TOWARDS AN ARCHITECTURE FOR PERSONALIZATION AND ADAPTIVITY IN THE SEMANTIC WEB

Veruska R. Aragão, Alvaro A. A. Fernandes, Carole A. Goble

Department of Computer Science
University of Manchester
Manchester M13 9PL, UK
{v.aragao, a.fernandes, c.goble}@cs.man.ac.uk

Abstract: *A semantic web is expected to underpin richer, and more automated, forms of interaction than web services and data allow today. However unlike syntactic forms, interpretations cannot be standardized across all potential users, or even be kept constant over time for a specific user. While a semantic web allows users to go beyond the syntax of resources by appealing to meta-resources and, more expressively, ontologies, it also raises the issue as to how are users to specify their own interpretations of web components and to avoid being overburdened with the need to keep those interpretations afresh. This paper argues that personalization and adaptivity are key technologies in addressing this particular issue for a semantic web. Without a proper solution the semantic web will be less usable than the vision would suggest it to be. The paper motivates the need for such technologies via an e-commerce use case and sketches an architecture that can be used to underpin their realization as concrete software components.*

1. Introduction

The web has developed into a complex of technologies that, taken together, serve a variety of purposes of great interest to both companies and individuals, notably researching information, and purchasing and selling goods. In most cases, activities such as these are goal directed and can be broken down into tasks whose execution is essential for either the effectiveness or the efficiency (or both) with which the goal is achieved. However, it has become clear both that users are sometimes hindered in the pursuit of their goal-directed activities by the absence of certain features in the web, and that this absence is inherent to the web as it stands. As a consequence, there is a mismatch between the view users have of their tasks and the ability of current web technologies to support that view. Thus, it is seldom the case that tasks can be performed in ways that approximate what users want. In many cases this mismatch can be traced back to the fact that services and data in the web are not describable by any formal semantics. This leads to situations in which, e.g., in searching, no context can be declared with which the returned results might be framed or filtered.

The idea of a semantic web has been proposed to make up for exactly these shortcomings in current web technologies. A semantic web is designed, among other things, to encompass from XML-expressed resources, through RDF-described services and data, to ontology-based reasoning capabilities [2, 8]. This view of a semantic web as comprising **resources**, **descriptions of resources**, and **ontologies** that express the concepts underpinning both resources and their descriptions is adopted throughout this paper. By making explicit the semantics of resources, the semantic web is expected to allow both richer and more automated forms of interaction. As an example of the former one might cite context-sensitive search with a consequent increase in the effectiveness of the task for users. An example of the latter might be the deployment of software agents acting as proxies for users to mediate interactions. However, even assuming that web services and data are endowed with a formal semantics, mismatches with individual users are bound to occur because semantic relationships are mappings from specifications to interpretations, and while the former can be (and already are) standardized, the latter cannot be assumed to be universally shared. By definition, interpretation is always relative to an interpreter (in the context of this paper, an individual user, who may be a person, a software agent or even an entire organization if the latter fixes an interpretation for all the individuals it has discretion over).

This paper expresses a position regarding how an architecture for user interaction can deliver user-initiated ways of personalizing interaction with the semantic web. One of the uses for the capabilities made possible by such an architecture could be to allow individuals to enforce their interpretation of semantic web components. The remainder of the paper is structured as follows. Section 2 provides some motivation for personalization actions using an e-procurement scenario. Section 3 sketches an architecture for a system that supports personalized, adaptive interaction. Section 4 briefly discusses related work. Section 5 presents some conclusions.

2. Motivation: E-Procurement in a Semantic Web

At the moment, there is no widely-available mechanism to allow users to personalize their interaction with web data and services. Much is being done by servers to adapt their provision to individual clients, but server-side personalization has little hope of being flexible and expressible enough at the grain of individual users. Client-side personalization is needed in order to allow users to express their preferences and have them taken into account in interactions with many organizations in a consistent manner. Even if server-side personalization were to deliver optimal results, each individual user would still have to negotiate the many different models that different servers, one by one, have constructed of each such user. In contrast, client-side personalization allows individual users to tailor once and for all, consistently and in tandem, their interaction with many servers.

This section employs a use case in e-procurement to show that, given user goals and some of the tasks that must be performed to achieve them, semantic web components allow the tasks to be executed in more desirable ways than are possible in the web as it stands. Then, it is shown that, given a semantic web, users may still benefit from personalization actions whose effect is to claim ownership of semantic web components by versioning their specification, insofar as this makes it possible for them to perform their tasks in the way they want, as opposed to the way prescribed by the standard interpretation associated with semantic web components by their respective providers.

Case Study: Goals and Tasks in E-Procurement The procurement task involves inspecting shelf stocks as well as back-room inventory levels, finding out whether, and in what quantities, supplies are needed, and, finally, ordering the required supplies from organizations capable of meeting the appropriate standard of quality and the desired delivery date [12]. In the example below, it is assumed that the procurement system is deployed over the web, and by e-procurement is meant the process underpinned by such a system [3]. Suppose that a large store is part of a chain with similar stores in many countries. Each store buys products to replenish their stock according to guidelines and policies set, in a centralized way, by headquarters. The store manager of the store needs to buy products to be delivered a few weeks before New Year's Eve. To achieve this procurement goal, the manager must determine stock levels, decide on the brand composition, search for products that are available for delivery in the specified period, and, finally, inform suppliers of the desired quantity of each product. Table *2.1* shows the relationship between this broad goal and some of the more prominent coarse-grained tasks involved in achieving it. Note that Table *2.1* is kept deliberately simple, e.g., other tasks may be involved in the pursuit of the stated goal, and tasks listed in the table may have to be undertaken in the pursuit of other goals than the stated one.

	GOAL	Buy products to be delivered for New Year's Eve
T	Determining the required quantity for each product	√
A	Stipulating brand composition	√
S	Searching for suppliers of the products	√
K	Checking availability	√
S	Ordering available products for delivery at the right time	√

Table 2.1: Some E-Procurement Goals and Tasks

The Added Value of a Semantic Web It is quite possible to achieve the goal in Table *2.1* by performing the required tasks using web services and data provided by potential suppliers. However, the lack of richer information about the services and data provided limits the scope of what the store manager can hope to achieve. If metadata were made available by the suppliers, a store manager would be better informed about whether a supplier can cater for specific needs. If, moreover, the store had access to ontologies used by suppliers to define and constrain their services and data, then the store manager might hope to avoid misunderstandings and misconceptions that impede the goal. If, in the future, most such interactions are to be delegated to software agents, this lack of information would make it impossible for interesting forms of interaction, delegation and mediation to be conducted in an automated manner. The semantic web comprises components whose role is to underpin such enriched forms of interaction.

For example, stipulating brand composition may require the concept to be defined in an ontology as a constraint that specializes the notion of order to the notion of well-proportioned order, i.e., one in which the targets for brand composition are met. In searching for products and suppliers, store managers might find themselves wasting time (or opportunities slip by) if the data is expressed in a language which they are not fully fluent in. However, if content (or tags, or both) were translated to local languages, stores in different countries would find it easier to consider suppliers in foreign countries. If they then relay their choices of potential foreign suppliers to headquarters, they give the latter an opportunity to increase the bulk of the order by collecting choices that might not otherwise have been made because of language barriers. Of course, with a bulkier purchase in hand, headquarters might be able to negotiate better terms and conditions. Given that XML (`http://www.w3.org/XML/`) tags annotate and structure content with server-side information, a client-side translation software, might be able to handle this sub-task well enough to underpin the strategy above. Suppose now that the country a store is located in (and all bespoke software that has been developed locally) uses the metric system. Suppose also that a potential foreign supplier expresses units using the imperial system. If so, then checking availability presupposes unit conversions. The measurement system in which units are expressed is metadata that RDF-based descriptions might capture at the supplier and at the stores thereby allowing conversions to be invoked appropriately. The observable effect, again, is a larger set of potential suppliers across all stores and this, in turn, may allow headquarters to negotiate better terms and conditions. In setting deadlines for delivery it may be important to define different holidays in different countries. Although this might be defined by an ontology whose scope is global, different stores might commemorate holidays that are local or contextual (e.g., the Chinese New Year in cities, like Manchester, where a large Chinese community has established itself). The above examples illustrate the fact that a semantic web would, in principle, allow the carrying out of tasks that are more ambitious in terms of the amount of knowledge deployed than is possible in the web as it stands. Table *2.2* associates with some of the tasks listed in Table *2.1* sub-tasks involving semantic web components that would allow the task to be performed more efficiently, or effectively, or both.

SEMANTIC WEB COMPONENTS		**Resource**	**Resource Description**	**Ontology**
T	Stipulating brand composition			Define the concept of well-proportioned order
A	Searching for suppliers of the products	Translate to local language		
S	Checking availability	Insert annotation regarding units used	Define conversion functions between measurement units	Define a measurement system
K	Ordering products that are available for delivery at the right time	Modify suppliers' web pages to show just data about a desired period	Mark suppliers that allow special conditions of payment	

Table 2.2: Semantic Web Components and E-Procurement Tasks

Each cell in Table *2.2* indicates that for the task represented in that row, the semantic web component represented in the column might be used in the sub-task described in the cell. For example, in stipulating brand composition, the ontology component of a semantic web might be used to define the concept of well-proportioned order. Sometimes, more than one semantic web component may play role. For example, in checking availability the resource might be rewritten to insert annotation regarding units used, the resource description component might be rewritten to define conversion functions between measurement units, and the ontology component might be used to might be rewritten so as to define a measurement system. Again, note that Table *2.2* is kept deliberately simple. It might have been possible to list another sub-task (or more sub-tasks) in each cell. Also, the sub-task that is listed in a cell as making use of one semantic web component might make use of another component (or more than one component).

The Added Value of Personalization One important observation that applies to all the cells in Table *2.2* is that the sub-tasks listed in each cell require the client-side applications to tailor to their own needs the server-supplied semantic web components. While not all sub-tasks for which a semantic web might prove useful will require tailoring of a semantic web component, Table *2.2* suggests that many may. The question then arises as to how can tailoring take place. As discussed in Section 2, while server-side personalization may go a long way towards providing a client with resources, resource descriptions and ontologies that are more likely to suit that client, then can do only on the granularity of groups, or communities, by associating a type to each client. Client-side personalization is needed for finer-grained tailoring. If so, it becomes crucial to consider how to design a solution that delivers such functionality. We suggest that *ownership transfer* – a particular approach to personalization stemming from research into adaptive hypermedia [10] – is a suitable candidate for delivering the kind of client-side personalization required in the scenarios of Tables *2.1* and *2.2*.

The notion of ownership transfer emerged from studies into the use of adaptive hypermedia for learning tasks. The idea proposed as a means of expressing the problem faced by users when trying to learn from hypermedia materials. Unlike traditional materials, e.g., a book, a user does not own a hypermedia course in the sense of being able to annotate it, decide on different routes through it that were never designed as such, and so on. One of the goals of adaptive hypermedia was then to allow ownership transfer to take place with benefits that accrue from it in terms of learning opportunities. In the context of this paper, the sub-tasks in the cells in Table *2.2* imply the need for personalization. If one rules out server-side personalization as too coarse grained, then the ability to transfer the ownership of semantic web component from their provider to each of their users, underpins the kind of fine grained personalization that is required. In this paper, ownership transfer is the outcome of personalization actions taken by the user. More concretely, personalization actions cause the versioning of the specifications of semantic web components so that, once a user has carried out a personalization action, what that user sees is the effect of the specifications have been versioned. So, personalizing resources, resource descriptions and ontologies by insertion, update and deletion causes the applications that use them to behave in a personalized way for the user that carried out the personalization actions from which the version stems. In other words, the range of effects that can be achieved for the benefit of users is determined by the range of observable outcomes that can be obtained by changes in the specification of the semantic web components upon which all interaction is founded. For example, each cell in Table *2.3* shows a personalization action on a semantic web component that might have the effect required by each sub-task in each cell in Table *2.2*.

SUB-TASKS	**SEMANTIC WEB COMPONENTS**		
	Resources	**Resource Descriptions**	**Ontologies**
Translate to local language	Apply translation software to XML source		
Define the concept of well-proportioned order			Update the ontology where the concept *product order* is defined
Insert annotation regarding units used	Add *unit* tags to XML source		
Define conversion functions between measurement units		Add pointer in the RDF to unit conversion function	
Define a measurement system			Generate an ontology with the concept *measurement* as root
Modify the suppliers' web page to show just information related with the desired period	Remove from XML source items with delivery date outside the desired period		
Mark suppliers that allow special conditions of payment		Mark the RDF information on suppliers with ranking data about terms and conditions	

Table 2.3: Personalization Actions to Enact Sub-Tasks in Table *2.2*

The kind of personalization actions envisaged here assumes a semantic web to comprise XML to specify resources, RDF [9] and RDFS[4] to specify resource descriptions and DAML+OIL (`http://www.daml.org/2001/03/daml+oil-index.html`) to specify ontologies. XML provides the basic syntax for semantic web components but in itself, it captures no semantics. RDF and RDFS ameliorate this problem. Although they are more expressive than XML they still lack expressive power, e.g., equivalence and disjointness of classes cannot be captured by RDFS. DAML+OIL is a full-blown description logic and sophisticated reasoning services can be developed over it, thereby solving some of these problems. By choosing these languages, our goal is to do justice to the different levels of expressiveness that an application may require. Thus, by resources are meant XML files, by resource descriptions are meant RDF files and by ontologies are meant DAML+OIL files. Personalization actions, in this case, reduce to operations on such files, as Table *2.3* suggests. It is this requirement to operate at the client on server-supplied files that necessitates versioning. Versioning is the outcome of personalization. Personalization causes ownership to be transferred. Ownership transfer allows users to enforce their interpretation of server-originated semantics.

3. An Architecture for Personalization and Adaptivity

This section sketches an architecture to support personalization and adaptivity of semantic web components. The assumption underlying this section is that in a semantic web, users will browse it, as they do in the web as it stands today. However, the functionality denoted by the verb *browse* need not be understood in a restrictive manner. As in the web today, front-ends to different protocols provide the comprehensive functionality available to users. In the remainder of the paper, the same is assumed for a semantic web. Thus, the verb *browse* stands as an example of the different protocols that a semantic web might make available. On a related issue, the word *user* is meant to denote not just people but software components too. The architecture proposed here aims to be minimally intrusive and, whenever possible, the set of new component comprising an extension that deliver additional functionality is layered (or loosely coupled) to other set of components delivering other functionalities.

Interacting with a Semantic Web The basic components deliver the kind of interaction with a semantic web that is delivered by the web as it is. The simplified architecture is depicted in Figure *3.1*. where arrows denote data flows and round-cornered boxes denote functional components, squares denote external entities (e.g., users, or other software components), and the semantic web components are denoted by a circle. In subsequent figures, open-sided rectangles denote stores. For ease of explanation, Figure *3.1* depicts a community of users $u_i, 1 \leq i \leq 21$ (where 21 is an arbitrary number with no special significance). In subsequent figures, explanations hinge on actions by user u_{21}. The intended interpretation of Figure *3.1* is that a community of users sends request to functional components (e.g., a browser). These in turn use different protocols to interpret the specifications embodied in generic (i.e., server-supplied, and hence not user-specific) components coming from a semantic web, and compose (and render visible) responses to user requests. Note that in Figure *3.1*, the fact that semantic web components are generic over all users is denoted by their not being specifically subscripted.
The availability of the specification of the semantic web components allows more powerful interactions with the web than those that can be provided by the web as it is. To see the contrast, consider that, for example, the task of ordering products available for delivery at a specific period in Table *2.1* can be modelled by applications over the web as it is. However, with the availability of semantic web component, as in Figure *3.1*, users could aspire to execute that tasks in a more effective way by, for example, modifying the specification associated with the resource, in this case the suppliers' web page, to show just information related with the desired period as suggested in Table *2.2*. Thus, Figure *3.1* represents an advance vis-à-vis the web as it is (in that it allows a shift from the tasks in Table *2.1* to those in Table *2.2*), but it provides no support for the personalization actions needed to realize them concretely. This requires extensions to the architecture in Figure *3.1* that are described below. The main elements of the complete architecture are: a *personalization component*, which allows users to modify specifications of semantic web components, performing personalization actions, in order to enforce their own preferences; a *learning component*, which recognizes patterns in personalization

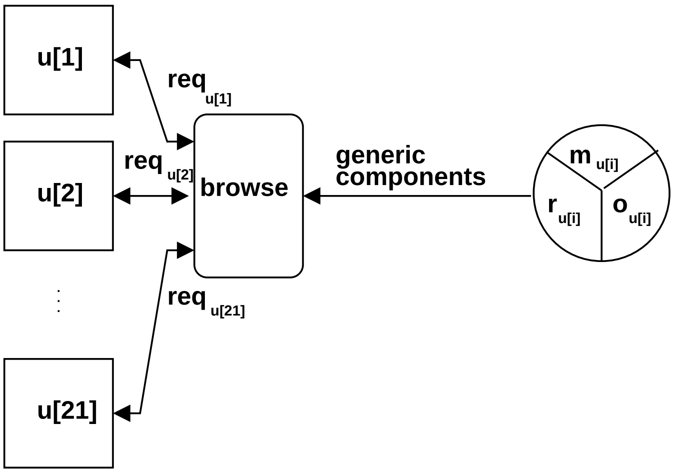

Figure 3.1: Interacting with a Semantic Web

actions and uses them to build a formal representation of a user; and finally, an *adaptation component* that defines how the behaviour patterns derived for a user can justify the system's carrying out personalization actions on her or his behalf. The next subsections describe the components above.

Personalizing Interaction The personalization component allows users to transfer ownership of the semantic web components by versioning them and modifying their specifications explicitly, and thereby enforce user preferences, by carrying out personalization actions. Users can enforce their preferences, even overriding the information provider's intended meaning. However, the original component is always kept so that it is possible to compare it with its current state at the server side. Thus, changes made by the information provider subsequently to a user having personalized it at the client side can always be discovered, and the users made aware of it. The extended architecture is depicted in Figure *3.2*.

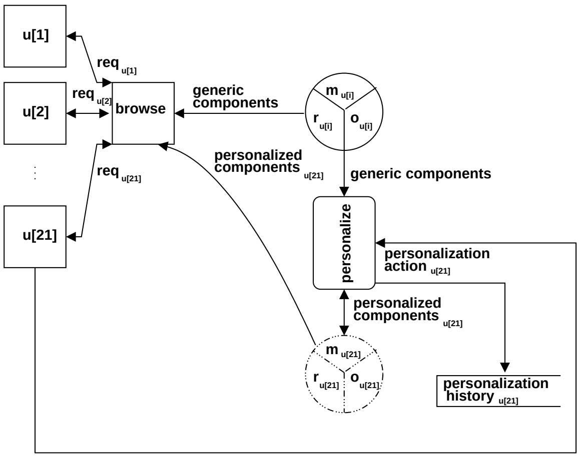

Figure 3.2: An Architecture for Personalization

Note that the generic components are versioned into personalized components as a result of personalization actions being requested of the functional component denoted by *personalize*. Of course, personalized components may be changed incrementally too, by successive personalization actions. The main import of this versioning process is that for a user (e.g., u_{21} in Figure *3.2*) who has requested personalization actions to be carried out, browsing reflects this fact, i.e., for u_{21}, requests are answered using the personalized component rather than the generic ones employed for all users who have not requested personalization actions to be carried out. By being reflected in browsing outcomes, the preferences expressed as personalization actions, the interaction process as a whole gradually becomes specific to a user, say, in the example, u_{21}.

Suppose that u_{21} is the supply manager of one of the retailer stores, then, once the set of personalization actions is defined, the user could modify supplier web pages, and delete from the XML source those items whose delivery date lies outside the desired period. Those actions are then stored in the personalization history. So, carrying out personalization actions as exemplified in Table *2.3* users can aspire to the more complex functionalities suggested in Table *2.2* In order to characterize which personalization actions a user might take, a task-based model of user interaction with the semantic web components needs to be defined. Such a model could then be used to relate user interactions and specification languages used in a semantic web. This relationship can be used to determine a well-defined set of actions that, when carried out, alter specifications in the way required to produce the required effect in terms of user interactions. Figure *3.2* presupposes that the *browse* function can be adapted into a just-in-time rendering mechanism which delays the evaluation of specifications into user-visible outputs in a similar way as already implemented in open-hypermedia systems like COHSE (`http://www.cohse.semanticweb.org/`). Personalization actions are stored and, over time, comprise a history of usage for a particular user. This history is a syntactic representation that can then be used to induce a usage model of personalization actions.

Learning about User Preferences Given a personalization history, an opportunity arises to identify the conditions under which the user has made personalization requests. Using machine learning techniques, a learning component might try to identify usage patterns in a personalization history. Identified patterns allow the system to postulate that a certain personalization action might be beneficial for the user concerned. In the example, the personalization component stores, in the personalization history, among others, the delete actions with which the supply manager eliminates items that cannot be delivered in the desired period. The learning component uses that personalization history as a training set to infer that the user wants items to be disregarded if they cannot be delivered within a certain period. This pattern can be stored and a collection of those builds up into a usage model for a particular user. The learning component therefore infers a usage model in the form of a set of conditions under which certain personalization actions are hypothesized as being beneficial. This is depicted in Figure *3.3*, which extends Figure *3.2*.
The usage model over resources, resource descriptions and ontologies can be said to define a user context. Each usage model is a formal representation of how the user has decided to act in the past. Notice that, across many users (e.g., in a group), there can be no simple assumption that the context is shared, but it may be legitimate to assume that the usage model is not of individuals but rather of an agent (human or otherwise) performing a role (e.g., supplies manager) that is bounded and constrained by policies (e.g., the business rules that characterize the chain of stores discussed above). So if groups have an identity that transcends their membership, and this identity is embodied in one distinguished user, the derivation of usage models for groups can become legitimate. Note that the usage model is explicit so, users can choose modify it, thereby indicating, whether they agree with the model induced by the learning component or not.

Adapting to Usage Models A final extension to the architecture in Figure *3.4* delivers adaptive functionality. An adaptation component can be envisaged that instantiates an active rule engine, i.e., an inference engine that executes event-condition-action rules.

In this setting, the induced usage model characterizes event patterns that trigger rules and rule con-

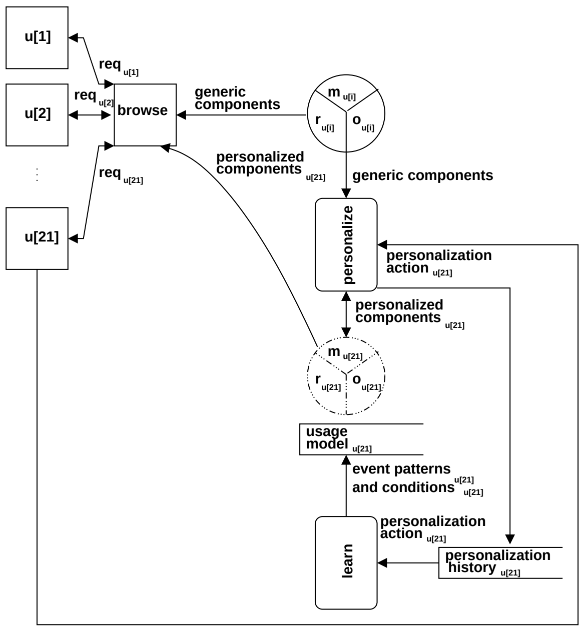

Figure 3.3: An Architecture for Inducing Usage Models

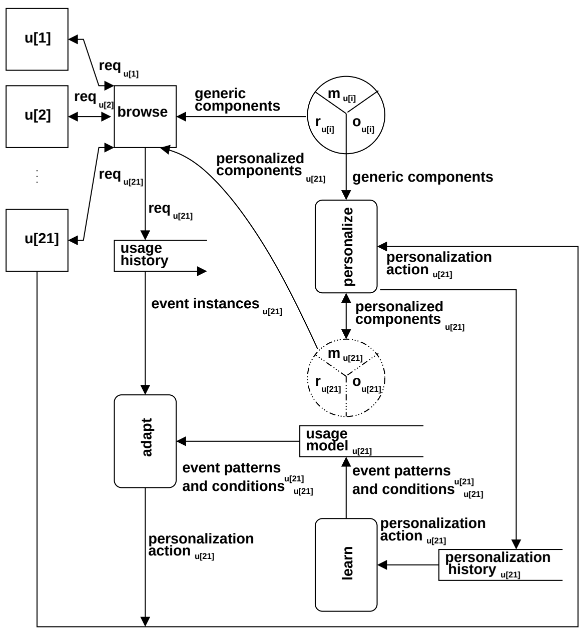

Figure 3.4: An Architecture for Adaptivity

ditions that ascertain whether the conditions obtain for actions associated with the rule. Events stem from the use of the system and are checked for detection against event patterns. In the example, after several interactions with supplier web pages and after the learning component has inferred that the supply manager wants to disregard items that cannot be delivered in a specific period, then if the manager browses a supplier web page, the adaptive component will either suggest the deletion or, if authorized, take action to delete items outside the period desired. The idea, therefore, is that when a user interacts with a semantic web component and this interaction is seen as an event which might instantiate a pattern, for which an action to personalize semantic web components may be deemed appropriate in order to enforce the user's preferences. The adaptation component enhances usability insofar it can proactively suggest possible personalization actions that a user might find useful to take. The more the user interacts with the system, the more the latter learns about usage patterns. The better the usage patterns, the better the recommendations that the adaptive component might make.

4. Related work

Research on personalization and adaptation technologies has been developed in many areas such as adaptive interface, adaptive hypertext and hypermedia, intelligent information retrieval, web mining, mobile technologies. They propose several frameworks and prototypes to personalized content and environment for users or group of users. [5] identifies six types of adaptive hypermedia systems using the criteria of application areas. Among them, *systems for managing personalized views in information spaces* are relevant for our work. For those systems, the web constitutes an immense hyperspace with a huge amount of information and on-line services, however users need to access just a subset of this hyperspace. Barsa [11], one of those systems, limits the hyperspace focus in search and filter using a usage profile which is build by agents that acquire knowledge about specific users' preferences, interests and tasks. The profile is created by explicitly asking users about tasks and implicitly observing how they perform tasks. In a different way, our framework keeps track of the user interaction with the system storing the personalization actions the user performed, building a personalization history and from these history learning patterns of usage and carrying out adaptations that maybe beneficial to the user. [6] proposes an approach to personalize data objects using metadata to support a much finer grained individualized description of them. In this approach to enable personalization of web application the metadata of the object is versioned and stored at the server-side. In our approach, metadata are provided to describe resources of the *semantic web* and users can modify it by transferring ownership of them through versioning. Different from the previous approach, the transfer of ownership causes the component personalized to be stored in the client-side. This client-side personalization allows users' preferences to be considered consistently while interacting with many organizations. [7] develops a framework of requirements for designing customizable web applications and includes some approaches that implement customization. This framework comprises context and adaption dimensions. Following, we fit our approach in those dimensions to stress the aspects our work comprises. Our approach is conducted by the *Dynamic User Context* dimension, gathering information when the user performs personalization actions, then the storage of the history of personalization actions performed comprises the dimension of *History Factor*. Using the history to infer and to perform some actions that may be beneficial to users includes the dimensions of *Adapting Dynamically* and *Mapping between Context and Adaption*. The systems cited here interact with the Web and users are passive in the sense that they cannot change aspects of the system.

5. Conclusions

This paper has sought to express the position that, useful as it may be, a semantic web does not do away with the need for user-centred interaction. In fact, because assigning semantics is an interpretation task that cannot be standardized, the need for techniques, such as personalization and adaptivity, that allow users to tailor web services and data to their own needs is likely to be more acutely felt. The paper has motivated this need by looking in some detail at circumstances arising in e-procurement where a semantic web improves the richness of interaction but does not remove the need for tailoring. Finally, the paper has sketched an architecture that suggests how personalization and adaptivity facilities can be integrated with browser-mediated interactions. Thus, the paper has contributed a position regarding complementary technologies that are likely to be indispensable in a semantic web. While much remains to be worked out in detail, the paper contributes an architecture with which some of these technologies can be implemented. The authors are now engaged in implementing the architecture so as to validate empirically the benefits stemming from the position expressed here.

References

[1] BERNERS-LEE, T. *Weaving the Web*. Orion Business Books, London, 1999.

[2] BERNERS-LEE, T., HENDLER, J., AND LASSILA, O. The semantic web. *Scientific American* (May 2001).

[3] BOTTO, F. *Dictionary of E-business*. John Wiley & Sons Pub., Jul 2000.

[4] BRICKLEY, D., AND R.V.GUHA. Resource description framework (rdf) schema specification - candidate recommendation. W3C, 2000. `http://www.w3.org/TR/rdf-schema/`.

[5] BRUSILOVSKY, P. Methods and techniques of adaptive hypermedia. *Adaptive Hypertext and Hypermedia 6* (1996), 87–129.

[6] HICKS, D., TOCHTERMANN, K., ROSE, T., AND EICH, S. Using meta-data to support customization. In *Proc. 3rd IEEE Meta-Data Conference* (April 1999).

[7] KAPPEL, G., RETSCHITZEGGER, W., AND SCHWINGER, W. Modeling customizable web applications. In *Kyoto Intl. Conf. Digital Libraries* (November 2000),

[8] KLEIN, M. XML, RDF, and Relatives. *IEEE Intelligent Systems* (March/April 2001), 26–28.

[9] LASSILA, O., AND R.SWICK. Resource Description Framework (RDF) Model and Syntax Specification. W3C, 1999. `http://www.w3.org/TR/REC-rdf-syntax`.

[10] OHENE-DJAN, J., AND FERNANDES, A. A. Modelling personalizable hyperlink-based interaction. In *2nd. Workshop Adaptive Hypertext and Hypermedia* (June 1998).

[11] THOMAS, C. B. Basar: A framework for integrating agents in the world wide web. *IEEE Computer 28*, 5 (May 1995), 84–86.

[12] WESTLAND, C., AND CLARK, T. *Global Electronic Commerce - Theory and Cases*. MIT Press, Feb 2000.