The Adaptation of a Web Information System: a Perspective of Organizations

Aivars Niedritis and Laila Niedrite

Abstract. We provide a different view on the problem of Web Information System (WIS) adaptation, looking from perspective of organizations that are interested in an adapted Web Information System for their needs if a unified system to support similar business processes is used. We propose an adaptation architecture for WIS. Two levels of adaptation are introduced – coarse grained adaptation for the organization level and fine grained adaptation for the user level. The architecture supports also the situation, when users can work with many instances of the system adapted for different organizations, which are integrated into one instance for a particular user.

1 Introduction

World Wide Web is not only an information source but it also has become a platform for application development. Fraternali [1] described a Web application as a hybrid between a hypermedia and an information system. He stated requirements for Web applications [1] and one of them was: the customization and possibly dynamic adaptation of content structure, navigation primitives, and presentation styles.

According to this requirement the data, navigation, and layout should be considered separately if an adaptation is performed. In [2] it is defined that a Web-based Information System (WIS) is adaptive if it is able to modify and personalize delivery of contents and services according to the context of the client. Personalization requires the adaptation of the applications as much as possible according to the preferences of the user and the context of the user [3].

In [3] the context is considered as a set of properties that describe the environment, where the user interacts with a WIS, e.g. in [3] the context is defined by the time, place, device, user, and environment dimensions.

A profile is a representation of an autonomous aspect of the WIS usage context [2]. Another term used in adaptation sphere is the definition of configuration as a specification how information has to be delivered to the user [2].

Many architectures and approaches [2], [3], [4], [5], [6], [7] exist for the WIS adaptation, based on different understandings of context, profile, and configuration. Also different properties of each notion are considered.

The authors of [3] provide a meta model, where a profile and context are defined, and they provide also a set of services for the definition of a personalized access model (PAM). PAM definition allows performing different personalization scenarios, including, preference-based recommendation, context-aware content delivery, personalized access to multiple contents [3].

In [2] an approach to the adaptation of a Web information system according to the context is presented. The authors use a general notion of profile, which is associated with a configuration. The configuration specifies how information has to be delivered according to the requirements of adaptation for the profile.

The paper [6] presents architecture for the personalized presentation layer of WIS to support navigation and different views on the presented data.

At the same time to deliver to the user the appropriate WIS also the approaches to the development of flexible and cost effective systems should be considered.

Software product line approach (SPL) [8] is based on predefined architecture and well known core functions. All reusable components are planned in advance. To get different instances of a particular software in traditional architectures usually the application code is changed. The SPL treats variations in different ways[9]: e.g. inclusion or exclusion of elements, usage of different versions of each component. Different can be e.g. behavior and quality features. For the SPL approach the architecture and configuration management is essential, because each new software product consists of numerous core components and variations.

Software as a service (SaaS) [10] defined as a “software deployed as a hosted service and accessed over the Internet” actually introduces a new way of providing the access to the software. SaaS can be services provided to organizations of different size to support business processes common for these organizations e.g. CRM applications. The organizations subscribe to use the software and pay for the usage. The service vendor hosts the SaaS application on their servers and maintains the software and infrastructure. This is often a cost efficient solution for small and medium size organizations.

SaaS applications can be provided at four different maturity levels, starting from serving a customized application instance per each customer on the SaaS vendor’s server. More advanced way is to use configurable, multi-tenant single instance solution. In this case the customization is performed using the metadata based configuration of SaaS application to adapt the single common application instance to different customer needs. The multi-tenancy means that users from different organizations use the same application instance, but data are distinguished between tenants of the service.

To effectively solve the problems concerning scalability, configuration and multi- tenancy, the SaaS applications need architectures designed for these purposes. These architectures also should resolve problems concerning the multi-tenant customization.

Our architecture is provided for the context of WIS usage, where many similar organizations use the same WIS, but each of them gets an adapted instance of the system. The two level adaptation is provided. Also the users get their own adapted instance of the WIS.

The article is structured as follows: Section 2 presents our proposed adaptation architecture of a Web Information System and describes two levels of adaptation. We provide description of a case study of the usage of proposed architecture in Section 3. In Section 4 we discuss our implementation results and make conclusions.

2 Architecture

We propose the adaptation architecture of a Web Information System (WIS) that supports the usage of WIS in many organizations of the same or connected business areas with similar processes. The architecture adapts the initial configuration of this WIS for each particular organization considering their specific needs.

The adaptation architecture of a Web Information System (Fig.1) consists of the following components:

- Context monitor identifies the context properties of the system usage in the time, when the user connects to the WIS,
- Usage monitor is planned to collect data about the usage patterns of the adapted WIS instances. The goal of this action is to provide information for the monitoring and analysis of processes supported by the WIS and to use these data to develop a strategy for the successful further development of adapted WIS.
- WIS data layer contains profiles that are used in both adaptation levels and the business data. During the coarse grained adaptation process a virtual data store is build along with the organization level WIS instance. The virtual data store contains the data owned by the organization. On the picture (Fig.1) the virtual data stores for different organizations are depicted as “B_SYS1 Business data” etc. WIS data layer also stores the data about the usage of the adapted WIS instances, collected by the usage monitor.
- Adaptation component starts with the identification of the initial configuration of WIS that is defined according to the proposed architecture.

  The adaptation component performs the adaptation in two levels:

- Coarse level adaptation process recognizes the groups of functions that are accessible to the system user according to the usage context, e.g. access point, time, and organization. According to the profiles defined in the system, e.g. configuration profile, this process adapts the WIS initial configuration and establishes an organization level instance of WIS. On the picture (Fig.1) the

function groups are depicted as FG1, FG2 etc. The organization level instances of WIS are depicted as B_SYS1, B_SYS2;

- Detailed level adaptation is based on the information stored in the user profile. The detailed adaptation consists of two steps:
  - Detailed adaptation of the navigation uses the profile information about the functions accessible to the particular user and within the framework of the organization level instance of WIS, provides an adapted navigation between all user functions.
  - Detailed adaptation of the content constructs an instance of WIS for the particular user according to the user profile, which describes the data accessible to the user, and to the user functions identified previously.

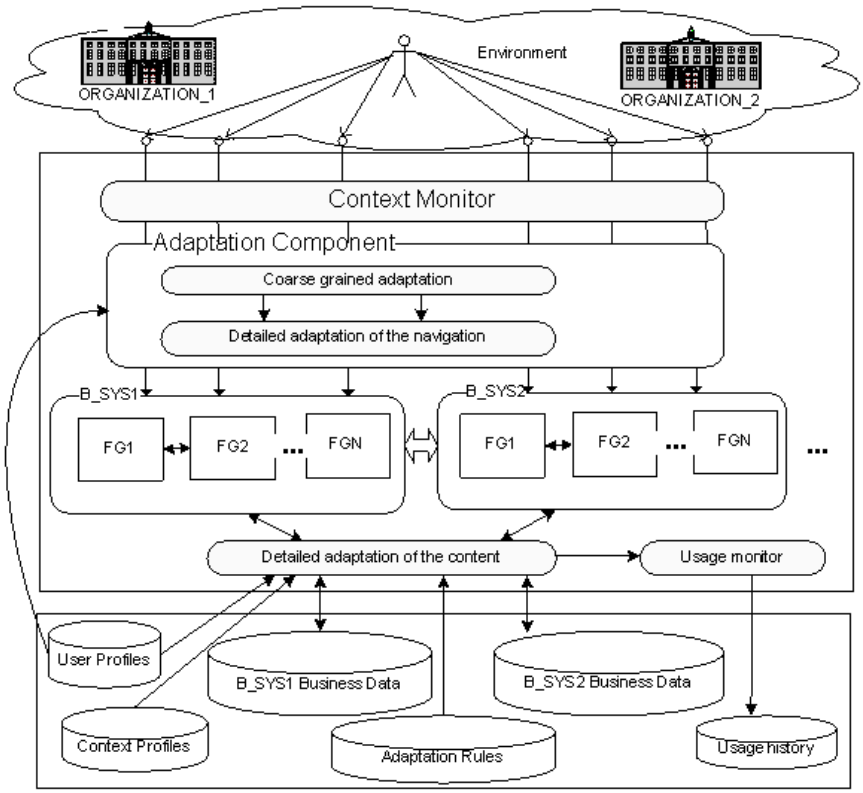

Fig. 1. Adaptation Architecture of WIS

These components will be described in the next sections of this paper in details.

2.1 Context Monitor

The usage context of the system is characterized by properties of the environment, which can influence the interaction between the user and the system in the time moment, when the interaction happens. The usage context is defined by the user, place, time, media used for the interaction, and the intention of the user to perform a particular action by means of the system (Fig.2).

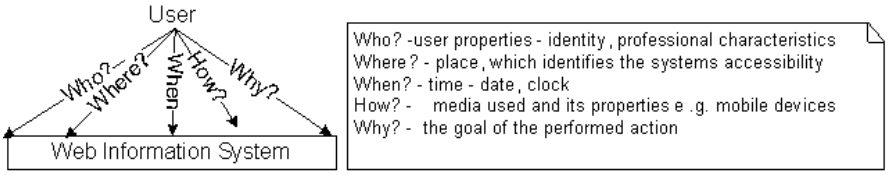

Fig. 2. The usage context of the system

The goal of the context monitor is to recognize the properties of the environment and to forward them to the adaptation component for usage in the adaptation of the WIS according to the particular environment.

2.2 Definitions

The profiles used in the proposed architecture will be defined by the UML class diagrams. Each class diagram defines one profile. The package diagram is used to describe dependencies between profiles (Fig.3). The class that is included in a diagram from other class diagram is depicted in grey.

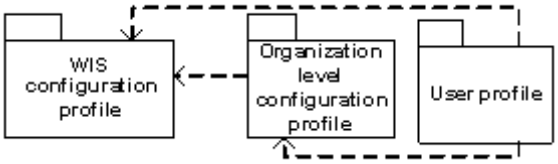

Fig. 3. The package diagram of WIS profile

WIS configuration profile is defined by the model given in the diagram (Fig. 4). The elements of this model are *Function group, Function*, and *Transition*. The *Function group* consists of many *Functions* implemented in WIS to support business functions of the organization. The function groups have *Transitions* defined between different instances of these groups. The transitions have allowed directions from one function group to another. The directions of transitions are modelled by relationships *From* and *To*.

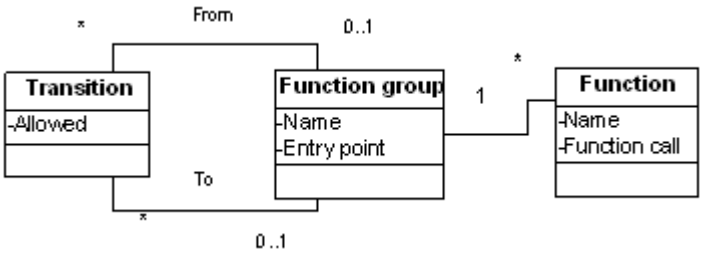

Fig. 4. Configuration profile of WIS

Function group is a grouping of WIS functions, which is characterized by a common *entry point*. The entry point is defined by entry location and entry type. Entry location in this case could be the procedure call in the browser, where the user chooses one of the links representing many entry points into WIS, and which activates a kind of login form. Each login form defines the entry type to the system.. For example, there may be different entry types for the authorized part of the system (the database authentication) and for the functions performing self-service tasks (the LDAP authentication).

Depending on the entry point, a joining of the function groups is defined to ensure for the user the option to access many function groups at once without additional navigation, login, and authentication. The joining of function groups is performed based on the allowed transitions defined in the configuration profile of the system.

Let FG1 and FG2 be two function groups. Two transitions between FG1 and FG2 are possible: the transition FG1→FG2 from FG1 to FG2 and the transition FG2→FG1 from FG2 to FG1. Which one of them is allowed, it is determined by the *Transition* attribute „allowed". The joining of two function groups could be performed, if the entry point into the function group will be used, from which the allowed transition starts. It could be defined joining FG1 and FG2 in one direction, but not in the opposite direction.

The configuration profile for the organization level instance of the WIS is defined by the model given in the diagram (Fig. 5).

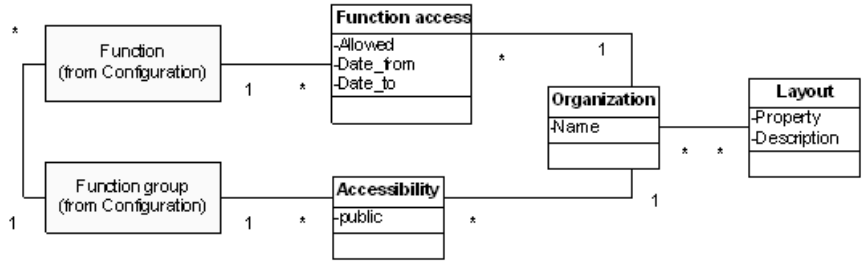

Fig. 5. The configuration profile for the organization level instance of the WIS

For each particular organization, this profile describes the individual properties (e.g. local configuration, layout etc.) essential for the organization level instance of the WIS. The elements of the model are *Organization, Function access, Accessibility,* and *Layout*. The model uses also classes from configuration profile of WIS (*Function* and *Function group*).

Organization represents those business organizations that use the WIS. *Accessibility* defines for the organization a WIS instance describing, whether access to particular function group is allowed or not to this organization. A function group could be defined as *public*. *Function access* defines for the organization, which functions are accessible in predefined time periods in the case, when the corresponding function group also is allowed. *Layout* describes properties of some elements, if a personalized look of the organization level instance of the WIS is needed. These elements are such as logo, background, font etc.

The user profile is defined by the model given in the diagram (Fig. 6). The user profile describes the affiliation of the user in the particular time period in one or more organizations using WIS. This description defines for the user the corresponding one or more organization level WIS instances. Additionally the user profile describes the user access rights to the functions and data restrictions in the WIS database. The elements of the model are *User, Affiliation, Table*, and *Row_Access*. The model uses also one class from configuration profile of WIS (the class *Function*) and one class from the Organization profile of WIS (the class *Organization*).

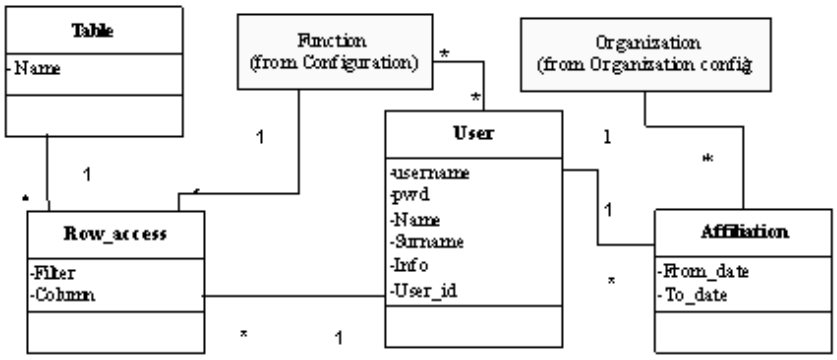

Fig. 6. User profile of WIS

User class describes the personal information of the user, login information, and *user_id* that is used to identify the data about the user in the WIS data base. Other data accessible to the user are described by *Row_access* and *Table*. The data restrictions are defined by column filters for the tables. The *Affiliation* describes the affiliation of the user in one or more organizations in the time period. The user has access to the *Functions* within the organization level instance of the WIS.

2.3 Adaptation Component

The adaptation component performs adaptation in two levels: a coarse grained and detailed adaptation. The result of the adaptation is a WIS instance adapted in two levels:

- The result of coarse grained adaptation is an organization level WIS instance adapted to the needs of the particular organization. The adaptation result has also a joined function groups from different WIS instances, if a user belongs to more than one organization using the WIS.
- The result of detailed adaptation is a WIS instance adapted for the individual needs of the user, which is characterized by personalized navigation and personalized content within the framework of the organization level instance of WIS.

2.3.1 Coarse Grained Adaptation

The adaptation component receives from the context monitor the information about the context of the system usage: user, entry point, and time. The adaptation component gets the profiles from the data layer according to the given context: the WIS configuration profile, the configuration profile for the organization level instance of the WIS, and user profile. The coarse grained adaptation is performed by the following operations:

- The construction of initial configuration of WIS
- The WIS adaptation in organization level
- The joining of instances of function groups
- The selection of allowed function groups
- The adaptation of layout

Let us denote the operation "The construction of initial configuration of WIS" with *start_config*. The initial configuration (IC) of WIS is formed according to the configuration profile of WIS. IC consists of all Function groups described in the profile.

Let us denote this operation "The WIS Adaptation in Organization Level" with *adapt_org* (*IC, user, time*). The goal of this operation is the adaptation of IC according to the organization level profile. The operation *adapt_org* indicates the organization from the user profile, which corresponds to the user identified by the context monitor. Then the IC is checked, if all function groups and functions are accessible to the particular organization. The adapted instance WIS_org is the result of this operation.

If the user belongs to more than one organization, then according to the profile of the user, adapted instances are made for each such organization: e.g. *WIS_org1*

and *WIS_org2*. For each organization of the user, the *adapt_org* operation is performed separately.

The operation "The Joining of Instances of Function Groups" is necessary in the case, when the user belongs to more than one organization and the corresponding WIS instances, e.g. *WIS_org1* and *WIS_org2* are made. Let us denote the joining operation with *FG_union (WIS_org1, WIS_org2).* *FGj_1* and *FGj_2* will be the two instances of the arbitrary Function group FGj in two WIS instances *WIS_org1* and *WIS_org2.*

FG_union is defined only for instances of the same function group, and only in the case, when the user belongs to both organizations having *WIS_org1* and *WIS_org2*. The joined instance *WIS_org1_org2* will consist of such joined function groups and Function groups, which are defined in the WIS organization profile only for one organization.

The operation "The Selection of Allowed Function Groups" is necessary in the case, when the user belongs to more than one organization and *adapt_org* and *FG_union* operations are performed. After that a selection of Function groups, according to the entry point is conducted. Let us denote the group selection operation with *FG_select (entry point).*

Let us assume that an entry point belongs to a function group *FGk*. Let *FGi* be an arbitrary function group; *Tij* be a transition from *FGi* to *FGj*, defined in the configuration profile of WIS. Then it is possible to define a transition chain *TCk,* which consists of sequential transitions *Tij* between function groups starting with function group *FGk*. The number of different possible *TCk* is predefined with the configuration profile of WIS.

The result of *FG_select (entry point)* is the adapted joined WIS instance *WIS_org1_org2*, which contains all function groups that are accessible with any of transition chains *TCk*.

Let us denote with *adapt_layout* the operation "The Adaptation of Layout", which performs the adaptation of the layout of WIS for the needs of the organization. When the user belongs to one organization, the layout properties are found out from the configuration profile for the organization level of WIS and applied to the WIS instance. In the case, when the user belongs to more than one organization, the layout properties are used according to the organization, which has the entry point used by the user.

2.3.2 Detailed Adaptation

The detailed adaptation is based on the user profile. The more detailed adaptation is made for the WIS instance that is already adapted by operations of the coarse grained adaptation. The detailed adaptation has two steps:

- detailed adaptation of the navigation,
- detailed adaptation of the content.

Let us denote these steps with *adapt_navig* (user) and *adapt_data* (user).

The operation *adapt_navig* (user) corresponding to the parameter „user" finds out the user rights to the functions. Within the framework of the adapted WIS instance gained during the coarse adaptation, an adapted navigation according to user rights is created. The adapted navigation provides access to allowed functions using links. We can say that an adapted instance with an adapted navigation of WIS for the user „WIS_user" is made.

The operation *adapt_data* (user) corresponding to the parameter „user" finds out the user restrictions to the data, which are defined using tables, their columns and filters for these columns. The WIS instance with the adapted navigation „WIS_user" is supplemented with the content, which in general case is defined by the data usage of the functions. The operation *adapt_data* (user) adapts the content corresponding to the restrictions defined for the user.

This approach could be used, when simple restrictions should be defined. More complicated adaptation rules are defined and stored in the Base for Adaptation Rules. The adaptation rules could be defined for example with OCL statements. These adaptation rules also define the data restrictions for the users.

3 The Case Study of WIS Adaptation

We will explain the usage of the adaptation architecture in the WIS for Universities (UWIS). Three function groups are defined in the configuration of the UWIS:

- We will denote with *FGa* the group of authorized functions. The authorized functions are the functions that are provided to support the administrative functions of the universities. These functions perform operations with the data of students and employees, e.g. "Entry of student grades", "Change of student statuses", etc. To perform these functions, the users should have access to data of other employees or students within the responsibilities of users' work.
- We will denote with *FGs* the group of self-service functions. These are functions, which the user can perform only with his personal data or data, which are connected with his data. For example, the user can apply for the study courses, he can control the actual status of his grades, statuses, etc.
- We will denote with *FGp* the group of public functions. These are accessible to all users without authentication and usually provide access to different reports that include public data.

The configuration profile of organization level instance of UWIS contains information about logo and other layout elements.

Let us consider a case, when the user is an employee at the Liepaja University (LiepU) and he studies at the University of Latvia (LU). Both universities use UWIS. UWIS has many entry points depending on the number of participants of UWIS project. The context monitor identifies the entry point, the user, and the

time, when the UWIS is accessed. The adaptation component finds out that the user belongs to two universities and constructs two instances of UWIS with all three function groups according to UWIS configuration profiles of LU and LiepU (Fig. 7).

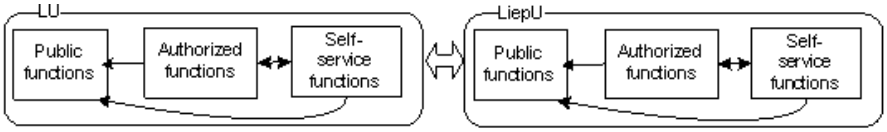

Fig. 7. WIS instances for the case study

The instances of UWIS contain also the predefined transition between function groups, the arrows denote the directions of transitions.

5 Results, Conclusion and Further Work

Our architecture is successfully implemented and is being used in two WIS. The proposed architecture of WIS adaptation is used in the project, where 12 universities are using each an adapted instance of the system. All universities (12) use the authorized function group, 8 universities use the public part of the system, but 10 universities use the self-service functions.

Fig. 8. Entry points into function groups in the UWIS (LAIS in Latvian)

The picture (Fig. 8) shows all entry points into the system, including one common entry point into authorized function group and entry points into self-service function groups of each particular university. These entry points into self-service groups serve also as entry points to public functions of each university.

The other implementation is used in the Road Safety Directorate and insurance companies for the car registration WIS.

Our architecture is not an exception among many architectures and approaches discussed previously. We also provide a little different view on the problem of

WIS adaptation, looking from perspective of organizations that are interested in an adapted WIS for their needs, if a unified system to support similar business processes is used. We introduce two levels of adaptation – coarse grained adaptation for the organization and fine grained adaptation for the user, we also support the situation, when users can work with many instances of the system adapted for different organizations, which are integrated into one instance for a particular user.

Some aspects of adaptation provided in the architecture were out of the scope of this paper, e.g. the process execution monitoring to provide more effective WIS and to find out new possibilities for the adaptation. We plan to use our work in process measurement and supplement it with adaptation aspects. Also more complicated adaptation scenarios that use adaptation rules should be formalized and described in detail.

Acknowledgments This work has been supported by ESF project No. 2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044.

References

1. Fraternali, P.: Tools and Approaches for Developing Data Intensive Web Applications: a Survey. J ACM Comput. Surv. 31(3), 227--263 (1999)
2. De Virgilio, R., Torlone, R.: A General Methodology for Context-Aware Data Access. In: Proceedings of the 4th ACM international Workshop on Data Engineering for Wireless and Mobile Access, MobiDE '05, pp. 9--15. ACM, New York (2005)
3. Abbar, S., Bouzeghoub, M., Kostadinov, D., Lopes, S., Aghasaryan, A., Betge-Brezetz, S.: A Personalized Access Model: Concepts and Services for Content Delivery Platforms. In: Kotsis, G., Taniar, D., Pardede, E., Khalil, I. (eds.) Proc. of the 10th Int. Conf. on Information Integration and Web-Based Applications and Services, iiWAS '08, pp. 41--47. ACM, New York (2008)
4. Valeriano, D., De Virgilio, R., Torlone, R., Di Federico, D.: An Efficient Implementation of a Rule-based Adaptive Web Information System. In: Frasincar, F., Houben, G.-J., Thiran, P. (eds.) Proc. of Int. CAISE Workshop on Web Information Systems Modeling (WISM'06), CEUR Workshop Proceedings, vol. 239 (2006)
5. De Bra, P., Houben, G.J., Wu, H.: AHAM: a Dexter-based Reference Model for Adaptive Hypermedia. In: Proc. of the 10th ACM Conf. on Hypertext and Hypermedia Hypertext'99, pp. 147--156, ACM Press (1999)
6 .Tvarožek, M., Barla, M., Bieliková, M.: Personalized Presentation in Web-Based Information Systems. In: Leeuwen, J., Italiano, G.F., Hoek, W., Meinel, C., Sack, H., Plášil, F. (eds.) Proc. of the 33rd Conference on Current Trends in Theory and Practice of Computer Science. LNCS, vol. 4362, pp. 796--807. Springer-Verlag (2007)
7. Jablonski, S., Petrov, I., Meiler, C., Mayer, U.: Guide to Web Application and Platform Architectures (Springer Professional Computing), SpringerVerlag (2004)
8. Clements, P. , Northrop, L. Software Product Lines: Practices and Patterns. Addison-Wesley, (2002).
9. Bass;L., Clements;P, Kazman, R. "Software Architecture in Practice", Second Edition, Addison-Wesley Professional, (2003), p. 560.
10. Microsoft, "Architecture Strategies for Catching the Long Tail", April 2006