ulm university universität
uulm

Technical Report VS-R19-2011

Aspectix Research Team | Institute of Distributed Systems | University of Ulm, Germany

A unified API for negotiation in multimedia middleware

Jan-Patrick Elsholz · Alexander Fromm · Sven Schober · Franz J. Hauck

2011

ART
Aspectix Research Team

A unified API for Negotiation in Multimedia-Middleware

Jan-Patrick Elsholz, Alexander Fromm, Sven Schober, Franz J. Hauck
Institute of Distributed Systems
Ulm University, Germany
Email: {jan-patrick.elsholz,alexander.fromm,sven.schober,franz.hauck}@uni-ulm.de

***Abstract*—The Next Generation Network (NGN) supports end-to-end negotiation between NGN entities and such of other networks [16]. However, interoperability between different negotiation protocols depends on gateways and benefits of upcoming protocols cannot be utilised. We proclaim, that a multimedia middleware avoids the need for gateways by enabling the application itself to cope with different protocols. This is realized by exchanging the underlying protocol transparently to the application through a unified API. Therfore, we define a generic programming model and introduce a multimedia ontology abstracting from specific protocol representations.**

I. Introduction

Multimedia communication becomes more and more important nowadays. Consider Voice over IP (VoIP) only connections by telephone providers and video telephony like Skype. Interoprability between all of these is indispensable to support communication between two participants. As providers might use different protocols, the Next Generation Network (NGN) suggests gateways. Inside the NGN a specific protocol for negotiation is used. Thus, applications are written for this protocol exclusivly. Consider upcoming protocols with advantages in bandwith and latency. Even though gateways enable us to talk to clients with such protocols, the NGN itself will still use the legacy protocol. We propse a multimedia middleware enabling the exchangebility and interoperability between different negotiation protocols. Thus, all advantages of upcoming protocols can be used even inside the middleware. Therefore, applications are developed using our unified application programming interface (API). This enables them to cope with different protocols without altering the implementations.

First, background technologies are discussed in Section II. Next, we present our unified API. Section IV shows our multimedia ontology. Finally we conclude in Section V.

II. Background

In this Section we discuss related work and describe basic technologies used in our prototype.

A. Multimedia Middleware

The International Telecommunication Union (ITU) defines a communication architecture called NGN where service-related functions are independent from underlying transport-related technologies with a separation of control functions [14]. The 3rd Generation Partnership Project (3GPP) leverages this approach with the IP Multimedia Subsystem (IMS) [1] to provide standardized access to different network services. Therefore, they specify an interface for accessing IMS services, called the Java Specification Request (JSR) 281 [25]. This interface abstracts from technical details and provides a high-level API for establishing sessions between different participants. However, it focuses on the Session Initiation Protocol (SIP) for session establishment and tightly couples the Session Description Protocol (SDP) for negotiation. Thus, there is no seperated API for negotiation and it is impossible to exchange the negotiation protocol seperately.

SailFin CAFE [22] provides a communication server based on the Java Enterprise Edition (EE) platform GlassFish [24] and the SIP Servlet API JSR 289 [17]. The programming model is a communication bean managed by the Java EE container. Again, it strongly focuses on SIP and SDP with gateways for interoperability.

Instant-X (IX) [3] is a middleware platform which provides multimedia functionality in terms of a Service Oriented Architecture (SOA) [23]. Thus, each task of a multimedia communication like signaling, transfer and negotiation is encapsulated into a seperated service. As IX is based on the component framework OSGi Alliance (OSGi) [19], it allows exchangebility of service components at runtime. Following the SOA approach, IX provides the programming model seen in Figure 1.

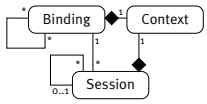

Fig. 1. Programming model of Instant-X

A *binding* represents a local end point, identified by a Uniform Resource Identifier (URI) of the local participant. It is responsible to activate its URI, e. g. keep a registration of a SIP URI alive. To keep track of this registration, an internal state machine similar to NGN is used. A *session* is a Peer-to-Peer (P2P) relationship between two participants. Each session is uniquely identified by the URI of its remote participant. Session and binding make up the tuple of sender and receiver. Again, a state machine reflects the status of this session. In contrast to the NGN, this state machine is decoupled from the negotiation related tasks. We distinguish three different session types: signaling, data and conference. The latter combines

many signaling sessions for multiple participants. The *context* contains configuration parameters for binding and session, e. g. credentials for registering a SIP URI.

B. Negotiation Protocols

The Session Description Protocol (SDP) [8] is a text-based protocol describing multimedia communication sessions. It negotiates the necessary parameters between end points based on the Offer/Answer Model [21]. Figure 2 shows an example containing a single video stream. The stream is sent and received bidirectional and may either be encoded with PCMU or G.723 and is transfered via Realtime-Transport-Protocol (RTP) over port 9948 from 134.60.70.200.

```
1  v=0
2  o=bob 2890844526 2890842807 IN IP4 134.60.70.200
3  s=call
4  e=bob@uni-ulm.de
5  c=IN IP4 134.60.70.200
6  t=2873397496 2873404696
7  m=audio 9948 RTP/AVP 0 4
8  a=rtpmap:0 PCMU/8000
9  a=rtpmap:4 G723/8000
10 a=sendrecv
```

Fig. 2. Example for an SDP session description

The Session Description Protocol next Generation (SDPng) [18] is a successor of SDP based on the Extensible Markup Language (XML). It specifies session and capability description in broadcast scenarios and dynamic negotiation for interactive media [18].

The End-to-End Negotiation Protocol (E2ENP) [6] is a Quality of Service (QoS) enabled negotiation protocol also based on XML. It allows references among capabilities and configuarations. This may reduce bandwith, sending only references instead of complete definitions. In contrast to SDP and SDPng, E2ENP distinguishs three negotiation phases: *prenegotiation*, *negotiation* and *renegotiation*. During *prenegotiation* configuration data valid for several E2ENP-Sessions is exchanged. The idea is to accelerate subsequent phases. *Negotiation* takes place once at each E2ENP-Session establishment. Parameters only valid for a single session are negotiated. They can be modify later several times during *renegotiation*.

H.245 [15] is a control protocol for multimedia communication specified by the ITU. Messages are binary encoded in ASN.1. It provides master[1], slave determination, logical channel signaling and terminal capability exchange. Logical channels are data transmissions like audio streams in a VoIP scenario. They must be explicitly established through a handshake. Therefore, an *OpenLogicalChannel* message is either transmitted seperatly (out-of-band negotiation) or within a signaling protocol like H.225 (in-band negotiation).

Besides the advantages of these negotiation protocols, they are not compatible to each other. So exchanging the negotiation protocol inside an application leads to completely new application implementations. We proclaim, that a unified API will decouple the specific protocol implementiation from the application.

[1]The master has a higher priority resolving conflicts.

C. Multimedia Ontolgies

A unified API enables the exchangebility of specific negotiation protocols. However, the application must specify capabilities and preferences somehow. Therefore, a multimedia ontology can be used to describe them in an abstract way. Ontologies define a set of representational primitives which model a domain of knowledge [5]. These primitives are identified by URIs and form a vocabulary. A so-called statement is a triple of such primites, e. g. subject, predicate and object. These statements add semantic information to the primitives which is machine readable. This enables reasoning about preferences.

Standard no. 7 specified by the Moving Picture Experts Group (MPEG) [12] is an XML based description language for multimedia data. Several approaches have been proposed to transfer MPEG-7 into an ontology [9] [4] as MPEG-7 meta-data terms lack semantic information [2].

MPEG-21 is an open framework for multimedia delivery and consumption [10]. Therefore, Digital Items (DIs) are defined as structured digital objects with a standard representation, identification and metadata [10]. MPEG-21 Digital Item Adaptation (DIA) [11] provides normative description formats supporting the adaptation of DIs, but it does not define interactions with existing transport and control technologies [7].

The Composite Capabilities/Preferences Profile (CC/PP) [26] is specified by the World Wide Web Consortiums (W3C) and describes device capabilities and user preferences. A so-called *profile* contains *components* describing major entities of a machine. They contain *attributes* which describe the particular capability or preference. However, CC/PP only defines the structure and not the vocabulary itself. For this purpose, MPEG-7 can be used.

III. Unified API

The main idea is to abstract from technical details and provide a unified API for negotiation in multimedia communication. Therefore, common phases from related negotiation protocols are extracted and combined under a common interface. These are *negotiation* and *renegotiation*. Both negotiate the necessary parameters for data sessions.

The *negotiation* occurs exactly once during session establishment. Depending on the negotiation protocol a number of negotiation messages are exchanged. Each message contains the desired preferences of a user. The preferences are classified into static and dynamic categories. The former have an informational function and are immediatly valid after specification, e.g. the IP-address. In contrast, dynamic preferences are considered as an offer from a session's participant. This offer becomes valid if the other participant agrees. Thus, the negotiation tries to find a consensus among the different dynamic preferences of two participants. Therefore, the result are session parameters for data transmission(s).

The *renegotiation* performs a negotiation as well. In contrast to it, the renegotiation is based on existing session parameter from a previous negotiation and can be executed more than once. It modifies partially or completly existing session parameters. For example, a video telephony is reduced to VoIP.

To reflect both in our unified API, the negotiation component contains a finite state machine. The generic programming model in Figure 3 is based on IX (see Section II). It is enhanced by a negotiation component.

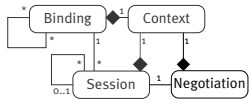

Fig. 3. Generic programming model of Instant-X enhanced by a negotiation component

The negotiation component is closely related to the session, because a negotiation protocol requires a separate transmission protocol for the negotiation messages. Thus, a signaling session must provide a valid transmission session to the negotiation. A `NegotiationHandler` is registered at this session to send and receive negotation messages. For example, SIP transmits SDP messages in-band. Capabilites and preferences for the negotiation are provided by the context in an abstract way by means of our multimedia ontology (see Section IV). Furthermore, negotiation and signaling component are loosely coupled, which allows any combination of negotiation and signaling protocol.

IV. Multimedia Ontology

In this Section we present a multimedia ontology to describe capabilities and preferences in an abstract way. Applications use this so-called Common Language (CL) exclusivly. Any negotiation protocol maps this CL to its specific representation. Thus, integration of upcoming protocols is easily done by a single mapping from CL to the new protocol.

Figure 4 shows the capabilities necessary for negotiation. Graphs are common visual representations of ontologies and in this figure depicted as `[subject]-predicate->[object]`. Prefixes specify the namespaces for the URIs. Literals are rendered in quotes. Installed and supported protocols are listed in line 3. They are categorized by transport, negotiation and signaling. Installed and supported codecs are listed in line 10. Their quality is characterized by an abstract level like the Mean Opinion Score (MOS) factor [13]. This enables the application to find appropriate codecs to the user's need. The user only specifies a very rudimentary desired quality. Through the SPARQL Protocol and RDF Query Language (SPARQL) [20], the multimedia ontology will deliver all available codecs satisfiying this quality metric. Additionally, the quality is personalizable to the user's preferences. The URIs to identify protocols and codec are reused from MPEG-7.

User preferences are described in *UsagePreferences* (see Figure 5). It contains general information like an email address

```
1  [m21-dia:TerminalCapability]
2   +-ix:protocol
3   |  +-> [rtp:avp-0]
4   |        +-rdf:type-------> "transport"
5   |        +-ix:codec-------> [mpeg7:pcmu]
6   |        +-rtp:profile---> "avp"
7   |        +-rtp:payloadtype-> "0"
8   |
9   +-m21-dia:Format
10     +-> [mpeg7:pcmu]
11          +-rdf:type-------> "audio"
12          +-ix:quality-----> "good"
13          +-ix:bandwidth---> "155"
```

Fig. 4. Graph notation of a capabilities example

```
1  [m21-dia:UsagePreferences]
2   +-sdpng:info
3   |  +-> [ix:info-01]
4   |        +-foaf:mbox-> [mailto:bob@uni-ulm.de]
5   |
6   +-sdpng:component
7      +-> [ix:audio-01]
8           +-rdf:type-------> [ix:audio]
9           +-ix:priority----> "2"
10          +-sdpng:alternative-> [<rdf:Bag>]
11                                  |
12   +--------------------------------+
13   |
14   +-rdf:type
15   |  +-> [ix:audio-01:avp-0]
16   |        +-m21-dia:Format-> [mpeg7:pcmu]
17   |        +-ix:transport
18   |           +-> [rtp:avp-0]
19   |                 +-rtp:ip-addr--> "134.60.70.200"
20   |                 +-rtp:rtp-port-> "9948"
21   +-rdf:type
22      +-> [ix:audio-01:avp-4]
23          +-m21-dia:Format-> [mpeg7:g722]
24          +-ix:transport---> [rtp:avp-4]
```

Fig. 5. Graph notation of a user's preferences example

as optional SDP contact information in line 3. Concrete *component* configurations are described in line 7. They define the transferred data type like audio, the transport protocol, a list of optional codecs and a priority. The latter specifies the importance of the particular *component* for the user. If no alternative for such a *component* matches any *TerminalCapabilites*, the negotiation component may find different but still appropriate formats via ontology reasoning.

The example depicted in Figure 5 is mapped to the representation seen in Figure 2 by an SDP negotiation component. Therefore, each component expands the ontology.

V. Conclusion and Future Work

In this work we present a unified API for negotiation in multimedia middleware. Unlike related work, our approach supports the exchangebility of different negotiation protocols. This avoids the necessity for gateways for interoperability. Even more, benefits of upcoming protocols are automatically integrated into the multimedia middleware as well. To enable applications to cope with these different protocols transparently, we introduce a multimedia ontology as a common negotiation language for any specific protocol. This decouples capabilities and preferences from protocol specific representations. Furthermore, we introduce a generic programming model for negotiation protocol implementations. Thus, applications

use our unified API only. In future work we will integrate additional negotation protocols to reveal conceptional problems and missing features.

REFERENCES

[1] 3GPP. IP Multimedia Subsystem (IMS); Stage 2 (Release 9). TS 23.228, 2009.

[2] S. Dasiopoulou, V. Tzouvaras, I. Kompatsiaris, and M. G. Strintzis. Enquiring mpeg-7 based multimedia ontologies. *Multimedia Tools Appl.*, 46(2-3):331–370, 2010.

[3] J.-P. Elsholz, H. Schmidt, S. Schober, F. J. Hauck, and A. J. Kassler. Instant-X: Towards a Generic API for Multimedia Middleware. In *Proceedings of IMSAA 2009*, Bangalore, India, December 2009. IEEE.

[4] García, R. Ontologies for MPEG-7 and MPEG-7 Classification Schemes, 2005.

[5] Gruber, T. Ontology. In *Encyclopedia of Database Systems*, pages 1963–1965. 2009.

[6] Teodora Guenkova-Luy. *Coordination of multimedia services and applications in mobile, heterogeneous network environment*. PhD thesis, Ulm University, 2007.

[7] Guenkova-Luy, T. and Schorr, A. and Hauck, F. and Kassler, A. and Wolf, I. and Feiten, B. and Timmerer, C. and Romijn, W. Harmonization of Session and Capability Descriptions between SDPng and MPEG-21 Digital Item Adaptation, 2005.

[8] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. RFC 4566 (Proposed Standard), July 2006.

[9] Hunter, J. MPEG-7 Ontology. http://metadata.net/mpeg7/, 2006.

[10] ISO/IEC 21000-2:2005. MPEG-21 – Part 2: Digital Item Declaration, 2005.

[11] ISO/IEC 21000-7:2003. MPEG-21 – Part 7: Digital Item Adaptation, 2003.

[12] ISO/IEC TR 15938-11:2005. Information technology – Multimedia content description Interface – Part 11: MPEG-7 profile schemas, 2005.

[13] ITU. Methods for Subjective Determination of Transmission Quality (ITU-T Recommendation P.800), 1996.

[14] ITU. General overview of NGN (ITU-T Recommendation Y.2001), 2004.

[15] ITU. Control protocol for multimedia communication (ITU-T Rec. H.245), 2009.

[16] ITU. NGN capability set 2 (ITU-T Recommendation Y.2007), 2010.

[17] JSR 289 Expert Group. JSR 281 SIP Servlet Specification, version 1.1, 2008.

[18] Bormann Kutscher, Ott. Session description and capability negotiation. Internet-Draft draft-ietf-mmusic-sdpng-08, IETF, August 2005. Work in progress.

[19] OSGi Alliance. OSGi Service Platform: Core Specification, Release 4, 2007.

[20] Prud'hommeaux, E and Seaborne, A. SPARQL Query Language for RDF, January 2008.

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

[22] R. B. Sankara and Pg Binod. Powering next generation media services and convergence with java ee. JavaOne 2009 Vortrag, 2009.

[23] W. Schulte and Y. V. Natis. 'Service oriented' architectures, part 1. SSA research note SPA-401-068, Gartner, April 1996.

[24] Sun. The GlassFish Community Delivering a Java EE Application Server, 2007.

[25] Sun. JSR 281 IMS Services API. http://jcp.org/en/jsr/detail?id=281, 2009.

[26] W3C. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0, 2004.