COMPOUND MEDIA STREAMING IN TIME

B. Feustel, T.C. Schmidt[1]
{feustel,schmidt}@fhtw-berlin.de

Computer Centre, Fachhochschule für Technik und Wirtschaft Berlin
Treskowallee 8, 10318 Berlin, Germany.

D. Marpe
marpe@hhi.de

Heinrich Hertz Institut für Nachrichtentechnik Berlin
Einsteinufer 37, 10587 Berlin, Germany

M. Palkow, H.L.Cycon
{mpalkow,hcycon}@fhtw-berlin.de

FB Ingenieurwissenschaften I, Fachhochschule für Technik und Wirtschaft Berlin
Allee der Kosmonauten 20-22, 10315 Berlin, Germany

ABSTRACT

The widespread availability of networked multimedia potentials embedded in an infrastructure of qualitative superior kind gives rise to new approaches in the areas of teleteaching and internet presentation: The distribution of professionally styled multimedia streams has fallen in the realm of possibility. This paper presents a prototype - both model and runtime environment - of a time directed media system treating any kind of presentational contribution as reusable media object components. The plug-in free runtime system is based on a database and allows for a flexible support of static media types as well as for easy extensions by streaming media servers.
The technique of pluggable subservers is used to include a real-time video codec, based on a fast low complexity wavelet transformation and an highly efficient lossless precoding scheme using ´partitioning, aggregation, and conditional coding (PACC). The JAVA implementation enables real–time video streaming in CIF format on IP connections with low bandwidth (≈ 200kb/s).

Keywords: Synchronized Media, Multimedia Modeling, Video Streaming.

1 INTRODUCTION

Today's standards of internet-connected computers provide associatively chained texts, images, sounds, movies or other information material and thereby confront students as well as teachers with a new paradigm of knowledge transfer: Networked multimedia accessories not only transport a formerly unknown multitude of presentation methods to the lecture hall or – in the framework of teleteaching – to students homes. They also present an unfiltered totality of present knowledge which in terms of quantity and rapidity of change no single teacher can compete with.

Individual aspects of classroom communication however remain bound to traditional lecture forms. All direct dialogs and related interchanges must be mentioned here, firstly. Of at least second importance is the notion of time and speed any teacher imposes on his students by determining order and speed the material is rolled out and by finally fixing the dates for testing and documenting success. It is this time control process which finally determines performance.

Many attempts are made to employ World Wide Web techniques to at least partially substitute traditional lectures, the vision of which being to liberate students from the restrictions of temporal

[1] This work was supported in part by the EFRE program of the European Commission.

and spatial presence and thereby offer knowledge to a wider public. But in contrast to a personal classroom attendance any http-based training module is driven by enduser's mouseclicks and thus delegates ordering and timing completely to the learning recipients. It hence burdens the learner with the responsibility not only for overall understanding but also for undertaking each individual progress step in time.

The important notion of time in teaching is one major reason for drawing a lot of attention in recent research works to World Wide Web techniques which distribute multimedia documents with temporal and spatial relations. The growing demand for synchronized handling of time-based media such as video and audio serves as a second motive for introducing temporal aspects to the Web. Finally, streaming data sources invent a new level of scalability by accounting for transport timing and therefore rapidly gain quantitative importance throughout the internet.

In the present paper we present the project Media Objects in Time (MobIT) which develops a model for time-directed presentation and processing of universal media objects as well as a prototype for a MobIT runtime environment.

This paper is organized as follows. In section 2 we introduce the basic ideas and functionalities of our work and exemplarily compare to related works. Section 3 presents the underlying Compound Flow Model (CFM). Architecture and implementation properties of the MobIT runtime environment will be discussed in section 4. A brief introduction to the built-in JAVA video-player based on high performance optimized wavelet codec is given in section 5. Finally, section 6 is dedicated to conclusions and an outlook on the ongoing work.

2 MOBIT AND RELATED WORKS

2.1 PRESENTING A MEDIA STREAM

The teaching and presentation system introduced here centers about the idea of media objects synchronizable in time which may be linked to form fairly complex presentations. But at the same time any object remains self consistent and of independent use. Roughly speaking our basic concept consists of defining media object instances and lining them up in time as is shown in figure 1. MobIt intends to provide an accurate scheme for temporal and spatial placement of presentation objects, where authors neither have to take care of interobject synchronization dependencies nor adaptation to possibly inaccurate network performance, the latter being subject to implementation of latency hiding techniques.

Presenting itself on a timeline any presentation becomes a time-based data object, even if composed only from timeless media such as texts or images. Any presentation component will carry an instance of initial appearance and a moment (possibly at infinitum) for fading away from the client's screen. Within this framework any streaming media such as video or audio may be included and synchronized to the scene and the overall data stream.

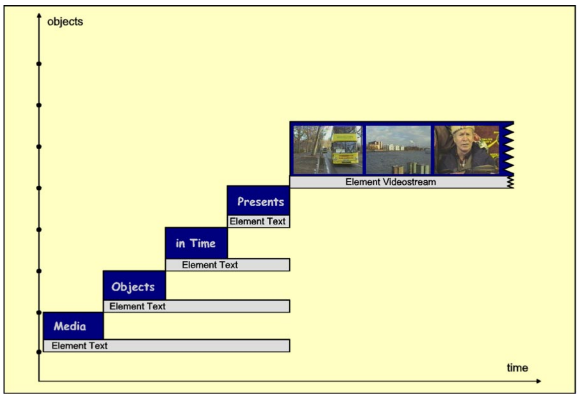


Figure 1: Object Instances aligned to Time

Aiming at the combined utilization in lecture rooms as well as teleteaching our model focuses on a clear, straight forward concept of reusable compound media components. Any of these will be accompanied by screenplay scripts arranging their behavior in space and time. Thus in place of the page oriented WWW concept or the typically event driven nature of CBT products MobIT runs as a flow oriented presentation model showing for example a crash-test video combined with charts of relevant statistics and vocally explained CAD car models in subsequence.

The implementation of the MobIT runtime prototype concentrates on a pure JAVA solution. Any recipient may load the corresponding JAVA applet into the browser which then will connect to the MObIT server, request for meta-data, open media data streams and eventually start timer and the display process. Spatial and temporal placement is done with great accuracy. Additional plug-in software is not required.

2.2 RELATED WORKS

As mentioned earlier several interesting research activities are presently enforcing the notion of time to the Web, the most prominent being the W3C recommendation Synchronized Multimedia Integration Language (SMIL) [Hos98]. As a declarative language SMIL allows for synchronization of media objects in an somewhat simplistic, HTML-style fashion. Synchronization is done in object pairs, either sequential or in parallel. The appearance of any object may be bound to a duration parameter. SMIL extends the meaning of hyperlink to connecting temporal and spatial subregions.

The runtime behavior of any SMIL interpreter thereby is more or less left open, which probably is the most important drawback of the model. Combined with the absence of a stringent handling of timelines temporal inconsistencies in more complex documents can be foreseen. Besides few reference implementations of SMIL players there is an attempt to include synchronization features into the Web browser named HTML+TIME [SYS98]. This proposal addresses temporal extensions to HTML and incorporates basic elements of SMIL.

Both ideas however suffer from strong limitations due to the simplistic ansatz of HTML omitting any structuring for media object use. Rutledge et al [RHO99] consequently report about severe difficulties in authoring SMIL presentations mainly due to the lack of reusability for object compositions as well as SMIL's inability to deal with complex object relations. In most recent works, the Boston specification of SMIL [Aya00], the World Wide Web Consortium heads towards a realization of SMIL as a module within the framework of the XHTML language. Most of this work is presently ongoing and far too incomplete from permitting implementations.

As a completely other example more similar to our work we like to mention the Nested Context Model (NCM) of Soares et. al. [SCR95]. With the aim of grounding a strong structure for flexible deployment of hypermedia documents the NCM provides a composite meta-structuring for different media types, called nodes, up to an arbitrary level of complexity. Those nodes may contain a reference list of denoted nodes giving rise to an arbitrary graph structure of the composed document. The model, which has been implemented in a system called HyperProp, treats hypermedia documents essentially as passive data structures. Synchronizations define through events which may occur as the result of object presentation or user interaction.

Since embedding of media objects within the NCM results in a passive mesh without further presentational meaning, an additional structure of activation, events and contexts (called perspectives), has to be superimposed. This characteristic on one hand leaves some liberty to the author (the same object structure may encounter different behavior in different contexts), on the other hand it adds an additional level of complexity to the modeled hypermedia system and denotes the major difference to our work.

3 THE COMPOUND FLOW MODEL

3.1 CONSTITUENTS

In designing the Compound Flow Model (CFM) much care was taken to define a simple structure of straight forward logic which intuitively appeals to document authors and is suited for light weight implementation especially on the client side. The CFM centers around a uniform hull entity named Media Object (Mob) serving as a universal container for embedding either subsequent Mobs or media data. By referencing one another Mobs allow for hierarchical compositions of unlimited depth, where the leaves of the resulting tree structure are fed with the actual media information by means of a distinct data object called Element (s. figure 2).

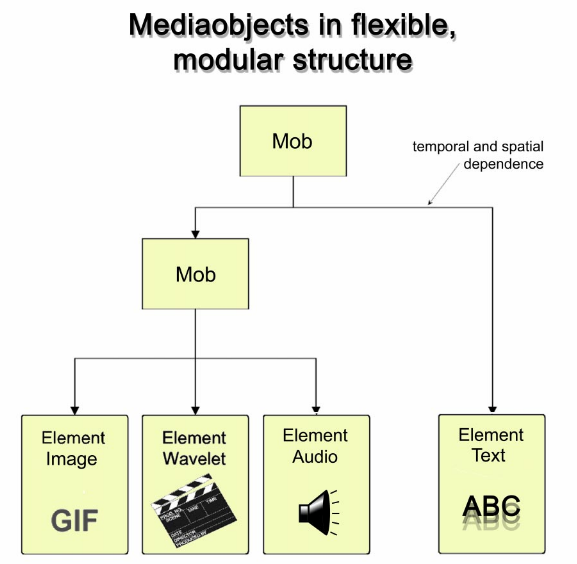

Mediaobjects in flexible, modular structure
Mob
temporal and spatial dependence
Mob
Element Image
GIF
Element Wavelet
Element Audio
Element Text
ABC

Figure 2: Media Object Hierarchy

Whereas Mobs as uniform composition objects exhibit neutral behavior with respect to embedded data, Elements are of specific, atomic type and enclose all properties related to the media data they control. Type informations of elements are exchanged via the MIME-type standards according to RFC 1341 and are used by the element class to appropriately receive media data, instantiate presentation methods and finally display and destroy them on scene. This data processing of elements applies for static media such as images or text, as well as for subserved data like video or audio streams.

Vital to the framework of CFM is an environment for generating and controlling the flow. As media objects for a given presentation may be widely branched, each one of them equipped with a complex structural inheritance and its own synchronization demands, a flow control module needs to resolve all structural data dependencies, to linearize resulting bulk information, to form an ordered flow and at last addict objects to the externally provided primary timer.

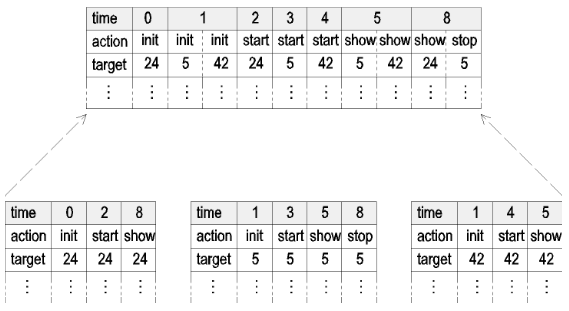
3.2 MEDIA OBJECTS

Media objects may be seen as the central constituents to comprise the CFM data structure. As the basic design idea a Mobs consists of both, the subordinate object reference list and a screenplay script for the references describing all parameters responsible for their behavior in time and space. Those scripts we denote as Playlists. Playlists describe the states attained by the corresponding Mobs in total.

Tightly bound to the concept of combined reference to objects and their states is the notion of generalized reusability for any component involved. Roughly speaking an object exhibits generalized reusability, iff it is self consistent and parameterizable in state space. The fundamental parameters of the state space up until now are the spatial size and the duration in time. Some additional features such as background color or font type change have been implemented. The definition of sustainable parameters however is still somewhat vague. The actual realization of parameter outcome needs to be done at the level of individual Elements, e.g. scaling, clipping, ... Nevertheless any Mob is equipped with the ability to receive and process available parameters.

Self consistency of Mob hierarchies may be viewed either from a structural or a state space sight: Structural self consistency is present as long as the object reference hierarchy omits recursions, i.e. a Mob must not contain any referral pointing to itself or a subordinate object. Self consistency in state space is ensured as long as any Mob stays within the temporal and spatial bounds of its superMob.

To strengthen this twofold notion our Mob design includes the congruence of consistency in structure and state space: As integral part of the CF model structural arrangements of Mobs carry the meaning of inclusion relations in space and time. As one result scale parameters are processed along the object hierarchy to hand down actual Element sizes. The exact appearance of a Media Object thus not only depends on its individual referenced parameters but also on its relative position within the hierarchy. As one benefit it should be noted that an author can place and parameterize any compound object in any context and the system will ensure spatial and temporal integrity.



time	0	1		2	3	4	5		8	
action	init	init	init	start	start	start	show	show	show	stop
target	24	5	42	24	5	42	5	42	24	5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

time	0	2	8
action	init	start	show
target	24	24	24
⋮	⋮	⋮	⋮

time	1	3	5	8
action	init	start	show	stop
target	5	5	5	5
⋮	⋮	⋮	⋮	⋮

time	1	4	5
action	init	start	show
target	42	42	42
⋮	⋮	⋮	⋮

Figure 3: Playlists under Linearization - aligning the Instruction Hierarchy to Time

3.3 THE COMPOUND FLOW

As it is clear from the above section every Mob provides its own relative co-ordinate system in space and time, each of them needing transformation into display co-ordinates during runtime. Reference identifiers of objects furthermore rely on a scheme of global database numbering and need to be transferred into local presentation co-ordinates as exploited in [KLS99]. Most importantly Mobs are structured in a possibly complex, highly nonlinear fashion which may suit for spatial representation but is inappropriate for display conformal with linear time.

Even though components of the Model are of active, self consistent nature an additional flow generator needs to be present. Generating a flow in our context has to fulfill the job of resolving all open object dependencies, collecting the data and en passant performing co-ordinate transforms and at the core linearize data with respect to time. As a result of such linear alignment all playlists are merged to form a complete script for the screenplay the whole presentation consists of. Additionally may be observed that the flow generator as described is – if properly implemented - well suitable for transmitting presentations data collection as a sequential stream over the network.

4 ARCHITECTURE AND IMPLEMENTATION

4.1 CONCEPTS

The technical core of the MobIT runtime environment is formed by an open multimedia architecture designed according to a 3-tiered principle as is shown in figure 4. A JAVA applet on the client side takes care of the presentation, processes the data object delivery and includes the flow generator by means of a sequentializing data interface and a time state machine. Media specific aspects are found encapsulated within element classes which control format processing and displaying on their own. The element class requests for implementations of methods according to the system's action types and is thereby open for easy extensions. The client also provides the general option to receive and instantiate additional subserver classes and eventually run them in a separate thread.

The MobIT server is likewise written in JAVA and primarily responsible for the session and transaction management with respect to client applets. Client and server communicate via a simple data exchange protocol which is spoken asynchronously through buffering cache layers which allow for latency hiding. To retrieve Media Data from a repository the server relies on an abstract data interface, presently

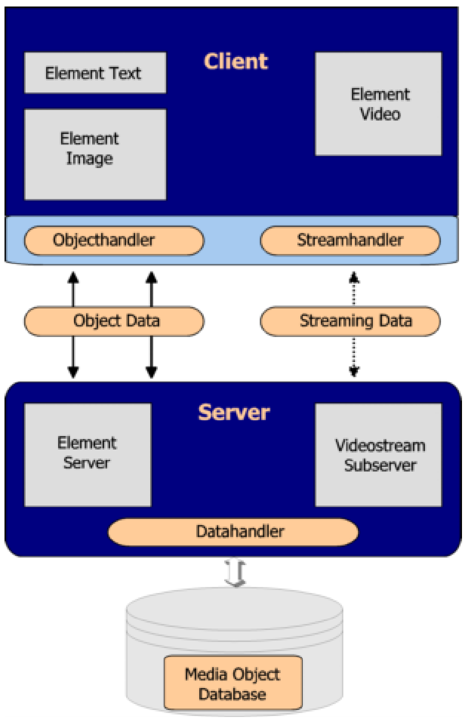


Figure 4: System Architecture

fed by the two data sources implemented: To store data just in a flat file system media object structuring may be XML-coded for direct server processing. More elaborate data access is donated from an intelligent, multifunctional multimedia database system which will be described in a forthcoming paper [KFS00]. Both, meta data and binary elements are readily handed over as JAVA object instances on MobIT application server's demand.

4.2 SUBSERVING

As an important feature of the platform introduced here may be seen its ability to deal with pluggable subservers. Subserving not only opens up the field for nonstandard media and streams, but also allows for incorporation of new, complex functionalities such as online data processing without fattening the thin applet client. For an overal stream oriented system it appears quite natural to include served media for streaming and such. MobIT provides a flexible and simple interface for this purpose. Currently subservers are used to incorporate the high performance optimized JAVA Wavelet video player (s. b.), a direct text sender which permits messaging to ongoing presentations and a LaTeX server which processes LaTeX formulas for display.

The interface to include any type of subserver has been purposely designed in minimal fashion: Any subserver in perspective must implement the methods `getPortCount` to allow for inquiry on requested number of ports, `setPorts` to permit port assignment, `setData` to receive data handles and the initialization. Additional information classes etc. are kept optional. The interface at the corresponding client site appeals as even simpler: `setServerInfo` and `getServerInfo` are the methods needed here. Within this open framework it should be easy to bring additional data servers to the system as for instance to include realtime visualization or live streams or …

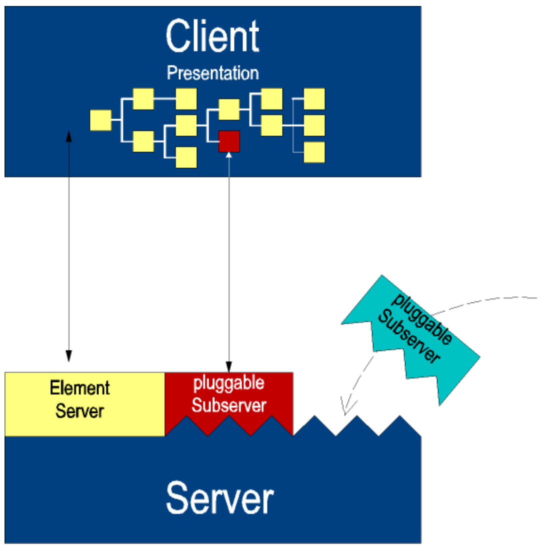


Figure 5: Subserver Extensions

5 WAVELET-BASED REAL-TIME VIDEO CODEC

5.1 TRANSFORMATION AND QUANTIZER

Transformation coding usually consists of three modules: a lossless transformation which decorrelates the signal, a quantizer and a lossless entropy coder which compactifies the data produced by the quantizer. The transformation we use is of wavelet type transforming the image as a whole. Thus no blocking artefacts occur. Filtering is done in a low complexity implementation with a 5/3 tab convolution – subsampling on three levels. The quantizer we chose as a simple uniform scalar with enlarged dead zone. The third module is a highly efficient scheme consisting of a precoder frame work (PACC) and a set of arithmetic coders. To reduce the

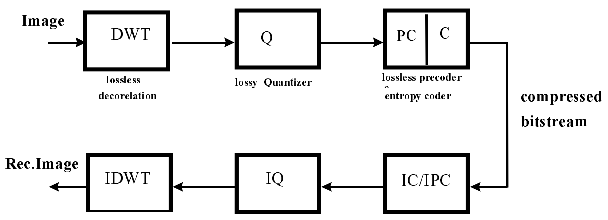


Figure 6: Transformcoding

temporal redundancies in a video sequence we use DPCM coding, i.e. the difference from a frame to the next will be coded only. Complex motion estimation and compensation are omitted for to achieve a fast running video codec implemented in JAVA.

5.2 THE PACC PRE-CODING FRAMEWORK

For encoding the quantized wavelet coefficients, we follow the conceptual ideas presented in [MaCy97]. We give a brief review of the involved techniques. For more details, the readers are referred to [MaCy97], [MaCy99].

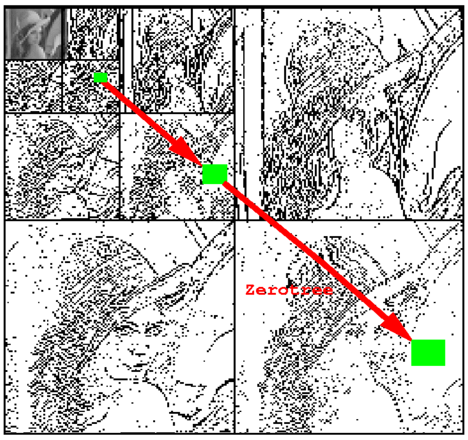


Figure 7: PACC Decomposition including 'Zerotree' coefficient aggregation

An initial ‘partitioning’ stage divides each frame of quantized coefficients into three sub-sources: a significance map, indicating the position of significant coefficients, a magnitude map, holding the absolute values of significant coefficients, and a sign map with the phase information of the wavelet coefficients. Note, that all three sub-sources inherit the subband structure from the quantized wavelet decomposition as shown in figure 7, so that there is another partition of each subsource according to the given subband structure.

As is indicated in figure 7 in a second stage, the pre-coder performs an ‘aggregation’ of insignificant coefficients using the ‘zerotree’ related data structure [LeKn92], [MaCy97] connecting insignificant wavelet coefficients which share the same spatial location across different scales of the octave-band decomposition. Note however, that in contrast to other zerotree-based coding methods, only zerotree roots localized in the lowest frequency bands are considered which guarantee a sufficient coding benefit.

The final ‘conditioning’ stage of the pre-coder supplies the elements of each source with a ‘context’, i.e. an appropriate model for the actual coding process in the arithmetic coder. Heuristically designed prototype templates are used for conditioning of elements of the significance map. Typically, they consist of two parts, where the first part is a causal neighborhood of the actual significance state c depending on scale and orientation of a given band. Except for the lowest frequency bands, the second part of the template utilizes additional information of the next upper level, i.e. significance information on the next coarser scale, which is given by the neighbours of the parent of the actual significance state c, thus allowing a “prediction” of the non-causal neighbourhood of c. The processing of subbands is performed in the order from lowest to highest frequency bands, and the partitioned data of each band is processed such that the significance information is coded (and decoded) first.

This permits the construction of special conditioning states for the coding of magnitudes using the local significance information. Thus, the actual conditioning of magnitudes is performed by classifying significant coefficients according to the local16 variance estimated by the significance of their 8-neighborhood. For conditional coding of sign information a higher-order Markov model is used whose states are built of two preceding signs of a given sign event with respect to the orientation of the related band [MaCy97], [MaCy99].

5.3 RESULTS

In native implementations the video codec encodes and decodes 25 CIF frames (352 x 288 pixels) simultaneously on a 500 MHz Pentium machine. Alternatively 5 frames in PAL (720 x 576) resolution may be processed, where framerate is expected to increase with forthcoming algorithmic improvements. The image quality is better or comparable with MPEG 4 / H.263 Coders. On moderate motion complexity this frame rate produces a bit rate of ca 200 kb/s. If we introduce an overlapped block matching motion compensation scheme, the image quality is better than 1dB PSNR compared to MPEG4 / H.263 codecs (see fig .8).

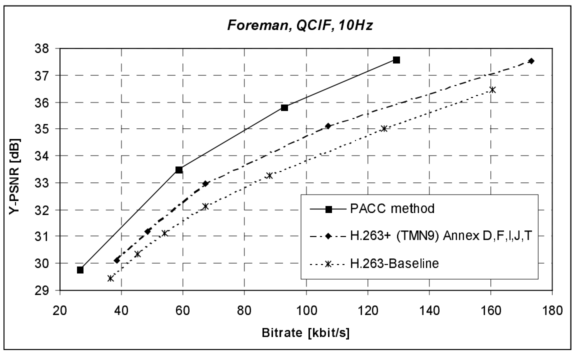


Figure 8: PACC Video vs H.263

As pure JAVA software without the block matching the codec still performs in real-time on moderate frame-sizes and frame-rates on the same type of standard PC: The JAVA codec running in an applet still decodes or encodes 5 CIF frames per second in real-time or - more appropriately - QCIF format with 25 frames.

Newer optimization techniques like HotSpot however and further optimization efforts within the codec most likely will increase JAVA code performance substantially so that even conferencing tools in decent quality may be included within the browser.

6 CONCLUSIONS AND OUTLOOK

The lack of temporal aspects within today's internet presentations and training attempts has been recognized as a serious deficit. Several interesting activities to overcome it can be presently observed. Relating to this debated context our paper presented the time oriented teaching and presentation system MobIT which has been designed to handle complex, reusable media objects in time. Grounded on the CFM stream oriented model we tried to introduce a clear, straight forward concept which copes with temporal and spatial object hierarchies.

Much work however has to be done in this ongoing project. Interactions have not been defined in CFM yet. As simple smil-type hyperlinks in our flow oriented model could only support hopping between - possibly nested - parallel timelines and as we do not intend to produce some sort of interaction programming language, our current activities concentrate on modeling an interaction paradigm. Accounting for the CFM potential to operate on self consistent media objects we are aiming at a small 'alphabet' of operations which enables authors to open up an unlimited number of navigational paths to the receptor with only a limited number of interactions defined. In addition an authoring tool must be brought to the web.

ACKNOWLEDGMENTS

We would like to thank Andreas Kárpáti who not only implemented the database system but always shared an open ear or three with us in many fruitful discussions.

REFERENCES

[Feu00] Feustel, B.: *Ein multimediales, zeitbasiertes Lehr- und Publikationssystem*, Diplomarbeit, TFH Berlin, 2000.

[FeuSch00] Feustel, B., Schmidt, TC.: *Time Directed Internet Teleteaching,* Proc. of ICL2000, Kassel 2000.

[KLS99] Kárpáti, A., Löser, A., Schmidt, TC.: *Interactive Picture Network*, Proc. of IEE Workshop on Distributed Imaging, 99/109, London 1999, pp. 18/1-18/8.

[Hos98] Hoschka, P. (ed.): *Synchronized Multimedia Integration Language (SMIL) Specification*, World Wide Web Consortium Recommendation (June 1998), http://www.w3.org/TR/REC-smil.

[SYS98] Schmitz, P., Yu, J., Santangeli, P.: Timed Interactive Multimedia Extensions for HTML (HTML+TIME): *Extending SMIL into the Web Browser*, Note for discussion submitted to the World Wide Web Consortium (September 1998), http://www.w3.org/TR/NOTE-HTMLplusTIME.

[Aya00] Ayars, J. et al (ed.): *Synchronized Multimedia Integration Language (SMIL) Boston Specification*, World Wide Web Consortium Working Draft (June 2000), http://www.w3c.org/TR/smil-boston.

[RHO99] Rutledge, L., Hardman, L., v Ossenbruggen, J.: *The Use of SMIL: Multimedia Research currently applied on a Global Scale*, in Karmouch, A (ed.): Proceedings of Multimedia Modeling 1999 (Ottawa, Canada, October 1999), pp. 1-17.

[SCR95] Soares, LFG., Casanova, MA., Rodriguez, NLR: *Nested Composite Nodes and Version Control in an Open Hypermedia System*, International Journal on Information Systems, 20, 6, Elsevier, 1995, pp. 501-519.

[KFS00] Kárpáti, A., Feustel, B., Schmidt, TC.: *MIR - A Multimedia Information Repository*, in preparation.

[ShSu00] Shi, Y., Sun, H.: *Image and Video Compression for Multimedia Engineering*, CRC Press, Boca Raton, 2000.

[LeKn92] A. LEWIS and G. KNOWLES: *Image Compression Using the 2D Wavelet Transform*, IEEE Trans. on Image Proc., 1 (2), 1992, pp. 244–250.

[MaCy97] D. MARPE and H. L. CYCON: *Efficient Pre-Coding Techniques for Wavelet-Based Image Compression*, 1997, Proc. PCS '97, pp. 45–50.

[MaCy99] D. MARPE and H. L. CYCON: *Very Low Bit-Rate Video Coding Using Wavelet-Based Techniques*, IEEE Trans. on Circ. and Sys. for Video Techn., 1999, 9 (1), pp. 85–94.

[MHCP99] D. MARPE, G. HEISING, H. L. CYCON and A. P. PETUKHOV: *Video Coding Using A Bilinear Image Warping Model and Wavelet- Based Residual Coding*, Proc. SPIE Vol. 3813, Wavelet Application in Signal and Image Processing VI, Juli 1999, pp. 401-408.