Adaptation of the Presentation in a Multi-tenant Web Information System

Aivars Niedritis[1], Laila Niedrite[1]

[1] University of Latvia, Faculty of Computing, Raina boulv.19, Riga, Latvia
{Aivars.Niedritis, Laila.Niedrite}@lu.lv

Abstract. We introduced a Web Information System (WIS) adaptation architecture that is based on Software as a Service (SaaS) ideas. It includes adaptation components, which allow adaptation in two levels: the organizations and the users get their own adapted instance of the WIS. The user interface in case of multi-tenancy should be dynamically adapted according to the particular organization and user. The same application component that contains a set of fields, controls and other interface elements should be varied according to the usage context. In this paper we provide a method for adapting the user interface within the proposed adaptation architecture that uses a set of rules describing the sequence of allowed actions generated as a response to the performed user's activities.

Keywords: Multi-Tenancy, Software as a Service, Adaptation, Presentation

1 Introduction

Web application is a hybrid between a hypermedia and an information system [1]. Customization and dynamic adaptation of content structure, navigation primitives, and presentation styles should be provided as features of a Web application [1].

A Web-based Information System (WIS) is adaptive if the delivery of content and services is dynamically modified according to the context of the client [2]. Personalization requires an adaptation of applications according to preferences of the user and the context of the user [3].

The context is a set of properties that describe the environment where the user interacts with a WIS [3], e.g. time, place, device, user, and environment. An autonomous aspect of the WIS usage context is represented by a profile [2]. Another term used in the adaptation sphere is the definition of configuration as a specification of how the information has to be delivered to the user [2].

There are many approaches [2], [3], [4], [5] for the WIS adaptation based on different understandings of the context, profile, and configuration.

WIS development projects should also take into account the issues of flexibility and cost efficiency in order to deliver the most appropriate system to the user. So, Software product line approach (SPL) [6], Software as a service (SaaS) approach [7] or others could be used.

SaaS is "software deployed as a hosted service and accessed over the Internet" [7]. In this case services can be provided to different organizations to support common business processes. Organizations subscribe to use the software and pay for the usage. The provided application is hosted on vendor's servers. One of the possible implementations of SaaS applications is multi-tenant single instance solution [7]. The tenants are different organizations that use the same application instance; however, the data is distinguished between tenants. Architectures for the SaaS applications should solve the problems of how to deliver an adapted application to multiple users using the same application instance and how to support data processing and delivery to an organization and to a particular user. The customization of SaaS application can be performed using the metadata-based configuration.

In our previous research [8] we introduced a WIS adaptation architecture that is based on SaaS ideas. It includes additional adaptation components which allow adaptation in two levels: organizations and users get their own adapted instance of the WIS. We provided also a method for the adaptation of the content [9] where problems concerning data storage, integration, and delivery to the user are solved within the proposed adaptation architecture.

Besides the dynamically provided appropriate data, the user interface in case of multi-tenancy should also be dynamically adapted according to the particular organization and user. The same application component that contains a set of fields, controls, and other interface elements should be varied according to the usage context. In this paper we provide a method for adapting the user interface within the proposed adaptation architecture that uses a set of rules describing the sequence of allowed actions generated as a response to the performed user's activities. The adapted user interface contains elements needed to perform this action sequence.

The rest of the paper is organized as follows. Section 2 presents the adaptation architecture of a Web Information System and describes two levels of adaptation. In Section 3 the profiles that are used to ensure the adaptation are defined. Section 4 describes the presentation metamodel. Section 5 describes how the adapted user interface is constructed based on a presentation metamodel and profiles. In Section 6 the related work is analyzed. Section 7 ends the paper with conclusions.

2 WIS Adaptation Architecture

We will provide a method for user interface adaptation as a part of the adaptation architecture of the Web Information System (WIS) that we presented in our previous work [8]. This architecture could be used for organizations in the same business domain and with the same target user group. This architecture uses the ideas of SaaS, e.g. multi-tenancy.

The adaptation architecture of the WIS consists of following components:

1. Context monitor that determines context properties,

2. An adaptation component that determines the initial configuration of WIS and performs the adaptation of WIS in two levels. Organization level instances of WIS are created during *coarse level* adaptation. The detailed adaptation performs navigation adaptation and detailed adaptation of content that integrates user's data from all WIS instances available to the particular user.
3. WIS data layer contains profiles that are used in both adaptation levels and business data. Data is physically stored in one database instance, but it consists from virtual data stores owned by particular organizations.

Let's consider an example when the customer is a user of the same WIS used by two different organizations. The adaptation architecture components in this case are shown on the Figure 1.

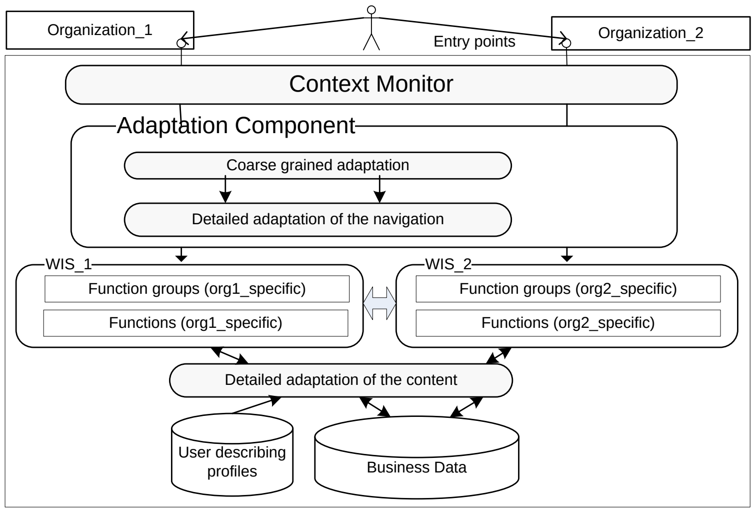
Organization_1
Entry points
Organization_2
Context Monitor
Adaptation Component
Coarse grained adaptation
Detailed adaptation of the navigation
WIS_1
Function groups (org1_specific)
Functions (org1_specific)
WIS_2
Function groups (org2_specific)
Functions (org2_specific)
Detailed adaptation of the content
User describing profiles
Business Data

Fig. 1. Adaptation architecture of WIS [7]

The organization level instances of WIS in our example are depicted in Figure 1 as WIS_1 and WIS_2. This is a simplified schema of the architecture. The whole description of the architecture is given in [8].

The architecture allows adapting the initial configuration of the WIS for each particular organization and, moreover, for each particular user, and supports the integration of all functions and all user data from all WIS instances which are used by the particular user into one user interface.

The problems discussed later in this paper are mainly connected with coarse level adaptation of the WIS, but to give an insight of the whole process of adaptation in two levels a short description of adaptation operations is given.

During the coarse level adaptation organization level instances are dynamically built. The adaptation operations of coarse level adaptation are the following [8]: the construction of initial configuration of WIS, the WIS adaptation in organization level, joining of instances of function groups, selection of allowed function groups, and adaptation of layout.

Afterwards, the detailed level adaptation is performed based on the user profile. The detailed adaptation has two operations [8]: detailed adaptation of the navigation and detailed adaptation of the content. The adaptation of the content corresponding to the particular user finds out his restrictions to the data, which are defined in user profile, for example, by using tables, their columns and filters for these columns. The WIS instance with the adapted navigation is supplemented with the content which in default is defined by the data usage of the functions. During the adaptation of the content it is adjusted to the restrictions defined for the user.

3 Extensions of WIS Profiles

The adaptation architecture uses a number of profiles to perform the adaptation: configuration profile, organization profile, and user profile. In our previous research [8] we focused on constructing different adapted instances of web information system from the set of all functions that form the whole WIS. Model diagrams of all profiles in their initial version and more detailed description are given in [8]. In this work we will explain how the different adapted WIS instances are presented to the user. To ensure the adaptation of the presentation we will extend the initial profiles. The already existing classes of profiles are depicted in grey.

3.1 Extended Configuration Profile

Configuration profile describes the structure of the WIS, e.g. components, subsystems, etc., without applying adaptation. Figure 2 represents the extended WIS configuration profile.

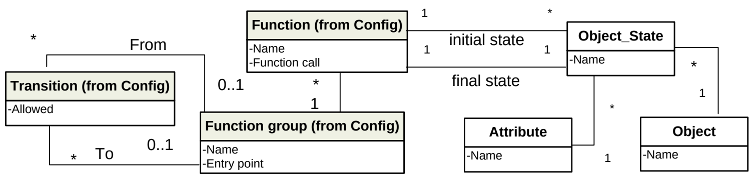

Fig. 2. Extended configuration profile

WIS configuration profile in its initial version consists of such classes as *Function group*, *Function*, and *Transition*. *Function group* is a grouping of WIS functions defined by an *entry point* to allow different technical implementations of WIS functions. For example, one possible situation where this grouping is necessary is the usage of different authentication methods for different function groups, e.g. authorized part of the system uses the database authentication and the self-service functions use the

LDAP authentication. User chooses a link to one of the login forms and then is directed to the corresponding function group of WIS. In the adaptation architecture of WIS different function groups can be joined to allow the user to work with many WIS parts and instances based on the allowed *Transitions* defined in the configuration profile of the system.

The initial profile is extended by the class *Object State* which represents the situation where functions can be state dependent. A function may be allowed in some definite states and forbidden in others. The state in our proposed architecture is introduced as an *Attribute* or set of attributes of an *Object*. If the status allows, the function can perform manipulations with an object that defines the state. A function is allowed according to the initial state of the object. After the function is fulfilled it changes the state of the object to the final state.

3.2 Extended Organization Profile

The organization profile describes properties essential for the organization level instance of the WIS, e.g. local configuration specific for the particular organization, layout, etc.

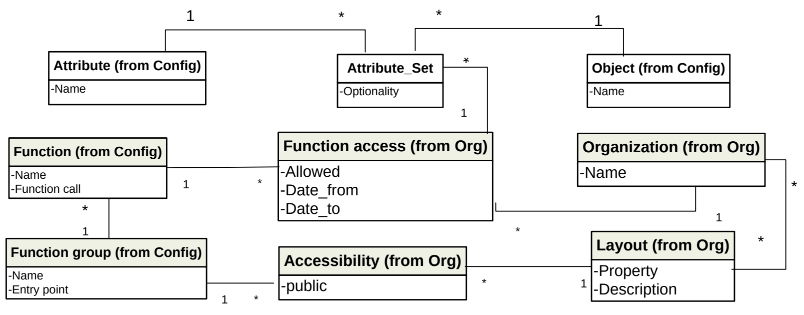

Fig. 3. Extended organization profile

WIS organization profile in its first version consists of classes *Organization, Function access, Accessibility,* and *Layout* (see Fig. 3). *Organization* represents the organizations that have their own instance of WIS. *Accessibility* defines an adapted instance of WIS by describing allowed function groups for the particular organization. *Function access* describes the adapted WIS instance for an organization in more detail by describing the accessible functions in predefined time periods. *Layout* specifies properties of some elements, if a personalized look of the organization level instance of the WIS is needed, e.g. logo, background, font, etc.

For each WIS instance of a particular organization the extended organization profile allows definition of an *Attribute Set* different from the initial attribute set used by a function in original configuration of WIS. The *Attribute Set* has associations with Attributes and Objects from the configuration profile. The feature “optional” of the attribute set describes if an attribute included in the attribute set is mandatory for the function execution logic.

3.3 User Profile

We do not introduce an extended user profile. The initial user profile describes the affiliation of the user in the particular time period in one or more organizations that are tenants of WIS and his access rights to functions and data. The elements of the user model are *User*, *Affiliation*, *Table*, and *Row_Access*. The profile uses also the class *Function* from the configuration profile and the class *Organization* from the organization profile. A more detailed description is given in [8].

4 Presentation Metamodel for WIS

To describe how an instance of WIS is presented to the user on the organization level, we described the elements of the user interface with a presentation metamodel (see Fig. 4).

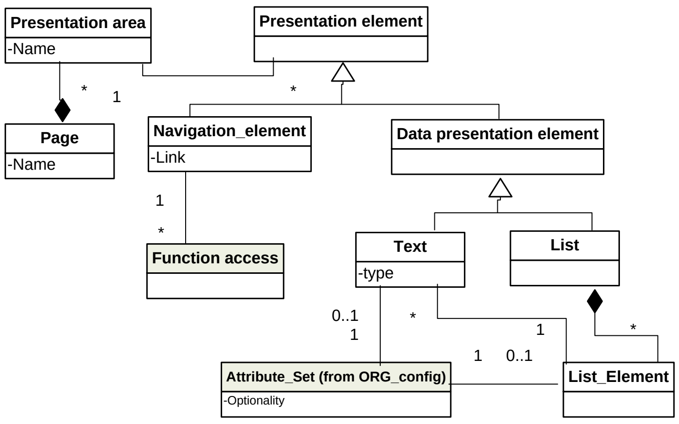

Fig. 4. Presentation metamodel for WIS

Page class is the main presentation element that contains all other user interface elements. Page class presents the Function group from the WIS basic configuration that is described with the configuration profile.

Page consists of at least one *Presentation area* which is provided for grouping of *Presentation elements.* A presentation element can be *Navigation element* or *Data presentation element*.

Navigation element is displayed as a link that represents a function call. In our WIS adaptation architecture the navigation element is not connected directly to the WIS function described in configuration profile, but is associated with a *Function* *access* class which supports presenting in the user interface only allowed functions of a particular WIS instance for a particular organization.

The data presentation element can be a kind of *Text* or *List* which both are different ways of displaying *Attributes* of the *Objects*. In our WIS adaptation architecture the data presentation elements are not connected directly to the attributes but are associated with an *Attribute Set* class which supports presenting in the user interface only allowed attributes of a particular WIS instance for a particular organization.

A List consists of *List elements* that also can represent attributes from the Attribute Set. The goal of introducing such presentation element is to allow a multiple choice from a set of attribute values. List elements also can be connected with Text presentation elements that can present additional information for each List element.

The property “Type” of the Text element describes whether the given presentation element for a particular WIS instance can only display the attribute value or the updates are allowed.

5 Construction of a Metamodel-Based and Adapted User Interface

The later described workflow (see Fig. 5) performs the construction of the user interface based on the previously described metamodel and uses the extended configuration and organization profiles of WIS adaptation architecture. The interface is generated with a close involvement of the user into the process. The workflow is adjusted to the situation when the choices of the user and the later interface construction depend on statuses of an object that is a focus of activities during business process completion. The user of a web-based business information system in such way gets an interface of WIS that is suited for the specific business situation.

We will consider each step of the process to describe the usage of profiles and adaptation operations, if applicable for the particular step, bearing in mind that only the organization level adaptation is made. The detailed level adaptation that uses the user profile is not the issue of this paper and is described in [9].

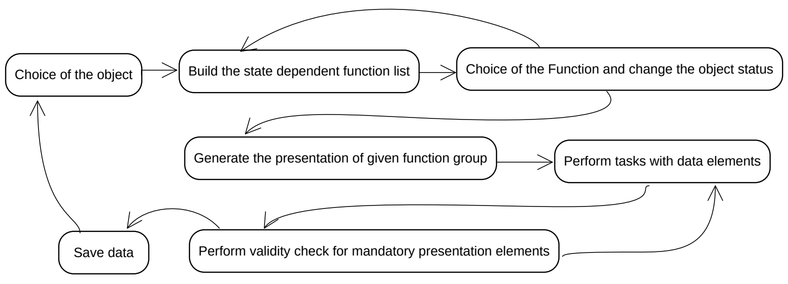
Choice of the object
Build the state dependent function list
Choice of the Function and change the object status
Generate the presentation of given function group
Perform tasks with data elements
Save data
Perform validity check for mandatory presentation elements

Fig. 5. User development in WIS adaptation

Chose the object. The user finds the object that will be the focus of activities during the later business process completion. For example, the user finds the record of an existing vehicle to perform later a number of operations with it.

Build the state dependent function list. The initial state of the object determines the available operations that can be applied to the object.

Let's see an example (see Fig, 6), which describes a situation with chosen object O1 having a state S1 and a list of allowed three functions F1, F2, and F3 with defined initial and final states. The user can then select either a function F1 or F3 whose description by a parameter 'Initial_state' also contains S1 among other allowed states.

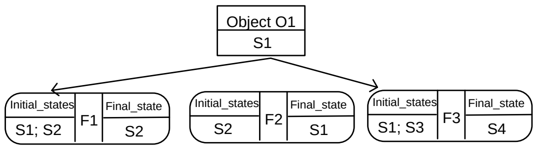

Fig. 6. An example of states of the object and functions

The list of operations allowed for the object (or functions in WIS context) is also based on configuration profile of WIS. The function list is presented in a presentation area as navigation presentation elements.

Chose the function and change the object status. According to the business needs, the user selects iteratively all necessary operations that should be performed with the object from the list of allowed operations displayed to him. After each choice the potential state of the object, if the operation will be later completed, is registered and it determines the content of the new state dependent function list.

In the previous example (See Fig, 6.) the status of the object is changed according to the value of the parameter 'Final_state' of the function. All possible scenarios of workflow composition for this example are: 1) F1→End 2) F1→F2→End; 3)F1→F2→F3→End; 4) F3→End. The 'End' in this case denotes that the user finishes the selection process.

The chosen functions are presented in a separate presentation area as navigation presentation elements.

Generate the presentation of the given function group. The user finishes the selection of functions and generates the presentation area that contains data presentation elements. The list of chosen functions and the organization profile of the company where the user works determines the list of attributes for particular WIS instance. The presentation metamodel also influences the look of the generated page. For example, the same attribute can be presented as text presentation element either of type "Input" or "Display" according to different user's choice of functions in presentations of different WIS instances.

For the previous example (see Fig. 6.) the presentation of WIS instance is given in Figure 7, which shows also the decisions of a user and state-dependent dynamic supply of allowed functions to the user, if he follows the workflow scenario 2) F1→F2→End described in previous subsection. Object area contains the presentation element O1, Function area contains the functions allowed according to the state of the selected object O1. Selected functions area contains the user's selection of functions. The data area is built after the user ends the construction of the desired scenario or workflow, and contains the attributes associated with functions also taking into account the configuration and organization profile for constructing an adapted WIS instance. All elements in the Figure 7 are stereotypes according to the presentation metamodel given on Figure 4.

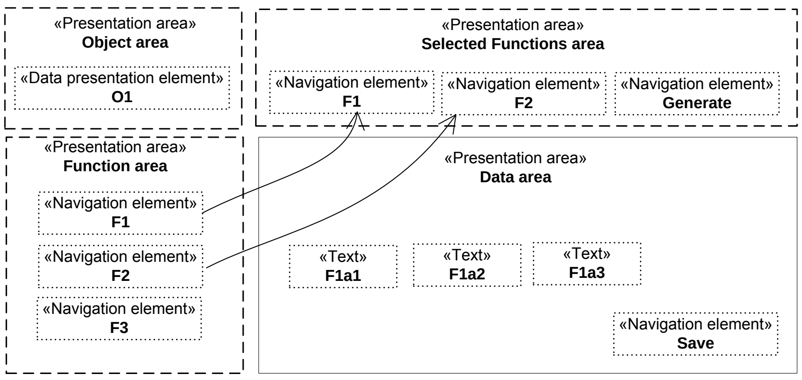

Fig. 7. User interface for the example

Perform tasks with data elements. This step does not influence the user interface and can only influence the values of object's attributes. Also, the attribute (or attributes) that defines the state of the object can be changed.

Perform validity check for mandatory presentation elements. Validity check performs evaluation, if all mandatory presentation elements have received their values, and before starting a new operation from the user's selection of functions also the actual state of object is always examined to ensure that a user has fulfilled all tasks from the workflow constructed by himself and has not overlooked any functions, thus, permitting illegal situation according to the objects state.

Save data. If the validity check is successful, the user can save changed data elements and start constructing a new workflow by selecting a new object.

Principles described in this section and also the whole WIS adaptation framework is implemented in the Vehicle Register. The Object in this implementation is a vehicle. The current status of the vehicle can be, for example, 'active', 'inactive', 'temporary', 'cancelled', etc. The operations allowed for a vehicle in the state 'active' are, for example, 'Rebuilding', 'Change the owner', 'Remove from the register', and many others. Examples of two organizations using the same WIS in form of their adapted instances are in this case Ministry of Transport that holds the vehicle register and certified service centres that among other services also provide technical rebuild possibilities which are later documented in their WIS instance.

6 Related Work

A number of works are published about WIS adaptation. WIS can be adapted according to the context using a profile that corresponds to a configuration [2]. The configuration in this approach defines the information delivery according to the requirements of adaptation for the profile. In [3] different personalization scenarios including preference-based recommendation, context-aware content delivery, and personalized access to multiple contents are presented, which are based on a set of services that implement a personalized access model (PAM). Also, a metamodel is given that defines notions of a profile and context. Personalized presentation layer of WIS is introduced in an architecture proposed in [5]. This presentation layer supports adapted navigation and different views on the presented data.

Also there exist metamodels for Web applications, for example, UWE [10], WebML [11], and many others. These metamodels define semantics of Web applications from different viewpoints.

The research in the fields of SaaS and multi-tenancy is concentrated on configurability issues. Applications can be configured by utilizing the software architecture to build connections between functional elements and configuration [12]. In [13] the authors use patterns to formalize the configuration requirements in seven categories, e.g. data, user type, business rules. Different configurable aspects of SaaS applications are also discussed in [14], and architecture to support configurability is provided.

Our WIS presentation metamodel is provided as a part of WIS adaptation architecture and defines how the adapted WIS instance is presented to the user. Our approach is complimentary to mentioned adaptation approaches and metamodels, and describes specific aspects of Web applications, namely, the aspects connected with SaaS and Multi-tenancy, when many different WIS instances should be provided to the users.

7 Conclusions

Presented approach to the WIS adaptation uses two levels of adaptation – coarse-grained for the organization and fine-grained for the user. The adapted WIS organization level instances are presented to users of different organizations.

We consider in our approach a specific type of business information systems that are implemented as WIS and that support business functions. We consider specific way of performing business processes when the workflow starts with a choice of an object whose state determines the later steps of performing business activities.

The construction of the user interface in our approach is based on a presentation metamodel and uses the configuration and organization profiles of WIS adaptation architecture. The interface is generated with a close involvement of the user into the process. The choice of the user and the interface construction depends on state of an object that is a focus of activities during business process completion. The user of a web-based business information system in such way gets an interface of WIS that is suitable for the specific business situation.

The described architecture and user interface construction according to the presentation metamodel is implemented and is being used in two different WIS; one of them is used for the car registration WIS serving more than 20 regions of the state with approximately 100 different vehicle registrars who all can get their adapted interface to the particular business situation.

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. 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)
6. Clements, P., Northrop, L. Software Product Lines: Practices and Patterns. Addison-Wesley, (2002)
7. Microsoft, Architecture Strategies for Catching the Long Tail, (2006)
8. Niedritis, A., Niedrite, L.: The Adaptation of a Web Information System: a Perspective of Organizations. In: Pokorny, J., Repa, V., Richta, K., Wojtkowski, W., Linger, H.; Barry, C., Lang, M. (eds.). Information Systems Development, Business Systems and Services: Modeling and Development, pp. 539–550. Springer (2011)
9.. Niedritis, A.: Delivery of Consistent and Integrated User's Data within a Multi-Tenant Adaptive SaaS Application. In: Niedrite, L., Strazdina, R., Wangler, B. (eds.), Perspectives in Business Informatics Research, Local Proceedings of 10th Int. Conf., BIR 2011 Associated Workshops and Doctoral Consortium, pp. 307–314. Riga Technical University, Riga (2011)
10. Koch, N., Kraus, A.: Towards a Common Metamodel for the Development of Web Applications. In: Cueva Lovelle, J.M., Gonzalez Rodriguez, B.M., Joyanes Aguilar, L., Labra Gayo, J.E., Paule Ruiz, M.P. (eds.), Proceedings of 3rd International Conference on Web Engineering (ICWE 2003). LNCS, vol. 2722, pp. 497–506. Springer Verlag (2003)
11. Schauerhuber, A., Wimmer, M., Kapsammer, E.: Bridging existing Web modeling languages to model-driven engineering: a metamodel for WebML. In: Workshop proceedings of the 6th Int. Conf. on Web Engineering, ACM Press: Palo Alto, California (2006)
12. Wang, H., Zheng, Z.: Software Architecture Driven Configurability of Multi-Tenant SaaS Application. In: Wang, F.L., Gong, Z., Luo, X., Lei, J. (eds.), Proceedings of the 2010 In-

ternational Conference on Web Information Systems and Mining (WISM'10). LNCS, vol. 6318, pp. 418–424. Springer-Verlag, Berlin, Heidelberg, (2010)

13. Shim, J., Han, J., Kim, J., Lee, B., Oh, J., Wu, C.: Patterns for Configuration Requirements of Software-as-a-Service. In Proceedings of the 2011 ACM Symposium on Applied Computing (SAC '11), pp. 155–161. ACM, New York (2011)
14. Nitu: Configurability in SaaS (Software as a Service) Applications. In Proceedings of the 2nd India Software Engineering Conference (ISEC '09). pp. 19–26, ACM, New York (2009)