Adapting web sites by spreading activation in ontologies

Tim Hussein
University of Duisburg-Essen
Lotharstr. 65, 47057 Duisburg, Germany
tim.hussein@uni-due.de

Jürgen Ziegler
University of Duisburg-Essen
Lotharstr. 65, 47057 Duisburg, Germany
juergen.ziegler@uni-due.de

ABSTRACT
In this paper we introduce SPREADR, a model-based technique for creating context-adaptive web applications. In this approach, the domain knowledge as well as context factors are represented by an ontology. Context factors depend on the application domain and may include location, time, user role, weather or any other relevant context information, often information which can automatically be sensed. Both domain objects and context factors are treated as concepts or instances in the ontology, which can be linked to each other, thus forming a semantic network. In addition to this representation, we introduce scalar activation levels for concepts as well as weights for relations. Concept activation represents the level of user interest whereas activation of context factors can be seen as the level of fulfilment based on some measurement. The structure of the semantic network remains the same for each user while activation levels may differ. Recognition of context factors or user actions trigger an activation flow through the network, thus increasing the activation of contextually important nodes. Relations are associated with different weights resulting in different amounts of activation spreading along different relations. The resulting spreading activation network can be used both for generating a web site as well as for controlling the adaptive behaviour of the system. Adaptation may include effects such as sorting navigation items by relevance, showing or hiding information depending on the activation value and highlighting or recommending important items. The system is also able to learn user preferences through implicit feedback. We present a demonstrator application as well as initial evaluation results.

Author Keywords
Adaptation, Ontologies, Spreading Activation, Recommender Systems

ACM Classification Keywords
H.3.3 Information search and retrieval: Selection process

INTRODUCTION
The increasing amount of information presented in current web based applications often makes them difficult to use. Finding the appropriate content within the flood of data can be challenging and may cause a user to reject a web application. Various approaches have been proposed to overcome these problems by adaptation, each with its particular advantages and drawbacks. Like [12] we propose an integrated view of context and domain information to contextually offer the appropriate content. For this purpose we make use of two concepts originally known within the scope of artificial intelligence and information retrieval: Ontologies and spreading activation. The basic idea behind SPREADR is that all domain items which are supposed to be displayed are semantically linked to each other in an ontology. Furthermore all relevant context-factors that should have an influence on the content selection are modeled in ontologies, too. For example, a location ontology can be used to describe possible places from which the user can access the system, certain time (seasons, time of day,...), weather or device aspects (PDA, standard PC,...). The domain ontology plus the different context ontologies are aggregated into a single Spreading Activation Network – a graph-based representation of the domain and all relevant contextual information. Each item or concept is a node in this graph and has a certain activation value[1] assigned to it, which represents the current degree of importance.

Nodes are semantically related to other nodes by weighted links[2] representing the strength of the relationship. As a result of appropriate context sensing techniques, a certain amount of activation is injected into the corresponding context node. This activation spreads through the network along the relations defined until a predefined termination constraint is met. The amount of activation propagated from a sending to a receiving node depends on weight of the relation between them. Since relations are directed activation can only flow in the direction of the link.

The result is a weighted network that can be interpreted as a profile of interests. It is now possible to realize various kinds of adaptations within the web application: Rearrange the navigation menu based on the relevance of each entry according to the current usage context, recommend certain items that are supposed to be very interesting, create relevance based tag clouds or simply adjust certain colors according to the context (e.g. by using higher saturation for very important items).

SPREADR keeps track of the recent activation transmissions and can look up the paths that led to certain adaptation effects. If this adaptation is accepted by the user (because he clicks on the recommended item), SPREADR can amplify the path leading from the initially activated node to the recommended one. The idea behind this is that this certain item

[1]The activation value is a real number $x \in [0..1]$
[2]The weight is also a real number x $x \in [0..1]$

actually is very important in this specific context. The items should be coupled more tightly, so that even more activation energy should be transmitted through that link the next time. Vice versa the relation weights along the path should be attenuated, if the recommendations are rejected. Thereby the system learns from the user's behaviour and decisions.

In section *Domain and Context models* we illustrate our context engineering approach and show a method to integrate contextual information into an existing domain ontology. The Spreading Activation technique we make use of in order to adjust the activation weights is explained in section *Adaptation by spreading activation*. Then we give a brief overview of the SPREADR architecture followed by a section about our experiences with the system. Finally we compare our system to other approaches that aim in a similar direction and finish with a conclusion.

DOMAIN AND CONTEXT MODELS

When building a model-driven context-aware web application, it is not only necessary to model the domain, but also the context. Here, context is "any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves" [11]. SPREADR can be used for any imaginable scenario as long as you can map it to an ontology represented in the Web Ontology Language (OWL) format. For the sake of clarity it is recommended not to mix up context dimensions but create one ontology file per dimension. SPREADR aggregates them on system startup into a Spreading Activation Network.

Like [19] we believe that the context model(s) should not be mixed with the domain model(s). A domain model should be created by domain experts without considering the context-adaptations that are supposed to take place. Several distinct context dimensions that might be meaningful for most scenarios can be found in [34].

The domain model

One ontological model that is absolutely necessary is the domain model. Herein the essential domain knowledge is represented, that means: All concepts[3], all instances of those concepts and the relations between them are modeled. According to custom those relations are semantically labeled[4]. The result is a semantic network containing the complete domain knowledge.

Surely it is big effort to design such a domain ontology accurately. But this information can be exploited during the web engineering process, too, and helps keep the data model independent from the web application logic. Thereby it is possible to adapt SPREADR to any given scenario with minimal effort once the corresponding ontologies have been created. Thus SPREADR can also be seen as an example of a model driven web application.

[3]For instance the concept of a DVD

[4]For instance *hasDirected*

The context models

It is possible to integrate several other models that will be taken into account by the system. For each context dimension the system is supposed to be aware of, one ontology should be defined (with the desired granularity). Depending on the scenario it may be useful to model continents, regions and countries or perhaps it may be more reasonable to describe cities, districts, buildings or departments. Of course one has to make sure that proper location sensing mechanisms are supported - for instance by IP resolving. Currently SPREADR does not offer sophisticated context sensing mechanisms itself, only a context simulator, that can be used to evaluate the system's behaviour in a certain context.

In this manner it is possible to create models for any context dimension imaginable – time of day, season, weather, mood, social status, whatever makes sense in that particular scenario. However, for the context to have an effect on the content selection, it is also necessary to model appropriate *context relations*. A context relation defines a link between a context factor and an item from the domain model. A mapping between identified context factors and domain items is defined in a so called *context relations ontology*. Thus certain domain items can be directly activated in a given context at runtime (and of course spread the activation to semantically related items. Figure 1 illustrates how context factors can be connected to the domain model. For details see section *Adaptation by spreading activation*).

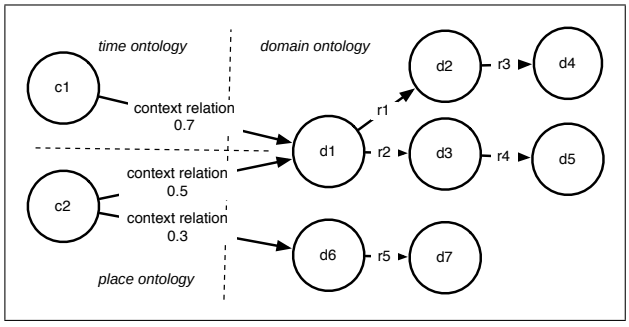

Figure 1. Connecting context factors to the domain model

Relation weights and activation values

As already mentioned in the *Introduction* we use relation weights to determine the degree of relationship between two concepts or instances. The resulting networks are identical in structure for each user, but the weights of the links are individualized. A relation between two nodes can be seen as a matter of fact, that is universally valid. The relation "Arnold Schwarzenegger is the Governor of Californa" is identical for every user, but the importance of the connection between "Arnold Schwarzenegger" and "Governor of Californa" may vary individually. One may for instance be very interested in his movies but absolutely not in his political career. By assigning such relation weights we can create individual weighted networks for each user to represent his view on the domain.

Nodes are handled in the same way by assigning activation values to them. If one considers an item to be important it

receives a higher activation value. Regarding context factors the activation level can be seen as a degree of fulfilment (as applied in fuzzy systems). In mid July the node "Summer" will surely get an activation value of 1, in early September maybe only 0.2. In both cases it is summer, but mid July is much more typical for that season than early September. The resulting models in conjunction with the individual link weights and activation values can be interpreted as an individual profile of interest and context.

The models should be absolutely reliable and reflect reality. Thus it is necessary to define the relations and the initial weights manually. An explanation of how the node's activation levels and the link weights are adjusted is given in the next section.

Once created they are reliable sources to base adaptations upon. Furthermore ontologies are can easily be extended or aggregated. We explained earlier that our context model consists of several ontologies, each one representing a certain context dimension plus one ontology only representing the relations between nodes of different models. So the ontologies can be reused in different scenarios. A time ontology may be useful in many scenarios and only has to be created once.

Moreover the semantic relations between the domain items and the context nodes can be exploited in several ways:

1. Parts of the navigation structure can be automatically created from the extracted class information described in the ontology.

2. By using very short queries[5] it is easily possible to filter nodes by their types or relations.

3. As the whole scenario including domain and context knowledge is entirely modeled in those ontologies, the system is totally independent of the content to be displayed. We tested SPREADR with several different scenarios and it is a matter of minutes to adapt the system after the owl-files, that represent the ontologies, are copied into the appropriate folders.

4. Changing the structure of the information can be achieved without hassle. It is not necessary to create or change a database schema or modify the program code.

All listed features are already integrated into SPREADR. But as this belongs to the field of model driven web engineering it is not described further in this paper.

ADAPTATION BY SPREADING ACTIVATION

In this section we describe the mechanism that is responsible for the activation and weight adjustments. For that purpose we make use of a technique called "Spreading Activation" - a concept that was proposed in the 1970s by Collins and Loftus [9]. Their model of spreading activation networks was originally applied in the fields of psycho linguistics and semantic priming [2]. Later, the idea was adopted by computer scientists: Spreading activation techniques have successfully been used in several research areas in computer science, most notably in information retrieval ([8], [10] and [3]). The principles of spreading activation have also been used by [29] in their information foraging theory.

The basic concept behind Spreading Activation is that all relevant information is mapped on a graph as nodes with a certain "activation level". Relations between two concepts are represented by a link between the corresponding nodes. If for any reason one or more nodes are activated their activation level arises and the activation is spread to the adjacent nodes (and the ones related to them and so on) like water running through a river bed. Thereby the flow of activation is attenuated the more it strides away from the initially activated node(s). At the end several nodes are activated to a certain degree that are semantically related to the concepts originally selected.

At the beginning, each node has an initial activation value of 0. We use the current context as the starting point for the Spreading Activation: When a new session starts all necessary context information is sensed and the nodes representing the recognized context factors are used as initial nodes to trigger the activation flow. As a result concepts and items that are related to the current context have a high activation value. A little example should illustrate this: The setting may be an information portal for leisure activities. Imagine the user starts a new session from Duisburg and the current time is evening. The system recognizes the time and the current location. This may result in raising the activation levels of all venues located in her city and events that take place on that particular evening are highly activated, too. Perhaps the system is configured to take local weather information into account as a context factor. So perhaps open air events may get a higher activation than indoor events. The flexible architecture allows a combination of arbitrary context information. Finally all nodes have a certain activation value that represents their degree of relevance in the current situation.

After that run the activation values are not reset but refined with every user action. If he clicks on a certain domain item - for instance a concert - this interaction is taken into account, too. The node representing that particular concert now is the initial node to another Spreading Activation run and transmit activation energy to all related concepts and items. That may include the concept of a concert in general (and thereby activating other concerts), the artist, the music genre, the venue of the concert and so on. Each interaction adopts the weights more and more to the current context and the user's interests.

At this point no adaptation is performed yet. Only the underlying models are adjusted to the current context and usage behaviour. We believe that it is a good idea to separate these model adjustment mechanisms from the process of web page generation. Any changes in the page generation and presentation framework won't affect the reasoning part of the system and vice versa.

[5]We use SPARQL, an RDF query language for that purpose.

Several algorithms have been developed to implement the concept of spreading activation. Details and a comparison can be found in [18]. We chose the so called *Branch-and-bound* approach.

The branch-and-bound algorithm

The following steps describe a single Spreading Activation run. As mentioned earlier the activation values are not reset after each cycle so that the networks can develop into a kind of user and context profile. During a Spreading Activation run two phases can be distinguished: Initialization and execution.

Initialization:

Before the actual execution of spreading activation begins, the network must be initialized:

1. The weights for the links are set based on the user's individual context model. Moreover, in our approach, the network is not necessarily in a blank state when a spreading activation run starts. Therefore, initial activation levels for each node in the network are set. These are based on the resulting activation levels of the previous run.
2. The initial nodes are activated with a certain value. The activation received by the start nodes is added to their previous state. Optionally the new activation level is calculated by applying an activation function to this sum.
3. The initial nodes are inserted into a priority queue ordered by descending activation.

Execution:

After initialization, the following steps are repeated until a defined termination condition is fulfilled or the priority queue is empty. The termination condition can be configured freely, but two pre-defined termination conditions are provided: (1) A maximum of *activated* nodes is reached, (2) a maximum of *processed* nodes is reached. A processed node is a node that has itself propagated activation to adjacent nodes.

1. The node with the highest weight is removed from the queue.
2. The activation of that node is passed on to all adjacent nodes, if this is not prevented by some restriction imposed on the spreading of activation. If a node j receives activation from an adjacent node i, a new activation level is computed for j.

$$A_j(t+1) = A_j(t) + O_i(t) \times w_{ij} \times a$$

   where $A_j(t)$ is the previous activation of j, $O_i(t)$ is the output activation of i at the time t, w_{ij} is the weight of the relation between i and j and a is an attenuation factor. The output activation of a node is the activation it has received. An arbitrary function can be used to keep the values in a predefined range. In most cases a linear or parabolic function will be meaningful.
3. The adjacent nodes that have received activation are inserted into the priority queue unless they have already been marked as processed.
4. The node that passed on its activation to the neighboring nodes is marked as processed.

When a new spreading activation run is triggered the values are not reset, so that the network is refined every time the process is executed. We implemented an aging mechanism by attenuating each activation value by 5% before each new spreading activation run.

Configuration

Certain constraints and termination conditions can be defined to to prevent activation from spreading through the whole network and eventually activating every single node. Additionally this allows for refining the Spreading Activation process regarding performance. The process can be influenced for instance depending on the concept type, the outgoing edges or the path-distance between nodes. Details about those constraints can be found in [8] and [30].

In addition, the sub-functions described above can be configured - for instance the attenuation factor - or reverberation can either be allowed or prevented. This means that a node j must not propagate activation to a node i if node j has itself been activated by node i before in the same run. Finally, our spreading activation mechanism allows the adjustment of relation type weights. A relation type weight is used for each relation for which no individual weight has been set in the initialization phase of the algorithm.

ADAPTATION EFFECTS

The weighted networks illustrated in the previous chapter now can be used as a foundation for adaptation effects. Our flexible architecture based on the Java Spring Framework allows for rapid development of components that make use of those weighted ontologies. Being in session scope all components have access to the user's individual network. We developed a configurable recommendation box that lists the Top 10 items from a predefined list of node types[6](see (1) in figure 2). Also the Top 3 items of a category are being displayed first in an accentuated position (3).

It is also possible to sort elements of the navigation menu by relevance or perhaps automatically show certain sub-elements of a category considered to be y important in the current context (2). In area (4) newsfeeds can be displayed depending on the current context[7] Although context adaptation can significantly improve a systems usability, it is important not to confuse the user. Interesting works about whether automatic adaptation of the user interface is helpful or confusing can be found in [25], [31] and [13].

Beyond that various adaptation effects are possible entirely focussing on a single context factor: The color scheme may

[6]For instance DVDs, CDs, Concerts

[7]This is not implemented yet.

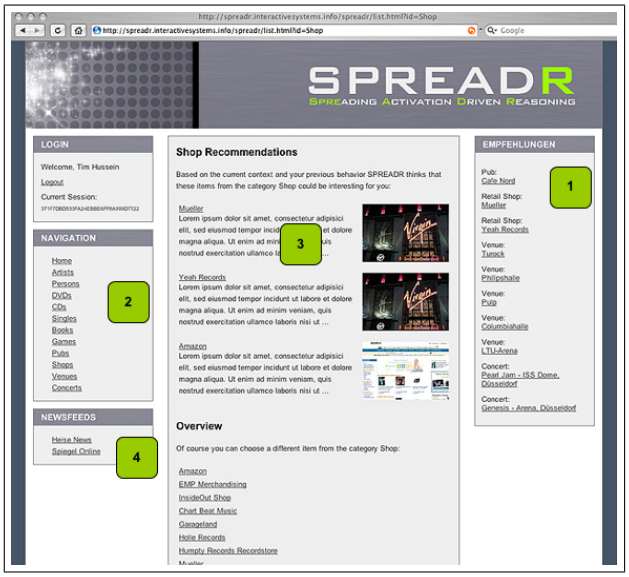

Figure 2. A typical web application powered by SPREADR

depend on the recognized season or time of day, simple services like weather information can show the temperature at the current location or certain items may simply be hidden if they are not considered to be important for the recognized user role. Of course the sense of such adaptations strongly depends on the setting.

LEARNING BY ADJUSTING THE RELATION WEIGHTS

In our opinion reasonable learning mechanisms are essential for a truly adaptive system, but we have to clarify what "learning" exactly means for a context adaptive web application like SPREADR. As specified in Section *Domain and Context models*, we use so called Spreading Activation Networks to represent the user's view on the domain and his current usage context. First the Spreading Activation itself can be seen as a learning mechanism, because thereby the system learns from the user's actions and adjusts the importance of certain concepts and items according to that. On the other hand this has no effect on future adaptations and does not incorporate any feedback.

To overcome that drawback the system can keep track of the model adjustments and the user's reactions to them. By default, each link in the base context model is defined globally in the context model[8]. That means that all relations between concepts are equal regarding their importance. This will certainly not reflect reality, but the context engineer unfortunately does not have exact knowledge about each user's view on the domain (a relation weight represents the importance that the relation has for an individual user). Thus, it is the same for each user first.

After each Spreading Activation run, the activated nodes store the path to the initial node whose propagation led to its activation – together with information on how much activation it has received via that path. The system keeps a this information for a certain amount of requests and if the user navigates to a domain item i that has recently been activated in a Spreading Activation run from a initial node j via path p, it means that the relation between i and j obviously is important. As a result the importance of that particular path is amplified by increasing each link weight along p. In this way the system learns what is important in a specific usage context and will transmit more activation through the amplified links in future runs.

Vice versa, the relation is considered to be of little importance if the user does not "confirm" the activation path within a certain period of time by requesting the recommended node. Consequently the link weights are attenuated. This idea was inspired by Hebbian Learning [15]. Details about the learning mechanism can be found in [33].

[8]We usually use 0.5 as a default value

ARCHITECTURE

Our framework currently concentrates on context reasoning and learning. Sophisticated context sensing mechanisms are planned for the future, but yet not realized. Therefore we separated the reasoning components from sensing and implemented a login mechanism to simulate certain context states. Thus it is possible to integrate any context sensing mechanism imaginable without touching the reasoning components. For the rest of this section we will consider the current context as being sensed.

In section *Domain and Context models* we explained that each context dimension is modeled in a separate ontology model and at system startup all models - including the domain model - are aggregated into a Spreading Activation Network. As a result there is an isomorphic network for each user that contains all domain and context information in semantically related nodes.

Whenever the system recognizes certain context factors, activation energy is injected into the context model in order to activate relevant domain items. Based on this information, adaptations of content or navigation can be initiated. The actual generation of the web pages that are sent to the browsers is not part of it, which allows for a maximum freedom concerning the technologies used for generating pages. We use the Java based Spring Framework for that purpose. In an MVC inspired manner a controller component usually triggers a spreading activation run and can use the results to prepare a model that can be handed over to the appropriate view component.

EXPERIENCES

We developed SPREADR as a model based approach in order to easily adapt it to different scenarios. So far we created two settings to test the effectiveness of our adaptation mechanism and to clarify our methodology: A context adaptive music portal, that currently is being extended to a more broader product and event information portal, and a typical company intranet. These are typical scenarios where adapting to the user and his current context is often considered to make sense. Both settings focus on the location and time context in addition to the user's interaction. Furthermore the

"user role" was modeled as a context factor in the intranet scenario. Methodically our evaluations refer to [32].

General Evaluation

For evaluation purposes we used the intranet scenario mentioned above including 24 service categories, 50 services, 38 contact persons, 18 events, 25 news items and 30 different support materials. We proposed the hypothesis that an intranet user finds important information with significantly less interaction when using the SPREADR system than with system without such context adaptivity. For that purpose we developed a control system in which the recommendations are chosen entirely by chance.

Eight users participated in this evaluation, all of them being familiar with the intranet domain and the content presented. Their task was to search for services or events which they felt were of interest to them. After having used the system the users had to rate several usability aspects using numbers on different scales, for instance:

- Had the recommendations presented directly after system login been helpful?
- Could the number of interaction steps be reduced to find the desired information?
- Had the adaptation effects and recommendations during system usage been helpful?

The results of this evaluation provide evidence that context adaptation supported the users task. On a scale ranging from 1 (applicable) to 6 (inapplicable) the context adaptive variation had an average rating of 1.375, whereas the non adaptive one had was rated 4.5 on average. For a detailed illustration of this evaluation see [23].

Evaluation of the learning mechanism

To evaluate the effects of the implemented learning mechanisms, another study was done with the adaptive music portal. People often have a small number of favourite artists but are not aware of other artists they might like, do not notice dates of interesting concerts taking place close to their current location, or that the music they are interested in is dependent on context such as time. In the music portal scenario, we target these problems by adapting the content of the portal to the current usage context, i.e. to the user profile enriched with activations of items by the current context. Our music portal provides album reviews, artist biographies, concert information and several kinds of additional information about events, pubs and items.

In order to evaluate the learning effets five users each had to simulate a single hypothetical user, albeit for both the experimental condition and the control condition without any link weight learning at all. The users did not know about those technical details and had to rate the quality of the adaptation effects. Although the specific tasks the users were supposed to perform were different for each of them, basically, the general procedure was the same: Adopt the given hypothetical profile and perform certain tasks like finding an interesting pub depending on your music preferences. During and after about 10 sessions with different tasks they had to rate the systems behaviour. Are the recommendations useful? Do they improve over time?

The subjects' ratings supported our hypothesis that the learning variation led to a much better usability, because the recommendations and adaptation effects have been rated considerably better when context learning was enabled. In that case the users were able to find interesting items with significantly less clicks and the presented content matched the users' interests better compared to the variation without learning. Alas the sample size is too small to provide statistically significant results. Detailed information about this evaluation are described in [33].

RELATED WORK

With increasing complexity of web applications, context becomes a substantial factor in terms of usability. Content that may be interesting for a user under certain circumstances may be totally uninteresting in a different context. Approaches that take contextual information into account for purposes of personalization have been introduced by various authors such as [17], [1] and [21].

The field of context-aware computing is sometimes called an "immature" one [16]. Though this is a harsh judgement we agree that context-adaptivity still has a long way to go. Especially there is a lack of approaches that take the user history into account as well as multidimensional context information. Unfortunately existing approaches mostly focus on single context factors like the user's current location [5]. The *a CAPella* system introduced by [12] is a context aware system, that can be "trained" by the user to automatically recognize certain events depending on the current context via multi-modal sensing: Information obtained from a microphone, a camera, RFID antennas and other devices is being used and interpreted. As a result, *a CAPella* for instance recognizes the start of a meeting and automatically presents certain documents that have been used in a similar context in the past. This interesting and promising approach is a good example for sensing and unifying context information. However, it needs certain external equipment for context sensing and strongly focusses on real-world interaction, which makes it not directly applicable for the purpose of web engineering.

For the context- and domain-models we use ontologies, which have have been proven to be a good choice for knowledge representation. Having their roots in philosophy, ontologies have become popular for computer science since the 1990s [26]. [24] is an example for using ontologies in recommender systems. In this paper *Quickstep* and *Foxtrot* are illustrated – two systems that make use of ontologies to recommend scientific research papers a particular user might be interested in. By using ontologies the authors want to overcome the typical disadvantages of pure collaborative filtering systems like the *ramp-up-problem* often referred to as explained in [4].

Similar approaches have been presented in [7] and [22]. Although these solutions do not take the particular context into account, collaborative filtering algorithms have been proven to be a simple but effective instrument that can be integrated into more sophisticated systems. In our opinion there is a strong need for an integrated strategy that incorporates as much context information as possible: The current usage context in terms of situation-awareness as well as past interactions together with the according context information for the time being.

An interesting approach can be found in [21], where classical associative spreading activation networks are enriched with "link types" and "context nodes" to generate context-adaptive recommendations. [18] use spreading activation to avoid the sparsity problem in collaborative filtering and explore transitive associations between users.

CONCLUSIONS

In this paper we introduced a novel approach to determine the most important elements of a given ontology with regard to current context and past user interaction. The resulting weighted network of concepts and instances can then be used as a foundation for adaptation effects. In our approach context relations are fully integrated into the propagation process and thus affect the adaptation activities. We agree with [28] that those web sites are adaptive, which "automatically improve their organization and presentation by learning from visitor access patterns". The proposed system meets this requirements.

After developing a fully functional prototype that we tested with entirely different settings, we are now looking forward to improve our system in various aspects. One of our short-term goals is to cluster user profiles and thus allow cross-network-propagation of activity action. Other goals include more sophisticated visualization and configuration options along with the possibility of directly manipulate the learning process by giving explicit feedback. Beyond that the context adaptive detection and integration of newsfeeds and web services is planned, hopefully leading to a truly information-centered application with context-adaptive interfaces as a long-term goal.

REFERENCES

1. G. Adomavicius and A. Tuzhilin. Multidimensional recommender systems: A data warehousing approach. *Lecture Notes in Computer Science*, 2232, 2001.

2. J. R. Anderson. A spreading activation theory of memory. *Journal of Verbal Learning and Verbal Behavior*, 22:261–295, 1983.

3. H. Berger, M. Dittenbach, and D. Merkl. *An Adaptive Information Retrieval System Based on Associative Networks*, volume 31 of *Conferences in Research and Practice in Information Technology*. ACS, Dunedin, New Zealand, 2004. First Asia-Pacific Conference on Conceptual Modelling (APCCM2004).

4. R. D. Burke. Hybrid recommender systems: Survey and experiments. *User Model. User-Adapt. Interact*, 12(4):331–370, 2002.

5. G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dartmouth College, 2000.

6. K. Cheverst, N. Davies, K. Mitchell, and A. Friday. Experiences of developing and deploying a context-aware tourist guide: The GUIDE project. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM-00)*, pages 20–31, N. Y., Aug. 6–11 2000. ACM Press.

7. M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper, 1999.

8. P. R. Cohen and R. Kjeldsen. Information retrieval by constrained spreading activation in semantic networks. *Inf. Process. Manage*, 23(4):255–268, 1987.

9. A. M. Collins and E. F. Loftus. A spreading-activation theory of semantic processing. *Psychological Review*, 82:407–425, 1975.

10. F. Crestani. Application of spreading activation techniques in information retrieval. *Artif. Intell. Rev*, 11(6):453–482, 1997.

11. A. K. Dey. Understanding and using context. *Personal Ubiquitous Computing*, 5(1):4–7, 2001.

12. A. K. Dey, R. Hamid, C. Beckmann, I. Li, and D. Hsu. a CAPpella: programming by demonstration of context-aware applications. In E. Dykstra-Erickson and M. Tscheligi, editors, *Proceedings of the 2004 Conference on Human Factors in Computing Systems, CHI 2004, Vienna, Austria, April 24 - 29, 2004*, pages 33–40. ACM, 2004.

13. L. Findlater and J. McGrenere. A comparison of static, adaptive, and adaptable menus. In E. Dykstra-Erickson and M. Tscheligi, editors, *Proceedings of the 2004 Conference on Human Factors in Computing Systems, CHI 2004, Vienna, Austria, April 24 - 29, 2004*, pages 89–96. ACM, 2004.

14. N. Guarino and P. Giaretta. Ontologies and knowledge bases: Towards a terminological clarification. In N. J. I. Mars, editor, *Towards Very Large Knowledge Bases*, pages 25–32. IOS Press, Amsterdam, 1995.

15. D. O. Hebb. *The Organization of Behavior*. John Wiley Sons, 1949.

16. K. Henricksen and J. Indulska. Personalising context-aware applications. In R. M. et al., editor, *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops, Agia Napa, Cyprus*. Springer, 2005.

17. J. L. Herlocker and J. A. Konstan. Content-independent task-focused recommendation. *IEEE Internet Computing*, 5(6):40–47, 2001.

18. Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1):116–142, 2004.

19. J. W. Kaltz. *An Engineering Method for Adaptive, Context-aware Web Applications*. PhD thesis, Universitaet Duisburg-Essen, Campus Duisburg, 2006.

20. G. Kappel, B. Pröll, W. Retschitzegger, and W. Schwinger. Customisation for ubiquitous web applications a comparison of approaches. *Int. J. Web Eng. Technol*, 1(1):79–111, 2003.

21. A. I. Kovacs and H. Ueno. Recommending in context: A spreading activation model that is independent of the type of recommender system and its contents. In V. Wade, H. Ashman, and B. Smyth, editors, *Proceedings of the AH2006*. Springer, 2006.

22. P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*, pages 187–192, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

23. J. Metz. Kontextadaptive web systeme in der unternehmenskommunikation. Master's thesis, University of Duisburg-Essen, 2007.

24. S. Middleton, N. Shadbolt, and D. D. Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88, 2004.

25. J. Mitchell and B. Shneiderman. Dynamic versus static menus: an exploratory comparison. In *SIGCHI Bull.*, 1989.

26. R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):16–36, 1991.

27. R. Oppermann. From user-adaptive to context-adaptive information systems. *iCom, Zeitschrift für interaktive und kooperative Medien*, 3/2005:4–14, 2005.

28. M. Perkowitz and O. Etzioni. Towards adaptive web sites: Conceptual framework and case study. *Artifical Intelligence*, 118(1-2):245–275, 2000.

29. P. Pirolli and S. K. Card. Information foraging in information access environments. In *CHI*, pages 51–58, 1995.

30. C. Rocha, D. Schwabe, and M. P. de Aragão. A hybrid approach for searching in the semantic web. In *WWW*, pages 374–383, 2004.

31. A. Sears and B. Shneiderman. Split menus: Effectively using selection frequency to organize menus. *ACM Transactions on Computer-Human Interaction*, 1(1):27–51, 1994.

32. S. Weibelzahl and G. Weber. Advantages, opportunities and limits of empirical evaluations: Evaluating adaptive systems. *KI*, 16(3):17–20, 2002.

33. D. Westheide. Spreading-activation based learning for adaptive web applications. Master's thesis, University of Duisburg-Essen, 2007.

34. J. Ziegler, S. Lohmann, and J. W. Kaltz. Kontextmodellierung für adaptive webbasierte systeme. In C. Stary, editor, *Mensch & Computer 2005: Kunst und Wissenschaft*. Oldenbourg Verlag, München, 2005.