Delivering Interactive Multimedia Services in Dynamic Pervasive Computing Environments

Cristian Hesselman
Telematica Instituut, Netherlands
cristian.hesselman@telin.nl

Pablo Cesar, Ishan Vaishnavi
Centrum voor Wiskunde en Informatica, Netherlands
{p.s.cesar, i.vaishnavi}@cwi.nl

Mathieu Boussard
Alcatel-Lucent, France
Mathieu.Boussard@alcatel-lucent.fr

Ralf Kernchen, Stefan Meissner
University of Surrey, United Kingdom
{r.kernchen, s.meissner}@surrey.ac.uk

Antonietta Spedalieri, Albert Sinfreu
Telefónica I+D, Spain
{alberts, aspedal}@tid.es

Christian Räck
Fraunhofer Fokus, Germany
christian.raeck@fokus.fraunhofer.de

ABSTRACT
This paper introduces a distributed system for next generation multimedia support in dynamically changing pervasive computing environments. The overall goal is to enhance the experience of mobile users by intelligently adapting the way a service is presented, in particular by adapting the way the user receives multimedia content from the service and how he interacts with the service. The system tailors this presentation to the user's context, for instance based on his current activity (e.g., driving or walking) or the characteristics of the devices that surround him (e.g., devices for multimedia rendering and user interaction devices). The system integrates with the (IMS-based) service platforms of "beyond 3G" telecom operators and allows multimedia content to be rendered on multiple devices at the same time. It can also dynamically and seamlessly adapt the way a service is presented while it is being used (e.g., from gesture-based input to voice commands). These changes are triggered by changes in the user's context (e.g., when the user gets into his car). This paper discusses the system's requirements, presents an architecture proposal and describes its current implementation.

Categories and Subject Descriptors
C.2.1 [**Computer Communication Networks**]: Distributed Systems – *distributed applications;* H.5.2 [**Information Interfaces and Presentation]**: User Interfaces – *Input devices and strategies, Interaction styles;* H.5.1 [**Information Interfaces and Presentation]**: Multimedia Information Systems – *video*.

General Terms
Design

Keywords
Adaptive multimodal interfaces, IMS, non-monolithic rendering

Ambi-sys'08, February 11-14, 2008, Quebec, Canada.

1. INTRODUCTION
The past few years have seen a revolution in mobile multimedia technologies. While the deployment of these technologies has been impressive, their use for interactive multimedia services has been limited to the simplest situation: to the delivery of a single set of content streams to a single device and by restricting user interaction to the keypad for basic media functionalities (play/pause/stop).

A well-known way to go beyond such a basic user experience is by means of multimodal user interfaces, which enable users to interact with services in a variety of ways [3][4][21]. Mobile environments are however highly dynamic, which means that it must be possible to dynamically tailor a multimodal interface to the situation of a particular user. For example, when a user gets into his car, the interface of the services he is using will typically need to change for safety reasons.

The contribution of this paper is a distributed system called the Multimodal Delivery and Control System (MDCS) that provides this form of adaptation. While adaptive multimodal services have been investigated before [3][4][7][8][11][21], the MDCS addresses and combines three of the major challenges that remain in this area:

- How to make adaptive multimodal services available on a large scale;
- How to use multiple devices at the same time for content rendering; and
- How to dynamically adapt the rendering and interaction features of a service based on the user's situation ("context").

To make adaptive multimodal services *widely available*, the MDCS integrates into the service infrastructure of "beyond 3G" telecom operators (typically based on the IP Multimedia Subsystem (IMS) [12]). The advantage of this approach is that service providers can rely on the telecom operator to dynamically adapt the modality of their services to the context of a particular user, which allows them to deploy multimodal services more easily. Moreover, the telecom operator can provide these multimodal facilities for multiple service providers, thus reducing the costs of operating these facilities and fueling the uptake of adaptive multimodal services. A final advantage is that telecom

operators usually have a large customer base, which makes support for adaptive multimodal interfaces available to a large number of mobile users.

The MDCS supports non-monolithic rendering [10], which means that it allows for the simultaneous use of *multiple devices* in a user's vicinity to render a particular multimedia item in a synchronized manner. This enhances the experience of a (mobile) user and is unlike current solutions for adaptive multimedia [7][8][11] that provide a fixed and self-contained set of content streams targeted at a single device. The MDCS focuses on multimedia items that take the form of multimedia presentations and supports presentations described in declarative languages such as SMIL [20] or MPEG-4 [22].

To provide truly adaptive multimodal services, the MDCS dynamically and seamlessly adapts the rendering *and* interaction features while a user is utilizing the service. These adaptations are triggered by changes in the user's context, for instance when his activity changes or when new rendering devices appear. This is unlike existing systems, which traditionally only adapt a service's interaction features or its rendering features, but not both [7][8][11] .

This paper is structured as follows. Section 2 introduces a motivating scenario in order to clarify the concepts presented elsewhere in the paper. Section 3 identifies the requirements of the MDCS and Section 4 discusses its architecture. Section 5 describes the system's implementation and Section 6 discusses related work. Finally, Section 7 concludes the paper and proposes future directions to follow in this research.

2. SCENARIO

The scenario revolves around a user called Consuelo. Consuelo is staying in a hotel and wants to watch a movie before she goes into town for dinner. Consuelo has a subscription with the VoD service provided by her mobile operator and uses the large touch screen in her room to select the movie "Casino Royale". Because Consuelo is in a hotel room and has a high-definition TV screen and her handheld device at her disposal, she receives the service in three synchronized parts: (1) the video component of the movie on the high-definition television, (2) the audio on the room's hi-fi system, and (3) the user interface for interacting with the VoD service on her handheld device [10]. Figure 1 shows a graphical representation of the scenario.

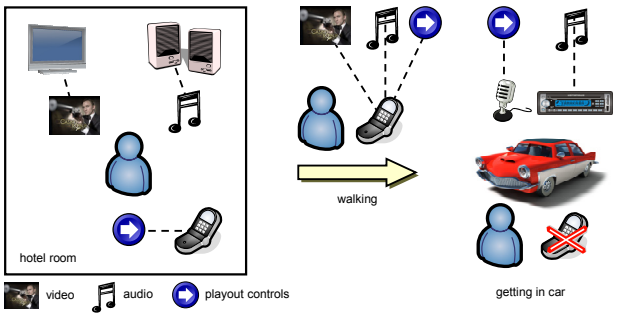

Figure 1. Scenario.

When it is time to leave for dinner, Consuelo heads for the parking lot to get to her car. As she gets out of her room, she decides to keep on watching "Casino Royale" and uses the controls on her phone to transfer the video part of the movie from the TV to her mobile phone.

When Consuelo gets in her car, her mobile phone indicates that its microphone, speakers, and display have been disabled for safety reasons. As a result, the audio of the movie is now playing on the car's audio system, while the in-car information system signals that the microphone in the car has been enabled. Since the movie is now audio only, Consuelo pauses it through a voice command and uses the in-car microphone to quickly annotate the movie with a few words that will enable her to remember the movie's story line once she gets back. She then drives off into town and uses the controls on the steering wheel to select an Internet radio station to listen to during the drive.

3. REQUIREMENTS

The scenario of Figure 1 requires a system that can dynamically adapt the delivery of a particular service in response to changes in a user's situation. This involves adapting (1) the way in which a user receives multimedia content from a service (the service's output) and (2) the way in which that user interacts with that service (the user's input). To accomplish this, the system has to go through three high-level steps (also applicable to a wider class of adaptive multimodal systems [3]):

1. Detect changes in a user's situation and discover the capabilities of nearby devices;
2. Given the user's new situation, decide on (1) the best way to deliver the content of the service to the end-user (e.g., in a non-monolithic way) and (2) on the best way for that user to interact with the service (e.g., in a multimodal way); and
3. Enforce these decisions by rendering content in a non-monolithic manner, by realizing multimodal interactions, or by transferring streams to other devices.

In the scenario of Figure 1, the system goes through these steps twice: once when Consuelo leaves her hotel room and once when she gets into her car. This high-level behavior gives rise to seven requirements, which are discussed below.

3.1 Context Detection & Resource Discovery

The system should be able to detect changes in a user's context, for instance in the set of devices in the vicinity of the user or the activity the user is engaged in. The devices in a user's vicinity can be considered as a dynamic "sphere" and are therefore called the user's *Distributed Communications Sphere* (DCS) [23] in this paper. As illustrated in Section 2, a user's DCS might change over time: Consuelo's DCS changes when she leaves her hotel room (the TV disappears from her DCS) and when she gets into her car (the car's audio system and built-in microphone become part of her DCS). A DCS also includes the network connections through which the devices in the DCS communicate with service in the infrastructure.

An important factor is the strategy that the system uses to handle changes in the user's context. The system could for instance respond to every event that signals such a change. This maximizes the match between a user's context and the way the system delivers the service (rendering and interaction-wise), but may also result in the system 'ping-ponging' between different devices and modalities. This reduces the stability of the system, which in turn negatively affects its usability. The system should therefore apply some hysteresis in responding to changes (cf. [6]), but this topic is outside the scope of in this paper.

In addition to detecting changes in a user's context, the system also needs to be able to discover the capabilities of the resources (devices and networks) in the user's DCS. Resource

discovery will however also not be considered in this paper as we expect the system to build on existing work in this area.

3.2 Decision Making

The system has to be able to decide on (1) the best way to deliver the content of the service and (2) the best way for a user to interact with that service. This decision should be based on the properties of the multimedia service involved and on the capabilities of the devices and networks in the user's DCS. In addition, it should take the user's preferences into account, which define what constitutes "best" based on the context of the user. These preferences could for instance specify a preference for certain devices (e.g., wall-mounted displays) or certain modalities (e.g., gesture-based interaction) in certain situations.

3.3 Non-monolithic Rendering

The system needs to support non-monolithic rendering, which means that it has to be able to simultaneously render the output of a particular multimedia service onto two or more devices in a user's DCS. When Consuelo is in her hotel room, for example, the system should be able to relay the audio component of "Casino Royale" to the hotel room's hi-fi system, the user interface to Consuelo's cell phone, and the video component to the wall-mounted display.

Non-monolithic rendering requires the involved devices to be time synchronized. This means that the system should be capable of transforming the timing and layout description of a service into stream and/or device-based synchronization constructs implemented in the transport/network layer. Furthermore, the user's DCS needs to have an efficient local device synchronization protocol, which works across varying network types.

3.4 Multimodal Interaction

The system should enable users to interact with a service in a multimodal way. This means that it should allow users to utilize different forms of input to interact with a multimedia service. This may require using the input capabilities of different devices in a user's DCS. The scenario of Figure 1 is a multimodal example in that Consuelo initially interacts with the movie service through the key pad of her cell phone, but afterwards uses the microphone in her car to stop the movie.

The system should also allow users to utilize multiple modalities at the same time to interact with a service. An example would be a user giving a service the instruction to zoom in on a certain part of a video by simultaneously making a gesture (e.g., pointing) and providing a voice command. This form of interaction requires a function called *input fusion* [24] [27], which combines these different inputs into a semantically meaningful action for the service.

3.5 Session Mobility

The system should be able to transfer part of the service's output to another device. This for instance allows Consuelo to transfer the video stream from the TV screen to her mobile phone (cf. Figure 1). This function is usually called "session mobility" [1][2][5].

3.6 Unified Interface

The system should provide a unified interface that allows service providers to easily deliver interactive multimedia services to a wide variety of users. The interface should abstract away from (1) the modalities that the system uses to present the service and to interact with the user, and (2) the devices and networks through which the system delivers the service. The latter is particularly important in "beyond 3G" environments in which a DCS usually consists of multiple devices and wireless networks, each with their own capabilities.

To provide a unified interface to interactive multimedia services, the system should be able to handle different types of description languages, for instance languages that describe the properties of multimedia content (e.g., SMIL [20], MPEG-4 [22], MPEG-7 [18][19]) and devices (e.g., CC/PP [25]).

4. ARCHITECTURE

This section discusses the architecture of the Multimodal Delivery and Control System (MDCS), a distributed system that fulfills the requirements outlined in Section 3. The MDCS focuses on deciding how to deliver a particular service to a particular user and on how to enforce these decisions through non-monolithic rendering, multimodal interaction, and session mobility. It relies on existing components for dynamic resource discovery and for detecting context changes.

The MDCS is specifically designed to run in the infrastructure of mobile telecom operators and forms a value-added service that, for the most part, runs on the operators' platform. The MDCS integrates into the service infrastructure of "beyond 3G" telecom operators, which will typically be based on IMS [12]. Figure 2 illustrates this.

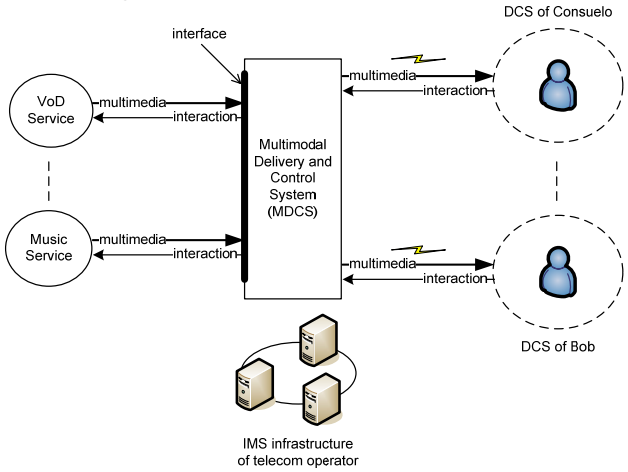

Figure 2. High-level view of the MDCS.

The MDCS architecture consists of three types of components: media components, control components, and supporting components. Media components process and render the content of a multimedia service and handle the interactions with users. Control components govern how media components are arranged and dynamically reconfigure these components to adapt to the end-user's current context. Finally, support components are components that are not part of the MDCS, but run elsewhere on the operator's platform.

The rest of this section considers the MDCS architecture in detail. It first discusses the concept of binding, which is an abstraction that associates a service to a user. Next, it introduces the MDCS' media, control, and supporting components and provides an overview of the system's behavior.

4.1 Bindings

A binding provides an association between an interactive multimedia service and a user. It enables a service to deliver multimedia content to a user and enables that user to interact with the service without the service having to be aware of the internal state of the user's DCS. This form of abstraction simplifies the development and deployment of interactive multimedia services, which is beneficial to the end-user, the service provider, and the platform operator. To allow the service to abstract away from the specifics of non-monolithic rendering, a binding also keeps track of the rendering state for a user. As a result, bindings can for instance be stopped, paused, or fast forward. Observe that a binding is not an association between a service and a device, which would expose the internals of a user's DCS to the service and would require the service to deal with the multiple modalities and devices in the DCS.

Figure 3 shows an example of a binding, in this case between the VoD service of Figure 1 and Consuelo.

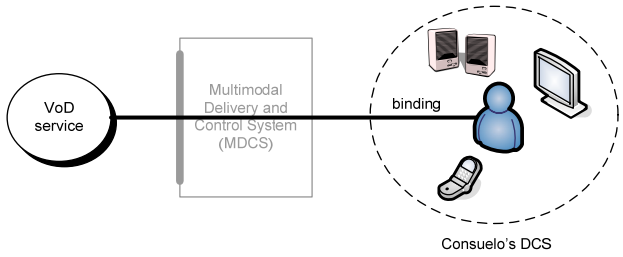

Figure 3. Example of a binding.

While a binding provides a convenient abstraction for interactive multimedia services, it is an abstract notion that the MDCS needs to realize. In the MDCS architecture, such a realization consists of a set of media components and a set of sessions that interconnect them.

4.2 Media Components and Sessions

The media components in a binding are responsible for rendering the content of the multimedia service and for handling the interactions with the user. These components are:

- *Renderers.* A renderer is a component capable of rendering multimedia elements (i.e., video, audio, images, and text). A renderer runs on a device in a user's DCS. There may be multiple renderers on different devices in the same DCS, thus allowing for non-monolithic rendering [10].
- *Output transformers.* An output transformer runs in the infrastructure of a telecom operator and can manipulate multimedia content (e.g., transcode it).
- *Activators.* Similar to a renderer, an activator runs on a device in a user's DCS. An activator catches user inputs such as speech commands, gestures, or more traditional desktop inputs like mouse clicks. The activators in one DCS may be distributed across multiple devices.
- *Input recognizers.* An input recognizer receives raw user input from an activator, determines what kind of action the input represents (e.g., a certain gesture), and sends a description of that action to the fusion component.
- *Fusion components.* A fusion component merges multiple user inputs into a single action that is understood by the service (e.g., "zoom") and conveys this action to the service. A fusion component is service-specific in that it knows which actions the service supports.

A binding contains instances of these components, which are interconnected by sessions that transport data. Figure 4 shows what the binding of Figure 3 could look like. In this example, the binding consists of an audio renderer (on the hi-fi system) and a video renderer (on the wall-mounted display) that receive audio and video packets from the VoD service through an audio and a video session, respectively. To handle interactions with the user, the binding also contains two activators, one for capturing voice commands and one for capturing gestures. Both activators run on Consuelo's cell phone and feed their data to an appropriate input recognizer (one for voice and one for gestures). The recognizers interpret the user's voice and gesture actions and forward a description of these actions to the fusion component. The fusion component combines them into a single action that is specific to the VoD service and conveys it to the service. The sessions that interconnect activators, input recognizers, fusion components, and services convey user interactions. Observe that the rendering watchers in Figure 4 are control components that are co-located with media renderers, but are not part of the binding. The example binding of Figure 4 does not contain any output transformers.

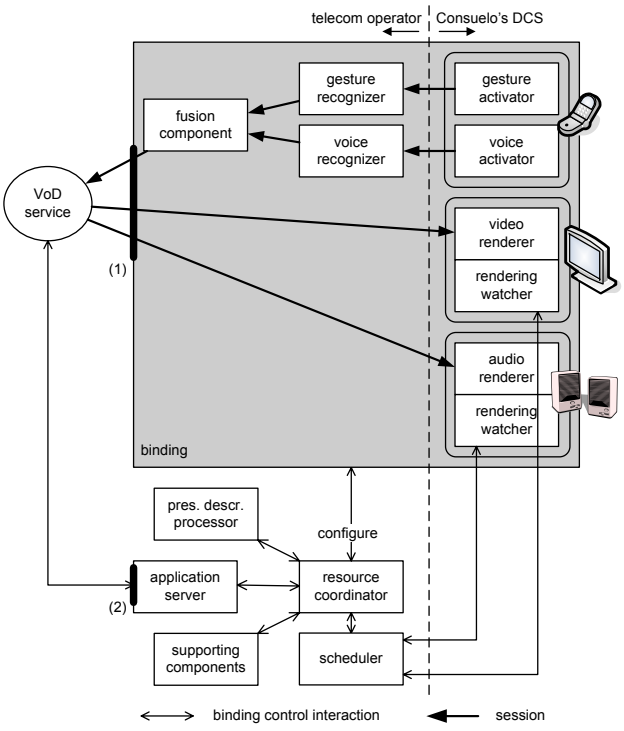

Figure 4. MDCS architecture.

The interfaces that media components expose to interactive multimedia services together form the *multimodal delivery interface* of the MDCS (interface 1 in Figure 4). This interface enables services to easily deliver their output to users in a multimodal and multi-device manner without having to be aware of the specifics of DCSs. It also allows them to receive multimodal commands from end-users without having to be aware of the actual modalities through which the user interacts with the service (e.g., a combination of gestures and voice commands).

Together, this contributes to the realization of the MDCS' unified interface (Section 3.6).

4.3 Control Components

Control components govern how bindings are organized in terms of media components and sessions. They reconfigure a binding when an event occurs that indicates that the context of a user has changed (e.g., when Consuelo leaves her room or gets into her car). The MDCS architecture contains five control components (also see Figure 4): MDCS application servers, resource coordinators, presentation description processors, rendering watchers, and schedulers.

An *MDCS application server* allows multimedia services to request the establishment of a new binding with a particular user. The MDCS application server forwards these requests to resource coordinators, which are components that control the current bindings of specific users. The interface the application server provides is called the MDCS' *binding control interface* (interface 2 in Figure 4). It abstracts away from the details of managing bindings, thus reducing the complexity of the interactive multimedia services that use it. Because the MDCS takes care of non-monolithic rendering (see Section 4.1), the binding control interface also provides basic methods that enable services to manipulate the playout of their multimedia content (e.g., stop and fast forward). The MDCS application service is also responsible for managing the life-cycle of resource coordinators and handles requests to release an existing binding.

A *resource coordinator* manages the bindings associated with a specific user. For each service, it decides (1) through which devices the user will receive the multimedia content and in which modalities (e.g., text and audio), and (2) through which devices and modalities the user will interact with the service (e.g., using gestures and keystrokes). A resource coordinator enforces these decisions by instantiating and configuring several media components, and by establishing the appropriate sessions to interconnect them using the Session Initiation Protocol (SIP). The resource coordinator also handles events that signal that the context of a user has changed, re-computes the way the media components of a binding should be arranged in the new context, and then enforces that decision by reconfiguring the set of media components in the binding. To facilitate this adaptation process, the resource coordinator keeps track of the bindings of a particular user and their realizations in terms of media components and sessions. The inputs for the decisions that a resource coordinator makes are the description of the service (input and output characteristics), the user's preferred devices and modalities, and a description of the characteristics of the available devices in the user's DCS.

A *presentation description processor* is responsible for mapping specific declarative languages such as XHTML, SVG, SMIL, or MPG-4 into an MDCS-internal data model. This allows the resource coordinator to become a language-agnostic component, thus allowing it to operate in heterogeneous environments (cf. Section 3.6).

A *scheduler* is a component that determines in what order the presentation provided by an interactive multimedia service will be rendered on the devices in a DCS. A scheduler offloads an interactive multimedia service from having to deal with non-monolithic rendering and keeps track of the state of a multimedia presentation. The goal is to keep the clocks of the scheduler and the renderers synchronized.

A *rendering watcher* is a component that keeps track of the state of an individual renderer and communicates timing information about the renderer to the scheduler (e.g., when a renderer has finished playing a certain multimedia element).

4.4 Supporting Components

The resource coordinator makes use of a set of supporting components that are not part of the MDCS, but that run elsewhere on the operator's platform. They are:

- *Recommender* [14]. The recommender is a learning component that provides modality recommendations. A modality recommendation suggests a certain set of modalities for delivering a particular service in a particular user's DCS on a particular device. A modality recommendation includes a confidence level that indicates the probability of the recommendation being correct. The recommender provides an interface for obtaining modality recommendations and for feeding information back into the recommender to improve its performance using learning algorithms. The recommender takes the user's current context into account.
- *Resource discovery facility* [23]. The resource discovery facility dynamically discovers the devices in a user's DCS and provides information about their capabilities (e.g., in terms of supported network interfaces). The discovery facility supports a request-response interface as well as a publish-subscribe interface that provides asynchronous callbacks when the composition of a user's DCS changes.
- *Context manager* [29]. The context manager provides information about the context of users, for instance in terms of their location or their current activity. The context manager supports a request-response interface and a publish-subscribe interface that provides asynchronous callbacks when the context of a particular user changes.

4.5 System Behavior

Resource coordinators form the central point of control in the MDCS. They manage the life-cycle of bindings in four different phases: an establishment phase, a delivery phase, an adaptation phase, and a release phase. These are similar to the phases discussed in [3].

The *establishment phase* begins when a user selects a content item from an Electronic Program Guide (EPG). The EPG sends a delivery request to the selected multimedia service, which in turn requests the MDCS application server to establish a binding with the user. The request contains the ID of the user, a description of the multimedia presentation that the user requested, and a description of the input commands that the service supports (e.g., "zoom" and "fast forward"). The application server forwards this request to the user's resource coordinator, possibly after creating it first. Next, the resource coordinator obtains a personalized modality recommendation from the recommender and uses the discovery facility to get a description of the set of available devices in the user's DCS. It combines this information with the description of the multimedia presentation to decide on (1) the modalities to use for delivering the multimedia content and on which devices, and (2) on the modalities to use for the interaction between the user and the service and through which devices these interactions should take place. Based on this decision, the resource coordinator creates a new binding by instantiating and configuring a set of media components (cf. Figure 4). It for

instance creates a fusion component and configures it with a description of the commands that the service supports. Finally, the resource coordinator passes a description of the content along with associated timing details to the rendering components.

The *delivery phase* begins when the resource coordinator has established a new binding and the user is receiving the service's content and is interacting with that service. During the delivery phase, the rendering watchers synchronize the playout of the content at the renderers and report their status back to the scheduler. In parallel, the service receives user inputs via the fusion component and calls the MDCS application server to execute actions that influence the rendering state of the service (e.g., pause).

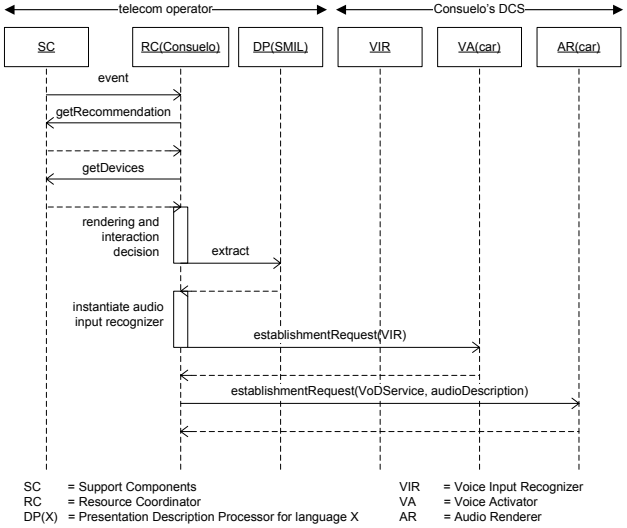

Figure 5. Example behavior.

The *adaptation phase* begins when the resource coordinator receives an event that indicates that the user's context has changed. It can receive such event from the context manager (e.g., when the user's activity changes), from the resource discovery facility (e.g., when a new device appears in the user's vicinity), or explicitly from the user. During the adaptation phase, the resource coordinator goes through the same steps as during the establishment phase, in the end possibly re-configuring the binding in terms of its media components (and hence the modalities and devices involved in the delivery of and interactions with the service). The resource coordinator also updates its internal model of the binding to match the new situation and sends updated media descriptions to the renderers involved in the delivery of the service's content.

Figure 5 illustrates this behavior when Consuelo gets into her car. Her resource coordinator receives a context change event from one of the supporting components (e.g., the context manager). The resource coordinator then consults the recommender and the discover facility in order to determine how to adapt the VoD service in Consuelo's new context. Based on their output, the resource coordinator instantiates a voice input recognizer and requests the audio activator in Consuelo's car to establish a session with it. The resource coordinator also requests the voice input recognizer to connect to a fusion component, but this is not shown in Figure 5.

Besides changing the interaction part of the binding with Consuelo, the resource coordinator also connects the audio renderer in Consuelo's car to the VoD service and disconnects the audio and video renderers on her mobile phone (not shown in Figure 5). The resource coordinator interacts with a presentation description processor to extract the audio part from the description of the service's output (in this example a SMIL description).

An important aspect of the resource coordinator in the adaptation phase is that it applies a certain degree of hysteresis to the events that it receives to avoid the MDCS from continuously changing modalities and devices. This could for instance be regulated by a set of configurable policies [28]. The MDCS goes back to the delivery phase when the adaptation phase ends.

The *release phase* begins when the MDCS is in the delivery phase and the MDCS application server receives a request from a service to release its binding with a particular user. The application server forwards this request to the user's resource coordinator, which releases the binding by stopping the associated media components and releasing the sessions.

5. IMPLEMENTATION

This section discusses the state of the MDCS implementation, in particular of the media and control components introduced in Section 4. The supporting components are third-party implementations and will not be discussed here. For more information on these components, the reader can consult the following articles: recommender [14], resource discovery facility [23], and context manager [29].

As explained in the previous section, the target platform for the MDCS is the IP Multimedia Subsystem (IMS) [12], which means that the different rendering devices, activators, and the multimedia service are connected using the IMS standard[1]. The particular implementation in use is Open Source IMS[2].

5.1 Media Components and Sessions

The rendering components (Section 4) are instances of the Ambulant Player[3], which is an open source implementation of SMIL 2.1. It runs on wide range of platforms and devices, from desktop computers to mobile devices such as Pocket PC, Linux-based PDAs, and Windows Mobile 5 systems. The major advantage of using SMIL for describing multimedia services is that it provides the timing and synchronization information of the constituting media elements. As a result, it becomes easier to perform transformations on the content, such as resizing, and to render to content in a non-monolithic manner. The output transformer is an extension of the Ambulant Annotator [10] implemented in Python.

The implementation of the MDCS integrates the scheduler into the resource coordinator (cf. Figure 4). The scheduler takes the form of an instance of the Ambulant Annotator [10], which allows users to annotate multimedia content. The Ambulant annotator is able to receive XML-RPC commands and is based on a user interaction model that provides multimodal and multi-device interaction. The model separates an RPC call for a

[1] http://www.3gpp.org/

[2] http://www.openimscore.org/

[3] http://www.cwi.nl/projects/Ambulant/distPlayer.html

particular user interaction into action semantics, action handling, and activator. The semantics provides a description of the action in terms of its name, what it provides, and what are the needed input data. The handler provides an actual implementation of the action. Finally, the activator is the implementation of the interface to the user. This model, analogous to the Model-View-Controller (MVC) paradigm, provides the benefit of a clear separation between the action handling and the actual look and feel of the interactions.

In the implementation, each interactive device provides its own implementation of the activators depending on their internal capabilities. The system includes a number of activators such as traditional graphical user interfaces in mobile phones, gesture interfaces, and voice interfaces. The raw input is then transformed into the Extensible MultiModal Annotation Markup Language (EMMA) [26] standard by the specific input recognizer and sent to the fusion component. The major advantage of using EMMA is that a device-specific input is transformed into a device-agnostic input. Finally, the fusion component, based on the actions semantics, is responsible for producing one action that will be handled by the resource coordinator.

5.2 Control Components

The following control components of the MDCS architecture have been implemented: the presentation description processor, the resource coordinator, and the rendering watcher.

The implementation of the presentation description processor targets interactive multimedia presentations and is therefore based on the SMIL language. The SMIL processor parses the description of a multimedia presentation and identifies the media elements, their characteristics, and their synchronization (e.g., video, audio, and subtitles). Based on the parsing of the document, a general description of the output model of the service is provided to the resource coordinator. Future work includes the development of presentation description processors for XHMTL, SVG, and XForms, among others.

The resource coordinator decides where and when to render the multimedia content. Its decision engine is implemented as a matching algorithm of the output model of the service, the results of the device discovery mechanism, and on recommendations. The resource coordinator is written in Java and interacts with the Ambulant Player and Annotator

The rendering watcher is responsible for enforcing the timing description and is implemented as part of each renderer. In order to synchronize the different renderers, the implementation uses the NeighbourCast-NM (Non-Monolithic) algorithm [9][15].

6. RELATED WORK

Adaptive multimodal services have been investigated before [3][4][7][8][11][21], but to the authors' knowledge they have not been integrated with the (IMS-based) platforms of "beyond 3G" telecom operators before. The MDCS is a distributed system that does exactly this. A derived contribution in this regard is the notion of a binding, which allows an interactive multimedia service to connect to a user without having to worry about the internal organization and dynamics of that user's DCS.

In terms of multimodality, a number of research papers also study full multimodal adaptation (i.e., adaptation while a user utilizes a service) based on contextual information, but they do this independently for user interaction [4][21] and rendering [3]. The MDCS, on the other hand, is designed to adapt both in an integrated manner. Another difference with existing work is that the MDCS largely runs on the servers of a telecom operator, which allows for more content adaptation possibilities (e.g., transcoding). Pure client-side solutions are less flexible in this regard [3][4].

As for rendering multimedia, traditional adaptive multimedia [7][8][11] solutions provide a self-contained stream targeted to a specific device. The MDCS, on the other hand, is capable of re-adapting the way content is rendered depending on changes in the user's context. Moreover, its architecture permits the output (or different parts of the output) of the service to be rendered through different synchronized rendering components simultaneously (i.e., in a non-monolithic manner), using the NeighbourCast-NM algorithm. As discussed in [15], the algorithm meets all the requirements for non-monolithic rendering.

Multimodal interaction should incorporate available interaction and rendering devices in the close environment into the user interaction. Aspects such as context-awareness, media content transformations, user input integration, content and device selection, device and modality service discovery are needed to be integrated in order to achieve a flexible and modular overall software architecture allowing for a better user experience [13]. Additionally, learning aspects in order to improve system response based on observed user interaction patterns can improve the overall user experience significantly as described in [14]. All these requirements are being taken into account in the proposed architecture.

7. CONCLUSIONS AND FUTURE WORK

This paper introduces the Multimodal Delivery and Control System (MDCS), a distributed system that enhances the experience of mobile users by dynamically adapting the way these users consume and interact with multimedia services. The MDCS integrates with the service platforms of "beyond 3G" telecom operators, in particular with the IP Multimedia Subsystem (IMS). It also allows a multimedia presentation to be rendered on multiple devices in parallel (non-monolithic rendering) and can dynamically and seamlessly adapt the rendering and interaction features of a service *while* a user is utilizing the service. These adaptations are triggered by changes in the user's context. The main abstraction of the system is that of a binding, which enables services providers to deliver interactive multimedia services to mobile users without having to worry about the details and dynamics of the user's communications environment (networks, devices). We discussed the MDCS' requirements, its architecture, and the state of its implementation.

Future work includes a more detailed study of mechanisms for media synchronization, advanced multimodal interaction, and device sharing (e.g., when two users share a device such as a wall-mounted display). Other future work will involve a formalization of the algorithms that the resource coordinator uses to decide how to present a particular service to a particular user (rendering and interaction-wise). The implementation of the MDCS will be extended as well, as it has a prototype status at this point. This will for instance include developing new types of presentation description processors for other for declarative languages such as SVG and XHMTL.

8. ACKNOWLEDGMENTS

This work has been conducted within the project SPICE (027617), which targets intelligent extensions of next generation service platforms.

9. REFERENCES

[1] H. Song, H.-H. Chu, and S. Kurakake, "Browser Session Preservation and Migration", the 11th International World Wide Web Conference (WWW2002), Hawaii, May 2002.

[2] H. Song, H.-H. Chu, N. Islam, S. Kurakake, and M. Katagiri, "Browser State Repository Service", International Conference on Pervasive Computing (Pervasive 2002), Zurich, Switzerland, August 2002.

[3] R. Kernchen, B. Mrohs, M. Sałaciński, K. Moessner, "Context-aware multimodal output selection for the Device and Modality Function (DeaMon)", 6th International Workshop on Applications and Services in Wireless Networks, Berlin, Germany, May 2006.

[4] R. Kernchen, P. Boda, K. Moessner, B. Mrohs, M. Boussard, G. Giuliani, "Multimodal user interfaces for context-aware mobile applications", 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Berlin, Germany, 2005.

[5] K. Ohta, T. Yoshikawa, T. Nakagawa, Y. Isoda, S. Kurakake, "Adaptive Terminal Middleware for Session Mobility," 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03), May 2003.

[6] C. Hesselman, H. Eertink, and A. Peddemors, "Multimedia QoS Adaptation for Inter-tech Roaming", Proceedings of the 6th IEEE Symposium on Computers and Communications (ISCC'01), Hammamet, Tunisia, July 2001

[7] S. Boll, "MM4U – a Framework for Creating Personalized Multimedia Content", International Conference of Distributed Multimedia Systems, Miami, Florida, September 2003.

[8] T. Lemlouma and N. Layaida, "Adapted Content Delivery for Different Contexts", International Symposium of Applications and the Internet, Orlando, Florida, 2003

[9] I. Vaishnavi, D. Bulterman, P. Cesar, B. Gao, "NeighbourCast: A synchronisation algorithm for wireless ad hoc networks", IASTED International Conference on Parallel and Distributed Computing and Systems, 2007.

[10] P. Cesar, D.C.A. Bulterman, Z. Obrenovic, J. Ducret, and S. Cruz-Lara, "An Architecture for Non-Intrusive User Interfaces for Interactive Digital Television", EuroITV, pp. 11-20, 2007.

[11] J.V. Ossenbruggen, L. Hardman, J. Geurts, and L. Rutledge, "Towards a Multimedia Formatting Vocabulary", 12th International Conference of the World Wide Web, Budapest, Hungary, 2003, pp. 384-393.

[12] 3GPP TS 23.002: "Network architecture"

[13] D. Bonnefoy, O. Droegehorn, and R. Kernchen, "Multimodality and Personalisation", in Enabling Technologies for Mobile Services: The MobiLife Book, M. Klemettinen, Ed. Chichester: John Wiley & Sons, LTD, 2007, pp. 153 - 184.

[14] Y. Du, R. Kernchen, K. Moessner, C. Räck, O. Sawade, and S. Arbanowski, "Context-aware Learning for Intelligent Mobile Multimodal User Interfaces", 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2007), Athens, Greece, 2007.

[15] I. Vaishnavi, D. Bulterman, P. Cesar, and B. Gao, "Media Synchronisation in Non-Monolithic Rendering Architectures," IEEE International Symposium on Multimedia, 2007.

[16] F.D. Keukelaere, R.D. Sutter, and R.V.D. Walle, "MPEG-21 session mobility on mobile devices," International Confererence on Internet Computing, pages: 287-293, 2005.

[17] R. Shacham, H. Schulzrinne, S. Thakolsri, W. Kellerer, "Session Initiation Protocol (SIP) Session Mobility", Internet Draft, draft-shacham-sipping-session-mobility-02.txt, February 2006

[18] F. Nack and A.T. Lindsay, "Everything you wanted to know about MPEG-7 (part 1)," IEEE Multimedia, 6 (3), July-September 1999, pp. 65-77.

[19] F. Nack and A.T. Lindsay, "Everything you wanted to know about MPEG-7 (part 2)," IEEE Multimedia, 6 (4), October-December 1999, pp. 64-73.

[20] D.C.A. Bulterman and L. Rutledge, "SMIL 2.0: Interactive Multimedia for Web and Mobile Devices," Springer-Verlag, Heidelberg, 2004.

[21] M. Honkala and M. Pohja, "Multimodal Interaction with XForms," International Conference on Web Engineering (ICWE2006), 2006, pp. 201-208.

[22] F. Pereira and T. Ebrahimi, "The MPEG-4 Book," Prentice Hall, Upper Saddle River (NJ), 2002.

[23] R. Kernchen, M. Boussard, C. Hesselman, C. Villalonga, E. Clavier, A.V. Zhdanova, and P. Cesar, "Managing Personal Communication Environments in Next Generation Service Platforms," IST Mobile and Wireless Communications Summit, Budapest, 2007, pp. 1-5.

[24] S. Oviatt , A. DeAngeli, and K. Kuhn, "Integration and synchronization of input modes during multimodal human-computer interaction," SIGCHI conference on Human factors in computing systems, 1997, p.415-422.

[25] G. Klyne, et al., "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies", W3C Recommendation, 2004.

[26] P. Baggia, J. Carter, D.A. Dahl, G. McCobb, D. Raggett, "EMMA: Extensible MultiModal Annotation markup language," W3C Working Draft, April 2007.

[27] R.A. Bolt, "Put-that-there": Voice and gesture at the graphics interface, International Conference on Computer Graphics and Interactive Techniques, pp. 262-270, 1980.

[28] C. Hesselman, H. Eertink, and A. Peddemors, "Multimedia QoS Adaptation for Inter-tech Roaming", 6th IEEE Symposium on Computers and Communications (ISCC'01), Hammamet, Tunisia, July 2001

[29] H. van Kranenburg, M. S. Bargh, S. Iacob, A. Peddemors, "A Context Management Framework for Supporting Context-Aware Distributed Applications", IEEE Communications Magazine, August 2006, pp. 67-74