DVB Interim Specification

IP Datacast Baseline Specification;

Specification of Interface I_MT

DVB Document A080

April 2004

Contents

Introduction 4
1 Scope 5
2 References 5
3 Definitions and abbreviations 7
3.1 Definitions 7
3.2 Abbreviations 9
4 Types of terminals 10
4.1 Terminal Type I 10
4.1.1 Application Platform services for Applications 10
4.1.2 Terminal services for Application Platform 10
4.1.2.1 Broadcast channel 10
4.1.2.2 Interactivity channel 10
4.2 Terminal Type II 10
4.3 Terminal Type III 10
4.3.1 Application Platform services for Applications 10
4.3.2 Terminal services for Application Platform 11
4.3.2.1 Broadcast channel 11
4.3.2.2 Interactivity channel 11
5 Services available to the terminal applications 12
5.1 Service discovery 12
5.1.1 Scope 12
5.1.1.1 Categories 13
5.1.1.2 Service sets 13
5.1.1.3 Bundles 13
5.1.1.4 Services 13
5.1.1.5 Service sessions 13
5.1.1.6 Service components 13
5.1.2 Logical structure of the service discovery channels 14
5.1.2.1 Hierarchy channel 15
5.1.2.2 Session channel 15
5.1.2.3 Main page channel 15
5.1.2.4 Security association channel 15
5.1.2.5 Notification channel 16
5.1.2.6 Dynamics 16
5.1.3 Transport of announcement files in Service Discovery Channel 17
5.1.3.1 Option 1: SAP 17
5.1.3.2 Option 2: DVBSTP 17
5.1.3.3 Option 3: WFDP 18
5.1.3.4 Option 4: FLUTE (Recommended transport protocol as a working assumption) 18
5.1.4 Descriptions and formats of announcement files in Service Discovery Channel 19
5.1.4.1 Hierarchy information 19
5.1.4.2 Session information 25
5.1.4.3 Main page channel 29
5.1.4.4 Security association channel 29
5.2 Service access and Service key delivery 31
5.2.1 Overview of Operation 31
5.2.2 Service access scenarios 32
5.2.3 Service access protocols in the interaction channel 33
5.2.3.1 SMS –based Service Access 33
5.2.3.2 HTTP(S) –based Service Access 34
5.2.4 First key delivery 34
5.2.5 Re-keying 34
5.3 Service interaction 35
5.3.1 Scope 35

5.3.2 Protocols required to support interaction with remote application server 35
5.4 Audio/Video streaming 36
5.5 Push file delivery 36

History 37

Introduction

As defined in [1] the I_MT is a logical interface representing the collection of interfaces between the mediation platform and the mobile terminal.

The current specification of Interface I_MT aims to provide a common framework for IP Datacasting over DVB-T/H and co-operating cellular networks. Current document is non-normative and is meant to provide implementation guidelines for early trials and pilot projects, in particular.

The further specification of Interface I_MT will be part of the effort to convert "Commercial Requirements, IP Datacast in DVB-H" [2] to technical specifications.

The current document will be superseded by ETSI standards for IPDC which are expected to be developed during 2004.

1 Scope

The present document specifies the interface I_MT as defined in [1]. I_MT is the interface between the mediation platform and the mobile terminals. For convenience, the DVB-UMTS system model [1] is shown in Figure 1.

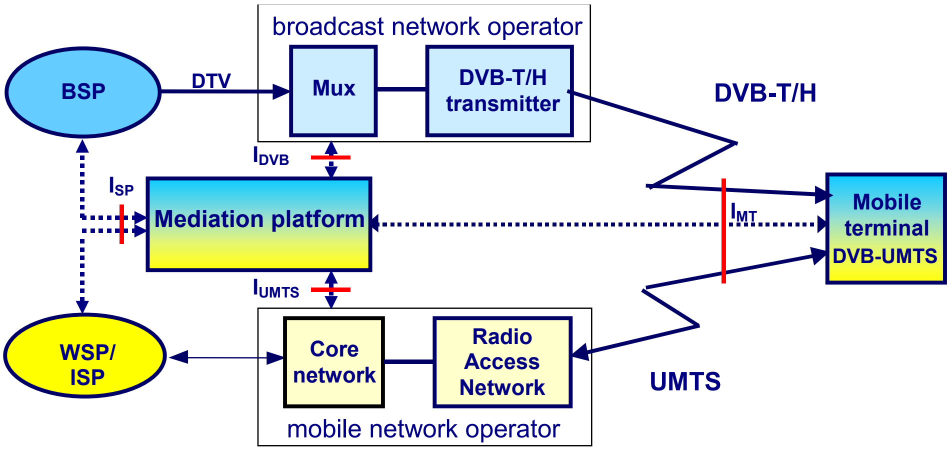

Figure 1: Mediation platform concept and interface I_MT in its context.

In Section 4 we describe different terminal classes of DVB systems. The I_MT interface specification has been arranged according to a number of services that the interface I_MT shall offer to applications running in the terminals to access communication services. These services have been detailed in Section 5.

The scope of this “Specification of Interface I_MT” is in the first hand to enable common direction to pilots and technical trials for IP Datacast over DVB-T/H. Both DVB-T and DVB-H will be assumed to be supported. In the context of this document we consider the use of Terminal type III only (as defined in Sect. 4.3). We note that DVB-H is more suitable for Terminal type III than DVB-T.

It is especially important to note that due to the scope limited to non-commercial systems Interface I_MT specifications in this document are non-normative. However, whenever possible, specifications should aim to be applicable to IP Datacast over DVB-T/H system solutions in the future.

In this document, unless specified otherwise, UMTS and GSM/GPRS are interchangeable. Furthermore, IP means IPv4 or IPv6. However, working assumption is that IPv6 is highly recommended for DVB-T/H broadcast channel in the air interface.

2 References

For the purposes of the present document , the following references apply:

[1] DVB TM2784 Rev. 2.: “Digital Video Broadcasting (DVB); Delivery sub-system for DVB and UMTS systems ”.

[2] DVB TM2967: “Commercial Requirements, IP Datacast in DVB-H.”

[3] ETSI TS 102 812: “Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification”.

[4] ETSI TR 101 154: “DVB Implementation Guidelines for the use of MPEG-2 Systems, Video and Audio in Satellite, Cable and Terrestrial Broadcasting Applications”.

[5] ETSI EN 300 744: “Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television”.

[6] ETSI EN 300 468: “Digital broadcasting systems for television, sound and data services; Specification for Service Information (SI) in Digital Video Broadcasting (DVB) systems”.

[7] ES 202 218 (See also DVB-TM2786 Rev. 2): “DVB-UMTS Interactive Channel Through the General Packet Radio Service (GPRS)”.

[8] DVB TM3037: Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB H).

[9] DVB A079 (TM3025Rev.1): “IP Datacast Baseline Specification: PSI/SI guidelines for IPDC DVB-T/H Systems”.

[10] ETSI EN 301 192: ”DVB Specification for Data Broadcasting”.

[11] IETF RFC 2327: “SDP – Session Description Protocol”.

[12] IETF RFC 2974: “SAP – Session Announcement Protocol”.

[13] DVB TM3022: IPI: DVB-IP Phase 1 Handbook. DVB SD&S Transport Protocol (DVBSTP) for Multicast Delivery of SD&S Information.

[14] IETF Internet Draft, Expired, RMT Group: "Versatile File Delivery Protocol, a Nack-based reliable multicast file transfer Protocol Instantiation", draft-richon-vfdp-protocol-00.txt, 6.12.2001. Available from

http://www.potaroo.net/ietf/idref/draft-richon-vfdp-protocol/ or

http://www.watersprings.org/pub/id/draft-richon-vfdp-protocol-00.txt

[15] IETF RFC 3450: "Asynchronous Layered Coding (ALC) Protocol Instantiation",

http://www.ietf.org/rfc/rfc3450.txt

[16] IETF RFC 3451: "Layered Coding Transport (LCT) Building Block",

http://www.ietf.org/rfc/rfc3451.txt

[17] IETF RFC 3452: "Forward Error Correction (FEC) Building Block", http://www.ietf.org/rfc/rfc3452.txt

[18] IETF Internet Draft: “FLUTE - File Delivery over Unidirectional Transport”,

http://www.ietf.org/internet-drafts/draft-ietf-rmt-flute-07.txt

[19] Tampere University of Technology, Finland (open source code),

http://atm.tut.fi/mad/

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE (INRIA), France (open source code),

http://www.inrialpes.fr/planete/people/roca/mcl/mcl_download.html

NOKIA, Finland (source code not known).

[20] DVB TM-UMTS 1035: Service Purchase Specification, v2.0.

[21] OMA DRM 1.0.

[22] OMA DRM 2.0.

[23] ETSI TR 102 005: “Digital Video Broadcasting (DVB); Implementation guidelines for the use of audio-visual content in DVB services delivered over IP”.

[24] 3GPP SA4: “FLUTE for MBMS downloading”, TD S4-030772 (Nov. 2003).

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

Application: A piece of software that runs making use of Application Platform provided.

Application Platform: Abstracts away the terminal implementation details and provides services for Applications.

Broadcast: Unidirectional point-to-multipoint data transport service in which data is transmitted from a single source entity to all users in a certain geographical area or areas.

Broadcast content provider: Typically produces and aggregates content and sometimes provides broadcast services to broadcast multiplexes.

Broadcast network operator: Operates multiplexes, compiles DVB-SI, delivers transport streams to transmission sites and maintains radio links and network infrastructure. [Note, limited to DVB-T/H network operator in this scope].

Broadcast service provider: Programs (aggregates, schedules and delivers) one-to-many content. This may include compiling service announcements (e.g. DVB-SI) and content adaptation.

Content Protection: The control of access to content and use through usage rights and rules.

Content provider: Provides interesting content for users/consumers. May also aggregate content from other content providers and/or package content as services.

Delivery system: The physical medium by which one or more multiplexes are transmitted. E.g. satellite system, wide-band coaxial cable, fibre optics, or terrestrial radio channel.

DVB network: A collection of MPEG-2 Transport Streams, each carrying a multiplex, and transmitted on a single delivery system.

DVB service: A collection of associated Elementary Streams. DVB service is identified by program_number (in PSI), or service_id (in SI).

DVB signal: Radio signal carrying a Transport Stream of a DVB network.

E-Commerce system: System responsible for enabling purchasing transactions between a service provider and a user.

Electronic Service Guide application (ESG application): Application in the terminal responsible for displaying the ESG, ordering the services from service provider, and initialising viewing of the broadcasts.

Electronic Service Guide metadata (ESG metadata): Descriptions and metadata about the service hierarchies, sessions and related content available. Through that information, using ESG application, the user can select the services and items he/she is interested in and find stored items on the terminal.

Elementary Stream: Stream of Transport Stream packets within a Transport Stream sharing a common packet identified (PID).

I_MT interface: A logical interface representing the collection of interfaces between the mediation platform and the mobile terminal.

Interactivity channel: A bi-directional interaction channel is established between the service provider and the user for interaction purposes. Can also provide connection to mobile network operator providing billing etc.

Internet service provider: A type of service provider which specialises in IP connections and IP-based content (including Internet and world-wide-web).

IP Datacast: Broadcast of various kind of content over IP over DVB transport system.

IP Datacast Baseline: The minimum core protocol profile an IPDC DVB-T/H Receiver may expect to be available on IPDC DVB-T/H Bearer (data transmission baseband) and the IPDC DVB-T/H Network is expected to make available on the IPDC DVB-T/H Bearer.

IP datagram stream: A stream of IPv6 or IPv4 datagrams each sharing the same IP source and destination address. An IP datagram stream is identified within an IP platform by its source and destination addresses. IP datagram stream on different IP platforms may have the same source/destination addresses, but are considered different IP datagram streams. IP datagram stream may be delivered over one or more IP streams.

IPDC DVB-T/H Bearer: The link and physical layers into which IP platform is encapsulated.

IPDC DVB-T/H Network: DVB network that makes the IP Datacast based services and the IP Datacast Baseline available over DVB-T/H for an IPDC Datacast Receiver.

IPDC DVB-T/H Receiver: The equipment or the system that consumes or uses IP Datacast based services provided over the IP Datacast Baseline on DVB-T/H.

IPDC DVB-T/H System: Consists of one or more IPDC DVB-T/H Networks and one or more IPDC DVB-T/H Receivers.

IP platform: A set of IP datagram streams managed by an organisation. The IP platform represents a harmonised IP address space that has no address collisions. An IP platform may span several Transport Streams within one or more DVB networks. Several IP platforms may co-exist in the same Transport Stream.

IP stream: A data stream delivering exactly one MPE encoded IP datagram stream. IP stream is identified by transport_stream_id, original_network_id, service_id, component_tag, and IP source/destination addresses.

Mediation platform: Platform that collects all the functions that enable interworking between legacy domains (broadcast, cellular), or new functions that are not available in any legacy domain.

Mobile network operator: Operates GSM/GPRS and/or UMTS cellular network(s) providing and maintaining various communications links between users (terminals) and external (service provider) networks.

Multicast: Unidirectional point-to-multipoint data transport service in which data is transmitted from a single source entity to all users in a geographical area, but which is consumable (decodable) by only (addressed) sub-group of users in a certain geographical area or areas.

Multiplex: A set of DVB services multiplexed together into a form that can be carried on a DVB Transport Stream.

Network operator: Operates a network including data and control aspects of backbone (core) and radio access infrastructure. Provides data connections between various entities and maintains agreements to the other (commercial) entities, such as users, service provider and other network operators to determine the scope of these connections. [Note, in this document network operators operate digital wireless systems.]

Pull-multicast: Multicast transport service where content is accessed to users after their explicit request.

Push-multicast: Multicast transport service where content is proposed to users without their explicit request.

Service: Combination of application-level objects that are needed for terminal end user.

Service discovery: A communication service by which the terminal discovers the end-user services available through its network interfaces, e.g., list the services available free-to-air or purchasable in an organized way.

Service Discovery Channel: A logical channel used to transport information about the available services. The information includes metadata about the services, how they can be received, and how they can be consumed.

Service protection: The control of access to a service (package of services) and use through encryption and usage rights and rules.

Service provider: Provides users with access to content and related services. May aggregate and transport (e.g. stream) content, and provide service descriptions and discovery.

Service system: System responsible for service management and content distribution.

Session: Combination of transport channels that are used to push the Service content from a source to the terminal.

Terminal: Comprises of all the hardware and software components of end user equipment. Supports Application Platform.

Transport Stream: Stream of transport_packets, as defined in ISO/IEC 13818-1.

Unicast: Unidirectional point-to-multipoint data transport service, in which data is transmitted from a single source entity to all users in a geographical area, but which is consumable (decodable) by only one (addressed) user. In the IP context this corresponds to IP packets being transmitted with a unicast address.

Wireless Service Provider: Provides users multimedia content and related services through bi-directional wireless interaction channel.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

ADSL	Asynchronous Digital Subscriber Line
ALC	Asynchronous Layered Coding
DCO	Datacast Operator
DRM	Digital Rights Management
DVB	Digital Video Broadcasting
DVB-H	Digital Video Broadcasting – Handheld
DVBSTP	DVB SD&S Transport Protocol for Multicast Delivery of SD&S Information.
DVB-T	Digital Video Broadcasting – Terrestrial
DVB TM	DVB Technical Module
DVB TM AVC	DVB TM AudioVideo Content Formats Technologies
DVB TM CBMS	DVB TM Convergence of Broadcast Mobile Services
DVB TM GBS	DVB TM Generic Data Broadcasting and SI Protocols
DVB TM IPI	DVB TM Internet Protocol Infrastructures
DVB TM UMTS	DVB TM Universal Mobile Telecommunications System
ESG	Electronic Service Guide
FEC	Forward Error Correction
FLUTE	File Delivery over Unidirectional Transport
GPRS	General Packet Radio Service
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
IETF	Internet Engineering Task Force
INT	IP/MAC Notification Table
IP	Internet Protocol
IPv4	Internet Protocol, version 4
IPv6	Internet Protocol, version 6
IPDC	IP Datacast
IPE	IP Encapsulator
I_MT	Interface Mobile Terminal
IPSec	IP Security Protocol
ISP	Internet Service Provider
LCT	Layered Coding Transport
MHP	Multimedia Home Platform
MIDP	Mobile Information Device Profile
MIME	Multipurpose Internet Mail Extensions
MPE	Multiprotocol Encapsulation
MPEG	Moving Picture Expert Group
MSISDN	Mobile Station Integrated Services Digital Networks number
NIT	Network Information Table
NTP	Network Time Protocol
OMA	Open Mobile Architecture
POTS	Plain Old Telephone Service
PSI	Program Specific Information
RMT WG	Reliable Multicast Transport Workgroup
RO	Rights Object
SA	Security Association
SAP	Session Announcement Protocol
SDC	Service Discovery Channel
SDP	Session Description Protocol
SI	Service Information

SMS	Short Message Service
UDH	User Data Header
UMTS	Universal Mobile Telecommunications System
WBXML	WAP Binary XML
VFDP	Versatile File Delivery Protocol
WAP	Wireless Application Protocol
WFDP	Wireless File Delivery Protocol
WSP	Wireless Service Provider
WUCP	WAP Unconfirmed Push over SMS
XML	Extensible Markup Language

4 Types of terminals

Depending on the area of application, several types of terminals can coexist. In the sequel, the various terminal types are described.

4.1 Terminal Type I

Terminal Type I is a full DVB implementation for fixed, nomadic or automotive reception. Type I terminal uses interactivity channel as a GPRS/UMTS modem or as a fixed IP connection for interactivity channel functions of MHP.

4.1.1 Application Platform services for Applications

Applications of Type I terminal shall expect full MHP implementation [3]. The applications expect access to a full set of DVB MPEG2 A/V services [4] as well as MHP services available on broadcast channel.

4.1.2 Terminal services for Application Platform

In order to provide the required services for Application, the Application Platform needs to be supported by the Terminal. The bearers are divided into two groups: broadcast channel and interactivity channel.

4.1.2.1 Broadcast channel

The Type I terminal shall implement the broadcast channel reception as specified by the following specifications: [3], [4], [5] and [6].

4.1.2.2 Interactivity channel

The terminal/receiver platform of Type I terminal shall implement the interactivity channel as specified in [7].

4.2 Terminal Type II

Terminal Type II shall implement both type I and type III terminals. Terminal Type II shall be able to run applications of Type I and Type III terminal.

4.3 Terminal Type III

Terminal type III is a mobile handset for mobile, indoor and handheld reception of IP-based services. Type III terminal uses interactivity channel to access broadcast and mobile interactive services.

4.3.1 Application Platform services for Applications

Applications of Type III terminal either expect MIDP environment or run natively on the terminal. The applications expect IP network services on network level. The IP network service is receive-only for the broadcast channel. The IP network service is bi-directional for the interactivity channel.

4.3.2 Terminal services for Application Platform

In order to provide the required services for Application, the Application Platform needs to be supported by the Terminal. The bearers are divided into two groups: broadcast channel and interactivity channel.

4.3.2.1 Broadcast channel

In order to support the delivery of IP, the Type III terminal shall implement the broadcast channel reception as specified in the following:

- DVB-T/H [8]
- PSI/SI guidelines [9]
- IP over MPE as defined in [10]. The Type III terminal shall support reception of elementary streams where Time Slicing and / or MPE/FEC are used.

IPv6 is highly recommended for DVB-T/H broadcast channel in the air interface.

4.3.2.2 Interactivity channel

In order to support bi-directional IP communications, the terminal/receiver platform of type III terminal shall implement the interactivity channel as specified in [7].

5 Services available to the terminal applications

The I_MT protocol stack is a means to provide communication services to a terminal connected to one or more network interfaces.

In the sequel, we detail the services that must be available in the terminal for applications to access communication services.

5.1 Service discovery

This is the service by which the terminal discover the services available through its network interfaces, e.g., list the services available free-to-air or purchasable in an organized way. Sufficient data model and protocol definitions are needed to enable pilots and technical trials. The typical application that will exploit this service is the Electronic Service Guide (ESG).

5.1.1 Scope

Electronic Service Guide (ESG) contains information about the services available. Through the information in the ESG, the user can select the services and items he/she is interested in and find stored items on the terminal. The ESG must be regularly updated via the broadcast channel. The mechanism for that is called the service discovery.

Service Discovery Channel (SDC) is the channel used to transport information about the available services. The information includes metadata about the services, how they can be received, and how they can be consumed. The information is sent as a set of announcement files by using IP based protocols.

To be able to show the service information to the user, the terminal must perform service discovery and receive the information via Service Discovery Channel (SDC). The transmission of the announcement files is done using carousels, meaning that the files of a carousel are transmitted in a loop. The terminal can start listening to the carousel when it needs the information update.

The service carried over the IP streams are described with descriptions carried over IP as well. Multicast / broadcast addressing of IP datagrams are used.

The IP streams carried over IP are organized in form of *services*, composed of *service components*, carried within *sessions*. Services are grouped in *service sets*, presented under *categories*. Services can be purchased *bundled*. Definitions are given for these terms.

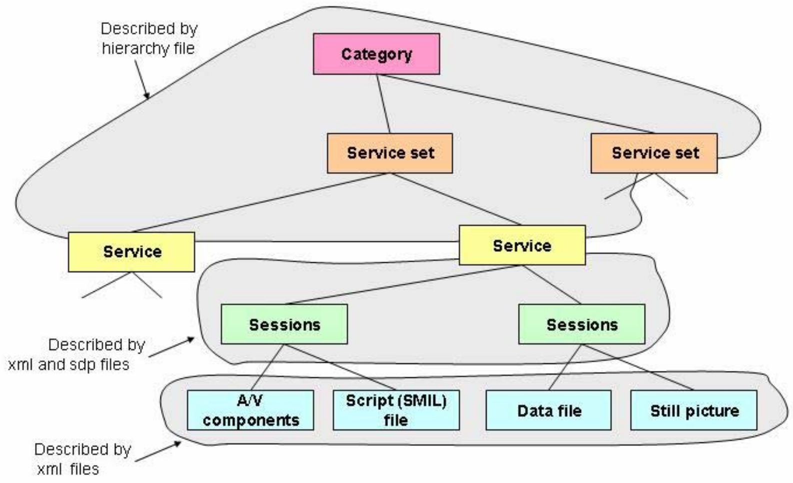

Figure 2 Service concept (service components are examples).

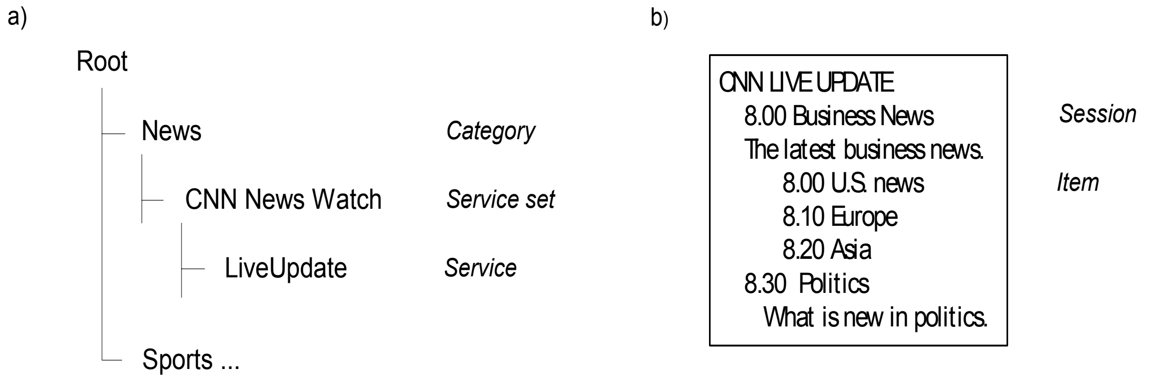

Figure 3 Concrete example of a) Service hierarchy, b) Session hierarchy.

5.1.1.1 Categories

The services are classified with categories. Below a category in the hierarchy there can be subcategories or service sets.

5.1.1.2 Service sets

Service sets combine the same kind of services of one service/content provider together. The same service can belong to many different service sets.

5.1.1.3 Bundles

A bundle is a sellable entity, a group of services that are sold together.

5.1.1.4 Services

Service is the lowest level in the hierarchy of categories, service sets and services. A service always belongs to one Content Provider / Service Provider.

5.1.1.5 Service sessions

Services have service sessions, which mean the scheduled transmission of content related to the service. A service session can contain one or more IP sessions. Each Session should provide enough content to form a Service by its own.

Service sessions comprise of service items, which are a piece of content that can be individually used.

(Note: In the ESG schema the type “session” corresponds also to the one-off purchases vs “bundle” corresponding to subscription type of sellable items.)

5.1.1.6 Service components

The *Service* is the combination of objects with different purposes:

Resource components:

“User oriented” content such as video, sound, text, images, 3D objects, downloaded applications, etc.

Packaging components:

These objects describe the other service components (they represent the *Service signalling)*. For that purpose, several XML based frameworks exists.

The package component indicates how the resource components (A/V, pictures, text, etc) are bound together; it should reflect how the application provider has structured the resource components all together to build the end-user application. This view should be terminal independent.

Presentation components:

These objects exploit the package component and some of the resource components to present the *Service* to the user.

For instance, the presentation component associated with the hierarchy file and the service descriptions are html files.

Interaction components:

These objects enable the user to interact with the applications (resident or downloaded) or with the terminal outside.

Terminals receive dedicated presentation component, specific to terminal capabilities, indicating which and how resource components should be presented to the end user on the terminal display. The resource components are consumed by the resident applications in the terminal. For implementation purpose, servers on the network side may generate from the terminal dependent resource components (and package) in order to simplify processing in the terminal.

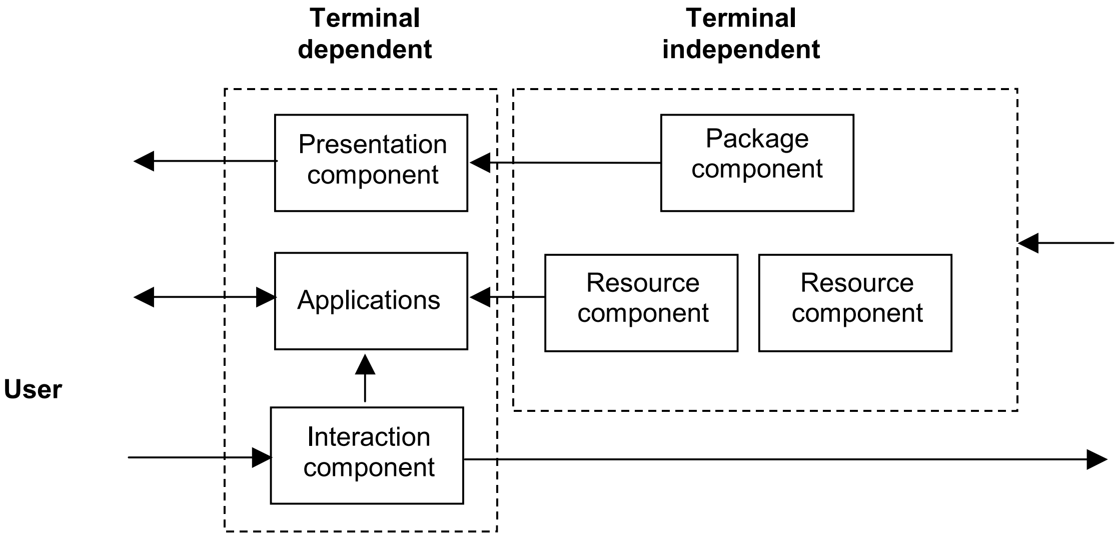

Figure 4 Service components overview.

5.1.2 Logical structure of the service discovery channels

When the terminal is turned on for the first time, the terminal does not have any service information but the information must be received from the network. The information reception can be divided into five phases: get the IP address for the hierarchy file, receive hierarchy file, main pages and service session descriptions, and retrieve detailed information.

There are two kinds of announcements related to service discovery information that is transmitted for the user: service hierarchy announcements and service session announcements. There is one carousel for each announcement type. In addition to these carousels, there is a carousel for main pages, for Security Association (SA) files and for alerts. Thus, there are five carousels: hierarchy, session, main page, SA files and alert carousels.

Figure 5 depicts the protocol stack architecture of the SDC.

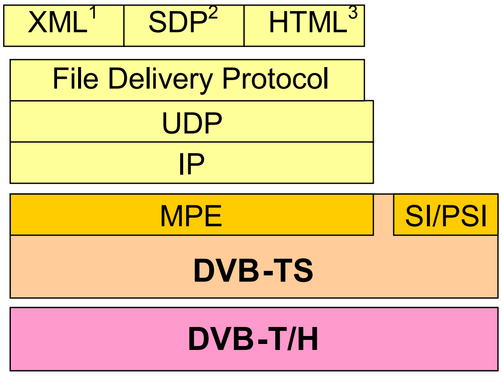

Figure 5 Protocol stack architecture of the Service Discovery Channel [1,2,3].

5.1.2.1 Hierarchy channel

The hierarchy contains categories, service sets and services. The hierarchy is described in one XML file. Thus, if only one language is supported, the carousel transmits the same XML file over and over again. If multiple languages are supported, one file for every language is transmitted. The XML file might be compressed using WAP binary compression, or similar.

5.1.2.2 Session channel

Session announcements include information about the transmitted service sessions of the services and items inside those service sessions. Session announcements are described using SDP [11] and XML.

SDP files contain information that the terminal needs in order to be able to receive and consume the content of the service session. Every SDP file relates to one service session. The SDP file is the same for all the languages, because the file does not contain information that is shown to the user. The XML file includes the information that helps the user to choose what service sessions he/she is interested in and how the sessions can be bought. There is one XML file for every supported language.

5.1.2.3 Main page channel

The main page carousel contains the main pages of the service sets as well as the operator main page. The main page is one file or a set of files. The files can be for example HTML files.

5.1.2.4 Security association channel

The content is encrypted using IPSec. Collections of IPSec keys called Security Association (SA) files are protected using Digital Rights Management (DRM) and the SA files are then transmitted to the user via air interface as a carousel. When the user subscribes to a service bundle, the user buys a Rights Object (RO), delivered via the interactivity channel. With RO the user is able to open the protected SA and receive the content. One SA file is associated to one bundle and it contains the information that is needed for opening the protection of all the service sessions of the bundle . If service sessions can also be purchased separately, a SA file can also refer to one service session.

The security association carousel can also carry the IPSec policy file.

[1] Section 5.1.4.1 Hierarchy information, 5.1.4.2 Session information to users, 5.1.4.4 Security Association information.

[2] Section 5.1.4.2 Session information to terminals.

[3] Section 5.1.4.3 Main page files

5.1.2.5 Notification channel

The alert carousel carries alert notification. The notifications can be part of a service and shown to the end-user or they can be information that is meant just for the terminal.

5.1.2.6 Dynamics

The service discovery process based on the channels described previously is described in the sequel.

- Reception of the correct hierarchy IP address in case of DVB-T/H network.

  The DVB-T/H client consists of a terminal and a receiver. The receiver has an air interface with the DVB-T/H network. The terminal has an interface with the receiver and gets the information from the network through the receiver. When the client is turned on and the ESG information should be received for the first time, the information between the network, the receiver and the terminal flows as follows:

  1. The receiver locates the desired network. The frequencies are scanned through until the desired DVB-T/H network is found. The network can be the first network found, a user defined network, a network configured to the terminal etc.
  2. The receiver receives the Network Information Table (NIT). In this context, the receiver also locates the IP/MAC Notification Table (INT).
  3. One DVB-T/H link may carry several IP platforms. The terminal asks the DVB receiver, which IP platforms are available and the receiver sends the information parsed from the NIT table to the terminal
  4. Terminal selects an IP platform and sends the information to the receiver.
  5. The receiver receives the INT table and parses it.
  6. The terminal wants to get the IP address for the hierarchy file and asks the receiver if it has a mapping for an address belonging to the SDC IP address range.
  7. The receiver returns the address belonging to the address range.
  8. Terminal joins the address and starts to receive hierarchy information.

- Hierarchy file

  When the terminal has found out the correct IP address, it opens the address and starts to listen. The terminal receives an XML file containing the hierarchy, parses it and saves the information to the terminal database.

- Main pages and service session descriptions

  From the hierarchy file the terminal among other things gets the IP addresses for the other carousels. Thus, the terminal can now move to listen to the IP address of the session announcements or the address of the main pages. The main pages are saved to the terminal such that they can be showed to the user when needed. The service session information is described using SDP and XML. SDP files contain the information that the terminal needs to be able to receive and consume the sessions. The XML is used for describing the content of sessions for the user. When the terminal receives the session SDP files and the XML file, it parses them and saves the information to the database.

- Detailed information

  If after receiving the hierarchy and session announcements, the end-user still wants to get more information about the services, service sessions or items, more detailed information can be fetched via web portal. The URLs for the more detailed information is given in the announcements.

- Updates

  In order to make sure that the user does not have outdated information, the information must be updated every now and then. The announcements are sent as carousels, thus the terminal could be listening to the latest announcements all the time. However, because of power saving constraints, the announcements contain expiry time, which is the absolute time when the next version is due. The new version is not necessarily different from the old one, but it is guaranteed that there will not be any changes during the validity of one version. The update interval is different for the different parts of the ESG because the probability of a change is different. Typically, the update interval of the hierarchy is much longer than the update interval for the sessions.

Therefore, after receiving service discovery information for the first time, if the terminal does not move from a network area to another, the terminal only listens to the announcements when it is time for the update. In case of hierarchy file update, the terminal receives the hierarchy file, checks if the modification time of the hierarchy is later than the modification time of the hierarchy in its database. If the received hierarchy is newer, it is updated to the database. Otherwise the terminal just marks down the next expiry time. When checking the hierarchy file, the terminal also reads the expiry time for the session and main page carousels. However, because the update interval for them can be shorter than the update interval for the hierarchy, also the update interval is given for sessions and main pages such that the following expiry times can be calculated without checking the hierarchy file.

If unexpected updates occur during the update interval, the terminal can be informed through alert service that the ESG information should be updated. Alert service is a carousel with notifications to the terminal or to the end-user. If the terminal moves to another network area, the terminal must start to update the information according to the announcements in that area.

5.1.3 Transport of announcement files in Service Discovery Channel

This section addresses transport protocols for announcement files in hierarchy, session, main page, SA files and alert carousels.

5.1.3.1 Option 1: SAP

The proposal is to use Session Announcement Protocol (SAP) [12] as a multicast session description protocol:

1. UDP-based
2. No interactivity channel
3. No synchronization

SDP descriptions would link towards the various channels composing the Service Discovery Channel (SDC). Actual service descriptions, which are in principle heavier than session descriptions, could use better-adapted mechanisms (like DVBSTP) and/or more reliable ones such as FLUTE.

5.1.3.2 Option 2: DVBSTP

DVB SD&S Transport Protocol (DVBSTP) for Multicast Delivery of SD&S Information [13]:

1. UDP-based
2. No interactivity channel
3. No synchronization

A "lightweight file transfer protocol similar to SAP but more efficient" to multicast SD&S (service discovery and selection) information

1. Data table mechanism to enable the receiver to easily identify the payload type and the fragment.
   a. A table is for instance an XML file:
      i. Service discovery information file for example
      ii. TV-Anytime metadata file
2. A version number allows the end device to easily identify if the information has changed

3. Fragmentation: section_number and last_section_number fields allow to fragment a table into variable length sections, each section being carried in a single UDP packet.

4. Very low overhead (16 mandatory bytes including 4 byte CRC + optional Private_Header variable length extensions for private optional data)

5. Efficient processing at the receiver side:

    a. The receiver can receive data immediately after power-on (no need to wait until the beginning of a new cycle)

    b. The receiver can reconstruct the tables progressively (over several cycles)

5.1.3.3 Option 3: WFDP

WFDP: Wireless File Delivery Protocol

To have optimised transmission in a cellular radio environments, France Telecom R&D developed a new protocol for broadcasting files towards mobile devices.

This protocol handles coverage discontinuity (a mobile can leave and enter non overlapping coverage areas and still recover the original file) and allows simultaneous transmission of information over several distinct channels (as long as mobile receivers are able to listen to these channels they will be able to assemble the original message from packets received from distinct channels and not necessarily in the right order).

VFDP: Versatile File Delivery Protocol

This session/transmission layer protocol is an IETF RFC [14] and is engineered by France Telecom R&D to handle transmission of bulky files via satellite network (ensures file integrity and handles high rate and highly asymmetrical transmission, packet loss due to weather, long transmission delays, interactivity channel congestion)

It allows one-to-many transmission, handling group of receivers, and is adapted to always-on-packet-based interactivity channel (e.g. GPRS/ADSL, ISDN) or switched interactivity channel (GSM, POTS). It can ensure 100% reliable transmission using combined acknowledgement + retransmission.

5.1.3.4 Option 4: FLUTE (Recommended transport protocol as a working assumption)

The proposal is to use the results of IETF Reliable Multicast Transport WG (RMT WG). Thus, the following suite of technologies should be applied in file delivery.

- RFC 3450, "Asynchronous Layered Coding (ALC) Protocol Instantiation", [15]

- ALC is based on two RMT "building blocks"
    - RFC 3451, "Layered Coding Transport (LCT) Building Block", [16]
    - RFC 3452, "Forward Error Correction (FEC) Building Block", [17]

- FLUTE - File Delivery over Unidirectional Transport, [18]
    - Currently passed 2nd round of WG Last Calls. As soon as FLUTE version07 is out, it will be submitted to IESG.
    - Three genetically independent reference implementations exist [19].

5.1.4 Descriptions and formats of announcement files in Service Discovery Channel

This section addresses service and session descriptions and formats carried in the hierarchy, session, main page, and security association files.

5.1.4.1 Hierarchy information

IP address range

The IP address range chosen and reserved for the hierarchy information is from FF15::1 to FF15::3FFF.

The port used is 4732.

File naming

The hierarchy XML files are named esg_<esgId>_<modified>_<lang>.xml, where esgId is the unique ID of the ESG, <modified> is the modification time of the file and <lang> is the language used. The values are the same as the corresponding values inside the XML file.

Carried information

PurchaseChannels

Purchase channels describe the channel through which the bundles can be bought. The attributes describing a channel are

- Purchase channel ID: unique ID for the channel
- Request bearer (Enum): the bearer through which the purchase request is sent
- Request address: the purchasing address
- Request: description of the request
- Response bearer (Enum): the bearer through which the purchase response is sent
- Response format (Enum)
- Operator ID: unique ID for the e-commerce operator

Bundles

Services can be bought in bundles. Thus, the bundles are the sellable items. These bundles have

- Bundle ID: unique ID for the bundle
- Bundle name: the name describing the bundle
- Bundle expiry time: The time after which the use of the bundle ends (can be missing when the ending time is not known).
- PurchaseItem for the bundle (e.g. price information for what the user can buy)
    - Whole period price: The price for the whole period
    - Remaining period price: The price for the remaining time of the period (e.g. if the price of the whole January is 5 EUR, on the third of January the price is 5*29/31= 4,68 EUR)
    - Subscription period start time (NTP time, the exact time that the period changes. Just the date is not enough, because the start time of an operational day can be different than the start time of a calendar day)
    - Subscription period end time (NTP time, the exact time when the period ends)

  - Currency
  - Purchase code: Code that is used in e-commerce system to identify the purchased bundle. For every bundle and every period, which can be bought, there is an own purchase code.

    In this example model, there are two possible purchase periods for every bundle: one month and three months. The periods start always from the beginning of the month. The one-month period can be bought as a one-time purchase or a continuous purchase. However, the purchase code is the same in both cases. The purchase codes for these possibilities are:

    b_<dcoId>_<bundleId>, for one month

    b_<dcoId>_<bundleId>_<purchaseMonth>, for three moths

  - PriceLabel: Label that can be shown to the user to explain the use of the purchase item for the user e.g. "3 months purchase", "purchase to the end of the current month" etc.

- Purchase channel ID(s): the ID(s) of the purchase channel(s) through which the bundle can be bought

Categories

- Name: the name of the category
- Category ID: unique numeric ID for the category
- Description: short description for the category
- Display order: in which order the operator wants the alternatives of the category level to be shown
- Parental rating (Enum)
- Parent category ID

Service sets

- Name: the name of the service set
- Service set ID: unique numeric ID for the service set
- Description: short description for the service set
- Display order: in which order the operator wants the service sets of the category to be shown
- Parental rating (Enum)
- Reference to the main pages of the service sets. The main pages are sent in a separate main page carousel
- Information about the services of the service set
  - Service ID(s) of the services that belong to the service set
  - Display order of the service in a service set (in which order the services are shown to the user
  - Default service ID
- Other operator specific attributes (e.g. author, creation date)
  - Value: The value of the attribute, the type of the value can be string, integer, date or URL or the combination of these types (e.g. both an integer and a string value). There are different value fields for all types.

  - Name: name that describes the attribute (e.g. author)
  - Description: describes the possible uses of the attribute
  - Attribute ID: unique ID for the attribute
- Bundle ID(s): the ID(s) of the bundle(s) the service belongs to

Services

- Name: the name of the service
- Service ID: unique numeric ID for the service
- The genre of the service (Enum)
- Description: short description for the service
- URL for detailed description
- Bundle ID(s): the ID(s) of the bundle(s) the service belongs to
- Parental rating (Enum)
- ClientContentPath: path to the service's directory in the end-user client
- Rating URL: URL for rating the service
- Service type (Enum)
- Other operator specific attributes (e.g. author, creation date)
  - Value: The value of the attribute, the type of the value can be string, integer, date or URL or the combination of these types (e.g. both an integer and a string value). There are different value fields for all types.
  - Name: name that describes the attribute (e.g. author)
  - Description: describes the possible uses of the attribute
  - Attribute ID: unique ID for the attribute

Other information

- Carousel information:
  - CarouselEnum: The type of the carousel (Enum)
  - The IP addresses (+ports) of the other carousels. IPv6 is in use.
  - The expiry times and update intervals: the time when the information expires (as NTP time). It is guaranteed that the announced information will not change before that. The expiry time is informed for hierarchy, session and main page carousels. For session and main page carousels, also the update interval (in seconds) is told so that the times for the next updates can be calculated without checking the hierarchy file.
- Modification time: the time, when the hierarchy has been last time modified (as NTP time).
- References to the operator main page: This is the page that the user can see first when opening the ESG. It can include top10 - services, newest services, recommendations etc. The main page should be available (for showing to the user) very quickly after opening the ESG. The main page is sent in a main page carousel.
- ESG ID: unique numeric ID for the ESG. There is one ESG for every network area.
- The languages of the ESG: There is one separate file for every supported language.

Schema

The XML file is built according to the schema (Element and attribute names are bold and their type is wrote using italics for making it easier to read the schema):

	<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="esg" type="esgType" />
EsgType	<xs:complexType name="*esgType*"> <xs:sequence> <!-- Elements must be in this order --> <xs:element name="**mainpage**" type="xs:*string*" /> <!-- Reference to the operator mainpage --> <xs:element name="**carousel**" type="*carousel*" minOccurs="3" maxOccurs="4" /> <!-- Carousel information, alert carousel is not mandatory --> <xs:element name="**purchaseChannels**" type="*channelsType*" /> <xs:element name="**bundles**" type="*bundlesType*" /> <xs:element name="**services**" type="*serviceType*" /> <xs:element name="**hierarchy**" type="*hierarchyType*" /> </xs:sequence> <xs:attribute name="**esgId**" type="xs:*integer*" use="required" /> <xs:attribute name="**lang**" type="xs:*language*" use="required" /> <xs:attribute name="**modified**" type="xs:*integer*" use="required" /> <!-- Last modification time as NTP time --> <xs:attribute name="**expiryTime**" type="xs:*integer*" use="required" /> <!-- Expiry time for the hierarchy file as NTP time --> </xs:complexType>
Carousel	<xs:complexType name="*carousel*" > <xs:attribute name="**carouselEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**firstUpdate**" type="xs:*integer*" use="required" /> <!-- Next time for update (as NTP time), --> <xs:attribute name="**updateInterval**" type="xs:*integer*" use="required" /> <!-- In seconds--> <xs:attribute name="**address**" type="xs:*string*" use="required" /> <xs:attribute name="**port**" type="xs:*unsignedShort*" use="required" /> <!-- range of unsignedShort: 0..65535 --> </xs:complexType>
ChannelsType	<xs:complexType name="*channelsType*" > <xs:sequence> <xs:element name="**channel**" minOccurs="0" maxOccurs="unbounded" > <xs:complexType> <xs:attribute name=”**id**” type="xs:*integer*" use="required" /> <xs:attribute name="**requestBearerEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**requestAddress**" type="xs:*string*" use="required" /> <xs:attribute name="**request**" type="xs:*string*" use="required" /> <xs:attribute name="**responseBearerEnum**" type="xs:*integer*" use="optional" /> <xs:attribute name="**responseFormatEnum**" type="xs:*integer*" use="optional" /> <xs:attribute name="**operatorId**" type="xs:*integer*" use="required" /> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>
BundlesType	<xs:complexType name="*bundlesType*" > <xs:sequence> <xs:element name="**bundle**" minOccurs =”0” maxOccurs="unbounded" > <xs:complexType> <xs:sequence> <xs:element name="**purchaseItem**" type="*purchaseItemType*" maxOccurs="unbounded" /> <xs:element name="**purchaseChannelId**" type="xs:*integer*" maxOccurs="unbounded" /> </xs:sequence>

	<xs:attribute name="**name**" type="xs:*string*" use="required" /> <xs:attribute name="**id**" type="xs:*integer*" use="required" /> <xs:attribute name="**expiryTime**" type="xs:*integer*" use="optional" /> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>
HierarchyType	<xs:complexType name="*hierarchyType*" > <xs:sequence> <xs:element name="**category**" type="*categoryType*" minOccurs="0" maxOccurs="unbounded" /> </xs:sequence> </xs:complexType>
categoryType	<xs:complexType name="*categoryType*" > <xs:sequence> <xs:element name="**description**" type="xs:*string*" minOccurs=”0” /> <xs:choice> <xs:element name="**category**" type="*categoryType*" maxOccurs="unbounded" /> <xs:element name="**serviceSet**" type="*serviceSetType*" maxOccurs="unbounded" /> </xs:choice> </xs:sequence> <xs:attribute name="**name**" type="xs:*string*" use="required" /> <xs:attribute name=”**id**” type="xs:*integer*" use="required" /> <xs:attribute name="**parentId**" type="xs:*integer*" use="optional" /> <!--The id of the parent category, not given if this is the highest level, otherwise mandatory --> <xs:attribute name="**parentalRatingEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**displayOrder**" type="xs:*integer*" use="required" /> </xs:complexType>
serviceSetType	<xs:complexType name="*serviceSetType*"> <xs:sequence> <xs:element name="**mainpage**" type="xs:*string*" /> <xs:element name="**description**" type="xs:*string*" minOccurs="0" /> <xs:element name="**serviceSetMembers**" type="*setMembers*" /> <xs:element name="**bundleId**" type="xs:*integer*" minOccurs="0" maxOccurs="unbounded" /> <xs:element name="**attributes**" type="*attributes*" minOccurs="0" /> </xs:sequence> <xs:attribute name="**name**" type="xs:*string*" use="required" /> <xs:attribute name="**id**" type="xs:*integer*" use="required" /> <xs:attribute name="**parentalRatingEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**displayOrder**" type="xs:*integer*" use="required" /> </xs:complexType>
serviceType	<xs:complexType name="*serviceType*"> <xs:sequence> <xs:element name="**service**" minOccurs="0" maxOccurs="unbounded" > <xs:complexType> <xs:sequence> <!-- Elements must be in this order --> <xs:element name="**description**" type="xs:*string*" minOccurs="0" /> <xs:element name="**detailedInfoURL**" type="xs:*anyURI*" minOccurs="0" /> <xs:element name="**bundleId**" type="xs:*integer*" minOccurs="0" maxOccurs="unbounded" /> <xs:element name="**clientContentPath**" type="xs:*string*" minOccurs="0" /> <xs:element name="**ratingURL**" type="xs:*anyURI*" minOccurs="0" /> <xs:element name="**attributes**" type="*attributes*" minOccurs="0" /> </xs:sequence> <xs:attribute name="**name**" type="xs:*string*" use="required" />

	<xs:attribute name="**id**" type="xs:*integer*" use="required" /> <xs:attribute name="**parentalRatingEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**genreEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**serviceTypeEnum**" type="xs:*integer*" use="required" /> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>
setMembers	<xs:complexType name="*setMembers*"> <!-- information about the services that belong to the service set --> <xs:sequence> <xs:element name="**member**" maxOccurs="unbounded" > <xs:complexType> <xs:attribute name="**serviceId**" type="xs:*integer*" use="required" /> <!-- the id of the service, which belongs to the service set --> <xs:attribute name="**displayOrder**" type="xs:*integer*" use="required" /> </xs:complexType> </xs:element> </xs:sequence> <xs:attribute name="**defaultServiceId**" type="xs:*integer*" use="required" /> </xs:complexType>
purchaseItemType	<xs:complexType name="*purchaseItemType*"> <xs:attribute name="**remainingPeriodPrice**" type="xs:*decimal*" use="required" /> <xs:attribute name="**wholePeriodPrice**" type="xs:*decimal*" use="optional" /> <xs:attribute name="**startTime**" type="xs:*integer*" use="required" /> <xs:attribute name="**endTime**" type="xs:*integer*" use="required" /> <xs:attribute name="**currency**" type="xs:*string*" use="required" /> <xs:attribute name="**purchaseCode**" type="xs:*string*" use="required" /> <xs:attribute name="**priceLabel**" type="xs:*string*" use="required" /> </xs:complexType>
attributes	<xs:complexType name="*attributes*" > <xs:sequence> <xs:element name="**attribute**" maxOccurs="unbounded" > <xs:complexType> <xs:sequence> <xs:element name="**description**" type="xs:*string*" minOccurs="0" /> <!-- description how the attribute can be used--> <xs:element name="**stringValue**" type="xs:*string*" minOccurs="0" /> <!-- The value of a string attribute --> <xs:element name="**integerValue**" type="xs:*integer*" minOccurs="0" /> <!-- The value of a integer attribute --> <xs:element name="**dateValue**" type="xs:*date*" minOccurs="0" /> <!-- The value of a date attribute--> <xs:element name="**URLValue**" type="xs:*anyURI*" minOccurs="0" /> <!-- The value of a URL attribute --> </xs:sequence> <xs:attribute name="**name**" type="xs:*string*" use="required" /><!-- The name of the attribute --> <xs:attribute name="**id**" type="xs:*integer*" /> <!-- Unique id for the attribute --> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>
	</xs:schema>

5.1.4.2 Session information

File naming

The SDP files are named <sessionId>.sdp, where <sessionId> is the unique ID of the service session.

The name of the XML file is <esgId>_<lang>.xml, where esgId is the unique ID of the ESG and <lang> is the language used.

Information for the users

Carried information

There is session related information and program related information in the XML file. Session related information is common to the whole service session. A session has one program or it can be divided into many programs. Descriptions, schedules etc. are given for the programs thus programs are the ones that the user is interested in. Programs can map to for example items but the whole session can also be a program.

Session information

- Session ID: a numeric ID that uniquely identifies the service session (unique in the scope of the session IDs of the operator)
- PurchaseChannel ID(s): the ID(s) of the purchase channel(s) through which the session can be bought (The ids refer to the same channels that are listed in the hierarchy XML file)
- Price for the session (value, currency)
- Purchase code: Code that is used in e-commerce system to identify the purchased session. The form of the session purchase code is s_<dcoId_<sessionId>.
- The SDP file of the session
- Service ID: the unique ID of the service, which the session belongs to. To map the session to the service unambiguously.
- Genre classification information (Enum)
- Encryption information: if the session is unencrypted, thus it is available for everybody; the flag is set to 0 or “false”. For the encrypted sessions, the value of the flag is 1 or “true”. A session is usually encrypted when the session can be bought separately or the service belongs to a bundle. However, a service, which belongs to a bundle, can contain also free, unencrypted sessions. This flag is meant especially for distinguishing those free sessions from the encrypted ones.

Program information

- Program ID: unique ID for the program
- Name of the program
- Scheduling (start time, stop time)
- Description
- URL for more detailed description
- Parental rating (Enum)
- Genre classification information (Enum)
- URL for voting

- URL for rating
- Maximum size of the program
- Program type (Enum)
- Other operator specific attributes (e.g. author, creation date)
    - Value: The value of the attribute, the type of the value can be string, integer, date or URL or the combination of these types (e.g. both an integer and a string value). There are different value fields for all types.
    - Name: name that describes the attribute (e.g. author)
    - Description: describes the possible uses of the attribute
    - Attribute ID: unique (alphanumeric) id for the attribute

Other information

- The expiry time: the time when the information expires (as NTP time). It is guaranteed that the information announced will not change before that.
- Modification time: the time, when the file has been last time modified (as NTP time).
- ESG ID: unique numeric ID for the ESG of the network area
- The language of the ESG: There is one separate file for every supported language.

Schema

	<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="**sessions**" type="*sessionsType*" />

sessionsType	<xs:complexType name="*sessionsType*">
	  <xs:sequence>
	    <xs:element name="**session**" type="*sessionType*" minOccurs="0" maxOccurs="unbounded" />
	  </xs:sequence>
	  <xs:attribute name="**esgId**" type="xs:*integer*" use="required" />
	  <xs:attribute name="**lang**" type="xs:*language*" use="required" />
	  <xs:attribute name="**modified**" type="xs:*integer*" use="required" /> <!-- Last modification time (as NTP time) -->
	  <xs:attribute name="**expiryTime**" type="xs:*integer*" use="required" /> <!-- Expiry time for the session file (as NTP time) -->
	</xs:complexType>

sessionType	<xs:complexType name="*sessionType*">
	  <xs:sequence>
	    <xs:element name="**purchaseCode**" type="xs:*string*" minOccurs=”0” /> <!— Missing if sessions cannot be separately bought -->
	    <xs:element name="**purchaseChannelId**" type="xs:*integer*" minOccurs=”0” maxOccurs=”unbounded” />
	    <xs:element name="**price**" type="*priceType*" minOccurs="0" />
	    <xs:element name="**SDPFile**" type="xs:*string*" />
	    <xs:element name="**program**" type="*programType*" maxOccurs="unbounded" />
	  </xs:sequence>
	  <xs:attribute name="**id**" type="xs:*integer*" use="required" />
	  <xs:attribute name="**serviceId**" type="xs:*integer*" use="required" />
	  <xs:attribute name="**genreEnum**" type="xs:*integer*" use="required" />
	  <xs:attribute name="**encrypted**" type="xs:*boolean*" use="required" />
	</xs:complexType>

programType	<xs:complexType name="*programType*">

	<xs:sequence> <xs:element name="**name**" type="xs:*string*" /> <xs:element name="**description**" type="xs:*string*" minOccurs="0" /> <xs:element name="**time**" type="*timeType*" /> <xs:element name="**detailedInfoURL**" type="xs:*anyURI*" minOccurs="0" /> <xs:element name="**votingURL**" type="xs:*anyURI*" minOccurs="0" /> <xs:element name="**ratingURL**" type="xs:*anyURI*" minOccurs="0" /> <xs:element name="**maxSize**" type="xs:*integer*" minOccurs="0" /> <xs:element name="**attributes**" type="*attributes*" minOccurs="0" /> </xs:sequence> <xs:attribute name="**id**" type="xs:*integer*" use="required" /> <xs:attribute name="**parentalRatingEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**genreEnum**" type="xs:*integer*" use="required" /> <xs:attribute name="**programTypeEnum**" type="xs:*integer*" use="required" /> </xs:complexType>
timeType	<xs:complexType name="timeType" > <xs:attribute name="**start**" type="xs:*integer*" use="required" /> <!— start time as NTP time --> <xs:attribute name="**stop**" type="xs:*integer*" use="required" /><!—stop time as NTP time --> </xs:complexType>
priceType	<xs:complexType name="*priceType*"> <xs:attribute name="**value**" type="xs:*decimal*" use="required" /> <xs:attribute name="**currency**" type="xs:*string*" use="required" /> </xs:complexType>
attributes	<xs:complexType name="*attributes*" > <xs:sequence> <xs:element name="**attribute**" maxOccurs="unbounded" > <xs:complexType> <xs:sequence> <xs:element name="**description**" type="xs:*string*" minOccurs="0" /> <!-- description how the attribute can be used--> <xs:element name="**stringValue**" type="xs:*string*" minOccurs="0" /> <!-- The value of a string attribute --> <xs:element name="**integerValue**" type="xs:*integer*" minOccurs="0" /> <!-- The value of a integer attribute --> <xs:element name="**dateValue**" type="xs:*date*" minOccurs="0" /> <!-- The value of a date attribute--> <xs:element name="**URLValue**" type="xs:*anyURI*" minOccurs="0" /> <!-- The value of a URL attribute --> </xs:sequence> <xs:attribute name="**name**" type="xs:*string*" use="required" /><!-- The name of the attribute --> <xs:attribute name="**id**" type="xs:*integer*" /> <!-- Unique ID for the attribute --> </xs:complexType> </xs:element> </xs:sequence> </xs:complexType>
	</xs:schema>

Information for the terminals

The technical information of service sessions, which is needed for being able to consume the content, is given in an SDP file. The information given in the SDP file is strongly dependent on the application used in terminal. Because the fields vary a lot, nothing final can be specified here. This chapter gives only some guidelines what fields could be used.

Mandatory fields per SDP

The basic SDP structure is defined in SDP RFC [11]. There are lots of optional fields but also some mandatory fields have been specified. The mandatory fields are:

- v= (protocol version) (always = 0)
- o= (owner/creator and session identifier)

  <username> <session_id> <version> <network_type> <address_type> <address>
- s= (session name) (Single space used if no name given)
- c=* (connection information – not required if included in all media)

  <network type> <address type> <connection type>
-
- t= (time the session is active)

  <start time> <stop time>
- m= (media name and transport address) (separated medias, may occur many times)

  <media> <port> <transport> <fmt list>
- c=* (connection information - optional if included at session-level)

  <network type> <address type> <connection type>

Extra attributes

One mean for extending the SDP structure is by using attributes. The attributes can be property attributes (a=<flag>) or value attributes (a=<attribute>:<value>). The commonly used attributes can be registered with IANA and the SDP specification recommends that unregistered attributes are defined using the form a=X-<attribute> or a=X-<attribute>:<attribute value>. However, many applications ignore the recommendation and do no differentiate unregistered and registered attributes.

- a= X-ApplicationType:<MIME type enum>
  - Application MIME type: The application used can be concluded from the MIME type.

This attribute can be supplemented with an argument attribute:

- a=arguments:<additional arguments>
  - Additional arguments: Any additional arguments that are needed for the application.

Other attributes related to SDC that can be used:

- a= priority:<priority value>
  - Priority: if the receiver has created too many filters and QoS problems arise, the sessions with the lowest priority is dropped. The range: min 0, max 255
- a= monitoringFlag:<value> <time interval> <address_type> <address>

- If this flag is on, the terminal must inform the network when it starts to listen to the session and it must keep informing periodically that it is still listening. When the terminal stops receiving the session, it is informs that also to the network. This mechanism enables for example getting real time statistics.
- Transmitted information:
  - If the flag is on or not: If monitoring in use, value is 1 and also the other information must be given. Default value 0, monitoring not in use.)
  - The interval for informing that the terminal is listening
  - The IP address for which the terminal takes contact when informing if the terminal is listening

(In the future, these extra attributes could be included into session XML file.)

5.1.4.3 Main page channel

The files are sent in a defined directory structure: /<esgId>/<serviceSetId>/, where <esgId> is the unique ID of the ESG and <serviceSetId> is the unique ID of the service set. Every file that is in the /<esgId> directory belongs to the operator main page. Every file in the /<esgId>/<serviceSetId> directory belongs to the service set main pages. The name of the index file of the main page is given in the XML hierarchy file.

The operator mainpage can be used in two modes, for example: full screen mode and normal screen mode. The size of the mainpage is different in these two modes and because of that there are two operator mainpages transmitted in the mainpage carousel. The files are named:

- Operator small mainpage: <mainpageName>_s.html
- Operator large mainpage: <mainpageName>_l.html

It is recommended that the main pages do not contain links to outside.

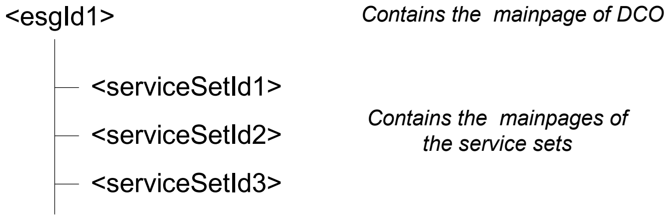
<esgId1>
Contains the mainpage of DCO
<serviceSetId1>
<serviceSetId2>
<serviceSetId3>
Contains the mainpages of the service sets

Figure 6 Mainpage directory structure

5.1.4.4 Security association channel

File naming

The SA files are named in case of bundles b_<expiryTime>_<bundleId>.dcf, where “b” refers to “bundle”, <expiryTime> is the time when the SA expires and <bundleId> is the unique ID of the bundle. If the SA file is for the service session, the name of the file is s_<expiryTime>_<sessionId>.dcf, where “s” refers to “session”, <expiryTime> is the time when the SA expires and <sessionId> is the unique ID of the service session.

SA file structure

Carried information

The values in parenthesis describe the suggested values.

- IPv6 multicast address: (FF38:0:0:0:0:0:XXXX:XX, 3 bytes can change)
- IPSec protocol: (ESP)
- Security Parameter index (SPI): 32 bit value as HEX
- ESP Encryption algorithm: (DES-CBC)
- ESP Encryption keys: 64bit value described as 16HEX chars
- ESP authentication algorithm: (NULL, not used initially)
- ESP authentication keys: (0, not used initially)

Schema

	<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="ServiceProtection" type=" ServiceProtectionType" />
ServiceProtectionType	<xs:complexType name="ServiceProtectionType" > <xs:sequence> <xs:element name=”KeyCrypt” type="keyCryptType" maxOccurs=”unbounded" /> </xs:sequence> </xs:complexType>
keyCryptType	<xs:complexType name="keyCryptType"> <xs:sequence> <xs:element name=”SA” type="SAType" maxOccurs=”unbounded" /> </xs:sequence> <xs:attribute name="IPSecProtocol" type="xs:*string*" use="required" /> <xs:attribute name="ESPEncryptionAlgorithm" type="xs:*string*" use="required" /> <xs:attribute name="ESPAuthenticationAlgorithm" type="xs:*string*" use="required" /> </xs:complexType>
SAType	<xs:complexType name="SAType"> <xs:attribute name="IPv6MulticastStartAddress" type="xs:*string*" use="required" /> <xs:attribute name=”numberOfMulticastAddresses" type="xs:*integer*" use="required" /> <xs:attribute name="SPI" type="xs:*string*" use="required" /> <xs:attribute name="ESPEncryptionKey" type="xs:*string*" use="required" /> <xs:attribute name="ESPAuthenticationKey" type="xs:*string*" use="required" /> </xs:complexType>
	</xs:schema>

IPSec policy file

The SA carousel can also transport the IPSec policy file. The policy file describes IP address ranges and security actions applied to the transmitted data.

In the simplest case the security policy may be defined static. The policy file is transmitted in SA carousel.

5.2 Service access and Service key delivery

This is the service by which the terminal has access to a specific service pointed out in the ESG. Service access relates to services and keys necessary to decipher/descramble the exchanged information. Due to the close relationship between the service access mechanisms and service key delivery they are both treated in this section which is structured as follows:

- Overview of the Service access operation.
- Mapping Service access scenarios, a) service access with payment, b) service access with free registration, c) Free-to-air service access, to the general service access operation.
- Service access protocols in the interaction channel: a) SMS –based, b) HTTP(S) –based.
- Service key delivery: a) First key delivery and b) Re-keying.

5.2.1 Overview of Operation

Figure 7 describes the participants and basic interactions for service access including service key delivery.

Service System (Mediation platform) is responsible for protecting the content, producing Security Association (SA) Files and DRM Rights Objects (RO) associated with them. The sellable item list, the RO files and the mapping of sellable item identities to RO file names are all delivered to the E-commerce system via HTTP interface. Service system delivers SA files through DVB-T/H air interface to a Terminal in SA carousel as described in Section 5.1.

E-commerce system is responsible for distributing the DRM ROs to the Terminal and for making and delivering the bills to the mobile operator's environment. E-Commerce System may sell items from several datacast service providers' lists each of which is uploaded separately from the Service System of each service provider.

Terminal is capable of receiving IP streams through DVB-T/H network and capable of making purchase transactions through cellular network (e.g., GSM). Terminal accessing a service pointed out in the ESG have to purchase the service through an e-commerce system which has part of the ESG information. End-user purchasing a service actually buys RO which are sent as a response to the purchase request. The channel to use for purchasing or registration (incl. purchasing address, bearer for purchase request and response) is indicated in the ESG announcements for each service. SMS –based and HTTP(S) –based service subscription protocols are identified.

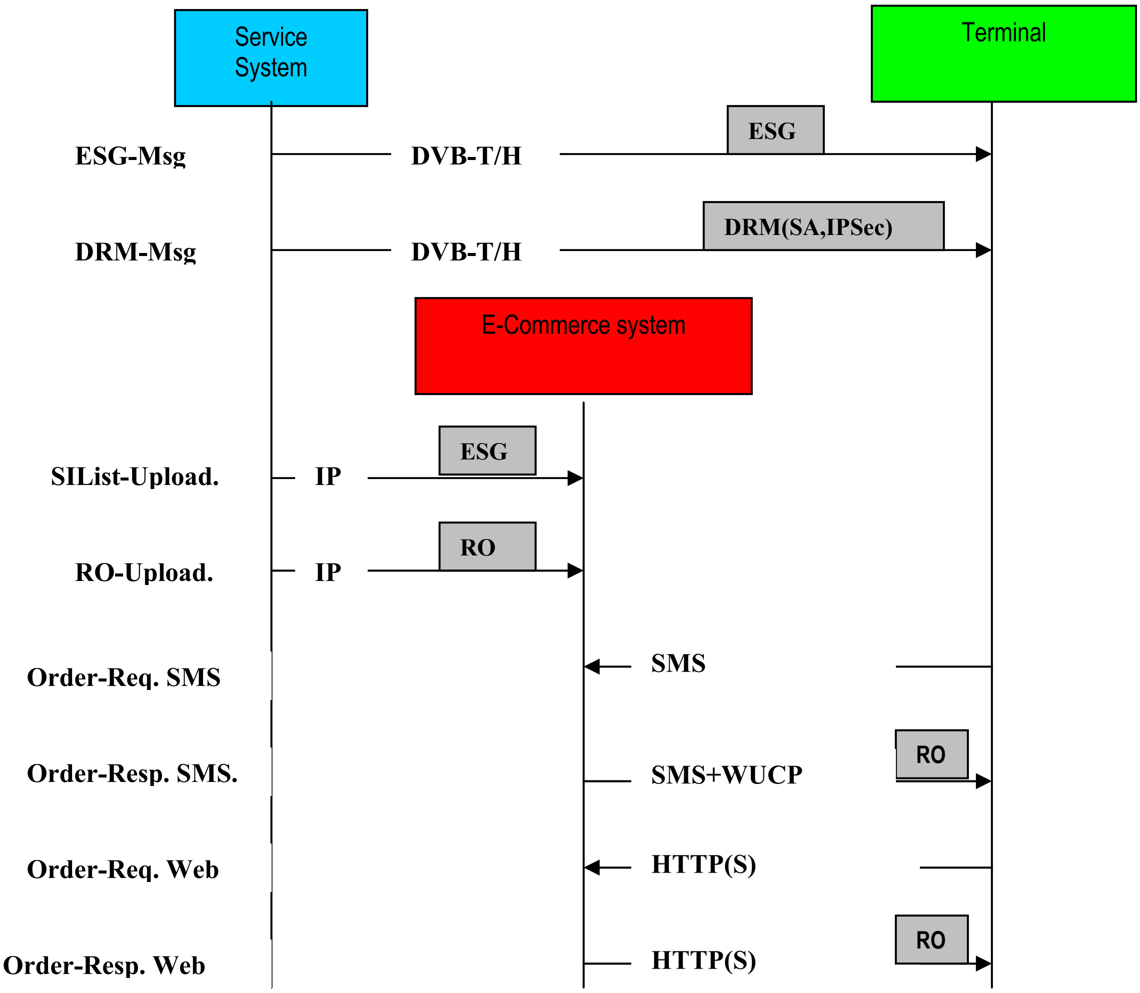

Figure 7: Participants and basic interactions for service access including service key delivery.

5.2.2 Service access scenarios

The following scenarios are recognized for service access:

a) Service access with payment, i.e., purchase a service

b) Service access with free registration

c) Free-to-air service access available to all holders of IPDC terminals.

Free-to-air service access does not require any actions through interaction channel when the service has been discovered from ESG, for example. Terminal capability to receive IP streams through DVB-T/H is needed only.

Service access with free registration follows the same interactions than Service access with payment, in principle. The only difference is that ROs are delivered for free.

5.2.3 Service access protocols in the interaction channel

Purchasing digital rights for a service includes two types of paying models – subscriptions and one-off purchases. In the ESG schema the type “bundle” corresponds to subscription items, and the type “session” to the one-off purchases [20]. Subscriptions are purchased for service packages over a period of time (e.g., continuous subscription, on monthly basis). One-off purchases are for services with instant consumption (e.g., pay per view) or for services lasting with a very limited time (e.g., a day).

Important functions/commands for a system to support are:

- SUBSCRIPTION/PURCHASE: Order to purchase a sellable item with confirmation of subscription and RO delivery as a response.
- CANCEL: Subscription cancellation with confirmation message as a response.
- ACCOUNT: Current subscription status of a user returned as a response.
- RELOAD: Sequence of all verified ROs held by a user are returned in case one or more ROs are damaged within Terminal DRM server module in the Terminal or not received.

Discontinuation of a bundle of services by a Service provider should be announced to all the users of that bundle. (ESG stops announcing the bundle to Terminals and E-Commerce system. The E-Commerce system takes care of sending notices to existing users of the bundle).

Multiple user languages must be supported, in the sense of all confirmations and error messages concerning particular sellable item must always use the name of the sellable item in the language indicated in the purchase order.

The commands supported by the system are specified rigidly for SMS access channel and more flexibly for HTTP(S) access channel.

Two service access protocols for purchasing specified in this specification are a) SMS –based and b) HTTP(S) –based ones.

5.2.3.1 SMS –based Service Access

SMS -based Service access model specified in [20] is recommended to be used. The core elements of the method are described in the following.

Order Request

SMS message have to include the following items:

- Message type ID
- Sellable item unique ID
- Purchase type parameters (e.g., one-off, one month, three months, continuous)
- Price of the sellable item
- Validity period (starting and ending times) of the sellable item
- User’s language
- Response channel identification
- User authentication attribute

SMS request is sent to SMS Center available to the Terminal. SMS Center number is retrieved from the requestAddress attribute of the purchaseChannel in ESG. SMS Center adds user identification (e.g., MSISDN in case of GSM) to the request which serves as an authorisation identity. MSISDN number is inserted into request without user’s intervention (retrieved from network components/SMSC).

Order Response

Confirmation of subscription is sent to the end-user as SMS after user authentication and processing the subscription request.

The delivery of RO shall be done using Wap Unconfirmed Push Over SMS (WUCP) as defined in OMA DRM 1.0 [21]. If WAP Push Proxy is not used, delivery is done utilizing Used Data Header (UDH) in SMS. Content itself is in the form of WBXML.

For other type of messages response channel is SMS.

5.2.3.2 HTTP(S) –based Service Access

Order Request

Web-based access is through an HTTP(S) dialog. The user browses to the URL of a Web shop using 2G/3G access link.

Order request will include similar request attributes (service identifications) as in the SMS –based service access but the dialog can follow different protocol for each Web shop, since it does not depend on ESG application of the terminal. An appropriate target could be that the basic order request functionalities (purchase, cancel, account, reload) defined for SMS could be supported by the HTTP(S) dialog to enable similar kind of purchase experience for an end user.

Order Response

Order response for RO delivery can be either HTTP(S) or SMS+WUCP, depending on the capabilities of the terminal. Other response messages are over HTTP(S).

There is also an alternative that both order request and response are through HTTP but secured with OMA DRM2.0 [22] (user specific service access keys, ROs) as opposed to HTTP(S).

5.2.4 First key delivery

A service is encrypted using IPSec. Digital Rights Management (DRM) is used to protect collections of IPSec keys called SA files. No content protection mechanisms are assumed. One SA file is associated to one service bundle and it contains the information that is needed for opening the protection of all the service sessions of the bundle. If service sessions can also be purchased separately, a SA file can also refer to one service. With a purchased DRM RO associated to a SA file the user is able to open the protected SA and receive the content.

SA files are transmitted to the user via DVB-T/H air interface as a SA carousel. DRM ROs are delivered via cellular network with SMS+WUCP or HTTP(S).

5.2.5 Re-keying

System should be able of self-initiated re-keying, i.e., delivery of new ROs for service subscriptions. Key updates weekly or monthly for service bundle subscriptions could be a reasonable frequency. Service system delivers ROs to e-commerce system and the e-commerce system pushes them to end-users with SMS+WUCP.

SA updates are through SA carousel.

IPSec keys could be different even within every session.

In case of generic ROs used frequency of re-keying should not be a problem for the system (no network load issue). Meanwhile, user-specific DRM ROs allow better control of authorisation of different receivers. However, generation and delivery of keys in this case (one key per user, millions of users in a commercial system) may become problematic. This kind of protection method maybe suitable for trials / pilots only. These are, however, in the scope of this specification. Individual encryption of service access keys can be complementary to generic RO approach.

Mechanisms to prohibit key redistribution to non-authorized parties are out of scope for this specification. There will be a need to look solutions for commercial deployment but pilots are not expected to be affected by lack of such specifications. Moreover, appropriate solution proposals are not available for the time being.

5.3 Service interaction

5.3.1 Scope

In this section service means an end-user service like playing a game rather than a service as seen from the viewpoint of a telecom operator, which might for example be simply connection to a communications network, carriage of IP, or a positioning service. Interaction with a service can take place in several manners. Some might use an existing communications protocol and end-user application. For example some services might be accessed by the user sending and receiving SMS messages on a normal phone with no extra software or standardisation needed. This type of service interaction does not need to be covered here.

This section is concerned with services which are provided by means of a Java application program which is downloaded into the terminal. The program environment would be MIDP. In this case the downloaded program will communicate, where necessary, with an application server to provide the experience or information that the user requires. The interaction will consist of the reception of data in the broadcast stream and two-way communication via a telecom system. What we are describing here is the necessary protocol provision to allow for this communication between terminal and application server.

5.3.2 Protocols required to support interaction with remote application server

Given that we have a downloaded program and a remote service which have been designed to work together, many details of the interaction are service designer issues and do not need standardisation. All that is needed is the means for establishing communication between the downloaded program and the server part of the application. These communication means are expected to be:

- Full IP data transfer in both directions on the telecommunications link (e.g. GPRS or UMTS) and full IP data reception on the broadcast link (DVB-T/H).
- Alternative telecommunication data transport mechanisms such as short message -based communication.
- Alternative broadcast data transport mechanisms such as the object carousel or private sections over the broadcast link.

Short message communication provides a means for sending and receiving packets of data. The interpretation of that data (e.g., SMS, MMS, or WAP-Push messages...) is not covered here; it is an application service design issue. Similarly objects arriving over an object carousel will either be privately designed by the application designer or will have a format defined in other specifications.

This leaves the IP links, both broadcast and interactive which provide the basis for a very general form of interaction. There is no clear need for protocols above the IP layer to be defined here since code for these can be downloaded as part of the application code. The two reasons why these might be standardised would be:

- Because of high processor resource usage of execution of a protocol which demands a built-in native implementation. AV transport and file download are the most likely to incur performance penalties because of the potentially high data rate. Since these are covered elsewhere in this specification, we do not specify any protocols for processing load reasons.
- It may be desirable to specify some protocol handling libraries to be present in the terminal to reduce program size and thus download time and cost. Those protocols such as UDP and TCP, which are anyway needed in the terminal, should be included in this list. The set of protocols to be covered in this way is for definition at a later stage.

5.4 Audio/Video streaming

This is the service by which the terminal access Audio and/or Video content real-time, e.g., watch live TV.

To deliver Audio/Video over IP in a DVB-T/H broadcast channel usage of TR 102 005 is proposed. [23].

5.5 Push file delivery

This is the service by which the terminal receives files. A delivery of a single file or a (recursive) directory of files are examples of this service. Delivered file is to be opened with a separate application matching the type of the file delivered.

Protocol specification is required for simple push delivery as well as for more reliable delivery. The protocol should not restrict the file type.

The preferred proposal is to use FLUTE/ALC suite of technologies (results of IETF Reliable Multicast Transport WG) in file delivery [15], [16], [17], [18]. FLUTE has been accepted as a working assumption for MBMS download in 3GPP SA4 [24].

FLUTE/ALC is also a recommended candidate for unidirectional multicast service announcements in this specification (see Section 5.1.3.4).

The following alternative solution proposals are also recognized for file delivery:

- DVB SD&S Transport Protocol (DVBSTP) for Multicast Delivery of SD&S Information [13]. The protocol is also a candidate for unidirectional multicast service announcements in this specification (see Section 5.1.3.2).
- Wireless File Delivery Protocol. The protocol is also a candidate for unidirectional multicast service announcements in this specification (see Section 5.1.3.3).

History

Document history		
<Version>	<Date>	<Milestone>
V0.0.1	24.09.2003	Put together work of subgroup I_MT.
V0.0.2	01.10.2003	Incorporated review comments from subgroup I_MT.
V0.0.3	25.11.2003	First inputs from Call for Proposals –process.
V0.0.4	10.12.2003	First edited draft specification based on the contributions from CfP –process.
V0.0.42	18.12.2003	Main comments included from I_MT review meeting 11th Dec. 2003.
V0.0.5	12.01.2004	Comments from I_MT review meeting 8th Jan. 2004. Version proceeded to be reviewed by DVB TM UMTS group on the 14th Jan. 2004.
V1.0.0	30.01.2004	DVB TM has approved the document on the 29th Jan. 2004 to be proceeded to a DVB Blue Book. Editorial modifications compared to version 0.0.5.
V1.0.0	01.04.2004	Publication as A080