ISBN: 978-972-8939-68-7 © 2012 IADIS

CONCEPTUAL ASPECTS OF USER-ORIENTED ADAPTIVE SYSTEMS

Inese Supulniece
Institute of Information Technology, Riga Technical University, Kalku 1, Riga, Latvia, LV-1658

ABSTRACT

Adaptive systems are a common and trendy concept within the field of Computer Science. Scientific literature about this topic includes a wide spectrum of systems from different computer science domains, starting from automation till HCI. When someone reviews the scientific literature with keywords, e.g. adaptation, adaptive system, there is a huge amount of researches and it is difficult to link together similar researches. The contribution of this paper lies in (1) defining four types of adaptive systems based on system and environment interaction view, which can be applied to any adaptive system; (2) proposing environment-system-user interaction view - the list of core concepts of adaptive system and extended list with user-oriented aspects; (3) describing possible adaptation results and formalize the main target of adaptation. The environment-system-user interaction view can serve as a basis for classification of adaptive systems and building a clear understanding about different types of adaptive systems and their essential (not all) differences. The proposed adaptive system types and core concepts are illustrated with examples.

KEYWORDS

Adaptation, adaptive systems, user-oriented

1. INTRODUCTION

Adaptive systems are at the present a common and trendy concept within the field of Computer Science, especially in the subfield of Information Systems. One difficulty faced by adaptation researchers is that relevant work is spread over many fields and communities and some of this work does not describe itself as "adaptation" (Kell, 2008). Literature about computer science includes a wide spectrum of systems that are described as adaptive in order to solve different problems, starting from automation till human-computer interaction. There is a lack of consensus among researchers and practitioners on the points of variation among adaptive systems. In fact, often the models of adaptation are represented implicitly in the form of domain knowledge or the engineer's expertise in the development of these systems. This in turn makes it rather infeasible to systematically compare the different approaches (Andersson, et al., 2009).

The most frequently used terms in definitions of adaptation and adaptive systems are "system" and "environment" and adaptation occurs during the interaction between system and environment. Thereby initially we will describe interaction of system and environment.

In the context of adaptive systems there is a lack of structured approaches, how to describe and classify them. Most of classification methods do not cover all adaptive systems, because often are created for a narrow particular research sub-area. In this paper we attempt to identify key concepts of adaptive systems to describe any type of adaptive system. It can serve as a basis for classification of adaptive systems and building a clear understanding about different types of adaptive systems and their essential (not all) differences. At the same time it should be simple enough to characterize and understand the system without going into all details.

Despite theoretically expected utility from adaptive systems, real success of adaptive systems is not unambiguous. If we understand adaptation as purposeful process, then there exists someone, who defines the goals, which are realized by adaptation process. Therefore we propose to include user-oriented aspects into the view.

The contribution of this paper lies in (1) defining four types of adaptive systems based on the system and environment interaction view, which can be applied for any adaptive system; (2) proposing the environment-

system-user interaction view - the list of core concepts of adaptive system and extended list with user-oriented aspects; (3) describing possible adaptation results and formalization of the main target of adaptation.

The rest of paper is structured as follows – Section 2 briefly describes terms *adaptive systems* and *adaptation* in computer science and some classification methods. Section 3 presents system and environment interaction and the key concepts of adaptive systems to understand and describe any kind of adaptive systems. The set of core concepts is extended with user-oriented aspects in Section 4. The proposed concepts are illustrated in practice using three scientific researches in Section 5. The paper concludes with Section 6, where research results and further research is discussed.

2. ADAPTIVE SYSTEMS

The term adaptation is widely used in biology and sociology. Since sixties this term started to appear in scientific literature about mathematics, technology and cybernetics (Rastrigin, 1981). The next section presents an overview of adaptive systems in computer science and some classification approaches.

2.1 Adaptation in Computer Science

In general, adaptation can be interpreted as 'making suitable by modifying the current state', but this is only one of the innumerable explanations around this concept (Garcia-Barrios, 2007). Specific interpretations of the terms related to adaptation, can be identified within particular contexts or disciplines.

Adaptive systems are directly related to the adaptation process. Tsypkin (1971) and DeJong (1980) defines adaptation as a process of changing system's parameters and structure, and maybe also managed impact based on current information with the aim to reach determined (often it is optimal) system's state under uncertain and changing working conditions. Sometimes adaptation is associated with learning, because learning is used for collecting information about system's state and characteristics, which is required for effective management in uncertainty surrounding. However, the most important feature of adaptation consists of the aim to optimize selected quality measures (Tsypkin, 1971) while accumulating current information and applying it for uncertainty prevention because of lacking a priori solution.

Adaptation typically includes two approaches (Rastrigin, 1981): 1) adapting to a particular environment (passive adaptation) or 2) searching environment, which is appropriate for particular system (active adaptation). In the first situation, an adaptive system realizes its functions the most effectively in the particular environment. In opposite, active adaptation refers to environment changes or searching for other environment, where functioning criteria might be maximized. Obviously, in reality both types of adaptation are merged. To realize adaptation, one needs to understand the goal of adaptation (what makes system operation effective?) and the algorithm to reach this goal. Adaptation is arrangement of purposeful activities on the object, which results in successful reaching the set goals, but adaptation task originates in situations, when some information needed for object optimization is missing.

Werbos (1987) describes adaptive systems as tools which develop new information about how to do the task better by analyzing past experience and relating it to performance criteria set by humans.

Jameson states - adaptive system adapts its behavior to individual users based on information about them (information is implicitly collected during the user-system interaction or users are explicitly asked for it), and the adaptive system performs the adaptation using some form of learning, inference or decision making (Jameson, 2003).

Chronological roadmap of adaptive systems within the context of computer science is given in (Garcia-Barrios, 2007). It shows that mainly adaptation is referred to the notion of changing something to meet some specific requirements or purposes.

2.2 Classification of Adaptive Systems

McKinley (2004) differentiates between parameter and compositional adaptation. Parameter adaptation modifies software system's variables that determine behavior. The Internet's Transmission control Protocol is an often-cited example. But parameter adaptation has an inherent weakness – it does not allow new algorithms and components to be added to an application after the original design and construction. It can

tune parameters, but can not adopt new strategies. By contrast, compositional adaptation exchanges algorithmic or structural system components with others to improve a program's fit to its current environment.

Trapp and Schurmann (2003) use two questions to come to a classification: what is the basic objective of the adaptation and which type of adaptation is used. Regarding the objective of adaptation, they consider classes: (1) function-based adaptation – the capability of a system to adapt its functionality dynamically to the changing conditions of its environment; (2) fault based adaptation – capability of a system to adapt its behavior in the case of faults to achieve graceful degradation. Regarding the different types of adaptation, they consider classes: (1) adaptive node mapping – capability of distributed systems to relocate certain software modules to other hardware nodes at runtime; (2) adaptive collaboration – achieved if the modules of a system can be reconnected at runtime; (3) adaptive functionality – capability of a system to adapts its actual functionality, whereby everything from adapting certain parameters to a complete exchange of the algorithm is possible.

Lachner et al. (2007) states three catalysts for adaptation: inter-individual, intra-individual and environmental differences. Inter-individual differences address varieties among several users, e.g. user preferences (towards language, color, menu options, security), interests and psychological personality characteristics. Intra-individual differences consider the evolution and further development of a single user, as well as the task over time, e.g. user's expertise increase. Environmental differences result from the mobility of computer devices, applications and people.

Awang et al. (2009) classifies four prominent approaches to software adaptation in Architecture-based, Component-based, Middleware-based and Agent-based. Architecture-based approach makes use of architecture model to achieve adaptation at run time. Component-based approach views application system as a "composition of components", where adaptation is achieved with dunalic modification of component composition to create variants. At any one time, the variant that best suited user requirements and operating conditions will be selected. Middleware-based approach attempts to implement software adaptation by segregating the development of application systems into different layers. In Agent-based approach software agent is used to enable software adaptation by acting as adaptation mediator in providing externalized adaptation mechanism.

The classification approaches mentioned are too general to understand the main differences between adaptive systems and the form of adaptation. Some of them are context specific and can not be applicable for adaptive systems from other disciplines of computer science.

Andersson et al. (2009) proposes modeling dimensions of adaptive systems, that describe various facets of self-adaptation and allows better to understand the scope of adaptive systems. These dimensions are classified in terms of four groups. First group (goals) includes dimensions associated with self-adaptability aspects of the requirements; second (change), the dimensions associated with causes of self-adaptation; third (mechanisms); the dimensions associated with the mechanisms to achieve self-adaptability, and fourth (effects), the dimensions related to the effects of self-adaptability upon a system. This classification approach seems to be applicable for any type of adaptive system as it strives to be a conceptual model for adaptive systems. This approach is better suited for describing behavior and working details of adaptive systems, e.g. frequency dimension (how often a particular change occurs), duration dimension (how long the adaptation lasts), triggering dimension (whether the change that triggers adaptation is associated with an event or a time slot), rather than for identifying key conceptual differences among classes of adaptive systems. Thus for this purpose our approach is more general and unsophisticated.

3. SYSTEM-ENVIRONMENT INTERACTION VIEW OF ADAPTIVE SOFTWARE SYSTEM

The most frequently used terms in definitions of adaptation and adaptive systems are "system" and "environment" and adaptation occurs during the interaction between system and environment. Thereby initially we will describe interaction of system and environment. Later system-environment view will be extended with user-oriented concepts to reflect purposefulness of adaptation process.

Let assume that in environment E exists a software system S. Then E can be characterized by several states E=<e_1, e_2, …e_n> in determined time moments and S can be characterized by states S=<s_1, s_2, …, s_n> as in figure 1.

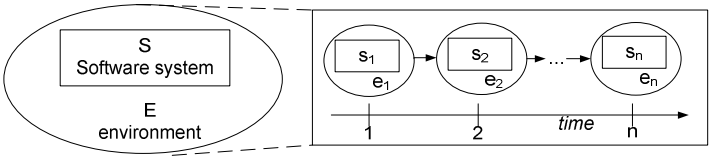

Figure 1. Software system and environment in determined time moments

Five alternatives for adaptation can be distinguished for described situation – see summary in Table 1.

Table 1. Formalization for system-environment interaction view of adaptation

Object whose changes cause adaptation	Object to be adapted	Formalization
Environment E	Software system S	$s_{n+1}=f(e_n, s_n)$
Software system S	Environment E	$e_{n+1}=F(e_n, s_n)$
Environment E, Software system S	Environment E, Software system S	$s_{n+1}=f(e_n, s_n)$; $e_{n+1}=F(e_n, s_n)$
Software system S	Software system S	$s_{n+1}=f(s_n)$

System is adapted to changing environment

Environment E is changing and software system S is adapting its behavior according to environment changes. Thus, S state depends on E state, which can be formalized as follows: $s_{n+1}=f(e_n, s_n)$ where s_{n+1} is S state after adaptation, e_n and s_n is E and S state before adaptation and f is algorithm for adaptation.

For example, cooperative driving systems, which adapt their behavior according to the surrounding vehicles and road conditions (Fritsch, et al., 2008).

Environment is adapted to changing system

Software system S is changing and environment E is adapting to S changes. Thus, E state depends on S state, which can be formalized as follows: $e_{n+1}=F(e_n, s_n)$, where e_{n+1} is E state after adaptation, e_n and s_n is E and S state before adaptation, F is algorithm for adaptation.

For instance, during implementation of packaged enterprise applications, there is an option to change either enterprise business processes or software itself. Under this option goes situations, when these processes (as part of environment) are adapted to e.g., to Enterprise Resource Planning systems. Unfortunately similar situations in computer science are not referred to adaptation process.

System and Environment is changing and adapting to each other

Both S and E is changing and adapting to each other. Thus, S state depends on E state and E state depends on S state, which can be formalized as follows: $s_{n+1}=f(e_n, s_n)$ and $e_{n+1}=F(e_n, s_n)$, where s_{n+1} and e_{n+1} is S and E state after adaptation, e_n and s_n is E and S state before adaptation, f and F are algorithms for adaptation.

User-adaptive software (Jameson, 2003) belongs to this type, because user (as part of environment) is changing over time and also adapting to system. At the same time also software system is adapting to the user.

System is changing and adapting to itself

Software system S itself is changed and is adapting to itself. Thus, S state depends on previous S state and can be formalized as follows: $s_{n+1}=f(s_n)$.

A software system consists of multiple components including software components and hardware, thus an example of this type is compositional adaptation (Mckinley, 2004). It enables software to modify its structure and behavior dynamically in response to changes in its execution environment.

Environment is changing and adapting to itself

Environment changes that cause environment changes are not the scope of computer science research (and this paper).

Adaptation mainly refers to the notion of changing something to meet some specific requirements or purposes, so there exists something, which is adapted (Garcia-Barrios, 2007). System to be adaptive must meet some dynamic requirements or purposes, thus there exists something, which is changing. Consequently, when someone refers to adaptation in computer science, it is important to understand, where the adaptation

takes place and what changes are causing the adaptation. To summarize mentioned situations and examples, a simplified input/output view of the core concepts of the adaptive software system is presented in Figure 2.

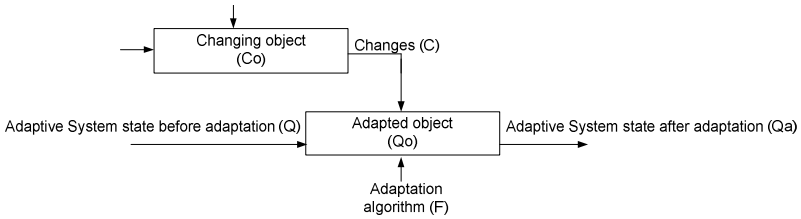

Figure 2. Simplified input/output view for core concepts of adaptive software system.

The adaptive software system state after adaptation (Q_a) depends on the adaptive system state before adaptation (Q), changes (C) and adaptation algorithm (F) or $Q_a=F(C, Q)$.

Consequently main core concepts of adaptive systems are:

* *Co* (changing object) - concept whose changes cause adaptation or changing object;
* *Qo* (adapted object) - concept to be adapted;
* *F* (adaptation algorithm).

4. ROLE OF THE USER IN ADAPTIVE SYSTEMS

Despite theoretically expected utility from adaptive systems, real success of adaptive systems is not unambiguous. Therefore, we propose to include user-oriented aspects into the view. In next sub-sections we will analyze two types of users (stakeholder and end-user) and their relevance to adaptive systems. Stakeholder and End-user views potential result of adaptivity in different abstraction level (Goals and Expectations), which is explored in sub-section 4.2. Findings are used to extend the list of core concepts with user-oriented aspects, describe possible adaptation results and formalize the main target of adaptation.

4.1 Stakeholder and End-user Concept

If we understand adaptation as a purposeful process, then there exists someone, who defines the goals, which are realized by adaptation process (Rastrigin, 1981). Thus we can add *stakeholder* (*H*) to our view – see figure 3. Stakeholder is actor which benefits from adaptive system. We can assume that stakeholder always formulate the goal set (*G*) towards adapted object.

Another important actor in adaptive system is *end-user* (*U*), who has expectations in his mind, what should be state of system after adaptation process. End-user has receptors and sensors which help him to perceive environment and software system. End-user always formulates the expectation set (*X*) towards adaptation result. If software system state corresponds to end-user expectations, then adaptation is not needed. If system's state by some reason does not correspond to end-user expectations, then adaptation is necessary.

End-user uses system to reach some goal thus all end-users are also stakeholders, but not all stakeholders are end-users.

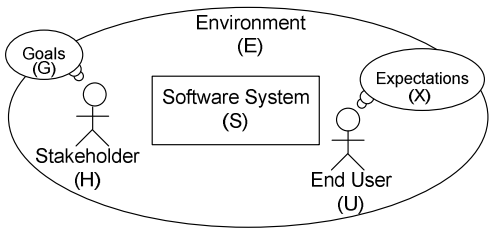

Figure 3. Stakeholder and End-user concepts in adaptive software systems

Consequently the extended list of core concepts from Section 3 is:

* *Co* is changing object - concept whose changes cause adaptation or changing object;
* *Qo* is adapted object - concept to be adapted;
* *U* is end-user;
* *H* is stakeholder;
* *G* is set of goals towards adapted object, which are set by stakeholder;
* *X* is set of expectations towards adapted object, which are set by end-user;
* *F* is adaptation algorithm.

There can be situations, when stakeholder, end-user and changing object is the same. For example, user adaptive system (Jameson, 2003), which adapts to individual user. User is Stakeholder who is setting goals towards the system; he is also end user of the system. There are many users and each user can have different characteristics and one user can have changing characteristics over the time, thus at the same time user is also changing concept, which cause system adaptation.

4.2 Goals and Expectations

Expectations (*X*) differ per each individual end-user *U*, because they are based on mental model of each individual end-user. These expectations also are different in separate time moments (e.g. depend on user's mood or particular situations) even if subject is the same. Most probably it is not realistic to capture fully these individual expectations in any software system.

But software system can capture Goals (*G*) - objectives the system under consideration should achieve. These are on higher generalization level than expectations as they are common for all individual end-users and also for stakeholders. User expectations (*X*) are linked to Goals, thus $X(U)=y(G)$.

According to Figure 4, adapted object state after adaptation or adaptation result is $Q_a=F(C, Q, G, X_c(U))$, where Q_a is adapted object state after adaptation, *F* is adaptation algorithm, *Q* is adapted object state before adaptation, *G* is set of goals towards adaptation and $X_c(U)$ is set of calculated expectations (see for explanation at the end of this section) of user *U* towards adaptation result.

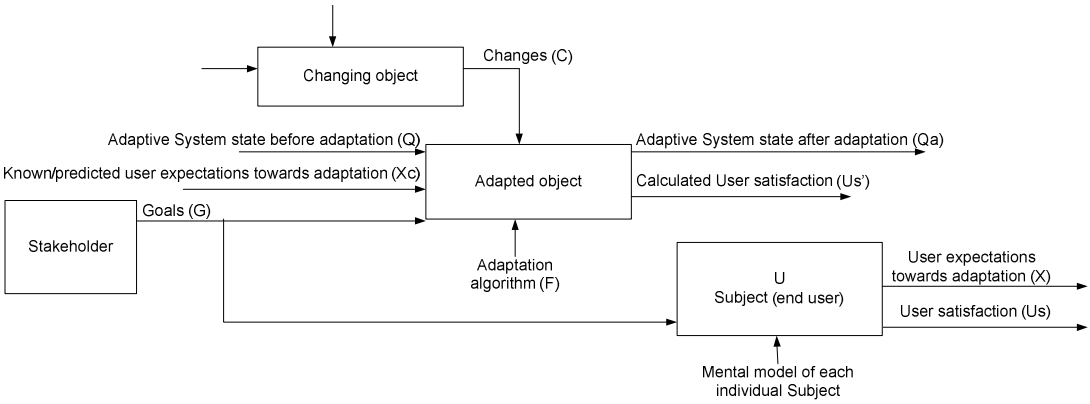

Figure 4. Extended input/output view for core concepts of adaptive software system

Theoretically expected result or adaptive system state after adaptation should be equal to expectations set by the end-user or $Q_a=X(U)$. But practically it is hardly achievable. The user satisfaction index (*Us*) shows proportion of user expectations, which are fulfilled as result of adaptation:

$$Us(U) = \frac{X(U) - Qa}{X(U)},$$

where $X(U)$ is set of expectations of user *U* and Q_a is adapted object state after adaptation.

1) If $Us(U)=0$, then user expectations are equal to system state after adaptation (this is ideal adaptation), where user is completely satisfied with adaptation result.
2) If $Us(U)>0$, then all user expectations are not met and user is not satisfied;
3) If $Us(U)<0$, then system had done more than user had been expected and user is not satisfied.

Thus, the main goal of adaptation would be to minimize $\Delta|Q_a\text{-}X(U)|$.

Even if it is not realistic to capture fully individual expectations, the software system might know or predict some user expectations towards adaptation (X_c), e.g. user preferences, which are listed on software system. Known/predicted user expectations (X_c) should be subset of real user expectations towards adaptation, thus $X(U)=X_c(U)+X_p(U)$, where $X_p(U)$ are personal expectations, which software system does not capture.

Coming back to user satisfaction index $Us(U)$ we can see that

$$Us(U) = \frac{Xc(U) + Xp(U) - Qa}{Xc(U) + Xp(U)}.$$

As $X_p(U)$ are personal expectations that are not captured by software system, we can exclude these expectations and get user satisfaction level, which can be calculated by adaptive system $Us'(U)$.

$$Us'(U) \frac{Xc(U) - Qa}{Xc(U)}.$$

And then main goal of adaptive system would be to minimize $\Delta|Q_a\text{-}X_c(U)|$.

This is exactly what is done by adaptive software systems in practice, but it it vital to remind that X_c is not full set of user expectations, thus even if mentioned function is minimised, user still might be not satisfied with adaptation result and planned benefits might be not reached.

When user expectations are not fulfilled by adaptation and $Q_a \neq X(U)$, the end user/stakeholder needs to make a decision:

1) to keep the adaptation result and continue the work with loss caused by adaptation or potential benefit that was not reached;

2) to the optimize adaptation algorithm to minimize $\Delta|Q_a\text{-}X_c(U)|$ with available resources (adaptation must be worth of spending these resources);

3) to change the set of expectations/goals.

5. APPLICATIONS

In this section the system-environment-user interaction view is used to describe several adaptive systems investigated in literature about adaptive software systems from different computer science domains. That allows for better understanding of essential aspects and differences between described researches. The following cases about adaptation and adaptive systems are used as examples:

1) Cooperative driving system (Fritsch, 2008) is adapting its behavior according to the surrounding vehicles and road conditions.

2) Learner model (Nguyen, Do, 2008) is intended to apply to Adaptive Education Hypermedia System (adaptive learning system supporting personalized learning environment).

3) Adaptive distributed middleware for data replication (Milan-Franco, et al., 2004) is able to adjust to changes in the amount of load submitted to the different replicas and to the type of workload submitted.

The system-environment interaction view of applications is listed in Table 2. Only three types are included, because two situations (where environment is changing and adapting to itself and where system is changing and environment is adapting) are not denoted with adaptive systems keyword in computer science.

Table 2. The system-environment interaction view of applications

Cooperative driving system [4]	Learner model in adaptive learning [12]	Adaptive distributed middleware for data replication [11]
System is adapted to changing environment.	System and environment is changing and adapting to each other.	System is changing and adapting to itself.

Table 3 summarizes core concepts in each of adaption cases considered. There might be several changing objects, adapted objects and end-users for papers describing high level idea or group of the systems, methods or technology, e.g. cooperative driving system. For a lower generalization level, description of adaptation (e.g. learner model in adaptive learning, adapted object, changing object and end-users) are denoted more specifically. The goal is noted in all papers, only the goal type (business, operational or technical level) varies. The biggest challenge is to identify the set of calculated user expectations, because most of researches do not provide this information.

Table 3. Core concepts of applications

Core concepts	Cooperative driving system [4]	Learner model in adaptive learning [12]	Adaptive distributed middleware for data replication [11]
Changing Object (Co)	Environment (e.g., another vehicle, infrastructure)	User (student) and his/her characteristics. Different students have different goals and student can change his goals over the time (inter- and intra-individual differences)	The database load (and the achieved throughput) and the state of individual components (monitoring for crashes)
Adapted object (Qo)	Cooperative driving system	Adaptive Education Hypermedia System (content/presentation and navigation (links))	System configuration of database replication at the middleware level (multiprogramming level and the distribution of transactions across replicas)
End-user (U)	Driver	Individual student	End-user of application, which is connected to particular database
Stakeholder (H)	Driver, drivers of other vehicles, other participants of road traffic	Teacher, individual student	End-user of application, end-user of database (e.g., administrator of data base)
Goals (G)	Adapt system's behavior according to the surrounding vehicles and road conditions (e.g., adaptive cruise control has to maintain a safe time-headway distance between vehicles)	Learning material should be adapted and delivered to every individual student	Adjust distributed middleware to changes in the amount of load submitted to the different replicas and to the type of workload submitted. At the local level, the focus is on maximizing the performance of each individual replica. At the global level, the system tries to maximize the performance of the system as a whole.
Calculated User Expectations (Xc(U))	In this article are not presented personalized or individualized driver expectations, the adaptation rules are common for all drivers	Learner (user) model – personal knowledge model and domain independent information (goals, interests, background, experience, individual traits, aptitudes and demographic information)	Max response time of queries to database

6. CONCLUSION

Four types of adaptive systems based on the system and environment interaction view are defined and core concepts (environment-system-user interaction view) of adaptive systems are proposed in this paper. The system and interaction view can be used to describe any adaptive system in computer science. The environment-system-user interaction view illustrates main differences between researches about adaptive systems. This can help researchers and practitioners from different computer science domains to understand each other, while using keywords 'adaptive systems' and 'adaptation'.

Comparing to Andersson, et al., (2009) dimensions, the common concept/dimension is the set of Goals (objectives the system under consideration should achieve), similar concept/dimension is the set of Change (the cause for adaptation). Our concept Adaptation Algorithm is union of Mechanisms (reaction of the system towards change) and Effects (impact of adaptation upon the system). Additionally we have added user-oriented concepts - two types of actors (stakeholder and end-user) and calculated expectations, which are not included in Andersson modeling dimensions.

The user-oriented concepts are separated, because adaptation is arrangement of purposeful activities on the object (Rastrigin, 1981) and there exists someone, who defines the goals and set expectations towards adaptation result. The difference between user expectations and system state before adaptation is missing

information, where adaptation task originates and where adaptation can be distinguished from optimization task.

Currently, only the main concepts of adaptive systems are identified. To use these concepts for classification of adaptive systems, content of each concept need to be explored and grouped. For example, changing part of adaptive system might be grouped into the layers, where adaptation might happen – environment (e.g., user, other systems, external environment), adaptive system itself (e.g., application, middleware, infrastructure). Thus adaptive systems can be classified by changing object, goal, adapted object, etc.

Our goal towards further research is to expand the proposed model of adaptive system and apply it for building user adaptive enterprise applications.

ACKNOWLEDGEMENT

This work has been supported by the European Social Fund within the project «Support for the implementation of doctoral studies at Riga Technical University».

REFERENCES

1. Andersson, J., et al., 2009, Modeling Dimensions of Self-Adaptive Software Systems. *Software Engineering for Self-Adaptive Systems*, Eds. B. H. C. Cheng, et al. LNCS, *5525*, pp.27-47, Springer Berlin/Heidelberg.
2. Awang, N.H., et al., 2009, Comparative Evaluation of the State-of-the Art on Approaches to Software Adaptation, *4th International Conference on Software Engineering Advances*, pp.425-439.
3. DeJong, K.A., 1980, Adaptive system design: a genetic approach, *IEEE Trans. Syst., Man, and Cyber.*, vol. SMC-10, no. 9, pp. 566-574.
4. Fritsch, S., et al., 2008, Time-bounded adaptation for automotive system software, *ICSE'08*, May 10–18, Leipzig, Germany., pp. 571 - 580
5. García-Barrios, V.M., 2007, *Personalization in Adaptive E-Learning Systems - A Service-Oriented Solution Approach for Multi-Purpose User Modelling Systems*; Dissertation at Institute of Information Systems and Computer Media, Faculty of Computer Science, Graz University of Technology.
6. Jameson, A., 2003, Adaptive Interfaces and Agents. In J. Jacko & A. Sears (Eds.), *Human-computer interaction handbook*, pp. 305–330, Mahwah, NJ: Erlbaum.
7. Jameson, A., 2003, Systems That Adapt to Their Users: An Integrative Overview, *Tutorial presented in: 9th International Conference on User Modeling*, Johnstown, USA.
8. Kell, S., 2008, A Survey of Practical Software Adaptation Techniques, *UCS*, Vol. 14, Nr. 13, pp. 2110-2157.
9. Lachner, J., et al., 2007, Challenges toward User-centric Multimedia, *Proc. 2nd International Workshop on Semantic Media Adaptation and Personalization (SMAP 2007)*, London, UK.
10. Mckinley, P. K., et al., 2004, Composing Adaptive Software. *IEEE Computer* 37(7), pp. 56-64.
11. Milan-Franco, J.M., et al., 2004, Adaptive Middleware for data replication, *In Middleware '04*, Springer-Verlag, pp.175-194.
12. Nguyen, L., Do, P., 2008, Learner Model in Adaptive Learning, *World Academy of Science, Engineering and Technology*, Vol.45, pp.395-400.
13. Rastrigin, L.A., 1981, *Adaptation of complex systems. Methods and applications*, (Adaptatsiya slozhnykh sistem. Metody i prilozheniya), (Russian), Akademiya Nauk Latvijskoj SSR. Institut Ehlektroniki i Vychislitelnoj Tekhniki. Riga: "Zinatne". 376 p. R. 1.60.
14. Trapp, M., and Schurmann, B., 2003, On the Modeling of Adaptive Systems, *In Intl. Workshop on Dependable Embedded Systems*, Italy.
15. Tsypkin, Y.Z., 1971, *Adaptation and Learning in Automatic Systems*, New York: Academic.
16. Werbos, P.J., 1987, Building and understanding adaptive systems: a statistical/numerical approach to factory automation and brain research, *IEEE Transactions on Systems, Man, and Cybernetics*, 17, pp.7-20.