Useware modeling for ambient intelligent production environments

Daniel Görlich
University of Kaiserslautern
Center for Human-Machine-Interaction
67663 Kaiserslautern
goerlich@mv.uni-kl.de

Kai Breiner
University of Kaiserslautern
Software Engineering Research Group
67663 Kaiserslautern
breiner@informatik.uni-kl.de

ABSTRACT
The impact of user interface quality has grown in software systems engineering, and will grow further with upcoming new paradigms such as Ambient Intelligence or Ubiquitous Computing, which confront the production industry with a huge diversity of new usage situations. In this paper, we will show the adaptation of a task-oriented useware modeling language, which is employed in the model-based useware development process, to future paradigms by extending its existing models with respect to the new upcoming requirements. This language reflects several user groups' tasks and user interface structure preferences in a common use model described in a system-independent language. It is being enhanced to describe spatial relations, connections between device compounds, and different ways of fulfilling tasks within different interaction zones. For the future, this model is intended to be used for the run-time generation of user interfaces for adaptive software and intelligent environments, especially in the area of production and manufacturing.

Categories and Subject Descriptors
D.5.2 [**User Interfaces**]: Theory and methods, User-centered design

General Terms
Performance, Design, Reliability, Experimentation, Human Factors, Languages.

Keywords
Useware, User Interfaces, Ambient Intelligence, Intelligent Production Environments.

1. INTRODUCTION
The level of acceptance of a user interface depends largely on its ease and convenience of use. A user can work with a technical device more efficiently when the user interface is tailored to the users' needs, on the one hand, and to their abilities on the other hand. Therefore, during a systematic development process, the users' needs, preferences, tasks, and mental models have to be surveyed, in order to subsequently deploy them into the development of a task-oriented and user-friendly device that will be as convenient as possible to use.

Still, people think and act quite differently, even when they perform the same task. Their personal requirements may depend on a large variety of influences ranging from their qualification, their area of activity and their tasks, up to rapidly changing conditions such as mood, time of day, current location, or recent events. In production environments, it is nowadays common to train employees in the operation of devices and restrict their access to safety-critical device functions, so all users (operators) are taught how to handle a device in advance. In a private environment, e.g., at home, however, users often have nothing more than a manual describing the functionality of a certain device – and they often refuse to read it right away.

Obviously, a developer of consumer goods must design the appliance in an intuitive way for a large variety of users. Nearly all kinds of home appliances are already equipped with computing power or are being replaced with equivalent – or even higher valued – technical devices. Hundreds of so-called "smart homes" and "living assistance scenarios" all around the world, centers of excellence regarding the technologically modern way of life, demonstrate networked, adaptable, "smart" devices that can be personalized or even actively and automatically adapt themselves to their users, resulting in environments proactively supporting the users during their daily lives, or are even able of detecting and thereby preventing critical situations [6]. With the *SmartFactory*^{KL} in Kaiserslautern/Germany, a first intelligent factory has been built to provide a testbed and demonstration platform for smart technologies based on ad-hoc networks, dynamic system collaboration, and context-adaptive human-machine interaction systems. In the future, these systems will provide information at any time and in any place, making them more flexible and remotely accessible. This, however, results in a huge number of usage situations dependent on user, situation, machine, environmental conditions, task, etc. A smart device's flexibility may thus become a disadvantage when information is not presented properly in terms of format, structure, and a context- and location-sensitive, task-oriented way [1].

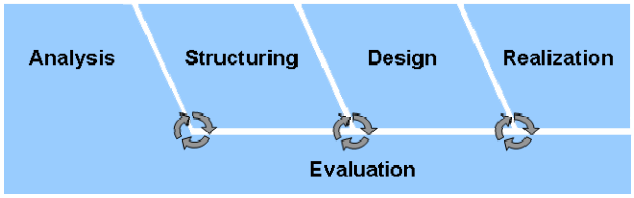

Figure 1. Systematic useware development process [4].

Within a systematic useware development process (see Figure 1), the Center for Human-Machine-Interaction (ZMMI) has applied its Useware Markup Language (useML, see Figure 2) harmonizing multiple users' mental task models in a common use model, in numerous successful joint ventures and industrial projects. With the launch of the *SmartFactory*^{KL} [8], it carries the future interaction paradigm of Ambient Intelligence into the area of production industry. This paper focuses on the description of

the enhancement of the Useware Markup Language to meet the mentioned challenges in future production environments, and presents the current state-of-the-research project GaBi (German abbreviation for "Generating task-oriented user interfaces in intelligent production environments"), which aims at developing possible solutions for human-machine interfaces in production facilities ten years from now – in the year 2017.

2. PROJECT STATUS

The research project GaBi aims at adapting and enhancing the already existing and approved Useware Markup Language, as well as at establishing a repository of usability patterns (tailored to the production industry), which will be used during the model-based generation of user interfaces at run-time. The project is scheduled for 2 (+1) years and is entering its second year.

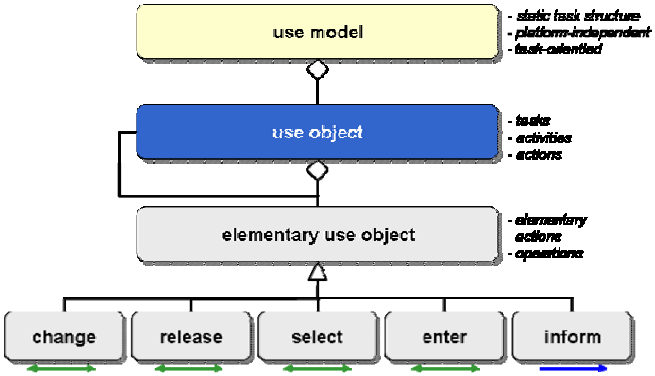

Figure 2. Classic useML scheme according to [4] and [5].

The early analysis of user requirements for interactions with intelligent environments was complemented with a so-called Future Workshop, which took place on February 13th, 2007. It was attended by participants from different manufacturing companies and research institutes, including requirements analysts, data protection officers, philosophers, software engineers, jurisprudents, and usability experts. All these specialists gave brief overviews of the current state-of-the-art in human-machine interaction from their own professional points of view, and recapped deficiencies of today's systems. After collecting visions for the year 2017 in a second phase of the workshop, these ideas were finally evaluated against the identified deficiencies with respect to the feasibility of their implementation. The results were incorporated into a scenario describing natural human-machine interaction in a production facility in the year 2017 and into the extension of the use model [7].

One fact that emerged, among others, was that there will be no one-fits-all solution meeting all kinds of tasks and personal preferences. The experts instead pleaded for more flexible systems, which adapt themselves to each user's needs and the current context of use automatically and further provide the possibility of being adapted manually by the user according to his personal preferences. Still, human-machine interfaces should not be too flexible, in order to still meet safety specifications and offer rarely needed, but important functions, for example. Therefore, some basic standards are needed to increase the recognition value of a system to the user, and to facilitate the use of different software products by increasing their compatibility. This tightrope walk between automatic self-adaptation to the individual user on the one hand, and the standardization of user interfaces on the other hand requires a well-adjusted combination of model-based user interface generation and previously defined user interface components, or the use of so-called usability patterns.

Another common mistake pointed out by the experts concerned the design of easily and intuitively usable interfaces. While simplification by reduction of complexity is an honorable goal, the systems must rather present interfaces corresponding to the task, qualification, preferences, and needs of the individual user. In this context, reducing complexity is not always the best way to optimize a user interface, because highly qualified users often need or simply want to access extra information and functionalities. From this point of view, it becomes evident that users make different, but always high demands on technical devices. They tend to interact with the same device in different ways, so that developers are advised to consider different types of users throughout the whole development process.

3. MODELING WITH useML

Originally invented by [5] to structure user interfaces in a user- and task-oriented useware development process (see Figure 1), the XML-based Useware Markup Language (useML) arranges user or machine operator tasks in a hierarchy of abstract use objects (UO) and five types of different elementary use objects (EUO), which are well-suited for today's machine operations. The overall model will be arranged as a tree, using UOs as nodes and EUOs at the leaf level. Starting from a high-level task description at the root node, the UOs are refined from high-level abstract tasks into more concrete subtasks, activities, actions, and, finally, elementary actions or operations such as pressing a button, entering a value, or reading displayed information from a screen (see Figure 2), which can be directly mapped to the corresponding functionality of a certain device. This task model is platform- and modality-independent and self-sufficient in terms of concrete design and realization, which are added during a later phase of the useware development process. Based on the multitude of task models determined from (potential) users in the analysis phase, the use model is integrated from these models through harmonization and systematic structuring. The use model is designed to incorporate several user groups' different approaches to their specific tasks in a single model, which can be filtered by attributes such as user group and device type. It is also possible to build one single use model for a whole family of devices, i.e., a company's product line. This way, all devices developed on the basis of the same use model will share a consistent, recognizable, and thus intuitive interaction scheme, which might also cross platforms: Only in the subsequent design and realization phases following the use model structuring (see Figure 2), the actual target platforms and interaction modalities are derived, which might be Graphical User Interfaces (GUI) or speech interfaces.

Although the Useware Markup Language is well suited for the development of single devices or device families, it was not designed to describe more complex production processes or even facilities incorporating a high number of devices or machines of different types. Therefore, the GaBi project aims at improving the Useware Markup Language by expanding its scope, providing

compatibility for future interaction paradigms such as Ambient Intelligence or Ubiquitous Computing. Such progressive environments will comprise hundreds or even thousands of cooperating devices and embedded systems with which we will quite naturally interact. Traditional interaction paradigms such as GUIs dedicated to a single device may not be sufficient any longer, and users may employ numerous devices at the same time to fulfill their tasks. An appropriate use model therefore must contain a spatial representation of the relevant environments or spaces, as well as a description of devices and device compounds involved in all potential users' works.

Within the GaBi project, we therefore adapted the use model scheme to these requirements (compare Figures 2 and 3). It now includes relations between locations, devices, and users, beginning with a hierarchical structure of (mobile or stationary) organizational rooms. The meets relation is used to model adjoining rooms. Using the joint relation, rooms can be structured into physical or logical subspaces. Completely different rooms are expressed by the disjoint relation. All rooms are identified by names, but can also have unique IDs, coordinates, or descriptions. As just mentioned, the rooms do not have to exist as physical rooms in the real world, but can also identify purely logical (i.e., organization) rooms.

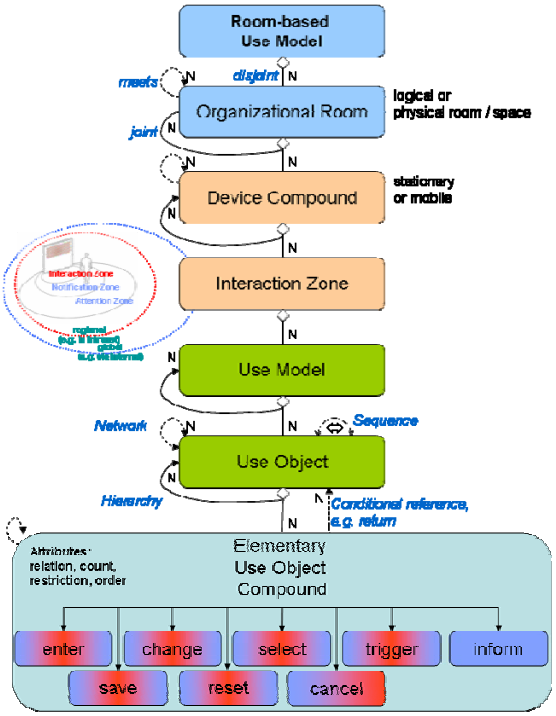

Figure 3. Integrated, room-based use model.

Within every room, multiple (mobile or stationary) device compounds can be located. Again, each device compound can recursively comprise other device compounds, devices, components, or parts. If a device is subordinated to or is a child element of another device, respectively, it is considered to be a part of that device. If a mobile device is subordinate to another device, it is considered to work as long as it is an integral part of its parent device. Mobile devices can also be direct children of organizational rooms; in this case, they can only be used within these rooms to fulfill the tasks modeled later on.

Furthermore, each device (compound) can be operated differently depending on the interaction zone that its user or operator is in. For example, a remote control panel for a robot picker arm might be configured to control the robot only within an effective range of a few meters, while it can request status information from a wider distance or even remotely via intranet. Such interaction zones can be defined for each device or device compound, but always belong to at least one of three abstract zones, i.e., the local (at or near the device), regional (within a sealed-off data network), or global zone. The local zone is further subdivided into an interaction zone, where the user can operate the device, a notification zone, where he can still gather information presented by the device (i.e., a display or loudspeaker), and finally an attention zone, where he cannot yet gather detailed information, but may notice warning signs, blinking lights, unexpected messages, color codes, and so on. In certain cases, these zones can be identical, e.g., when a user possesses a remote control that lets him operate a device from a distance that exceeds his physical limitations. Under normal circumstances, however, the zones would overlap as shown in Figure 3.

For each interaction zone, every device in any room should possess at least one use model, and preferably even exactly one. This, however, is not the classic use model anymore as invented by [5] and described above (see Figure 2); it has been extended to not only span hierarchies of UOs and EUOs. Rather, sequences of use objects can be defined, and elementary use objects can be combined into compounds (EUOCs) with elaborated selection and execution rules. Further, any UO can be linked to other ones, even in other subtrees of the hierarchy, thereby spanning a network of associated use objects within the classical hierarchy (see Figure 3).

Within an EUOC, execution rules can define how many of the given EUOs can or must be executed in which order. For example, it can be stated that at least 3 of all 5 components in a compound must be executed sequentially. Finally, conditional references between EUOCs and their parent UOs can be enclosed, such as a break or post condition.

4. CREATING THE UI

Based on this integrated use model, by applying platform-specific (stylesheet) transformations, it is possible to build the corresponding UI. Thus, only one use model is needed to describe the human-machine interaction independently of the device that will be used to communicate with the user. An important property here is the fact that the UI can be created at development time, can then be deployed at the destination platform and used there.

Due to the highly dynamical environment, new interaction devices can be integrated seamlessly at any time. Thus, the use model needs to be reinterpreted accordingly. Therefore, it is important to alter the appearance of the UI at run-time, integrating new functionality to reflect the current configuration of the whole production environment. Unlike the previous, single-device approach, it is self-evident that while the usage situation is no longer static at run-time, the UI code has to be generated as well as deployed and executed at run-time. In a previous approach, which generated a model-based user interface at run-time [2], we observed that the time consumption of the entire process is very high. This means that performing all activities necessary for providing a complete user interface built from an abstract

description (task model and usage situation) takes far too long to be really usable.

This raised the idea of every device to be integrated into this environment already having to provide a set of user interface components, with each being well developed for a certain pre-defined platform. At run-time, the corresponding user interface components have to be transferred to the interaction device and there need to be combined into an integrated UI acting as a universal controller. When a new device appears in this environment, only the new UI component needs to be deployed at the interaction device.

In useML, such UI components represent the implementation of EBOCs or even entire UOs, depending on the current granularity. Originally, an EBOC consists of a description of how the human-machine interaction has to be performed and which steps can be mapped directly to the device's functionality. Since each UI component is a complete encapsulated interaction unit, there is no longer a need for explicit modeling. Therefore, the enhanced useML will accept also components instead of EBOCs and UOs.

Task	Priority
Maintainance	2
Improvement	3
Redesign	4
Production	1

Task	Priority ▼
Production	1
Maintenance	2
Improvement	3
Redesign	4

Figure 4. Effect of usability patterns.

Another important aspect concerns the so-called usability patterns and usability guidelines [3], which capture "best practice" knowledge in Usability Engineering. There are different types of these patterns, e.g., some describing the interaction of humans and machines, others including layout descriptions. Figure 4 shows the effect of the alternating-row-color and the table-header pattern on the example of a regular table. The enhanced useML allows for potentially applicable patterns being annotated to entire sub-trees, fulfilling certain pre-conditions.

5. SUMMARY

After our requirements research comprising the Future Workshop with a heterogeneous set of participants, the fact emerged that user interfaces in future intelligent production environments have to be task-oriented in order to achieve a reduction of complexity compared to the human-machine devices that are currently used in such factories. Therefore, we extended the approved useML according to the new paradigm of Ambient Intelligence in future production environments. Hence, we included the possibility of structuring the spatial environment (spatial use model), which is essential to these context information sensitive systems. Also, another important factor is the configuration model of interacting devices, which was also included in the altered model. Now every device can be equipped with a separate use model, describing its own way of interaction.

6. FUTURE WORK

Nevertheless, many problems remain regarding the introduction of the Ambient Intelligence paradigm into intelligent production environments. For example, the fact that the device configuration will change at run-time, according to the factory configuration, needs to be reflected. Therefore, methods have to be developed that address the integration of actual context information into the transformation process of the model.

Another important issue is the implementation of this adaptive system. Due to the awareness that is not effective to create an entire UI from scratch at run-time, we already mentioned the idea of composing the UI from single components, which need to be provided in the first place (e.g., by the devices themselves). The composition of the single components is also a topic of interest, as is the way the composition will be influenced by usability patterns. These design guidelines already exist, but are neither formalized in a machine readable way, nor have explicit patterns been identified for production environments.

Finally, an important step in our evaluation process will be a feasibility study implementing and testing our concept. For this purpose, the *SmartFactory*^{KL} is the ideal testbed for simulating future production environments.

7. ACKNOWLEDGMENTS

This work was supported in parts by the GaBi project at the University of Kaiserslautern, which is funded by the German Research Foundation (DFG).

8. REFERENCES

[1] Bödcher, A., Mukasa K. and Zühlke D. Capturing Common and Variable Design Aspects for Ubiquitous Computing with MB-UID. In *Proceedings of the Workshop on Model Driven Development of Advanced User Interfaces*. Montego Bay, Jamaica, 2005.

[2] Trapp, M., and Schmettow, M. Consistency in Use through Model-based User Interface Development, CHI 2006, *Workshop on The Multiple Faces of Consistency*, Montreal, Canada, 2006.

[3] Welie, M. v., Veer, G. C. v. d. and Eliëns, A. Patterns as Tools for User Interface Design. In *International Workshop on Tools for Working with Guidelines*, (Biarritz, France, 2000), 313-324.

[4] Zuehlke, D. *Useware-Engineering für technische Systeme.* Springer, Berlin, 2004.

[5] Reuther, A. *useML – Systematische Entwicklung von Maschinenbediensystemen mit XML.* Ph.D. Thesis, University of Kaiserslautern, 2003.

[6] Nehmer, J., Becker, M., Karshmer, A., and Lamm, R. Living assistance systems: an ambient intelligence approach. In *Proceeding of the 28th international Conference on Software Engineering* (Shanghai, China, May 20 - 28, 2006). ICSE '06. ACM Press, New York, NY, 43-50.

[7] Görlich, D., and Breiner, K. Intelligent task-oriented user interfaces in production environments. In *1st International Workshop on Model-Driven User-Centric Design & Engineering* (Seoul, Korea, September 2007). IFAC, 2007.

[8] Pohlmann, E. G., Bödcher, A., and Zühlke, D. *SmartFactory*^{KL} – Informationstechnik für die Fabrik der Zukunft. In *atp – Automatisierungstechnische Praxis* 47(12), 2005, S. 48-52.