Computer Supported Cooperative Work **13:** 283–303, 2004.

Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation

TOM GROSS [1] & WOLFGANG PRINZ [2]
[1] *Faculty of Media, Bauhaus-University Weimar, Bauhausstr. 11, 99423 Weimar, Germany (E-mail: tom.gross@medien.uni-weimar.de);* [2] *Fraunhofer Institute of Applied IT, Schloss Birlinghoven, 53754 St. Augustin, Germany (E-mail: wolfgang.prinz@fit.fraunhofer.de)*

Abstract. Users who work together require adequate information about their cooperative environment: about other group members' presence and activities, about shared artefacts, etc. In the CSCW literature several concepts, prototypes, and systems for providing this group awareness information have been presented. In general, they capture information from the environment, process it, and present it to the users. This paper addresses the processing aspect; in particular, we present a concept for processing awareness information by means of awareness contexts. With this concept we address the problem of contextualising event notifications enabling the presentation of notifications in the appropriate user situation. We describe a lightweight model and its integration into an event and notification infrastructure. We report on an empirical study, and draw some conclusions for the design of context-awareness for cooperative environments.

Key words: contexts, CSCW, evaluation, group awareness, modelling

1. Introduction

In the Computer-Supported Cooperative Work literature it has been emphasised for years that efficient and effective cooperation requires that the cooperating individuals are well informed about their partners' activities (Dourish and Belotti, 1992). They require information about the people they are cooperating with, about their actions, about shared artefacts, and so forth. This information is often referred to as awareness (sometimes with prepositions such as group awareness (Begole et al., 1999; Erickson et al., 1999) or workspace awareness (Gutwin and Greenberg, 1998).

In situations where the cooperating individuals are at the same place this information is often perceived implicitly (Heath and Luff, 1991). Humans thereby apply their natural abilities to process information in the periphery of their attention. For instance, when somebody is working in an office with open doors, the person often is able to capture activity in the hallway without being

disrupted from the task at hand. In other situations where individuals, who are at different places, have to cooperate as a group, technological support for the cooperation process as well as the perception of cooperative activities is essential. The basic parts of this technological support are sensors for capturing the information, indicators for presenting the information, and models for processing the information and providing users with adequate information.

In the literature of Computer-Supported Cooperative Work, Human–Computer Interaction, and Ubiquitous Computing several approaches and systems for *sensors* (e.g. smart-its (Gellersen et al., 2002), or ELVIN (Fitzpatrick et al., 1999)) and for *indicators* (within single applications such as ClearBoard (Ishii et al., 1994), or toolkits such as GroupKit (Roseman and Greenberg, 1996), or in ambient displays such as ambientROOM (Wisneski et al., 1998) have been presented.

Concerning the information *models* some concepts and implementations have been presented in the area of Human–Computer Interaction, and Ubiquitous Computing. Schilit et al. (1994) point out that for context models the following questions have to be answered: where you are, who you are, and what resources are nearby. Later, Dey (2000) developed the Context Toolkit and took the following parameters for modelling contexts: location of the user, identity of the user, time, and environment or activity. Although these models are very elaborated, they primarily address the single-user and do not support cooperative settings and shared contexts.

Some examples of models for cooperative settings are the following: The AREA system describes situations as relationships among objects, where objects are single persons, artefacts, or aggregations such as groups of people. Users can specify which events and artefacts they are interested in and when and in which intensity they want to be informed (Fuchs, 1999; Sohlenkamp et al., 2000). The Atmosphere model (Rittenbruch, 1999) describes contexts as 'spheres'. Users classify their actions on artefacts by means of 'contextors' and map them to specific contexts. When an action is performed a predefined contextor has to be selected. The AETHER model defines the relations between objects using semantical networks (Sandor et al., 1997b). The Model of Modulated Awareness (MoMA) applies a reaction–diffusion metaphor. This metaphor is based on the idea that whenever two or more entities have contact their state is modified in some way. Group awareness is produced and consumed through fields (Simone and Bandini, 2002).

These latter models provide good support for cooperative situations, but have their shortcomings. They become clear, when exposed to some core requirements for good models. Good models should fulfil the following requirements:

- A precise model with close correspondence to the modelled part of the reality.
- A clear mapping of real events and situations to parts of the model.

- Simple and easy modelling and the model's adaptation to changes in the modelled part of the reality.
- No or very little additional effort on the users' side for capturing and presenting the information.

Although these are generic model requirements, they have a specific importance for awareness models. Since awareness needs of users change quickly with their current work situation it is essential that the model allows a precise modelling combined with the ability to be easily configured to changing situations (Heath et al., 2002). The model must therefore provide either means to adopt specific instances to the changing environment or it must adopt itself to changes. Since users wish to concentrate on their (cooperative) work and not on the configuration of awareness tools, the (semi) automatic adoption should be supported to reduce the users' effort.

The AREA-model is quite precise and entails no additional effort for its users, but fails sometimes to map events and situations of the reality to the model. It is difficult to adapt the model to changes in the reality. The Atmosphere model supports detailed modelling and succeeds in mapping events and situations to the model. However, the adaptation of the model is very difficult and the system requires considerable additional effort by the user. For AETHER and MoMA the evaluation is the same: they are very elaborated and provide the most precise modelling of reality, and they entail no additional effort from the users. The mapping of events and situations to the model is in general more adequate than in AREA, but less adequate than in Atmosphere. However, because of the complex models and mechanisms, the adaptation to changes in the reality is extremely high.

In the next section we present our own model for group awareness information, which fulfils the above requirements by clustering information into contexts. We then describe the integration of the model into an awareness information environment. Furthermore, we report on a long-term empirical study of the model and discuss the results providing some general design guidelines for the modelling of contexts for group awareness.

2. Modelling and implementing contexts

In this section we provide a general and a specific definition of group awareness and context, and we describe how contexts can be modelled in a cooperative environment.

2.1. GROUP AWARENESS AND CONTEXTS

In our model contexts are used to cluster information semantically. If we have a closer look at the term context, the Merriam – Webster defines a

context as "(1): the parts of a discourse that surround a word or passage and can throw light on its meaning; (2): the interrelated conditions in which something exists or occurs". In a cooperative setting a context can be defined as the interrelated (i.e. some kind of continuity in the broadest sense) conditions (i.e. circumstances such as time and location) in which something (e.g. a user, a group, an artefact) exists (e.g. presence of a user) or occurs (e.g. an action performed by a human or machine).

Awareness contexts can emerge in various dimensions: geographical contexts and locations such as buildings, floors, offices; organisational contexts such as departments or projects; personal and social contexts like family, close friends; technological contexts such as users of specific technologies (e.g. programmers using Eclipse for their Java programming); action or task contexts such as users who perform similar actions or tasks with similar tools; and so forth. A context can have the following attributes with corresponding values:

- Each awareness context has a unique *name*.
- The *administrator* of a context is the person who sets up and manages the context.
- *Members* of a context are all users who work in a context and who consequently produce events through their actions.
- *Locations*, at which events can be produced, are either electronic (e.g. a shared workspace) or physical areas (e.g. a meeting room).
- The *artefacts* of a context are all objects on which users can operate.
- Each context is associated with various single-user and cooperative *applications* (e.g. text editors, programming environments, groupware applications).
- Events produced in a context are described by their *types*.
- An *access control list* for an awareness context comprises a list with all the rights that exist for each context; each member of an awareness context may have the right to produce events, to subscribe to events or event types, and to decide how she wants the events to be presented. Context-specific ACLs ensure that the members of a context are informed about the events within the context, but that privacy is preserved concerning users who are not members of the context. For each context, context members can define their own privacy policy. This can be seen as an extension of the pure reciprocity that is often claimed (e.g. Borning and Travers, 1991; Tang and Rua, 1994). In some contexts members can agree upon reciprocity, in some they can define other models.
- Each awareness context has various connections to its *environment* and to other contexts (e.g. two projects with one awareness context respectively, which have overlapping membership). Large contexts consisting of many members, many shared artefacts might be spread

over several locations and might be organised in sub-contexts (Agostini et al., 1996).

It is important to note that the context description does not require the specification of values for all attributes. For instance, a context can be created and some attributes like locations or applications are specified later on; or a context could have no locations or no applications at all. Nevertheless, the more details are available for a context, the better events can be matched to it. In many cases the attributes of a context can be generated automatically. For instance, if a context consists of a shared workspace the list of members and artefacts of the context can be dynamically obtained from information about the shared workspace. Furthermore, it is possible to use patterns or predicates over a set of possible attribute values to specify an attribute value.

Departing from this basic notion and characterisation of contexts, we now describe the event and notification infrastructure, and the integration of awareness contexts into it.

2.2. THE EVENT AND NOTIFICATION INFRASTRUCTURE

The Event and Notification Infrastructure (ENI) is a generic extensible awareness environment, which includes simple but powerful and lightweight mechanisms for the generation and user configurable presentation of awareness information at the standard desktop interface (Prinz, 1999). The concept of ENI is based on sensors, events, and indicators. Figure 1 shows the architecture of ENI.

Sensors are associated with actors, shared artefacts, or any other object constituting or influencing a cooperative environment, and generate events related to them. Sensors can capture actions occurring in the electronic space (e.g. changes in documents, presence of people at virtual places) and actions

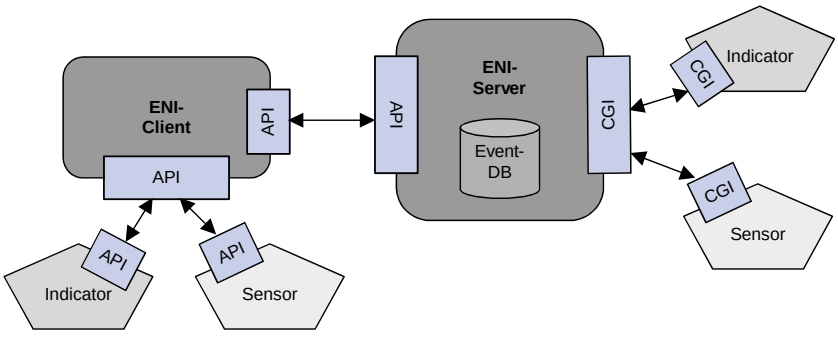

Figure 1. The notification architecture.

in the physical space (e.g. movement or noise in a room). Some examples of sensors we have realised so far are presence sensors capturing for the presence of users; web presence sensors capturing the visits of users on Web sites; a web content watcher capturing updates to specific Web pages (e.g. newspapers on the Web); sensors integrated in office documents, and a shared workspace system.

The generated *events* are sent to the ENI server – either via the application-programming interface (API) of the ENI client, or via a common gateway interface (CGI). They are described as attribute-value tuples. The ENI server stores the events in an event database. This database is realised as a semi-structured database (Abiteboul et al., 1999) using XML formatted tuples. This decision was made to allow for a flexible handling of different event types, which would not be possible in a relational database. The communication protocol between the server and client application is HTTP and the data exchange is XML-based.

Users can use ENI clients to subscribe to events at the ENI server and to specify indicators for the presentation of the awareness information. Subscriptions have the form of event patterns. The client registers these patterns at the server via the API. When the server receives an event that matches the pattern, the event is forwarded to the respective client.

Additionally, users can specify how they want to be informed about the event (i.e. which indicators should be used for the presentation). Various kinds of *indicators* are offered ranging from pop-up windows, to applets in Web pages, to ticker tapes (Fitzpatrick et al., 1998), to 3D graphical presentations in a multi-user environment (Prinz et al., 2003). Cadiz et al. (2002) present numerous examples for the integration of indicators into the windows desktop environment.

On a whole ENI is a flexible tool for the application-independent support of group awareness (Prinz, 1999). A disadvantage that was discovered in everyday use is the immediate notification of the users who have specified interest for a certain event. In many cases users are notified regardless of their current context of work. In order to provide users with proper information at the right time in an adequate quality and quantity, we introduce awareness contexts.

2.3. APPLYING AWARENESS CONTEXTS

For the provision of awareness information it is important that event notifications are presented in the appropriate situation – that is, in the situation in which the information is most relevant to the user.

For those systems in which the awareness information is presented in the context of its origin (e.g. a document), awareness widgets (Gutwin et al., 1996; Sohlenkamp et al., 2000) are often used overlaying the presentation of

a document. Thus, whenever users open the document they immediately see the awareness information attached to the document.

In other cases this problem cannot be solved that straight-forward – for example, when awareness information is not directly coupled with a document or when information shall be presented independently from a document access. In these latter cases the context of origin of an event and the context of work of a user who receives a notification are distinct, which entails the following requirements for context processing:

1. The system has to know or deduce the *context of origin* of an event.
2. The system has to know the current *context of work* of the users who need to be informed.
3. The user has to be able to specify in which *situation* she wants to be informed about events from a specific context in a specific format.

Figure 2 illustrates the processing of events according to these requirements. The left side shows the association of an event with a context; the right side shows the association of a user with a context of work based on his/her current activities. We will describe the three processing steps subsequently.

2.3.1. *Step 1 – Identifying the context of origin*

Events can either be mapped to a context when they are captured at the sensors or when they reach the server. Events can only be mapped to a context at the time of creation, if either the sensor has information about the context specification (from the Context-DB describing the contexts of origin) or if the sensor is used for only one context. In these cases the sensor can immediately add a context attribute to the produced event. However, often this is not the case and therefore we describe an alternative method for the association of a context at the server later in this paper.

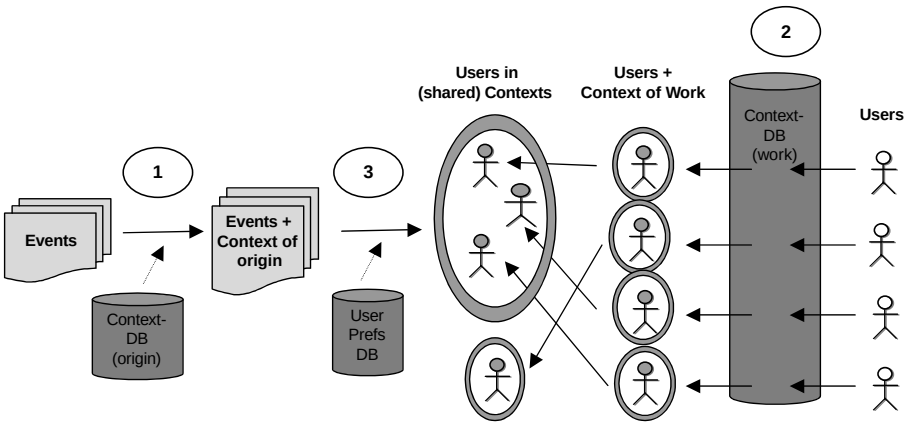

Figure 2. Three steps to bring user in context.

2.3.2. *Step 2 – Identifying the users' context of work*
Once the computed context attribute is added to the respective event the system needs to detect the current context of a user. For this purpose, the interaction of the user with the physical and electronic world is captured and analysed. Fine-grained activities such as typing on the keyboard or passing light barriers have to be aggregated and matched to contexts. That is, the attributes of the events produced by the user are compared to the attributes of the known contexts (from the Context-DB describing the work contexts of users). The result is the selection of a single context or a list of contexts from the context descriptions that matches best with the current activities.

2.3.3. *Step 3 – checking the users' preferences*
Users can specify in which context of work – that is, in which situation – they want to be informed about an event from a specific context of origin. Furthermore, they can specify the format of the presentation of the event information and the schedule for the presentation. For instance, a user can specify that whenever she is in the work context 'Project A' she wants to receive information related to 'Project A' and 'Project B' in a tickertape with a 15 minute interval.

The system knows what is happening in the environment on a whole and in which part the user is involved. The system can then associate the user with a context. Users with similar activities are informed about each other – and are put in a shared context. After presenting these general concepts of awareness contexts and their processing, we will now show how awareness context are applied in ENI.

2.4. AWARENESS CONTEXTS IN ENI

In order to support awareness contexts in ENI two main extensions were made: at the ENI server we added a context module, and at the ENI client we added a situation module. Figure 3 shows the extended ENI architecture.

The context module maps incoming events to a context of origin. It compares attributes of the incoming events with the attributes stored in the context database. Table I shows the attributes of contexts.

Table II shows the mapping of event attributes to context attributes (see also Figure 4 for an example event).

The *sensor* attribute of the event indicates the application, by which the user activity was performed. It is therefore matched with the *context-app* attribute of the context description. The *event-originator* is compared with the people listed as *context members* – that is, it verifies if the user is a member of one of the registered contexts. If a location-based service submits the event (indicated by a special sensor type) it is compared with possible

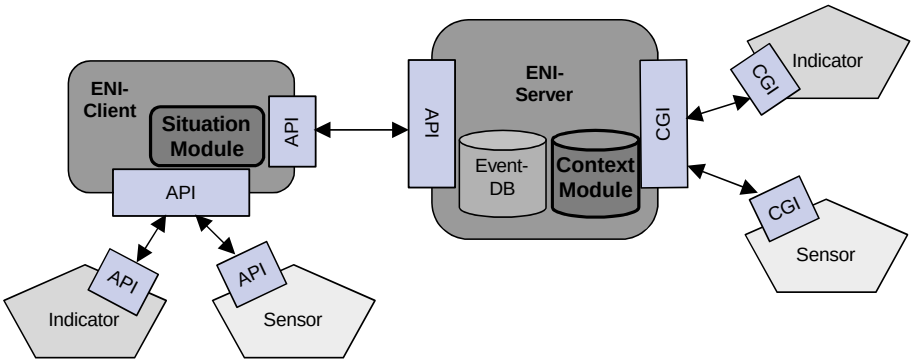

Figure 3. The extended ENI architecture.

context-locations to determine the context based on the location where the activity took place. The *artefact* attribute identifies the object on which the user action took place (e.g. a document or file). In addition, further *event-attributes* are evaluated if the context description contains a specification of additional *context-event* attributes. Examples for such attributes are the operation performed by the event-originator, the folder in which the artefact is stored, or the date and time at which the event was submitted. The latter is useful if the context is valid only during a certain period.

The result of comparing event attributes and context descriptions is a list of one or more contexts that match with the event attributes, combined with a factor indicating the strength of the match. The factor depends on the number and importance of the matching attributes. The importance of each context attribute is described as part of the attribute specification. This result is attached to the event by a context attribute (*event-context*).

Users may use this attribute for the specification of their interest profiles in the ENI client. Thus, the specification of the interest cannot only be based on

Table I. Awareness context attributes

Attribute	Description
context-name	Name of the context
context-admin	Human or non-human actor who created the context
context-member	Human members of a context
context-location	Physical locations related to a context
context-artefact	Artefacts of a context
context-app	Applications related to a context
context-event	Events relevant to a context
context-acl	Access control list of a context
context-env	Related contexts

Table II. Mapping of event attributes to context attributes

Event attributes	Context attributes
sensor	context-app
event-originator	context-member, context-location
artefact	context-artefact
event-attributes	context-event

discrete events, but also on the users' specification of a general interest on events of a specific context. This ensures that a user is informed about future events of a context. There is no need to explicitly describe new event patterns for new events. This addresses a drawback often criticised with event based notification systems (Sandor et al., 1997a).

With an additional element of the interest profile users can specify the situation in which they would like to be informed about an event. Table III shows the options for the timing of the presentation of the awareness information. If users wish to be informed immediately, no scheduling has to be done by the Context Module in the ENI server; if users wish other timing of their notifications, a scheduling component of the context module has to make sure that the users are informed according to their preferences.

These situations are not exclusive – that is, users can combine different schedules. For instance, a user could decide to see events of a specific context

```
<EVENT>
  <ATTRIBUTE type="sensor" value="BSCW"/>
  <ATTRIBUTE type="event-originator" value="Uta.Pankoke"/>
  <ATTRIBUTE type="operation" value="ReadEvent"/>
  <ATTRIBUTE type="artefact" value="weiser_cacm.pdf"/>
  <ATTRIBUTE type="bscw-object-id" value="135578"/>
  <ATTRIBUTE type="folder" value="Related Work"/>
  <ATTRIBUTE type="bscw-folder-id" value="132978"/>
  <ATTRIBUTE type="bscw-class" value="Document"/>
  <ATTRIBUTE type="bscw-content"value="application/pdf"/>
  <ATTRIBUTE type="acl" value="tom.gross, karl-heinz.klein,
              uta.pankoke, wolfgang.prinz"/>
  <ATTRIBUTE type="date" value="2003-02-03 09:09:45"/>
  <ATTRIBUTE type="expires" value="2003-02-04 09:09:45"/>
</EVENT>
```

Figure 4. An example event produced by a user operation in BSCW.

Table III. Schedule of notifications and description of situations

Time	Description
Immediately	An event is presented immediately
In the same context	An event is presented if the interested user is in the context in which the event was generated
Specific context	An event is presented when the user is in a specific context
Date	An event is presented at a defined date and time (e.g. at lunch time)
Age	An event is presented after some time – if it has not been presented because of one of the above rules

any time she is working in that context and additionally at a certain time (e.g. at login).

In order to analyse the current context of work of a user, a situation module is added to the ENI client. This component monitors the current activities of a user and tries to match them to a context. For this purpose, the module uses information from the operating system about running applications and processed objects. This information, as well as the events that are generated by the actions of the user, is also compared to the descriptions of the working contexts. This is done in analogy to the above mapping of events to contexts of origin. And, similarly, the result can contain an ambiguous match with more than one context with different weights. In this case the attributed *context-env* of the attribute description is evaluated. If the contexts are connected, the system assumes that the user is in one of the contexts and the respective events are presented. If no match can be found, a sequence of actions of a user is monitored and the system tries again to match them to a context. When a context of work is found with a probability that exceeds the threshold defined by the user, then the respective events are presented.

3. Method

In this section we will first describe how we applied this context modelling and processing approach to a shared workspace system, and then we will describe some results from a study of its use.

3.1. DEPLOYMENT SETTING

Our model of contexts was applied in a specific Web-based shared workspace system, the Basic Support for Cooperative Work (BSCW) system (Appelt,

1999; OrbiTeam, 2004). The BSCW system allows users with standard Web-browsers to set up nested folders and documents and to share them with others by inviting them to root folders or sub-folders. All the members of a shared workspace (typically a root folder in BSCW) can see all the contents and change it in various ways (e.g. add, delete, or cut, copy, paste folders and documents).

The motivation for applying our context model to BSCW was the following: cooperative work is often organised with the help of shared workspaces (Pankoke-Babatz and Syri, 1997). These shared workspaces specify lists of members and contain shared objects as well as shared applications. If the workspaces of a project are considered as a context, many attributes of a context can be obtained from the workspace description. Thus, actions of the members of shared workspaces can be mapped to contexts likewise. Therefore, we have chosen such an application to validate the applicability of the proposed model. Although this is a specific example, we believe that the results are of general nature, since the considered data is common to almost all shared workspace systems.

A special sensor, which has been integrated into the BSCW server, generates an event in an XML-based format for each user action and forwards it to the ENI server via HTTP. Figure 4 presents an example for a BSCW event.

The *sensor* attribute denotes the application that submitted the event. The *event-originator* attribute contains the user-id of the user who performed the *operation* on a document (*artefact*) in a *folder* (i.e. the containing folder). The complementary *bscw-object-id* and *bscw-folder-id* attributes contain a unique system identifier. The *bscw-class* and *bscw-content*-attributes specify the document (mime)-type. The *acl* attribute lists the user-ids of all users who have access to the artefact. This list is constructed from the list of members who have access to the folder that contains the accessed object. The ENI server uses this list as access control list for the validation of access operations on the event. Therefore, only users who have access to an object in BSCW can read the corresponding events. *Date* contains the date/time of the user action and *expires* the expiration time of the event.

3.2. CONTEXT SUPPORT FOR SHARED WORKSPACES

In order to describe a possible mapping of events to a context we now explain the understanding of a context for this shared workspace application.

Very often users are members of many different workspaces, whereas each workspace is associated with a project, a task, or an organisational unit. For the following, we consider this use and intention of a workspace as the context of this workspace. Often workspaces contain a large number of

folders or subfolders, each containing a number of documents of different types. Thus, we interpret all user actions on these objects as actions that happen in the context of this workspace.

However, we can see from the example in Figure 4, that the events themselves do not contain an indication of the workspace itself. That is, the event does not indicate the context, which it belongs to. This is due to the fact that the BSCW system itself has no notion of a context since this is an external user- or group-specific interpretation of a shared folder. Therefore, it is necessary that the association of a context to an event is computed externally to the application.

Figure 5 shows a screenshot of a BSCW workspace of a project on 'Mobile and Ubiquitous Cooperation Support'. In this example workspace we have four folders (which contain respectively four, nine, one, and nine documents) and three documents. Figure 6 shows a screenshot of the members of this BSCW workspace. This example workspace only has four members.

When a new workspace is created, the BSCW server sends corresponding information to the context module of the ENI server. The context module of the ENI server then creates a context description and starts to query the BSCW server periodically (in general, in a 24 hour interval) for information of all workspaces it knows (the folders and documents, the members, etc.). The BSCW server sends back a list of properties of the respective workspace in XML for each workspace. Figure 7 shows an excerpt of this list (for simplification we only include parts of the folder and document list as well as the user list).

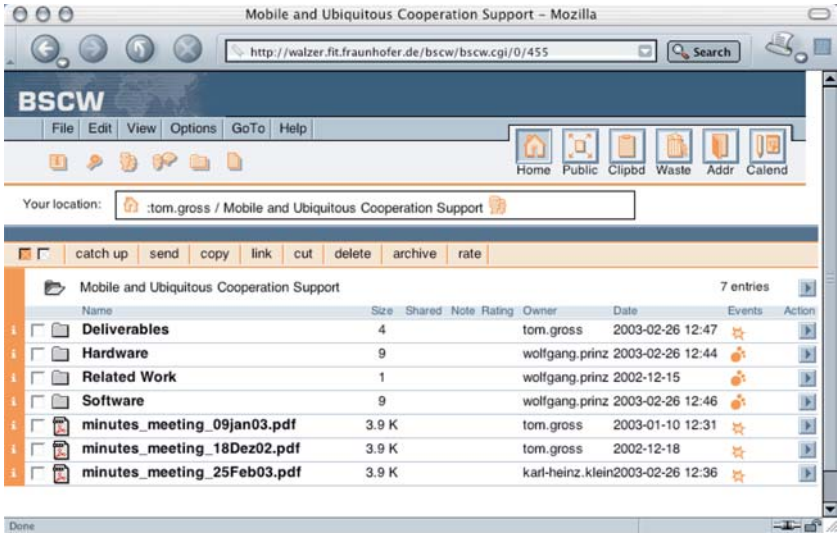

Figure 5. Screenshot of a BSCW workspace (including folders and documents).

Figure 6. Screenshot of the members of this BSCW workspace.

The context module of the ENI server then updates its context descriptions accordingly.

3.3. PARTICIPATION AND DATA COLLECTION

We have created a logging mechanism for the BSCW server. It generates log files containing all events that are produced by user actions over time.

We have captured data over 18 months and analysed the log files subsequently. For a specific (real) user whom we have selected as a test user, it contained 15,800 events, to which the user had access (i.e. in which the user-id was contained in the ACL attribute of the event). From our experiences with the use of BSCW and a comparison of the user's workspaces with that of others, we can consider this user as a representative user.

4. Results

Each event had the structure of the example event shown in Figure 4. The user used the BSCW server to support cooperation processes in eleven different projects or tasks. The workspaces contained between 25 and approximately 1200 different objects, but most contained more than 100 objects. The

```
<Mobile and Ubiquitous Cooperation Support>
   <ATTRIBUTE type="bscw-folder-id" value="455"/>
   <ATTRIBUTE type="date" value="2003-02-03 09:09:45"/>
   …
   <ATTRIBUTE type="artefacts" value="Deliverables, D1.1.doc,
                D1.2.doc, D2.doc, D3.doc, Hardware,
                …
                minutes_meeting_25Feb03.pdf"/>
   <ATTRIBUTE type="members" value="tom.gross,
                karl-heinz.klein, uta.pankoke,
                wolfgang.prinz"/>
   …
</EVENT>
```

Figure 7. Workspace property list from BSCW server (excerpt).

number of members for a workspace ranged from eight to twelve (for 75% of all workspaces) and 30 (seminar group at the university). In the following, we describe our approach for the specification of a context mapping by which each event is associated with one of the eleven different contexts.

The previously presented context model requires the specification of the following parameters (see Table II): context-app, context-member or context-location, context-event, or context-artefact. In all cases of our specific application, the context-app is equal to ''BSCW''; the context-location does not apply since we do not consider real-world locations. Thus, the following mappings remain for comparison, which will be discussed subsequently:

- event-originator attribute to context-member;
- artefact attribute to context-artefact;
- event-attributes to context-event.

The *event-originator attribute to the context-member* mapping allows a selection of a context based on the membership of the event-originator in a certain team or community that is described as the group of people that constitute a context. For a project, this would be the project-members. Analysing the data set, we found, that for our particular user seven out of eleven contexts contain similar membership list (i.e. the list of members differed only in two or three members). These were typically users from outside the local organisation, whereas the overlapping set contained colleagues from the local organisation. This shows that the event-originator attribute is not sufficient for context selection. Since the distribution of events is almost equal over all event-originators, an unambiguous mapping can be made only for less than 20% of all events. This is the percentage of members who are member in just one context.

A more reliable mapping is accomplished by a comparison of the *acl* attribute of an event with the *context-member* attribute of a context. Due to the complete enumeration of the workspace members in this event attribute and the complete listing of the context members in the context-member attribute, a reliable selection of a context is possible. However, this requires that the context-member list is always kept up-to-date with the membership list of the shared workspace. Although automatic update procedures can guarantee this consistency, this approach still has a drawback. Our data shows that three out of the eleven contexts contain the same members. Two of the three contexts are related to each other, thus this ambiguity might not be very problematic from the user's viewpoint. However, the third context covers a different topic than the others. Thus, we need to consider additional criteria for the context selection.

The *artefact attribute* to *context-artefact* comparison enables an unambiguous mapping, but it requires that the context artefact attribute

enumerates all artefacts that constitute the context. Consequently, the context database mirrors almost the complete context data. To avoid this duplication we need to find properties that classify or aggregate artefacts of a context. In our application scenario the *folder,* respectively the *bscw-folder-id* attributes provide such an aggregation. Since the number of folders is much smaller than the number of objects (in our case only 13% of the number of objects), this reduces the number of duplicates by a significant factor.

However, compared to the membership list investigated above, the folders that constitute a context are more dynamic. Therefore, automatic update mechanisms must ensure that the context database is consistent with the actual context data. Since the context module is informed about the creation of a new subfolder by an event, this update mechanism is realised as a simple learning process. Whenever a new folder is created as a subfolder of a folder that is already listed in a context description, this new subfolder is automatically added to the context description. This ensures that the context description is always consistent.

5. Discussion

This examination of the contextualisation approach for notification systems illustrates that the context description as well as the contextualisation of event information requires a careful investigation of the application data. In the presented case different properties of the event information as well as the context description were considered to find a suitable mapping.

In the easiest case, the context is hardwired in the event by the application, such that each event contains an attribute that identifies the context unambiguously. However, this would result in an inflexible solution. The definition of a new context or the modification of an existing context would require an adaptation of all involved applications. Furthermore, such an approach makes it difficult to realise a user specific mapping, in which the same event is mapped to different contexts, because of different user- or group-preferences. For example, one user group might decide that all events, which originate from a workspace containing organisational material (e.g. forms, organisational statistics, etc.) belong to the context ''organisational stuff'', while the user group that produces this information regards this as the project context ''corporate identity''.

Thus, the presented approach of a centralised context module that provides a flexible and lightweight approach to model awareness contexts in a user specific way provides more flexibility. Nevertheless, we have learned that the event to context matching requires a very detailed and specific context description. This result is important for the development of future context based systems since it implies that all activity representation must be as

detailed as possible and that successful matching algorithms must rely on very detailed context description. This relates to our first and second requirement that the model should correspond closely to the modelled part of the reality and that it supports a clear mapping of real event and situations.

However, a very detailed context description is problematic for dynamically changing context data, since it requires continuously updated context descriptions. The solution for this problem is twofold. First, it is important to find categories and aggregations of context data to avoid the necessity to enumerate and thus duplicate workspace information in the context description. The folder-id serves this purpose for the shared workspace application. Second, a simple learning mechanism that automatically updates the context description by interpreting incoming events that contain update information is useful. The proposed model supports this by a simple learning process that automatically updates the context description based on an interpretation of update events received from the shared workspace system. The model thus satisfies also the third and fourth requirement by adapting itself to changes in the modelled part of the reality and by reducing the users' configuration effort. The capturing of event and context information requires no user effort. The actual presentation of event information is not addressed in this paper, but various presentation means that are compatible with the context model presented here can be found in Prinz et al. (2003).

We believe that these findings that were made by applying awareness contexts to BSCW are generalisable and thus also valid for other cooperative applications of contexts.

6. Conclusions

In this paper we contributed a model, an implementation and an examination for the contextualisation of awareness information. We believe that – as it has been said in the area of global information systems like the World-Wide Web – in future it will not only be important that information can be provided at all. Rather, one of the big challenges will lie in the selection of the relevant information, and on creating calm technology. According to Weiser (1997) who coined the term ubiquitous computing we will have increasing access to information and will use an increasing number of tools and gadgets. As a consequence a challenge for the design will be that these tools and gadgets remain in the background and only contact the users with questions or information, when it is really important or relevant to the user. In our opinion, awareness contexts are an interesting step towards this direction.

The evaluation of the context model for the contextualisation of events from a shared workspace system demonstrated the applicability of the model. But, it also indicated that contextualisation requires a careful investigation of

the application to identify properties that permit a unique mapping of events to a context. In cases where such properties cannot be identified, the presented approach allows a graded mapping of events to a context.

Shared workspaces can also be used for sharing context descriptions. The creator of an awareness context uses a shared workspace for the storage and administration of the descriptions. In a shared workspace, the administrator can then specify the members of the context and grant them access to the context description. So, all members of the awareness context can update the context description. For instance, they can add new applications or event types.

As mentioned above user can specify preferences that describe in which situation a user wants to be informed about which context, in what format and by which media the events should be presented, and when events should be presented. Similar to the sharing of context descriptions, shared workspaces can also be used for sharing user preferences. So, the members of a context cannot only share context descriptions, but also their preferences. This allows for context members to be uniformly informed.

We expect that this kind of support for awareness contexts will allow users to establish conventions – which we call 'Context-iquette'. Members of an awareness context can establish conventions for the kind of information that is monitored and also conventions for the presentation of this information. This is a major step towards the protection of privacy – in each context users can find a context-specific solution to this challenge. In one context users might want to have reciprocity; in another context users might accept asymmetry.

Some future challenges are questions of the evolution of contexts. Questions like: who will model awareness contexts?; how will the evolution of these models be supported?; who will be allowed to change the model are very important for the success of awareness contexts?.

Further future challenges lie in the presentation of awareness information. Because users are members of several awareness contexts and want to be informed about several awareness contexts at the same time, we need mechanisms for merging information from different awareness contexts and displaying it. This leads also to a problem of prioritising awareness contexts; that is, it has to be constantly decided which kind of information from which awareness context is to be displayed immediately and which kind of information of which awareness context can be displayed after a delay. Algorithms could calculate the current actuality of an awareness context form information like the number of present users (in absolute figures and relatively to the whole number of members of an awareness context), the fluctuation of an awareness context, the frequency of changes to documents in an awareness context (either with equally important documents or with a hierarchy of importance of documents). Furthermore, the current awareness

context a user is in will vastly influence the type of information to be displayed and also the means of presentation.

Finally, we believe that contexts play a vital role for proper understanding of any kind of information (cf. the above-mentioned first definition of context above). This cannot only support people who already share an awareness context, but also newcomers because the awareness context can be used as guidance of the new users.

Acknowledgements

The research presented here was carried out by the IST-10846 project TOWER, partly funded by the EC. We would like to thank all our colleagues from the TOWER team. In particular with thank our colleague Karl-Heinz Klein for the implementation of the concepts presented in this paper and many useful discussions on the applicability and usefulness of concept itself. We would also like to thank the anonymous reviewers and Christoph Oemig for their comments on earlier versions of this paper. The research was done while both authors were at Fraunhofer FIT.

References

Abiteboul, S., P. Bunemann and D. Suciu (1999): Data on the Web – From Relational to Semistructured Data and XML. Morgan Kaufmann

Agostini, A., G.d. Michelis, M. Grasso, W. Prinz and A. Syri (1996): Contexts, Work Processes, and Workspaces. *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, vol. 5, no. 2–3, pp. 223–250.

Appelt, W. (1999): WWW Based Collaboration with the BSCW System. In SOFSEM'99. Milovy, Czech Republic, Heidelberg: Springer-Verlag.

Begole, J., M.B. Rosson and C.A. Shaffer (1999): Flexible Collaboration Transparency: Supporting Worker Independence in Replicated Application-Sharing Systems. *ACM Transactions on Computer-Human Interaction*, vol. 6, no. 6, pp. 95–132.

Borning, A. and M. Travers (1991): Two Approaches to Casual Interaction Over Computer and Video Networks. In *Proceedings of the Conference on Human Factors in Computing Systems - CHI'91*. Apr. 27–May 2, New Orleans, LO. N.Y.: ACM, pp. 13–20.

Cadiz, J., G. Venolia, G. Jancke, and A. Gupta (2002): Designing and Deploying an Information Awareness Interface. In *Conference on Computer Supported Cooperative Work, CSCW 2002*. New Orleans, New York: ACM Press.

Dey, A.K. (2000): *Providing Architectural Support for Building Context-Aware Applications*. Ph.D. thesis, Georgia Institute of Technology.

Dourish, P. and V. Belotti (1992): Awareness and Coordination in Shared Workspaces. In *Proceedings of the Conference on Computer-Supported Cooperative Work – CSCW'92*. Oct. 31–Nov. 4, Toronto, Canada. ACM, N. Y., pp. 107–114.

Erickson, T., D.N. Smith, W.A. Kellogg, M. Laff and J.T. Richards (1999): Socially Translucent Systems: Social Proxies, Persistent Conversation, and the Design of Babble. In

Proceedings of the Conference on Human Factors in Computing Systems – CHI'99. May 15-20, Philadelphia, PE, New York: ACM, pp. 72–79.

Fitzpatrick, G., T. Mansfield, S. Kaplan, D. Arnold, T. Phelps and B. Segall (1999): Augmenting the Workaday World with Elvin. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work – ECSCW'99*. Sept. 12–16, Copenhagen, Denmark, Dortrecht, NL: Kluwer Academic Publishers, pp. 431–450.

Fitzpatrick, G., S. Parsowith, B. Segall and S. Kaplan (1998): Tickertape: Awareness in a Single Line. In *CHI'98: ACM SIGCHI Conference on Human Factors in Computing*. Los Angeles, CA, New York: ACM Press.

Fuchs, L. (1999): AREA: A Cross-Application Notification Service for Groupware. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work – ECSCW'99*. Sept. 12–16, Copenhagen, Denmark, Dortrecht, NL: Kluwer Academic Publishers, pp. 61–80.

Gellersen, H.W., A. Schmidt and M. Beigl: Multi-Sensor Context-Awareness in Mobile Devices and Smart Artefacts. *Mobile Networks and Applications* vol. 7, no. 5 (2002). pp. 341–351.

Gutwin, C. and S. Greenberg (1998): Design for Individuals, Design for Groups: Tradeoffs Between Power and Workspace Awareness. In *Proceedings of the ACM 1998 Conference on Computer-Supported Cooperative Work – CSCW'98*. Nov. 14–18, Seattle, WA, NewYork: ACM, pp. 207–216.

Gutwin, C., M. Roseman and S. Greenberg (1996): A Usability Study of Awareness Widgets in a Shared Workspace Groupware System. In *CSCW'96: Conference on Computer Supported Cooperative Work*. Nov. 16–20, Boston, MA., New York: ACM Press, pp. 258–267.

Heath, C. and P. Luff (1991): Collaborative Activity and Technological Design: Task Coordination in London, UK Underground Control Rooms. In *Proceedings of the Second European Conference on Computer-Supported Cooperative Work – ECSCW'91*. Sept. 24–27, Amsterdam, NL, Dortrecht, NL: Kluwer Academic Publishers, pp. 65–80.

Heath, C., M.S. Svensson, J. Hindmarsh, P. Luff and D.Vom Lehm (2002): Configuring Awareness. *Computer Supported Cooperative Work: The Journal of Collaborative Computing,* vol. 11, no. 3–4, pp. 317–347.

Ishii, H., M. Kobayashi, and K. Arita: Iterative Design of Seamless Collaboration Media. *Communications of the ACM,* vol. 37, no. 8 (1994) pp. 83–97.

OrbiTeam. BSCW - The Knowledge Management System of OrbiTeam. OrbiTeam Software GmbH, http://www.bscw.de/index_en.html, 2004. (Accessed 23/7/2004).

Pankoke-Babatz, U. and A. Syri (1997): Collaborative Workspaces for Time Deferred Electronic Cooperation. In *GROUP'97: International ACM SIGGROUP Conference on Supporting Group Work*. Phoenix, AZ, New York: ACM Press, pp. 187–196.

Prinz, W. (1999): NESSIE: An Awareness Environment for Cooperative Settings. In *EC-SCW'99: Sixth Conference on Computer Supported Cooperative Work*. Copenhagen, Dordrecht: Kluwer Academic Publishers, pp. 391–410.

Prinz, W., U. Pankoke-Babatz, W. Graether, T. Gross, S. Kolvenbach, and L. Schaefer (2003): Presenting Activity Information in an Inhabited Information Space. In D. Snowdon, E. Churchill and E. Frecon (eds.): *Inhabited Information Spaces: Living with Your Data*. NewYork: Springer-Verlag, pp. 181–208.

Rittenbruch, M. (1999): Atmosphere: Towards Context-Selective Awareness Mechanisms. In *Human-Computer Interaction: Communication, Cooperation and Application Design - HCI'1999*. Aug. 22–26, 1999, Munich, Germany, Hillsdale, NJ: Lawrence Erlbaum.

Roseman, M. and S. Greenberg (1996): Building Real-Time Groupware with GroupKit, A Groupware Toolkit. *ACM Transactions on Computer-Human Interaction*, vol. 3, no. 1 (1996). pp. 66–106.

Sandor, O., C. Bogdan and J. Bowers (1997): Aether: An Awareness Engine for CSCW. In *ECSCW'97: Fifth European Conference on Computer Supported Cooperative Work*. Lancaster, UK, Dordrecht: Kluwer Academic Publishers, pp. 221–236.

Sandor, O., C. Bogdan and J. Bowers (1997b): Aether: An Awareness Engine for CSCW. In *Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work – ECSCW'97*. Sept. 7–11, Lancaster, UK, Dortrecht NL: Kluwer Academic Publishers, pp. 221–236.

Schilit, B.N., N.I. Adams and R. Want (1994): Context-Aware Computing Applications. In *Proceedings of the Workshop on Mobile Computing Systems and Applications*. IEEE Santa Cruz, CA: Computer Society, pp. 85–90.

Simone, C. and S. Bandini (2002): Integrating Awareness in Cooperative Applications through the Reaction-Diffusion Metaphor. *Computer Supported Cooperative Work: The Journal of Collaborative Computing*. Dordrecht: Kluwer Academic Publisher, vol. 11, no. 3–4, pp. 495–530.

Sohlenkamp, M., W. Prinz and L. Fuchs (2000): PoliAwac – Design and Evaluation of an Awareness Enhanced Groupware Client. *AI and Society – Special Issue on CSCW,* vol. 41, no. 1. pp. 31–47.

Tang, J.C. and M. Rua (1994): Montage: Providing Teleproximity for Distributed Groups. In *Proceedings of the Conference on Human Factors in Computing Systems – CHI'94*. Apr. 24–28, Boston, MA, New York: ACM, pp. 37–43.

Weiser, M. and J.S. Brown (1997): The Coming Age of Calm Technology. In P.J. Denning, and R.M. Metcalfe (eds.): *Beyond Calculation: The Next Fifty Years of Computing*. New-York, ACM, pp. 75–85.

Wisneski, C., H. Ishii, A. Dahley, M. Gorbet, S. Brave, B. Ullmer and P. Yarin Ambient Displays (1998): Turning Architectural Space into an Interface between People and Digital Information. In *Proceedings of the First International Workshop on Cooperative Buildings: Integrating Information, Organisation, and Architecture Workshop – CoBuild'98*. Feb. 25–26, Darmstadt, Germany, Heidelberg: Springer-Verlag.