Some Issues on Presentations in Intelligent Environments

Christian Kray

Saarland University
P.O. Box 15 11 50
66041 Saarbrücken
Germany
kray@cs.uni-sb.de

Antonio Krüger

Saarland University
P.O. Box 15 11 50
66041 Saarbrücken
Germany
krueger@cs.uni-sb.de

Christoph Endres

Saarland University
P.O. Box 15 11 50
66041 Saarbrücken
Germany
endres@cs.uni-sb.de

Abstract. Intelligent environments frequently embed a varying number of output means. In this paper, we present an analysis of the task of generating a coherent presentation across multiple displays. We review problems and requirements, and propose a multi-layer approach aimed at addressing some of the previously identified issues. We introduce a low-level architecture for the handling of devices as well as mechanism for distributing coherent presentations to multiple displays. We then discuss what further steps are needed on the way from an abstract presentation goal to a concrete realization.

Introduction

From a technical point of view the trend of Ubiquitous Computing is tightly bound to the notion of the disappearing computer, where most of the processing power and communication abilities will be embedded behind the scenes. However, from the user's perspective Ubiquitous Computing will lead to ubiquitous input and output facilities in a given environment. New display technologies (e.g. organic displays) will increase the amount of displays in an instrumented environment significantly, allowing intelligent multimedia presentation systems to plan and render presentations for multiple users on multiple displays. Modern airports are already good examples of environments equipped with several different types of displays: Information on arrivals and departures is presented on huge public boards, at gates plasma screens display information on the actual flights, throughout the building small touch screens are used to provide single users with information on the airport facilities. Finally wireless LAN hotspots allow the usage of private PDA screens to access information on the web at several locations. Until now those displays cannot be used together to present information, but several technical suggestions were recently made to move towards a solution for this problem. In [1] a framework is presented that incorporates different devices to render audio and graphics files. An approach that allows users to access publicly available displays (e.g. from an ATM) for personal use is presented in [2]. Some research has been also carried out on the combined usage of a PDA and large screens [3,4,5,7].

However, only little work was done on issues regarding the planning and rendering of multimedia presentations on multiple displays. In this paper we reflect both on the different technical prerequisites and the concepts that will allow for distributed multimedia presentations in instrumented environments. This involves the handling of devices, e.g. their registration and access, the rendering of the presentations, e.g. synchronizing presentations in time and space and the planning of presentations, e.g. how to guide the users attention from device to device. While the processes underlying a presentation using multiple devices are rather complex and hence call for a sophisticated approach that takes into account a number of factors ranging from device control to guiding the attention of the user, the benefits are worth the effort: On the one hand, the users will enjoy consistent and non-interfering presentations that are adapted to the current situation. On the other hand, these presentations will use all the available output means instead of just small limited set. Hence, the system will deal with the complexity while the users benefit from a simplified interface.

We will first review some key issues in the context of intelligent environments in general, and more specifically point out problems related to handling presentations in such a setting. In order to address some of the issues raised, we will then present a low-level infrastructure capable of dynamically handling various devices. Based on this foundation, we will introduce a mechanism for rendering presentations that are distributed across multiple devices in a synchronous way. These presentations are the result of a planning process that can only partially rely on traditional presentation planning. Consequently, we will point out some areas where further reasoning is necessary. After sketching out a possible solution for this problem, we will shortly summarize the main points of this paper, and provide an outlook on future research.

Issues

During the process of planning, generating, and rendering a presentation in intelligent environments several issues arise that go beyond those encountered in traditional multimedia presentation [9]. The issues are mainly caused by the dynamicity of the ubiquitous setup and the diversity of the devices used for a presentation. A further factor that complicates the process is the potential presence of multiple users.

A fundamental issue that does not only impact presentations but all services and tasks running 'on' an intelligent environment is the discovery, handling, and control of devices. In the case of presentations mainly output devices such as displays or loudspeakers are of interest in this area. The infrastructure has to provide means to determine what devices are present and available, to control these devices, and to dynamically add and remove devices.

The latter point is also closely related to another key issue: the constant change an intelligent environment may be faced with. On the one hand, new devices and users entering or leaving a room may require the system to re-plan all presentations that are currently active. On the other hand, presentations running in parallel may interfere with each other, e.g. two unrelated but simultaneous audio messages. Therefore, a suitable presentation engine has to constantly reassess the current device assignment, and potentially re-plan frequently. In doing so, it must also guarantee a smooth transition from one assignment to another in order to not confuse the users. The requirement for constant re-planning also calls for a representation format that supports this process.

The potential presence of multiple users causes further problems in addition to interference. On the one hand, planning several presentations instead of a single one results in increased complexity. On the other hand, the sharing of devices (such as a large display) adds a new dimension to the planning as well as to the rendering process. An additional issue concerns the actual rendering or output of a presentation. In order to realize a coherent presentation, it is vitally important to properly synchronize the various output devices. In the synchronization process, device-specific properties such as bandwidth, memory, and speed have to be taken into account. A suitable mechanism for synchronized rendering of presentations would therefore have to provide means for prefetching and pre-timed events.

In the following, we present a basic infrastructure for device handling, a rendering mechanism for distributed presentations, and some ideas on the planning process that are aimed at addressing several of the issues raised above.

Device handling

In a dynamic, instrumented environment, there are several requirements for managing the devices. One key factor to take into consideration is the fluctuation of people and their personal devices (e.g. PDAs or mobile phones), the concurrency of several applications/presentations and the differing features of similar devices (e.g. PDAs with color or monochrome displays, or different display sizes).

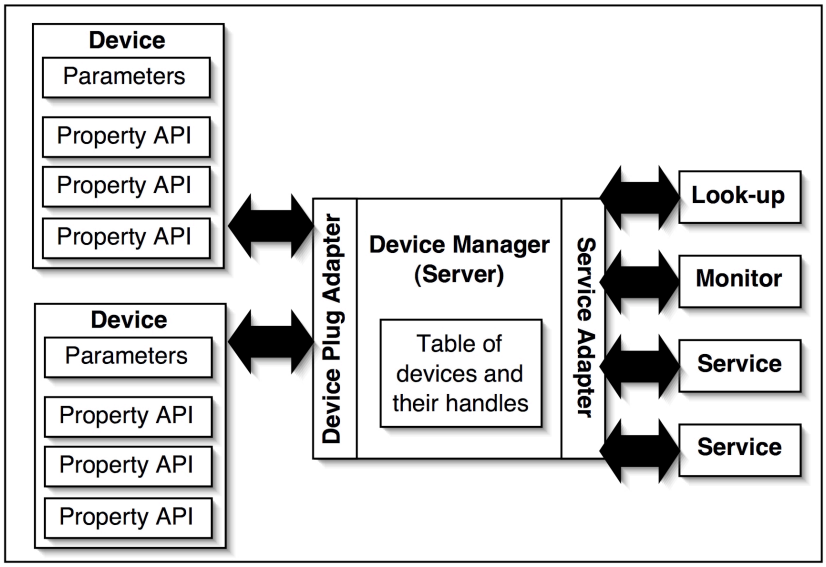

Fig. 1. Architecture of the Device Manager

The task of classifying devices turns out to be rather complex. One possible approach, which was realized in the FLUIDUM project [8], is to assume a different point of view: Instead of classifying devices, we define device feature objects (e.g. video in, audio out, tag reader, etc) and then describe a device using a list of feature objects it possesses.

As underlying infrastructure, we use a device manager with two remotely accessible interfaces. On the one hand, devices can register and announce their availability; on the other hand, services can register and check for devices with certain features. The structure of the device and the architecture of the device manager are shown in Figure 1.

A device consists of a table containing parameter value pairs (for instance "name=camera01") and a collection of property objects (or "features"). The advantage of realizing these objects as remotely accessible APIs is, that we do not only know which features a device possesses, but we can also access it directly. (We decided to use Java remote method invocation (RMI) since it is an advanced technology for object marshalling and sending, but does not have the communication overhead of more sophisticated approaches such as CORBA or JINI.)

The device registers with the device manager over a "device plug adapter" (an RMI interface). The device manager keeps an up-to-date list of registered devices, and provides information to connected services that are interested in knowing about available devices. The device manager thus serves as a matchmaker between services and devices, and also passes events concerning the deregistering of devices to the services. It hence provides a lookup or yellow page service for a given environment such as a room.

Presentation rendering

In the previous section, we described an infrastructure aimed at handling the (de-) registration and low-level control of various devices including output means such as displays. While this certainly provides a basis to build upon, further services are needed in order to render presentations in a coherent way. As we have seen previously, the synchronization of a presentation that is spread across multiple displays or output devices is a key concern that still needs to be addressed. Furthermore, the spatial constellation of the devices used for the presentation plays an important role.

For example, we want to avoid to generate a sophisticated presentation and to display it on a screen that is behind the user. In addition, we have to cope with the removal of some devices while a presentation is running, and thus with the incremental and continuous re-planning of the presentation. Therefore, there is a need for a representation format that not only incorporates space and time but that is also modular and can be rendered by a variety of different output devices.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <smil>
   <head>
     <meta name="title" content="example"/>
     <layout>
       <root-layout title="demo" id="root" width="240" height="300"/>
       <region title="main" height="320" id="main" z-index="1"
         width="240" top="0" left="0"/>
     </layout>
   </head>
   <body>
     <par id="par1">
       <audio begin="0s" region="main" id="audio1" src="song.wav"
         dur="10s"/>
       <seq id="seq1">
         <img begin="0s" region="main" id="img1" src="one.png"
           dur="5s"/>
         <img begin="0s" region="main" id="img2" src="two.png"
           dur="5s"/>
       </seq>
     </par>
   </body>
 </smil>
```

Fig. 2. Example SMIL presentation

We would like to propose the Synchronized Multi-Media Integration Language (SMIL) [16] to address these issues for several reasons. Firstly, the language incorporates a sophisticated concept of time allowing for the sequential and parallel rendering of various types of media. In addition, it is possible to specify the duration, the beginning and the end of the actual rendering of a part of a presentation both in absolute and relative time. Secondly, there is a similar concept of space, which supports both absolute and relative coordinates. Furthermore, it incorporates a region concept (providing local containers) as well as a simple layer concept that allows for the stacking of regions. Unfortunately, the underlying space is only two-dimensional, which is of limited use in an inherently three-dimensional scenario such as intelligent environments. However, we will point out ways to overcome this limitation later in this section.

A third reason supporting the adoption of SMIL for ambient presentations lies in its inclusion of various media such as text, images, sound and videos at the language level. The short example shown in Figure 2 illustrates the simplicity of including various media. The example describes a presentation that displays a slide show of two pictures (``one.png'' and ``two.png'') that are each shown for five seconds while a sound file is playing (``song.wav''). This brings us to a fifth reason for using SMIL, which is the conciseness of the format. Since it is text-based, it can even be compressed and adds very little overhead to the media-data itself. This is especially important in an intelligent environment where many devices will not have a high-speed wired connection but rather an unreliable wireless link with a lower bandwidth.

A final reason favoring the use of SMIL in ambient presentations consists of the availability of the corresponding player software on a wide range of devices. SMIL is a subset of the format used by the REAL player [15], which runs on desktop computers, PDAs and even some mobile phones. Similarly, a slightly outdated version of SMIL is part of the MPEG-4 standard [13] that is at the heart of a variety of services and media players such as QuickTime [12]. The Multimedia Message Service (MMS) [11] -- a recent addition to many mobile phone networks -- is also based on SMIL, and consequently, the current generation of mobile phones supporting MMS provide some basic SMIL rendering capabilities. Furthermore, there are several free players such as S2M2 [14] and X-Smiles [17]. For our purpose, especially the latter one is interesting as it is continuously updated and its source code is available. In addition, it has been specifically designed to run on various devices ranging from desktop computers to mobile phones.

However, before we can actually design a service for controlling and rendering SMIL on multiple output devices, we have to address the problem of SMIL only supporting two dimensions. Fortunately, this is a straightforward task. Assuming that we are dealing with bounded spaces – e.g. we do not consider interstellar space -- we can develop a function that maps the surface of a bounded space, e.g. a room, to a two-dimensional plane. Figure 3 shows an example for such a function. If we just want to deal with rectangular rooms and wall-mounted displays the mapping is very straightforward and consists basically of an `unfolding' of the corresponding three-dimensional box.

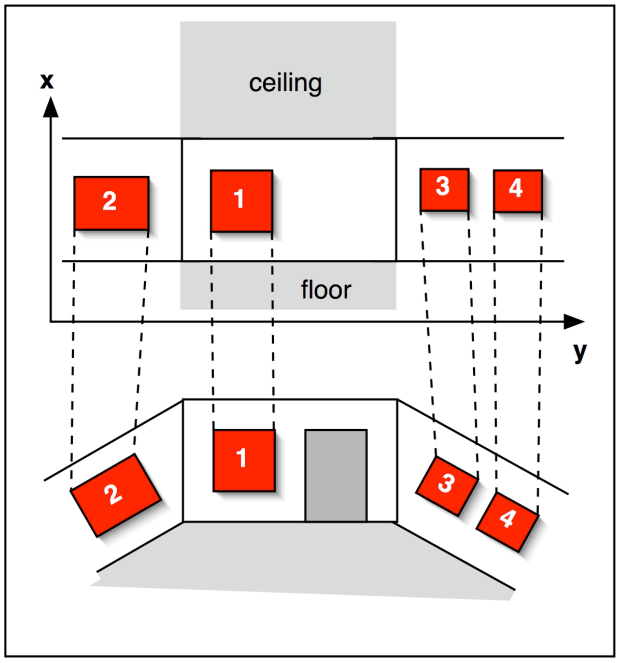

Fig. 3. Mapping of 3D surfaces to a 2D plane

Obviously, the more complex a room is, the harder it is to find an `intuitive' mapping that preserves some of the spatial properties. This is even more so, if we take into account mobile devices such as PDAs or tablet computers. However, as long as the mapping function is bijective, we can always determine which part of a SMIL presentation corresponds to a specific location in 3D space, and vice versa. Even if the mapping does not preserve spatial properties such as neighborhood relations, we can perform spatial reasoning in 3D space by re-projecting 2D parts using the mapping function. Therefore, it is safe to assume the existence of a table-based bijective mapping function based on the basic infrastructure described in the previous section. Since a service maintaining this function, i.e. the presentation-planning component, receives constant updates on the availability and location of various output devices, it can ascertain the integrity of the mapping function. Using such a mapping function, a presentation planner can generate a complete SMIL presentation covering multiple devices in a room. At this stage, we need a piece of software that takes this presentation and sends the various parts of it to the corresponding devices, and that assures that the overall presentation is delivered in a coherent way. We have implemented such a service - the SMIL Manager shown in Figure 4 - that takes care of this part of the presentation pipeline. By introducing a SMIL property object for all devices capable of rendering (parts of) a SMIL presentation, we can rely on the standard service interface for device access and control. The API of the SMIL property object implements the actions of the protocol listed in Table 1.

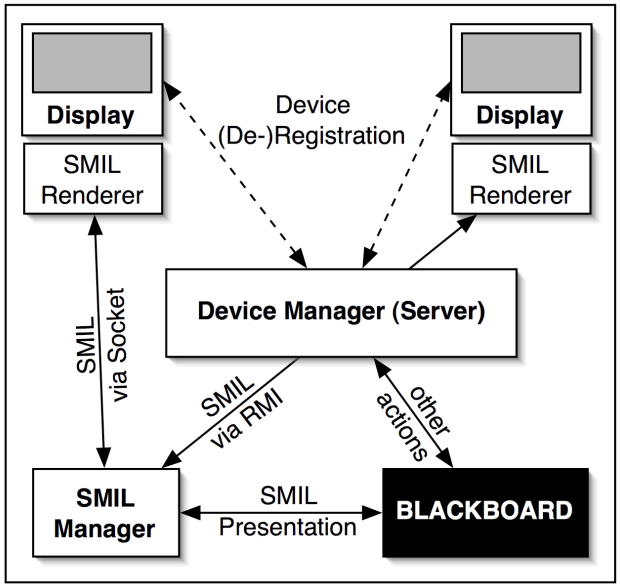

Fig. 4. Architecture of the SMIL Manager

Using the Device Manager (presented in the previous section), the SMIL Manager can hence keep track of all devices capable of rendering SMIL presentations. It relies on the mapping function to identify, which part of the overall presentation is to be rendered on which device. It then sends the various parts of the presentation to the corresponding devices for prefetching. Once all devices have reported back after completing the prefetching of media included in the presentation, the SMIL Manager triggers the simultaneous start of the presentation on all devices.[1]

However, there are a few details in the process that deserve proper attention. In order to assure a synchronized presentation, the SMIL Manager sends out synchronization messages to all attached output devices on a regular basis. Otherwise, the different clocks and time-servers that the various devices rely on would result in unwanted behaviors such as a part of the presentation starting too early or too late. Furthermore, the prefetching of media is vitally important. Especially on a shared wireless network, bandwidth is limited and can result in media not being available when they are needed in the presentation if we do not perform prefetching. But even in case prefetching is included in the process, it is important to actually check whether it has completed on *all* devices since there are great differences in terms of the time that is required to download all media. For example, a screen attached to a wired workstation will probably fetch the required media for a presentation much faster than a PDA connected through WLAN.

[1] Note that the simultaneous start of the presentation on all devices does not necessarily imply that all devices will actually produce output right away. More often, some will render something directly while others will wait for a predefined time before actually outputting something.

Action	Explanation
<LOAD>	Load a presentation specified by an included URL.
<START>	Start a presentation specified by an included URL, optionally a specified time.
<STOP>	Immediately stop a presentation specified by an included URL.
<SYNCHRONIZE>	Synchronize internal clock with time stamp of SMIL Manager.
<REPORT>	Send a message whenever the user activates a link.
<LOADED>	Presentation at included URL has been fully prefetched.
<FINISHED>	Presentation at included URL has finished playing.
<LINK>	User has activated a link pointing to included URL.

Table 1. Protocol for interaction between the SMIL Manager and output devices

Table 1 lists the various actions that the SMIL Manager can perform on the connected devices. These are either transmitted using remote method invocation (RMI) or through a socket connection using plain text messages such as the ones listed in the table. Obviously, there has to be a means to instruct a device to load a presentation, and to start it (at a given time). Additionally, it should be possible to stop a running presentation, and to synchronize the devices with the clock of the SMIL Manager. Finally, there is rudimentary support for interaction (through the <REPORT> action). When enabled, the device reports back to the Manager in case the user clicks on a link embedded in the SMIL presentation. On touch-sensitive screens, for example, this allows for simple interactions. We will discuss some implications of this feature in the last section of this paper. Furthermore, the devices report back once the current presentation has finished playing.

The SMIL Manager as well as the client application are written in Java and have successfully been run on various desktop computers and PDAs. Since they only require version 1.1 of the Java virtual machine, we are confident that they will run on further devices in the future (such as mobile phones). SMIL rendering relies on the X-Smiles engine [13], and is realized as a dedicated thread that is separate from the communication (either through RMI or a socket connection). Therefore, both the server and the client can communicate while performing other tasks (such as playing a presentation or registering further clients). This is an important feature especially in the context of stopping a currently running application (e.g. when the device has to be used by another service, or when the presentation has to be re-planned due to the removal of some device).

Presentation planning

Now that we have discussed how to control the different devices and how to handle distributed SMIL presentations, there is a final stage still missing in the presentation pipeline: the planning and generation of the presentation. As we have noted in the introduction, there is a lot of research going on in the context of single-user single-device presentations, i.e. for a person sitting in front of a desktop computer. However, when faced with a multi-device (multi-user) presentation, there are some key properties that distinguish the corresponding planning process from traditional setups.

Firstly, time and space play a much more prominent role since not only are the various output devices distributed across space but also do they have to be properly synchronized. The latter point refers to the issue discussed in the previous section as well as to the planning stage since the presentation planner has to take into account when to present what on which display. Secondly, it may frequently occur that multiple users are present in a room, which entails several problems not present in single-user scenarios. For example, a number of users may share certain devices, which may interfere with one another, e.g. when one person is making a phone call while a second person uses a voice-based approach to access a database. In addition, security and privacy come into play if there is more than one person involved.

A third important differentiating factor consists of the dynamicity of the environment. While traditionally, it is assumed that the devices used for a presentation do not abruptly disappear, this assumption is not realistic in an intelligent environment. Even more so, it is possible that further output devices actually appear, e.g. a person carrying a PDA enters an intelligent room. All these factors call for an approach that facilitates re-planning on various levels, which in turn is a further difference to more static and traditional presentation planning. In order to take these special requirements into account, we distinguish three different planning problems that are interleaved with each other: (1) the planning of the *content, (2)* the planning of the *media distribution in space and time* and (3) the planning of the *attentional shift (focus control)* that has to be supported when users have to focus on different displays in their environment over time.

The first problem is similar to content selection in classical intelligent multimedia presentation systems designed for single display scenarios (e.g. [9]). Hence, we will not discuss it here, as it can be addressed using a classical plan operator based planning technique such as STRIPS [10]. For the planning of the spatio-temporal distribution we plan to adopt a constraint-based mechanism described in [6], where the rendering abilities of each display (e.g. resolution, audio quality, interaction requirements) is modeled and scored with a certain value. The rendering of a video on a PDA display would for example achieve a lower score than on a big plasma screen. Although most of the constraints are soft (e.g. it is possible to render a video on a PDA), some are not: if the presentation contains interactive elements (e.g. buttons) it is not possible to render them on a screen without input capabilities. In this case a plasma screen without a touch screen overlay for example could not be used. One possible solution would consist of rendering the interactive parts of the presentation on a device with a touch-screen (e.g. a PDA) and the rest of the presentation on the plasma screen.

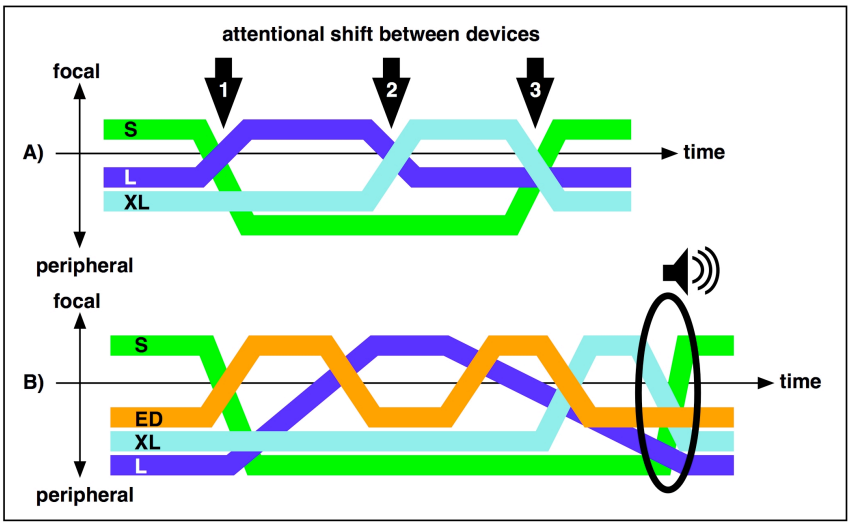

Fig. 5. A distributed presentation: A) without focus control, and B) with focus control using an everywhere display and audio meta-comments

From the users perspective such a distributed presentation can cause problems, if they do not know where to focus their attention at a certain moment of time. One way to overcome this problem is illustrated in Figure 5. The diagram in Figure 5 A) shows an example of a presentation distributed over time on a small ('S' - e.g. a PDA), a large ('L' - e.g. a 17-inch touch-screen) and an extra large ('XL' - e.g. a plasma screen) device. From the user's perspective devices in the environment can be either in focus or peripheral. The three arrows in Figure 5 A) indicate that the user has to perform an attentional shift, whenever a device switches from the focus role to a peripheral role and vice versa. The user's ability to perform this shift may be hindered by several factors, e.g. the spatial distance between the devices and distracting cues in the environment (e.g. noise). If a user is not able to shift the attention to the right device at the right moment in time, the probability increases of the presentation becomes confusing or, even worse, that it is misinterpreted.

Therefore it makes sense to support this shift of attention in the instrumented environment. In Figure 5 B) two additional means are used to facilitate the attentional shift. The shifts 1 and 2 are supported by an everywhere display [18] (ED). An ED is a device that is able to project images on arbitrary surfaces in the environment, i.e. a ceiling-mounted projector attached to a movable stand that can be controlled remotely. Therefore it can used to guide the attention of the user from one place in the instrumented environment to another place in the environment. If the environment knows the user's position, an ED can even be used to guide the attention from a PDA that is carried by the user towards a fixed display in the environment. Other means to shift the user's attention can be meta-comments that explicitly tell the user where to look next (e.g.: "To stop the video please press the corresponding button on your PDA"). In Figure 5 B) such an audio meta-comment is used to close the third "attentional gap".

We can extend the constraint-based planning approach by introducing another score that represents the difficulty of the users to direct their attention from one display to another in a given situation. Of course these additional elements of the presentation increase the overall presentation time leading to other problems that may result into a backtracking process and a modified distribution of the presentation.

Conclusion and outlook

In this paper, we discussed several issues and possible solutions in the context of generating coherent presentations in an intelligent environment. We presented an analysis of the task of generating a coherent presentation across multiple displays, namely the planning, the rendering and the control of the output devices. We reviewed several problems and requirements such as synchronization and three-dimensionality, and introduced a multi-layer approach aimed at addressing some of the previously identified issues. We described a low-level architecture for the handling of devices as well as mechanism for distributing coherent presentations to multiple displays using the Synchronized Multimedia Interface Language (SMIL). We then discussed the issues related to presentation planning, and provided some ideas for implementing the corresponding process.

Consequently, a major part of future work consists in actually designing a working system for the planning process, and to test it with various services such as navigation, localized information or smart door displays. A second field of major interest is the capture of user interaction. We have already started on this point by incorporating a simple link following action in the protocol used to connect the SMIL Manager and its clients. However, the underlying Device Manager allows for a much more fine-grained interaction, which we intend to include in future versions of the system.

Acknowledgements

The authors would like to acknowledge their funding though the Deutsche Forschungsgemeinschaft (DFG) within the Special Research Center 378: Resource-Adaptive Cognitive Processes and the Fluidum research group.

References

1. T. L. Pham, G. Schneider, S. Goose : A Situated Computing Framework for Mobile and Ubiquitous Multimedia Access Using Small Screen and Composite Devices, in Proc. of the ACM International Conference on Multimedia, ACM Press, 2000

2. Roy Want, Trevor Pering, Gunner Danneels, Muthu Kumar, Murali Sundar, and John Light: The Personal Server: Changing the Way We Think about Ubiquitous Computing, Proc. of Ubicomp 2002, LNCS 2498, Springer, 2002, pp. 192
3. Scott Robertson, Cathleen Wharton, Catherine Ashworth, Marita Franzke: Dual device user interface design: PDAs and interactive television, Proc. of CHI'96, ACM Press, 1996,pp. 79.
4. Yasuyuki Sumi , Kenji Mase: AgentSalon: facilitating face-to-face knowledge exchange through conversations among personal agents, Proc. of Autonomous agents 2001, ACM Press, 2001.
5. Brad A. Myers: The pebbles project: using PCs and hand-held computers together, Proc. of CHI 2000, ACM Press, 2000,pp. 14.
6. Antonio Krüger, Michael Kruppa, Christian Müller, Rainer Wasinger: Readapting Multimodal Presentations to Heterogenous User Groups, Notes of the AAAI-Workshop on Intelligent and Situation-Aware Media and Presentations, Technical Report WS-02-08, AAAI Press, 2002.
7. Michael Kruppa and Antonio Krüger: Concepts for a combined use of Personal Digital Assistants and large remote displays, Proceedings of SimVis 2003, SCS Verlag, 2003.
8. The Fluidum project. Flexible User Interfaces for Distributed Ubiquitous Machinery. Webpage at http://www.fluidum.org/.
9. E. Andre, W. Finkler, W. Graf, T. Rist, A. Schauder and W. Wahlster: "WIP: The Automatic Synthesis of Multimodal Presentations" In: M. Maybury (ed.), Intelligent Multimedia Interfaces, pp. 75--93, AAAI Press, 1993.
10. R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2: 198-208, 1971.
11. The 3rd Generation Partnership Project (3GPP). The Multimedia Message Service (MMS). Available at http://www.3gpp.org/ftp/Specs/html-info/22140.htm.
12. Apple Computer Inc. The QuickTime player. Available at http://www.apple.com/quicktime.
13. MPEG-4 Industrial Forum. The MPEG-4 standard. Available at http://www.m4if.org.
14. The National Institute of Standards and Technology. The S2M2 SMIL Player. Available at http://smil.nist.gov/player/.
15. RealNetworks. The REAL player. Available at http://www.real.com.
16. The World Wide Web Consortium. The synchronized multimedia integration language (SMIL). Available at http://www.w3.org/AudioVideo/.
17. X-Smiles. An open XML-browser for exotic devices. Available at http://www.xsmiles.org.
18. Claudio Pinhanez: The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces, Proc. of Ubicomp2001, Volume 2201, Lecture Notes in Computer Science, 2001, pp 315.