Improving collaboration by using context views

Tim Hussein, Werner Gaulke, Timm Linder, and Jürgen Ziegler
University of Duisburg-Essen
Lotharstr. 65, 47057 Duisburg, Germany
{tim.hussein, werner.gaulke, timm.linder, juergen.ziegler}@uni-due.de

ABSTRACT

In this paper, we explain our notion of context, considering for instance membership in a group as context. We derive a model for context-adaptivity from the well-established one for user-adaptivity proposed by Jameson, and introduce *context views* as means for facilitating group-based work. Context views aim at identifying the most important elements within an application in a generic way by exploiting context information.

Author Keywords

Context, Collaboration Support, Adaptivity, Recommender Systems, CSCW

ACM Classification Keywords

D.1.2 Information Systems: Information Storage and Retrieval—*Information Search and Retrieval*

INTRODUCTION

Approaches for context-adaptive collaboration support have been proposed in *user modeling* as well as *context-awareness* research. Unfortunately, not much research has been done in terms of bringing these fields together to supports group-based work in a context-aware and generic fashion. In this paper, we therefore aim at extending a classical model for user-adaptivity by adding context-awareness. Contributions of this paper are

(a) A conceptual model combining user-modeling and context-awareness in a generic fashion.

(b) Context views as a conceptual technique for identifying the most important elements and artifacts within a system with regard to the current context.

(c) A running example illustrating how both contributions can facilitate collaborative work.

After discussing related work in the next section, we present our notion of context and how systems should deal with it, followed by a model for context-adaptive systems that extends Jameson's well-established one for user-adaptivity [17]. After describing a conceptual framework for context-adaptive applications, we introduce *context views* as a generic way of applying arbitrary personalization techniques like recommendation algorithms in such systems and explain how it can be used to support group-based work in a context-adaptive fashion. We conclude this paper with a short summary.

RELATED WORK

During the 1990s, the fields of *adaptive hypermedia* [6, 5] and *recommender systems* [12, 21] emerged, both aiming at automatically adapting software systems according to the user's behavior. These fields produced quite successful solutions – in terms of academic as well as commercial success – and produced effective algorithms for product or content recommendation. Some of them are actually context-aware [1, 15] or provide recommendations for groups [18, 20]. However, introducing inspiring ideas, these approaches are mostly designed for e-commerce and thus are not directly applicable for collaborative work scenarios with the characteristics mentioned above.

A knowledge context model for virtual workgroup support systems has been introduced by Ahn et al. [2]. Again, this approach is not generically applicable. Gross and Prinz propose a context model consisting of events, artifacts, locations, etc. [13], which can, however, only be used to update and visualize awareness information. Kimura [23] or aCAPella [10] use external tools like cameras, microphones, electronic whiteboards, and others. All these approaches work well in their respective scenarios, but focus on real-world interaction. Furthermore, they neither aim at nor offer context-adaptivity for groups in a generic way.

CONTEXT DEFINITIONS

In computer science, the term *context* is usually either defined as an explicit list of (mostly external) influencing factors like location, time, etc. [22], or in a rather broad and abstract way [19, 9]. One of the most frequently cited definition of context was proposed by Dey and Abowd [9].

They define context as any information that can "characterize the situation of an entity" including information about the user and the application. This rather broad definition lets the designer decide what he considers as relevant contextual information and allows to include external data as well as

Caicoll 2010.

information derived from system usage.

Examples for the first kind of information are time, place, weather, etc., and browser history or latest purchases for the latter. A mobile application, for instance, could need the location for adaptation decisions whereas for a desktop application the last interaction could be more important. From this perspective, co-workers or group members are part of the personal context, too. A system should provide means for storing and exploiting all these types of contextual information in a generic way.

ADDING CONTEXT TO USER-ADAPTIVITY

The main goal of adaptive software systems is to be an alternative to the *one-fits-all* approach of traditional software systems by means of (automatic) adjustment to the user's needs, goals or behavioral patterns. A system is regarded as (user-)adaptive, if it is able to interpret the user's input and thereupon performs changes that improve the usability for the user. Jameson [17] illustrates the concept of user-based adaptivity by a model like the one in Figure 1 and defines it like that:

> "A user-adaptive system can be defined as an interactive system that adapts its behavior to individual users on the basis of processes of user model acquisition and application that involve some form of learning, inference, or decision making." – Jameson [17]

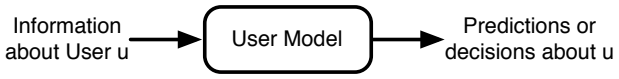


Figure 1. User Adaptivity according to Jameson

Collaborative work often is organized loosely (open source communities for instance) and users may participate in several projects simultaneously. These are, among others, challenges in creating tools for group support. As a consequence, tools aiming at facilitating collaborative work should take the particular context into account in order to determine the users' current goals and give optimal support.

For that purpose, we extend Jameson's model for user-adaptivity to a more general one for context-adaptivity, replacing the user model with a *state model*. This extended model may contain more than just user information (we will illustrate this in an example later). Furthermore, we think that context-adaptive systems, as the name suggests, have to consider the particular view on the situation as well. This leads to a modified model for adaptive systems (see Figure 2).

Depending on the usage *context*, the system should select the most appropriate content and/or features: The system should provide a *view* on the current state, highlighting the most important items, features, etc. with regard to the current context. Providing this is exactly the goal of what we call *context views*. But before explaining this concept, we introduce a generic framework for context-aware applications that builds the foundation for context views.

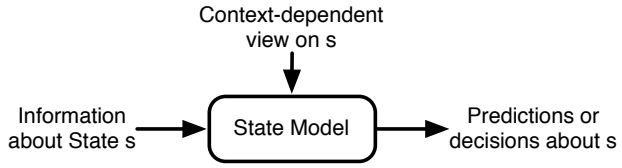


Figure 2. Extending Jameson's model to reflect Context Adaptivity

A GENERIC CONTEXT FRAMEWORK

In this section, we explain our notion of how context-adaptive systems should be organized. For a clear separation of concerns, we propose to distinguish between information on 4 different layers, which are now discussed briefly. A more detailed introduction can be found in an earlier publication of some of the authors [14].

(1) Knowledge Layer

On this layer, conceptual and factual (stable) knowledge about the application and the particular domain is stored (for instance in an ontology). It can be seen as a user- and situation-independent, neutral and objective view of application and domain that includes all information that is not likely to be changed. Information on that layer, can later be used to draw conclusions by applying inference rules (which are to be stored on that layer as well).

(2) State Layer

The state layer contains information about the current situation including information about physical environment, computing environment, resources and user model: Where is the user? What time is it? What are the current circumstances? Considering the constitutional information defined in the knowledge layer, a model representing the current state of usage is being defined. Sensing rules express how both external and internal information sources can be used to take information from outside the application into account as well as information derived from system behavior (User A clicked on Item B). Thereby new objects or properties can be established under the terms defined in the domain model. The single users' states can then be merged into a global state. Figure 3 illustrates an example.

(3) Contextualization Layer

On this layer, information about how to react on certain context conditions should be stored using arbitrary algorithms, like for instance if-then-rules. The result of a contextualization process is a *virtual state* reflecting a view on the state under a certain contextual perspective. In one of the next sections, we introduce *context views* as a generic method for contextualization.

(4) Adaptation Layer

Based upon context views, suitable adaptations defined in the adaptation layer are selected. From a set of adaptation rules, the relevant ones are identified using the virtual (contextualized) state. For instance, if contextualization revealed, that certain files are important for User A, an adaptation rule

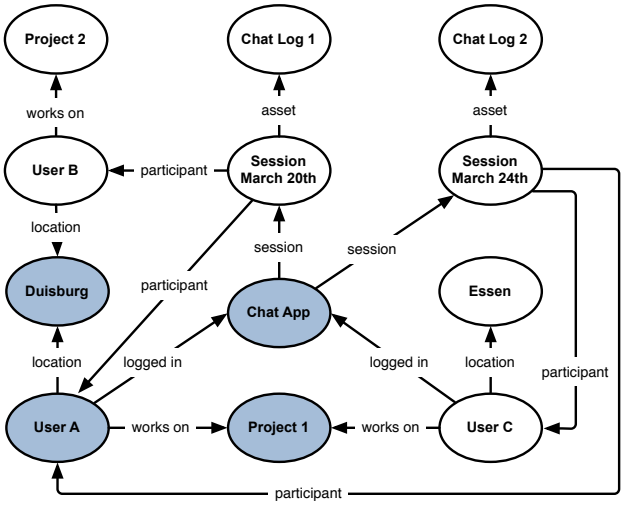


Figure 3. User A's state (filled) is merged with other information to obtain a global state.

could change the user interface so that it offers shortcuts to these files.

APPLYING THE FRAMEWORK FOR GROUP CONTEXT AND ADAPTATION

While the notion of context is inherently multifarious for individual users, it becomes even more complex in the case of group context and adaptation. The framework presented above has already shown to provide a useful basis for structuring and implementing context-adaptive systems for individuals [15]. We claim, however, that it can also be applied and extended to deal with context for groups and collaboration. The different layers can serve to separate and structure the various issues and options of defining group context and adaptation. Concrete methods for context-adaptation could be allocated at a single level or could use combinations of techniques that refer to different levels of the framework.

At the knowledge layer, a primary issue is how to link or merge individual terminologies or conceptual models into shared models. This problem arises, for instance, when different individual tag sets are to be integrated. At a more formal level, personal taxonomies or ontologies might need to be merged into a group model. Various techniques have been proposed, for example, in the ontology engineering community to address this issue. Both approaches can be used to establish a common terminological ground for groups from which a current context can be dynamically extracted to generate, for example, recommendations for relevant resources. Another interesting aspect at this level is the possibility to exchange individual conceptual structures among users. This can the support mutual understanding and might be used to view a set of resources from the perspective of another user or group. In accord with the model of perspective taking/perspective making proposed by Boland and Tenkasi [4] such an exchange of conceptual models can foster learning and cross-group knowledge exchange.

The state layer as defined above captures the current situation of users in their entirety. Integrating individual states can yield group state in various ways: Firstly, explicitly defined relations among group members may be instantiated to express current group-relevant associations among members. Co-location is one example of such explicitly defined relations. When several persons of a group (and relevant resources) are simultaneously at the same physical location, this might be expressed by instantiating a co-location relation among the entities.

How such relations can be detected, e.g. based on sensor input and inferencing techniques, will not be discussed further here. A second approach to merging individual states would be to aggregate state-dependent weights on entities or relations of the knowledge layer. As an example, individual interest profiles derived from interaction behavior and represented as numerical weights over the set of concepts could be aggregated for the group in a summative fashion. More sophisticated techniques might use statistical methods or voting approaches to aggregate states into group state.

At the contextualization layer, the guiding question is to determine which state information is relevant for deciding about potential adaptations to be performed in the next step. In the present paper, this subset of the state information is called a context view. The issue related to group context at this level is therefore how context views for single users can be merged into group context views. Methods to achieve this are currently investigated in our research.

Finally, in the adaptation layer, approaches for merging different individual adaptations have to be considered. For adaptations taking the form of recommendations of documents or Web resources, for example, the problem can be framed in terms of combining different (ranked) lists of recommendations into a single list of items recommended to group as a whole. Research on group recommending has proposed a number of techniques for this purpose, such as averaging across rankings or 'least misery' strategies that try to avoid recommendations that are unacceptable for one of the group members.

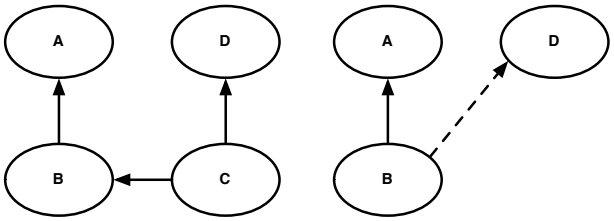
The issues and examples described show that the 4-layer context framework proposed helps to separate different concerns also for group context and adaptation and can support a more systematic approach to designing context-adaptive systems for groups and collaborative work. Since the design space spanned by this framework is very large, more research is planned with respect to providing useful techniques addressing the different levels as well as combinations or patterns of such techniques that are useful in realistic settings.

CONTEXT VIEWS

A truly adaptive system should consider both the users' current state and context information to select the most important entities and concepts with regard to the current situation. Context-views aim at that goal, applying contextualization operations to a state which results in a *virtual state* repre-

senting a certain view on the state. The name context view refers to the view concept in database theory, where a certain sequence of operations leads to a *virtual table* that can thereupon be queried as if it were a regular table. Each view shifts the focus to other information from the state model thus enabling different forms of adaptation.

Context views work similar to layers in image processing applications, but instead of using an image as the bottom layer and and a non-destructive filter layer above, we use the current state as the bottom layer and overlay it with non-destructive context views on top of it. The example in Figure 4 illustrates a (very simple) state and a possible resulting virtual state that might emerge after contextualization.



(a) A simple state. (b) Applying a context view leads to a virtual state.

Figure 4. A state before contextualization and a (virtual) state after contextualization. From this perspective, C is not important (and thus removed), whereas a virtual connection between B and D is generated (dashed) as a result of the contextualization process. Virtual new entities could be added as well (although this is not the case here).

Context, in our understanding, can be virtually anything from external circumstances (location, weather, etc.) to internal resources (click-stream, state of the application, etc.). This rather broad understanding of context goes along with the already mentioned context-definition by Dey and Abowd [9]. Applying the *view* concept to context-adaptivity, we define a context view as a sequence of contextualizing operations leading to a virtual state. With contextualizing operation we mean any mechanism that is able to analyze a certain initial state (with respect to the context if necessary), draw conclusions and create a virtual state thereupon (see Figure 5).

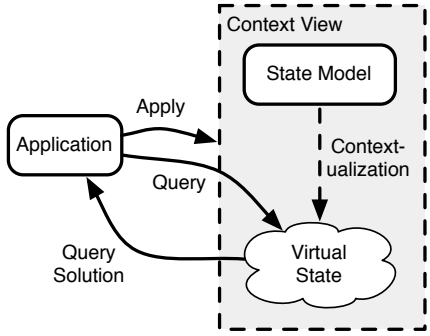


Figure 5. Applying a context view to the state to obtain a virtual state including the most important elements with respect to the current context. These elements are passed to the application when needed (query).

In the *context* of a certain project, a company's chat application could for instance perform operations like this:

```
(1) Identify all users working on the same project
    attending the same chat room.

(2) Mark past chat protocols of sessions, in which
    at least 2 users working on that particular
    project participated.
```

This example highlights the project context, but any other aspect like the current location could be used as well. Applying these contextualization operations in a context view leads to a virtual state as illustrated in Figure 6. The contextualization does not affect the original state at all.

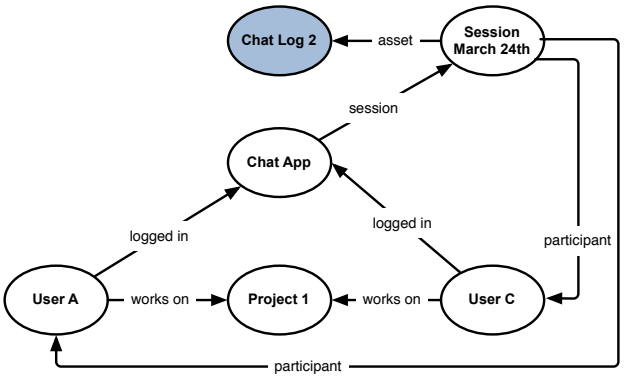


Figure 6. After applying a context view performing the contextualization operations mentioned above, the state from Figure 3 could lead to such a virtual state. The system can now decide what to do with the marked assets (for instance, offer links to them in a sidebar).

Technically, the contextualization process is spawned with a copy of the original state. From a more abstract point of view this means, that only events change the current state. Reasoning, on the other hand, leads to a cognitive model *reflecting* reality but does not change it per se.

ALGORITHMS FOR CONTEXT VIEWS

For contextualization purposes, arbitrary personalization techniques can be used, some of which are now illustrated exemplary.

Rule-based

The example in the previous section used a simple rule-based approach to highlight the most relevant elements. Personalization by means of fixed rules often is a good trade-off between administration and implementation effort on the one hand and good results on the other.

Spreading Activation

Spreading Activation was introduced in the 1970s [7] and originally applied in the fields of psycho linguistics and semantic priming [3]. Spreading activation techniques have successfully been used in several research areas in computer science, most notably in information retrieval [8, 11] or e-commerce [16].

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 adjacent nodes (and the ones related to them and so on). Thereby the flow of activation is attenuated the more it strides away from the initially activated node(s). In the end, several nodes are activated to a certain degree that are semantically related to the elements originally selected.

This technique can directly be adapted to context views as all relevant information already is encoded in a graph. If, for instance, user A and B attend a chat, the nodes representing A and B could be initially activated. The activation would now spread through the graph, increasing adjacent nodes. As a result, exactly those elements that A and B have in common (for instance a project both are working on) would receive the most activation.

Collaborative Filtering

Collaborative filtering methods are supposed to be the most widely implemented recommendation techniques. They can be partitioned into classical User-based-CF [12] and Item-based CF approaches [12]. The basic assumption in both variants is, that those who "agreed" in the past tend to agree again in the future. In e-commerce – a classical application domain for CF – users are regarded as similar, if they have similarities in their past purchases or product ratings.

Transferred to context views and collaborative work, users could be regarded as belonging to the same group, if they have certain features in common. In this case, instead of items in a shopping cart, recently opened documents could for instance be the foundation to compute similarity upon, resulting in something like "users who opened Roadmap for Project X also opened Work Packages of Project X".

Hybrid approaches

Hybrid recommender systems can be applied as well (in fact we did just that in a previous publication [15]). For instance could both algorithms mentioned above be combined by first applying collaborative filtering to identify certain elements and, in a second step, using the the results as initial nodes in a spreading activation process for refining.

These approaches only sketch the basic ideas in order to demonstrate the generic applicability of context views regardless of the algorithm. In reality, things like privacy or rights management have to be considered as well, but the principles and ideas of context views should be clear now.

SUMMARY

In this paper, we explained our notion of the context with regard to group-based systems following rather broad context-definitions by McCarthy or Dey and Abowd. The definition "any information that can be used to characterize the situation of an entity [is context]" leads to the implication that for instance the membership in a certain group can be regarded as context as well – which affects the notion of context-adaptive systems in turn: Not only should context-adaptive systems exploit external context factors like time, location, etc., but also take any other information into account "that can be used to characterize the situation".

With this in mind, we extended the well-established notion of user-adaptivity proposed by Jameson and derived a model for context-adaptive systems. We implemented this model by introducing the concept of *context views*, which can be used to identify the most important elements of a situation with regard to the particular contextual perspective. A system can thereupon use them for adaptation purposes. In the algorithms section, we explained how several well-established techniques can be used for context views, either solely or in combination.

Context views can be used in group-based scenarios (as shown in this paper), but in fact they are meant as a more general concept for arbitrary context-adaptive systems. However, in this paper, we focused on the derivation of the concept as well as its applicability for group-based work. Moreover, we skipped the details of the implementation concentrating on the concept itself. The technical details of the implementation will be presented in our upcoming work as well as the application on other scenarios than group-based work.

ACKNOWLEDGEMENTS

The research presented in this paper is part of the CONTici project funded by the German Research Foundation (Deutsche Forschungsgemeinschaft).

REFERENCES

1. G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*, 23(1):103–145, 2005.

2. H. J. Ahn, H. J. Lee, K. Cho, and S. J. Park. Utilizing knowledge context in virtual collaborative work. *Decis. Support Syst.*, 39(4):563–582, 2005.

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

4. R. Boland and R. Tenkasi. Perspective making and perspective taking in communities of knowing. *Organization Science*, 6(4):350–372, 1995.

5. P. D. Bra, P. Brusilovsky, and G.-J. Houben. Adaptive hypermedia: From systems to framework. *ACM Computing Surveys (CSUR)*, 31(4):12, 12 1999.

6. P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction (UMUAI)*, 6(2-3):87–129, 1996.

7. A. M. Collins and E. F. Loftus. A spreading activation theory of semantic processing. *Psychological Review*, 82(6):407–428, 1975.

8. F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, 1997.

9. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Proceedings of the CHI 2000 Workshop on the What, Who, Where, When, and How of Context-Awareness*, The Hague, Netherlands, 2000. ACM Press.

10. A. K. Dey, R. Hamid, C. Beckmann, I. Li, and D. Hsu. a cappella: Programming by demonstration of context-aware applications. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '04)*, pages 33–40, New York, NY, USA, 2004. ACM.

11. W.-T. Fu and P. Pirolli. Snif-act: a cognitive model of user navigation on the world wide web. *Human-Computer Interaction*, 22(4):355–412, 2007.

12. D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

13. T. Gross and W. Prinz. Modelling shared contexts in cooperative environments: Concept, implementation, and evaluation. *Comput. Supported Coop. Work*, 13(3-4):283–303, 2004.

14. J. Haake, T. Hussein, B. Joop, S. Lukosch, D. Veiel, and J. Ziegler. Context modeling for adaptive collaboration. Technical Report 2, University of Duisburg-Essen, ISSN: 1863-8554, 2009.

15. T. Hussein, T. Linder, W. Gaulke, and J. Ziegler. Context-aware recommendations on rails. In *Workshop on Context-Aware Recommender Systems (CARS-2009) in conjunction with the 3rd ACM Conference on Recommender Systems (ACM RecSys 2009)*, New York, NY, USA, 2009.

16. T. Hussein and J. Ziegler. Adapting web sites by spreading activation in ontologies. In L. Bergman, Kim, Jihie, B. Mobasher, S. Rueger, S. Siersdorfer, S. Sizov, and M. Stolze, editors, *Proceedings of International Workshop on Recommendation and Collaboration*, New York, USA, 2008. ACM.

17. A. Jameson. *Adaptive interfaces and agents*, pages 305–330. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 2002.

18. A. Jameson and B. Smyth. Recommendation to groups. In P. Brusilovsky, A. Kobsa, and W. Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, pages 596–627. Springer, Berlin, 2007.

19. J. McCarthy. Notes on formalizing context. In *Proceedings of the 13th international joint conference on Artifical intelligence (IJCAI'93)*, pages 555–560, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

20. J. A. Recio-Garcia, G. Jimenez-Diaz, A. A. Sanchez-Ruiz, and B. Diaz-Agudo. Personality aware recommendations to groups. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 325–328, New York, NY, USA, 2009. ACM.

21. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *EC '00: Proceedings of the 2nd ACM conference on Electronic commerce*, pages 158–167, New York, NY, USA, 2000. ACM.

22. B. N. Schilit, N. Adams, and R. Want. Context-aware computing applications. In D. Long, editor, *Proceedings of the Workshop on Mobile Computing Systems and Applications (WMCSA 1994)*, pages 85–90, Santa Cruz, CA, USA, 1994. IEEE Computer Society.

23. S. Voida, E. Mynatt, B. MacIntyre, and G. Corso. Integrating virtual and physical context to support knowledge workers. *IEEE Pervasive Computing*, 3(1):73–79, 2002.