Tiziana Catarci
University of Rome

The Life Cycle of Multimedia Metadata

Harald Kosch, László Böszörményi, Mario Döller, Mulugeta Libsie, Peter Schojer, and Andrea Kofler
University Klagenfurt

During its lifetime, multimedia content undergoes different stages or cycles from production to consumption. Content is created, processed or modified in a postproduction stage, delivered to users, and finally, consumed. Metadata, or descriptive data about the multimedia content, pass through similar stages but with different time lines.[1] Metadata may be produced, modified, and consumed by all actors involved in the content production-consumption chain (see the "Life-Cycle Spaces" sidebar for more information). At each step of the chain, different kinds of metadata may be produced by highly different methods and of substantially different semantic value.

Different metadata let us tie the different multimedia processes in a life cycle together. However, to employ these metadata, they must be appropriately generated. The CODAC Project, led by Harald Kosch, implements different multimedia processes and ties them together in the life cycle. CODAC uses distributed systems to implement multimedia processes. Figure 1 gives the architectural overview of this system.

The project's core component is a Multimedia Database Management System (MMDBMS),[2] which stores content and MPEG-7-based metadata.[3] It communicates with a streaming server for data delivery. The database is realized in the Multimedia Data Cartridge (MDC)—which is an extension of the Oracle Database Management System—to handle multimedia content and MPEG-7 metadata.

Use case scenario

To demonstrate metadata's life cycle, let's consider a use case scenario. A user watches an interesting music video on a colleague's screen and wants to retrieve the same music video. The only information she retained was that the singer in the music video was Avril Lavigne, and she remembers the song's melody. She can't ask her colleague directly, so she wants to access it from a multimedia database.

The query service of our Multimedia Data Cartridge (MDC) offers a solution for finding such a music video. First, the user can enter a query by thematic means, thus specifying that the music video's singer is Avril Lavigne. (See Figure 2a, p. 80, for the query interface.) In response to the first query, the service returns several music videos that pertain to the singer. To narrow the search, the user can hum the melody to the query with a humming service, as Figure 2b shows.

The multimedia database retrieves information on the music video that meets the query request and delivers it to the user. Such information includes the full title, full information on the singer, the production date, and so on, and finally the address of the media server where the user can obtain the video. Thus, the user finds out that the song she has searched for is entitled "Skater Boy" and can now access the music video.

Luckily, the user is registered to the media server storing the video and can request the music video from this server. In addition, the user specifies in the video request her mobile device's terminal capabilities. Unfortunately, the media server has only copies of the music video in a quality that doesn't meet the terminal constraints. The server examines metadata generated in postprocessing of the video to generate a variation of the video with the best possible quality satisfying the constraints and then delivers this variation.

Let's further assume that the request goes over an authorized proxy cache that examines if a cached copy is present. If it's there, but not in the appropriate quality, delivery-related metadata describing the possibility for the video to be adapted to resource constraints is used by the proxy cache to adapt the video accordingly.

1070-986X/05/$20.00 © 2005 IEEE

Published by the IEEE Computer Society

Life-Cycle Spaces

Metadata's life cycle spans content, metadata, and user spaces. In a content space, we can identify four major stages of a content's life span: production/creation, postproduction processing, delivery, and consumption. Once multimedia content is created in the first stage, users can also modify it either by editing it or generating different versions of it.

The metadata space involves two parts:

- Metadata production. This occurs during or after content production. During production, content producers generate globally valid metadata such as creation information (about authors, actors, date of production, and so on). Additional metadata are produced in a postproduction stage either by automatic extraction (typically low-level features, such as color histograms or shape recognition) or through human intervention (typically high-level semantic information, such as scene descriptions or emotional impressions). Such metadata might include transcoding hints and references to the actual media instance or the media profile. In addition, metadata to support content adaptation can be generated.

- Metadata consumption. This occurs at the media delivery and consumption stages of the content's life cycle. For instance, metadata helps end users filter the content according to their preferences. The data helps the system to identify the most appropriate variation of the content that meets the required quality of service. Even proxies and routers might take advantage of metadata to carry out efficient adaptation. They might also reproduce metadata—in effect closing the metadata life cycle—to reflect the changes made to the corresponding content.

The user space includes users that produce, process, and consume content. Three major user classes are involved:

- Content providers and producers are in charge of creating and producing content. They can enrich content by generating and attaching globally valid metadata.

- Processing users are those involved in postproduction processing and include those who index multimedia data in a multimedia database (MMDB) environment or those who prepare the multimedia content for adaptation, such as variation creation.

- End users consume metadata and content. Their role is in browsing, searching, and consuming the multimedia data. They play an important role adapting as well as, for example, specifying their viewing preferences and device characteristics. Their roles can also be in completing the life cycle. For instance, a proxy server is an end user for the media storage server and at the same time a content provider of possibly modified content for the terminal end user.

The metadata life cycle presents a global view of multimedia processing from production through processing to the consumption of multimedia metadata. It shall help other multimedia applications to structure their workflows.

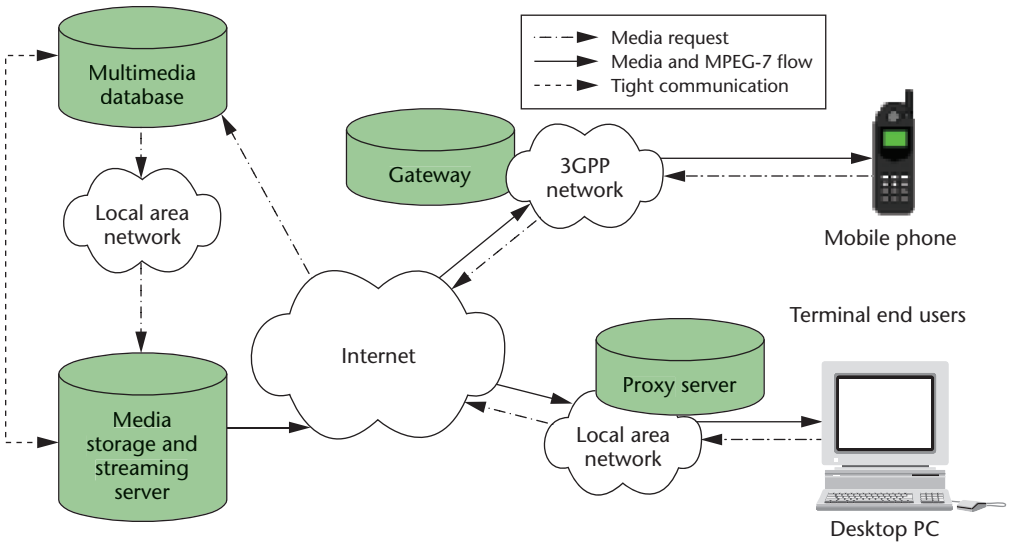
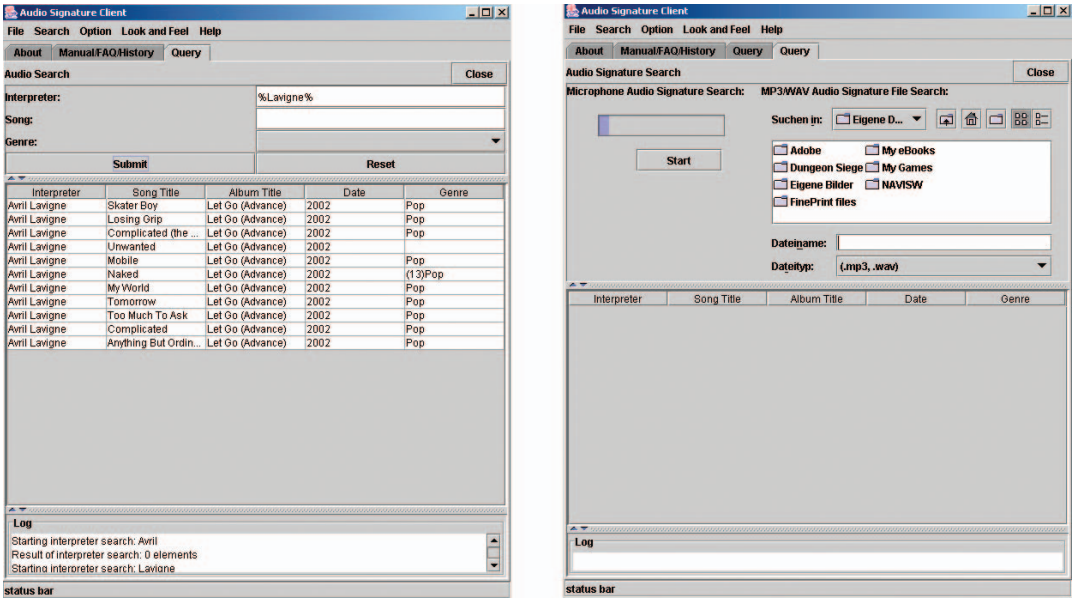


Figure 1. Architectural overview of the CODAC distributed system.

Metadata production

Our multimedia database, the Multimedia Data Cartridge (MDC), defines several application libraries for metadata production. The libraries offer a means for multimedia annotation and content-based retrieval for different media types. For



(a) (b)

Figure 2. Query interface snapshots for seeking a music video. (a) Query by semantics; (b) query by humming.

instance, the use case scenario we just described uses video and audio retrieval methods at the same time. The production process involves a combination of semiautomatic annotation methods through user intervention and full automatic means. For this scenario to work, the metadata resulting from the production process must be stored in the MDC. We can use the produced metadata either for search and retrieval or content adaptation.

Metadata for search and retrieval

The MDC has facilities for annotating videos (and/or video segments), audio, and images.

Video annotation includes attributing semantics to videos and video segments. Semantic information includes listing the creator, genre, actors, and so on, as well as information relying on classification schemes. For instance, we can attribute a term from the Internet Content Rating Association (ICRA) classification schema to a video containing strong language or hate speech. Videos are also annotated using the so-called low-level features, such as accumulated color histograms based on the MPEG-7 ScalableColor descriptor.

Video retrieval is based on semantic criteria—for example, all music videos of the singer Avril Lavigne—and on searching for similarities. In this instance, the user could include an image representing a key frame of a video to retrieve videos containing similar images.

An audio annotation and recognition framework is available in the MDC that lets users identify a music composition with the help of a 10-second audio signal, which can for instance be obtained by humming the melody. In the annotation phase, a fingerprint is extracted from each recorded audio signal in the form of an MPEG-7 AudioSignature and inserted into the MDC. When the user places a query, the same fingerprint algorithm is applied to the sample, and this fingerprint is compared to the database entries using a Nearest-Neighbor search (NN-search) relying on the built-in Similarity-Search trees (SS-trees) of the MDC indexing framework. (The SS-tree was introduced in 1996. It was the first index tree to employ minimum bounding spheres instead of rectangles to partition the feature spaces.)

MDC's image annotation and retrieval framework uses a blob-based image segmentation taken from Blobworld.[4] A blob is a more or less homogenous area in an image. Blobs are discovered by grouping pixels with similar color, texture, and shape. These features are mapped by the annotation framework to the MPEG-7 descriptors ScalableColor, HomogeneousTexture, and RegionShape.

Video variation and metadata for content adaptation

The annotation framework automatically produces metadata for video variation as a means for adaptation. In video variation, the hope is to generate new videos (variation videos) from the

source video, with a reduced data size and quality by applying a variation or reduction method. The supported variations include

- *temporal variation*, which reduces the visual data's frame rate through frame dropping;
- *spatial variation*, which encodes fewer pixels (pixel subsampling) and thereby reduces the level of detail in the frame; and
- *color variation*, which reduces each pixel's color depth.

The CODAC project has defined and implemented two new methods of video variations:[5]

- *Object-based variation* extracts the foreground and background objects in a source video and re-encodes the video with two visual objects. This facilitates object-based adaptation, which otherwise would be impossible for a video encoded without objects. For instance, in our use case scenario, object-based variation would be useful for segments with a dynamic singer foreground and static image background. This lets an adaptation engine discard, in case of resource bottlenecks, the static background image.
- *Segment-based variation* lets us apply variation methods selectively for video segments based on the physical characteristics of motion, texture, and color, thereby minimizing the quality loss and/or maximizing the reduction in data size. We segment the source video into its component shots and automatically select and apply a set of appropriate methods by analyzing the degree of physical characteristics within the segment.

We accomplish variation creation by implementing a server module called the Variation Factory, which generates the variations.[5] All the variation methods—including the newly defined object and segment-based variation methods—are implemented in the Variation Factory. A source video can be subjected to a single method or a combination of them as deemed necessary—for instance, temporal variation followed by spatial variation, and so on. At each step, the Variation Factory produces a variation video, thereby creating a tree of variations. The Variation Factory is an application programming interface (API) including user interfaces to guide the metadata generation process. The user can control the adaptation results and rearrange parameters—for instance, if the perceived quality is too low.

The Variation Factory's architecture is developed in Java under Java Media Framework (JMF). The API supports the integration of audio and video playback into Java applications and applets.[6] The input is video, and the outputs are one or more adapted variation videos and an MPEG-7 metadata document that describes the source and the variation videos using descriptors of the VariationSet description scheme. The MPEG-7 Document Processor produces metadata. It uses Java API for XML processing (JAXP), which is generally used to parse and transform XML documents.

First, a Document Object Model tree is constructed. Then, the DOM tree is parsed and an XML text file is produced. The descriptors include information on the variations' fidelity, data size, and priority. We store the created variation videos in the media server along with the source video and the MPEG-7 document in the metadatabase. During delivery, the MPEG-7 metadata document is streamed together with the requested video.

Multimedia Data Cartridge

The MPEG-7 MDC is at the center of the metadata life cycle because it must manage all the metadata produced and deliver it to the consuming elements. The MDC is an extension of an Oracle database that can store, index, and query multimedia metadata based on the MPEG-7 standard. It currently consists of three main parts (see Figure 3, next page). The core system consists of a multimedia database schema based on MPEG-7, the Multimedia Indexing Framework (MIF) supporting query optimization, and a set of internal and external libraries for incoming requests and queries.

The MDC has been implemented by a small group of database programmers with experience working with the Oracle DBMS kernel. They kept the extensions to the Oracle database as modular as possible. Part of these modules, mainly the indexing framework, is in preparation of a SourceForge project (http://sourceforge.net/index.php), which CODAC plans to make available to the public in Spring 2005.

MPEG-7-based database schema

The multimedia schema relies on the MPEG-7 standard to provide a complete multimedia metadata schema for low-level descriptions (such as

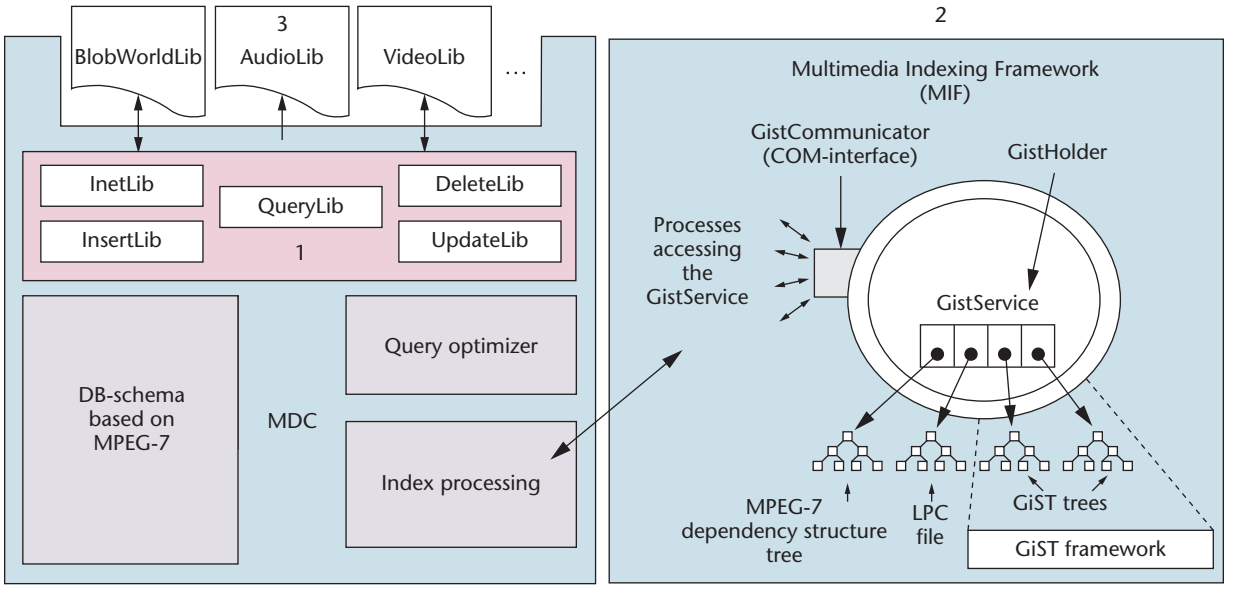


Figure 3. MPEG-7 Multimedia Data Cartridge (MDC).

color, texture, and shape for images) and high-level descriptions (such as structural and semantic descriptions) for all media types. We have mapped the MPEG-7 descriptors, formulated as XML-types, to object-relational tables to enable fine-grained querying on these data. (A detailed explanation of the mapping is available elsewhere.[7])

Library support

A set of internal and external libraries are used for incoming requests and queries. The set of internal libraries is used as access points to the core system and consists of InitLib, used for creating new instances of the MDC data type; InsertLib, which provides insert functionality of MPEG-7 documents; DeleteLib, for deleting MPEG-7 documents; UpdateLib, for updating parts of stored MPEG-7 documents; and QueryLib, for query services. Furthermore, external libraries are used to offer application-specific services.

The services we described in the use case scenario are VideoLib, for obtaining videos with semantic search criteria, and AudioLib, for querying with the humming functionality. Both external libraries (VideoLib and AudioLib) rely on the search functionality of the QueryLib, which is basically a translation of search criteria to complex SQL and XPATH statements on the schema tables.

Multimedia Indexing Framework

We divided MIF into two modules. The first is GistService, which is located in an external address space and manages all available access methods. The current version supports Generalized Search Trees (GiST[8]) and further access methods not relying on balanced trees (such as LPC files) to support NN-search in high-dimensional vector spaces.

The second is the Multimedia Index Type, which is located in the Oracle address space. It consists of several index types, their corresponding operators (for NN-search, range search, and so on), and the appropriate implementation. AudioLib's query-by-humming service relies on the SS-tree index and the related NN-search operator.

Metadata consumption at the proxy

Multimedia adaptation is a key technology for assuring the quality of end-to-end delivery in the network. Adaptation can dynamically react on different kinds of presentation devices and on unpredictable resource availabilities. Proxy servers situated in the middle of the delivery chain constitute an ideal place to control multimedia delivery and adapt the stream if resource availability changes. In this context, we built the Quality-Based Intelligent Proxy (QBIX), which is a terminal capabilities- and metadata-aware mapped (RTSP; http://www.rtsp.org) proxy server that supports real-time adaptation in the compressed (temporal adaptation only) and uncompressed domains. Adaptation improves the hit rate in the proxy cache and lets the proxy act as a media gateway supporting transcoding in real time based on metadata.[9] One part of the metadata is sent by the server and describes the video variations (MPEG-7 VariationSet descriptions), and the other part is provided by the terminal end user (in the form of Composite Capabilities/Preferences Profiles, which we describe later in this

article) sending terminal capabilities to the proxy.

The proxy extracts the terminal capability information from the video request and checks if it has this music video in its cache in a quality that matches the terminal capabilities. If the video is already available, but its properties aren't in accordance with the terminal's characteristics—for example, the video's bit rate is too high to be consumed at the end user's site—the MPEG-7 descriptions that accompany the video are examined by the proxy. They contain hints on which variations of the original video should be selected or created (if one doesn't exist) to meet the delivery and presentation constraints. These hints contain the video's expected size, bit rate, and quality. The proxy then chooses from among these variations the one with the highest quality that meets the restrictions of the end user's terminal. The proxy can either load the required variation from the server or generate it with a sequence of appropriate transcoding procedures.[9]

We implemented metadata consumption at the proxy and the terminal using our Video ToolKit (ViTooKi). ViTooKi is principally a set of libraries that support adaptive standard-compliant video streaming, transcoding, and proxy caching. It supports MPEG-1/2/4/7/21 and MP3/AAC file types, stored in various containers like mp4 and avi using standard protocols with retransmission and the server, proxy, and player included. (ViTooKi is a SourceForge project and is available at http://sourceforge.net/projects/vitooki/.)

Metadata consumption with terminal devices

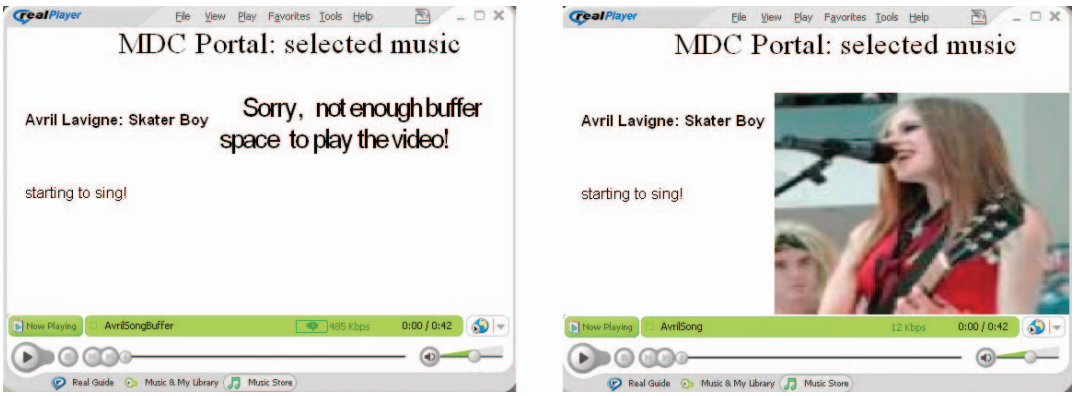
Using multiple terminal devices (such as PCs, PDAs, and mobile phones) makes it necessary for us to think about adapting presentation interfaces according to the terminal capabilities and user preferences. To serve multiple types of client devices, the content and metadata are consumed through an adaptive presentation interface.

In this context, adaptation means dynamically adjusting the interface to the end user's terminal capabilities (such as different hardware characteristics—including the buffer for video players—and software, such as browsers and corresponding multimedia plug-ins) as well as the user's preferences—for example, the user might prefer to receive only images and no video.

We describe terminal capabilities with a composite capabilities and preferences profile (CC/PP), which is a standardized framework developed by the World Wide Web Consortium (W3C), based on the resource description format.[10] Using RDF gives us a standard way of exchanging information about the end user's capabilities. We extended a CC/PP standard implementation, DICE,[11] and integrated it into the end user or proxy and server environments to adapt the query and presentation interfaces to the respective usage environment descriptions.

To demonstrate how the terminal end user can use metadata, let's assume that the video request (in our use case scenario) is successfully accepted and the music video starts to stream from the server or the proxy. Unfortunately, the presentation interface detects a buffer constraint with respect to the video player that would degrade the video quality if the video plays in full resolution. Based on the metadata accompanying the media, the presentation interface can now choose the appropriate resolution that meets the buffer constraints.

Obviously, video display might not be possible in all cases, depending on resource availabilities, and the consumption might be restricted to the audio track only. Figure 4 shows two possi-



(a) (b)

Figure 4. Adaptive presentation interface for (a) audio only and (b) video and audio.

bilities of media consumption at the end user realized through our adaptive presentation interface. The presentation can include video and audio (Figure 4b). If the user doesn't have enough buffer for video display, we can play only the audio track (Figure 4a).

Further research

Researchers have done little work so far on security issues within the database or the legal effects of using multimedia metadata. This is in part because of the missing license credits in the MPEG-7 standard. Integrating the Intellectual Property Management and Protection (IPMP) tools of the emerging MPEG-21 standard can give us the necessary metadata to manage legal effects. Furthermore, we could use the MPEG-21 descriptions to realize a multimedia access and control system within our multimedia database. We're following the progress of the MPEG-21 standard and will consider these issues in the near future. **MM**

Acknowledgments

This work was carried out in cooperation with Jörg Heuer from Siemens AG, Holger Crysandt from RWTH Achen, and funded in part by the Austrian Science Fund (Fonds zur Förderung der wissenschaftlichen Forschung) under project number P14789.

References

1. F. Nack, "All Content Counts: The Future in Digital Media Computing Is Meta," *IEEE MultiMedia*, vol. 7, no. 3, July-Sept. 2000, pp. 10-13.
2. H. Kosch, *Distributed Multimedia Database Technologies Supported by MPEG-7 and MPEG-21*, CRC Press, 2003.
3. J.R. Smith, "MPEG-7 Multimedia Content Description Standard," *Multimedia Information Retrieval and Management*, Part I, D. Feng, W.C. Siu, and H. Zhang, eds., Springer Verlag, May 2003.
4. C. Carson et al., "Blobworld: A System for Region-Based Image Indexing and Retrieval," *Proc. 3rd Int'l Conf. Visual Information Systems*, LNCS 1614, Springer Verlag, 1999, pp. 509-517.
5. M. Libsie and H. Kosch, "Video Adaptation Using the Variation Factory," *Proc. 7th IEEE Int'l Workshop on Multimedia Signal Processing* (MMSP 2004), IEEE CS Press, 2004, pp. 120-123.
6. R. Gordon and S. Talley, *Essential JMF: Java Media Framework*, Prentice Hall, 1999.
7. M. Döller and H. Kosch, "An MPEG-7 Multimedia Data Cartridge," *Proc. SPIE Conf. Multimedia Computing and Networking 2003* (MMCN 03), 2003, pp. 126-137.
8. J.M. Hellerstein, J.F. Naughton, and A. Pfeffer, "Generalized Search Trees for Database Systems," *Proc. 21st Int'l Conf. Very Large Databases*, Morgan Kaufmann, 1995, pp. 562-573.
9. P. Schojer et al., "Architecture of a Quality Based Intelligent Proxy (QBIX) for MPEG-4 Videos," *Proc. ACM World Wide Web 2003 Conf.*, ACM Press, 2003, pp. 394-402.
10. *Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0*, W3C Recommendation, 15 Jan. 2004; http://www.w3.org/Mobile/CCPP/.
11. M. Butler et al., "Device Independence and the Web," *Internet Computing*, vol. 6, no. 5, Sept./Oct. 2002, pp. 81-66; http://dice.ccpp.info/.

Readers may contact the authors at {harald.kosch, laszlo, mdoeller, mulugeta, pschojer, andrea}@itec.uni-klu.ac.at.

Contact MultiMedia at Work department editor Tiziana Catarci at catarci@dis.uniroma1.it.