Ambient Intelligence: Towards Smart Appliance Ensembles*

Thomas Kirste
December 2004

Dept. Computer Science, Rostock University, Germany
tk@informatik.uni-rostock.de

Abstract. The vision of Ambient Intelligence is based on the ubiquity of information technology, the presence of computation, communication, and sensorial capabilities in an unlimited abundance of everyday appliances and environments. Today's experimental smart environments are carefully designed by hand, but future ambient intelligent infrastructures must be able to configure themselves from the available components in order to be effective in the real world.
We argue that enabling an ensemble of devices to spontaneously act and cooperate coherently requires software technologies that support self-organization. We discuss the central issues pertaining to the self-organization of interactive appliance ensembles and outline potential solution paradigms: Goal-based interaction and distributed event processing pipelines.

1 Introduction

The vision of Ambient Intelligence (AmI) [1, 3, 10] is based on the ubiquity of information technology, the presence of computation, communication, and sensorial capabilities in an unlimited abundance of everyday appliances and environments.

"Ambient Intelligence", a term coined by the European Commission's Information Technologies Advisory Group (ISTAG) and Philips, is the vision of a world, in which we are surrounded by smart, intuitively operated devices that help us to organize, structure, and master our everyday life. The notion "Ambient Intelligence" specifically characterizes a new paradigm for the interaction between a person and his everyday environment: Ambient Intelligence enables this environment to become aware of the human that interacts with it, his goals and needs. So it is possible to assist the human proactively in performing his activities and reaching his goals.—If my car stereo tunes in to exactly the station I just listened to at the breakfast table, then this is a simple example for such an aware, pro-active environment; just as the mobile phone that automatically redirects calls to my voice mail in case I am in a meeting, or the bathroom mirror that reminds me of taking my medications.

Hitherto, it is the user's responsibility to manage his personal environment, to operate and control the various appliances and devices available for his support. But, the more technology is available and the more options there are, the greater is the challenge

* This work has been partially supported by the German Federal Ministry of Education and Research under the grant signature BMB-F No. FKZ 01 ISC 27A.

Fig. 1. Environments we'd like to be smart: Conference rooms (left) and living rooms (right)

to master your everyday environment, the challenge not to get lost in an abundance of possibilities. Failing to address this challenge adequately simply results in technology becoming inoperable, effectively useless. Through Ambient Intelligence, the environment gains the capability to take over this mechanic and monotonous control task from the user and manage appliance activities on his behalf. By this, the environment's full assistive potential can be mobilized for the user, tailored to his individual goals and needs.

Technical foundation of Ambient Intelligence is *Ubiquitous* resp. *Pervasive Computing*: the diffusion of information technology into all appliances and objects of the everyday life, based on miniaturized and low cost hardware. In the near future, a multitude of such "information appliances" and "smart artifacts" will populate everyone's personal environment. In order to make the vision of Ambient Intelligence come true, a coherent teamwork between the environment's appliances has to be established that enables a co-operative, proactive support of the user. Wireless ad-hoc networking and embedded sensors provide the basis for coherent and coordinated action of an appliance ensemble with respect to the current situation. By enabling multi-modal interaction—such as speech and gestures—an intuitive interaction becomes possible. On top of this, strategies, models, and technologies for the self-organization of appliance ensembles are required that allow an adaptation to the user's needs and desires.

Clearly, Ambient Intelligence covers more aspects then just supporting the interaction of a human being with its environment. In general, any "active" entity engaging in situated behavior can benefit from a responsive environment, which provides smart assistance for this behavior – this holds for human beings, for livestock, for robots, as well as for smart goods (in logistics and smart factory applications). However, within the scope of this paper, we will concentrate on a specific sub-area of Ambient Intelligence: the software infrastructure needed for supporting user interaction with smart environments.

2 A Scenario ...

A rather popular scenario illustrating this application area is the *smart conference room* (or *smart living room*, for consumer-oriented projects, see Figure 1) that automatically

Fig. 2. Appliance Ensembles: physical constituents (left), and compound ad-hoc ensemble (right)

adapts to the activities of its current occupants (cf. e.g. [2, 5, 9, 11]). Such a room might, for instance, automatically switch the projector to the current lecturer's presentation as she approaches the speaker's desk[1], and subdue the room lights—turning them up again for the discussion. Of course, we expect the environment to automatically fetch the presentation from the lecturer's notebook.

Such a scenario doesn't sound too difficult, it can readily be constructed from common hardware available today, and, using pressure sensors and RFID tagging, doesn't even require expensive cameras and difficult image analysis to detect who is currently at the speaker's desk. Setting up the application software for this scenario that drives the environment's devices in response to sensor signals doesn't present a major hurdle either. So it seems as if Ambient Intelligence is rather well understood, as far as information technology is concerned. Details like image and speech recognition, as well as natural dialogues, of course need further research, but building smart environments from components is *technologically* straightforward, once we understand what kind of proactivity users will expect and accept.

3 … and its Implications

But only as long as the device ensembles that make up the environment are anticipated by the developers. Today's smart environments in the various research labs are usually built from devices and components whose functionality is known to the developer. So, all possible interactions between devices can be considered in advance and suitable adaptation strategies for coping with changing ensembles can be defined. When looking at the underlying software infrastructure, we see that the interaction between the different devices, the "intelligence", has been carefully handcrafted by the software engineers, which have built this scenario. This means: significant (i.e. unforeseen) changes of the ensemble require a manual modification of the smart environment's control application.

This is obviously out of the question for real world applications, where people continuously buy new devices for embellishing their home. And it is a severe cost factor for institutional operators of professional media infrastructures such as conference rooms

[1] For the smart living room this reads: "switch the TV set to the user's favorite show, as he takes seat on the sofa."

and smart offices. Things can be even more challenging: imagine a typical ad hoc meeting, where some people meet at a perfectly average room. All attendants bring notebook computers, at least one brings a projector, and the room has some light controls. Of course, all devices will be accessible by wireless networks. So it would be possible for this chance ensemble to provide the same assistance as the deliberate smart conference room above. Enabling *this* kind of Ambient Intelligence, the ability of devices to configure themselves into a coherently acting ensemble, requires more than setting up a control application in advance. Here, we need software infrastructures that allow a true self-organization of ad-hoc appliance ensembles, with the ability to afford non-trivial changes to the ensemble. (See also [12] for a similar viewpoint on this topic.)

Besides providing the middleware facilities for service discovery and communication, such a software infrastructure also has to identify the set of fundamental interfaces that characterize the standard event processing topology to be followed in all possible ensembles. This *standard topology* is the foundation for an appliance to be able to smoothly integrate itself into different ensembles: In a conference room, the user's notebook may automatically connect to a projector and deliver the user's presentation, while it will hook up to the hi-fi system and deliver an MP3 playlist when arriving back home.

When looking at the challenges of self-organization indicated above, we can distinguish two different aspects here:

Architectonic Integration – refers to the integration of the device into the communication patterns of the ensemble. For instance, the attachment of an input device to the ensemble's interaction event bus.

Operational Integration – describes the aspect of making new functionality provided by the device (or emerging from the extended ensemble) available to the user. For instance, if you connect a CD player to an ensemble containing a CD recorder, the capability of "copying" will now emerge in this ensemble.

Clearly, both aspects eventually have to be accounted for by an "Ambient Intelligence Software Architecture". In this paper, we will concentrate on the aspect of architectonic integration.

4 Appliances and Event Processing Pipelines

When developing at a middleware concept, it is important to look at the communication patterns of the objects that are to be supported by this middleware. For smart environments, we need to look at *physical devices*, which have at least *one* connection to the physical environment they are placed in: they observe user input, or they are able to change the environment (e.g. by increasing the light level, by rendering a medium, etc.), or both. When looking at the event processing in such devices, we may observe a specific *event processing pipeline*, as outlined in Figure 3: Devices have a User Interface component that translates physical user interactions to events, the Control Application is responsible for determining the appropriate action to be performed in response to this event, and finally the Actuators are physically executing these actions. It seems reasonable to assume that *all* devices employ a similar event processing pipeline (even if

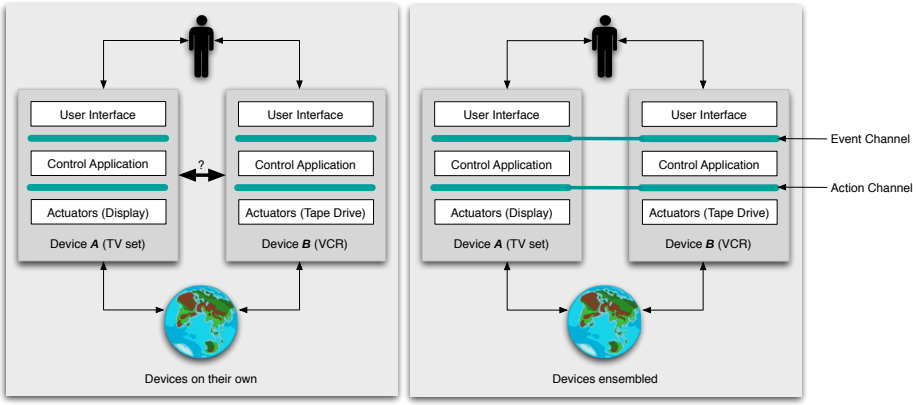

Fig. 3. Devices and Data Flows

certain stages are implemented trivially, being just a wire connecting the switch to the light bulb).

It would then be interesting to extend the *interfaces* between the individual processing stages across multiple devices, as outlined in the right side of Figure 3. This would allow a dialogue component of one device to see the input events of other devices, or it would enable a particularly clever control application to drive the actuators provided by other devices. By turning the private interfaces between the processing stages in a device into public *channels*, we observe that the event processing pipeline is implemented cooperatively by the device ensemble on a per-stage level. Each pipeline stage is realized through the cooperation of the respective local functionalities contributed by the members of the current ensemble.

So, our proposal for solving the challenge of architectonic integration is to provide a middleware concept that provides the essential communication patterns of such data-flow based multi-component architectures. Note that the *channels* outlined in Figure 3 are not the complete story. Much more elaborate data processing pipelines can easily be developed (such as outlined in [7]). Therefore, the point of such a middleware concept is not to fix a *specific* data flow topology, but rather to allow *arbitrary* such topologies to be created ad hoc from the components provided by the devices in an ensemble.

In the next section, we will look one additional channel of particular importance for interaction with ad-hoc ensembles: the *goal* channel.

5 Goal-Based Interaction

When interacting with appliances in everyday settings, we are used to think of interaction in terms of the individual “functions” these devices provide: functions such as “on”, “off”, “play”, “record”, etc.. When interacting with devices, we select, parameterize, and then execute functions these devices provide. When these functions are executed, they cause an effect: a broadcast is recorded on videotape, the light is turned brighter, and so on.

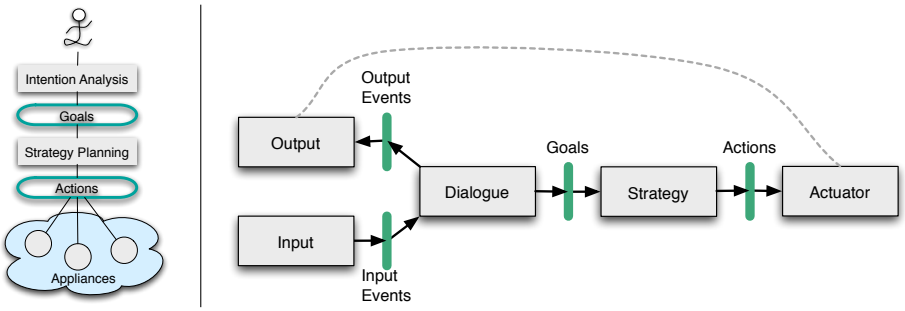

Fig. 4. Principle of goal based interaction (left); location in the appliance pipeline (right)

Of course, different devices have different functions, similar functions in different devices behave differently, and staying on top of all features is not altogether easy. So, interaction with devices is usually not intuitive and straightforward—as anybody trying to coax an unfamiliar projector into adjusting his contrast or programming a new VCR will probably acknowledge. Such activities can get very much in the way and interfere massively with user's foreground task, such as giving a lecture or enjoying a show on TV.

The proliferation of computational capabilities and the advent of ad-hoc ensembles will not make things easier. On the contrary: interesting devices in an ad-hoc ensemble may be completely invisible to a user, such as a rear-projection facility in an unfamiliar conference venue. Now: how is a user expected to interact with components and devices he is not even *aware* of (even *if* he knew, how to operate them . . .)?

But then, a user is not really interested in the *function* he needs to execute on a device – it is rather the function's *effect* which is important.

This observation immediately leads to the basic idea of goal-based interaction. Rather than requiring the user to invent a sequence of actions that will produce a desired effect ("goal") based on the given devices and their capabilities, we should allow the user to specify just the goal ("I want to see 'Chinatown' now!") and have the *ensemble* fill in the sequence of actions leading to this goal (Find the media source containing media event "Chinatown". Turn on the TV set. Turn on the media player—*e. g.*, a VCR. Position the media source to the start of the media event. Make sure the air condition is set to a comfortable temperature. Find out the ambient noise level and set the volume to a suitable level. Set ambient light to a suitable brightness. Set the TV input channel to VCR. Start the rendering of the media event.).

Once we abstract from the individual devices and their functions, we arrive at a system-oriented view, where the user perceives the *complete* ad-hoc ensemble as a single system of interoperating components that helps him in reaching his specific goals. Once we abstract from (device) actions and have the system communicate with the user in terms of (user) goals, we also have effectively changed the domain of discourse from the system's view of the world to the user's view of the world.

Goal-based interaction requires two functionalities: *Intention Analysis*, translating user interactions and context information into concrete *goals*, and *Strategy Planning*,

which maps goals to (sequences of) device operations (see Figure 4, left). With respect to the information processing inside appliances as outlined in Section 4, these two functionalities can be interpreted as components of the Control Application, resulting in the extended processing pipeline (shown in Figure 4, right).

Note that goal based interaction is able to account for the operational integration, called for in Section 3: Operational integration can now be realized based on an explicit modeling of the semantics of device operations as "precondition / effect" rules, which are defined over a suitable environment ontology. These rules then can be used by a planning system for deriving strategies for reaching user goals, which consider the capabilities of all currently available devices (see [7] for details).

We will now briefly look at the core concepts of a middleware for supporting the dynamic instantiation of the distributed event processing pipeline that coordinates the operation of an appliance ensemble.

6 Towards a Middleware for Self-Organizing Ensembles

From an engineering point of view, it is not sufficient just to enumerate the desired features of a software system – one would also like to know if it is possible to construct a effective *solution* to this feature set, lest we should follow a pipe dream. And we are well aware of the fact that the set of features we have called for in the previous sections is a rather tall order.

Therefore, we have begun investigating potential solutions for software infrastructures supporting self-organizing ensembles. In this section, we outline the basic properties of the SODAPOP infrastructure, a prototype middleware for ambient intelligence environments based on appliance ensembles. SODAPOP[2] development is currently pursued within the scope of the DYNAMITE project—the interested reader is referred to [4, 8] for a more detailed account.

The SODAPOP model introduces two fundamental organization levels:

- Coarse-grained self-organization based on a data-flow partitioning, as outlined in Section 4.
- Fine-grained self-organization of functionally *similar* components based on a kind of "Pattern Matching" approach.

Consequently, a SODAPOP system consists of two types of elements:

Channels, which read a single message at time *point* and map them to multiple messages which are delivered to components (conceptually, *without delay*). Channels have no externally accessible memory, may be distributed, and they have to accept *every* message.

Channels provide for *spatial distribution* of a single message to multiple transducers. The specific properties of channels enable an efficient distributed implementation.

Transducers, which read one or more messages during a time *interval* and map them to one (or more) output messages. Transducers are *not* distributed, they may have a memory and they do not have to accept every message.

[2] The acronym SODAPOP stands for: **S**elf-**O**rganizing **D**ata-flow **A**rchitectures su**P**porting **O**ntology-based problem decom**P**osition).

SODAPOP provides the capability to create channels—message busses—on demand. On a given SODAPOP channel, messages are delivered between communication partners based on a refined publish / subscribe concept. Every channel may be equipped with an individual strategy for resolving conflicts that may arise between subscribers competing for the same message (the same request).

Once a transducer requests a channel for communication, a check is performed to see whether this channel already exists in the ensemble. If this is the case, the transducer is attached to this channel. Otherwise, a new channel is created. Through this mechanism of dynamically creating and binding to channels, event processing pipelines emerge automatically, as soon as suitable transducers meet.

When subscribing to a channel, a transducer declares the set of messages it is able to process, how well it is suited for processing a certain message, whether it allows other transducers to handle the same message concurrently, and if it is able to cooperate with transducers in processing the message. These aspects are described by the subscribing transducer's *utility*, which encodes the subscribers' handling capabilities for the specific message.

When a channel processes a message, it evaluates the subscribing transducers' handling capabilities and then decides, which transducer(s) will effectively receive the message. Also, the channel may decide to *decompose* the message into multiple (presumably simpler) messages, which can be handled better by the subscribing transducers. (Obviously, the transducers then solve the original message in cooperation.)

How a channel determines the effective message decomposition and how it chooses the set of receiving transducers is defined by the channel's *decomposition strategy*. Both the transducers' utility and the channel's strategy are eventually based on the channel's ontology—the semantics of the messages that are communicated across the channel. A discussion of specific channel strategies for SODAPOP is out of the scope of this paper. It may suffice that promising candidate strategies for the most critical channels—the competition of Dialogue Components for Input Events, the competition of Strategists for goals on the Goal Channel, and the cooperative processing of complex output requests—have been developed and are under investigation (see [6, 8] for further detail on SODAPOP).

7 Ensemble Organization by SODAPOP

To summarize, self-organization is achieved by two means in SODAPOP:

1. Identifying the set of channels that completely cover the essential message processing behavior for any appliance in the prospective application domain.
2. Developing suitable channel strategies that effectively provide a distributed coordination mechanism tailored to the functionality, which is anticipated for the listening components.

Then, based on the standard channel set outlined in Figure 4, any device is able to integrate itself autonomously into an ensemble, and any set of devices can spontaneously form an ensemble.

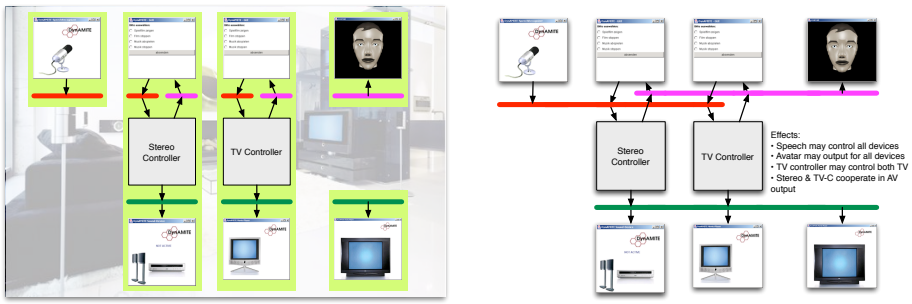

Fig. 5. A simple Smart Living Room Ensemble. Before assembly (left), and after topology formation (right)

Currently, SODAPOP is available as experimental software from the DYNAMITE web site [4]. Formation of an ensemble based on experimental SODAPOP is outlined in Figure 5. On the left, the individual appliances are shown (the green boxes symbolize their hardware packages), where this example contains a stereo and a TV set (both with standard WIMP-type user interfaces), a solitary speech input, a solitary display, and a solitary avatar (possibly on a mobile display). For all devices, their internal transducers and channel segments are shown. On the right, the resulting "ensembled" appliance set is shown, after the matching channel segments have linked up by virtue of SODAPOP. Note how the vertical overall structure at left has been replaced by a horizontal overall structure. Note also, that now stereo and TV both afford speech control, output may be done anthropomorphic through the avatar by all components, and the audio for a movie will be automatically rendered by the stereo system (winning competition with the TV-set's audio system by offering a higher quality).

8 Conclusion

Ambient Intelligence promises to enable ubiquitous computing technology to provide a new level of assistance and support to the user in his daily activities. An ever growing proportion of the physical infrastructure of our everyday life will consist of smart appliances. In our opinion, an effective realization of Ambient Intelligence therefore inherently requires to address the challenge of self-organization for ad-hoc ensembles of smart appliances.

We argue that a possible solution should be based on the fundamental concepts of *goal based interaction* and *self-assembling distributed interaction pipelines*. In guise of the SODAPOP middleware, we have given evidence that such an approach indeed is viable.

We do not expect the solution proposal we have outlined above to be the only possibility. However, we hope that we have convinced the reader that there is *at least one* possible and sufficiently concrete approach towards solving the substantial challenges of dynamic ensembles, which are raised by the proliferation of ubiquitous computing technology.

As stated in the introduction, Ambient Intelligence is in the focus of European IST research initiatives. As can bee seen from our discussion, software infrastructures for Ambient Intelligence need to cope with complex systems, being able to readjust themselves dynamically to changing circumstances. European research has a strong and successful background in designing and implementing complex systems—Europe therefore is well positioned to response to the challenge of Ambient Intelligence.

So we are looking forward to exciting new research and results in this important area of future ICT infrastructures.

References

1. Aarts E. Ambient Intelligence: A Multimedia Perspective. *IEEE Multimedia*, 11(1):12–19, Jan–Mar 2004.
2. Brummitt B, Meyers B, Krumm J, Kern A, Shafer S. Easy Living: Technologies for Intelligent Environments. *Proc. Handheld and Ubiquitous Computing*. Springer, 2000.
3. 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, 2001.
   `ftp://ftp.cordis.lu/pub/ist/docs/istagscenarios2010.pdf`
4. DynAMITE -– Dynamic Adaptive Multimodal IT-Ensembles (Project Web Site)
   `http://www.dynamite-project.org`
5. Franklin D, Hammond K. The intelligent classroom: providing competent assistance. *Proc. fifth Int'l Conf. on Autonomous Agents*. ACM Press, 2001.
6. Heider T, Kirste T. Architectural considerations for interoperable multi-modal assistant systems. In Forbig P (Ed.) *Proc. Interactive Systems, Design, Specification, and Verification (DSV-IS 2002)*. Springer, 2002.
7. Heider T, Kirste, T. Supporting goal-based interaction with dynamic intelligent environments. *Proc. 15th European Conf. on Artificial Intelligence (ECAI'2002)*, Lyon, France, 2002.
8. Hellenschmidt M, Kirste T. Software Solutions for Self-Organizing Multimedia-Appliances. *Computers & Graphics, Special Issue on Pervasive Computing and Ambient Intelligence* 28(5), 2004.
9. MIT Project Oxygen, Pervasive, Human-centered Computing. (Project Web Site)
   `http://oxygen.lcs.mit.edu/`
10. Shadbolt N. Ambient Intelligence. *IEEE Intelligent Systems*, 18(4):2–3. Jul/Aug 2003.
11. Stanford Interactive Workspaces—iWork. (Project Web Site)
    `http://iwork.stanford.edu/`
12. Servat D, Drogoul A. Combining amorphous computing and reactive agent-based systems: a Paradigm for Pervasive Intelligence? *Proc. first Int'l Joint Conf. on Autonomous Agents and Multiagent Systems*. ACM Press, 2002.