

# Strategies for Self-Organization and Multimodal Output Coordination in Distributed Device Environments

**Christian Elting**

European Media Lab GmbH  
Heidelberg, Germany  
Christian.Elting@eml-d.villa-bosch.de

**Michael Hellenschmidt**

Fraunhofer IGD Darmstadt  
Darmstadt, Germany  
Michael.Hellenschmidt@igd.fhg.de

## Abstract

One of the great research challenges currently facing the Ubicomp community is the development of mechanisms for groups of devices to dynamically coordinate with one another in meaningful ways. This paper presents results of the project DynAMITE, which develops mechanisms for self-organizing device environments. Based on the SodaPop middleware model for architectural integration we developed conflict resolution strategies for distributed device environments. The multimodal presentation strategy supporting graphical user interfaces, speech synthesizers and virtual characters is presented here as well as the open source implementation, which is available from the project web site.

## Keywords

Self-organization, ambient intelligence, ubiquitous computing, multimodal output presentation, conflict resolution strategies.

## INTRODUCTION

In the near future, a multitude of information appliances and smart artifacts will characterize everyone's personal environment. To make the vision of co-operative device ensembles, which enable proactive support of the user come true, coherent teamwork among the environment's appliances needs to be established. This is a main requirement to realize the visions of Ambient Intelligence [1,8,37]. The following scenarios illustrate the behaviour of smart devices within a mobile environment. Imaging a person in a living room watching TV while she uses a personal digital assistant (PDA) to browse the TV program. How should the device ensemble (TV set, loudspeakers, PDA) present the information to her? One way to do this might be to present a short introduction by speech on the loudspeakers, to render textual information about the movies on the PDA screen and to show stills of the movies on the screen of the TV set. It is not difficult to imagine such a scenario, which is able to deal with a fixed set of devices, situations and exceptions. But this system will only function properly only as long as the developers anticipate the device ensemble, which makes up the environment. Considering ubiquitous and mobile device scenarios this is not always possible. Imagine that the user is leaving her flat and she is taking her PDA with her. She might be entering an environment, which is entirely new to her (e.g. a lecture room or a railway station) and features previously unknown devices. In the lecture room the slides of the lesson given might be presented on the PDA of the user. And if the PDA owns loudspeakers, the speaker's voice might also be presented acoustically. In the railway station a ticket-selling machine might present its graphical user interface on the PDA of the user.

In all these environments, the PDA has to co-operate with the other devices in order to realize a coherent device ensemble.

Enabling this kind of ambient intelligence within mobile environments and the ability of devices to configure themselves into such a coherently acting ensemble, requires more than anticipating possible device combinations. Usually it is also impossible for the user to define the behaviour of device ensembles, because she would need detailed knowledge of the devices and their abilities. Moreover devices ensembles could be too large and the contexts of use too numerous to cover every situation by hand. Here software-infrastructures are needed, which are able to deal with dynamic changes in the device ensemble without requiring manual modification. Self-organizing mechanisms and conflict resolution strategies are necessary to establish co-operation between devices and to realize a coherent ensemble.

The project DynAMITE (Dynamic Adaptive Multimodal IT-Ensembles, [9]) faces the challenge to develop and implement software-infrastructures, which allow self-organization of ad-hoc appliance ensembles. To realize these challenges, DynAMITE covers different research areas within the field of ubiquitous computing and ambient intelligence:

- The development and application of a software infrastructure for self-organizing environments
- The development of ensemble topologies for Ambient Intelligence
- And the development of conflict resolution strategies for competitive devices

When looking at the challenges of self-organization of device ensembles, we can distinguish two different aspects of self-organization:

The *architectural integration* refers to the integration of a device into the ensemble, whose functionalities can be expressed by means of an ontology. E.g. headphones can be characterized as an acoustic output device, which makes it possible to route a lecturer's speech input to this device. The *operational integration* describes the aspect of integrating new functionality, which is unknown to the system and cannot be expressed by an existing ontology. This is also true for assembling a completely new functionality, which was not present on a single device (e.g. the copy functionality provided by means of two VCRs). See [17] for more details about operational integration.

DynAMITE focuses on the aspect of architectural integration. That means that device ensembles can be built from individual devices in an ad-hoc fashion (by the user or by the environment itself). Besides self-organizing mechanisms such ensembles need conflict resolution strategies to deal with competitive device situations. Furthermore device ensembles may change over time – due to hardware components entering or leaving the infrastructure

or due to changes in the quality-of-service available for some infrastructure services. Especially the coordination of different multimodal output devices is challenging due to the large set of possible output realizations (e.g. graphic user interface, synthesized speech or virtual character).

This paper is structured as follows: the next section gives a short overview about the SodaPop system, which we developed within the EMBASSI project [20] and which is basis for the development of the DynAMITE infrastructure. Then we explain how conflict resolution strategies are applied within a SodaPop device ensemble by means of utility value functions and describe our strategy to coordinate multimodal output devices within a dynamic mobile environment. The types of output devices we consider here are graphical user interfaces, speech syntheses and virtual characters. After that we describe our implementation and give an overview of the current version of our open source system, which is available from our project website [9] and give some examples of the dynamics of our multimodal output strategy. After the discussion of related work we summarize our results and give an outline of our next steps.

### SELF-ORGANIZATION IN DYNAMIC AGENT INFRASTRUCTURES

A central requirement for a software infrastructure supporting architectural integration is the support of ensembles, which are built from individual devices in an ad-hoc fashion (either manually by the user or automatically by the infrastructure). Therefore it must not rely on a central controller – any device must be able to operate stand-alone. Such an architecture has to meet the following objectives:

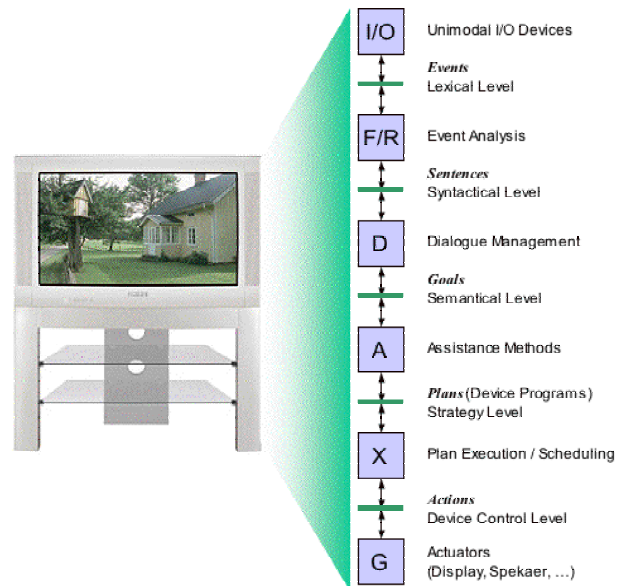
- ensure the independence of components
- allow dynamic extensibility by new components
- avoid central components (to prevent single point of failures or bottlenecks)
- support a distributed implementation
- allow flexible re-use of components
- and provide transparent service arbitration

To meet these challenges we developed the SodaPop (self-organizing data-flow architectures supporting ontology-based problem decomposition) middleware model for self-organizing infrastructures (for more details please refer to [16] and [19]). This section only outlines the basic principles of SodaPop for a common understanding of our ideas for self-organizing environments and explains the application of strategies for conflict resolution.

First we looked at the internal event processing pipeline of the physical devices we wanted to support [20]. In principle, they observe user input, interpret this input into device actions, change the environment and render output. Figure 1 illustrates this by means of an TV set. Physical user interactions are translated to events. Different components for interpreting these events determine the appropriate action to be performed, and finally the actuators are physically executing these actions. In SodaPop we call the components *transducers* and the interfaces between them *channels*.

The basic idea of SodaPop to bring more than one device together in order to turn private interfaces between processing stages into public channels. Thus it is possible to achieve an architectural

integration by introducing two fundamental organization levels: Coarse-grained self-organization based on data-flow partitioning (the classification of a device into different components dealing with different parts of the processing pipeline), and fine-grained self-organization for functionally similar components based on a kind of “pattern matching” approach (similar components are connected to the same channels). SodaPop does not rely on a specific data flow topology, but rather allows arbitrary topologies to be created ad-hoc from the components provided by the devices in an ensemble.



**Figure 1. A home entertainment appliance and its internal component structure: Here six transducers are present, which are connected by means of five channels.**

The discrimination of the topology into channels and transducers makes it also possible to apply different conflict resolution strategies to each channel. These channel strategies make the communication of components flexible and guarantee the extensibility of the system. In the following section we describe in detail the application of channel strategies by means of utility value functions and outline our approach for a conflict resolution strategy for self-organizing multimodal presentations within a dynamic mobile device ensemble.

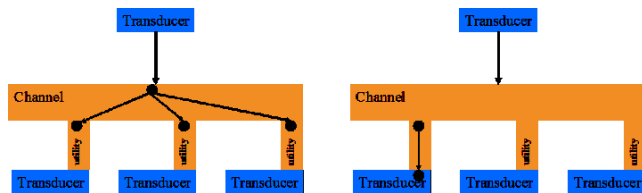
### CHANNEL STRATEGIES

In order to guarantee the dynamics of the device ensemble (i.e. the possibility to add and remove components and devices at runtime) the application of strategies for conflict resolution must not be hard-wired. In our living room scenario the user wants to inform herself about the today’s movies. We assume that the description of the movies is coming from an electronic program guide as text information. How should this text be presented to the user by means of the output functionalities of the devices, which are connected to the presentation channel? Of course it might be possible to implement a hard-wired channel strategy that covers all possible device configurations (e.g. presence of a PDA, loudspeakers busy with a movie, etc.). But what will happen if a complete new situation is encountered (e.g. the introduction of a second PDA)?

The next section explains the general channel-transducer architecture in DynAMITE and how utility value functions are used to solve this problem. After that we illustrate the MMO channel strategy. This strategy achieves the ad-hoc integration of new output transducers, which might be located on several output devices, into a coherent presentation.

### Channel-transducer model

A transducer, which subscribes to a channel, declares the set of messages it is able to process and how well it is suited for processing a certain message (for more details please refer to [19]). These aspects are described by the transducer's utility value function. A utility value function is a function that maps a message to an utility value, which encodes the subscriber's handling capabilities for the specific message. When a channel processes a message, it evaluates all utility value functions of the connected transducers and then decides – dependent from the channel strategy – which transducers will effectively receive the message (see Figure 2) or parts of the message for distributed problem solving.



**Figure 2. The evaluation mechanism of the utility value functions. A channel, which has to deliver an event, evaluates all utility value functions of the connected transducers (left picture) and then decides, according to the channel strategy, which transducer will receive the message (right picture).**

Both the transducers' utility and the channel's strategy are eventually based on the channel's ontology – the semantics of the message, which are communicated across the channel. The definition of these ontologies is possible, because we are offering solutions for the architectural integration of dynamic device ensembles. As Figure 1 indicates there are several possible channels within a device topology. Each channel is responsible for delivering certain types of messages to the connected transducers. It is obvious that on each stage of the processing pipeline different strategies might be necessary. For example user output events might be processed by every transducer, which is able to handle it. But some stages of the processing pipeline might make it necessary that only one transducer gets an event even though more than one is able to process it.

Thus, DynAMITE realized at first three simplistic channel strategies, to demonstrate the dynamic abilities of SodaPop. As there are:

- The *all*-strategy, which forwards an event to all those components, whose utility value functions give back an positive value.
- The *random*-strategy forwards the event to only one of the components whose utility value function is evaluated positive. This is determined by a random mechanism.
- The *best*-strategy selects the component, whose utility value function is most appropriate with respect to the task associated with the event (for more details please refer to [19]).

The strategy we present in this paper implements an approach for problem decomposition within the domain of multimodal presentation planning. Problem decomposition stands for the decomposition of an event into multiple events, which are processed by different components cooperatively.

### Multimodal output coordination strategy

System output to the user can be presented by means of different output devices (e.g. displays or speakers) as well as different output modalities (e.g. pictures or speech). In ubiquitous computing scenarios the set of output devices and available rendering components can change dynamically. The strategy we present here is able to include new output devices and rendering agents into the presentation planning process. The types we are supporting are graphical user interfaces (GUIs), virtual characters and speech synthesizer components. The strategy achieves the decomposition of a single output event to multiple events for each rendering component, which are distributed among multiple devices. We call this strategy *MMO* (multimodal output coordination) strategy.

Imagine a situation, in which a user is working on a desktop PC. System output is currently realized by a GUI-type rendering component and a speech synthesis-type rendering component, which are rendering output on the PC monitor resp. speakers. Both components are connected to a channel implementing the MMO strategy. However the user wants to augment the system by means of a new rendering component, which realizes an animated character. Therefore the user starts a third rendering component on a laptop. This component connects to the same channel as the GUI and the speech components.

After the channel receives an output event (e.g. from a dialog managing component), the MMO strategy evaluates the utility value functions of all connected rendering agents. The utility values contain information about the type of the rendering component (e.g. character or GUI) as well as its current state (e.g. its location, width and size).

Therefore the MMO strategy handler realizes that a new rendering component has been connected, which realizes an animated character consisting of lip movements, head movements, gestures and speech. Due to the fact that both components contain speech output the richer output of the character component is preferred over the speech synthesis component located on the PC. Moreover as the GUI and the character are not located on the same device no layout coordination needs to take place. As a result system output is now realized by means of the GUI on the PC and the animated character on the laptop omitting the speech synthesis.

We use the rendering component profiles to describe the functionality and restrictions of rendering components concerning the multimodal output, which they synthesize. This allows the automatic inclusion of previously unseen rendering components into a presentation at run-time. These profiles are based on Bernsen's model of output unimodalities [7]. Our model of rendering component profiles is described in detail in [10].

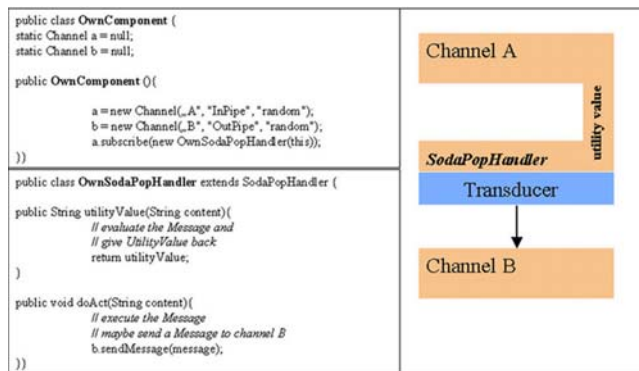
In this work we focus on three different types of output components: virtual characters, speech syntheses and graphical user interfaces. However of each type of component an arbitrary number of instances can be started, which may be distributed among different devices (like PDAs, laptops or desktop PCs).

For the execution of the MMO strategy we use the presentation planning tool described in [5]. This planner is the core of the MMO strategy. The software uses a hierarchical AI planning approach to decompose a complex presentation goal into primitive presentation acts. The primitive acts send rendering messages to a set of rendering components together with instantiated parameters for each component (e.g. screen location or emotions to be expressed). After that each rendering component displays the content to be rendered accordingly.

## IMPLEMENTATION

In this section we give an overview of our current implementation of the SodaPop Framework and its channel strategies.

The SodaPop-framework as well as the Java API and the demonstrators are freely available from the DynAMITE website [9]. The API allows the implementation of own transducers and channels, as well as the application of the introduced channel strategies. Thus the realization of topologies as illustrated in Figure 1 becomes possible.

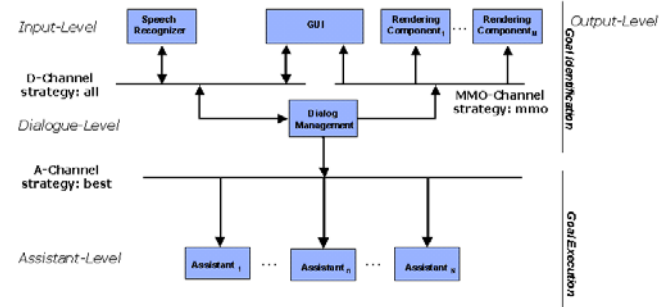


**Figure 3. A transducer consists of two classes: the transducer itself and a SodaPopHandler, which handles the utility value evaluation and the final execution of a message. The picture on the right shows what physically happens.**

Figure 3 illustrates some details of the SodaPop Java API. First a transducer instantiates the channels it wants to connect to. A channel description consists of a channel name and a parameter indicating whether the transducer only listens to messages on this channel or also wants to send events to it. Finally the channel strategy name is given. The transducer in the example illustrated in Figure 3 is connected to two channels, whereas both channels have the strategy “random”. Because the transducer is connected to channel “A” as a listener, a SodaPopHandler must be instantiated that handles the evaluation of the utility value function (Figure 3 top left). The SodaPopHandler handles also the execution of a message, if the channel strategy decides that this transducer will receive the event for further processing (Figure 3 bottom left).

By means of this API a demonstrator was built, especially illustrating the dynamics of the multimodal presentation components. This demonstrator realizes a home entertainment scenario. The topology of the demonstrator is illustrated in Figure 4. The topology implements the main channels of our generic appliance topology as illustrated in Figure 1. The user is able to interact with this environment either by speech or by graphical user interfaces to define movies or pieces of music that should be played by means of MP3-player-applications or MPEG-player-applications. Its topology introduces three different channels. The dialog channel (the D-channel in Figure 4 top left) communicates

user interaction events (user’s speech or graphical user interface events). The assistant channel receives the task for playing a movie of music and transfers it to the most appropriate MPEG-player or MP3-player assistant (Figure 4 bottom). Finally the multimodal output channel (Figure 4 top right) realizes the coordination of the different rendering components to map the amodal text information into different multimodal output representations. As illustrated in one of the previous sections its amount of output components is unlimited. There is also no restriction concerning the combination of the different output components.



**Figure 4. The example topology we provide together with our framework. The most appropriate assistant is chosen by means of the best-strategy, while the decomposition of rendering events is done by the MMO-strategy.**

To demonstrate the dynamic abilities of the SodaPop model and the channel strategies we implemented some components:

- a ViaVoice-based speech recogniser [39] to recognize user utterances
- a graphical user interface
- a dialog management component, which controls dialogs with the user
- a speech synthesis rendering component
- a virtual character rendering component
- two MPEG-player applications with different device properties
- a MP3-player application

If the user wants to watch a movie, she simple says “I want to see a movie please”, or if available presses the corresponding button of the graphical user interface. Interpreting this sentence, the dialog management component starts a disambiguation dialog with the user. Every system output is triggered by means of an output event, which is sent by the dialog management to the multimodal output channel. As soon as a movie (or a piece of music) is defined, the dialog management gives the task to the assistant channel, where it is conducted by one of the different available player applications.

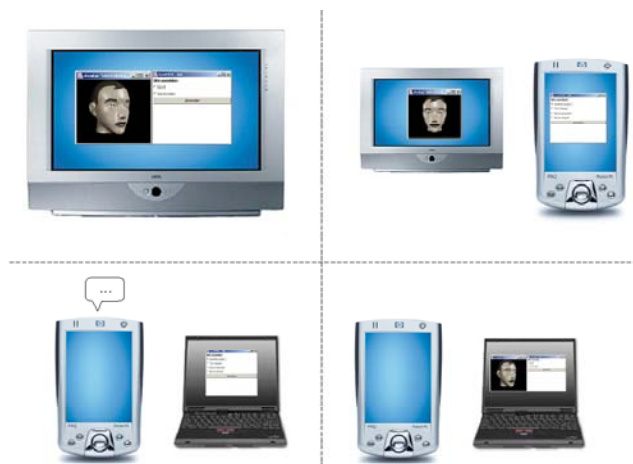
The graphics part of the virtual character component [2] is implemented in Java 3D. The character synchronizes its lip movements with speech generated by the MBrola speech synthesis [30]. The graphic user interface component consists of a Java GUI, in which the user can make selections by means of a choice box. The speech synthesis component uses an IBM Via Voice implementation of the Java speech API [39]. We also implemented a second type of speech synthesis, which is based on MBrola. This integration of an additional speech synthesis

component shows the modularity of the demonstrator concerning the integration of new rendering components.

Please note that the application of only one dialog management component is no restriction of the introduced principles of dynamic self-organizing device environments. The application of channel strategies makes it able to apply as many dialog management components as needed. We decided to apply only one dialog management component to keep the demonstrator as simple as possible. Thus one is able to concentrate on the applicability of the different channel strategies.

## EXAMPLES

In this section we provide examples of the application for the MMO channel strategy in our implementation. We start with a virtual character rendering component and a graphical user interface running on the same device (Figure 5, top row, left picture). Together they instantiate a channel, which applies the MMO-strategy (cf. MMO channel in Figure 4).



**Figure 5. Examples for the coordination of different output components running on different physical devices. The coordination is done by the MMO-channel strategy.**

The window of the animated character is adapted according to the size and location of the GUI window. After the evaluation of all utility values functions the MMO strategy knows by means of the rendering component profile that the character is able to look at the GUI, which is also executed.

However the user rather prefers to access the system by means of a PDA and starts a GUI component and a speech synthesis component on it (Figure 5, top row, right picture). Together they extend the previously existing MMO-channel. The GUI is still hidden until the MMO strategy decides to pop it up. With the next presentation task, the presentation planning strategy takes the utility value of the newly connected graphical user component into concern. The utility value indicates that the user has chosen the GUI on the PDA for processing an input (by means of a choice box). Therefore the strategy hides the other GUI on the TV screen assuming that the users rather want to work on the PDA than on the TV screen. The GUI on the PDA is also adapted according to the resolution of the PDA screen. The character on the TV screen is now looking at the user instead of the GUI, which is not visible on the TV screen anymore. The speech synthesis on the PDA is not used as the richer graphical-acoustical output of the character is preferred.

After that the user switches off the TV and boots a laptop, because it contains some movies, which she did not find on the TV set system (Figure 5, bottom row, left picture). On the laptop only a GUI rendering component is present. After the GUI on the laptop is used for input the GUI switches again from PDA to laptop. However the speech synthesis contained on the PDA is used to augment the GUI output on the laptop. However the user prefers a richer output and installs the software for the animated character on the laptop. In order to prevent two speech renderings at once (by means of the speech synthesis as well as by the animated character) the speech synthesis on the PDA is switched off, a layout coordination between the character and the GUI is conducted and the user achieves the desired result (Figure 5, bottom row, right picture).

## RELATED WORK

Research in the field of self-organizing device ensembles covers both co-operation and communication mechanisms of agent platform technologies and coordination strategies for multimodal output. We are especially interested in approaches for assisting the user while dealing with unknown device ensembles. Concerning the automated generation of multimodal presentations a lot of research has been conducted focusing on single-device scenarios. We give an overview over known approaches with respect to problem decomposition and point out differences to multi-device environments.

### Self-organization of device ensembles

In the past, many initiatives presented concepts for managing agent communication processes or addressed the problem of dynamic, self-organizing systems. The KQML [26] and FIPA [12] agent communication languages provide powerful methods for defining communication acts, as well as methods for service discovery. But the deployment within high dynamic ensembles is not possible. KQML as well as FIPA are using central routing components for delivering messages from agent to agent or a hard-wired peer-to-peer communication mechanism. Furthermore service discovery can be done by another heavy weight component, called yellow-page-service or facilitator, which allows the subscription for events as well as appropriate agents offering special services. A mechanism for conflict resolution does not exist. There is no solution if the service discovery function offers more than one possible address component. The Open Agent Architecture (OAA) (see [27] and [35]) and the Galaxy Communicator Architecture [13] provide architectures for multi-agent systems supporting multimodal interaction. Whereas Galaxy uses a centralized hub-component, which uses routing rules for modeling how messages are transferred between different system components, the OAA uses a special meta-agent with Prolog-based decomposition and recombination strategies. Both approaches have advantages as well as disadvantages. The routing rules of Galaxy as well as the strategies implemented within the OAA meta agents make dynamic sampling of non-static ensembles very difficult. Rules have to be adapted each time the configuration of the ensemble changes. On the other hand, on top of Galaxy and OAA systems can be built, which behave predictably in every possible situation. Galaxy as well as the OAA do not support mechanisms for structuring a component topology, which means that a designer has to know the whole system (and its behavior rules) and all interfaces, which connect the different components.

There are other research approaches (e.g. the Task Computing initiative [28]) supporting task aware computing, which bear

some resemblance to the ideas of activity based computing, which Don Norman described in 1976 [34]. Also the concept of situation-aware assistance realizes more assistance for the user and disburdens her from the responsibility to control every task and device separately. Here, Kirste [22] and Heider [17] describe the principles that are used to develop the EMBASSI architecture [20] and are the starting point for our self-organizing software infrastructure SodaPop. SodaPop is described in detail in Kirste [16] and Hellenschmidt [19]. The Speakeasy project that makes device and service interfaces available to users on multiple platforms [32] seems to be a similar approach as the task computing initiative. But the basic principle is very different to the DynAMITE approach. Whereas Speakeasy and Task Computing make the control of all devices (and their services) available for the user – and thus give all responsibilities to her – the project DynAMITE, and the work we present here, goes one step ahead. SodaPop, the software infrastructure of DynAMITE realizes self-organizing device ensembles that offer conflict resolution strategies to make the co-operation of devices possible without the need of an engagement of the user.

### Multimodal output coordination

There has been a lot of research in the past focusing on the generation of multimodal presentations on a single output device. Classic presentation planning systems described in [3], [11], [21] or [40] dealt with static output device environments, which were not supposed to change during a system run. Especially presentation planning in multimodal dialog systems is still restricted to a single output device. The presentation planning system described in [31] generates and coordinates multimodal output consisting of an animated character, pictures, maps and speech synthesis. The output is adapted to different display sizes, however the system does not support more than one output device.

Moreover all of these approaches rely on a central presentation planning component, which is physically located on a certain device. However for ubiquitous computing scenarios, this is a setback. If the user switches off the device, on which this component is located, then it is not possible anymore to plan a presentation. Nowadays systems begin to emerge, which also take the dynamic addition of new output devices and services into account. This is crucial for ubiquitous computing environments, in which the device environment frequently changes.

The Berlin subway system is playing Macromedia Flash adverts, which are presented synchronously on two displays in parallel without using sound [6]. A user study showed 82% approval of the system, which shows that coherent multi-device presentations can result in appealing presentations. However the system uses a static output device environment, both screens have the same size and are adjacent to each other, which needs not necessarily to be true in other ubiquitous computing environments.

Robertson [36] built a prototype system for combining a PDA with a TV set. In this system the TV screen is used to add additional information to the graphic PDA output like pictures or maps. This system also relies on a static device environment. Han [15] describes the WebSplitter system, which is able to split a multimedia HTML document among available output devices like a laptop, a PDA or speakers. However the distribution of the content among the devices has to be done manually by the user. Kruppa and Krüger [24] investigate interaction paradigms between PDAs and large displays. Kray [23] presents an architecture for presentation planning in a multi-device and multi-

user environment. A central planning component generates and synchronizes SMIL [38] presentations on multiple displays and speakers. However it is not possible to take care of different presentation functionalities available on each device e.g., playing a movie might be possible on a desktop computer, but might not be possible on a PDA.

In multi-device environments teams of animated characters can be employed to mediate between devices [25]. In the system presented in [14] three avatars are split across two screens. André and Rist [4] also showed that the use of presentation teams results in believable dialogs.

In order to be able to dynamically include new devices containing new rendering agents (e.g. a speech synthesis agent on a laptop) into a multi-device presentation, the new rendering agents have to identify themselves to the rest of the system by means of a service description. Ideally it should be possible to describe any rendering agent type by means of this description (e.g. movie player agents or speech synthesis agents). The model presented in [7] contains a complete and unique classification of all basic output modalities within the media of acoustics, graphics and haptics.

### CONCLUSION

In this work we presented the results of the project DynAMITE (Dynamic Adaptive Multimodal IT-Ensembles). We introduced our implementation of the SodaPop middleware model for self-organizing data-flow architectures that separates a component topology into channels and transducers. The goal of DynAMITE is the architectural integration of new components, which might be located on mobile devices. We illustrated the four conflict resolution strategies for competing components provided by our framework. One of these strategies, the multimodal presentation strategy, coordinates presentation components of three types: graphical user interfaces, virtual characters and speech synthesizer. This strategy makes the application of an unlimited number of output components running on different devices possible. The advantages of our approach is the absence of a central component or a hard-wired intelligence by means of enabling self-organizing topologies and the application conflict resolution strategies based on the evaluation of utility value functions.

We introduced our demonstrator, which can be downloaded from our project web site. The demonstrator realizes a home entertainment application illustrating the applicability of our approach. The next steps of our work in the DynAMITE project include the definition of basic component topologies usable as a blueprint for advanced scenarios as well as the definition of common vocabularies and ontologies for describing services and devices (e.g. UPnP). In previous work we already investigated the use of user profiles and output preferences within respect to multimodal environments [18]. We intend to include similar preference models into the strategies of DynAMITE system.

We also intend to evaluate our approach under real-life conditions. Here we intend to investigate issues related to the usability of our system. A lot of research has been conducted, which covers the cognitive effects of multimodal presentations on a single device. However the effects of multi-device presentations still remain to be investigated. Moreover in Ubicomp systems, where computers disappear, there is a danger that users feel lost and cannot exploit the system's functionality [33]. We will investigate this in relation to different types of interface paradigms for Ubicomp home entertainment systems [29].

## ACKNOWLEDGEMENTS

The work presented in this paper was funded by the German ministry for education and research under the grants 01 IS C27 A and 01 IS C27 B as well as by the Klaus Tschira foundation.

## REFERENCES

- [1] Aarts E.: Ambient Intelligence: A multimedia Perspective, IEEE Multimedia (2004), 12-19.
- [2] Alexa M., Berner U., Hellenschmidt M., Rieger T.: An Animation System for User Interface Agents. Proceedings WSCG 2001, The 9<sup>th</sup> International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2001
- [3] André E., Müller J., Rist T.: WIP/PPP: Automatic Generation of Personalized Multimedia Presentations, 4th ACM International Multimedia Conference, Boston, MA, November 1996.
- [4] André E., Rist T.: Presenting Through Performing: On the Use of Multiple Lifelike Characters in Knowledge-Based Presentation Systems, Int. Conf. on Intelligent User Interfaces IUI 2000, New Orleans, LA, 2000.
- [5] André E., Baldes S., Kleinbauer T., Rist T.: CkuCkuCk 1.01 Planning Multimedia Presentations - User Manual and Installation Guide, available from <http://www.dfki.de/imedia/miau/software/CkuCkuCk/manual/manual.html>, 2000.
- [6] Berliner Fenster – Das Fahrgastfernsehen, <http://www.berliner-fenster.de>, 2004.
- [7] Bernsen, N. O.: Multimodality in Language and Speech Systems - From Theory to Design Support Tool, in: Granstrom (ed.), Multimodality in Language and Speech Systems, Kluwer Academic Publishers, 2001.
- [8] Ducatel K., Bogdanowicz M., Scapolo F., Leijten J., Burgelman J.-C.: Scenarios for Ambient Intelligence 2010, ISTAG report, European Commission, Institute for Prospective Technological Studies, Seville, Nov. 2001.
- [9] DynAMITE, <http://www.dynamite-project.org>, 2004.
- [10] Elting C., Möhler G.: Modeling Output in the EMBASSI Multimodal Dialog System, Int. Conf. on Multimodal Interfaces ICMI02, Pittsburgh, PA, October 14-16, 2002.
- [11] Feiner S. K., McKeown K. R.: Automating the Generation of Coordinated Multimedia Explanations, in: Mark Maybury & Wolfgang Wahlster (eds.), Readings in Intelligent User Interfaces, pp. 89-97, Morgan Kaufman Publishers, 1998.
- [12] FIPA, The Foundation for Intelligent Physical Agents, <http://www.fipa.org>, 2002.
- [13] Galaxy Communicator Infrastructure, Project Homepage, available from <http://sls.csail.mit.edu/sls/technologies/galaxy.shtml>, 2001.
- [14] Gebhard P., Kipp M., Klesen M., Rist T.: What Are They Going to Talk About? Towards Life-Like Characters that Reflect on Interactions with Users, Int. Conf. on Technologies for Interactive Digital Storytelling and Entertainment TIDSE'03, Darmstadt, Germany, March 24-26, 2003.
- [15] Han R., Perret V., Naghshineh M.: WebSplitter: A Unified XML Framework for Multi-Device Collaborative Web Browsing, ACM Conference on Supported Collaborative Work CSCW, Philadelphia, PA, 2000.
- [16] Heider Th., Kirste Th.: Architecture Consideration for interoperable multi-modal assistant systems, In: Forbig, Peter (Ed.). Interactive Systems, Design, Specification, and Verification, Proceedings, Berlin, Heidelberg, Springer International, 2002, pp. 253 – 267.
- [17] Heider Th. Kirste Th.: Supporting goal-based interaction within dynamic intelligent environments, Proc. 15<sup>th</sup> European Conference on Artificial Intelligence (ECAI 2002), Lyon, France, 2002.
- [18] Hellenschmidt M., Kirste Th, Rieger Th.: An Agent Based Approach to Distributed User Profile Management within a Multi-Modal Environment, IMC Workshop 2003 Proceedings: Assistance, Mobility and Applications Stuttgart, Fraunhofer IRB Verlag 2003, p. 129-135.
- [19] Hellenschmidt M., Kirste Th.: SodaPop: A software infrastructure supporting self-organization in Intelligent Environments, 2<sup>nd</sup> IEEE Conference on Industrial Informatics, INDIN 04, Berlin, Germany, June, 2004.
- [20] Herfet Th., Kirste Th. and Schnaider M.: EMBASSI-Multimodal Assistance for Infotainment and Service Infrastructures, In: Computers & Graphics 25, 4, pp. 581 – 592, 2001.
- [21] Kerpedjiev S., Carenini G., Roth S, Moore J.: Integrating Planning and Task-based Design for Multi-media Presentation, International Conference on Intelligent User Interfaces IUI'97, pp.145-152, Orlando, FL, 1997.
- [22] Kirste Th.: Situation-Aware Mobile Assistance, Proc. 1<sup>st</sup> EC/MSF Advanced Research Workshop, Bonas, France, June, 1999.
- [23] Kray C., Krüger A., Endres C.: Some Issues on Presentations in Intelligent Environments, European Symposium on Ambient Intelligence, Eindhoven, Netherlands, Nov. 3-4, 2003.
- [24] Kruppa M., Krüger A.: Concepts for a Combined Use of Personal Digital Assistants and Large Remote Displays, Proceedings of SimVis 2003, Magdeburg, March 2003.
- [25] Kruppa M.: The better remote control – Multiuser interaction with public displays, Workshop on Multi-User and Ubiquitous User Interfaces (MU3I), January 13, 2004.
- [26] Labrou Y., Finin T., A Proposal for a new KQML Specification, TR CS-97-03, Computer Science and Electrical Engineering Department, University of Maryland Baltimore County, Baltimore, MD 21250, February 1997.
- [27] Martin D.L., Cheyer A.J., Moran D.B.: The Open Agent Architecture: A Framework for Building Distributed Software Systems. Applied Artificial Intelligence, Vol 13., 20. 1-2, pp. 91 – 128, January- March 1999.
- [28] Masuoka R., Parsia B., Labrou Y.: Task-Computing – the Semantic Web meets Pervasive Computing, Proc. Of the 2<sup>nd</sup> International Semantic Web Conference (ISWC), Sanibel Island, Florida, USA, October, 2003.

- [29] Meyer zu Kniendorf, Ch.: Interaktionskonzepte in verteilten und vernetzten Systemen am Beispiel der Unterhaltungselektronik, 5. Berliner Werkstatt Mensch-Maschine-Systeme. ZMMS Spektrum, Band 18. Auszug aus Fortschritt-Berichte VDI, Reihe 22, Nr. 16, S. 488- 506, Düsseldorf: VDI Verlag GmbH, 2004.
- [30] MBROLA – Towards a Freely Available Multilingual Synthesizer, Faculté Polytechnique de Mons, MULTITEL-TCTS Lab, available from <http://tcts.fpms.ac.be/synthesis/mbrola.html>, Mons, Belgium, 1999.
- [31] Müller J., Poller P., Tschernomas V.: Situated Delegation-Oriented Multimodal Presentation in SmartKom, Workshop Intelligent Situation-Aware Media and Presentations ISAMP, Edmonton, 2002.
- [32] Newman M.W., Izadi S., Edwards W.K., Sedivy J.Z., Smith T.F.: User Interfaces When and Where They are needed: An Infrastructure for Recombinant Computing, Proceedings of UIST 02, Paris, France, 2002.
- [33] Nijholt, A., Rist, T., Tuijnjenbreijer, K., Lost in Ambient Intelligence?, CHI2004 Workshop: Lost in Ambient Intelligence?, Vienna, Austria, April 25, 2004.
- [34] Norman D.A.: The Psychology of Everyday Things, New York, Basic Book, first edition, 1976.
- [35] Open Agent Architecture, <http://www.ai.sri.com/~oaa/>, 2001.
- [36] Robertson S., Wharton C., Ashworth C., Franzke M.: Dual Device User Interface Design: PDAs and Interactive Television, CHI, Vancouver, British Columbia, Canada, 1996.
- [37] Shadbolt N.: Ambient Intelligence, IEEE Intelligent Systems, 2-3, 2003.
- [38] SMIL, W3C Recommendation: Synchronized Multimedia Integration Language (SMIL 2.0), 2001.
- [39] Via Voice, IBM, <http://www.ibm.com/software/voice/viavoice/>
- [40] Zhou M. X., Feiner S. K.: Efficiently Planning Coherent Visual Discourse, Journal of Knowledge-based Systems, 10(5) pp. 275-286, 1998.