# A Browser User Interface for Digital Television

**Juha Vierinen and Petri Vuorimaa**

Telecommunication Software and Multimedia Laboratory,
Helsinki University of Technology,
P.O. Box 5400, FI-02015 HUT, Finland.
jvierine@cc.hut.fi and Petri.Vuorimaa@hut.fi

**ABSTRACT**

This paper discusses a process of designing and implementing a graphical user interface (GUI) for an XML browser. The process consists of four steps: a) a concept of a multimedia browser for television is defined; b) the GUI requirements are defined; c) a prototype is designed and tested with multimedia authoring tools; and d) finally, the prototype is implemented, which is done in Java, and integrated with an existing XML browser. The result is a browser application that can be run on digital television.

**Keywords:** Java, XML, digital television, user interface design

## 1 Introduction

Until now, most of the software applications have been developed for desktop devices with keyboards, mice, and monitors as standard input and output devices. Recent developments have introduced a need for developing and converting software for different types of devices. Examples of these devices are PDAs, mobile phones, and digital television set-top-boxes. These devices expand computing to new environments where traditional desktop computers do not go.

In this paper, we seek a solution for displaying media-rich content for the emerging digital television broadcasting needs. Media-rich content could be super teletext, web browsing, or commercials. In an ideal situation, all this content can be distributed in the same format and viewed with the same application.

Extensible Markup Language [Bray98] (XML) is a structural document description language that is independent of its presentation. "XML is the key enabling technology for the next generation of data-intensive enterprise applications on the Web." [Zurek97] Being independent of presentation, XML is especially useful for delivering the same content for different devices.

Previously, XML has been used in digital television for a specific super teletext Java application [Vuori00a]. In the paper, they concluded that "XML can be used in digital television text TV services". They also predicted that "In the future, set-top-boxes will very likely have their own browsers, which can even be XML based."

## 2 Aim of the Study

Nowadays a web browser is an important channel for information in desktop computers. Some media-rich content that is distributed through web-sites already competes with TV. It is very propable, that this kind of content, e.g., short movies and music, will also be viewed with a TV in the future. For this reason, we wanted to see how a browser could be fitted in digital TV. We tried to follow the future digital televion standards.

We used Java as the implementation language. It was used, because the standards of many future multimedia devices promise support for Java. Set-top-boxes are not an exception; the Digital Video Broadcasting (DVB) specification states that European set-top-boxes will have a Java Virtual Machine [DVB00]. The DVB has defined a framework called the Multimedia Home Platform (MHP), which defines a set of classes that can be used to make applications for set-top-boxes.

We had access to a Java based XML browser called X-Smiles [1] [Vuori00b]. It has been developed in the GO-MM project at Helsinki University of Technology. The browser is intended for different devices supporting Java. It has a modular GUI part, which makes it possible to bundle different GUIs, for different devices, using the same core module. The digital TV GUI was the first device specific user interface that was designed for it. Apart from designing this one GUI, we were also developing a process, that could possibly be used to design browser GUIs for other multimedia devices.

The process we used in developing the user interface is quite a common one. It is a fuzzy set of steps to create a product, iterating from the concept to the final product. A Similar process can also be found in many user interface design related books:

**Step 1.** The first step is to define the concept of the product.

**Step 2.** The requirements and contraints are listed based on the concept, user interviews, case scenarios, and the environment in which the product is to be used.

**Step 3.** A GUI is designed iteratively. A prototype is designed and evaluated for usability aspects, system constraints, and the requirements defined earlier.

**Step 4.** The product is implemented. In an object-oriented world, this is also an iterative process. The usual steps are analysis, implementation, and testing.

## 3 Concept and Requirements

### 3.1 Concept

As said earlier, the idea was to make a XML browser for TV use. The concept of a browser for television is already familiar to many people. There are several solutions for web browsing with a TV (e.g., OpenTV Device Mosaic [OpenT00]), but they do not take full advantage of the new emerging XML standards.

Anybody using television in the future will be a potential user of a browser. This implies that users will have different levels of technical skills. Because of this, we tried to make the GUI more simplified, compared to normal desktop browsers, still maintaining as much similarity with existing browsers.

[1] The X-Smiles browser is available as open source at www.x-smiles.org

### 3.2 Constraints

The NorDig specification [NorDi00] is a standard for future Scandinavian digital TV devices. It defines the minimum capabilities of input and output devices. They were the basis of our system specific constraints.

The input device can be an infrared keyboard, but not mandatory. Usually, it will be a remote control. The most important buttons that can be used by applications are the four color coded buttons, the arrow buttons, and an ok.

Because the resolution of TV is not the best possible (minimum of 720x576 according to the NorDig specifications), the size of the fonts has to be large enough. Otherwise, the onscreen text will be hard or even impossible to read.

The TV screen is also different than a normal computer monitor. The display is viewed from a much longer distance, it flickers, and the color depth is less. There are some color combinations which enhance clarity and there are also some which dramatically reduce it [Darby97]. In designing the user interface, there should be clear on screen cues which resemble the buttons of the remote [DalyJ00].

The physical environment of TV is usually the living room. A digital TV can be thought to be more of an entertainment center, than a tool. A study on television usage behaviors conducted by Logan et al. [Logan95] showed, that "TV can provide a forum within the home for people to sit down together and share daily experiences." There results imply that operating a TV browser should not use too much mental resources, as the user might be doing something else at the same time.

### 3.3 Requirements

We used several common techniques to come up with the requirements. These included mapping of the user groups, their needs, goals, and the physical environment of use. We also wrote imaginative scenarios, studied existing solutions, and researched the terminology related with document browsing.

To keep our browser simular with existing solutions, we incorporated existing needs and goals from users of teletext and desktop web browsers to our requirements. These included *Go Backward/Forward, Follow link,* and *Type in URL*. We also chose to incorporate exising terminology for browsing.

The tasks related with information retrieval and browsing are broad. One may want to read the news, watch a short movie, or maybe do some on-line shopping. By writing down these kinds of imaginative scenarious and analysing them, we tested the requirements. In the process we found more requirements, such as the *Feedback*, that is described later.

### 3.4 Use Cases

We listed the requirements and gave each one a name. In an object oriented terminology, they are called use cases [Fowle97]. Every use case defines a one or more requirements for the GUI. The purpose of the rather unformal use cases was to give a direction to follow in designing and testing the user interface.

The most important requirements from the user point of view are: *Accessibility*, *Bookmarks*, and *Feedback*. Because text input is difficult, there has to be some kind of a portal, where the user can start off browsing. Managing and adding bookmarks is also important, since following a

link is much more convenient than entering a long hard URL with difficult text input methods. The following use cases are listed in a supposed priority order:

**Go backward/forward** There must be some way to navigate backward and forward between pages. This is one of the most basic idea of web browsing. The *backward/forward* functionality must be easy to access, since it is used very often.

**Feedback** As the TV is more distant to the user than a computer, there has to be more cues, which inform the user and give feedback on what the browser is doing.

**Follow link** XML offers a way to hyperlink documents. Basically *follow link* will mean the traditional web-based follow link functionality, even though XML offers more complex ways of linking documents. Follow link is also an often used task.

**Status** The user should know of which state the browser is in. In addition to this, the user should also be given a title or URL of the page that is currently open.

**Scroll up/down** There must be some way to scroll content, if it doesn't fit the screen. One possibility would also be, to require the content to fit into the screen, thus avoiding annoying scrolling of content.

**Accessibility** There has to be some home page, which offers access to the most often browsed pages. It can be a portal combined with a bookmarking system.

**Bookmarks** Because the TV environment offers only a poor URL input possibilities, there must be an efficient bookmarking system. It should also be possible to create and manage sections. This can be achieved with a simple local document which all the bookmarked URLs.

**Type in URL** There must be some way to manually type in an address for the browser, even though it will not be used so often in the TV environment.

**Exit browser** There must be an easy way out—there must also be a possiblity to keep the browser in the background and toggle between the TV picture and the browser.

**Transparency** There should be a possibility to cover only part of the TV screen with the browser window.

## 4 Design Phase

### 4.1 Methods

The actual prototype development began with drawing different imaginative views of the browser. We tried different components that were TV friendly and could also accomplish the different use cases. After exploring different solutions with paper prototypes, we came up with a design that worked together and would be fairly easy to use. We also made a partly interactive version of the final prototype and did some usability testing with it.

The tools that were used in the prototyping phase were typical multimedia authoring tools. The components and the prototype were designed with a vector graphics tool (i.e., Adobe Illustrator). It offers good scalability of drawn components. The layering also makes it easy to try out different views of the GUI. Macromedia Director was used to create the interactive prototype.

The usability tests that we conducted in the design phase on the prototype were small. Usually, one or two people giving their comments to explicit questions, such as: "Can you read the text here?" or "How would you add this page to the bookmarks?". The main idea was to test whether the user interface could be used in the way that it was defined. In the beginning, prototypes were tested on paper. More thorough usability tests were done on the final interactive prototype with a TV connected to a PC and a remote control.

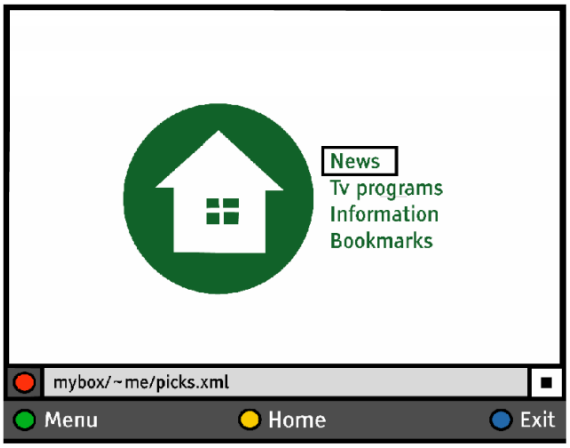


Figure 1: The XML/Java browser user interface. The content area in the middle represents a TV portal. The lower area is the browser frame.

We came up with a design that is shown in Figure 1. The browser resembles a normal browser in some sense. It has similar components, but the graphics are more simplified and there is less information on the screen.

### 4.2 Functionality

The remote control buttons that are used for navigation are the arrow buttons, the color buttons, and the ok button. The arrow buttons left and right go back and forward; up and down moves the highlight to the next link or page. The ok button follows a link, or activates controls of multimedia components, such as play or stop. The color buttons open either a menu, go home, or exit the browser. There are four color button cues telling which function the button on the remote will invoke. The text field serves as a multipurpose statusbar, telling what the browser is doing and which page is currently open. On the right side of the statusbar, there is a little box, which animates when the browser is loading some page. The red and green button both bring up a menu, from which the other seldomly used functions can be accessed. The colors of the cues are in the following order: red and green for the menus, yellow for home and blue for exit. All the components are shown in Figure 2. The numbers correspond to the following list:

1. **Main Menu** Arrow keys scroll up and down. OK key activates the selection.

2. **Highlight** A surrounding rectangle informs the user, which link or functionality is active. The arrow keys up and down change it's position. This is similar to the textbased browser Lynx. The ok button follows a link or activates the seleection.

3. **Configuration Menu** This is similar to the main menu, but the user can also toggle the items in the menu.

4. **Content Area** All the documents will be rendered in this window. No scrollbars are used, the content is displayed page by page.

5. **Arrow** This component does not contain any functionality. An arrow flashes to visualize the Go Back and Go Forward functions.

6. **Animator** An animator is something that most browsers have. It visually

indicates that the browser is busy doing something.

7. **Statusbar** The text field functions a statusbar. It tells the user which page is open and what the browser is doing.

8. **Lower Bar** The colorfill circle combined with a label indicates which functions are launched with the color buttons of the remote control. Similar components have been used in many applications designed for TV usage.

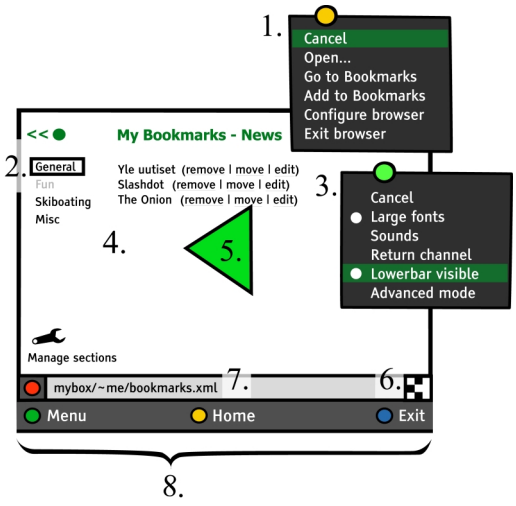


Figure 2: The components that were designed. Most components are already familiar from TV user interfaces. The numbers correspond to the numbers in the list of components.

## 5 Implementation

The browser, for which we designed the GUI, is called X-Smiles. It is written purely in Java and it supports the new emerging document description standards. It has support for Formatting Objects, Synchronized Multimedia Interchange Language, and Scalable Vector Graphics.

A simplistic model of the browser, shown in Figure 3, contains three essential classes: `Browser`, `UIBridge`, and `XSmilesUI`. `Browser` contains all of the core functionality of the browser. `UIBridge` is a handler, which delegates information between `Browser` and the GUI. `XSmilesUI` is an interface for the GUIs. The implementation is based on a design pattern called the *bridge pattern* which is intended especially for situations where a class (i.e., the GUI in this case) can be swapped on the fly [Gamma94].

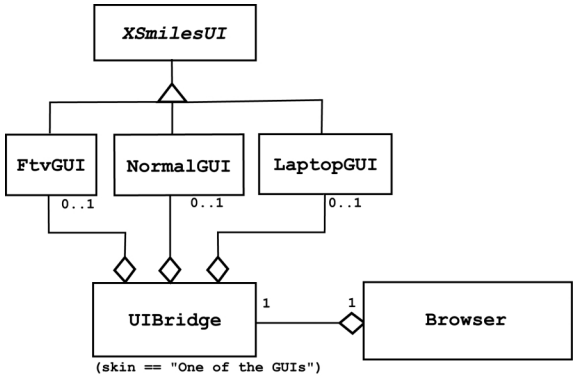


Figure 3: The most important classes related with the GUI. The UIBridge is the agent, which delegates information between the Browser and the GUI that is chosen.

To build the GUI, we needed a set of GUI components. The MHP specifies some user interface components for TV (i.e., MHP-HAVi components), but there were no known implementations or designs for them. The MHP states that "Alternatively, applications can derive custom widgets by subclassing the HAVi widgets, using the abstract widget framework, or by employing Java's *Lightweight User Interface Framework*." [DVB00] The Lightweight User Interface Framework offers a way to implement custom user interface components. In this case, this was what we did.

Custom lightweight components have to extend either the `Component` or `Container` class. The component has to be drawn using methods of the `Graphics` class. In most cases, it is also wise to have some sort of double buffering to avoid flickering.

## 6 Usability tests

Preliminary informal usability tests were conducted to find how user friendly the GUI was. The tests were carried out with 20-28 year old english speaking testees familiar with web browsing. The tasks presented to the testees, were based on the use cases listed in the requirements. In the tests, the subjects were able to perform most of the tasks, but there were also a few problems in the design and in the implementation.

One of the problems in the design was the use case "Typing in URL". We designed an input field, but it couldn't be used with the normal remote control. A mobile phone type of text insert could have been used, but we didn't implement it in the prototype. In any case, inserting text is a bit annoying.

## 7 Conclusion

The final digital TV browser is shown in Figure 4. First, the requirements were defined. Then the GUI was designed iteratively with multimedia authoring tools.

The Java lightweight component framework made it possible to create exactly the kind of user interface components designed. They were implemented with the basic Java drawing functions and had double buffering to avoid flickering.

During the study, knowledge was gained in general on making interfaces for exotic devices. With a similar process it would be possible to make browser user interfaces also for handheld or wearable computers.

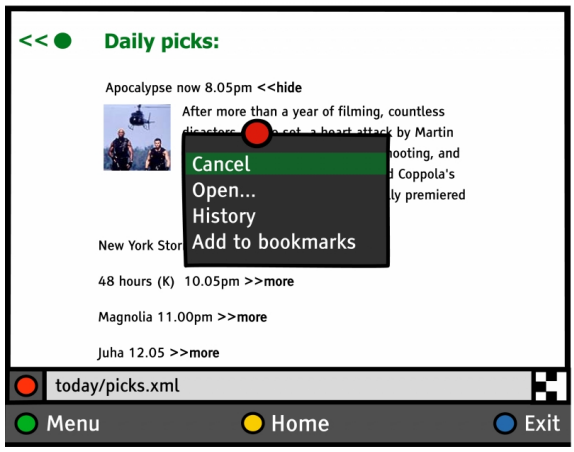


Figure 4: A Picture of the final browser GUI that was implemented. A prototype of an EPG is shown

## 8 Discussion

In this study, our goal was to make a prototype of an XML browser for TV. More research should be done to map possible uses of the browser. The future broadcasting stream will, for sure, include different kinds of documents. What kind of services can be offered with the browser? Different possibilities include commercials, interactive shopping, questionnaires, and web content. The browser and network connectivity also offers possibilities for totally new formats of TV programs.

Will the network connection be one-way? If not, how fast does the connection have to be, to ensure painless browsing?

We did not include a possibility for text input with a remote control, even though there are different possible ways to implement it. Which would be the best device for text input: a remote or a IR keyboard?

There are many questions still left open considering the use of a browser in the TV environment. One thing is for sure—there will be a browser in many appliances of the future, and TV is not an exception.

## REFERENCES


[Vuori00a] P. Vuorimaa and C. Sanchos: *XML Based Text TV,* Web Information Systems Engineering, WISE 2000, Hong Kong, June 19 - 20, 2000.

[Zurek97] B.Zurek, VP, Web Development Tools, Sybase Corp.

[Bray98] T.Bray, J.Paoli, and C.M.Sperberg-McQueen: *Extensible markup language (XML) 1.0, W3C Recommendation,* Feb. 1998. Available at http://www.w3.org/TR/REC-xml/

[Vuori00b] P.Vuorimaa and T.Ropponen: *X-Smiles XML Browser,* International Workshop on Networked Appliances, IWNA 2000, Brunswick, NJ, USA Nov 30 - Dec. 1, 2000.

[OpenT00] OpenTV Inc: *Device Mosaic 4.1.0 Browser* August 2000. Available at http://www.opentv.com-/docs/devicesmosiac410.pdf

[DVB00] DVB: *Multimedia Home Platform,* p 586. Feb 2000. Available at http://www.etsi.org

[Fowle97] M.Fowler and K.Scott: *UML Distilled,* Chapter 3, Use Cases, Addison-Wesley, 2000.

[NorDi00] NorDig: *The Nordig II Digital Integrated Reciever Decoder Draft Specification,* The remote control, p.53, Feb 2000. Available at http://www.nordig.org/specifications.html

[DalyJ00] O.Daly-Jones and R.Carey: *Interactive TV And Electronic Program Guides: Usability Guidelines,* p. 2, 2000. Available at http://-www.egroups.com/group/Usableitv

[Gamma94] E.Gamma, R.Helm, R.Johnson, and J.Vlissides: *Design Patterns: Elements of Reusable Software,* Addison-Wesley, Oct 1994.

[Logan95] R.J.Logan, S.Augaitis, R.H.Miller, and K.Wehmeyer: *Living Room Culture–An Antropological Study Of Television Usage Behaviors,* Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting 1995.

[Darby97] S.Darby: *Enchancing the Accessibility of Digital Television for Visually Impaired People, Visual Information,* May 1997. Available at http://www.rnib.org.uk-/wesupply/products/digitv.htm