable for multicast, this can be used to reduce the traffic. The presented radio architecture uses a central radio server which controls the voice communication. Hence, multicast would bypass the radio server and the sector could not

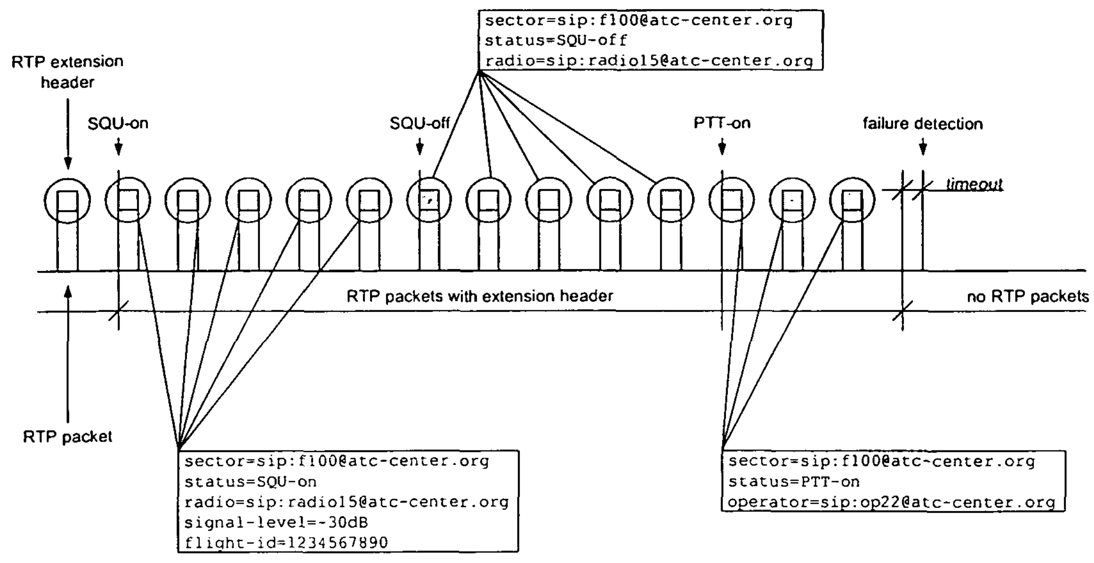

sector=sip:f100@atc-center.org
status=SQU-off
radio=sip:radio15@atc-center.org
RTP extension header
SQU-on
SQU-off
PTT-on
failure detection
timeout
RTP packets with extension header
no RTP packets
RTP packet
sector=sip:f100@atc-center.org
status=SQU-on
radio=sip:radio15@atc-center.org
signal-level=-30dB
flight-id=1234567890
sector=sip:f100@atc-center.org
status=PTT-on
operator=sip:op22@atc-center.org

Figure 3.14: Explicit RTP signaling scenario

be controlled anymore. The only exception is the direction from the radio server to the operator positions, but using a mixture of multicast and unicast causes interoperability problems as currently VoIP devices support either unicast or multicast, but never both at the same time.

3.3.3 Explicit RTCP Radio Signaling

RTCP [SCFJ03] is a complement to RTP and its primary function is to provide feedback on the quality of the data distribution. Although RTCP is an integral part of voice transmission with RTP, not all available VoIP products support RTCP. There are several products [Gra][Sim][Bilb], which neglect RTCP, anyhow they work. Therefore, to use radio signaling over RTCP, all particpants have to support RTCP, nevertheless the radio server may grant reduced functionality to user agents which do not support RTCP.

RTCP defines several type of packets:

SR (Sender report): To report transmission and reception statistics from participants that are active senders.

RR (Receiver report): To report reception statistics from participants that are receivers only.

SDES (Source description items): To describe the RTP source, e.g. the email address or the geographic location of the participant.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P| subtype |   PT=APP=204  |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           SSRC/CSRC                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          name (ASCII)                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   application-dependent data                ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3.15: RTCP APP type packets

BYE: To indicate end of participation.

APP: To transmit application-specific functions. The APP RTCP packets are to experiment with new, application specific functions without the need to register new RTCP packet types with IANA[8].

RTCP packets are not sent continuously like RTP packets, but rather on certain time intervals (SR, RR, SDES, APP) or on certain events (BYE, APP). Using APP packets in a continuous manner to signal radio events is a waste of bandwidth as RTP packets can be reused as described in section 3.3.2. Therefore, the RTCP based radio signaling should be used as an event based signaling method.

Figure 3.15 shows the header of the APP packet, which offers several ways to transport radio signaling data:

- The subtype field
- The name field
- The application-dependent data field

The name field should represent the name of the application which uses this RTCP extension, e.g. atcs for air traffic control services. The subtype field will be used to differ various services for the atcs application, e.g. 0x00001 can be used to identify the radio signaling service. The application-dependent data field will carry the signaling information using the same syntax as used in the SIP based signaling approach (figure 3.5). This approach, as well as the SIP based signaling method, can be compared to out-band signaling in circuit switched systems as the radio signaling information is separated from

[8]Internet Assigned Numbers Authority, http://www.iana.org/

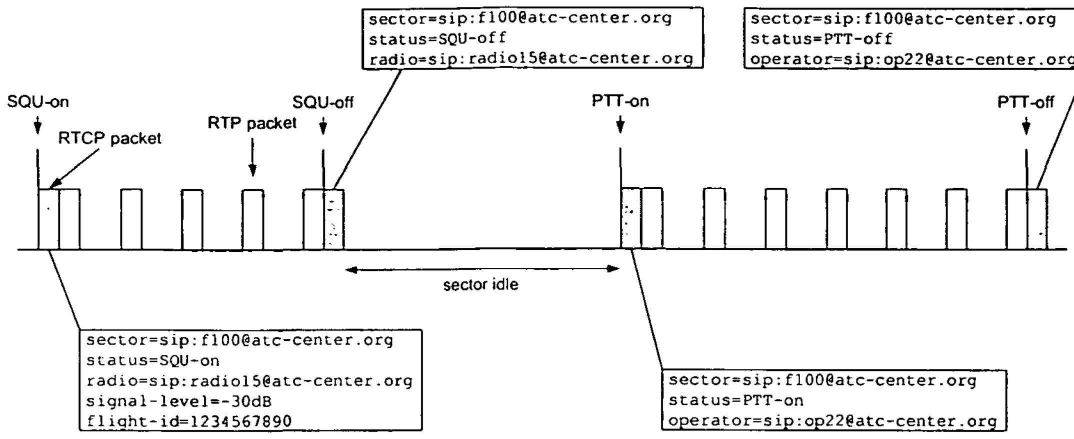

sector=sip:f100@atc-center.org
status=SQU-off
radio=sip:radio15@atc-center.org
sector=sip:f100@atc-center.org
status=PTT-off
operator=sip:op22@atc-center.org
SQU-on
RTCP packet
RTP packet
SQU-off
PTT-on
PTT-off
sector idle
sector=sip:f100@atc-center.org
status=SQU-on
radio=sip:radio15@atc-center.org
signal-level=-30dB
flight-id=1234567890
sector=sip:f100@atc-center.org
status=PTT-on
operator=sip:op22@atc-center.org

Figure 3.16: Explicit RTCP signaling scenario

the voice data, as shown in figure 3.16. Therefore, the receivers and senders have to be aware of the characteristic of packet switched networks as discussed in section 3.2.

RTCP is an unreliable protocol, as it relies on UDP. As event based signaling requires reliable transmission of the status information, a retransmission mechanism has to be implemented on top of RTCP. RTCP APP packets which transport radio events have to be acknowledged by every receiver or the sender has to retransmit the event. Therefore, it is not reasonable to use explicit RTCP signaling together with multicast as acknowledgments and retransmissions are not easy to implement on top of a multicast or broadcast protocol. The *Push-to-talk over Cellular* (PoC) service also uses RTCP to signal floor control [Eria], which is similar to the sector arbitration problem (refer to section 6.2).

In contrast to the SIP based signaling, this method requires the establishment of a voice session, even if the operator is not interested in any voice (`status` state in figure 2.9), using the SIP `INVITE` method. Nevertheless, the voice session will only be used for sending and receiving RTCP packets. Thus, no RTP packets have to be sent. RFC 3264 [RS02a] suggests the usage of the `a=inactive` parameter to setup such a voice session.

3.4 Proprietary Signaling

A self-defined proprietary radio signaling protocol can be event based or continuous. Introducing a proprietary continuous signaling protocol is unnecessary, because the status information can easily be integrated into the RTP stream without additional protocol overhead. A proprietary event based signaling protocol may be fitted exactly onto the needs of radio signaling and therefore will be useful if particular features like multicast should be used or the system has to be trimmed on performance. Nevertheless, it may cause interoperability problems between different vendors and would require more research work in developing the protocol. The requirements for a proprietary protocol are:

Subscription: Enables every node to subscribe the radio events at a notifier.

Notification: To notify all subscribed nodes about the status change of a resource.

Routing and address resolution: The protocol has to provide an address resolution mechanism to enable subscribers to resolve the system internal address of the notifier into an IP address.

Failure detection: The protocol should be able to detect failures of subscribers and notifiers. Furthermore, when multicast should be used, this requires the usage of UDP and therefore a retransmission mechanism must be introduced to deal with packet loss.

Compared to the above presented solutions, which are based on standardized protocols, the only advantage of a proprietary protocol is the usage of multicast or broadcast.

3.5 Signaling Summary

Table 3.2 compares the above presented radio signaling methods. SIP was chosen, as it is interoperable, delivers the messages reliably and does not need to set up an RTP session if a subscriber is only interested in the events. Although SIP is used to signal radio events, the clients may act fault tolerant and treat incoming RTP packets without prior SIP-based PTT/SQU signals also as radio messages and treat such scenarios as errors in the SIP signaling.

signaling method	SIP	RTP	RTCP	Proprietary
reliable	yes	not necessary	no	yes
signaling type	event based	continuous	event based	event based
RTP packets required	no	yes	no	no
out-band/in-band	out-band	in-band	out-band	out-band
multicast/broadcast applicable	no	no	no	yes

Table 3.2: Comparison of different radio signaling methods

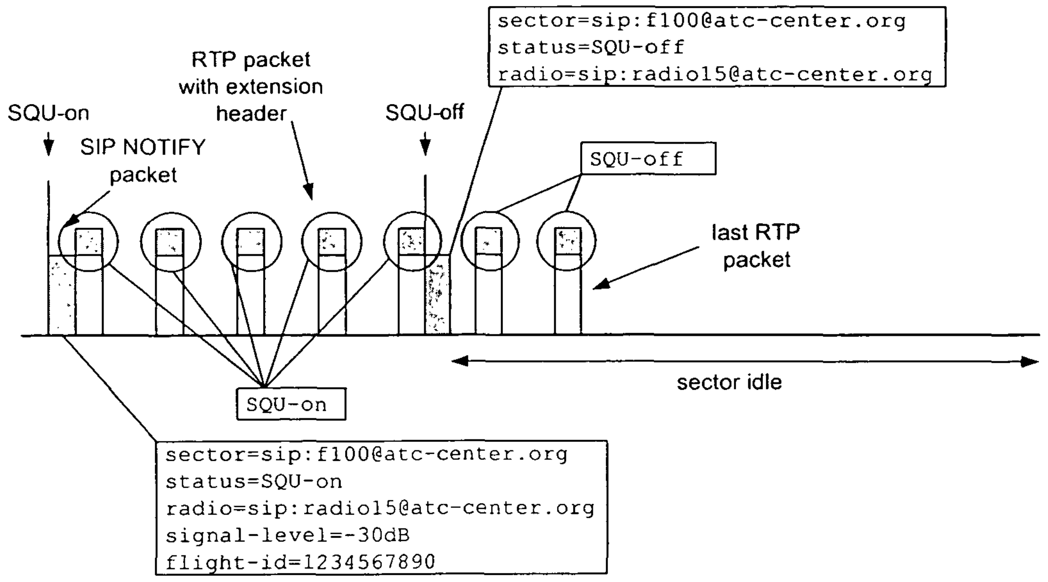

SQU-on
SIP NOTIFY packet
RTP packet with extension header
SQU-off
sector=sip:f100@atc-center.org
status=SQU-off
radio=sip:radio15@atc-center.org
SQU-off
last RTP packet
sector idle
SQU-on
sector=sip:f100@atc-center.org
status=SQU-on
radio=sip:radio15@atc-center.org
signal-level=-30dB
flight-id=1234567890

Figure 3.17: Combination of SIP, explicit RTP and implicit RTP signaling

To enhance fault tolerance, a combination of the presented signaling methods like in figure 3.17 can be used. SIP will be used for sector arbitration and radio signaling related additional data. Implicit RTP signaling will be used to detect failures either in the SIP or in the RTP connection while saving bandwidth by avoiding RTP packets being sent in idle times. Thirdly, explicit RTP signaling is used to mark each RTP packet with a minimal extension header to identify packets. After a radio message, some RTP packets will be sent to explicit signal PTT-off also in the RTP stream, before switching into the idle mode. For a more detailed discussion of fault tolerance refer to chapter 5.

The further sections focus on the SIP based signaling approach and its implementation and verification.

3.6 Putting It Together: The Radio Service

Section 2.5 showed how several basic SIP services and the audio transmission service based on RTP will be combined to build up the radio service. Radio sessions exist in two places: between the operator position and the radio server, and between the radio server and the radio gateway. The operator position requests from the radio server three different services, Status, Rx and RxTx, whereby both, Rx and RxTx, include the Status service. The radio server also request these services from the radio gateway and one more; the Tx service, which is necessary to connect to radio transmitters.

As the session initiation between operator position and radio server on one hand and between radio server and radio gateway on the other hand is nearly identical, the following examples will show only the signaling between operator position and radio server. When requesting the services from the radio server, the operator position has to act according a special sequence as shown in the state diagram in figure 2.9. Every state transition corresponds with a SIP transaction. Table 3.3 shows the state transitions and the corresponding SIP messages.

event	SIP messages
subscribe status granted	`SUBSCRIBE + 200 OK`
unsubscribe status	`SUBSCRIBE (Expires: 0)`
key-in Rx granted	`INVITE (SDP: a=recvonly) + 200 OK`
key-in RxTx granted	`INVITE + 200 OK`
key-out	`BYE`

Table 3.3: State transitions and their corresponding SIP requests

The Status state can be reached by a successful subscription to the PTT/SQU events at the radio server of the corresponding radio sector. To subscribe, the operator position uses the `SUBSCRIBE` method, which is an extension to SIP (refer to section 3.2.1). The status subscription corresponds with messages 1-4 of figure 3.18. The radio server will verify the request according to the local security policies and if the requesting operator position is allowed to subscribe, the radio server accepts the subscription and immediately notifies the subscriber about the current state of the radio sector.

To reach the Rx state, an audio session has to be set up (messages 5-7). This call setup differs slightly from standard SIP call setups. If an operator requests an Rx session, it signals this to the radio server by using a special `type=value` pair in the session description [HJ98]:

```
a=recvonly
```

Hence, the radio server will grant Rx access to this operator and includes an `a=sendonly` line into the SDP answer according to the offer/answer model in [RS02a].

If the operator requests an RxTx session, the operator position application sends a standard `INVITE` to the radio server. When the radio server receives an invitation for a RxTx session, it tries to subscribe the PTT/SQU events from the requesting operator position. This is necessary to receive the PTT-on and PTT-off events from the operator. If the subscription is successful, that means the requesting party is an operator position

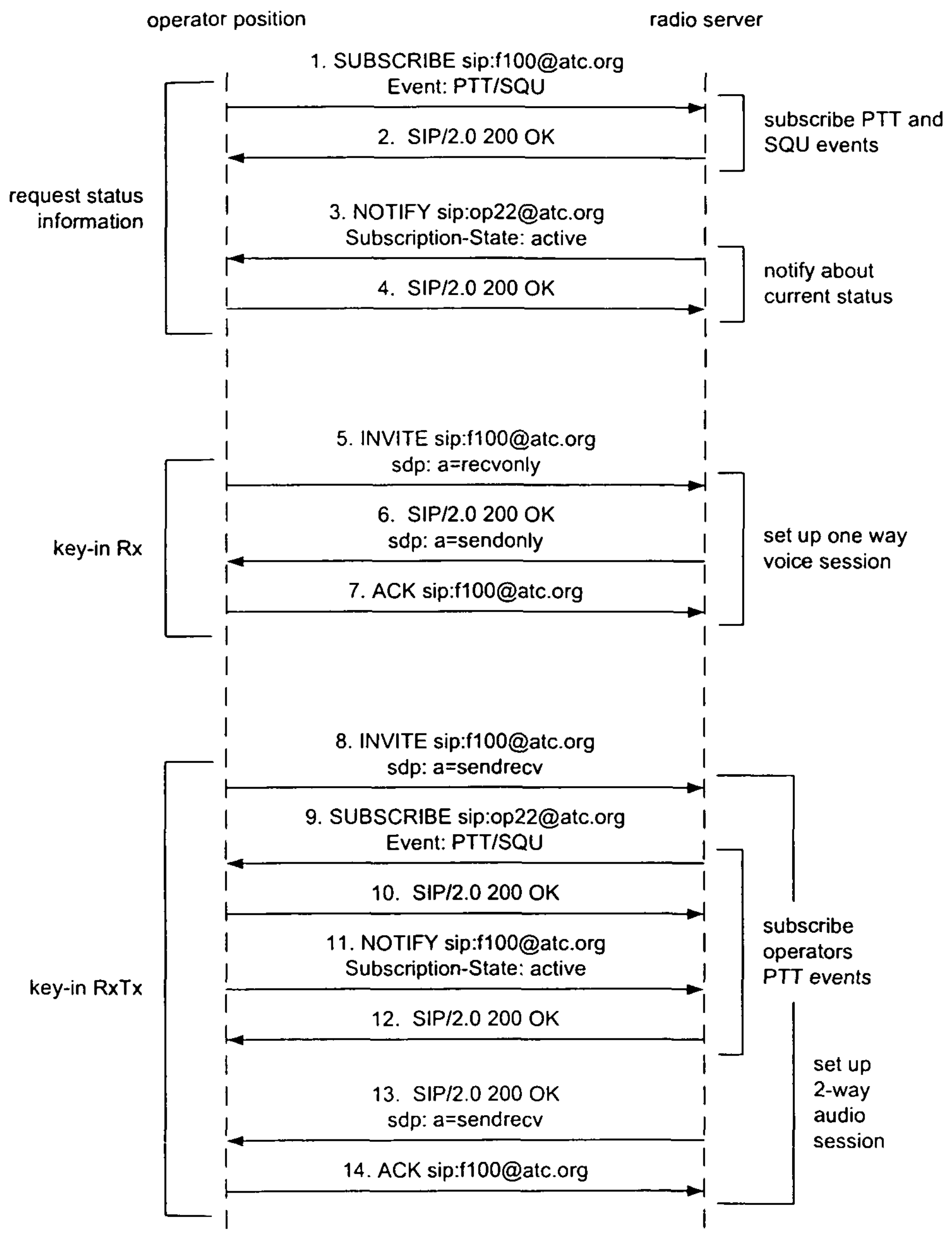

operator position
radio server
1. SUBSCRIBE sip:f100@atc.org
Event: PTT/SQU
2. SIP/2.0 200 OK
subscribe PTT and SQU events
request status information
3. NOTIFY sip:op22@atc.org
Subscription-State: active
4. SIP/2.0 200 OK
notify about current status
5. INVITE sip:f100@atc.org
sdp: a=recvonly
6. SIP/2.0 200 OK
sdp: a=sendonly
7. ACK sip:f100@atc.org
key-in Rx
set up one way voice session
8. INVITE sip:f100@atc.org
sdp: a=sendrecv
9. SUBSCRIBE sip:op22@atc.org
Event: PTT/SQU
10. SIP/2.0 200 OK
11. NOTIFY sip:f100@atc.org
Subscription-State: active
12. SIP/2.0 200 OK
13. SIP/2.0 200 OK
sdp: a=sendrecv
14. ACK sip:f100@atc.org
key-in RxTx
subscribe operators PTT events
set up 2-way audio session

Figure 3.18: Operator position key-in scenario

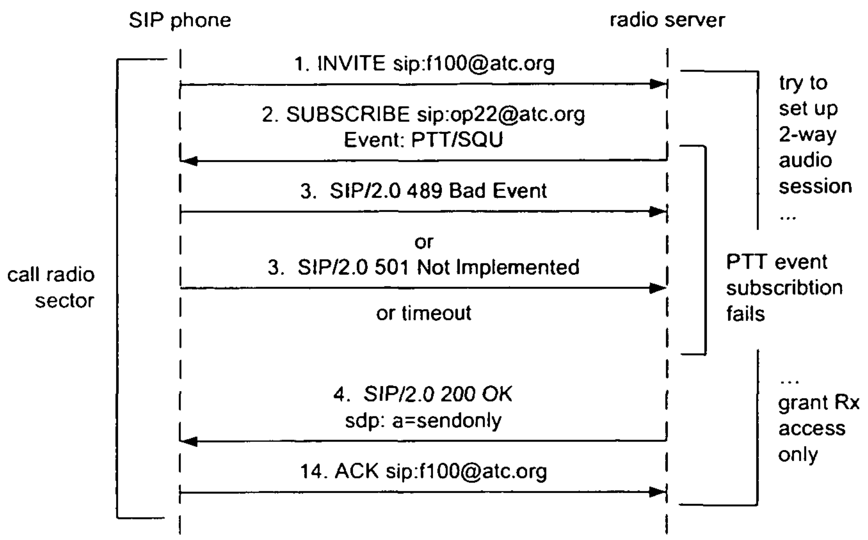


Figure 3.19: Rx compatiblity mode for standard SIP phones

which is capable of radio signaling, it grants RxTx access to the operator (messages 8-14).

Figure 3.19 shows a scenario where a standard SIP phone tries to call into the radio server. As the phone requests a two-way audio session, the radio server again tries to subscribe the PTT events. As the requesting party is not capable of radio signaling, the subscription will fail. SIP UAs which do not know the `SUBSCRIBE` request will answer with a `501 Not Implemented` , wheres UAs which understand the `SUBSCRIBE` request but do not support the PTT/SQU event package will response with a `489 Bad Event` message. The radio server will reject the request or allow an Rx session only (depending on the local security policy). This will be indicated to the caller by sending an `a=sendonly` line in the session description.

If the operator wants to change from Rx to RxTx or vice versa, the controller working position sends a Re`INVITE`. This allows changing the key-in state of the session without tearing down the existing call and setting up a new call. The described key-in procedure allows to key-in a radio channel in Rx mode even with a standard SIP phone.

To be fault tolerant, the radio server has to deal with non-standard-conform implemented UAs. Some of them will answer the `SUBSCRIBE` request with various error codes or do not answer to the request at all. The worst case would be a UA which answers with a `200 OK` message although it is not capable of radio signaling. Therefore, the radio server should not trust the `200 OK` message but should rather wait until the first `NOTIFY` message from the UA, which has to include an `Event: PTT/SQU` and a

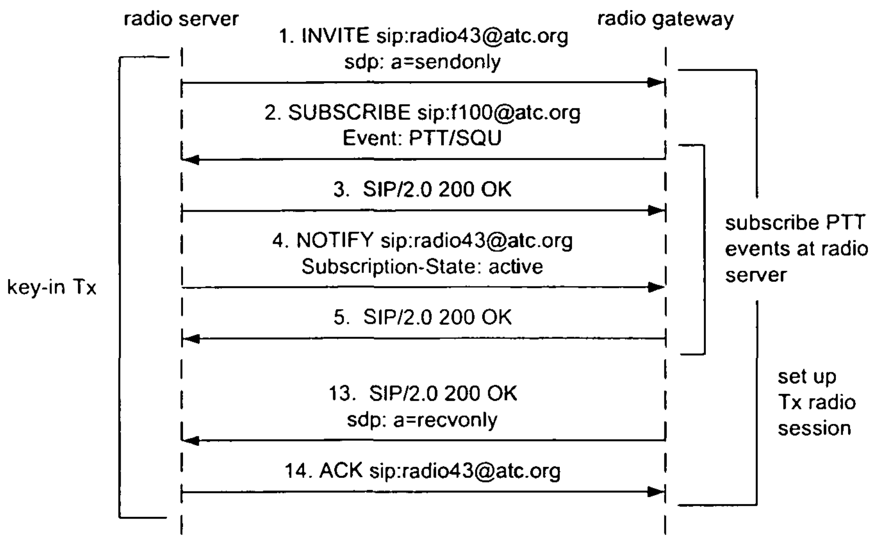

radio server
radio gateway
1. INVITE sip:radio43@atc.org
sdp: a=sendonly
2. SUBSCRIBE sip:f100@atc.org
Event: PTT/SQU
3. SIP/2.0 200 OK
4. NOTIFY sip:radio43@atc.org
Subscription-State: active
5. SIP/2.0 200 OK
13. SIP/2.0 200 OK
sdp: a=recvonly
14. ACK sip:radio43@atc.org
key-in Tx
subscribe PTT events at radio server
set up Tx radio session

Figure 3.20: Tx mode session for radio transmitters

`Subscription-State: active` header.

SIP allows to send requests in-dialog or out-dialog. In the case of a radio session this means that the first `SUBSCRIBE` request establishes a dialog and the following `INVITE` and `SUBSCRIBE` requests can be sent in-dialog using the same call-id and the same `From:` and `To:` tags. The advantage of in-dialog requests is, that they will be sent directly to the other participant without any need to contact the SIP proxy[9]. Nevertheless, some SIP stacks like *dissipate2* [Bila] have problems with in-dialogs when mixing `INVITE`s and `SUBSCRIBE`s. Hence, using separate dialogs will ensure interoperability amongst several SIP stacks. This will lead to maximum of 3 dialogs for a radio session in case of an RxTx session, which consists of one `INVITE`-dialog and two mutual `SUBSCRIBE`-dialogs.

The radio service between the radio server and the radio gateway also supports the Tx service, which enables to transmit radio messages without reception of radio messages. This service is intended to connect to pure radio transmitters. Figure 3.20 shows the message flow, which is similar to the message flow for RxTx sessions except that the radio server adds the `a=sendonly` attribute into the offered SDP and the radio gateway answers with `a=recvonly` in the SDP answer.

Although the radio signaling is essential for the radio service, there are other important parts. The radio server is the central point of the radio service and arbitrates the sector. For this purpose it receives the status events from the radio gateways and the operators, applies the sector arbitration logic and signals the result to all subscribed UAs. This sector arbitration logic is explained in detail in section 4.3.

[9]Except the SIP proxy added a `Record-Route:` header during the dialog establishment.

Chapter 4

Radio Server Implementation and Verification

This chapter addresses the design of the reference implementation of the presented architecture and a verification of the radio service. This includes detailed performance measurements and a comparison of several scenarios. Performance measurements on a SIP proxy conclude this chapter to relativize the results of the performance measurements of the reference implementation. Beforehand, an addressing schema based on SIP-URIs (Uniform Resource Identifier), tel-URIs and ENUM (E.164 Number Mapping) will be presented.

4.1 Addressing Schema

To access various services and communication participants, the demander needs to know the address of the service or participant. A new system with a new technology offers the possibility to introduce a new addressing schema to overcome current limitations, nevertheless backward compatibility to the existing addressing schema is necessary to ease integration and to enable a soft transition from existing systems to the new system.

The addressing schema has to support the addressing of local destinations, of destinations in the PSTN, and of destinations in legacy and proprietary communication networks. Especially the linking of several ATC centers, for example to share radio gateways and radio services amongst several centers, requires a common addressing schema. A SIP based communication environment can cope with multiple resource locaters like SIP-URIs, tel-URIs and ENUM to address various services and destinations. Therefore, SIP is not restricted to a certain type of network or communication service.

4.1.1 SIP-URI

SIP URIs have a form similar to email addresses, typically consisting of a user name and a domain to which the user belongs. To distinguish SIP addresses from email addresses, SIP-URIs starts with `sip:` and SIPS-URIs starts with `sips:` . A SIPS URI points to the same destination as a SIP-URI, but indicates that the resource has be contacted securely. As SIP-URIs may consist of letters and numbers, they allow to address a resource in an intuitive manner, e.g. the SIP address `sip:k.darilion@ict.tuwien.ac.at`, where ict.tuwien.ac.at is the domain of the company K. Darilion is working for. Based on the configuration of the authoritative SIP proxy for a certain domain, user names which consists of numbers only, can be treated as phone numbers and will be routed according to a local dialing plan. These phone numbers can be local numbers or global phone numbers, identifiable by prefixes, e.g. `sip:+4315880138422@ict.tuwien.ac.at` will be treated by the SIP proxy as global valid E.164 phone number [Inta] and forwarded to a SIP-PSTN gateway. Possibly, the SIP proxy tries to resolve the phone number into a SIP-URI using ENUM (refer section 4.1.3) before forwarding.

4.1.2 TEL-URI

The tel-URI [VS00] addresses resources in the telephone network. The address starts with `tel:` and includes the phone number and the context in which the phone number is valid. The following examples explain the use of the phone context:

`tel:+43-1-58801-38422` This is a globally valid E.164 phone number identifiable through the + sign in front of the number. The "-" between the numbers are only to improve the readability and will be ignored by the processing system. As this is a worldwide valid number no phone context is necessary.

`tel:38422;phone-context=TU-Vienna` This is a phone number which is only valid in the context of the Vienna University of Technology, e.g. an extension in the PBX system of the Vienna University of Technology. This does not imply that this resource is only availably in the area of the TU-Vienna, it can also be reached from outside, if the routing nodes know how to resolve the tel-URI, for example by inserting TU-Vienna's phone number in front of the extension and changing the name of the context to `tel:58801-38422;phone-context=Vienna`.

4.1.3 ENUM

The ITU-T Recommendation E.164 [Inta] is the international public telecommunication telephony numbering plan. An E.164 phone number consists of a maximum of 15 digits.

The ENUM protocol [Fal00] describes how E.164 addresses can be translated into a domain name which can be resolved by domain name system (DNS) [Moc87] to look up the URIs [BLFM98] of several services concerned with this phone number. This protocol can be used to resolve E.164 phone numbers into SIP-URIs [PLYC03]. At first, the E.164 phone number must be translated into a DNS name, as shown in table 4.1.

E.164 number	+43-1-58801-38422
1. remove all non-number characters	4315880138422
2. reverse the order	2248310885134
3. insert dots between the numbers	2.2.4.8.3.1.0.8.8.5.1.3.4
4. append the .e164.arpa domain	2.2.4.8.3.1.0.8.8.5.1.3.4.e164.arpa

Table 4.1: Using ENUM to resolve an E.164 phone number into a domain

Afterwards, Naming Authority Pointer records (NAPTR) [MD00] are used to identify available ways or services for contacting a specific resource identified by the E.164 number. Figure 4.1 shows the NAPTR record to resolve the E.164 address from table 4.1 into a SIP URI with higher priority, into an email address, and into a telephone URI. Usually, then the client has to further resolve the received URI, e. g. a SIP URI can be resolved using the DNS SRV records [GVE00] to find the domain name of the next SIP proxy. This domain name then will be resolved to an IP address.

```
$ORIGIN 2.2.4.8.3.1.0.8.8.5.1.3.4.e164.arpa.
  IN NAPTR 100 3 "u" "E2U+voice:sip"    "!^.*$!sip:k.darilion@ict.tuwien.ac.at!"    .
  IN NAPTR 101 3 "u" "E2U+voice:sip"    "!^.*$!sip:darilion@pernau.at!"    .
  IN NAPTR 105 4 "u" "E2U+voice:tel"    "!^.*$!tel:+4315880138422!"             .
  IN NAPTR 105 1 "u" "E2U+email:mailto" "!^.*$!mailto:darilion@ict.tuwien.ac.at!" .
```

Figure 4.1: NAPTR records for two SIP, one email and one telephone resource

The main purpose of ENUM is to find contact information about a resource, which is addressed by an E.164 phone number, but like a SIP client not connected to the PSTN network. Furthermore it is used to check if an E.164 addressed destination can also be reached via IP communication techniques like SIP or H.323.

4.1.4 Numbering Plan for ATC centers

SIP allows the use of intuitive addresses using SIP-URIs with alphanumerical characters, but it is not always reasonable to use them, because ATC operators are used to dial phone numbers. The user interface of typical working positions offers speed dial buttons

destination/service	number
operator of a certain sector	1xxxxx
a particular operator	2xxxxx
a particular operator position	3xxxxx
a particular standard phone	4xxxxx
a particular radio sector/frequency	5xxxxx
a particular radio (gateway)	6xxxxx
voice enabled services (voicemail ...)	7xxxxx
external (gateway) call xx = particular gateway yy = number in the particular network	8xxyyyyyyyyy
PSTN calls	9xxxxxxxxxxx

Table 4.2: Dial plan

and a dial pad. Therefore, using addresses, which include letters would also require to change the GUI. Furthermore, devices with limited input capabilities like phones are also connected to the communication system. This requires that every resource can be addressed by a phone number. Anyway, the SIP based addressing schema allows the use of intuitive addresses, which can be used with operator positions which support keyboards and voice recognition. Those addresses will be aliases which will be resolved by the SIP proxy to the real address. Therefore, the following addressing schema also supports intuitive names.

SIP was designed for en-bloc signaling, whereas many communications networks, especially in Europe, were designed for overlap signaling. This may cause problems when calling SIP destinations from the PSTN [CRPP03]. If an operator uses the dial pad to enter a phone number on a SIP UA, the operator has to indicate to the application that the dialed number is complete. This can be done by pressing a certain button (e.g. the *Dial* button) or by doing nothing, and after a timeout the SIP user agent assumes the entered number as complete. To avoid that the operator has to press the *Dial* button, the SIP UA may use the *early dial* feature. A UA which uses early dial sends an `INVITE` message to the SIP proxy every time the user enters another digit of the phone number. The SIP proxy analyzes the phone number in the request URI, and responds with a `484 Address Incomplete` message as long as the number does not fit into the local dial plan. To implement overlap dialing using the "early dial" feature, there must not be a number which is the prefix of another number, e.g. there must not be the number 123456 and the number 1234 as the second number is part of the first number.

Therefore, all destinations (or at least each destination within a certain service group), which might be addressed by a phone number, should use a fixed number of digits.

The kind of service might be differentiated by the first digit as shown in table 4.2. For example, a phone call to a certain operator (regardless of its current operator position) is done be calling the SIP-URI `sip:212345@atc.org`.

Note: For calls into the PSTN network, the early dial feature can not be used if the call is directed into a network, which uses a dial plan without a fixed number of digits, as the proxy does not know, if a dialed number is complete or not.

ENUM (refer section 4.1.3) should be used by the SIP proxy for routing calls from the PSTN to pure SIP destinations to identify the SIP address. Additionally, ENUM should be used for calls originating from SIP UAs to E.164 numbers to detect if the called destination is also reachable by a SIP-URI. Additionally, SIP-CGIs and the call processing language (CPL) can be used to extend the routing logic in the SIP proxy on a per user basis [RLS99].

As SIP allows to address resources with intuitive addresses by using alphanumerical characters, this can be done in addition to the plain number based addressing to ease operation. For instance, the operator John Q. Public should be reachable by a SIP-URI which consists of the operators name in the user part, i.e. `sip:john.q.public@atc.org`. Table 4.3 shows the previous dial plan with intuitive names instead of numbers.

destination/service	SIP-URI
operator of a certain sector	`sip:operator.f100@atc.org`
a particular operator	`sip:john.q.public@atc.org`
a particular operator position	`sip:op22@atc.org`
a particular standard phone	`sip:phone.reception@atc.org`
a particular radio sector/frequency	`sip:f100@atc.org`
a particular radio (gateway)	`sip:radio15@atc.org`
voice enabled services (voicebox ...)	`sip:voicebox@atc.org`
generic SIP call	`sip:user@domain`

Table 4.3: Intuitive addressing schema

4.2 Dissipate SIP stack

The *dissipate* SIP stack was developed as integral part of *KPhone* [Bilb], an open source SIP softphone for Linux. It was preferred against other SIP stacks as, at the beginning of this work, it was the only freely available open source SIP stack, which offered a high level API and support for the SIP specific event notification. *Dissipate* is written in

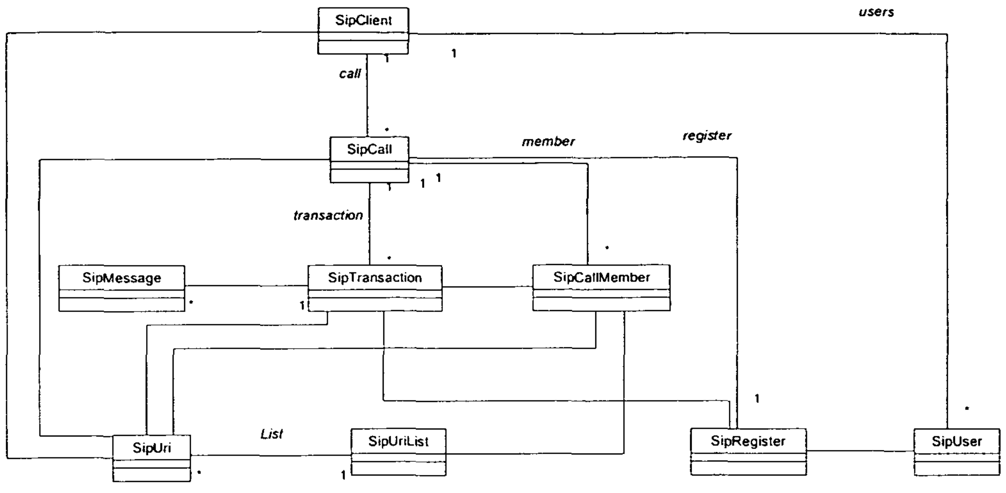


Figure 4.2: Class diagram of the *dissipate* SIP stack [Bila]

C++ and uses the Qt library[1] [Tro]. *Dissipate* offers a high level API, i.e. it handles retransmission of SIP messages if UDP is used as transport protocol, keeps track of the call state of all ongoing calls, and handles all the signaling. Thus, the application does not have to care about sending the right SIP messages at the right time as this is done by *dissipate*. For example, if the application hangs up a call, it does not have to care about sending a BYE or a CANCEL message, depending on the call state, as this is done by the stack. *Dissipate* supports RFC 3261 [RSC+02] and instant messaging as defined in RFC 3428 [CER+02]. *Dissipate* also supports the SIP specific event notification, but only for transmitting presence information. Therefore, to use dissipate as SIP stack for all the SIP UAs of the radio service, the presence part of the stack was modified to support PTT/SQU events as defined in section 3.2.2.

Figure 4.2 shows the class diagram of the SIP stack. The main class is the SipClient, which manages all ongoing calls and all local SIP users. Furthermore, it polls for incoming SIP messages, processes them and forwards them to the corresponding objects. The SipUser objects represent local users, for which the stack is responsible. The SipCall represents a call to another party, which is represented in the SipCallMember class. SipTransaction reflects the currently active SIP transactions which consists of several SipMessages. The stack notifies the application about incoming calls and state changes in active calls via signals, which can be subscribed by the application. Furthermore, the SipRegister class peforms the registration at several SIP proxies.

The next sections shows excerpts of the design and implementation of SIP applications

[1]Amongst others, Qt offers GUI classes, string parsing and manipulation functions, timers, and inter-object communication using signals and slots.

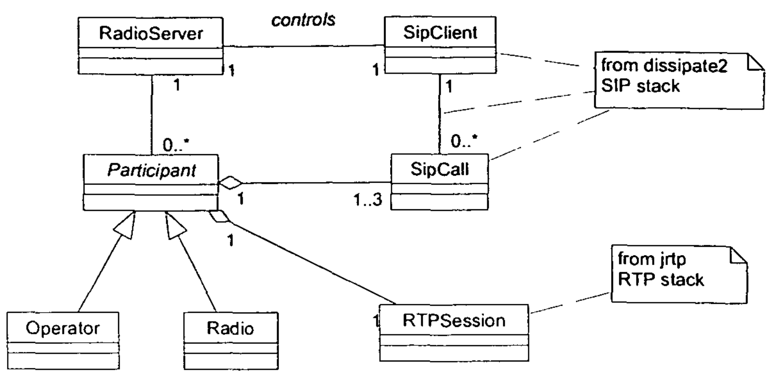


Figure 4.3: Class diagram of the radio server

at the operator position, the radio server and the radio gateway, and how they interact with the SIP stack.

4.3 Radio Server Implementation

At startup, the radio server registers its sector[2] at the SIP proxy. After a successful registration, the radio server connects to the radio gateways in the configured mode (Rx, Tx, or RxTx). After setting up the connections to the radio gateways, the radio server accepts connection requests from other clients, i.e. operator positions and standard SIP phones. The radio server is designed as shown in figure 4.3. The participant class handles all the SIP and RTP connections to the other participants. Although operators and radio gateways are handled similar, they have some significant differences. For instance, the Operator class is passive, therefore, it never tries to connect to an operator position, whereas the Radio class is active, as the connection to the radio gateway is always initiated from the radio server. This is a result of the client-server architecture presented in section 2.6. Therefore, the Operator and Radio class inherit their common functionality from the Participant class. The SipCall objects represent the standard phone call and the two mutual subscription dialogs. The RTPSession object handles the reception and sending of RTP packets for the standard phone call.

Figure 4.4 shows the state diagram of the RadioServer class. If there is no sector activity, the class is in the idle state. On incoming SQU signals, that means at least one of the radio gateways received a radio message and sent a SQU-on notification to the radio server, the state transits to receiverVoting. In this state, the RadioServer waits for further

[2]The SIP URI of the sector and the SIP addresses of the radio gateways have to be configured in advance.

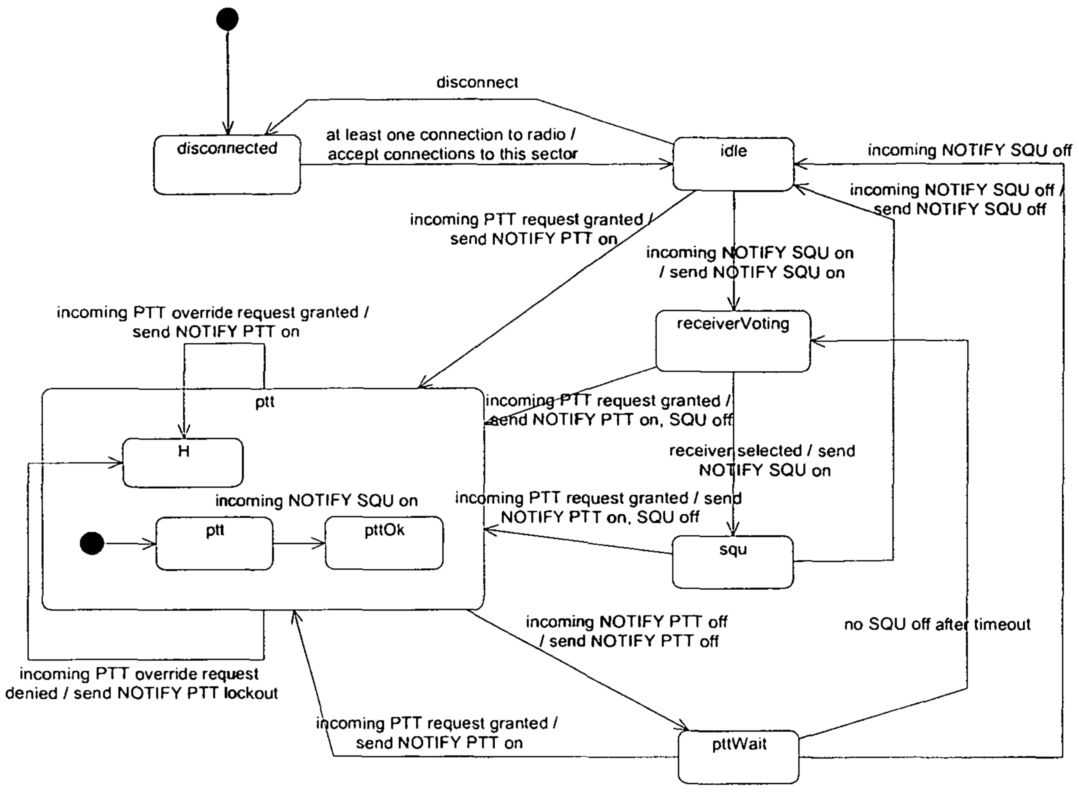

disconnect
disconnected
at least one connection to radio / accept connections to this sector
idle
incoming NOTIFY SQU off
incoming NOTIFY SQU off / send NOTIFY SQU off
incoming PTT request granted / send NOTIFY PTT on
incoming NOTIFY SQU on / send NOTIFY SQU on
receiverVoting
incoming PTT override request granted / send NOTIFY PTT on
ptt
incoming PTT request granted / send NOTIFY PTT on, SQU off
H
receiver selected / send NOTIFY SQU on
incoming NOTIFY SQU on
incoming PTT request granted / send NOTIFY PTT on, SQU off
ptt
pttOk
squ
incoming NOTIFY PTT off / send NOTIFY PTT off
no SQU off after timeout
incoming PTT override request denied / send NOTIFY PTT lockout
incoming PTT request granted / send NOTIFY PTT on
pttWait

Figure 4.4: State diagram of the RadioServer class

SQU-on notifications and uses a defined algorithm to choose the radio gateway with the best reception[3], then the state transits to squ and stays in this state until the selected radio gateway sends a SQU-off notification. In any of these states, an incoming PTT-on request triggers a state transition into the ptt state.

The ptt state is divided into two sub-states. As discussed in section 2.7, every radio transmission also triggers an incoming radio signal, which causes SQU-on notifications, which triggers the transition into the pttOk state. This incoming signal can be used to verify the operation of the radio transmitter and receiver. If there is no SQU-on signal received within a certain period of time, the radio server may treat this as a failure and should react using one of the failover solutions discussed in section 5.3 to restore a failure-free status. Furthermore, the operator has to be informed that the transmission of the radio message may have failed.

If the RadioServer receives a PTT request from another Operator although it is already

[3]The prototype uses a very simple algorithm: the radio gateway which signals first will be chosen.

in the ptt state, it will check the priority of the request. If the priority of the incoming request is higher than the priority of the ongoing radio transmission, the incoming request overrides the current radio transmission and the radio server signals to all operators that the currently speaking operator has been overridden. At the end of the radio transmission, when the currently active operator signals PTT-off, the RadioServer transits into the pttWait state. As the begin of a radio transmission always triggers an incoming radio message, the end of the radio transmission has to trigger the end of the incoming radio message, which will be indicated by a SQU-off from the radio gateway. Therefore, the non-appearance of the SQU-off notification within a certain period of time indicates that there is an overlapping radio message from a pilot and the RadioServer immediately transits into the receiverVoting state.

The state diagram showes only the state transitions in failure-free activity. Additional state transitions may occur, if the radio server detects failures of the connected clients.

The second task of the radio server besides PTT/SQU signaling is the forwarding of the audio stream. The forwarding unit of the RadioServer will accept RTP packets from the current active participant, an operator or a radio gateway, and forward them as described in table 4.4.

stream from	**forwarding to**
radio gateway	- all Rx or RxTx subscribed operator positions
Tx subscribed operator	- all radio gateways in Tx or RxTx mode - all Rx or RxTx subscribed operator positions except the one that sends

Table 4.4: RTP forwarding rules

4.4 Radio Gateway Implementation

The radio gateway is the gateway between the IP based packet switched communication infrastructure and circuit switched radio. Implementing a prototype system with a real gateway would also require a radio transmitter, a radio receiver, and antennas. To simplify the prototype implementation, a radio gateway simulator was used as shown in figure 4.5. The simulator uses real voice conversations between operators and pilots, recorded at several airports in the USA [NIS], which are stored in WAV files. The simulator reads the radio messages from the WAV files and sends them, together with the SQU-on and SQU-off notifications to the radio server. Radio messages from the radio server will be played back at a loudspeaker using the sound interface of a standard PC.

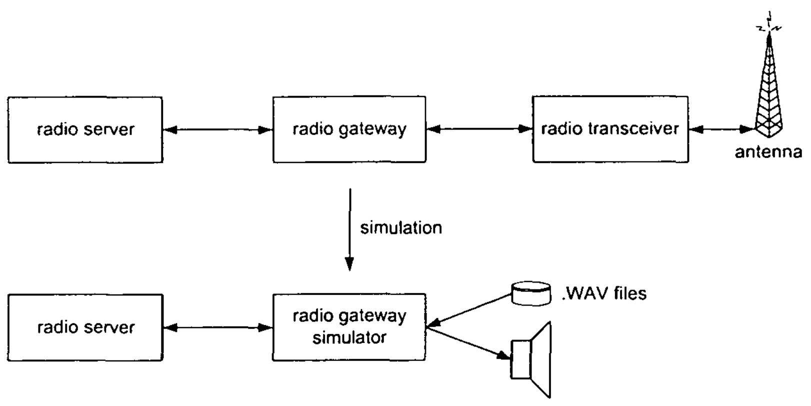


Figure 4.5: Radio gateway simulator

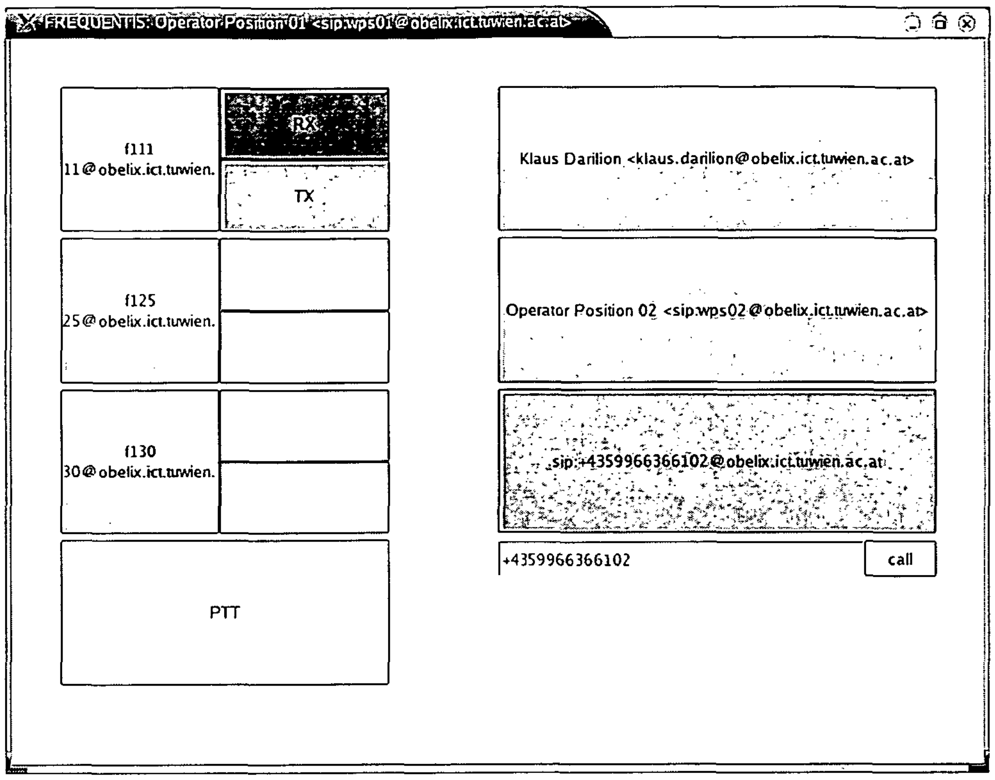

FREQUENTIS: Operator Position 01 <sip:wps01@obelix.ict.tuwien.ac.at>
f111
11@obelix.ict.tuwien.
RX
TX
f125
25@obelix.ict.tuwien.
f130
30@obelix.ict.tuwien.
PTT
Klaus Darilion <klaus.darilion@obelix.ict.tuwien.ac.at>
Operator Position 02 <sip:wps02@obelix.ict.tuwien.ac.at>
sip:+4359966366102@obelix.ict.tuwien.ac.at
+4359966366102
call

Figure 4.6: Screenshot of the operator position application

4.5 Operator Position Implementation

The operator position application, called *WPos*, is the only application of the radio service, which needs a user interface. The WPos was developed under Linux and uses the Qt library for the graphical user interface. Figure 4.6 shows a screenshot of the WPos application. The left side of the user interface is the radio part, which allows to subscribe to pre-configured radio sectors and signals radio activity, whereas the right part consists of direct access (DA) buttons to initiate standard phone calls.

The GUI is based on the entity-boundary-control concept [HK99]. The class diagram of the operator position application is shown in figure 4.7. The boundary class (e.g. the SectorButton) is the graphical representation of the entity class (e.g. SectorEntity) which handles the SIP based signaling. The controller class (SectorControl) controls the entity and boundary objects and initiates certain tasks on incoming events received from the entity object and when the user presses the buttons.

The operator position is the only place where audio processing is necessary. The operator

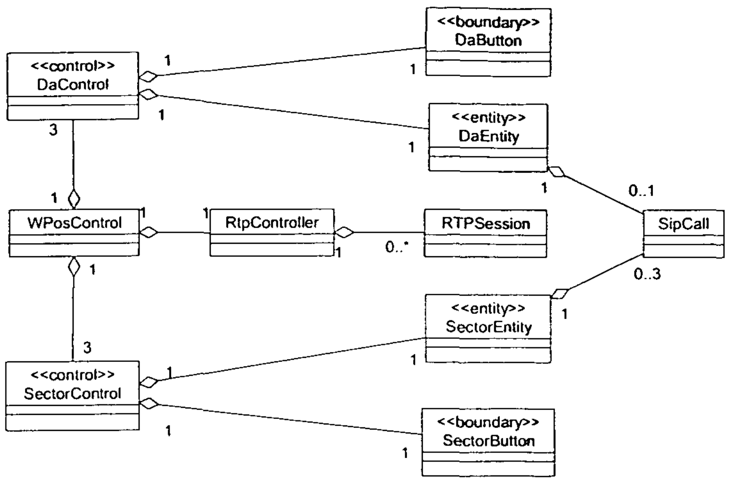


Figure 4.7: Class diagram of the operator position application

position receives the RTP streams from all the radio servers it has subscribed to and from the participants of ongoing phone calls. The RtpController mixes all incoming audio streams and sends the result to the sound device. In the opposite direction, it records from the attached microphone and sends RTP streams to the participants. Thus, it uses an RTPSession for each participant.

As each operator position mixes the audio stream itselfs, multiple receptions may occur as shown in figure 4.8. In this scenario, operator 1 and operator 2 are connected to the same radio sector. Therefore, every radio message sent from operator 1 is forwarded from the radio server to operator 2. If there is also an active phone call between both operators, all the voice recorded at operator 1 would also be sent to operator 2 directly. At operator position 2, it would be possible to detect that both receiving streams come originally from the same operator if operator position 1 (OP1) uses the same synchronization source identifier (SSRC) [SCFJ03] for both RTP streams and the radio server includes this identifier into the list of the contributing sources (CSRC) in the RTP stream sent to operator 2. Nevertheless, the packets from both streams arrive at different times at OP2 (due to the additional delay introduced by the radio server) and have different timestamps. Therefore, OP2 would have to compare the payload of both streams to identify doubly received packets.

A much simpler approach is to block the second stream at the originating operator position. During the transmission of a radio message, i.e. the PTT button is pressed, all currently active calls will be set inactive (on-hold) and no RTP packets are sent on these streams.

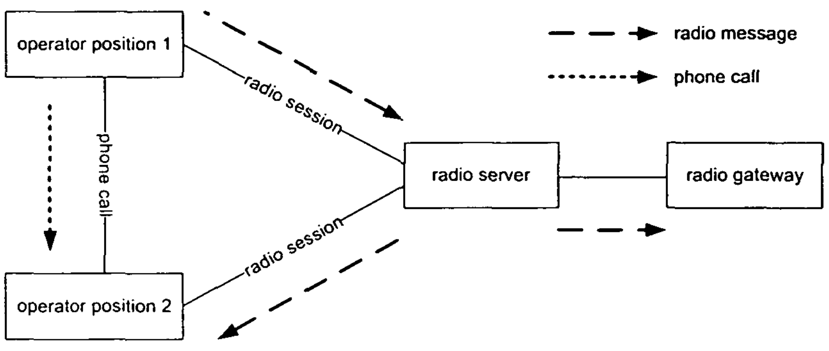


Figure 4.8: Multiple receptions from the same operator

4.6 Signaling Delay Measurements

The operators and pilots have to obey the listen-before-talk procedure before transmitting a radio message. Nevertheless, it can happen that an operator and a pilot transmit a radio message at the same time, which will lead to a collision and both radio messages are useless and have to be repeated. The higher the signaling delay between the operator position and the radio gateway, the higher is the probability of collisions. Furthermore, if the operator speaks immediately after pressing the PTT button, it is possible that the first syllables are lost because of the transmitter attack-time (refer to section 3.1). Therefore, it is important to signal PTT and SQU events very fast through the system to keep the probability of collisions low and to avoid the loss of syllables.

To evaluate the feasibility of the radio architecture, the event distribution of the radio server was analyzed and the signaling delays were measured. To relativize the measurements, the forwarding delay of a SIP proxy in a forking scenario was measured. This scenario is similar to the notification distribution scenario and can be used to compare the results.

The delay introduced by the radio server for PTT/SQU signaling will probably be higher than for the audio streams, as the processing of the signaling is more complex. The PTT/SQU signals are SIP messages, which will be parsed by the SIP stack to identify the SIP request, corresponding transactions, and dialogs. Furthermore, the radio server application has to arbitrate the sector according to the incoming notifications, configure the RTP forwarding unit, and finally it signals the decision to the operator positions and the radio gateway. This processing takes more time than the reception and transmission of RTP packets, as they do not have to be matched to ongoing transactions and the RTP is simpler than SIP. This can be verified by comparing the SIP signaling delay with the RTP forwarding delay in the following sections.

In all scenarios UDP was used as transport protocol, to avoid additional delay due to

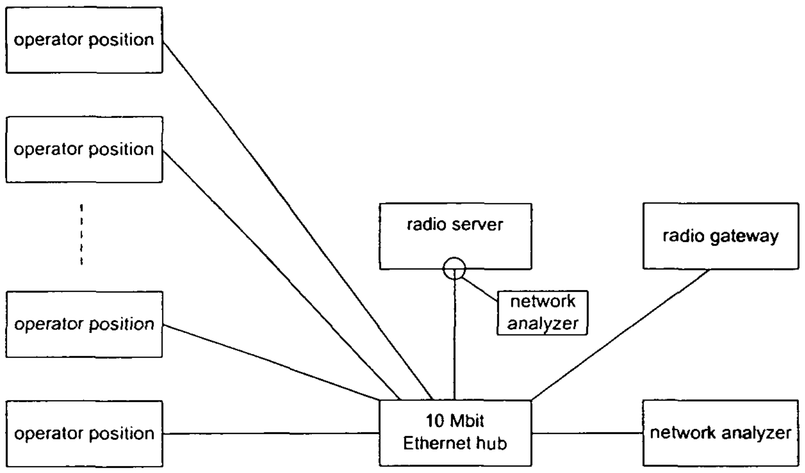


Figure 4.9: Setup for signaling delay measurements

the handshake of TCP during connection establishment and possible buffering in the TCP stack. Figure 4.9 shows the setup for the measurements. The network in the test setup was a 10 Mbps Ethernet collision domain and all PCs were connected to the same hub[4]. All PCs were configured not to use the DNS to resolve domain names but the domains were entered in the local *hosts* file. This ensured, that there were no additional delays caused by domain name lookups.

Basically, the test setup recorded the timestamps of all signaling packets sent to and received from the radio server. The difference between the timestamps of certain packets is the signaling delay. Every scenario was measured at least ten times and the average signaling delay was calculated. The packet flows were captured by network analyzers. On the radio server, Ethereal, a software network analyzer, recorded the packets. Furthermore, a dedicated hardware network analyzer[5] was connected to the network to verify the results. This setup also allows to measure the delay introduced by the network interface card in the radio server as the difference between the signaling delay measured by the external network analyzer, and the signaling delay measured by the software network analyzer.

The radio server runs on a Celeron 1.8 GHz PC with Linux 2.4.20. As SIP proxy, the SIP Express Router on an Athlon 800 MHz PC with Linux 2.4.20 is used.

[4]Although there is a 100 Mbps Ethernet available in the laboratory, a 10 Mbps Ethernet collision domain was used as the 100 Mbps network is switched, which makes packet capturing more complicated. Nevertheless, measurements, which reached the limit of the 10 Mbps network, were repeated in the switched 100 Mbps Ethernet.

[5]Finisar THG and Finisar Surveyor

4.6.1 PTT Signaling: Results and Analysis

In the PTT signaling scenario, shown in figure 4.10, the operator requests sector access by sending a PTT-on event to the radio server. If the radio sector is idle, the radio server will grant access to the sector and forwards the PTT-on event to all connected radio gateways in Tx or RxTx mode, to activate the radio transmitters. Furthermore, the radio server sends the PTT-on event to all subscribed operator positions, even to the PTT-requesting operator, which provides a confirmation for the sector request.

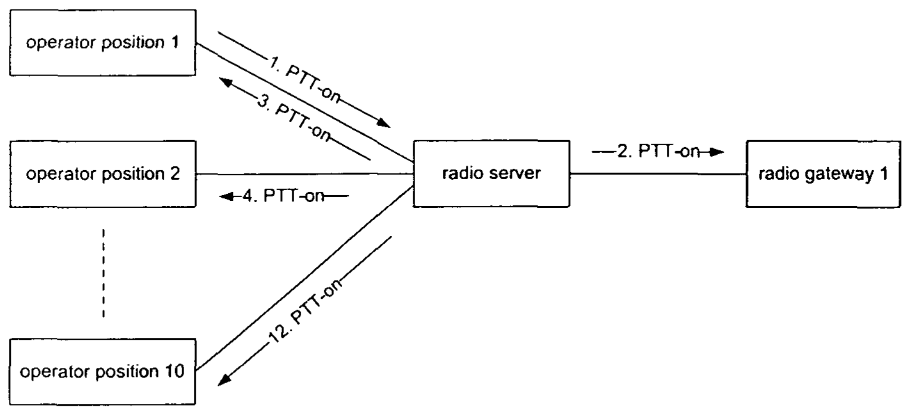


Figure 4.10: PTT signaling scenario

The setup consisted of 9 operator positions and one radio gateway connected to the radio server. Figure 4.11 shows the detailed SIP message flow including the `200 OK` responses to the `NOTIFY` requests. In reality, the `200 OK` messages 4a–4k will overlap the `NOTIFY` messages 3a–3k, but they are drawn at the bottom of the figure for clarity.

Following, all delays are explained in order of their occurrence:

t_{send} defines the time from the application's request for a `NOTIFY` message to the SIP stack till the `NOTIFY` message leaves the PC and is on the Ethernet. This time period includes:

1. The generation of the `NOTIFY` message in the SIP stack.
2. The processing delay in the UDP/IP stack of the operating system.
3. The processing delay in the network interface card and its device driver.

For a detailed explanation of the delays introduced by the operating system and the network card refer to [Kam01][KG03].

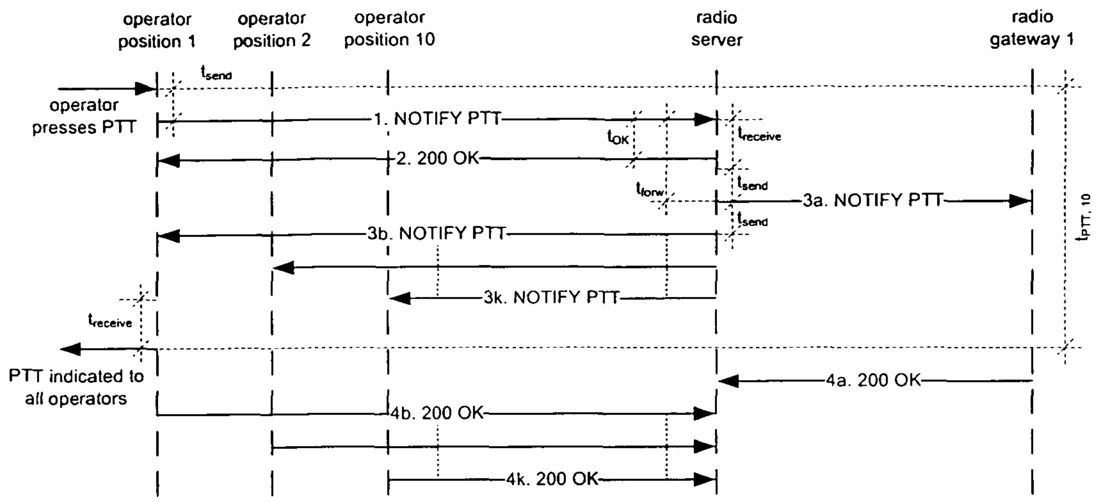


Figure 4.11: PTT signaling delay

t_{OK} is the time between the incoming NOTIFY request and the 200 OK response. The 200 OK message is produced by the *Dissipate* SIP stack without any interaction from the application. So, this time again includes the processing delay in the network cards and the IP stack.

$t_{receive}$ is the time period from the reception of the incoming NOTIFY message until the notification event is available in the radio server application.

t_{forw} is the time between the incoming NOTIFY request and the first forwarded NOTIFY request. $t_{forw} = t_{receive} + t_{send}$

$t_{PTT,n}$ is the time needed to inform n nodes (radio gateways and operator positions) about a PTT/SQU event.

As the measurements are done on the network layer only, it is not possible to measure $t_{receive}$, but it can be derived from t_{forw} and t_{send}. Although the measurements were done only against the radio server, it is assumed that $t_{receive}$ and t_{send} at the operator position and the radio gateway are identical to the radio server as the same hard- and software is used.

Table 4.5 shows average values and the standard deviation for the signaling delays of the radio server implementation. Please note that the measurements are done in a pure signaling scenario without any RTP relaying. The 200 OK response to the incoming notification is sent by the SIP stack itself and takes about 5.0 ms. The generation of a NOTIFY transaction for each subscriber takes about 1.6 ms and is also done by the tb

processing time	average (ms)	standard deviation (ms)	messages
t_{OK}: send 200 OK	5.0	0.074	1–2
t_{forw}: receive and send NOTIFY	8.1	0.28	1–3a
t_{send}: send NOTIFY	1.6	0.31	3a–3b

Table 4.5: PTT signaling delays

SIP stack itself. The required time to signal a PTT event to n nodes can be calculated with equation (4.1). The generation of the notification at the operator position plus the processing at the radio gateway takes approximately the same time as the forwarding at the radio server ($t_{send} + t_{receive} = t_{forw}$). Therefore, t_{forw} is doubled. Forwarding of the PTT signal to one radio gateway and nine operator positions ($n = 10$) takes at least 30 ms.

$$t_{PTT,n} = t_{forw} * 2 + t_{send} * (n - 1) \tag{4.1}$$

4.6.2 SQU Signaling: Results and Analysis

Figure 4.12 shows a SQU signaling scenario. The SQU signaling is quite identical to the PTT signaling scenario in terms of processing needs and message flow. Therefore, the results from the PTT signaling scenario in section 4.6.1 and equation (4.1) can be used also for SQU signaling.

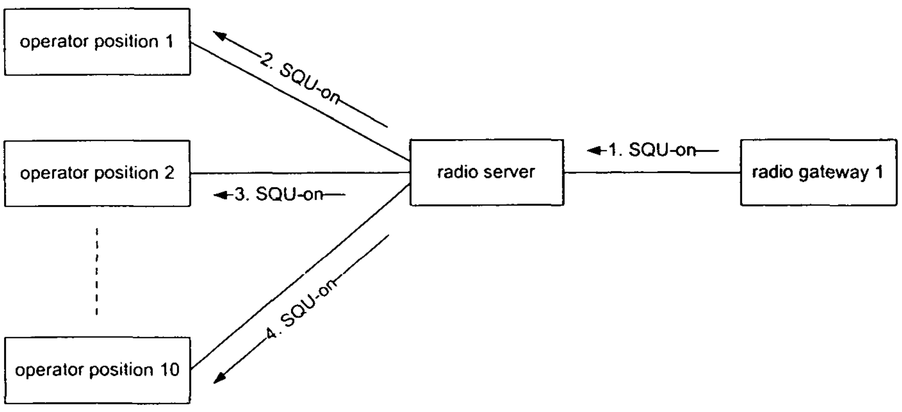


Figure 4.12: SQU signaling scenario

4.6.3 RTP Forwarding: Results and Analysis

The RTP switching unit of the radio server is very simple. For each connected radio gateway and operator position, there is a dedicated port for receiving the RTP stream. If an RTP packet is received, the application lookups the participant which corresponds to the port on which the packet was received. If the sender of the RTP packet is the currently active radio sector holder, than the RTP packet will be forwarded accordingly to the forwarding rules, if not, the RTP packet will be discarded. As RTP requires no transaction matching, acknowledge packets and retransmission mechanism, the audio forwarding is faster than the PTT/SQU forwarding.

Using unicast for the audio transmission results in the same characteristic as the PTT/SQU forwarding, the more operators and radio gateways are connected to the radio server, the higher is the forwarding delay. The additional RTP delay was about 0.2 ms per packet. The RTP packets included 20 ms of voice encoded with G.711. Adding the various protocol headers as shown in table 4.6 leads to a minimum delay of 0.19 ms when using 10 Mbps Ethernet as derived in equation (4.2). Therefore, the introduced RTP delay is a limitation of the Ethernet. This also shows that the SIP forwarding delay is much higher than the RTP forwarding delay.

Type	Size (Byte)
RTP payload G.711 20 ms	160
RTP header	12
UDP header	8
IP header	20
Ethernet Version II interframe gap	12
Ethernet Version II preamble	8
Ethernet Version II header	14
Ethernet Version II checksum	4
sum	**238**

Table 4.6: RTP packet overhead budget

$$t_{min} = \frac{238\,\text{Byte}}{10\,\text{Mbps}} = 0.19\,\text{ms} \tag{4.2}$$

4.6.4 SIP proxy forking performance

If multiple contacts are registered for the same address-of-record (AOR), a SIP proxy forks requests for this AOR to all registered contacts. If the proxy is transaction stateful, it has to maintain the state of all forwarded requests. In the following setup, ten contacts for one address-of-record were registered and a MESSAGE request was sent to this address, which in turn was forwarded to all ten contacts. The proxy sent the messages approximately every 0.48 ms, which is nearly the theoretical maximum in 10 Mbps Ethernet for a packet size of 576 Byte. Therefore, the test was repeated in a switched Fast Ethernet, which reduced the packet interval to one tenth. Capturing on the PC running the SIP proxy revealed that the SIP proxy produces the messages every 0.02 ms. As a consequence, even Fast Ethernet is to slow to send the packets as fast as the proxy generates them. As the delay introduced by layer 2-4 (Ethernet, device driver, IP, UDP) is identical for the radio server delay and the SIP proxy delay, the crucial factor is the processing delay in the SIP stack.

The radio server can not be as fast as a SIP proxy as it has to create separate transactions for each notification whereas the SIP proxy may treat the forking scenario as one transaction with several branches. The radio server also has to maintain the call state of every connected subscriber. Nevertheless, this comparison showed that there is a high potential to speed up the radio server and the event distribution by using a faster SIP stack.

4.6.5 Discussion of the results

In [Eurb], Eurocontrol suggests a maximum PTT/SQU signaling delay of 25 ms (local). This is hard to achieve, even in traditional circuit switched systems. The prototype implementation achieved this requirement only with six or less connected nodes. Therefore, the SIP stack must be faster, especially the generation of new transactions and requests. In the current configuration, the system can be used only for small systems like airport tower systems, which typically consists of up to five operator positions and up to five radios.

Another way to avoid sequential request generation for each subscriber would be the use of multicast or broadcast. SIP uses multicast only to discover registrars. Therefore, SIP would have to be adopted to support multicast messages. Unfortunately, such a SIP extension would violate the existing SIP standard.

Plain status information broadcast with UDP would be a simple and fast way to signal events in the network. However, using the SIP based event notification has the advantage that also the status signaling is done by an open standardized protocol achieving interoperability among several manufacturers. Furthermore, the SIP based approach is also suitable for status signaling over WANs as it uses unicast.

One reason for the required fast PTT/SQU signaling is the characteristic of the current circuit switched systems and the radio transmitters, as described in section 3.1. If the PTT/SQU signaling requirements would be softened to reflect the characteristics of packet switched system, the prototype implementation could be used for more than six connected nodes.

Chapter 5

A Dependable Radio Service

As the radio service is the only possibility for communication between operators and pilots, this service has to be highly dependable, as even short outages may lead to perilous situations. Failures of the radio service can be caused by failures of the involved nodes or by failures of the underlying network. Even if all components in the system are faultless, outages of the service can occur if the network is congested.

In the previous chapters, the term radio service is used to describe the service offered by the radio server and consumed by the operator position application (refer to figure 5.1). From the operator's view, the main user of the system, the radio service is offered by the application at the operator position. Thus, the radio service depends on the following main components:

- operator position
- radio server
- radio gateway
- radio device[1]

Furthermore, the radio service depends on a network infrastructure and a SIP infrastructure, which in turn relies on the DNS. The SIP infrastructure consists of a registrar, a proxy, and a location service. These three logical entities, which are usually combined in one application, typically use a database system as back-end for storage of user data. Therefore, these SIP services and the database system will be treated as one block in the reliability block diagram [Smi01]. The network infrastructure, which consists of network

[1]As the radio device is an external device which will be connected to the proposed IP based communication system, it wont be considered in the following dependability analysis.

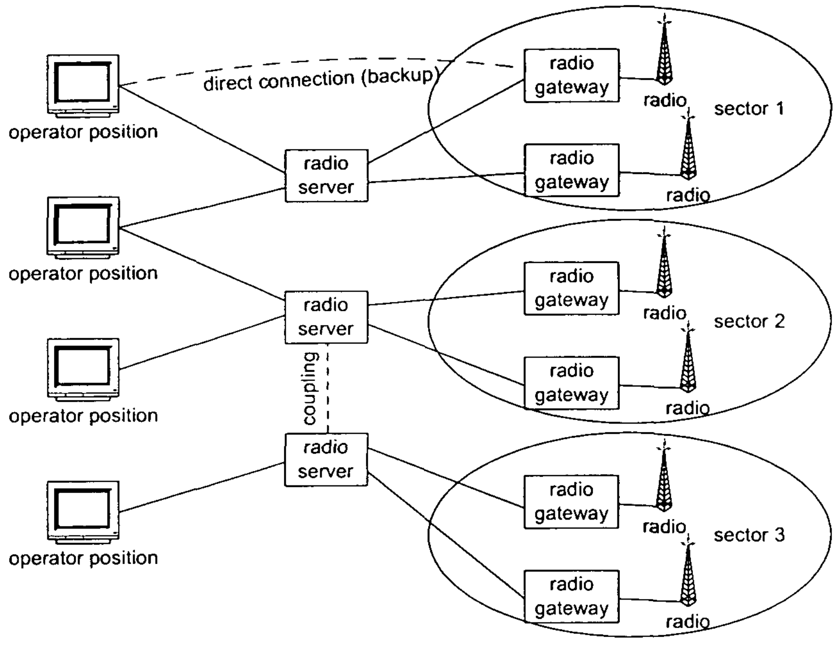


Figure 5.1: Radio architecture

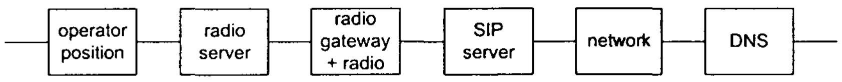


Figure 5.2: Reliability block diagram of the radio service

cables, switches and routers, will also be treated as one block. The DNS system usually consists of local, self managed DNS servers and the public DNS infrastructure[2], which will be treated as one block.

Figure 5.2 shows the reliability block diagram of the radio service. The DNS and the network are out of the scope of this work, nevertheless, they are mandatory for a dependable SIP based communication system [Ohl03][Ogg01]. To achieve higher overall availability, redundancy has to be introduced and failover mechanisms should be used. Regardless of the used failover mechanism, the availability of the failover mechanism must be higher than the availability of the single component [Hat03].

[2]For local communications, the local DNS infrastructure is sufficient.

5.1 Dependable SIP Proxies

The SIP proxy[3] is a central element in a SIP infrastructure. Every SIP UA uses the SIP proxy for registration of its current contact address and for location lookups when sending requests to other users. There are two possibilities to cope with outages of the SIP proxy:

1. Intelligent clients: The SIP client detects the failure of the SIP proxy and switches to a backup proxy. This can be done by configuring a backup proxy at the client or using multicast[4] to locate a SIP registrar [RSC+02] and the techniques of [RS02b] to find SIP servers.

2. Intelligent proxies: There is a proxy cluster and a backup proxy replaces the failed main proxy transparently to the SIP clients. If the SIP proxy fails during an ongoing transaction and there is no state sharing mechanism between main and standby proxy, the transaction may fail, but following transactions will succeed.

The first approach requires additional intelligence in the SIP UAs. Currently, many SIP UAs, especially hard phones, do not support failover mechanisms. Therefore, a high availability solution which is transparent to the SIP UAs is preferable. In [Ohl03], several solutions for high available SIP proxies are presented. For local scenarios, i.e. the SIP clients resides in the same LAN as the SIP proxies, IP takeover is sufficient and is the proposed solution[5]. This solution is transparent to the SIP clients. If the main SIP proxy fails, a backup proxy takes over the IP address of the failed proxy and acts as SIP proxy (see figure 5.3). This requires fail-stop [CDK01][TvS02] behavior of the faulty proxy.

The monitoring of the active proxy can be done by the backup proxy or by a dedicated monitoring device, which initiates the takeover in case of a failure. In the LAN, MAC addresses are used for packet delivery. To map IP addresses to MAC addresses, the address resolution protocol (ARP) is used. In case of an IP takeover, the IP address of the SIP proxy remains constant, whereas the MAC address changes. Therefore, the new SIP proxy advertises the new MAC address by broadcasting ARP-packets with the new IP/MAC-address mapping. This prevents other host in the LAN to send packets to the MAC address of the failed proxy. The virtual router redundancy protocol [KWW+98] as well as the *heartbeat* utility of the Linux high availability project [Lin] use IP takeover and can be used to create high available SIP proxy clusters. A typical value for the duration of the take over is 10 seconds: 5 seconds for detecting the failure and 5 seconds for starting the SIP proxy service at the standby server.

[3]In this work, the term SIP proxy means the combination of the registrar, the proxy and the location service

[4]The *all SIP servers* multicast address is `sip.mcast.net`, which resolves to 224.0.1.75 for IPv4

[5]For global scenarios, the SIP proxies should be distributed global too.

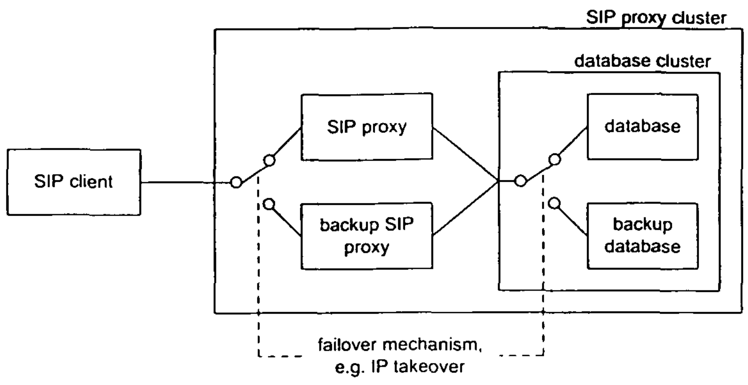


Figure 5.3: SIP proxy cluster and IP takeover

If several ATC centers are interconnected via WANs, each center should have its own SIP proxy. This is necessary to allow local communication in the ATC centers in case of broken WAN-links.

5.2 Dependable Operator Position

If the operator position fails, the operator has to switch over to a spare operator position. Some of the failures will be detected by the operator immediately (e.g. system crash, power supply failures, defect display, ...) whereas other failures are not obvious to the operator (e.g. loss of network connectivity). Such failures have to be signaled by the application to the operator immediately. If the operator has changed to another operator position, it is important that incoming calls to the operator are forwarded to the new operator position. Therefore, the new operator position registers the same SIP address-of-records as the failed operator position. Thus, incoming calls will be forked by the proxy and forwarded to both operator position. If the old operator position failed in a way that it will not respond to the requests anymore, this branch will timeout, whereas the branch to the backup operator position will handle the request.

Problems might occur if the failed operator position still responds to requests, e.g. only the screen or the input devices of the operator position are broken. In this case, the fail-stop assumption is not valid. For example, if there is an intercom[6] request and the partly broken operator position answers the request faster than the backup position, the call will be established to the broken operator position. Therefore it is essential that the partly broken operator position will be deactivated to prevent it from interfering with

[6]Intercom request will be accepted by the operator position without any interaction of the operator. This service allows to talk to an operator instantly in case of emergency situation.

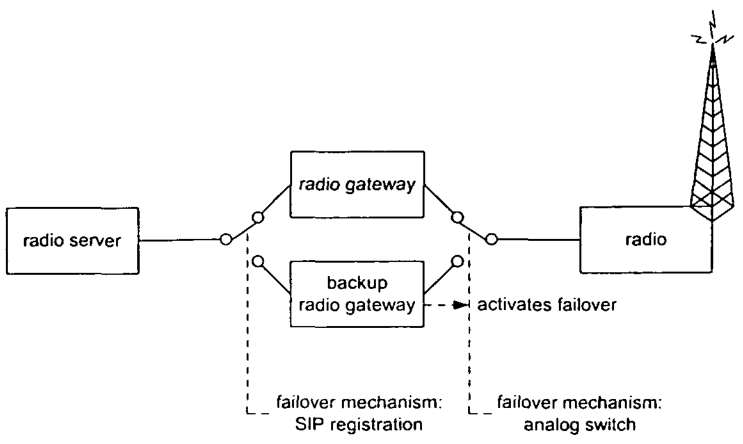

radio server
radio gateway
backup radio gateway
radio
activates failover
failover mechanism: SIP registration
failover mechanism: analog switch

Figure 5.4: Redundant radio gateways with failover transparent to the radio server

the backup operator position. The user location entries of the broken operator position will expire and therefore will be deleted automatically by the registrar.

Another possibility is, that the backup position deletes all contacts at the registrar before registering using a `REGISTER` message with `Expires: 0` and `Contact: *`. Unfortunately, this method will also delete valid entries, i.e. if another operator is registered with the same address-of-record, and can hence not be used.

5.3 Dependable Radio Gateways

The radio gateways connect analog radio devices to the packet switched SIP based VoIP system. Therefore, for high availability considerations, its also important how radios are connected to radio gateways. Figure 5.4 shows a setup with only one radio and redundant radio gateways in an active/standby setup.

As failover mechanism, the SIP location service is used. If the backup radio gateway detects a failure of the active radio gateway (e.g. by sending keep alive messages between the radio gateways), it registers the address-of-record of the corresponding radio device at the SIP registrar and deletes the old contact addresses for this address-of-record. The radio server also detects the failure of the radio gateway (refer to section 5.5) and tries to re-establish the session to the radio gateway. This time, the SIP proxy forwards the request to the backup radio server. Furthermore, the radio needs to be connected to the backup radio gateway. This can be ensured if both radio gateways are permanently connected or by using an analog switch, which is controlled by the backup radio gateway (figure 5.4). Although the analog switch is a single point of failure, a higher overall

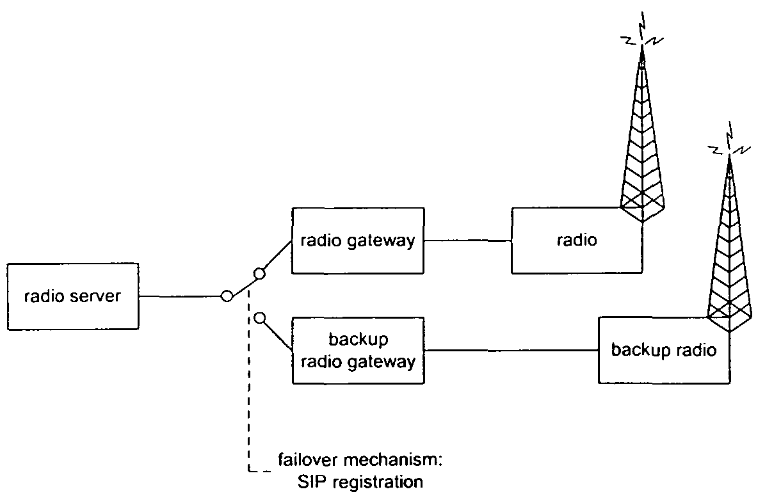


Figure 5.5: Redundant radio gateways and radio devices with failover transparent to the radio server

availability will be reached as switches are very simple devices, which usually have a significantly higher availability as complex devices like radio gateways.

The previous setup has one big drawback: the radio device is a single point of failure. Hence, in mission critical applications like air traffic control there will always be a backup radio. Figure 5.5 shows a setup which is similar to the previous one, except that there is an explicit radio gateway for each radio. The failover again is transparent to the radio server handled by the radio gateways themselves. The radio server only detects a broken session, tries to re-connect and will be directed to the backup radio gateway.

If the radio gateway should be kept simple (e.g. an embedded device) and the radio configuration changes rarely, the failover could be handled by the radio server as shown in figure 5.6. If the radio server detects a failure of the active radio gateway, it connects to the configured backup radio gateway.

In all presented scenarios, there will be a short service disruption[7] during failover. Therefore, the radio server should inform the operator about the short outage, e.g. by playing a warn tone, so that the operator can repeat recent radio messages.

[7]The time period where the service is not available consists of the time needed to detect the failure, the time to takeover (successful REGISTER transaction), and the time to reconnect to the radio gateway. A conceivable time period would be 20 seconds.

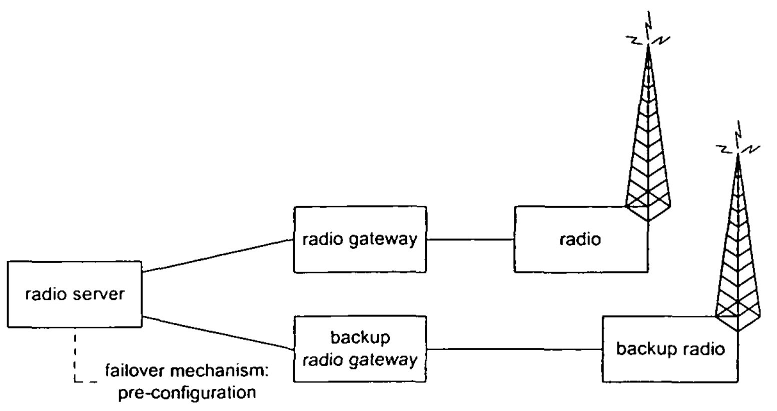


Figure 5.6: Fail over with pre-configured backup radio gateway in the radio server

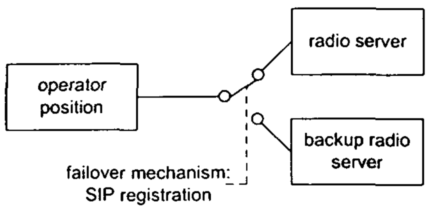


Figure 5.7: Dependable radio server in active/standby configuration

5.4 Dependable Radio Servers

In mission critical applications it is preferable that failovers to backup components happen without service interruption. Active/standby configurations which use the SIP registration as failover mechanism require the clients to re-establish the connections in case of a failover. Therefore, a failover without service interuption is not possible in an active/standby configuration. This might be overcome in an active/active setup where the operator position and the radio gateway are permanently connected to two radio servers. The monitoring of the radio servers and the comparison of their output is very complicated as the radio servers are not synchronized. Furthermore, the radio servers have to share their state to avoid inconsistent situations, e.g. radio server A gives sector access to operator A whereas radio server B gives sector access to operator B. This arrangements complicate the whole radio service, introduce additional delays and are a source of additional faults.

The following active/standby solution is less complex by using the SIP registration service to handle the failover. Figure 5.7 shows the setup and the failover sequence diagram is shown in figure 5.8. When the operator keys-in a certain frequency, the operator positions sends an INVITE to the SIP proxy with a request URI of the corresponding radio sector, e.g. INVITE sip:f100@atc.org. The proxy resolves the request URI to the current contact address of the corresponding radio server (the active radio server) and forwards the request. After session establishment (INVITE-200 OK-ACK handshake), the operator position monitors the radio server by methods described in section 5.5. Furthermore, the backup radio server monitors the active radio server. If the active radio server fails, the backup radio server deletes the old location entry for this sector and registers at the proxy using the address-of-record of the radio sector, which were served by the failed radio server. The old contact, which points to the failed radio server, will be deleted by sending a REGISTER message with Expires: 0 and Contact: *.

The operator position also detects the failure of the radio server, terminates the broken session, waits some seconds and then tries to re-establish a session with the radio server. Again, the INVITE is sent to the proxy, but this time the proxy forwards the request to the backup radio server. If the operator sends the INVITE immediately after the detection of the broken radio server, without giving the system time to re-configure, it may happen that the proxy still forwards the request to the broken radio server. As the radio server does not respond, it may take up to 31.5 seconds[8] until the transaction fails and another attempt can be initiated.

During failover, the radio service on the failed sector is unavailable. This has to be signaled to the operator, in a way that the operator is informed about possibly lost radio messages.

The radio gateway should have the possibility to be located at remote sites which are connected via low bandwidth connections. Therefore, it is essential, that the radio gateway accepts only one voice connection from a radio server. During radio server failover, the bakup radio server tries to connect to the radio gateway. Therefore, the radio gateway has to accept the new call and end the existing call to the broken radio server. Nevertheless, this is no perfect solution, as problems occur if a non privileged SIP client sends an INVITE message to the radio gateway, which causes the gateway to quit the current session with the radio server.

The setup in figure 5.7 requires a dedicated backup radio server for each active radio server. Another setup, which is shown in figure 5.9 and figure 5.10, reduces the amount of needed backup radio servers as the backup servers are shared amongst active servers. This setup can save hardware costs, but it requires 3rd party intelligence, which monitors the active radio servers and controls the backup radio servers. In case of a failure, the control instance activates a backup radio server with the configuration of the failed

[8]31.5 seconds results from the retransmission algorithm for SIP over UDP in the SIP specification [RSC+02]. To reduce this value, the SIP timer T1 has to be reduced as described in [DKG03]. The timeout for SIP over TCP depends on the implementation of the TCP/IP stack, i.e. depends on the used operating system.

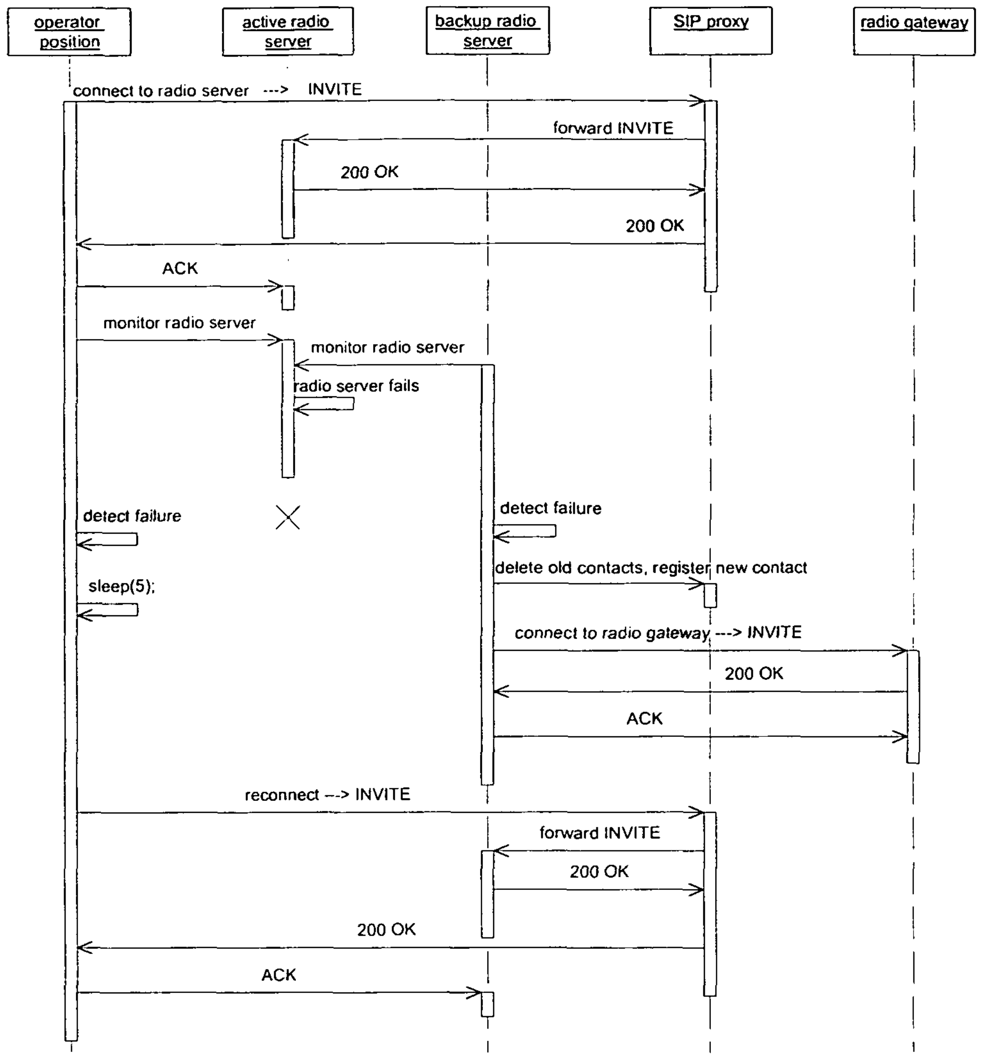


Figure 5.8: Failover sequence diagram for SIP based radio server failover

server. The configuration can be done only after a failure, as the backup radio server does not know, in advance, which radio server will fail.

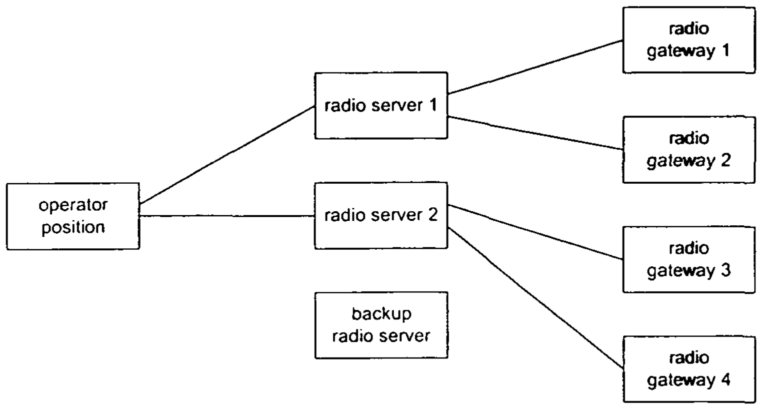


Figure 5.9: Shared backup radio server in standby

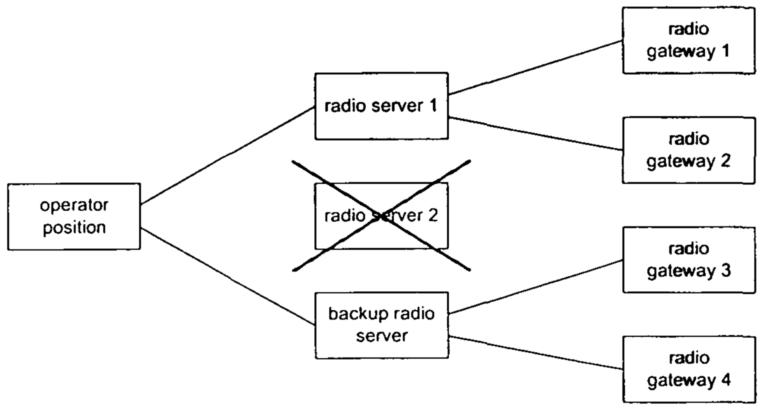


Figure 5.10: Shared backup radio server activated

5.5 Failure Detection

The radio service uses a client/server based model. The application at the operator position is a client and requests the radio connectivity from the radio server. In turn, the radio server acts as client by requesting the radio connectivity from the radio gateways (refer to figure 5.1). The servers do not care if the clients fail, but as the clients need the service from the server, and if the server's reliability mechanism is not totally transparent to the client[9], the clients have to detect failures of the server. Following, the various types of failures and several ways to monitor the availability of certain nodes are analyzed. In certain cases, the operator position might act as a server for other operator position, e.g. if an operator is interested in the availability of another operator.

5.5.1 Failure Types

Server response failures are differentiated into the following types[10]:

- no response
- wrong response

For the client, it is irrelevant if the response arrives too late, or there is no response at all. Furthermore, if there is no response at all, there is no way for the client to detect if the request or response was lost in the network, or if the server is "down". Server outages are typically due to application failures, hardware failures, or power supply failures. Software crashes could be resolved by restarting the software, nevertheless it would be wise to start the application on a backup server to analyze the crash and to find the fault. Hardware failures and power supply problems usually require to maintain the device, which takes some time. Therefore, a backup device is necessary.

If the whole device fails, the broken node won't disturb the remaining system, but if for instance only the application crashes, certain failover setups like IP takeover (section 5.1), require a fail-stop behavior of the particular device to avoid interferences with the backup device. This is also necessary, if a service may be provided only by one server at a time, like the radio service. Typically this occurs due to temporary network outages. The server is not reachable, so a backup server starts the radio service. If the network returns it will cause problems, if the service is offered by two servers. Therefore, the redundant servers also have to check for the return of a failed server as one of them has to shut down its service.

[9]Totally transparent means that the client can not detect a failover of the server. This is very difficult, as it requires state sharing between the servers in the cluster.

[10]Network failures are out of the scope of the following analysis.

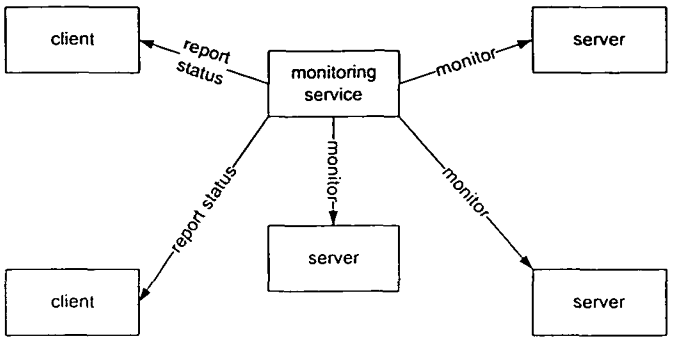


Figure 5.11: Central server monitoring

A correct, but late response typically indicates that the resource is overloaded, the service is misconfigured, a requested sub-service failed (e.g. DNS), or any other service, which runs on the same device behaves unexpected and consumes most of the resources of the device. Nevertheless, this makes the service useless and a failover to another server is necessary. In this case it is essential, that the "slow" server is shut down (fail-stop) to avoid interferences with the backup server.

Detecting Byzantine failures [CDK01][TvS02]—the server responses to requests with a wrong behavior—is a non trivial task. Fortunately it is not necessary, as the operator will detect malfunctions of operative components and he will trigger a manual failover to standby components in case of failures.

5.5.2 Server Monitoring

Knowing the availability status of a server in advance, before requesting the service, is essential for fast failover. Therefore, the servers have to be monitored. This can be done centralized or distributed. The centralized version is shown in figure 5.11. A dedicated monitoring service monitors all service offering nodes (e.g. radio servers, radio gateways, gateways, directory services ...) and reports the status to the clients. To reduce network load, it is useful to gather information from several or all of the servers and send a single status report using broadcast or multicast. As the whole system relies on the status reports, the monitoring service itself has to be highly available.

The use of broad-/multicast requires that all clients are within the same subnet as the monitoring service or in a multicast capable network. If several hosts are scattered around several subnets, e.g. the hosts are on several sites, which are connected by WANs, it might be useful to have a dedicated monitoring device in every subnet, which is responsible for monitoring and notifying the hosts within this subnet (figure 5.12). The monitoring services share the state of their servers amongst each other using unicast

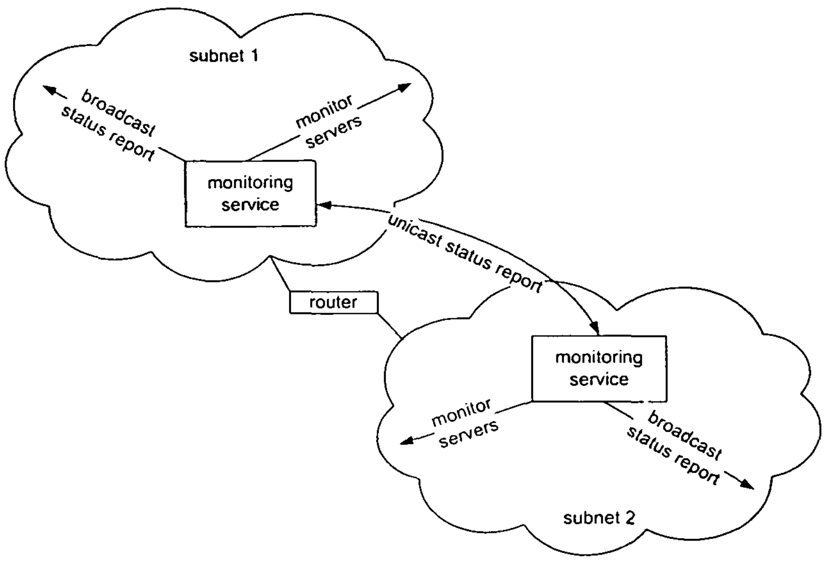


Figure 5.12: Relay status information between several subnets

connections. Additionally, this reduces the WAN traffic, but increases the delay of the status report.

In the distributed monitoring approach, every client which is interested in the availability of a service, monitors the service itself as shown in figure 5.13. This will lead to a lot of monitoring sessions and to a lot of traffic in a full meshed scenario, i.e. every client is interested in the availability of every server. On the other hand, if the clients are only interested in certain servers, i.e. the radio server is only interested in the radio gateways which belong to its sector and the operator position is only interested in the radio servers the operator works on, and existing sessions (like audio sessions) can be re-used to deliver status information, this approach introduces only small additional traffic. Furthermore, as there is a direct client-server monitoring session, the failure detection can be faster as with a dedicated, central monitoring service.

Regardless whether the monitoring will be central or distributed, the availability information has to be retrieved by one of the following methods:

- Servers send keep alive messages in periodic intervals and the clients listen for them.
- The clients send availability requests to the servers and wait for responses.
- Existing data connections will be reused to deliver keep alive messages.

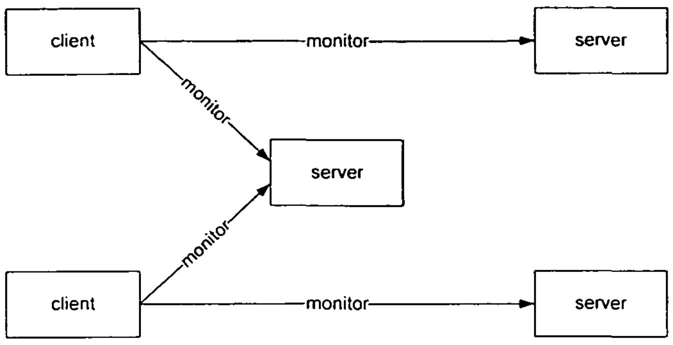


Figure 5.13: Distributed server monitoring

In the first scenario, the server sends availability information in periodic intervals to the clients. The absence of the status reports will be treated as a failure of the server. Therefore, the clients need to know the time between the status updates. If the server knows which clients are interested in receiving status information, the server may use unicast to send its status reports. The address of the clients can be pre-configured or the clients subscribe to the server to retrieve the status information. If the server does not know the clients, it has to use broadcast or multicast to send messages. As broadcast and multicast are unreliable, and packet loss might occur, a missing status report can not be considered as an error in this case, but as a warning only. However, the absence of several subsequent status reports should be considered as failure of the server. Furthermore, broadcast and multicast are limited to the same subnet[11] and would require status relays as described above and shown in figure 5.12. Multicast is better than broadcast in terms of network utilization and host load but is more complex, especially in non-standard network configurations[12].

In the second scenario, the client polls all services for their availability status. If no answer is received, the server may be treated as failed. The client polls only servers, which it is interested in. Therefore, it has to know the address of the server by configuration or by querying a location service. The polling can be performed on various layers of the network—the higher the layer, the more accurate is the received availability status. ICMP messages (ping) can be used to test the network link, the hardware, and the

[11]Multicast can also be used over multiple subnets and WANs if multicast capable networks are used, but they are rare.

[12]In high available systems, hosts ore often connected to the network via two independent connections, i.e. a dual port network interface card or two network interface cards which are connected to different switches, to switch over to the backup link if the primary link fails. When using multicast, such a failover requires to re-join the multicast group to activate the multicast forwarding on the switch port of the backup link. The multicast join is done by the application - therefore the failover must not be transparent to the application

network stack of the server. To test the server application, the polling must be done at application layer, e.g. an `OPTIONS` request will be used to test SIP based applications. As the availability check is initiated by the client, each client can choose the polling interval individually. As this scenario uses unicast, it is not suited for large systems where every client wants to know the status of every server. Furthermore, the polling requires at least 2 packets per availability check, whereas the previous scenario requires only 1 packet per status report.

The third scenario is well suited, if there are already existing connections between the server and the client with constant data flow like audio sessions, where RTP packets are sent in periodic intervals. If the client receives no incoming RTP packets for a certain time, it assumes that the server has failed, terminates the current session and applies one of the failover mechanisms described in sections 5.3 and 5.4. If the RTP stream is used as availability indicator, silence suppression can not be used.

The various methods can be compared using the following parameters:

Failure detection time In packet switched networks, failures can not be detected immediately. The client always has to wait for a timeout which can be assumed to indicate a failure. To detect failures faster, the timeout value has to be reduced. This requires to send availability reports more frequently which in turn may stress the network and the hosts.

Host load The monitoring leads to additional load on the hosts. Servers need to generate and send status reports, clients need to process incoming reports. Using broadcast or multicast will reduce the servers' load in generating reports, but it will increase the utilization of all hosts in the network as they have to process the broadcast requests. If there is one multicast address used for sending status reports which is used by all servers, each client receives the reports of all servers and has to process them, although a client might be only interested in the status of one of the servers.

Network load Each availability check produces network traffic. The shorter the detection time and the higher the number of clients and hosts, the higher is the network load. In LANs, high bandwidth is cheap and the network can be dimensioned onto the needs of the system, but in WANs, bandwidth is much more expensive and it is desired to keep traffic low to save money and to avoid interference with other services.

Complexity The availability check mechanism should be simple as complex mechanisms increase the possibility of failures in the failure detection.

The proper method and the above parameters depend on the network structure and the number of clients, servers and other network hosts. Also a combination of the several methods can be used.

Chapter 6

Summary and Conclusion

6.1 Summary

Voice Communication for air traffic control and public safety is a rather new field of application for VoIP. Therefore, current protocols can't be used out-of-the-box as they don't support radio communication signaling natively. Furthermore, this field of application has much higher dependability requirements than typical VoIP applications like enterprise telephony, and therefore an architecture which supports fault tolerance is needed.

These requirements, along with the environment the system has to fit in (section 2.6), lead to the presented architecture consisting of operator positions, radio servers and radio gateways. The central radio server is needed to distribute the incoming radio messages, to keep the bandwidth requirements to the radio gateway low, and to arbitrate concurrent sector access from several operators.

In chapter 3, several possibilities to deliver the radio signaling have been shown. They differ in the complexity of the protocol, their implementation complexity, their capability to deliver additional data and their bandwidth demands. SIP was chosen as the most promising protocol due to the possibility of transmitting the radio signals without the need to setup an RTP stream and its extensibility, which supports delivery of additional PTT/SQU related signaling data in the body of the SIP messages. Furthermore, as SIP is a reliable protocol, it reduces implementation complexity by avoiding a retransmission mechanism in the application.

As proof of concept, a prototype was implemented, which used radio gateway simulators instead of real radio devices for simplicity reasons. The radio server connects to the radio gateways (simulators) in their respective operation mode (Rx, Tx, RxTx) and supports connections from the operator positions in Status, Rx and RxTx mode. Furthermore, the radio server supports a compatibility mode, which allows connections from standard SIP phones in Rx mode.

The prototype implementation proved that the required functionality can be achieved by using SIP as protocol for radio signaling. To verify if this implementation also fulfills the required maximum signaling delays, several delay measurements were performed. The measurements show that the generation of SIP requests takes a considerable amount of time, caused by the complexity of SIP. As the notifications are sent one by one using unicast, the number of subscribers that can be informed about the radio event within the required time is limited. The comparison with the SIP proxy in a forking scenario shows that an application, which is trimmed for high performance, tremendously speeds up generation of SIP requests. This shows that there is potential to increase the performance of the SIP stack.

The required high availability in ATC systems can only be overcome by means of redundancy. Chapter 5 shows redundancy concepts for the SIP proxy, the operator position, the radio server and the radio gateway.

6.2 Related Work

6.2.1 AudioLAN

The AudioLAN project [GG99], developed by the Eurocontrol Experimental Centre, is part of the OpenATC initiative [Eura]. AudioLAN is a voice communication system based on VoIP technology and was developed especially for simulation and operator training. AudioLAN is a distributed system without any central components. The system consists of operator positions, gateways to the PSTN and radio gateways. Furthermore, a supervision and configuration application, and a voice recorder are available and can be placed on any host within the AudioLan network. AudioLAN uses IP multicast for the audio transmission, therefore the used IP network must be multicast capable. To connect multicast LANs via not multicast capable WAN connections, a multicast tunneling application is used.

Figure 6.1 shows the layered software architecture of the operator positions. The voice manager controls the RTP streams and manages two independent audio interfaces to separate radio and phone. The communication manager is responsible for phone and radio communications. The human machine interface (HMI) and the optional configuration and supervision tool are Java based applications which interact with the communication manager.

Standard phone calls and conferences are set up using H.323 as signaling protocol. Information about the PTT/SQU signaling and redundancy scenarios to achieve fault

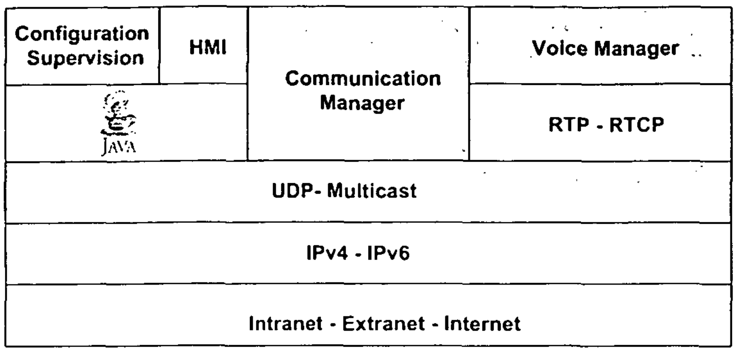


Figure 6.1: Software layers of the AudioLAN system [GG99]

tolerance are not available. Furthermore, no information is available how the sector is arbitrated is case of concurrent access to a radio resource. The end-to-end audio latency time is about 60 ms due to jitter buffer [GG99].

The most important differences to the presented voice communication solution are:

- AudioLAN does not support operator mobility but requires manual configuration of operator and gateway addresses. Therefore, an operator can not change its working position without re-configuration of the system. The presented VCS uses SIP mobility based on the REGISTER method to deal with mobile operators. Therefore, mobile operators can be reached at their current operator position without any re-configuration.
- AudioLAN does not use a radio server but each operator positions communicate directly with the radio gateway. Therefore, there is no central component which is a single point of failure. Nevertheless, the radio gateway can not be located on remote places, which typically only have low bandwidth connections.

6.2.2 Internet Radio Linking Project

The aim of the Internet radio linking project (IRLP) is to "... provide a simple and easy system to link radio systems together using the Internet as the communications backbone ..." [IRL]. IRLP's software is based on "Speak Freely" [spe], a freely available software for Internet telephony, which supports RTP for audio transmission. Figure 6.2 shows the main idea of IRLP: ham radio sites will be interconnected by IP networks.

The standard PC acts as radio gateway using a standard PC soundcard which is connected to the audio in- and output of the ham radio. Furthermore, the dedicated IRLP

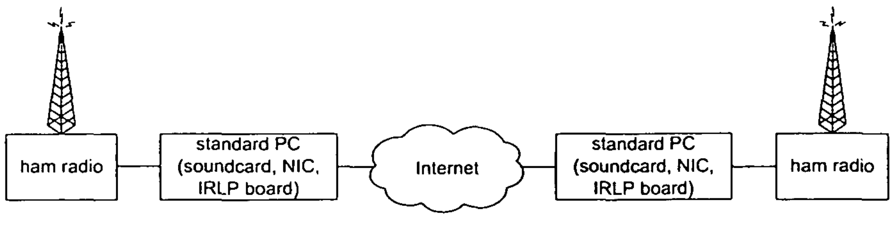

ham radio
standard PC (soundcard, NIC, IRLP board)
Internet
standard PC (soundcard, NIC, IRLP board)
ham radio

Figure 6.2: Linking of ham radios using IRLP

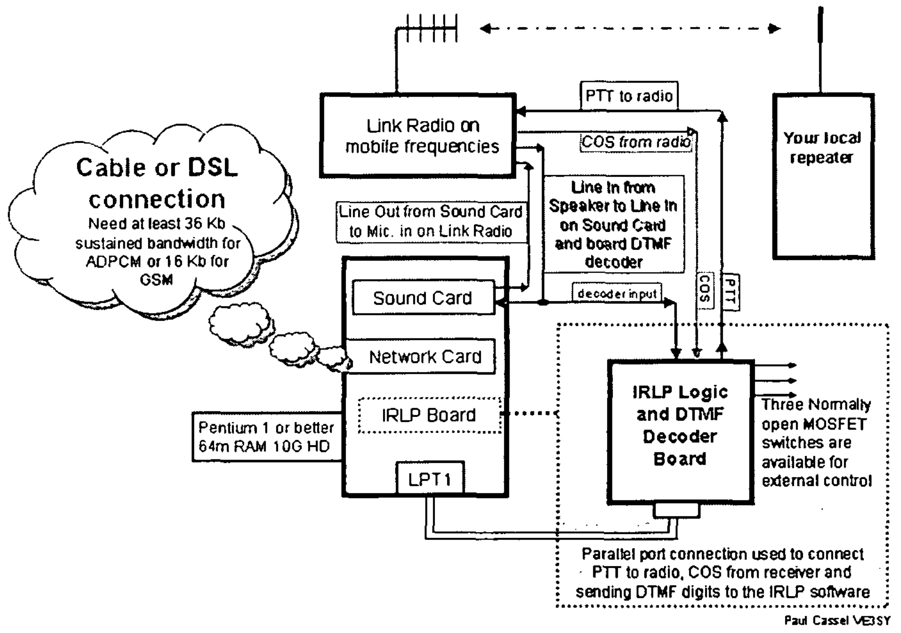

Link Radio on mobile frequencies
Your local repeater
PTT to radio
COS from radio
Line In from Speaker to Line In on Sound Card and board DTMF decoder
Line Out from Sound Card to Mic. in on Link Radio
Cable or DSL connection
Need at least 36 Kb sustained bandwidth for ADPCM or 16 Kb for GSM
Sound Card
decoder input
COS
PTT
Network Card
IRLP Board
Pentium 1 or better 64m RAM 10G HD
LPT1
IRLP Logic and DTMF Decoder Board
Three Normally open MOSFET switches are available for external control
Parallel port connection used to connect PTT to radio, COS from receiver and sending DTMF digits to the IRLP software
Paul Cassel VE3SY

Figure 6.3: Typical IRLP Node in Repeater Configuration [Intb]

board is necessary to process and generate the PTT and SQU signals as shown in figure 6.3. This board receives the carrier operated squelch (COS) or continuous tone coded subaudible squelch signals (CTCSS) from the radio and sends the information to a control software running on the PC using the parallel interface. The control software will start and stop the audio stream according to the SQU signal received from the IRLP board. As the setup is symmetrical, SQU on one site causes PTT on the other site and vice versa. The PTT signal is derived from the presence of an incoming audio stream (implicit PTT in section 3.3.1) and will be sent to the IRLP board, which generates the PTT signal for the ham radio. Furthermore, the IRLP board can decode DTMF tones and trigger several activities on reception of certain DTMF codes, e.g. shut down the IRLP node.

The IRLP system can be seen as two interconnected radio gateways to couple two sectors (frequencies). The users still utilize their ham radio equipment to send and receive radio messages, so the operator position has no IP connectivity but is still analog. The voice delay between the IRLP nodes mostly depends on the connectivity of the sites and the jitter buffer size at the receiver and is typically less than 200 ms [Intc].

The most important differences of the IRLP compared to the presented voice communication solution are:

- The IRLP project uses IP only to interconnect radio transceivers. The operators still use their analog radio equipment whereas in the presented VCS also the operator position uses IP technology for voice communication.
- The IRLP does not use explicit PTT and SQU signaling but derives these signals from the presence of RTP packets, wheres the presented VCS uses explicit signaling by means of SIP.

6.2.3 Push-to-Talk over Cellular

In August 2003, Ericsson, Motorola, Nokia and Siemens Mobile published the specification for a push-to-talk over cellular (PoC) service for GSM/GPRS networks [Eria], which is based on the IP Multimedia Subsystem (IMS) [PMKN04] as defined by the 3rd Generation Partnership Project (3GPP) [3GP]. The PoC service builds virtual rooms in which users can talk directly to each other just by pressing the PTT button, without the need of explicitly calling the other participants. The PoC service is a plain IP-based service and is intended to use GPRS in current existing GSM networks.

Figure 6.4 shows the architecture of the PoC service and the integration with the IMS. The user equipment (UE) communicates with the Group and List Management Server (GLMS) to manage groups, contacts and access lists, with the IMS to initiate and terminate sessions, and with the PoC server which controls the group calls and relays the voice data between the PoC users. The UE uses HTTP [FGM+99] and an XML [HMM02] based payload format to communicate with the GLMS [Eric]. The session signaling between the UE and the IMS is done with SIP and the audio communication between the UE and the PoC server uses RTP/RTCP.

Figure 6.5 shows a typical signaling flow. A user wants to transmit a voice message to all group members and therefore presses the PTT button on its mobile phone. The UE sends an `INVITE` request to the IMS, which signals the session invitation to the PoC server[1]. The PoC server establishes the calls to the other group members via the IMS. As soon as the first callee accepts the invitation, the PTT pressing user gets a

[1]This can be done by SIP or any other protocol. This is not defined in the current specification.

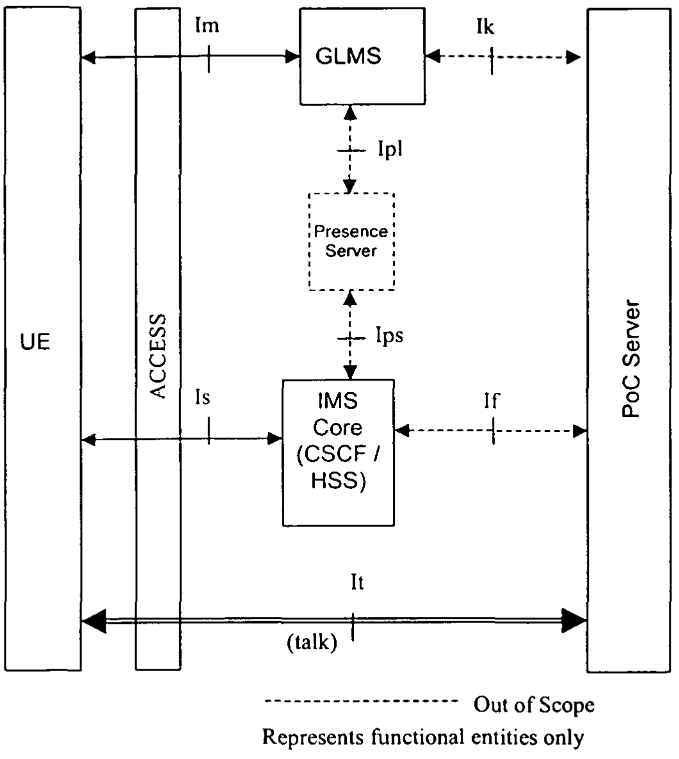


Figure 6.4: PoC architecture [Erib]

Talk indication and can start talking. The caller's UE will send RTP packets to the PoC server which forwards the RTP packets to the other group members. The PoC server also buffers the voice stream for group members which accept the call later and therefore would otherwise miss some seconds of the voice message. The next time a group member wants to transmit a voice message, the sessions are still established. Therefore, the signaling takes place directly between the UEs and the PoC server. The arbitration of the virtual radio room is called *floor control*[2] and is done by the PoC server. The PTT signaling is event based and uses application defined (APP) RTCP packets [Erie] as described in section 3.3.3. After a certain time of inactivity[3], the sessions will be terminated and further voice messages require a new session setup.

The PoC specification also defines other setup scenarios where the user may talk immediately after pressing the PTT button and the RTP stream is buffered at the PoC server until the group members are ready to receive the RTP stream. Table 6.1 shows the floor control messages which are defined in PoC. As these messages are sent by

[2]The term *floor control* is comparable to the term *sector arbitration* used in this thesis.

[3]The time period is not defined in the specification.

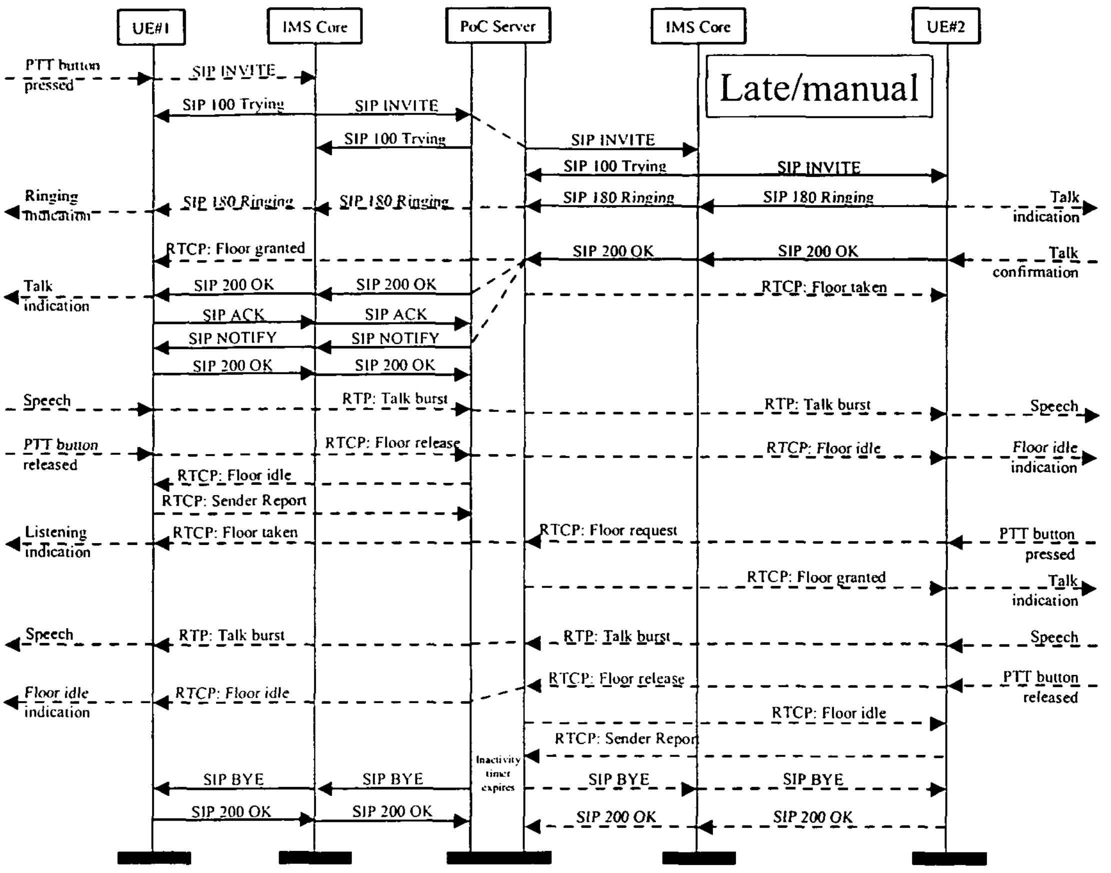


Figure 6.5: Late media establishment and manual answer [Erid]

RTCP, the delivery is unreliable and the UEs and the PoC server have to implement a retransmission mechanism. Thus, the messages will be retransmitted, if the floor control requests are not answered within a certain period of time. PoC does not acknowledge all floor control actions, but only the `floor request` and `floor release` actions. The `floor request` action will be acknowledged by the `floor grant`, `floor taken` or `floor deny` action, whereas the `floor release` message will be acknowledged by the `floor idle` or `floor taken` action. The retransmission timeout is not defined in the PoC specification.

PoC only supports UDP as transport protocol for the SIP signaling whereas this thesis also supports TCP as transport protocol. Furthermore, PoC uses compression of the SIP signaling [PBC+03][HCF+03][GMBO+03][Cam03] to reduce the packet size and lower the signaling delay. UEs are addressed with tel-URLs [VS00] or SIP-URIs. The phone numbers in the tel-URLs have to use a local number format or E.164 [Inta] numbers.

floor control actions	**description**
`floor request`	sent from a UE to the PoC server to request for the permission to talk
`floor release`	used by a granted user to release the floor
`floor grant`	produced by the PoC server to inform the requesting participant that the floor has been granted
`floor idle`	from the PoC server to the participants indicating that the floor is idle
`floor deny`	from the PoC server to the requesting participant indicating that the floor request was denied
`floor taken`	from the PoC server to the participants indicating that the floor has been granted to a certain user
`floor revoke`	from the PoC server to the granted participant indicating that the users permission to talk has been revoked

Table 6.1: *Floor control actions*

Both of them have to be resolved to SIP-URIs using ENUM (refer to section 4.1).

Although the PoC service is similar to the presented radio communication solution, there are some significant differences:

- The PoC service delivers most of the floor control actions unreliably, whereas in the presented solution every radio event is delivered reliably, acknowledged by a `200 OK` SIP message.
- The PoC service is like the presented radio communication solution, but without radio gateways, all participants have IP connectivity. Therefore, there is no differentiation between incoming and outgoing radio messages and no SQU signaling, only PTT signaling.
- The PoC service currently does not signal additional data which is essential for ATC (e.g. the signal level at the receiver, data sent by a radio data link).
- Within the PoC service, if the active UE releases the floor by sending a `floor release` action to the PoC server, this action includes the sequence number of the last RTP packet of the ongoing talk spurt. This information is currently not available at the radio server and should be integrated with future releases (refer to future work in section 6.3).
- The PoC service does not guarantee the delivery of the voice messages in real-time. This is caused on one hand by the use of GPRS, which introduces much more delay

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P| service |  PT=CNTR=205  |            length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        SSRC of sender                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      control information                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6.6: RTCP extension packet format

than typical in wired networks, and on the other hand by the setup scenario, in which sessions are teared down after a certain time of inactivity, which requires a new session setup for further voice messages.

Using RTCP messages for floor control enables faster signaling than using SIP, though it increases the application's complexity by introducing retransmission mechanisms. Therefore, it might be worth to adopt some characteristics of the PoC service into the presented work with respect to the requirements for radio communication in public safety.

6.2.4 RTCP extension

The Internet draft "Session control through RTCP" [Hal02] introduces call control and session signaling into RTCP for purposes like push-to-talk as floor control in multi-party conferences, instant messaging, and muting of audio streams for "listen only" participation in conferences. Therefore, it defines a new RTCP packet using the payload type 205 as shown in figure 6.6. The service field will define the type of service this control packet belongs to and the control information field will hold the requested activity within this service. The control signaling will be event based, therefore reliable transmission is necessary. Neither RTCP [SCFJ03] nor this extension cares about reliable transmission, therefore a retransmission and packet acknowledge mechanism has to be applied by the application.

This draft is incomplete and therefore hard to compare with other solutions.

6.2.5 Proprietary Solutions

Several radio equipment manufacturers like Catalyst Communications Technologies Inc. [Cat], Telex-Vega [TEL] or JPS Communications [Ray] already sell products to interconnect radio transceivers by VoIP. They use proprietary signaling protocols and therefore, these products are not interoperable with products of other vendors and information about implementation details are hardly available. Recently, Cisco Systems, Inc. also announced a solution to interconnect radio equipment, called "Land Mobile Radio over IP" [Cisb]. This system uses the presence of an RTP stream to generate the PTT and SQU signals locally as described in section 3.3.1. This method is also used by the IRLP (see section 6.2.2).

6.3 Future Work

In the current implementation, the radio server starts relaying RTP packets as soon as a PTT/SQU-on notification arrives and stops relaying after receiving the corresponding PTT/SQU-off notification. Depending on the operator position and radio gateway implementation it may happen that the PTT/SQU-off notification is sent prior to the last RTP packet. Furthermore, this can also happen when the notification is sent after the last RTP packet as in IP networks a packet can overtake another packet. In this case, the radio server would stop relaying too early and the last RTP packet will not be relayed. This can be solved by sending the sequence number of the last RTP packet in the PTT/SQU-off notification as done within the PoC service [Erie]. This enables the radio server to wait for the last RTP packet which is identified by its sequence number.

Currently used voice communication systems support enhanced services like coupling of certain radio sectors and interconnecting radio session with ongoing phone calls. This requires extensions at the operator working positions and the radio server. A possible coupling solution was already presented in section 2.6, nevertheless is has to be implemented and verified that the system still fulfills the timing requirements despite the additional delay introduced by successive radio servers.

Although the presented architecture already supports coverage—the radio server can connect to several radio gateways in Rx, Tx, and RxTx mode—an algorithm is needed to choose the best receiver for incoming radio messages. The current implementation chooses the radio gateway which signals SQU-on first. This should be replaced by a logic, which uses the signal level of the corresponding receiver, which will be integrated into the SQU-on notifications.

Many scenarios desire automatic service location . This can be solved using service location protocols like Salutation [Con] or SLP [GPVD99].

In the current implementation, operators can use a standard SIP phone to dial into the radio server and receive the radio messages of the corresponding sector, but they are not able to transmit a radio message. This feature can be added to the system by generating the PTT signal in the radio server using voice detection techniques (VOX-PTT). For

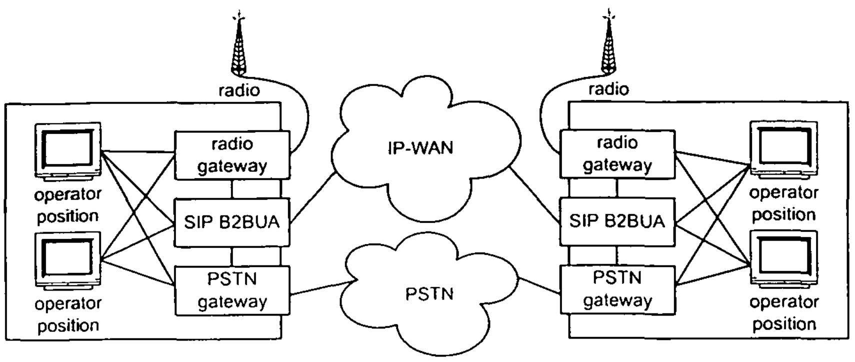


Figure 6.7: The B2B UA controls all external calls

this purpose, the radio server analyzes the RTP stream from the phone and if the audio signal is above a certain level the PTT will be generated locally.

In public safety environments it is often required to guarantee phone calls also in congested network situation. These high priority calls have to disrupt existing low priority calls to free bandwidth resources. To disrupt established calls, a SIP back-to-back user agent (B2B UA) is necessary. The B2B UA resides on the border of the ATC center and stays in between all established calls to external destinations as shown in figure 6.7. The B2B UA has to know the available bandwidth for all the possible call destinations and if the maximum bandwidth is reached, other phone calls will be disrupted in favor of the high priority calls. The call priority can be signaled using the SIP priority header [RSC+02], e.g. `Priority: emergency`.

The current implementation relies on the SIP signaling for PTT and SQU information. If for any reason the SIP signaling fails, but the audio transmission using RTP is still alive, it would be useful to get some basic PTT/SQU information also from the RTP streams, e.g. by adding some bits of PTT/SQU data into every RTP packet using the extension header. This will keep the radio service alive, although with reduced functionality.

6.4 Conclusion

The prototype implementation shows that the SIP based signaling approach is appropriate to signal PTT, SQU, and related additional data. However, the delay measurements show that the delay requirements are hard to fulfill. The currently used SIP stack takes 1.6 ms for each `NOTIFY` transaction, which limits the number of servable operator positions and radio gateways.

The current implementation can not be used in systems in which a lot of operator positions are subscribed to the same sector. Nevertheless, it can be used in applications with reduced number of operator positions and less strict timing requirements, e.g. as in voice communication systems for public transport and in airport tower applications.

The comparison with the RTP distribution delays shows that RTCP notifications, which are quite similar to the generation of RTP packets, are produced much faster with the consequence of serving more subscribers within the same timing constraints. Therefore, using an RTCP based signaling approach increases the number of servable subscribers. Nevertheless, the number of subscribers is still limited due to the serial notifications caused by unicast signaling and the not avoidable retransmission mechanism. This can be eliminated by using multicast, with the disadvantage of using an unreliable signaling.

List of Figures

1.1 Airspace sectors in the area of airport Linz [Aus] 3
1.2 A typical voice over IP architecture 6
1.3 The SIP UA registers at the SIP proxy 8
1.4 The SIP trapezoid—a typical call setup 9

2.1 *Transmit a Radio Message* Use Case 13
2.2 System analysis model of the radio system 14
2.3 Service environment for air traffic control 15
2.4 Layered service integration 16
2.5 Enhanced voice services 17
2.6 Radio architecture 18
2.7 Coupling of radio sectors 19
2.8 High level operator position design model 20
2.9 Subscription state diagram 20
2.10 Sector activity state diagram at operator position 21
2.11 Radio gateway positioning 22
2.12 High level design model of the radio server 22
2.13 Remote sidetone 23
2.14 Remote sidetone delay 23
2.15 Blocking of the remote sidetone 24

3.1 PTT signaling scenario 26
3.2 SQU signaling scenario 27
3.3 PTT delay and voice delay 29

3.4 Event subscription and notification 31
3.5 SIP notification of an incoming radio message event 33
3.6 Event subscription 34
3.7 Event notification 34
3.8 SIP based radio signaling scenario 35
3.9 Audio playback despite PTT-off signal 36
3.10 4-way PTT handshake 36
3.11 Implicit PTT signaling 37
3.12 RTP header 39
3.13 RTP extension header 39
3.14 Explicit RTP signaling scenario 40
3.15 RTCP APP type packets 41
3.16 Explicit RTCP signaling scenario 42
3.17 Combination of SIP, explicit RTP and implicit RTP signaling 44
3.18 Operator position key-in scenario 46
3.19 Rx compatiblity mode for standard SIP phones 47
3.20 Tx mode session for radio transmitters 48

4.1 NAPTR records for two SIP, one email and one telephone resource 51
4.2 Class diagram of the *dissipate* SIP stack [Bila] 54
4.3 Class diagram of the radio server 55
4.4 State diagram of the RadioServer class 56
4.5 Radio gateway simulator 58
4.6 Screenshot of the operator position application 59
4.7 Class diagram of the operator position application 60
4.8 Multiple receptions from the same operator 61
4.9 Setup for signaling delay measurements 62
4.10 PTT signaling scenario 63
4.11 PTT signaling delay 64
4.12 SQU signaling scenario 65

5.1 Radio architecture 70
5.2 Reliability block diagram of the radio service 70

5.3 SIP proxy cluster and IP takeover 72
5.4 Redundant radio gateways with failover transparent to the radio server . 73
5.5 Redundant radio gateways and radio devices with failover transparent to the radio server 74
5.6 Fail over with pre-configured backup radio gateway in the radio server . . 75
5.7 Dependable radio server in active/standby configuration 75
5.8 Failover sequence diagram for SIP based radio server failover 77
5.9 Shared backup radio server in standby 78
5.10 Shared backup radio server activated 78
5.11 Central server monitoring 80
5.12 Relay status information between several subnets 81
5.13 Distributed server monitoring 82

6.1 Software layers of the AudioLAN system [GG99] 87
6.2 Linking of ham radios using IRLP 88
6.3 Typical IRLP Node in Repeater Configuration [Intb] 88
6.4 PoC architecture [Erib] 90
6.5 Late media establishment and manual answer [Erid] 91
6.6 RTCP extension packet format 93
6.7 The B2B UA controls all external calls 95

List of Tables

1.1 Air traffic control terms 2

3.1 PTT/SQU events sending and receive matrix 33
3.2 Comparison of different radio signaling methods 43
3.3 State transitions and their corresponding SIP requests 45

4.1 Using ENUM to resolve an E.164 phone number into a domain 51
4.2 Dial plan 52
4.3 Intuitive addressing schema 53
4.4 RTP forwarding rules 57
4.5 PTT signaling delays 65
4.6 RTP packet overhead budget 66

6.1 Floor control actions 92

Bibliography

[3GP] 3rd Generation Partnership Project. http://www.3gpp.org.

[AF03] F. Andreasen and B. Foster. *Media Gateway Control Protocol (MGCP) Version 1.0*, January 2003. RFC 3435.

[AST] Asterisk, the open source linux pbx. http://www.asterisk.org/.

[Aus] Austrocontrol. Sectional Drawing of the airspace above Linz. http://www.austrocontrol.at/ais/graphics/linzarea.pdf.

[Bila] Billy Biggs and Pekka Raisio and Jouni Vuorela and Juha Heinanen. dissipate2 SIP (Session Initiation Protocol) stack, integrated into KPhone SIP phone. http://www.wirlab.net/kphone/.

[Bilb] Billy Biggs and Pekka Raisio and Jouni Vuorela and Juha Heinanen. Kphone, SIP (Session Initiation Protocol) user agent for Linux. http://www.wirlab.net/kphone/.

[BLFM98] T. Berners-Lee, R. Fielding, and L. Masinter. *Uniform Resource Identifiers (URI): Generic Syntax*, August 1998. RFC 2396.

[Cam03] G. Camarillo. *Compressing the Session Initiation Protocol (SIP)*, February 2003. RFC 3486.

[Cat] Catalyst Communications Technologies, Inc. http://www.catcomtec.com/.

[CDK01] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems - Concepts and Designs.* Addison-Wesley, 2001.

[CER$^+$02] B. Campbell, Ed., J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle. *Session Initiation Protocol (SIP) Extension for Instant Messaging*, December 2002. RFC 3428.

[CGR+00] F. Cuervo, N. Greene, A. Rayhan, C. Huitema, B. Rosen, and J. Segers. *Megaco Protocol Version 1.0*, November 2000. RFC 3015.

[Cisa] Cisco Systems, Inc. Cisco Technical Solution Series: IP Telephony Solution Guide. Version 2.1, October 2001.

[Cisb] Cisco Systems, Inc. Land Mobile Radio over IP. http://www.cisco.com/univercd/cc/td/doc/product/software/ios123 /123newft/123t/123t_7/lmrip/gtlmrip.htm.

[Con] The Salutation Consortium. Salutation architecture specification (part-1), version 2.1.

[CRPP03] G. Camarillo, A. B. Roach, J. Peterson, and J. Peterson. *Mapping of Integrated Services Digital Network (ISDN) User Part (ISUP) Overlap Signalling to the Session Initiation Protocol (SIP)*, August 2003. RFC 3578.

[DKG03] Klaus Darilion, Wolfgang Kampichler, and Karl M. Goeschka. Event-based Radio Communication Signalling using the Session Initiation Protocol. In *Proceedings of the 11th IEEE International Conference on Networks (ICON 2003)*, 2003.

[DKGK04] Klaus Darilion, Wolfgang Kampichler, Karl M. Goeschka, and Christoph Kurth. A service environment for air traffic control based on SIP. In *Proceedings of the 'Hawaii International Conference On System Sciences HICSS-37'*, Jan 2004.

[Don00] S. Donovan. *The SIP INFO Method*, October 2000. RFC 2976.

[DoT] Federal Aviation Administration Department of Transportation. System requirements document next generation air/ground communications (nexcom). http://www.faa.gov/nexcom/nexdoc.htm.

[Eria] Ericsson and Motorola and Nokia and Siemens. Push-to-Talk over Cellular Specification. http://www.ericsson.com/multiservicenetworks/distr /PoC_specifications.ZIP.

[Erib] Ericsson, Motorola, Nokia, Siemens. Technical Specification: Push-to-talk over Cellular (PoC), Architecture V1.1.0 (2003-08), PoC Release 1.0.

[Eric] Ericsson, Motorola, Nokia, Siemens. Technical Specification: Push to Talk over Cellular (PoC), List Management and Do-not-Disturb V1.1.3 (2003-08), PoC Release 1.0.

[Erid] Ericsson, Motorola, Nokia, Siemens. Technical Specification: Push-To-Talk over Cellular (PoC), Signaling Flows V 1.1.3 (2003-08), PoC Release 1.0.

[Erie] Ericsson, Motorola, Nokia, Siemens. Technical Specification: Push-to-Talk over Cellular (PoC) User Plane, Transport Protocols V1.1.0 (2003-08), PoC Release 1.0 .

[Eura] Eurocontrol. Audiolan - radio/telephone voice communication system based on internet technologies. http://www.openatc.org/.

[Eurb] Eurocontrol. Eurocontrol guidelines for implementation support (egis), part 5, communication and navigation specifications, chapter 2, voice communication system (vcs). http://www.eurocontrol.int/eatmp/egd/vcs.pdf.

[Fal00] P. Faltstrom. *E.164 number and DNS*, September 2000. RFC 2916.

[FGM+99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. *Hypertext Transfer Protocol - HTTP/1.1*, June 1999. RFC 2616.

[GG99] Gilles Gawinowski and Laurent Guichard. Audiolan: Internet technology inside voice communication systems. In *Journal of Information Technology Impact, Vol. 1, No. 3*, pages 105–112, 1999.

[GMBO+03] M. Garcia-Martin, C. Bormann, J. Ott, R. Price, and A. B. Roach. *The Session Initiation Protocol (SIP) and Session Description Protocol (SDP) Static Dictionary for Signaling Compression (SigComp)*, February 2003. RFC 3485.

[GPA+03] C. Groves, M. Pantaleo, T. Anderson, T. Taylor, and Eds. *Gateway Control Protocol Version 1*, June 2003. RFC 3525.

[GPVD99] E. Guttman, C. Perkins, J. Veizades, and M. Day. *Service Location Protocol, Version 2*, June 1999. RFC 2608.

[Gra] Grandstream. Grandstream BudgeTone-100 SIP Phone Product Literature. http://www.grandstream.com/Product_Spec.pdf.

[GVE00] A. Gulbrandsen, P. Vixie, and L. Esibov. *A DNS RR for specifying the location of services (DNS SRV)*, February 2000. RFC 2782.

[H32] Packet-based multimedia communications systems, recommendation h.323 (07/03). ITU Telecommunication Standardization Sector.

[Hal02] D. Haldar. *Session control through RTCP, an extension to RTCP*, November 2002. Internet Draft draft-haldar-rtp-session-control-00.

[Hat03] W. Hatzinger. *Verfuegbarkeitsanalyse eines paketorientierten Sprachkommunikationssystems.* Diploma thesis at the Vienna University of Technology, 2003.

[Hay94] Simon Haykin. *Communication Systems, 3th Edition.* Wiley, 1994. page 122-131.

[HCF+03] H. Hannu, J. Christoffersson, S. Forsgren, K.-C. Leung, Z. Liu, and R. Price. *Signaling Compression (SigComp) - Extended Operations*, January 2003. RFC 3321.

[HJ98] M. Handley and V. Jacobson. *SDP: Session Description Protocol*, April 1998. RFC 2327.

[HK99] Martin Hitz and Gerti Kappel. *UML@Work.* dpunkt.verlag, 1999. page 234-236.

[HMM02] Elliotte R. Harold, W. Scott Means, and W. Scott Means. *XML in a Nutshell.* O'Reilly & Associates, 2002.

[I80] Iso/iec 15802-3: 1998 ansi/ieee std 802.1d, 1998 edition. incorporating IEEE supplements P802.1p, 802.1j-1996, 802.6k-1992, 802.11c-1998, and P802.12e.

[Inta] International Telecommunication Union. Recommendation E.164. The international public telecommunication numbering plan.

[Intb] Internet Radio Linking Project (IRLP). A Typical IRLP Node in Repeater Configuration. http://www.irlp.net/typical_node.html.

[Intc] Internet Radio Linking Project (IRLP). IRLP FAQ Page. http://www.irlp.net/faq.html#q25.

[IRL] The internet radio linking project (irlp). http://www.irlp.net/.

[JCJO92] I. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison Wesley, 1992.

[JVP00] Jan Janssen, Danny De Vleeschauwer, and Guido H. Petit. Delay and distortion bounds for packetized voice calls of traditional pstn quality. In *Proceedings of the 1st IP-Telephony Workshop (IPTel 2000), GMD Report 95, Berlin, Germany*, pages 105–110, 2000.

[Kam01] Wolfgang Kampichler. *Measurement of voice readiness in local area communication systems.* Dissertation at the Vienna University of Technology, No.:599622 II, 2001.

[KG03] Wolfgang Kampichler and Karl M. Goeschka. On measuring quality of service limitations in local area networks. In *Proceedings of the 'IEEE International Conference on Communications'*, 2003.

[KWW+98] S. Knight, D. Weaver, D. Whipple, R. Hinden, D. Mitzel, P. Hunt, P. Higginson, M. Shand, and A. Lindem. *Virtual Router Redundancy Protocol*, April 1998. RFC 2338.

[Lin] High-Availability Linux Project. http://www.linux-ha.org.

[Mar01] George Marsh. Air Traffic Control Switches. In *Avionics Magazine*, February 2001.

[MD00] M. Mealling and R. Daniel. *The Naming Authority Pointer (NAPTR) DNS Resource Record*, September 2000. RFC 2915.

[Mic] Microsoft. Microsoft Windows Messenger. http://www.microsoft.com/windows/messenger/default.asp.

[Moc87] P.V. Mockapetris. *Domain names - implementation and specification*, November 1987. RFC 1035.

[MUA90] K. Murano, S. Unagami, and F. Amano. Echo cancellation and applications. In *Communications Magazine, IEEE, Vol.28, Iss.1, Jan 1990*, pages 49–55, 1990.

[NIS] Air Traffic Control Corpus, NIST speech discs. Logan International Airport, Washington National Airport, Dallas Fort Worth Airport. Available from Linguistic Data Consortium, University of Pennsylvania; http://www.ldc.upenn.edu/.

[Nol94] M. S. Nolan. *Fundamentals of Air Traffic Control, Second Edition.* Wadsworth, 1994.

[Obj] Object Management Group. Unified Modeling Language. http://www.omg.org.

[Ogg01] C. Oggerino. *High Availability Network Fundamentals.* Cisco Press, 2001.

[Ohl03] N. Ohlmeier. *Design and Implementation of a High Availability SIP Server Architecture.* Diploma thesis at the Technische Universitaet Berlin, 2003.

[PBC$^+$03] R. Price, C. Bormann, J. Christoffersson, H. Hannu, Z. Liu, and J. Rosenberg. *Signaling Compression (SigComp)*, January 2003. RFC 3320.

[PLYC03] J. Peterson, H. Liu, J. Yu, and B. Campbell. *Using E.164 numbers with the Session Initiation Protocol (SIP)*, September 2003. Internet Draft draft-ietf-sipping-e164-04.

[PMKN04] Miikka Poikselka, Georg Mayer, Hisham Khartabil, and Aki Niemi. *The IMS: IP Multimedia Concepts and Services in the Mobile Domain.* Wiley, 2004.

[Pos81] J. Postel. *Internet Protocol*, September 1981. RFC 791.

[Ray] Raytheon JPS Communications. http://www.jps.com/.

[RLS99] J. Rosenberg, J. Lennox, and H. Schulzrinne. Programming Internet telephony services. In *Internet Computing, IEEE, Vol.3, Iss.3, Pages:63-72*, May/Jun 1999.

[Roa02] A. B. Roach. *Session Initiation Protocol (SIP)-Specific Event Notification*, June 2002. RFC 3265.

[RS02a] J. Rosenberg and H. Schulzrinne. *An Offer/Answer Model with Session Description Protocol (SDP)*, June 2002. RFC 3264.

[RS02b] J. Rosenberg and H. Schulzrinne. *Session Initiation Protocol (SIP): Locating SIP Servers*, June 2002. RFC 3263.

[RSC$^+$02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. *SIP: Session Initiation Protocol*, June 2002. RFC 3261.

[RZR02] J. Radecki, Z. Zilic, and K. Radecka. Echo cancellation in ip networks. In *Circuits and Systems, 2002. MWSCAS-2002. The 2002 45th Midwest Symposium on, Vol.2, Iss., 4-7 Aug. 2002*, pages II–219– II–222 vol.2, 2002.

[SCFJ03] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications*, July 2003. RFC 3550.

[Sim] Simon Morlat. Linphone, SIP user agent for Linux. http://linphone.org/.

[SJ01] Henry Sinreich and Alan B. Johnston. *Internet Communications using SIP.* Wiley, 2001.

[Sky] Skyguide. Skyguide - Air Traffic Management Processes and Procedures. http://www.skyguide.ch/e/control/process_e.html.

[Smi01] D.J. Smith. *Reliability, Maintainability and Risk - Practical Methods for Engineers, 6th Edition.* Butterworth-Heinemann, 2001.

[spe] Speak freely. http://speak-freely.sourceforge.net/.

[TEL] TELEX Communications, Inc. http://www.vega-signaling.com/.

[Tro] Trolltech. Qt, a multiplatform, C++ application development framework. http://www.trolltech.com/.

[TvS02] A.S. Tanenbaum and M. van Steen. *Distributed Systems - Principles and Paradigms.* Prentice Hall, Upper Saddle River, New Jersey 07458, 2002.

[VS00] A. Vaha-Sipila. *URLs for Telephone Calls*, April 2000. RFC 2806.

[W3C] W3C. Soap version 1.2 part 0: Primer; w3c recommendation 24 june 2003.

Curriculum Vitae

Klaus Darilion
Institute of Computer Technology
Vienna University of Technology
Gußhausstrasse 27-29/384
A-1040 Wien, Austria
Email: `klaus.darilion@gmx.at`

2nd July, 1975	Born in Wels, Austria
1985 - 1989	Primary School, Wels
1989 - 1994	Higher Technical Engineering School for Electrial Engineering, Wels
1994 - 2000	Diplom-Ingenieur (M.Sc.) in Electrical Engineering with 'distinction' at the Vienna University of Technology. Thesis: A ticket sale solution with WAP and CORBA
10/2000 - 06/2001	Military Service
07/2001 - present	Research Assistant at the Institute of Computer Technology, Vienna University of Technology.