Towards an MPEG-7 Query Language

Mario Döller[1], Harald Kosch[1], Ingo Wolf[2], and Matthias Gruhne[3]

[1] Department of Distributed Information Technology, University Passau
{Mario.Doeller,Harald.Kosch}@uni-passau.de
[2] Department Platforms for Media Broadband, T-Systems Enterprise Services GmbH
WolfI@t-systems.com
[3] Institut Digitale Medientechnologie (IDMT), Fraunhofer Illmenau
ghe@idmt.fraunhofer.de

Abstract. Due to the growing amount of digital media an increasing need to automatically categorize media such as music or pictures has been emerged. One of the metadata standards that has been established to search and retrieve media is MPEG-7. But it does not yet exist a query format that enables the user to query multimedia metadata databases. Therefore the MPEG committee decided to instantiate a call for proposal (N8220) for an MPEG-7 query format (MP7QF) framework and specified a set of requirements (N8219). This paper introduces a MP7QF framework and describes its main components and associated MP7QF XML schema types. The framework makes use of the MPEG-21 digital item declaration language (DIDL) for exchanging MP7QF Items along various MP7QF frameworks and client applications. An MP7QF Item acts as container for the input query format and output query format of a user query request.

This paper concentrates on components of the framework such as session management, service retrieval and its usability and excludes consciously definitions and explanations of the input and output query format.

1 Introduction

Stimulated by the ever-growing availability of digital audiovisual material to the user via new media and content distribution methods an increasing need to automatically categorize digital media has emerged. Descriptive information about digital media which is delivered together with the actual content represents one way to facilitate this search immensely. The aims of so-called metadata ("data about data") are to e.g. detect the genre of a video, specify photo similarity or perform a segmentation on a song, or simply recognize a song by scanning a database for similar metadata. A standard that has been established to specify metadata on audio-visual data is MPEG-7 and has been developed by MPEG. This organization committee also developed the successful standards known as MPEG-1 (1992), MPEG-2 (1994) and MPEG-4 (version 2 in 1999). MPEG-7 had a broad impact to experts of various domains of multimedia research. For instance, there are several proposals for storing MPEG-7 descriptions in multimedia databases (e.g., MPEG-7 MMDB [6]). As MPEG-7 bases on XML Schema, one distinguishes research in native XML databases (e.g., Tamino [15]) and on mapping strategies for (object)-relational databases (e.g., Oracle [12]). Then, there are several multimedia applications supporting and using MPEG-7. To mention only a few: VideoAnn [18], Caliph and

E. Damiani et al. (Eds.): SITIS 2006, LNCS 4879, pp. 10–21, 2009.

Emir [10] etc. Finally, many document query languages such as [SDQL 96], [XML-QL 99], [Lorel 00], [YATL 98], [XQL 98], recent W3C [XQuery], etc., have been proposed for XML or MPEG-7 document retrieval. However, these languages cannot adequately support MPEG-7 description queries mainly due to two reasons. Either, they do not support query types which are specific for retrieving multimedia content such as query by example, query based on spatial-temporal relationships, etc. Or, there is no standardized interface defined and each query language or MPEG-7 database offers its own query interface, which prevents clients experiencing aggregated services from various MPEG-7 databases.

At the 77^{th} MPEG meeting in July 2006, the MPEG committee decided to instantiate a call for proposal (N8220 [1]) for a MPEG-7 query format (MP7QF) framework and specified a set of requirements (N8219). The objective of the MP7QF framework is to provide a standardized interface to MPEG-7 databases allowing the multimedia retrieval by users and client applications, based on a set of precise input parameters for describing the search criteria and a set of output parameters for describing the result sets.

In this context, we briefly analyze currently available query languages and approaches for multimedia and XML data in consideration of the introduced MP7QF requirements and propose a new architecture for a MP7QF framework. This paper concentrates on components of the framework and its usability and excludes consciously definitions and explanations of the input and output query format. The query format will be introduced in a separate paper.

The remainder of this paper is organized as follows: Section 2 specifies requirements for an MP7QF framework. Then, Section 3 deals with related work to MPEG-7 query languages and databases. The MP7QF framework is described in Section 4 which is divided in 4 subsections namely, subsection 4.1 describing a multimedia retrieval scenario using the proposed framework, subsection 4.2 presenting the architecture and its components, subsection 4.3 specifying the MP7QF Item and subsection 4.4 dealing with the retrieval of MP7QF-Interpreter. The MP7QF-Interpreter deals as adaptor for MPEG-7 databases and is responsible for transforming MP7QF queries into queries of the query language of the respective database. Finally, this paper is summarized in Section 5.

2 Requirements for an MP7QF Framework

In general, one can distinguish between two main types of requirements: requirements for the framework and requirements for the query language. In the following, selected requirements are explained in more detail. A complete list is given in MPEGs requirement paper (N8219).

2.1 Requirements for the Framework

1.) *Allowing simultaneous search in multiple databases:* The framework should support the distribution of a single query to multiple search engines.

[1] see `http://www.chiariglione.org/mpeg/working_documents/explorations/mp7_qf/mp7_qf_reqs.zip`

2.) *Querying database capabilities:* The framework should allow the retrieval of database capabilities such as supported media formats, supported MPEG-7 descriptors/ descriptor schemes, supported search criteria, etc.

2.2 Requirements for the Query Language

1.) *Query Types*
   a.) *Query by description:* Here, the query language should provide means for the retrieval based on textual descriptions (e.g., names) as well as the retrieval based on desired MPEG-7 descriptions and/or description schemes.
   b.) *Query by example:* The query language should support the retrieval based on representative examples of the desired content.
   c.) *Multimodal query:* The query language should provied means for combining the retrieval based on different media types.
   d.) *Spatial-temporal query:* The query language should support retrieval based on spatial and/or temporal relationships (e.g., Search for images where a red ferrari is in front of a white house.)
2.) *Specifying the result set:* The query language should provide means for specifying the structure as well as the content in the sense of desired data types.
3.) *Querying based on user preferences and usage history:* The query language should consider user preferences and usage history for retrieval.
4.) *Specifying the amount and representation of the result set:* The query language should provide means for limiting the size as well as supporting paging of the result set.

3 Related Work to MPEG-7 Query Languages and Databases

A very good overview of the usability of XML databases for MPEG-7 is provided by Westermann and Klas in [20]. The authors investigated among others the following main criteria: *representation of media descriptions* and *access to media descriptions*. To summarize their findings, neither native XML databases (e.g., Tamino [15], Xindice [16]) nor XML database extensions (e.g., Oracle XML DB [12], Monet XML [14], etc.) provide full support for managing MPEG-7 descriptions with respect to their given requirements. Based on their various limitations (e.g., supported indexing facilities for high-dimensional data), the retrieval capabilities for multimedia data are restrictive.L

In the following, we will have a closer look to the used query languages in various approaches and evaluate them on the requirements presented in Section 2.

XPath [2] (XML Path Language) is a recommendation of the W3C consortium that enables the access to individual parts of data elements in the XML document. In general, an XPATH expression consists of a path (where the main difference to filenames or URIs is that each step selects a set of nodes and not a single node) and a possible condition that restricts the solution set. The main disadvantage of XPATH expression is their limited usability in querying XML documents. For instance, it does not provide means for grouping or joins. In addition, XPath on its own provides no means for querying multimedia data in MPEG-7 descriptions based on the presented criteria.

XQuery [19] is a declarative query language and consists of the following primary areas: The main areas find their counterparts in SQL. For instance, the *for/let* clauses represent the SQL SELECT and SET statements and are used for defining variables respectively iterating over a sequence of values. The *where* clause complies to the SQL WHERE statement by filtering the selection of the *for* clause. The *order-by* finds their analogous in the SQL SORT BY statement and provides an ordering over a sequence of values. The *return* clause respectively SQL RETURN uses a custom formatting language for creating output. A main part of XQuery is the integration of XPath 2.0 and their functions and axis model which enables the navigation over XML structures. Additional parts provide the ability to define own functions analogous to SQL stored procedures and the handling of namespaces. With regard to our requirements, XQuery does not provide means for querying multiple databases in one request and does not support multimodal or spatial/temporal queries.

SQL/XML [3] is an extension of SQL and was developed by an informal group of companies, called SQLX[2], including among others IBM, Oracle, Sybase and Microsoft. A final draft has been standardized as SQL part 14 (SQL/XML) by ANSI/ISO in 2003. Its main functionality is the creation of XML by querying relational data. For this purpose, SQL/XML proposes three different parts. The first part provides a set of functions for mapping the data of (object-) relational tables to an XML document. The second part specifies an XML data type and appropriate functions in SQL for storing XML documents or fragments of them within a (object-) relational model. The third part describes mapping strategies of SQL data types to XML Schema data types. SQL/XML supports the requirement for specifying the representation and content of the result set but on its own (based on its alliance to SQL) there are no means for supporting multimedia retrieval in combination with MPEG-7 descriptions.

The authors in [9] propose an XML query language with multimedia query constructs called MMDOC-QL. MMDOC-QL bases on a logical formalism path predicate calculus [8] which supports multimedia content retrieval based on described spatial, temporal and visual data types and relationships. The query language defines four main clauses: OPERATION (e.g.: generate, insert, etc.) which is used to describe logic conclusions. The PATTERN clause describes domain constraints (e.g., address, etc.). Finally, there exist a FROM and CONTEXT clause which are paired together and can occur multiple times. The FROM clause specifies the MPEG-7 document and the CONTEXT is used to describe logic assertions about MPEG-7 descriptions in path predicate calculus. Of all presented query languages, MMDOC-QL fulfills best the presented requirements. Nevertheless, there are several drawbacks such as simultaneous searches in multiple databases or the integration of user preferences and usage history which are not considered in MMDOC-QL.

XIRQL [4] is a query language for information retrieval in XML documents and bases on XQL [13]. The query language integrates new features that are missing in XQL such as weighting and ranking, relevance-oriented search, data types and vague predicates and semantic relativism. A similar weighting and relevance approach has been introduced in [17].

[2] `http://www.sqlx.org`

Besides, there exist several query languages explicitly for multimedia data such as SQL/MM [11], MOQL [7], $POQL^{MM}$ [5] etc. which are out of scope of this paper based on its limitation in handling XML data.

4 Approach

4.1 Scenario

The following scenario describes a simple interaction process of a user (or client application) with our MP7QF framework during a content-based multimedia retrieval.

Scenario: Query by example with one database. This scenario deals with a query by example request where one image database is included. The query considers the following two low level feature descriptors ScalableColorDescriptor and DominantColorDescritpor for retrieval. In the following, the separate steps are explained in detail:

1. *Connection to the MP7QF Framework:* During the connection process of a user to the MP7QF framework, a new session is established. A session administrates a session id, the connected MP7QF-Interpreter, user preferences and user history. In this context, the user can set its user preferences (as MPEG-7 description) for the established session.
2. *Database selection:* The next step is the selection of desired MP7QF-Interpreter which the framework should consider for retrieval. In this scenario, the user selects the database by its own (see service retrieval scenario presented in Figure 3(a)). Several different connection approaches are described in Section 4.4.
3. *Formulate the query:* This can be realized in combination with a graphical editor (maybe selecting desired descriptors from a tree) or by a command shell. The outcome of this step is an XML instance document based on the input query format (IQF) XML Schema.
4. *Formulate Result Set:* Formulate how the result set should look like (structure and content). The basis for this process is the output query format (OQF) XML Schema.
5. *Transmit Query:* Transmit the query (IQF and OQF instance document) to the MP7QF-Interpreter which is responsible for the transformation of the input query format to the respective target database query language.
6. *Receive Result:* The MP7QF-Interpreter returns the result based on the OQF to the MP7QF framework where its validity is checked. Then the result is forwarded to the user. In this stage, the MP7QF framework extracts user history information and stores the result in the users session information.

4.2 Proposed Architecture

The proposed MP7QF Architecture presented in Figure 1 comprises the main components for fulfilling most of the requirements described in MPEGs requirement paper (N8219) and enables the demonstrated scenario in a standardized way. In the following, the components are described in detail whereas the authors concentrate on components of the framework and its usability and excludes consciously definitions and explanations of the input and output query format. The query format will be introduced in a separate paper.

– **Session Management:** The session management provides means for the storage of user information such as user preferences and user history. In addition, the session management stores the query result for allowing relevance feedback (requirement 4.4.3 of N8219) and the search within the result set of previous searches (requirement 4.4.4 of N8219). For this purpose, our MP7QF proposal introduces a SessionType (see Appendix A) which contains the following elements:

  1.) *ActiveConnection:* The active connection element stores all currently active connections of type ConnectionType. The connections are manipulated (added, deleted) by the use of the Service Management tool. Whenever a retrieval operation is initiated (by calling the Search method), the distributor tool forwards the query to all active connections.
  2.) *QueryResultCache:* The query result cache element stores the result (in form of an OQF type) of the last successful executed retrieval operation. This cache will be used to allow relevance feedback and the search within the result set of previous searches.
  3.) *UserDescription:* The user description complies with the MPEG-7 UserDescription type and manages user data, their preferences and their history. The user preferences and history can be used during the retrieval process to personalize the query (e.g., adding some specific filter operations, etc.).

  An instance of the SessionType is established and maintained for every active user by the session management tool. For this purpose, the session management tool provides the following methods:

  - *SessionID createSession (UserDescription):* This method allows the establishment of a new session for a user. The input parameter contains available user description and corresponds to the MPEG-7 UserDescription descriptor. We suggest that at least the user element is filled. After a successful execution of the operation the created session ID is responded. This session ID applies during the retrieval process for identifying the participating session.
  - *UserDescription closeSession (SessionID):* This method closes an active session which is identified through the given session ID. In addition, the user description is returned which may contain updated history data.
  - *UserDescription closeSession (UserDescription):* Same as before, this method closes an active session. The session is identified through the user description type and here especially by the user element. The main difference to the previous method is, that in this case all open sessions of the respective user are closed.
  - *OQF mp7Search (SessionID, MP7QF Item):* The search method initiates the retrieval process and takes as input parameter the responsible session and the query formulated by IQF and OQF stored in a MP7QF item (see Section 4.3). After completion, the method returns the result as XML document based on the given OQF.

– **Service Management:** This component manages two parts. First, it provides methods for connecting MP7QF-Interpreter to active user sessions. Second, the service management gathers all active MP7QF-Interpreter their ServiceCapabilityDescriptors. For this purpose, every MP7QF-Interpreter must provide such a possibility,

e.g., *ServiceDescriptor getServiceDescription ()*. The management of connections can be realized by using one of the following methods:

- *connectService (SessionID, <vector> ServiceDescriptor):* This method corresponds to the described service retrieval scenario in Figure 3(b) and adds all given MP7QF-Interpreter to an internal list hold at the session management. The ServiceDescriptor contains basic service information such as Host ID, connection information, etc. (see Section 4.4 and Appendix A for a detailed description).
- *<vector> ServiceDescriptor searchService (ServiceCapabilityDescriptor, Boolean):* This method filters the list of registered MP7QF-Interpreter based on a given ServiceCapabilityDescriptor. The ServiceCapabilityDescriptor is described in more detail in Section 4.4. The second parameter specifies a Boolean value which determines whether the filter process should be restrictive meaning that the result set only contains MP7QF-Interpreters which match the retrieval criteria to 100 percent. Otherwise the result set would also contain MP7QF-Interpreters which do not match the criteria for 100 percent (e.g., a specific low level descriptor is missing, a specific IQF operation is not supported, etc.).
- *releaseService (SessionID, <vector> ServiceDescriptor):* This method releases the connected MP7QF-Interpreter from the defined session.

– **Aggregator:** The aggregator is used for combining the result sets of different databases.
– **Distributor:** The distributor splits the user request into calls specific to a certain database. Note: The aggregator and the distributor are only necessary when more than one database is involved in a query.
– **Input Query Format:** The Input Query Format (IQF) specifies the syntax and structure of an MP7 query.
– **Output Query Format:** The Output Query Format (OQF) specifies the syntax and structure of the MP7 query result set.
– **Query Composer:** The Query Composer defines an overall syntax and structure which combines IQF and OQF elements to one request. This request is described and transmitted with the MP7QF Item to respective MP7QF-Interpreter. The Query Composer is also used to assemble the OQF response .
– **Query Interpreter:** The MP7QF Items are transformed into specific bindings (e.g., XQuery, SQL, etc.) of the target databases. The result set is handled by the Query Composer to produce the OQF response.

4.3 MP7QF Item

The MPEG-7 Query Format Item (MP7QF Item) is used for the exchange of MP7 query information between MP7 management tools on different machines and can also be used by the client for interacting with the MP7QF framework. The description of the MP7QF Item bases on the MPEG-21 digital item declaration language (DIDL) standard [1]. The architecture of the MP7QF Item (see Figure 2) consists of the following elements:

– *MP7QF ConnDescriptor:* This descriptor represents the connection information/ session object information.

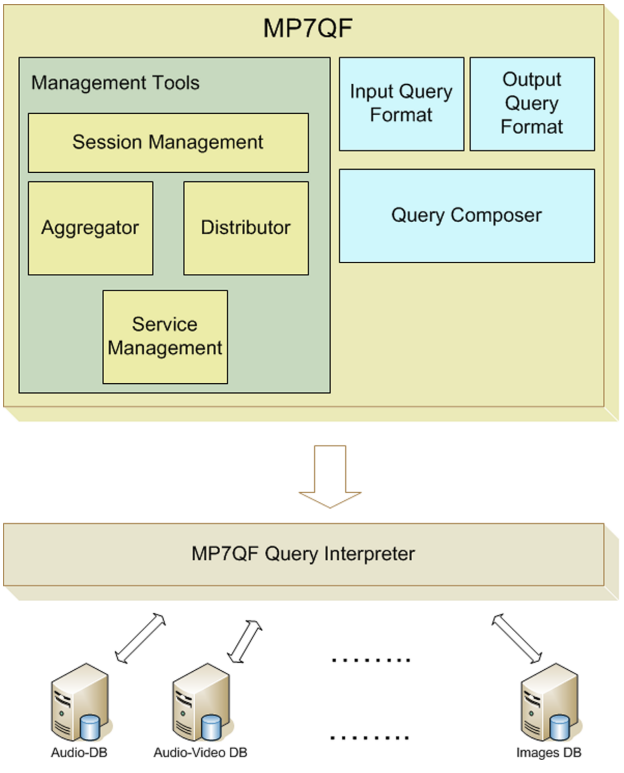


Fig. 1. MPEG-7 Query Format Framework Architecture

- *Query Input Item:* The Item contains a set of Descriptors, including MPEG-7 DescriptionUnits, the actual query and optionally on or more components (in case of query by example).
- *Query Output Item:* The Item contains a Result set Item and optionally an Item carrying presentation information, e.g. an XSLT

An MP7QF Input Item and corresponding output item is stored in a container.

4.4 MP7QF-Interpreter Retrieval

In order to query MPEG-7 databases, we have to introduce an MP7QF-Interpreter which serves as adaptor of the MP7QF framework to available MPEG-7 databases. Here we have to note, that every MPEG-7 database type (e.g., native XML-database, or OR SQL-database, etc.) must provide such an interpreter. During query execution, the MP7QF-Interpreter is responsible for transforming the MP7QF Item into database specific calls. One possible binding of the MP7QF-Interpreter would be the use of Web-Service technology which is explicitly mentioned by the MPEG requirements.

The MP7QF framework supports two different MP7QF-Interpreter retrieval scenarios. First, as displayed in Figure 3(a), the user has the possibility to connect to any

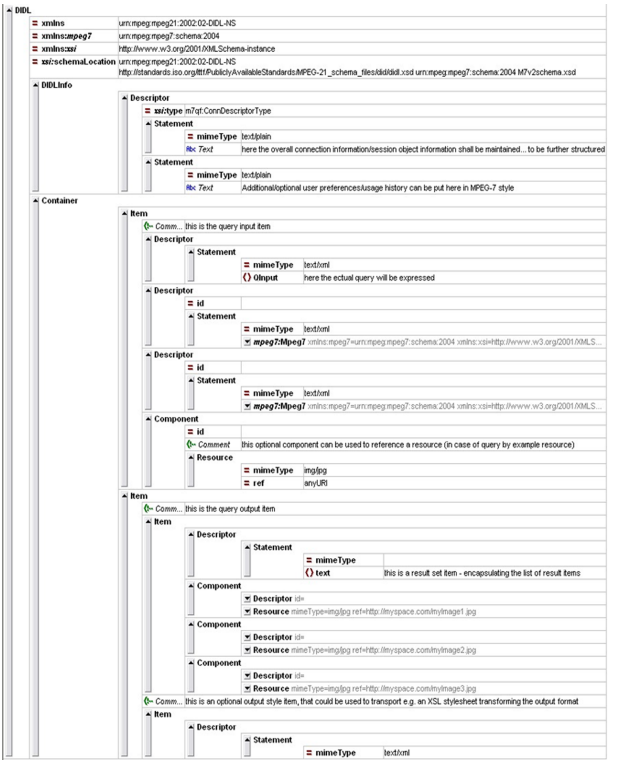


Fig. 2. MPEG-7 Query Format Item

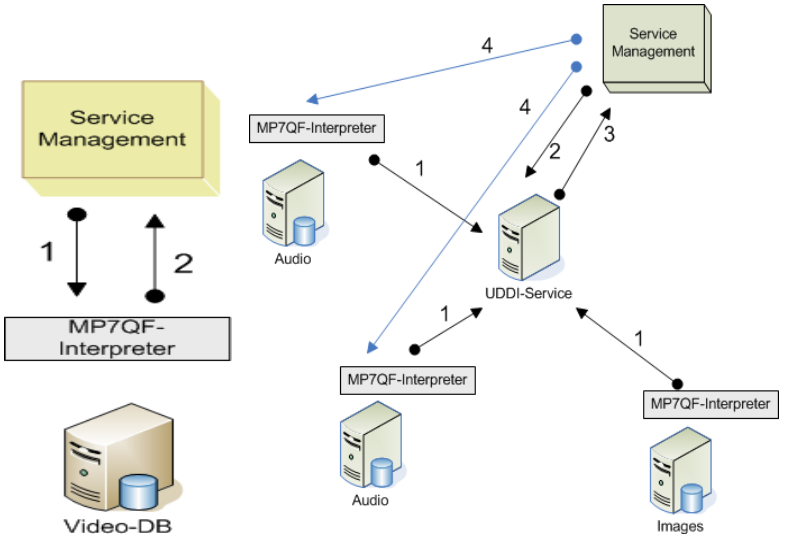
desired database. Second, as shown in Figure 3(b), one can select several databases based on its stored media type (e.g., audio) and by their provided MPEG-7 Profile, IQF query types and operations.

In the following, the different steps of both scenarios are explained in detail:

– **Manual Service Retrieval**
  1. In a first step, the service management component connects to the MP7QF-Interpreter based on a fixed URL given by the user. This is realized by calling the *connectService* method of the service management component.
  2. During this step, the MP7QF-Interpreter transmits its supported service capabilities and connection information in form of the ServiceDescription type to the service management tool. This information is stored for the currently active session (SessionType) in the session management.

– **Automatic Service Retrieval**
  1. First, every participating media database has to register at an exclusive server (UDDI Service [3] in case of Web-Service binding). During this registration all MP7QF-Interpreters send their ServiceDescriptions.

[3] Universal Description, Discovery and Integration (UDDI) see `http://www.uddi.org/`



(a) Manual Service Retrieval **(b)** Automatic Service Retrieval

Fig. 3. Service Retrieval Approaches

2. Then, the service management component of the MP7QF framework queries the UDDI Service based on the user needs. This query can contain requests based on supported MPEG-7 Profiles or IQF query types and operations and is represented by an instance of service capability type.
3. The UDDI Service responds a list of databases fitting the required constraints.
4. The MP7QF framework connects to the proposed databases and receives their capabilities

ServiceDescriptorType. The ServiceDescriptor type combines all information about one MP7-Interpreter and its database. In detail, it includes the connection interface described by the ConnectionType, and the supported capability of the MP7QF-Interpreter and its database in form of the ServiceCapability type. The type is used as either input or output parameter of the following methods: *connectService, searchService* and *releaseService*.

ServiceCapabilityType. The ServiceCapability Type describes information of a MP7QF-Interpreter concerning its supported query types and the underlying MPEG-7 profile. A MP7QF-Interpreter represents the interface to an MPEG-7 database and is responsible for transforming incoming MP7QF Items containing IQF and OQF requests to the respective query language of the target database (e.g., XQuery). Due to the diversity of the MPEG-7 standard it is very likely that an MPEG-7 database only supports a subset of available MPEG-7 descriptors. In series, this is true for the IQF data types and operators. For this purpose, the ServiceCapability Descriptor defines the following elements:

– *SupportedProfile:* This element specifies the MPEG-7 profile the target database provides.

- *SupportedIQFTypes:* This element contains a list of supported input query format types the target database is able to process.
- *SupportedIQFOperations:* This element provides a list of supported input query format operations the target database is able to evaluate.
- *UsageCondition:* This element contains a set of predefined usage conditions such as free of charge, authentication required, payed service etc.

The ServiceCapability Type is used during a service retrieval process in two different cases: First, it describes the capabilities of an MP7QF-Interpreter and the database. Second, a user can formulate its desired requirements an MP7QF-Interpreter must support.

5 Summarization

This paper introduced an MPEG-7 query format (MP7QF) framework based on the defined requirements in the 77^{th} MPEG meeting in July 2006. The framework provides means for session and service management which have been described in detail. In addition, parts of our MP7QF XML Schema have been introduced such as ServiceDescription, ServiceCapabilityDescriptor, SessionType, etc. Nevertheless, it has to be noted that this paper concentrated on components of the framework such as session management and service retrieval and its usability and excludes consciously definitions and explanations of the input and output query format.

References

1. Bormans, J., Hill, K.: Overview of the MPEG-21 standard. ISO/IEC JTC1/SC29/WG11/N5231 (October 2002)
2. Clark, J., DeRose, S.: XML Path Language (XPath). W3C Recommendation (1999), `http://www.w3.org/TR/xpath`
3. Eisenberg, A., Melton, J.: SQL/XML is Making Good Progress. ACM SIGMOD Record 31(2), 101–108 (2002)
4. Furh, N., Grossjohann, K.: XIRQL: A Query Language for Information Retrieval in XML Documents. In: Proceedings of the 24th ACM-SIGIR Conference on Research and Development in Information Retrieval, New Orleans, Louisiana, USA, pp. 172–180 (2001)
5. Henrich, A., Robbert, G.: POQL^{MM}: A Query Language for Structured Multimedia Documents. In: Proceedings 1st International Workshop on Multimedia Data and Document Engineering (MDDE 2001), pp. 17–26 (July 2001)
6. Kosch, H., Döller, M.: The MPEG-7 Multimedia Database System (MPEG-7 MMDB). Journal of Systems and Software (accepted for publication) (in Press by Elsevier) (to appear in spring 2007)
7. Li, J.Z., Özsu, M.T., Szafron, D., Oria, V.: MOQL: A Multimedia Object Query Language. In: Proceedings of the third International Workshop on Multimedia Information Systems, Como Italy, pp. 19–28 (1997)
8. Lui, P., Charkraborty, A., Hsu, L.H.: Path Predicate Calculus: Towards a Logic Formalism for Multimedia XML Query Language. In: Proceedings of the Extreme Markup Languages, Montreal, Canada (2000)

9. Lui, P., Charkraborty, A., Hsu, L.H.: A Logic Approach for MPEG-7 XML Document Queries. In: Proceedings of the Extreme Markup Languages, Montreal, Canada (2001)
10. Lux, M., Klieber, W., Granitzer, M.: Caliph & Emir: Semantics in Multimedia Retrieval and Annotation. In: Proceedings of the 19th International CODATA Conference 2004: The Information Society: New Horizons for Science, Berlin, Germany, pp. 64–75 (2004)
11. Melton, J., Eisenberg, A.: SQL Multimedia Application packages (SQL/MM). ACM SIGMOD Record 30(4), 97–102 (2001)
12. Murthy, R., Banerjee, S.: XML Schemas in Oracle XML DB. In: Proceedings of the 29th VLDB Conference, Berlin, Germany, pp. 1009–1018. Morgan Kaufmann, San Francisco (2003)
13. Robie, J.: XQL (XML Query Language) (1999), `http://www.ibiblio.org/xql/xql-proposal.html`
14. Schmidt, A., Kersten, M.L., Windhouwer, M., Waas, F.: Efficient relational storage and retrieval of XML documents. In: Suciu, D., Vossen, G. (eds.) WebDB 2000. LNCS, vol. 1997, p. 137. Springer, Heidelberg (2001)
15. Schning, H.: Tamino - a DBMS designed for XML. In: Proceedings of the 17th International Conference on Data Engineering (ICDE), pp. 149–154 (April 2001)
16. Staken, K.: Xindice Developers Guide 0.7. The Apache Foundation (December 2002), `http://www.apache.org`
17. Theobald, A., Weikum, G.: Adding Relevance to XML. In: Suciu, D., Vossen, G. (eds.) WebDB 2000. LNCS, vol. 1997, pp. 35–40. Springer, Heidelberg (2001)
18. Tseng, B.L., Lin, C.-Y., Smith, J.R.: Video Personalization and Summarization System. In: Proceedings of the SPIE Photonics East 2002 - Internet Multimedia Management Systems, Boston, USA (2002)
19. W3C. XML Query (XQuery). W3C (2006), `http://www.w3.org/TR/xquery/`
20. Westermann, U., Klas, W.: An Analysis of XML Database Solutions for the Management of MPEG-7 Media Descriptions. ACM Computing Surveys 35(4), 331–373 (2003)