RESTful User Model API for the Exchange of User's Preferences among Adaptive Systems

Martin Balík and Ivan Jelínek
Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University, Karlovo náměstí 13, 121 35 Prague, Czech Republic

Keywords: Adaptive Hypermedia, Personalization, User Model, Data Exchange, REST API.

Abstract: Adaptive Hypermedia Systems observe users' behavior and provide personalized hypermedia. Users interact with many systems on the Web, and each user-adaptive system builds its own model of user's preferences and characteristics. There is a need to share the personal information, and the current research is exploring ways to share user models efficiently. In this paper, we present our solution for personal data exchange among multiple hypermedia applications. First, we designed a communication interface based on the REST architectural style, and then, we defined data structures appropriate for the data exchange. Our user model is ontology-based and therefore, the data from multiple providers can be aligned to achieve interoperability.

1 INTRODUCTION

Hypermedia applications on the World Wide Web (WWW) are used by people on a daily basis. People use applications intended for work, entertainment, communication, learning, etc. To improve usability and facilitate the orientation within the large amount of information, many of the applications provide automated personalization and adaptation features. Such applications are called Adaptive Hypermedia Systems (AHS) (Brusilovsky, 2001; Knutov et al., 2009) and belong to the category of user-adaptive systems.

User-adaptive systems monitor users' behavior and keep track of characteristics, behavior, preferences, etc. of each individual user. The collection of personal data associated with a specific user is called the User Model (UM). The User Model is typically built individually within each user-adaptive application. This can cause a lot of issues, including the cold start problem (Salton and McGill, 1983), information inconsistency (Vassileva et al., 2003), bothering users with initial setup and asking for similar data input, narrow personal information domain, etc. In our work, we focus specifically on Adapitve Hypermedia Systems (AHS). Many different AHSs are used by a single person on a daily basis. The effort of extracting user's characteristics is commonly repeated in multiple self-contained applications. Moreover, individual user models can include complementary information. Therefore, it would be very beneficial to share the personal information and keep it synchronized between all applications used by the person.

Sharing a User Model is a required, but a very challenging task. Fortunately, the users are willing to share personal information (Kobsa and Teltzrow, 2006; Gross and Acquisti, 2005). It is important to find a balance between information revelation and personal privacy. The convenient choice of information providers and proper integration of data sources will bring many benefits to the user.

The user usually works with diverse services on the Web. Sometime, even the domain is quite similar. In other cases, there can be connections with other people with corresponding interests and new links to reusable data. On the web, everything is interconnected. Though, adding semantics to the raw data brings new possibilities how the data can be used. Through semantic annotations, the vision of Linked Data (Bizer et al., 2009) is becoming true.

The main problem of personal information sharing is the heterogeneity of User Models. To deal with the interoperability problems, especially the semantic heterogeneity of User Models needs to be addressed (Carmagnola, 2009). Users' data are stored in AHSs in different formats and use different syntactic and conceptual structures. Solving the AHS interoperability problems by utilizing an Application Program Interface (API) is an appropriate research direction as discussed in the context of adaptive educational systems in (Aroyo et al., 2006).

Balík M. and Jelínek I..
RESTful User Model API for the Exchange of User's Preferences among Adaptive Systems.
DOI: 10.5220/0004896806270634
In *Proceedings of the 16th International Conference on Enterprise Information Systems* (ICEIS-2014), pages 627-634
ISBN: 978-989-758-028-4

The paper proposes a new adaptive hypermedia system architecture and a method of personal data exchange. RESTful web service is used to expose both user's preferences and through user modeling process obtained characteristics.

The paper is structured as follows. Section 1 deals with the description of AHS and the tasks to solve. In Section 2, a current state of the art of the discussed topic is being reviewed. In Section 3, both the proposed personal data storage structure and the RESTful user model integration interface are described in detail. In Section 4, a use case scenario utilizing the benefits of the RESTful user model interface is demonstrated. Finally, the paper concludes by summarizing results of the research and indicates the directions of the future work; see Section 5.

2 RELATED WORK

User-adaptive applications are fundamentally based on a process of observing user's behavior and storing relevant user-specific information. This process is called user modeling, and data is stored into a user model. The user modeling process is a marginal topic for the presented research, and an overview of existing user modeling techniques can be found in (Sosnovsky and Dicheva, 2010; Aroyo and Houben, 2010). In next subsections, we will first review existing approaches used to share user-specific information stored in the user model, and second, we will explain fundamentals of the Web architecture, Web services and Representational State Transfer.

2.1 User Model Sharing Approaches

Recently, motivated by the expansion of mobile and ubiquitous devices, the researchers noticed the need of sharing personal profiles of users, to enhance the adaptation abilities of user-adaptive applications. Various techniques of sharing personal data have been proposed (Carmagnola, 2009; Sosnovsky et al., 2009; Cena and Furnari, 2009). The main obstacle is usually to solve the data interoperability problems. There are currently two main approaches. The first possibility is the way of standardization. The systems need to adhere to a fixed representation that needs to be respected by all service providers. The second approach is using mediation techniques to transfer the data from one representation to another.

Standardization-based approach of AHS integration assumes a common semantic representation of user models within all participating systems, usually expressed by a shared ontology (Heckmann et al., 2005). Implementing domain models of adaptive systems as ontologies is the first step toward interoperability. A standardized user-modeling ontology is a possible solution to make the information exchange possible. However, the fundamental requirement of this approach is that all participants agree upon the standardized ontology, which may pose an issue for some of them (Berkovsky et al., 2009).

As a generalization of a standardization-based approach, the central user modeling server can be assumed. A solution, where adaptive systems do not need to support user modeling was used in (Kay et al., 2002).The networked adaptive applications act as clients, they simultaneously update the central user model on the server, and they can request back personal information when needed. Another solution based on a central server is presented in (van der Sluijs and Houben, 2005), where the exchange of user data between applications is supported by Semantic Web technologies. The authors call the component providing user model storage the Generic User Modeling Component.

More complex solutions than standardization are utilized by mediation (Berkovsky et al., 2007). Without any standard vocabulary, it is necessary to solve syntactic and semantic heterogeneity issues. Ontology mediation is the process of reconciling differences between heterogeneous ontologies in order to achieve inter-operation between data sources annotated with and applications using these ontologies (de Bruijn et al., 2005). To overcome the heterogeneity of user modeling data, two steps are required. First, the reasoning and inference mechanisms for converting data between various representations, applications and domains need to be developed and applied. Second, the semantically enhanced knowledge bases are exploit, facilitating the above reasoning and inference (Berkovsky et al., 2009).

Although, there have been projects like (Heckmann et al., 2005) primarily focusing on the standard, widely accepted user modeling ontology, (Martinez-Villaseñor et al., 2012) claims that the standardization approach is not a feasible solution. Such statement suggests to follow the second direction, the mediation of different domains based on natural language processing and artificial intelligence. However, there is also a possibility of a hybrid approach combining both standardization and mediation approaches. Such unification is the aim of our approach.

2.2 The Web Architecture

Representational State Transfer (REST) (Fielding and Taylor, 2002) architectural style was proposed by

Table 1: Vocabulary namespaces used in the REST API.

Prefix	Namespace URI	Description
dcterms:	http://purl.org/dc/terms/	Dublin Core vocabulary
foaf:	http://xmlns.com/foaf/0.1/	Friend of a Friend (FOAF) vocabulary
sioc:	http://rdfs.org/sioc/ns#	Semantically-Interlinked Online Communities (SIOC) vocabulary
owl:	http://www.w3.org/2002/07/owl#	Web Ontology Language (OWL) vocabulary
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF vocabulary
rdfs:	http://www.w3.org/2000/01/rdf-schema#	RDF Schema vocabulary
xsd:	http://www.w3.org/2001/XMLSchema#	XML Schema (XSD) vocabulary
um:	http://intelleo.eu/ontologies/user-model/ns	IntelLEO User Model Ontology
gomawe:	http://fel.cvut.cz/gomawe/0.1/	GOMAWE Architecture Ontology

Roy Fielding and is based on a set of principles for designing network-based software architectures. The set of principles was defined by four interface constraints: identification of resources, manipulation of resources through representations, self-descriptive messages, and hypermedia as the engine of application state.

Web services based on the principles of REST are called RESTful and can be considered as an alternative to the SOAP-based web services. In the past, RESTful services were used only for simple ad-hoc services, and the area of enterprise systems was scoped to the WS-* standards, e.g., SOAP, WSDL, WS-Addressing, WS-Security. This is no longer the case and RESTful services have been successfully applied in many enterprise applications. The challenge is to use them correctly and to be able to align them to solve the real problems (Adamczyk et al., 2011). The comprehensive comparison of both technologies was presented in (Pautasso et al., 2008; Pautasso and Wilde, 2009). Conclusions of the comparison give an advantage to the RESTful services in Web integration scenarios and prefer WS-* Web services for enterprise application integration, where advanced security and Quality of Service (QoS) is required. The practical comparison in (Guinard et al., 2011) concludes that REST is lightweight, scalable, very easy to understand, learn, and implement.

There are two important terms when speaking about REST methods – safety and idempotence (Fredrich, 2012). Safety means that calling the method does not cause side effects and does not change the state of the server. For example, the `GET` method of an API must adhere to the safety definition, otherwise it can cause problems for other services and result in unintended changes on the server. Idempotence refers to a method that will produce the same results if executed once or multiple times. The `PUT` and `DELETE` methods are defined to be idempotent. Therefore, it should be ensured that making multiple requests produce the same result on the server. Safe methods are idempotent at the same time, because they do not cause any changes on the server.

3 GENERIC AHS ARCHITECTURE

In our previous work, we have formalized the Generic Ontology-based Model for Adaptive Web Environments (GOMAWE) (Balík and Jelínek, 2008). This model defines the fundamental components of an adaptive hypermedia application. One of the components is the integration interface intended for exposing user-specific data in the local storage to external applications, and, at the same time, for obtaining user-specific data from external providers.

Based on the GOMAWE formal specification, the Adaptive System Framework (ASF) (Balík and Jelínek, 2013) was built to support AHS development. ASF provides the most typical AHS components serving as building blocks for further development. The most recent extension of the framework is the integration component based on the principles described in this article.

3.1 User Model

In our design, the user model has a special architecture that was devised from the requirements, including types of stored information and methods of information retrieval. We divided the user data into two parts – the User Profile and User Model. This division corresponds to explicit and implicit personalization styles. Implicit personalization is performed by the adaptive system. On the other hand, explicit personalization is performed by the user using special features of the system. A system with such personalization features is called adaptable system. Our architecture can be perceived as a hybrid solution combining both personalization methods.

The User Profile contains explicit user's preferences, i.e., preferences explicitly filled by the users. This data is stored as key-value pairs, see Definition 1.

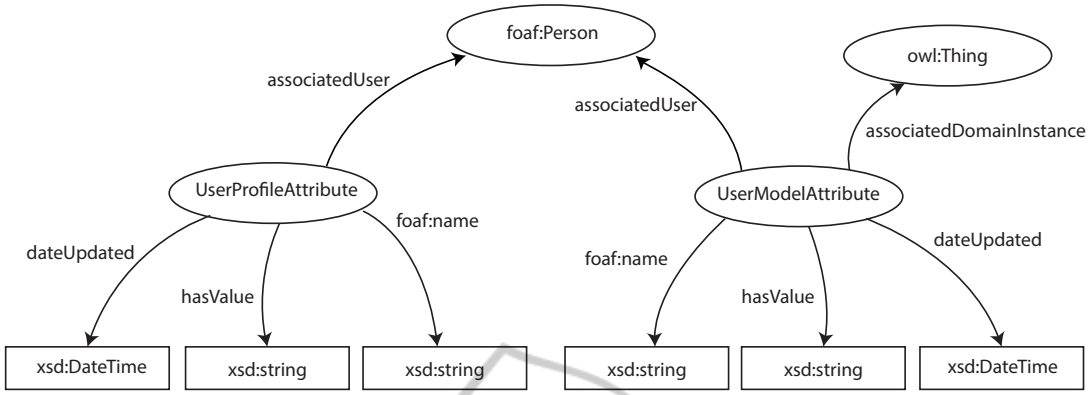

Figure 1: Selected parts of the GOMAWE Architecture Ontology.

The key is usually a constant string defined by the developer of the application. Typical origin of the data is the "settings page" of the application. However, when we assume integration with other personal data providers, the User Profile data can be filled utilizing social networks' user profiles available in services like Facebook, Twitter or LinkedIn.

Definition 1 (User Profile)**.** The User Profile of a user u is a tuple $UP_{(u)} = (A,V,r)$

$$r : A \rightarrow V | \forall a \in A : r_{(a)} \in V_a, \quad (1)$$

where A is a set of attributes defined as the vocabulary of user's characteristics, $V = \bigcup_{a \in A} V_a$ is a set of attribute values and V_a is the range of attribute a.

The User Model denotes in our terminology a model containing implicit user's characteristics. While the explicit characteristics are set by the user oneself, the implicit characteristics are devised by the adaptive system. The Domain Model captures the most important types of objects in the application context. It defines the conceptual framework and semantics of hypermedia content. The system collects various values related to a specific object of the domain and stores the values into the User Model, see Definition 2. Moreover, the characteristics are categorized to a set of dimensions. The assignment of attributes and dimensions is utilized in the user modeling process, however, it will not be discussed here as it is not substantial for the web services and the application integration interface.

Definition 2 (User Model)**.** The User Model of a user u is a tuple $UM_{(u)} = (D,A,V,r)$

$$r : D \times A \rightarrow V | \forall a \in D \times A : r_{(a)} \in V_a, \quad (2)$$

where D is a set of domain instances, A is a set of attributes defined as the vocabulary of user's implicit characteristics, $V = \bigcup_{a \in A} V_a$ is a set of attribute values and V_a is the range of attribute a.

As examples of User Model attributes we can mention *knowledge* of a lesson in an educational application, or *bought* item indication in a web commerce application. The User Model is able to capture even relationships among users. Based on the fact that a user entity is part of the Domain Model, the User Model can include attributes *friend*, *follower*, etc. relating the user to other users. This information can be acquired from social networking applications.

3.2 A RESTful User-specific Data Interface

Based on the comparison in Section 2, we decided to use RESTful web services to design the user-adaptive application's personal data interface. First, the interface does not require advanced security, and second, the most important requirement is flexibility supplemented with easy and intuitive development.

The exchange of information is based on standard metadata vocabularies and ontologies. Utilizing the proposed unified user model data structure, ontologies can be aligned with relative ease and translation between two domains can be achieved. At the same time, the design does not force participating systems to agree fully on a fixed domain model ontology, and advanced mediation techniques can be used when needed.

3.2.1 RDF Vocabularies

Our user-data-integration interface uses three types of vocabularies – first, the standard vocabularies, second, user-model-specific vocabularies and finally, the domain-specific vocabularies. RDF resources and their attributes reuse existing, widely adopted vocabularies such as the Dublin Core (Dub,), Friend of a Friend (Brickley and Miller, 2010), and Semantically-Interlinked Online Communities (Sio,)

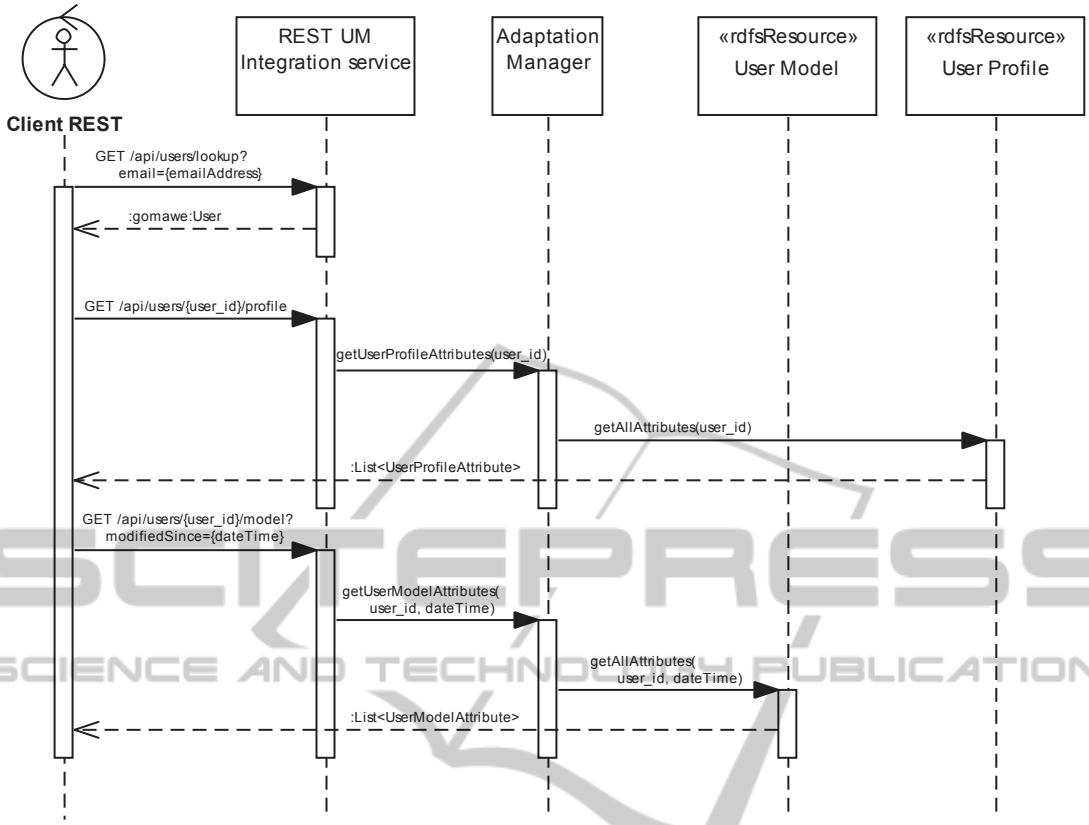

Figure 2: Example of request sequence to retrieve users's personal data.

vocabularies. To ensure good interoperability, ASF maps as many attributes as possible to these standard vocabularies. User-model-specific vocabularies used in our design include IntelLEO User Model Ontology (Jovanovic et al., 2012) and GOMAWE Architecture Ontology, see Fig. 1. Domain-specific vocabularies differ application from application and refer to domain-specific concepts to which the user's characteristics are related.

The Table 1 lists the vocabulary namespaces used in the ASF-based application REST API.

3.2.2 ASF-based Application API Resources

The fundamental resources of the RESTful API are listed in Table 2.

Table 2: Primary resources of the REST API.

Type	Example
gomawe:UserProfileAttribute	/api/users/{user_id}/profile
gomawe:UserModelAttribute	/api/users/{user_id}/model
foaf:Person	1) /api/users/{user_id} 2) /api/users/lookup?email={emailAddress}

In Table 2, `gomawe:UserProfileAttribute` refers to the list of User Profile attributes assigned to the user with the matching ID. `gomawe:UserModelAttribute` refers to the list of User Model data of the user with the matching ID. `foaf:Person` refers to the owner of the user account corresponding to the specified id (1) or email address (2).

3.2.3 REST Operations

This subsection summarizes the REST operations supported by the API. The API supports three HTTP operations: `GET, PUT, POST`. The `DELETE` operation is not supported, as the user data can only be extended or updated, and no attributes can be removed.

Only the RDF/XML format is supported at this time, therfore the HTTP Accept header value should be set to:

```
GET /api/users HTTP/1.1
Accept: application/rdf+xml
```

The following parameters can be used to refine the requests:

- Modified since – Parameter `modifiedSince` can be used to limit the listed attributes to only those that were modified after the entered time value.

The parameter value is of type `xsd:date` or `xsd:dateTime`

- Resource paging – Parameter `limit` can be used to limit the number of listed attributes to avoid large data transfers. Parameter `offset` can be used to request additional pages starting with the specified item. For both parameters integer type value is allowed.

The following are the typical uses of HTTP methods with explanatory examples:

- HTTP GET – The `HTTP GET` method is supported by all API resources. It is used to retrieve the user's account and his or her user modeling data. First, the application needs to negotiate the correct resource IDs and after matching user's account, the user model attributes can be requested, see Fig. 2. The following request can be used to retrieve user profile attributes of a user:

```
GET http://example.org/api/users/
   {user_id}/profile HTTP/1.1
Accept: application/rdf+xml
```

- HTTP PUT – The `HTTP PUT` method is supported only by UserProfile and UserModel resources. It is used to update the models by an external application. The `PUT` method operation needs to be idempotent, therefore, using this method means creating new attributes or replacing values of existing attributes.

```
PUT http://example.org/api/users/
   {user_id}/profile HTTP/1.1
```

- HTTP POST – The `HTTP POST` method is supported only by UserProfile and UserModel resources. The operation of this method is reserved for the cases of incremental updates of the models. It will be more extensively utilized in future extensions of the REST API. The more precise value updates can be based on arithmetic mean or the time of last update. This operation is not idempotent and should be called only once to avoid inappropriate user model changes.

3.2.4 Authentication

To avoid misuse of both the personal and the domain data provided by the RESTful application interface, authentication of requests is required. We use HTTP basic authentication. Using HTTPS protocol is recommended. Otherwise, the username and password would be sent without encryption.

4 APPLICATION USE-CASE

The proposed approach of AHS integration will be demonstrated in an adaptive learning scenario. There are currently three separate systems used by students in a programming course, each with a different purpose. Although each of the systems stores different information, they can benefit from each other, exchange user's data and extend understanding of the user's knowledge and preferences.

The integration module implementation is a work-in-progress project, and the results will be evaluated in the future. Currently, we do not focus on ontology mapping issues. Our aim is to verify the web service interface and the method of exposing user-specific personal data.

There are three participating information systems in our scenario. The first system is an adaptive learning system containing learning materials and simple test questions at the end of each lesson. The second system is intended for selection of a programming project topic and for submission of the completed works. The third system performs an automatic evaluation of several programming tasks assigned to students in the course of a year. The adaptation features of the learning system can benefit to a considerable extend by additional information about students's progress on solving the assigned tasks and their achieved results. The adaptive guidance within the learning course can be also really well tailored based on the student's project topic, and the subtopics related to his or her project.

In the current setup, see Fig. 3, each of the systems is equipped with the RESTful data interface, and all systems are interconnected through a central mediation service. Each of the systems conforms to a similar domain, and their domain models overlap with some mutually related concepts.

The User Model Attributes exchange is subject to a certain level of mutual understanding of the domain semantics by the communicating counterparts. In our experimental scenario, for the ease of personal-data interface demonstration, all three systems use the domain ontology of the adaptive learning system. The data structure defined by the ontology is mapped locally. Furthermore, the RESTful API of the learning systems is extended by the domain resources and systems can negotiate the correct domain instance references. All domain instances are represented as RDFS-resources identified by a unique Uniform Resource Identifier (URI). The URIs are used as a link between the overlay User Model and the domain layer of the application.

The User Profile Attributes are not domain-

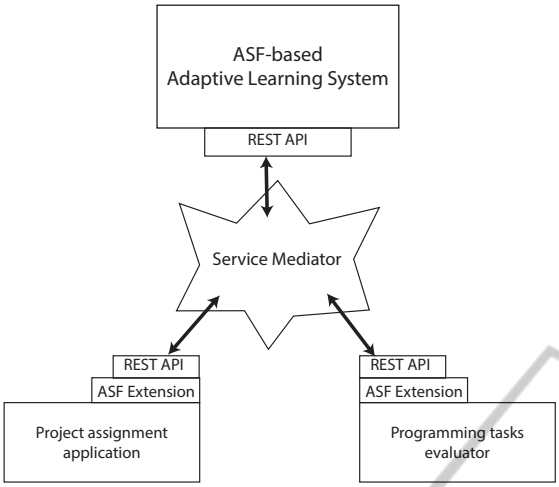

Figure 3: Applications participating in the use-case integration scenario.

dependent. The meaning of the included preferences needs to be well represented using common and widely used ontologies. In later implementations, public services like Facebook, Twitter or LinkedIn can be used as user's personal information providers, and therefore, it is important to be able to match the preferences adequately.

5 CONCLUSION

The paper has proposed our novel solution to deal with the syntactic and semantic heterogeneity of personal information in adaptive hypermedia systems. To make the integration of users' preferences and characteristics possible, a special architecture of User Profile and User Model was defined, and a RESTful web service application interface was introduced.

Compared to other solutions presented in Section 2, our proposal combines both shared format and conversion approaches resulting in a hybrid solution and utilizing advantages of both approaches.

The strength of our approach is the generic architecture, and the fact that realization can be supported by the Adaptive System Framework. Even non-adaptive systems can be extended by the ASF-based integration module and operate as providers of users' personal data. An integration use-case of that kind was presented using the case of multiple learning-support applications. As a result, the adaptive learning application can benefit from the integration and promptly react on achievements of students recorded by related learning-support systems.

In our future work, we will thoroughly evaluate the integration impact on a deeper understanding of user's knowledge, preferences and needs.

The special user modeling data storage and the introduction of the web-service-based application interface of a user-adaptive application is one of the important steps towards the generic AHS architecture formalization.

ACKNOWLEDGEMENTS

The results of our research form a part of the scientific work of a special research group WEBING.[1]

REFERENCES

Dublin Core Metadata Initiative. http://dublincore.org/. [Accessed: August 7, 2013].

SIOC Project. http://sioc-project.org/. [Accessed: August 7, 2013].

Adamczyk, P., Smith, P. H., Johnson, R. E., and Hafiz, M. (2011). REST and Web Services: In Theory and in Practice. In Wilde, E. and Pautasso, C., editors, *REST: From Research to Practice*, chapter 2, pages 35–57. Springer New York, New York, NY.

Aroyo, L., Dolog, P., Houben, G.-J., Kravcik, M., Naeve, A., Nilsson, M., and Wild, F. (2006). Interoperability in personalized adaptive learning. *Educational Technology & Society*, 9(2):4–18.

Aroyo, L. and Houben, G.-J. (2010). User modeling and adaptive Semantic Web. *Semantic Web*, 1(1–2):105–110.

Balík, M. and Jelínek, I. (2008). Towards Semantic Web-based Adaptive Hypermedia Model. In *ESWC Ph.D. Symposium*, pages 1–5, Tenerife, Spain.

Balík, M. and Jelínek, I. (2013). Adaptive System Framework: A Way to a Simple Development of Adaptive Hypermedia Systems. In *The Fifth International Conference on Adaptive and Self-Adaptive Systems and Applications (ADAPTIVE2013)*, pages 20–25, Valencia, Spain. IARIA.

Berkovsky, S., Heckmann, D., and Kuflik, T. (2009). Addressing challenges of ubiquitous user modeling: Between mediation and semantic integration. In *Advances in Ubiquitous User Modelling*, pages 1–19.

Berkovsky, S., Kuflik, T., and Ricci, F. (2007). Mediation of user models for enhanced personalization in recommender systems. *User Modeling and User-Adapted Interaction*, 18(3):245–286.

Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data-the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22.

Brickley, D. and Miller, L. (2010). FOAF vocabulary specification 0.98. http://xmlns.com/foaf/0.1. [Accessed: August 7, 2013].

[1]Webing research group – http://webing.felk.cvut.cz.

Brusilovsky, P. (2001). Adaptive Hypermedia. pages 87–110.

Carmagnola, F. (2009). Handling semantic heterogeneity in interoperable distributed user models. In *Advances in Ubiquitous User Modelling*, pages 20–36.

Cena, F. and Furnari, R. (2009). A model for feature-based user model interoperability on the web. In *Advances in Ubiquitous User Modelling*, pages 37–54.

de Bruijn, J., Martín-Recuerda, F., Ehrig, M., Polleres, A., and Predoiu, L. (2005). Ontology Mediation Management v1. Deliverable D4.4.1. Technical report, SEKT.

Fielding, R. T. and Taylor, R. N. (2002). Principled design of the modern Web architecture. *ACM Transactions on Internet Technology*, 2(2):115–150.

Fredrich, T. (2012). RESTful Service Best Practices.

Gross, R. and Acquisti, A. (2005). Information revelation and privacy in online social networks. In *ACM Workshop on Privacy in the Electronic Society (WPES)*.

Guinard, D., Ion, I., and Mayer, S. (2011). In Search of an Internet of Things Service Architecture: REST or WS-*? A Developers' Perspective. In Puiatti, A. and Gu, T., editors, *Mobile and Ubiquitous Systems: Computing, Networking, and Services (MobiQuitous 2011)*, pages 326–337, Copenhagen, Denmark. Springer.

Heckmann, D., Schwartz, T., Brandherm, B., and Schmitz, M. (2005). GUMO The General User Model Ontology. pages 428–432.

Jovanovic, J., Gasevic, D., and Siadaty, M. (2012). IntelLEO User Model Ontology. http://intelleo.eu/ontologies/user-model/spec/. [Accessed: August 7, 2013].

Kay, J., Kummerfeld, B., and Lauder, P. (2002). Personis: A server for user models. In De Bra, P., Brusilovsky, P., and Conejo, R., editors, *Adaptive Hypermedia and Adaptive Web-Based Systems (AH2002)*, pages 203–212, Malaga, Spain. Springer.

Knutov, E., De Bra, P., and Pechenizkiy, M. (2009). AH 12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. *New Review of Hypermedia and Multimedia*, 15(1):5–38.

Kobsa, A. and Teltzrow, M. (2006). Convincing Users to Disclose Personal Data. *Privacy-Enhanced Personalization*, pages 19–21.

Martinez-Villaseñor, M. D. L., Gonzalez-Mendoza, M., and Hernandez-Gress, N. (2012). Towards a ubiquitous user model for profile sharing and reuse. *Sensors (Basel, Switzerland)*, 12(10):13249–83.

Pautasso, C. and Wilde, E. (2009). Why is the web loosely coupled?: a multi-faceted metric for service design. In *Proc. of the 18th international conference on World Wide Web (WWW '09)*, pages 911–920, Madrid, Spain. ACM.

Pautasso, C., Zimmermann, O., and Leymann, F. (2008). RESTful web services vs. "big" web services: making the right architectural decision. In *WWW 08: Proceeding of the 17th international conference on World Wide Web*, pages 805–814, New York, NY, USA. ACM.

Salton, G. and McGill, M. J. (1983). *Introduction to modern information retrieval*. McGraw Hill Book Co., New York.

Sosnovsky, S., Brusilovsky, P., Yudelson, M., Mitrovic, A., Mathews, M., and Kumar, A. (2009). Semantic Integration of Adaptive Educational Systems. In Kuflik, T., Berkovsky, S., Carmagnola, F., Heckmann, D., and Krüger, A., editors, *Advances in Ubiquitous User Modelling*, pages 134–158. Springer.

Sosnovsky, S. and Dicheva, D. (2010). Ontological technologies for user modelling. *International Journal of Metadata, Semantics and Ontologies*, 5(1):32–71.

van der Sluijs, K. and Houben, G. (2005). Towards a generic user model component. In *Proc. of PerSWeb05, Workshop on Personalisation on the Semantic Web*, pages 47–56, Edinburgh, UK.

Vassileva, J., McCalla, G., and Greer, J. (2003). Multi-agent multi-user modeling in I-Help. *User Modelling and User Adapted Interaction*, 13(1-2):179–210.