User Modeling and User-Adapted Interaction (2005) 15:275–302
DOI 10.1007/s11257-005-1092-2

Personalization and Context Management

ANDREAS ZIMMERMANN, MARCUS SPECHT, and ANDREAS LORENZ
Fraunhofer Institute for Applied Information Technology, Schloss Birlinghoven, 53754 Sankt Augustin, Germany.
e-mail: {*Andreas.Zimmermann,Marcus.Specht,Andreas.Lorenz*}*@fit.fraunhofer.de*

(Received: 21 April 2004; accepted in final form 5 May 2005)

Abstract. Supporting the individual user in his working, learning, or information access is one of the main goals of user modeling. Personal or group user models make it possible to represent and use information about preferences, knowledge, abilities, emotional states, and many other characteristics of a user to adapt the user experience and support. Nowadays, the disappearing computer enables the user to access her information from a variety of personal and public displays and devices. To support a new generation of contextualized and personalized information and services, this paper addresses the problem of *context management*. Context management is a new approach to the design of context-aware systems in ubiquitous computing that combines personalization and contextualization. The presented framework for context management integrates user modeling and context modeling, which can benefit from each other and give rise to more valid models for personalized and contextualized information delivery. The paper will introduce a base framework and tools for designing context-aware applications and decompose the underlying framework into its foundational components. As two illustrative application cases, the paper discusses implementations of an intelligent advertisement board and an audio-augmented museum environment.

Key words. context-aware computing, context management, context-toolkit, mobile authoring tools, personalization

1. Introduction

Supporting the individual user in her working, learning, or information access is one of the main goals of user modeling. Personal or group user models make it possible to represent and use information about preferences, knowledge, abilities, emotional states, and many other characteristics of a user to customize the user experience and support her with adapted interaction. On the one hand, these applications try to establish personalization as a central concept for indidualized services and information; on the other hand, applications also try to integrate contextual information into their services to provide the end user with easier access to personalized information, as in location-based services. Today, mobile devices deliver information that is adapted to the current location or network bandwidth of the user's device. But most approaches lack a consistent integration of user modeling and contextual modeling for contextualized and personalized applications (Jameson, 2001a). In this sense, future applications must take a crucial step towards new information services that take into account the user's personal profile and history and also the current context of use (Oppermann and Specht, 2000).

Because of the rapid development in mobile information technology and display technology, the user's access to information is becoming increasingly ubiquitous. Heterogenous devices can be used to access information and services, and there is an urgent need to filter information, adapt it, and customize it, not only to the indivdual user but also to the current context of use. A simple example of such a need is the fact that today's information displays are not always private; therefore, a public projection display should not present personal information to a group of users standing in front of it. Nevertheless those public displays can fulfill an important role in the contextualized support process for an individual: They can select and filter information relevant to the common characteristics of the physical environment or even take into account user group models that include, for example, sociodemographic parameters. From our point of view, a main challenge and also potential for future contextualized and personalized support lies in the combination of public and private information displays and the combination of personalization and contextualization. That is, personal displays can be combined with public displays to yield personalized contextualized contents and services.

In the effort to create added value in ubiquitous environments, the current context of use becomes more important as the computer disappears. From our point of view, it is not enough to supply content or services that consider single environmental or user characteristics; we need to identify approaches for the integration and interpretation of different sensing components for modeling the user and the context more appropriately. The combination of context sensor data with user modeling appears to be one of the key challenges for personalized information systems of the future. In this sense, delivering information that is to the point has two main facets:

1. Personalization allows users to obtain information that is adapted to their needs, goals, knowledge, interests or other characteristics. User models deliver the main parameters for selecting and adapting information presentation to the individual user.
2. Contextualization complements personalization so that environmental states or the context of use can also be taken into account.

In this paper, we discuss the relationship between user modeling and context modeling and how to combine the two approaches for contextualized and personalized applications. After discussing relevant previous work, we introduce an implemented framework and a toolkit for integrated personalization and contextualization of applications. We then demonstrate with two case studies how we have built personalized and contextualized systems that are based on this infrastructure.

2. Contextualization and User-Adaptive Systems

Since the 1950s, software architects have been designing systems for the adaptation of information to the user for a variety of reasons. Many works about user-adaptive systems have looked at questions like:

- how to acquire data about users and their behavior patterns;
- how to represent and store information about users;
- how to analyze a user's behavior and draw inferences about a user's knowledge, interests, goals, and other characteristics;
- how to adapt system parameters to individual users or groups of users;
- how to plan and realize adaptations.

With the help of recent methods from the research field of context modeling, we can take these approaches a step further by integrating different sources of user data, extending inference capabilities, working on more reliable user data, and enhancing channeling and distribution mechanisms. A detailed mapping between context parameters, user behavior and the available interaction channels forms the basis for more contextualized interactions (see also Schmidt, 2002).

When we analyze the literature on context-aware information systems, the trend toward the integration of (auditive or visual) ubiquitous displays becomes obvious. These systems enable the user to move in physical space and use different public displays or to access systems via personal displays that they carry with them. Therefore, it becomes necessary to adapt the information not only to the user's preferences but also to an enviroment that changes rapidly as the user moves through it. This need raises the challenge of defining new approaches for applying adaptive methods with new rendering and interface technologies. We see this challenge as a major paradigm shift in user interaction towards multimodal ubiquitous interaction. Initial examples of new adaptive methods in content delivery are location-based services and museum information systems like hippie (Oppermann and Specht, 2000). Mostly, those systems have taken into account the user's location and employed simple room models or tag-based approaches for identifying hot spots in the environment of the user. In the hippie project, a tag-based approach for identifying the physical environment of the user was combined with the recording of individual user tracks in the physical space, which again allowed inferences about the user's interests. In terms of combining user modeling and contextualization, this approach enabled hippie to learn about user interests from a combination of tracking technology in physical space and simple taxonomies connected to physical locations. New forms of augmented reality training systems enable the user to explore a domain and its artifacts either in virtual reality training simulations (Johnson et al., 1998) or in tracked real training environments (Fox, 2001). This approach allows for a detailed diagnosis of the learner's skill level, especially with respect to motor skills. It also supports tutorial interactions. Another example of the extension of adaptive methods comes from the field of adaptive augmented reality systems. The LISTEN system (Zimmermann and Lorenz, 2003) tracks the user's position and head orientation with a resolution of

5 cm and 5°, which allows it to identify whether a user is looking at a detail of a work of art in a gallery or just at the frame or at a point beside the artwork (Gossmann and Specht, 2001). On the basis of this detailed tracking data, an auditory display is created in LISTEN that guides, entertains, and informs the user with an audio-augmented real world experience. As a new instantiation of adaptive prompting, LISTEN introduced the possibility for the system to attract the user's attention by whispering or whistling at her from a certain direction in the 3D auditory display. On the basis of a model that relates the physical world situation and its parameters with electronic artifacts, different classes of applications can be distinguished that match user and context parameters in order to recommend content, recommend other users, or personalize the system experience in some other way (Gross and Specht, 2001).

2.1. EXTENDING CLASSICAL APPROACHES

Basically, our understanding of contextualization is derived from an extension of the classical adaptive hypermedia approach (e.g. Brusilovsky, 1996). In adaptive hypermedia, adaptive methods can be described interms of four main dimensions:

– *Information used for adaptation: What information about the user is known, and what information can the system use for adaptation?* For contextualized systems, this question can be extended to the user's environment and to new methods for integrating sensor data and making inferences about it. For example, the system can use all of the sensors in a room in which a user is currently moving, or the system can use context information concerning related entities in another place. These new types of information make possible more valid and richer inferences in contextualized computing.
– *Adaptation component: What aspect of the system adapts to the given information about the user?* For contextualized systems, this question can be extended to multimodal interaction and sensing. In classical user modeling, the user's environment is her PC desktop. The input channels for user tracking are mostly limited to the GUI interactions of the user. In ubiquitous computing, new channels for both user input and system output become available. Location can be seen as just one parameter for contextualization. Furthermore, a system can use all available information displays that can be perceived by the user. For example, a system could use a loudspeaker in the room of the user, a big screen display, or a PDA of the user.
– *Adaptation goal: Why does the system adapt to this information?* Mostly, adaptive systems adapt to their user for ergonomic or pedagogical reasons. In contextualized computing, the interaction of the user with his/her physical environment becomes more important. One goal of contextualized systems could be to create more natural and authentic interactions and to make the adaptive system appear smarter in that it is appropriately embedded in real-life experiences.

– *Adaptation strategy: What steps are taken to adapt the system to the user, and how active or reactive are the user and the system in the adaptation process?* For contextualized computing, shorter feedback loops can be predicted for the implicit tracking of the user's behavior, as more parts of the user's behavior can be sensed and taken into account. Additionally, because of the variety of input and output channels, new forms of interaction and continuous updates of user and context models become more important.

Adaptive hypermedia systems collect information with a variety of explicit and implicit acquisition methods. In order to give an integrative overview of the various forms of adaptivity, Jameson (2003) has defined a user-adaptive system as

> "an interactive system that adapts its behavior to individual users on the basis of processes of user model acquisition and application that involve some form of learning, inference, or decision making".

In our terms:

> "an adaptive system (contextualized or personalized or both) follows an adaptation strategy (e.g., pacing or leading) to achieve an adaptation goal (e.g., intuitive information access or easy use of a service). To achieve an adaptation goal, it considers relevant information about the user and the context and adapts relevant system components on the basis of this information".

Combining contextualization and personalization also has important impacts on the processes of user model acquisition, inferencing, and interaction adaptation. If the process of *user model acquisition* also takes into account a variety of environmental variables for the adaptation to individual users, the system can have shorter feedback cycles and more valid information about the real needs of the user in the context of use. Furthermore, the adaptive system could adapt not only to the individual user model and explicit user feedback but also to more concrete information about the user's behavior and implicit feedback loops (*user model acquisition*; Jameson, 2003). By integrating environmental sensors and user sensors, new applications can collect direct feedback from user sensors in a way determined by data from environmental sensors, as is shown in Figure 1. A good example of such an application might be a training system (e.g., a medical training application like the echo tutor of Grunst et al., 1995) that monitors the users' movements while handling a complex machine and gives direct feedback for training purposes. As a consequence, new interaction metaphors are also created for the user interface. While at first glance these metaphors do not seem to make a big difference to the user modeling, they have important impacts on the information available for inferencing, information filtering, and presentation. In a broader sense, the adaptive methods applied can build on more valid information from the *inferencing* components. Additionally, new forms of adaptive methods can instantiate classical adaptive methods in a new way. For example, adaptive annotation could take into account the different modalities and output channels available in the context of use

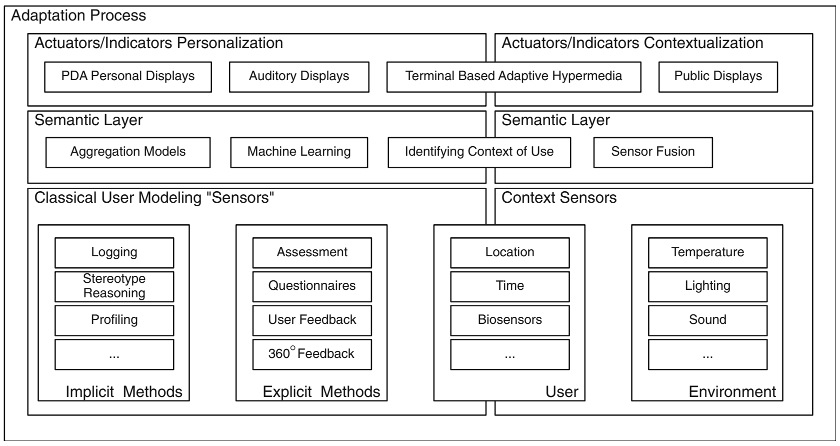


Figure 1. User sensors and environmental sensors for more valid inferences and validation of implicit user tracking methods.

and give explanatory audio instructions when a user looks at a real-world object (*user model application*; cf. Jameson, 2003; Kobsa et al., 2001). Furthermore, the different input and output channels can be used to collect information in the current context, as when new forms of interactive mobile games ask the user multiple-choice questions that are related to the current physical environment.

2.2. RELATED WORK ON CONTEXT MANAGEMENT

In recent research, the term *context management* usually refers to the administration of contexts. Context servers or other context-managing components store contexts and provide access for retrieving, comparing, and updating the knowledge. Beforehand, programmers and other computer scientists have defined sensors and established the system’s behavior, either directly in the source code or via tools like the Context Toolkit of Salber et al. (1999).

2.2.1. *Context Toolkit*

As a very first step, Schilit (1995) presented a system architecture that supports context-aware mobile computing and that made it relatively easy to build such applications. The Context Toolkit developed at the Georgia Institute of Technology (cf. Dey et al., 2001; Salber et al., 1999) aims to provide a reusable solution for handling context that facilitates the implementation and deployment of interactive context-aware applications. The toolkit incorporates various services related to the gathering and supplying of context, including an encapsulation of context sensors, access to context data, context storage, and a distributed infrastructure. In an analogy to GUI widgets, they defined *context widgets* for supporting the acquisition and delivery of context.

The Context Toolkit addresses many issues related to context gathering and supply. Its main flaw is the absence of a formal context model and a means for controlling an application or changing dynamic parameters inside an application. In addition, the functionality of interpreters for context derivation is limited, as they are usually employed only for simple data type conversions. As a result, support for context comparisons is limited as well.

2.2.2. *Context-Framework*

A recent and ongoing research activity is the approach to a context management infrastructure for pervasive computing environments pursued by Henricksen (2003). This infrastructure facilitates the development of context-aware applications through the provision of generally required functionalities such as context gathering, context management and context dissemination. These functionalities are based on a context model that addresses the diversity of contextual information, its quality, and complex relationships among context data and temporal aspects. The context management infrastructure is organized in a ‚three-layer hierarchy of interacting components: a Context-Aware Application layer, a Context Management layer and an Awareness Module layer. The communication among these layers is handled by a notification service.

The context model provided by this approach contains some simplifications that make a comprehensive representation of complex contextual information difficult, since the question remains how entities, attributes, and their values are to be modeled. In conjunction with context gathering, no augmentation processes have been considered so far. In addition, context data filtering can be carried out only if all of the data that is to be filtered refers to the same context attribute.

2.2.3. *Context Middleware*

Within the scope of the Context Service project of the IBM Research Center, a middleware infrastructure aimed at gathering and disseminating context in pervasive computing environments has been developed (Ebling et al., 2001). This service-based approach provides applications with contextual information at a high level of abstraction. This information provision covers the gathering of context information via sensors, maintaining context, and responding to queries by clients. Further issues like context quality, context storage, and extensibility are also addressed. In contrast to many other approaches, this project explicitly addresses privacy and security concerns. The context model used is based on a form metaphor that describes a particular type of contextual information and is composed of a set of attributes (Lei et al., 2002).

The concepts introduced by the Context Service seem to require further development, since the form of abstraction used to represent context evidently does not distinguish between different context attributes. This fact makes it difficult to handle the potential complexity of context. Furthermore, the architecture of the Context Service considers processes of context data augmentation only marginally, since neither context data derivation nor filtering of context data are carried out.

3. CXMS: A Context Management Framework

Many projects at Fraunhofer FIT have made use of user modeling components and personalization engines in order to make systems adaptable by and adaptive to the individual user. Many of these components are specialized applications, tailored to one specific domain or environment and rarely reusable. Common problems that emerge during the development and the reuse of components are the strong dependency on the domain, the lack of open and standardized interfaces, the lack of uniform representation of user profiles and models, and the use of different and distributed data sources. On the basis of our experiences in working in different application domains, we were able to abstract from the variety of technical implementations to create a general framework where different methods for context knowledge acquisition, domain inference, and adaptive methods for personalization and contextualization can be combined.

3.1. DESIGNING CONTEXT-AWARE SYSTEMS

The types of domains for context-aware and adaptive systems are manifold. Applications like e-learning systems, adaptive hypermedia, or mobile services share a certain property: they involve interaction with a user or with groups of users. If we treat users as one special kind of entity within a domain, we can identify the components such a domain is composed of as: domain entities, entity attributes, relations between entities, and messages or events exchanged among entities.

Dey and Abowd (1999) define an entity as a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves. Every context-aware and adaptive system needs to map parts of these components onto an internal model. First of all, the implementation of our framework aimed at providing support for the development of contextualized systems. Therefore, we developed a component-based architecture for hosting several components on different abstraction levels and providing facilities like database access and information transfer between the components. The resulting framework separates its components by functionality, giving the opportunity to change the composition on each layer. Figure 2 illustrates the four main layers in a transparent way, distinguishing among the Sensor, Semantic, Control and Indicator/Actuator Layers.

On its way through all of the layers, unstructured data becomes semantically enriched step by step, until a decision can be taken about how the domain can be adapted. In the following subsections, the components of this model are described in more detail.

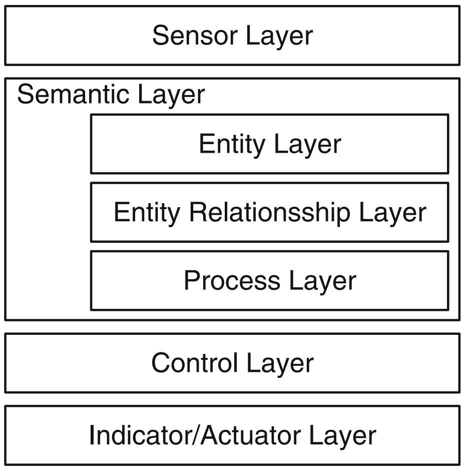


Figure 2. A layered approach to contextualization.

3.1.1. *Sensor Layer*

The first functional layer serves as an information collector. Each context-adaptive system relies on a network of sensors placed in the physical environment and delivering an image of the current situation that the user is acting in. Sensor filters detect every change within the environment and perform an analysis of the user's behavior and interaction with the system; the system's sensitivity, speed, and accuracy will depend on the technology of the sensing infrastructure. The sensor satellites work both synchronously, triggered and filtered by time intervals, and asynchronously, by events or on demand (via push and pull); hence they deliver a temporally diffuse stream of data to the sensor layer.

3.1.2. *Semantic Layer*

The Semantic Layer defines the context model of a context-adaptive system. In our understanding, a user may have a context and a context may enclose a user. The context model captures the current situation that the user is acting in, including her preferences, interests, social dependencies, and physical and technical environment. Overall, the context model has to cover the four dimensions mentioned in Gross and Specht (2001): Identity, location, time, and environment. The semantic layer semantically enriches the data delivered by the sensor layer and assigns values to corresponding attributes of the entity's context. Thus, a context always represents all available up-to-date information describing the current situation. Starting from that, the sub layers entity, entity-relation and process further utilize the context information.

3.1.2.1. *Entity Layer.* In the entity layer, all entities of the domain are defined, and the mapping of sensor data streams to attributes of entities is specified. Furthermore this layer is used to specify relations between domain entities and their contexts, and it defines the model for nested entities. In recent research on user-adaptive systems (Schilit et al., 1994), countless approaches for the identification of different context types exist, such as user context, computing context, and physical context or identity, location, time, and environment (Gross and Specht, 2001). An important distinction is made between static and dynamic parts of context (Zimmermann et al., 2002). From our point of view, all domain entities can have static or dynamic parts in their context definition. Especially the environmental attributes often have highly dynamic attibutes if users are supported in mobile applications.

3.1.2.2. *Entity Relationship Layer.* All entities that are part of the interaction relate to each other in certain ways. On the entity relationship layer, dependencies are modeled, so as to express associations like x communicates with y or x is localized in-front-of y. The entity relationships can be dynamically added and deleted. For example, a user moving in a building can have a dynamically instantiated relation to the room entity he currently is in.

3.1.2.3. *Process Layer.* The process layer observes the evolution of the abovementioned contexts or context parts over time. The application of time series and history modules, statistical models, and intelligent algorithms support this analysis. At this level, beliefs about the user's behavior, preferences, interests, plans, and other characteristics are acquired through inference and learning, as are beliefs about future entity relationships or environmental changes. Since different inference mechanisms may derive different beliefs about the context, a system for the resolution of contradictory evidence and inconsistent beliefs should be employed if necessary.

The semantic layer completely covers the profiling task of personalization engines and provides different views on the data captured about context to the succeeding matching task. The semantic layer supplies the control layer with an accurate image of the current interaction situation.

3.1.3. *Control Layer*

On the basis of the context model and data provided by the semantic layer, the control layer decides what actions should be triggered if particular conditions in the model become true. As a consequence, it generates sequences of commands for the control of the behavior of the domain. A direct communication link between the control layer and the domain enables the system to respond to simple requests from external systems or applications. It also allows the realization of shared-initiative and shared-control approaches, which have been proposed and used for end user development in the last 10 years for adaptive information and learning systems (Oppermann, 1994; Oppermann and Thomas, 1996; Oppermann and Specht, 2000; Wulf, 2000).

The commands assembled by the control layer may vary in their level of abstraction. We can distinguish between commands and strategies. Commands may be simple actions like turn-volume-down, while strategies represent more complex macros or plans like attract-user-attention. On this strategic layer, the system decides on the highest level whether to behave actively or reactively in the interaction. The basic underlying principles, which the user will experience as the essential distinction made on this layer, are *pacing* and *leading*.

The selected action or strategy can be seen as the link between these two questions:

1. *What information is taken into account for adaptation?* Using the terms of the entity layer, either direct attribute values of entity contexts like the language or some preference can be taken into account. Concerning the entity-relationship layer, the system can also evaluate a relation from the current user entity for possible adaptation, like the room temperature or the speed of the vehicle that the a user is traveling in. As information from the process layer, current plans of the user can also be taken as the relevant information for adaptation.
2. *Which part or functionality of the user interface is adapted, and how?* The other central parameter of the adaptive method is the way in which the adaptation is displayed or realized in the domain. The answer to this question is strongly dependent on the concrete rendering methods implemented on the indicator/actuator layer.

Within this layered model approach, the control layer is also responsible for context management issues like changing context values of specific entities, deleting contexts in context histories, and the fine-tuning of several models and algorithms implemented on the semantic layer. To the succeeding indicator/actuator layer, the control layer delivers the assembled sequence of commands.

3.1.4. *Indicator/Actuator Layer*

The indicator/actuator layer handles the connection back to the domain by mapping the decisions taken by the control layer to real world actions. Specialized software components process the delivering of information snippets or the displaying of data on a particular device. Engines on this layer implement domain-dependent methods that directly change variable parameters of the domain. Depending on the level of integration with the domain, these methods may be part of the target application; or the commands assembled by the control layer may have to be transformed into appropriate actions. As feedback for the control layer, messages indicating the success or failure of actions are sent back.

3.2. COMPONENTS OF A CONTEXT MANAGEMENT SYSTEM

From our point of view, product managers or application designers know best about the required system behavior, necessary data sources, and constraints.

Instead of system developers, these end users should be able to define appropriate information sources, sensors, parameters, and rules defining the adaptive behavior. Since content providers handle different contents for different scenarios, they should also have influence on the adaptation process and results. For reasons like these, we will shift the meaning of the term *context management* from the software level towards an application design level:

> *Context Management* involves the construction, integration, and administration of context-aware behavior. Context management considers the definition of relevant context parameters, the link between these parameters and information sources, their utilization for the targeted adaptive behavior, and the definition of that behavior.

The Context Management System (CXMS) developed by the Fraunhofer Institute for Applied Information Technology offers a tool suite that is intended to facilitate the development and maintenance of context-aware systems and services through the hiding of complex technical details from developers and end users. Additionally, the CXMS allows the flexible enhancement of existing applications and services that lack adaptive and context-aware functionality.

The Context Management System comprises two main parts, shown in the center of Figure 3: The Context Modeling Tool (left) and the Content Management System

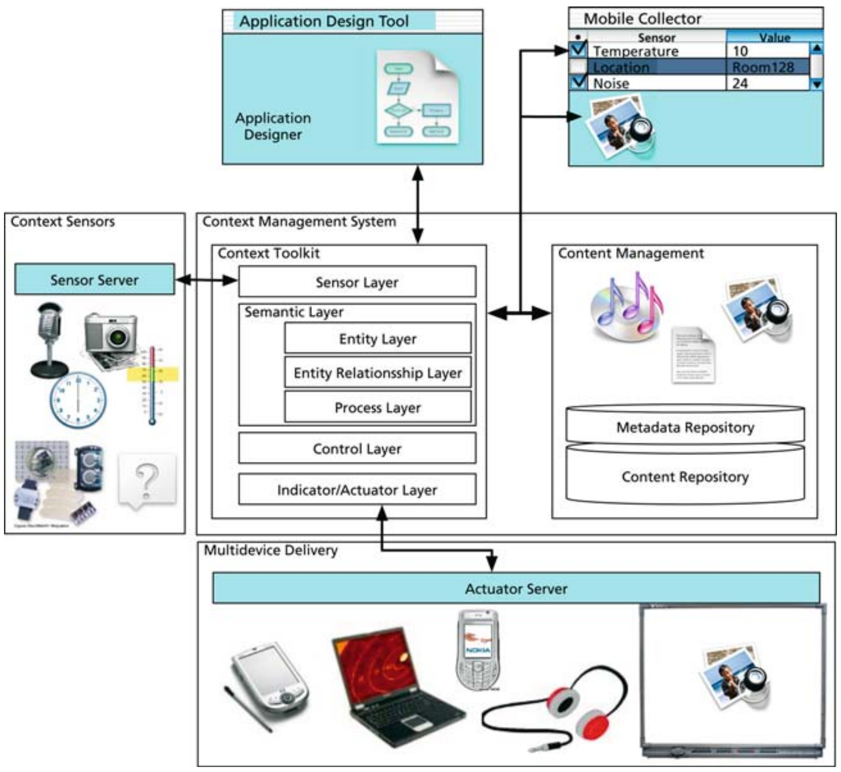


Figure 3. The components of the Context Management System.

(CMS) (right). As a repository of content, the CMS enables the administration of content and typically provides access to a huge amount of content. In the following, we describe in more detail the four main components of the CXMS: the Context Toolkit, the Design Tool, the Mobile Collector, and the Content Player.

3.2.1. *Context Toolkit*

The Context Toolkit developed by the Fraunhofer Institute provides software developers with an extension to the JAVA programming language for the implementation of context-aware applications. The libraries consist of ready-to-use software components that hide complex technical details from the developer. The implementation of the Context Toolkit realizes the layered approach presented above.

The sensor layer consists of software objects that receive incoming data, perform cleaning and fusion of sensor values, and fire sensor change events that notify objects in the semantic layer. Depending on qualities like precision and signal availability, a technology hand over switches between several sensor technologies that deliver the same kind of information (e.g., GPS or WLAN for tracking position). Already existing sensors for position tracking and for noise or motion detection can be extended through the sensor interface. Since the Context Toolkit was designed as a modeling tool, its current implementation does not support the automatic detection or recognition of sensors.

Each context attribute on the semantic layer is connected with zero or more sensors, and in turn sensors deliver information to one or more attributes, creating across-linked network between sensors as information sources and attributes as information interpreters. Context attributes receive sensor values and map them onto semantically more enriched values (for example, a ‘time of day’ attribute interprets the value of a time sensor). The current implementation provides a model-based interpretation of attributes that map sensor data to different abstractions for time, to locations based on location models and to certain degrees of noise and motion. Additionally, attributes may derive their values from algorithmic datafusion from more than one sensor (for example, speed is derived from position and time).

In our modeling approach, we have chosen attribute-value pairs as a simple but flexible context representation. Each context is an enumeration of one or more context attributes, and each entity owns one static context by default and several types of dynamic contexts in addition. For each type of context, the implementation allows for the configuration of a persistent storage of changes of this type of context as a history. The memorization of contexts may be triggered in three ways: after every change event, on the basis of a fixed time interval, or on the basis of explicit input of other entities. As a part of the process layer, this history function supports the profiling task of personalization engines and enables the derivation of preferences, interests, and so forth.

The Context Toolkit supports three types of matching procedures for entities: First, two entities can be compared to see if their contexts are similar to each other (e.g., users in the same room). This matching procedure gives information about relationships between entities. Second, a matching can be performed to see if the entity's context fulfills certain criteria (e.g., for the application of stereotypes). A third matching procedure enables the filtering of entities from a list (e.g., for a database query). Therefore, the toolkit offers the configuration of qualifiers (whose results are true or false) and of more complex similarity measures (whose results lie in the interval [0..1]).

The components described so far form the basis for a rule system that controls the desired behavior of the target application. This rule system is a set of hierarchically ordered rules, each rule comprising a precondition and an action part. If the precondition of a rule is fulfilled, the rule fires and all associated actions are executed. Preconditions consist of Boolean expressions built up from qualifiers. Currently available actions bring about changes in context attribute values and in relations between entities, the selection of content, and the presentation of content on different devices. Accordingly, the indicator/actuator components focus on displaying content on different devices.

The temporally diffuse stream of data delivered to the sensor layer may strain the operation of the entire system (some domains have to deal with highly dynamic alterations of sensor values, cf. Zimmermann et al., 2002). To mitigate this strain, we optimized the event flow of the system using the event handling mechanisms of the JAVA programming language. In a very first step, a discretization procedure prunes and cleans the sensor value stream (e.g., measuring the position in 2-second intervals for visitors in a museum). If the sensor value changes, the sensor fires an event notifying all attributes that have registered as listeners to this event. The attribute receiving the new sensor value performs a mapping of this value to a model, which in general further reduces the precision of the original value (e.g., a certain position falls into a specific zone of a location model). If the value of one context attribute changes, a context change event is fired. The control layer receives this event and triggers the evaluation of the rule system, which results in an adapted behavior of the application. This dovetail connection of events limits the overall computational complexity of the system.

3.2.2. THE DESIGN TOOL

The mode of operation of the software components provided by the Context Toolkit and the entire design of the targeted context-aware application are to a certain extent configurable using the language XML. We therefore designed an appropriate design and authoring interface, which is a regular editor for XML files augmented with some intelligence that is intended to prevent users from making mistakes. This interface puts together panels that are necessary for the design of the application: panels for sensors, attributes, modeling, control, queries, and actuators, respectively.

The SENSOR panel enables the administrator to add and remove sensors. Each sensor has to be labeled with a unique name, which is used as an identifier throughout the application. Distributed sensors, to which the application has access only over a network, have to be configured with respect to their IP address, ports, and communication protocol (e.g., TCP/IP or HTTP).

In a similar way, the ATTRIBUTES panel allows for the addition and removal of context attributes, again identified by unique names. Because attributes are referred to via names, they can be assigned to the context model in the next step, and rules or other system components have access to current attribute values. The composition of actuators is editable in the same manner as sensors and attributes. In the ACTUATORS panel, actions for changing the behavior of the application can be selected or removed. At this stage, all basic components of the context-aware application are defined and ready to use for the modeling step.

In the MODELING panel, the user can design the context model used in the application. This panel allows for the definition of domain entities and several types of context. Furthermore, it supports the allocation of sensors to attributes. The static part of the context model contains some meta information about the content, such as its identification and the category that it belongs to. Additionally, the allocation of sensors to attributes is shown; for example the attribute ‘time of day’ depends on the TimeSensor.

The QUERIES panel allows the preparation of qualifiers and preconditions for the rules controlling the application behavior. Queries analyze the current situation expressed by the values of the context attributes and define conditions for the execution of actions that determine the system’s behavior. To allow the creation of such conditions, the QUERIES panel offers a tree-like representation, in which the root node represents the operation and two child nodes the respective operands. An example can be taken from the Intelligent Advertisement Board (cf. Section 4.2): The precondition IS_INTERESTED has the form:

$$\text{IS\_INTERESTED} = (11 < \text{MOTION\_LEVEL} < 35)$$

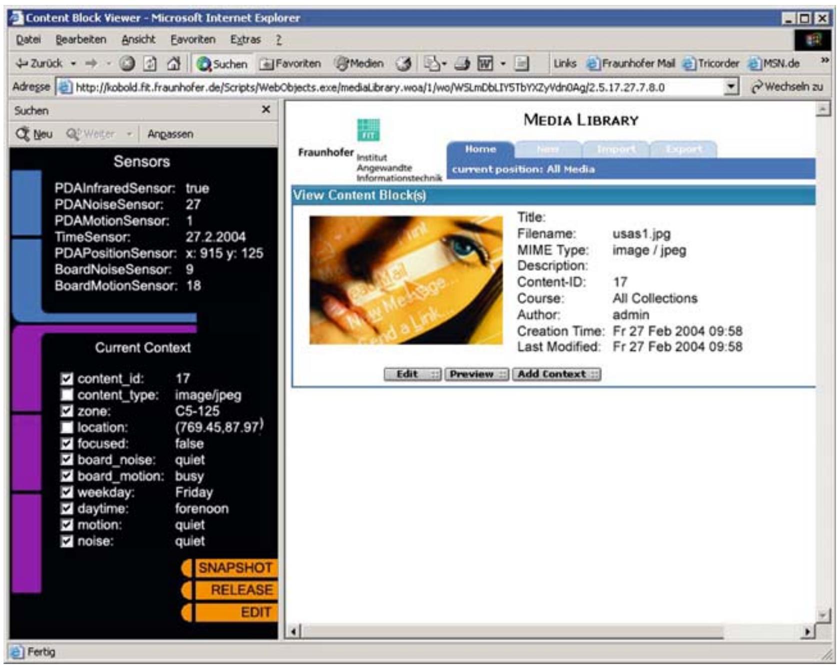
This condition will yield the value ‘true’ if (and only if) a person’s body exhibits some movement (presumably reflecting interest) while the person is standing in front of the advertising board (as opposed to changing location). With the user interface for the CONTROL panel, the designer specifies aspects of the system’s behavior. She can define rules with one or more pre-conditions (expressed by preconditions specified via the QUERIES panel) and a list of actions to be executed if the preconditions are fulfilled. Both the names of queries and the names of actuators refer to the components defined in earlier steps.

3.2.3. THE MOBILE COLLECTOR

The annotation of content with context data is a special task which can most effectively be done on the move or directly in the context of use. Our approach supports users with a tool for recording context data together with content from a Content Management System. The Mobile Collector is designed to be an efficient tool for the production of contextualized content by content providers.

Figure 4 illustrates the Mobile Collector running on a Tablet PC. The right-hand side shows the web front-end of the Content Management System. This screen provides the author with functionality for adding, removing, searching, and browsing content such as images, sounds, videos, or even entire HTML pages. The left-hand side of the figure depicts the current context of the device in the lower panel and the current sensor values in the upper panel. If there are any changes to the values of the sensors or context attributes, both panels are immediately updated. Since the left-hand panel is a browser plug-in, it does not affect the user while he or she is browsing the internet. It reflects the context model defined with the Context Toolkit and allows the user to capture the current context (i.e., to freeze its values), to edit context attribute values, and to (un)select attributes that are considered (ir)relevant in this specific situation.

If the correct content has been selected and the context has been adjusted appropriately, the author can create the link between the two by clicking on the snapshot button, thereby storing this link in the persistent memory of the Mobile Collector. One context snapshot consists of current values for each context attribute at the specific moment at which the snapshot button was pressed. The designer can specify that a given attribute is irrelevant by leaving the checkbox


Content Block Viewer - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück Suchen Favoriten Medien Links Fraunhofer Mail Tricorder MSN.de
Adresse http://kobold.fit.fraunhofer.de/Scripts/WebObjects.exe/mediaLibrary.woa/1/wo/WSLmDbLIY5TbYXZyVdn0Ag/2.5.17.27.7.8.0 Wechseln zu
Suchen
Neu Weiter Anpassen
Media Library
Fraunhofer Institut Angewandte Informationstechnik
Home New Import Export
current position: All Media
View Content Block(s)
Edit Preview Add Context
SNAPSHOT
RELEASE
EDIT
Fertig

Sensors

PDAInfraredSensor:	true
PDANoiseSensor:	27
PDAMotionSensor:	1
TimeSensor:	27.2.2004
PDAPositionSensor:	x: 915 y: 125
BoardNoiseSensor:	9
BoardMotionSensor:	18

Current Context

✓	content_id:	17
☐	content_type:	image/jpeg
✓	zone:	C5-125
☐	location:	(769.45,87.97)
✓	focused:	false
✓	board_noise:	quiet
✓	board_motion:	busy
✓	weekday:	Friday
✓	daytime:	forenoon
✓	motion:	quiet
✓	noise:	quiet

Title:	
Filename:	usas1.jpg
MIME Type:	image / jpeg
Description:	
Content-ID:	17
Course:	All Collections
Author:	admin
Creation Time:	Fr 27 Feb 2004 09:58
Last Modified:	Fr 27 Feb 2004 09:58

Figure 4. The Mobile Collector.

for that attribute unchecked. The value vectors stored with the content blocks are the basis for a filtering process that retrieves content in a specific context later on: The retrieval procedure compares the stored context snapshots with the values of the user's current context and returns the content block associated with the best match. Since the Mobile Collector is a tool for collecting context snapshots and linking them to appropriate contents, it is not used for administrative purposes. Breaking up these links between contexts and contents – i.e., removing the context annotation for a specific content block – requires further treatment using an administrator tool on a desktop PC.

The role of the Mobile Collector in the design process of context-aware applications can be illustrated with reference to the example of a simple location-based museum guide (cf. Section 4.5). As part of the LISTEN system, this museum guide enables visitors to a museum to obtain personalized information about exhibits displayed on a Personal Digital Assistant (PDA) that they carry around. In the first step, the Design Tool supported the developer in the specification of sensors, attributes and actuators. During the preparation of the exhibition hall, the curator of the museum used the Mobile Collector to annotate the exhibits with suitable information. In front of each painting, the curator selected the content blocks from the database that were to be associated with the exhibit and adjusted the values of the contexts in which these content blocks will be retrieved during operation. For the museum guide, the content blocks were mainly images, printed text, and spoken text; and the context attributes were the visitors' position, the time of day, and the day of the week. During the operation of the system, the Content Player (cf. Section 3.5) running on each device filters and presents this content in dependence on the visitor's current context.

3.2.4. CONTENT PLAYER

The Content Player is an adapted browser that runs on a mobile device like a Personal Digital Assistant (PDA). In a similar way as with the Mobile Collector, the sensors connected to the mobile device are read out and their values are sent to the server. The server interprets the sensor values and determines the behavior of the targeted device. In this case, the determination of the behavior of the targeted device involves the selection of contents from the content management system as a function of the current context. The server sends the contents suiting the current situation to the content player, where the browser refreshes the displayed page with the new contents. The Context Toolkit supports the determination of the appropriate behavior and the content playback in the way described above.

4. Example Applications

Unlike most recent work in this area, our work aims to support system developers with different implementation skills in implementing components on each layer described in Section 3.2. The component-based approach facilitates the independent addition, removal, and replacement of components on each layer. This approach allows for a focused allocation of resources to the several layers (e.g., experts in sensoring may work independently from experts in machine learning or decision making). Furthermore, our aim is a close integration of the work of software engineers with the work of product managers, application designers, and content providers. The Context Management System has been the basis for the realization of several applications in various domains. Two examples will be presented in this section as an illustration of how the described infrastructure is applied and how the abstract models defined earlier are instantiated. The two examples are an intelligent advertisement board and a museum guide: At the trade fair CeBIT 2004, the Fraunhofer Institute for Applied Information Technology presented an intelligent advertisement board as an illustrative application of the Context Management System. The museum guide was one of the developments within the LISTEN project, which a 3-year research project that was funded by the European Commission. Those two applications have in common the property that they adapt their behavior when particular context attributes change their values.

4.1. INTELLIGENT ADVERTISEMENT BOARD

At CeBIT 2004, Fraunhofer FIT presented a context-sensitive advertisement board as an application scenario for the Context Management System. The Intelligent Advertisement Board can be used, for example, at train stations, in airports, and in shop windows. The advertising board is equipped with simple and reliable sensors, and it reacts intelligently to the surrounding environment. It is able to respond to changing conditions like noise, trains arriving and departing, the time of day, or people who show interest while standing in front of the board. The content the board's advertisements consists of images from the categories »sports«, »food«, »shopping« and »news«, suitable for the times of day »morning«, »noon« and »evening«. In addition, all advertisements have at least one further level of detail on which »continuative information« is shown – i.e., more detailed information that is related to the main message of the advertisement. If it did not take context into account, the system would simply present the advertisements randomly. With simple contextualization, the display reacts more intelligently. Figure 5 depicts the instantiation of the layer model proposed in Section 3.2 for the realization of this specific application.

4.1.1. *Contextualization*

We plugged a simple and robust webcam and a standard microphone into the system as sensors. The motion sensor is connected to the data stream of the small camera. Through pixel analysis of consecutive pictures, this sensor returns a value within 0 and 100 as the degree of activity in front of the camera. In the same manner, the noise sensor analyses the audio stream of the microphone. In addition, the time of day plays an important role in this application.

The context associated with the content consists of a static and a dynamic part. The static part contains attributes that represent some metainformation about the contents, such as their identifier, the category they belong to (i.e. sports, news, food, or shopping), the time of the day to which they are relevant (i.e. morning, noon, or evening), and an identifier for their more detailed successor. The attributes of the dynamic part of the context model alternate their values in short time intervals. The attributes MOTION and NOISE listen to the corresponding sensors and indicate the activity level in front of the advertisement board. The attributes TIME_OF_DAY and NEXT_TRAIN determine their values on the basis of specific models that abstract from the time delivered by the time sensor. In particular, the attribute NEXT_TRAIN maps the time to the train schedule of the train station Messe/Laatzen near the CeBIT fair ground.

The rules specifying the behavior of the Intelligent Advertisement Board can be seen in the control layer section of Figure 5. Because the time of the day is taken into account, advertisements for nearby restaurants are shown in the evening rather than at breakfast time. Furthermore, the board presents eye-catching and attractive pictures

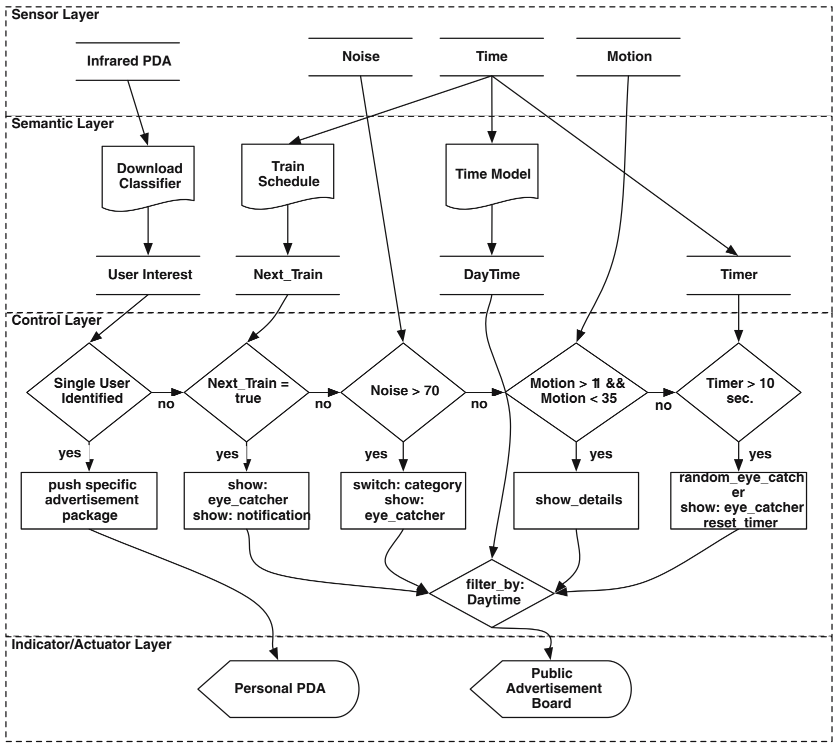


Figure 5. The layer model mapped to the realization of the Intelligent Advertisement Board.

if people do not show any interest. If someone's attention is attracted, the advertising board displays additional and continuative information that is relevant to its eye-catching parent. The train schedule superposes all other context-dependent information. In the case of arriving or departing trains, the advertising board presents with an appropriate announcement and warning and shifts its presentation style to less eye-catching, more informative notifications. The five actuators of the advertising board SHOW: IMAGE, SHOW_DETAILS, SWITCH_CATEGORY, RANDOM_EYE_CATCHER, RESET_TIMER and FILTER_BY: DAYTIME select the most appropriate advertisement and present it on the screen.

4.1.2. *Personalization*

The very first adaptation step described above adapts the behavior of the advertisement to an entire group and does not distinguish among the individuals involved. A limitation of this type of contextualization for user groups is that the adaptation in general does not match exactly any one individual user's needs. A constraint is that no personal information about an individual should be displayed to the public. In some cases, it may be desirable for the contextualized information presented on the public display to be adapted to the special needs of an individual user. Via an infrared pull via a personal digital assistant, an individual user is able to show specific interest in a particular advertisement. This user action triggers a process whereby the advertisement in question, together with all of its associated successors and all advertisements of the same category and the same binding to the time of day, are downloaded to the user's personal device. Each download refines the group model of the advertisement board and redistributes the scores for each advertisement. Furthermore, special needs of individual users can be supported through the delivery of customized forms of advertisement contents to a user, For example, to support deaf users we have produced simple snippets of sign language videos that could be watched on a PDA that a deaf user is carrying with her.

4.2. MUSEUM GUIDE

The LISTEN project (LISTEN, 2005) conducted by the Fraunhofer Institute for Media Communication is an attempt to make use of the inherent everyday integration of the aural and visual perception (Eckel, 2001). In October 2003, this system was made available to the visitors of the August Macke art exhibition at the Kunstmuseum in Bonn (Unnützer, 2001). The users of the LISTEN system move in physical space wearing wireless headphones that render 3-dimensional sound. Therefore, the users can listen to audio sequences emitted by virtual sound sources placed in the environment. The visitors to the museum experience personalized audio information about exhibits through their headphones. While using the LISTEN system, users automatically navigate an acoustic information space designed as a complement or extension of the real space. The selection, presentation, and adaptation of the content of this information space takes into account the user’s current context. In the Hyper Audio system described by Petrelli and Not in this special issue the corresponding infrared signal is detected when the visitor approaches an exhibit. Moreover, the system can be activated selecting a displayed link on a palmtop computer with a pen. Similarly, the user of the ec(h)o system described by Hatala and Wakkary (in this issue) uses a wooden cube for selecting audio presentations.

One of the main objectives of the LISTEN project was to avoid any portable device or remote control for the user except for the headphones and the movement in physical space. For the personalization process, this means that the user’s movements are the only interface for interaction. The personalization process in LISTEN is based on the context dimensions of time, position, and head orientation and on an extensive amount of annotation of both visual items and the different sound items that are to be played for each exhibit. The use of enhanced localization technologies allows precise user tracking in physical space, which delivers the knowledge necessary to adapt the system’s behavior.

Figure 6 provides an overview of how the abstract model proposed in the Context Toolkit framework is instantiated for the LISTEN system. The following paragraphs describe this instantiation in more detail.

4.2.1. *Contextua1ization*

The utilization of implicit feedback is an important issue for the realization of a personalized immersive environment. The only information on which the entire personalization process can be based is the spatial position and the head orientation. Both values are delivered by a fine-grained tracking system (ARTracking, 2005). The third important variable is time: Snapshots of the current context are time-stamped, so that an interaction history database is built up. The annotations for the sound items and the visual objects stored in the content management system cover matters like:

- The relation to the physical space (i.e. location and focus);
- classifications in terms of phases of work, image genre, or aspects of art technique;
- more subjective descriptions of the objects (e.g., in terms of the mood they reflect);
- sound effects that might match the type of content that the objects address (e.g., emotional impacts or dramaturgy);
- the link to a successor, if any.

In particular, speech sound items could be further classified into subcategories like Citation, Artistic Collage, Diary Readings, Letter, and Newspaper Articla to describe their style of presentation.

The location model allows the LISTEN system to interpret the user’s position and head orientation and gain values for the context attributes ‘location’ and

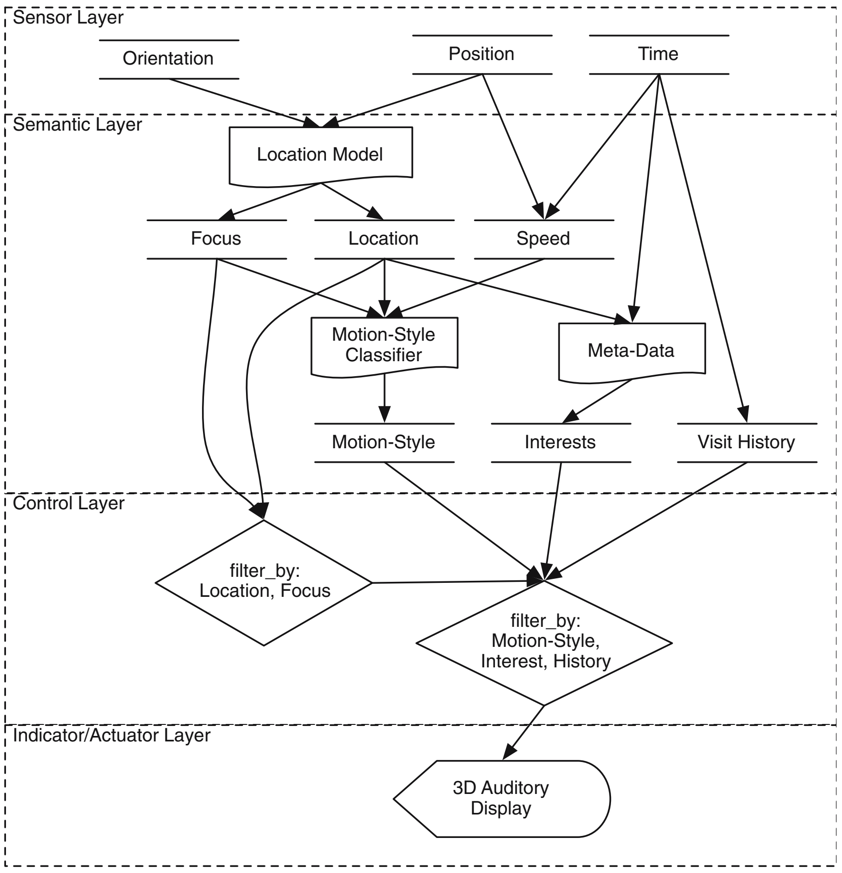


Figure 6. Instantiation of the context toolkit for the LISTEN audio guide.

'focus'. This model maps the position to virtual zones and the orientation to the identifiers of visual objects. An alternation in the values of these two context attributes triggers a prefiltering process, which retrieves sound items from the database that include these two values. For the selection of an appropriate item from the list of results, the subsequent personalization process is activated.

4.2.2. *Personalization*

From the list generated by the contextualization process, the best-suited sound item is selected on the basis of the user's interaction history, motion style, and interests. The analysis of the interaction history plays an important role in the adaptation of the system. To offer a corresponding range of content choices the visitor's interest needs to be infered from the interaction with the system. The ec(h)o system (this issue) of Hatala and Wakkary integrates the movement interaction with the explicit visitor’s content selection using a wooden cube. Because the LISTEN system did not make use of explicit feedback from the user, statistical models calculate the average time users spend in front of a painting and allow for the determination of points of interest in the exhibition. A comparison of the user’s spatial position with the time yields the user’s speed of movement. From this information, we can derive the motion style as a stereotypical behavior of the user. The concept of stereotypes is well known in the field of user modeling as a way of categorizing users in terms of their attributes or behavior (Rich, 1989). In a museum environment, it is not trivial to define meaningful stereotypes. We found it useful to introduce the following motion styles, which can be seen as representing possible ways of looking at exhibits:

– sauntering around
– goal-driven directed movement
– standing, focused
– standing, unfocused

The motion styles are defined in XML in a way similar to that of the definition of rules. Each change in the user’s context defined by the rules triggers the next determination of the best-suited stereotype.

A score-based interest model induces a procedural aspect concerning time, space and metainformation. We followed the approach of Pazzani and Billsus (1997) and assumed that the more time the visitor spends with a specific exhibit, the more she likes it. What the user appears to like is expressed in terms of the characteristics that all visual objects and sound items are metatagged with. The metadata are transferred into the interest model, and scores are given to each metadata attribute depending on the time that the user spends with aspecific visual or acoustic object.

On the basis of the interaction history, the motion styles, and the interest model, the system is able to adapt the auditory scenery and to generate different sound presentations.

5. Lessons Learned

From the systems presented here and from other prototypes we have built in Fraunhofer FIT, we learned a lot about designing and implementing personalized and contextualized systems and applications. Furthermore, through the work new open questions arose. We plan to continue our work with special focus on those new issues and to integrate the lessons learned from the previous work. Some main questions and lessons learned are summarized and stated here:

– The integration of multiple sensor values allows for more valid user models and enables new forms of contextualized interaction and user model acquisition in pervasive computing. Nevertheless the identification of the right combination of sensors is vital in the design of context management applications. Even if the right sensors are used and integrated, their scaling onto semantic layer attributes and triggers can be very important; in particular, it can have unexpected effects on the system behavior.

– Because sensors put a high load on computing and network infrastructures, it is important to have efficient multilevel filtering and to combine contextualization and personalization filters on different levels. It is useful to have the concept of an integrated sensor server that collects sensors that are available in the environment, a server that allows for the reuse and distribution of sensors with multiple entities and for the integration of existing infrastructure sensors.
– For the combination of personalization and contextualized information and technology support, different displays or actuators/indicators and their characteristics become important. The channeling of information to different displays raises not only important technical questions but also questions about the privacy of information and displays.
– Designing end user tools for building contextualized applications must follow very simple approaches, with visual approaches probably being most suitable in general. As our first prototypes seemed to be barely understandable for external users, we are currently prototyping different user interfaces that should enable end users to integrate entities and sensors and to define actions and rules for contextualized applications in a simple way. Nevertheless, the inferencing process from low-level sensor values to higher-level user model attributes is often highly complex and domain-specific. The instant combination of content and context snapshots as realized in the mobile collector seems to be a promising approach for user interaction. Currently, this approach has some limitations when it comes to defining actions and more complex system behavior.
– A framework approach for personalized and contextualized applications seems highly promising in terms of the reuse of libraries for sensor and tracking devices, as well as the use of rendering libraries for different displays. The modeling on the semantic layer and the control layer is still in an early stage, and we plan to investigate it in more depth. Even with simple rule sets, a system can get so complex that the consequences of particular rules cannot be predicted clearly.

6. Conclusions and Future Work

The work presented gives an overview of current developments towards a context management system at Fraunhofer FIT. Within a functional layer architecture, different layers have been identified and prototypically implemented that support sensor data management, context abstraction, and the control of actuator output. We have described implementations of sample applications for intelligent information distribution and selection, as well as the collection and connection of context data with contents. Especially on the semantic enrichment layer and the control layer, we are currently experimenting with different approaches for generalization and inferencing mechanisms. From our experience, we see that even a simple set of rules defined by end users can have very complex output.

In our future work, we plan to build a variety of different applications and tools based on the described infrastructure and layer architecture, so as to be able to evaluate the approach more thoroughly and to refine it. Furthermore, improved methods for the integration of static and dynamic contexts and for the combination of user data and environmental data have to be worked out. The demand for context-sensitive functionalities constitutes a crucial challenge for system developers as well as for product managers, application designers and system integrators. They have to deal with the problem of heterogeneity, which is found in hardware and software protocols, different mobile and wireless environments, and the services of multiple providers. At the same time, managers, developers, and system integrators have to react to shortened delivery times in a competitive and dynamic market. In order to integrate the development of the basic components by software engineers with the tailoring of components to build the final application logic, we plan to provide a framework that is completely configurable via XML and the user front-end by persons with little or no training in computer science.

References

ARTracking: 2005, A.R.T. GmbH – Your Expert for Infrared Optical Tracking Systems. http://www.ar-tracking.de/

Brown, P. J.: 1996, The Stick-e Document: A framework for creating context-aware applications. *Proceedings of the International Conference on Electronic Documents, Document Manipulation, and Document Dissemination Publishing*, Palo Alto, CA, pp. 259–272.

Brusilovsky, P.: 1996, Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction* **6**(2–3), 87–129.

Dey, A. K. and Abowd, G. D.: 1999, Towards a better understanding of context and context awareness. *Technical Report GIT-GVU-99-22*, College of Computing, Georgia Institute of Technology.

Dey, A. K., Abowd, G. D. and Salber, D.: 2001, A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction Journal* **16**(2–4), 97–166.

Dieterich, H., Malinowski U., Kuehme, T. and Schneider-Hufschmidt, M.: 1993, State of the art in adaptive user interface. In: M. Schneider-Hufschmidt, T. Kuehme, and V. Malinowski (eds.): *Adaptive User Interfaces*. North-Holland, Amsterdam, Netherlands, pp. 13–48.

Ebling, M. R., Hunt, G. D. H. and Lei, H.: 2001, Issues for context services for pervasive computing. *Workshop on Middleware for Mobile Computing*, Heidelberg, Germany.

Eckel, G.: 2001, LISTEN – Augmenting every day environments with interactive soundscapes. *Proceedings of the 13 Spring Days Workshop 'Moving between the physical and the digital: exploring and developing new forms of mixed reality user experience*', Porto, Portugal.

Ekahau: 2005, Ekahau – The Most Accurate Wi-Fi Positioning. http://www.ekahau.com/

Fox, T.: 2001, Präsentation neuer interaktiver Lehrmedien in der Sonographie. *Proceedings of the 25th Dreiländertreffen DEGUM SGUM ÖGUM*, Nürnberg, Germany.

Gossmann, J. and Specht, M.: 2001, Location Models for Augmented Environments. *Proceedings of the Workshop on Location Modeling for Ubiquitous Computing*, Atlanta, USA, pp. 94–99.

Gross, T. and Specht, M.: 2001, Awareness in Context-Aware Information Systems. *Mensch & Computer – 1. Fachübergreifende Konferenz*, Bad Honnef, Germany, Teubner-Verlag, pp. 173–182.

Grunst, G., Fox, T., Quast, K. -J. and Redel, D. A.: 1995, Szenische Enablingsysteme – Trainingsumgebungen in der Echokardiographie. In: U. Glowalla, E. Engelmann, A. de Kemp, G. Rosbach, and E. Schoop (eds.): *Deutscher Multimedia Kongreß'95, Auffahrt zum Information Highway Highway*, Springer-Verlag, pp. 174–178.

Henricksen, K.: 2003, A Framework for Context-Aware Pervasive Computing Applications. Ph.D. thesis, University of Queensland, Queensland.

Jameson, A.: 1999, User-adaptive systems: An integrative overview. *Proceedings of the Tutorial Presented at Seventh International Conference on User Modeling*, Banff, Canada.

Jameson, A.: 2001a, Modeling both the context and the user, *Personal Technologies* **5**(1), 29–33.

Jameson, A.: 2001b, User-adaptive and other smart adaptive systems: Possible synergies. *Proceedings of the First EUNITE Symposium,* Tenerife, Spain, pp. 13–14.

Jameson, A.: 2003, Adaptive interfaces and agents. In: J. A. Jacko and A. Sears (eds.): *Human-computer Interaction Handbook.* Mahwah, NJ: Erlbaum, pp. 305–330.

Kobsa, A., Koenemann, J. and Pohl, W.: 2001, Personalized hypermedia presentation techniques for improving online customer relationships. *The Knowledge Engineering Review* **16**(2), 111–155.

Kravcik, M., Kaibel, A., Specht, M. and Terrenghi, L.: 2004, Mobile collector for field trips. *Journal Educational Technology & Society* **7**(2), 25–33.

Lei, H., Sow, D. M., Davis, J. S., Banavar, G. and Ebling, M. R.: 2002, The design and applications of a context service. *ACM SIGMOBILE Mobile Computing and Communications Review* **6**(4), 45–55.

LISTEN: 2005, LISTEN – Augmenting everyday environments through interactive soundscapes. http://listen.imk.fraunhofer.de.

Oppermann, R.: 1994, Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software. Hilldale, New Jersey: Lawrence Erlbaum Associates.

Oppermann, R. and Thomas, C. G.: 1996, Supporting learning as an iterative process in a social context. *Proceedings of the European Conference on Artificial Intelligence in Education* Lisboa, Portugal, pp. 150–156.

Oppermann, R. and Specht, M.: 2000, A context-sensitive nomadic exhibition guide. *Proceedings of the Second Symposium on Handheld and Ubiquitous Computing*, Bristol, UK, pp. 127–142.

Pazzani, M. J. and Billsus, D.: 1997, Learning and revising user profiles: The identification of interesting web sites. *Machine Learning* **27**, 313–331.

Rich E.: 1989, Stereotypes and User Modeling. In: A. Kobsa, and W. Wahlster, (eds.): *User Models in Dialog Systems*, Springer-Verlag, pp. 35–51.

Johnson, W. L., Rickel, J., Stiles, R. and Munro, A.: 1998, Integrating pedagogical agents into virtual environments. *Presence* **7**(6), 523–546.

Salber D., Dey, A. K. and Abowd, G. D.: 1999, The context toolkit: Aiding the development of context-enabled-applications. *Proceedings of the 1999 Conference on Human Factor in Computing Systems,* Pittsburgh, PA, pp. 434–441.

Schilit, B. N.: 1995, System architecture for context-aware mobile computing. Ph.D. Thesis, Columbia University.

Schilit, B. N., Adams, N. I., and Want, R.: 1994, Context-Aware computing applications. *Proceedings of the Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, pp. 85–90.

Schmidt, A.: 2002, Ubiquitous computing in context. Ph.D. Thesis, Lancaster University UK.

Unnützer P.: 2001, LISTEN Kunstmuseum Bonn. *KUNSTFORUM International* **155**, 469–470.

Weiser M.: 1991, The Computer for the twenty-first-century. *Scientific American* **265**, 94–104.

Weiser M.: 1994, The world is not a desktop. *Interactions* **1**, 7–8.

Wulf, V.: 2000, Exploration environments: supporting users to learn groupware are functions. *Interacting with Computer* **13**(2), 265–299.

Zimmermann, A., Lorenz, A. and Specht, M.: 2002, Reasoning from contexts. *Proceedings of the 10th Workshop on Adaptivity and User modeling in Interactive Systems*, Hannover, Germany, pp. 114–120.

Zimmermann, A. and Lorenz, A.: 2003, Listen: Contextualized presentation for audio-augmented environments. *Proceedings of the 11th Workshop on Adaptivity and User Modeling in Interactive Systems*, Karlsruhe, Germany, pp. 351–357.

Authors' Vitae

Andreas Zimmermann

Andreas Zimmermann is a research associate and PhD candidate at the Fraunhofer Institute for Applied Information Technology in Sankt Augustin (Germany). He received a MS in Computer Science from the University of Kaiserslautern (Germany) in 2000. After one year of business work he joined the research group "Information in Context" to acquire his doctor's degree in context-management. He has got a strong research background in artificial intelligence and his further research interests include areas like user modeling, personalization, contextualization and nomadic systems. He was responsible for user modeling and implementation of context-awareness in the EU-founded LISTEN-project.

Dr. Marcus Specht

Marcus Specht received his Diploma in Psychology in 1995 and a Dissertation from the University of Trier in 1998 on adaptive learning technology. From 1998 until 2005 he worked as a senior researcher at the Fraunhofer Institute for Applied Information Technology (FIT). He has rich experience in educational and contextualized technologies for learning and personalized services. His main research interests are adaptive systems, contextualized computing, and intelligent interfaces. He was involved in several national and European projects for applying contextualization and personalization in the fields of e-learning and collaborative systems. Since 2005 he works as Professor for educational technologies at the Open University of the Netherlands.

Andreas Lorenz

Andreas Lorenz completed his master's degree in computer science at the University of Kaiserslautern (Germany) in 2001, and joined the research group "Information in context" at the Fraunhofer Institute for Applied Information Technology in Sankt Augustin (Germany) in spring 2002. He is a research associate and PhD candidate in the research field of multi-agent systems. His further research interests include user-adaptive systems, mobile and nomadic systems, and software engineering. He was responsible for software-design and implementation in the LISTEN-project.