Context-aware Web Engineering: Modeling and Applications

J. Wolfgang Kaltz — Jürgen Ziegler — Steffen Lohmann[1]

IIIS
Universität Duisburg-Essen (Campus Duisburg)
47057 Duisburg, Germany

{kaltz,ziegler,lohmann}@interactivesystems.info

ABSTRACT. *This article presents an approach to Web Engineering which aims to account for context-awareness in a comprehensive and integrated fashion, thus enabling an enhanced adaptation of the application to the end-user. A conceptual model, permitting the combination of a domain ontology with context-relevant parameters and a degree of relevance, is presented. Subsequently, the use of such a model in a Web Engineering process is discussed, including appropriate modeling software, and requirements for a runtime system.*

KEYWORDS: *Web Engineering, context-awareness, adaptation, context model, domain ontology*

1. Funded by the German Federal Ministry of Education and Research (BMBF) under project nr. 01ISC30F

1. Introduction

The next generation of Web software holds the promise of "mass customization" (Rheingold, 2002). If we look at customization of software as its ability to fulfill an individual user's needs, we realize that the software must be aware of several factors: the user's profile, her current task or goal, and possibly additional factors such as location, time, or device used. The combination of all relevant factors can be termed the "context", and thus a Web software which takes them into account is a context-aware application.

Several approaches to integrating such context-awareness within software applications have been used in the past. Hypermedia applications have typically focused on the areas of learning and information retrieval, with the goal of guiding the user's navigation and presenting relevant information. An overview of this field of research, usually called adaptive hypermedia, is given in (Brusilovsky et al., 1998). In this field, the system adapts its offering according to the user model, where this user model is generally constructed from the user's behavior. Adaptivity in this sense takes into account user categories, sometimes also the types of tasks (Koch, 2001; Staff, 1997). Another recent field of development in Web applications with respect to context-sensitivity has focused on providing context-relevant information to mobile users, usually based on their location as context information. A typical scenario is a touristic visit in which the user is guided by a mobile device such as a PDA - see e.g. (Cheverst et al., 2000). (Chen and Kotz, 2000) provide a survey of context-aware mobile computing research. These applications use a context model which is specific to mobile scenarios. Research such as (Dix et al., 2000) does suggest that other types of context, like *domain context*, are also relevant to mobile applications, but does not explore this issue in detail.

We believe that context modeling is potentially useful in all types of applications, and that therefore a more general model, including mechanisms for context integration and usage, which would cover a wide variety of application domains, would be beneficial. Attempting to integrate context and adaptation capabilities in a comprehensive approach poses a number of interesting challenges. A conceptual model integrating a multitude of contextual factors is needed. This raises the question of how to structure and systemize the analysis and design processes. There is the risk of high complexity resulting from a large number of interactions between contextual factors. Generally speaking, modeling context is a difficult and time-consuming task, so an important question is how it can be made more effective, with the help of an appropriate base model and tools.

Our specific focus is the area of Web Engineering, which advocates a process and a systematic approach to development of Web-based systems (Murugesan et al., 2001). We will aim to provide for context modeling in a more general way than for mobile scenarios only, as we believe the benefits of context models can apply in general to Web applications, whether they are used by mobile devices or not. In fact, future Web applications should attempt to provide services for all sorts of devices, the device being just an additional contextual factor. The main focus of this article is to present an approach to modeling context on a broad base, integrated with other elements of the application domain. Furthermore, we will discuss how this model may be used within a Web Engineering process, and address design issues of a corresponding context system.

2. Related Work

The goal is to provide an environment for the development of Web software which takes into account context in a comprehensive and integrated fashion; to achieve this goal, one can distinguish two types of approaches.

Using a network approach, the network in itself would hold knowledge about which information and which services are relevant in which context. This network approach is explored e.g. by (Samulowitz et al., 2001), who focus on providing context-relevant information and services through a smarter network. The client, a mobile device, sends “Context-aware-packets” to the network. Within the network, nodes listen for such packets and respond to them, e.g. by presenting a relevant service to the client, which may then choose to activate it. The scenario presented is a mobile user attempting to discover a printer she may use, and has physical access to, from the present location. This type of approach has the disadvantage of requiring additional infrastructure for its realization and thus seems less suited for describing a general Web Engineering approach. It is, however, potentially better suited to the specific challenges of mobile computing (Forman and Zahorjan, 1994); it could thus be interesting to integrate such an approach for better support of mobile clients.

The other approach is data-centric, meaning that context data needs to be modeled completely at the application level, and made available to the Web application. Thus, relevant context data is modeled in the Web application environment, and within the Web application certain decisions will be based on this information. (Cannataro et al., 2002) is an example of such an approach; it defines XAHM, which has three conceptual adaptivity dimensions: user’s behavior, external environment, technology. A context in this model is a specific point within these three dimensions; although additional “dimension parameters” might be added. Another recent development is the COBRA-ONT architecture described in (Chen et al., 2004), which has the goal of supporting context-aware systems in smart spaces. This research, like ours, is also based on the observation that previous systems often lack a formalized model describing the contextual knowledge. The COBRA-ONT focuses on OWL (Bechhofer et al., 2004) as the key element for pervasive context-aware systems.

Our approach is also data-centric, and has similarities to XAHM in that it attempts to model context information in a general, integrated manner. A significant difference in our approach is the integration within a general Web Engineering activity, specifically with a domain ontology. Furthermore, our goal of service integration would allow for inclusion of context relevant services and/or information on a *discovery* basis, thus overcoming somewhat the limitation of the data-centric approach. We further agree with (Chen et al., 2004) that OWL is a good candidate for specifying an interchangeable format for storing contextual knowledge; however we attempt to place the focus on the definition and usage of a context model, whereas OWL to us represents a means of representation of this information.

From a modeling perspective, the approach of modeling terms in connection with a relevance value bears a resemblance to Bayesian networks, in which terms can be connected with probabilistic values. (Tipping, 2001) for instance has used Bayesian networks to associate relevance values, though for the purpose of learning algorithms. Such networks may in fact provide a basis for implementation for us; however, we believe that for a business user's perspective, our - more specific - modeling approach has a higher chance of being understood, and, therefore, of being used effectively in a Web Engineering process.

3. A Scenario for Context-aware Web Software

In this section, we present an example scenario for Web software for which we believe context-awareness to be a useful paradigm, and which will help to motivate the context categories introduced in the next section, additionally to typical context categories, such as *Location*, known from mobile computing research such as (Cheverst et al., 2000).

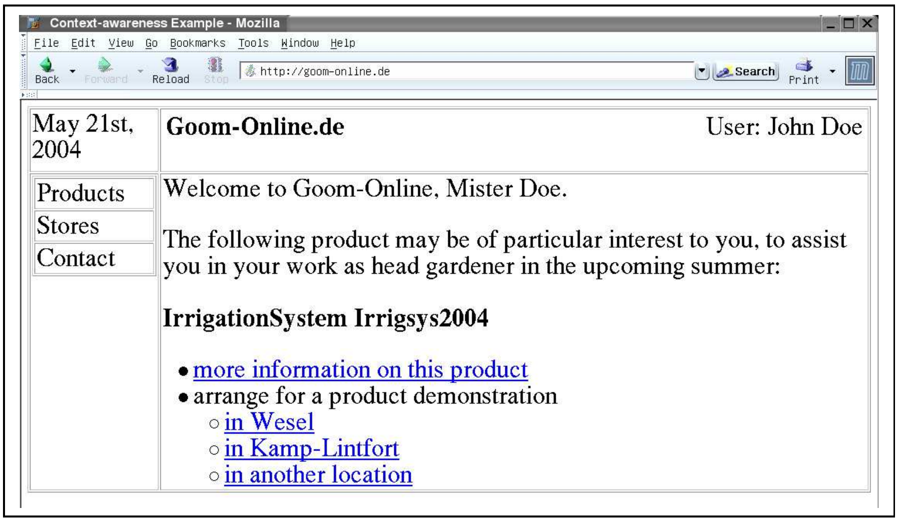

Figure 1. *A context-aware Web application*

Consider a construction market servicing various geographical areas (as an online store and/or as actual outlets). Figure 1 shows a scenario in which the user is known to the system and the user's profile is used in order to predict which types of offering may be of interest to this user. We call this context dimension the *User&Role*. Furthermore, the date is used in order to provide season-relevant information: this is the *Time* context dimension. The application shows the user a product which is potentially of interest, along with the ability to gain further information on this product, perhaps as a product description document which is attached. Furthermore, the application offers the user to arrange for a product demonstration in one of the store locations that the system assumes are near the user, whilst not precluding other possible locations.

Generally speaking, what is presented to the user is context-relevant pieces of information, but also further possibilities for interaction with the software: we term these *services*. Such a service is thus also offered in a context-sensitive manner, and is itself context-aware in that it may use context information as parameters. In the example scenario, a service is provided to enable the user to arrange for a product demonstration. To summarize: in a *context-aware* Web application, navigation, content selection and display type and services (as further going interaction items) may be *adapted* according to the context. The concepts introduced by this example scenario will be elaborated upon in the remainder of this article.

4. Context Modeling in Web Engineering

Recent studies have shown that a systematic engineering approach is often lacking in the development of Web software (Barry and Lang, 2001), leading to negative effects such as limited re-usability and difficult maintainability. The focus of current research - see, e.g., (Wissen and Ziegler, 2003) - is thus to provide a methodology and tools for systematic engineering of Web applications. Such efforts are often based on establishing an ontology, which represents the terms and relations of the domain relevant to a specific application.

In our modeling approach, the domain ontology plays a central role and may already have been developed in the overall engineering process. The goal is then to connect the domain ontology with context information. Context parameters are classified and connected with a certain weight to elements of the domain ontology, resulting in a *relevance space*. This approach is detailed in this section; the subsequent section will focus on how this model may be used by a Web application.

It should be noted that, although we shall aim at having as big a part as possible of the model generated by the system, we consider it indispensable for an expert user to be able to validate and extend the model. Therefore the model will need to be flexible with regard to representing context information, and yet remain comprehensible by an expert user. The goal of attaining this compromise will motivate the decisions presented here.

4.1. *A General View of Context*

Our general notion of context is that elements of the conceptual design space of an application, such as concepts of the domain ontology, can be contextualized through relations to elements of the context space. A certain information object may be, for example, particularly relevant for a specific geographic region or location. This relation is reciprocal: the information object is relevant if the current context is determined by the specific location; conversely, a specific location may be relevant if the current context focus is on a particular information object.

Context is thus determined by a set of contextual factors and

– a current perspective on these factors, or, as termed in (Brezillon, 2004), the *contextual knowledge*. This knowledge results from the context sensing and user behavior;

– an integrated ontology;

– a structuring of the contextual factors, resulting from the context modeling.

4.2. *Categorization of Context Elements*

The question of how to integrate context elements, or parameters, within an application model can be approached in two different ways. First, it is conceivable to allow for arbitrary definition of context parameters, and arbitrary combination with elements of the domain ontology. This would provide for maximum flexibility in the linking of context parameters and domain elements, and ensure that the model has a theoretically unlimited expressive power.

The second approach is to define a categorization of context parameters, and require specific values relevant to the application domain at hand to be assigned to one of these classes. This would seem to be a major restriction for the conceptual model, however, for the practical usage, we postulate that such a grouping of parameters is unavoidable, from the modeling perspective and from the usage perspective. Since we assume that some form of human interaction will be required to complete and adjust the model, the grouping of parameters is indispensable for the user to maintain an overview of the model. On the other hand, we assume that not all context elements will be linked with the domain ontology. This means that there will be a concept of "neighborhood", or more precisely, "distance" between context elements: certain elements will be relevant in a given context, even though they are not directly linked to the domain. In order to achieve a neighborhood, some form of categorization is required. Furthermore, such a categorization will allow for a large degree of re-use, through the possibility of context catalogues.

Given that we wish to classify context elements, we now should ask which, and how many, classes are useful. Existing approaches in mobile scenarios, as described above, usually focus on one or two classes, which are key to the scenarios they wish to cover: these are typically *Location* and/or *Device*, sometimes *Time*. The limitation to one or two classes has the advantage that, in theory, the model will remain easier to understand; however such a model is less expressive and limits the combination possibilities. For context modeling representing a broad variety of Web application scenarios, we propose the following categories of context parameters (see also Figure 2): *User&Role*, *Process&Task*, *Location*, *Time*, *Device*.

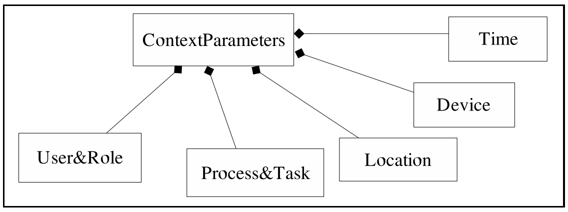

Figure 2. *Overview of the context parameter categories*

User&Role provides a categorization of users according to their role, such as various types of customers, or different types of employees.

Process&Task represents a functional context, such as work items for employees.

Location is a categorization of locations relevant to the application, in the desired granularity: for some applications, the country may be sufficient location information, for others, the city, and so forth. These locations are not to be confused with the locations *sensed* by the system, such as by GPS positioning, user input, or network address (IP address or the like). Furthermore, this location categorization does not refer to locations in the information space (i.e. where is a certain Web media located) nor their proximity location-wise to other elements in this space: in our terminology, this type of relationship is part of the domain ontology, not of the context dimensions.

Time refers to different types of time information may be relevant, such as the time-zone of the client, the actual time, a virtual time, etc.

The *Device* category contains that device and device-related information which may be a relevant context parameter (such as in mobile scenarios), e.g. the device type, display properties, etc. A property such as bandwidth can be used for (automatic) application adaptation: if bandwidth (as a context parameter) is sensed to be below a certain threshold, or changes to fall below that threshold, the streaming of audio or video can be reduced in quality, in order to maintain the same refresh rate nonetheless - see e.g. (Nepal and Srinivasan, 2003).

Related work aiming at integrated modeling of contextual factors sometimes makes more abstract categorizations of context factors, such as "physical environment, social environment", or “material context, social context” (Petrelli et al., 2001). Such categorizations are useful for high-level reasoning about context, and/or for experts in knowledge representation. For user-oriented modeling (which, in our opinion, is required for Web Engineering), we believe that more concrete categories, in a user-oriented terminology are more helpful. This will hopefully facilitate the modeling process and, consequently, the meaningfulness of the model.

4.3. *Definition of Context*

Given the above categorization of context parameters, the *context space* C may be defined as the combination of context parameters, domain ontology elements and service descriptions:

$$C = \{U, P, L, T, D, I, S\} \quad [1]$$

where U is the set of user & role factors, P the processes & tasks, L the locations, T the time factors, D the device factors, I the available information items, and S the available services (or service descriptions). A specific *context* is then a point in the context space.

The relevance space R can then be defined as the product of the context space with relevance factors:

$$R = C \times \mathbb{R} \quad [2]$$

Relevance factors are associated with items in the hierarchy, i.e. context parameters and elements from the domain ontology, including service descriptions. Sub-items in the hierarchy implicitly carry the same factor, unless overridden by a more specific value.

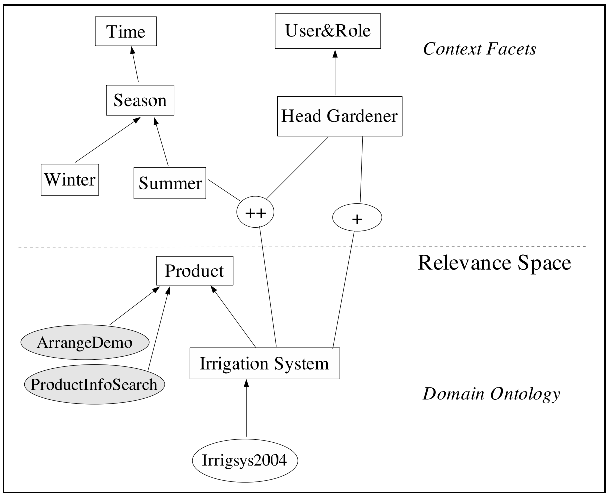

Figure 3. *Relevance space: an example*

Figure 3 provides a context model and an associated relevance space which models the example scenario discussed in section 3. A certain type of product from the domain ontology (“Irrigation System”) is modeled to be relevant (marked as “+”) for a certain user role (“Head Gardener”). Furthermore, in a certain season (“Summer”), and for the same user profile, the element of the domain ontology is modeled to be even more relevant (marked as “++”), making it more susceptible in appearing e.g. in a customized offering. All product instances (such as “Irrigsys2004”) attached to this product type can then be used in that context. Attached to the “Product” hierarchy are two *services*, “ArrangeDemo” and “ProductInfoSearch” which may be offered to the user for any product.

As shown in this example, the business user may model the relevance factors in the granularity of her choosing; not all elements need be explicitly linked with relevance factors. However, the elements’ presence is useful even if not explicitly linked: such elements can nonetheless carry a relevance factor, by their proximity in hierarchies to elements which *are* linked. Note also that although our definition of context allows for arbitrary relevance values, we presume that a higher-level definition as shown in the example will yield more meaningful results, given that the relevance factors will need to be verified by a business expert in a sensible manner.

At this stage of context modeling, the services to be integrated are viewed as simple domain elements, in analogy to elements of the information space. A service can be associated to the context parameter hierarchy, with a certain relevance, just as the other elements. A more detailed specification of the service will later be necessary in order to provide a meaningful mapping of application parameters (such as a GPS location) to a context parameter used in the model (such as a geographical zone relevant to the current business domain). (Strang et al., 2003) gives an approach in particular for specifying Web Services such as to integrate context information, focusing on the environmental context. The hierarchy of location parameters might also be modeled by an approach such as "semantic location" (Pradhan, 2000).

4.4. *Relevance Values*

The issue of precision of the relevance setting was introduced by the example in Figure 3. In principle, it is necessary to maintain the possibility of arbitrary precision for a relevance setting: a relevance setting need not be explicitly specified by the user, but could either be the result of automated extraction from company resources, or be a result of an item’s proximity to other items which do have explicit relevance factors. For the latter, an algorithm will compute the actual relevance factor. For such items for which the *user* will want to set a relevance factor, the possibility of entering a high precision value (such as a probability value) may be counter-productive, as it is more likely to result in confusion and errors. Instead, the business expert might specify the relevance factor at a "high-level", perhaps in five categories: “++” meaning very highly relevant, “+” meaning highly relevant, no marking meaning somewhat relevant, “−” meaning rather irrelevant, “−−” meaning totally irrelevant.

A "−" or "−−" setting would indicate items which are inappropriate in a given context, and thus may bother the end-user, or even be considered offensive if presented in a certain context. Consider the following example to illustrate the need for such an "irrelevance relationship": a retail store services many types of consumers; its products based on a certain type of meat should not be recommended to customers of whom the system knows they are of a religion which forbids the consumption of such a product. Thus, this knowledge could be modeled by a "−−" relation, indicating "irrelevant". Note however, that we do not wish to rule out completely the offering of this product for this type of customer: the system will make an assumption that this product will likely be irrelevant for this user, however there may be situations where it is not, for instance if this particular user is not concerned with such constraints, or is shopping for somebody else who is not concerned. Alternatively, the latter scenario could also be modeled as a task "buy for a friend" within the *Task&Process* context dimension, in which case the personal profile would not be used in the same manner. To summarize, the system should avoid recommending a product if it is probably not relevant, however it should not forbid showing it altogether, for instance consequently to the user explicitly choosing it, by navigation or searching.

4.5. *Practical Considerations*

The availability of several dimensions in the context model obviously raises issues of practicability. Potentially, an item of the relevance space may need to be displayed in as many dimensions as there are parameters, making this display very difficult to interpret. Yet, a useful interpretation for a user is necessary, as we believe user adjustments to the model should be possible at any time. We postulate that the modeling in such a manner is nonetheless useful: a given relevance link will usually not be a link between all dimensions, but more likely between two or at most three, and recent research offers new tools which assist in visualizing a multi-dimensional information space - model visualization software is discussed in the next section.

Modeling relevance relations between context parameters and domain ontology elements is conceivably a lengthy task, which we believe can be considerably supported by the system, through mining of the data stock available at the customer's site. One approach would be to look for co-occurrences and set relevance factors according to statistical values. The expert user would then be asked to verify the accuracy of the model. For an applied example of generation of a concept network through data mining, see (Staudt et al., 1998). Orthogonal to data mining, catalogues of context parameters can be established, some domain independent (such as e.g. geographic locations), others business domain specific, such as typical user roles for a specific business domain. One approach which may be useful in this regard, particularly if business rules are also to be represented, is "adaptive object-models" (see (Yoder et al., 2001)).

5. The Context Model in the Web Engineering Process

5.1. *Modeling Activity*

5.1.1. *Overview*

The modeling for a specific application domain will consist of linking elements of the domain ontology (i.e. the domain knowledge in the strict sense) with elements from the context dimensions, and assigning a relevance factor to this link. The resulting model is called the *relevance space* (see Figure 3 for an example). This general idea is detailed in this section.

As stated, elements of context are linked to elements of the domain ontology: these may be concepts in the ontology, and information items attached to these. They may also be services which are available to the Web application, or, more precisely, to service descriptions. By service, we understand a mechanism through which the application will provide the user with a dynamic offering: e.g., a reservation service through which the user can book a flight. Typically, the Web application would offer such a service to the user, and to implement the service would rely on an actual, back-end service, which is either locally implemented or is a third-party offering. These services could in turn be Web Services (as understood in (Haas and Brown, 2004)), for which the Web application would act as consumer, but could in fact be supported by various back-end technology. Thus, in our model, service descriptions are used for describing a service which is relevant in the given context.

Note that not all parts of the context model need be explicitly relevant in the modeling process. However, they remain available to the application, and may become a part of the decision-process even if they are not directly linked to the domain resources.

A further important consideration is the potential overlapping between the domain knowledge and context parameters. In the scenario presented in section 3, it is conceivable that the ontology for the construction market already contains elements that we have thus far considered as context parameters: for instance, locations of markets may already be modeled in the domain ontology. In our perspective on the engineering process, these would then be considered elements of the context dimensions as well. To assist the user in such a case, these elements can then be imported into the context dimensions within the modeling environment.

5.1.2. *Modeling Process*

The actual context parameters might come from various sources:

– predefined context catalogs, such as categories of time like spring, summer, fall and winter; geographic zones; user roles typical to certain business domains; etc.

– the domain ontology, as it may already contain elements pertaining to context (see above). For instance, geographic zones might already be represented in the ontology, if inherent to the business domain.

– custom additions by business experts.

A typical modeling scenario might then be as follows:

1) the domain ontology is read from an existing source, or established (independently of context-awareness considerations),

2) a business expert chooses categories of context parameters which are appropriate for the business (e.g. a "season" catalogue from the "Time" dimension),

3) a data-mining tool runs through the available data pools and makes suggestions as to which items from the domain ontology are relevant in connection with which elements from the chosen context catalogues,

4) the business expert chooses to import into the model those context relationships offered by the mining process that she considers to be appropriate, and fine-tunes the model in a user-centered context editor.

As suggested in the above scenario, our aim is that much of the model will be automatically generated using the available sources, and then fine-tuned by a business expert. This generation activity is however not the focus of this article.

5.1.3. *Model Use in the Development Process*

Once defined, the contextual model is available to the remainder of the Web Engineering process. One can distinguish between two types of approaches: complete integration or part-wise integration. By complete integration, we understand that all parts of the Web application would be context-sensitive, i.e. all menu navigation and information presented would be generated according to context. In a part-wise integration, the application would in some parts be context-independent, and in other, selected parts, present context-sensitive menu navigation and information.

We believe that, typically, not all parts of a Web application will require a context-sensitive model. In fact, for some aspects adaptation may be counterproductive. Therefore it is an issue of the Web Engineering process to determine in which parts, which types of context-sensitive resources will be useful. It should be noted that, in any case, context parameters will need to be modeled in an integrated fashion with the application domain if a useful usage of contextual factors is to be achieved: in this light, the decision whether a Web application should be fully or partly context sensitive is secondary.

5.2. *Applications for Model Visualization and Editing*

For user-centered visualization and manipulation of context relationships as outlined in this article, general-purpose ontology editors are clearly not suited. A particular challenge consists in assisting the user for the task of integrating the various context dimensions with the domain ontology. Recall that our context space consists of the connection between elements from the domain ontology and one or several other elements, typically coming from the context dimensions. This connection is a contextualization instance and carries further information (such as a relevance value).

To support the user in visualizing an existing context space and to allow for editing it, we have investigated using the MatrixBrowser (Ziegler and Kunz, 2002). MatrixBrowser is a tool which assists in visualizing and editing a two-dimensional information space. Figure 4 shows an example of contextual factors (as understood in this

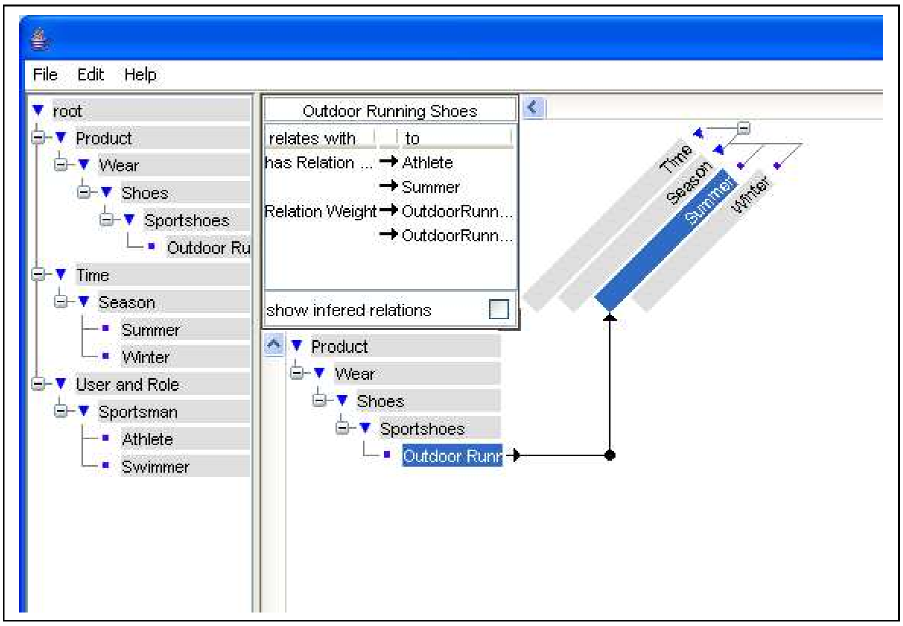

Figure 4. *Editing contextual factors using the MatrixBrowser*

article) being displayed in the MatrixBrowser. The advantage of the MatrixBrowser's approach is to provide an intuitive visualization of the connection between two hierarchies of information.

For the purpose of an n-dimensional context editor, we are currently developing a tool code-named ContextBrowser. Figure 5 shows the same context information as that shown in Figure 4 displayed in this system. The goal of this prototype is to investigate how n-dimensional context connections may be handled in a user-intuitive fashion. In this visualization, a hierarchy in the domain ontology ("Product") is shown; each element possessing a context relationship displays an arrow and up to two numbers indicating the number of relationships: the first number indicates implicit relationships due to the relationship of an element higher up in the hierarchy; the second number indicates the number of explicitly modeled relationships. Clicking on the arrow will unroll the connected context dimensions. The user can switch perspectives to the connected ontologies, or have an extract shown in a separate window (in this case, "Time").

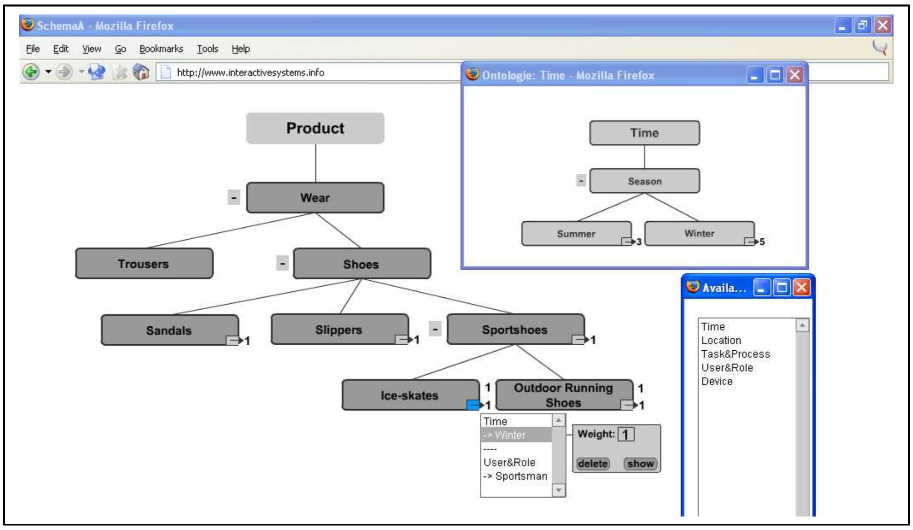

Figure 5. *ContextBrowser: multi-dimensional context relationships*

The MatrixBrowser provides a good overview in situations where a large number of domain elements and relationships exist, but lacks functionality regarding detailed viewing and handling of a single relationship - this is the feature we are exploring with the ContextBrowser. In the future, we hope to merge both approaches in order to achieve a fully usable context editor.

5.3. *Runtime Context System*

5.3.1. *Context-sensitive Resources*

Given specific context parameters (such as described in the previous section), the system will provide resources which are deemed to be relevant in this context - these are what we term the context-sensitive resources. Reciprocally, given a specific resource, the system can provide contextual factors for which this resource is deemed to be particularly relevant - this information could then be used e.g. to search for other resources relevant in the context.

We distinguish three types of context-sensitive resources (see Figure 6):

– $Navigation$ refers to navigation possibilities which are relevant to this context, through a menu or through other types of links. For instance, the context-model can provide links which are probably of interest, thus permitting adaptive navigation, where it is desired.

– $Services$, in analogy to $Navigation$, refers to available services relevant to this context. Service description for relevant services can be used to dynamically search

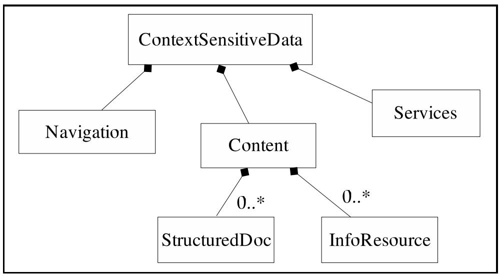

Figure 6. *Classes of context sensitive data (output)*

for available, relevant services. Such a service can be presented to the user, leaving it up to her whether to execute this service; but such a service could also be transparently executed, e.g. to dynamically retrieve information without requiring user interaction.

– $Content$ refers to documents or other types of structured information which are relevant in a given context and can be retrieved. The structuring could consist of DocBook (OASIS, 2002) or similar XML mark-ups, thus enabling a straightforward adapted presentation through the use of stylesheets. Another type of content elements would be arbitrary information resources, such as office documents, which the user could then choose to retrieve and open as a whole.

5.3.2. *Precision and Filtering*

Given that there is a categorization of context parameters in the model (see above) and thus a concept of neighborhood, a request to the context model for relevant resources may have different results. The request may be more or less granular; typically not only exact matches will be wanted, but also neighboring resources. Since the relevance space is linked with a relevance factor, any element of the context space will have a certain relevance factor. We propose the use of a *context filter* or *context lens*, which will have a certain relevance setting; according to the setting, those elements of the context space will be matched which have a relevance factor equal to, or greater then the lens setting. Once the conceptual model has been established as described above, several inference mechanisms for the generation of the adaptation model might be used; however, the issue of inference is beyond the scope of this article and a subject for future work.

5.3.3. *Model Representation in the System*

The Web Engineering process as outlined in this article being based on the usage of a domain ontology and context categories as ontologies, representing additional context information such as our relevance relationships within an ontology as well yields the advantage of easy integration with the remaining information. This section provides some extracts of a representation of this ontology using OWL [1], with the goal of clarifying how basic context information is represented in the system.

A relevance relationship is a connection between two, or more OWL “things”; this connection is weighted (see Listing 1).

```
<owl:Class rdf:about="#Relevance">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#relevanceOnt" />
      <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">2</
          owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#relevanceWeight" />
      <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

<owl:ObjectProperty rdf:ID="relevanceOnt">
  <rdfs:domain rdf:resource="#Relevance"/>
  <rdfs:range rdf:resource="&owl;Thing"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="relevanceWeight">
  <rdfs:domain rdf:resource="#Relevance" />
  <rdfs:range  rdf:resource="&xsd;decimal" />
</owl:DatatypeProperty>
```

Listing 1: *OWL representation of context information*

Listing 2 provides examples of relevance instances based on the examples outlined in previous sections (see section 3 and Figure 3): elements existing in other ontologies (domain ontology plus context ontologies, not shown in this extract) are connected by a relevance weight.

```
<Relevance rdf:ID="re1">
  <relevanceOnt rdf:resource="&product;IrrigationSystem"/>
  <relevanceOnt rdf:resource="&user;HeadGardener"/>
  <relevanceWeight rdf:datatype="&xsd;decimal">0.5</relevanceWeight>
</Relevance>

<Relevance rdf:ID="re2">
  <relevanceOnt rdf:resource="&product;IrrigationSystem"/>
  <relevanceOnt rdf:resource="&user;HeadGardener"/>
  <relevanceOnt rdf:resource="&time;Summer"/>
  <relevanceWeight rdf:datatype="&xsd;decimal">1.0</relevanceWeight>
</Relevance>
```

Listing 2: *OWL representation of relevance instances*

5.3.4. *Context System*

In our approach to Web Engineering, the context system is an independent module, which may be “plugged” into the Web software where deemed useful by the Web En-

1. Ontology Web Language, (Bechhofer et al., 2004)

gineer. Indeed, experience, most notably in the adaptive hypermedia field, has shown that adaption, respectively context-sensitive behavior, is not useful in all situations and can in fact be counter-productive. The context system we propose is meant to integrate with a process which includes the creation (or usage) of a domain ontology. The context system also uses this domain ontology, upon which the relevance space is based (see above). The application design process will then determine at which points context-sensitive resources are useful. The application would thus provide the context system with current parameters, and receive in return the desired context-sensitive resources.

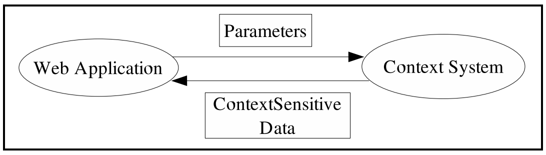

Figure 7. *Context system (black-box view)*

Figure 7 gives a simple, black-box type view of the context system. What is noteworthy here is that context-relevant parameters as given by the Web application are not identical to the context parameter modeling - the useful matching is a task of the context engine. The context engine returns to the Web application the desired context-sensitive resources (see above).

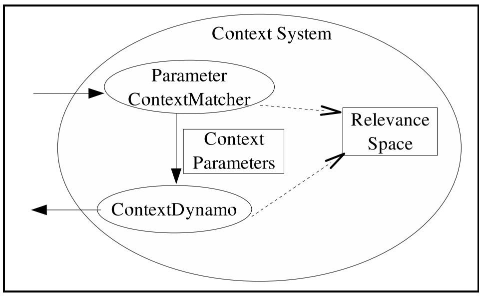

Figure 8. *Context system - interior overview*

Figure 8 provides an overview of the context system's interior workings. Parameters provided by the Web application can be of two types. The first type are elements of the domain ontology: given that the Web application is built upon a domain ontology, it "knows" which elements of the ontology are currently relevant. One basis for this knowledge is ontology-based navigation, where parts of the menu navigation structure were constructed from an ontology, and the user has navigated through such a structure. This information is provided as parameters to the context system. The second type of parameters are classical Web application parameters, such as client device parameters. Such parameters are transformed into context parameters as known by the context engine (defined here as $ContextParameters$). For instance, the Web application may provide a time context in date, hours and minutes; however in the relevance space only higher-level distinctions of time, such as spring, fall etc. are used, so this parameter is transformed accordingly. The same mappings may be needed for the other types of parameters as well. The context parameters are then passed on to the $ContextDynamo$, which determines which resources are relevant in this context, according to the $RelevanceSpace$ and the current context-filter setting. These resources are then returned to the Web application, which may then present them to the user.

6. Outlook and Future Work

We believe the development of an integrated conceptual model for context-aware Web Engineering to be a promising approach: from an engineering perspective, the integration with a domain ontology can enable the systematic development of a Web application for which contextual factors may be used; it can support the automatic generation of the context model in a significant proportion; and it can lead to the creation of context catalogs pertinent for specific business domains. It is thus reasonable to assume a meaningful degree of re-usability may be achieved not only for the domain ontologies, but also for the context models. From a usability perspective, the end-user can be presented with relevant information and menu navigation possibilities, and furthermore can interact with available, relevant services, thus providing for Web applications better tailored to the user's needs and, hopefully, raising the productivity of people working with these applications.

Future work will focus on the following issues: (1) detailed definition of context-adaptive service integration, (2) to what degree can the context-sensitive conceptual application model be automatically generated, and with what kind of tools? (3) the development of useful context catalogs, depending on application domains (e.g. business catalogs, geographical catalogs, etc.).

References

Barry C., Lang M., "A Survey of Multimedia and Web Development Techniques and Methodology Usage", *IEEE MultiMedia*, vol. 8, num. 2, 2001, p. 52–60, IEEE Computer Society Press.

Bechhofer S., van Harmelen F., Hendler J., Horrocks I., McGuinness D. L., Patel-Schneider P. F., Stein L. A., "OWL Web Ontology Language Reference", Recommendation, http://www.w3.org/TR/2004/REC-owl-ref-20040210/, 10 February 2004, W3C XML Protocol Working Group.

Brezillon, P.: A Context Approach of Social Networks. In: Proceedings Workshop on Modelling and Retrieval of Context, Vol-114, CEUR (2004)

Brusilovsky P., Kobsa A., Vassileva J., *Adaptive Hypertext and Hypermedia*, Kluwer Academic Publishers, 1998.

Cannataro M., Cuzzocrea A., Pugliese A., “XAHM: an adaptive hypermedia model based on XML”, *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, ACM Press, 2002, p. 627–634.

Chen G., Kotz D., “A Survey of Context-Aware Mobile Computing Research”, report num. TR2000-381, November 2000, Dept. of Computer Science, Dartmouth College.

Chen H., Finin T., Joshi A., “An Ontology for Context-Aware Pervasive Computing Environments”, *Special Issue on Ontologies for Distributed Systems, Knowledge Engineering Review*, vol. 18, num. 3, 2004, p. 197–207, Cambridge University Press.

Cheverst K., Davies N., Mitchell K., Friday A., Efstratiou C., “Developing a context-aware electronic tourist guide: some issues and experiences”, *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, 2000, p. 17–24.

Dix A., Rodden T., Davies N., Trevor J., Friday A., Palfreyman K., “Exploiting space and location as a design framework for interactive mobile systems”, *ACM Trans. Comput.-Hum. Interact.*, vol. 7, num. 3, 2000, p. 285–321, ACM Press.

Forman G. H., Zahorjan J., “The Challenges of Mobile Computing”, *IEEE Computer*, vol. 27, num. 4, 1994, p. 38–47, IEEE Computer Society Press.

Haas H., Brown A., “Web Services Glossary”, Working Group Note, http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/, W3C Note, 11 February 2004, W3C Web Services Architecture Working Group.

Koch N., “Software Engineering for Adaptive Hypermedia Systems: Reference Model, Modeling Techniques and Development Process”, PhD thesis, Ludwig-Maximilians-University Munich, 2001.

Murugesan S., Deshpande Y., Hansen S., Ginige A., “Web Engineering: A New Discipline for Development of Web-Based Systems”, Murugesan S., Desphande Y., Eds., *Web Engineering. Managing Diversity and Complexity of Web Application Development*, vol. 2016 of *LNCS*, Berlin, Heidelberg, 2001, Springer, Page 3 ff.

Nepal S., Srinivasan U., “DAVE: a system for quality driven adaptive video delivery”, *Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*, ACM Press, 2003, p. 223–230.

OASIS Consortium, “The DocBook Document Type”, Committee Specification, http://www.oasis-open.org/docbook/specs, 2002, OASIS.

Petrelli D., Not E., Zancanaro M., Strapparava C., Stock O., “Modelling and Adapting to Context”, *Personal Ubiquitous Comput.*, vol. 5, num. 1, 2001, p. 20–24, Springer-Verlag.

Pradhan S., "Semantic Location", *Personal Ubiquitous Comput.*, vol. 4, num. 4, 2000, p. 213–216, Springer-Verlag.

Rheingold H., *Smart Mobs: The Next Social Revolution*, Perseus Publishing, 2002.

Samulowitz M., Michahelles F., Linnhoff-Popien C., "Adaptive interaction for enabling pervasive services", *Proceedings of the 2nd ACM international workshop on Data engineering for wireless and mobile access*, ACM Press, 2001, p. 20–26.

Staff C., "HyperContextr: A model for Adaptive Hypertext", Jameson A., Paris C., Tasso C., Eds., *Proceedings of the Sixth International Conference on User Modeling (UM97)*, p. 33–39, Springer, Berlin, June 2-5 1997, (Chia Laguna, Sardinia, Italy).

Staudt M., Kietz J.-U., Reimer U., "A Data Mining Support Environment and its Application on Insurance Data", *Knowledge Discovery and Data Mining*, 1998, p. 105-111.

Strang T., Linnhoff-Popien C., Frank K., "Applications of a Context Ontology Language", Begusic D., Rozic N., Eds., *Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCom2003)*, Split/Croatia, Venice/Italy, Ancona/Italy, Dubrovnik/Croatia, October 2003, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, Croatia, p. 14–18.

Tipping M. E., "Sparse bayesian learning and the relevance vector machine", *J. Mach. Learn. Res.*, vol. 1, 2001, p. 211–244, MIT Press.

Wissen M., Ziegler J., "A Methodology for the Component-Based Development of Web Applications", *Proceedings of 10th Int. Conf. on Human - Computer Interaction (HCI International 2003), Vol. 1, Crete, Greece*, 2003.

Yoder J. W., Balaguer F., Johnson R., "Architecture and design of adaptive object-models", *SIGPLAN Not.*, vol. 36, num. 12, 2001, p. 50–60, ACM Press.

Ziegler J., Kunz C., Botsch V., "Matrix browser: visualizing and exploring large networked information spaces", *CHI '02 extended abstracts on Human factors in computing systems*, ACM Press, 2002, p. 602–603.