Standardized Multimedia Retrieval based on Web Service technologies and the MPEG Query Format

Mario D’oller, Kerstin Bauer, Harald Kosch
Chair of Distributed Information Technology
University of Passau, Germany
{forename.surname}@uni-passau.de
Matthias Gruhne
Institut Digitale Medientechnologie (IDMT)
Fraunhofer Ilmenau, Germany
ghe@idmt.fraunhofer.de

ABSTRACT: In recent years, the growing number of digital audiovisual media files available over the internet or even on users hard discs is overwhelming. In order to support efficient storage and retrieval of those data, several comprehensive and rich multimedia retrieval systems (MMRS) have been introduced. Unfortunately, a standardized query format does not yet exist and almost every retrieval framework provides its own proprietary solution. Therefore, the ISO/IEC SC29 WG11 committee decided to contribute to this application by establishing the MPEG Query Format (MPQF). The MPQF is currently in Final Committee Draft (FCD) status and provides besides the standardization of messages from and to multimedia services also functionalities for service discovery, aggregated services and definition of service capability descriptions. In this context, the paper briefly introduces the MPQF and concentrates thereby on the management part. Based on this, a Web Service based framework is presented that realizes the management functionalities of MPQF. One of its central novel features is the distribution of a MPQF request to multiple multimedia services and the aggregation of individual result sets.

Categories and Subject Descriptors
H.5.1 [Multimedia Information Systems]; Distributed data structure; H.3.5 [Online Information Services]; Web-based services; H.5.2 [User Interfaces]; Standardization

General Terms
MPEG Systems, Multimedia Services,
Keywords: Audio visual media, ISO multimedia standards, MPEG query format

Received 10 June 2007; Reviewed and accepted 7 Sep. 2007

1. Introduction

During the last years, a variety of multimedia information has been brought to recipients due to the growing abilities of computer, telecommunications and electronic industry. Furthermore a high amount of digital video and audio information became publicly available over the years. By having more and more content digitally available, the ability increased to deliver rich information to customers. But at the same time, we are facing several major barriers which prevent us experiencing a broad and unified access to different multimedia data collections.

First, due to intensive research during the last decade, a multiplicity of multimedia retrieval systems (MMRS) has been emerged. Some concentrate on a specific domain (e.g., only audio data is considered) others are trying to provide access to multimodal data. Current promising systems are for instance Cortina 1 [18], which combines text and image retrieval techniques or SIMPLicity 2 [25], which is an image retrieval system that bases on low level features (color, texture, shape) in combination with region identification and classification. R.C. Veltkamp and M. Tanase compared (updated in 2004) 43 available products [22].

Second, in order to improve retrieval efficiency, MMRS apply metadata descriptions of multimedia data. Some apply proprietary description models, whereas others focus on the use of available standard technologies such as MPEG-7 3, METS 4 or Dublin Core 5. Recent MPEG-7 databases supporting multimedia retrieval are for instance MPEG-7 MMDB [5] or PTDOM [27].

Third, almost every MMRS provides a different retrieval interface. Some base on well known query languages such as SQL/MM [15], MMDOC-QL [14], etc., others use proprietary request protocols. However, a client faces a variate set of different interfaces which excludes the interoperability among those systems.

Finally, the data contained in database systems are very heterogeneous, too. Multimedia data can be, for example, audio, images, videos, text documents or any combination of them. Varying kinds of data require different features from query languages or annotation descriptions.

In order to cope with this situation and to overcome these barriers, a unified access to MMRS needs to be introduced. Therefore, the MPEG consortium decided to focus on the standardization of a new query format. The MPEG Query Format [1] (MPQF) is described by an XML Schema and bases on three main pillars. Besides standardizing the format of multimedia query requests (by the Input Query Format) and its response messages (by the Output Query Format), MPQF also provides means (by Query Management Tools) for a standardized service discovery, service selection and the implementation of aggregated services. Related to this, the paper introduces all three parts of MPQF and a corresponding Web Service framework implementation. One of its central novel features is the distribution of a MPQF request to multiple multimedia services and the aggregation of the individual result sets.

1 http://water.ece.ucsb.edu/Cortina/
2 http://wang.ist.psu.edu/IMAGE/
3 http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm
4 http://www.loc.gov/standards/mets/
5 http://dublincore.org/
result sets. For this purpose, current result aggregation algorithm are examined for their use with MPQF services. The remainder of this paper is organized as follows. In Section 2 a brief overview of XML based query languages is given and current middleware models and result aggregation techniques are reviewed. Then, in Section 3 the MPEG Query Format is introduced where a specific focus is pointed to the management part. An examination of the use of various result aggregation algorithm for MPQF is provided in Section 4. The implementation of the management part as a Web

2. Related Work

As outlined beforehand, metadata descriptions (e.g., MPEG-7) of multimedia data based on XML are used in order to improve the retrieval efficiency. An analysis of XML databases for MPEG-7 provided by Westermann and Klas in [26] bares several shortcomings of native XML databases (e.g., Tamino [20]) and XML database extensions (e.g., Oracle XML DB [16]). Therefore, several systems explicitly for MPEG-7 have been emerged (e.g., MPEG-7 MMDB [5] or PTDOM [27]). In addition, several XML based multimedia query languages have been introduced. There are for instance: XQuery [23] which consists of four primary areas where every part finds 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. A main part of XQuery is the integration of XPath 2.0 and their function and axis model which enables the navigation over XML structures. XQuery on its own has several limitations in expressing multimedia queries (e.g., spatio-temporal, range based queries, etc.). Nevertheless, there is ongoing work in this direction. For instance, the authors in [30] describe an XQuery extension for MPEG-7 vector-based feature queries. Furthermore, the authors in [7, 6] adapted XQuery for the retrieval of MPEG-7 descriptions based on semantic views. Its adaptation, called Semantic Views Query Language (SVQL) is specialized for retrieving MPEG-7 descriptions in TV news retrieval applications.

Other approaches for MPEG-7 (e.g., MMDOC QL [14]) or XML in general (e.g., XIRQL [9]) provide some local solutions and suffer by their low distribution.

Besides, there exist several query languages explicitly for multimedia data such as SQL/MM [15], POQLMM [11] etc. which are out of scope of this paper based on its limitation in handling XML data.

As mentioned above, multimedia collections and their retrieval systems are spread over the whole globus. The concatenation of those distributed databases and systems is a well known research area (see e.g. [17]).

Besides, basic, middleware frameworks such as CORBA 6, several systems, especially designed for the connection of multiple multimedia database systems, exist (e.g., dLIMIT [13], MOCHA [19] or HERON [24]).

In all these systems, merging results from different sources plays an important role. Result aggregation is necessary when several results from different locations need to be merged to one single result. Current approaches (see for instance [3]) can be classified according to the following characteristics:

• The degree of overlap [29] of the content among the involved database systems. The data can be identical, overlap or pairwise disjoint.

• The degree of interaction can be taken into consideration.

They can be either cooperative/integrated or uncooperative/isolated [4]

• The amount of additional information such as analysis of training data, that the algorithm relies on (see e.g., [4]).

3. MPEG Query Format

The normative parts of the MPEG Query Format define three main components: The Input Query Format provides means for describing query requests from a client to a MMRS. The Output Query Format specifies a message container for MMRS responses and finally the Query Management Tools, where its main purpose is to organize all open issues before a query is sent. This includes functionalities such as service discovery, service aggregation and service capability description. Note, the term service refers to all MMRSs including single databases as well as service providers administrating a set of MMRSs.

3.1 Input Query Format

As outlined in Figure 1, the input query format consists of three (optional) components, namely the QFDeclaration, the OutputDescription and the QueryCondition.

The QFDeclaration part supports the stockage of elements (e.g., the description of a color layout) that may be used for several times within the query condition or output description part. This ensures the reuse of already defined descriptions and flattens down the whole query.

>Figure 1. MPQF Query Architecture

The main purpose of the OutputDescription part is to provide a means for a user agent to specify the content as well as the structure of the result set. This is realized by either absolute or relative XPath expressions pointing to the desired information at the target XML schema (see Code 1 for a MPEG-7 example). By using absolute expressions, structural information can be applied. The target XML schema can be constituted by the outputNameSpace attribute and therefore opens MPQF for all XML based multimedia metadata descriptions.

>Code 1 Example output description

```
<OutputDescription maxItemCount="10" maxPageEntries="10" freeTextUse="true" outputNameSpace="urn:mpeg:mpeg7:schema:2004">
    <ReqField typeName="MediaLocatorType">/boxshadowdwn</ReqField>
    <ReqField typeName="Volume">/boxshadowdwn</ReqField>
    <ReqField typeName="MediaUri" />
</OutputDescription>
```
Besides the name space attribute, the standard provides further optional attributes for signalling the maximum amount of result items (maxItemCount), paging functionality (by maxPageEntries) and the use of thumbnails, free text or other media resources.

The QueryCondition part determines filter criteria in order to prepare the result set according to users interests. For this purpose, a user first can denote the focus of retrieval by specifying the mime types of some target media types the query should respond and by specifying its granularity (e.g., concentrate on whole video or video segments). Of course, the second point is strongly related to the power of the underlying target XML schema. The query condition itself is described by the Condition element and builds a filter tree consisting of query types (e.g., QueryByMedia) and expressions (arithmetic or string) in its leaf nodes concatenated by boolean expressions in its inner nodes. A preference value can be assigned to every node within the filter tree indicating which filter criteria is more important than others.

Let us suppose, one is interested in the following query: Give me all videos of type MPEG where George Clooney poses as actor and one scene plays in the city Paris! A possible query realizing the given example which target to a MPEG-7 database is presented in Code 2.

The query condition first filters available media types with the TargetMediaTypes element to videos of type MPEG (MPEG-1, MPEG-2, etc.). Second, the Condition element triggers a combined retrieval (by AND) for videos by the use of two QueryByDescription query types. The first QueryByDescription query type receives as input a MPEG-7 description (with ID agent) that restricts the movies to those where George Clooney is an actor. This is realized by using the classification scheme for actor at the Role element and specifying the person by the Agent element. The second QueryByDescription query type narrows down the retrieval to movies where at least one scene plays in the city Paris. Therefore, the available MPEG-7 EventType type is used where the SemanticPlace element specifies the city Paris and its relation (Relation element) the desired characteristic.

3.2 Output Query Format

The Output Query Format is defined by the QueryOutputType type. It provides a container for all response related information. Every result set consists of one optional GlobalComment element containing any descriptive information a service want to respond and a set of ResultItem elements substituting the individual results. Furthermore, a result set can have a SystemMessage element in case of errors or warnings. Besides, the result set can be split into different pages which is indicated by the occurrence of the currPage (current page number) and totalPages (total number of pages) attributes. The validity of the result is notified by the expirationDate attribute.

The individual items are described by the ResultItem type. This type encapsulates optionally MPEG-7 feature descriptions or any other metadata, that is not specified by MPEG. Each item has a required attribute recordNumber which is unique within a result set and signalizes its position. In addition, there are three optional attributes, namely rank indicating the overall rank of this item within the result set, confidence indicating the services confidence value for the match and the originID determining the service. Besides feature descriptions, every item can contain a Thumbnail, a Comment and a text based result (within the TextResult element).

The result set presented in Code 3 might be a possible response to the query request demonstrated by Code 2.

3.3 Management Part

Whereas the input and the output query format directly focus on the multimedia retrieval itself (the request and its response), the management part copes with the problem of
searching and choosing desired multimedia services for retrieval. For this purpose, the standard provides a service capability description (see Code 4) and four different service discovery requests. In the following, the main elements of the management part are explained in detail:

- **Service Capability Description**: A service capability description determines what kind of retrieval functionality the respective service supports. A service capability description can be composed of the following elements (see Code 4 for an example description): The SupportedQFProfile points to the profile version of MPQF. The available metadata description at server side is given by the SupportedMetadata element. Note, that the metadata description must base on XML Schema. The supported media formats are indicated by mime types listed in the SupportedExampleMediaTypes and SupportedResultMediaTypes elements. The available constituents for formulating the search criteria are listed by the SupportedQueryTypes and SupportedExpressions elements. Note, that the information bases on classification scheme terms (e.g., urn:mpeg:mpqf:2008:CS:full:100.3.6.1 corresponds to QueryByMedia). Potential usage restrictions on the level of individual query types and expression (e.g., payment is needed) can be assigned by using the UsageConditions element.

- **Service Discovery**: The aim of service discovery is to provide some kind of registry functionality where services can register and lend themselves to a broader community. In series, a client can perform service discovery tasks in order to find the MMRS of desire. Our framework implements the service discovery functionality in terms of the Service Management Web Service.

### 4. Result Aggregation Algorithm for MPQF

Result aggregation is one of the challenging issues in information retrieval when multiple distributed heterogenous retrieval systems are involved. In general, it ensures the merging of individual result sets to a single one. This process comprises several obstacles such as the occurrence of duplicates when the individual data sets overlap. The available approaches can be classified according to the mentioned characteristics presented in Section 2. In this section, we examine which of these algorithms can be applied to MPQF services. We assume in our examination that only information contained in the OutputQueryType is available. Furthermore, the MPQF services can include any kind of content like video, audio and image files. But since no centralized database is available, it is unknown whether the content of the services overlap or not. Therefore, the following algorithms and techniques can be applied for MPQF services:

- **Combination techniques** [12]:
  Some of the earliest merging algorithms were presented in [8], e.g., Min/Max: Minimum/Maximum value of the individual score values, CombSum: Sum of the individual score values (analog to averaging) and CombMNZ: Multiplication of the
total scores and the number of non-zero-scores (analog to weighted averaging). In [12], it was shown that CombMNZ is the best out of these merging algorithms. The problem of these combination techniques is that they rely on the presence of a lot of overlap among the databases.

Regarding to MPQF, they are applicable under the assumption that duplicate result items can be detected. Duplicates play an important role in these methods. If duplicate detection is provided then these methods represent good alternative result aggregation algorithms.

- Weighted combination techniques [28]: These methods (see [28]) are an improvement of the combination techniques described above. The weights can be estimated very conveniently for each query and there is no need for training queries. For instance, the CombMNZ has been improved by the use of weights. The new method WMNZ (weighted MNZ) first sums the scores of the retrieved documents before summing the weights of each database that has a non-zero score for this document. Finally, both sums are multiplied.

Regarding to MPQF, these methods are applicable for MPQF databases if the weight values for all databases can be determined.

- Borda-fuse voting model [2]: This procedure was introduced for metasearch engines by [2] and relies on the Borda-fuse voting model (based on democratic election strategies). In this procedure, the voters rank a set of c candidates according to their preferences. The top ranked element gets c points, the second c -1 points, and so on. Finally, the candidates are ranked in order of their total points and the one that received the most points is at the top of the final result list. The merging strategy has several advantages: First, only the rank information is necessary for the aggregation process. No further information is required, especially neither relevance scores nor training data are needed. Second, the algorithm is quite simple and efficient.

Regarding to MPQF, this model is similar to the Max method of the combination techniques and therefore, as only the rank of the elements is used, this method can be applied to MPQF services as well.

- Data fusion techniques to retrieve documents of the Web [21]: In this approach, the individual retrieved elements can be duplicates or be disjoint. Furthermore, different weighting schemes can be used by the individual sources. The merging strategies proposed by the authors not only use the rank of the result documents but also their title and summary, with or without downloading the documents.

Regarding to MPQF, one has to consider that result items of MPQF services can consist of image, video or audio data. If only raw data (e.g., the image itself) is received then this approach can only rely on rank information and it degenerates to the approaches mentioned above. Nevertheless, in many cases there is some additional information that can be used for this merging technique: multimedia data is often annotated with metadata. One still needs to decide which metadata can be used and whether the values are comparable if different metadata standards are involved.

- Shadow Document Method [29]: This method can be used for databases which contain partial overlaps, but require a method in order to detect duplicates. The processing of the duplicates is a major issue in this approach and depends on the decision whether its occurrence increases the evidence of relevance. The advantages of this merging method are that no training data and no downloads are necessary. The only information needed from the retrieved results are the score values of the documents.

Regarding to MPQF, this algorithm could also be applied for result aggregation since it only uses score values. Nevertheless, an open question remains: the detection of duplicates and the assignment of an adequate score value for the shadow documents (see [29]) needs to be examined.

- Round Robin [3]: This algorithm is one of the most well-known approaches and selects the individual items of the individual result sets in the order of their rank (ascending) in a round robin fashion [3]. A detection of duplicates does not take place. So if there are any duplicate elements in the individual results, they will also occur as duplicates in the merged result. The Round Robin algorithm works best, if there are no duplicates and if the individual result lists from the databases contain an almost equal amount of elements. If the distribution is bad, the algorithm will produce a worse result.

Regarding to MPQF, this approach can be used as it only requires the rank information from the databases. The elements from the individual results can be merged easily. The duplicates still remain a problem. They are not considered in this algorithm. But it is not unlikely that MPQF services contain many identical elements. Even so, this algorithm was chosen for the framework in this paper. The detection of duplicates is not in the focus of this work. The following aggregation algorithms are not applicable, at least not without considerable expenses:

- Raw score merging [3]: This method is very simple. It bases on the hypothesis that the same search model is used in all databases and thus the score values returned are comparable. But since this is almost never the case, this technique is seldom used [3]. The same argument is applicable for MPQF.

- Reference statistics [4]: This algorithm is designed for isolated sources and uses a designated centralized database to collect relevant statistics for some of the documents (e.g. top 10).

Regarding to MPQF, this result aggregation method can not be applied (based on our assumptions) as it relies on a centralized database that collects statistical information about the contents of all MMRs. As we aim for a purely distributed approach, a central database is not adequate.

- Feature distance ranking algorithms [4]: This approach is also designed for isolated sources and requires the partial download of the retrieved documents. Obviously, as we handle multimedia data, partial download is not envisaged. Besides, a feature vector has to be applied to determine the quality of a result item. Defining such a feature vector will be very difficult in the case of heterogeneous multimedia data.

5. Web Service based framework for MPQF

The implemented framework relies on Web Service technologies and consists of three principal components (see Figure 2): Service Management, Retrieval Management and Validator Web Service. Web Services have many advantages. They are, for example, very flexible, powerful and can be combined easily with each other. Using the Apache Axis7 environment and its WSDL2Java command, the types of the MPQF XML Schema are transformed into Java classes which supersedes XML/String parsing.
5.1 Service Management Web Service

The Service Management Web Service realizes service discovery and offers registry functionality. It provides two methods for MMRSs and clients.

A MMRS can register (by calling the registerService method) at the service management. The method receives as input an instance of the MPQF AvailableCapabilityType type which is a coverage for the service capability description (see Code 4).

Clients are able to search (by calling the searchService method) for particular MMRSs at this registry. The method requires as input a valid MPQF input management query (see e.g., Code 5) and results in a list of matching and available services delivered as a valid MPQF management output. The input management query supports four different standardized types of requests.

First, an empty query (only the < Input/> tag is provided) will result in the whole list of available MMRSs. Second, a query containing one or more service IDs results in the descriptions of the requested service IDs if available. The third case is analog to the second one unless no service IDs are given but the desired capability of the MMRS is specified (see Code 5). The client will receive all MMRSs that satisfy the query. The fourth case combines the second and third one. Clients specify one or more services IDs as well as one desired capability and receive as response the registered MMRSs that satisfy all specified criteria.

Code 5 Example Input Management Query

```xml
<MedQuery>
  <Management>
    <Input>
      <DesiredCapability>
        <SupportedMetadata>urn:mpeg:mpeg7:2004</SupportedMetadata>
        <SupportedMetadata>urn:mpeg:mpegq:2008:CS:full:100.3.6.4"/>
        </DesiredCapability>
      </Input>
    </Management>
  </MedQuery>
```

5.2 Retrieval Management Web Service

The Retrieval Management Web Service realizes service selection and result aggregation. This Web Service provides one method (search) in order to query multiple MMRSs at a particular time. The method requests a MPQF input query and responses with the merged result sets of the selected MMRS. The selection of MMRSs is realized by using the ServiceSelection element in the input query multiple times. The two principal components of this Web Service are the Aggregator and the Distributor (see Figure 2). For each incoming query the retrieval management instantiates one Aggregator and as many Distributors as MMRSs are specified inside the query. The task of the distributors is to forward the query to the respective MMRS and to wait for the query result. After all query results are returned to the distributors, the aggregator merges them to a single result.

In general, several algorithms can be considered for result aggregation as outlined in Section 2. As only the information provided by the MPQF XML-Schema is available, only a few result aggregation algorithm can be taken into account.

5.2.1 Requirements on a full-fledged MPQF based service aggregation

The management part of MPQF introduces several concepts a framework model must consider. The following list provides a brief overview:

- **Mode of communication**: The mode of communication describes how messages between clients and MMRSs can be exchanged. It is determined using the attribute immediateResponse of the InputQueryType type. If the value of this attribute equals to true or if it is not present, the synchronous communication mode will be used. Otherwise the mode is set to asynchronous communication. The main difference is in the point in time when the result set is forwarded to the client. In the synchronous mode, the query request is executed and the result set is received immediately. In asynchronous mode, the result set needs to be fetched by the client in a separate operation (by using the FetchResult operation).

- **Timeout**: The timeout value is no mandatory attribute of the InputQueryType type. If it is not specified by the client, it is up to the framework to assume a default timeout value. This attribute can be used in the synchronous as well as in the asynchronous communication mode and depending on this, its use is interpreted differently. In the synchronous type of communication, a client defines the timeout value as the duration of time that the database servers are allowed to spend for information retrieval. If the query result from a database takes longer than the specified timeout value, the client is assumed not to be interested in the query result any more. In the asynchronous communication mode, the timeout value specifies the duration of time after which the client will send its FetchResult query to retrieve the merged query result. Thus, the timeout attribute determines the time that the database servers is allowed to take for query processing.

- **Result buffering**: When retrieving query results from several databases, it is inevitable for a framework to provide a particular capacity to buffer the individual query results. This is strongly required as the query results from different databases have to be merged before they are returned to the client. Until the merging process is finished, the results not already integrated have to remain at the Web Service. Hence, the timeout value is very important for result buffering as it...
defines how long the intermediary results have to be hold until the final merged query result can be returned to the client.

**Relevance feedback:**
This paradigm is very popular in multimedia retrieval systems [10]. It allows the comprehension of the client by evaluating the result set and choosing good and/or bad examples for a continuous retrieval. The MPQF supports this paradigm by the QueryByRelevance feedback query type which receives as input a list of recordNumbers indicating the selected result items.

### 5.2.2 Impact of result aggregation on MPQF result sets

The algorithm chosen for this framework is Round Robin [3]. It works straightforward by simply choosing the next element of all result sets in turn (see Figure 3). The following list iterates over all attributes and elements of the result sets and demonstrates what kind of modifications are needed in order to generate a valid output.

- **GlobalComment**: The GlobalComment elements of all individual result sets are simply concatenated as they are represented as strings. In doing so, the original information of the services is not lost.

- **SystemMessage**: Since it is hard to merge multiple SystemMessage elements to a single one, a new SystemMessage is created where all individual messages are combined. Unfortunately, the information which service has stated which system message is lost. In order to keep this information, one solution is to modify the description of the system message which is not standard conform.

- **currPage**: This attribute is supposed to be the same for all individual results, so simply one of the present values is chosen.

- **totalPages**: This attribute indicates the total number of pages retrieved from a service. During the aggregation process this value is set to the sum of the single total pages that were returned. Thus, the final value of totalPages might be bigger than the actual amount of pages. The overall amount of result items should not exceed the given maxItemCount and maxPageEntries values of the input query. Therefore, the new totalPages is accumulated by the sum of all individual totalPages, but must hold the following condition: totalPages <=\text{maxItemCount} \times \text{maxPageEntries}.

- **expirationDate**: This attribute describes the date when the service expires the result. The new expiration date is set to the next date to take place.

- **recordNumber**: The attribute recordNumber is specified to be a numeric value in strictly ascending order. This value has to be adjusted when several result sets are merged to a single one. Its occurrence is required and after the aggregation took place, it has to be strictly ascending again. This uniqueness is necessary since the recordNumber is referenced when relevance feedback is used. As mentioned previously, the overall amount of result items should not exceed the maxItemCount. Therefore, a merging process can only select items from the individual result sets until this upper boundary is reached. How this selection is realized, depends on the used aggregation algorithm. Another problem tackles the process of an ensuing relevance feedback operation. A relevance feedback operation selects several result items by its recordNumber value and activates a new retrieval based on the given good and/or bad examples. As the record number is adjusted at the merged result set, a mapping table has to be maintained in order to extract the original record numbers. Nevertheless, this solution only solves a part of the problem. A relevance feedback request only contains a collection of record numbers and it assumes that the target MMRS has the original result set by hand and can extract additional information of the selected items for retrieval. This assumption is only true in a single MMRS scenario, but not in a scenario where multiple services are involved and the result merging process is realized by a third party implementation (as in our scenario).

- **originID**: The attribute originID has to point to the respective MMRS wherefrom the result was retrieved. Its assignment is not mandatory for the MPQF standard. If the MMRS already assigned its ID, it does not have to be changed during the result aggregation. Otherwise it is activated. This is necessary since the client needs to know the origin of a result item, e.g. for a following relevance feedback.

- **rank and confidence**: The attributes rank and confidence do not have to be adjusted when several result items are aggregated. Their occurrence is optional and their value depends on the database from where they were retrieved.

- **mpqfID**: This attribute identifies a single message within the communication process between clients, services and middleware components. During the result aggregation process, the framework is confronted with a set of different mpqfIDs as response to one query request. Therefore, a new ID needs to be generated which serves as identifier for the merged result set. In this case, several issues are related. On the one side, in asynchronous communication mode the client is not able to fetch the individual result sets on its own and on the other side a direct relevance feedback operation to the involved MMRS is also not possible. Therefore, a mapping (merged result set ID to the IDs of the individual messages) needs to be maintained at the Retrieval Management Web Service until the expiration time is reached and all proximate messages need to be processed by the framework.

As already examined, the Round Robin algorithm works best when the contents of all MMRSs are almost pairwise disjoint. Nevertheless, duplicate elements in the individual result sets will lead to duplicate elements in the final query result. This problem occurs since no duplicate elimination takes place. After the result aggregation is successfully completed, one final query result is returned to the requesting client. For the time being, only the synchronous communication mode was implemented. Therefore, the retrieval management and the clients wait for the query results to be returned from the MMRS. The client can specify a timeout value to determine the maximum time span that the MMRSs are allowed to take.

### 5.3 Validator Web Service

The Validator Web Service provides a means for a syntactic verification of a MPQF query. As MPQF bases on XML Schema, the validator has to deal with XML instance documents. For this purpose, the current implementation embeds the XSV \(^8\) XML validator. Ongoing implementations will also consider a rule based verification of semantic relationships and conditions within a MPQF query.
6. Summarization

This paper presented the implementation of a framework for the MPEG Query Format (MPQF) based on Web Service technologies. MPQF is a query language based on XML Schema and standardizes the interaction among clients and multimedia retrieval systems. The query language provides means for interoperability among distributed multimedia search and retrieval services and makes intelligent content navigation in e.g., MPEG-7 enabled multimedia repositories possible that are loosely connected via distributed heterogenous networks. The introduced framework implements service discovery, service selection and service aggregation. Furthermore, this paper provided an analysis of existing result aggregation algorithms for the use with MPQF and identified the Round Robin approach as valuable. In future extensions, the framework needs to support asynchronous mode and the MPQF-FetchResult operation. Furthermore, an advanced result aggregation algorithm needs to be taken into consideration as the current approach does not include the elimination of duplications. In addition, it will be investigated how statistics and ratings can be obtained by the evaluation of the MPQF-RelavenceFeedback operation in order to judge the confidence of MMRSs.

References


