Dynamo internal document (D-001)

Thursday April 6, 2000

WebEPG performance analysis

M.G.L.M. van Doorn

Philips Research, New Media Systems & Applications

mark.van.doorn@philips.com

Keywords: Personalization, customization, semi-automatic presentation generation, XML, XSL, DOM

Abstract: Presentation of information on the WWW adapted to different presentation environments is gaining importance as more and more consumer devices with Internet connectivity appear. Moreover, the presentation of this information must also be tailored to the personal user interests and preferences because of the huge amount of content available on the WWW and the fact that different users have different tastes. This document summarizes and evaluates the WebEPG framework, developed at Philips Research, that addresses these problems. The framework uses XML and XSL to exchange information in a generic and flexible way over the Internet. This way information can be tailored to different machine capabilities and user preferences. Within this framework, an application, the WebEPG, has been built that adapts the presentation of TV program information to different consumer access devices and user profiles. The TV program information itself is stored in XML and through dynamically modified XSL stylesheets that model the capabilities of the information appliance and the user preferences, this information is formatted in a Web presentation language format like HTML. This way the semantics of the information is separated from its presentation language syntax.

To test the performance of the WebEPG application, evaluation experiments for various XML EPG document sizes have been conducted on two machines with different amounts of internal memory. The evaluation measured XML parsing and XSL processing times related to document size for two different stylesheets.

Conclusions: The WebEPG framework and application in particular show that Web technologies can be used to present information on a wide range of consumer access devices, modeled as XSL stylesheets, tailored to different users, represented as XML profiles, in a generic and flexible way.

The results of the performance analysis test show that XML parsing and XSL processing require a lot of processing power and large amounts of memory. This indicates that while on one hand optimization of XML parsers and XSL processors may be possible and certainly needed, especially for use in consumer access devices, efficient storage and retrieval techniques of DBMS and IR systems are also required in order to speed-up the selection process that precedes the presentation. Another concern of the current framework is that the presentation personalization is domain dependent: A different stylesheet and user profile is needed for every domain. Both issues are under further investigation in the Dynamo project.

Contents

1 Introduction **1**

2 WebEPG Architecture **2**

3 Presentation generation **3**

3.1 Platform customization 3

3.2 User personalization 3

3.3 Program recommendation 5

4 WebEPG Implementation **5**

5 Experiments with the WebEPG **7**

6 Performance analysis **10**

6.1 Evaluation 12

6.2 Improvements 12

7 Future directions and related work **13**

8 Conclusions **14**

9 Acknowledgments **14**

1 Introduction

As the amount of information on the Web grows rapidly, effective and efficient information retrieval and extraction is becoming increasingly important. It is also important to present this information to the user in a natural way [23]. If computers and devices are to vanish in the background of our lives as is advocated in the ubiquitous computing [28] view, they must adapt to different users with different goals. Obviously, presentation of this information tailored to specific consumer devices and adapted to user profiles is an important issue for retrieval and presentation systems in such environments.

The Extensible Markup Language (XML) [4] is rapidly emerging as the universal format for structured documents and data on the Web: XML makes it relatively straightforward to define new document types, author and manage documents and transmit and share those across the Web. The Extensible Stylesheet Language (XSL) [5], used for expressing stylesheets, provides a language for translating XML documents and an XML vocabulary for specifying formatting semantics. Information in XML format on the Web can be transformed in a presentation format like HTML [6], WML [14] or SMIL [12] with XSL stylesheets. This separation of content and presentation information enables platform customization and user personalization of Web content.

This document discusses the design and implementation of a system (WebEPG) made by Philips Research that uses XML/XSL for tailoring Electronic Program Guide (EPG) information. However, the framework that adapts information to different conditions and users is also applicable to other domains such as tourist information, weather reports and, of course e-commerce. EPG information is available on many sites [3, 9, 13] and provides schedule and background information on TV programs. As both the number of consumer access devices with Internet connectivity and the amount of offered TV content are rapidly increasing such EPG information is getting more and more important.

The presentation of EPG information can be tailored to users needs expressed or influenced by their personal profiles which may include, taste preferences, education level and available time [25, 1] constraints such as user disabilities or social-cultural background (e.g. language) [15]. Furthermore, information can be adapted to the available network bandwidth and capabilities of information appliances [26, 27] like TV sets, smart phones, PDAs and remote controls: A smart phone has a smaller display and less memory than a PC, for example, so the information must be conveyed in a different manner.

The remainder of this document is organized as follows: First the architecture for the WebEPG framework is described, then the presentation generation process is discussed in the current framework; more particularly the way in which the presentation of the EPG document is tailored to different environments and user preferences stored in user profiles. The next sections discuss the current implementation of the WebEPG service and show some results of EPG presentations adapted to user profiles on different consumer devices followed by an evaluation of the WebEPG algorithm. The documents ends with future research questions and conclusions in the last section that are of interest for the Dynamo project. In this project we are exploring ways to generate hypermedia presentations in an effective, efficient and generic manner.

2 WebEPG Architecture

The WebEPG framework consists of three separate layers and two process in between as shown in figure 1: The datasource layer (1) contains distributed meta-data about documents or parts of documents stored by content access providers, made available on the Web. The document generation process uses this meta-data to extract or retrieve documents from these different, possibly distributed datasources. In the case of the WebEPG service the content of these documents is TV program and related information. The document layer (2) imposes an application-defined structure on these retrieved documents. A presentation generation process adds layout and style information to the documents in the document layer and adapts the presentation to the capabilities of the information appliance and the preferences of the user. The presentation layer (3) then displays these documents on the end-platform.

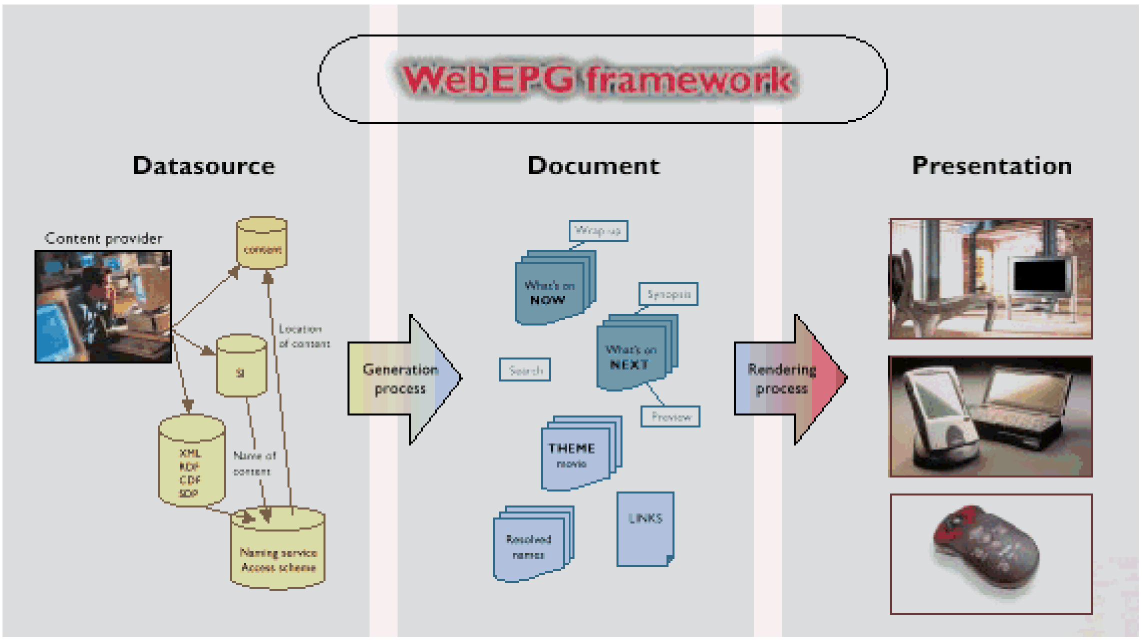


Figure 1: WebEPG framework

In the current implementation, it is assumed that the EPG information is already retrieved and available in the document layer. The structure of this EPG document is described in an XML DTD. The focus of the current WebEPG application is on the design and implementation of the presentation generation process (rendering process) in figure 1.

Figure 2 shows the client-server architecture of the document and presentation layers and the rendering process: The EPG information, delivered by an information provider, is stored in XML document(s) with corresponding DTD on the content access server. This EPG information must then be presented in an appropriate way to different users on a wide variety of presentation environments. This is done by XSL stylesheets. The presentation generation process describes how the content on the content access server is adapted to a particular presentation environment (client) and will be discussed next.

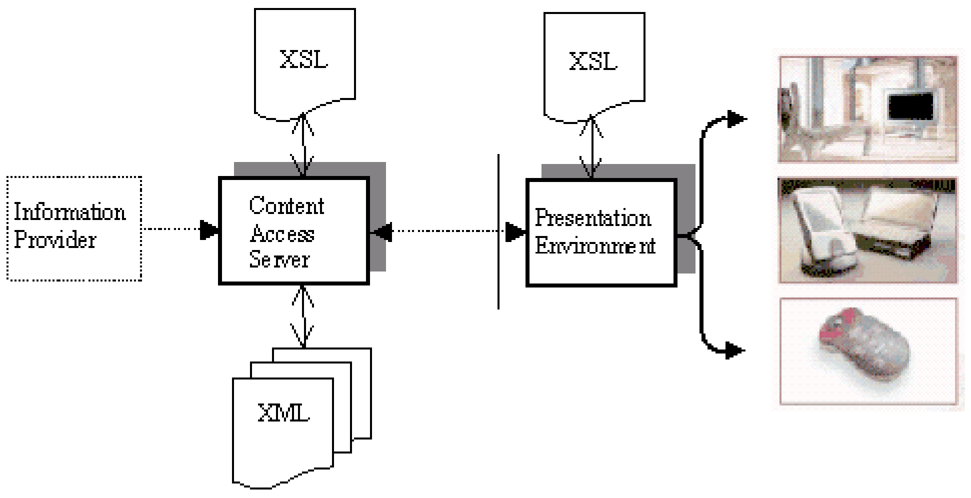


Figure 2: WebEPG framework

3 Presentation generation

The presentation generation process adapts the EPG document to the capabilities of the presentation environment and the preferences of users. Users are interested in different topics and have different personal tastes regarding the ”look-and-feel” of the presentation for example. In general, the EPG document described in XML is transformed in a presentation language format like HTML, WML or SMIL through XSL stylesheets which provide platform customization and user personalization. Figure 3 shows the entire process of presentation adaptation.

3.1 Platform customization

The capabilities and requirements of the presentation environments or consumer devices are modeled as a set of XSL rules in an XSL stylesheet. A stylesheet selection mechanism chooses the right kind of stylesheet for a particular presentation environment from a set of stylesheets, one for each type of consumer device (see figure 3, right).

3.2 User personalization

Whereas platform customization could be modeled as a static set of XSL rules, personalization aspects such as preferred colors, layout, parental rating, channels and themes cannot be translated into generic XSL rules because these preferences are dynamic. Since each user has different preferences that can even change within a session, it is undoable to create stylesheets for all possible user preference combinations. To solve this problem the DOM [2] is used to change a particular stylesheet at run-time based on a user's profile

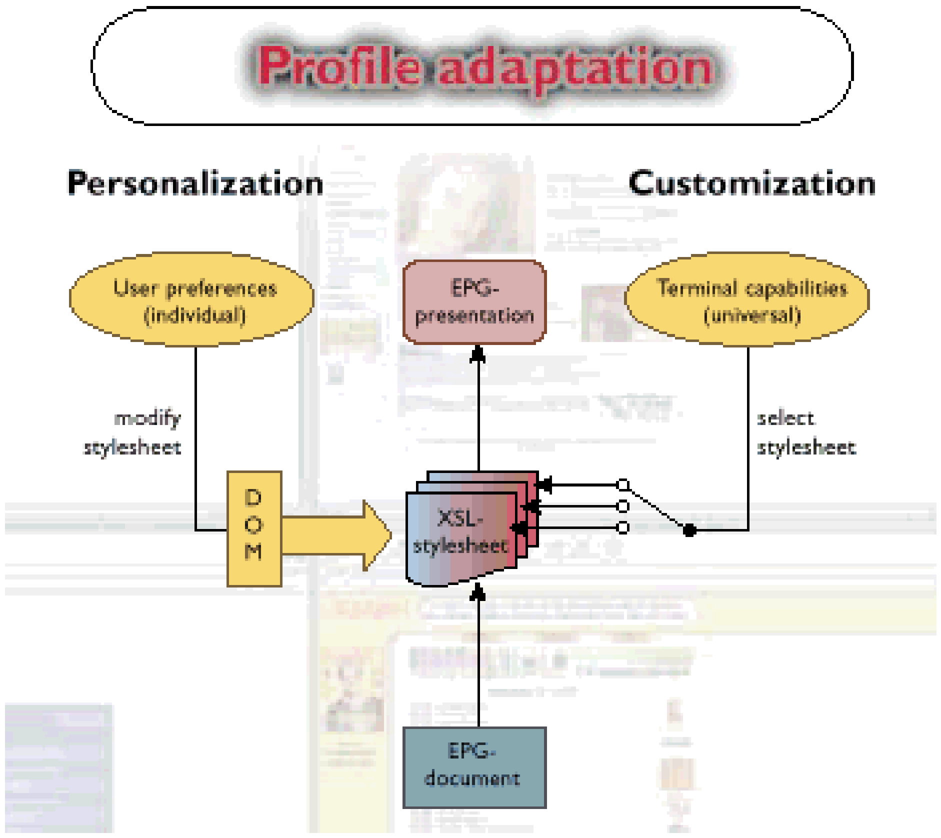


Figure 3: Presentation adaptation

(see figure 3, left).

At the moment a proprietary XML vocabulary for modeling is used to model user preferences, however a meta-data language like RDF [11] seems more appropriate. The user preferences are modeled in an XML document as a set of pattern-value pairs. As an example, consider the next line in an user profile that indicates that a particular user is only interested in the TV channels SBS6 and RTL4:

```
<profile pattern="EPG_SERVICE_SELECTION"
         value="(.='SBS6') or (.='RTL4')">
```

Each pattern attribute in the user profile corresponds with the name attribute of an element in a generic XSL stylesheet. See for example, the following generic XSL stylesheet rule that matches the user profile sample above:

```
<xsl:variable name="EPG_SERVICE_SELECTION"
     value="service/@station-name[EPG_SERVICE_SELECTION])">
```

During processing the value attribute of this element in this stylesheet is (partially) replaced by the value attribute belonging to this pattern attribute in the user profile. The following XSL rule is the result:

```
<xsl:variable name="EPG_SERVICE_SELECTION"
     value="service/@station-name[(.='SBS6') or (.='RTL4')]">
```

This stylesheet rule is applied to the EPG document.

3.3 Program recommendation

Another example of personalization is TV program recommendation; users can get personal advice on TV programs which are of interest according to their personal profiles [19]. Personalized filtering of offered TV programs mechanism in the WebEPG application is implemented in the following way: Each event (TV program) in the EPG document is classified into one or more theme categories (e.g. movies, sports, news) and each event-category pair is assigned a value between 0 and 255 that indicates the probability that the event belongs to the theme category. The user can add theme ratings to her user profile ranging from 0 ("I hate this category") to 7 ("I love this category") via a simple relevance feedback dialog to personalize the selection process: Filtering (searching) takes place by summing up per event the user theme rating with the system theme rating and selecting those events that are above a certain threshold. This is done in a similar way as described in the previous paragraph.

The final result of the presentation generation process is an EPG document in a presentation format like for example HTML, WML or SMIL. This document is then ready to be displayed on the user's presentation platform.

4 WebEPG Implementation

The current WebEPG application is implemented as follows (see figure 4): The EPG document layer and presentation generation process reside on a webserver that offers Java servlets [10] support. This webserver contains the EPG document and system ratings in XML format and the platform capabilities modeled as rules in XSL stylesheets. The client sends an HTTP request to this webserver that contains:

- a reference to the user profile and ratings (the profile and ratings are actually stored in two separate XML documents but the profile points to the location of the user ratings so we only need to send the profile reference to the server)
- a pointer to the location of the EPG document
- a reference to the stylesheet of the user's presentation platform

Note that this is just one example of a conceivable location reference or placement strategy; others are also possible. This HTTP request is then handled by a Java servlet that adapts the EPG document to the user profile and the presentation environment of the user. The servlet uses the IBM XML4J parser [8] to parse the XML documents and the

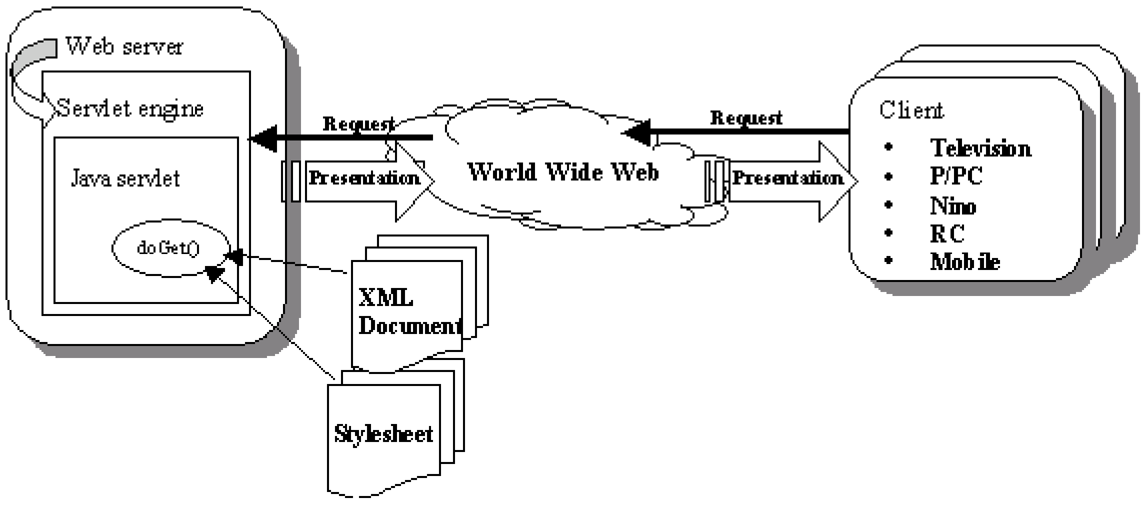
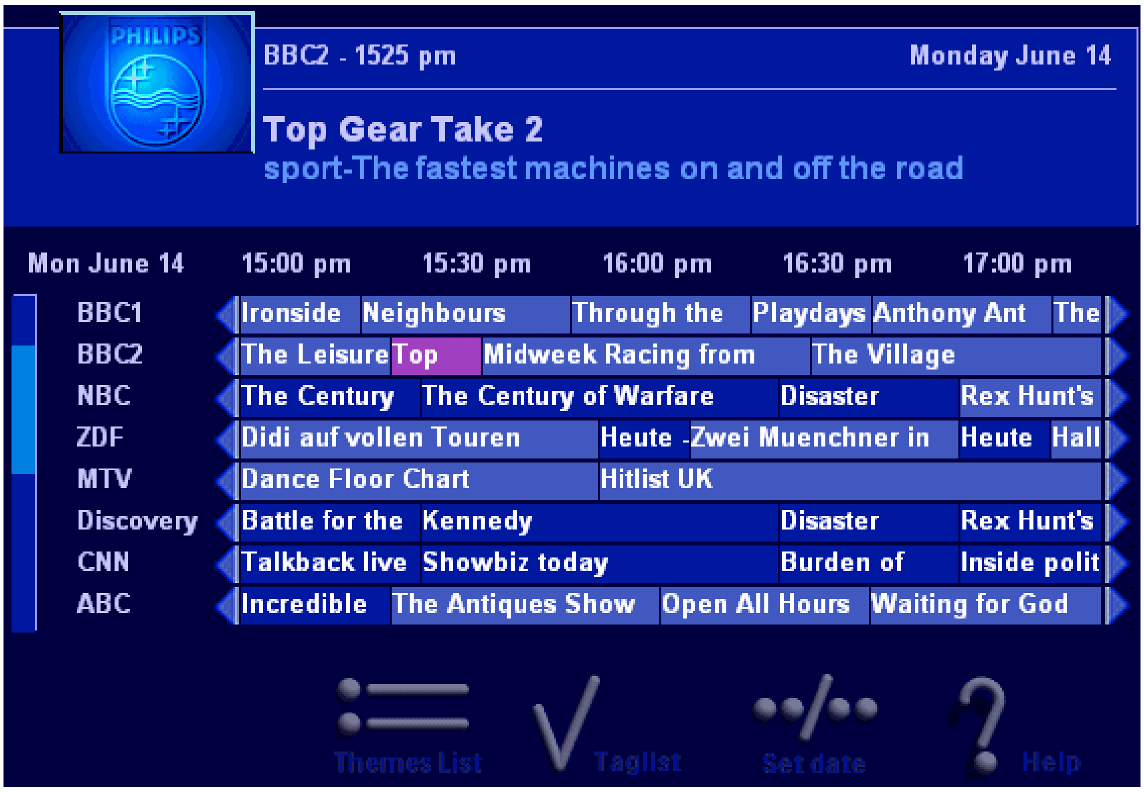


Figure 4: WebEPG implementation

Java LotusXSL processor [7] to format and transform it into a presentation language like HTML. The XSL processor interfaces with the XML parser through the DOM 1.0 API which allows access to in-memory parse trees of the XML parser; this is necessary for the XML (and XSL) formatting and transformation by the XSL processor. Figure 4 shows the implementation of the current WebEPG application.

5 Experiments with the WebEPG

No user evaluation has been conducted so far. However, to demonstrate the effects of platform customization and user personalization stylesheets have been implemented stylesheets for a wide range of presentation environments including TV, PC, remote control and Palmsize PCs. As an example consider figure 5, which shows the EPG presentation in HTML format on a TV set. Figure 6 shows how the same EPG looks on a Personal Digital Assistant (PDA) formatted in the WML language.


BBC2 - 1525 pm
Monday June 14
Top Gear Take 2
sport-The fastest machines on and off the road

Mon June 14 15:00 pm 15:30 pm 16:00 pm 16:30 pm 17:00 pm

BBC1 Ironside Neighbours Through the Playdays Anthony Ant The
BBC2 The Leisure Top Midweek Racing from The Village
NBC The Century The Century of Warfare Disaster Rex Hunt's
ZDF Didi auf vollen Touren Heute - Zwei Muenchner in Heute Hall
MTV Dance Floor Chart Hitlist UK
Discovery Battle for the Kennedy Disaster Rex Hunt's
CNN Talkback live Showbiz today Burden of Inside polit
ABC Incredible The Antiques Show Open All Hours Waiting for God

Figure 5: TV look

Figure 8 shows a PC look adapted to the theme ratings of a particular user. The theme ratings were specified in XML, as shown in figure 7. The score of the ratings is indicated with star bars. The XML EPG data and theme ratings originated from TV Advisor, a recommender system for TV [19]: The output of TV Advisor was parsed into the WebEPG XML format and later customized to different platforms and personalized to different user profiles using the adaptation process described earlier. Figure 8 shows the presentation of the TV Advisor results that has the same look as the original TV Advisor interface, customized and personalized using these theme ratings: A program about automobiles like ”Top Gear Take 2” (value 2) scores considerably lower (only two stars) than a documentary like ”Firepower 2000” (value 6).

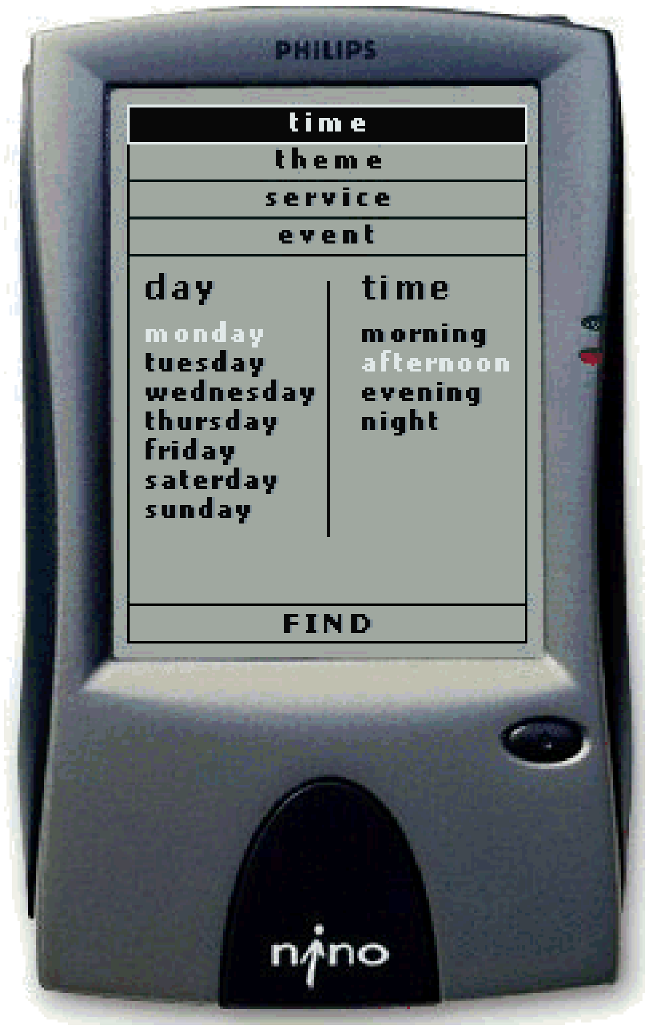
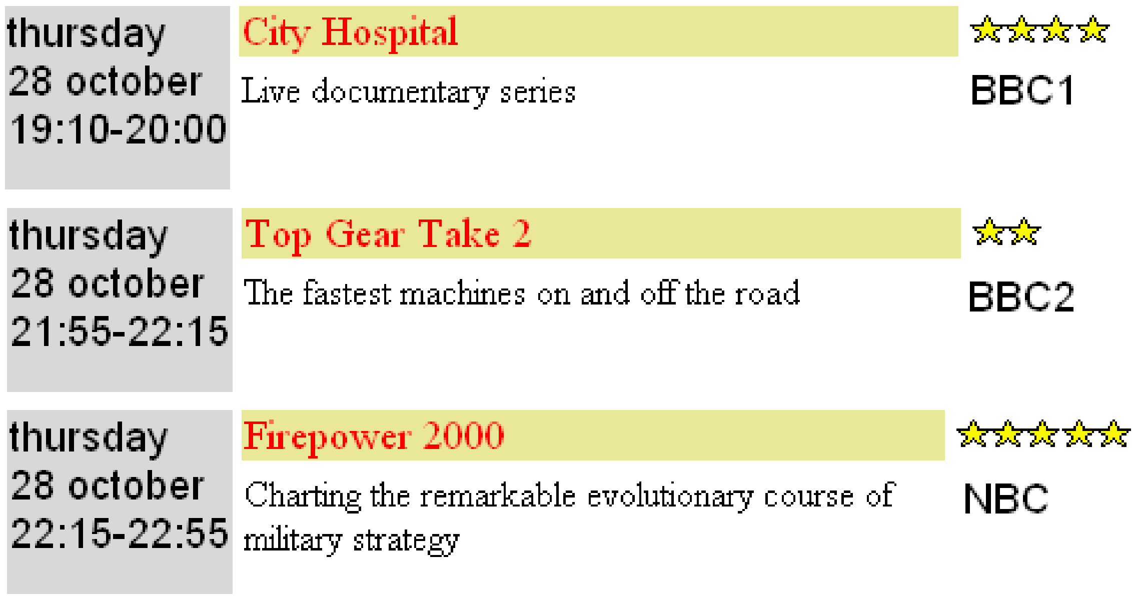


Figure 6: Nino look

```
<theme-rating>
  <profile name= "Automobiles"      value= "2"/>
  <profile name= "Cartoon"          value= "5"/>
  <profile name= "Documentary"      value= "6"/>
  <profile name= "Science fiction"  value= "1"/>
  <profile name= "Thriller"         value= "5"/>
  <profile name= "Adventure"        value= "4"/>
</theme-rating>
```

Figure 7: Theme ratings



thursday 28 october 19:10-20:00	City Hospital Live documentary series	★★★★ BBC1
thursday 28 october 21:55-22:15	Top Gear Take 2 The fastest machines on and off the road	★★ BBC2
thursday 28 october 22:15-22:55	Firepower 2000 Charting the remarkable evolutionary course of military strategy	★★★★★ NBC

Figure 8: The TV Advisor PC look, adapted to user theme ratings in figure 7

6 Performance analysis

The WebEPG implementation has been tested with different XML EPG documents (taken from TV Advisor) using a Java test program that takes an XML EPG, XML rating, XML profile and an XSL stylesheet document as input and returns timing information and the created presentation. This test program was run for the different document sizes on two PCs with different amounts of internal memory:

- Intel Pentium II, 400 MHz, 128 Mb RAM running Windows NT 4.0
- Intel Pentium II, 350 MHz, 64 Mb RAM running Windows NT 4.0

The tests on both machines used the standard JVM in JDK 1.2.2 and IBM's XML4J parser (version 2.0.15) and Lotus XSL processor (version 0.18.2). The two most time-consuming procedures identified in these tests were:

- Parsing the original XML EPG document
- Applying the modified stylesheet to the EPG document in memory

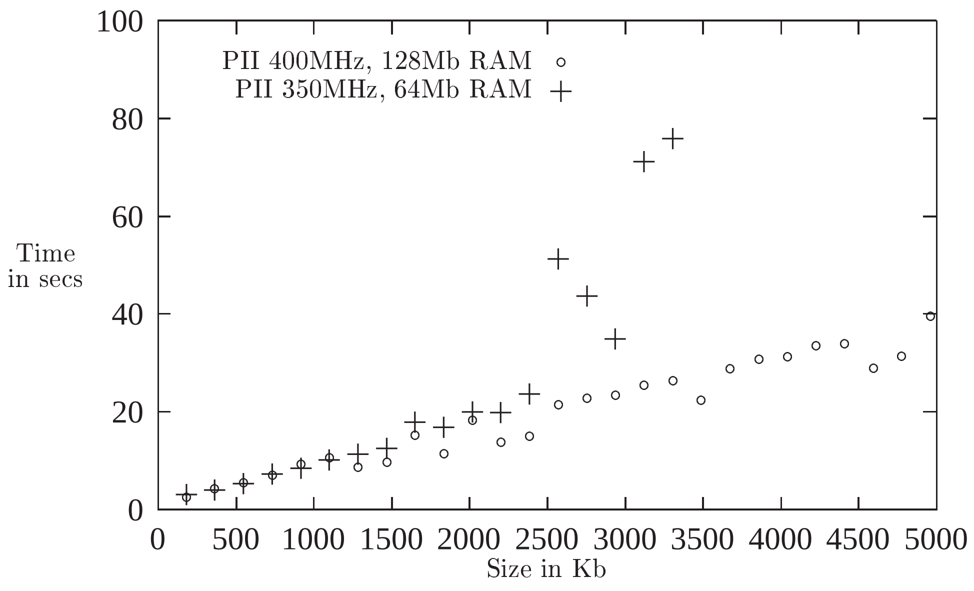


Figure 9: XML parsing durations

Based on these observations, the XML parser and XSL processor times were compared to different document sizes. Figure 9 shows the performance of the IBM's XML parser for various sizes of the XML EPG document generated by TV Advisor. The XSL processor performance has been tested on two different XSL stylesheets: The first stylesheet (figure 10) has been generated by the WebEPG application, the second stylesheet (figure 11) is a stripped down version of this stylesheet that only selects those XML elements that are specified by the user profile: It demonstrates some of the information retrieval capabilities of XSLT.

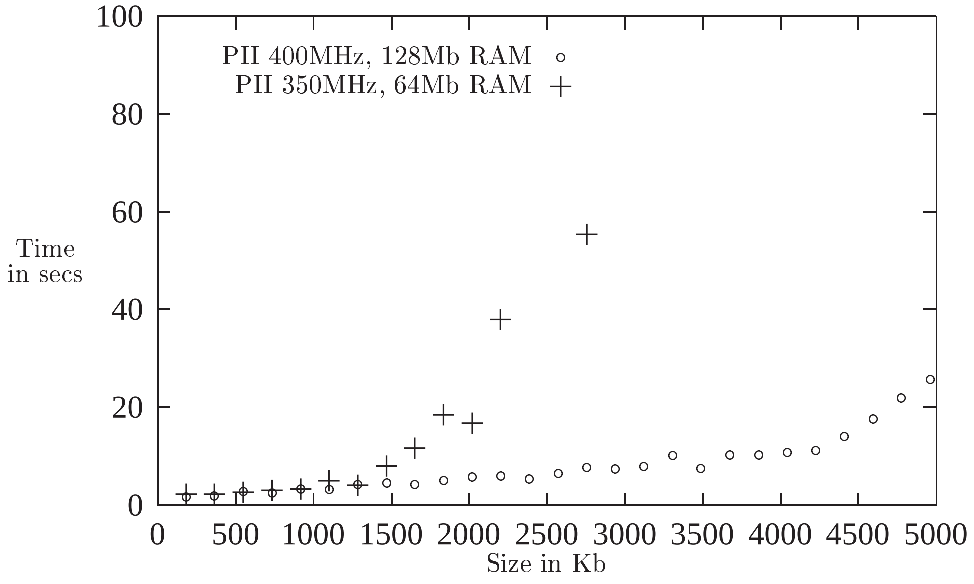


Figure 10: XSL processing durations, stylesheet 1

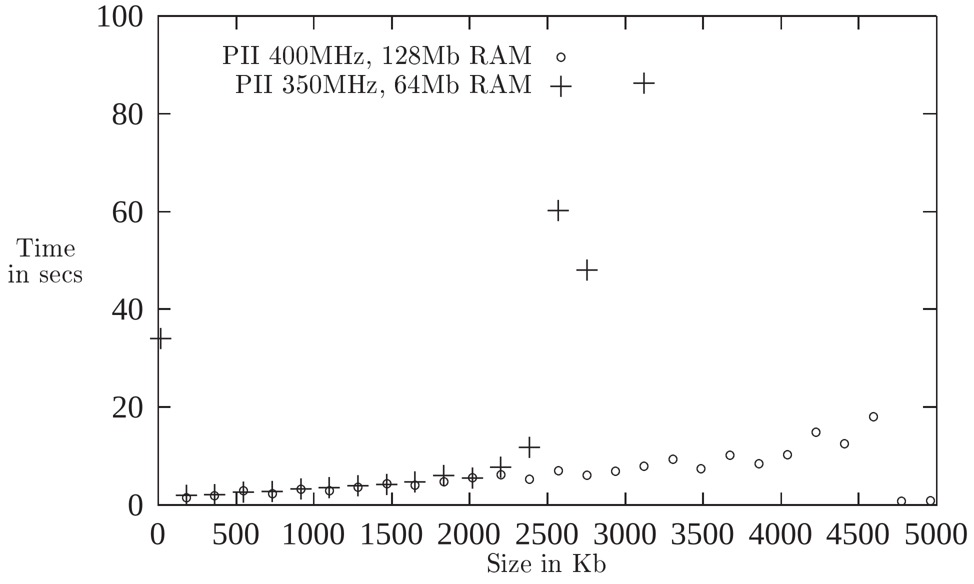


Figure 11: XSL processing durations, stylesheet 2

The complexity and order of rules in the XSL stylesheet were not tested in more detail because they depend on the implementation of the XSL processor. However, these may be important test parameters for XML parser and XSL processor builders. For more XML parser benchmarking on other platforms and in other programming languages, see [16].

6.1 Evaluation

Figure 9 shows that XML parsing is linear in time if there is sufficient memory. The Pentium II with 64 Mb runs out of memory if the XML document is around 2.5 Mb. Figures 10 and 11 show the XSL processor performance: Applying an XSL stylesheet on an XML document is also linear in time until the system runs out of memory and must swap to disk[1]. These two figures also show that complex stylesheets need more processing power and memory than simple ones as figure 10 shows a sharper increase than figure 11 for both machines. XML parsing and XSL processing are memory expensive processes that take up a lot of processing power as well. Figure 9 indicates for example that it takes 10 seconds to parse a 1 Mb XML file on a Pentium II running at 400 MHz.

The XML/XSL products have not been optimized as they were not meant for commercial use so part of the results may be contributed to inefficient programming: The XML parser and XSL processor may keep references to objects that are not used anymore for example. Nevertheless, the results show that XML parsing and XSL processing is best used on small XML document sizes: Efficient DBMS and IR systems are needed to reduce the amount of data for XML parsing and XSL processing as much as possible.

6.2 Improvements

The previous section already mentioned that the XML parser and XSL processor implementations might be improved. But there is also room for improvement to the WebEPG application itself: In the original algorithm the large XML EPG document is parsed again for each user request. This is not very efficient because the XML EPG document does not change very often. An obvious improvement was therefore to parse the XML EPG only when needed, i.e. when the right XML EPG document is not available in memory. This has been implemented and the new version runs considerably faster on a second request.

At this moment the XSL presentation stylesheet is modified by a piece of software that uses the DOM to change the presentation stylesheet. However, it is also possible to replace this piece of software by a “profile” stylesheet that describes how the profile must be mapped to the presentation stylesheet and an XSL processor that changes this profile stylesheet and the user profile into a stylesheet that can be applied on the presentation stylesheet (figure 12. This way it is not necessary to recompile the stylesheet adaptation process if the profile has been changed.

XSLT is now used both for selecting program (or event) elements from the XML EPG file

[1]The Pentium II with 128 Mb RAM gave a memory exception during all the tests when the XML document size reached 5 Mb

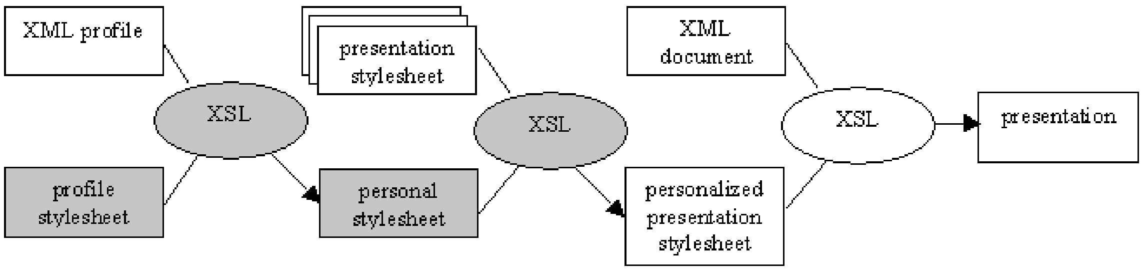


Figure 12: Replacing the stylesheet modification software with XSLT (gray blocks).

and formatting. But XSLT is not a meant to be used as a query language. The results above also suggest that it is better to split up the stylesheet in a retrieval query specification in for example XQL or an object query language like OQL and a presentation specification in a format like XSL or CSS: First the user query is used to retrieve (select) the relevant programs from the database and then the XSL stylesheet(s) is applied to format the results. See also figure 13.

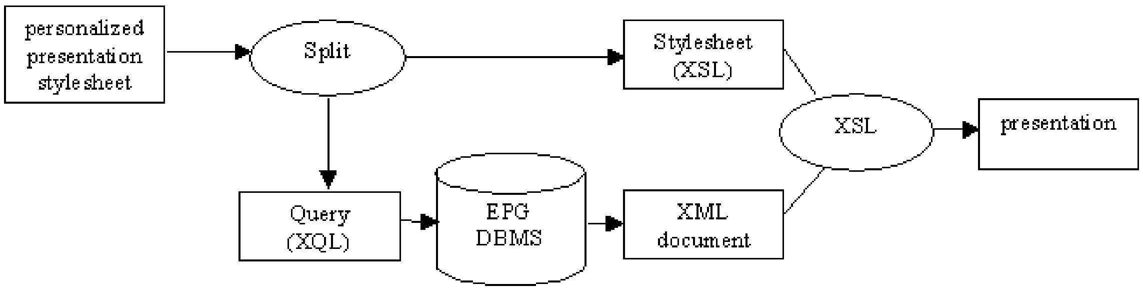


Figure 13: Separating retrieval from presentation of Web content

7 Future directions and related work

In the current framework the presentation personalization is domain dependent: A different stylesheet and user profile is needed for every application. Furthermore, the evaluation of the WebEPG shows that presentation without retrieval (selection) first does not scale up to large document sizes: Personalization of the query is clearly needed before the information has been retrieved (or selected). In the Dynamo project we are investigating these issues in more detail as the goal of this project is to design and implement a system that generates hypermedia presentations in an efficient, semi-automatic and domain independent way. The WebEPG application itself is an example of such an intelligent presentation systems just like [20, 18, 22, 24] for example. It generates presentations adapted to platform capabilities and user preferences. The architecture under proposal in Dynamo is partly based on the Standard Reference Model (SRM) for Intelligent Multimedia Presentation Systems (IMPS) [17].

8 Conclusions

Presentation of information on the WWW adapted to different presentation environments is gaining importance as more and more consumer devices with Internet connectivity appear. Moreover, the presentation of this information must also be tailored to the personal user interests and preferences because of the huge amount of content available on the WWW and the fact that different users have different tastes. The WebEPG framework and application in particular show that WWW technologies like XML, XSL and the DOM can be used to present TV program information stored in XML on a wide range of consumer access devices, modeled as XSL stylesheets, tailored to different users, represented as XML profiles, in a generic and flexible way.

XML parsing and XSL processing time for different XML EPG document sizes were measured to test the performance of the WebEPG application: The results of these tests show that XML parsing and XSL processing require a lot of processing power and large amounts of memory. This indicates that while on one hand optimization of XML parsers and XSL processors is needed, especially for use in consumer access devices, efficient storage and retrieval techniques of DBMS and IR systems are also necessary in order to speed-up the selection process that precedes presentation. Another concern of the current framework is that the presentation personalization is domain dependent: A different stylesheet and user profile is needed for every domain. These are important considerations in the Dynamo project in which we are investigating ways to generate hypermedia presentations in an efficient, semi-automatic and more generic, domain-independent manner.

9 Acknowledgments

This document that also appears as a technical note was not possible without the advice and help of Warner ten Kate, Duco Das, Leon van Stuivenberg, Astrid Dobbelaar and the work done by Johan Gielen who designed and implemented the WebEPG framework and application.

References

[1] Client-specific web services by using user agent attributes, W3C note-agent-attributes-19971230. http://www.w3.org/TR/NOTE-agent-attributes-971230.html.

[2] Document object model (DOM). http://www.w3.org/DOM/.

[3] EuroTV. http://www.eurotv.com/index2.htm.

[4] Extensible markup language (XML). http://www.w3.org/XML/.

[5] Extensible stylesheet language (XSL). http://www.w3.org/Style/XSL/.

[6] Hypertext markup language (HTML). http://www.w3.org/MarkUp/.

[7] IBM alphaworks: Lotusxsl. http://www.alphaworks.ibm.com/tech/LotusXSL.

[8] IBM alphaworks: XML parser for java. http://www.alphaworks.ibm.com/tech/XML.

[9] Infomedia electronic program guides. http://www.infomedia.lu/9epg.htm.

[10] Java servlets. http://www.javasoft.com/products/servlet/index.html.

[11] Resource description framework (RDF). http://www.w3.org/RDF/.

[12] Synchronized multimedia interpretation language (SMIL). http://www.w3.org/AudioVideo/.

[13] TVShow - worldwide television shows, stars and guides. http://www.tvshow.com/tv/.

[14] WAP wireless markup language specification. http://www.wapforum.org/what/technical/SPEC-WML-19990616.pdf.

[15] Web accessibility initiative (WAI), W3C WAI working group. http://www.w3.org/WAI/.

[16] XML parser benchmarks. http://www.xml.com/pub/Benchmark/article.html.

[17] M. Bordegoni, G. Faconti, S. Feiner, M.T. Maybury, T. Rist, and P. Trahanias S. Ruggieri. A standard reference model for intelligent multimedia presentation systems. *Computer Standards and Interfaces*, 18:477–496, 1997.

[18] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *Journal of User Modeling and User-Adaptive Interaction*, 6:87–129, 1996.

[19] D. Das and H. ter Horst. Recommender systems for TV. In *In AAAI-98 Workshop on Recommender Systems*, Madison, USA, 1998.

[20] P. de Bra, G. Houben, and H. Wu. AHAM: A dexter-based reference model for adaptive hypermedia. In *Proceeding of the ACM Conference on Hypertext and Hypermedia*, pages 147–156, 1999.

[21] J. Gielen. WebEPG: Personalized information presentation in a device independent manner. Technical Report TN 354/99, Philips Research Eindhoven, November 1999.

[22] A. Kobsa, D. Muller, and A. Nill. KN-AHS: An adaptive hypertext client of the user modeling system bgp-ms. *Review of Information Science*, 1(1), 1996.

[23] M.T. Maybury and W. Wahlster, editors. *Readings in Intelligent User Interfaces*. Academic Press / Morgan Kaufmann, April 1998.

[24] J. Rabinowitz, N. Mathe, and J.R. Chen. Adaptive hyperman: A customizable hypertext system for reference manuals. In *In Proceedings of the AAAI Fall Symposium on Artificial Intelligence Applications in Knowledge Navigation and Retrieval*, Cambridge, MA, November 1995.

[25] F. Rousseau, J. Antonio Garca-Macas, J. Valdeni de Lima, and A. Duda. User adaptable multimedia presentations for the www. In *In Proceedings of the 8th international conference on the WWW*, Toronto, Canada, 1999. http://www8.org/w8-papers/2b-customizing/user/user.html.

[26] W. ten Kate, D. Bulterman, P. Deunhouwer, L. Hardman, and L. Rutledge. Presenting multimedia on the web and in tv broadcast. In D. Hutchison and R. Schafer, editors, *ECMAST Lecture notes in computer science*, volume 1425, pages 56–69, Berlin, Germany, May 1998. Springer.

[27] W. ten Kate and H. Radha. Bringing the web to the tv: Convergence scenarios. In *W3C Workshop "Television and the Web"*, pages 29–30, Sophia-Antipolis, France, June 1998. http://www.w3.org/Architecture/1998/06/Workshop/.

[28] M. Weiser. The computer for the twenty-first century. *Scientific American*, pages 94–100, 1991.

Appendix: Raw performance analysis data

Below the data used to plot the performance graphs. Stylesheet 1 refers to the fully personalized stylesheet with presentation rules that is used in the WebEPG application, stylesheet 2 is a stripped down version of this stylesheet which only selects those XML elemements that correspond with the user profile.

XML parsing (Pentium II, 350 MHz, 64Mb), see also figure 9:

```
#GNU plot PC5648
#XML document: <variable sizes>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel.xsl
#Parsing the XML EPG document (size / time)
184 3184
367 4076
551 5408
735 7291
919 8564
1103 10174
1287 11406
1471 12578
1655 17876
1839 16924
```

```
2022 20069
2206 19929
2390 23744
2574 51324
2757 43673
2941 34960
3125 71152
3309 75979
3492 126622
3676 124539
```

XML parsing (Pentium II, 400 MHz, 128Mb), see also figure 9:

```
#GNU plot PC5856
#XML document: <variable sizes>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel.xsl
#Parsing the XML EPG document (time / size)
184 2513
367 4376
551 5538
735 7100
919 9263
1103 10625
1287 8672
1471 9714
1655 15182
1839 11406
2022 18306
2206 13770
2390 15052
2574 21431
2757 22763
2941 23374
3125 25366
3309 26278
3492 22422
3676 28782
3860 30735
4044 31235
4228 33478
4411 33858
4595 28922
4779 31445
4963 39507
5146 66555
```

XSL processing (Pentium II, 350 MHz, 64Mb, stylesheet 1), see also figure 10:

```
#GNU plot PC5648
#XML document: <variable size>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel.xsl
#Applying (modified) XSL stylesheet on XML EPG document (size / time)
184 2233
367 2173
551 2574
735 3004
919 3325
1103 5037
1287 4056
1471 7931
1655 11667
1839 18457
2022 16734
2206 37895
2390 93114
2574 100264
2757 55350
2941 152019
3125 132831
3309 265892
3492 346138
3676 569519
```

XSL processing (Pentium II, 350 MHz, 64Mb, stylesheet 2), see also figure 11:

```
#GNU plot
#XML document: <variable size>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel2.xsl
#Applying (modified) XSL stylesheet on XML EPG document (time / size)
184 1942
367 2053
551 2643
735 2814
919 3234
1103 3525
1287 3925
1471 4156
1655 4656
1839 5959
2022 5457
2206 7732
2390 11787
```

```
2574 60227
2757 48029
2941 124579
3125 86264
3309 332118
3492 320030
3676 418952
```

XSL processing (Pentium II, 400 MHz, 128Mb, stylesheet 1), see also figure 10:

```
#GNU plot PC5856
#XML document: <variable size>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel.xsl
#Applying (modified) XSL stylesheet on XML EPG document (size / time)
184 1532
367 1843
551 2794
735 2403
919 3214
1103 3155
1287 4176
1471 4467
1655 4136
1839 5007
2022 5568
2206 5949
2390 5268
2574 6439
2757 7551
2941 7281
3125 7891
3309 10035
3492 7501
3676 10265
3860 10214
4044 10726
4228 11166
4411 13960
4595 17535
4779 21892
4963 25677
5146 751
```

XSL processing (Pentium II, 400 MHz, 128Mb, stylesheet 2), see also figure 11:

```
#GNU plot
```

```
#XML document: <variable size>
#XML profile: profile.xml
#XML rating: rating.xml
#XSL stylesheet: AdvisorChannel2.xsl
#Applying (modified) XSL stylesheet on XML EPG document (size / time)
184 1452
367 1823
551 2904
735 2274
919 3134
1103 2914
1287 3595
1471 4287
1655 3936
1839 4726
2022 5538
2206 6159
2390 5198
2574 6960
2757 5989
2941 6870
3125 7862
3309 9313
3492 7361
3676 10105
3860 8442
4044 10235
4228 14821
4411 12468
4595 17955
4779 701
4963 741
```