Personalized Presentation in Web-Based Information Systems

Michal Tvarožek, Michal Barla, Mária Bieliková

Institute of Informatics and Software Engineering, Faculty of Informatics and Information Technologies, Slovak University of Technology
Ilkovičova 3, 842 16 Bratislava, Slovakia
{Name.Surname}@fiit.stuba.sk

Abstract. Large information spaces and complex functionality of contemporary systems together with the advent of the Semantic Web are big challenges for the design of simple yet powerful user interfaces. The complex nature of systems thus requires the use of several presentation, personalization and user modeling methods and techniques to address the growing needs and requirements of both users and systems. We present a design of an architecture for the personalized presentation layer of a web-based information system that employs a set of interconnected software components implemented as autonomous software tools for presentation, personalization and user modeling to support features such as navigation support and different views on the presented data, acquisition and evaluation of user characteristics and user adaptation and personalization. The feasibility of our design is evaluated by its realization in the labor market application domain.

1 Introduction

The web contains vast amounts of information in its visible "surface" part and in the "deep web". These include both billions of publicly accessible web pages and large searchable databases. Moreover, contemporary web-based information systems (WIS) need to deal with increasing amounts of data while new challenges are introduced by emerging technologies, such as the vision of the Semantic Web.

Furthermore, the enormous amount of available information provided by content-oriented WIS and the overall complexity of applications resulted in the need for adaptation of their interfaces. Presentation tools today need to create personalized output based on the user characteristics stored in a user model maintained e.g., by a user modeling server [1] or by the information system itself. Although the user model can be acquired in many different ways, the current focus is on methods that minimize user involvement in the process.

While many existing WIS work with the aforementioned databases as parts of the "deep web", a web-based presentation interface is also the preferred means of presentation for many new applications, partly because it can also be used for the integration of different functionality into a single user interface. To facilitate the development of WIS, entire presentation frameworks were created that support the operation of several interconnected presentation tools. However, the design and architectures of integrated WIS with support for adaptation of content, layout and presentation, (semi)automatic user modeling and integration of a set of different presentation tools have not yet been explored sufficiently.

Our aim is to design and evaluate a presentation layer architecture including also personalization facilities to effectively serve the evolving field of WIS development. The rest of the paper is structured as follows. Section 2 presents related work. In section 3, we propose an architecture for the personalized presentation layer of WIS with the respective "user modeling back-end" that allows for the creation of an effective adaptive user interface. We provide an overview of the concepts, goals and methods used in our design. We also describe the requirements, purpose and integration of individual tools that are necessary to realize the proposed functionality. Next, in section 4 we describe our evaluation of the proposed architecture in the domain of online job offers. Finally, in section 5 we summarize our contribution and present ideas for future work.

2 Related Work

The issues of large Web-based Information Systems (WIS) have already been explored in several works from different perspectives with focus on their design, architecture and performance. In [2] authors discuss the problems of large enterprise WIS on the example of a banking system. They identified the need for strong usability, performance and maintainability and defined a "3G-WIS" as "*realize a pleasant use of high quality services, and they will come*" type of service for customers. The authors propose WIS adaptation based on context-awareness and present the corresponding architecture design on an example e-banking WIS.

Existing reference models for adaptive hypermedia can serve well as a basis for WIS architecture design. Apparently the best known is AHAM – Adaptive Hypermedia Architecture Model [3], which is an extension of the generic Dexter reference model for hypermedia. It focuses on the architecture of educational adaptive hypermedia and adds adaptation support based on a domain, user and teaching model with focus on presentation specifications, concepts and concept relationships. Furthermore, unlike the Dexter model, AHAM introduces a permanent and continuously updated user model used for adaptation throughout multiple sessions. The Munich Reference Model for Adaptive Hypermedia [4] follows object-oriented approach to design. Based on the Dexter and AHAM models it specifies an architecture using UML and formal specification expressed by OCL (Object Constraint Language).

With the growing complexity and requirements of adaptive WIS and their practical deployment, issues concerning efficiency and overall performance increase in importance. In [5] the authors propose a flexible presentation architecture for adaptive hypermedia based on the generic Model-View-Controller architecture. They identify problems associated with caching of personalized content, propose caching at different stages of the presentation pipeline and describe examples and performance measurements on the example of an educational WIS.

To facilitate the design and development of WIS for the Semantic Web, the existing model based Hera methodology for WIS was extended with support for adaptability and adaptivity [6]. The authors proposed the use of a profile and user model for adaptation and as well as the use of a conceptual model and domain model that took advantage of ontologies with RDF(S) markup.

In [7] the authors further elaborate on the use of the Hera methodology for Semantic Web WIS design and describe a series of model-driven transformations that are used to generate a hypermedia presentation for the retrieved data. The proposed architecture consists of a semantic layer that specifies the data content, an application layer that describes the abstract view on the data and a presentation layer that specifies the presentation details.

Furthermore, the authors in [8] claim that personalization and adaptation is inevitable in presentation design but has not yet been the central issue of WIS methodologies unlike adaptation in conceptual and navigation design. They describe the AMACONT project that introduced a component-based XML document format, which enables the composition of personalized ubiquitous web presentations from reusable document components encapsulating adaptive content. They combine Hera at the conceptual and navigation levels and AMACONT at the presentation level into an integrated framework for WIS design.

To summarize, many current approaches focus on the design and modeling of the domain model (WIS information content) with extensions related to personalization and performance improvements. We stress the incorporation of user modeling as autonomous client and server side software tools into the system's architecture and its integration with presentation and personalization tools.

3 Personalized presentation layer architecture

Many contemporary information systems employ a standard three-layer architecture consisting of a data layer, an application layer and a presentation layer. In this context, the data layer stores and retrieves data from a database, the application layer performs the core business logic of the system while the presentation layer takes care of the presentation and user interaction [9]. Our focus in this paper is the presentation layer of the typical three-layer architecture with additional focus on web-based information systems for Semantic Web applications. In this respect, we consider the support for the Semantic Web and adaptive hypermedia technologies to be imperative.

Consequently, we think of the presentation layer as a *personalized presentation layer* that performs two primary tasks. First, it provides a user interface that offers simple access to all of the system's functionality while effectively hiding all of the system's inner complexity from the user. Second, it dynamically adapts the user interface to the needs, usage patterns and goals of individual users in order to increase their comfort, productivity and satisfaction by exploiting information stored in a user model. A third – somewhat related task is to create a comprehensive log of user activity, evaluate it and extract and store meaningful user characteristics, which will then be used in the adaptation process. Thus a suitable personalized presentation layer must fulfill the following requirements:

- Provide an easy to use, user-friendly and intuitive user interface.
- Provide simple access to the system's functionality.
- Offer support for personalization by individual users, which requires
  - support for user model creation and maintenance (logging and acquisition of user activity and characteristics), and
  - support for user model usage to provide adaptation to individual users' needs and usage patterns.

As such, the aforementioned tasks are performed by the presentation part, the personalization part and the user modeling part of the system respectively. Since these tasks depend on each other, our design employs an integrated set of cooperating software components (tools) to realize the necessary functionality (see Figure 1).

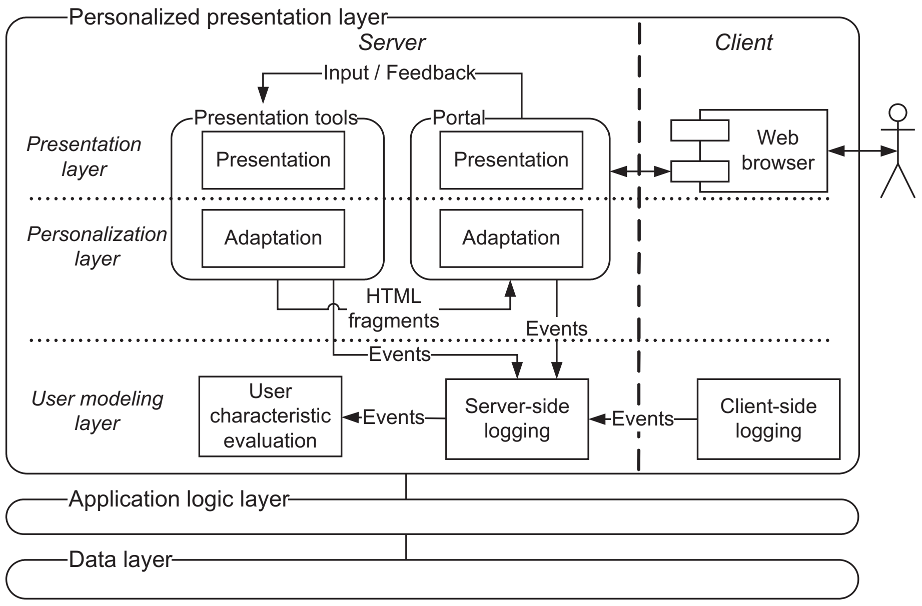

Fig. 1. WIS architecture of the personalized presentation layer.

Presentation tools and the portal work with the domain and user models and consist of a presentation and a personalization layer respectively. Individual presentation tools are depicted in the center and forward output to the web portal, which in turn provides an interface to the client web browser (right). The user modeling layer is shown at the bottom and includes both client and server side logging and user characteristic evaluation. All of the already described functionality of a WIS can be examined at the following two levels of abstraction.

Portal level. The primary purpose of the portal level is to act as means of integration for the system's functionality by providing a common environment for individual tools. The portal provides a common global navigation interface, authentication, authorization and user management services to individual tools (e.g., information about the current user session). It defines the overall high-level functionality and layout of items such as global menus, fields, links and that of individual tools, as well as the overall navigation structure of the whole site.

Tool level. The purpose of individual tools is to realize or provide access to the functionality of the system. Presentation tools are responsible for the presentation of information and user interaction with each tool being responsible for the handling of its part of the user – system interaction. Thus the tool level defines the internal low-level functionality and layout of items (e.g., controls, menus, text areas, images) for individual presentation tools and the functionality of processing tools which are not directly used for presentation (e.g., for user modeling or business logic).

3.1 Presentation

The task of the presentation part is to provide an effective user interface. While the focus of successful presentation is the layout, style and consistency of the interface, using the most suitable means to present information is also of prime importance. Thus, our portal design consists of pluggable user interface components called portlets. Each portlet represents some logically well defined functionality realized by one or more cooperating tools. The advantage of this approach is that various types of presentation of the same content can be provided to different users. For example, two different presentation tools – one producing graphical output, another producing textual output can be used and the user can choose the most suitable way of presentation.

Consequently, at the tool level, presentation deals with the specifics of individual tools such as layout and style of labels, controls, tables and images all of which are handled by the tools themselves. However, at the portal level the main focus is the overall layout of tools – portlets, each of which contains some specific functionality such as: global navigation (e.g., menus), authentication (e.g., login), notifications (e.g., RSS), user input (e.g., forms), data presentation (e.g., faceted browsers, RDF and OWL visualizers).

3.2 Personalization

Both levels of presentation would be insufficient without proper support for personalization, which can either affect the *visualization* by adapting the layout, style and feel of the user interface or the *functionality* and/or *behavior* of individual tools.

At the portal level, the layout of individual tools can be adapted by changing the order of the displayed portlets. In this way, the usability of the interface can be improved or the accessible functionality can be adapted to the current needs and goals of the user. This can be done by adaptively adding or hiding items in global menus or by adding or hiding portlets. Furthermore, the portal provides a “skinnable” interface, which allows users to choose their preferred presentation style. Also, additional restrictions can be defined based on user roles and access privileges thus allowing or denying access to specific portlets and their functionality.

The tool level offers possibilities for the adaptation of visualization of individual tools as well as for the adaptation of their functionality. First, a set of guidelines that make it possible to adapt the interface of individual tools to better suit user characteristics can be defined. Next, the functionality of individual tools can be adapted by the tools themselves by exploiting knowledge about user characteristics stored in the user model.

The use of personalization introduces new issues that must be considered in an adaptive information system. The user should have the possibility to explore his own user model in order to know, understand and possibly modify the information, which the system stores about the user and uses for adaptation [1]. Furthermore, support for scrutability is important especially in cases when the system behaves unexpectedly, e.g., recommends illogical items. These needs translate at least into a simple user model browser/editor and a “scrutability explorer” that might be built into each presentation tool. Alternatively it might be a separate tool that would be invoked by other tools.

3.3 User modeling

Adaptation performed by *adaptation tools* is based on data stored in the user model. This model needs to be created and maintained during the user modeling process. The goal of the process is to identify user characteristics, which might be consecutively used for the adaptation of the system’s operation (e.g., information filtering), for the adaptation of its output (e.g., sorting of search results) or input (e.g., interpretation of user commands) [10]. Thus a successful set of user modeling tools must fulfill the following requirements:

– Provide means for both manual and automatic acquisition of user behavior and user characteristics ideally with little user involvement.
– Provide means of automatic evaluation of the acquired data.
– Allow for the scrutability of the user model by allowing the user to view, inspect and modify its content.

In general, the process of user modeling is perpetual and includes collection of data about the user and processing of the collected data and the update of the user model [11].

User data collection. The primary purpose of the user data collection stage is to acquire as much relevant data (e.g., user behavior, documents, preferences) about users as possible while ideally keeping the amount of necessary user involvement as low as possible. This is the reason why among many possible user modeling approaches, we focus on an approach which is based primarily on the analysis of user behavior within a system. While this approach reduces the amount of user involvement, it requires comprehensive logs of user actions, which must fulfill the following requirements:

1. All relevant user actions (e.g., load of a page, following of hyperlink, use of the back button) and the corresponding (server) events must be logged.
2. For each action the exact time (timestamp) must be logged.
3. Each action must be semantically described (i.e., its meaning must be known).

To fulfill the aforementioned requirements, we chose to expand the presentation and personalization layer onto the client where we use client side monitoring tools to log such events (representing user actions), which the server is normally unaware of (due to browser caching mechanisms, JavaScript interactions etc.). To fulfill the requirement for semantically described actions, we enhance the standard server side logging mechanism with appropriate semantics. Every presentation tool is responsible for supplying the semantics of the performed actions by using a predefined ontology of events.

User model update. The aim of the data processing stage is to evaluate the acquired data and estimate meaningful user characteristics with high confidence that might be consequently used for adaptation. The processing depends on the chosen method of data collection and on the nature of collected data. If data represent user actions, individual user characteristics can be estimated by analyzing user navigation on a web site or in the visible information space and by analyzing user feedback on the displayed content. Since user modeling is a perpetual process, each time a user characteristic is identified, the user model is updated to reflect the newly discovered knowledge.

4 Design evaluation

To evaluate the proposed design we used it for the presentation layer of a system developed in the NAZOU research project[1] aimed at retrieving information in a heterogeneous information space (job offers in the Internet), processing it and presenting it to the user employing adaptive techniques [12].

4.1 Presentation and personalization

In section 3 we introduced the basic requirements on the functionality of the personalized presentation layer, which consists of several cooperating software tools. We used a single tool for portal level functionality – *JOP – Job Offer*

[1] NAZOU Project, `http://nazou.fiit.stuba.sk`

Portal which is the primary user interface and processes all user inputs while returning the corresponding outputs.

The main purpose of *JOP* is to integrate different functionality and provide a form-based user interface tailored for ontology editing with support for dynamic form generation for a given ontology [13]. *JOP* also allows users to register, log in and customize the global layout of the user interface.

Since *JOP* is based on Apache Cocoon[2] which supports flexible inclusion of existing functionality into the portal solution, the individual functionality and results of all other tools are easily integrated into the user interface provided by *JOP*. Each tool (or set of tools) is represented by *a coplet* (cocoon portlet) whose position within the overall layout of the portal can be adjusted by the user, who can also minimize, maximize or hide it completely.

On the tool level we developed two cooperating presentation tools, with each tool being responsible for the adaptation of its content according to data from user model. We also developed several simpler tools for the editing of the user profile and for forms filling, both of which are integrated into our portal solution.

Factic – Faceted Semantic Browser [14] is a tool which allows users to effectively navigate the information space by choosing restrictions on the displayed content. It is fully integrated into *JOP*, which supplies it with information about the current user for adaptation purposes. As input it takes user actions and returns the logical description of the content that should be displayed. Its output can be further processed by a set of XSL transformations to directly create valid XHTML output or alternatively it can be sent to *Prescott* for further processing.

Prescott [15] is a presentation tool able to visualize domain dependent content (e.g., job offers) in a flexible and configurable manner using the *Fresnel* presentation ontology. It defines various views on domain content using "lenses", which can be defined dynamically based on user preferences. As input, *Prescott* takes the logical description of the content (e.g., the URIs of job offer instances) and returns an XHMTL fragment with the visualization of the content. Additionally, *Prescott* can take advantage of user characteristics stored in a user model to choose the most appropriate lens to apply on the ontology individuals that should be displayed. *Prescott* is invoked mainly by *Factic* every time the user changes the selected restrictions or decides to view details of a job offer.

4.2 User Modeling

To fulfill the requirements on the data collection stage (see Section 3.3), we developed the client side monitoring tool *Click* [16]. This JavaScript based tool captures and logs browser events and sends them to the server.

Server side logging is enhanced by the *SemanticLog* [16] tool, which combines information from presentation tools and logs acquired by *Click* to create a comprehensive log of user actions with added semantics which are suitable for further processing.

[2] The Apache Cocoon Project, `http://cocoon.apache.org`

The consecutive data processing is performed by the *LogAnalyzer* [17] tool, which estimates user characteristics and stores them in the user model. Since the used method implies the mainly domain-dependent nature of the revealed characteristics, *LogAnalyzer* needs better "understanding" of the displayed domain content and uses the services provided by the *ConCom* [16] tool, which compares ontological concepts by using various comparison strategies.

4.3 Evaluation

We evaluated the proposed architecture in the domain of job offers by taking advantage of the domain and user ontologies created at our university for project NAZOU. The domain ontology of job offers is used to model domain concepts and their relations as well as to store data about individual job offers. Similarly, the user ontology models user specific concepts and contains data about individual user characteristics. Both of these ontologies are used by the respective tools to process job offer or user related requests (e.g., *Factic* presents job offer instances, while *LogAnalyzer* creates and updates user characteristics).

The domain ontology consists of about 700 classes of which 670 belong to hierarchical classifications with a maximum depth of 6 levels. The ontology contains a test base of about 1 000 job offer instances acquired by different means prior to system evaluation (e.g., manual input, web wrappers)[18]. It is stored in the semantic part of the corporate memory [19] that is responsible for providing means for querying and manipulating the information content enriched by semantics as well as providing the physical back-end for persistent and transient storage of the semantics. The semantic model of the Web content is represented using an ontology (in OWL DL language). Moreover, the user ontology was partially derived from the domain ontology by adding new concepts and relations. The content of the user ontology – individual user characteristics is created and updated by the *LogAnalyzer* tool during system operation.

We developed the above mentioned presentation and user modeling tools and integrated them in a complex WIS thus successfully using the proposed personalized layer architecture design. To verify the integration aspects of the proposed architecture we integrated additional tools that, for example, acquire and present job offer clusters from the application layer or display personalized instance ratings. The integration and interaction of individual tools during request processing is shown in Figure 2.

Client requests are first received by *JOP* which forwards them to appropriate presentation tools or processes them directly in which case it next sends a response to the client. The *Factic* presentation tool prepares the response by querying the domain and user models and invoking *Prescott* to generate XHTML fragments as necessary. It then logs the respective event and resulting display state via the *SemanticLog* web service. Lastly it combines individual XHTML fragments into the final response and sends it to *JOP*, which in turn forwards it to the client.

Furthermore, *JOP* and *Click* independently log events that resulted from user actions via *SemanticLog*, which notifies *LogAnalyzer* about new events for

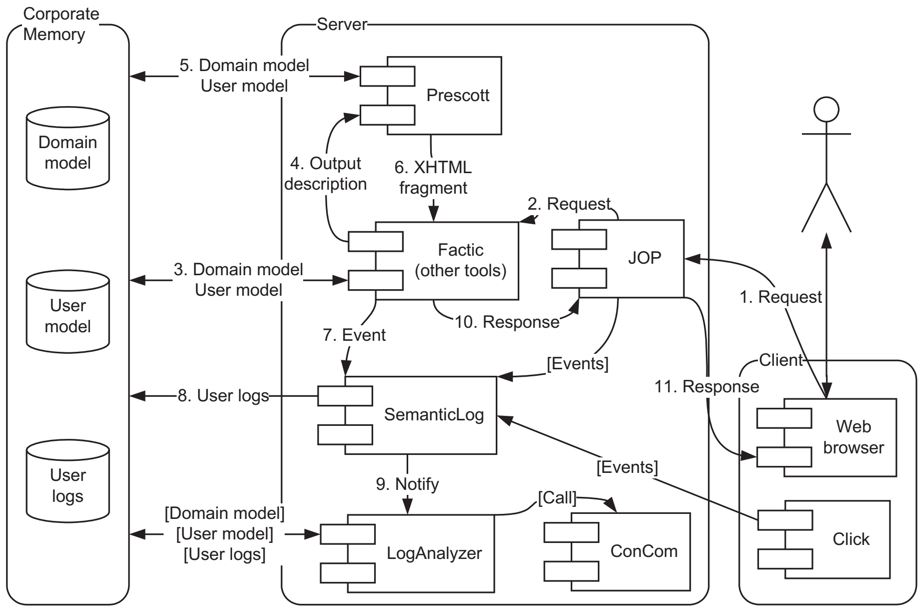

Fig. 2. Request processing in a WIS based on the proposed architecture of the personalized presentation layer.

processing. *LogAnalyzer* in turn processes these events asynchronously taking advantage of *ConCom* and updates the user model with newly identified user characteristics.

The user characteristics are used by several software tools from the application layer which contribute to further refinement of user interests. The *Top-k aggregator* tool retrieves the most relevant job offers with respect to user preferences (e.g., salary, education requirements, place) based on ordered lists of user preferences [20]. The *Aspect* tool searches for similar documents (job offers) based on a probabilistic model for soft clustering [21]. Using the described approach for the comparison of domain and user ontology instances the devised clusters are presented to the user based on her characteristics.

5 Conclusions

In this paper we presented requirements that should be met by successful web-based information systems and their respective presentation, personalization and user modeling parts. We proposed an architecture of the personalized presentation layer that consists of several cooperating tools that in combination provide the presentation of data, adaptation of the system's operation and the automatic acquisition and evaluation of user characteristics.

Furthermore, we have realized and evaluated the proposed architecture as well as its individual tools in the domain of online job offers. We continuously cope with the problem of general immaturity of Semantic Web technologies, which are still in a development (e.g., ontologies and ontological repositories).

We also intend to use the proposed architecture in another research project aimed at adaptive presentation and navigation in the space of scientific publications [3]. This second evaluation will be performed with a different and much larger domain ontology and with an adapted user ontology, thus putting more focus on domain independence of individual tools and performance issues.

Future work includes deeper analysis and experimentation with the performance and communication overhead associated with the use of independent tools. We also work on the integration of additional tools developed in the course of our research project into the portal *JOP*, the evaluation of additional interconnections between tools and the addition of new tools for specific tasks.

Acknowledgment. This work was partially supported by the Slovak Research and Development Agency under the contract No. APVT-20-007104, the State programme of research and development under the contract No. 1025/04 and the Scientific Grant Agency of Slovak Republic, grant No. VG1/3102/06.

References

1. Bieliková, M., Kuruc, J.: Sharing user models for adaptive hypermedia applications. In: 5th Int. Conf. on Intelligent Systems Design and Applications, ISDA'05, Wroclaw, Poland, ACM Press (2005) 506–511
2. Binemann-Zdanowicz, A., Kaschek, R., Schewe, K.D., Thalheim, B.: Context-aware web information systems. In: Proc. of the 1st Asian-Pacific Conf. on Conceptual Modelling, APCCM '04, Australian Computer Society, Inc. (2004) 37–48
3. 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, New York, NY, USA, ACM Press (1999) 147–156
4. Koch, N., Wirsing, M.: The munich reference model for adaptive hypermedia applications. In De Bra, P., Brusilovsky, P., Conejo, R., eds.: Proc. of Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems, AH'02, Springer, LNCS 2347 (2002) 213–222
5. Ullrich, C., Libbrecht, P., Winterstein, S., Mühlenbrock, M.: A flexible and efficient presentation-architecture for adaptive hypermedia: Description and technical evaluation. In Kinshuk et al., ed.: Proc. of 4th IEEE Int. Conf. on Advanced Learning Technologies, ICALT'04, IEEE Computer Society (2004)
6. Frasincar, F., Houben, G.J.: Hypermedia presentation adaptation on the semantic web. In De Bra, P., Brusilovsky, P., Conejo, R., eds.: Proc. of 2nd Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems, AH'02, Springer, LNCS 2347 (2002) 133–142
7. Houben, G.J., Barna, P., Frasincar, F., Vdovjak, R.: Hera: Development of semantic web information systems. In: Proc. of 3rd Int. Conf. on Web Engineering, ICWE'03, (Springer, LNCS 2722)

[3] MAPEKUS Project, `http://mapekus.fiit.stuba.sk`

8. Fiala, Z., Frasincar, F., Hinz, M., Houben, G.J., Barna, P., Meißner, K.: Engineering the presentation layer of adaptable web information systems. In N. Koch et al., ed.: Proc. of 4th Int. Conf. on Web Engineering, ICWE'04, Springer, LNCS 3140 (2004) 459–472
9. Fowler, M.: Patterns of Enterprise Application Architecture. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2002)
10. Kay, J. Human Factors and Ergonomics Series. In: User Modeling for Adaptation. Lawrence Erlbaum Associates (2000)
11. Brusilovsky, P.: Methods and techniques of adaptive hypermedia. User Model. User-Adapt. Interact. **6**(2-3) (1996) 87–129
12. Návrat, P., Bieliková, M., Rozinajová, V.: Methods and Tools for Acquiring and Presenting Information and Knowledge in the Web. In: Int. Conf. on Computer Systems and Technologies, CompSysTech'05, Varna, Bulgaria (2005)
13. Barla, M., Bartalos, P., Sivák, P., Szobi, K., Tvarožek, M., Filkorn, R.: Ontology as an Information Base for Domain Oriented Portal Solutions. In: Proc. of 15th Int. Conf. on Information Systems Development, ISD'06. (2006)
14. Tvarožek, M.: Personalized Navigation in the Semantic Web. In V. Wade et al., ed.: Proc. of 4th Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems, AH'06, Springer, LNCS 4018 (2006) 467–471
15. Bieliková, M., Grlický, V., Kuruc, J.: Framework for presentation of information represented by an ontology. In Vojtáš, P., ed.: Workshop on Theory and Practice of Information Technologies, ITAT'05, Račkova dolina (2005) 325–334
16. Andrejko, A., Barla, M., Bieliková, M., Tvarožek, M.: Tools for User Characteristics Acquisition. In: Datakon'06. (2006) Accepted.
17. Barla, M.: Interception of User's Interests on the Web. In V. Wade et al., ed.: Proc. of 4th Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems, AH'06, Springer, LNCS 4018 (2006) 435–439
18. Bartalos, P., Barla, M., Frivolt, G., Tvarožek, M., Andrejko, A., Bieliková, M., Návrat, P.: Building an ontological base for experimental evaluation of semantic web applications. In van Leeuwen, J., Italiano, G.F., van der Hoek, W., Sack, H., Meinel, C., Plášil, F., eds.: Proc. of SOFSEM '07, Springer, LNCS (2007) Accepted.
19. Ciglan, M., Babik, M., Laclavik, M., Budinska, I., Hluchy, L.: Corporate memory: A framework for supporting tools for acquisition, organization and maintenance of information and knowledge. In J. Zendulka, ed.: Proc. of 9th Int. Conf. on Inf. Systems Implementation and Modelling, ISIM'06, Perov, Czech Republic (2006) 185–192
20. Gurský, P., Lencses, R., Vojtáš, P.: Algorithms for user dependent integration of ranked distributed information. In et al., M.B., ed.: TCGOV 2005 Poster Proceedings, Bozen-Bolzano, Italy (2005)
21. Polčicová, G., Tiňo, P.: Making sense of sparse rating data in collaborative filtering via topographic organisation of user preference patterns. Neural Networks **17** (2004) 1183–1199