Reinhard Oppermann und Andreas Zimmermann

Context Adaptive Systems

Kontextadaptivität

Adaptivität_Adaptierbarkeit_Kontext_Situiertheit_Benutzermodell_Systemmodell_Aufgabenmodell

Zusammenfassung. Eine Analyse der Adaptivität im Rahmen kontextsensitiver Systeme wird vorgestellt, die die Langzeitperspektive des Kontext mit der aktuellen Situiertheit von Handeln in einer speziellen Raum- und Zeit-Stelle in Beziehung setzt. Drei Funktionen von Adaptivität werden in diesem Kontext behandelt, die den Adaptivitätsprozess bestimmen: die sensorische Funktion, die Merkmale der Mensch-Computer-Interaktion aufnimmt, die eine Adaption erfordern; die Inferenzfunktion, die aufgrund von Modellen des Benutzers, der Aufgabe, des Systems und der Umgebung eine Schlussfolgerung über die zu vollziehende Adaption trifft; und die Effektorfunktion, die die zuvor ermittelte Anpassung vollzieht. Schließlich werden die Akteure, die diesen Prozess ausführen, besprochen, die Benutzer (besonders für die adaptierbare Version von Adaptivität), das System (besonders für die adaptive Version von Adaptivität) und eine Kombination von beiden evtl. in Verbindung mit Domain-Experten, die das Aufgabenfeld und die Benutzer kennen und Hilfestellung für die Bestimmung und Ausführung von Adaptionen geben können.

Summary. An analysis of approaches to adaptive and context-aware systems will be given. Adaptation between the user needs and the system performance will be discussed reflecting the context of use. Context-awareness will be discussed affecting the adaptation in a long-term perspective; situation-awareness will be discussed affecting the adaptation in a specific point in time and space. Three functions are presented that determine the adaptation process: the sensory function collecting the internal and external stimuli for the adaptation; the inference function identifying the adaption mechanisms based on adaptation models; and the effector function performing the identified mechanisms. Finally the actors in the process of adaptation will be discussed like the user (adaptable version of adaptations), the system (adaptive version of adaptations) and a combination of both eventually together with an end-user expert.

1. The Need for Adaptation

A software system passes through a potentially long software engineering cycle (Booch et al., 1999, or Gilb, 1998) and before delivery requirement engineers, designers and developers realize the components of the system. However, it is impossible to anticipate the requirements of all users, and a single best or optimal system configuration is impossible. The active involvement of users and clear understanding of user and task requirements in the context of use is a challenge in the development of computer-based interactive systems for two reasons: first, the potential user groups are not known a priori, but can only be anticipated according to future scenarios; these groups need to be revised as the visions evolve because there may be various groups of potentially affected users. Second, the visions of the aspired project are far-sighted and not close to users' current experiences; therefore, users may not be confident and precise about their needs concerning this future system. With their norm for "Human-centred design processes for interactive systems" ISO 9241, part 210, (formerly 13407) the International Organization for Standardization gives guidance on user-centred design activities throughout the life cycle of computer-based interactive systems. One of the core tasks of user-centred design is to negotiate and facilitate the communication across the well-known user-developer gap while acknowledging the different forms of expression and different requirements on each side. However, despite the implementation of a human-centred design process, some types of modern ap-

DOI 10.1524/icom.2011.0004

plications require instant adaptation due to their exposure to increasing contextual dynamics.

1.1 Adaptation and Adaptivity

Even if the user-centred design process implemented in a project guarantees a certain degree of user acceptance and yields a richer understanding of the context of use, the completed product's ability to adapt to changing conditions still plays a central role for a broad acceptance. The operational environment will change, the tasks will be distinct, the end-users will be heterogeneous, and their competences and expectations will evolve. It is impossible for developers to identify all possible requirements modifications. The dynamics of changing conditions shifts the customisation process of the system's characteristics from the development phase to its usage and operation phase and from the developer to the user.

For this reason, developers implement techniques of adaptation into the system in order to react to changing conditions as far as possible. There is an important differentiation between manually and automatically performed adaptation processes. The term adaptation decomposes into the two terms adaptivity and adaptability. Adaptivity indicates a system that adapts automatically to its users according to changing conditions; such a system is called an adaptive system. Adaptability refers to users that can substantially customise the system through tailoring activities by themselves; such a system is called an adaptable system. This distinction can be more fine-grained and complemented by activities dedicated to the specific actors in the entire adaptation process.

The aim of adaptivity is to have systems that adapt themselves to changes in user-related or environmental properties. Such automatically performed adaptations base on the evaluation of the user behaviour and assumed user needs, or taking explicit user input into account. The objective is to assist the users by proactively supplying what is actually needed. Adaptive systems try not to distract users from their primary task by searching and selecting information or services, as well as by performing extensive customisation tasks. Systems automatically adapt to situations where computers outperform the user (e.g. faster response time, identification of toxic gases) or the user is unable to perform a specific task and the computer takes over control (e.g. unsuited clothes, cognitive overload).

The aim of adaptability is to empower end-users with or without at least limited programming skills to customise or tailor computer systems according to their personal or task-specific requirements. Such adaptable systems allow for fast adaptations to dynamically changing requirements by letting the end-users put their domain-specific expertise to the task of system customisation. After the user identified a change of her needs, she is enabled to manually adapt the system's features. This approach provides adaptation methods and tools that are under the control of the user instead of the system. Therefore, adaptable systems enable users to override certain functions or correct decisions if the system's model of the user or the environment and the reality mismatch (e.g. refinement of preferences, change modality).

Adaptive and adaptable systems are complementary to each other (Oppermann, 2005). Both methods increase the match between user needs and system behaviour once the development of the system has been finished. Thus, the system is kept flexible during usage.

1.2 Context and Situation

Since the term context-aware computing was first introduced by Schilit and Theimer in 1994, a large number of further definitions of the terms context and context-awareness has been proposed in the area of computer science. Each of the provided definitions introduces a considerable amount of expert knowledge that needs to be incorporated in further research. Dey's definition (Dey, 2001) is intended to be adequately general to cover the work conducted by research into interaction based on context. In order to further constrain its universality, this general definition needs to be enclosed by a formal and an operational part.

Context is any information that can be used to characterize the situation of an entity (Dey, 2001). Elements used for the description of context information fall into five categories: individuality, activity, location, time, relations. The activity predominantly determines the relevancy of other context information in specific situations. Location and time primarily drive the establishing of relations to other entities that enable the exchange of context information among entities.

The formal additive constricts and clusters context information into five fundamental categories, which facilitates the handling of context in concrete applications and the specification of a context model. The operational additive of the definition qualifies the reference to a certain task and the exploitation of context information (Zimmermann et al., (2007)). In the literature, the situation is regarded as a part of context. Brézillon (2002) describes a situation as "a subpart of the overall context", and (Oppermann, 2005) defines situation as "the relevant context characteristics at a specific pointing time and space." These two perspectives form the basis of the definition of the term situation in a general form:

A situation is the state of a context at a certain point (or region) in space at a certain point (or interval) in time, identified by a name.

A situation is considered as a structured representation of a part of the context, which can be named and where location and time are used as spatio-temporal coordinates of this situation. Like a snapshot taken by a camera a situation captures the momentary profile of the context attributes. A situation does not include the long-term development of the context attributes (Oppermann, 2005), i.e. the history, while the context covers the whole history of the interaction, in order to establish continuity. However, the observation that each process can be subdivided in periods with sequences of momentary situations influenced Coutaz and Rey (2002) in their understanding of a context at a certain point in time as a composition of multiple situations over a period of time. Thus, no situation exists without history.

2. The Methodology of Adaptive Systems

This section introduces the methodology of adaptive systems and depicts the three functionalities within the adaptation process that serve as a structure of adap-

tivity. The intension of adaptive systems is the adaptation of the system behaviour based on inferred properties by the system. The adaptation of system services and performance includes three logically and temporally separable functionalities. We express these functionalities as a trilogy of the terms of a sensory, an inference and an effector function that build upon each other. Based on internal interaction sensors or external environment sensors, all incoming events are collected and reported to a central unit. This central analysing unit handles the incoming event messages, evaluates them and generates decisions or proposals on adaptation activities. The result consists of actions of adaptation executed by corresponding effectors.

The sensory function indicates the observation of the task performing users by the system. During this step the adaptive system registers all information collected about the user and about the context: the location, time, environment, domain, physical conditions and social actors, and records this data in a systematic and continuous way.

The inference function refers to the intelligent analysis of the accumulated data through statistical methods and learning algorithms. Hereby, different models like system model, domain model, user model, task model and environment model represent the knowledge needed for drawing inferences.

The effector function transfers the results of the inferences into respective options of operations adapting the functions or the interface of the system to the current user's needs.

The distinction of the three components, a sensory, an inference and an effector function is an analytical one. This distinction was motivated backwards from the end of the process to its beginning: The adaptations performed by the adaptive system (effector function) must be in line with the needs of the user in the current context of use. The system needs to have an idea about potentially executable adaptations. The kernel of the adaptive system, the inference component, contains mechanisms to infer these adaptive effects from relevant indicators. These indicators in turn are provided by the sensory function and serve as an information basis for inference algorithms to draw conclusions about the user behaviour, task and location, time, environment, domain, physical conditions and social actor characteristics. Like in the process of human apperception, perception categories are necessary, otherwise the recognition is "blind". The following sections illustrate this threefold structure of adaptive systems in more detail.

2.1 Sensory: Identifying Characteristic User Attributes

Changing conditions trigger the execution of an adaptation. Therefore, sensory mechanisms perform an "observation" of the user behavior and her environment, in the first step. Generally, the system needs to detect and record the characteristics respectively to the task over a certain time period.

Types of Characteristics

Many characteristics might be taken into account as catalysts for such an adaptation process. They can be clustered into three main categories: inter-individual, intra-individual and environmental differences.

- Inter-individual differences address varieties among several users along manifold dimensions. Physiological characteristics like disabilities are of major concern for application designers if they want to have their system accepted by a large community. The consideration of user preferences like language, colour schemes, modality of interaction, menu options or security properties, and numberless other personal favourite preferences is one of the most popular sources of adaptation and can be reused in different applications. An important prerequisite for the content adaptation is the awareness of the user's interests and disinterests. The user's interests configure filters for the information presented to the user. Additionally, a broad spectrum of psychological personality characteristics exists like emotions, self-confidence, motivation, beliefs or idols, which are difficult to assess automatically. The same holds for the user's level of factual, general and domain-specific knowledge (e.g. beginners or experts), which is a valuable source of adaptive operations.
- Intra-individual differences consider the evolution and further development of a single user, as well as the task over time. A static system falls short of changing user requirements as the user's activities and goals evolve. In an extreme case users are overstrained by the system in the beginning and perceive the same system as cumbersome and restricted as the user's expertise and needs increase. In the same manner, the need for a higher flexibility of computer systems is pushed by the changing of the tasks to be accomplished with such a system.
- Environmental differences result from the mobility of computing devices, applications and people, which leads to highly dynamic computing environments. Unlike desktop applications, which rely on a carefully configured and largely static set of resources, ubiquitous computing applications are subject to changes in available resources such as network connectivity and input and output devices. Moreover, they are frequently required to cooperate spontaneously and opportunistically with previously unknown software services in order to accomplish tasks on behalf of users. Thus, the environment surrounding an application and its user is a major source to justify adaptation operations.

Inter- and intra-individual differences particularly refer to adaptation effects based on changes in characteristics of the users. Jameson (2003) defines all systems that automatically perform an adaptation to the individual user in some nontrivial way as user-adaptive systems. User-adaptive systems must be able to observe the user's behaviour, generalise these observations and make assumptions about the user.

Acquiring Characteristics

Traditionally and factually the user's interaction with the system is the most important dimension for adaptivity. In here lie the biggest inter-individual and intra-individual varieties and in here lies the biggest need for adaptation during the usage of the system. Generally, over a certain time period the system needs to detect and record relevant indicators for adaptation

needs. The knowledge of how to acquire context information includes the selection of the appropriate acquisition method. These methods supply information that is provided explicitly by the user, implicitly through observation or from external sources: Questionnaires, tests, sensors or queries.

Questionnaires or forms present an effective means for capture information explicitly provided by the user. The user fills in questionnaires and delivers a variety of valuable information to the system. This sort of input events oftentimes enables the system to learn about the user's properties, preferences, interests, task, etc., since they are explicitly specified by the user. Like questionnaires, tests are filled in by the users, in order to make information available to the adaptive system. In general, tests serve as means for the determination of the user's knowledge level, competence and expertise. In addition, third party evaluation, assessments and results of tests (e.g. though tutors in learning systems) can be supplied to the system.

As additional sources of information and technical components, sensors measure physical or chemical properties of the users and their environments. Sensors quantify temperature, humidity, pressure, sound, lightness, magnetism, acceleration, force, and many more other properties of the environment. Furthermore, sensors measure the user's blood pressure, body temperature or eye movements, as well as the location of the user in- and outdoor. More recently, the sensors are equipped with built in microprocessors and are more and more capable of autonomously processing their signals (so called 'smart sensors' or 'smart dust' (Satyanarayanan, 2003)). In self-organizing networks (Culler and Mulder, 2004) sensor technologies builds ad-hoc sensor-networks and deliver requested information on demand. Furthermore, there exists a multiplicity of external sources like databases, pools or external applications that manage valuable information (e.g. user profiles, weather forecasts). Information cannot be gained exclusively through direct interaction of the user with the system or through observation of the user. Additional queries to external information sources may augment the system with a variety of valuable information.

The careful selection of appropriate acquisition methods requires a lot of knowledge, since one acquisition method might deliver valuable information for one domain, but might be useless in another. In addition, possible restrictions given by the law have influence on the selection process. Knowledge about the capability and limitations is indispensable for almost any acquisition method, and decisions on possible methods must base upon the following properties:

- Sample rate, i.e. the time span between two acquired values.
- Value range, i.e. the span between the extreme values
- Scale, i.e. the minimal difference between two values

Particularly for the sensing technologies, the availability of infrastructural aspects, like suitable technology, network connectivity, computing power, storage capacity, and costs, plays an important role in selecting an appropriate context acquisition method.

The adaptation of the system concerning the tasks to be processed and the technology to be used may often be done previous to the application of the system for a certain configuration. For the user the needs for adaptation often turn out during the application of the system. This is because several users with different dispositions make use of the system, respectively because during the system application the user changes due to familiarisation, training, mental fatigue etc. This identification logically and temporally takes place as the first step of the adaptation process. Indeed, its composition aligns with and implicitly depends on the second component, which is described in the following section.

2.2 Inference: Knowledge-Based Decision Finding

The second function, inference, implements the analysis of the data recorded by the sensory function on the basis of model assumptions on user needs, heuristics or ontology models of the application domain. The way the system reacts or adapts its behaviour is based on the interpretation of certain peculiarities and attributes of the user's personality, the degree of expertise or tasks collected by the sensory function. The information about the observed interactions serves as a source for inference mechanisms that identify the most appropriate adaptation operation the system should offer. Theories about personality and learning, or pragmatic considerations about efficiency are relevant for the decision finding process as well.

The implementation of knowledge-based decision finding requires derivation knowledge, which encompasses knowledge of how to combine and process several types of information in order to enrich already available information. The explicit input provided by the user, the values measured by sensors and the data requested from external sources offer a broad range of information about the properties of the users and their environments. In order to fill the models with meaningful information and to perform an adequate adaptation, this information may not be sufficient, and thus, parts of this information need to be analysed, assessed and interpreted. Basically, the extraction of such higher-level information is a complicated process because boundary conditions apply, double entendres possibly emerge, and potentially undefined or uncertain results arise.

Inference mechanisms

The following non-exhaustive list describes techniques for derivation:

- Statistical Models: The simplest way of data interpretation is based on statistical models, which are available as large software packages today. The use of statistical methods such as correlation, regression, clustering and time series analysis separates real effects from flukes, and thus, avoids misinterpretations of the user behaviour. Statistical coherences are depicted and backgrounds are uncovered that emerge from interdependencies between events.
- Fuzzy Logic: Fuzzy logic provides logic operations to process fuzzy value sets (Zadeh, 1965), while available knowledge incorporates the processing. These sets emerge from the fact that discreet values only can be seen as discreet within a reachable precision of measurement. Fuzzy sets are based on vague definitions of sets and fuzzy logic enables the determination of membership in vaguely defined sets.

Thus, fuzzy logic constitutes a means of expressing uncertainty.

- Data Mining: Data Mining goes one step further and refers to techniques for finding interesting and useful patterns and rules in huge data sets. The term data mining includes a number of technologies that allow for the analysis and prognosis of behaviour patterns and trends, and deliver insights and coherences that have been hidden so far. Very often, data mining technologies base on algorithms from the field of artificial intelligence, knowledge management and statistics. The appliance of these algorithms depends on the aim and the purpose of the desired analysis.
- Nearest Neighbour: The nearest neighbour algorithm (Duda and Hart, 1973) operates by storing examples or experiences in a training set. A new and unseen instance is classified through the assignment to the class of the most similar example. A prominent example of this algorithm is case-based reasoning. The aim of case-based reasoning is to copy the human way of solving new problems on the basis of solutions for similar problems of the past and to map this procedure to machine processes. Case-based reasoning (Aamodt and Plaza, 1994) may be suitable for problem areas, in which the knowledge of how a solution is created is poorly understood (Watson, 1998).
- Probabilistic Procedures: Probabilistic procedures like hidden Markov models or Bayesian networks offer a method of the representation of uncertain knowledge and resultant possible reasoning. They enable the compilation and representation of coherences, dependencies and independencies of objects in a probability network. Such a network contains the qualitative effects existing among the objects. Following the Bayesian theorem, the relevancy of an (unknown) object can be calculated on the basis of the sum of all probabilities of the effecting objects.
- Neuronal Networks: A neural network is an interconnected group of neurons that uses a mathematical model or computational model for information processing based on a connectionist approach to computation (Hertz et al., 1991). The weighted connections between the neurons change their structure based on external or internal information that flows through the network. During the training phase of the network the algorithm iteratively or recursively adapts the connection weights between the neurons based on the presentation of input/output data sets. These weights store the knowledge to the solution of specific problems.
- Evolutionary Algorithms: Evolutionary algorithms (Bäck, 1996; Bäck et al., 1997) are approaches for solving difficult optimisation and search problems and are geared to the well-known evolutionary theory: characteristic descriptions of problem solutions are coded and constitute so-called individuals that are assessed by a fitness function. Through repeated selection, recombination and elimination new and improved individuals are created consistently. In the course of generations the fitness of individuals is improved and optimised problem solutions are reached.

Inference mechanisms derive conclusions based solely on information that is already known. Statisticians have developed formal rules for inference from quantitative data and artificial intelligence researchers develop automated inference systems. The selection of appropriate inference mechanisms requires experience and knowledge about their proper implementation, application, training and further processing of their results. Inference techniques address wide areas like search problems, planning, logical deduction, optimisation, and approximation methods. The adaptation process of these techniques to a new problem domain needs to take into account aspects such as flexibility of the applied algorithm, robustness against uncertain input, performance and response time, learning aptitude, transparency, and understandability.

Modeling

The knowledge about the domain, the users and their tasks has to be represented in an easily accessible manner for the system. For this reason, the extracted information and derived knowledge are mapped to adequate models making the data and its meaning available for the system. The following models are necessary for adaptation purposes.

- System model: Within a complex application domain a model of the system is needed in order to better understand, shape and control complex decisions. This model describes the system as a whole and covers specifications about what the system is and how it works. A system is well defined and draws a clear dividing line between itself and the environment. It consists of elements that are related to each other und that give the system its structure. Therefore information about the following components is of major importance: Knowledge of the system about itself, functionalities of the system, platform, and possibilities for interaction and errors, structures and branches, and help facilities. The system model serves as a basis for the definition of a formal semantic for the above-mentioned components and for the abstract illustration of the system behaviour. In combination with the task model the system model enables the clarification of the user's role and thus, is the fundament for adaptive operations.
- Domain model: The types of domains for adaptive systems are manifold. Possible application domains are e-learning systems, hypermedia, mobile services etc., each introducing a research field with conferences and workshops on its own. Obviously, these domains share one certain aspect: they are interacting with a user or groups of users. If we treat users of a domain as one special kind of object within this domain, we can list the components a domain is composed of: domain objects, attributes describing these objects, relations between objects and messages interchanged among objects. Every adaptive system needs to map parts of these components to an expressive internal model, in order to interpret and process information about the domain it is applied for.
- User Model: The user model stores the knowledge that is currently available about a user. On the one hand, this

model will contain parts, such as the name and the age of the user, which can also be reused within numberless other domains. Other attributes like the user's interests in arts are strongly connected with one specific domain and thus will not apply in all domains. As a result, the user model has to reflect a distinction between domain-dependent and domain-independent information. Depending on the application field the user model may cover more or less relevant information about the user like preferences, habits, interests, knowledge, competences, restrictions or personality characteristics. Besides this distinction, the information covered by a user model is classified by the frequency of its alternation. Fix or very rarely changed attributes (e.g. name, age, but also personality attributes like intro-/extroverted) are called static attributes. On the other hand, there exist dynamic attributes that change often or even in regular short intervals (e.g. position, head orientation).

- Task Model: Task models as well as dialog models play a central role in the model-based development of user interfaces. Task models capture and reflect knowledge about user tasks concerning the system. Through abstraction of the tackled task of the user such a model normally describes the current purpose of application of the system. Dialog models represent abstracted descriptions of the dialog flows, which are composed in appropriate specification languages. The task model is built upon the results of the task analysis and describes the functionality of the given overall system in a structured and in many cases modular manner. In addition, it may give details about the order of workflows, the purpose of the system and the tasks of the human. Formally, a task model consists of a goal, a subject of work, one or more tools and supplies for work, one or more user roles, and subtasks with temporal relations. Modeling tasks constitutes one phase within the software development process before the system design. Possible methods like observations, interviews and other empiric procedures are applied that investigate a possible respectively planed embedding of a context-aware system.
- Environment Model: The environment model describes the outside view of the system. The environment is something surrounding a user or an entity and giving a meaning to this object. This model contributes to the user model and stores information about the current situation the user acts in together with the preceding situations in the activity process. Its main purpose is to define the scope of the system, which should neither be too large (influences time or money), nor too small (the system is not doing what the user wants or needs). Information about the environment is shared knowledge that is explored and exploited by participants in the interaction. Shared knowledge includes the domain, the users, and the interaction with the system (transaction history). One distinction has to be drawn between static and dynamic aspects of an environment, respectively between knowledge that remains constant throughout the interaction and knowledge that changes throughout the interaction. Presently the most important parts are space and time with regard to the user or an entity. Based on this information, recommendations may be generated that comply with the user's interests as well as with the spatial and temporal reachability (e.g. location based services). Other types of information about the environment are social (e.g., friends, enemies, neighbours) and technical properties.

2.3 Effector: Implementing Adaptive Behaviour

The effector function comprises the realization of the adaptation activity that has been decided in the inference step. Possibly, a dialogue for clarification about the upcoming adaptation has to be held with the user in advance. Effective modifications of the system require knowledge about the realisation of the selected adaptation method or of the entire adaptation process. The methodology of the way how adaptation is performed constitutes an important aspect of the adaptation process. Not only has this methodology an effect on the content of the potential adaptation, but also the necessary skills of the user are determined that are required for the execution of the adaptation in the case of manual adaptation and for the intelligibility of the adaptation in the case of automatic adaptation. The tighter the methodology of the adaptation is coupled with the actual usage of the system the easier the methodology can be applied and learnt. The chosen rendering of the adaptation immediately affects the methodology. Therefore, the implementation of adaptive behaviour needs to consider the selection of the appropriate adaptation target and the associated adaptation method. The following sections describe these two aspects in more detail.

Adaptation Targets

The algorithms rendering the adaptation method contain actuation knowledge about targets available for adaptation. An adaptation of these system parts may be immediately visible or noticeable to the user or the adaptation effects may be hidden from the user and display their impact at a later point in time. Adaptive applications consider five properties or parts of the system, i.e. adaptation targets that can be tailored to the context and the user:

- Human-Computer Interaction, i.e. the modality needed to enter commands or data, and receive information and services.
- Information Presentation, i.e. the methods and coding required for receiving and displaying content (front-end).
- Functionality, i.e. the features needed to perform tasks (back-end) and the ability of the application to solve a single task or a set of tasks.
- Information and Service Selection, i.e. the information content, density and depth as well as the functionality and complexity of necessary services.
- Knowledge Base, i.e. the collection, organisation, and retrieval of the knowledge about and the model of the user, the context and the application.

In a ubiquitous computing system, traditional modalities for data input (e.g. a keyboard) are expanded by other information acquisition methods such as sensors. The same holds for traditional

information presentation displays (e.g. a monitor), which can be extended by every possible actuator that may have influence on the environment like motors or Light Emitting Diodes (LEDs). For example, a warning for the user may be rendered as a "Warning" writing on a monitor, a flashing red light or an alerting noise coming through the loudspeaker.

Adaptation Methods

Independently from the adaptation targets mentioned above an array of basic methods of adaptation can be specified. The rendering and execution of an appropriate adaptation method is part of the actuation knowledge. Additionally, these adaptation methods are independent from the way an adaptation is performed, i.e. manually or automatically. The basic adaptation methods are (with an increasing complexity):

- Null Adaptation, i.e. the simplest way of adaptation if there is no adaptation necessary. Null adaptation can also mean that an adaptive system leaves the adaptation entirely to the user.
- Parameterisational Adaptation, i.e. changes that affect the internals of elements or components of the adaptation target. This adaptation method changes accessible values of these components and alters parameters in order to reconfigure their behaviour.
- Transformational Adaptation, i.e. transforming the old structure or composition of the adaptation target into a new one. This kind of adaptation supports the reorganisation of parts of the adaptation target and permits modification, addition and removal of these elements under certain conditions.
- Generative Adaptation, i.e. the generative from-scratch (re-)programming or change of the functionality of an element of the adaptation target. Such generators need to be tightly integrated with the adaptive system and might perform a generation automatically or in correspondence with the user. In practice, a pure automatic generative approach is mostly insufficient because of the computational complexity of the generation process or because of the insufficient quality of the results it produces.

3. Actors in the Process of Adaptation

Several actors can be identified, who are part of the realisation of the adaptation process. This identification of actors is essential to better understand the creation, usage and management of adaptive systems, and also to provide an accordant tool suite that supports each actor in its specific role in the software engineering process. During the entire software engineering process of adaptive systems the same human actor may assume different roles.

The five main actors in the adaptation process of an adaptive system are: the application, the user, the developer, the domain expert and the author. This distinction can be more fine-grained, but additional actors can be complemented by activities dedicated to the five basic groups of actors. In the following, these five types of actors are described in more detail.

The two obvious actors playing the main part in the adaptation process are the context-aware application for automatic adaptation methods and the user for manual adaptation methods. Traditionally, the adaptive system initiates and executes the adaptation process. The user needs to be actively involved in the adaptation process, and thus, the user participates even in the application of automatic adaptation methods. Dieterich et al. (1993) show the interplay between the user and the application in the different phases of the adaptation process.

Prior to the delivery and operational use of an adaptive system, developers implement such a complex system following a structured software engineering process. After the requirement engineering and planning phase this third group of actors completes the system comprising software and hardware step by step. The developers integrate the targeted adaptive and adaptable methods into the application in order to broaden the usage of the system. Additionally, the developers implement mechanisms that allow the user to accept or reject automatic adaptations proposed by the application or have influence on an automatic adaptation process. Developers play an important role as an actor at adaptive systems because they decide on and design the algorithms and conditions for adaptation activities.

The developers should build systems that are usable and suit the needs of many users. Certain dynamics may limit the fit of the application characteristics to the user requirements after system rollout. Therefore, tailoring of the system or adding new features might become necessary after the development phase during the operational use of the system. Since the time between the emergence of new user needs and a professional development might be too short, domain experts perform the required adjustments and configurations to the system. Such an expert knows more about the adaptation possibilities than the user and has the better ability to execute them. As an interface or adapter between the system (or the developer) and the user, the domain expert (power user) might propose adaptation alternatives, which are neither automatically uncovered by the system nor investigated by the user. The expert's knowledge about the essential methods and tools links to approaches like end-user development (Fischer, 2002; Lieberman et al., 2006) addressing the configuration and the further development of applications.

A domain expert can accompany the adaptation process from the initial phase on and cooperate with the users, which all contributes to a better fit of the adaptive system with the requirements. Depending on the usage context of the application, authors administrate the content and services adaptive systems provide. In their role as domain experts, authors mange the information and service repository of context-aware applications. They care for the addition of new information and the removal of outdated information, and keep the provided content and services up-to-date. The author creates or composes appearance of the context-aware application without actually knowing the entire internals of the application. Thus, authorship bases on skills regarding creativity rather than programming. Furthermore, the author conducts test runs of the application without any user to check whether or not the application presents the desired content and services correctly.

4. Conclusion

In this paper we have presented a review of approaches to adaptive and context-aware systems. For such systems we have proposed a model of three functions: a sensory function, an inference function and an effector function. Context-aware systems we have understood as an extension of adaptive systems including beyond user and task characteristics also the time and space as well as the environment and technological infrastructure of the usage. Situation-awareness was proposed for an episode in the history of context-awareness where the momentary values of context variables are evaluated for adaptations not necessarily reflecting the long-term history of the user.

Special attention shall be given to the shared control of the adaptivity between the user and the system. Monitoring the system's adaptive behaviour allows the user to be in control of the adaptivity and use adaptive features to augment his or her capabilities for more effective and more efficient task performance. This shared control approach between user and system about system adaptations shows the complementary of adaptive and adaptable systems. The user decides about adaptations while the system proposes and performs adaptations. Adaptive systems focus on the system's contribution for the adaptation like proposing, negotiating and performing adaptations. Adaptable systems focus on the user's contribution like performing the adaptation autonomously or negotiating adaptation with the adaptive system and decide about the final result. The adaptive aspect of system's adaptations was the focus in this chapter. The relationship between adaptive and adaptable features establishes the contribution of adaptivity for the autonomy of the user: the user finds initiative and support for adaptations but remains in control about the process and results.

Finally several roles were discussed in the process of adaptations increasing the effectiveness and effectivity of the adaptations without overstraining the user. The roles reflect the phases of the lifecycle of the system and the different competences of the human beings like developer, users and consultancies as well as the contribution of an adaptive system in reducing the effort for the end-user and increasing the performance of the user-system interaction.

Danksagung

Wir danken Andreas Lorenz für seine Mitarbeit an einer früheren Fassung dieses Aufsatzes.

Literature

Aamodt, A. and Plaza, E., 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. Artificial Intelligence Communications, 7(1): 39–52.

Bäck, T., 1996. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, USA, 328 pp.

Bäck, T., Fogel, D. and Michalewicz, Z., 1997. Handbook of Evolutionary Computation. Oxford University Press, 988 pp.

Booch, G., Rumbough, J. and Jacobson, I., 1999. The Unified Software Development Process. Object Technology, Massachusetts, 463 pp.

Culler, D.E. and Mulder, H., 2004. Smart Sensors to Network the World. Scientific American, 290(52–59).

Dieterich, H., Malinowski, U., Kühme, T. and Schneider-Hufschmidt, M., 1993. State of the Art in Adaptive User Interfaces. In: M. Schneider-Hufschmidt, T. Kühme and U. Malinowski (Editors), Adaptive User Interfaces. North-Holland, Amsterdam, Netherlands, pp. 13–48.

Duda, R.O. and Hart, P.E., 1973. Pattern Classification and Scene Analysis. John Wiley & Sons, New York, USA.

Fischer, G., 2002. Beyond ‚Couch Potatoes': From Consumers to Designers and Active Contributors. First Monday (Peer-Reviewed Journal on the Internet), 7(12).

Gilb, T., 1998. Principles of Software Engineering Management, 464 pp.

Hertz, J., Krough, A. and Palmer, R.G., 1991. Introduction to the Theory of Neural Computation. Perseus Books Group, Redwood City, CA.

Jameson, A., 2003. Systems that adapt to their users: An integrative Overview, Tutorial presented at 9th International Conference on User Modelling, Johnstown, PA, USA.

Lieberman, H., Paternó, F. and Wulf, V., 2006. End User Development. Springer, Berlin, Germany, 492 pp.

Oppermann, R., 2005. From User-Adaptive to Context-Adaptive Information Systems. i-com Zeitschrift für interaktive und kooperative Medien, 4(3): 4–14.

Satyanarayanan, M., 2003. Of Smart Dust and Brilliant Rocks. Pervasive Computing, 2(2–4).

Watson, I., 1998. Applying Case-Based Reasoning: Techniques for Enterprise Systems. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 289 pp.

Zadeh, L.A., 1965. Fuzzy Sets. Information and Control, 8(3): 338–353.

Zimmermann, A., Lorenz, A. and Specht, M., 2003b. User Modeling in Adaptive Audio-Augmented Museum Environments. In: P. Brusilowsky, A. Corbett and F. de Rosis (Editors), Proceedings of the 9th international Conference on User Modeling (UM-03). Lecture Notes in Computer Science. Springer Verlag Berlin Heidelberg, Johnstown, PA, USA, pp. 403–407

1

2

1 Reinhard Oppermann, Diplom Psychologe und Dr. phil., seit 1993 Honorarprofessor an der Universität Koblenz-Landau. Seit 1979 wissenschaftlicher Angestellter in der Gesellschaft für Mathematik und Datenverarbeitung (GMD) – heute Fraunhofer-Gesellschaft, jetzt Forschungsbereichsleiter für kontextualisierte Informationsdienste. Forschungsthemen sind Benutzerbeteiligung bei Systementwicklungen, Auswirkungen von Informationstechnik auf das Verhältnis Bürger – Staat, Mensch-Maschine-Kommunikation, Softwareergonomische Evaluationsmethoden, adaptive und adaptierbare Benutzerschnittstellen, Kontextanalyse von mobilen Informationssystemen, E-Learning im Arbeitskontext.

2 Andreas Zimmermann arbeitet als leitender Wissenschaftler und Projektmanager im Forschungsbereich "Information in Context" am Fraunhofer Institut für Angewandte Informationstechnik FIT in St. Augustin. Im Rahmen der Europäischen Forschungsprojekte, die er leitet, ist er für die benutzer-zentrierte Software-Entwicklung und das Design von Software-Architekturen verantwortlich. 2007 erhielt der Diplom-Informatiker für seine Promotion im Bereich kontext-sensitiver Systeme einen Doktortitel. Vor seiner Anstellung am Fraunhofer Institut war er als Product Innovation Manager bei TRAIAN Internet Products in Bonn und als Berater für die T-Systems in Aachen tätig.