Software inventarisation for DYNAMO/RTIPA

Mark van Doorn

October 21, 1999

Abstract

This document gives a short overview of the available architectures, software components and documentation currently present at CWI, TUE and Philips Research relevant to the DYNAMO and RTIPA projects. The similarities and differences between the software architectures and components are summed up in the final section.

1 CWI

The Multimedia and Human-Computer Interaction group at CWI has developed an authoring system for hypermedia called GRiNS and invented the Amsterdam Hypermedia Model [5]. It is also heavily involved in the W3C workgroup SYMM which designs SMIL, a specification language based on XML for synchronized, interactive multimedia presentations on the internet [12]. The Standard Reference Model (SRM) for hypermedia suggested by Maybury et al. [3] is also investigated as starting point for the design and implementation of a system for semi-automatic generation of hypermedia presentations.

1.1 GRiNS

The GRiNS system contains an editor and player for hypermedia presentations and runs on a variety of platforms (Windows 95/98/NT, Unix (SGI IRIX 6.5 and Sun Solaris 2.x) and Macintosh). GRiNS 1.0

is now made commercially available by Oratrix[1]. The editor graphically displays the structure of a SMIL document as a tree of nodes and can also give an overview of the relations (i.e. spatial, time and link) between the different content items (channels). The internal format of a document is represented in the CMIF format which has some similarities with SMIL but needs to be translated to SMIL before it can be displayed in the SMIL player. The player features (besides a capability for displaying SMIL documents) a scheduler which detects network congestion and can provide debug information during the presentation. GRiNS has been implemented in Python, the scripting language developed at CWI and uses standard C libraries for multimedia processing. The GUI uses standard, platform dependent software libraries.

At the moment an XML-DOM interface is being build on top of the GRiNS-editor. The choice for an XML-DOM interface is twofold: First, the internal representation of a document within GRiNS (CMIF) is like the XML-DOM tree-structured and secondly, being compliant with the uniform standard XML-DOM interface makes it easier to maintain and extend in the future. In order to use the XML-DOM API on top of GRiNS in many applications (as an implementation of the presentation layer in the SRM model for example) and across different platforms, CORBA is being used: The GRiNS server can be accessed by software components that need this functionality through a common, generic interface specified in IDL.

1.2 Partial SRM implentation

Some prototype work has already been created for the realization and design layers of the Standard Reference Model for Intelligent Multimedia Presentations. Basically, the realization layer first parses the design plan, an instance of a realization DTD that captures all of the necessary constraints, using IBM's XML4J parser and creates an in-memory representation of the document tree. The Java Cassowary constraint solver solves the temporal and spatial constraints, while the links are further processed for quick retrieval during output generation. These results are passed to a SMIL (or HTML) generator which maps the elements, attributes and relationships from the realization DTD to the corresponding entities in SMIL. The SMIL document can

[1] http://www.oratrix.com

then be displayed in a SMIL player like the GRiNS player (presentation layer of the SRM). Preliminary work on the design and content layer has resulted in some design and domain semantic rules in Prolog. This work is described in more detail by Brian Bailey in [1] and can be found on the DYNAMO website[2]

1.3 Documentation

Information about the GRiNS authoring environment and player can be found on http://www.oratrix.com. The Amsterdam Hypermedia model is described in [5]. The SMIL 1.0 and SMIL "Boston" specifications can be found on the W3C website [12]. The Standard Reference Model for generating adaptive hypermedia presentations is explained in [3] and an example implementation of the realization and presentation layers is described in [1].

2 TUE

2.1 Overview of work

The information systems group at the Technical University of Eindhoven research is concerned with both the generation of hypermedia from volatile data, i.e. data for which no design has explicitly been defined such as ad-hoc user queries [7], and the presentation of this hypermedia according to specific user and domain models and so-called "teaching" models that specify how the information from the domain model is influenced by the user model [2]. Currently, the generation and presentation stages are being combined and examined in more detail to gain more insight in the processes involved in the overall process. The software that is available is described in the next section.

2.2 Software Architectures

2.2.1 AHA

The AHA model has been implemented using Java servlets that extend the functionality of the webserver, i.e. implement the adaptive

[2] http://www.cwi.nl/multimedia/Dynamo/

engine. The user and domain models are specified in XML and corresponding DTD documents. The AHA system consists of three models that work together for the presentation of hypertext documents: The domain model describes concepts, relations between concepts and dependencies (like prerequisites and inhibitors), the user model describes the knowledge the specific user has about the domain and is updated each time the user interacts with the presentation. The teaching model describes how the domain knowledge influences the user model.

Content-adaptation and link adaptation such as high-lighting and hiding are performed by adding conditional text to an HTML document as comment. This additional information provides the pedagogical rules of the teaching model. How the system deals with these rules has been implemented in the adaptive engine. The document is translated to XML and the adaptive engine reads this XML document and uses the domain and user models (the specific user can be identified using a login screen) to create an HTML page that is best suited for the user's current knowledge of the domain according to the rules put in the document by the author. The HTML page is then send back to the user and the user model is updated and written back to disk. This is done each time the user accesses and leaves the page but the overall performance is quite acceptable for a single user that uses a local webserver.

2.2.2 Volatile database output

The information contained in multimedia databases is typically less structured than the information in traditional administrative databases. Hypermedia can help to represent the output of such databases and improve the overall quality of the presentation. A methodology for designing such structured hypermedia applications is the Relationship Management Methodology [9]. RMM helps in designing presentations for (whole) information objects contained in a database and in generating navigational structures such as indices and guided tours. The methodology is best suited for applications that have a regular structure that is frequently changed. However, a significant portion of database output is volatile due to the nature of ad-hoc queries and cannot easily be translated into a hypermedia presentation because of the dynamically generated structure of such queries. So RMM either needs to be dropped or extended to handle volatile data efficiently.

The basic idea about the extension of this model is that the user is offered a predefined, default representation deduced from the data dictionary in the RMDM representation that is adjusted in correspondence with the number of items found in the result. A heuristic algorithm is used for selecting the best slice [3] from a set of predefined slices. A slice definition contains information about the layout class [4] and the parameters of this layout class and describes which attribute fields are used and how these are ordered. The user can override this default representation (the attributes in a slice and the navigational structure between relations for instance) for efficient use of the query result in her work.

The language used for data manipulation is an extension of SQL: It offers, besides data retrieval, the user the possibility to add representation information (i.e. navigation and presentation) inside a query. A preprocessor interprets this SQL-like syntax and creates a SQL query that is send to a JDBC/ODBC complaint database to retrieve the data. After the data has been retrieved, a database servlet creates an XML document from this data that is passed on to the presentation servlet. The preprocessor also creates and adjusts the default representation deduced from the data dictionary according to the information provided by the user in the extended SQL query. This representation is then used by the presentation servlet to generate a HTML document which is then send back to the user and displayed in her browser. The implementation does not support multimedia at the moment and presentation is limited to the layout of fields and hyperlinks. The navigation options currently available are: index, guided-tour or index-guided tour.

2.3 Documentation

For more information on the generation of hypermedia see [8, 7] and the master thesis of Erik de Kort. RMM is explained in [9]. AHAM is described in for example [2]. These and other publications can be found on [6].

[3] Group of one or more attributes of an entity in RMM

[4] The layout managers in the Java AWT API are used

3 Philips Research

3.1 WebEPG

The Streaming Internet and Media cluster (SIM) of the New Media Systems & Applications group of Philips Research has developed and implemented a framework for building Electronic Programme Guides (EPGs) [5] using Web technology. The design consists of three separate, but interconnected parts: (1) a (distributed) database containing meta data about, among perhaps many other things, the content of TV programmes, (2) the EPG-document layer which extracts information from this database and generates documents and relations between these documents that together describe the semantics of an EPG and finally (3) the EPG-presentation layer which is concerned with the "look-and-feel" of the EPG documents by adding layout and style information. The presentation can be customized to specific user profiles and machine capabilities. The work has been concentrated on rendering EPG documents on different machines for fixed user profiles (interaction between the document and presentation layer). The software that is available will be describe next. Furthermore, the cluster is developing a SMIL player written entirely in Java.

3.2 Software architecture

The structure or semantics of an EPG is described in a Document Type Definition (DTD) document. The EPG itself is described as an XML document that satisfies this DTD. The transformation of DTD's and presentation of XML documents in a particular format (HTML,WML,SMIL etc.) is done via XSL stylesheets and the Document Object Model (DOM) API. To parse and validate the EPG a parser is needed. Next an XSL processor can be used to render it according to a particular XSL stylesheet.

3.2.1 Current WebEPG

All processing is done at the server side. The Microsoft Internet Information Server (IIS) / Personal Webserver (PWS) is used as the default HTTP server. Microsoft Active Server Pages (ASP) extend the functionality of the webserver through the VBScript scripting language.

[5] But other domains would also easily fit in this framework

This scripting language in combination with MSXML components is used for XML document handling, such as parsing and XSL processing. This combination uses static XSL stylesheets for different platforms and user profiles that are selected at run-time through HTTP-requests from the client. Unfortunately, this results in an enormous amount of XSL stylesheets for each possible combination of profile and platform.

XSL stylesheets, XML documents and corresponding DTD are available for an example EPG as well as active server pages to invoke the right MSXML components for XML document handling.

3.2.2 Improved WebEPG

In the improved version servlets and Java take over the role of ASP and VBScript. Furthermore, each webserver that supports servlets can be used. XML parsing is done with IBM's XML4J parser and XSL processing with IBM's Lotus XSL processor. User profiles and platform capabilities are passed on as XML documents to the servlet in a HTTP request. The servlet uses the XML parser, XSL processor and the DOM API which offers support for traversing and manipulating XML documents. Through the DOM API, XSL stylesheets can be adapted at run-time resulting in less and dynamic XSL stylesheets.

Generic XSL stylesheets, an XML EPG document, some sample user profiles and corresponding DTDs are available. The servlet that calls the IBM XML parser and Lotus XSL processor and modifies the generic XSL stylesheets to match the user's profile and machine capabilities has also been implemented.

3.3 Documentation

For more information see technical reports [4] and [11] and the many pointers for XML, XSL and DOM on the internet ([10] gives a good overview).

4 Conclusions

The projects in the different groups show some overlap but are complementary to a large extent. Most of the work done, could be of use in the DYNAMO project. Little work has been done with regard to adaptation to network resources in the context of the RTIPA project.

The WebEPG framework shows the overall process of information retrieval, extraction and presentation of hypermedia. Generation of hypermedia content from volatile database output (information extraction and adding structure to the retrieved multimedia information) is addressed by the TUE. Presentation of this hypermedia according to simple user profiles, knowledge is done by both Philips and the TUE. The TUE also includes the intentions of the author in the presentation process (teaching model). CWI has an authoring environment for multimedia and mostly addressed effective editing and efficient runtime support for hypermedia and started some prototype work on the implementation of the SRM model which is less abstract than the WebEPG framework and focusses more on the presentation than on the extraction and retrieval part.

All groups use XML and DTD documents to describe structured information to some degree. Philips uses XSL style sheets to do XML translations and presentation while the TUE and CWI do this in software. CWI uses SMIL as the format for representing hypermedia while the TUE and Philips groups generate HTML presentations (no hypermedia at this moment). Furthermore, the WebEPG and the TUE systems are implemented as servlets on a webserver (server side processing) while the GRiNS system is a stand-alone application on the client-side which replaces the browser.

Much work still needs to be done. Some possible questions might be (this list is by no means complete):

- How is knowledge represented and how is it used in the different layers and agents to assist in hypermedia presentation?
- What kind of adaptation is required?
- To what features should the system adapt?
- How does the user give feedback to such a system in an unobtrusive way?
- Network adaptation?
- What should the different layers and agents of the SRM model do in more detail (in case the SRM model is chosen)?
- How are tasks and resources allocated between project members?
- Is the software written by the different project partners available to all the other project members?

References

[1] Brain Bailey. Automatic generation of electronic program guides.

[2] Paul De Bra, Geert-Jan Houben, and Hongjing Wu. AHAM: a Dexter-based reference model for adaptive hypermedia. In *Conference on Hypertext and hypermedia*, pages 147–156, Darmstadt, Germany, Februari 1999.

[3] Mark T. Maybury et al. A Standard Reference Model for intelligent multimedia presentation systems. *Computer Standards & Interfaces*, 18(6-7):477–496, December 1997.

[4] Johan Gielen. WebEPG; applying XML, XSL and DOM for tailored information exchange via the Web. Technical Report 220/99, Philips Research, August 1999.

[5] Lynda Hardman, Dick Bulterman, and Guido van Rossum. The Amsterdam Hypermedia Model: Adding time and context to the Dexter model. *Communications of the ACM*, 37(2):50–62, Februari 1994.

[6] Geert-Jan Houben. Adaptive hypermedia links:. http://wwwis.win.tue.nl/˜houben/pub.

[7] Geert-Jan Houben and Paul de Bra. WWW presentations for volatile hypermedia database output. In *WebNet97, the World Conference of the WWW, Internet and Intranet*, pages 229–234, Toronto, 1997. AACE.

[8] Geert-Jan Houben and Paul de Bra. Retrieval of volatile database output through hypermedia applications. In *Hawaii International Conference on System Sciences*, Maui, Hawaii, Januari 1999.

[9] Tomas Isakowitz, Edward Stohr, and P. Balasubramanian. RMM: A methodology for structured hypermedia design. *Communications of the ACM*, 38(8):34–44, August 1995.

[10] Mark Johnson. XML for the absolute beginner: A guided tour from HTML to processing XML with Java, April 1999. See also: http://www.javaworld.com/javaworld/jw-04-1999/jw-04-xml-p.html.

[11] Warner ten Kate. WebEPG for use in enhanced broadcast. Technical Report 256/98, Philips Research, August 1998.

[12] The SMIL specification:. http://www.w3.org/AudioVideo/.