

Information and Communication Technologies (ICT) Programme

Project N°: FP7-ICT- 214063

**SEA**



---

Deliverable D4.1

## ***D4.1: Cross layer control, adaptation modelling and metadata specification***

---

**Author(s):** Th. Zahariadis, G. Leoleis (Synelixis), O. Negru (TGV), Th. Schierl, K. Grüneberg (Fraunhofer HHI), M. Naumann (Nomor), L. Celetto, E. Quacchio (STM), Costis Contopoulos (VOD)

**Status -Version:** Final

**Date:** 30 September 2008

**Distribution - Confidentiality:** Public

**Code:** SEA\_D4.1\_SYN\_FF\_20080930.doc

### **Abstract:**

This deliverable focuses on the second SEA key content delivery pillar, namely the multi-source/multi-network streaming & adaptation. In more details, SEA aims to offer on-the fly content adaptation, inherited resiliency and enriched PQoS by dynamically combining different content layers, views and representations of the same resource transmitted from multiple sources (different servers or peers) and/or received over multiple diverse paths or networks. Cross-network adaptation and cross-layer optimization will offer traffic adaptation and optimal use the available resources (bandwidth).



## Disclaimer

This document contains material, which is the copyright of certain SEA contractors, and may not be reproduced or copied without permission. All SEA consortium partners have agreed to the full publication of this document. The commercial use of any information contained in this document may require a license from the proprietor of that information.

The SEA Consortium consists of the following companies:

No	Participant name	Participant short name	Country	Country
1	ST Microelectronics	STM	Co-ordinator	Italy
2	Synelixis	Synelixis	Contractor	Greece
3	Thomson Grass Valley	Thomson	Contractor	France
4	Philips Consumer Lifestyle B.V.	Philips	Contractor	Netherlands
5	Vodafone Panafon AEET	Vodafone	Contractor	Greece
6	Nomor Research	Nomor	Contractor	Germany
7	Fraunhofer HHI	HHI	Contractor	Germany
8	Politecnico di Torino	Polito	Contractor	Italy
9	Universidad Politécnica de Madrid	UPM	Contractor	Spain
10	University of California, Los Angeles	UCLA	Contractor	USA

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.



This page has been intentionally left blank.



## Table of contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>8</b>
1.1	SEA ABSTRACT NETWORK ARCHITECTURE.....	8
1.2	ADAPTATION SCENARIOS .....	11
1.2.1	<i>Scenario 1: Adaptation of SVC stream .....</i>	<i>12</i>
1.2.2	<i>Scenario 2: Adaptation of MVC stream.....</i>	<i>12</i>
1.2.3	<i>Scenario 3: Adaptation of MDC/SVC base-layer stream.....</i>	<i>12</i>
1.2.4	<i>Scenario 4: Adaptation of SVC stream over P2P network.....</i>	<i>12</i>
1.2.5	<i>Scenario 5: Adaptation of MVC stream over P2P network .....</i>	<i>13</i>
1.2.6	<i>Scenario 6: Adaptation of MDC stream over P2P network.....</i>	<i>13</i>
1.2.7	<i>Scenario 7: Adaptation of SVC stream with hierarchical distinction of P2P chunks .....</i>	<i>13</i>
1.2.8	<i>Scenario 8: Adaptation of MDC/SVC base &amp; enhanced-layers.....</i>	<i>13</i>
1.2.9	<i>Scenario 9: Adaptation of MDC stream at the terminal .....</i>	<i>14</i>
<b>2</b>	<b>CROSS-LAYER CONTROL &amp; ADAPTATION.....</b>	<b>15</b>
2.1	MPEG-21 DIGITAL ITEM ADAPTATION (DIA) .....	16
2.1.1	<i>MPEG21-DIA tools.....</i>	<i>16</i>
2.1.2	<i>Drawbacks of a pure MPEG21-DIA implementation .....</i>	<i>18</i>
2.2	SIGNALLING CONTENT ATTRIBUTES BY SDP .....	18
<b>3</b>	<b>SEA CROSS-LAYER ADAPTATION ARCHITECTURE.....</b>	<b>22</b>
3.1	SEA CROSS-LAYER MODULES .....	22
3.2	ADAPTATION ENGINE ARCHITECTURE .....	23
3.3	INITIALIZATION AND ADAPTATION OF STREAMING SESSIONS .....	24
<b>4</b>	<b>ADAPTATION DECISION MODULE (ADM).....</b>	<b>28</b>
4.1	MODULE FUNCTIONALITY .....	28
4.1.1	<i>Stream Initialisation.....</i>	<i>28</i>
4.1.2	<i>Stream Adaptation.....</i>	<i>30</i>
4.1.3	<i>End-to-end Content Adaptation Mechanism.....</i>	<i>36</i>
4.2	SOFTWARE ARCHITECTURE.....	37
4.3	INTERFACE – FUNCTIONS DEFINITION .....	38
<b>5</b>	<b>ADAPTATION EXECUTION MODULE (AEM).....</b>	<b>41</b>
5.1	MODULE FUNCTIONALITY .....	41
5.2	SOFTWARE ARCHITECTURE.....	44
5.2.1	<i>Description of SW architecture.....</i>	<i>45</i>
5.2.2	<i>VLC SW architecture mapping.....</i>	<i>46</i>
5.2.3	<i>Scenario mapping .....</i>	<i>47</i>
5.3	INTERFACE – FUNCTIONS DEFINITION .....	48
<b>6</b>	<b>NETWORK AWARENESS MODULE (NAM).....</b>	<b>51</b>
6.1	MODULE FUNCTIONALITY .....	51
6.2	SOFTWARE ARCHITECTURE.....	52
6.3	INTERFACE – FUNCTIONS DEFINITION .....	52
6.3.1	<i>Interface from the NAM to the ADM.....</i>	<i>53</i>
6.3.2	<i>Interface from the ADM to NAM.....</i>	<i>53</i>
<b>7</b>	<b>TERMINAL AWARENESS MODULE (TAM).....</b>	<b>54</b>
7.1	MODULE FUNCTIONALITY .....	54
7.1.1	<i>Plurality of terminals .....</i>	<i>54</i>
7.1.2	<i>List of parameters .....</i>	<i>54</i>
7.1.3	<i>Logical grouping.....</i>	<i>54</i>
7.1.4	<i>Aggregating into usable concepts .....</i>	<i>55</i>



7.2 SOFTWARE ARCHITECTURE FOR THE TAM IN THE HOME ENVIRONMENT ..... 56

7.3 INTERFACE – FUNCTIONS DEFINITION ..... 56

**8 CONTENT STORAGE MODULE & P2P AWARENESS MODULE ..... 57**

8.1 MODULE FUNCTIONALITY ..... 57

8.2 SOFTWARE ARCHITECTURE ..... 57

8.3 INTERFACE – FUNCTIONS DEFINITION ..... 59

**9 CONCLUSIONS ..... 60**

**10 REFERENCES..... 61**



## Abbreviations

<b>AMBR</b>	<i>Aggregate Maximum Bit Rate</i>
<b>AMC</b>	<i>Adaptive Modulation and Coding</i>
<b>AP</b>	<i>Access Point</i>
<b>ARQ</b>	<i>Automatic Repeat Request</i>
<b>BER</b>	<i>Bit Error Rate</i>
<b>BSD</b>	<i>Bitstream Syntax Description</i>
<b>CLC</b>	<i>Cross Layer Control</i>
<b>DIA</b>	<i>Digital Item Adaptation</i>
<b>DID</b>	<i>Digital Item Declaration</i>
<b>DIM</b>	<i>Digital Item Methods</i>
<b>DIME</b>	<i>DIM Engine</i>
<b>DIML</b>	<i>DIM Language</i>
<b>DIP</b>	<i>Digital Item Processing</i>
<b>DMP</b>	<i>Digital Media Project</i>
<b>DVB-x</b>	<i>Digital Video Broadcasting – x (where x = Cable, Satellite, Terrestrial, ...)</i>
<b>GOP</b>	<i>Group of Pictures</i>
<b>HAN</b>	<i>Home Area Network</i>
<b>ICT</b>	<i>Information and Communications Technologies</i>
<b>IETF</b>	<i>Internet Engineering Task Force</i>
<b>IMS</b>	<i>IP-based Multimedia sub-Systems</i>
<b>IPMP</b>	<i>Intellectual Property Management &amp; Protection</i>
<b>IPTV</b>	<i>IP Television</i>
<b>ISMA</b>	<i>Internet Streaming Media Alliance</i>
<b>ISP</b>	<i>Internet Service Provider</i>
<b>MANE</b>	<i>Media Aware Network Element</i>
<b>MDC</b>	<i>Multi Description Coding</i>
<b>MIMO</b>	<i>Multiple Input Multiple Output</i>
<b>MME</b>	<i>Mobility Management Entity</i>
<b>MMUSIC</b>	<i>Multiparty Multimedia Session Control</i>
<b>MOS</b>	<i>Mean Opinion Score</i>
<b>MPEG</b>	<i>Moving Picture Experts Group</i>
<b>MVC</b>	<i>Multi-view Video Coding</i>
<b>NAL</b>	<i>Network Abstraction Layer</i>
<b>NAM</b>	<i>Network Awareness Modules</i>
<b>P2P</b>	<i>Peer-to-Peer</i>
<b>PDA</b>	<i>Personal Digital Assistant</i>
<b>PQoS</b>	<i>Perceived Quality of Service</i>



<b>QoS</b>	<i>Quality of Service</i>
<b>RAN</b>	<i>Radio Access Networks</i>
<b>RAT</b>	<i>Radio Access Technology</i>
<b>RTCP</b>	<i>RTP Control Protocol</i>
<b>RTP</b>	<i>Real Time Protocol</i>
<b>RTSP</b>	<i>Real Time Streaming Protocol</i>
<b>RTT</b>	<i>Round Trip Time</i>
<b>SAE</b>	<i>System Architecture Evolution</i>
<b>SAP</b>	<i>Session Announcement Protocol</i>
<b>SDP</b>	<i>Session Description Protocol</i>
<b>SEA</b>	<i>SEAmless Content Delivery</i>
<b>sHMG</b>	<i>SEA Home Media Gateway</i>
<b>SIP</b>	<i>Session Initiation Protocol</i>
<b>sNMG</b>	<i>SEA Network Media Gateway</i>
<b>SNR</b>	<i>Signal-to-Noise Ratio</i>
<b>SVC</b>	<i>Scalable Video Coding</i>
<b>TAM</b>	<i>Terminal Awareness Module</i>
<b>TCP</b>	<i>Transmission Control Protocol</i>
<b>UED</b>	<i>Usage Environment Description</i>
<b>UCD</b>	<i>Usage Constraints Description</i>
<b>VLC</b>	<i>VideoLanClient</i>
<b>XSLT</b>	<i>Extensible Stylesheet Language Transformation</i>



# 1 Introduction

Towards the Future Internet age, SEA (SEAMless Content Delivery) aims to offer a new experience of seamless video delivery, maintaining the integrity and wherever applicable, adapting and enriching the quality of the media across the whole distribution chain. In more details, SEA aims to introduce novel services and new business models, by innovating over three key-content delivery pillars:

- a) Multi-layered/Multi-viewed content coding,
- b) Multi-source/multi-network streaming & adaptation and
- c) Content Protection and lightweight asset management.

This deliverable focuses on the second key content delivery pillar, namely the multi-source/multi-network streaming & adaptation. In more details, SEA will offer on-the fly content adaptation, inherited resiliency and enriched PQoS by dynamically combining different content *layers*, *views* and *representations* of the same resource (video stream) transmitted from multiple sources (different servers or peers in case of P2P streaming) and/or received over multiple diverse paths or networks. Reconstruction of the content segments may take place either within the network (offering transparent streaming to low-end terminals) or at the end-user terminal in case multi-network connectivity is available. Cross-network adaptation and cross-layer optimization will offer traffic adaptation (load balancing to avoid network flooding) and optimal use the available resources (bandwidth).

The cross layer control and content adaptation needs input parameter from the radio network, the terminal characteristics and feedback about the perceived QoS from the user. The SEA Network Awareness Modules (NAM) and the Terminal Awareness Module (TAM) will provide qualified feedback on a per user base for system parameters, which can help to adapt and distribute the content in the best possible way. Parameters that have been considered include: current throughput, max throughput (if limited by subscription), packet delay, packet delay over time, jitter, round trip time, etc.

Moreover, this deliverable specifies the format and structure of the cross-layer metadata. Within SEA, we have selected the MPEG-21 framework for cross-layer metadata exchange, adapted to the SEA needs and architecture, while an SDP solution is selected for lower-end terminals. By means of the MPEG-21 DIA metadata, the sHMG/sNMG gather information about the characteristics of the connected terminals, producing a decision about the best compromise between quality and bandwidth reserved to each device. Terminals describe the contents of the bit streams, allowing an easier adaptation of the bit stream to a specific scalability point at the sHMG side.

Cross-layer control and adaptation is an important part of SEA, which aims to provide the means to distribute A/V user-centric services with superior quality and striking, flexibility, improving citizens quality of life, entertainment and safety.

## 1.1 SEA Abstract Network Architecture

A general overview of an integrated network environment from all the network technologies described in [1][2] is summarized in Figure 1. This depicts the overall environment that SEA intends to utilize and provide solutions for cross-layer control and adaptation, content protection and peer-to-peer communications. In case (a), we assume an integrated service/content provider-driven business model, reflected to the network architecture. The service/content provider is located at the edge of the core network and is using a client/server's paradigm-based distribution chain. It offers streaming A/V services over broadcasting, interactive/on-demand networks (including the Internet) and 2G+/3G/4G mobile networks. It is worth to note the great heterogeneity in the network segments characteristics of



this approach. On one hand, the broadcasting networks (DVB-S/S2, DVB-T, DVB-C, DVB-H, etc.) have specific channels/frequencies allocated per content stream, thus a certain quality of the transmitted content is guaranteed. However, the user interaction is very limited, thus personalisation and value-added services are also quite limited. State-of the art bi-directional broadband wireless (WiMax) or wireline (xDSL, Cable, FTTx) networks offer adequate bandwidth in the (extended) home environment and enable feedback and user control, therefore they allow for interactive and on-demand services. Mobile networks (3G/4G) offer dedicated channel (and QoS) per terminal/service; thus a certain PQoS can be guaranteed.

Figure 1b depicts the case, where the end-users, apart from being content/service consumers, they can as well contribute as content providers. This has effectively become a possibility through the availability of sufficient uplink capacity that most today’s access technologies typically offer.

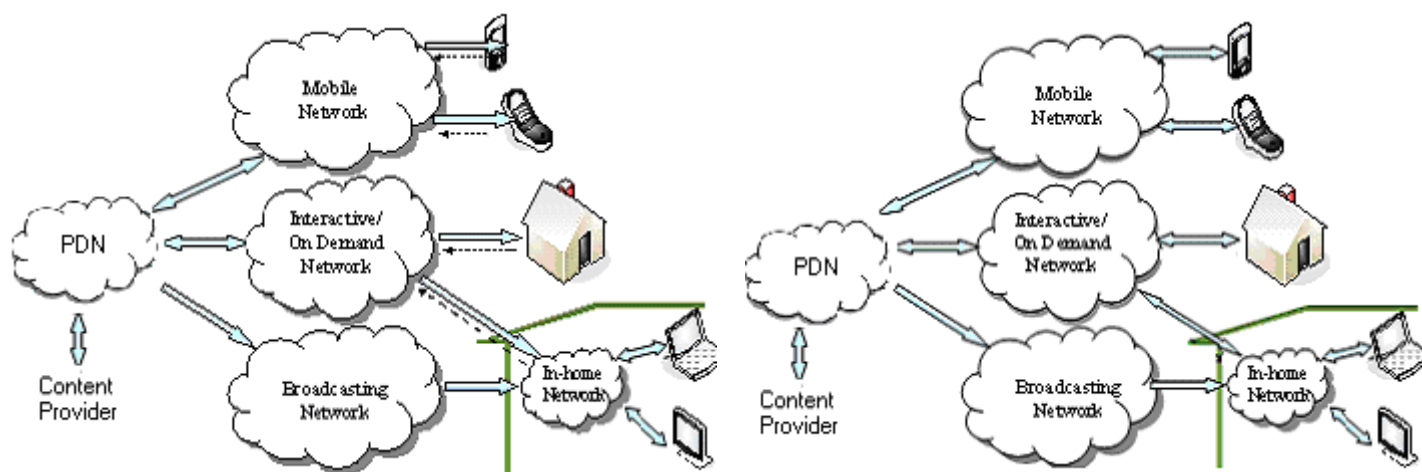


Figure 1: Logical Network Architecture a) Broadband downlinks, b) Broadband bidirectional links

Moreover, the advanced coding schemes will facilitate video distribution with enriched QoS, especially in case of high-end multi-modal terminals able to receive and reconstruct multiple video streams segments (i.e. layers, views, descriptions).

However, home terminals or common, less-expensive low-end mobile terminals may be only capable of decoding at a particular bit-rate or may be only feasible to correctly display up to a particular image resolution. Thus, in order to meet all SEA innovative features, the media delivery service architecture should be content aware and have knowledge of the access technologies as well as of the utilized end-user device capabilities and characteristics. The network has to provide the relative adaptation functionalities, in order to seamlessly support the majority of terminals. On the other hand, particular access technologies (e.g. 3G networks) can support services up to a particular bit-rate and with certain QoS, while in P2P networks the end-to-end path may be unknown or time variant.

In order to build a service architecture upon the described variety of access networks, it is desirable to have as much information and adaptation at the lower layers (up to the network layer) as possible, along with scalability functionality coming with the media codec. Certain functions such as content caching in the network, content adaptation and cross-layer optimization would certainly need knowledge of the network conditions/characteristics. In order to overcome this problem, wherever applicable in the SEA architecture, we introduce intelligent media/ network aware nodes.

As shown in Figure 2, we introduce in the SEA architecture two content aware edge Media Aware Network Element (MANE):

- a) a seamless Home Media Gateway (sHMG), located at the edge of the extended home environment and
- b) a seamless Network Media Gateway (sNMG) at the edge of the access networks, e.g. the 3GPP Service Architecture Evolution (SAE)

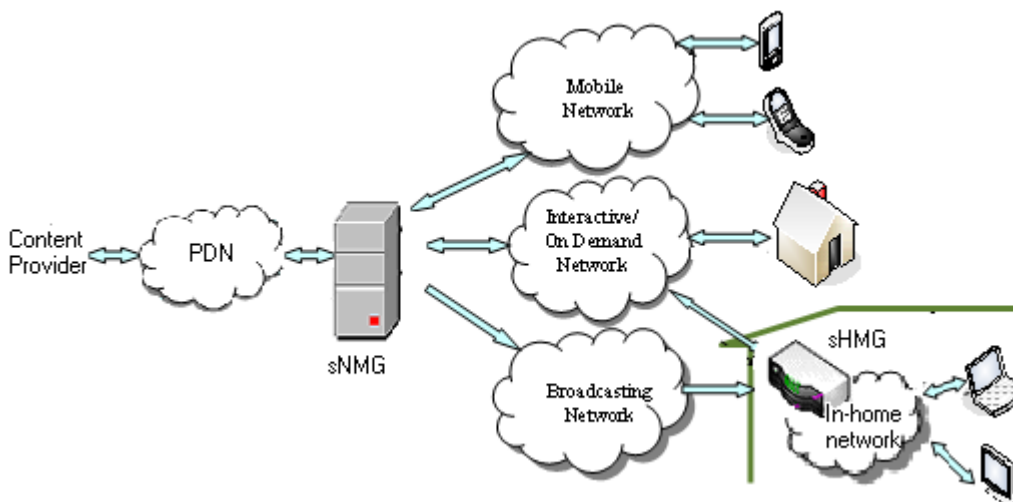


Figure 2: SEA abstract network architecture

The SEA content aware MANE edge nodes (sHMG and sNMG) are network-based components for SEA architecture and support the intelligent, seamless content distribution. They offer functions like network and terminal awareness, content enrichment and content protection. In the longer term, they may be integrated with IP-based Multimedia sub-Systems (IMS) as define by ETSI TISPAN. These MANE nodes can offer multimedia storage, dynamic content adaptation and enriched PQoS by dynamically combining multiple multimedia content layers from various sources. Moreover, as they have knowledge of the underlined networks, this information on the network conditions/ characteristics can be provided to and utilized by the Cross Layer Control mechanism and adapt the multimedia streams to the next network in the delivery path. This will be extremely important in case of a low bandwidth but guaranteed QoS mobile networks and in the broadband but best effort P2P topologies.

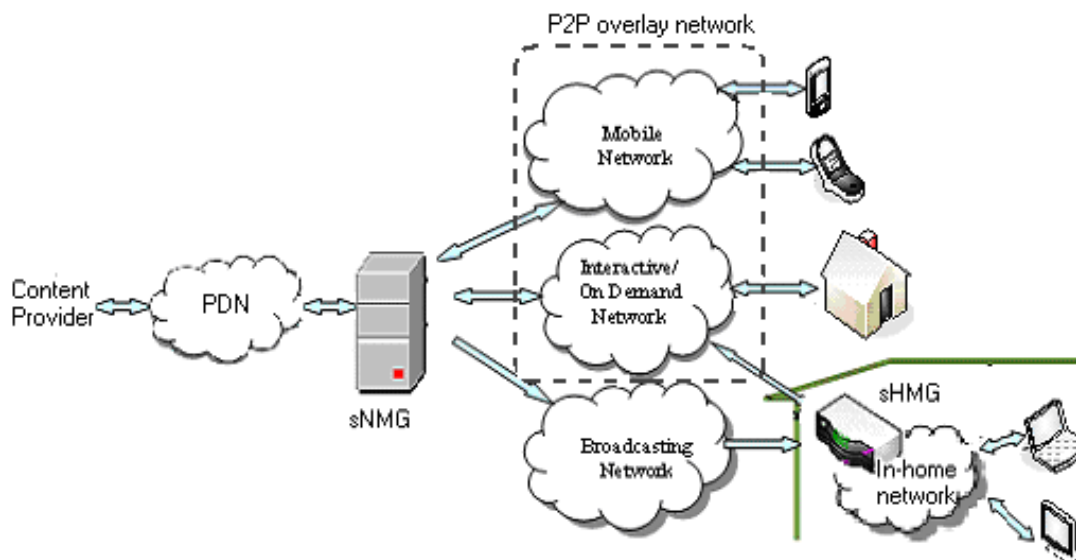


Figure 3: The SEA concept of P2P overlay architecture

Introducing the two MANE’s network nodes at the edges of the networks also enables SEA to realize a Peer-to-Peer (P2P) overlay topology as shown in Figure 3. In this case, services may be offered not only by centrally located media streaming servers, but by groups of end-user devices and the sHMG acting as distributed content repositories. Given content protection and management is in place, network operators and service providers may offer value-added streaming services with remarkable



PQoS. Moreover, individuals may produce their own (real-time) content and make it publicly available to a larger audience, without having to rely on a specific, expensive networking infrastructure. In this environment, video streaming scalability, resilience and PQoS may be increased, as multiple sources may stream video segments, enriching the content on the fly, either at the network and/or at the end-user terminal.

The sHMG and the sNMG are key components in the foreseen SEA network architecture and play an important role in the cross layer control and adaptation. Both have been analysed in deliverable “D2.3 - Cross Layer Interfaces Specification”. Though their processing capabilities differ severely, in this document we consider both as Adaptation Engines (AE), which play a similar role in the SEA adaptation process..

## 1.2 Adaptation Scenarios

Taking into account the SEA architecture, the SEA network nodes and the final terminal capabilities (ranging from laptops to mobile phones), we may foresee a number of adaptation scenarios. In order to optimize adaptation and increase the number of available scenarios, we adopt the general SEA adaptation network architecture as shown in Figure 4. In this view, we assume that in the path from the Content Provider to the terminal, we may have N+1 Adaptation Engines (AE). Each engine is responsible for adapting the video stream to the next network in the path i.e.  $AE_i$  adapts the video stream to the characteristics/capabilities of Network  $i$ , always taking into account the final terminal capabilities and user requirements.

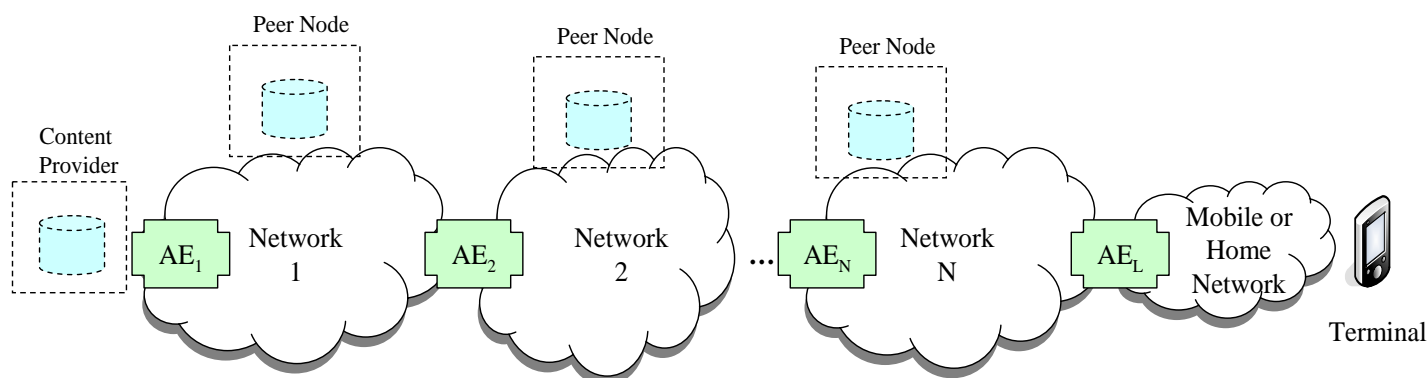


Figure 4: SEA Adaptation network architecture

As the adaptation options may be limited, some adaptation engines may perform stream adaptation, or some of them may just forward (relay) network, streaming, terminal or user characteristics to the next AE along the connection path. It is important to note however, that the last adaptation engine will also have the responsibility to terminate the adaptation in case the terminal is not able to handle it. For example in case of P2P streaming, the terminal may not be able to handle the required extended buffering and streaming reconstruction, and this functionality may be handled by the *Last Node AE* ( $AE_L$ ). For simplicity reasons, we’ll assume that the  $AE_L$  will always terminate the adaptation process, reconstruct the streamed video according to the terminal needs, and then stream the re-constructed video to the terminal<sup>1</sup>. Moreover, the  $AE_L$  will be responsible for streaming A/V content optimised for the end-user’s terminal and access connection. Taking into account the above architecture we may summarize a number of scenarios, as follows.

<sup>1</sup> In case the terminal is able to handle the adaptation process itself, we can assume that the  $AE_L$  is collocated at the terminal.



### **1.2.1 Scenario 1: Adaptation of SVC stream**

In this scenario we assume that the video in SVC format is streamed from to Video Server to the Terminal. The SVC stream description (metadata) is forwarded in parallel, thus each AE will have knowledge of the stream characteristics.

In case a network (e.g.  $N_x$ ) in the path is not able to handle all SVC layers (both the base and the enhanced layers), the relevant AE ( $AE_x$ ) may decide to drop a number of the enhanced layers. Moreover, if the terminal is not able to handle SVC reconstruction or the user preferences/permissions call for only SVC base layer streaming, the  $AE_L$  will implement the adaptation function. In case of a VoD scenario, this adaptation may be managed at AE closer to the video server in order to avoid consuming  $N_1-N_L$  network resources.

If the operation environment changes e.g. the network conditions are improved or the terminal is now connected via a broadband network, the adaptation will stop, enhanced layers will also be streamed and the quality of the video will increase.

### **1.2.2 Scenario 2: Adaptation of MVC stream**

In this scenario we assume that the video in MVC format is streamed from to Video Server to the Terminal. The MVC stream description (metadata) is forwarded in parallel, thus each AE will have knowledge of the stream characteristics (number of views included in the stream).

In case a network (e.g.  $N_x$ ) in the path is not able to handle all views, the relevant AE ( $AE_x$ ) may decide to drop the additional views and keep only the ones selected by the user. Moreover, if the terminal is not able to handle MVC reconstruction or the user preferences/permissions call for only one view, the  $AE_L$  will implement the adaptation function. In case of a VoD scenario, this adaptation may be manipulated at AE closer to the video server in order to avoid consuming  $N_1-N_L$  network resources.

If the operation environment changes e.g. the network conditions are improved or the terminal is now connected via a broadband network, the adaptation will stop, all views will be transfer to the terminal and the user will be able to select his favorite view (or more than one view in parallel).

### **1.2.3 Scenario 3: Adaptation of MDC/SVC base-layer stream**

In this scenario we assume that the video is H.264 SVC base-layer only (i.e. equivalent to AVC) encoded in MDC format. The MDC stream description (metadata) is forwarded in parallel, thus each AE will have knowledge of the stream characteristics.

In case the terminal is not able to handle MDC reconstruction, the  $AE_L$  will implement the adaptation function. Contrary to the previous scenarios, the MDC-adapted layer will be propagated to an adaptation engine closer to the terminal (e.g.  $AE_L$ ), since MDC is intended to increase the perceived quality of the received video. After MDC reconstruction, the  $AE_L$  will have the responsibility to optimally stream the video to the terminal.

In case the terminal is capable of handling the MDC reconstruction, we can assume that the  $AE_L$  is collocated at the terminal.

### **1.2.4 Scenario 4: Adaptation of SVC stream over P2P network**

In this scenario, we assume that the video in SVC format is streamed from to Video Server to the Terminal utilizing the P2P technology as a network overlay. In this scenario, we assume that the P2P mechanisms are not aware of the actual H.264 SVC format (i.e. peers are indifferent to the distinction between base and enhanced layers).

In case the terminal is not able to handle P2P and SVC reconstruction, the  $AE_L$  will implement the adaptation function. The  $AE_L$  will receive the P2P video chunks, reconstruct them in order to recreate the SVC stream and based on the last network characteristics, the terminal requirements and the user



preferences/permissions may additionally adapt the SVC stream (i.e. drop enhanced layers) before it streams it to the terminal.

### **1.2.5 Scenario 5: Adaptation of MVC stream over P2P network**

In this scenario, we assume that the video in MVC format is streamed from to Video Server to the Terminal utilizing the P2P technology as network overlay. In this scenario, we assume that the P2P mechanisms are not aware of the actual H.264 MVC format (different views of the same scene).

In case the terminal is not able to handle P2P and MVC reconstruction/view selection, the  $AE_L$  will implement the adaptation function. The  $AE_L$  will receive the P2P video chunks, reconstruct them in order to recreate the MVC stream and based on the last network characteristics, the terminal requirements and the user preferences/permissions may additionally adapt the MVC stream, i.e. stream all views or just the ones which were selected by the user.

### **1.2.6 Scenario 6: Adaptation of MDC stream over P2P network**

In this scenario, we assume that the base layer of SVC video (AVC) is coded in MDC format, and it is streamed from to Video Server to the Terminal utilizing the P2P technology as a network overlay. In this scenario, we assume that the P2P mechanisms are not aware of the actual H.264 SVC base layer format.

In case the terminal is not able to handle P2P and MDC reconstruction/view selection, the  $AE_L$  will implement the adaptation function. The  $AE_L$  will receive the P2P video chunks, reconstruct them in order to recreate the MDC stream, extract from that the SVC base layer and stream it to the user terminal.

### **1.2.7 Scenario 7: Adaptation of SVC stream with hierarchical distinction of P2P chunks**

In this scenario, we assume that the video in SVC format is streamed from to Video Server to the Terminal utilizing the P2P network technology. In this scenario, we assume that the P2P mechanisms are aware of the actual H.264 SVC format and are able to perform separation between base and enhanced layers.

In case the terminal is not able to handle P2P and SVC reconstruction, the  $AE_L$  will implement the adaptation function taking advantage of the P2P network capability. In this scenario, the P2P module located at the  $AE_L$  will receive the P2P video chunks, giving different priorities to the chunks that carry the SVC base layer as compared to the chunks that carry the enhanced layers. In this way, the P2P network may guarantee that the base layer has higher priority over the enhanced layers and in case that only a subset of the chunks is received in time, the P2P network will ensure that these chunks belong to the SVC base layer. The  $AE_L$  will then reconstruct the SVC stream and based on the last network characteristics, the terminal requirements and the user preferences/permissions may additionally adapt the SVC stream (i.e. drop enhanced layers) before it streams it to the terminal.

### **1.2.8 Scenario 8: Adaptation of MDC/SVC base & enhanced-layers**

In this scenario we assume that the H.264 SVC base-layer and enhanced layers of the video are streamed, but for enhanced reliability and robustness the SVC base-layer is encoded in MDC format. The SVC and the MDC stream description (metadata) is forwarded in parallel, thus each AE will have knowledge of the stream characteristics.

In case the terminal is not able to handle the SVC and the MDC reconstruction, the  $AE_L$  will implement the adaptation function taking advantage of the MDC capability. In this scenario, the MDC module located at the  $AE_L$  will receive the MDC descriptions and reconstruct the SVC base layer, while the enhanced layer will be received without MDC encoding. In this way, the SEA network may guarantee that the base layer streaming will be more robust and in case that the number of packets in error is too high, at least the base layer will be most probably received unimpaired. This



adaptation is not propagated at AE closer to the video server, as the MDC will increase the quality of the received video (PQoS). After MDC reconstruction and based on the last network characteristics, the terminal requirements and the user preferences/permissions, the  $AE_L$  may additionally adapt the SVC stream (i.e. drop enhanced layers) before it stream it to the terminal.

### 1.2.9 Scenario 9: Adaptation of MDC stream at the terminal

In this scenario we assume that the terminal has multimodal capabilities and is able to reconstruct MDC signals. The video is streamed to the  $AE_L$  following any of the 1-8 scenarios; the  $AE_L$  will receive and reconstruct the video stream, adapt it according to the last network characteristics, the terminal requirements and/or the user preferences/permissions (i.e. drop enhanced layers or select a specific view) and then encode/transcode it in MDC format before it sends it to the terminal. On the other hand, the terminal will receive the MDC encoded stream and reconstruct the video.

Adaptation Scenarios	1	2	3	4	5	6	7	8	9
SVC Base Layer (AVC)	X		X	X		X	X	X	
SVC Enhanced Layers	X			X			X	X	
MVC		X			X				
MDC @ $AE_L$			X			X		X	X
MDC @ Terminal									X
P2P (layers agnostic)				X	X	X			
P2P (w/hierarchical chunks)							X	X	

**Table 1: SEA adaptation scenarios**

The adaptation scenarios are summarized in Table 1. In scenarios 1-8, we may assume that if the terminal is capable of handling the adaptation/reconstruction/buffering functions, we can assume that the  $AE_L$  is collocated at the terminal. In adaptation scenario 9, we highlight the MDC to MDC adaptation. Yet, other options may also be available. Moreover, in SEA we will mainly focus on scenarios 1-6. Scenarios 7-9, due to the complexity that they introduce to the network nodes, will not be implemented/demonstrated during the lifetime of the current project; however, they will be further studied and analyzed, at a theoretical level.



## 2 Cross-layer Control & Adaptation

Adaptation techniques within a single layer have been proven to have little effect. Such local adaptations are deficient in providing global optimal setting for the system. In contrast, cross-layer approach has been extensively discussed in recent research literature for its viability for providing better performance than traditional layered architecture. Deliverable D2.3 [3] provides a detailed description of various cross layer techniques available in the literature. Cross-layer approach increases interaction among different layers to exploit the inherent characteristics of underlying network to maximize the utility (e.g. QoS) and reduce the cost (e.g., battery life, bandwidth). The involvement of multiple layers in cross-layer adaptation is important otherwise various mechanisms available at different layer are likely to counteract each other's effect.

Layered architecture divides the responsibilities among different layers. Hence each layer has a set of distinct mechanisms and associated parameters to fulfil its functionality. Different cross-layer proposals exist that enhance performance, increase throughput and reduce power consumption by adapting mechanisms and using parameters at different layers [3].

<b>Layers</b>	<b><i>Mechanisms (to optimize)</i></b>	<b><i>Parameters</i></b>
User	User priority selection	Terminal characteristics, objective quality metrics (like PSNR, VQM, SSIM), Subjective quality metrics (like MOS, SAMVIQ and DSIS)
Application	Transrating, Transcoding, Forward Error Correction (FEC), Unequal FEC, Automatic Repeat Request (ARQ), Adaptive and scalable encoding/decoding	Rate, Codec, Protection level
Transport	TCP Congestion Control, UDP, Header Compression	Packet loss information, receiver window, congestion window, retransmission timer
Network	Packetization, DiffServ, TE	IP packet size, DiffServ Code Point, Handoff information
Data Link	MAC Protocols, Radio resource control, FEC, ARQ, Framing	Retransmission attempts, Error rate, retry limit, RTS/CTS, Handoff, Traffic classes, TDMA time slots, OFDM carriers
Physical	Channel modulation and coding	BER, signal strength, transmission power, capability profile
Context information	Dynamic Voltage Scaling, Soft real time scheduling	Battery status, Architectural capability profile

**Table 2: Mechanisms and parameters at different layers**

Table 2 classifies useful parameters and adaptation mechanisms by their respective layers, user level, architecture and OS level. Parameters presented in this table include both tunable and read-only parameters.

In this chapter we highlight two options that will be used in SEA. The first one is the MPEG-21 Digital Item Adaptation, a well proven solution, which however is sometimes quite complicated and generates a lot of overhead. In parallel, for the lightweight terminals, we propose a solution based on utilisation of the SDP protocol



## 2.1 MPEG-21 Digital Item Adaptation (DIA)

MPEG21 is an XML based standard from the Moving Picture Experts Group that aims to define an open framework for multimedia applications, providing solutions for management of contents, delivery of contents based on consumer and device capabilities, protection of rights, protection from unauthorized access/modification, protection of privacy of producers and consumers etc.

MPEG21 is an open framework and is based on two essential concepts: the definition of a fundamental unit of distribution and transaction (the *Digital Item*) and the concept of *User* interacting with *Digital Items*. The *User* is any entity that interacts within the MPEG21 framework and makes use of a Digital Item. *Users* may include individuals, consumers, communities, organizations, governments or any other initiatives around the world. A *Digital Item* is a structured digital object with a standard representation and metadata; it aggregates in other words multimedia resources together with metadata, licenses, identifiers, intellectual properties management and protection (IPMP) information etc. *Digital Items* are the fundamental units of the interaction among *Users* within the MPEG21 multimedia framework.

The MPEG-21 standard currently comprises 18 parts which can be clustered into six major categories each one dealing with different aspects of the Digital Items. Among these **MPEG21 Digital Items Adaptation (DIA)**[9] aims to provide the tools for managing audio-visual contents and services in today's diversity of networks, devices and user preferences.

Three are the keys factor that have driven the design of MPEG21-DIA:

- The growth of digital multimedia contents available and the diversity of the media formats in which these contents are represented
- The increasing number of devices for accessing and interacting with multimedia content, from PCs and Set-Top-Boxes (STBs) to portable devices like PDAs and mobile phones. Each device has its own constraints in term of display size, color capabilities, processing power, memory resource and power supply. Additionally, these access devices are used in different locations and environments.
- A wide spectrum of networks available for the transmission of multimedia contents has emerged, enabling users to access the web and multimedia contents from different locations, in different contexts (mobility for example), and with varying connectivity characteristics

MPEG21 DIA specifies a set of description tools (syntax and semantics) that can be used to assist the adaptation of multimedia contents. These tools could be used to satisfy transmission, storage and consumption constraints, as well as Quality of Service management by the various Users. Firstly, DIA provides a way to express the correspondence between usage context and content characteristics, resulting in possible adaptation operations required for obtaining the optimized version. Secondly, it provides the means to perform content adaptation in an efficient and coding format independent way.

### 2.1.1 MPEG21-DIA tools

The Digital Item Adaptation tools are grouped in eight major categories. Four of them that are directly related to SEA are briefly described in this paragraph.

- **Terminal and network quality of service (QoS)** addresses the problem of selecting optimal parameter settings for media resource adaptation to satisfy constraints imposed by terminals and/or networks while maximizing the QoS. *These tools establish relationship between constraints and feasible adaptation operations satisfying the constraints.* A proper schema is defined, in a way that, terminal and network QoS management is efficiently achieved by adaptation of media resources to imposed constraints. In particular Adaptation QoS tool (AQoS) specify the syntax to select the optimal adaptation parameter. The *Utility Function* tool instead describes the relationship between constraints, adaptation operators and utilities in a list format.



- **Usage Environment Description (UED) Tool.** This category includes *User* characteristics, terminal capabilities, network characteristics and natural environment characteristics. The tool provides syntax and semantics to describe the various properties of the usage environment in order to obtain, for the user, the optimal adaptation of Digital Items for transmission, storage and consumption. It contains syntax and semantics (in XML format) for the description of User characteristics.
  - *Terminal Capabilities*; these tools provide description of terminal capabilities in terms of power characteristics (average consumption, battery capacity etc.), processing constraints, codec capabilities, device properties (which include display and audio output capabilities) the average/maximum input/output bit-rate.
  - *Network characteristics*; are intended network capabilities and conditions (available bandwidth, delay, error characteristics etc.) in order to have a more efficient and robust transmission of resource
- **Universal Constraints Description (UCD) Tool.** The UCD tool describes the syntax and semantic to define constraints that are applied in the AdaptationQoS tools described previously. These constraints can be applied not only for the whole media resource, but also at logical sub-partition (such as GOPs, ROIs etc.) referred as adaptation units. The UCD provides means for specifying supplementary information to further constrain the usage scenario and usage environment of a *Digital Item*, beyond that is possible by a UED.
- **Bitstream Syntax Description (BSD).** With binary media resources a variety of adapted versions can be retrieved from a single bitstream by performing editing style operations such as data truncation and transformation. In order to have an adaptation processor that is not aware of the specific bitstream coding format, a generic approach is considered by providing a method based on XML for a manipulating bitstreams. XML in fact is used to describe the high-level structure of the bitstream; the resulting document is called Bitstream Syntax Description (BSD). This description acts as an additional layer and describes how the bitstream is organized in term of layers or packets of data. The BS description is scalable, in the sense that may provide a description of the bitstream with different levels of detail; furthermore, *the BSD is itself adaptable in order to properly reflect the bitstream adaptation*. Once that the BS description is available, it is possible for the adaptation engine to modify it by means of XSLT style sheet depending on the different constraints, and generating back the adapted bitstream.

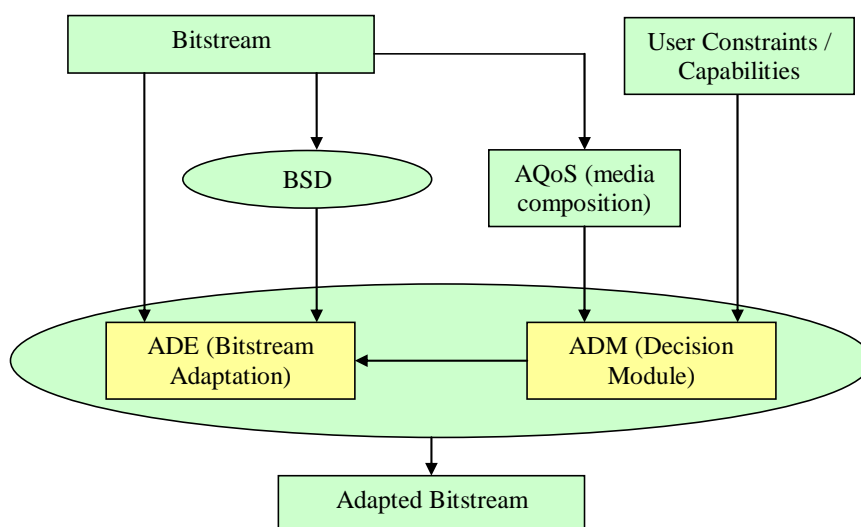


Figure 5: MPEG21-DIA Adaptation Process



### 2.1.2 Drawbacks of a pure MPEG21-DIA implementation

MPEG21-DIA is a general framework based on XML metadata. One of the main advantages of the MPEG21-DIA framework within SEA architecture would be that, by means of the BSD approach (2.1.1), Adaptation Execution process could be performed independently of the codec used to generate the media bitstream. Furthermore, as stated 2.1.1, the MPEG21 framework is based on the exchange of metadata documents among terminals and network nodes (HMG/NMG). This means that an optimal Adaptation Decision (ADM) for each terminal could be taken within the MG, by collecting capabilities and preferences provided by means of the UCD/UED metadata files. As an example, when used in conjunction with a scalable codec like H.264/SVC, the MPEG21-DIA tools make possible the creation of efficient and coding format independent adaptive streaming framework as described in [9].

Nevertheless we envisage some drawbacks and inefficiency in the implementation of the ADM and AEM modules by following strictly the rules of the MPEG21-DIA approach.

- As stated in §2.1.1, the BSD approach requires to generate and manage XML files containing the high level description of the bitstream; such high level description is necessary to perform adaptation (i.e. by dropping enhancement layer of an SVC content). Usually the BSD document need to be generated off-line by processing the original media sequence. This approach may be unpractical, as it force content provider to pre-process contents prior transmission, in order to have a BSD file for each video content. The creation and management of such XML documents may be computationally heavy, as the BSD may result in a huge file compared to the size of the bitstream, particularly for low-resolution video sequences. For an H.264/SVC bitstream for example, the BSD document will contain an indication of the location inside the bitstream of each access unit for each scalability layer; a low resolution video sequence composed by many scalability layers will result then in a BSD document comparable in size with the compressed bitstream. A relevant part of the storage area will therefore be occupied by XML documents
- BSD should be available in each network node that performs adaptation (HMG, NMG). Therefore transmission of BSD from content provider to MG would be necessary; actually there is not a standard way foreseen by standardization committee to transmit them
- Even for the UCD/UED documents there are not actually available standard protocols to properly transmit them over the network;
- The management of huge XML documents and the usage of style sheet libraries (like XSLT) to process them may require big memory resources. This could be a problem for devices with memory and processing constraints (such as STB)

In order to overcome the aforementioned problems, and to avoid potential inefficiency in the implementation of the SEA infrastructure, we propose in this document a slightly different method with respect to the pure MPEG21-DIA approach to perform automatic and optimal adaptation. While keeping some main concepts and tools of the MPEG21-DIA standard (for example the usage of UCD/UED documents to describe preferences and constraints), for the lightweight terminals, we propose a more efficient solution, based on SDP mechanisms. In such cases, we do not use the BSD approach to perform adaptation of the media contents, and avoid the transmission of UCD/UED documents among network nodes and terminals.

## 2.2 Signalling content attributes by SDP

The Session Description Protocol is defined by the Multiparty Multimedia Session Control (MMUSIC) Group of the Internet Engineering Task Force (IETF) in RFC 4566, [12]. SDP is a protocol to be used for session negotiation or declaration. SDP is also used by other IETF protocols



such as the Real Time Streaming Protocol – RTSP (RFC 2326,[10]) for controlling point-to-point multimedia streaming sessions, the Session Announcement Protocol – SAP (RFC 2974, [14]) for indicating multicast multimedia streaming sessions, or the Session Initiation Protocol – SIP (RFC 3261, [15]) for negotiation of multi-directional conversational multimedia sessions. SDP can be also used standalone where the protocol text is exchanged as an ASCII text file by other means.

SDP is an ASCII based clear text protocol which allows human as well as machine readability. A sessions description contains of a number of lines of ‘<type>=<value>’ format. An overview of the line types is given in Table 3. Mandatory and optional line types are indicated in the second column.

A session description consists of different sections:

(a) Session wide information section

The session wide information section contains, e.g. a "v=" line with information about the protocol type and an "s=" line specifying a session name. The session wide information may also contain a "c=" line including information about the connection, e.g. the IP address.

(b) Media description sections

If present, media description sections contain information about the different media types of a session (associated line types are "m=", "a=rtptime:", "a=fmtp:"), e.g. the payload type used in the Real Time Protocol – RTP (RFC 3550,[16]), UDP port, encoding parameters. Additional parameters describing the media are placed in one or more attribute lines ("a=") following the "m=" line.

Line Type	Mandatory/optional	Description
<i>Session description</i>		
v=	mandatory	protocol version
o=	mandatory	originator and session identifier
s=	mandatory	session name
i=	optional	session information
u=	optional	URI of description
e=	optional	email address
p=	optional	phone number
c=	optional	connection information--not required if included in all media
b=	optional	zero or more bandwidth information lines
t=...	optional	<i>One or more time descriptions ("t=" and "r=" lines; see below)</i>
z=	optional	time zone adjustments
k=	optional	encryption key
a=	optional	zero or more session attribute lines
m=...	optional	<i>Zero or more media descriptions ( see below )</i>
- <i>Time description:</i>		
t=	mandatory	time the session is active
r=	optional	zero or more repeat times
- <i>Media description:</i>		
m=	mandatory	media name and transport address
i=	optional	media title



c=	optional	connection information -- optional if included at session level
b=	optional	zero or more bandwidth information lines
k=	optional	encryption key
a=	optional	zero or more media attribute lines

**Table 3: Parameters specified by Session Description Protocol**

For the purpose of adaptation, the media description lines provide crucial information which can be evaluated by different network elements. Transport of layered media offers opportunities for adaptation by selective manipulation of the different layers, if the attributes of each layer are described in the SDP. Table 4 shows an example of a declarative session description offering one video media that contains three layers with different encoding/transport parameters described by different attributes.

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 192.0.2.12
s=SVC SDP example
i=SVC Scalable Video Coding session
c=IN IP4 234.66.88.66/64
t=2873397496 2873404696
m=video 20000 RTP/AVP 96
b=AS:64
a=framerate:15
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=4d400a; packetization-mode=1; sprop-parameter-sets=Z01ACprLFicg,aP4Eag==; sprop-spatial-resolution=176,144
a=mid:1
m=video 20002 RTP/AVP 97
b=AS:128
a=framerate:15
a=rtpmap:97 H264-SVC/90000
a=fmtp:97 profile-level-id=53000c; packetization-mode=1; sprop-parameter-sets=Z01ACprLFicg,Z1MADEsA1NZYWCWQ,aP4Eag==,aEvgRqA=,aGvgRiA=; sprop-spatial-resolution=352,288
a=mid:2
a=depend:lay 1
m=video 20004 RTP/AVP 98
b=AS:256
a=framerate:30
a=rtpmap:98 H264-SVC/90000
a=fmtp:98 profile-level-id=53000c; packetization-mode=2;init-buf-time=156320; sprop-parameter-sets=Z01ACprLFicg,Z1MADEsA1NZYWCWQ,aP4Eag==,aEvgRqA=,aGvgRiA=; sprop-spatial-resolution=352,288
a=mid:3
a=depend:lay 1 2
```

**Table 4: SDP example describing SVC content**

The "b=AS:" line gives an application specific bitrate for the media resp. the layer this line applies to, which typically states the maximum bitrate over the whole session. In general, the SDP is send only once during session setup, so it is most useful for media coded at a fixed bitrate (per level). Adaptive transmission can be achieved by selecting a set of layers that match best the available throughput of the network. As dependencies between different layers have to be respected, an "a=depend:" attribute line is given for all layers except the base layer of an SVC stream or the base view of an MVC stream. Adaptation may also consider frame rate or spatial resolution.

<i>Syntax</i>	<i>Semantics</i>	<i>Unit</i>	<i>Context</i>
b=AS:	Max. application bitrate	kbit/sec	media specific
a=framerate:	Max. framerate	frames/sec	media specific



a=group, a=mid, a=depend	Indicates dependency of "m=" lines	-	session & media specific
a=prop-spatial-resolution	Max spatial resolution	pixels	media specific, only with "H264-SVC" payload format
a=profile-level-id	Profile and level of the contained H.264, MVC or SVC bitstream	-	media specific, only with "H264", "H264-SVC" payload formats
a=prop-scalability-info	Scalability SEI describing the SVC bitstream	-	media specific, only with "H264", "H264-SVC" payload formats
a=prop-parameter-sets	Sequence and Picture Parameter Sets of the H.264, MVC or SVC stream	-	media specific, only with "H264", "H264-SVC" payload formats

**Table 5: Media description parameters**



### 3 SEA Cross-Layer Adaptation Architecture

For the design and implementation of the Cross-Layer adaptation architecture, SEA has adopted and adapted wherever needed, the tools and mechanisms of both the MPEG-21 dynamic and distributed adaptation models [4]. The first one refers to the *adaptation according to dynamically changing usage environments* (e.g. varying bandwidth), while the latter to *multiple adaptation steps successively performed on different MPEG-21 peers*.

Moreover, for lightweight terminals, we utilise a more flexible SDP based adaptation approach. In this case, we assume that the last node is responsible for terminating the streaming session, reconstruct the video stream if needed, perform all major adaptations and finally streams a unicast, adapted stream to the terminal, based on terminal characteristics, network conditions and user preferences.

#### 3.1 SEA Cross-Layer Modules

To accommodate the SEA models as described in section §1.2, we have extended the architecture shown in [2] with the CLC functional nodes as shown in Figure 6.

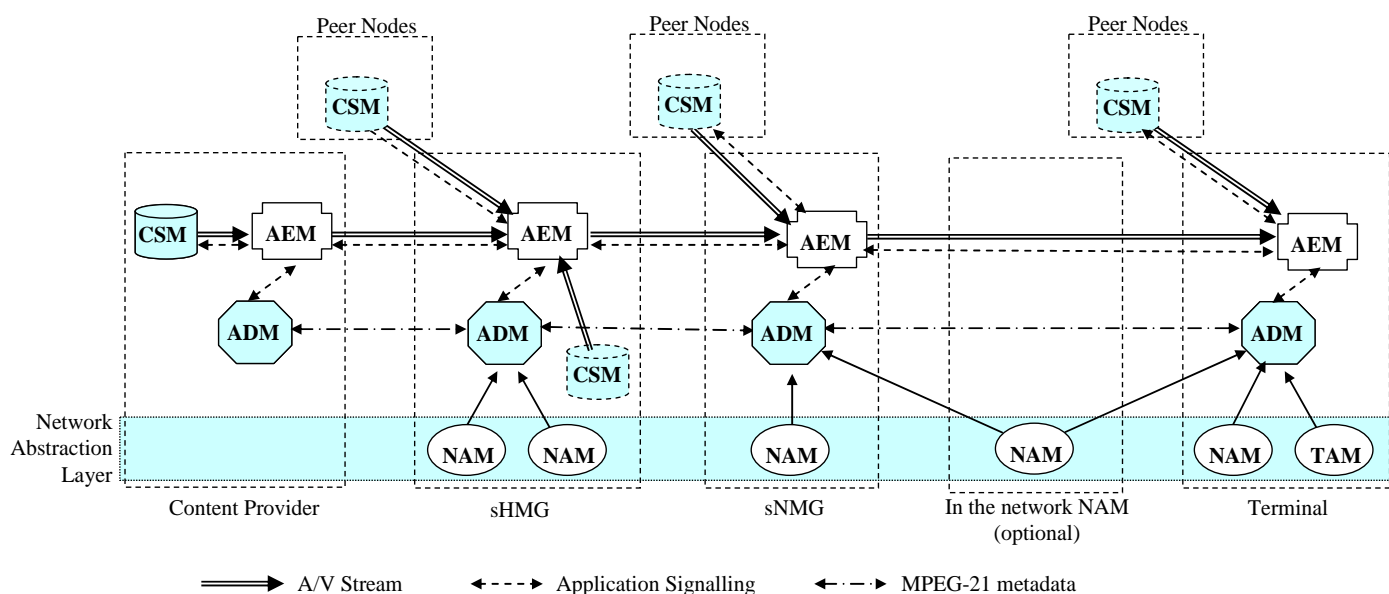


Figure 6: Cross Layer Modules communication

It has to be noted that as compared to [2], the  $AE_L$ , which is the Adaptation Engine of the *Last Node*, in Figure 6 is shown as sNMG. Yet, in another realisation of the network architecture, it could have been an sHMG.

Following a MPEG-21 like approach, the major CLC nodes in SEA are the Adaptation Decision Module (ADM) and the Adaptation Execution Module (AEM). These modules offer the following functionality:

- a) **Adaptation Decision Module (ADM):** This module is able to decide if and what adaptation has to take place. Based on a multi-criteria decision framework, and network and terminal capabilities sensing, ADM will allow tuning of the encoding/streaming parameters, optimizing the end-to-end rate, the distortion image quality and the resilience strategies at the application layer, as well as information regarding the connecting terminals.



- b) **Adaptation Execution Module (AEM).** This module is the context aware module, which actually performs the A/V handling. AEM functions will include for example dropping or combining SVC layers and initiating MDC distribution over different paths.

The ADM and the AEM modules will be located on all intelligent SEA nodes i.e. the Content Provider node, the sHMG, the sNMG and the terminal. Additionally, based on the business model, the ownership of the sNMG, the sHMG and the capabilities of the terminal, three supporting entities may also be defined:

- c) **Content Storage Module (CSM).** This module is utilised to store or cache the A/V content segments (layers, views, descriptions). It may act as an A/V server or a peer node in a P2P delivery environment supporting on-the fly content enrichment.
- d) **Network Awareness Module (NAM):** This module has knowledge of the physical characteristics of the network (multiple access, QoS classes, coverage). Moreover, it may be able to measure or probe network parameters e.g. number of users, available bandwidth, etc. This module may be located at all intelligent SEA nodes i.e. the Content Provider node, the sHMG, the sNMG and the terminal. Moreover, it may be optionally located in the network, providing additional information which is directly retrieved by the network nodes.
- e) **Terminal Awareness Module (TAM):** This module is located at the user terminal and has knowledge of the physical characteristics of the terminal (display, network interfaces, processing power, decoding capabilities). Moreover, it may be able to measure parameters at the terminal e.g. CPU load, battery life, free storage space, and network conditions e.g. SNR, BER, etc.

At the application level, CLC poses a number of challenges, namely: a) Scalability (SVC) and b) Resilience/concealment. Such issues shall be addressed in a cross-layer optimization scenario, to achieve the full exploitation of their capabilities. We consider the situation of a video content to be distributed from a service provider to a plethora of users with different capabilities and requirements, and employing different access technologies (e.g. GPRS, WiMax, ADSL, etc.) and we would like to provide the possibility that the moving end-user can seamlessly change the transmission medium during the streaming session.

## 3.2 Adaptation Engine Architecture

Taking into account the SEA adaptation scenarios as they are described in section §1.2 and summarized in Table 1, adaptation may be applied to a large set of received stream types. In more detail, the received stream that may be adapted may be SVC (base layer with or without enhanced layers), MVC (with a number of views), MDC encoding different types of video (e.g. SVC base layer, MVC), P2P video chunks (either used as transport where P2P is unaware of the video format that it is carrying or the P2P network is aware and gives different priorities to the different chunks) and a number of their combination.

Thus, a more detailed view of the Adaptation Engine is shown in Figure 7. As it may be seen, the stream may be P2P, SVC/AVC, MVC or MDC and may be adapted by one or more AEM sub-modules.

For simplicity reasons, the SEA AEM sub-modules are specialized for the video streams that we face and do not cover other video types. Moreover, we avoid transcoding at the network as it poses scaling limitations, while adaptation is focused on scaling characteristics of the streams (e.g. layers, views, descriptions etc.).

As shown in Figure 7, in a first layer the NAM/TAM modules provide input to the ADM module. In a second layer, the ADM modules communicate horizontally to exchange information and make decisions. It is important to note that each ADM is making a decision for the network that will follow (please refer also to Figure 4), while in VoD cases this information is propagated to the ADM



modules that are closer to the Video Server. Finally, the ADM that are selected will actually trigger the associated AEMs to perform the adaptation.

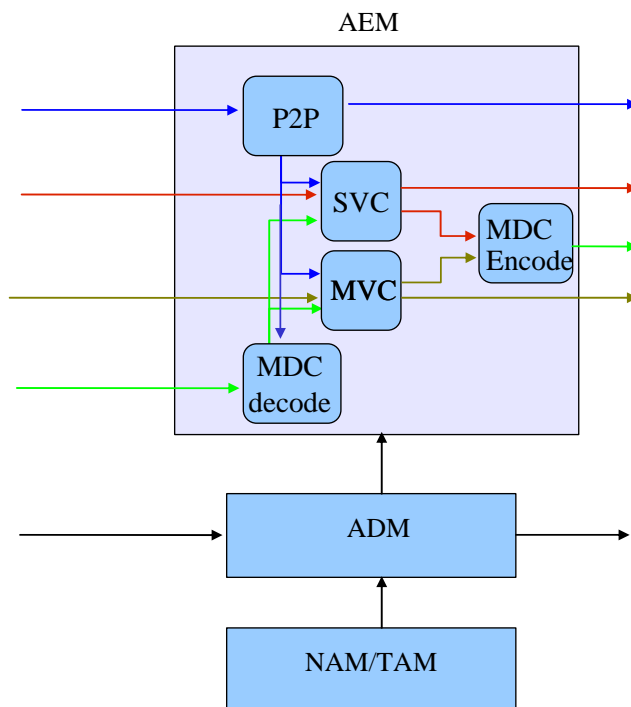


Figure 7: Adaptation Engine Architecture

We further analyze the required functionality of the main components of the SEA Cross-layer Adaptation Architecture in sections 4 and 5.

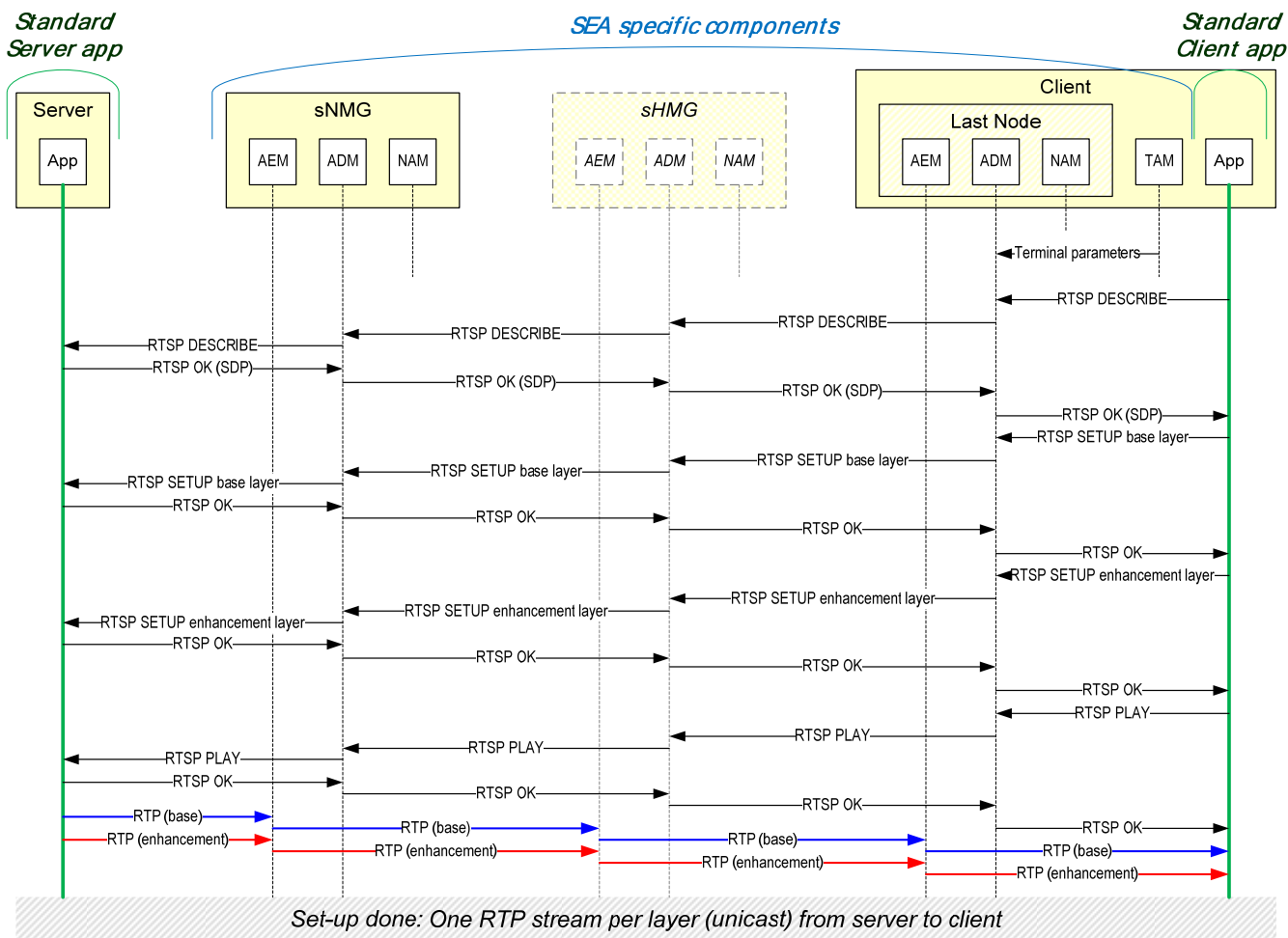
### 3.3 Initialization and Adaptation of Streaming Sessions

In order to better understand the background on which media adaptation takes place and how the adaptation modules have to collaborate, we give an example below how to set up streaming sessions initially. Thereafter, the streaming sessions will be adapted to various conditions of the environment as also explained further below.

We assume that a client (terminal) starts to connect to a resource offered by a streaming server, and that a number of SEA media gateways (MGs), either sHMGs or sNMGs, are passed in the path between the server and the terminal. These media Gateways may be utilised latter for adaptation of the streamed video. We assume that within SEA, the video part of the media requested by the client is an SVC, MVC or MDC stream. Any of these media streams are scalable in the sense that they can be divided into different portions, which are distinct layers, views or descriptions, respectively, of the total media stream.

The complete description of the media is conveyed by RTSP using the SDP. Each of these portions is transported in its own RTP session. Consequently the client has to open several sessions once it has received a complete description of the media. Both the messages for session setup (RTSP) and the media streams itself (RTP) are passing the SEA MGs, which forward the data as proxy servers (Figure 8).

Note that both the server and the media player do not have to fulfil special SEA requirements, but may be common applications. Moreover, the number of SEA specific nodes (GWs) involved may vary depending on the use case as indicated by the dashed outline of the sHMG.



**Figure 8: One-RTP-stream-per-layer, end-to-end unicast**

The underlying intention to use distinct RTP sessions for each layer is that this transportation method can also be applied if multicast sessions are involved, which would be possible either end-to-end or in parts of the transmission chain, which support multicast. In Figure 9, multicast is used from the server to the sNMG only. This set-up reduces the traffic in that part of the network significantly, compared to mere point-to-point streaming, as in practice there is more than one sNMG connected to the same server.

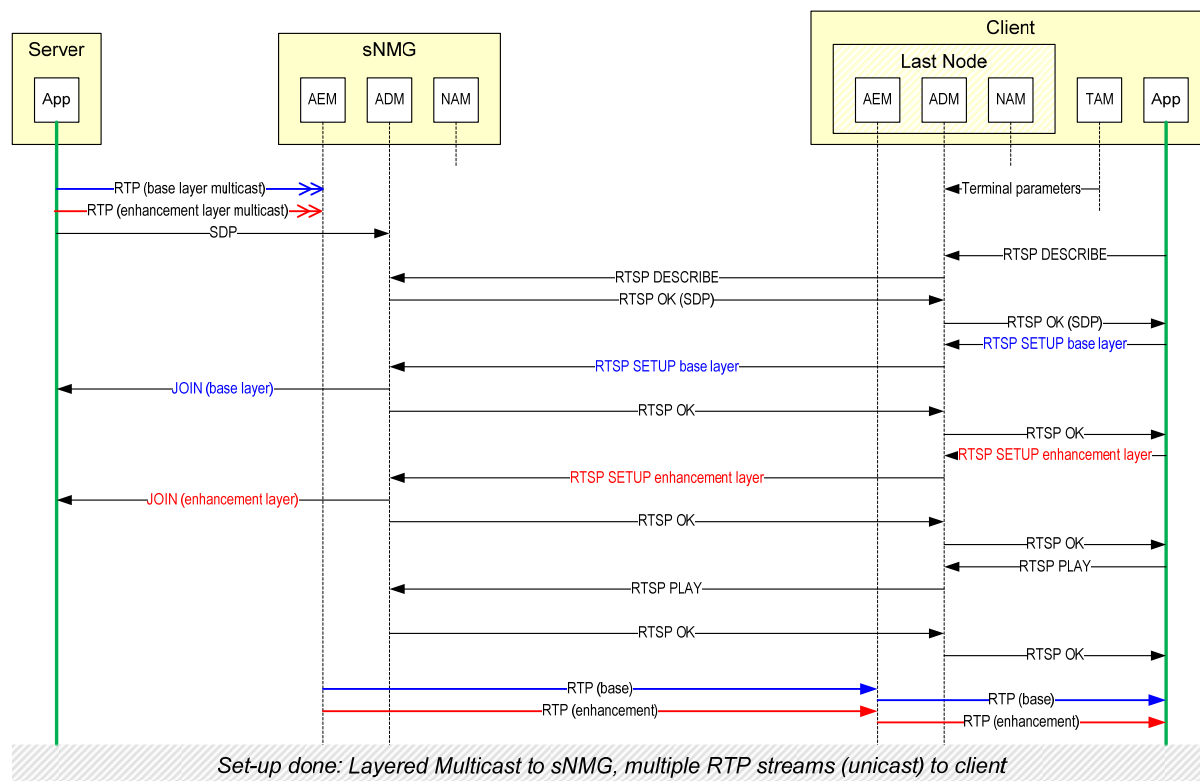


Figure 9: Layered Multicast to sNMG only

Once a streaming session is set up, a SEA specific MG will monitor the network performance and execute appropriate media adaptation. In order to keep metadata information up-to-date on all ADMs involved in the media chain, a mechanism for metadata exchange is established between each ADM and the ADM in the neighbouring nodes as shown in Figure 10.

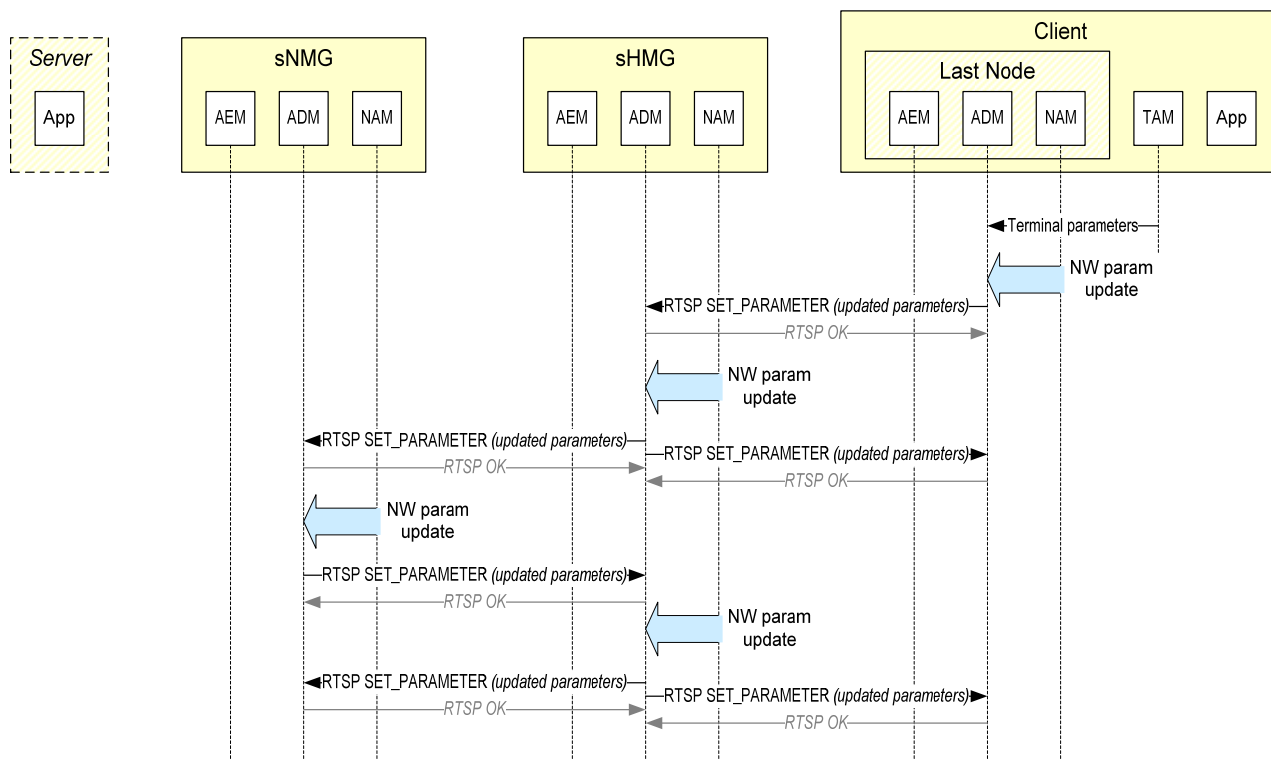


Figure 10: Inter-ADM communication



An adaptation example is shown in Figure 11. Here, the sNMG and the “Last Node”, which is an adaptation engine collocated with the media player application at the client machine, are involved in the adaptation. Both the server and the media player itself are common applications capable of RTP reception, RTSP communication and SVC (resp. MVC, MDC) decoding.

The server provides one (unicast) RTP stream per layer. These streams are forwarded through the complete streaming chain to the client application which combines and decodes all layers it receives. The  $ADM_{(sNMG)}$  in the sNMG is triggered by a “low bandwidth” condition reported from its  $NAM_{(sNMG)}$ , see the reddish shaded block in Figure 11. The  $ADM_{(sNMG)}$  decides to drop the enhancement layer and asks the  $AEM_{(sNMG)}$  to drop all packets of the enhancement layer. The RTSP command SET\_PARAMETER is used for the downstream communication between two adjacent ADMs., By this command, information about which adaptation has been executed by the  $ADM_{(sNMG)}$  is sent to the  $ADM_{(L)}$  using a SEA-specific parameter.

In case the bandwidth increases again, the  $ADM_{(L)}$  decides to forward the request to add a layer to the  $ADM_{(sNMG)}$  to add the enhancement layer again. There are two possibilities:

- **Yellowish block:** The  $ADM_{(sNMG)}$  decides not to ask the  $AEM_{(sNMG)}$  to forward all packets for the enhancement layer again, but sends a negative acknowledge (RTSP 453) back to the  $ADM_{(L)}$
- **Greenish block:** The  $ADM_{(sNMG)}$  asks the  $AEM_{(sNMG)}$  to forward all packets for the enhancement layer again and sends a positive acknowledge (RTSP 200) back to the  $ADM_{(L)}$ .

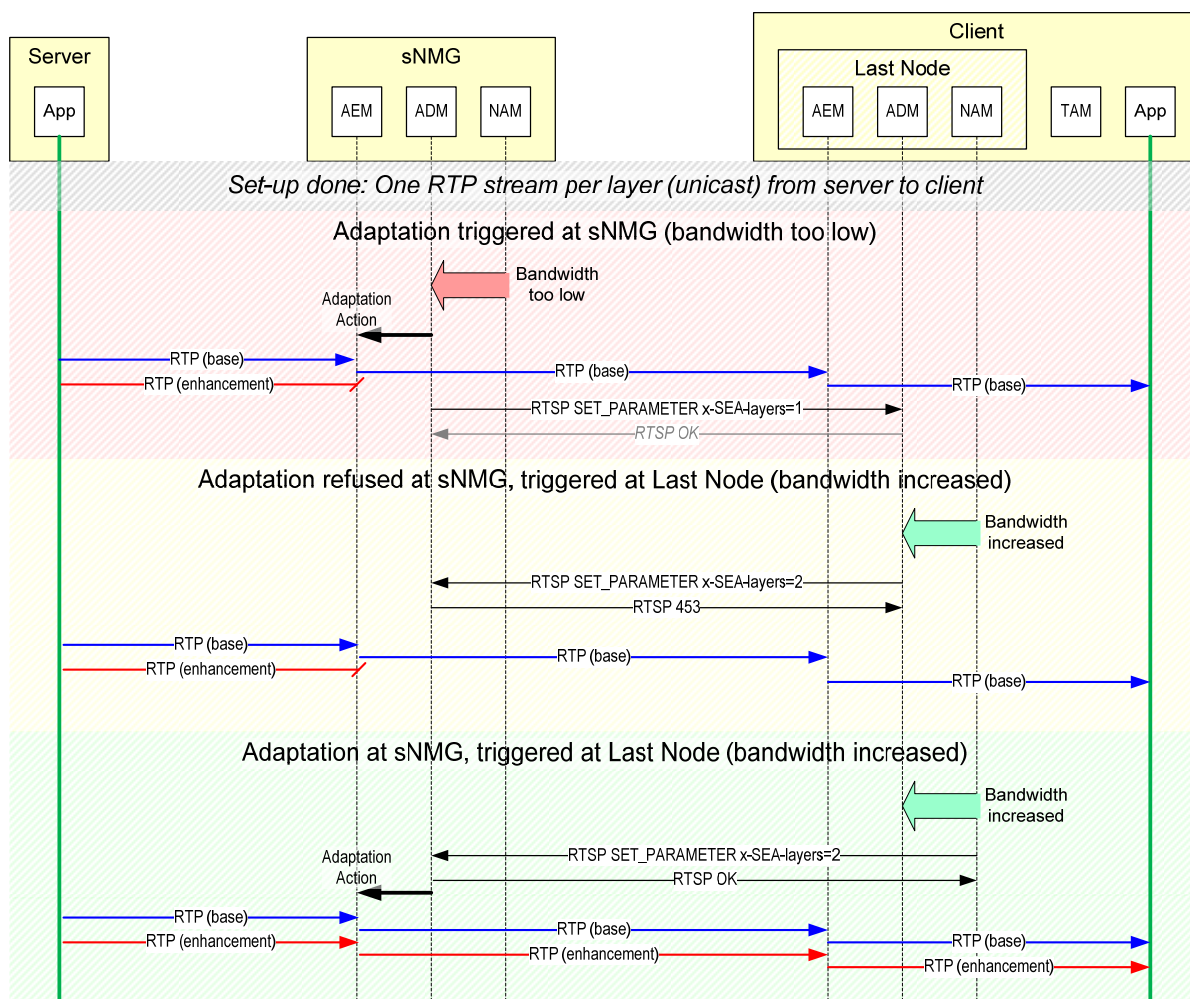


Figure 11: Adaptation triggered first at sNMG, then at Last Node



## 4 Adaptation Decision Module (ADM)

The ADM decides if, when and what adaptation has to take place in the streaming process. ADM is a software component that is able to take adaptation decisions based on normative metadata. Adaptation decisions are mainly a set of values that are used as parameters for steering the encoding or adaptation mechanisms. ADM is hosted on all the content aware nodes of the networks, namely the Content Provider server, the sNMG, the sHMG and the terminal.

In order to make decisions, the ADM needs knowledge of the network and the terminal characteristics. This information is provided by the NAM and TAM entities respectively, described in a format compliant to MPEG-21 Part 7 Usage Environment Description (UED) [5]. Communication between the ADM modules achieves the end-to-end PQoS enhancement.

### 4.1 Module Functionality

Based on a multi-criteria decision framework, and sensing of the network (NAM) and the terminal (TAM) capabilities and environment, ADM allows tuning of the encoding/streaming parameters, optimizing the end-to-end rate, the distortion image quality and the resilience strategies at the application layer, as well as information regarding the connecting terminals. ADM will further adapt pre-encoded scalable SVC and MVC media content based on network and terminal information.

Based on the SEA adaptation scenarios as they are described in section §1.2, ADM may decide that:

- a) SVC layers/MVC views may be dropped due to network congestion or terminal inability to decode them,
- b) the terminal has multiple radio access network interfaces available, thus reception via a more broadband network will be better,
- c) P2P technologies have to be initiated to enrich the AV content.

Additionally, the ADM will take into account various non-technical criteria: user preferences, service contract, operator business model etc.

The main decision tacking and adaptation functionality of the ADM is described in §3.3. In more details the following adaptation activities will be supported:

#### 4.1.1 Stream Initialisation

The first step is to set-up the RTSP session between the server and the client. All ADMs that are located in the path play an active role in the initialisation process, as information about the terminal, the network and the stream has to be stored. More precisely the following functions take place:

##### TAM registration/negotiation

The first task in the stream initialisation is the TAM registration process. During this process the terminal module sends information to the ADM located on the terminal about the current terminal capabilities. In case an ADM is not located on the terminal, information may be sent to the ADM<sub>L</sub> (last node ADM).

As explained in the next session, the TAM messages are based on MPEG-21 UCD/UED messages; though intra-SEA or application specific messages may also be utilised. In case of terminals with very small capabilities, TAM registration may be omitted, and the ADM<sub>L</sub> will assume that all intensive adaptation functions should take place at the ADM<sub>L</sub>, and the adapted stream will be forwarded via a unidirectional connection established between the ADM<sub>L</sub> and the terminal.

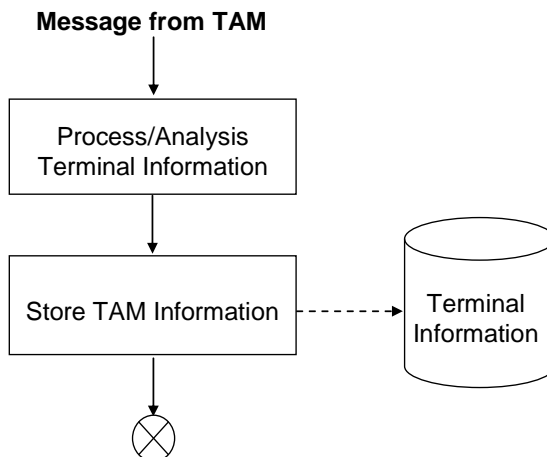


Figure 12: Terminal information is registered at the ADM

**Session Setup Request**

When the terminal initiates a new stream session, it will send a session set-up RTP message. The messages flows are explained in section §3.3, while the ADM functionality is shown in Figure 13.

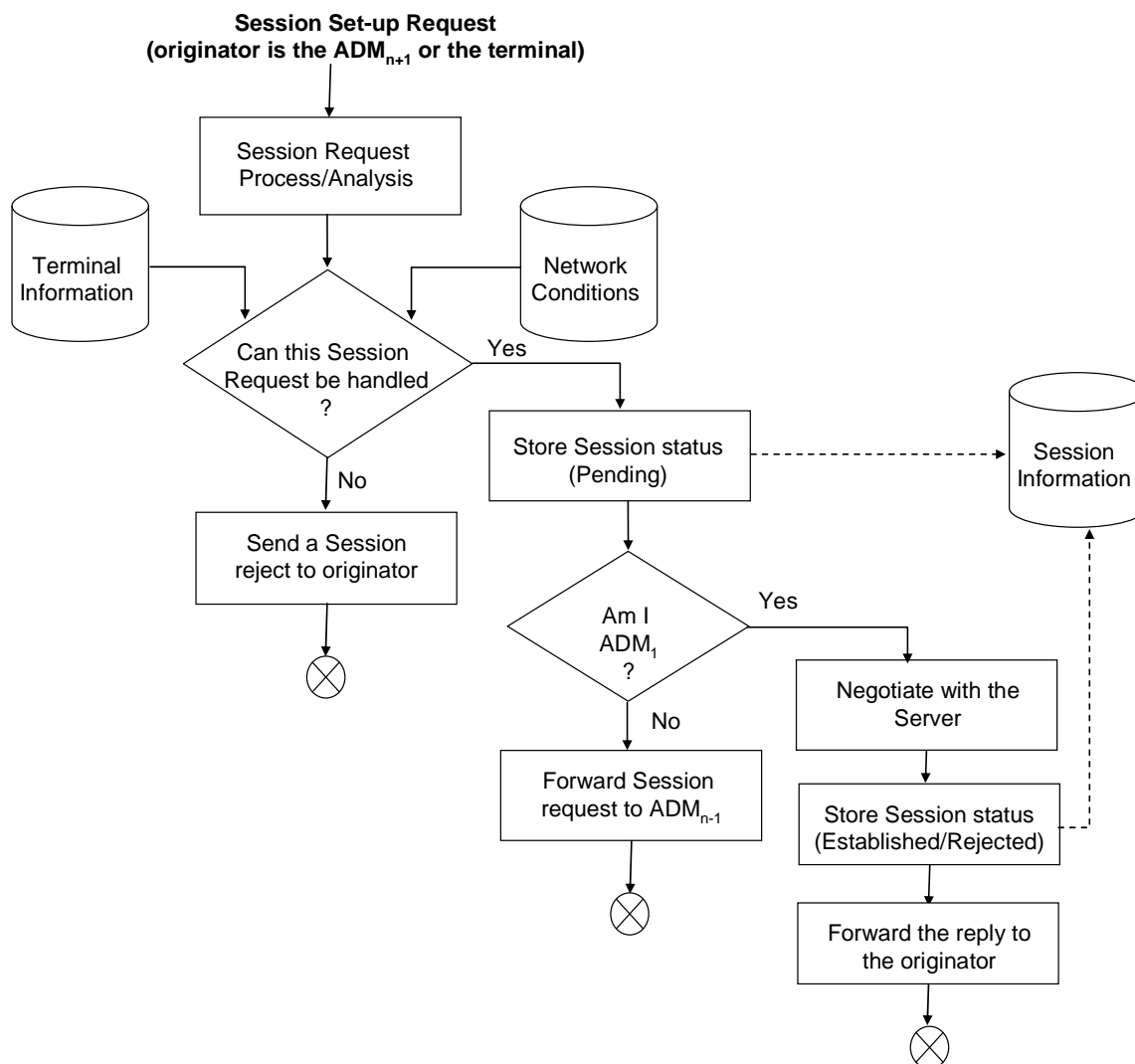


Figure 13: Stream session set-up request processing at the ADM



Whenever  $ADM_n$  receives a set-up message from the  $ADM_{n+1}$  or the terminal, initially it processes and analyses the session request. Then it proceeds to a decision process to derive if it is able to do the session set-up or not. In this process the terminal information (if available) and the network information (provided by NAM) are taken into account. If  $ADM$  decides that the session request can't be handled it responds to the originator, either the  $ADM_{n+1}$  or the terminal, with a Session reject response message.

In case  $ADM$  decides that the session can be handled, it temporally stores the session information (as pending, yet some resources are allocated), and then checks if it is the  $ADM_1$ .  $ADM_1$  is the first  $ADM$  just after the server. If yes, it starts the negotiations with the server. In some cases (e.g. layered multicast), the  $ADM$  may decide that this is the  $ADM_1$ , as it may directly establish the connection, though it does not negotiate directly with the server. In case the  $ADM$  is not the  $ADM_1$ , the session set-up message is forwarded to the  $ADM_{n-1}$ .

Finally, the set-up request is propagated to the  $ADM_1$ . The negotiation with the server may result in a session establishment or rejection. In both cases, the session information is updated and the result is returned to the originator.

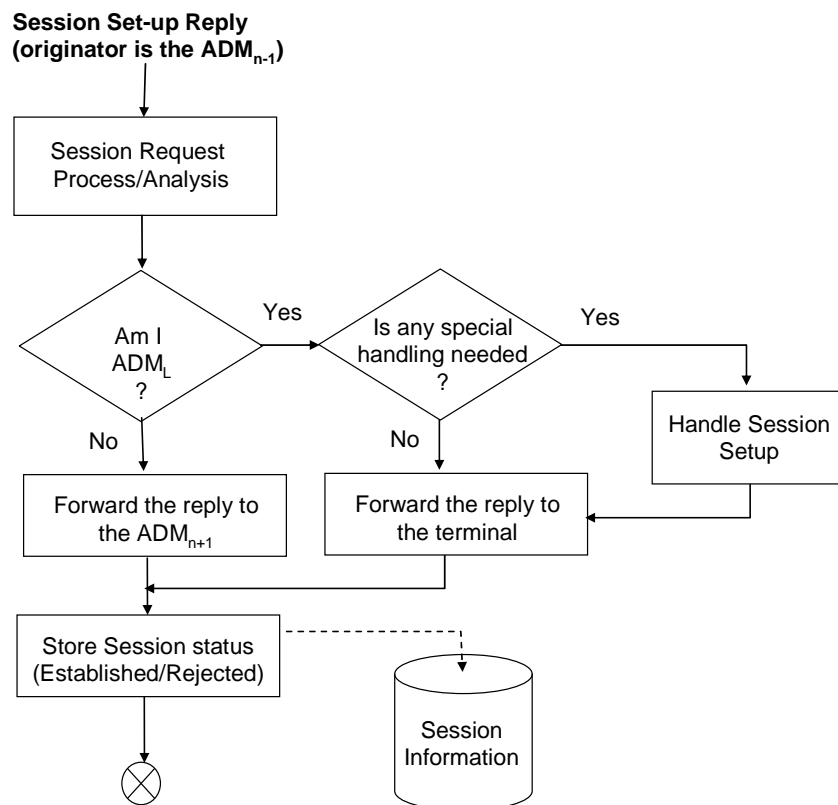


Figure 14: Stream session set-up reply processing at the ADM

Whenever a  $ADM$  receives a session set-up reply message, after an initial processing and analysis, the  $ADM$  checks if it is the Last Node  $ADM$  ( $ADM_L$ ). If it is not, it forwards the reply message to  $ADM_{n+1}$  and updates the session status (which was in pending status).

If it is the  $ADM_L$ , it checks if the session requires special handling (e.g. if the terminal can't handle adaptation, P2P buffering, MDC encoding etc). In either case, the reply is forwarded to the terminal and the session status is updated.

#### 4.1.2 Stream Adaptation

Stream adaptation may be initiated by:

- a) the terminal NAM



- b) the NAM of a Media Gateway in the network
- c) the AEM in case of end-to-end adaptation.

Case (c) is examined in chapter 4.1.3 "End-to-end Content Adaptation Mechanism", thus we'll not consider it here further. In case (a), the terminal's NAM measures a major change at the network conditions and thus it informs the terminal ADM. As shown in Figure 15, the terminal ADM after an initial process and analysis of the network information, it derives a decision on the necessity for an adaptation. In this decision it takes into account the Terminal Information (provided by TAM), the active sessions information and the stored (previous) network information. In case, ADM decides that no adaptation is needed, it will just store the new network condition.

Adaptation takes place at the ADM<sub>L</sub> or at an ADM node as close as possible to the server. In case the terminal ADM decides that adaptation is needed, then for each session that is active, it will forward an adaptation request to the ADM<sub>L</sub>.

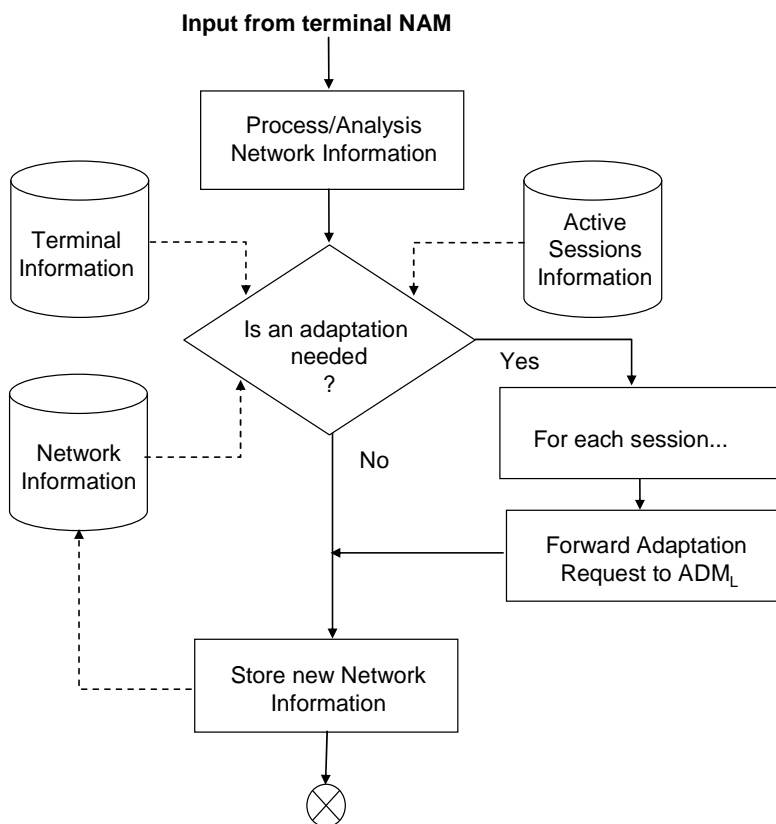


Figure 15: Stream adaptation initialised by terminal NAM

In case (b), the Media Gateway's NAM measures a major change at the network conditions, either positive, the network condition is better, or negative. Thus it informs the ADM. As shown in Figure 16, the Media Gateway ADM after an initial processing and analysis of the network information, it derives a decision on the necessity for an adaptation. In this decision it takes into account the Terminal Information (provided by TAM), the active sessions' information and the stored (previous) network information. In case, ADM decides that no adaptation is needed, it will just store the new network condition to use it as a reference point in the next interaction.

In case the Media Gateway ADM decides that adaptation is needed, then for each terminal and for each session that is active, it will execute the procedure "Handle Adaptation", which is explained in Figure 17.

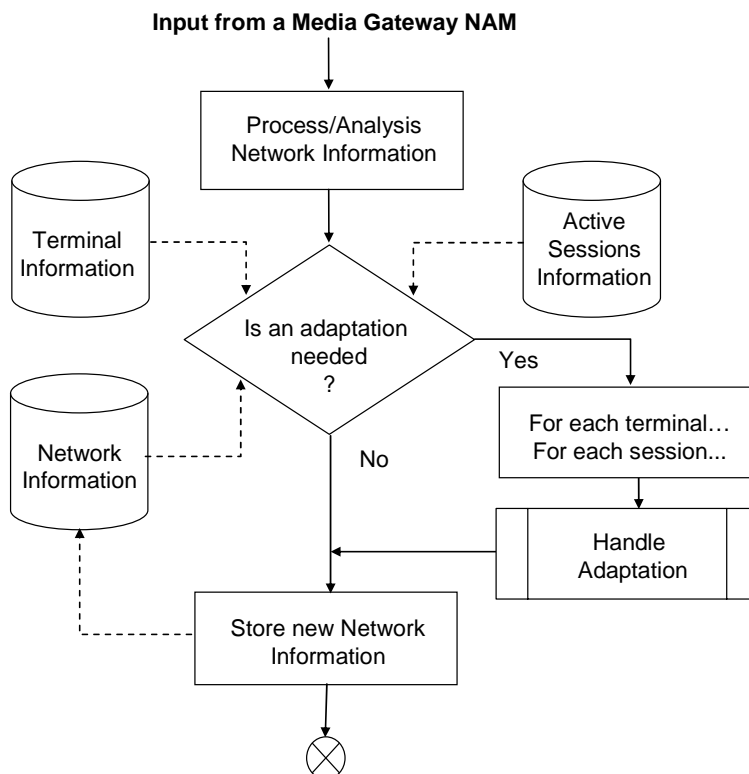


Figure 16: Stream adaptation initialised by a MG NAM

In the adaptation handling procedure (Figure 17), initially the ADM checks if it “has to do”, “should do” and “can do the adaptation”. In the first question, the ADM checks if it is the ADM<sub>1</sub>. If it is, then it has to proceed with adaptation as there is no other ADM in path to the server. If it is not ADM<sub>1</sub>, it checks if it should do adaptation at this layer. This may happen in specific cases i.e. layered multicast. If it is not ADM<sub>1</sub> and there is no specific reason for doing the adaptation here, the adaptation request is forwarded to the ADM<sub>n-1</sub>, which is one hop closer to the server.

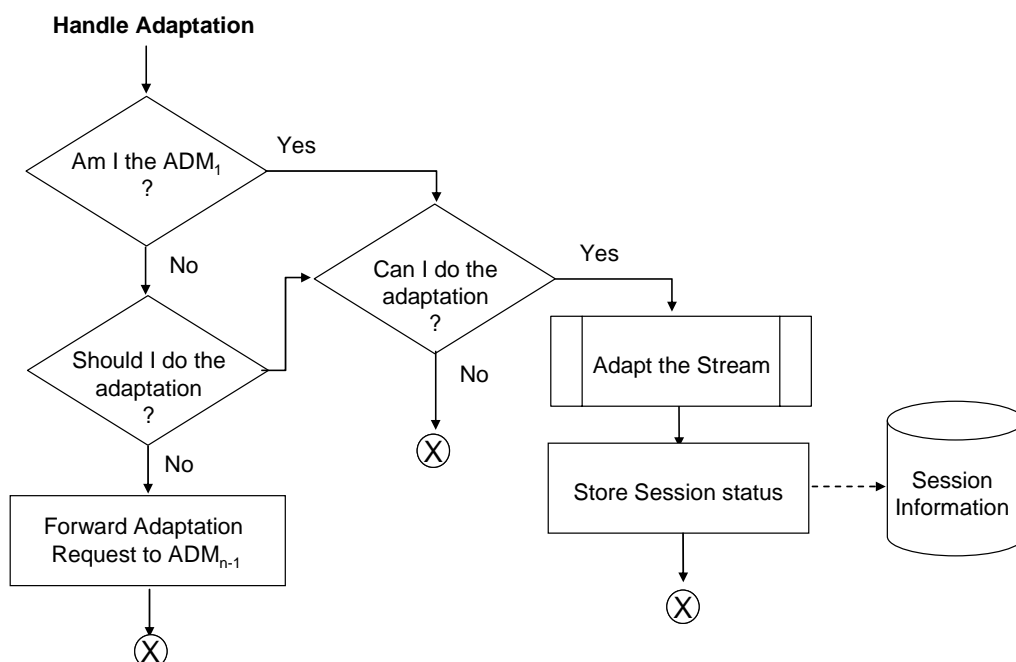


Figure 17: Handle Adaptation procedure

If the adaptation *has* or *should* take place in this ADM, the next step is to check if the ADM can do the adaptation. In a negative answer, the adaptation is rejected. In case the adaptation will take place



an “Adapt the Stream” procedure (shown in Figure 20) and the new session status is stored a session database.

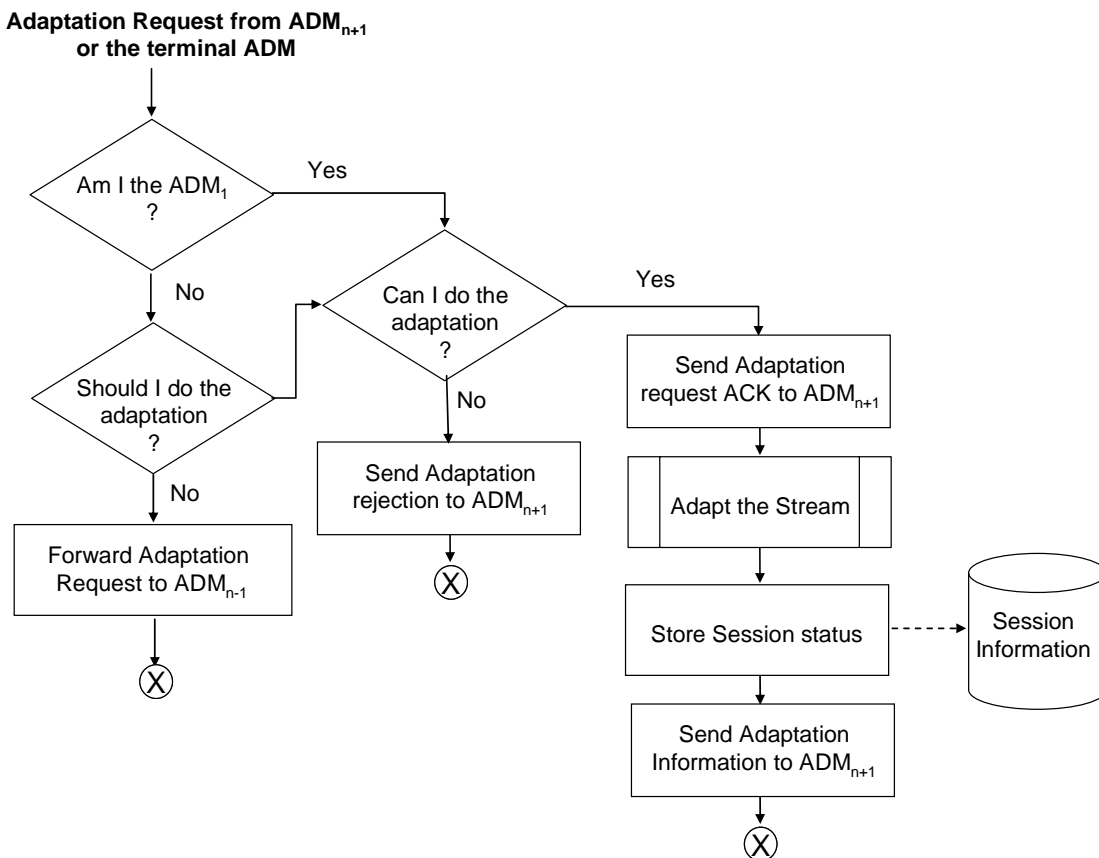


Figure 18: Stream adaptation initialised by the ADM<sub>n+1</sub> or the terminal ADM

In case the ADM receives an adaptation request from the ADM<sub>n+1</sub> or the terminal ADM, it follows the logical diagram shown in Figure 18. Again the ADM check if it “has to do”, “should do” and “can do the adaptation”. In case it is not the ADM<sub>1</sub> or there is no specific need to do the adaptation here, the adaptation request is forwarded one hop closer to the server at the ADM<sub>n-1</sub>. In case it is decided that adaptation has to take place here and the ADM is able to do the adaptation, the ADM sends a message to ADM<sub>n+1</sub> (or the terminal ADM) that adaptation will be handled (here), it adapts the stream (shown in Figure 20), stores the new session status and finally forwards a message with the adaptation information to ADM<sub>n+1</sub> or the terminal ADM.

In case it is decided that adaptation has to take place here, but the ADM is not able to do the adaptation, an adaptation reject message is send to the ADM<sub>n+1</sub>. This message is received by the ADM one hop closer to the terminal (Figure 19). This will decide if it can do the adaptation. If it can’t do the adaptation, if it is the ADM<sub>L</sub> a reject message will be sent to the terminal ADM; if it is not ADM<sub>L</sub> a reject message will be sent to ADM<sub>n+1</sub>. In case it is decided that the ADM is able to do the adaptation, the ADM sends a message to ADM<sub>n+1</sub> (or the terminal ADM) that adaptation will be handled (here), it adapts the stream (shown in Figure 20), stores the new session status and finally forwards a message with the adaptation information to ADM<sub>n+1</sub> or the terminal ADM.

So in this way the adaptation request is forwarded as close to the server as possible, having as cornerstones at the adaptation procedure the ADM<sub>1</sub>, the one closer to the server and the ADM<sub>L</sub>, the one closer to the terminal.

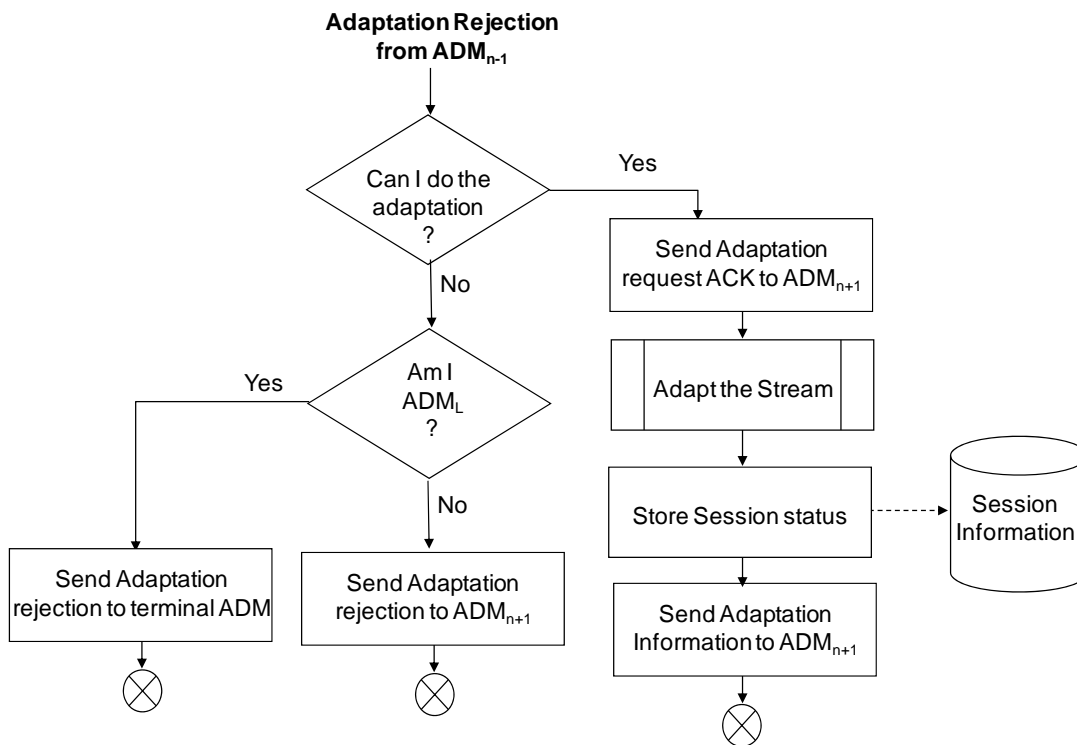


Figure 19: Adaptation rejection message from ADM<sub>n-1</sub>

In some ADM adaptation is decided to take place. The ADM extracts information from the SDP message regarding the composition of the bit stream in terms of image resolution, bitrate, layer dependencies etc. The ADM decides some form of adaptation and in general it can decide that some layers resp. views may be dropped during this phase. In particular – if the Scalable SEI message is present – the ADM decision may benefit from extra information provided in this message for a finer adaptation. In this special case the adaptation can go inside a single RTP layer and some of its NALUs may be dropped<sup>2</sup>.

In the adaptation procedure as shown in Figure 20, the first step is to check if the ADM is ADM<sub>L</sub>. If yes, then additional functionality and adaptation may take place. First of all, the ADM checks if the session packets that it receives are coming via the SEACast and they are P2P chunks. In that case, based on the terminal profile and capabilities, the ADM may decide to terminate the P2P here in order to avoid terminal overloading with P2P buffering. Thus, it will send a message to the SEACast/AEM to terminate P2P.

The next step would be to check if the received stream is encoded using MDC. If yes, it may also decide to terminate MDC in ADM<sub>L</sub>, by sending a relevant message to the AEM.

After the P2P and/or the MDC streaming has been terminated or if this ADM is not the ADM<sub>L</sub>, the ADM will check if this is a SVC encoded stream. If yes, and taking into account the terminal, user and stream information, ADM will decide if adaptation can take place e.g. “How many SVC layers are available?”, “Would the user accept just SVC base layer?”, “Should an additional SVC layer be multicasted?” etc. If adaptation at SVC context can take place, ADM will send the relevant command to the AEM..

<sup>2</sup> An example based on SVC coding. Hierarchical B-frames allow several layers of temporal scalability (depending of the GOP size), but the RTP session may aggregate all or some of these layers in a single RTP session to avoid fractioning the bit stream in too many RTP session, with regard to reordering complexity and firewall pinhole opening. Analysing each layer of scalability may allow a finer adaptation. If we have a device displaying at 15Hz, with a bit stream that presents a base layer of QCIF 30Hz and enhancement of CIF 30Hz in the SDP searching for presence of hierarchical B-frames, we can discard them fitting the 15Hz native monitor resolution.

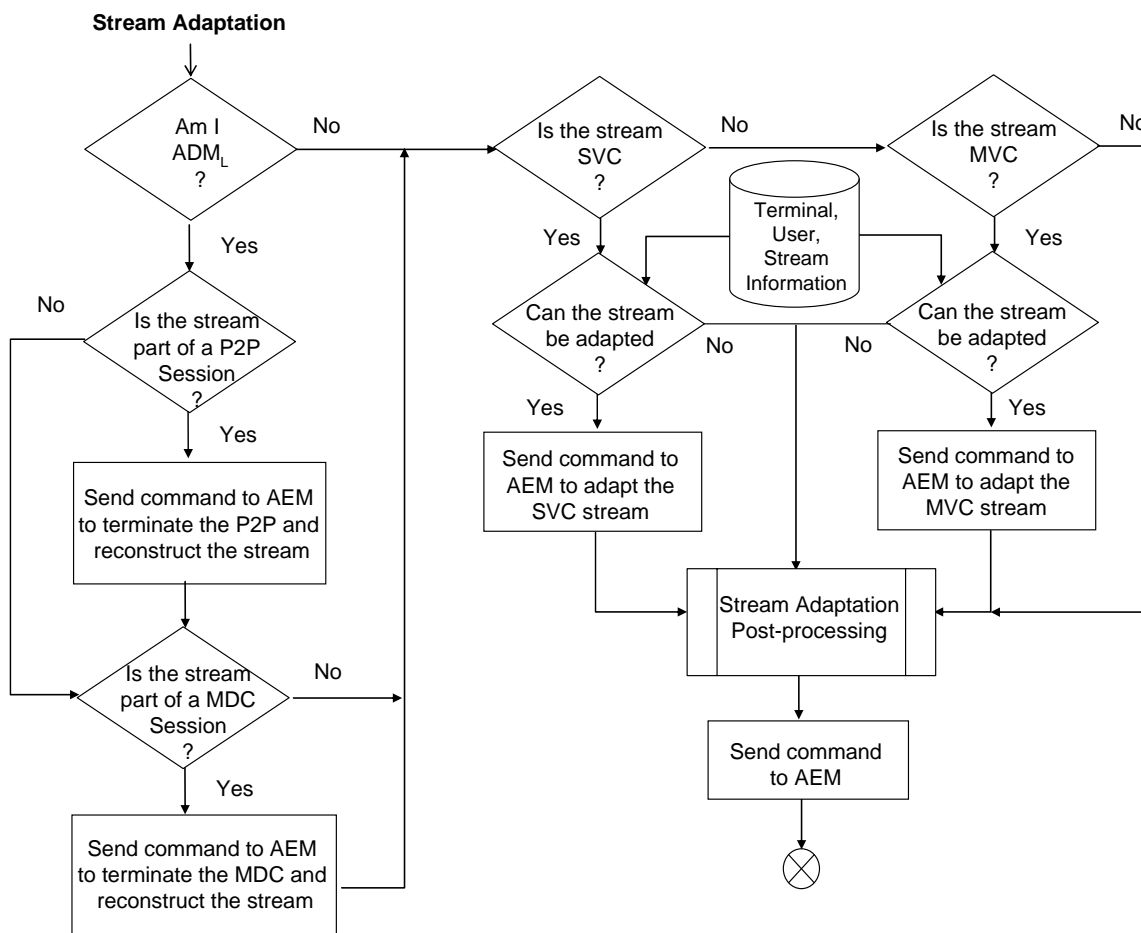


Figure 20: Stream adaptation procedure

In case the stream is an MVC stream, the ADM will decide if one or more views can be rejected (or a new view can be added) and send the relevant command to the AEM.

It should be noted that when the ADM forwards the SDP to the next hop in the adaptation chain, the dropped layers are not removed from the SDP in order to ensure that – in case of some improvement in the network links – the next hop is able to subscribe some higher layers and provide a better quality to the terminal. Through updated UCD/UED information the ADM in the terminal/MG may decide to change the quality for a specific media session during the streaming phase. Nonetheless, some manipulation of the SDP message is required. In case the MDC is terminated or– optionally – initiated in the MG, the SDP should be modified to inform the next hop of the changed transmission.

After the SVC or MVC stream has been adapted, and based on the media gateway processing power, the system load and the terminal capabilities, a stream adaptation post-processing function will be initialized. In the stream adaptation post-processing, initially the ADM checks if this is the ADM<sub>L</sub>. If not the post-processing function will exit.

If it is the ADM<sub>L</sub>, it will check if the network session can be adapted, taking into account the terminal capabilities, the user preferences and the network information. In a positive result, the ADM<sub>L</sub> will decide (in co-ordination with the terminal ADM) to adapt the session by modifying the terminal’s active network interfaces (activate or deactivate an interface), change the window size of the IP packet size, increase/decrease the allocated bandwidth if supported, or even change the TDMA time slots or the OFDM carriers. All these network adaptations are subject to the actual network interface and terminal.

After the network session adaptation has been check and handled, ADM<sub>L</sub> will check if the terminal has multiple active network interfaces. In case of negative answer, the process will exit. In case of positive answer, the ADM<sub>L</sub> will check if additional MDC transcoding should be applied in the last



path (between the  $ADM_L$  and the terminal), taking advantage of the diverse propagation paths and reflections that the different network interfaces may have. The decision will be based on the terminal capabilities, the user preferences (including QoS vs cost preferences), the media gateway processing congestion and load and of course the network conditions. If the result is positive, the MDC module will be initiated and the stream will be transcoded to multiple descriptions, before it is streamed to the terminal. Yet, from current experimentation it seems that MDC transcoding in real-time faces a number of scaling problems; thus it's final applicability will be further studied.

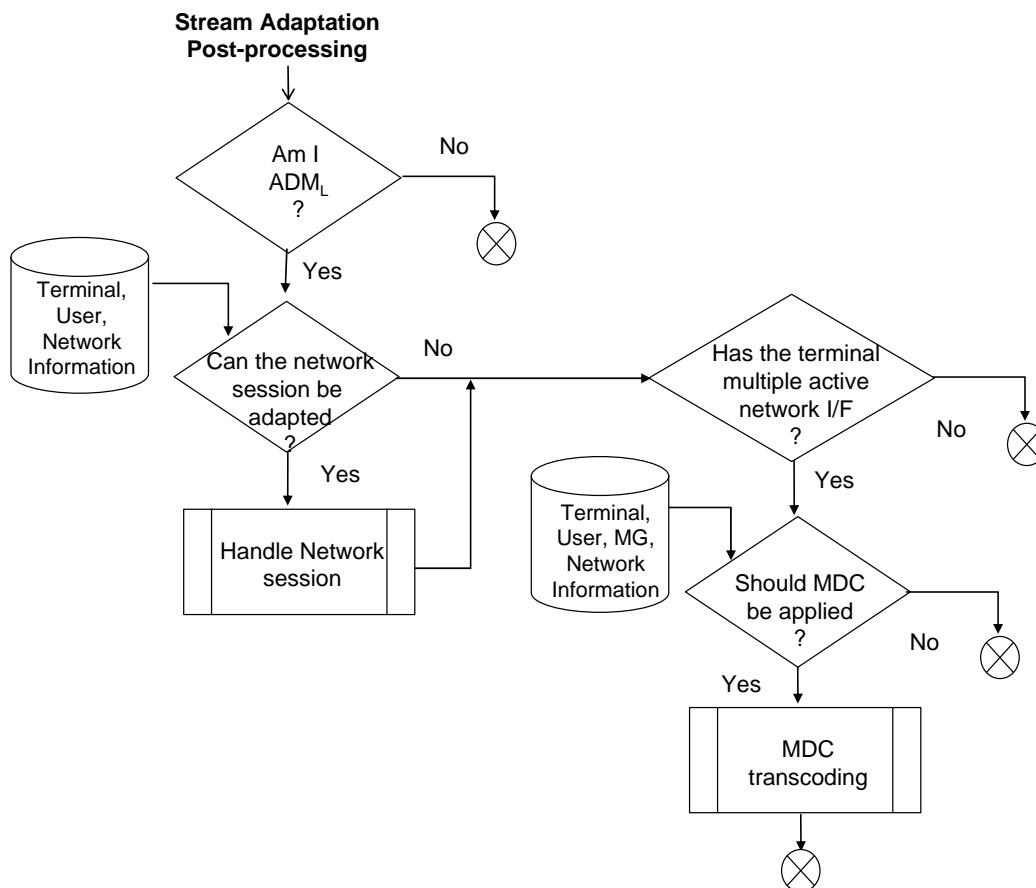


Figure 21: Stream adaptation post-processing

### 4.1.3 End-to-end Content Adaptation Mechanism

In general end-to-end adaptation is performed via the collaborating ADMs, and actually content adaptation is realised at one ADM/AEM node, either the  $ADM_I$ , the  $ADM_L$  or the ADM that is closer to the server, if no other special conditions apply (e.g. layered multicast). Yet, in some cases QoS and Perceived QoS (PQoS) may be calculated only at the two ends of the streaming session. For instance the bit error rate, the end-to-end delay, the jitter are normally measured/calculated at the edges. In that case, the adaptation may be initiated by the AEM.

As it is shown in Figure 22, the AEM may initiate an adaptation request. This request is processed and analysed, taking into account the current terminal, active sessions and network information and if an adaptation is needed this is handled as described earlier (Figure 17).

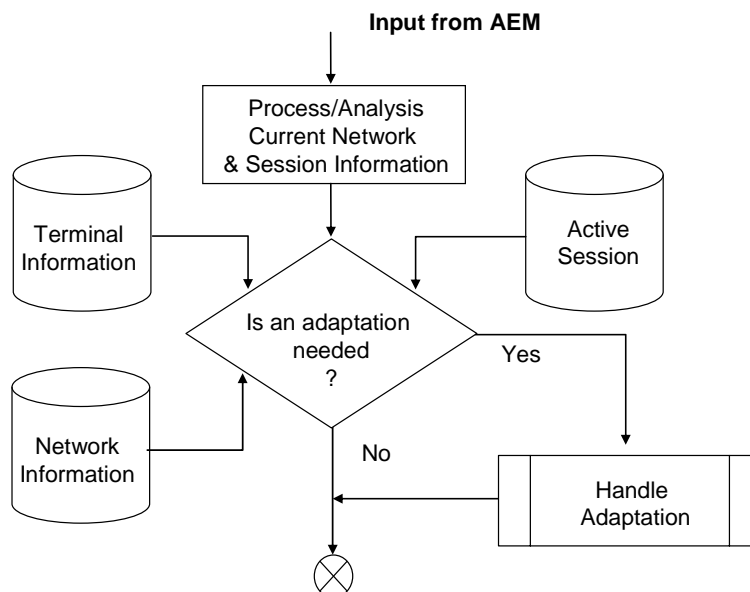


Figure 22: Adaptation initialised by the AEM

Finally, in the end-to-end scenario, the co-operating ADMs may decide that adaptation may take place in two places in the network:

- a) to the ADM that is closer to the server (or to the last point that layered multicast is applied) and
- b) at the ADM<sub>L</sub>, when additionally stream adaptation post-processing is required.

## 4.2 Software Architecture

The ADM software module will be implemented in all content aware nodes of the networks, as well as in content providers’ servers and terminals. It will expose interfaces with the NAM and TAM modules in order to collect information about terminal parameters and preferences as well as network conditions; on the other hand, interfaces will be necessary also to provide the results of the adaptation decision mechanism to the execution modules (AEM).

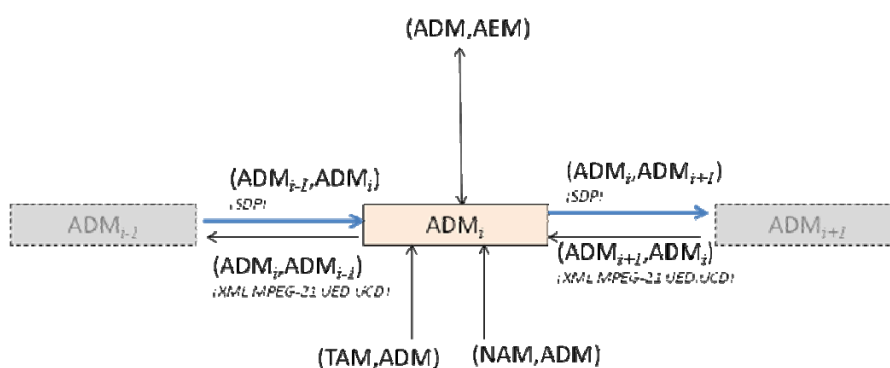


Figure 23: Adaptation Decision interfaces

As illustrated in Figure 23, based on information about terminal and network conditions retrieved by the NAM/TAM and the description of the session type and media composition provided by the SDP message, the ADM will take the optimal adaptation decision.



### 4.3 Interface – Functions Definition

The ADM software module communicates with NAM/TAM in order to collect the necessary information about network condition, physical features of the terminal and user preferences. Such information will be formatted in MPEG21-DIA standard compliant metadata files, namely the UCD/UED documents. Figure 24, gives an example of the UED document that describes the capabilities of the terminal, in terms of display size and maximum input.

```
<?xml version="1.0" encoding="UTF-8"?>
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS" xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg21:2003:01-DIA-NS ..\DIASchema\UED.xsd">

  <Description xsi:type="UsageEnvironmentType">
    <UsageEnvironmentProperty xsi:type="NetworksType">
      <Network xsi:type="NetworkType">
        <NetworkCharacteristic xsi:type="NetworkConditionType">
          <AvailableBandwidth maximum="3200" average="3200" minimum="3200"/>
        </NetworkCharacteristic>
      </Network>
    </UsageEnvironmentProperty>
    <UsageEnvironmentProperty xsi:type="TerminalsType">
      <Terminal>
        <TerminalCapability xsi:type="DisplaysType">
          <Display xsi:type="DisplayType">
            <DisplayCapability xsi:type="DisplayCapabilityType">
              <Mode>
                <Resolution horizontal="352" vertical="288" activeResolution="true"/>
              </Mode>
            </DisplayCapability>
          </Display>
        </TerminalCapability>
      </Terminal>
    </UsageEnvironmentProperty>
  </Description>
</DIA>
```

**Figure 24. Example of a UED metadata document for a scalable video content**

Figure 25, shows an example UCD description. The UCD document specifies different types of constraints (cf. chapter 2.1.1 above): the *limitation constraints*, which restrict the resolution space for the adaptation decision (for example imposing constraints on frame width and frame height) and the *optimization constraints*, which are used to choose the optimal solution after applying the limitation constraints (in the example, the maximization of the bitrate is chosen as optimization constraint).



```

<?xml version="1.0" encoding="UTF-8"?>
<DIA xmlns="urn:mpeg:mpeg21:2003:01-DIA-NS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:mpeg:mpeg21:2003:01-DIA-NS ..\DIASchema\UCD.xsd">
  <DescriptionMetadata>
<ClassificationSchemeAlias alias="SFO" href="urn:mpeg:mpeg21:2003:01-DIA-StackFunctionOperatorCS-NS"/>
<ClassificationSchemeAlias alias="AQoS" href="urn:mpeg:mpeg21:2003:01-DIA-AdaptationQoS-CS-NS"/>
<ClassificationSchemeAlias alias="MEI" href="urn:mpeg:mpeg21:2003:01-DIA-MediaInformationCS-NS"/>
  </DescriptionMetadata>
  <Description xsi:type="UCDType">
    <AdaptationUnitConstraints>
      <LimitConstraint>
        <!-- FrameWidth <= 352 -->
        <Argument xsi:type="SemanticalRefType" semantics=":MEI:17"/>
        <Argument xsi:type="ConstantDataType">
          <Constant xsi:type="IntegerType">
            <Value>352</Value>
          </Constant>
        </Argument>
        <!-- Bool IsLessThanOrEqualTo operation-->
        <Operation operator=":SFO:38"/>
      </LimitConstraint>

      [...]

      <!-- Bitrate <= Average Bitrate of the Network -->
      <LimitConstraint>
        <Argument xsi:type="SemanticalRefType" semantics=":MEI:9"/>
        <Argument xsi:type="SemanticalDataRefType" semantics=":AQoS:6.6.5.3"/>
        <Operation operator=":SFO:38"/>
      </LimitConstraint>
      <!-- FrameWidth <= Horizontal Resolution of the Device -->
      <LimitConstraint>
        <Argument xsi:type="SemanticalRefType" semantics=":MEI:17"/>
        <Argument xsi:type="SemanticalDataRefType" semantics=":AQoS:6.5.9.1"/>
        <Operation operator=":SFO:38"/>
      </LimitConstraint>
      <!-- FrameHeight <= Vertical Resolution of the Device -->
      <LimitConstraint>
        <Argument xsi:type="SemanticalRefType" semantics=":MEI:18"/>
        <Argument xsi:type="SemanticalDataRefType" semantics=":AQoS:6.5.9.2"/>
        <Operation operator=":SFO:38"/>
      </LimitConstraint>
      <!-- maximize the Bitrate -->
      <OptimizationConstraint optimize="maximize">
        <Argument xsi:type="SemanticalRefType" semantics=":MEI:9"/>
      </OptimizationConstraint>
    </AdaptationUnitConstraints>
  </Description>
</DIA>

```

**Figure 25. Example of an UCD metadata file for a scalable content**

By parsing the incoming SDP message, the ADM is able to extract the information concerning the format in which the media is received (SVC/MVC/MDC), the session type (layered, single session) and the media composition (available layers, views, descriptions). By collecting this information, it is possible for the ADM to generate the format dependent AdaptationQoS description; in the example of Figure 26 the AQoS describes the adaptation capabilities of an H.264/SVC content. By matching the properties and preferences described in the UCD/UED documents with the available SVC layers extracted from the SDP message and described in the AdaptationQoS, it is possible for the ADM to find the optimal adaptation parameters. This matching process can be seen as a mathematical optimization process as described in literature [9]. The resulting parameters will contain for the SVC bitstream the value of target bitrate, frame size, frame rate and will be provided to the AEM module that will use them to perform the proper editing operation on the bitstream.



```

<Constraint iOPinRef="Bitrate">
  <Values xsi:type="IntegerVectorType">
    <Vector>219 262 298 727 1064 1430</Vector>
  </Values>
</Constraint>
<AdaptationOperator iOPinRef="QualityLevel">
  <Values xsi:type="IntegerVectorType">
    <Vector>0 0 0 0 0</Vector>
  </Values>
</AdaptationOperator>
<AdaptationOperator iOPinRef="DependencyId">
  <Values xsi:type="IntegerVectorType">
    <Vector>0 0 0 1 1 1</Vector>
  </Values>
</AdaptationOperator>
<AdaptationOperator iOPinRef="TemporalLevel">
  <Values xsi:type="IntegerVectorType">
    <Vector>0 1 2 0 1 2</Vector>
  </Values>
</AdaptationOperator>
<Utility iOPinRef="Framerate">
  <Values xsi:type="IntegerVectorType">
    <Vector>7 15 30 7 15 30</Vector>
  </Values>
</Utility>
<Utility iOPinRef="Width">
  <Values xsi:type="IntegerVectorType">
    <Vector>176 176 176 352 352 352</Vector>
  </Values>
</Utility>
<Utility iOPinRef="Height">
  <Values xsi:type="IntegerVectorType">
    <Vector>144 144 144 288 288 288</Vector>
  </Values>
</Utility>

```

**Figure 26. Example of AQoS with layer composition and dependencies of a scalable video content**

When dealing with MVC bitstream, the AdaptationQoS will contain also indication of the view composition and dependencies, that will be possible to be extracted even in this case from the SDP description. On the other hand, the UCD/UED documents will contain constraints considering also the maximum number of views that can be received and processed by a terminal. After the adaptation decision, the ADM will be able to forward then the *id* of the view(s) that the AEM will have to extract from the bitstream.

In case the UED/UCD impose constraints on the QoS that is necessary to guarantee to a terminal when streaming over an error prone network, the ADM may decide to initiate an MDC session, and forward such decision to the AEM. On the other hand, if constraints on the QoS are relaxing, the ADM will have the capability to terminate an MDC session, and to enrich the content by adding SVC/MVC layers.



## 5 Adaptation Execution Module (AEM)

The AEM module will be a software module, hosted on all the content aware nodes of the networks, namely the Content Provider server, the sNMG, the sHMG. This module is the context aware module, which actually performs the A/V handling. It relies on media stream information generated by the encoder/server and offered via the Sessions Description Protocol (SDP) container as described in 2.2. Adaptation of SVC, MVC and MDC streams is controlled by requests from the ADM (cf. subsection 5.3).

Supplemental Enhancement Information (SEI) messages, if included in the bitstream by the encoder, can also be evaluated to find out about bit rates or other characteristic features of the media stream or sub-bitstreams thereof. In SVC streams, Scalability Information SEI messages can give information about the contained layers which can be useful for adaptation purposes. These messages can hold basic information for each layer such as `priority_id`, `discardable_flag`, `dependency_id`, `quality_id`, and `temporal_id`. Further optional information fields can give a more detailed view on each of the layers, e.g. about coding of regions of interest, average and maximum bitrate of each layer, frame rate or frame size. For MVC, similar information can be found in the View Scalability Information SEI messages, e.g. average and maximum bitrate of each view.

The AEM will have also the capability to terminate or initiate a MDC session anytime, depending on the decision provided by the ADM.

### 5.1 Module Functionality

The AEM functions will be mainly application specific. Functions may include dropping or combining SVC base and enhancement layers, adapting MVC streams and/or initiating MDC distribution over different network paths. Therefore, the MPEG21-DIA BSD approach as described in 2.1.1 won't be considered for AEM.

Moreover, the AEM will be P2P aware. It may have the ability to search for additional content in peer nodes, select them and retrieve additional information or content pieces.

Based on application layer's information exchange, SEA will be able to utilise the above modules and support on the network or at the terminal side content adaptation. More precisely, it may take advantage from a scalable coding scheme and exploit the flexibility of the H.264/SVC standard to adapt with a small computational effort the bit-rate of concurrent video streams, obtaining a better visual quality compared to transcoding solutions. Spatio-temporal and quality layers can provide several steps of visual quality improvement – from a minimum quality given by decoding the pure H.264/AVC base-layer up to the maximum quality achievable by decoding the full SVC bitstream.

Below, the adaptation process is illustrated by an example. Figure 27 shows the internal structure of five subsequent SVC access units of an example bitstream, coded with a hierarchical prediction structure. Each Access Unit (AU) is the coded representation of a single picture. An AU consists of a set of coded slices that belong to different sub-bitstreams. In Figure 27, the AUs are numbered in decoding order. In this example, AU 1 and AU 2 are key frames and contain a data portion that belongs to the base layer (Layer 0). If Layer 1 is decoded, which is present also in AU 3, the frame rate is increased by a factor of two. Decoding of Layer 2, which is present also in AU 4 and AU 5, again doubles the frame rate.

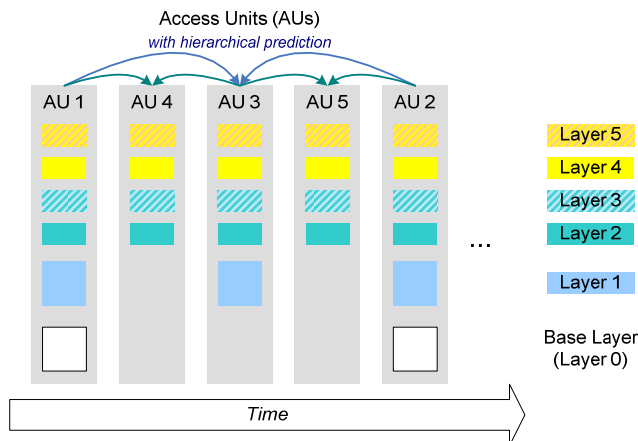


Figure 27: SVC Access Unit structure example

The association of the data portions to different operation points and their arrangement in the three scalability dimensions are shown in Figure 28. In this example, data portions are grouped to form six layers marked by different colors. Layer 1 not only doubles the frame rate (as mentioned above), but also doubles the spatial resolution. Layer 2 belongs to the same spatial resolution and doubles the frame rate. Layer 3 contains data of a higher fidelity levels (SNR). Layer 4 doubles the spatial resolution, and Layer 5 again enhances the fidelity of the video stream.

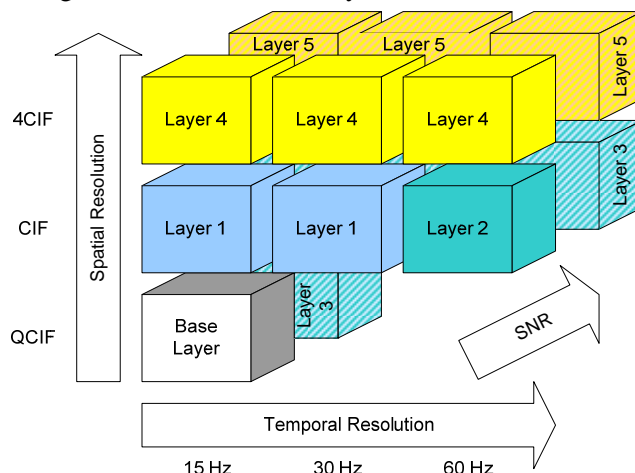


Figure 28: SVC Scalability dimensions

The adaptation process can start from any operation point. For example, if the AEM is forwarding the complete bitstream, the ADM can decide to drop one or more layers, starting from the highest. Figure 29 shows the sequence of AUs in decoding order, which in general is also the transmission order.

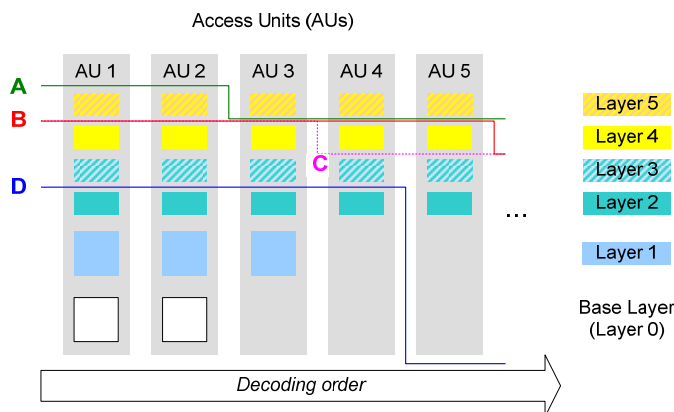


Figure 29: Adaptation process examples



If the ADM decides to drop Layer 5 after the AEM has processed AU 2, the AEM can react immediately (green line "A"), and any decoder will be able to decode frames at a lower quality level.

If the ADM decides to drop Layer 4 and the AEM responds to this decision immediately after processing AU 3 (pink dashed line "B"), the decoder which has been operating at 4CIF resolution has to fall back to CIF resolution which is a more complex requirement. It will be much more convenient for the decoder to switch spatial resolution at the next key frame (red line "C").

If the ADM decides to decrease the frame rate to 30 Hz, the AEM can drop both non-referenced AU 4 and AU 5 completely. If this decision is received after processing AU 4, only AU 5 will be dropped (blue line "D").

If the AEM is already dropping one or more upper layers, the ADM can also decide to add one or more layers. The AEM can add a fidelity enhancement layer at any time, but it is not always possible to respond immediately. If the enhancement layer contained in the next AU references past frames, and the referenced information has been dropped, the AEM has to delay its reaction until the next key frame. If the next AU is an IDR frame, any adaptation is feasible.

Similar rules apply for H.264/MVC streams. For such streams the ADM can decide to drop one view in order to react to the preferences expressed by the terminal or in order to fit network constraints. For example one terminal may be able to manage only two views among a set of views offered, in order to provide stereoscopic 3D vision; on the other hand, the ADM may decide to drop some view in order to fit bandwidth requirements of the network to which it is connected. In both cases the inter-view prediction scheme used to generate the bitstream must be taken into account. The current MVC reference model adopts a prediction scheme as the one illustrated in Figure 30. Therefore, in order to extract the desired sub-set of views it is necessary to identify the dependencies in term of prediction, between the views (S0, S1, ...,Sn). If the ADM decides to select the view S4 of the example shown in Figure 30, it will be necessary for the AEM to keep views S0 and S2, too. Information on views dependencies can be provided in the SDP message in the form of dependencies information as stated in 2.2.

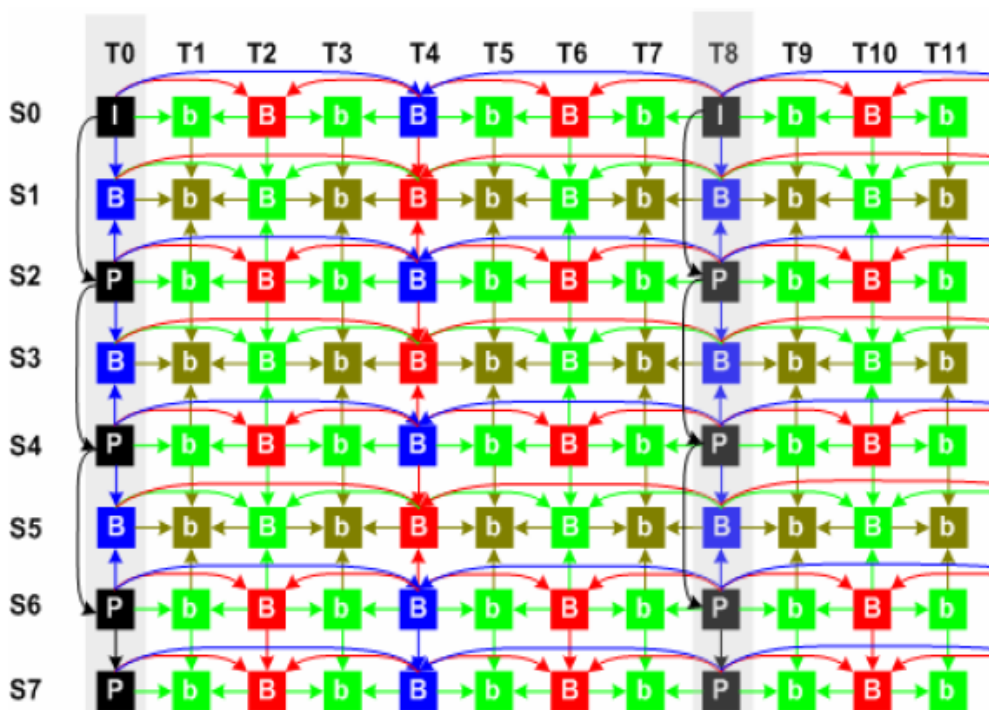


Figure 30. MVC reference prediction scheme

In order to improve the resilience of the transmitted stream, and to guarantee a fixed level of QoS at terminal, the ADM can decide to activate MDC scheme. As illustrated in Figure 7, the AEM will implement an MDC handler, able to manage, terminate and initiate MDC connections. In case of



incoming MDC sessions, the MDC handler will forward the descriptions to the next hop. In case of incoming SVC/MVC session, it could be possible to initiate the MDC session using directly the SVC/MVC base layer by transcoding/re-coding the content on the proper MDC format. The MDC handler may also be able to terminate the MDC session by merging the received descriptions and generating a full quality base layer for SVC/MVC streaming.

## 5.2 Software Architecture

The Adaptation Engine can be seen as a RTSP proxy server that adapts VoD connections depending on its links condition. Inside the Adaptation Engine, the AEM forwards either one input bitstream or a specific combination of complete input bitstreams or parts thereof, depending on the scenarios depicted in section 1.2.

We now focus on RTP input and output AEM data. Referring to Figure 7, input data transmission from the network is based on RTP layered or single sessions and P2P sessions controlled by RTSP. Output data transmission to the terminals is based on RTP layered or single sessions (again using RTSP). The AEM forwards the data of a layered RTP session or drop the layer, depending on the control message passed by the ADM. The AEM can also join multiple RTP sessions in a single RTP session. The AEM can also analyze the content of the RTP session and decide to remove some of the NALUs it contains.

The AEM of the last node terminates the P2P sessions, adapts the received multimedia information to the terminals' capabilities and forwards the adapted data to the connected terminals. The AEM can terminate MDC sessions, forward MDC data to the terminal or produce MDC sessions for the terminal.

The ADM exposes a control interface to the AEM (see subsection 5.3), which allows for a fixed set of media specific adaptation operations. Its main functionality reflects the selection of decodable sub-bitstreams included in the total set of bitstreams received by the AEM and a control about termination/adaptation of the streaming session to the terminal for what regards, layered coding and MDC session generation/termination.

A scheme of the Adaptation Execution Module elements and its interfaces with ADM can be seen in Figure 31. Through the following paragraphs we provide a description of the different elements, as they will be activated to work with the flow of data.

As it has already been noticed in the ADM functional description section, the ADM forwards the SDP to the next hop in the adaptation chain without removing from the SDP the dropped layers. In this way it ensures that, in case of some improvement in the network links, the next hop is able to subscribe some higher layers and provide a better quality to the terminal. Nonetheless, some manipulation of the SDP message is required. In case the MDC is terminated or— optionally — initiated in the MG (post-processing), the SDP should be modified to inform the next hop of the changed transmission.

If layered RTP sessions are present as an input of one streaming session, sessions can be forwarded, or some layers can be dropped depending on the command given by the ADM. If the RTP sessions are encrypted this is the only way of adaptation, as we do not have access to the payload information. If the sessions are transmitted in clear, a dedicated *Packet Analyzer* (PA) can be used for each layered session. The PA of the layer performs the finer adaptation process if the ADM requires so.

In an alternative path for the layered RTP sessions, they form a unique outgoing session. In this case, the *RTP reordering* block uses the *Decoding Order Numbers* (DONs) inside the RTP session to compose the multiple streams in a unique flow. This can be done only if the MG can access the internal RTP data of the session, hence the information is transmitted in clear. In this case a unique *Packet Analyzer* will decide which NALU to keep and which to drop.

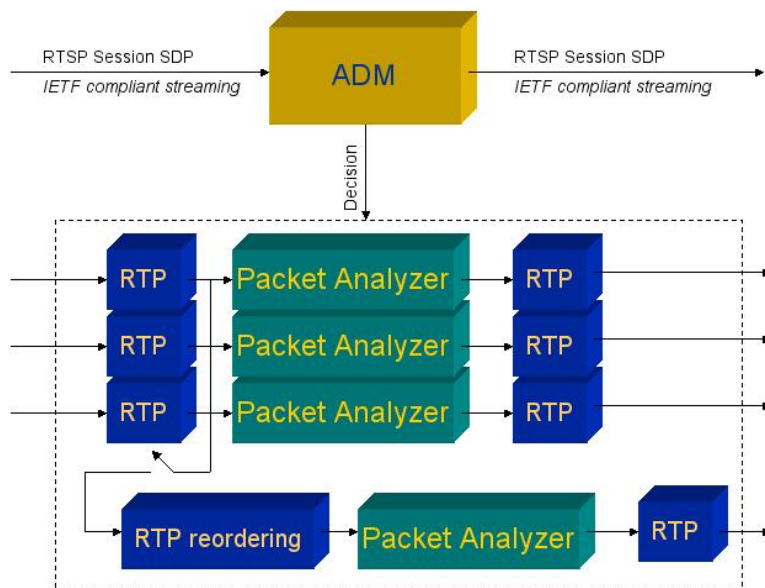


Figure 31. Adaptation Execution Module interfaces

**5.2.1 Description of SW architecture**

The software architecture of ASTRALS may serve as a basis for the development of the SEA project. The VideoLanClient (VLC) has proved to be a highly flexible and well-structured source code and we propose to extend it with the innovative functionalities we are exploring. A part of this section describes the extensions required for the project.

SEA proposes a new peer-to-peer technology – SEACast – with several innovative algorithms. SEACast is a P2P application composed by C++ and Python modules. SEACast connects to other SEACast applications in the network to create the P2P overlay and creates a socket to transmit the reconstructed Multimedia Content to a local client application. The VLC in the AEM will connect to the local port opened by the SEACast and will adapt and forward the bitstream according to the information passed by the AEM on the decision point for a specific terminal.

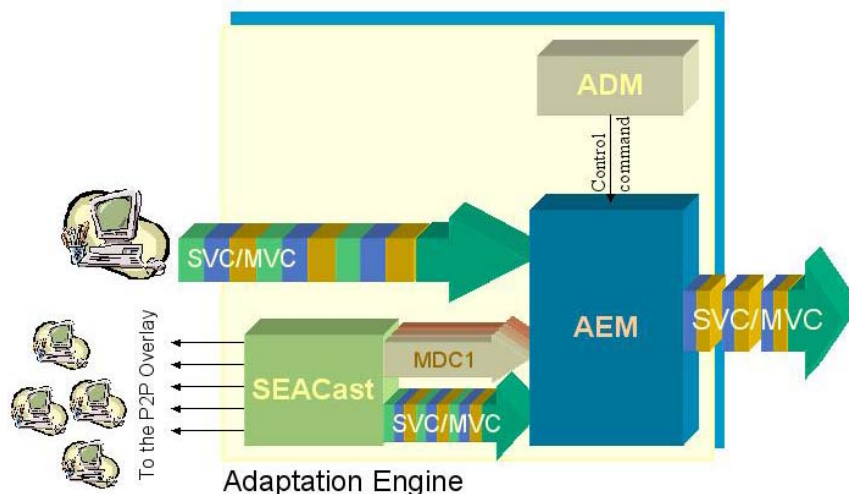


Figure 32: Adaptation Engine software architecture

Figure 32 summarizes the AEM software architecture. The dashed box represents the AEM module.



The VLC is the core of the implementation. It receives data from some server over the Internet (when considering the VoD scenario) or the bit stream from the SEACast module. Depending on the information passed by the ADM, the VLC may adapt the incoming data pruning some of the scalability layers, eliminating some of the transferred views, transmitting the information in layered or in one unique session, reconstructing the MDC session in a unique H.264 bit stream, when incoming MDC sessions are present

The ADM module performs the setup of the VLC. The interface of the ADM with the AEM provides information about setting of the AEM for this specific streaming session. The ADM signals the media format of the incoming data, if the data is encoded in SVC, MVC or MDC format and if it is coming from a VoD connection or from the P2P engine. The ADM signals if the incoming stream should be adapted for the specific terminal. It can specify if some of the scalability layers should be pruned for SVC, if some of the views should be dropped for MVC.

The ADM signals the desired output format for streaming. The output data may be formatted using a layered unicast approach or a unique session. The reason for that is that adaptation in encrypted session may be performed only dropping some layered sessions, so encrypted data may transit through the VLC at this stage.

Data may come from the network in VoD sessions using RTSP/RTP IETF protocols. In this case the VLC should be used as a proxy server. The ADM parses the SDP and extracts information regarding the characteristics of the incoming bit stream. This information, with information coming from the TAM and NAM, will be the basis for the decision.

Data is also coming from the active P2P session in the SEACast. In this case, data is streamed on a local interface from the P2P application to the VLC and here it is adapted according to the terminal capabilities.

### 5.2.2 VLC SW architecture mapping

This section describes the software implementation of the AEM module inside the VLC. Figure 33 describes the architecture we want to implement for the VLC as we envisage it. The VLC software structure is organized in modules that are divided in different families, depending on their functionality. For instance, the *Demux* family demuxes audio/video data in different streams. The *livedotcom.cpp* implementation in the Demux family uses the LiveDotCom libraries to implement the RTSP and RTP session handling.

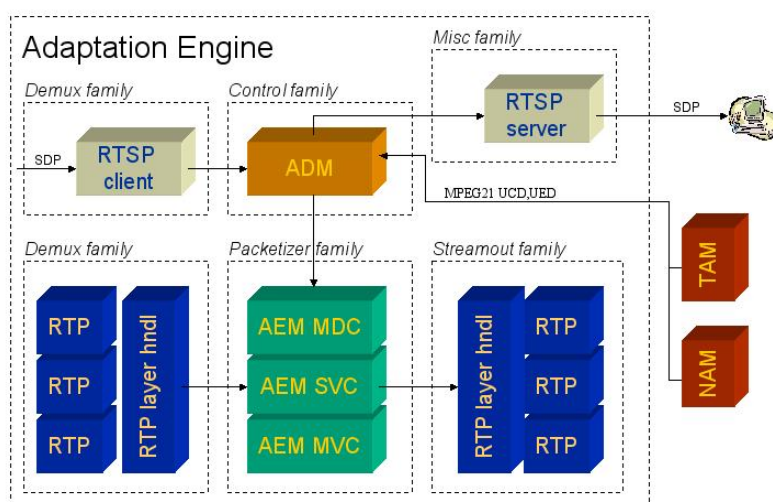


Figure 33: Mapping of modules in VLC architecture



The possibility of analyzing and partitioning the bitstream in the VLC contains many similarities to the goal of a class of objects in VLC called *packetizer*. A packetizer is a module that partitions a flow of data for a specific coding standard. The chunks created by the *packetizer* will be passed to the decoder or the streamer modules. In case of SVC/MVC (and also MDC), each chunk will contain an Access Unit. We can evolve the *packetizer* module for the SVC and implement the *packetizer* module for MVC in such a way that they will be able to understand commands as “*drop a layer*” or “*discard a view*”. Such packetizers will be able to analyze the content of the Access Units and decide for each NALU if it can be kept or discarded.

The ADM can also be implemented in the VLC. This can be an additional module that can be part of the class of objects called *control* in VLC. This module implements functionalities of RTSP proxy in the VLC and may take advantage and extend some existing implementation of RTSP client and server already existing in the original software. When, – the ADM module detects an incoming MDC session (by parsing the SDP), it may activate the MDC decoding module and produce a concealed bitstream

When encrypted sessions are present, scalability layers and views are transmitted in separate sessions. As the data is encrypted, the adaptation is limited to joining/dropping some sessions. The receiving module (in the terminal) should be able to handle the composition of multiple incoming sessions in a single flow of data in the receiving module. This can be done handling multiple incoming sessions in the RTP module through a RTP layered handler that collects information from all the RTP sessions.

### 5.2.3 Scenario mapping

In this session, we are grouping the scenario presented in section 1.2 in sets that are mapped on the same software architecture, from an AEM point of view. We are also referring to the previous Figure 33 to depict how the communication between the different modules is progressing over time.

#### 5.2.3.1 VoD ADM decision on layer/view selection and MDC termination

This section comprises Scenario 1, 2, 3, 8 and 9 as depicted in Section 1.2. In these scenarios, the MG is considered a RTSP proxy server, adapting and forwarding the multimedia session to the following hop in the network.

In scenario 1 and 2, the SDP message is analyzed by the ADM but passes unmodified through the ADM. In particular if the Scalable SEI message is available, a finer optimization may be performed on the link. The adaptation is performed in the following phase, when the RTSP SETUP commands are issued by the terminal, the MG can signal the adaptation issuing an RTSP response with status code 453 “not enough bandwidth” [10] for some layers that are excluded in the adaptation phase (see, [10] sect 7.1.1). This standardized operation informs the terminal that some of the selected sessions may not be streamed at this moment due to network constraints. In scenario 3, the SDP message is modified. The incoming SDP message contains a description of the different descriptions and the IP addresses from where the multimedia data is flowing. In the outgoing message the MG will aggregate the information of the different descriptions, perform some concealment and forward the reconstructed bit stream to the terminal in a single flow. The SDP message should be modified to reflect these changes.

In the following we give a brief description of modifications from a terminal point-of-view. Different modules are involved and require implementation of new functionalities. The current implementation of the VLC does not have functionality to deal with the proxy server for the RTSP protocol. In order to support RTSP we need to change some parameters passed during the communication.

In the current VLC implementation, the SDP generation and interpretation does not comply with the last IETF specifications regarding signaling layered and MDC streaming session [4]. This point regards both the server and the client implementation. We need to modify the RTSP client and server modules to identify MDC sessions and take proper actions to conceal and merge MDC-codec media



sessions. The MVC decoder should be integrated in the terminal and the streaming protocols for RTSP/RTP should be implemented as far as handling of MVC and MDC sessions is concerned.

The MG proxy server implementation starts from scratch. Some of the functionalities – eg. RTSP/RTP client server – are already present, but they should be integrated to form a complete solution, that exchanges information and aggregates it for the ADM decision.

Scenario 8 and 9 are considered for further study and investigation. Scenario 8 can be considered as a special case of scenario 3. Complexity resides in the implementation of the solution more than in the streaming technology or the software architecture and it should fit in the current software architecture. Scenario 9 requires some modifications to the SDP message. The outgoing SDP should contain information regarding the MDC and the channels where the data is flowing. The ADM should start a VLC module that handles the MDC data generation and this module will feed the RTP module with the information on the flowing data.

### 5.2.3.2 P2P session termination

This section comprises Scenario 4, 5, 6 and 7. In these scenarios, the VLC receives data from the SEACast P2P engine and adapts the received data to the terminal and network characteristics. In all these scenarios, the P2P sessions are terminated at the MG and data is forwarded to the final terminal. In all these situations, VLC acts as an adaptation engine.

Both the VLC and the SEACast resides on the same machine (the MG). The transmission between the two elements occurs via a localhost IP port. Information about media characteristics should be exchanged between the ADM and the SEACast.

## 5.3 Interface – Functions Definition

The AEM will export interfaces with ADM, in order to receive information about the format of the incoming media (SVC/MVC/MDC) and the proper command to adapt the bitstream; according the input media format, such commands will include instructions about the layer resp. view to prune or whether a MDC session has to be started or terminated. Finally the ADM will provide to the AEM details about the composition of the output streaming session (layered or single session). In case of P2P streaming, the AEM will be directly connected with the SEACast module. In such situation the AEM will receive the P2P video chunks, and reconstruct them in order to recreate SVC/MVC bitstream; based on the decision taken from the ADM (last network characteristics, terminal requirements and the user preferences/permissions) it will additionally adapt the stream.

Depending on the incoming session and media type, the commands issued by the ADM will compose an instruction set that will make possible the selection of:

- A. the pure base layer of an SVC stream, respectively the base view (2D compliant, AVC compatible subset) of an MVC stream
- B. the complete SVC/MVC media stream,
- C. any decodable SVC/MVC sub-bitstream which are available
- D. the activation of an MDC session,
- E. the termination of the MDC session and the composition of the incoming MDC description as base layer for SVC/MVC

In case of (C), the selection can be based on

- a) fidelity level,
- b) target bitrate,
- c) image resolution,



- d) frame rate,
- e) number of views,
- f) specific set of views,
- g) or any combinations of the above parameters.

In case of (a), it could be convenient to just step up or down to the next level along the fidelity dimension of the scalability cube.

In case of (b), (c), (d), (e), and (f), the selection could be based on numeric values representing the targeted bitrate, image resolution, frame rate, or either the number or a specific set of views, respectively. For SVC streams, the number of views shall always be one, and the parameter which identifies a specific set of views will be ignored. For MVC, the only meaningful parameters will be the number of views and, if applicable, an identifier for a specific view or set of views.

VLC will be one of the main SW frameworks adopted in the SEA project, due to its flexibility and modularity, that make it suitable to extend its capabilities according to the innovative functionality foreseen in the SEA project. Several different objects are defined within the VLC source code, each one implementing a class of functionalities, grouped by similarity (i.e. decoding, displaying, streaming, etc.) and characterized by the same input/output format.

AEM module shares functionalities with many module families inside the VLC. Looking at Figure 33, the AEM can be mapped in the RTP input module for the part regarding layered session forwarding and joining (*Demux* family) and to the *stream\_out* family for RTP output. The part pruning the discarded layers of scalability or views could be mapped in the *packetizer* family. The *packetizer* functions in fact are able to perform editing operations directly on the bitstream, partitioning a flow of data for a specific coding standard. Such functionality can be easily extended providing to the *packetizer* the capability to process each incoming block of data, deciding which part eventually drops, according to the instructions coming from the ADM. Each elementary block of data coming from the network (NAL unit, *Network Abstraction Layer Unit*) will be analyzed, and the crucial information about the layer/view to which the NALU belongs to will be parsed from the NAL unit header.

According to the described functionality of the AEM, there will be mainly two types of interfaces. The first interface deals with the input bit stream coming from the RTP sessions (the *Demux* family of functions in VLC). Here the interface is implicitly already defined by the VLC communication between the two module families. The only difference will be that with layered transmission, the set-up of the RTP sessions might be different, depending if the sessions would be joined or not. This interface is used by the AEM to receive the data stream, incoming from VoD layered unicast sessions, layered multicast, and also from the P2Pcast module. In the latter case – besides adapting the bitstream – the AEM will have the task to properly compose the chunks provided by the SEAcas application. The same interface will be used by the AEM once the adaptation process is terminated to transmit the adapted bitstream to the following modules. In case of incoming layered streaming sessions and depending on the instructions received from ADM, the AEM will be able to compose the layered incoming sessions and provide an output single stream, feeding a single output RTP session.

The second interface regards the commands exchange between the AEM and the ADM. In order to properly adapt the bitstream, the AEM should receive a set of commands and information from the ADM. These will include the possibility to drop layers or views. Such information can be formatted into a unique data structure that will be provided by the ADM. The operation of exchanging data structure among internal modules is quite simple in VLC, as proper functions are available to perform it; it is possible in fact in VLC to browse among the loaded modules, and inspect the allocated data structures. Therefore the *packetizer*, before performing the adaptation operation will be able to check the values of the variables containing the information about the adaptation decision taken by the ADM.



For the ADM to AEM interface, we foresee the following parameters

- the did parameter for SVC bit streams;
- the ql parameter for the SVC bit streams;
- the tl parameter for the SVC bit streams;
- a list of active views for the MVC bit streams;

We describe now how we would handle the different cases. Layered sessions can be unified setting the RTP receiver parameter in the proper mode (unify/layered). MDC decoding can be decided by ADM, calling the plug-in for decoding MDC streams. SVC/MVC adaptation can be decided by the ADM as it activates the appropriate packetizer (either the one dedicated to the SVC or the one dedicated to the MVC) and sends the previously defined parameters, signaling which layer resp. view to keep, to that packetizer. Peer-to-peer cases will be handled as normal SVC/MVC sessions at this stage.

The AEM will feedback the ADM about the status of the open RTP connections. AEM receives RTCP Sender Reports (and also feedback the sender with its own RTCP Receiver Reports). The reports are important to understand the session statistics inside the link (as network may provide only aggregated data on the link status). The ADM can use the provided information to take actions to provide the best overall quality to all session on the specific link.

We foresee the following parameters exchange, based on the RTCP Sender Report information received from the previous Adaptation Engine

- *sender packet count*, total number of RTP data packets transmitted since starting transmission;
- *sender octet count*, total number of payload data transmitted in the RTP session;
- *fraction lost*, the fraction of lost RTP data packets collected from previously sent Receiver Reports;
- *cumulative number of packet lost*, the total number of RTP data packets lost since the beginning of transmission;
- *inter-arrival jitter*, statistical variance of the RTP data packet inter-arrival time;

These parameters will be passed to the ADM using VLC internal variables. They are used by the ADM to match the current media transmission statistics with the network link conditions and take actions to improve the quality when required.

In case of layered RTP data transmission, the single RTP sessions may be interrogated separately and provide single session statistics.



## 6 Network Awareness Module (NAM)

The Network Awareness Module (NAM) provides measurements and has knowledge of the physical characteristics of the network.

### 6.1 Module Functionality

The NAM module will be introduced in the sHMG, the sNMG and optionally in the terminals, including mobile networks to sense characteristics of the underlying networks connected to them. The NAM module can further process these low-level and network specific information into performance indicators of the network that are more generic and simpler for the ADM to use. This allows the ADM to be technology-independent and hides low-level technical aspect of the access networks within the NAM instead.

Each different network would require a specialized NAM module. For example, in case of in-home network video delivery, a number of NAM modules located at the sHMG will retrieve information associated with the access network (e.g. an ADSL NAM and a DVB-S NAM) and the in-home networks (e.g. a WiFi NAM). Similarly, a terminal in a mobile network, which can be a simple mobile terminal or a highly-capable terminal such as a laptop would have at least one NAM for the access technology it is connected to. In case it is a multi-mode terminal capable of roaming between several technologies, there would be the same number of NAM modules at the terminal as the number of technologies it supports.

Based on the underlying network characteristics and the network administration policy, NAM may be able to retrieve the following statistics and information from the network.

- the available QoS classes
- the packet loss
- the packet error rate
- the bit error rate (BER)
- the DiffServ Code Point
- the Handoff information
- the number of retransmission attempts
- the Traffic classes
- the signal strength
- the capability profile

Note that not all information from all the connected networks is important for the adaptation in a specific direction. As an example, if the node is the sHMG which is connected to the home terminal via a WLAN, information of the WLAN provided by the NAM at the sHMG would be reception quality and network conditions the sHMG is experiencing. This would be more important for uplink traffic than for downlink traffic. Therefore, the adaptation of a downlink streaming service would have to rely on network information from the NAM at the terminal instead, but not the NAM at the sHMG. This means the ADM has to decide which information it would use for adaptation in which direction by itself.

In addition, the NAMs of the inactive mobile networks, e.g. in case the terminal is multimodal and is not using some of its network interfaces, are likely to be inactive also. This is due to the fact that most of the low-level measurements of the radio channel are done when the terminal is in active state.

In general, the NAM will act like an entity that collects network characteristics from several layers and translates them into a meaningful representation as listed in 6.3 to the ADM.



## 6.2 Software Architecture

A general architecture of the NAM in both the media gateway and the terminal is shown in the following figure. There is a NAM responsible for each network the node is connected to. The interfaces between the NAM, the MAC and Physical layers are shown for generality. In practice, it is not necessary to have both but it depends on the network the node is connected to and whether any useful low-level information can be obtained from any of these layers.

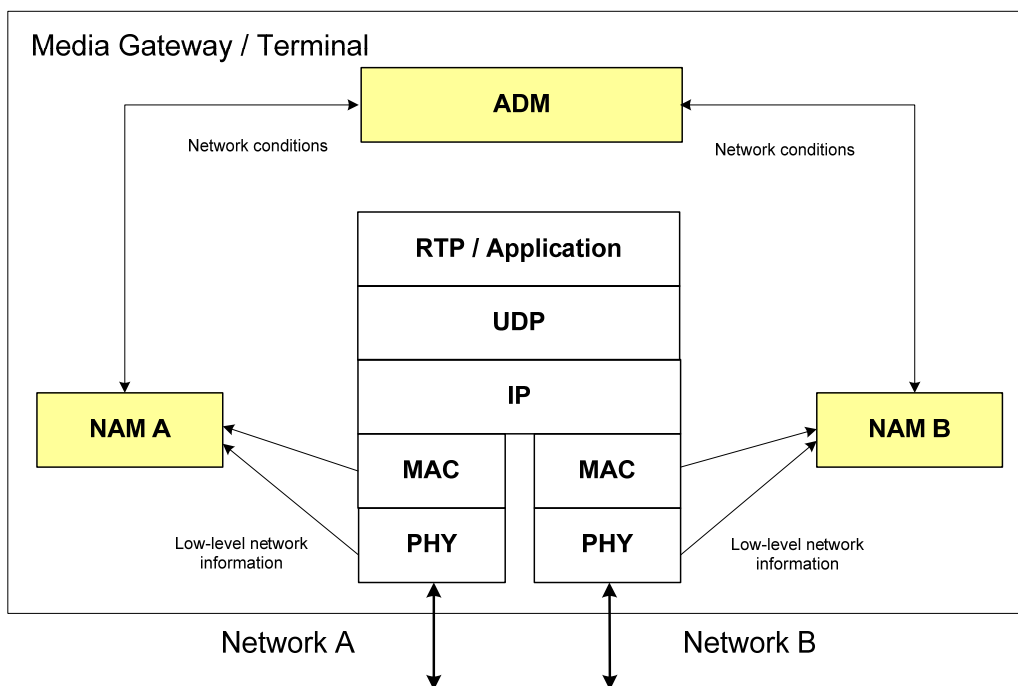


Figure 34: NAM software architecture in a media gateway / terminal

In the case the node is a terminal in a radio access network, a NAM in the terminal would be sufficient to obtain all necessary/available/accessible information regarding the Handover decision and channel characteristics from its measurement reports without having to implement additional NAM modules in the network itself. The NAM will use inputs from only 2 layers of the terminal’s protocol stack, the MAC and Application layers. Inputs from the MAC layer are channel measurement reports to be sent to the network. The NAM can therefore predict future Handover that the network will make based on this report by using the same decision criteria employed by the network in addition to monitoring channel’s quality. In the case that the terminal makes the Handover decision by itself as in several inter-system Handover cases, this NAM can then use this decision instead without making predictions. Other information obtainable from these two layers are the QoS level, number of retransmission, etc. More information regarding the NAM’s interactions with these two layers and low-level inputs to the NAM can be found in [3].

## 6.3 Interface – Functions Definition

The interfaces for the NAM as shown in the last figure are between the NAM, the ADM, the MAC and Physical layers of the hosting node. However, information regarding the interfaces from NAM to the MAC and Physical layers can be found in [3]. The following discussion will be the interface from the NAM to the ADM and the information contained therein.



Note that this interface is local interfaces between NAM and ADM modules within the same physical node and are implementation specific. It is to be a proprietary implementation based on UED / UCD of the MPEG 21 DIA standards.

### 6.3.1 Interface from the NAM to the ADM

With regard to the NAM of the mobile radio networks as in [3], these NAMs can provide the following generic performance parameters of the underlying networks;

- A. Semi-Static Information** : This type of information would be provided to the ADM during the session setup and are likely to be static throughout the session. Nevertheless, if there are changes to these parameters, e.g when the terminal changes from one network to another, the NAM can also notify the ADM about the changes during the session.
- Maximum network throughput
  - Minimum guaranteed throughput
  - Guaranteed packet delay variation (delay jitter)
  - Guaranteed packet delay
  - Guaranteed packet loss rate
- B. Aperiodic Information** : This type of information would be provided to the ADM in aperiodic manner, depending on the nature of the information.
- **Generic metric for channel condition** : This is a generic metric to indicate the quality of the current radio channel from 0 to 100 with 100 being the best possible channel condition for the terminal. This metric is not meant to be used to compare the channel conditions from different networks nor terminals with different capabilities. It is derived from low-level measurements, e.g. SINR, BER and terminal-specific capabilities. Hence, a low-capability terminal with 100 on the score may be able to handle lower throughput to another high-capability terminal with also 100 on the score.  
  
The intended use of this metric is to imply the channel conditions of that specific terminal in relation to its worst and best conditions it can handle.  
  
The ADM could set the thresholds for the NAM to report whenever this metric drops below or above the interested points, thus implying degrading or improving channel conditions for the terminal
  - **Handover Notification** : This is a notification from the NAM to the ADM regarding a possible Handover, including the type whether it is an intra or inter-system Handover in the near future.

Apart from the NAMs in mobile networks, other NAMs responsible for other access networks, e.g. DSL or WLAN should provide similar generic network parameters. Parameters such as the generalized metric for the channel conditions from the WLAN should be made possible, for example.

### 6.3.2 Interface from the ADM to NAM

This interface is to allow the ADM to configure and command the NAM as follows.

- A. Configuration of the Thresholds** : This is to configure the reporting threshold values of the generic metric for channel condition measured by the NAM in a mobile terminal in 6.3.1.
- B. Report Request Command** : This is used to command the NAM to report all its measurements to the ADM.



## 7 Terminal Awareness Module (TAM)

Similarly to the Network Awareness Module, the Terminal Awareness Module (TAM) has knowledge of the physical characteristics of the terminal. The TAM provide a functionality similar to the Terminal Capabilities information located inside the Usage Environment Description (UED) of MPEG-21. It provides description of the terminal capabilities in terms of power characteristics (average consumption, battery capacity etc.), processing constraints, codec capabilities, device properties (which include display and audio output capabilities), the average/maximum input/output bit-rate (as a function of the processor capabilities).

### 7.1 Module Functionality

#### 7.1.1 Plurality of terminals

Mobile terminals will typically vary in screen resolution, available codecs, processing power, battery life. In general laptops have screen sizes 14"- 15" and resolutions of 1024x768 pixels and upwards, sub-notebooks have screen sizes of 7"-13" and resolutions of 800x480 and upwards, PDAs have screen sizes of 3" and resolutions of 240x320 and upwards, and smart-phones are alike PDAs.

Computational power is likewise more limited in PDAs and smart-phones and more available in laptops and can range from several hundred MHz to several GHz on multiple cores. However taking a measurement of the computational load at the time of the session startup is in itself not valuable information, as computational load fluctuates over time based on the number of tasks the user is performing or finalizing and the tasks the system is performing autonomously.

Mobile terminals have the characteristic of being battery powered and having a specific battery life, apart from PDAs and smart-phones, laptops are also used while connected to its AC charger. The value of knowing the remaining battery life of the terminal at session startup is in itself low, since the impact of the increase of energy consumption in the terminal will be different between terminals. Let us consider both a laptop and a PDA with 1 hour battery life remaining. When a computational intensive task is started on both devices, the rate of decrease in battery life will be different. Besides this aspect, the computational load at the time of the session startup has an influence on the battery lifetime indication at that time.

#### 7.1.2 List of parameters

The TAM may be able to retrieve user and terminal information and statistics, such as:

- Display
- Multiple Network interfaces
- Simultaneous Active interfaces
- Processing power
- Coding capabilities
- Decoding capabilities
- MDC capability
- CPU load
- Battery life
- Free storage space
- General User preferences
- Specific User preferences

Moreover, it may be able to extract network conditions at specific interfaces e.g. packet loss information, coverage, SNR, BER, etc.

#### 7.1.3 Logical grouping

Parameters can be grouped in three groups: Static characteristics, Measurements, and User Preferences.



**Static characteristics** are the fixed capabilities of the terminal, such as the codec capabilities: e.g. the capability to process streams such as H.264 SVC or H.264 MVC. Further static characteristics are input-output capabilities, such as the screen size, display resolution, and audio output capabilities, and the static network capabilities. Other properties that fall into the static characteristics group can be power-related attributes, such as full battery capacity, storage capacity, such as total storage size and medium, and data I/O characteristics, such as the audio and video buffer sizes. Typically the static characteristics are determined once and do not change over time.

**Measurements** are representations of the changes of the terminal over time. The conditions that are changing can be CPU load, free storage space, battery life remaining, whether the device is connected to a charger, and network conditions. However, the measurements will need a layer of abstraction to be meaningful for the ADM, i.e. they are terminal dependant.

**User Preferences** provide a means to the user of the terminal to influence the adaptation process in such a way that the viewing experience is optimized for that user. Possible user preferences can be whether or not to view videos full-screen, whether or not the aspect ratio must be maintained, or preference for a specific AV quality.

#### 7.1.4 Aggregating into usable concepts

The Static Characteristics, Measurements and User Preference information available at the TAM can be aggregated into more usable concepts for the ADM to base the adaptation decisions on. These concepts are the **Performance Index** and **Optimizing for User Preference**.

The Performance Index provides a means to aggregate CPU load, processing power, and battery power, into a more generic concept. This will provide the ADM a terminal independent indication of whether or not the terminal is operating at its maximum performance. This concept is FFS.

The optimization for User Preference concept can be best explained by the use of two scenarios.

*In the first scenario John is using his PDA to watch a 1 hour movie. At the session startup, the battery life remaining amongst other static terminal characteristics and measurements resulted in the highest quality stream being sent to the terminal. However, this highest quality stream also results in a fast decrease in remaining battery life. After 30 minutes the battery life remaining has dropped significantly, to the point that it becomes clear that it will not be possible to watch the remaining 30 minutes in this quality.*

*John has chosen to Optimize for Quality. This means that he will watch the remainder of the movie in the quality that was initiated, until the battery runs out. Rather than watching the movie until the end, John rather watches as much of the movie in the highest quality and finish watching the movie at a later moment, e.g. after recharging the PDA.*

*If John had chosen to Optimize for Battery life, the adaptation mechanism can take this user preference into account, knowing that John would rather watch the entire movie from start to end, with less regard to the quality in which it is viewed. This could mean that at the session startup a specific AV encoding or network interface is chosen that less power hungry at the terminal, or that a trigger is sent during playback to the ADM that there is need to reconsider the adaptation to be able to finish watching the stream.*

The benefit for the user is that by using the tradeoff between Quality and Battery life, he or she can have an influence on the adaptation process, which results in a better viewing experience.

The second scenario is similar to the first scenario; however the tradeoff is now between audio quality and video quality.

*John wants to watch a stream with the latest news on his PDA. In this case he has indicated on his terminal that his preference is to have best quality audio, rather than having the best quality of video.*



The difference in the second scenario is that the choice here is not between best quality audio with best effort video and best quality video with best effort audio. In-between these extremes is the option to have both best quality audio and best quality video, as audio and video quality are not mutually exclusive as in the case of performance and battery life in scenario one.

## 7.2 Software Architecture for the TAM in the Home Environment

The architecture of the TAM would be like as shown in the following figure.

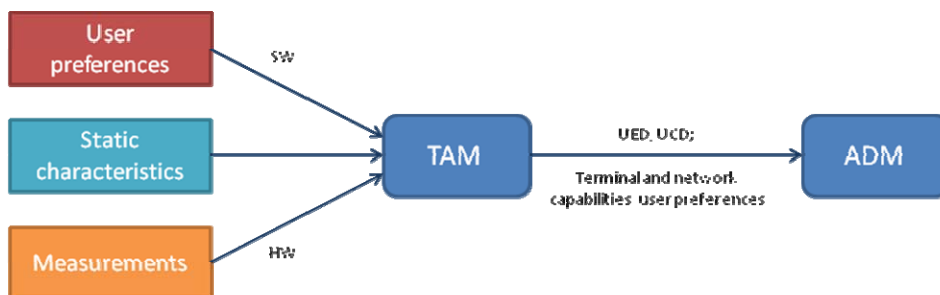


Figure 35: TAM Architecture

The TAM collects characteristics of the terminal over three categories. The TAM will act like a module that collects these terminal characteristics and translate them into a standardized and meaningful representation to the ADM by means of using the UED and UCD of MPEG 21.

## 7.3 Interface – Functions Definition

The TAM provides the physical characteristics of the terminal. The input interface of TAM is labeled (Term, TAM) in the figure below. The presence of such interface depends on the actual terminal. The TAM exhibits an output interface (TAM, ADM) towards its own ADM; (TAM, ADM) conveys MPEG-21 UED, UCD metadata in the form of XML files.

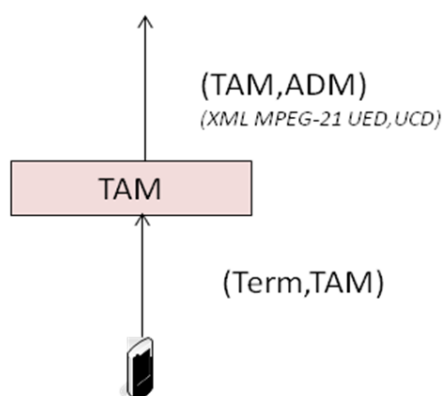


Figure 36: TAM interfaces



## 8 Content Storage Module & P2P Awareness Module

As it has been described in [2], SEA has introduced a module, called SEAcast, which is responsible for the peer-to-peer communications. The SEA network nodes consist of a storage device, namely the Content Storage Module (CSM) that enables the temporary caching or the more permanent storage of content material. These storage devices allow for the peer-to-peer sharing of streaming video, in combination with the special module of SEAcast which is exactly responsible for the management of the P2P network overlay. SEAcast module comprises a core module and a network awareness module, the SEAcast-NAM or S-NAM.

### 8.1 Module Functionality

The Content Storage Module (CSM) will store and/or cache the A/V content segments (layers, views, descriptions, chunks). These storage media may be located either in the network, inside the sNMG as network storage device, or in the extended home environment, inside the sHMG. In the case where the terminal has adequate processing capabilities (e.g. a PC) so that the sHMG node resides inside the terminal, CSM can simply be thought as a PC hard disk.

It may act as an A/V server or a peer node in a P2P delivery environment supporting on-the-fly content adaptation which will be implemented by the AEM. By utilizing distributed CSM modules, SEA aims to lower the delay and the bandwidth consumed in buffer maps exchanged between the peers, improve the quality of the video content, and cope with a huge number of simultaneous users.

Moreover, a significant gain is expected if the video data is not considered as ordinary data chunks, but organized in a succession of descriptions and/or layers, optimized for video playout (ref. Scenario 7). In fact, in critical transmission situations, e.g. a streaming node disconnects suddenly or thousands of users request the same content simultaneously, the peers may resort to lower quality, skipping refinement layer packets, but avoiding service break down.

### 8.2 Software Architecture

Figure 37 shows the architecture of the sHMG with an emphasis on the CSM and the SEAcast modules. CSM inside the sHMG is partitioned in two storage units. One partition is allocated to the user where personal or private content can be stored. When a user wants to distribute some personal content, acting as a content issuer, he or she keeps that item stored in the CSM to be available to a qualified requestor. The locally available content is made known to the license issuer, during the process of content encryption (ref. to D5.1 about the concepts of encrypted content management); through this license server, other users can be informed about the availability of the content and the accompanying usage rights.

The other partition, called herein the *Community Data* storage, is allocated for temporally caching the streaming videos delivered to the sHMG. This feature promotes the efficient distribution of an A/V content, both in the home environment (Home-LAN) as well as in the P2P network overlay. In the first case, when different members of the same LAN group wish to view the same content at different terminals, the content gets downloaded once and stored inside the CSM temporal cache, from which it afterwards is distributed to the requesting end-users. This is a noteworthy saving of available network resources, when compared with the alternative of each requestor having initiated a distinct session with the content provider.

In the latter case, content segments (i.e. layers, views, descriptions) that are temporarily buffered in the Community Data CSM, can contribute in a faster P2P transfer mechanism established between



peer clients. This mechanism is controlled by the SEAcast module embedded in the MG, as already mentioned and detailed below.

The SEAcast core module is devoted to the implementation of all the enhanced P2P functionalities, i.e. overlay network management, exchange of SVC/MDC/MVC encoded contents. This contains another special module, S-NAM, which is aware of the P2P network. S-NAM is dedicated to the collection of information about the status of the application-layer overlay network. Such information will possibly consist of: the number of and reliability of peers participating in the swarm, the quality of the contents owned by the peers (in terms of SVC/MVC layers or descriptions) the aggregate bandwidth, etc. This information is passed to the ADM module from which proper adaptation decisions can be derived.

When sHMG acts as a content publisher, the procedure flow starts by the owner of sHMG uploading and storing inside the CSM partition of Private Data the items to be published. These can be stored either encrypted or not. In the first case, the Entitlement Control Message (ECM, ref. D5.1) which corresponds to that particular encrypted item is also stored in pair. The successive procedure follows the normal functions of content adaptation control, as seen already. A description of the network's environment interfaces ADM, so that adaptation decisions can be extracted. As an example, if NAM reports high traffic in the uplink direction, ADM will decide to lessen the bandwidth allocated for streaming out the personal content. This decision may translate to any of the adaptation mechanisms examined so far (e.g. drop of enhancement layers) and is executed in AEM.

In the case where sHMG/sNMG requests some content from a P2P network, the SEAcast module finds the available sources of retrieval. To achieve this each node is publishing the available content over the SEAcast overlay, either stored data or live video. All the other nodes in the network access such content in real-time with a simple buffering scheme (in current version of SEAcast buffering amounts to 5 s). This information together with the parameters coming from NAM and TAM characterizing the conditions of the requestor's network connection and the terminal capabilities or personal preferences and user profile, respectively, are input to the ADM in order to decide on adaptation actions. These decisions consist of selecting the number and type of SVC/MVC layers or descriptions from the chosen peers. Thus, the requestor's ADM decisions should be communicated to the ADMs of the selected peers, where, based on the current conditions of their network connections, the final adaptation outcome will dictate which of each peer's available segments will be streamed to the requestor.

As regards the architecture and function of the sNMG, with respect to the storage of content and the mechanisms for P2P management, these will be similar to what described about sHMG, with the only exception that sNMG can not act as a content provider server. Therefore, its CSM can be regarded as a network storage medium that can only cache content providers' items transferred through the network, on a temporary basis. This way, the sNMG contributes to the distributed dispensation of the same content, therefore enabling a more efficient balanced traffic which offloads certain routing paths from getting heavily congested, as might have, should a server-client delivery model had been employed. The advantages of this distributed content acquisition become more evident in the scenario where a large number of users request to view the same content simultaneously (e.g. in a broadcast programme) or over a short period of time (e.g. a VoD case). Consequently, it is sensible to select the frequency of requests that any particular content receives as the criteria that will determine which content items will be temporarily cached inside sNMG. Counters implemented inside the CSM can keep track of the number of times that any stored content gets requested, which will dictate the permanence of the content in the storage space. This effectively will promote the faster streaming of most popular content.

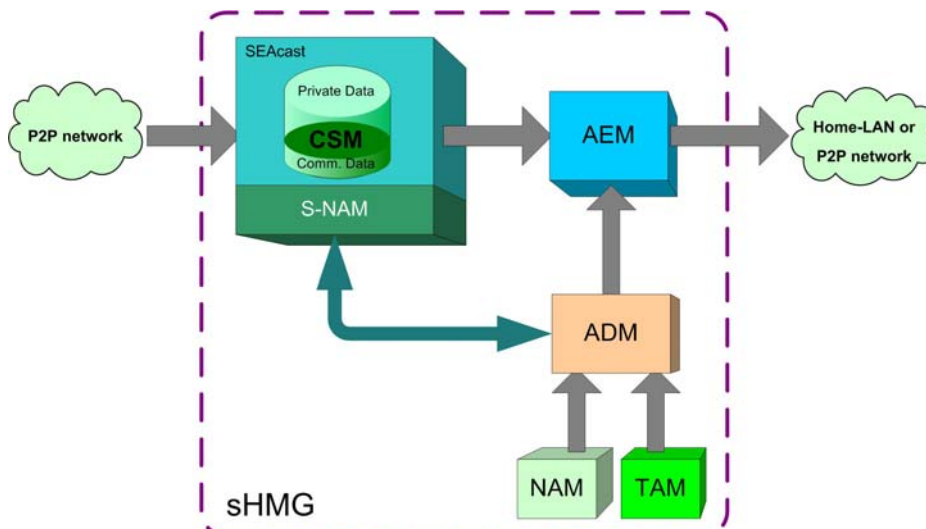


Figure 37: CSM and SEAcast modules in sHMG

### 8.3 Interface – Functions Definition

CSM should basically have the functions and qualities of a data container in a modern storage device. CSM should offer fast indexing of all content items kept in it. A request for retrieving any stored item will return a handle by which that item can be manipulated by the invoking module. Further, the storage space of caching temporary content should provide a self-organizing disk management mechanism. This is purposeful since a volume of digital items, issued from different content publishers, will be kept in a storage medium which obviously will have not unlimited size. The management of available free storage space will be determined by the permanence of the stored files, which will be based on criteria, like for instance, the age of file storages, the popularity of file requests, etc.



## 9 Conclusions

This document begins with the general concepts and objectives of the SEA project; that is, to enable the seamless delivery of multimedia content with enhanced peer-to-peer communications, cross-layer control and adaptation of the content along with content protection and rights management over the whole distribution chain and heterogeneous networks. Following, the SEA logical network architecture, including the added sNMG and sHMG nodes, their interfaces to different networks, their functionalities and mapping to the SAE architecture, has been introduced and explained in more details. This logical network has been designed to achieve that the stated objectives are in a practical, innovative and efficient manner and can be well integrated with currently existing and foreseen network technologies.

The SEA peer-to-peer concept and guidelines for further steps have been described in chapter 3. It introduces the SEAcast module and identifies modifications needed for the current P2P protocols to support SVC, MDC and MVC content management.

The SEAcast module is to be implemented at the sHMG and sNMG. This allows content aggregation to be done at the home network level (in case of the sHMG) and at the network level (in case of the sNMG) which results in improved overall bandwidth utilization. The SEAcast module will work in coordination with the adaptation modules also residing in those network nodes to efficiently determine suitable actions related to adaptation and resilience of the content.

The SEA cross-layer adaptation architecture has been discussed along with related functional modules. The ADM and AEM are both to be placed at the sHMG and sNMG. Working in coordination with the TAM, NAM and the SEAcast module, the ADM uses knowledge of both static and dynamic information of the terminals, the underlying networks and the overlaying peer-to-peer network to make decision for adaptation of the content accordingly. The AEM then performs actual adaptation of the content by removing or adding enhancement layers, descriptions and viewpoints.

The NAM will be implemented at the sHMG, sNMG and at the terminals. In addition, it is also speculated that having the NAM in the mobile radio networks would also be beneficial to the adaptation modules since some parameters useful for the adaptation that are not available at the terminal or at the sNMG could be obtained there.

For the aspect of content protection and rights management, SEA intends to investigate the feasibility of adopting existing content protection standards such as the MPEG-21 along with the ISMACryp as a major encrypting scheme to be employed at the content provider. SEA's approach is that access to the content should be requested first and the license is permitted with usage guidelines for the users. If one desires to use the content under different conditions than in the given usage terms, he or she has to renegotiate with the License Server for new permission.

Additional functional nodes related to the content protection aspect for SEA are to be mainly implemented at the content providers. These are the Permission Manager, Authentication Server, License Conductor and License Server while the terminals are only required to have a Digital Rights Management (DRM) Proxy implemented.



## 10 References

- [1] SEA Consortium, Deliverable 2.1 “Service Requirement Specifications”, March 2008, [http://www.ist-sea.eu/Public/SEA\\_D2.1\\_VOD\\_FF\\_20080414.pdf](http://www.ist-sea.eu/Public/SEA_D2.1_VOD_FF_20080414.pdf)
- [2] SEA Consortium, Deliverable 2.2 “Architecture Specification”, July 2008, [http://www.ist-sea.eu/Public/SEA\\_D2.2\\_NOM\\_FF\\_20080711.pdf](http://www.ist-sea.eu/Public/SEA_D2.2_NOM_FF_20080711.pdf)
- [3] SEA Consortium, Deliverable 2.3 “Cross Layer Interfaces Specification” September 2008, [http://www-iset-sea.eu/Public/SEA\\_D2.3\\_POL\\_FF\\_20080930.pdf](http://www-iset-sea.eu/Public/SEA_D2.3_POL_FF_20080930.pdf)
- [4] MPEG-21, “Digital Item Adaptation, FCD”, ISO/IEC JTC1/SG29/WG11 N5845
- [5] I.S. Burnett, S.J. Davis, G.M. Drury, "MPEG-21 Digital Item Declaration and Identification—Principles and Compression", IEEE Transactions on Multimedia, vol. 7, no. 3, pp. 400–407, June 2005
- [6] ISO/IEC 21000-10:2006, Information technology — Multimedia framework (MPEG-21) — Part 10: Digital Item Processing, 2006.
- [7] ISO/IEC 21000-4:2006, Information technology — Multimedia framework (MPEG-21) — Part 4: Intellectual Property Management and Protection Components, 2006.
- [8] IEEE 802.16e, Part 16: “Air Interface for Fixed and Mobile Broadband Wireless Access Systems”, Amendment 2 “Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands”
- [9] ISO/IEC 21000-7:2007, Information technology — Multimedia framework (MPEG-21) — Part 7 Digital Item Adaptation
- [10] H. Schulzrinne, A. Rao, R. Lanphier: "Real Time Streaming Protocol (RTSP)", IETF RFC 2326, April 1998, <http://tools.ietf.org/html/rfc2326>
- [11] L. Choi, W. Kellerer, and E. Steinbach, “On Cross-Layer Design for Streaming Video Delivery in Multiuser Wireless Environments”, EURASIP Journal on Wireless Communications and Networking, vol. 2006, pp. 1-10, 2006.
- [12] M. Handly, V. Jacobson and C.Perkins, “SDP: Session Description Protocol”, IETF RFC 4566, July 2006, <http://tools.ietf.org/html/rfc4566>
- [13] H. Schulzrinne, A. Rao, R. Lanphier: "Real Time Streaming Protocol (RTSP)", IETF RFC 2326, April 1998, <http://tools.ietf.org/html/rfc2326>
- [14] M. Handley, C. Perkins, E. Whelan: Session Announcement Protocol", IETF RFC 2974, October 2000, <http://tools.ietf.org/html/rfc2974>
- [15] H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler: SIP: Session Initiation Protocol", IETF RFC 3261, June 2002, <http://tools.ietf.org/html/rfc3261>
- [16] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, IETF STD 0064, RFC 3550, July 2003, <http://tools.ietf.org/html/rfc3550>
- [17] T. Schierl, S. Wenger, Signaling media decoding dependency in Session Description Protocol (SDP), <http://tools.ietf.org/html/draft-ietf-mmusic-decoding-dependency-03>, 25 Sep 2008
- [18] Q. Li and M. van der Schaar, “Providing adaptive QoS to layered video over wireless local area networks through real-time retry limit adaptation”, IEEE Transactions on Multimedia, vol. 6, no. 2, pp. 278-290, April 2004.
- [19] Micheal Eberhard, Luca Celetto, Christian Timmerer, Emanuele Quacchio, Hermann Hellwagner, “An Interoperable Streaming Framework for Scalable Video Coding based on



- MPEG21”, Proc. of the 5<sup>th</sup> International Conference on Visual Engineering VIE, Xi’an, China, August 2008
- [20] D. Mukherjee, E. Delfosse, J.-G. Kim and Y. Wang, “Optimal Adaptation Decision-Taking and Network Quality-of-Service”, *IEEE trans. On Multimedia*, vol. 7, no. 3, Jun 2005
- [21] T.Schierl, S. Wenger, “Signaling media decoding dependency in Session Description Protocol (SDP),” IETF Draft, May 25, 2008, <http://tools.ietf.org/html/draft-ietf-mmusic-decoding-dependency-02>