A distributed media server for the support of multimedia teaching

Michael Liepert[1], Carsten Griwodz[1], Giwon On[1], Michael Zink[1], Ralf Steinmetz[1,2]
[1]KOM, Darmstadt University of Technology, Merckstrasse 25, 64283 Darmstadt, Germany
[2]GMD IPSI, Dolivostr. 15, 64293 Darmstadt, Germany
medianode@kom.tu-darmstadt.de

ABSTRACT

One major problem of using multimedia material in lecturing is the trade-off between actuality of the content and quality of the presentations. A frequent need for content refreshment exists, but high quality presentations can not be authored by the individual teacher alone at the required rate.

Several past and current projects have had the goal of developing so-called learning archives, a variation of digital libraries. On demand, these deliver material with limited structure to students. For lecturing, these systems provide just as insufficient service as the unreliable WWW.

Based on our system HyNoDe [HYN97] we address these issues in our distributed media server built of "medianodes". We add content management that addresses teachers' needs and provide guaranteed service for connected as well as disconnected operation of their presentation systems.

medianode aims at a scenario for non-real-time, shared creation and modification of presentations and presentation elements. It provides user authentication, administrative roles and authorization mechanisms. It requires an understanding of consistency, versioning and alternative content tailored to lecturing. To allow for predictable presentation quality, medianode provides application level QoS supporting alternative media and alternative presentations. Viable presentation tracks are dynamically generated based on user requests, user profiles and hardware profiles. For machines that are removed from the system according to a schedule, the systems guarantees availability of consistent, complete tracks of selected presentations at disconnect time.

In this paper we present the scope of the medianode project and afterwards its architecture, following the realization steps.

Keywords: Distributed Servers, Multimedia Servers, Quality of Service, Multimedia Lecturing

1 INTRODUCTION

The use of electronic teaching material nowadays is done in either of two ways. The foremost way of giving the learners access to learning material are CD-Roms. The second way that gains in popularity among teachers is the use of computers as a flexible replacement of overhead projectors in lectures. In both cases, the main reason for the development is the possibility of extending and maintaining the typical teaching material that comprises text and images, as well as a cost reduction that can be expected in the long term.

In both cases, the material is prepared for each single presentation manually, which is work intensive and error prone. Furthermore it is unlikely that the collection and assembly process of the material is documented, which would allow the re-use of the material in another context. Timeliness of the material can thus only be achieved by a teacher through repeating the process of collection and assembly again and again. Furthermore, information material that is the basis for the teaching material must be bought repeatedly.

In contrast to this, the World Wide Web seems like an endless source of knowledge that is kept up to date continuously. It provides textual as well as graphical material and, in a rather restricted way, additional audiovisual material. For distributed learning and teaching, however, this system is not used. Because of a lack of long-term continuity and immediate access to the content, the lack of means for combining the web material with additional material, as well as obvious problems with the consistency of complete presentations and in the transmission quality, the web can not be used for teaching, while it has some restricted short-term value as add-on informative material.

The goal of the medianode project is the creation of an infrastructure for the support of multimedia-enhanced teaching in Hessian schools and universities. Especially for the education in technical areas, it is today hardly possible for the individual teachers to keep their teaching material continuously up-to-date, if this teacher works for himself and does not exchange material with others. An education offer that is not up-to-date anymore, however, will not be accepted by the students.

Using the rapidly evolving presentation tools of the world wide web for access, the project will create a distributed system through integration of the approaches for production, distribution and consistency maintenance, which will grow from a central node that operates as a production center and archive for the end user through a flexible, decentrally organized system of regional nodes.

The future teaching support system is envisioned to support the creation of presentations from raw material rather than being only a material archive for lessons. In order to do this, it is relevant to note that material can rarely be used without modifications. Teachers' personal style, personal opinion and directions of specific classes differ with each teacher and their expression is as necessary as the re-use of the content itself. Furthermore, the quick integration of the latest material must be achieved.

With medianode, we create a system that takes products and previous research into account and we plan to go beyond these developments for some specific research issues. However, we want our system to fulfil real world requirements and to be used early within the timeframe of the project. For the design and development of our system, however, the earliest possible use and the adaptation to long-term technical extensibility compete with each other. This document provides directions for the development of the system and provides weak limits to the developments issues. We expect these limits to harden when feedback from initial system users requires immediate solutions for some of the issues.

1.1 Lecture Archive

Special lectures, guest lectures and seminars are frequently interesting beyond the scope of one institute or even a single location. For this reason, the creation of video archives or even the creation of commented presentations with central video content are frequently interesting. For the appropriate use of this material, various modes should be supported:

- If the lecture is interesting only for a single person, the possibility of local replication of the content in case of very bad network infrastructure, and the adaptation of bandwidth requirements by format conversion to lower quality with consecutive streaming of the lecture must be distinguished.
- If the lecture should be re-used as a part of another presentation, additional information is relevant, e.g. supportive presentation material such as slides or information that concerns the applicable combinations of individual presentation parts.
- If the lecture is used by groups of students in a single institute, the distribution system should apply caching on the receiver side to achieve an appropriate quality.

1.2 Availability of Functionality

Frequently ignored but absolutely relevant for the use of electronic means in lectures and presentations is the possibility to continue the presentation in case of various faulty components. Besides the loss of electrical power and a breakdown of projectors -which can be compensated only with major efforts- a variety of problems can be addressed automatically:

- No network access: all on-line access to information becomes impossible, thus a complete, consistent presentation must be available on the presentation system, even if the presentation loses its audio-visual parts
- Screen resolution the use of on-site projection systems restricts the available screen resolutions
- Machine crash: the presentations must be easy to be identified, found and transferred
- Audio support: loud speaker systems are frequently incompatible with the lecturer's equipment. Various options should be considered and tested to solve this; finally textual presentation of audio parts should be considered as an alternative
- Video support: Because of its size, video material usually can not be kept in more than one version on the presentation system. In case of a last-minute machine change, it is most likely that video presentation will become impossible. The presentation should provide an alternative to this video presentation.

1.3 Access control

Because of the financial situation of research and teaching in Germany, lecturers are not interested in the free access to their teaching material by non-students, especially companies. Because of this, all parts of the media server network must integrate access control and copyright verification mechanisms, which should be implemented in a number of steps:

- System security of the individual machines of the server network must be ensured in order to protect the data from illegal access. This is a system administration issue, but the project must not introduce or enforce security holes into a system.
- The access of the data on a server must be protected by authentication and authorization mechanisms.
- The system has to reliably control, log and eventually prohibit the introduction of (mis-) information. This protection is sensible only when consistency maintenance for presentation is supported by the system, and while consistency checking includes access control information into the checking procedure.

- Future extensions with accounting mechanisms on an application level based on user authentication can be used to allow the opening of access to external users. The use of electronic cash may be considered for an integration of billing into the system.
- Since the use of the distributed system in the Internet is the goal, additional protection mechanism for the network access must be included. This requires the use of encryption techniques and probably the introduction of an additional trust center for provision of keys.

2 ARCHITECTURE

The initial design of the multimedia node implements only minimal functionality, but that allows for dynamic extensions that support more complex requirements (Figure 1). The initial functionality includes a web server front-end, a standard file system and localization information that represents exclusively the placement of data on the local disk. Figure 2 in Section 3.2 shows the structure of such a generic multimedia node.

The kernel of the server allows for the dynamic loading of modules, which adhere to interfaces that are defined in the initial phase of the project. In order to do this, consistency and security mechanisms are implemented in addition to the communication among modules.

A module of the server implements the interface of a function block out of the set of defined function blocks (currently Access, Storage, Verifier), where each interface of a module must be implemented at least a rudimentary basic implementation.

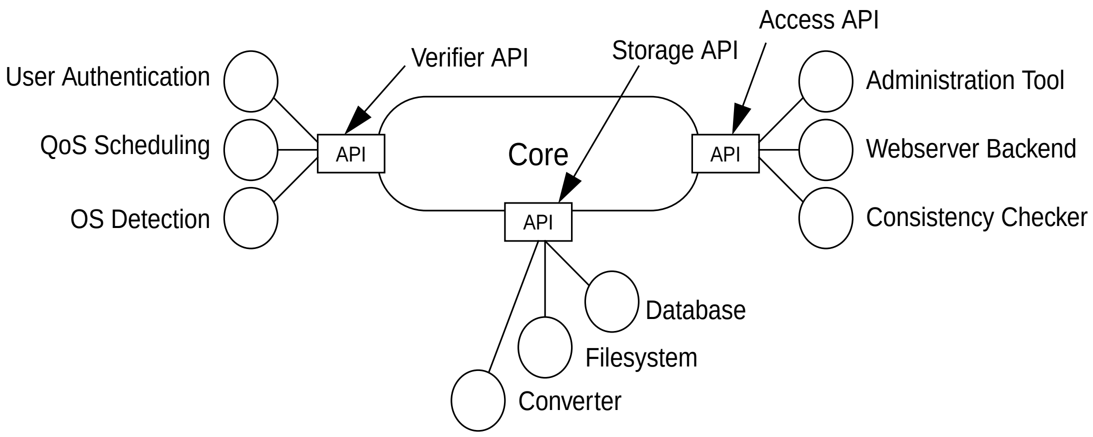

Figure 1: Example modules and their APIs assignment

Each interface is implemented as a base class.

A variety of techniques for automatic distribution can be tested in such a configuration: web caching, AV caching, pre-distribution (“push caching”), prefetching, segment replication, scene replication, version control, consistency of versioned documents, and the selection of alternative content. Furthermore, the distribution problems on the application level are extended by other research aspects, such as security aspects, billing systems, copyright protection, and the processing of expired copyrights.

The Core: A Dynamic Loader, stripped off functionality

The web server will be built around a core that implements only the functionality of dynamically loading and unloading modules and provides basic platform dependent operating system services to these modules. This design decision is motivated mainly by the targeted scalability.

Dynamic loading in a platform independent way is a main issue of middleware. We will use appropriate concepts and definitions from the ISO middleware standard CORBA [OMG98] where possible. Still, the core has to be able to be implemented without including any CORBA runtime implementation, again for resource reasons.

The following list summarizes the initial decisions concerning the core module:

1. There will be a core module only capable of dynamically loading further functionality.
2. This module will be implemented in a platform dependent way (Our Choice: C/C++)
3. This module will be provided also in a form not requiring a CORBA runtime implementation, but we will try to adopt

interface specification concepts from CORBA to allow easy connection to CORBA implementations. This may include important CORBA interface definition language IDL concepts like IDL basic data types and procedure call qualifiers.
4. The core module will provide a core platform abstraction to access machine resources to cover two goals: the division of platform dependency from functionality, more important, to supply the system with means to know and manage its own state, including allocated resources.

Loadable Functionality in Bows

Complementing the core, all target functionality will be packed into loadable software pieces. What we require from the loadable functionalities implementation is analogous to the concept of component software like OpenDoc [AC95], ActiveX [CK] and JavaBeans [JF97]. Thus, we benefit from experiences with the widely accepted, mature component approach.

We adopt the general component characteristics for our lodable modules, e.g. they are deployable, maintainable, functional, specific, self describable etc., but extend and adapt in a way specific to medianode's needs. As we decided for the core/components approach, we wanted to be able to use the mediarug to store, distribute and configure its components. This results in eliminating the difference between data and system. Further, high reliability and also remote testing demand for extended configuring and debugging features. Thus, requirements on our lodable modules, called *bows*, additional to standard components are:

1. A bow has to provide two means of high-level reflection: a human readable description of itself and a system-usable description of its application interface, in terms of mediarug meta-data.
2. Bows have to provide means to collect dynamically generated control information like warnings or error messages; the generation and redirection of these information streams has to be dynamically controllable.
3. To be safely integratable into the medianode, all bows have to use platform abstractions provided by the core module to access the platform it is running on.

We define one uniform interface for bows, providing the functionality listed above. There are more concrete subclasses of the bow root class, namely storage bows, verifier bows and access bows. Examples for mediabow classes are network access bows for the various network protocols, management access bow, (distributed) file system storage bows, database storage bow, conversion storage bows. Examples for verifier bows are security verifier bow controlling access, but also QoS verifier bow controlling e.g. playout schedules.

3 REALIZATION

In order to implement a flexible media server, but to provide a basic functionality quickly, the implementation schedule is split into multiple phases. The two main targets of the project are to provide means to maintain and present lectures to multiple, distributed lecturers and to provide means to prototype and evaluate new concepts in multimedia communication. Thus, a useful (i.e. cheap, fast, usable) product for the end-users and a widely extensible experimentation platform should evolve. We conclude from that, that we can not rely completely on given platform abstractions for resource reasons (they are very expensive or slow) and for maintenance reasons (we have to have control on platform abstractions).

3.1 Step I: Definition phase

In the definition phase various use case scenarios are formulated and the technical requirements for the implementation of these scenarios are collected. This is done in cooperation with the teachers at KOM and other institutions in Hessen who are interested in the use of the system prototype. Based on this, the interfaces for the functional blocks and the information paths in the server are defined.

The implementation of functionally “empty” blocks is also part of the definition phase. This includes the interface definitions and the implementation of the dynamical access mechanism.

3.2 Step II: Web server

The next implementation phase delivers a web server (see Figure 2) that operates on a standard file system and delivers data exclusively through the HTTP protocol. This implementation is not intended to exceed the functionality of any web servers that are currently in existence. It is rather implemented as proof-of-concept for the core functionality of the media node.

The insertion of content into the system in this phase works exclusively through simple mechanisms such as NFS mounts or FTP. The required storage module is a 1:1 mapping of the file system. First access control mechanisms are already required in this phase, e.g. by using a firewall that is administered in conjunction with the server.

The web server itself consists of the latest version of the Apache web server.

The Apache web server supports many features that are needed by the medianodes. Additionally special requirements for the medianode architecture can be included through extensions written as dynamically loadable libraries. If necessary also changes on the server itself can be made since the source code is available.

The design decisions for that phase are based on the following high level goals:

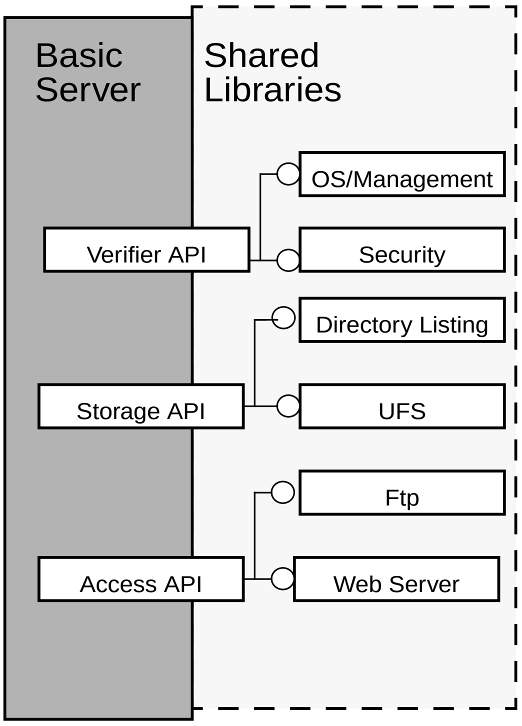

Figure 2: generic multimedia node

- scalability
- uniformity of interfaces and implementation rather than on platform dependent optimizations (Still, platform independence using a virtual machine like sun's Java is not feasible due to e.g. speed demands of the targeted network functionality), source code portability
- abstraction of platform dependent functions should be separated from the target functionality where possible.
- We want to enable the use of the mediarug's storage and distribution facilities to store, distribute and manage its components and its dynamic state.

Personalization: Authorities, Roles, Profiles, Copyrights

The mediarug has to provide support for the storage and use of multimedia documents for distributed groups of teachers and students. From the user's view, that demands for a) fully reliable reproduction of decisions and designs of the authors and for b) reuse of work. The first goal a) is an unconditional requirement, so only the reuse has to be optimized without damaging the integrity of the produced content. User authentication and access verification is a central issue to be considered and integrated into the medianode form the beginning.

Authorities demand reliable identification and authentication of all users. The basic concepts the mediarug has to implement are the Access Control List and user groups, which are used by web servers to provide privacy. Since ACLs are specific to the organization of data in web servers, i.e. in hierarchical directories, and since the medianode will provide OS like functionality, it has to change and extend the pure ACL approach:

- web serving: ACLs, groups
- copyright: ownership, expiry
- OO/SQL data organization: ACLs should be 'inherited' along various relations, like *part-of*
- OS-like fine granularity: ACLS should also be valid for relation slots/OO-parameters

Especially the third point recommends to integrate authorities into the meta-scheme of the data organization .

The authorities allow for the implementation of **roles**. These roles can include various types of users like consumers, authors and system administrators. Out of practical experience on HyNoDe and InfoDesigner [STKO95], we will divide the role of the author into author and designer, to separate the personal design from the factual content.

As a user is authenticated, he can be linked to a **profile**, i.e. the data known about him. This data includes a profile, that provides the system with information about factual and environmental characteristics of that user. Factual characteristics can identify preferred content or structure (like language, sequence or explanation depth), technical characteristics identify system constraints or preferences, mostly concerning size or format . A user profile is that user's preferred meta data of content including demanded mandatory and optional meta data. Here, also weights for preferred meta data can be introduced.

To maintain and enforce **copyrights**, an initial owner has to be defined to set access rights. Naturally, this will be the originator. As the owner restricts access to his property, the mediarug also could grant means to define strength and eventually time span of that restriction, e.g. to use watermarking and secure transport protocols for a special content.

3.3 Step III: Streaming Server

The Streaming servers enhance the web server by additional bows that include proprietary video servers into the system. The intention is to have an extension towards AV capabilities in a very early phase of the medianode project. Content distribution and insertion remains a manual issue in the beginning.

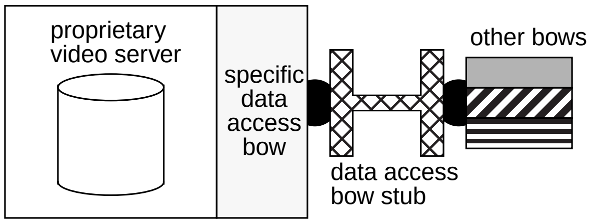

Figure 3: Integrating a video server as a data access bow

To the rest of the system, the streaming servers present themselves as a storage bow. Since the bow is frequently expected to be located on a different machine then the main access node of a clustered medianode, these bows need to implement major parts of the functionality for monomedia handling by themselves. Furthermore, upload, distribution and delivery mechanisms of these servers are more or less proprietary and need to be handled outside the main communication channels. Figure 3 shows that a generic data access bow stub should be used to pass control information back and forth between the medianode that hosts the video server and media nodes that handle less specific parts of a presentation.

The same mechanism is used for all other specialized data access bows that provide a local control API only but that should be integrated into the server. It may not be immediately clear why this approach should be taken. The following three products that we intend to use for data access bow implementations are candidates for the use of a data access bow stub.

3.4 Step IV: Introducing presentation term and consistency

Before the consistency of presentations can be used for their distribution, the term presentation is defined. This step defines the structural model for presentations. This model will include alternative presentation, alternative presentation paths, alternative content, temporal order and the use of on-line and off-line conversion. Depending on this model, partial presentations can be identified and distributed consistently based on the QoS requirements of their individual elements. In this step, the file system must be augmented by a database system that supports the storage of sufficient information for the identification of consistent sub-presentations.

Presentation consistency was also an issue in earlier projects that we have been working on, such as GLASS, IBM's InfoDesigner and the HyNoDe system.

The Berkom GLASS (Globally Accessible Services) system was an experimental on-demand system with the goal of evaluating the viability of an MHEG-1 implementation. MHEG-1 was generally considered too generic to be implemented, and the GUI implementations of our distributed engine had cooperation problems due to freedom of interpreting execution order in the standard. The functional subset that is necessary for real-world applications is reflected in the comparatively simple MHEG-5 standard.

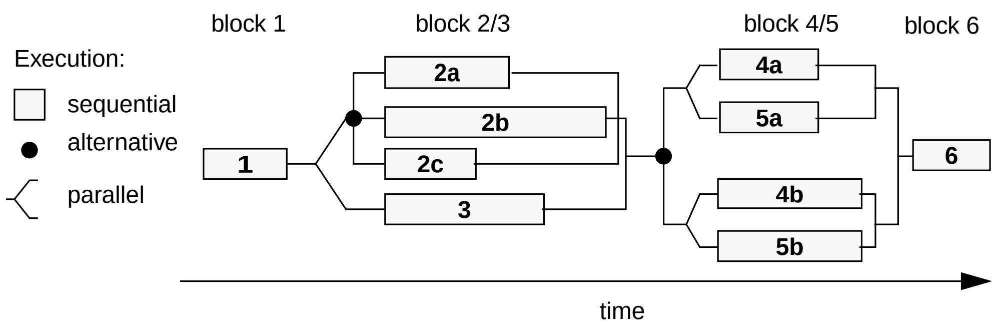

Figure 4: Execution order

The problems of the GLASS engine provide important input for the specification of the presentation term. Several models exist for the scheduling of multimedia presentations. We demand for our model that presentations have simple, strict terms in which to describe order and co-presentation. We believe that at least delay sequences are required in addition to sequentially and parallelity. Furthermore, we need a definition of separating and joining alternative paths.Obviously, the presentation specification must explicitly or implicitly define when timelines join after parallel or alternative execution. of 2b?

IBM InfoDesigner/2 (ID/2) is a product that aims at the multimedia kiosk market. Key issues are adaptability to the quickly changing monomedia and multimedia encoding formats, long-term stable execution of the running system, the use of templates, access rights, varying levels of permission for customers, and the integration of (black-boxed, partly instable) third-party code.

The original ID/2 system has introduced an interesting OO approach that allows the dynamic creation of object relations from basic objects. Subsequently, these relations can be fixed and become objects themselves. Figure 5 shows the principle.

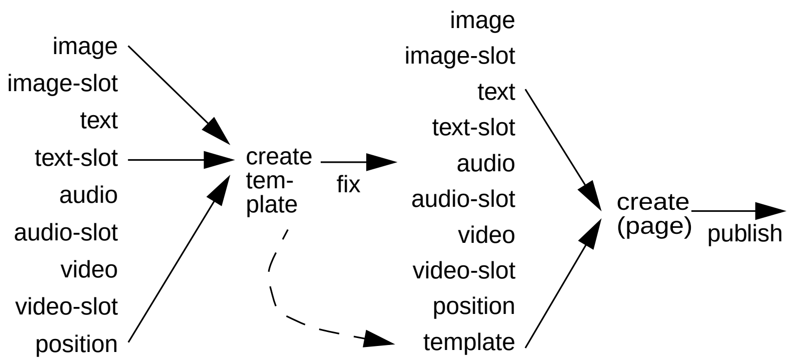

Figure 5: Generations of presentation elements

An extension to the scheme has become necessary: the system is supposed to allow importing and exporting of and working with a variety of encoding formats for the multimedia presentation itself (e.g. ID/2 format, HTML, MHEG-5).

The EC-sponsored Hypermedia News-on-Demand (HyNoDe) project is an integration project that aims specifically at the business news market. While the commercial interest in the project centers around delivery of consistent, personalized, multimedia-enhanced news by means of a hierarchical distribution system, the initial design focused very strongly on QoS issues. The initial design of recursive QoS calculations and the interaction with network level QoS negotiation was dropped due to the limited time. Figure 6 gives an impression of the original design. The figure describes a part of a presentation that combines an HTML text with optional audio-visual information. The audio-visual information may either be an MPEG video, a combination of H.263 video with G.723 audio, or a simple image that is accompanied by the same audio track. The combined QoS description is not very attractive. But unfortunately, one typical assumption of QoS-related work is wrong: continuous media presentation elements are not freely scalable in arbitrary combinations. Whether the QoS computation results in overlapping alternatives, or in empty regions (as is the case in Figure 6), a decision must be made to deliver one encoding or another. Furthermore, state-of-the-art client software tends to be too inflexible to make this change on the fly.

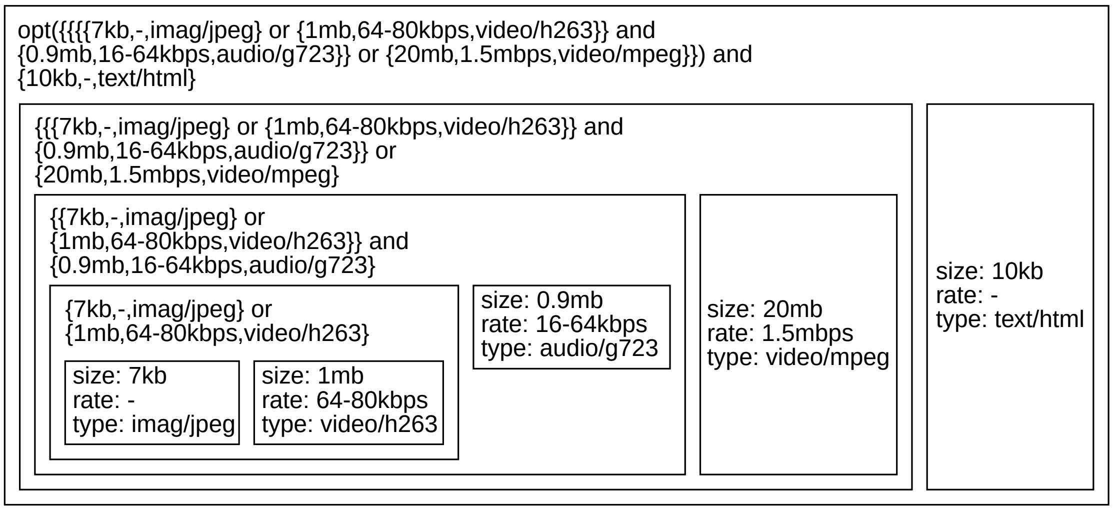
opt({{{{7kb,-,imag/jpeg} or {1mb,64-80kbps,video/h263}} and {0.9mb,16-64kbps,audio/g723}} or {20mb,1.5mbps,video/mpeg}}) and {10kb,-,text/html}
{{{7kb,-,imag/jpeg} or {1mb,64-80kbps,video/h263}} and {0.9mb,16-64kbps,audio/g723}} or {20mb,1.5mbps,video/mpeg}
{{7kb,-,imag/jpeg} or {1mb,64-80kbps,video/h263}} and {0.9mb,16-64kbps,audio/g723}
{7kb,-,imag/jpeg} or {1mb,64-80kbps,video/h263}
size: 7kb rate: - type: imag/jpeg
size: 1mb rate: 64-80kbps type: video/h263
size: 0.9mb rate: 16-64kbps type: audio/g723
size: 20mb rate: 1.5mbps type: video/mpeg
size: 10kb rate: - type: text/html

Figure 6: Recursive QoS definition

3.4.1 User Abstraction/Interface to QoS

An abstract interface must be built that allows the user to set these parameters in a simple way. One of the solutions could be a simple slider which is used to improve the QoS. To realize this an algorithm must be found that translates the slider information into parameters which can be used for the QoS signalling. There are already some existing ideas of algorithms which could be used for this translation like in [RRWA98]. Some of the existing or even new created algorithms will be implemented in the QoS interface to figure out which of these provides the best translation method.

Afterwards it should be investigated how local QoS could be involved to raise the quality of a presentation. Local QoS means for example to execute certain functions like a decoder for video data with a higher priority by allocating pinned memory and CPU cycles than a word processing software that is used for showing some text, because the video is jitter and delay sensitive.

3.4.2 Alternative Media

An important concept to provide a guaranteed quality of service to the user and to adapt to the resources available to the user are alternative contents. This means that the system is able to provide various content representations to a user. To avoid confusion, we will use the term media brick for a set of content representations equivalent in content, but eventually not in dimension, coding format or coding parameters. Thus, a media brick is the definition of a set of (media brick or content) representations.

There are three concepts to achieve alternative content:

1. Statically storing all different representations of a content and explicitly enumerating them inside the media brick;
2. Dynamically applying a converter on a representation but explicitly define the possible converters and thus target formats inside the media brick and
3. Dynamically applying conversions independently from the media brick definitions.

The initial version of the mediarug will only support standard WWW access, only allowing for the first two alternative content concepts. The data structures and interfaces to access representations using the first and the second method are identical, so the first approach still allows to introduce the second method without changing any of the system interfaces. The third approach can be introduced by explicitly introducing new access methods or protocols into mediarug applications. Thus, the system data structures and interfaces will explicitly only state the first possibility, because this is sufficient to cover the mediarug system requirements concerning alternative media.

There are consequences on the technical system design that can already be drawn from the necessity of alternative media representations and media bricks:

1. The data structure of the mediarug has to provide a data entity *media brick*, representing a relation *has equivalent content* by naming a set of representations.
2. The data scheme of the mediarug has to provide the necessary meta data for each media representation allowing for identifying an appropriate representation.
3. All references from media content to other content have to be references to media bricks rather than representations

3.5 Step V: Loading presentations

The insertion of presentation elements into the archive will be automatized in this step. Depending on the available material, one or two approaches for adding meta information to existing presentation material and for subsequent transmission of the information to the (remote) central server should be supported initially. The first control approach should be command line-oriented, the second should be graphical.

3.6 Step VI: Re-use of archived material

In the next step, the available material should become usable so it can be included in new presentations. Especially the development of very simple specification means is planned, which allows the creation of new presentations on a very high abstraction level. In such a case, support for the creation of alternative presentation paths would be provided by server-sided processes.

3.6.1 Work Coordination & Content Versioning

At a more elaborate stage of the mediarug, multiple persons will be able to edit mediarug content using the same access methods as when retrieving content. This introduces access conflicts and the need for explicit versions of content. We predetermine the following minimum requirements from the strong editor-, i.e. professor-centric, driven mediarug functionality specification:

Because editors have to be absolutely positive about the content that will be displayed,

- there has to be a definite, easy, way (i.e., concept and (G)UI) to identify current, own content.
- therefore, a media brick and a special media representation will always be the user's newest edition to that user. Access of former versions can be stated as experts' action, and it is likely that we will not allow branched versions but only newly copied content.
- change of multiple content use: as a user x changes content that is (indirectly) part of a document X other than his current work focus, the system will
  (1) notify the user that at change confirmation time, requiring confirmation to adopt the changes into X, if X is in possession of user x; or the system will
  (2) inform the owner of document X of the possibility to adopt a change into X at a convenient time
- thus, there has to be security on owner/content-relationship. We will keep the option of introducing the concept of a users' group as document owner.

The system has to support reuse of content for many reasons. This is in a trade-off to the reliable reproduction stated above and can only be optimized as long it does not violate that first target. Reasonable answers to that are:

- Content has to be in modules, pieces, i.e. media bricks
- Information in a content likely to be personal (style and personal/corporate design) has to be split from factual information (content) where possible, introducing roles in the set content and editors.
- There has to be a definite way to identify an unchangeable piece of content, therefore there has to be an archive of versions and editions. For that, a concept like a release or final document is needed to characterize a set of versions. Even the owner of content can not be allowed to change a former version (this latter point could force the introduction of branches)
- There has to be support (a concept and GUI) to query and find other versions and also whole consistent releases.

For the mediarug, we will explore the various solutions to this kind of problems already existent, and only introduce new ones if not available.

Operating and file systems that allow access by concurrent users provide means to solve access conflicts and to lock resources. Database and transaction approaches used for computer supported collaborative work, CSCW, provide solutions mainly for concurrent work, locking and safe transactions. Configuration management and version control tools are mainly used for software development. These tools cover all aspects of versioning and configurations in document editing, thus defining an upper bound for the respective target functionality. Especially handling of versions, differences and branches will be adopted from these tools.

The main, and in the beginning only access method of the mediarug will be the WWW, which means the protocol HTTP. As an upcoming standard, a main starting point to introduce work coordination and content versioning capabilities into the mediarug is the work of the WebDAV working group of the IETF. This working group is defining the HTTP extensions necessary to enable distributed web authoring tools to be broadly interoperable, while supporting user needs.The WebDAV group has analyzed the functional needs of several organizations, and has developed requirements for distributed authoring and versioning [IETF98].We will extend the IETF WebDAV concepts by

1. adaptation to the medianodes' authentication and verifying concept
2. defining a way to integrate non-HTTP protocols
3. identification and adaptation of implementations from OS/FS, CSCW and CM/VC to the WebDAV-specifications.

3.7 Step VII: Caching server

In this step, a medianode variation will be created that supports the distribution of data from the archive. Initially we will implement pure cache servers that work exclusively on data that has been requested at least once. In contrast to web caching, however, we will take temporal behavior and temporal dependencies among elements into account.

Since our final goal assumes a decentralized system, respectively the possibility to merge multiple distributed servers that have grown around central administrative servers, into a single one, we must not build a system that relies on contacting a central server for each operation.

We will also experiment with cooperative caching approaches to provide sufficient disk space for the support of audio-visual material.

3.7.1 Updating

Our approach is bound to support updating of cached presentations. The best approach to do this is not clear. We can envision various options:

- Central maintenance
- Verify access: For each access to a document, the origin of the document is contacted and the cache server verifies that the content is still up to date. In case of a hierarchical distribution system, the verification is made with the next server in the hierarchy rather than with the origin of the server.
- Callbacks: Install callbacks at origin server of each document.f we assume this approach, it is noteworthy that callbacks are to be installed among servers, not among client and server of our system. We will also need callbacks at various abstraction levels. It is necessary to determine whether client and server hold synchronized callback information, especially after connection or system downtime when both sides have been affected and simple reference counting in the way of DNS does not work any more. Note that update notifications may be handled with multicast.

In all of these approaches we must take into account that documents are modified at other servers than the central server, and that such modifications are made while the medianode is in disconnected operation.

3.7.2 Positioning

Positioning of content can be considered at more than a single granularity for each system, in our case we might consider positioning in a single machine, a single medianode or among medianodes. Positioning in clusters is done e.g. by the Hermes Long Term Research project [CZ98]. We may be able to receive the code for such operations from that project and will not consider positioning in a single medianode.

Positioning in distributed servers depends on the load and accessibility of the server's nodes. We will consider the need for content first - need meaning replications that are necessary because it is relevant to enable access to content for sites that are badly connected or disconnect from the rest of the distributed server frequently. In this respect, we want to consider presentation consistency.

Since the system can be given some meta-knowledge about the network infrastructure, it is aware of network bottlenecks and the distribution of content can be scheduled appropriately to make a minimal consistent presentation available first, followed by alternative content of higher quality. When such a download has been applied, caching strategies are used to keep the most relevant presentation available at the medianode.

Since we assume that splits and joins of the distributed server are part of the everyday operation, we are limited in the application of optimized content distribution in a distribution network. However, we consider the optimization of content positioning in a distribution network with multiple origins an issue to be addressed, and the results to be introduced into the system. In contrast to web caching, which has the goal of unloading origin servers, we are interested in exploiting the resources of the medianodes of our distributed server fairly. Furthermore, we want to use local decisions of the individual medianodes -that are in communication with their neighbors and receive hints-, we do not want to centralize decision-making for content positioning. We consider this decentralized operation an asset mainly for two reasons:

- central control is incompatible with the assumption of server splits, which are considered part of our regular operation,
- "uncontrolled" growth of cooperating systems without central control may be more attractive to other users and can be faster.

Our resynchronization mechanism provides a restricted protection from network downtime. However, we intend to consider aspects of backup routes as they are implemented in video server work at USC.

Lectures are typically used once per year or once per term by a teacher, they are used by students for post-processing the lecture soon after the lecture and again, before tests.

We want to use these observations to introduce advanced distribution mechanisms and to exploit the knowledge about the presentation schedule and the presentation consistency that is stored in the database, in order to support a variety of networking situations. This ranges from the prefetching of complete presentations for off-line operations to the prefetching of individual elements in real time of the presentation based on user behavior.

3.7.3 Server (re-) synchronization

In the case of a failure (network or hardware) communication between to medianodes is not possible. This leads to the problem that information in the different medianodes are not consistent anymore. By implementing a server (re-)synchronization technique it will be assured that this problem can be avoided. Here it seems reasonable that the medianode which was not able to communicate with the other nodes sends a note to one of his peers that it was out of sync. Since all nodes log there synchronization processes they know from which point on a resynchronisation has to be performed.

3.8 Step VIII: Additional Tasks

The following issues are not bound to any special implementation phase, but require some maturity of the system. Additional tasks is introduced by adding additional specific bows, using the specified system interfaces rather than changin vital system characteristics.

3.8.1 Extensible Format Awareness; Functionality Brokerage

To achieve format awareness, we introduce semantic information about representations' formats into the systems (meta-) content. So, the system will be able to know semantic relations of its formats. The most important semantic relation is possible equivalence (JPEG and GIF e.g. as two image coding formats) and possible conversion/converter. Since the multitude of e.g. multimedia formats evolves quickly, means have to be defined to extent the system of formats while reliably maintaining meta data consistence.

Using format awareness, we will try to achieve a process of functionality brokering by evaluating the available meta data to automatically identify the functionality and its appropriate representation to achieve a certain addressable mediabrick representation. Format awareness and functionality brokering require a uniform format and semantic of meta information on system state, users, media and available implementations (bows).Transport/Computation Load Levelling ml

Format awareness and functionality brokering enables the mediarug to compute various ways to deliver content. One use of that ability could be achieved by introducing time information into meta data. Available time information on static and dynamic items (network lines and connections, user preferences and computation speed of machines and bows) could enable the system to chose better sequences of computation and transport of addressable content, e.g. to tune its configuration.

3.8.2 Network level QoS

The communication between medianodes in a mediarug is done via the Internet, which nowadays does not allow the transmission of real-time data like audio and video in an acceptable quality. To be able to transmit these data with a better quality the Internet protocols have to be extended by QoS mechanisms (Network level QoS). Therefore the existing IP implementations of the medianodes have to be extended (if not already done) in a manner that allows to support network level QoS. In addition the client has to be extended to be able to signal the desired QoS to the medianodes. The QoS signalling between the medianodes will be done by RSVP (Resource ReserVation Protocol) [BRA97]. Further more mechanisms which allow to handle data with a higher priority in a preferred way (packet scheduler and classifier) will be implemented in the medianodes. Figure 7 shows the modules an their interaction for the network level QoS.

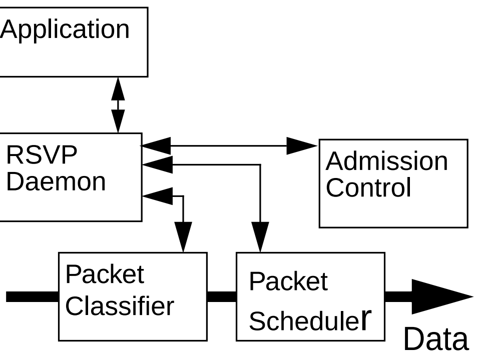

Figure 7: Network level QoS

3.8.3 Neighbor detection

A whole medianode system consists of any number of single medianodes which act also as servers, but can also have client functionality (e.g. the notebook of lecturer). There is no centralized instance in medianode that keeps track about the single nodes and their functionality and possible special task (video server, authoring,...). Each node has to find all the other nodes of a system and also collect information about their functionality.

The single medianodes use a specific multicast address to send a multicast message to all the other medianodes which have a server running listening for this messages on this multicast address and a specific port. Both multicast address and port can be set freely but must be the same on all medianodes in one system. When a medianode starts it sends a request as a multicast message including the specified multicast address and port and afterwards waits for answers from other medianodes.

If one of the servers goes down in a controlled way it also sends a multicast message to the other peers in order to inform them that it is not available anymore. The other nodes delete the entry they have been informed about from their databases and consequently do not contact the node that became off-line anymore. To make sure that medianodes also recognize a crash of another medianode the protocol should be updated in a way that the multicast requests are repeated in certain intervals. If one of the already known address is not validated it will be deleted from the local peer database.

4 CONCLUSION

This document presents the steps for the creation of a distributed multimedia server that will be installed in various locations throughout Hessen under a central control, which is intended to provide teachers at schools and universities in Hessen a means for common use and interchange of multimedia-enhanced teaching material.

The basic idea of the project is the necessity of adapting teaching material to the state-of-the-art continuously and to present the material in a state-of-the-art manner as well. We want to take into account that the effort for continuous maintenance of the teaching material is too big for a single person, especially where multimedia-enhanced presentations are concerned. The means that we propose to overcome this is a cooperation that includes the exchange of teaching material among teachers, which can lead to a distribution the effort and achieve the important timeliness. Since multimedia elements gain relevance in state-of-the-art presentation, we need to address also the effort that is required for the inclusion of such elements into the presentations. The media node system wants to address both problems and to provide a practical solution.

In order to achieve early initial use as well as the extensibility of the system for a long time, the ease of learning the system, using the system, long-term consistency of the available data and resistance against a multitude of errors will be addressed. Platform independence and scalability are issues that must be addressed in order to remain open for future extensions. Commercial considerations as well as security considerations play an increasing role in teaching that is funded by the public, which must also be taken into account.

The envisioned software for a network of media servers will be growing in steps. This approach is taken to make early version of the system available, to get user feedback early and the adapt flexibly to the requirements of the users. The abstract goal of the project is an achievement of the listed functionalities and high a degree of adaptability to future requirements.

REFERENCES

[OMG98] The Common Object Request Broker: Architecture and Specification. Object Management Group; February 1998. http://www.omg.org/corba/c2indx.htm

[AC95] OpenDoc Programmer’s Guide; Apple Computer, 1995

[JF97] Web Developer’s Guide To JavaBeans. Jalal Feghhi; Coriolis Group Books, 1997

[CK] ActiveX and the Wb, Arcitecturte and technical Overview. Charlie Kindel, Microsoft Developer Relations Group

[HYN97] "Personalized News on Demand: The HyNoDe Service", HyNoDe Consortium, Lecture Notes in Computer Science: Proceedings IDMS ’97, 1997, Springer Verlag, Heidelberg/Germany

[IETF98] Requirements for Distributed Authoring and Versioning on the World Wide Web", IETF 1998

[STKO95] Ein verteiltes Multimedia-Kiosksystem: Anforderungen, Architektur und Erfahrungen; Ralf Steinmetz and Daniel Köhler, Themenheft it+ti, 1995

[RGVW98] Mapping User Level QoS from a Single Parameter. Athony Richards et. al. . MMNS’98

[KG98] Möglichkeiten für den Einsatz von Lastverteilungsstrategien verteilter Systeme in der Videoverteilung, KOM Technical Report, TR-KOM-1998-07

[CZ98] Data Base Design Principles for Striping and Placement of Delay-Sensitive Data on Disks, PODS’98, June 1998

[RRWA98] Richards, Rogers, Witana, Antoniades, “Mapping User Level QoS from a Single Parameter”, MMNS’98, Toronto, Nov. 1998