UNIVERSITE DE GENEVE

CENTRE UNIVERSITAIRE
D'INFORMATIQUE
GROUPE VISION

Date: First version: October 6, 1999
This version: October 28, 1999
N° 99.04

TECHNICAL REPORT

VISION

MRML: Towards an extensible standard for multimedia querying and benchmarking

Draft Proposal

Wolfgang Müller Zoran Pečenović[1] Arjen P. de Vries[2]*
David McG. Squire Henning Müller Thierry Pun

Computer Vision Group
Computing Science Center, University of Geneva
24 rue du Général Dufour, CH - 1211 Geneva 4, Switzerland

[1]Laboratoire de Communications Audio-Visuelles and Groupe de l'Ergonomie des Systèmes Intelligents
Ecole Polytechnique Fédérale de Lausanne, Switzerland

[2]Department of Computer Science
University of Twente, The Netherlands

e-mail: Wolfgang.Mueller@cui.unige.ch,
Zoran.Pecenovic@lcav.de.epfl.ch

*Arjen was happy to test that thanks still worked.

Abstract

In recent years, the need for databases which query multimedia data by content has become apparent. Many commercial and non–commercial research groups are trying to fulfill these needs.

The development of research can be described as moving in two directions

- search for new, useful query and interaction paradigms
- deeper research to improve the performance of systems that have adopted a given query paradigm.

The search for new better performance given a query paradigm has led to "clusters" of systems which are similar in their interaction with the user, and which give a certain set of interaction capabilities to the user.

It is already visible, that research will move towards systems which enable the user to formulate multi–paradigm queries in order to further improve results.

As a consequence of the above, there is the need for

- A common mechanism for shipping multi–paradigm queries and their results , which assures that the right query processor processes the right query.
- For each paradigm a common language, which enables to formulate queries in this paradigm.

This mechanism, of course, has to be extensible in order not to constrain ongoing research.

We propose MRML (Multimedia Retrieval Markup Language), a new XML-based language that provides an extensible mechanism for shipping multi–paradigm queries. As a demonstration of its use, the current version of MRML already provides a detailed extensible language for shipping Query–By–Example (QBE) queries in a CBIRS (Content Based Image Retrieval) context.

In this article we highlight

- which advantages the use of MRML can have for use in the CBIRS and other communities. Especially we show how our groups (Lausanne, Geneva) use MRML to minimize programming work and to share user data, and how *MRML can be used for a common CBIRS benchmark.*
- that MRML is explicitly designed to minimize the programming work imposed by its use,
- how MRML can be extended towards other query paradigms,

In this paper, we also propose a development model, that will keep MRML an open standard that is usable while it grows.

Keywords: multimedia database,interoperability,query language

1 Introduction

The development of research can be described as moving in two directions

- search for new, useful query and interaction paradigms
- deeper research to improve the performance of systems that have adopted a given query paradigm.

The search for new better performance given a query paradigm has led to "clusters" of systems which are similar in their interaction with the user, and which give a certain set of interaction capabilities to the user.

It is already visible, that research will move towards systems which enable the user to formulate multi–paradigm queries in order to further improve results.

As a consequence of the above, there is the need for

- A common mechanism for shipping multi–paradigm queries and their results , which assures that the right query processor processes the right query.
- For each paradigm a common language, which enables to formulate queries in this paradigm.

Fulfilling this need would also enable the different communities

- to share user interfaces. At present almost every group has its own interface suiting its purposes. However, many interfaces are very much alike.
- to share implementations without sharing the code. Comparing the resulting systems of different groups of a research domain would be easier, if one could make them accessible to outside scripts without publishing the actual code.

The query shipping–mechanism should be designed in a way that it does not constrain ongoing and future research regarding both the search for optimal query formulation for each paradigm and the research for new query paradigms for MMDB queries. In short, the query–shipping mechanism should *separate the communication problem from the query formulation problem*, letting research groups evolve freely to find good domain specific query formulation schemes.

MRML, as proposed in this paper, provides such a shipping–mechanism. In addition to the requirements stated above it was designed for making the use of it as simple as possible, thus allowing groups with exotic development environments and little manpower to be able to use MRML.

MRML in the current version provides a complete solution for QBE in CBIRS, which was the initial use of MRML.

MRML is an XML based language for the communication between MMDB server and user interface. In this paper we demonstrate the utility of the framework provided by MRML by giving the example how we use MRML in our *Viper* system.

The paper is organized as follows:
First we describe the general framework with emphasis on extensibility. After this we give the first application of MRML which is the use in our CBIRS systems together with the CIRCUS interface written by one of us (Z.P.).

2 The design of the MRML query shipping framework

Common positions always limit the freedom of the individual. However, in this case the design is easily extensible. In this section we propose a development strategy which will preserve the freedom of the individual research groups, while keeping the standard.

The primary goals of our design are:

1. Extensibility is, as we said above, our primary goal. Main problem here is to provide a framework which permits independent growth of the products of different research groups (followed by periodical code merging).
2. We want to leave the developer the freedom of choice of the implementation language. A standard like this is unlikely to be adopted by the research community, if it works only with a given "mainstream" computing environment.

3. We want the use of the communication protocol to be as independent from third party libraries as possible. A group should be able to provide its own tools within finite time.

Our choice is to use an XML (eXtensible Markup Language) DTD (Document Type Definition – a grammar) for the specification of our communication protocol, together with specifications for the transmission of messages, and for extensions of the protocol.

When making this choice we saw mainly the alternative of using EJB (Enterprise Java Beans), CORBA, and other methods of remote procedure calls. However, we feared strong links with languages (Java/EJB) or large program packages (CORBA). Moreover, the use of an XML application implies a common user log file format for multimedia databases. Please note that the DTD is a way of expressing the capabilities of client and server, *i.e.* they can *negotiate* the kinds of query that are allowed.

The attractiveness of XML is further increased by the existence of free tools in numerous programming languages. XML has been designed explicitly for simplifying parser design. XML has to be parsable by deterministic parsers, thus it is simple to implement one's own XML/MRML parsers.

2.1 The structure of XML and "graceful degradation"

The structure of XML is similar to that of HTML, which stems from their common ancestry, *i.e.* SGML (Standard Generalized Markup Language): an XML document can be seen as a tree of "elements" which themselves contain other elements. The content of each node of the document tree is a list of attribute-value pairs, as well as a sequence of nodes (possibly interleaved with text). This structure is encoded using so called "tags" for the elements. The "opening tag" of an element with type `t` and attribute `anAttribute` being set to `x` would be <`t anAttribute="x"`>. The "closing tag" of an element `t` would be < /`t`>.

This free structure is constrained by a Document Type Definition (DTD) which is a grammar for the tree structure. The details can be found in [**?**].

"Graceful degradation" is the key to MRML's extensibility. It means *"build extensions in a way, that ignoring them causes minimal harm"*. Examples will be given in §6 which contains the MRML–DTD as well as its description. We believe that they demonstrate the feasibility of this approach.

2.2 The main MRML tags

Each message sent can be one of the following:

ihandshake The first text sent when the user connects to the database will be an ihandshake message. ihandshake contains the name of the user. Via the DTD one can detect the abilities of the interface.

shandshake The server will respond to the ihandshake message by giving a shandshake message. This message contains a list of sessions the user has done using the database, as well as a property sheet specification.

Sessions give the user the possibility to perform across–session learning using the system.

MRML property sheets are a description of parameters variables and their dependencies which permit the interface program to build property sheet GUI elements, which will permit the user to configure the database freely. How freely is entirely the decision of the server. Using this mechanism we sidestep the problem of defining a set of common database configuration parameters. We also give the possibility to send interfaces of different complexity to different kinds of users, etc. .

iconfiguration The user is now free to use the configuration options at will. If so, at each step the interface will send an iconfiguration message. We require that this iconfiguration message will have to consist of at least the configuration of the algorithm used. However, as it was said before, more can be configured if necessary.

If the user does not use the configuration facilities, a set of default values is used, until the user does use the configuration facilities.

inewsession,irenamesession For session management the user can choose to open a new session (the old one will be closed then), or to rename the current one the non-existence of an iclosesession tag is not an accident: in a WWW scenario, we cannot to rely on users properly closing their sessions.

iquerystep Fill this at your will. In this paper we propose an XML-based query language for QBE in CBIRS which will be described at §**??**, page **??**.

iresult The result is a list of URLs together with calculated similarities and information on which panel the information is to be displayed.

error Server and interfaces are able to send error messages to each other.

Each of the described messages uses other "helper"–XML–elements. The relationship is further described in the DTD.

3 Connection protocol of MRML client and Server

The interface will connect to a socket of the server, and send a text in MRML. After that it will wait for a response in MRML. After receiving the response the connection will be shut.

...more and some figures to come here...

4 MRML for interfacing and benchmarking in CBIRS

In current CBIRS research there emerge three groups of query techniques which are used in the different systems, most of the time in combination:

- Query By Example (QBE) and browsing queries: the user gives an image, and retrieves similar images using the system. He or she can increase the quality of the result using relevance feedback (for example [8]). As a modification of this scheme: "browsing queries", which could be summarized as QBE without first example (e.g. used by [3, 5, **?**]). In any of these cases the user feedback is limited to stating the relevance of display items.

- Query By Sketch/Query By Segment: these systems require more interaction from the user, who has to draw an example or who has to mark regions of interest in the example he or she has given [1, 4].

- Annotation and Query on Annotation, eventually linking annotation to low–level features: this, essentially, is extending standard database technology to image databases [2,7,?].

As one can see, each of these groups necessitates different activity from the user, as well as a different method of query formulation. Video and audio, again, need other methods of interaction.

The problem of query by example in content based image retrieval systems, is particularly simple in terms of the interaction required. In this paper, we use QBE in CBIRS as an example for the use and usefulness of MRML:

MRML, as it is now, will be able to help the CBIRS community

- Improve interface design: Given that groups could improve a common interface rather than starting their own complete system from scratch, the design, as well as the usability of interfaces of MM databases should improve. The area of MM databases could as well profit from new breakthroughs in HCI, which are to be expected from research areas like e.g. emotional computing.

- MRML will help meta-query research: MRML as it is now allows simple scripts to redirect one query (QBE) to several servers and collect their results. Once other query paradigms are implemented in MRML, research on multi-paradigm and multimodal queries also will profit from the development

- MRML will facilitate the development of real–world applications which use new research results (e.g. an CBIRS–plugin to the GNU Image Manipulation Program [?]). This could be very helpful for evaluating the immediate use of current research.

- MRML will help exchange user data: The log files of interaction between user and content based multimedia retrieval systems are rarely exploited for improving the system [3]. Logging the queries would provide a common format which could easily be exchanged, thus giving the research area the possibility to benefit from collected experience, as suggested in [6]

- It is our opinion the most important point for the CBIRS community is that MRML could be used as a common interface to a common benchmark for CBIRS, which thus could easily be distributed and employed, thus supporting work on a common image collection and benchmark.

  We would like to emphasize that a common benchmark for image database is likely to evolve strongly within short time, and thus needs a flexible framework as a basis.

An immediate advantage for the CBIRS community would be the possible use of the CIRCUS (Lausanne) interface developed by one of the authors (Z.P.).

4.1 Expressing Query By Example in MRML

A query will essentially be a *tree* of URLs together with user given relevance judgments together with information to which panel the results should be sent. Why not a list? A tree permits us to shorten the message: If we want to send a query to several algorithms with slight modifications, we can formulate the common parts at the root of the tree, and modify them in the inner nodes and in the leaves of the tree.

URLs in order to keep things open for true multimedia querying, queries do not send binary data, but an URL as a pointer to such data.