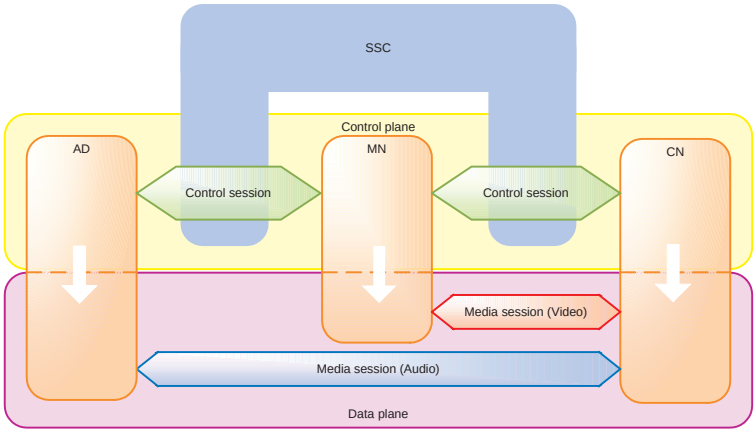
Partial session mobility in context aware IP-based multimedia subsystems

Thesis for a Master of Science degree in Telematics
from the University of Twente, Enschede, the Netherlands
Enschede, April 17, 2007

Jasper Aartse Tuijn



GRADUATION COMMITTEE:
Dr. ir. I.A. Widya (University of Twente)
Ir. D.J.A. Bijwaard (Alcatal-Lucent)
Dr. ir. B.J.F. van Beijnum (University of Twente)

Partial session mobility in context aware IP-based multimedia subsystems

Thesis for a Master of Science degree in Telematics
from the University of Twente, Enschede, the Netherlands
Enschede, April 17, 2007

Jasper Aartse Tuijn

UNIVERSITY OF TWENTE,
Faculty of Electrical Engineering, Mathemathics and Computer Science,
Department of Computer Science,
Division of Architecture and Services of Network Applications

Abstract

Nowadays, almost every person uses mobile devices to communicate with other people on a daily basis. Those mobile devices keep renewing rapidly following the latest developments in communication technologies supporting higher bandwidth and the newest services. The development of these technologies make new types of personalized context-aware services possible.

A typical goal of a personalized context-aware service could be to optimize the use of capable device in the close proximity of the user in ongoing communication sessions. This would typically be useful when a user who is engaged in an audio/video conference enters a meeting room. Whilst Upon entering, all media session components until then running on the PDA are transferred from the PDA to the projector, hifi-set and webcam all present in the meeting room. When the user leaves the room all media components are transferred back to the PDA. Moving parts of a multimedia session between different devices is here defined as partial session mobility.

For SIP numerous methods exist to enable partial session mobility both refer- and invite-based. None of those methods does support the ability to initiate partial session mobility from within the network, and especially not in combination with partial session mobility initiated by the user. Because of a number of advantages in the invite-based method, this method is taken as basis for the developement a method that supports both network initiated partial session mobility and user initiated partial session mobility.

Acknowledgments

I thank my supervisor at Alcatal-Lucent, Dennis Bijwaard for all the productive conversations and ideas during the development of this thesis, for the helpfull comments on the text and for investing a lot of time in supervising me.

I also thank my primairy supervisor at the University of Twente, Ing Widya for giving very good feedback on my thesis, and especially the conceptual parts.

I would like to thank my girlfriend, Lianne Meppelink, for supporting me. I also thank my family for making this possible and supporting me.

Finally I thank the employees of Alcatel-Lucent in Enschede for the pleasant and productive working environment.

Jasper Aartse Tuijn

Enschede, the Netherlands
8 December 2006

Contents

List of Figures

List of Tables

List of Acronyms

AAA ..... authentication, authorization and accounting

AD ..... audio device

B2BUA ..... back-to-back user agent A node in the signalling path of a SIP session having a SIP-session with both participants, while connecting these sessions internally. A B2BUA differentiates from a proxy-server because it keeps track of SIP-dialogs, while a proxy-server (stateful) only keeps track of SIP-transactions.

CN ..... correspondent node The correspondent node is the node the user is having a conversation with.

GPS ..... global positioning system

GUI ..... graphical user interface

ISUP ..... ISDN user part

MARQS ..... mobility management, AAA, resource management, QoS, security

MN ..... mobile node The device used by the user as network-terminal.

PIDF ..... presence information data format

QoS ..... quality of service

SDP ..... session description protocol

SIB ..... seamless integration of broadcast

SIP ..... session initiated protocol

SIP-UA ..... SIP user agent

SM ..... session mobility

SSC ..... sub-session controller The application server handling the sub-sessions with the local device.

TM ..... terminal mobility

UA ..... user agent

UI ..... user interface

UM ..... user mobility

USP ..... ubiguitous and seamless pervasiveness

VD ..... video device The video device that will be used to transmit and/or receive video-streams.

VID ..... virtual identity

VoIP ..... voice over IP

Chapter 1

Introduction

This chapter gives an introduction to this Master Thesis. The first section describes the motivation that led to this assignment. Based on this the second section presents the objectives of the assignment. The third section describes the approach taken to achieve the objectives. The last section gives an overview of the structure of the remainder of this thesis.

1.1 Motivation

Nowadays, the use of mobile phones is completely integrated in the daily living of most people. The technologies being used for mobile phones are rapidly changing to include more advanced networking technologies that offer a higher bandwidth (e.g. UMTS and WiMax) making a new generation of services for the end-user possible.

Another development of the last few years is the growth in use of the Internet as medium for conversations, using both audio and video. Providers and internet providers already offer internet telephone services to their customers. A lot of those voice over IP (VoIP) services use session initiated protocol (SIP) as session control protocol.

Most Telecom providers are already searching for solutions to have a uniform core network that couples the different access networks that use different technologies (e.g. PSTN, GSM). A number of telephone providers (e.g. KPN) already start updating their telephone network to a completely IP-based multimedia network like IMS. IMS uses SIP as session control protocol.

Those IP-based networks can be accessed through different access networks using different kinds of access technologies. These access networks can also use wireless networks like GPRS, UMTS, WLAN and WiMax. Especially when users have mobile devices to access the telephony network mobility becomes an important aspect. Some examples of the importance of mobility in this context: switching from access point when driving on the highway, switching to another telecom provider because the primary provider is not available in the area (roaming), switching from UMTS to GSM at the moment the user leaves the UMTS covered area. In each of those examples the user should not experience any interruption.

As stated before new types of services are possible with those new developed technologies. One of the developments in this area is using available devices in the direct environment of the user to support the user in communication. An example of these types of services:
Suppose a large network operator installed web-cams across the centre of a large city. Customers that registered for the service get offers on their phone to use the web-cams while

they are in a conversation. When they accept the offer, the video from the web-cam is being transferred to the other user in the conversation.

We define the type of mobility illustrated in this example as partial session mobility. Internet draft [45] describes two methods to execute partial session transfers specifically for SIP. However those methods focus purely on partial session transfers initiated by the user or user-terminal and do not consider transfers initiated by another node in the network. Especially in IMS this latter feature would be very interesting because it would allow a telecom provider to offer network initiated partial session transfer as a service to the end-user as shown in the example above.

In the context of user-friendliness it is important for an end user to have control on the execution of a network initiated partial session transfer. At the moment a user cannot choose whether a partial session transfer should be executed, the user could be faced with an unwanted partial session transfer that cannot be rejected. This is why a network initiated partial session transfer should always be proposed to the user or user-terminal, after which the user or user-terminal decides if the transfer is allowed to be executed.

1.2 Objectives

Based on the motivation in the previous section we defined the main research question: *'How can different devices in the vicinity of the user be used to enhance an existing multimedia-call the user participates in?'.*

We define the ability of moving media-part(s) of a session between different devices while preserving the continuation of this session as partial session mobility. The actual move of a media-part to another device is called a partial session transfer. The main objective is to find a suitable way to make this possible. To come to a more detailed objective we did a pre-study to find interesting aspects related to the research question. As a result a number of sub-objectives have been defined to give the main objective a more detailed interpretation:

- The different parts of a multimedia-call should be individually transferred to other devices. This does not concern the discovery of devices that can be used, and the reasoning process to decide when which media-stream must be transferred to such a device.
  - Support terminal initiated partial session transfers. This means it must be possible for the user or user-terminal to start a partial session transfer.
  - Support network initiated partial session transfers. This means it must be possible that an assigned node in the network proposes a partial session transfer. After the user or user-terminal accepts the transfer, this node really executes the transfer.
  - Both terminal initiated and network initiated partial session transfers should be possible in a single call. This means for example that after a network initiated partial session transfer as media-stream can also be transferred by a terminal initiated partial session transfer.
  - It should be possible to retrieve an already transferred media-stream both terminal and network initiated. This means once a media-part of the session has been

transferred, it should be possible to transfer this media-part to the original device or another device, initiated by the terminal or network.

  - The initiator of the partial session transfer should have control on the media-parameters of a transferred media-stream. This means the node initiating the partial session transfer should be able to propose the media-parameters (e.g the codec) that will be used.

- Find a way to handle partial session mobility in an IMS-compliant way. This means it should be a SIP based solution, because SIP is the session control protocol being used in IMS.

Besides those objectives there are a number of other important principles that must be considered during the development of the solution. Those principles are not as unambiguous as the objectives described above, however can be used to compare different possible solution with each other. Below are those principles:

Minimize disruptions

At the moment a stream is being transferred to another device there is a change of a disruption in that streams. From a user perspective it is important to minimize this disruption. A simple concept that can be used to accomplish this is 'Make before break'. In the context of this objective this means a 'new' media-stream should be set up before the 'old' media-stream is being closed.

Robustness

The solution should try to correctly handle a network failure of the devices involved. E.g. in case of a sudden disconnect of a device that is being used to play a video-stream, it must be possible to transfer that video stream to another device.

Compatibility

If a solution defines extensions to SIP that must be supported by a number of nodes to enable partial session mobility, the impact of this extension on currently available SIP user agents should be taken into account. This means a solution that needs only a few SIP user agents to be extended instead of all SIP users agents has preference.

Separation of concerns

When developing a system it is important to keep it simple. One way to do this is making sure functionality is provided by the components that logically are meant to provide that functionality.

1.3 Approach

To provide an answer to the research question described above and to fulfil the objectives we divided this process in different parts:

- Literature study on state of the art. Here the focus is on the standards surrounding SIP and IMS, and on existing SIP based techniques used for session mobility and specifically partial session mobility. Also the deliverables of the Daidalos project [5], to which this work is aligned, must be studied.
- Analyse the possible solutions for partial session mobility and their implications in the scope of IMS.
- Propose a suitable solution for partial session mobility. This includes the development of a method based on an existing method for session mobility or partial session mobility.
- Validate the solution using a prototype implementation.
- Present and document the results. This includes writing a master thesis, giving a presentation of the work done, and demonstrating the prototype.

The next section explains the scenario that is used as basis for the research and development described in this thesis.

1.4 Scenario

The scenario described in this section is partly derived from the scenarios in Internet Draft [34] and from the scenarios defined in Daidalos [20]. This scenario gives a typical use case where audio and/or video streams are transferred to different devices.

At the company Paul is working for, all employees are equipped with a PDA which they use for a number of purposes including video-conversations with colleagues located at other offices. To facilitate these communications optimally, the company uses a centralized system that helps employees in a conversation to use stationary multimedia devices located in the different rooms. Therefore the location and capabilities of all those stationary devices are known in this centralized system. The system also has an up-to-date view of the location of all the PDAs being used by the employees.

Paul is at work having a videoconference on his PDA with a colleague located at another office. While having the videoconference, Paul enters a conference room. This conference room is equipped with a beamer, web cam, microphone and sound system. A screen pops up on Paul's PDA, it gives Paul the opportunity to use the beamer for displaying the video received from the colleague at the other office. Paul decides to accept this offer, after which the video is immediately transferred to the beamer.

Another screen pops up at Paul's PDA offering to use the web-cam located in the room to use as video-source for the video-stream; Paul also accepts this offer. The system does not display another offer for using the integrated sound system because Paul indicated in his preferences for the system (Those preferences are stored centrally for each employee) that he always wants to use the capabilities of his PDA to play and record audio.

Because another colleague has reserved the room Paul has to leave the room during the videoconference. Before doing so he wants to transfer the video streams back to his PDA.

The user interface on his PDA shows the two video streams connected to respectively the beamer and web cam. Paul clicks on both streams and, given the option to transfer the streams back, he chooses to retrieve the streams on his PDA. Paul immediately notices the transfer of the streams back to his PDA, after which he can leave the room without missing any part of the video-conversation.

1.5 Document structure

The remainder of this thesis is structured as follows. Chapter 2 introduces a number of concepts that are important to the objective. The succeeding chapter introduces a number of technologies that are necessary as background information for the chapters after that. Chapter 4 describes the technical requirement based on the objective and uses these requirements to evaluate current solutions for partial session mobility.

In chapter 5 one of the current solutions is used as basis to develop a suitable solution. Chapter 6 describes the design of the different components and chapter 7 describes how these are implemented as a prototype. In chapter 8 the implemented prototype is used to validate the method developed. Finally chapter 9 contains the conclusions and future work.

Chapter 2

Concepts

This chapter describes the basic concepts of partial session mobility, this gives a clear view to the reader about what partial session mobility means and what concepts are involved. The concepts outlined in this chapter are used correspondingly in the remainder of this thesis, this gives the reader a good basis to understand, qualify and evaluate considerations made in this thesis. Each of the sections below describes a specific basic concept of partial session mobility.

2.1 Session

In the context of the objective of this thesis a session is seen as a lasting connection between two or more nodes located in a network. Hereby a session is always related to connections that concern multi-media communication. In this context we consider sessions on two different levels:

- The control plane. We define a session on the control plane as a control-session. A control-session defines and controls sessions that exist on the data-plane. During the control-session, the participants negotiate the media-parameters being used in the sessions on the data plane.
- The data plane. We define a session on the data plane as a media-session. A media-session consists of a media-stream between two or more endpoints; this media-stream can be two-way or one-way. When the media-session has been set up the participants send and/or receive media as specified by the control-session. A media-session can only exist if it has been negotiated in a control session; this means a control-session exists before the media-sessions specified by the control-session exists.

The example below shows the relation between the control plane and data plane. This is also illustrated by figure 2.1.
A mobile node (MN) wants to set up a video and audio call with the corresponding node (CN); therefore MN invites CN to start a control- session. While setting up this control-session MN and CN negotiate the parameters of the media-sessions they want to initiate. These parameters contain information about the type of media, the codec's, the protocol and endpoint-addresses. After both come to an agreement, the control-session is active. In the next step, both nodes set up the media-sessions as negotiated in the control-session. The set

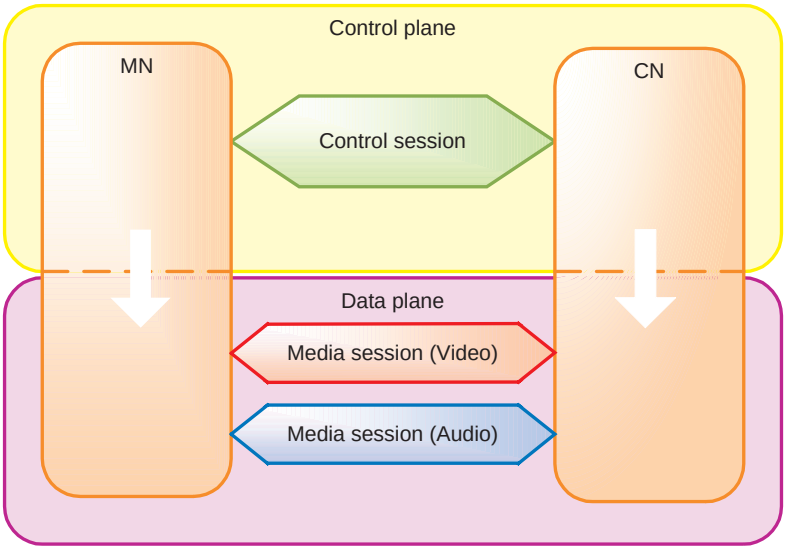


Figure 2.1: Session

up of the media-sessions does mean that both nodes start sending and receiving the audio and video-data using the protocol as negotiated.

In the example the two end-nodes are called mobile node (MN) and correspondent node (CN) The MN is the terminal the user uses as primary interface to communicate, this terminal is not limited to a mobile device, it can also be a fixed device. The CN is the terminal used by the user on the other side of the communication line. The remainder of this thesis also uses the same terminology. In the remainder of this thesis the term session should be interpreted as the control-session as described in this section, unless specifically called media-session.

2.2 Mobility

In this document we refer to mobility as the ability to stay connected to services and with other users while moving; moving can also mean not physical moving from one location to another, this can be illustrated by an example: Suppose that a guy with a GSM in his pants is at work, sitting at his desk. While the guy is sitting there the GSM switches a number of times to another GSM access point, because the signal strength varies. In this example the person is not moving, while mobility is necessary to make sure the access point is used that has the best signal strength. To explain this concept in more details we consider three different kinds of mobility [16]:

- *Terminal mobility (TM)* See [16][48]) is a form of mobility where terminals can switch its point of attachment, without services being disturbed/interrupted. This also includes the case where the other access point uses another access router and access network.

- *User mobility (UM)* (See [16]) handles users that switch to another terminal. When a user switches to another terminal the user should be able to access its own services. This

does not include the ability to continue an ongoing session while switching to another terminal.

- *Session mobility (SM)* (See [16][48]) handles sessions that should be able to move to other terminals or interfaces, while not disrupting this session.

As defined in section 4.2.2 of Daidalos deliverable D311 [16] based on their procedure there are a limited number of effective scenarios concerning mobility:

1. *UM only or TM only.* TM and UM, both without active sessions, lead to the same procedure. With TM the same terminal is used to connect to another access point, while with UM, the user switches to a different terminal. In each situation the user has to re-register. TM with active sessions sometimes also leads to the same procedure in case TM transparently makes sure layers on top do not notice the transfer.

2. *TM and SM.* The terminal changes to another access point to the network while there is an active session running. Because the terminal changes from access point and there is an active session, also session mobility is necessary. As describe above in some cases of TM this scenario does not apply because TM might transparently make sure layers on top do not notice the transfer.

3. *UM and SM or SM only.* The switch to another terminal with active session(s) and the redirect of a session to another user lead to the same procedure. In both cases SM leads to the transfer of the session to another terminal. If the user of the targeted terminal is another user then the user at the source terminal the procedure does not change.

This thesis focuses on scenario three, because this mobility scenario corresponds with the scenario given in section 1.4.

2.3 Partial session mobility

As described before from a user perspective partial session mobility means media-streams endpoints can be individually moved to another device. This section shows how this can be mapped on the concepts of sessions as described above, using an example situation to clarify this.

Lets say the MN did set up a session with the CN, including an audio- and video-stream. This would mean on the control plane the MN and CN did set up a control session, in which two media-sessions (One for audio and one for video) have been negotiated. On the data plane those two media-sessions have been set up between the MN and CN.

When the MN wants to use an audio device (AD) to handle the audio, the MN and CN change the control session such that it defines the audio media-stream as a stream between the CN and AD. The CN would stop the audio media-session and start a new one with the AD. However, in this situation the AD has not been involved in negotiating this media-session on the control plane.

Either the MN or CN can involve the AD on the control plane to negotiate the media-session. The MN is the one that wants the AD to be involved, and there is no reason why the CN

would explicitly be involved in this because the CN should not be responsible for something the MN wants. That is why it is most logical that the MN sets up a control-session with the AD and negotiates the media-session between the CN and AD.

As a result the MN negotiated with both the CN and AD about the audio media-session while the MN is not involved anymore in this media-session. Both the AD and CN did set up the media-session while on the control plane they did not directly negotiate, the MN did this negotiation on behalf of the CN and AD as intermediary. As shown here the media-session related to the video-stream did not change, and still exists between the CN and MN. This is illustrated by figure 2.2.

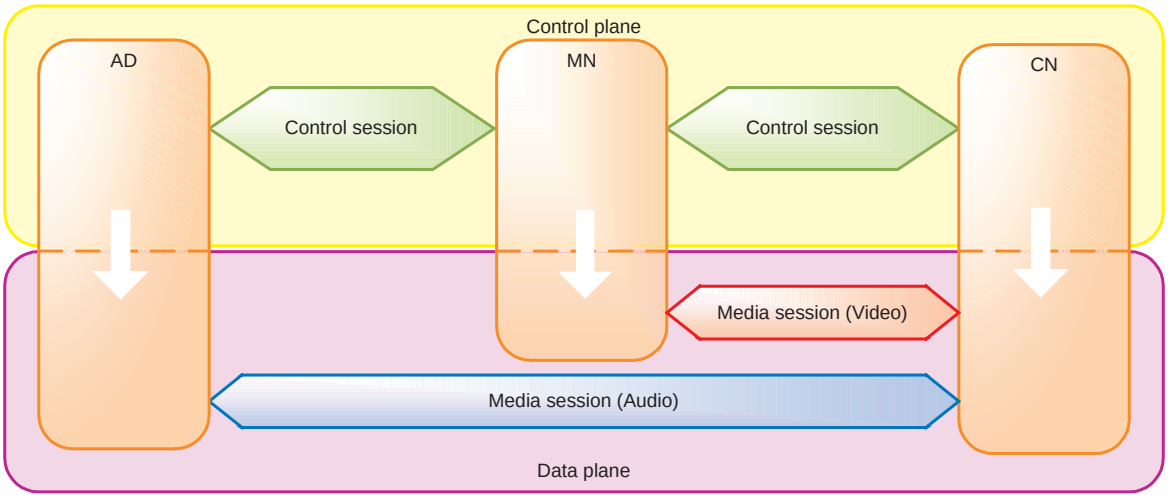


Figure 2.2: Partial session mobility

The MN involved the AD in the session it has with the CN. The control-session the MN set up with the AD has been purely set up to support the session the MN has with the CN. Because this session has a supportive nature we define it as a sub-session of the session between the MN and CN. We define the combination of the session between the MN and CN and the sub-sessions as a mobility session.

In this thesis every mobility session contains one device that plays the MN role. This means a device that is the MN in a mobility session can be the CN in another mobility session. The same principle goes for the CN and local devices (LDs) the MN uses in the mobility session to handle a stream (In the example described above the AD is considered to be an LD).

2.4 Network initiated partial session mobility

As described in section 1.1, network initiated partial session mobility means the ability to let a node in the network initiate a partial session transfer. This section describes how this would conceptually work.

As mentioned before in the objectives (See section 1.2, a partial session transfer should first

be proposed to the user. This proposal for a partial session transfer can arrive via different ways at the user or user-terminal, in this work we consider two possibilities:

- *The terminal.* Hereby the user self does initiate a partial session transfer using the user interface (UI) of the terminal. This possibility also considers the possibility that the terminal can act on behalf of the user, based on user-preferences stored at the terminal. This type of partial session mobility is also considered in the example described at the previous section.
- *The network.* Hereby another node connected to the network does propose a partial session transfer to the user or user-terminal (here the user-terminal can act on behalf of the user). To support this the node in the network cannot be any arbitrary node, because it must be informed about the exact specification of the current session(s) the user-terminal has. In this thesis we consider this node is located in the service provisioning part of the network, e.g. this provisioning part of the network could be located at a central place in a telecom provider network, a company network or a home network. In this thesis this node is called the sub-session controller (SSC).

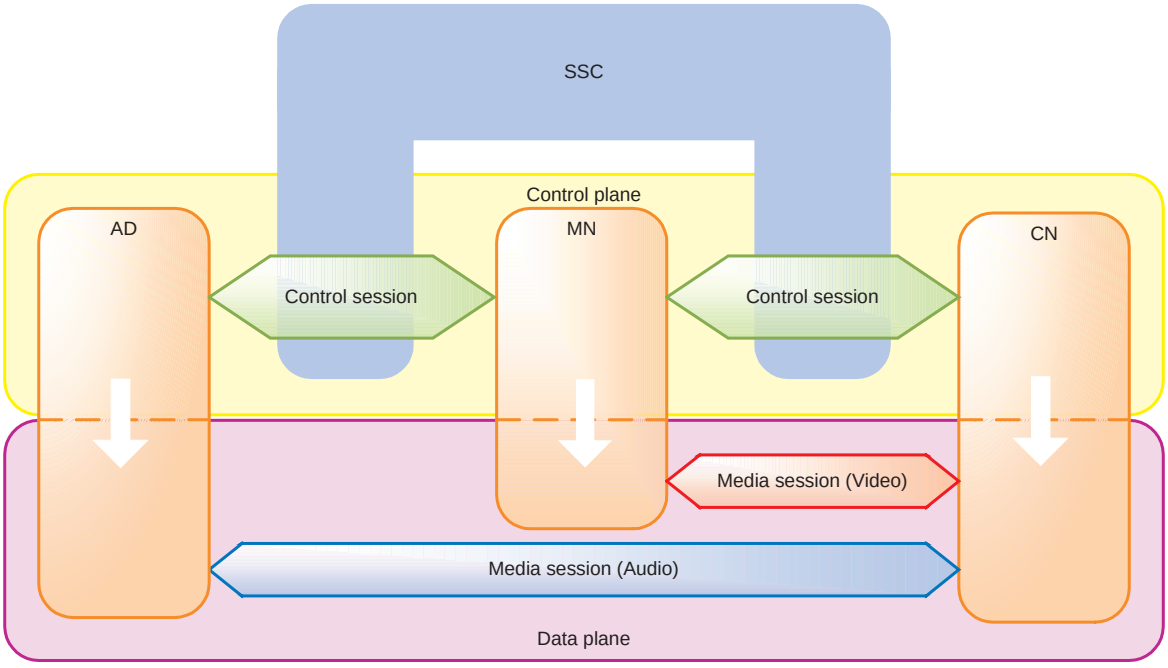


Figure 2.3: Network initiated partial session mobility

As mentioned before the SSC must be informed about the exact specification of the current session(s) the MN is involved in. Here we consider two mechanisms to make sure the SSC is informed about this, namely the SSC is involved in all control sessions as intermediary or letting the different nodes individually sent all changes made in these sessions they are involved in to the SSC. The latter mechanism has the disadvantage that more bandwidth is used at the nodes, because they need additional bandwidth to send the updates on the control

plane. Because of this it is the best solution to get the SSC involved as intermediary in all potential control sessions. Figure 2.3 illustrates the involvement of the SSC in all sessions the MN has, this way the SSC knows the exact situation, and can directly interact in those sessions to change them.

The concepts described in this chapter form the basic knowledge and understanding to develop a solution for partial session mobility that does both support terminal initiated and network initiated partial session mobility. The next chapter continues with technologies and recent developments that must be explored in order to develop a suitable solution.

Chapter 3

Background

This chapter describes techniques that are needed as background information for introducing further technical aspects, making decisions and coming to conclusions on the topic of partial session mobility in context aware IP-based multimedia subsystems.

3.1 Session initiation protocol (SIP)

SIP [44] is an application layer session management protocol developed by IETF MMUSIC working group [11] to specify an IP-based signalling protocol with a superset of functionality in the public switched telephone network (PSTN). [13]

3.1.1 Introduction

SIP standardizes the initialization, control and termination of multimedia sessions between two or more participants. These sessions can involve media like voice, video, application sharing, messaging, etc. Two nodes that want to setup a media-session have both to agree on the contents of the media. This mediation is not part of SIP although SIP does provide carrier functionality for protocols handling this mediation like the session description protocol (SDP) [31]; SDP describes the content of the session, e.g. the codec, the IP-endpoint, etc.

SIP is based on principles of the HTTP protocol [28], it uses the same request-response principle, is also human readable and uses many of the HTTP status codes, e.g. 200 (OK) and 404 (Not Found). The SIP messages also contain headers and a body; the headers have the same syntax and semantics as HTTP headers (Augmented Backus-Naur Form [25]). The Content-Type header field defines the data type of the content of the body.

To identify a SIP user, SIP uses URLs that have the form of an e-mail address such as 'sip:aartsetuijn@lucent.com'. For end-users it is comfortable to be able to use their own e-mail address for their SIP identity. SIP uses SIP proxies to route the SIP requests to the correct user. Before a SIP proxy knows the address the user is directly reachable on (E.g. aartsetuijn@135.85.87.11:5060), the user first must register at its home-domain (In this example 'lucent.com'). Therefore the user sends a REGISTER request to the SIP registrar of the home-domain, this message is routed via the proxy of that domain; the proxy knows the address of the registrar and proxies the REGISTER request to this registrar. The registrar stores the coupling between the SIP identity and the current address the user is reachable on.

The SIP proxy server within the home domain has access to this registrar, so it can always reach the user.

If user A (sip:userA@domainA) wants to send a SIP request destined for user B (sip:userB@domainB) via its VoIP telephone, the telephone routes the request to the proxy server in the domain A. The VoIP telephone discovers this proxy server using DNS [43]. The proxy server in domainA routes the request further on to the proxy server of domain B, which is also discovered using DNS. The media-stream that might be set up is not routed via those proxy-servers; instead the media will go directly to a specified IP-endpoint, which is specified by the corresponding SIP-endpoint in the SDP. Figure 3.1 illustrate this trapezoid construction.

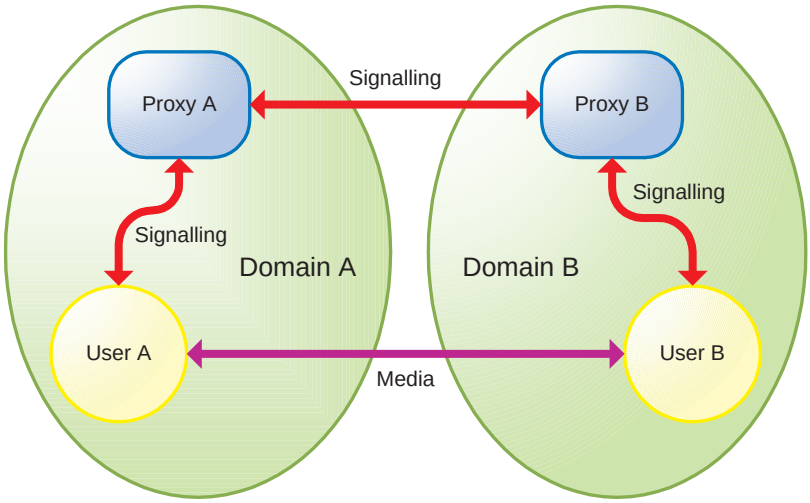


Figure 3.1: SIP trapezoid construction

To start a session the caller sends an INVITE SIP request to the callee. Each proxy server in the signalling path responds with a 100 (trying) response to the caller. This indicates the proxy server handles the routing of the INVITE request on behave of the caller. When the INVITE request arrives at the callee, this SIP endpoint responds with a 180 (Ringing) response. This response is routed through the same proxy servers, but in reverse direction. If the callee answers the call it response with OK (200); otherwise it sends an error response. The caller sends an acknowledgment to the callee in response to the OK (200) message to complete the three-way handshake. The three-way handshake, using the ACK message, is only used for (re-) INVITE requests.

Within this session initialization phase both nodes can exchange (Carried by the INVITE and OK messages) SDP messages to form an agreement on the content and end-points of the media-stream. If both nodes agree on the content of the session they start transmitting and receiving the media-stream. If one of the nodes wants to alter the media-stream during the session, it sends a re-INVITE containing the new media description. The other node answers with an OK response if it accepts the change.

Once the two nodes know each-others location, they normally do not use the proxy servers

anymore to route the SIP messages to the recipient. This means the proxy servers drop out of the signalling path after the INVITE request and OK response. If a proxy wants to stay in the signalling path it adds a 'Record-Route' field in the header of the INVITE message that contains the URL of the proxy. After this step the proxy stays in the signalling path of the complete peer-to-peer SIP relation between the caller and the callee; this relation is called a dialog. If one of the users 'hangs-up' (exits the dialog), its SIP-client sends a BYE request to the other node. This node confirms the reception of this BYE requests with an OK (200) response. Figure 3.2 shows the sequence diagram of the complete procedure in a situation where the proxy-servers do not stay in the signalling path and the user at domain B ends the call after it has been set up.

As described above SIP proxies help routing SIP messages to the intended user. Before a SIP message arrives at this user, the SIP message might have been routed via multiple SIP proxies. There are two types of SIP proxies: stateless and statefull. A stateless SIP proxy simply forwards SIP messages, where it makes the routing decision only based on that message. A statefull SIP proxy stores information (Typically information about transactions and/or complete sessions), which it can use afterwards. With this information it can affect the processing and routing of future messages. Specifically only a statefull proxy may fork messages to multiple destinations.

3.1.2 Session description protocol (SDP)

SIP uses the session description protocol [31] to describe multimedia sessions and negotiate the media-parameters. SDP itself does only define a format that can be used to describe multimedia sessions. RFC 3264 [32] provides an offer/answer-model using SDP to negotiate the media-streams in a session; SIP uses this model.

An SDP session description contains simple textual statements. These statements are related to the session-section or one of the media-sections. The statements in the session-section define some session-specific parameters and possibly some globally applicable parameters for the media-streams. The media-sections contain media-specific parameters; some of these can also be defined in the session-section. In that case the parameters in the media-section override the parameters in the session-section.

The box below gives an example of an SDP description:

```
v=0
o=aartsetuijn 0 0 IN IP4 135.85.86.61
s=-
c=IN IP4 135.85.86.61
t=0 0
m=audio 24224 RTP/AVP 0 3 4 5 16 6 17 14 8 15 18
m=video 24222 RTP/AVP 34 26 31 33
c=IN IP4 135.85.86.101
```

As can be seen each line starts with a single character followed by a '=' and some content. Each of those single characters is a parameter with a specific meaning:

v: Protocol version

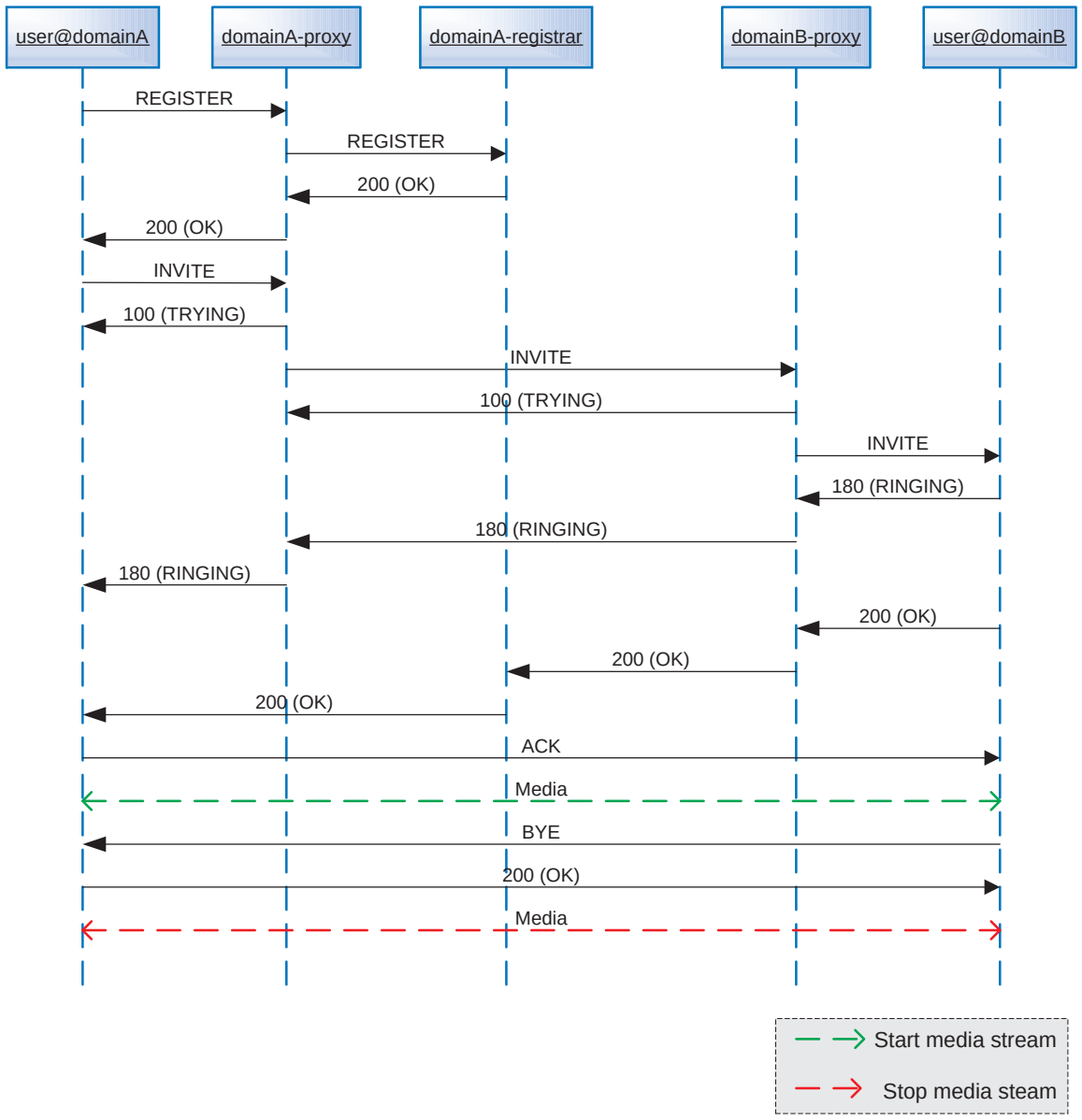


Figure 3.2: SIP sequence diagram

o: Origin, it contains a username (aartsetuijn), session id (0), version (0), network type (IN: Internet), address type (IP4: IPv4) and IP-address (135.85.86.61)

s: Session name, reasonably defined for multicast sessions, but in this case not defined ('-').

c: Connection data, it contains the network type (IN), address type (IP4) and connection address (135.85.86.61 and 135.85.86.61). In the example the second occurrence of this parameter is specifically defined for that media-description while the first occurrence is

the general connection parameter defined in the session-section.

t: The start and stop times for a conference session

m: Media description, it contains the media (audio/video), port (24224/24222), transport (RTP/AVP) and the supported media formats containing numbers that point to the actual media format description.

As said before SIP uses an offer/answer-model to negotiate the details of the media-streams between the two end-nodes. RFC 3264 [32] describes how this offer/answer-model uses SDP to negotiate a session. Here we give a short description of how the basics of this model work. With this offer/answer-model both nodes that want to negotiate the Media-streams use SDP to describe their own capabilities regarding to what they can send and receive. The connection parameter (c) and the port in the Media description (m) only describe the endpoint the sending node can receive the media on.

In case a node can only send or receive a certain stream it marks the stream with respectively the 'a=sendonly' or 'a=recvonly' attribute. The offerer sends an SDP message describing its capabilities; the answerer sends an SDP message describing its matching capabilities with respect to the SDP message send by the offerer. If the offering SDP contains the attribute 'a=sendonly' the answer must contain the attribute 'a=recvonly' (if the answerer accepts the offered stream), and the other way around.

3.1.3 3rd party call control (3PCC)

3rd party call control is the ability of a SIP entity to initiate, control and end a call between two other SIP entities. There are different ways to accomplish this using functionality from RFC 3261 [44] only. In this section we explain only one of the possibilities, namely 'Flow I' as specified within RFC 3725 [42].

The SIP user agent (SIP-UA) that wishes to create a session between two other user agents is called the controller. This controller starts the process by sending an INVITE request without SDP data to the first user agent. If the first user accepts the INVITE, the user agent responds with a 200 response that contains an SDP offer. Now the controller can invite the second user agent using the offer it got from the first user agent. If the second user accepts the invitation, its user agent responds with a 200 response that contains the SDP offer from the first user agent. The controller sends an ACK to the second user and an ACK to the first user, containing the answer from the second user. Now that both user agents agreed on the session, they can set up the media session. Since the controller sent the offer and answer to the other user agent without altering it, both user agents know each other's IP endpoint for the session media. According to SIP the multimedia data is not routed via the controller but send directly between the two user agents. Figure 3.3 illustrate the message flow between the controller, the first user agent and the second user agent.

Besides creating a session, with 3PCC it is also possible to control a call mid-session. In this case the controller must be in the signalling path of the call between the two end-nodes as a back-to-back user agent (B2BUA) (See also RFC 3261 [44]), this way the controller has a SIP

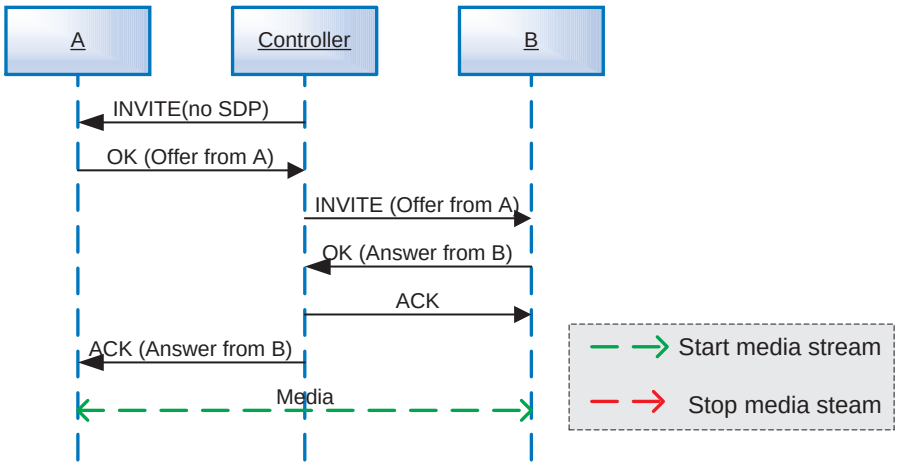


Figure 3.3: 3rd party call control, flow I

session with both the end-nodes and transfers the signalling messages between them. Figure 3.4 shows how a call is being transferred from B to C using 3rd party call control.

This solution is simple, straightforward and does not involve manipulation of SDP messages. The main drawback of the approach is that there can easily occur a timeout because the time between the invite to the first user agent and the acknowledgment of this session depends on the time it takes to invite the second user agent.

3.1.4 REFER header

The REFER method (See RFC 3515 [46]) is a method to let the recipient of the REFER request refer to a resource identified in the request. This provides the possibility to let another SIP entity invite a SIP entity that is identified in the initial REFER request. To support this functionality the REFER requests contains the new header field 'Refer-To', which contains the target to where the recipient must sent the request and information about the method of the request (E.g. Invite, Bye, etc.).

The recipient of a REFER request informs the sender about the results by means of NOTIFY requests (See RFC 3265 [41]). The recipient of the REFER request decides which information it wants to present to the sender about the status of the referred request and the response of the target. This means the recipient of the REFER request may only send part of the body/headers of the response(s). Figure 3.5 illustrate the message sequence diagram of the REFER method.

There are some security issues, which should be taken into account when working with the REFER method:

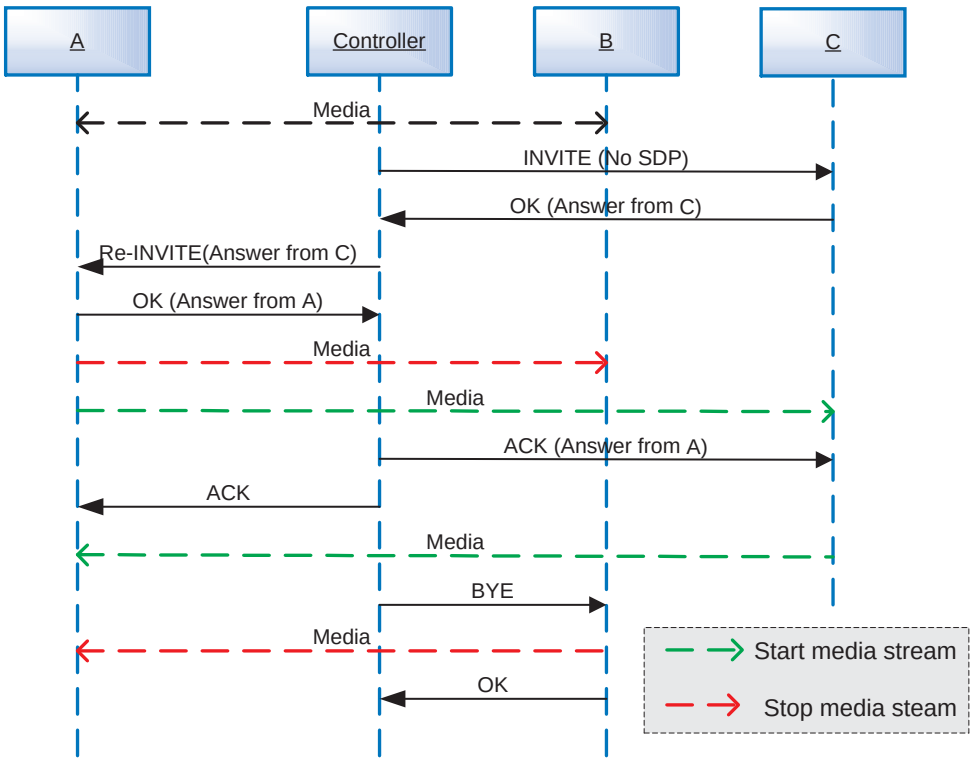


Figure 3.4: Using 3rd party call control to transfer a call

- The recipient of the REFER request can decide which information about the response of the target of the REFER is included in the notify messages. This could give the sender of the REFER request information about the target, which the target would not give the sender of the REFER request directly (E.g. the IP-endpoint of the user agent). To minimize this risk, the recipient of the REFER request should only return a carefully selected subset of the information available to the sender of the REFER request. This way, the least information about the targeted user agent is exposed.
- The sender of the REFER request sends the command to let the recipient send a certain message to the targeted user agent (UA). There should be some sort of protection against misuse of information concerning the recipient by the sender of the REFER. This can be done by letting the user of the receiving user agent make the choice about contacting the specified URI.

As an extension to the SIP REFER method Internet Draft [23] proposes a solution to refer to multiple SIP identities (Multiple refer) in a single REFER request. This solution uses the Refer-To header field as a pointer to an URI-list. This URI-list is included in the body. Also an extra option-tag 'multiple-refer' is added to the Require-header to indicate that the receiving user agent must support this extension to interpret the REFER request.

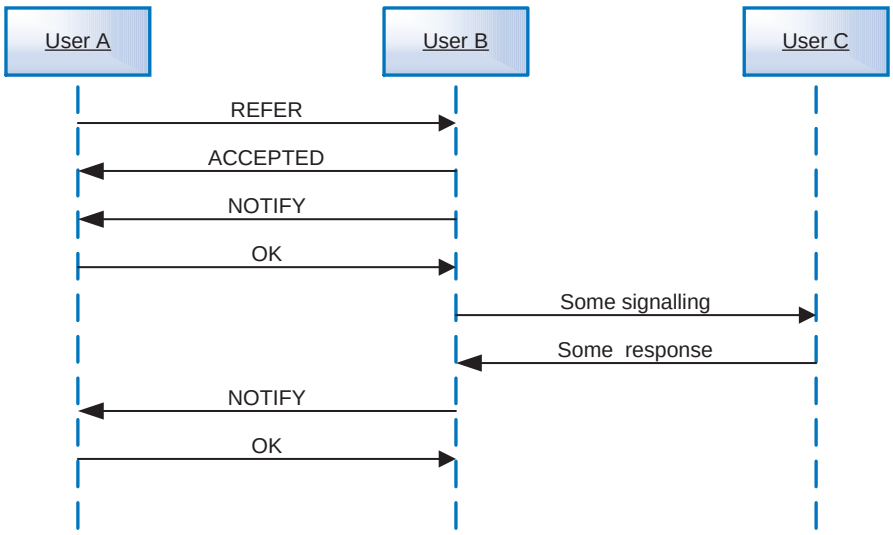


Figure 3.5: The REFER header

RFC 3891 [36] defines another extension to make it possible to replace an existing dialog with a new one. For this purpose a new header-field 'Replaces' is added, which can be used in an INVITE request. The Replaces-header contains information of the current existing dialog that should be replaced by the new one, as defined in the INVITE request. To replace an active dialog, the user agent that initiated the transfer must be authorized to do this. The user agent is automatically authorized if it is the user agent that is replaced. This extension can for example be used for 'parking' a call and retrieving the 'parked' call later on from the same or another device.

3.1.5 Security

Using a protocol like SIP involves considerations about security. There are a number of properties that bring along some security issues. RFC 3261 [44] describes the most important security issues and how they can be solved. The solutions offered take place at different phases; each of these phases and the security solutions is described below.

Security during registration on the network

As described in section 26.1.1 of RFC 3261 [44] there is the danger of a registration being hijacked. This issue is resolved using authentication between the Registrar and the user agent. The user agent sets up a TLS connection to the registrar. While connecting, the Registrar uses a certificate to authenticate itself. When the connection has been set up, the user agent uses this connection to send the SIP REGISTER request. The user agents uses HTTP Digest authentication [29] for authentication at the registrar.

Node-to-node security

Section 26.1.2 of RFC 3261 [44] describes a problem called 'impersonating a server'. This problem shows that an attacker can impersonates a SIP server in a certain domain. This means that the attacker receives messages sent to this domain. To solve this security issue all nodes in the signalling path should prove that they are who they say they are. This is done by authentication and a secure channel between the individual nodes using TLS [27] or IPsec [33]. This secure channel is used to send the SIP-messages over.

End-to-end security

An 'evil' SIP server in the signalling path can alter the body of the SIP message. E.g. this evil SIP server can alter an SDP message in the SIP body in order to set the media endpoint to a wiretapping device. Section 26.3.1 of RFC 3261 [44] offers a solution for this 'Tampering with Message Bodies' using end-to-end encryption of the SIP message body and possibly some of the header-fields. S/MIME [40] is used for this purpose. This way both users can be certain that SIP servers in between cannot read the body. End-to-end security might be not applicable in certain situations where certain SIP servers in the signalling path need to interpret the SIP body.

As an extension to the security solutions above, RFC 3329 [22] describes an extension of SIP that defines negotiation functionality for security protocols that is intended for the connection between a SIP UA and the first SIP-hop.

3.2 IP multimedia subsystem (IMS)

IP Multimedia Subsystem [6] is an IP-based system for providing multimedia services targeted at telecom providers, it is being developed by the 3rd Generation Partnership Project (3GPP) [1], a consortium of telecommunications standards bodies. It focuses on the continuing development and standardization of telecommunication systems like GSM, GPRS, EDGE and UMTS.

3.2.1 Introduction

The developments and technologies specified by 3GPP come together in the IP based Multimedia Subsystem. This system defines the IP-based core and the connections to all the other telecommunication technologies and services. The architecture of IMS can support all kind of services, which also contain the services telecommunication providers provide these days. These services must be available to the end-user while using different connection technologies and different telecommunication providers. In IMS, SIP is used to manage the multimedia sessions between different parts. Figure 3.6 shows the layered structure of IMS. This structure shows the dependencies of the layers with respect to each other, this means the upper layer uses the services provided by the lower layer. However these layers should not be read as protocol layers in which the upper layer offers the complete functionality to the user of the system. In IMS each of the layers offer directly certain functionalities to the end-users. Below the layers are described in more detail.

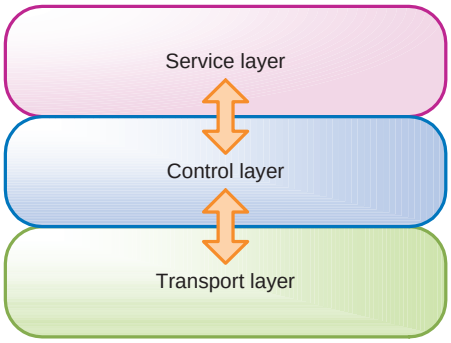


Figure 3.6: IMS layer structure

Transport layer

In the transport layer of IMS different transport network technologies can be used, depending on the connection the different users have. IMS supports packet switched networks (E.g. GPRS, UMTS, WLAN, WiMAX, DSL or cable) using IP and circuit switched networks using gateways between IP and the specific circuit switched networks (E.g. PSTN and GSM).

Control layer

The Control layer of IMS is responsible for call session control. In this layer routing decisions are made to ensure a call is routed to the correct user. To accommodate this functionality the control layer contains the user-information of the subscribers. The control layer is also responsible to connect certain services located in the service layer to specific sessions.

Service layer

Using the service layer of IMS the telecom provider can offer services to the subscribers. The Service layer is located on top of the control layer making sure services are only provided to users and sessions that are authorized for the specific services. To give an example, in this service layer the provider could offer a presence service to the subscribers, making it possible for those users to see the current presence of other users while sharing their own presence.

3.2.2 Architecture

This section describes the global architecture of IMS, this includes a description of the individual components and the functionality these components are responsible for and how the architecture of IMS makes sure subscribers are able to call each other using different access networks and different technologies. Figure 3.7 shows this architecture. The next sections describe the different components shown in the architecture in more detail.

Home subscriber server (HSS)

This entity holds the records of the subscribers of this domain. The subscribers are authenticated according to these records before they can set up sessions. The HSS also knows the locations of the users that are signed in.

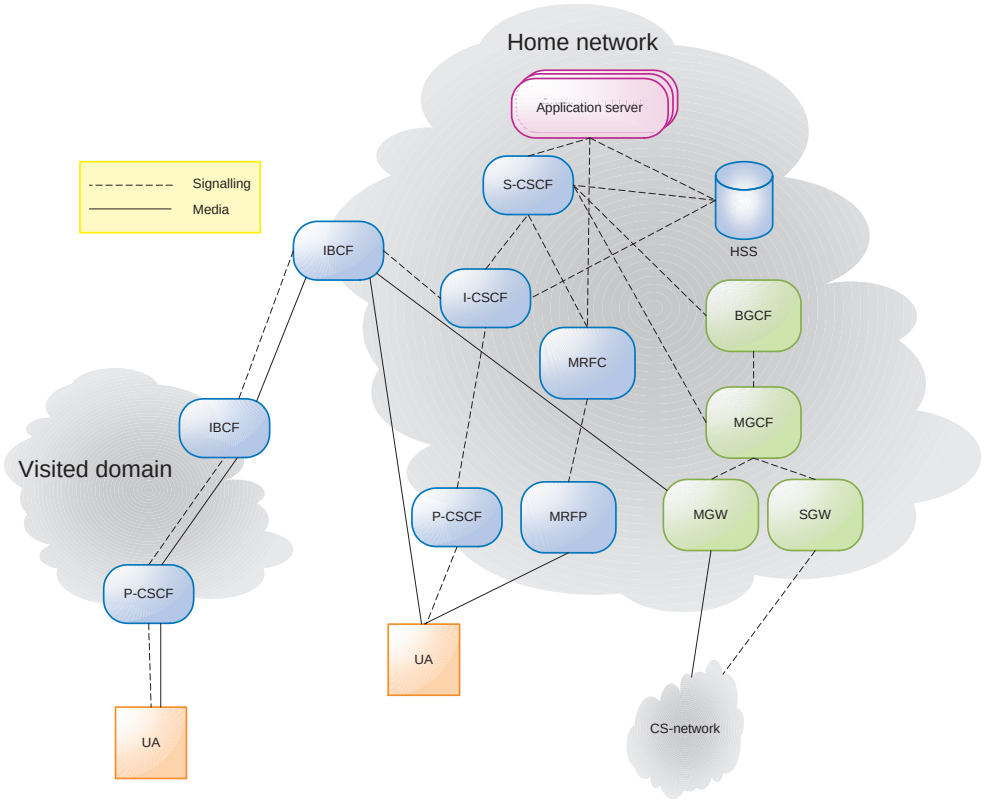


Figure 3.7: IMS architecture

Call session control function (CSCF)

The CSCF manages the SIP sessions and how these SIP sessions are routed through the system. There are three different roles, each having it own purpose in the big picture:

- *Serving (S-CSCF)*:
  This entity actually handles the sessions and stores the state of each session. It also interacts with application servers to support services.

- *Interrogating (I-CSCF)*:
  This entity is the contact point for SIP messages destined for a users currently located in corresponding domain or users of the corresponding network operator.

- *Proxy (P-CSCF)*:
  This is the contact point for any user agent in the IMS. The P-CSCF is located in the same network as the user agent and its only purpose is to route the SIP messages to the I-CSCF in the home-domain.

Application server (AS)

An application server offers services in an IMS. An AS can be included in the signalling path to let it act as a B2BUA. In that case it receives SIP messages from the S-CSCF, after

wich it can change these SIP messages, and it sends them back to the S-CSCF, which routes them further. The application server can reside in both the home network, or in an external network. To obtain information about subscribers it can interact with the HSS. Example applications: Presence, conference control, charging, and video telephony.

Interconnect border control function (IBCF)

This function defines the gateway between IMSs. Besides the signalling, this entity also controls the exchange of media between the different networks.

Breakout gateway control function (BGCF)

This entity controls the transfer to the CS domain (Circuit Switched domain). It selects which PSTN network is used to set up a session and chooses a local MGCF or a peer BGCF to control the gateway.

Media gateway control function (MGCF)

This entity is the gateway for the signalling to and from the CS network. It translates the SIP signalling to and from ISDN user part (ISUP), the signalling used to set up telephone calls in PSTNs and it keeps track of the state of the sessions that cross this gateway.

Media gateway (MGW)

This is the actual gateway for the media between the CS network and the PS network. Media conversion is done between the bearer channels on the CS-side and the media-stream on the PS-side.

Signalling gateway (SGW)

This is the actual gateway for the signalling between the CS network and the IP network, it transfers signalling from the MGCF to the CS network.

Media resource function controller (MRFC)

This entity controls the MRFP. It gets information from an AS and/or S-CSCF and controls the MRFP accordingly.

Media resource function processor (MRFP)

This entity can process media resources. This means it can be the source of them, mix them and transcode or analyze them.

3.2.3 Mobility

IMS includes mechanisms to support mobility [4], more specific terminal mobility. As described in section 2.2 this type of mobility makes sure a terminal can seamlessly switch to another base station. This also includes the possibility to switch to another provider, e.g. when somebody is travelling in a foreign country.

The only type of mobility IMS does currently support is terminal mobility, this means IMS does not yet support session mobility. However, research has been done on this subject resulting in a proposal to also add session mobility to IMS [37].

3.3 The IST Daidalos project

Daidalos [5] is a EU Framework Programme 6 Integrated Project. Daidalos stands for Designing Advanced network Interfaces for the Delivery and Administration of Location independent, Optimized personal Services. It is an operator driven project that consists of a network architecture, which focuses on a user-centred manageable communication infrastructure for the future.

3.3.1 Introduction

The goal of the Daidalos project is 'Seamless, pervasive access to content and services via heterogeneous networks that support user preferences and context'. The Daidalos project developed two scenarios that show the user perspective of the important subjects in the research goal. These two scenarios (See Daidalos Deliverable D111 [20]) describe 'A busy day in the life of Bart M. Watson' (Automobile Mobility) and 'A daily life on the campus' (Mobile University), each of them containing a number of scenes that describe the use of innovative mobile technologies from a users perspective. The scenarios form the foundation of the further research and development done in the project.

In the project a number of core subjects [18] are defined that are important from the users perspective:

- Mobility: Mobility is being described in more detail in section 2.2.
- Integration of heterogeneous network technologies: This means different wired and wireless network technologies are being integrated in the architecture of Daidalos to form a uniform whole that can take advantage of the services offered by Daidalos.
- Providing new profitable services: This means the concepts and ideas being used in the development must be validated according to the feasibility of the business case used for that concept or idea.
- Personalized multimedia services: This means the services being developed in the project must be focused on the user, where the service uses the personal preferences of the user to enhance the service to the user.

To enhance the subjects as described above, Daidalos addresses five key concepts [18]:

- MARQS. Mobility management, authentication, authorization and accounting (AAA), resource management, quality of service (QoS), security.
- VID. This means separating the user from the device, while making sure only a minimum of user details are exposed.
- USP. Ubiquitous and seamless pervasiveness.

- SIB. Seamless integration of broadcast
- Federation. This allows service providers and operators to offer and receive services.

3.3.2 Architecture

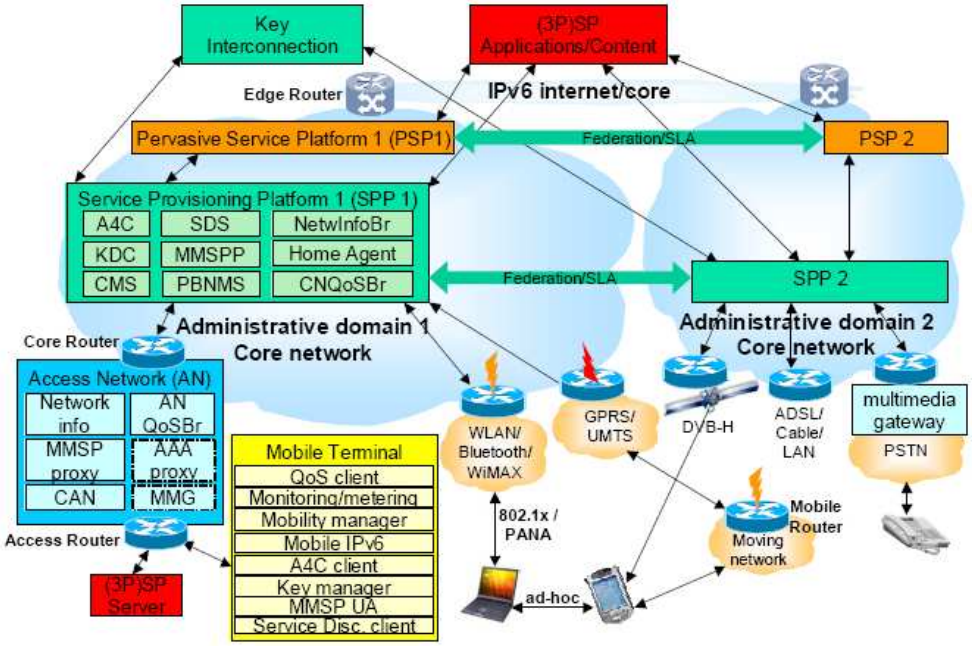


Figure 3.8: Daidalos architecture

Figure 3.8 illustrates the Daidalos architecture. In the architecture of daidalos [16] an administrative domain is for example the domain of a telecom provider; it is the core network of the system for a specific provider. As can be seen, different core networks of different administrative domains can communicate, this makes sure inter-domain sessions are possible. The architecture consists of a number of components that are distributed among different domains and different access networks. Below the main components are described:

- *Service Provisioning Platform.* This platform all service related issues are handled: identity management, multimedia services, mobility, security, authentication, authorization, auditing, accounting and charging. The sub-components shown in figure 3.8 handle these issues.
- *Pervasive Service Platform.* This platform is responsible for personalization and context awareness.
- *Access Network.* Via access networks users can connect to the core network of Daidalos. Different types of access routers make sure the user can access the network using different access technologies. Besides this the access network handles QoS and content adaptation.

- *Terminal.* The terminal is the device the user uses to connect to the network and control sessions. It contains functionality to authenticate, manage mobility and reserve QoS
- *(Third Party) Service Provider.* As the name already says the service provider provides services to the user, this can be applications, or content.
- *Key Interconnect.* This component makes sure certificate authorities in different domains can work together. This way users are able to connect via different domains using their own credentials.

3.4 Partial session mobility

This section gives an overview of current techniques to enable partial session mobility using SIP. Section 2.2 already described the concept of partial session mobility; the following sections describe four methods that implement partial session mobility in SIP, each having its pros and cons. The first two methods ('Mobile-node control' and 'Session handoff' are described in Internet Draft [45]. The third method (Multiple REFER messages) is described in section 4.11.3 of Daidalos deliverable D351 [17], while the fourth method is described in a master thesis [38]

3.4.1 'Mobile-node control'-mode

The 'Mobile-node control'-mode (See Internet Draft [45]) uses 3PCC (Flow I in RFC 3725 [42]) to let the mobile node transfer part of the session-media to another device, while keeping the control of the session. This split-up of session media is done by giving each media-part in the session a different end-point in the SDP-body.

Figure 3.9 shows an example of the message sequence diagram using the 'Mobile-node control'-mode. The example starts with an existing videoconference between the MN and the CN (arrow 1). The MN initiates the partial session transfer by inviting the audio device (AD) with the SDP-parameters of the CN (arrow 2). The AD accepts the invitation to start an audio session, by sending the OK with the audio-part of the SDP-parameters (arrow 3). The MN uses the same construction to invite the video device (VD) (arrow 5 and 6). After these steps the MN re-invites the CN using the SDP-parameters it got from the AD and VD (arrow 8), this way the CN starts sending the audio to the AD (arrow 11) while sending the video to the VD (arrow 12).

The example above uses the method described in section 9.1 of Internet Draft [45] to minimize the disruption of the media-streams during the transfer. This method uses the following mechanisms:

- After the Local devices received the INVITE message from the MN, they start listening and sending. According to RFC 3261 [44] the LDs have to wait sending before they receive the ACK message. By immediately starting to send media after the INVITE the disruption gets smaller. Also as soon as the LDs receive media-data, they use/play this data, even if they did not receive the ACK message yet.

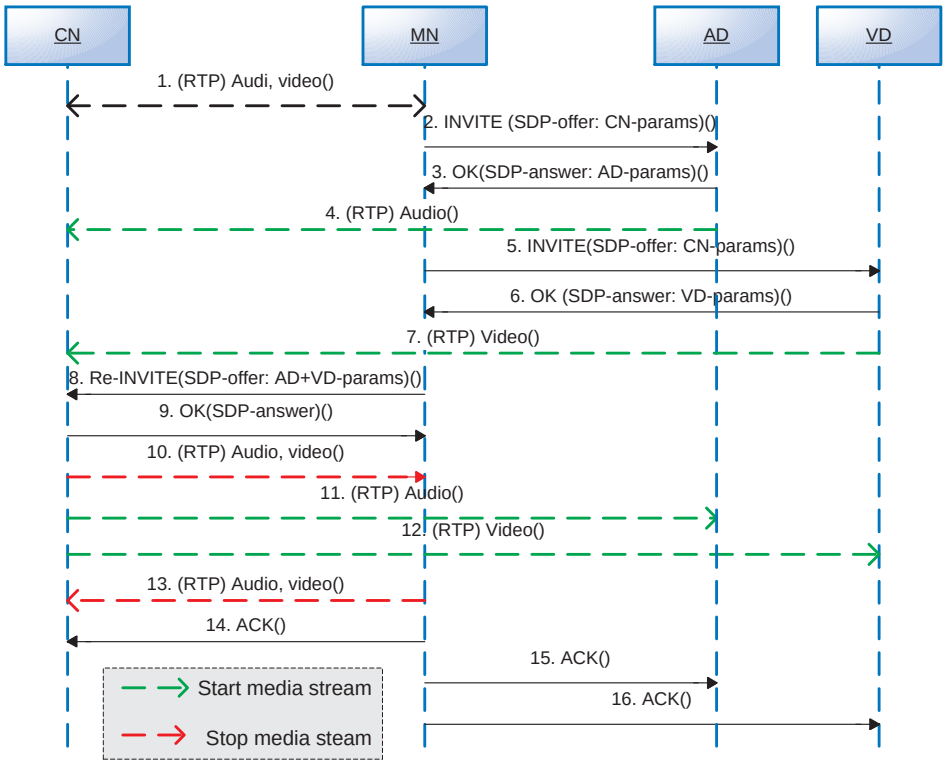


Figure 3.9: Message sequence diagram - 'Mobile-node control'-mode

- After sending the (Re-) INVITE message to the CN, the MN keeps sending media for a fixed amount of time. This way CN keeps receiving data even if the data that is being send by the LDs does not have been arrived yet.
- When the CN receives the re-INVITE message it immediately starts listening for the new data-stream while using/playing the old data-stream as long no data comes in on the new data-stream. This way, the CN starts using data from the LDs as soon as it receives it, but while that data does not arrive yet, it uses the data it still receives from the MN. The CN also immediately stops sending media to the MN and starts sending the media to the Local Devices after it receives the re-INVITE message. Hereby it makes sure the LD is being used as fast as possible to play the media.

Using this method, the CN does not notice any disruption in the stream, as long as the CN chooses a good period to keep sending the media. The Local devices immediately start playing the data when the CN starts sending to them, this makes the possible disruption very short.

According to Internet Draft [45] this method does support partial session transfers executed on both sides after each other. When both sides try to execute a partial session transfer at the same time, normal SIP behaviour (See the use of a 491-response in section 14.2 of RFC 3261 [44]) takes care of this by ensuring the transfers occur after each other.

Using this method it is important that the SDP-body in the Re-INVITE message contains the same order of media-streams. If a stream is being split-up (E.g. The video stream is being split-off from the audio-stream, whereas the video-stream is transferred to another device) the part of the original stream that stays at the original device is located at the same place in the SDP-body. The part of the stream that is being transferred to the other device must be added below the other streams in the SDP-body.

3.4.2 'Session handoff'-mode

The 'Session handoff'-mode (See Internet Draft [45]) uses SIP REFER messages with the 'replaces'-header extension (See section 3.1.4) to transfer the complete session to a multi-device system. This multi-device system makes sure the different media-parts in the session are transferred to the correct devices.

To transfer a part of the session-media to another device using SIP REFER messages it is necessary to set up multiple media sessions, one for each separate media-part. This method has the disadvantage that currently a single call cannot contain multiple SIP sessions. This means the CN will have a call with each single end-point devices. The CN must support this in order to make this method work.

To overcome this problem the 'Session handoff'-mode uses Multi-device systems (MDS). In an MDS there are multiple devices that can be reached via one SIP URI. This SIP identity points to one of the devices, that controls the MDS. This controller of the MDS is called the Multi-device system manager (MDSM). When the controller of the session between the CN and MN discovers the MDS, it chooses to hand-off the session to the MDS by referring the MDSM to the CN. The MDSM does set up sessions between itself and all the devices joined in the MDS (This is done using Third party call control, see section 3.1.3), after which it uses the REFER message to re-invite the CN using the combined SDP-parameters of all the devices in the MDS.

In the example illustrated by Figure 3.10 the MN has a video/audio session with the CN. The MN wants to use the audio-device to record and play all audio-streams of the session and the video-device to record and play all video-streams of the session. The AD and the VD are combined in a MDS, with the AD as MDSM (It acts as a B2BUA). The MD starts the partial session transfer by sending a REFER to the AD indicating that it should invite the CN. The AD might still have to set up the MDS by starting a session with the VD. The AD sends an INVITE to the CN containing the combined SDP of the AD and VD, it also contains a 'replace'-header to indicate that the session should replace the existing session between the CN and MN. The CN accepts the invite of the AD and the transfer of data between the CN and AD/VD is started. The AD informs the MN about the successful transfer using a NOTIFY, after which the MN can close the connection it has with the CN.

The 'Session handoff'-mode uses the same method as the 'Mobile-node control'-mode to minimize the disruption of the media-stream during the partial session transfer as described in the previous section.

In the case of a session transfer being attempted at both sides at the same time the Internet Draft [45] proposes to use the same 491 response (Request Pending) as being used for the

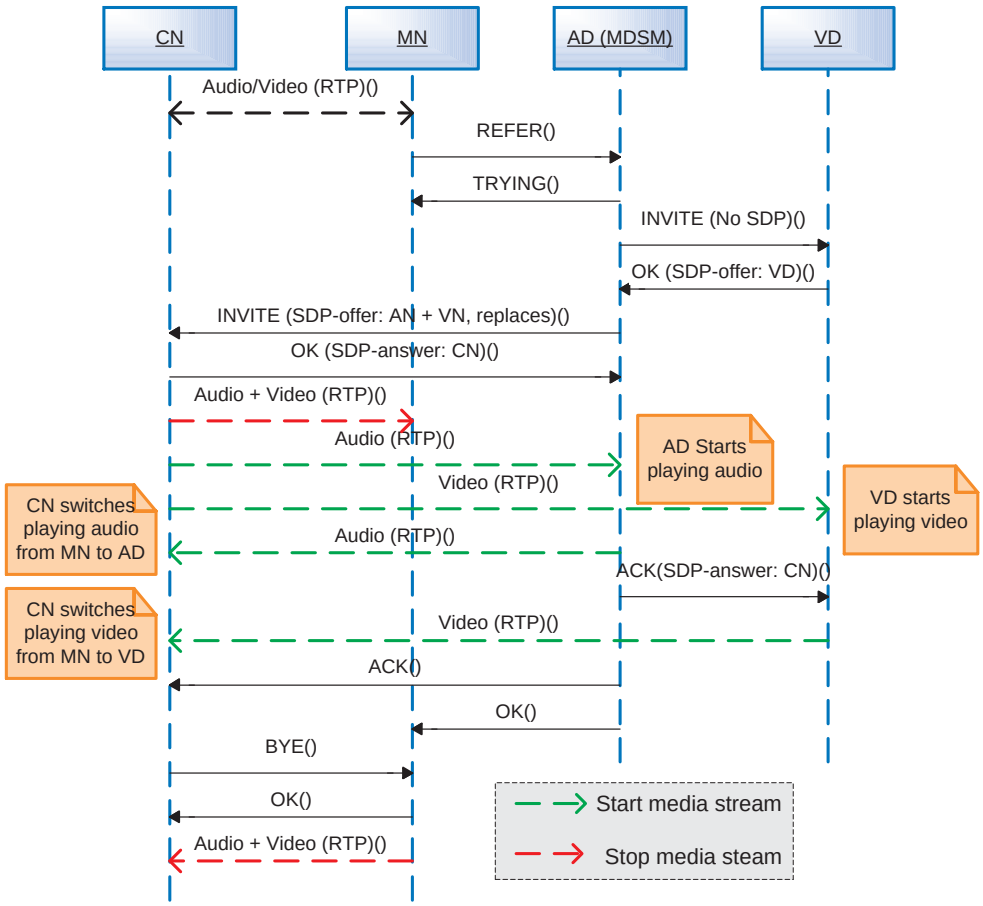


Figure 3.10: Message sequence diagram - 'Session handoff'-mode

'Mobile-node control'-mode. This way the REFERs are cancelled, and both sides try the session transfer again after a staggered time-interval expires.

3.4.3 Multiple-refer

This method also uses SIP REFER messages (See section 3.1.4) to transfer a part of the session media to another device. In comparison to the previous method, this method does not use MDSs to distribute the media-data to the correct devices. This solution uses the 'multiple-refer' extension (See section 3.1.4) to refer to multiple SIP identities in one REFER request.

The 'multiple-refer' mechanism lets the recipient of the REFER message sent one message to multiple targets. To support partial session transfer, it is necessary to sent different INVITE messages (Each target has another SDP-body) to multiple targets. So it must be possible to include all these message-templates in the REFER request. Daidalos deliverable D351 [17]

does not yet propose a specific extension needed to implement this.

In the example illustrated by Figure 3.11 the MN initiates the partial session transfer by sending a REFER message to the CN. This REFER message contains multiple targets, with an SDP-body for each of these targets that must be encapsulated in the INVITE message. The CN invites the AD and VD using these SDP-bodies after which the media-streams are transferred. The CN notifies the MN about the successful transfer, now the MN can disconnect the connection with the CN.

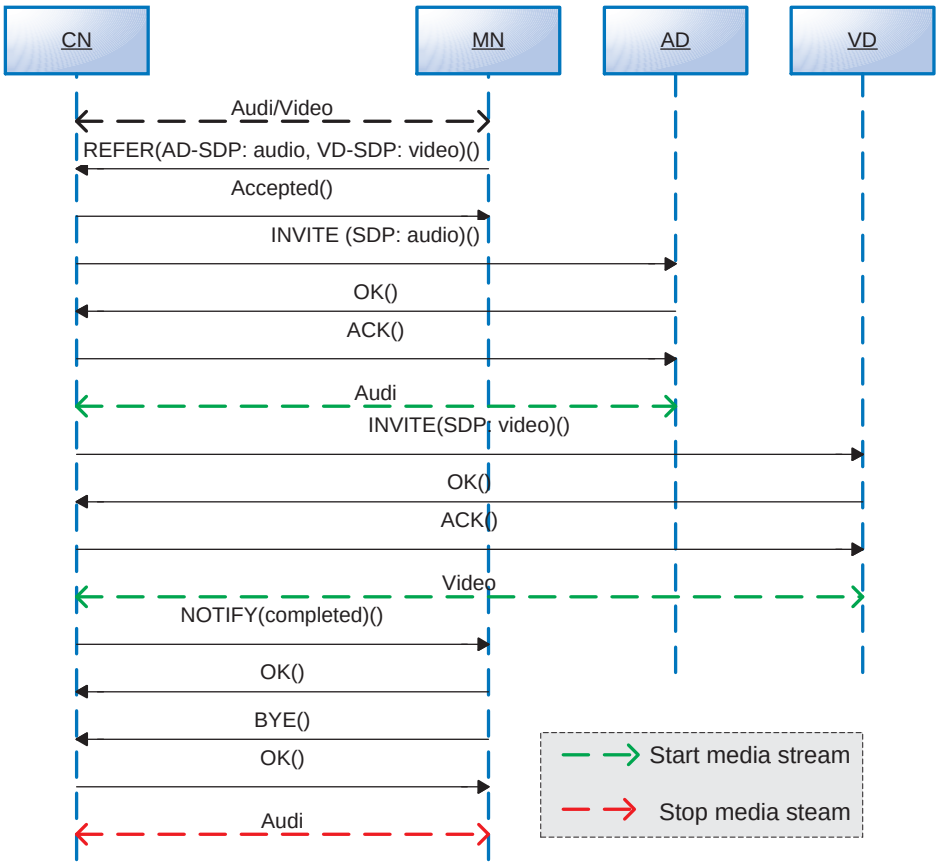


Figure 3.11: Message sequence diagram - 'multiple-refer'-method

3.4.4 Mobility header

The mobility header solution [38]) also uses SIP REFER messages to transfer a part of a session to another device. To solve the previous described problem about a call not being able to contain multiple SIP sessions (See section 3.4.2) this solution proposes an extension to SIP that introduces the mobility header.

The mobility header contains the unique call-id of the call that must be partially transferred to a new call and the information about the part of the call that must be transferred. The CN builds a tree structure to keep track of the calls that split-up the original call during the partial session transfer. The mobility header can occur in the INVITE, BYE and REFER message requests.

In the example illustrated by Figure 3.12 the MN initiates the transfer of the audio part of the stream to AD by sending a REFER to the AD indicating that it should invite the CN to transfer the audio-part of the session between the CN and MN to the AD. For this purpose the REFER contains the Mobility header containing the call-id of the session between the CN and MN; the header also contains a media-part indicating that only the audio-part of the media-stream must be transferred. The AD sends an INVITE message to the CN; this INVITE message also contains the mobility header that describing the call it should partly replace. The CN accepts the INVITE and the audio-stream between the CN and AD is being set up. After the MN got confirmation of the AD that the INVITE succeeded, it re-invites the CN to explicitly stop the audio-stream between the CN and MN. Now the transfer of the audio-part is completed, the MN initiates the transfer of the video-part using the same procedure.

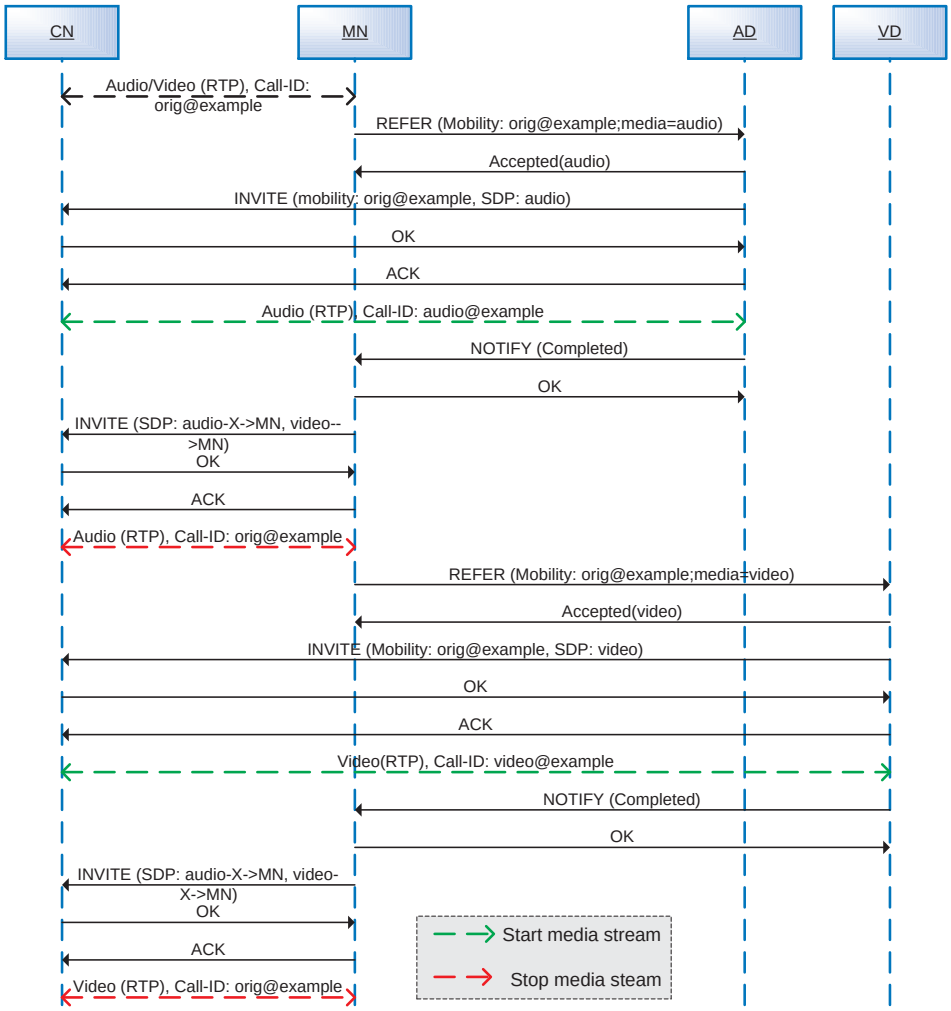


Figure 3.12: Message sequence diagram - 'mobility header'-method

Chapter 4

Requirements and evaluating available methods

This chapter describes how we can use the currently available methods to find a suitable solution that fulfils the objectives as described in section 1.2. The first section defines the technical requirements of the solution descriped in this thesis that form the basis of the development in the next chapter. A number of these requirements are being used by the subsequent sections to evaluate the current available methods.

4.1 Technical requirements

Based on the objectives and basic principles described in section 1.2, this section defines a number of technical requirements. Most requirements are also used as evaluation-criteria in the evaluation of the current available methods. Those methods are evaluated in the subsequent four sections. Some of the requirements are not used as evaluation criteria; these are described below together with the requirements that are used as evaluation criteria:

Terminal initiated partial session transfers

As described in the objectives (Section 1.2 the solution must support terminal initiated partial session transfers. The current available methods described in section 3.4 all support this, so this requirement used as an evaluation criteria would not distinguish the different methods. That is why this objective will not be used as an evaluation criterion.

Network initiated partial session transfers

As described in the objectives an assigned node in the network must be able to propose a partial session transfer to the user/user-terminal. The objectives address a number of functionalities and basic concepts that are related to network initiated partial session transfers:

- As the word 'propose' already indicates, there needs to be interaction with the user to have an agreement about the actual execution of the partial session transfer as also described in the objectives as the principle of user awareness and user consent. According to the objectives it is also possible that preferences set by the user are used to let the user-terminal decide on the actual execution of the partial session transfer.

  For the solution described in this thesis it is assumed that the following information is needed by the user to make the decision:

  - The exact media-stream that is being transferred.
  - The SIP-URI of the device the media-stream will be transferred to
  - The codec/quality of the stream after it has been transferred to the targeted device.

  In case the UA does not support network initiated partial session mobility, the UA cannot interpret the proposal sent by the SSC. A basic concepts in the development of protocols for the Internet is forward compatibility, this means a message contains information about which version of the protocol is being used for the message. In case the receiving node does not that version, it responds with a message indicating this. This principle and existing use of it in the used technologies should be prefered during the development of a solution.

  In case of a node that does not support necessary extentions the SSC should look at the user-preferences set for IMS to make the decission about actually executing the partial session transfer. None of the current available methods for partial session transfer does support this technical requirement, so it is not used as evaluation criteria.

- In this work the discovery of devices and reasoning about when a certain media-stream must be transferred to another device are not concerned. Here the focus is purely on the actual initiation and execution of the partial session transfer as described in the objectives (Section 1.2). Therefore the functionality to discover devices and execute the reasoning logic is separated from the actual network initiated partial session transfer. However to give the reader a clear view of the subject and the surrounding areas involved a number of examples of information that can be used in this reasoning process are attached in appendix A. In the remainder of this thesis it is considered that a context-reasoning node is present in the network that implements the reasoning process, and tells the SSC to initiate a partial session transfer at a certain moment in time.

- It must be possible to integrate the node that iniates partial session transfers from the network in IMS. In IMS the component meant to offer services to the end-users is called the application server as described in section 3.2.2. An application server is allowed to act as a B2BUA, which means it is allowed to act on behalf of the end-nodes involved on the control layer. An application server thus has full control on the involved control-sessions, and thereby also on the configuration of the media-sessions. Because none of the currently available methods do already support integration in IMS as an application server, this requirement is not considered as an evaluation criteria for these methods.

None of the current available methods do explicit support network initiated partial session transfer. However not every method is evenly able to be used for network initiated partial session transfers. Therefore the methods are evaluated by the possibility to support network initiated partial session transfers.

Combined terminal and network initiated partial session transfer

The objectives prescribe that functionality should be enclosed to support both terminal initiated and network initiated partial session transfers in a single call. So after a network

initiated partial session transfer it should still be possible to do a terminal initiated partial session transfer, and the other way arround.

This means both the node in the network that initiates partial session transfers as the terminal itself must always have an up-to-date view of the situation related to how the media-streams are connected to the different devices. Because none of the current available methods do support this as described by the actual evaluation, this objective is not used as an evaluation criterion.

Retrieve an already transferred media-stream

The objectives prescribe that the solution searched for must support the possibility to transfer an already transferred media-stream again to another device; this also includes the retrieval of the media-stream on the original device. This should be possible both network and terminal initiated.

Media-parameter control

As described in the objectives the initiator of the partial session transfer should have control on the media-parameters being used when transferring a media-stream. Ideally this means that the initiator is able to completely define the SDP body that describes the stream between the CN and LD. However the devices involved in the media stream eventually both also have to agree on the choice made by the initiator.

Minimize disruption

From the user perspective a minimization of the interruption while transferring a media-stream to another device is important. As described in the objectives the solution must be optimized in this issue by using the concept 'make before break'. This means the solution will be optimized by making sure the order in which media-streams are started and stopped is such that a new media-stream is being set up before the old media-stream is closed.

Because this optimization can be used in any method this will not be used as evaluation criteria. However this optimization will be used later on to optimize the final solution.

Robustness

As described in the objectives the solution should be robust against disconnecting end-nodes. This means that whenever a device involved in a session is disconnected from IMS, it is possible to retrieve the media-streams that where located on this device, are retrieved on the original device. So even when an involved device is disconnected from IMS it must be possible to undo a partial session transfer.

Compatibility

As also described in the objectives extensions to SIP can have great impact on the deployment of the solution. Therefore a solution that needs only a limited part of the amount of nodes to be upgraded is preferable to a solution that does need all nodes to be upgraded. Also a solution that does support a form of backwards compatibility is preferable to a solution that

does not. With backwards compatibility is meant the ability to both support extended and non-extended nodes in the same position, for example: Suppose a CN must be extended to fully support partial session mobility, however with limited functionality a non-extended CN is also supported.

Separation of concerns

As described in the objectives the concept of separation of concerns should be concerned when evaluating different solutions. In this context a solution that on the control plane does not involve the CN in a partial session transfer done in favour of the MN is preferable to a solution that does involve the CN in this situation.

4.2 'Mobile-node control'-mode

With this method the mobile node has control on the session and also keeps control on the session after a partial session transfer. The method itself does not take into account that there is a node in the network that should be able to control the session. Nevertheless the 'Mobile-node control'-mode does use 3rd party call control and with this technique it is possible for a network node to set up session between UAs. If the network node does set up a session between the UAs, after that it is also able to do a partial session transfer. This means network initiated partial session transfer is possible using the basic techniques used by this method.

While the 'Mobile-node control'-mode can support network initiated partial session transfers as described above it is not possible to support both network initiated partial session transfer and terminal initiated partial session transfer in a single session, because the point of control depends on the node that did set up the call. If the network node did set up the call, the MN does not control the session with the CN, because it only has a session with the network node.

The mobile node keeps control on the session because it is located in the signalling path of the session. This property of the 'Mobile-node control'-mode has a number of positive and negative consequences:

- The MN always has the capability to retrieve an already transferred stream.
- The MN has direct influence on the body of the invite message being sent to the LDs and CN; this way the MN has full control of the parameters of the media-streams.
- If a local device, which is being used in the session, suddenly disconnects, the mobile node can change the session accordingly by issuing a Re-INVITE message to the CN. This way, the stream can be changed away from the disconnected device, without the session being ended.
- When the MN suddenly disconnects, the media-streams between the CN and LDs continue while the signalling path is broken.

A big advantage of the 'Mobile-node control'-mode is that it is completely based on SIP functionality defined in RFC 3261 [44]. This means SIP clients do not need any additional

extensions to support it. As described in chapter 7 of Internet Draft [45] simultaneous session transfers with the 'Mobile-node control'-mode are handled by existing SIP functionality.

When using the 'Mobile-node control'-mode the CN is not directly involved in the transfer of a stream to another device on a signalling level. This is an advantage because when looking at the concept of separation of concerns it is not logically to involve the CN in the partial session transfer on the control plane.

4.3 'Session handoff'-mode

With this method the terminal gives away the direct control on a transferred part of the session. The terminal can only re-transfer a part of the session using a nested REFER, while the controlling node can easily reject this REFER message. This means the terminal can 'ask' the controlling node to transfer the stream(s) back, but it cannot force that node to transfer it back.

Another disadvantage of 'giving away' the direct control on the session occurs at the moment the current MDSM suddenly disconnects. In that case the MN is not able to transfer the stream back because it should 'ask' the MDSM nicely if it wants to transfer the streams back. Of course in this case the MDSM does not respond anymore. If the MN suddenly disconnects the session continues because the MDSM has control, however the user of the MN (Who probably is paying the bill) cannot stop the session anymore.

The SIP REFER message used in the 'Session handoff'-mode is being defined in RFC 3515 [46] as an extension to SIP. This means there could be a lot of UAs that do not support this method, this makes it harder to deploy a solution based on the REFER extension. Also the 'replaced'-header is being defined as an extension to SIP in RFC 3891 [36].

A big disadvantage of this method is that it is not possible for the MN to prescribe the exact media-parameters that should be used in the transferred streams. The MDSM decides the exact media-parameters that will be used after the partial session transfer at the moment the MN let the MDSM sent the INVITE to the CN. There is also the disadvantage that the MDSM decides which devices will be used, the MN does only know the MDSM and it capabilities, but it does not know what devices the MDSM exactly uses.

When combining the MN and the MDSM (The MN also acts as MDSM), the 'Session handoff'-mode would transform to the 'Mobile-node control'-mode. In that case the MN does not need REFER messages anymore to do anything, it only needs INVITE messages to transfer streams to and from local devices.

The 'Session handoff'-mode has just as the 'Mobile-node control'-mode not been designed with a node in mind that controls the session from an application server located in the network. However one could imagine that a BDSM being placed in the network. This would mean that only after the MN transferred the session to the BDSM the BDSM could initiate a partial session transfer from the network.

The 'Session handoff'-mode as defined in Internet Draft [45] does not define how and at which moment the MDSM does set up the MDS, the MDSM looks like a black-box with some

functionality, but without the actual description of how this functionality can be implemented. Also in literature nothing was found on this subject.

Also for this method chapter 7 of Internet Draft [45] describes how simultaneous session transfers are supported, however in contradiction to the 'Mobile-node control'-mode the solution for the 'Session handoff'-mode is not defined in the SIP RFC [44], it is described as an extension to the behaviour of SIP.

When looking at how this method scores related to the concept of separation of concerns the CN is not being involved in the partial session transfer on the control plane, this is an advantage of this method.

4.4 Multiple-refer

With the 'Multiple-refer' method the CN is the central point where all the related sessions combine. This means only the CN has direct control on the 'sub'-sessions with the LDs. The MN only has control on the session it has with the CN. At the moment the MN wants to start a partial session transfer it 'asks' the CN to make a connection with the local devices that must be involved using a REFER message extended with 'multiple-refer'. At this point the MN does control which devices are to be involved, but after the transfer it cannot direct influence the session that the CN sets up with those devices.

As with the 'Session handoff'-mode also another node besides the MN could send the REFER message to the CN, however this node would need the call-id of the session between the CN and MN. We assume the node in the network does have access to this information making sure network initiated partial session transfer is possible.

At the moment the MN wants to change something in the sub-sessions, e.g. change a codec, it has to use REFER messages to nicely ask the CN to change them. The CN is capable of changing the sub-sessions itself while the MN does not know anything about this, because it is just not involved in the session. In this case the MN is not able to decide whether or not it should change something in this sub-session or transfer the stream back, because it does not know the current situation.

When sending a REFER message that indicates to the receiver to send an INVITE message to the target node, it is not possible to include the exact body of the INVITE message in the REFER message. This means the sender of the REFER message cannot exactly specify which media-streams are transferred to which devices. A further extension to the 'multiple-refer' extension is necessary to solve this problem; this extension has not been developed yet.

As described above with this method the CN is the central node that combines the session between the CN and MN with the sessions it has with the local devices. The CN should somehow relate those sessions to each other; this means the CN contains the central intelligence needed for a partial session transfer to be executed. A lot of support for this is needed at the CN, while looking at the concept of separation of concerns ideally the CN would not be involved in a partial session transfer at all on the control plane.

As for now this method does not define any support for simultaneous session transfers. Probably the same method could be used as with the 'session handoff'-mode, but research should be done to find out exactly.

Because the CN is the central node in this solution this is also the weakest spot when it comes to sudden disconnects. If the CN disconnects, the MN does not have the possibility to explicitly close the connections with the local devices, while it does not have to pay for the connection with those devices. At the moment a LD disconnects, only the CN has direct control to change the sub-session, than MN can sent a REFER asking the CN to transfer the media-stream to the MN.

With the statement above we come to another negative property of this method, namely the CN sets up a session with the local devices instead of the MN. This means the accounting procedure is getting more complicated, charging can no longer be done based on the node that did set up a session. In this case the MN should pay for the connection while the CN does set up the session(s) with the local device(s).

The 'multiple-refer' method is not yet fully developed, there are a number of problems that must be solved and solutions that must be developed before this method can be used to support terminal initiated partial session mobility.

4.5 Mobility header

Just as the multiple-refer method, the mobility header method the CN is also the central node that combines the different sessions together and also here only the CN has direct control on the sub-sessions with the local devices. However with this method the MN does not send the REFER messages to the CN, but to the local devices which on their turn invite the CN. Even though when looking at this method with the concept of separation of concerns in mind, this becomes a disadvantage, because still the CN is directly involved because it keeps track of the sub-sessions as described above. Also with the mobility header method the MN does control which devices are involved when it initiates a partial session transfer.

This method has the same disadvantage as the 'multiple refer'-method, namely the MN does not have exact control on the SDP body that is being sent to the CN by the local devices. This means the mobility header method needs another extension to including the SDP body in the REFER message.

To make the mobility header method work, all nodes involved must support the 'mobility'-header. Besides the support needed by the CN (Just as the multiple-refer method), which already has the disadvantage of involving the CN in something that is not its business, also the local devices need to support the extension(s). This has a large influence on the deployment of a system that uses this method because every existing node must be updated before this method can be used.

To support network initiated partial session mobility a method should have the capability to have a node in the network that initiates the partial session transfer. With this method the MN sends REFER messages to the local devices to initiate the partial session transfer. A node from the network could do the same to initiate the transfer, however the MN does explicit

close the media-streams with the CN after the local device(s) did set up the connection with the CN. This cannot be done by the network node directly.

As with the 'multiple-refer' method there is no solution defined for simultaneous partial session transfers. However it is probably possible to use the extension proposed in the 'session handoff'-mode for this purpose, because this method is general for REFER messages.

At the moment the CN suddenly disconnects, the MN does not have direct influence on the session between the CN and the local devices, while the MN does have to pay for this connection. As with the 'multiple-refer' method this also causes charging difficulties because the CN does set up the connection with the local device(s), while the MN does have to pay for the sub-session.

The thesis specifying this method [38] does not explicitly define how the retrieval of a transferred stream works; however the MN can easily be involved again because it has still a session with the CN. By sending a re-invite to the CN indicating that the CN can send for example video and audio again to the MN, the MN re-activates the session. The only problematic issue that could arise is how the MN should disconnect the sessions the used local devices have with the CN. Before this method can be used it must be enhanced to fully support the retrieval of media-streams.

4.6 Conclusion

The previous four sections described the advantages and disadvantages of the currently available methods to execute partial session mobility using SIP based on the evaluation criteria mentioned in section 4.1. This section summarizes the results of this evaluation. To summarize the evaluation criteria as mentioned in section 4.1 below is a short list of the requirements:

1. Network initiated partial session transfers. Hereby it is only important if there are clear possibilities to let the method support network initiated partial session transfers.
2. Retrieve an already transferred media-stream
3. Media-parameter control
4. Robustness
5. Compatibility
6. Separation of concerns

The table below contains the evaluation results of the different methods compared with each other using the evaluation criteria mentioned. The evaluation criteria have all the same priority, which means the 'Mobile-node control'-mode is the most promising method to use as basis.

Below a summary is given of the evaluation criteria that differentiate the 'Mobile-node control'-mode from the other methods the most.

The methods 'Session handoff'-mode (See section 4.3), Multiple-refer (See section 4.4) and Mobility header (See section 4.5) all use REFER messages to execute a partial session transfer

Method	1	2	3	4	5	6
'Mobile-node control'-mode	+	+	+	+-	+	+
'Session handoff'-mode	+-	-	+	+-	-	+-
Multiple refer	+	-	-	+-	-	-
Mobility header	-	+-	-	+-	-	-

Table 4.1: Evaluation of currently available methods

while the 'Mobile-node control'-mode (See section 4.2) uses re-INVITE messages. All the REFER based messages have the disadvantage that it is not possible for the initiater of the partial session transfer to prescribe the exact SDP-body that has to be sent in the INVITE message that follows from the REFER message. Only the 'Mobile-node control'-mode that only uses re-INVITE messages does not have this problem.

Another problem that occurs at almost all methods that use REFER messages is that a partial session transfer cannot be undone, after a node transferred a media-stream, it is not anymore involved directly in the session that contains the transferred media-stream and therefore it cannot retrieve the media-stream directly. With the mobility header method however the original media-stream can be set up again, but full retrieval of the stream is not yet supported.

At both the multiple refer and mobility header method the CN is involved in a partial session transfer that is being executed for the MN. When looking at the concept of separation of concerns this is disadvantage when compared to the 'Mobile-node control'-mode and 'Session handoff'-mode. Besides this the 'Mobile-node control'-mode and the 'Session handoff'-mode both do not need an extensions of SIP.

None of the currently known methods described here are directly suitable in the context of this assignment, although those methods provide us with a good basis to start developing a more suitable solution. The next chapter describes the development of this solution. As already described this solution is based on the 'Mobile-node control'-mode.

Chapter 5

Development of a suitable method

As described in the previous chapter the 'Mobile-node control'-mode forms the basis of a more suitable method to achieve the objectives. This chapter describes the additions to the 'mobile-node control'-mode needed to fulfil the requirements as defined in section 4.1. The first section introduces the solution, while the succeeding sections go into more detail to describe how the requirements are being fulfilled.

5.1 Introduction

The previous chapter described the requirements based on the objectives (See section 1.2). The 'Mobile-node control'-modes already fulfils some of these requirements. Therefore, below an overview of the not yet supported functionality in the 'Mobile-node control'-mode and concepts that are important to keep in mind while developing the solution:

- Network initiated partial session mobility
  - The partial session transfer must be proposed to the user-terminal. This includes making sure this proposal contains information about the exact media-stream that is being transferred, the SIP-URI of the device the media-stream will be transferred to and the media-parameters of the media-stream after it has been transferred.
  - The SSC must be able to know when a SIP-UA does not understand the proposal, and handle accordingly.
  - It must be possible to integrate the SSC as an AS in IMS.
- Retrieve an already transferred media-stream. This must be supported for the combination of terminal and network initiated partial session mobility.
- The initiator must have as much control as possible on the media-parameters. This is already supported by the 'Mobile-node control'-mode, but it must also be supported when using combinations of terminal and network initiated partial session mobility.
- The concept 'Make before break' must be used to minimize the disruption in the media-streams while transferring the media-stream.
- Robustness. The 'Mobile-node control'-mode does not support the ideally wanted robustness, however robustness must be kept in mind.

- Compatibility. The 'Mobile-node control'-mode uses only SIP functionality as defined in RFC 3261 [44]. This way it is compatible with SIP-UAs that only support this functionality and it does not need SIP-UAs to support other extensions. While developing this method this should be kept in mind, to minimize the needed extensions necessary. The amount of SIP-UAs that should be extended in order to deploy the method should also be considered.
- Separation of concerns. While the 'Mobile-node control'-mode already does this very well, this must be kept in mind to make sure the method developed in this chapter optimally uses of this principle.

As described in section 3.4.1 the 'Mobile-node control'-mode uses an optimization to minimize the disruption of the media-streams during a transfer. As defined in the objective the solution should make use of the principle of 'make before break' to minimize the disruption of the involved media-stream while executing a partial session transfer. The optimization used by the 'Mobile-node control'-mode is not being used in the basis for the development of the solution dscribed in this chapter. The research necessary to find out if and how this optimization can be used in the solution presented in this chapter is considered as future work.

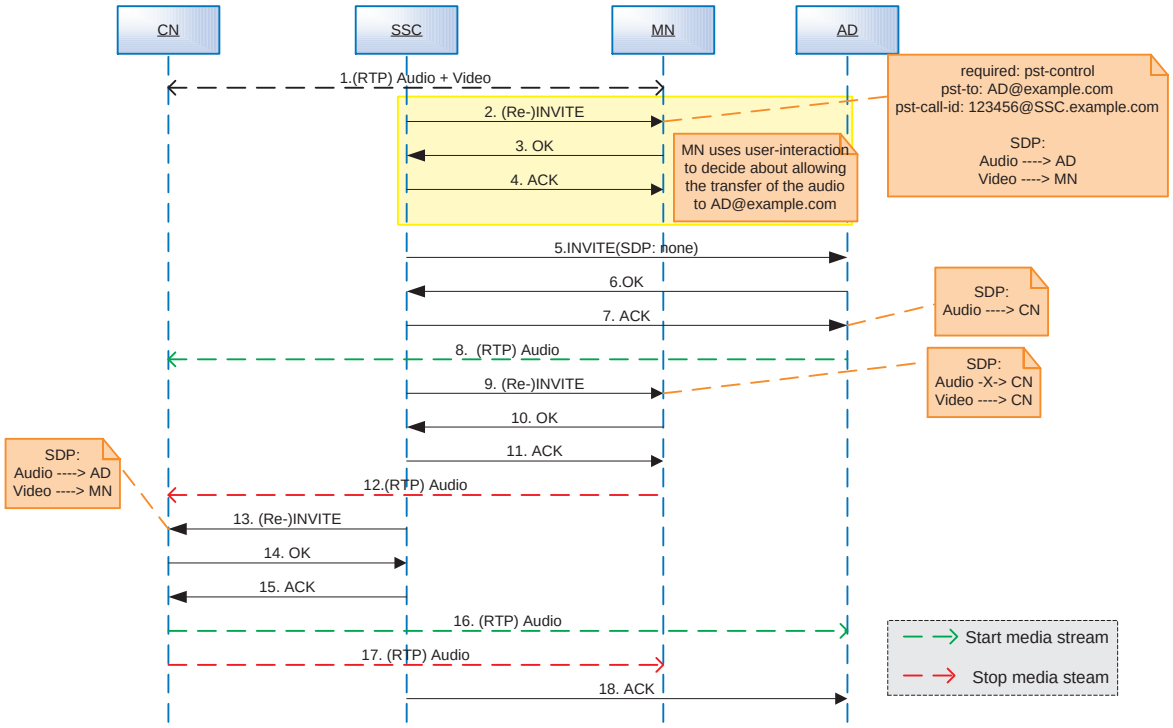


Figure 5.1: Message sequence diagram - Typical situation

Figure 5.1 shows the message sequence diagram of the developed solution in a typical situation where a network initiated partial session transfers is being executed. To explain the solution

in further detail the sections below each describe a part of the solution related to the related requirements described in the previous section.

5.2 The sub-session controller (SSC)

The SSC must be able to start sessions on behalf of nodes and interpret session between nodes. Therefore the SSC must implement B2BUA functionality (See RFC 3261 [44]), a B2BUA can participate in multiple sessions that it can connect with each other by acting on behalf of the end-nodes.

To make sure the SSC can offer its services to a certain MN, it must be located in the signalling path of all sessions the MN has with other nodes. Because the SSC can be integrated in IMS as an AS, it is automatically in the signalling path of sessions between the MN and other nodes if the MN wants the partial session transfer service.

As the SSC initiates the partial session transfer as shown by figure 5.1 here is a short explaination of the different steps. This diagram starts with a session that is already going on between the CN and MN (Arrow 1). In the next step (Arrow 2-4) the SSC proposes the partial session transfer to the MN, after which the MN accepts the partial session transfer. After that the SSC uses third-party invites the AD to start sending audio to the CN (Arrow 5-8). In the next step (Arrow 9-12) the MN is invited to stop sending audio to the CN. Than the SSC invites the CN to start sending audio to the AD instead of the MN (Arrow 13-17); nothing changes to the video-stream. The next section describes in detail how the SSC does propose a partial session transfer to the MN.

5.3 Proposing a partial session transfer

As described at the requirements in the first section of this chapter the SSC should propose a partial session transfer to the MN. The solution as illustrated in figure 5.1 uses an extended INVITE message (Arrow 2) send by the SSC to the MN on behalf of the CN. This extended INVITE message contains the method 'pst-control' in the required header. This implicates that according to SIP the receiver of this message is only allowed to process the message further in the application if it does support this extension.

If the MN that receives this extended INVITE message does not support the 'pst-control' extension it must, according to SIP, respond with a 420-response (Bad extension). After the SSC receives this response it knows the MN does not support the extension. In this case the SSC can refer to the user preferences in order to decide what to do next. Figure 5.2 illustrates the behaviour mentioned in case the user-preferences prescribe to continue the partial session transfer in this case.

As illustrated in figure 5.1 at arrow 3 the MN responds with an 200-response (OK), meaning the network initiated partial session transfer is accepted by the MN. In case the MN does not want to accept the network initiated partial session transfer it responds with a 603-response (Decline). Figure 5.3 illustrates this situation.

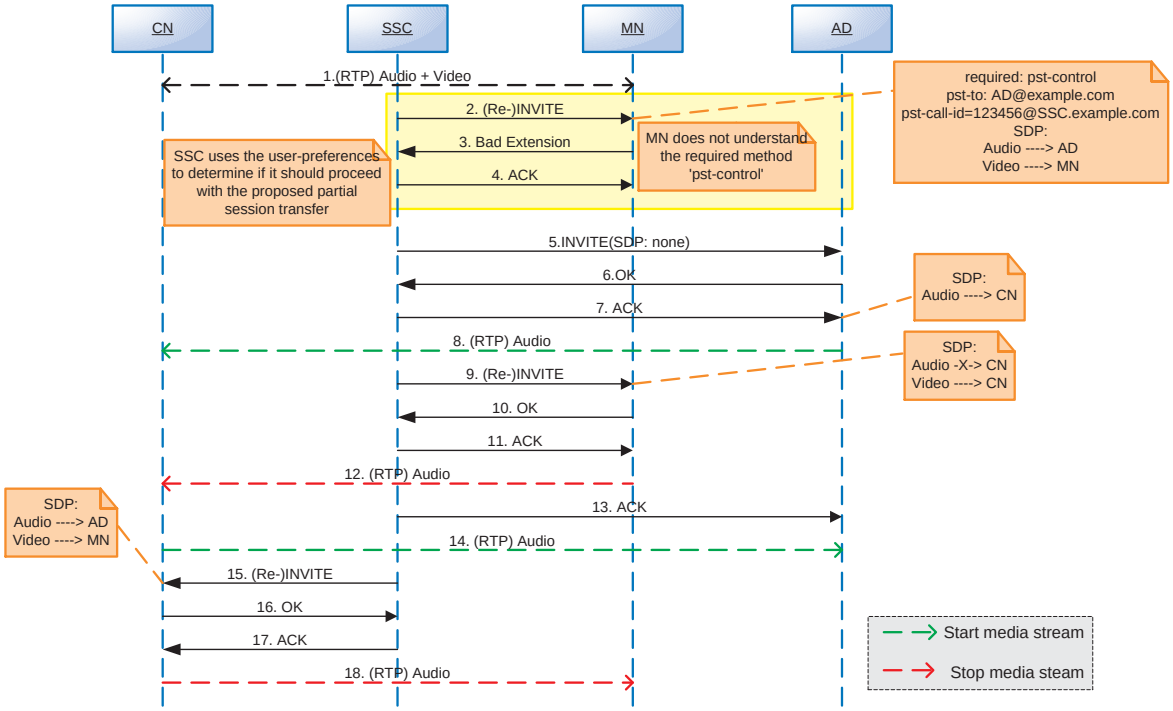


Figure 5.2: Message sequence diagram - User terminal does not support the extension

5.4 Information about the transfer

The requirements in the first section of this chapter describe that the proposal must contain certain information to let the user decide whether the network initiated partial session transfer must be executed:

- Which media-stream is being transferred
- The media-parameters of the media-stream that will be used at the target device after it has been transferred.
- The SIP-URI of the device the stream will be transferred to identify the device.

To make sure the extended INVITE message (Arrow 2) contains the information defined by the first two points as described above, it contains an SDP-body. This SDP-body however should not be interpreted as one in a normal negotiation (See section 3.1.2); the SDP-body has another meaning as will be described below.

As described in section 3.1.2 in SIP a media-stream is being defined by the combination of the SDPs of each endpoint of the session, each endpoint has its own ‘view’ on the media-session defined in the corresponding SDP.

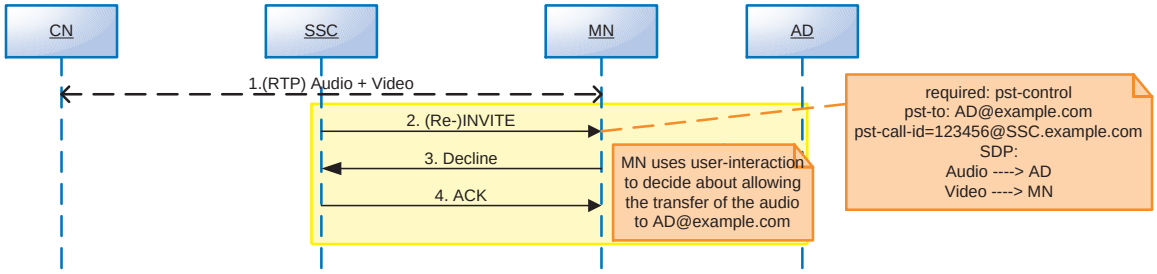


Figure 5.3: Message sequence diagram - User declines the proposed partial session transfer

The SDP-body contained in the 'pst-control'-INVITE message (Arrow 2) contains the SDP that the MN would normally sent to the CN in case the MN initiates a partial session transfer as defined in the 'Mobile-node control'-mode (See section 3.4.1). The MN compares, based on the place in the session description, the media descriptions in this SDP-body with the media description in SDP-body that was last sent to the CN to set up or change the session.

The media description that describes the media-stream that is being transferred always changes, because the IP-address or port number defined in the media description changes. This makes sure the MN is always able to recognize the stream that is being transferred based on the changed media description. Because the media description contains the media-parameters of the media-stream after it has been transferred to the target device, the MN also immediately knows exactly those media parameters.

SDP does not include a SIP-URI to identify the device the stream is being transferred to. Therefore the new header 'pst-to' is added to the extended INVITE message (Arrow 2) containing the SIP-URI of the device the stream will be transferred to. Figure 5.1 also shows this header in the comment at arrow 2.

5.5 Retrieve an already transferred media-stream

After the SSC or MN has executed a partial session transfer, it must be possible to start a terminal or network initiated partial session transfer to transfer the already transferred stream again to another device, where this could mean the media-stream is being returned back to the MN as described at the requirements in the first section of this chapter. A network initiated partial session transfer is always possible because the SSC is in the signalling path of each session the MN has with other nodes, and therefore can change those sessions accordingly.

However a terminal initiated partial session transfer after a network initiated partial session transfer is not that easy, because the MN does not have a session with the LD that is used. During the network initiated partial session transfer, the SSC did set up a session with the LD on behalf of the MN, but the MN was not involved in this session setup.

The MN does need the necessary information to identify the session the SSC set up with the

LD on behalf of the MN. Therefore the extended INVITE contains the new header 'pst-call-id' that contains the call-id of the session the SSC did setup with the LD on behalf of the MN. This new header is also illustrated in figure 5.1 in the comment at arrow 2. With this call-id to identify the session with the LD, the MN can close or change the session with the LD when it wants to. The SSC contains the logic to make sure SIP messages sent with this call-id are delivered at the LD. Figure 5.4 illustrates the situation where the MN retrieves the media-stream (Arrow 3-7) and closes the sub-session with the LD (Arrow 8-10).

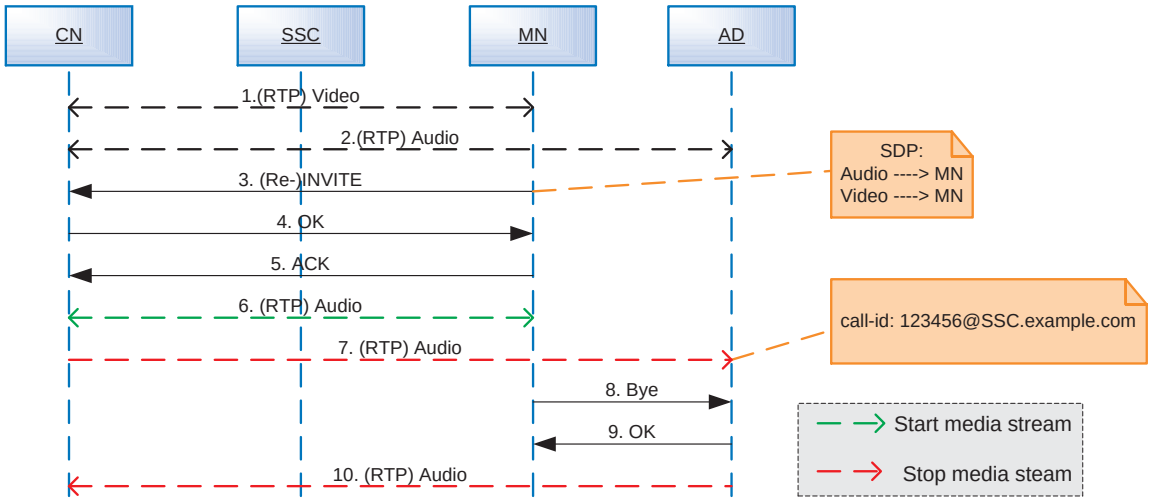


Figure 5.4: Message sequence diagram - Retrieved by terminal

5.6 User initiation

As described in the first section of this chapter the solution described here uses the 'Mobile-node control'-mode as basis. This basis already allows a MN to initiate a partial session transfer: the MN invites the local devices and than re-invites the CN. As described in section 5.2 the SSC acts as a B2BUA for all session the MN has with other nodes. The SSC can only initiate network initiated partial session transfers when it has an up-to-date and correct view of the current situation with respect to sub-sessions and how the media-streams are distributed over them. Because of this, the SSC does have to interpret the SIP messages that are being sent in these sessions correctly. This way, even when a terminal initiated partial session transfer occurred, the SSC still can initiate a partial session transfer afterwards.

When both terminal and network initiated partial session transfers occur in a single session a number of sub-session must be set up and closed at certain moments. The next section describes the responsibilities the SSC and MN have in relation to each other regarding those sub-sessions.

5.7 Managing sub-sessions

As described before a partial session transfer can be initiated by the network component or the mobile terminal itself. In each case sub-sessions are being set up between the MN and local device(s). This section describes which component is responsible for dealing with these sub-sessions in what situation. To explain the problem here follows an example: Assume there has been a network initiated partial session transfer. In case the MN does not support the pst-control extension and it does change the session with the CN, using a re-INVITE message to stop one of the streams, the SSC makes sure it handles the sub-session with the local device accordingly.

In this case the SSC does have the responsibility to look after all sub-sessions. However in other situations the MN might be responsible for this. How the responsibilities are divided amongst the SSC and MN is described below.

The easiest situations are those where there is no combination of network initiated and terminal initiated partial session transfers and the MN does not modify anything to the session with the CN after the transfer has been executed. If only network initiated partial session transfers occur, the SSC always has to deal with the sub-session. If only terminal initiated partial session transfers occur, it is the task of the terminal to handle these sub-sessions.

As the example above shows the question if the MN supports the 'pst-control' extension also has influence on the way the responsibilities regarding the sub-sessions are being handled. The SSC knows if the MN supports the pst-control extension after it sent the first pst-control re-INVITE message. Below the rules are shown in situations where both network initiated and terminal initiated partial session transfers can occur:

- MN supports pst-control
  - The node that made the last change to an individual sub-session is responsible for this sub-session. This means after a network initiated partial session transfer the SSC is responsible for the involved sub-session. After the MN does change something to the sub-session the MN is responsible. The SSC could become responsible again if it executes a partial session transfer while changing something in that specific sub-session.
  - This also means that if the MN is responsible for an individual sub-session and it does change the session with the CN, it should also make sure the sub-session with the local device is still consistent with the session between the MN and CN.
  - In case the MN is able to initiate a partial session transfer, it will never become responsible for a sub-session, because it never changes or starts a sub-session.
- MN does not support pst-control
  - SSC is always responsible for the sub-sessions
  - If the MN does change something in the session with the CN, the SSC has to make sure the sub-session(s) are still consistent with this session.

As described above the SSC does know if the MN supports the extension when it receives the response of the MN to the pst-control re-INVITE message. However if the user is at first connected to a device that supports the extension and than starts using another device (User mobility) that does not support the extension, this assumption does not hold. The solution described in this chapter does not consider this situation, because in this thesis the focus is purely on partial session mobility (As in section 1.2. To find a solution for this specific problem, which is out of scope for this work, further research must be done)‘.

As describes in section 3.4.1 the ‘Mobile-node control’-mode does have the problem related to robustness at the moment the MN suddenly disconnects while one of the media-streams has already been transferred to another device. In this situation the involved media-stream continues, while the user is not able to use its terminal to stop the media-stream. In this case the user should use the interface of the involved LD to stop the stream. With solution described here the SSC is always involved in all sub-sessions related to the MN. Therefore if the SSC is notified of the communication failure with the MN, it could easily stop all related sub-sessions if they did not already.

As described in the previous sections all the functionality described in section 5.1 is added to the ‘Mobile-node control’-mode in order to fullfill the requirements described in section 4.1. The next chapter describes how the MN and the SSC, which is introduced here, are designed to make sure they can support the method described in this chapter.

Chapter 6

Design of the prototype

This chapter describes the design of the prototype. When using the solution described in the previous chapter the SSC and possibly the MN support the extension to SIP. At least the SSC should know exactly how the different media-streams are distributed over the different available devices. To maintain this information a data model is needed, which is described in the first section. The following sections present the design of the SSC and the SIP-UA used as MN, CN and LDs.

6.1 Data model

The purpose of the data model is to maintain an up-to-date view of the relations between nodes, both on the control plane and data plane. This means this model can contain the following types relations:

- The control sessions (SIP sessions) between nodes
- The media sessions that are defined in the control sessions.
- The relations between the control sessions and the media sessions. With this type of relations it is possible to have two devices participating in a control session, where the control session defines a media session between two different devices.

The data model described here can contain this up-to-date view of the sessions and sub-sessions in a situation where currently no partial session transfer is going on. This means it can not contain ongoing transitions that occur during the execution of a partial session transfer. However the nodes using this data model do use the individual entities to execute the transitions; But only when the transitions are completed the data model is consistent again, fulfilling the constraints and relations as described here.

The data model consists of a number of entities that are related to each other. Figure 6.1 contains an UML class diagram showing the entities of the data model, modelled as classes, and how these are related to each other. The data model does not explicit model the control-session or media-session in the class diagram. Those relations are implicitly modelled by means of the distributed endpoints. The diagram does not show all constraints that exists on the different relationships, those are described in the text. Below each of these entities and the relations between them are described in more detail:

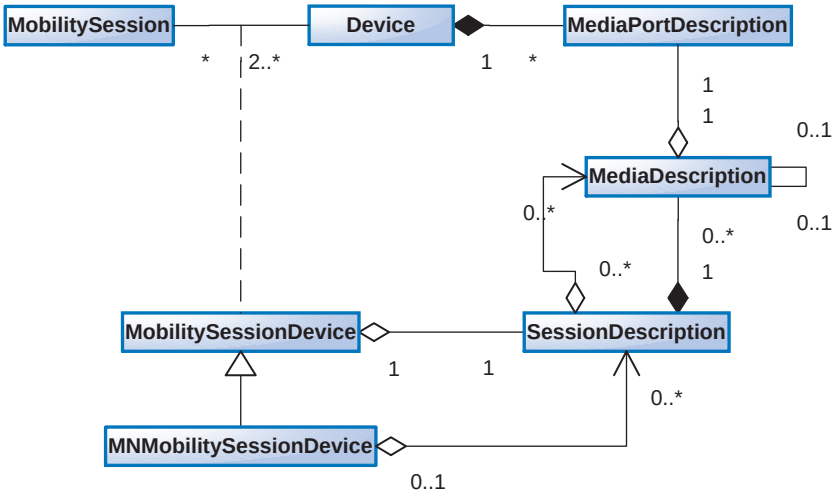


Figure 6.1: Class diagram - The data model of the system

- *MobilitySession*. This class containing a combination of control sessions (See the description of a mobility session in section 2.3) define how the different media-streams are related to the different devices. A number of devices can exist in a MobilitySession, each defining the relations with the other devices. In a MobilitySession precisely one MN and CN can exist, while there can exist multiple LDs.

- *Device*. This entity represents a certain registered SIP-UA, thereby not necessarily representing a real world device. The MN, CN and LDs as used throughout this thesis are all represented as devices in this data model. A Device can have multiple MediaPortDescriptions, that represent the endpoints of media-sessions

- *MobilitySessionDevice*. This component represents the relation a Device has with a MobilitySession. If a device is present in multiple MobilitySessions, then for each of these MobilitySessions a MobilitySessionDevice exists.

- *MNMobilitySessionDevice*. This is a sub-class of MobilitySessionDevice that has one difference: it can contain additional SessionDescriptions, one for each sub-session.

- *MediaPortDescription*. This component represents a media-session endpoint, which is the combination of an IP-address and port number. A MediaPortDescription always belongs to a MediaDescription that describes the media-capabilities of the media endpoint. Besides the MediaDescription a MediaPortDescription also belongs to one specific device. One device can contain multiple MediaPortDescriptions.

- *SessionDescription*. This entity represents a specific MobilitySessionDevice in a control-session. A control-session defines a number of media-sessions; these media-sessions are described from the perspective of the specific MobilitySessionDevice by means of MediaDescriptions.

A session description is related to one MobilitySessionDevice, meaning it is specifically created to represent a MobilitySessionDevice in a session. A MobilitySessionDevice generally has only one SessionDescription, however the MobilitySessionDevice that is the MN (MNMobilitySessionDevice) can have a number of additional SessionDescriptions for the sub-sessions. The class diagram shows this as directed aggregation relationship between MNMobilitySessionDevice and SessionDescription. The constraint in this relation is that a SessionDescription always only has a relation with one MobilitySessionDevice (Including a possible MN).

A SessionDescription can have a relationship with a MediaDescriptions in two ways:

- The MediaDescription has a MediaPortDescription that is related to the same device this SessionDescription is related to. This is normally the case when a session is being set up between two devices, where the media-sessions are also being set up between those devices. This relationship is shown by the composite relationship in the class diagram between MediaDescription and SessionDescription.
- The MediaDescription has a MediaPortDescription that is related to another device than this SessionDescription is related to. This is the case when for example device A sets up a session with device B and device A wants one of the media-sessions to exist between device B and C. In that case a sub-session must exist between A and C to make sure C also agrees with this session. In the class diagram this relationship is shown as a directed aggregation relationship between MediaDescription and SessionDescription.

  This relation can only exist if the SessionDescription is related to an MNMobilitySessionDevice, because within a MobilitySession only the MN uses MediaDescriptions in its SessionMobility that represent a media-stream endpoint that is located at another device. Besides this, this relation can also only exist if the SessionDescription linked with this MediaDescription via this relation exists in the same MobilitySession as the SessionDescription that is linked with the MediaDescription via the composite relation described above.

- *MediaDescription.* This entity represents a media-stream from one side of a control-session. It has a MediaPortDescription that describes the endpoint for the media session. A MediaDescription always belongs to one SessionDescription, that is a SessionDescription of the Device that has the specific endpoint of the media-stream. This relation with SessionDescription is shown by the composite relation in the class diagram. Besides this relation, a MediaDescription can also occur in other SessionDescriptions, this is shown by the aggregation relation in the class diagram.

  A MediaDescription can have a connection with another MediaDescription, the so-called 'partner'. If two MediaDescriptions have each other as partner, those MediaDescriptions form a pair; this means the media as described in both MediaDescriptions can flow between the two MediaPortDescriptions they are related to. A number of constrains exist for this relationship:

  - A MediaDescription cannot have itself as its partner.

- The partner of the MediaDescription may not be related to another MobilitySession.
- If a MediaDescription has a partner, that partner must have this MediaDescription as partner. It is also possible that a MediaDescription has no partner; This is possible in a situation where during the negotiation, the device on the other side of the session does not responds with an answer to the specific MediaDescription.

Both the SSC and 'pst-control' aware MN can use this data model to maintain the state of the MediaStreams.

6.2 Sub session controller (SSC)

As described in the previous chapter the SSC is the component that can initiate partial session transfers from the network side. To do this it needs an up-to-date view of the situation regarding to sub-sessions and how the media-streams are distributed. That is why the SSC acts as a B2BUA as described in section 5.2. This way the SSC handles all signalling for sessions the MN has with other nodes, and uses the data model as described in the previous section to keep an up-to-date view of the situation.

The first section describes the different components and interfaces used in the design, while the second section describes the behaviour of the SSC in different situations.

6.2.1 Interfaces and components

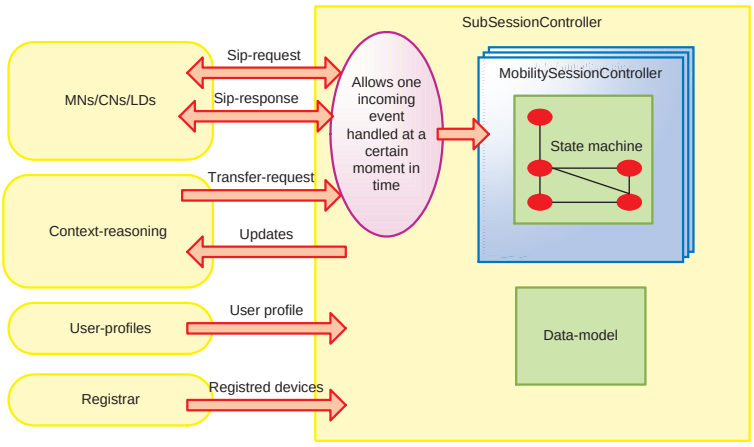


Figure 6.2: Design of the SSC

Figure 6.2 shows the design of the SSC. Below each of the external components and the relations with the SSC are described:

- *MN/CN/LDs* As described before the SSC does act as a B2BUA; this means the SSC communicates with the CN, MN and involved local devices using SIP. Hereby all control

plane messages are routed via the SSC, this means it receives and sends SIP requests and response to and from the MN, CN and LDs.

- *Context-reasoning* As describe in section 4.1 the context-reasoning process is done in the context-reasoning node. The SSC does need an interface to communicate with this node to inform this node about the current situation and to let the context-reasoning node order the SSC to initiate a partial session transfer.

- *User-profiles* After the MN does responded with a 420-response (Bad extension) the SSC has to make a choice between continue with the transfer or stop the transfer based on user preferences. Therefore the SSC needs an interface with the component that stores these user preferences in a user profile.

- *Registrar* The SSC does need the contact-address of SIP-nodes that are being involved in sessions to know how to route a message to a SIP-node. Therefore the SSC needs updates from the registrar about these nodes and address the nodes are reachable on.

To keep the prototype simple we decided that the SSC does only handle one incoming event at the same time. This means for example that the SSC does not yet handle an incoming SIP request from a SIP-UA before a transfer-request is fully handled, or the other way around. If the prototype would not contain this restriction it would need more complex methods to ensure the integrity of the data model, because different simultaneous processes can make updates to the model at the same time.

Below is a description of the internal components:

- Data-model This data model as defined in section 6.1 is used to keep an up-to-date view of the sessions and media-streams defined between the different nodes. As described this data model does not contain an up-to-date view during the execution of a partial session transfer.

- MobilitySessionController Each instance of a MobilitySessionController controls a specific MobilitySession and has one state-machine that processes the incoming events and making sure the result is reflected by the data model.

- State machine The state machine contains a number of states that represent the different stages in the communication with the external components. A number of events can occur, the state machine defines how these events are handled for each individual state. The next section describes the exact behaviour of this state machine.

  The state machine also defines how a network initiated partial session transfer is being executed. During these transition it does not use the data model, instead it uses the data model only in the beginning to view the current situation, while during the transfer it does use the classes of the data model to invite the different devices. Only after the transfer is completed it makes sure the data model is consistent again as described by the design of the data model (Section 6.1).

6.2.2 Behaviour

This section describes the behaviour of the SSC. To do this in a structured way this section contains a number of state diagrams. The state diagrams do not contain exactly all SIP messages that are being exchanged; BYE, RINGING and TRYING messages are not shown. Also the registration of nodes at the registrar is not shown in the state diagrams.

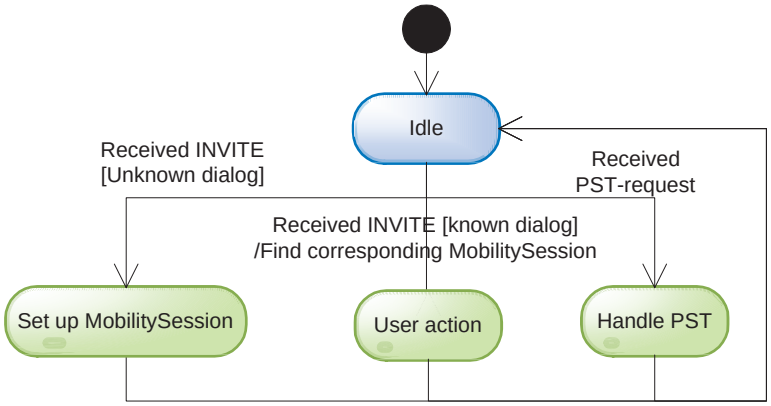


Figure 6.3: State diagram - SSC: Top-level

The first state diagram (Figure 6.3) describes the high level behaviour of the SSC. It shows how the behaviour is split-up in three parts, using sub-states:

- *Setup mobility session*: A new session is being set up by a user. This is initiated by an INVITE message send by a UA (Below this is called 'the source', just as the target of the INVITE is called 'the target') that contains an unknown call-id. This results in a new defined mobility session in the SSC. The INVITE message received from the source is being forwarded to the target, after which the response of the target is being forwarded to the source. Also the ACK message is being forwarded this way. This procedure is illustrated by the state diagram in figure 6.4.

- *PST handling*: This sub-state defines how the SSC handles a network initiated partial session transfer; figure 6.5 shows the state diagram (This figure does not contain the handling of ACK messages, in reality however they are being send). The SSC is ordered to initiate a partial session transfer via the interface the SSC has with the context-reasoning node (See section 6.2.1).

  The SSC first proposes the partial session transfer to the MN by sending the extended INVITE message to the MN. If the MN does not answer with 200 (OK), then the partial session transfer stops. In this design the SSC does use user-preferences to decide if it continues with the partial session transfer.

  In the next step the target device of the partial session transfer is being invited, if it does not responds with a 2xx-response then the partial session transfer is stopped. After that the CN is being invited, if it does not respond with a 2xx-response the session

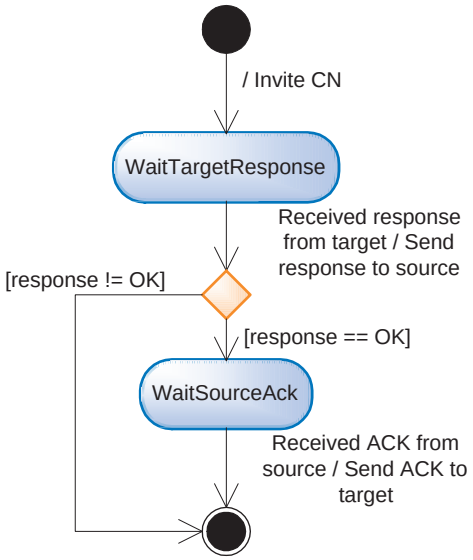


Figure 6.4: State diagram - SSC: handle set up of mobility session

with the target is being restored in the previous situation (Shown in the figure by the action 'Restore target'). Than as the last step the SSC invites the device the transferred media-stream was originally connected to. In case this device does not respond with a 2xx-response, the session with the CN and the target must be restored to the original situation, this is shown in the figure as the action 'Restore CN; Restore target'

- *User action*: This sub-state defines how terminal initiated changes in a session are handled. If the MN, CN or a LD sends a re-INVITE, the SSC has to manage this message as a B2BUA, while also making sure all sub-states it is responsible for are handled correctly.

  How the SSC handles these requests is illustrated by figure 6.6 containing the state diagram. As this state diagram shows it is subdivided in four composite states, each handling requests that have specific nodes as sender and receiver. Below each of them is described briefly:

  - *Handle CN to MN*. A request sent by the CN with the MN as recipient is being send to the MN. After the MN responses with a 2xx-response, the SSC has to make sure involved sub-sessions the SSC is responsible for are correctly handled. The exact behaviour in this case is not prescribed in this design. . After this a response is being send to the CN. This behaviour is illustrated by the state diagram in figure 6.7.
  - *Handle MN to LD*. In case MN sends a message to an LD on a sub-session, it does support partial session mobility. In this case the request is simply forwarded, after which the same is done with the response. Figure 6.8 illustrates this behaviour.
  - *Handle MN to CN*. The request sent by MN with CN as recipient is forwarded to CN. Also the response of the CN is being forwarded to MN. In case of a positive

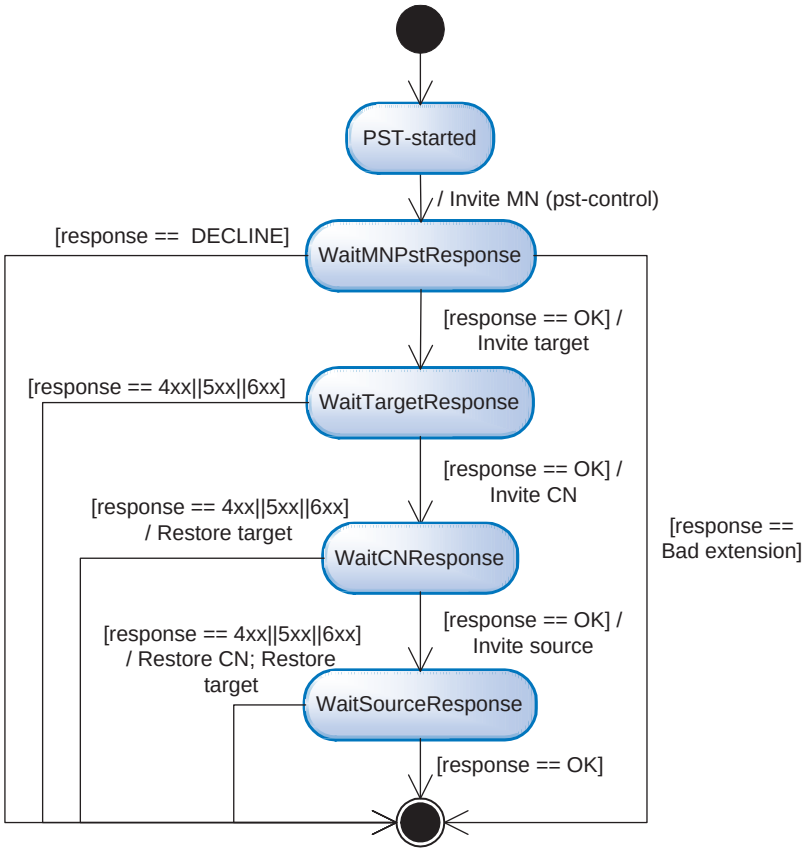


Figure 6.5: State diagram - SSC: Handling of network initiated partial session transfer

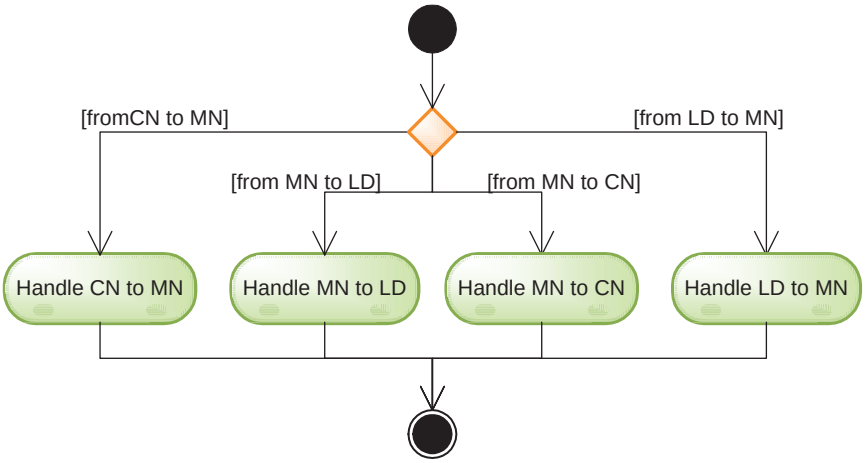


Figure 6.6: State diagram - SSC: Handling of terminal initiated re-invites

response the SSC has to handle the sub-session it is responsible for. Also in this

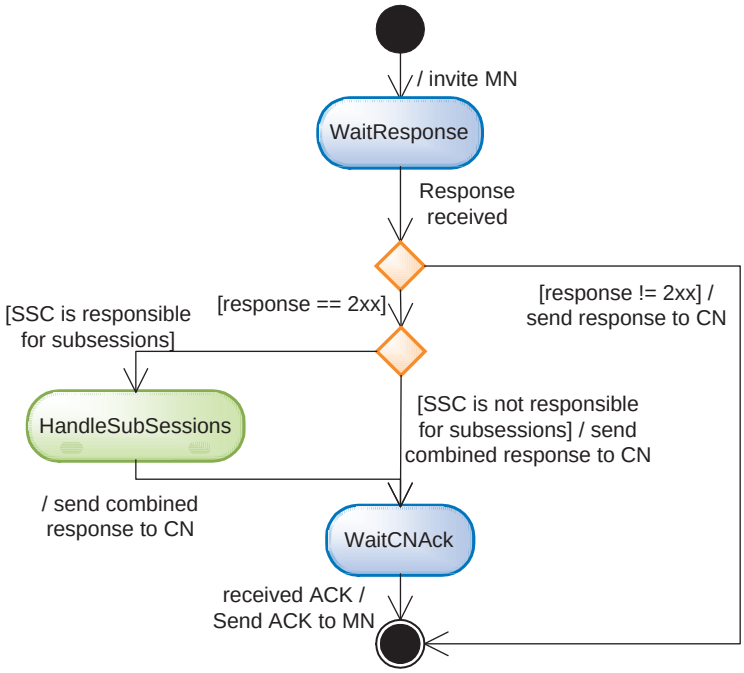


Figure 6.7: State diagram - SSC: Handling CN to MN

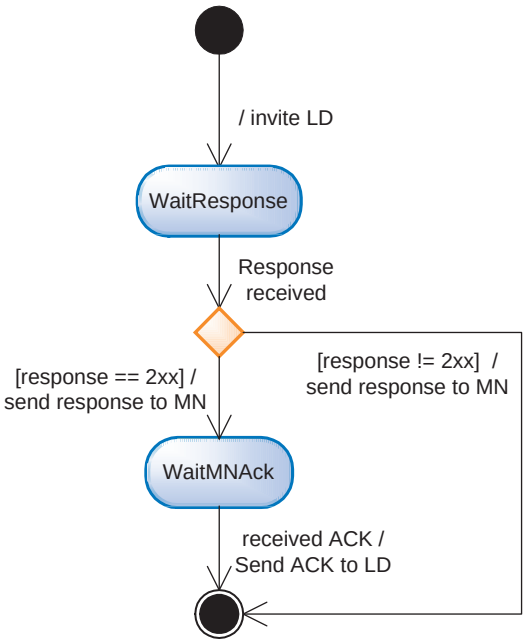


Figure 6.8: State diagram - SSC: Handling MN to LD

situation this design does not specify the exact behaviour on how to handle the sub-sessions, however this could for example mean that all sub-session are closed. Figure 6.9 shows the related state diagram.

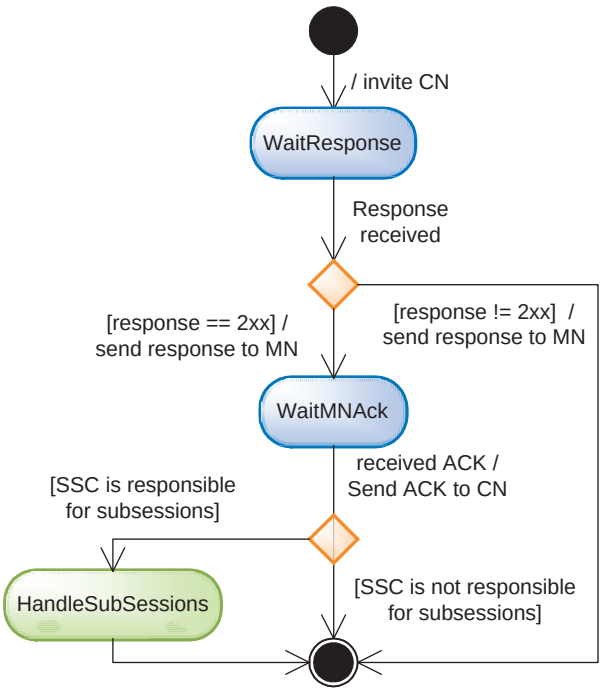


Figure 6.9: State diagram - SSC: Handling MN to CN

– *Handle LD to MN.* In case the LD sends an INVITE message to the MN in this design the SSC directly responds with a 603-response (Decline) making sure nothing changes in the session with the LD. In future versions it might be possible to accept a re-INVITE message from the LD.

6.3 Mobile node

The solution described in chapter 5 does support two different types of SIP-UAs as MN, namely basic SIP-UAs that do not support the pst-control extension and SIP-UAs that do support the pst-control extension. To describe the design of the latter, which is based on the basic SIP-UA, this section first introduces the design of the basic SIP-UA. After that the extended design is presented needed to support the pst-control extension.

Figure 6.10 shows a reference design of a basic SIP-UA. At the bottom this design consists of the SIP-stack and the RTP-stack, the SIP stack handles the signalling via SIP and the RTP stack handles the media-streams via RTP. Based on the signalling the module handling the media is being configured to sent and/or receive data via the RTP-stack. As the source of media the media handling module can use media-sources like a web cam or a microphone. The GUI defines the user interface that gives the user user-friendly controls to manage sessions. The GUI can also be used to playback incoming media. Although in this design the focus is on SIP-UAs with a GUI, SIP-UAs without GUI that automatically handle invitations can also exist. Besides the GUI that can playback media-streams it is also possible that other

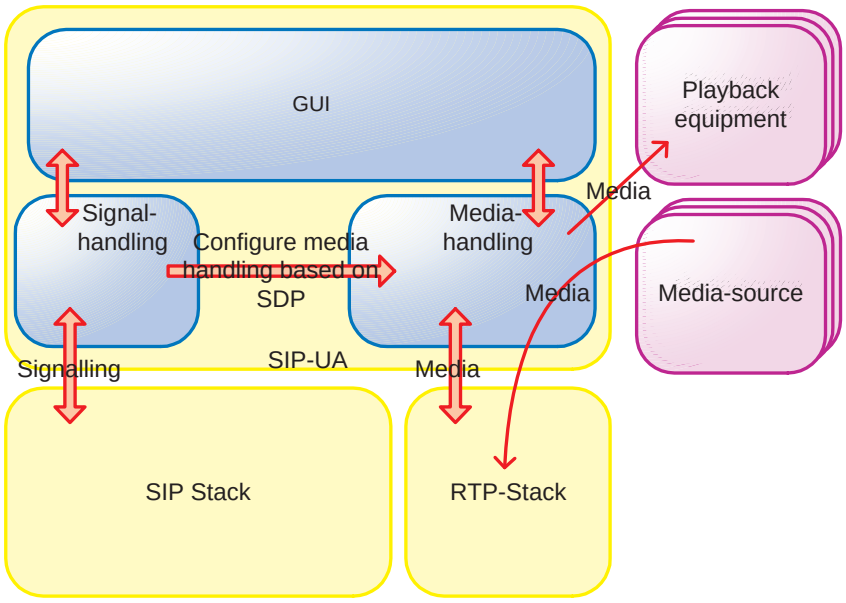


Figure 6.10: Design of a basic SIP-UA

playback equipment is used for this purpose.

To support the pst-control extension the SIP-UA should contain the following extra functionality in comparison to the basic SIP-UA:

- It should understand the (re)-INVITE message with the 'pst-control' method in the required header in order to understand proposals for partial session transfers.
- It should have a mechanism to interact with the user to display the proposal and offering a choice to the user. Or the user should be able to configure the UA to make the choice automatically.
- For the view of the user on a call with the CN it would be helpful to show information about which device is being used for the individual media-streams. This can only be shown if the SIP-UA is able to store the state of the session with the CN and the related sub-sessions.
- The user should also be able to manually initiate a transfer of these media-streams showed in the GUI. Instead of using the GUI the prototype could also implement an interface that can be used by another component to initiate a partial session transfer from the terminal. However in the design of the prototype SIP-UA only the GUI is considered as interface.

To make sure the design of the SIP-UA supports these new functionalities the following additions and changes are made to the design (Figure 6.11 illustrates these changes):

- The Signal handling should be extended to understand the pst-control method.

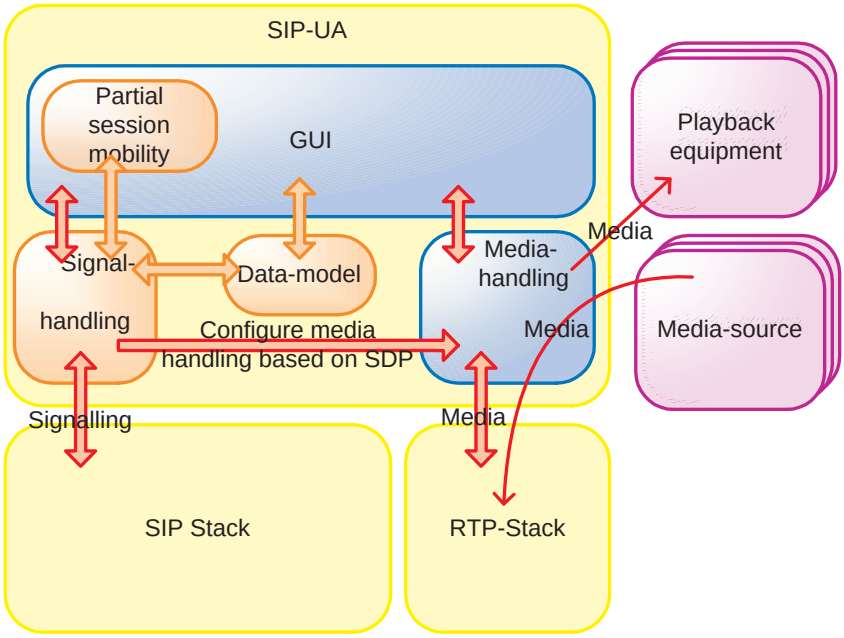


Figure 6.11: Design of the extended SIP-UA

- A data model that relates the different sessions and the media-streams in those sessions to each other should be added.
- The GUI could contain an extra section that shows these relations and gives the possibility of initiating a partial session transfer. As an alternative an extra interface (non UI) could be defined to let other components start a partial session transfer from the terminal. In our prototype we only consider the GUI variant.
- A GUI component should be added that can and will be used to propose a network initiated partial session transfer to the user.

Chapter 7

Implementation

To validate the method developed for partial session mobility we implemented a prototype. This chapter describes the implementation of this prototype. The prototype focuses on the basic functionality; this means not all the aspects of the design are implemented in this prototype. The first section describes the implementation of the sub-session controller while the second section does the same for the Mobile node. The last section describes the implementation of the data model.

7.1 Sub-session controller

The Sub-session controller is basically a B2BUA that is located in the signalling path of the involved devices. The specific behaviour of the B2BUA implements the functionality needed for partial session transfers.

There are multiple ways this B2BUA can be implemented, namely using an application server framework (E.g. the RADVISION SIP Server Platform [15]) as lower level service or direct using the SIP-stack for this purpose.

Because our prototype is build for proof-of-concept and we did not want to build in on top of a specific software framework, to keep it simple the prototype is being build directly on top of the SIP stack. As implementation for this SIP stack we use the JAIN SIP reference implementations (Formally known as the NIST SIP stack) [9]. This implementation has the following advantages:

- It is a Java implementation. This makes easy and fast implementing of the prototype possible.
- Implements the standardized JAIN-SIP API specified in JSR-32 [7]

To keep our validation set up simple and our prototype implements some very basic registrar functionality (See section 3.1.1). This way the SSC can internally search for the contact-information when it wants to send a SIP message to one of the registered terminals. With this registrar functionality the validation set up is being simplified because all terminals only have to use the SSC as proxy-server so they can communicate with each other.

Our prototype contains a number of packages, each containing classes. Below the structure of these packages is explained:

Package	Description
com.lucent.psm	The package all the new components is added
com.lucent.psm.controller	The package specific for the SSC
com.lucent.psm.controller.core	Contains some generic classes
com.lucent.psm.controller.events	The events that can occur on the states
com.lucent.psm.controller.states	All states in the state-machine of the SSC
com.lucent.psm.controller.states.setupms	The states being used for the process of setting up a mobility session
com.lucent.psm.controller.states.pst	The states being used in case of an network initiated partial session transfer
com.lucent.psm.controller.gui	Contains the GUI related classes.
com.lucent.psm.javax.sip.header	Extension to the stack to support the 'pst-to'-header and 'pst-call-id'-header
com.lucent.psm.model	The data model being used
com.lucent.psm.registrar	Registrar functionality

Table 7.1: Package structure

7.1.1 Jain SIP stack

This section describes how the sub-session controller uses the underlying NIST SIP stack using its Jain SIP API. Below the different stages and situations in communicating using SIP. Each of those stages is explained using code-snippets.

Initialization of the SIP stack

At the start-up of the sub-session controller one of the first things it does is initializing the SIP stack in the SipManager. Before creating the actual instance of the SIP stack two mandatory properties must be set, namely the IP-address the stack should be listen on and a user-friendly name to identify the stack. Because the sub-session controller must support both user agent and proxy functionality the stack must not automatically create dialogs when a transaction is created. To accomplish this we use the automatic dialog support property. Here we show the code necessary to set the options:

```
System.setProperty("javax.sip.IP_ADDRESS", "");
System.setProperty("javax.sip.STACK_NAME", "");
System.setProperty("javax.sip.AUTOMATIC_DIALOG_SUPPORT", "off");
```

After this step the SipManager instantiates a SIP stack, binds it to the correct IP-address and port, and configures the stack for call-backs:

```
// Initialize Sip factory
javax.sip.SipFactory sipFactory = SipFactory.getInstance();
sipFactory.setPathName("gov.nist");

// Create SIP-stack
javax.sip.SipStack sipStack =
    sipFactory.createSipStack(System.getProperties());

// Create and bound SIP-listening-point on port 5000 using UDP as transport
javax.sip.ListeningPoint listeningPoint = sipStack.createListeningPoint(
    System.getProperty("javax.sip.IP_ADDRESS"),
    5000, "udp");

// Create SipProvider
javax.sip.SipProvider sipProvider = sipStack.createSipProvider(listeningPoint);

//Register self as SipListener
sipProvider.addSipListener(this);
```

Incoming SIP messages

After the SIP stack receives a SIP request or a SIP response, it uses the SipListener interface to call one of the following methods on the listener:

```
public void processRequest(RequestEvent requestEvent);
public void processResponse(ResponseEvent responseEvent);
```

In case of a SIP request the SipManager first checks if the request is a REGISTER-message, it than delegates the processing of this request to the Registrar (See section 7.1.5). If it is not a REGISTER request it starts handling the request state fully.

If the Request does not belong to an existing dialog (The server transaction == null) then the SipManager creates a new transaction using the following construct:

```
SipProvider.getNewServerTransaction(Request);
```

Using the callId in the SIP message the SipManager determines the corresponding MobilitySession in which the request should be handled further. In case no corresponding MobilitySession exists a new one is created.

In case of a SIP response the SipManager also uses the callId to determine the MobilitySession that will handle the response further on.

Sending a SIP request

There are multiple situations where a SIP request must be send. The first situation is where the SIP request is not sent in an existing dialog. Below we show the interaction with the stack necessary to send an INVITE message not coupled to an existing dialog:

```
//Create the actual request.
Request req = SipFactory.getInstance().createMessageFactory().createRequest(
    URI,
    "INVITE",
    CallIdHeader,
    CSeqHeader,
    FromHeader,
    ToHeader,
    LinkedList, //LinkedList with ViaHeaders
    MaxForwardsHeader,
    ContentTypeHeader,
    byte[] //Content
);
//Add the Contact-header
req.addHeader(ContactHeader);

// Add a record-route header to make sure every SIP message send in this dialog
// is always routed via the SSC
req.addHeader(RecordRouteHeader);

//Create a Client transaction using the request
ClientTransaction ct = SipProvider.getNewClientTransaction(req);

//Send the request
ct.sendRequest();

//Create the dialog using the ClientTransaction
Dialog dialog = SipProvider.getNewDialog(ct);
```

It is also possible that an INVITE request is sent inside an existing dialog, this is called a re-INVITE. In this case the request can be created using the already existing dialog:

```
// Create the INVITE request using the existing dialog
Request req = Dialog.createRequest("INVITE");

// Fill the request with the correct content
req.setContent(Object, ContentTypeHeader);

// Add the record-route header necessary to route all signalling via the SSC
req.addHeader(RecordRouteHeader)

// Create at ClientTransaction using the SipProvider
ClientTransaction ct = SipProvider.getNewClientTransaction(req);

// Send the request
Dialog.sendRequest(ct);
```

Other requests that must be sent within an existing dialog are created and sent the same way. The only request that is treated differently is the ACK request. The acknowledgement of a certain call is not done using the 'createRequest'-method, instead the following method is being used:

```
//Create ack request using the CSeq number of the INVITE this ACK acknowledges
Request req = dialog.createAck(long);
```

7.1.2 pst-control support

As described in chapter 5 and 6 the developed method for partial session transfer uses a new extension to sip. The sub-session controller does have to support this extension; this section shows what implementation specific consequences this extension has.

The first extension to basic SIP is the addition of the require-method 'pst-control' to the require-header. Because this involves no addition of a header there are no SIP extensions necessary to support this require-method. To support both the pst-to header and the pst-call-id header however new SIP headers are introduced, this means the SIP stack must support those. This is why we extended the NIST SIP stack with those two headers.

Our prototype introduces the two classes 'PstCallId' and 'PstTo' that correspond to the two added headers. Both classes are located in the package 'com.lucent.psm.javax.sip.header' and extend the class 'gov.nist.javax.sip.header.SIPHeader' which is located in the NIST Sip stack.

The JAIN SIP API defines the HeaderFactory interface to build SIP stack implementation specific headers. To add support for a new header in the SIP stack properly the JAIN SIP API should be extended, this means the HeaderFactory should be able to create the new header. However because in this case focus is on creating a prototype to validate the developed prototype this is necessary. The code-snippet below shows how the prototype uses the new Headers, without using the factory pattern (See [30]).

```
// Add the pst_to header
Request.addHeader(new PstTo(URI));

// Add the pst_call_id header
Request.addHeader(new PstCallId(String));
```

7.1.3 State machine

As described in the previous chapter the sub-session controller contains a state machine to handle incoming SIP messages and partial session transfer requests, this state machine is implemented using the state pattern (See [30]). The generic part of this state-machine consists of a few abstract classes:

- *Event*: This class is the super-class of all event-classes defined in the state machine. The event sub-classes defined in the implementation are: RequestEvent, ResponseEvent, SipRequestEvent and TransferRequestEvent. The SipRequestEvent and TransferRequestEvent are both sub-classes of the RequestEvent. Each of these events corresponds to one of the events being defined in the design of this state machine (Section 6.2.2). Figure 7.1 illustrates the structure of the event package.

- *State*: This class is the super-class of all states defined in the state machine. It contains a method 'handleEvent' that is being implemented by the sub-classes. The sub-classes are separated between two sub-packages, namely 'setupms' and 'pst', according to the design of the state machine. Figure 7.2 illustrates the structure of the state package.

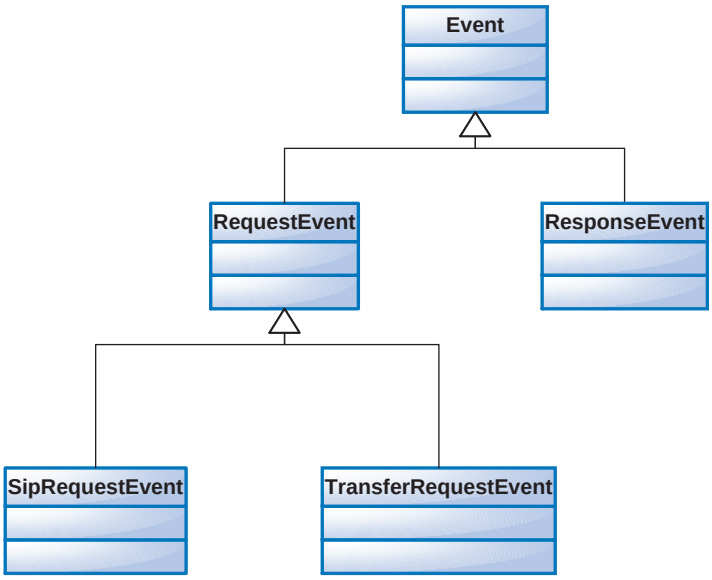


Figure 7.1: Class diagram - Event package

The implementation of the prototype differs from the design for the relations between the states during a network initiated partial session transfer. As described next in section 7.2 the software being used for the SIP-UA does not handle multiple streams as described in the design; this results in disrupted streams. Therefore the SSC implements another sequence of inviting the different devices. The original design used the following sequence (See section 6.2.2): MN(with pst-control extension), target of the stream, CN, source of the stream. The sequence as implemented by the SSC is as follows: MN(with pst-control extension), source of the stream, target of the stream, CN.

Another difference in the implementation of the prototype according to the design is the absence of the capability to recognize and process user initiated partial session transfers because the implementation of the network initiated functionality in the prototype had a higher priority than the terminal initiated functionality. The task of the SSC in a terminal initiated partial session transfer is acting as a B2BUA, understanding what is going on and process the result in the data model. The terminal itself does use the 'mobile-node control'-mode, which is already a proven method.

For each of the two composite-states(Set-up of MobilitySession and PST) implemented in the prototype, the sub-states have to store some information about the ongoing actions. Therefore two classes are defined to store this information, namely SetupMsInformation for storing information about the setup of a MobilitySession, and TransferInformation for storing information about the ongoing network initiated partial session transfer.

The current implementation of the prototype does handle BYE messages it receives. This means the if a SIP-UA closes a connection the prototype of SSC ignores this. Just as re-INVITE message coming from on of the involved nodes, a BYE message can have consequences to related sessions. The handling of the related sessions and the processing of this in the data

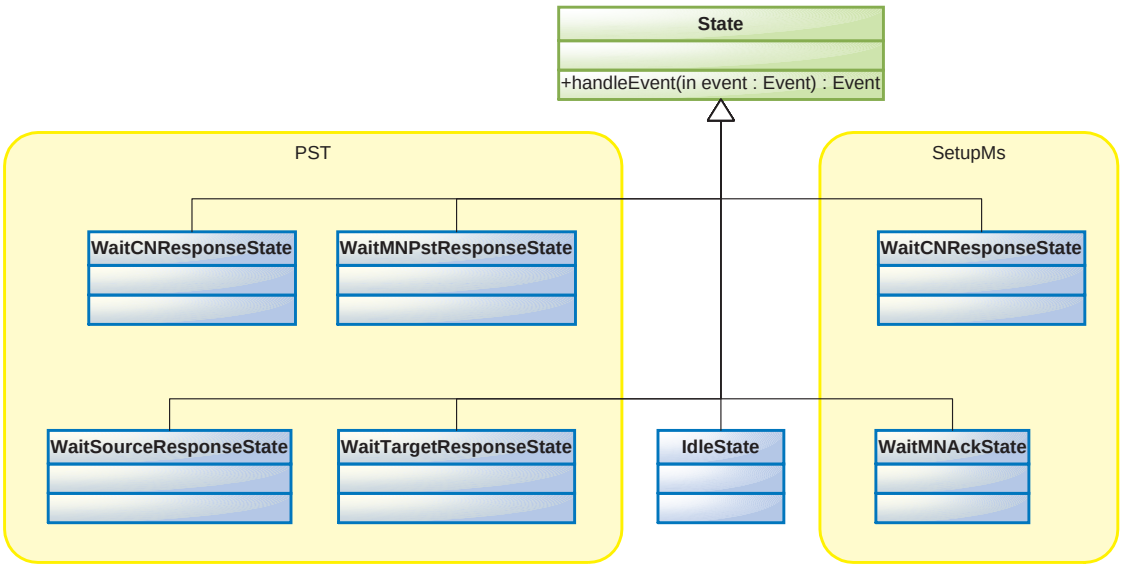


Figure 7.2: Class diagram - State package

model is considered as future work.

7.1.4 User interface

The implementation of the prototype contains a user interface to show the current situation of the sessions going on and to initiate a partial session transfer. This section first describes the process to configure a partial session transfer before actually initiating the partial session transfer.

The sub-session controller can control multiple mobility sessions at the same time. The first step in the configuration process is to choose a mobility session where we want to change something in the distribution of the media. After this, the user should be able to choose a media-stream between the CN and a LD or the MN, which it wants to transfer. The next step is to choose the device the media-stream should be transferred to. Now the SSC can start the actual partial session transfer.

Figure 7.3 shows the GUI of the SSC. At the top the GUI contains a select box that can be used to select an ongoing mobility session. With the 'Refresh'-button this list can be refreshed. Underneath is a detail-screen that graphically shows the media-streams between the different devices in the selected mobility session. An audio stream blue, while a video stream is painted in red. When clicking on one of the available streams, the stream is selected. The same counts for the devices. At the bottom the selected stream and/or Target device are being displayed. At the moment those have been selected, the transfer button becomes active and can be used to start a network initiated transfer of the selected stream to the selected device.

After the user chooses a device that should be used as target for the choosen media-stream

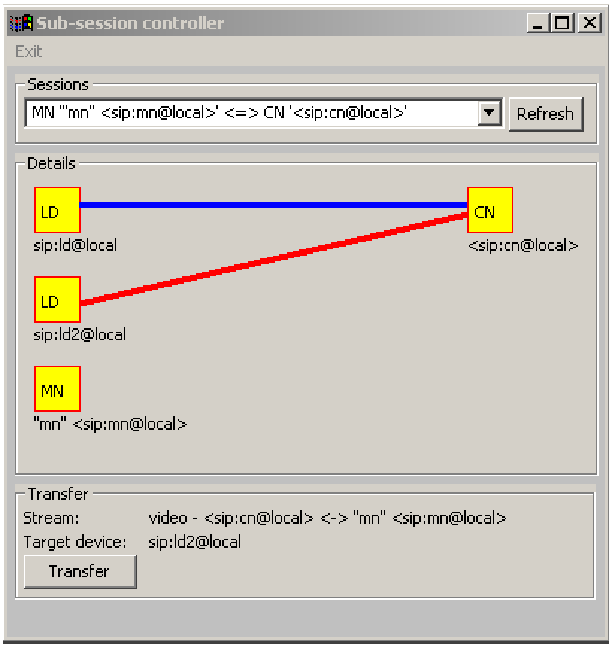


Figure 7.3: The GUI of the SSC

the prototype has to look at the capabilities of that device to find a media description that can be used as media-stream endpoint. For this the prototype uses an OPTIONS-request for which the device responds with an SDP-body describing its capabilities.

7.1.5 Registrar

As stated before the prototype of the SSC does contain a simple registrar to keep the validation setup simple and to let the SSC search internally for registered terminals without needing an external interface. The source-code containing the registrar functionality originated from the JAIN-SIP Service Platform [8].

At the start-up of the SSC it does create an instantiate of the Registrar. When the SSC receives a REGISTER-message it uses the registrar to process the message. The registrar internally stores the information about registered users. When a Device is created in the data model the ModelManager uses the registrar to find the contact-URI of the device. Also the GUI of the SSC uses the registrar to find out which devices are registered, so these can be used as target of a partial session transfer.

7.2 Mobile node

As described in section 6.3 a number of changes are needed to the basic SIP-UA to create the extended SIP-UA. Because the prototype does not have to support terminal initiated partial session transfers, not all functionality described has to be added. Actually the only functionality that should be added is the handling of re-INVITE messages with the pst-control extension. The SIP-UA should recognize this extension and respond appropriately. To keep

the prototype simple it will not present the transfer proposal to the user, it simply always accepts the proposal.

For the basic SIP-UA, the prototype uses the SIP Communicator [14], more specifically the SIP Communicator being used in the MMSP-UA of the Daidalos project. This SIP Communicator has been adapted in the Freeband 4g+ project [2] to support among other things re-INVITE and OPTION messages During tests done with the SIP Communicator a problem occurred related to the handling of SDP and the actual media, it does not support a media-stream specific connection-parameter in the SDP. Besides this, the way the SIP Communicator uses the RTP stack, it does not support separation of audio and video streams on this level.

The prototype of the SIP-UA has been adapted to solve this problem, the handling of the SDP and the coupling between the SDP and the RTP/Media stack has been changed to support a separated audio and video stream, making sure that a stream is not being reinitialized if not necessary. The basic SIP Communicator does not have the ability to use a file as audio or video-source. Because this functionality is very convenient during validation, the SIP Communicator has been extended to support this using configuration options.

As described in chapter 5 the developed solution uses the concept of 'make before break' to minimize the disruption of the media-streams during a partial session transfer. During tests with the SIP Communicator problems related to the sequence in which the different devices occured. The RTP stack being used by the SIP communicator only handles the first stream that comes in at a certain media endpoint. In case of the solution described in chapter 5 during the transfer first a new media-stream is being set up before the old one is being closed, with the SIP Communicator this new media-stream was never handled by the RTP stack. To solve this problem the prototype does not use the optimized sequence of invitations. Instead the sequence as described in section 7.1.3 is used.

A detailed list of changes that where made to the SIP Communicator in order to let it operated as SIP-UA in the prototype is given in appendix B.

7.3 Data model

The data model as designed in section 6.1 is used by both the SSC and 'pst-control' extended SIP-UA. However in the prototype as described in this chapter the SIP-UA does not implements all functionality, this means the data model is not needed by the MN to keep an up-to-date view of all the sub-sessions. Because with this prototype some functionality is not implemented the implementation of the data model is a bit simplified with respect to the design (See section 6.1). Figure 7.4 contains the class diagram of this data model that is being used by the SSC.

As the implementation of the MN cannot initiate partial session transfers, the MN will not have more than one session, which means the specific instance of MNMobilitySessionDevice does not have multiple instances of SessionDescription. This means the directed aggregation relation between the MNMobilitySessionDevice and SessionDescription as shown in figure 6.1 is not necessary anymore.

Also because a MN will not have any sub-sessions with the LDs, it will not use the directed aggregation relations between SessionDescription and MediaDescription as shown in figure 6.1. This relation is only used when the MN explicit uses sub-sessions, but because in the prototype only the SSC will handle the sub-sessions it is not used.

As described in the design of the data model (Section 6.1) the data model as presented there only show the consistent view on the relations and constraints, however while a partial session transfer is going on the data model might not be consistent. For the SSC this means for example that it first creates an instances of MediaDescription and an instance of SessionDescription before it actually invites a node. In this situation that instance of SessionDescriptions is not coupled to any MobilitySessionDevice; also MediaDescriptions that belong to one of the SessionDescriptions of the devices can be used in these SessionDescription.

So the data model as described in this section does only represent the view on the sessions and sub-sessions before a partial session transfer is initiated or after a partial session transfer has been finished. The period in between, while executing an actual partial session transfer is not modelled.

Another change in the data model with respect to the design is the coupling with the SDP API as defined by JSR 141 [10]. The prototype uses the reference implementation of this API to handle all SDP-bodies. The following classes of the implementation are related to SDP API classes:

- *SessionDescription*: This class is always related to an implementation of the interface javax.sdp.SessionDescription of the SDP API. The SessionDesription class uses this implementation of the javax.sdp.SessionDescription interface as basis, and adds extra functionality needed for partial session mobility.
- *MediaDescription*: An instance of MediaDescription is always related to an instance of an implementation of the javax.sdp.MediaDescription interface. Just as the SessionDescription, MediaDescription uses javax.sdp.MediaDescription as basis and adds extra functionality needed for partial session mobility. One constraint in this relation is that the instance of the javax.sdp.MediaDescription implementation should be coupled with the java.sdp.MediaDescription that is coupled with the same SessionDescription as the MediaDescription itself.
- *MediaPortDescription*: An instance of MediaPortDescription is always related to an instance of the implementation of the javax.sdp.Connection interface. As a constraint the instance of javax.sdpMediaDescription the instance of java.sdp.Connection is related must be the same instance as the MediaDescription that holds that MediaPortDescription is related to.

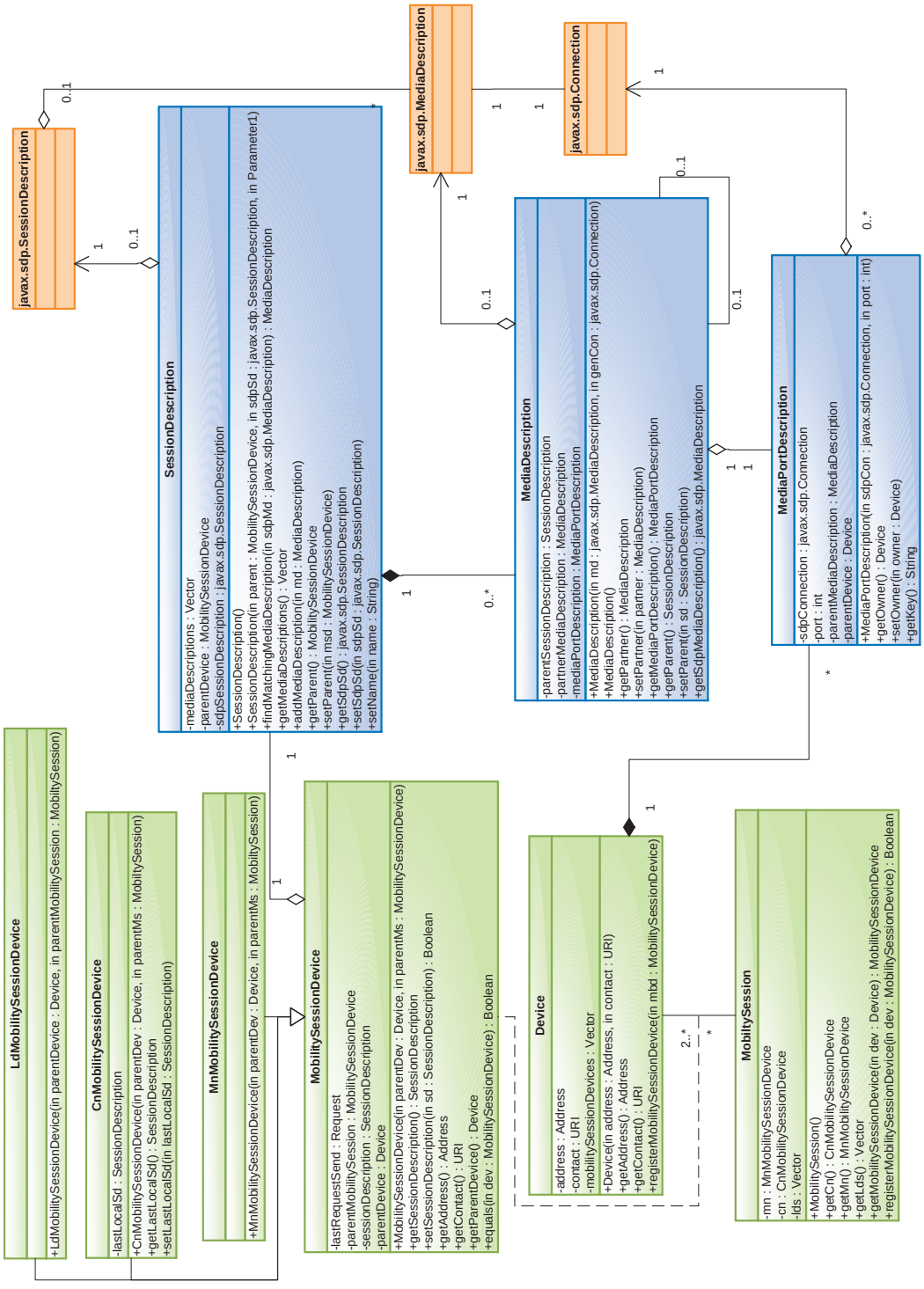


Figure 7.4: Class diagram - Data model

Chapter 8

Validation and discussion

This chapter describes the validation of the method for partial session mobility described in chapter 5 using the prototype designed (See chapter 6) and implemented (See chapter 7). The first section introduces the validation criteria based on the technical requirements (Section 4.1). The second section describes the validation setup, while the third section describes the procedure to validate the prototype. The last section describes the results of the validation.

8.1 Criteria

This section defines the validation criteria that will be used in succeeding sections. These validation criteria are based on the technical requirements as defined in section 4.1. Because the prototype implementated as described in the previous chapter does not contain all functionality contained in the developed method, a number of requirements are not used as validation criteria.

Below is the list of the technical requirements criteria describing the corresponding validation criteria and how these validation criteria can be used to validate the prototype.

1. *Terminal initiated partial session transfer.* To validate this requirement, the MN should be able to initiate a partial session transfer. Because the solution uses the functionality of the 'Mobile-node control'-mode for terminal initiated partial session transfers, this has already been validated using a prototype

2. *Network initiated partial session transfer.* This requirement can be validated by executing a network initiated partial session transfer using the prototype as part of the validation procedure.

3. *Combined terminal and network initiated partial session transfer.* To validate this requirement this should also be a part of the validation procedure. But the prototype does not implement the complete advanced SIP-UA as described in section 7.2, therefore terminal initiated partial session transfer cannot be executed with this prototype. However as described above, terminal initiated partial session transfers are already validated as described in [19] using the 'Mobile-node control'-mode. The only aspect of this requirement that cannot be validated is the processing of the SSC to understand exactly what is going on in case of a terminal initiated partial session transfer. The implementation and validation of this part can be considered as future work.

4. *Retrieve an already transferred media-stream.* As described in section 4.1 it must be possible to transfer an already transferred media-stream again both terminal and network initiated. This requirement implicates that in case of a partial session transfer that has been initiated by either the MN or SSC, it should be possible for either the MN or SSC to initiate another partial session transfer for the corresponding stream. This means this requirement depends on the requirement above.

   To validate this requirement it is therefore not necessary to have all possible situations concerning terminal and network initiation related to retrieving an already transferred media-stream in the validation procedure. This requirement can be validated by executing a network initiated partial session transfer that transfers an already transferred media-stream to another device. The other situations are implicitly validated by the validation of the previous requirement.

5. *Media-parameter control.* The method itself does support the initiator to use whatever SDP-body it wants to use, while the capabilities of a device are also concerned. The prototype SSC does not implement the ability to define the exact SDP-body; therefore this cannot be validated with this prototype.

6. *Minimize disruption.* The solution has been designed with the optimization as described at this requirement in section 4.1. The SIP Communicator that is being used as SIP-UA in the prototype however does not handle the optimized order of re-INVITE messages very well as already described in the previous chapter.

   Although it is not possible with this prototype to validate the optimized disruption time while doing a partial session transfer, it is possible to look at the associated timings for the different steps in a partial session transfer. As their are no hard requirements defined on the time that the different steps in the partial session transfer may take in a typical situation, this validation step only results in numbers that give the reader a better perception of partial session mobility.

   As the delays introduces at the SIP-UAs, depend very much on the specific SIP-UA that is being used by the users for this validation only the delays at the SSC will be measured. As basis for this measurements the logs produced by the SSC will be used; those logs contain timing statements that shows how much time is used to execute certain steps in the procedure as implemented in the prototype (See chapter 7). The reader should keep in mind that the logging to the file and terminal also takes time, and could lay a large effect on the total amount of time taken.

7. *Compatibility.* The level of compatibility does not have to be validated using the validation procedure, because no hard requirements have been defined. The solution has been designed such that the SSC can also operate with basic SIP-UAs. However full functionality is only achieved with extended SIP-UAs.

   As seen with the SIP Communicator during the implementation phase (See section 7.2) there could be problems with SIP-UAs not completely supporting RFC 3261 [44], also problems could occur when the RTP-stack does not handle multiple media-streams very

well. For this thesis no research is done to find the scale of these problems with existing SIP-UAs. This can be considered as future work.

8. *Separation of concerns.* The design of the solution is based on the 'Mobile-node control'-mode that makes sure the CN is not involved with the partial session transfer on the control plane. The same applies for the developed solution as can be seen in chapter 5. Because this is more a design issue, that has indeed been considered in the design, this is not a criteria used for validation.

9. *Robustness.* As described earlier the 'Mobile-node control'-mode does have an issue with robustness (As described in chapter 8) when the MN suddenly disconnects. This issue will not be validated with the validation procedure, however the results of the validation describe the theoratical approach on this issue.

8.2 Setup

This section describes the validation setup used to validate the prototype described in section 7. The setup consists of a number of computers all connected to the same switch that contain one or more of software components of the prototype. Below is a list of the computer systems that are used for the validation process:

- PC-1: This is a desktop PC with two p3-1Ghz processors and 512 Mb ram. It is running Windows 2000 and the IP-address of this machine is 135.85.86.61
- PC-2: This is a werver with a p3-1Ghz processor with 512 Mb ram. It runs Linux and the IP-address of this machine is 135.85.86.100. It is not used to play audio.
- PC-3: This is a server with two xeon-3Ghz processors with 2Gb ram. It runs Linux and the IP-address is 135.85.86.101. Also this server is not used to play audio.
- PC-4: This is a Desktop PC with two p3-1Ghz processors and 512 Mb ram memory. It runs Windows 2000 and the IP-address of this machine is 135.85.87.11

Each software component of the prototype or instances of a software component is located at one of the computer system described above. Below a list of the software components, describing the possibilities and the computer system they are located on:

- MN: Will be running on PC-2 using port 6000 for the sip signalling, 22224 for audio and 22222 for video. The MN does not use any source device or file to send data to other nodes.
- CN: This entity runs on PC-1 and uses port 7000 for the sip signalling, 23224 for audio and 23222 for video. The CN is the only entity that sends audio and video, the other entities only receive audio and/or video. It uses an audio-file and a video-file as source for the media.
- LD: Runs on PC-3. This is a video-only device, if an audio-stream is being transferred to this device it is not played. It uses port 8000 for sip signalling, port 24224 for audio and port 24222 for video A laptop, P4 2Ghz, 512mb

- LD2: Runs on PC-1. LD2 supports both video and audio playback. It uses port 9000 for SIP signalling, port 25224 for audio and port 25222 for video
- SSC: Runs on PC-4 and uses port 5000 for SIP signalling.

All SIP-UAs are configured such that the SSC is the proxy and registrar. The only device sending audio and video is the CN; the other devices do not start transmitting real audio of video data.

8.3 Procedure

This section describes the procedure needed to validate the solution described in chapter 5 using the prototype described in chapter 7. The procedure has been set up to make sure all validation criteria as described in section 8.1 are handled. Below is the list of steps that define the validation procedure. The list is divided in the separate partial session transfers that it contains. Each of these transfers is divided in detailed steps describing the procedure.

1. Setup
    - The point of departure is the situation where all components are registered at the SSC and no sessions have yet been set up.
    - The MN calls the CN. The GUI of the CN should show that it is being called by the MN.
    - Via the GUI of CN the call is accepted. (The MN should now start to display the video sent by the CN, and play the audio sent by the CN).
2. Audio from MN to LD2
    - The GUI of the SSC is used to change the endpoint of the audio-stream from the MN to the LD2.
    - As described in chapter 7 the MN automatically accepts the proposed partial session transfer.
    - Because the prototype does not use the optimized 'make before break' sequence of inviting the involved nodes, as described in chapter 7, the MN already stops playing the audio-stream.
    - The GUI of LD2 should show an incoming call from MN (The SSC invites LD2 on behalf of the MN).
    - Via the GUI of LD2 the call from MN is accepted. Now the audio stream should start playing on LD2.
    - The GUI of the SSC should also show now graphically that the audio stream exists between CN and LD2, while the video stream still exists between CN and MN.
3. Video from MN to LD
    - Via the GUI of the SSC a partial session transfer is proposed to change the endpoint of the video-stream from MN to LD.

- After MN automatically accepted the transfer, video stops playing at the MN (Because the sequence of invites is not optimized), and the GUI of LD shows it is being called by MN.
- The GUI of LD is used to accept the call.
- Now LD should start playing the video it receives from CN. The GUI of the SSC now should show the video stream exists between CN and LD, while the audio stream still exists between CN and LD2.

4. Video from LD to LD2

   - The GUI of the SSC is used to propose a partial session transfer to change the endpoint of the video-stream from LD to LD2.
   - MN automatically accepts, after which LD stops playing the video.
   - In contrast with the previous transfers the GUI of LD2 does not show it is being called by the MN, because there is already a call going on that is used to also contain the video stream. This means LD2 should automatically start playing the video stream it receives from CN
   - The GUI of the SSC should also show graphically that the video stream exists between CN and LD2

5. Audio from LD2 to MN

   - Via the GUI of the SSC the transfer of the audio stream to MN is proposed.
   - MN automatically accepts the offer, after which the audio playback at LD2 should stop.
   - The re-INVITE at the MN is automatically accepted, after which MN starts playing the audio it receives from CN.
   - The GUI of the SSC now should show the audio stream exists between CN and MN.

6. Video from LD2 to MN This procedure is the same as the previous, but instead of the audio stream now the video stream is transferred.

For each of the partial session transfers as described above the different steps that are taken by the SSC in such a transfer will be measured using timestamps printed in the logs. This way the reader gets a perception of the time it takes to execute a partial session transfer using the solution as described in chapter 5 with the prototype described in chapter 7.

Also using these time-measurements the most important areas can be identified in which optimizations have the most affect on the total disruption of the involved media-stream. The next section describes the results obtained from the procedure using the prototype with respect to the validation criteria.

8.4 Results

Below for each of the validation criteria described in section 8.1 the results are given:

- *Network initiated partial session transfer.* Almost all stages of the validation procedure contain a network initiated partial session transfer. Because all those stages where succesfully executed the prototype (See chapter 7) shows the solution developed (See chapter 5) succesfully executes a network initiated partial session transfer.
- *Retrieve an already transferred media-stream.* Stages 4,5 and 6 of the validation procedure (See section 8.3) describe transfers of already transferred media-streams, where stages 5 and 6 contain a transfer of the stream back to the original device (MN). As those stages where succesfully executed using the prototype it is shown that with the solution developed an already transferred media-stream can succesfully be retrieved on the original device or transferred to another device.
- *Minimize disruption.* As described this validation criteria here the delays of the different steps in the procedure used by the SSC are given. The reader should keep in mind the logging used to measure the different steps also introduce an extra delay, however no indication is given showing how big this delay precisely is. The following steps are considered in a complete network initiated partial session transfer:
  - Inviting the MN
    - From incoming task to initiate a partial session transfer to the extended INVITE message being send to the MN. On average this step took 122 ms.
    - From the incoming OK response sent by the MN to the end of the processing of this OK response. On average this took 50 ms.
    - From the end of processing of the OK response to the sending of the ACK to the MN. On average this took 37 ms.

    Combined the steps necessary to propose the partial session transfer to the MN took on average 209 ms.
  - Inviting the source
    - From the start of inviting the source to the actual sending of the INVITE to the source. On average this took 90 ms. One of the measured results has not been taken into account here, because the invites took a lot longer than the other four times caused by a re-transmit.
    - From the OK response received from the source to the end of the processing of this OK. On average this step took 77 ms.
    - From the end of processing of the OK response received from the source to the sending of the ACK to the source. On average this took 32 ms, however one of the results was not taken into account because a retransmit of the OK came in during the processing, therefore the time it took to execute this step was twice as long as the rest.

    The total time it took for the SSC to invite the source of the media stream that was transferred took on average 199 ms.
  - Inviting the target
    - From the start of inviting the target to the actual sending of the INVITE to the target. On average this step took 87 ms.
    - From the OK response received from the target to the end of the processing of this OK-response. On average this step took 65 ms.

    - From the end of the processing of the OK response received from the target to the sending of the ACK to the source. On average this took 34 ms.

  The complete invitation of the target device of the transferred media streams took on average 186 ms.

- Inviting the CN
  - From the start of inviting the CN to the actual sending of the INVITE to the CN. On average this took 81 ms.
  - From the receival of the OK-response from the CN to the end of the processing of this OK-reponse. This step took on average 75 ms.
  - From the end of the processing of the OK-response received from the CN to the sending of the ACK to the CN. This took on average 31 ms.

  The process of inviting the CN took 187 ms of processing time by the SSC.
- The finishing of the partial session transfer. This means making sure the data-model is consistent with the actual sessions. On average this took 37 ms.

When counting the measurements presented above, on average a network initiated partial session transfer took 818 ms of processing time at the SSC. As said before the reader should keep in mind the SSC uses logging, which probably has a considerable influence on these measurements.

When looking at how these measurements impact the total disruption time one should consider that some of the steps described above are not in the critical path, meaning a number of steps are taken before the media-stream stopped at the source device, and some of the steps are taken after the media-stream started at the target device. If looked at the procedure implemented in the prototype the critical path starts just after the invitation of the source when the source received the ACK message, meaning it should stop its side of the media-stream. The critical path ends just after inviting the CN, when the CN starts up its side of the new media-stream. On average this critical path took 373 ms.

As said before in section 8.1 the delay introduced by the SIP-UAs to set up the media-stream really depends on the specific SIP-UA implementation. At the same time whichever method is used to execute partial session transfers, this does not change anything to the time it takes to process the SDP-body and initiate the corresponding media-streams at the SIP-UA.

As also described in section 7 the prototype does not use the 'Make before break' concept because the SIP Communicator does not handles the media-streams right. However when this concept would be used, the critical path in the processing steps of the SSC whould be completely gone. when looking at the steps necessary to execute a partial session transfer the invitation of the source whould be done after the invitation of the CN as also shown in figure 5.1. This means only the time it takes for the SIP-UAs to start the streaming and the time it takes for the packages to be transferred over the network effect the total disruption time.

One issue all current available methods for terminal initiated partial session transfer have problems with is robustness. With the 'Mobile-node control'-mode at the moment the MN suddenly disconnects, the problem occurs that an already transferred media-stream continues and cannot easily be stopped by the user. However as described in section 5.7 the SSC is

involved in all sessions; if the SSC can detect the communication failure with the MN, it should be able to stop involved sub-sessions. Because the current solution does not yet contain a way the SSC could obtain this information, this solution is not as robust as it ideally should be. However this solution does provide a possible solution that should be worked out a bit further. This is considered as future work.

The results above show that the solution developed (See chapter 5) does indeed fulfil the requirements that could be used to validate the solution with this prototype (See chapter 7. Because the prototype does not yet implement all functionality, not all aspects of the solution could be validated.

8.5 Discussion

This section discusses a number of topics that are related to the assignment, but are not part of it. Those topice could however be important for future developments in this area and should be considered before partial session mobility as described in this thesis is being deployed as a service to real users.

A big issue related to any information send via wireless or non-wireless networks is privacy. In the context of partial session mobility the context reasoning node needs information about the location of available devices, the location of the users and information about the media-streams the users use. All this information could be privacy sensitive, e.g. a user might not want other devices or users to know that he is nearby. Also the maintainer of a device might not want certain users to know it is nearby the device.

Things could get really complicated because for every device or person other privacy constraints might apply, because every person might have other demands related to this issue. Another related issue is the possibility that a user can lend out its own device in case other users in the vicinity wants to use it and the device is not used at the moment. This could be the case if user A has a audio-conference with a friend. User B, a colleque of user A,does have a video-conferencing enabled PDA in his jacket. User B did set up a user profile that states that colleques in the vicinity are allowed to use its PDA for video-conferences for a fixed rate. This way every user could become a service provider using a centralized system located in the network that combines the offers and demands.

When talking about rates for certain services another issue is touched, namely how are the costs for using devices and network capacity for partial session transfers being spread among the involved users. There are different forms of charging, each have their implications. Session based charging enables charging based on SIP signalling. In case of partial session mobility it would be most logical that the MN has to pay for sessions with the LDs, because it is a service offered to the MN. With ession based charging this is relatively easy because the sub-sessions are being set up by the MN or on behalf of the MN. With volume based charging the actual bandwidth being consumed is being charged. When using partial session mobility, media-streams flow between the CN and MN, and between the CN and LDs. As stated before it would be logical the MN should be paying for this, however this would make charging more difficult, because in this case it is not one of the nodes involved in the media-streamthat that have to pay.

Another issue related to charging is the way how the billing is done, pre-paid or post-paid. In case of post-paid billing the total time or amount of data is being logged in the system, and afterwards a bill is send to the user. In case of pre-paid billing the system should on the fly keep record of how much time or data is left before the connections are disrupted. Maybe the a message should be sent to the user so the user knows the pre-paid money has almost been spend.

Another money related issue is the way the user is informed of how much it costs to use another device to handle a media-stream in its current session, what are the effects. One could imagine that transferring a stream to another device on one hand causes a decrease in costs related to the session with the MN, while on the other hand a new session is being set up, which also costs money. Anyway, the user should be notified about the financial effects of the partial session transfer.

Also an issue that arose during the development of the solution as described in this thesis, is the ability to support partial session mobility to two users in the same domain using the same sub-session controller (SSC). If two users in the same domain both want partial session transfer support and are both using the same SSC, in this SSC two mobility sessions would be needed. One mobility session having the first user as MN and the second as CN, and one mobility session having the first user as CN and the second as MN. In this situation the SSC should internally be able to send SIP messages between those mobility sessions and correctly recognize which mobility session should handle which incoming SIP message.

This gets even more complicated when partial session mobility is being deployed on a large scale, meaning multiple SSCs are necessary to handle the service. In that case SSCs should be aware of each other, the mobility sessions each of them has and the what devices are being present in those mobility sessions. For this to work the SSC must somehow be distributed in a number of nodes that are able to access the knowledge of each other; a centralized shared database could be used for this.

Chapter 9

Conclusion and future work

This chapter describes the conclusions that can be drawn from the work described in this thesis. This includes answers to the research question and objectives described in section 1.2. The second section gives an overview of related issues that are considered as future work.

9.1 Conclusions

As described in section 1.1 a number of ongoing developments, such as higher bandwidth access technologies (including wireless technologies), IP-based multimedia subsystems, and the corresponding developments in the networks of telecom providers, pave the way for new types of services for the end-users, enabling more succesfull communications. This motivation lead to the main research question of this thesis: *'How can different devices in the vicinity of the user be used to enhance an existing multimedia-call the user participates in?'*. Based on this research question and the objectives defined in section 1.2 this section describes the conclusions that can be drawn from the solution that has been developed and validated as described in previous chapters.

The big challenge in fulfilling the objectives was to develop a solution that combines a number of functionalities while also making partial session mobility accessible for the end-users. The core of this challenge is to make network initiated partial session mobility possible, while still making sure the user is aware and does approve. The solution described in chapter 5 supports both terminal and network initiated partial session mobility. If the user-terminal does support the needed extension the user is always aware and does agree with network initiated partial session transfers. Also in case of a user-terminal that does not yet support the extension a network initiated partial session transfer can be executed if the user did set this up in the its user preferences stored in the network. Terminal initiated partial session transfer is supported by using the 'Mobile-node control'-mode that was used as basis of the development of the solution; the sub-session controller interprets terminal initiated partial session transfers to make sure it has an up-to-date view of the sub-sessions and the related media-streams.

Using the prototype that was developed (See chapter 7) a lot of the aspects are validated as described in chapter 8. Not all aspects could be validated, because the prototype did not yet support all functionalities; it does not yet support terminal initiated partial session transfers, and also does the SIP-UA not yet show the user if it received a proposal for a partial session transfer. Despite the 'Mobile-node control'-mode that is used for these transfers has

already been validated by the authors of that method, the correct processing of a terminal initiated partial session transfer by the SSC could not be validated because of this. However as described in the design of the SSC (See section 6.2) the developed solution should in theory enable the SSC to actual do this.

As described in section 8.4 all currently available methods are not as robust as they ideally would be. The solution described in this thesis does contain a possible solution for this, but more research is necessary for this possibly the solution should be further extended.

Section 4.1 describes as part of the technical requirements that the component that initiates network initiated partial session transfers should be integratable in IMS. As also described there the component in the architecture of IMS that perfectly fits that job is an application server because an application server may act as a B2BUA, meaning it is able to keep track of ongoing SIP sessions and manipulate those SIP sessions and connection those with each other. The SSC as used by the method in this thesis therefore can be integrated in IMS as an application server.

One of the objectives contains the possibility to retrieve an already transferred media-stream back to the original device, or more general and a bit broader transfer an already transferred media-stream to another device. As a number of steps of the validation procedure show, the solution as described in this thesis does support this.

The initiator of a partial session transfer should be able to control the media-parameters of a transferred media-stream. The 'Mobile-node control'-mode is the only currently available method for terminal initiated partial session transfer that enables the MN to control the exact SDP body that is offered to the different nodes. As described in chapter 5 in case of a network initiated partial session transfer the SSC is able to have the same level of control on the SDP-body.

As described in chapter 7 with the current prototype it is not possible to use the optimized procedure of inviting the involved devices to minimize the disruption of the transferred media-stream that is used in the original solution described in chapter 5. However with the theoretical approach as described in section 8.4, it can be concluded that in case of the optimized procedure the processing time needed by the SSC to handle the network initiated partial session transfer does not influence the disruption of the media-stream during the transfer.

9.2 Future work

This section describes a number of issues, extensions or enhancements that came up during the research that where considered out of the scope of this particular research. Each of the issues described here can be a starting point for future research.

As described in chapter 7 the current prototype can be extended to support all functionality of the developed method (See chapter 5 The following list gives an overview of these recommended extentions::

- Add support for terminal initiated partial session transfers to the SIP-UA.

- Extend the handling of the proposal for a partial session transfer in the SIP-UA. This means the user should be prompted with the details of the partial session transfer, and accept or decline the partial session transfer. As an extension to this, the SIP-UA could handle the proposal based on user-preferences stored at the SIP-UA.
- Extend the SSC to handle terminal initiated SIP requests. Based on this the SSC can be extended to recognize terminal initiated partial session transfers. This includes the processing of a terminal initiated partial session transfer in the data model. Besides this it this extension to the SSC could also include the processing of BYE-messages in the data model.

Section 8.5 describes a scalability issue related to deploying the solution on a large scale. As described in that case the functionality of the SSC must be distributed among a number of different nodes. Further research is necessary to find a suitable solution to make the SSc scalable.

The 'Mobile-node control'-mode used as basis for the development of the solution uses an optimization to minimize the disruption of the involved media-streams. As described in chapter 5 the solution already does use an optimized process of inviting the involved devices to minimize this disruption. Further research can be done to find out if the optimization used be the 'Mobile-node control'-mode might further reduce the disruption of the involved media-streams.

When combining user mobility with the solution for partial session mobility as described in this thesis an issue could arise, this issue is related to the ability of the SSC to remember if a device supports the extension as described in chapter 5. In case a user at first uses a device that supports the extension, while later on the user starts using another device while keeping the same session, the SSC still thinks the device of the user does support the extension. This could become a problem if the MN is still responsible for a certain sub-session. Further research can be done to find out if there are other ways for the SSC to now that the device used by the user does not support the extension anymore.

An issue that arrised in almost all multimedia systems is the synchronisation between the different media. This issue often occurs when only one device is used that recieves both media-streams. In case of multiple devices that each recieve a media-strean, as with partial session mobility, the issue of synchronisation even get bigger. Research can be done to find solutions for synchronisation of different media-streams even when these media-streams end on different devices.

Future research can also be done on extending the possibilities of partial session mobility in general. One could thing of the merging of external multicast or broadcast streams into an ongoing session between two users. Another interesting topic could be the support of partial session mobility in sessions between more than two users, these sessions might use a conference server that might support partial session mobility.

The method described in this work has been submitted as internet draft to the IETF to finally become an internet standard. This internet draft is enclosed in appendix C.

Appendices

Appendix A

Context information

This appendix gives the reader an overview of the information that can be used to decide at what moment a media-stream should be transferred to another device and how this information can be obtained. As described in section 4.1 it is assumed there exists a context-reasoning node that does the actual reasoning [26], and therefore needs to obtain this context information. Research on these topics is for example done in the Awareness project [3] and WP4 of Daidalos-2 [5]. For information about managing context information and reasoning based on this context information we refer to previous research [50, 24, 49] that has been done on this subject.

Below are a number of different types of information that can be used by the context reasoning node to decide whether a certain media-stream must be transferred to another device at a certain moment in time:

- Information about how the device is connected. This includes information about the connection, such as the transfer speed, the delay and price.
- Information about where the device is located. This information can be acquired from GPS devices, the current connected access points, beacons, calendar information, etc.
- Information about the capabilities of the device? This information tells if the device can send or receive video or audio data, and which quality/codec's it does support.
- Information about the status of the device. This includes information about the availability of the device or user (Presence information).
- Information about preferences of the user. The context-reasoning node might have access to the HSS in IMS of the telecom provider it is installed at, so preferences that are stored there can be used. It is also possible for the user to configure preferences at the user agent. If the application server sends an offer to the user agent, the user agent might handle this offer without user-interaction.

The next question is how and at what moment (e.g. At registration or at the set up of a session) the context-reasoning node can obtain this information. This depends on the type of information and whether the information can be obtained during the registration, session-setup or in between. IMS supports presence services [12] using the presence information data format (PIDF) as defined in RFC 3863 [47]. As described below most of this information can

be encapsulated in the presence information. This ensures the context-reasoning node can easily obtain the necessary information using the presence services offered by IMS.

The location of devices might be one of the most important factors in deciding if and to which device certain session media should be transferred. The GEOPRIV Location Object Format (See RFC 4119 [39]) supports the encapsulation of the location of the device in presence information. Using this technology it is possible to define the location based on different methods such as civic location (Country, city, address, etc), global positioning system (GPS), manually, triangulation, etc.

Also the status of the device can be encapsulated in presence information. Internet Draft [21] proposes a method that makes it possible to add more details of the connection to the presence information, such as type of connection, throughput and the costs of using it. However this method is not yet supported by IMS.

Another type of information that might be valuable to the context-reasoning node is information about the capabilities of a specific UA. Internet Draft [35] describes an extension to presence information data format (PIDF) to add support for this kind of information. For example, using this extension the context-reasoning node can find out if a certain UA supports audio- and/or videostreams. With this extension the context reasoning node cannot acquire information about the supported codecs by a certain UA. However it can acquire this information using the SDP negotiation or via SIP OPTIONS [44] The choice between those two methods depends on the specific method used for partial session transfer.

Appendix B

Changes to SIP Communicator

This appendix contains a detailed list of changes made to the SIP Communicator as SIP-UA in the prototype as described in section 7.2.

- Changes in general part of SIP Communicator:
  - Added Log4J specific code.
  - Changed the main-function to have the log4j.xml file reference as parameter during the start up of the SIP Communicator.
  - Changed the handling of a stopped media-player, hereby not all player-components are removed from the GUI anymore. Only the really involved components are removed. With this possibility it is possible to let a non-altered media-stream be uninterupted playing while another stream is being altered.
  - Making the SIP Communicator Java 1.5 compatible: Some classes used a varibale with the name 'enum'; this is a restricted word in Java 1.5.
  -
- Changes in the SIP part:
  - Added 'pst-control' handling in the call-processing. The current version of the SIP Communicator does not look at the required header at all specified in RFC 3261 [44]. If the extended SIP Communicator receives an pst-control enabled INVITE message it now makes sure it does not handle the SDP-body in the normal way, and always responds with OK.
  - To do this also the handling of re-INVITE messages was changed.
  - The SIP Communicator did put port and other params in the address in the 'to'-header; this way the SIP Communicator on the other side could not recognize itself, because it expects its address without port and extra parameters. A change is made to make sure the SIP Communicator does not include those extra parameters in the 'to' and 'from' headers.
- Changes in the media-Management:
  - Changed the MediaManager class. This class is responsible to manager the handling of the actual media based on the SDP-bodies received. The following changes where made:

    - It keeps track of the different ongoing media-streams, this way it does not restart a media-stream when nothing changes.
    - It delegates the actual start-up of a mediastream (Two-way) to the new class MediaStream (See below).
    - It uses classes from the data model to inform the MediaStream-object about the content.
    - In the original SIP Communicator the audio and video-datasource where combined into one merged datasource. This way it was not possible to transmit the audio and video data to two different endpoints. In the extended version the two datasources are kept separated for this purpose.
  - Added a class MediaStream, to separate a part of the functionality from the MediaManager. This class corresponds with a specific media-stream. With this class this media-stream can be started and stopped. The changes to functionality that was done before in the MediaManager:
    - It supports Media-specific connections in the SDP. The MediaManager did only support global Session-description defined connections, this way partial session transfer using the 'Mobile-node control'-mode cannot occur because this method uses the Media-specific connection parameters. multiple mediaStreams in a single RTP-connection.
    - Each MediaStream uses an instance of RTPCommunicator to transmit and receive data via RTP.
  - Added a class RTPCommunicator, also to separate functionality from the MediaManager. This class manages the receiving from and receiving to an specific ip-address/portnr via RTP. To transmit data it uses the Datasource that has been initialized by the MediaStream. At the moment it receives a stream it starts a player. After a player started or a player stopped it communicates this with the MediaManager.

- Changes in GUI:
  - Individual removal of components displayed in the video-pane, that shows the media. This way it is possible to remove only a specific players visual component while other players can still display their contents.
  - Removed ControlPanelComponent from the video-pane. With this control component the user could control the video or audio handling (And see some information). It was not possible to remove this control component; so this way, once the visual component was removed, the control panel component would still be present. Now the control components are not placed on the video-pane this problem does not occur anymore.

Appendix C

Internet Draft: draft-aartsetuijn-nipst-00

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 27, 2007

J. Aartse Tuijn
University of Twente
D. Bijwaard
Alcatel-Lucent
February 23, 2007

A method for network initiated partial session transfers
draft-aartsetuijn-nipst-00

Status of this Memo

Abstract

This document describes a SIP-based method for network initiated partial session transfers that works together with terminal initiated partial session transfers. It uses the Mobile Node Control Mode for terminal initiated partial session transfers and it extends this method to support network initiated partial session transfers.

Table of Contents

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

As work in progress [shacham06] described, SIP Session Mobility is the seamless transfer of media of an ongoing communication session from one device to another. A system for session mobility is defined, aimed at allowing a Mobile Node (MN) to discover available nodes and to include them in an active session. [shacham06] also defines how the different parts of a session can be individually transferred to these devices, meaning the different stream endpoints in a multimedia-stream can each be transferred to another device, here called partial session mobility. Two methods are proposed, the Session Handoff mode and the Mobile Node Control Mode.

With both these methods the Mobile Node (The mobile terminal used by the user) initiates the partial session transfer. This document describes how a partial session transfer can be initiated from a dedicated node located in the network such that Mobile Node initiated partial session transfers are still possible based on de described method in [aartsetuijn06] and the work in progress [shacham06]. Network initiated partial session transfers would typically be triggered by external events like network-side discovery of nearby multimedia devices like beamers.

When the dedicated node in the network initiates a partial session transfers, the user may want to be involved in the decision; therefore the partial session transfer is proposed to the user-terminal to let the user decide about the actual execution of the partial session transfer. The following information is assumed to be necessary for the user decision:

- Which media-stream endpoint will be transferred
- The device to transfer the stream endpoint to
- The media-parameters to be used for the media-stream after the transfer

As described in [aartsetuijn06] the most applicable solution to base network initiated partial session mobility on is the Mobile Node Control Mode.

3. Overview of operation

This section describes the operation of network initiated partial session mobility; the Mobile Node initiated partial session transfers are assumed to be handled with the Mobile Node Control Mode as described in [shacham06].

3.1. Key concepts

The Mobile Node Control Mode uses the capabilities of SDP [RFC4566] to transfer media-stream endpoints to different devices. Since the media-stream endpoints are controlled solely on the control plane, network initiated partial session mobility can be supported entirely on the control plane, while the actual media-streams are set up directly between the devices.

A network node supporting network initiated partial session mobility should be part of all the session-specific SIP signalling of the nodes that want to use the service. This means the node can alter/ change and setup sessions on behalve of the involved nodes. A new session will be set up by the dedicated node in the network if it involves a new device to handle one or more media-streams. The sessions with other devices that help the Mobile Node (MN) handle media-streams are called sub-sessions

A mobility session is defined as a session of a Mobile Node with is peer in a session, and the sub-sessions between the Mobile Node and the additional devices that are used in the session with the peer.

Another important concept being used is using a proposal to let the user decide about the execution of a network initiated partial session transfer. This means the dedicated node located in the network must first propose a partial session transfer to the user or user-terminal, after which it is only executed if the user agrees with the proposed transfer.

3.2. Component overview

The network node to initiate partial session mobility is called the sub-session controller (SSC); it acts as a SIP B2BUA. In IMS [3GPP 23.228] this would typically be implemented as an application server.

The SSC can offer its service to each individual node on each individual session that has the SSC in its signalling path. In this document the MN is the user-terminal that the SSC is offering its service to. The MN does have an ongoing session with a corresponding node (CN). To accomodate the user-terminal other devices can be used

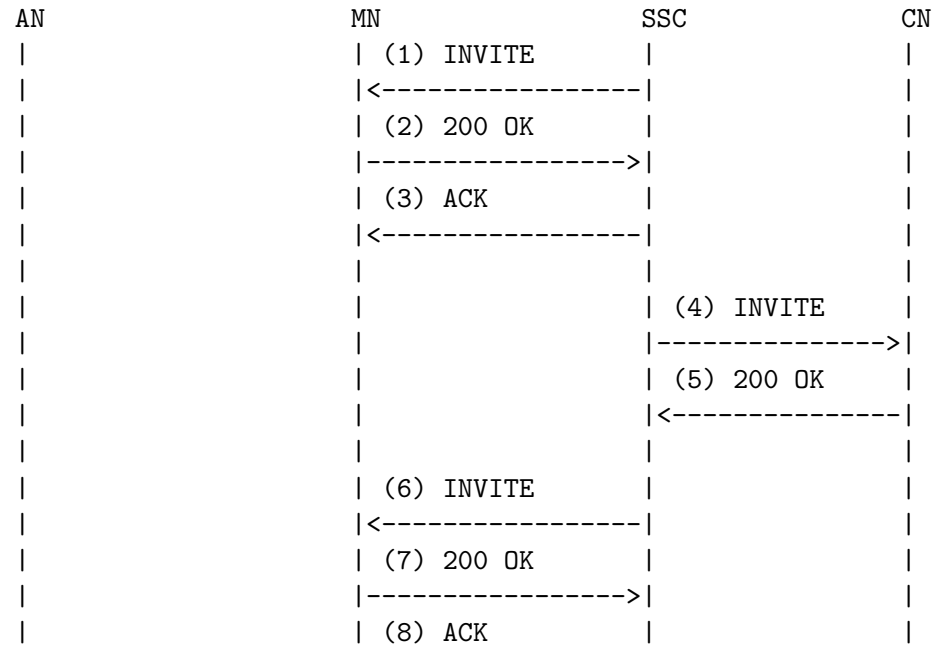
as endpoint for the involved media-streams, these devices are called local nodes (LNs). Examples of LNs are an audio node (AN) that can play and record audio and a video node (VN) that can play and record video. Ofcourse LNs can also act only as source or target for media-streams.

In case the CN also wants to use partial session mobility itself, it will play the role of MN in its own mobility session. This means a session between two devices can be represented by two mobility sessions, each of which contains the session between the two devices, but with each of the devices in another role.

3.3. Network initiated partial session transfer

Before a partial session transfer can be initiated a session must exist. The SSC is located in the signalling path of this session, and it acts as a B2BUA. Figure 1 shows the message sequence diagram of a typical network initiated partial session transfer. Below this message sequence diagram is explained in more detail.

To propose the partial session transfer to the user the SSC sends an INVITE message (1) to the MN. This message uses the option-tag 'pst-control' in the Required header field to make sure the MN can interpet the message correctly. If the MN does not accept the proposal it responses with 603 (Decline), as shown in Figure 2 , message (2). In that case the SSC should not proceed with the partial session transfer.



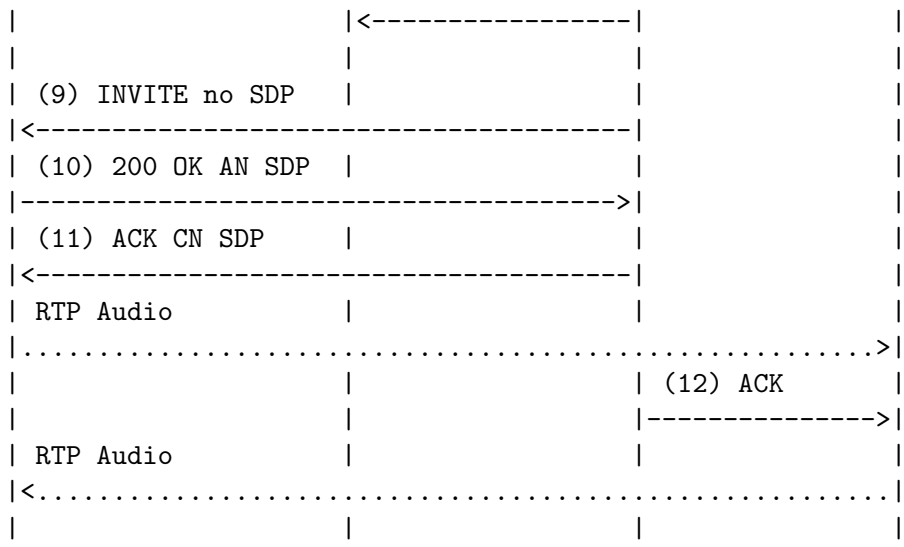


Figure 1: Network initiated partial session transfer

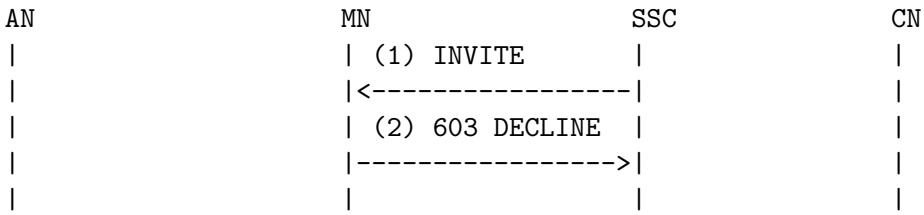


Figure 2: Decline of a network initiated partial session transfer

As described in Section 2 it is assumed the MN need to know which media stream will be transferred, to which device it will be transferred, and with which media-parameters. To show to what device the media-stream will be transferred, a new header field is introduced for INVITE messages, namely the Pst-to header field. This header field contains the SIP-URI of the device the media-stream will be transferred to.

To inform the MN about which media-stream will be transferred and with what media-parameters, the INVITE message meant as proposal contains a SDP-body. This SDP-body contains a session description proposal that will be sent to the CN when re-invited. By analyzing this session description and comparing it with the previous session description being offered to the CN, it knows which media-stream is being transferred, and what media-parameters will be used. Figure 4 contains an example of the SDP-proposal sent to the MN; the MN compares it to the last SDP session description it sent to the CN,

shown in Figure 3. The MN can determine which media-stream will be transferred by looking at the Connection parameter; the media-stream that is being transferred uses another connection parameter.

```
c=IN IP4 192.0.2.1
m=audio 24224 RTP/AVP 0 3 4 5 16 6 17 14 8 15 18
m=video 24222 RTP/AVP 34 26 31 33
```

Figure 3: SDP session setup

```
c=IN IP4 192.0.2.1
m=audio 24224 RTP/AVP 0 3 4 5 16 6 17 14 8 15 18
c=IN IP4 192.0.2.2
m=video 25222 RTP/AVP 34 26 31 33
```

Figure 4: SDP proposal

A new header field is added to the SIP INVITE message to inform the MN about the Call-id of the sub-session that will be set up with the target device of the transfer. The header containing this Call-id is called Pst-call-id.

SSC might know all capabilities in advance, so it can invite the LD and CN more precisely. If it does not, it invites the LN without any SDP.

The order in which the involved nodes are invited is important to the compatibility with SIPUAs. To make sure as much SIPUAs are supported as possible old media-streams first has to be stopped before new media-streams are being set-up as also described in [aartsetuijn06]. However this does mean a disruption in the media-streams. Figure 1 also illustrates this order by first inviting the MN (1) making sure the MN stops sending audio to the CN, then inviting the CN (4) making sure the CN stops sending audio to the MN and starts sending audio to the AN, and only after that inviting AN to let it send audio to CN. This sequence also makes sure multiple sources are not sending the media at the same time to the same endpoint.

When this method would be used in IMS, the SSC may also handle Circuit Switched call legs, and for this be combined with the CSI Application Server (See [3GPP 23.279]) that provides Voice Call Continuity (VCC) between Circuit Switched networks and IMS.

3.4. Retrieval of a media-stream

Retrieval of a media-stream can be initiated from both the terminal and network. In case the media-stream was transferred by the terminal, it can also easily be retrieved by the terminal as described in paragraph 5.3.2 of [shacham06]. If a media-stream was transferred by the SSC, the SSC can transfer the media-stream back to the MN by starting a network initiated partial session transfer.

When the SSC has transferred a media-stream, and the MN wants to retrieve that media-stream, the MN knows about the transferred media-stream because it was informed by the proposal (INVITE-message) sent by the SSC. Because it has an up-to-date view on the current situation it is able to determine if a retrieval is necessary, and how it should be accomplished. Figure 5 shows the sequence diagram of this, where the MN only has to re-invite the CN to retrieve the audio-stream. Here the SSC does have the responsibility (As described in section Section 7) to make sure the sub-session with the AN is consistent with the session between the MN and CN.

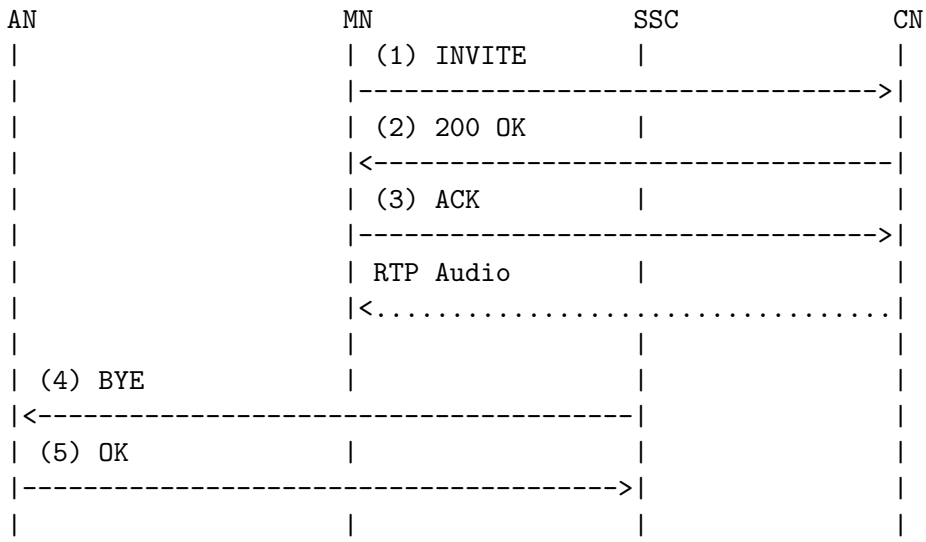


Figure 5: Retrieval by Mobile Node

The other way arround, where the MN has transferred a media-stream, and the SSC intends to transfer that media-stream back to the MN, the SSC must have an up-to-date view on the current situation. Because the SSC acts as a B2BUA on every session that is related to the MN, it is able to process all changes made in the session(s). This way the SSC always has an up-to-date view of the current situation.

4. 'pst-control' option-tag

A new SIP option-tag called 'pst-control' is defined for the Require and Supported header fields.

A user agent including the 'pst-control' option-tag in a header field indicates compliance with this specification.

This 'pst-control' option-tag may be included in the Require header field of re-INVITE's sent by a B2BUA. Nodes controlled by end-users should not use this option-tag in the Require header field.

A B2BUA that wants to propose a partial session transfer to a certain node for a specific call must include the 'pst-control' option-tag in the Require field of the re-INVITE message that represents the proposal.

When the 'pst-control' option-tag is present in an INVITE request, it also MUST contain a Pst-to header field and a Pst-call-id header field.

5. The Pst-to header field

The Pst-to header field can appear in INVITE requests where the 'pst-control' option-tag has been set. This field contains the SIP-URI of the device the media-stream will be transferred to. The following is the ABNF (augmented Backus-Naur Form) for the Pst-to header field:

Pst-to = "Pst-to" HCOLON (name-addr / addr-spec) * (SEMI generic-param)

6. Pst-call-id header

The Pst-call-id header field can appear in INVITE requests where the 'pst-control' option-tag has been set. This field contains the call-id of the sub-session that will be set up with the device identified by the Pst-to header field. The following is the ABNF (augmented Backus-Naur Form) for the Pst-call-id header field:

Pst-call-id = "Pst-call-id" HCOLON callid

7. Behaviour of SSC

When the SSC wants to offer a partial session transfer to a user, it sends an INVITE request to the MN containing the 'pst-control' option-tag in the Required header, the Pst-to header field to indicate the targer of the transfer, the Pst-call-id header field to identify the sub-session being set up with the target device and the SDP-body to indicate what stream is being transferred and what the parameters of it are after the transfer.

In case the SSC recieves a 306 response (Decline), meaning the MN does not want the proposed partial session transfer to be executed, the SSC does not continue with the execution. When the MN does not support pst-control it will respond with a 420 response (Bad Extension), while the Unsupported header field contains the 'pst-control' option-tag. The behaviour of the SSC in case the MN does not support pst-control is implementation specific. The SSC could for example use user preferences to determine if it should continue or stop the execution of the partial session transfer.

Besides a network initiated partial session transfer, a partial session transfer can also be initiated by the user-terminal. This influences the way how the sub-sessions are handled and by which node they are handled. In case the MN supports pst-control, the node that made the last change to a sub-session is responsible for that specific sub-session. This responsibility consists of making sure the sub-session is consistent with the session between the MN and CN. In case the MN does not supports pst-control, the SSC is always responsible for the sub-sessions.

Because the SSC act as a B2BUA that is located in the signalling path of all sessions between the device that acts as MN and it peers, it receives all messages of the involved nodes and must interpet them and make sure these messages are forwarded to other nodes accordingly. As described above, if the SSC is responsible for a certain sub-session it must make sure the sub-session remains consistent with the session between the CN and MN. This includes changes to a sub-session by the involved LN, where the SSC must make sure the sub-session and session between the MN and CN remains consistent. The exact behaviour necessary is considered implementation specific.

8. Behaviour when receiving INVITE with 'pst-control' in Require header

When a node receives an INVITE message with the option tag 'pst-control' in the Required header according to SIP [RFC3261] it should only process the message further if it supports the specified extension. In case it does not supports this extension it response

with a 420 response (Bad Extension).

In case the receiving node does support the extension it can decline (306 response) or accept (200 response) the proposed partial session. Which one applies depends should depend on what the user wants. This could for example mean user-interaction is necessary before the proposal is accepted or decline.

9. Security Considerations

All security considerations as described in [shacham06] also apply for this proposal. Besides those considerations here some other are described that are related to network initiated partial session transfers.

9.1. Sender of proposal can be any node in the signalling path

The INVITE send to the MN to propose a partial session transfer can virtually be send by any node in the signalling path of a certain session.

9.2. Sudden disconnection of the MN

In case a media-stream has been transferred to a LN and the MN suddenly disconnects, that specific media-stream does not stop working; it only stops when the LN or CN disconnects, or the LN, CN or SSC uses signalling to stop the media-stream. The proposed method does not contain a mechanism to register sudden disconnects, so one of the involved nodes can take appropriate actions.

9.3. MN is informed about the existence of a LN

In case of a network initiated partial session transfer, the MN is informed about the existence of the involved LN at the moment it receives the proposal for the partial session transfer (At (1) in Figure 1. It is the task of the SSC to make sure a LN is only exposed to nodes that are trusted. No mechanism has been defined for this.

9.4. LN is informed about the IP-address of the CN

At the moment the MN accepts a transfer, the involved LN is informed about the IP-address of the CN to inform the AN to send or receive media from the CN (At (9) in Figure 1). The CN might not want other nodes beside the MN to know about its IP-address.

10. IANA Considerations

Section 27 of RFC 3261 [RFC3261] creates an IANA registry for method names, header field names, warning codes, status codes, and option tags. This specification defines two new SIP header fields and a SIP option tag.

10.1. Pst-to and Pst-call-id headers

This document defines two new SIP header fields: Pst-to and Pst-call-id. These header fields needs to be registered by the IANA in the SIP Parameters registry under the Header Field subregistry

10.2. Option Tag

This section defines a new option tag per guidelines in Section 27.1 of RFC 3261 [RFC3261].

Name: pst-control

Description: This option tag is used to identify support for client control of network initiated partial session transfers. When present in a Supported header field, it indicates that an agent supports controlling a network initiated partial session transfer.

11. Acknowledgements

The work described in this Internet-Draft is based on results of IST FP6 Integrated Project DAIDALOS. DAIDALOS receives research funding from the European Community's Sixth Framework Programme. Apart from this, the European Commission has no responsibility for the content of this Internet-Draft. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

12. Normative References

[3GPP 23.228]
3GPP, "TS 23.228: IP Multimedia Subsystem (IMS) (Stage 2), Release 7", December 2005.

[3GPP 23.279]
3GPP, "TS 23.279: Combining Circuit Switched (CS) and IP

Multimedia Subsystems (IMS) services, Release 7", December 2006.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC4566] Handley, M., Jacobsen, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[aartsetuijn06] Aartse Tuijn, J., "Partial session mobility in context aware IP-based multimedia subsystems", december 2006.

[shacham06] Shacham, R., Schulzrinne, H., Thakolsri, S., and W. Kellerer, "Session Initiation Protocol (SIP) Session Mobility", November 2006.

Authors' Addresses

Jasper Aartse Tuijn
University of Twente
Calslaan 1-309
Enschede 7522MH
The Netherlands

Email: j.aartsetuijn@student.utwente.nl

Dennis Bijwaard
Alcatel-Lucent
Capitool 5
Enschede 7521PL
The Netherlands

Email: bijwaard@alcatel-lucent.com

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).

Bibliography

[1] The 3rd generation partnership project. Available from: http://www.3gpp.org.

[2] 4g+ project. Available from: http://www.freeband.nl/kennisimpuls/projecten/4gplus/ENindex.html.

[3] Awareness: Context aware mobile networks and services. Available from: http://awareness.freeband.nl.

[4] Core network protcols (release 7). Technical specification 24008-740. Available from: http://www.3gpp.org/ftp/Specs/archive/24_series/24.008/24008-740.zip.

[5] Designing advanced network interfaces for the delivery and administration of location independent, optimised personal services. Available from: http://ist-daidalos.org.

[6] Ip multimedia subsystem (ims). Technical specification 23228-740. Available from: http://www.3gpp.org/ftp/Specs/archive/23_series/23.228/23228-740.zip.

[7] Jain sip api specification. Available from: http://jcp.org/en/jsr/detail?id=32.

[8] jain-sip-appserver. Available from: https://jain-sip-appserver.dev.java.net.

[9] Java api for sip signalling. Available from: https://jain-sip.dev.java.net.

[10] Jsr 141: Sdp api. Available from: http://jcp.org/en/jsr/detail?id=141.

[11] Multiparty multimedia session control (mmusic). Available from: http://www.ietf.org/html.charters/mmusic-charter.html.

[12] Presence service uring the ip multimedia (im) core network (cn) subsystem (release 7). Technical specification 24141-710. Available from: http://www.3gpp.org/ftp/Specs/archive/24_series/24.141/24141-710.zip.

[13] Session initiated protocol. Available from: http://en.wikipedia.org/wiki/Session_Initiation_Protocol.

[14] Sip communicator - the java voip and instant messaging client. Available from: http://www.sip-communicator.org.

[15] Sip server software framework for sip proxy, sip serverc, etc. Available from: http://www.radvision.com/Products/Developer/SIPServer/.

[16] Initial network architecture design and sub-systems interoperation. 2004.

[17] Service creation platform design specification. 2004.

[18] Ist project summary: Daidalos ii, 2005. Available from: `http://www.ist-daidalos.org/daten/publications/EU-leaflet/EU-project_Daidalos_II_Summary.pdf`.

[19] *The virtual device: expanding wireless communication services through service discovery and session mobility*, volume 4, 2005. Available from: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1512952`.

[20] Work package 1- activity 1.1: Consolidated scenario description. 2005.

[21] J. Ala-Kurikka, E. Harjula, and M. Ylianttila. Pidfconn: Extension to the presence information data format (pidf) for expressing connectivity features. Internet draft, The Internet Society, March 2006. Available from: `http://tools.ietf.org/wg/simple/draft-ala-kurikka-simple-pidfconn-01.txt`.

[22] J. Arkko, V. Torvinen, G. Camarillo, A. Niemi, and T. Haukka. Security mechanism agreement for the session initiated protocol (sip). Rfc, The Internet Society, January 2003. Available from: `http://tools.ietf.org/html/rfc3329`.

[23] G. Camarillo, A. Niemi, M. Isomaki, M. Garcia-Martin, and H. Khartabil. Refering to multiple resources in the session initiation protocol (sip). Internet draft, The Internet Society, June 2006. Available from: `http://tools.ietf.org/wg/sipping/draft-ietf-sipping-multiple-refer/draft-ietf-sipping-multiple-refer-06.txt`.

[24] H. Chen, T. Finin, and A. Joshi. An ontology for context-aware pervasive computing environments. *The Knowledge Engineering Review*, (18):197–207, May 2004. Available from: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1295737`.

[25] D. Crocker and P. Overell. Augmented bnf for syntax specification: Abnf. Rfc, The Internet Society, November 1997. Available from: `http://www.apps.ietf.org/rfc/rfc2234.html`.

[26] A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, February 2001. Available from: `http://portal.acm.org/citation.cfm?id=593572&CFID=295096&CFTOKEN=91441915&qualifier=LU1007183`.

[27] T. Dierks and C. Allen. The tls protocol. Rfc, The Internet Society, January 1999. Available from: `http://www.apps.ietf.org/rfc/rfc2246.html`.

[28] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol – http/1.1. Rfc, The Internet Society, June 1999. Available from: `http://tools.ietf.org/html/rfc2616`.

[29] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart. Http authentication: Basic and digest access authentication. Rfc, The Internet Society, June 1999. Available from: `http://www.apps.ietf.org/rfc/rfc2617.html`.

[30] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, Reading, Massachusetts, 1994.

[31] M. Handley, V. Jacobsen, and C. Perkins. Sdp: Session description protocol. Rfc, The Internet Society, July 2006. Available from: `http://tools.ietf.org/html/rfc4566`.

[32] H. S. J. Rosenberg. An offer/answer model with the session description protocol (sdp). Rfc, The Internet Society, June 2002. Available from: `http://tools.ietf.org/html/rfc3264`.

[33] S. Kent and R. Atkinson. Security architecture for the internet protocol. Rfc, The Internet Society, November 1998. Available from: `http://tools.ietf.org/html/rfc2401`.

[34] D. Komiya, X. Mingqiang, and E. Shim. Use cases for session mobility in multimedia applications. Internet draft, The Internet Society, February 2006. Available from: `http://tools.ietf.org/wg/mmusic/draft-komiya-mmusic-session-mobility-usecases-00.txt`.

[35] M. Lonnfors and K. Kiss. Session initiation protocol (sip) user agent capability extension to presence information data format (pidf). Internet draft, The Internet Society, July 2006. Available from: `http://tools.ietf.org/wg/simple/draft-ietf-simple-prescaps-ext`.

[36] R. Mahy, B. Biggs, and R. Dean. The session initiated protocol (sip) "replaces" header. Rfc, The Internet Society, September 2004. Available from: `http://www.apps.ietf.org/rfc/rfc3891.html`.

[37] M. MANI and N. CRESPI. The tenth ieee international conference on communications systems (ieee iccs 2006). In *Session Mobility between Heterogeneous Accesses with the Existence of IMS as the Service Control Overlay*, November 2006. Available from: `www.int-evry.fr/rs2m/cgi-bin/publi/Biblio?A=DL&Id=2006P031`.

[38] C.-J. Peng. Ssip: Split a sip session over multiple devices. Master's thesis, Institue of Communication Engineering, National Chung Cheng University, Taiwan, Republic of China, June 2005. Available from: `http://140.123.21.32/ETD-db/ETD-search/view_etd?URN=etd-0727105-020000`.

[39] J. Peterson. A presence-based geopriv location object format. Rfc, The Internet Society, December 2005. Available from: `http://tools.ietf.org/html/rfc4119`.

[40] B. Ramsdell. S/mime version 3 message specification. Rfc, The Internet Society, June 1999. Available from: `http://tools.ietf.org/html/rfc2633`.

[41] A. Roach. Session initiated protocol (sip)-specific event notification. Rfc, The Internet Society, June 2002. Available from: `http://www.apps.ietf.org/rfc/rfc3265.html`.

[42] J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo. Best current practices for third party call controll (3pcc) in the session initiated protocol (sip). Rfc, The Internet Society, April 2004. Available from: `http://www.apps.ietf.org/rfc/rfc3725.html`.

[43] J. Rosenberg and H. Schulzrinne. Session initiated protocol (sip): Locating sip servers. Rfc, The Internet Society, June 2002. Available from: `http://www.apps.ietf.org/rfc/rfc3263.html`.

[44] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. Sip: Session initiated protocol. Rfc, The Internet Society, June 2002. Available from: `http://www.apps.ietf.org/rfc/rfc3261.html`.

[45] R. Shacham, H. Schulzrinne, S. Thakolsri, and W. Kellerer. Session initiated protocol (sip) session mobility. Internet draft, The Internet Society, February 2006. Available from: `http://tools.ietf.org/wg/sipping/draft-shacham-sipping-session-mobility-02.txt`.

[46] R. Sparks. The session initiation protocol (sip) refer method. Rfc, The Internet Society, April 2003. Available from: `http://www.apps.ietf.org/rfc/rfc3515.html`.

[47] H. Sugano, S. Fujimoto, G. Klyne, A. Bateman, W. Car, and J. Peterson. Presence information data format (pidf). Rfc, The Internet Society, August 2004. Available from: `http://tools.ietf.org/html/rfc3863`.

[48] Q. Wang and M. Abu-Rgheff. Next-generation mobility support. *Communications Engineer*, 1(1):16–19, February 2003. Available from: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1196353`.

[49] X. Wang, D. Zhang, T. Gu, and H. Pung. Ontology based context modeling and reasoning using owl. In *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communiations Workshop*, pages 18–22, March 2004. Available from: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1276898`.

[50] S. A. Xynogalase, M. K. Chantzara, I. C. Sygkouna, S. P. Vrontis, I. G. Roussaki, and M. E. Anagnostou. Context management for the provision of adaptive services to roaming users. *Wireless Communications, IEEE*, 11(2):40–47, April 2004. Available from: `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1295737`.