Multimedia Applications

Needed for Multimedia:
- Encoding of multimedia data
- Protocols for multimedia communication, network and system resources, synchronization
- Data storage on hard disk, CD, databases

Now: how to design and work with multimedia applications?
- Multimedia documents: How to describe, structure and access documents containing several media in an appropriate way?
- Design of multimedia system interfaces: How to design good interfaces for a human user to access the applications?
- Programming: How to ease the way of constructing multimedia applications?

First: multimedia documents

Multimedia Documents

Exchanging documents requires that the document architecture is known. This requires a definition of an architecture:
- Content: multi-/mono-media information
- Structure: spatial and temporal relations between information
- Manipulation Model: definition of operations for creation, change and deletion of information
- Representation Model: exchange protocol and data format
- Presentation Model: rules for document presentation

A multimedia document then is a document which comprises at least one continuous (time-dependent) medium and one discrete (time-independent) medium.

Mono-media Document Example: Hypertext

Hypertext System
- A system that allows to read several related textual documents nonlinearly

Node-Link Hypertext Model:
- Individually chunks of (textual) information (also called nodes) are set in relation by means of hyperlinks
- Hyperlink: relation between two or more nodes, defining a structure
- Hypertext Document: distinct set of nodes and links which constitutes a logical entity
- Hyperweb: distinct set of hyperlinks
Multimedia Documents: Hypermedia

- Hypermedia is the generalization of hypertext to include additional media like graphics, photos, audio clips, video sequences, animations.
- Synchronization and linking of these media elements to other elements must be possible.
- Hypermedia systems allow interactive, integrated, and synchronized presentation of multimedia information.

Hypermedia

3 Layers of different functionalities:

**Presentation and user interface**
Based on a given structure and user’s display, it is decided:
- which data to present
- how the data are presented

**Hypermedia abstract machine**
- Determine the structure of the document
- Knowledge about references, data structures, attributes

**Database**
- Storage of data as objects without semantics or structure definitions
- Storage management: consistency for multi-user access, persistency, fault tolerance, …

Description of hypermedia documents is possible e.g. with SGML, HTML/XML, or MHEG.

Standard Generalized Markup Language (SGML)

SGML was evolved from an IBM internal project, and was strongly supported by American publishers:
- Authors define titles, tables, etc. inside a document in a uniform way, without any description of the actual representation.
- Publisher determines layout.

Basic ideas:
- Author uses tags (markups) to mark parts of the text to be e.g. a title or a table.
- SGML determines how tags have to look like.
- User groups agree on the meaning of the tags.
- Formatter generates document layout from tags.

SGML defines a syntax, not semantics!

Example:
```xml
<title>Multimedia Systems</title>
<author>Otto Spaniol</author>
<site>RWTH Aachen</site>
<summary>In multimedia systems, …</summary>
```
**SGML: Tag Categories**

**Descriptive Tags:**
- Define the structure of the document in the form `<tag> text </tag>`

**Entity Reference:**
- For symbol substitution and file, data set, and variable embedding; the actual content can be inserted later at the corresponding place
- Example: `&Ae.rger ... means ... Ärger`

**Processing instructions:**
- Instructions used by other programs, e.g. a formatter
- E.g. for the embedding of different media

---

**SGML and Multimedia**

**Embedding of multimedia data:**
- As external document parts from separate files
- Originally embedded as CGM (Computer Graphics Metafile) graphic
  - Concrete data referred by using NDATA
  - Example:
    ```
    <!ATTLIST video id ID #IMPLIED>
    <!ATTLIST video synch #IMPLIED>
    <!ELEMENT video (audio, movpic)>  
    <!ELEMENT audio (#NDATA)> -- non-text media
    <!ELEMENT movpic (#NDATA)> -- non-text media
    ...
    <!ELEMENT story (preamble, body, postamble)>  
    ```
- Used now: HTML

---

**SGML Conformant Languages**

**HTML**
- HyperText Markup Language
- For description of hypertext / hypermedia in the World Wide Web (WWW)
- Most used SGML-language today

**HyTime**
- ISO standard for structured presentation of hypermedia information

**SMDL**
- Standard Music Description Language

**XML**
- eXtensible Markup Language
- Designed to enable the use of SGML on the WWW
- “Meta language”: lets you design your own markup language
- Conforms better to SGML as HTML does – “format of the future”
The World Wide Web (WWW)

- **WWW**: Internet-wide distributed hypermedia information retrieval system, provides access to large universe of documents
- **Client-server model**:
  - Clients: (Web browsers) collection of programs which can send requests for documents to WWW servers
  - Servers: programs that, after they receive requests, send back the appropriate answers - either documents or error messages
- **Major WWW standards**:
  - HTTP: HyperText Transfer Protocol (transmission of documents)
  - URL: Uniform Resource Locator (address of document)
  - HTML: HyperText Markup Language (document format)

**Proxy/Cache Approaches**

- **Autonomous Caching**
  - Each proxy itself decides whether to cache or to displace a document
  - No additional management overhead

- **Hierarchical Caching**
  - Information about requests is forwarded "upwards"
  - Central management

- **Cooperative Caching**
  - Groups of proxies may exchange documents
  - Even documents that are retrieved very often don't need to be stored everywhere (allows for better scalability)
  - Group management needed

**Hypertext in the WWW**

- **Nodes**
  - Hypertext documents represented by HyperText Markup Language (HTML) or arbitrary information objects referenced by means of an URL
- **Hyperlinks**
  - Embedded in HTML documents, directional point-to-point
- **Source anchors**
  - Text portions or images regions included in HTML documents
- **Destination anchors**
  - Either information objects addressable through an URL or specific parts of HTML documents
- **Link type**
  - Always a reference for further reading
  - Further information in addition to the pure link between two nodes
  - Example: differentiation between different kinds of destination nodes
**Hypertext Markup Language (HTML)**

HTML is a SGML Document Type Definition (DTD), giving syntax and semantics of hypertext documents by defining tags for:
- describing the text structure (not the layout)
- creating hyperlinks
- embedding objects like graphics
- describing tables
- ...

**Coding**
- HTML Documents are usually stored as text files

**Browsers**
- Can understand tags
- Can convert them into page layout

**Document: Internal Representation**

Some HTML tags:
- `<HEAD>...</HEAD>`: Page header
- `<B>...</B>`: Bold text
- `<P>`: New paragraph
- `<IMG SRC="...">`: Inclusion of picture
- `<A HREF="...">...<A>`: Link to another document

**Example:**
```
<HEAD>My Page</HEAD>
<BODY>
  This is my own web page.
  <B>Ain't it nice?</B>
  <P>Here's my picture: <IMG SRC="myself.jpg">
  <A HREF="http://www.me.info/myself.html">Here</A>
  you can find further information about me.
</BODY>
```

**Forms and CGI (Common Gateway Interface)**

Forms are used to convey information from client to server:

```
Please send me more information!
Name: [Input box for name]
Street: [Input box for street]
City: [Input box for city]  [Send]
```

HTML representation:
```
...<FORM ACTION="http://www.info.com/cgi-bin/order" METHOD=POST>
  <P>Please send me more information!</P>
  <P>Name <INPUT NAME="customer" SIZE=30></P>
  <P>Street <INPUT NAME="street" SIZE=30></P>
  <P>City <INPUT NAME="city" SIZE=30></P>
  <INPUT TYPE=SUBMIT VALUE="Send!">
</FORM>...
```
Forms and CGI

The `<FORM>` tag specifies
- Transmission method
  - GET: get document from URL
  - POST: in this case, a document is sent to the server, e.g.,
    `customer=Bill+Clinton&street=The+White+House&city=Washington`
- Action to be taken by server, e.g., execution of a CGI script

**CGI: Common Gateway Interface**
- CGI script: a program which is
  - executed at the server
  - on behalf of clients’ request (as defined by transmitted parameters)
  - e.g., a database query
- Output of the CGI script is returned to client, e.g., newly generated WWW page
- Addressing of scripts via URLs, usually in directory `cgi-bin`

Enhanced Client-Server Interaction

**Problem with CGI scripts:** client interactions are limited - programs are executed on the server

**Solution:** execute scripts/programs on client side

**Main idea**
- URL on a page points to small program
- Program is called applet
- When clicked: browser downloads applet to the client and executes it there

**Advantages:**
- WWW pages become more interactive
- Browser capabilities can be extended on-the-fly
- E.g., loading of new viewers when needed

Java: Fundamentals

**Java:**
- Programming language to write applets
- Originated from Sun Microsystems

**Applet Lifecycle**

```
import java.applet.*;
public class myApplet extends Applet {
    public void init() {
        ... // called on first startup
    }
    public void start() {
        ... // called when page entered (again)
    }
    public void stop() {
        ... // called when page left
    }
    public void destroy() {
        ... // called when browser/runtime environment is finally left
    }
    public void myMethod() {
        ... // do something
    }
}
```

Inclusion of applets into HTML documents by new tag `<APPLET>`
- E.g., `<APPLET CODE=game.class WIDTH=100 HEIGHT=200></APPLET>`
Chapter 5.1: Documents and Hypermedia

The World Wide Web – Problems

The WWW with HTML in combination with scripts and applets serves as a basis for presenting multimedia content. But:

- Users can get “lost in hyperspace” because documents are arbitrarily structured, no overview is possible
- Difficult to find information (search engines have scalability problems)
- No version control of documents (detection of outdated information)
- No support for document authoring
- No link consistency: dangling links are created by the deletion of one link end
- Low performance and high network load (solution: proxy servers)
- No linking between dynamic media possible (no real hypermedia)
- ... much more

Multimedia and Hypermedia Information Coding

Expert Group (MHEG)

MHEG is an “umbrella standard” for multimedia objects. It provides a language for control of delivery of multimedia objects:

- Exchange format and representation for composed (time-dependent) multimedia information
- Provides a structures for interactive, spatial and temporal related information
- Suitable for real-time (distributed) multimedia/hypermedia applications (no complex parsing and interpretation required)
- Platform independent description
- Typical application: devices with small resources like point-of-sales terminals, video on demand set-top boxes (MHEG is e.g. used in DVB), ...

MHEG Family

Outdated standards:

- MHEG-1: MHEG Object Representation, Base Notation (ASN.1)
- MHEG-2: MHEG Alternate Notation (SGML)
- MHEG-3: MHEG Extension for Scripting Language Support
- MHEG-4: Registration Procedures for Format Identifiers

Current standards:

- MHEG-5: Support for Base-Level Interactive Applications
- MHEG-6: Support for Enhanced Interactive Applications: JavaVM
- MHEG-7: Conformance Testing
- MHEG-8: XML

MHEG-5: Objects, Events, Links

Multimedia applications can be described using MHEG as collection of scenes. Both contain objects and links; on the happening of certain events for an object (mouse movement, clicks, ...) the effects of the corresponding link are processed.
MHEG Objects

MHEG standard: data structures and binary representation of MHEG objects (not: retrieval protocol or MHEG Engine)

MHEG pipeline:
- Information input (e.g. web of hypermedia documents)
- Interactive presentation

MHEG Objects Classes

- MH Object
- Component
- Macro
- Behavior
- Descriptor
- Action
- Link
- Script
- Content
- Interaction
- Composite

Instances can be created from all leaves of this MHEG class hierarchy tree. Instances cannot be created from internal nodes of the tree.

MHEG Objects Examples

Complex MHEG Objects include several other MHEG objects that determine:
- the content (output objects)
- the user interaction (input objects)
- the presentation (action objects)
- the relations between the different objects (link objects)

MHEG Presentable Object Example

Structural information included in MHEG objects allows to determine the presentation of the object:

Presentation as described by the included set of link, action and script objects.

MHEG Objects may be distributed.
MHEG Scheduling of Objects

MHEG supports real-time hypermedia presentations by allowing to add time dependent object preparation and presentation attributes (time stamps)

<table>
<thead>
<tr>
<th>Sec.</th>
<th>0</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Audio</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text A</td>
<td>prepare text A</td>
<td>prepare text B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text B</td>
<td></td>
<td>timestamps and actions stored in the MHEG-Object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>prepare audio</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The PREPARE action triggers the retrieval and initialization of the MHEG object.
Problem: delays cannot be exactly predicted by the MHEG engine.

Conclusions

Multimedia documents can be represented using several techniques
- **SGML** provides a description language for “everything”. But: only a theoretical construct
- **HTML** as reduced version of SGML for hypertext in the WWW
  - Different media can be linked in a document
  - Scripts and applets allow for dynamic content
  - But: some problems, e.g. no linking of different media possible
- **MHEG** as description language for multimedia objects
  - Allows for composing complex objects and specifying time dependencies
  - Also gives an integration with Java
  - Newer standards allow for the usage of XML for description