Awareness in Context:
A Light-Weight Approach

Tom Gross, Wolfgang Prinz
Fraunhofer Institute for Applied IT
{tom.gross, wolfgang.prinz}@fit.fraunhofer.de

Abstract. Users who work together require adequate information about their environment—group awareness. In the CSCW literature several models and systems for group awareness have been presented. They basically capture information from the environment, process it, and present it to the users. In general, only the information *per se* is captured without capturing information about its context of origin. Furthermore, the information is then often presented to the users regardless of their current context of work. In this paper we present a light-weight approach for modelling awareness contexts. We describe the concept, report on its examination, and discuss implications for the modelling of contexts and the design of group awareness support.

Introduction

In the CSCW 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 Bellotti, 1992) (Schmidt, 2002). They require information about the other persons 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 automatically (Heath and Luff, 1991). 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. This technological support ranges from workflow management systems to shared workspace systems and other groupware systems. Typically, these systems provide users with information about the members of the group, the shared artefacts, and the process of the group activities. However, most of the approaches and systems only provide this information within the borders of the respective applications.

We have designed and developed an infrastructure that provides group awareness across applications called ENI (Event and Notification Infrastructure). ENI is an event-based awareness environment, which includes various sensors for the capturing of events and various indicators for their presentation. Interviews with application partners and discussions with colleagues who have been using ENI showed that they have the impression that they are better informed than before using ENI. However, several users pointed out that the information was not always provided in the situation when it was of utmost use and that on some occasions the information was to coarse and on other occasions it was to detailed. On a whole the feedback of the users can be summarised to three major requirements for the support of group awareness:

- First, awareness environments should provide awareness information in a way that is adequate for the current situation of the user. The individual user requires personalised information that is adapted to the situation. Besides personal preferences the type of information and its presentation largely depend on the context in which a user is. The context itself depends on parameters like the current task, the current type of cooperation, the artefacts and tools used, and so forth.
- Secondly, awareness environments should not only provide the pure awareness information, but also information about the context of origin of the awareness information. The context in which an event occurs vastly determines its meaning.
- Thirdly, awareness environments should allow users to share awareness contexts. Users with shared interests should be able to share and exchange their awareness profiles.

For the provision of awareness within a closed application such as PoliAwaC (Sohlenkamp et al., 2000) these requirements can be satisfied in an application-specific way. However, for a generic awareness infrastructure a more open concept is required. In this paper we will present such a concept and the implementation of awareness contexts in a generic infrastructure reflecting these requirements. We will shortly introduce the ENI awareness environments, which served as a basis for the realisation of awareness contexts. We will then detail awareness contexts and specify their structure. We will describe how these awareness contexts were integrated into ENI and report on an examination of their quality in terms of effectively identifying actual work contexts. Related work and a summary will conclude this paper.

Event and Notification Infrastructure

The ENI event and notification infrastructure is a generic extendible 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.

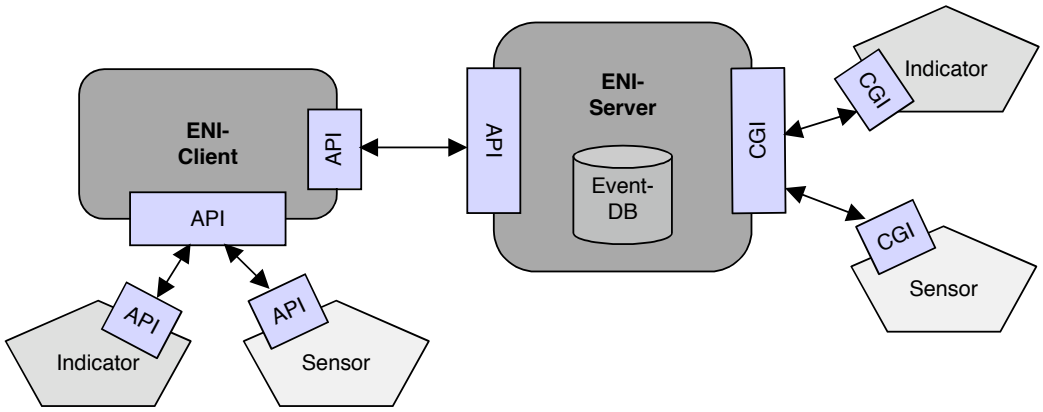

Figure 1. The notification architecture.

Sensors are associated with actors, shared material, or any other object constituting or influencing a cooperative environment and generate events related to them. Sensors can capture actions that take place in the electronic space (e.g., changes in documents, presence of people at virtual places) and actions in the physical space (e.g., movement or noise in a room). Some examples of sensors we have realised so far are presence sensors checking for the presence of users; web presence sensors checking the visits of users on Web sites; a web content watcher checking 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 realized as a semi-structured database (Abiteboul et al., 1999) using XML formatted tuples as the storage format. 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 the 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; that is, which indicators should be used for the presentation.

Indicators are offered in various shapes 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. 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. 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 the right information at the right time in an adequate quality and quantity, we introduce awareness contexts.

Awareness Contexts

In the Merriam-Webster a context is defined 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 this paper we see contexts in the second meaning: in this paper 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, but also clubs, where people are members of; personal and social contexts like family, close friends; technological contexts such as users of specific technologies (e.g., ICQ); action or task contexts such as users who perform similar actions or tasks with similar tools; and so forth.

In order to make context descriptions computable and interpretable by a computing system we decided to represent a context using the following set of attributes (cf. Table I).

These attributes allow the matching of events to contexts of origin and the detection of the current work context of the user. They are described subsequently:

- Each awareness context has a unique *name*.
- The *administrator* of a context is the person who created 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-use and cooperative *applications* (e.g., text editors, programming environments, groupware applications).
- *Events* that are 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 guarantee that the members of a context are informed about the events within the context, but that privacy is kept 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. In some contexts members can agree upon reciprocity in others 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). Big 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).

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 I. Awareness context attributes.

It is important to note that the context description does not require the specification of all attributes. For instance, a context can be created and some attributes like locations or applications are specified only 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 the context. 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 gained from information about the shared workspace. Furthermore, it is possible to use pattern or predicates over a set of possible attribute values to specify an attribute value.

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 the origin (e.g., a document), often awareness widgets (Sohlenkamp et al., 2000) (Gutwin et al., 1996) are used that overlay 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 straightforward, for example when awareness information is not directly coupled to a document or when information shall be presented independent of 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 the *context of origin* of an event or deduce the context of origin;
2. the system has to know the current *context of work* of the users who need to be informed; and
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 illustrates the association of an event with a context. The right side illustrates the association of a user with a context of work based on his/her current activities. We will describe the three processing steps subsequently.

Identifying the Context of Origin (1)

Events can either be mapped to a context when they are produced 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.

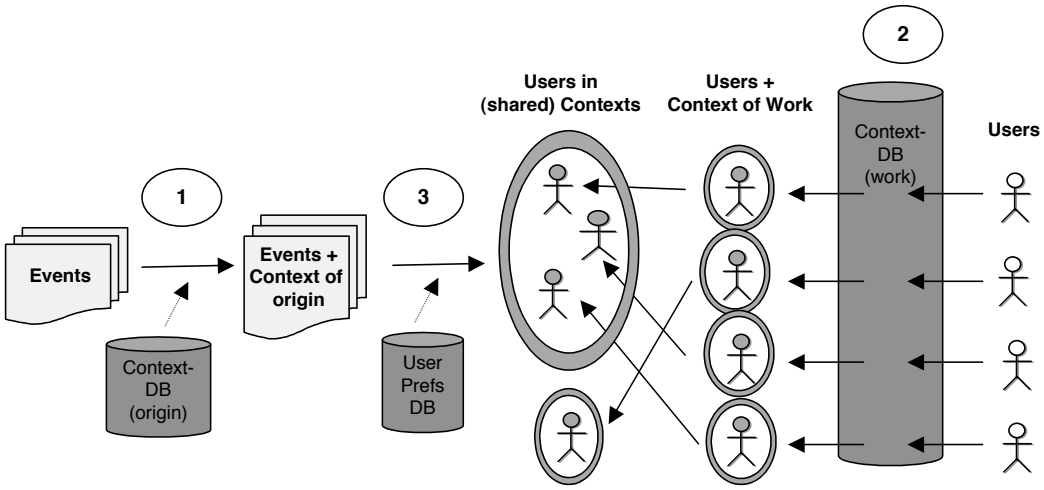

Figure 2. 3 Steps to bring user in context.

Identifying the Users' Context of Work (2)

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.

Checking the Users' Preferences (3)

The users can specify in which context of work, i.e., 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 rhythm.

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

Awareness Contexts in ENI

In order to support awareness contexts in ENI we added two main extensions: 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.

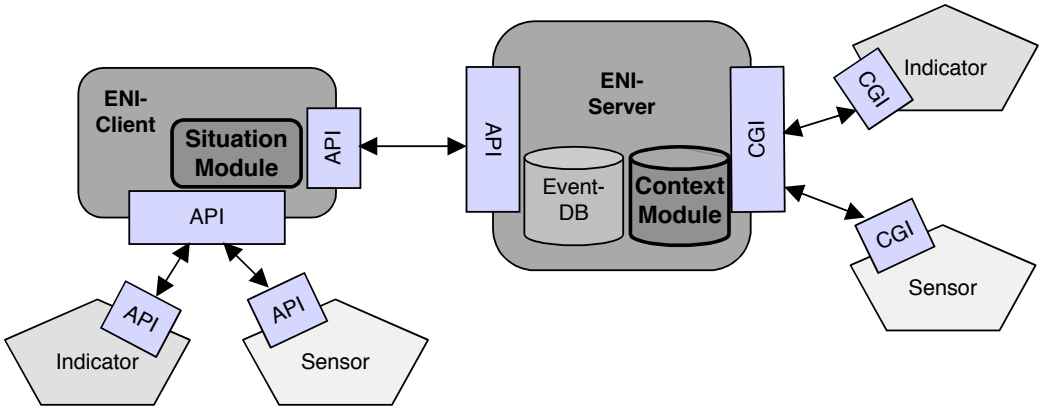

Figure 3. The extended ENI architecture.

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

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

Table II. Mapping of event attributes to context attributes.

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*, i.e., 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 *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 time interval.

The result of the comparison between the event attributes and the context description is a list of one or more contexts that match with the event attributes, combined with a weight indicating the strength of the match. The weight 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 can take this attribute for the specification of their interest profiles in the ENI client. Thus, the specification of the interest cannot only be based on discrete events, but users can also specify a general interest on events of a specific context. This guarantees 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 that is often criticized with event based notification systems (Sandor et al., 1997).

With an additional element of the interest profile users can specify the situation in which they prefer to be informed about an event. Table III shows the options for the timing of the presentation of the awareness information. If users want to be informed immediately, no scheduling has to be done by the Context Module in the ENI-Server; if users want 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.

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

Table III. Schedule of notifications and description of situations.

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

Application and Examination of the Model

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.

Context Support for Shared Workspaces

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 gained from the workspace. 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. We describe experiences and consequences of the application of our model for a specific Web-based shared workspace system, the BSCW system (Appelt, 1999). 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.

We first describe the event information that is provided from the system on each user activity, and then the context descriptions for the mapping of events to specific project contexts. These descriptions are used as a test case for an examination of the proposed context model. We use a data set of approx. 16.000 events gathered over period of 18 month to discuss the suitability of our mapping.

The BSCW server generates for each user action an event that is forwarded to the ENI server via http in XML-based format. For this purpose a special BSCW sensor has been realized in the BSCW server. 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.

```
<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.

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. It is common that a user is member of many different workspaces, whereas each workspace is associated with a project, a task, or an organizational 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 to which it belongs. This is due to the fact that BSCW system itself has no notion of a context since this is an external user- or group-specific interpretation of a shared folder. It is therefore necessary that the association of a context to an event is computed external to the application.

Figure 6 shows a screenshot of a BSCW workspace of a project on ‘Mobile and Ubiquitous Cooperation Support’. In this example workspace we have 4 folders (which contain respectively 4, 9, 1, and 9 documents) and 3 documents. Figure 7 shows a screenshot of the members of this BSCW workspace. This example workspace only has 4 members.

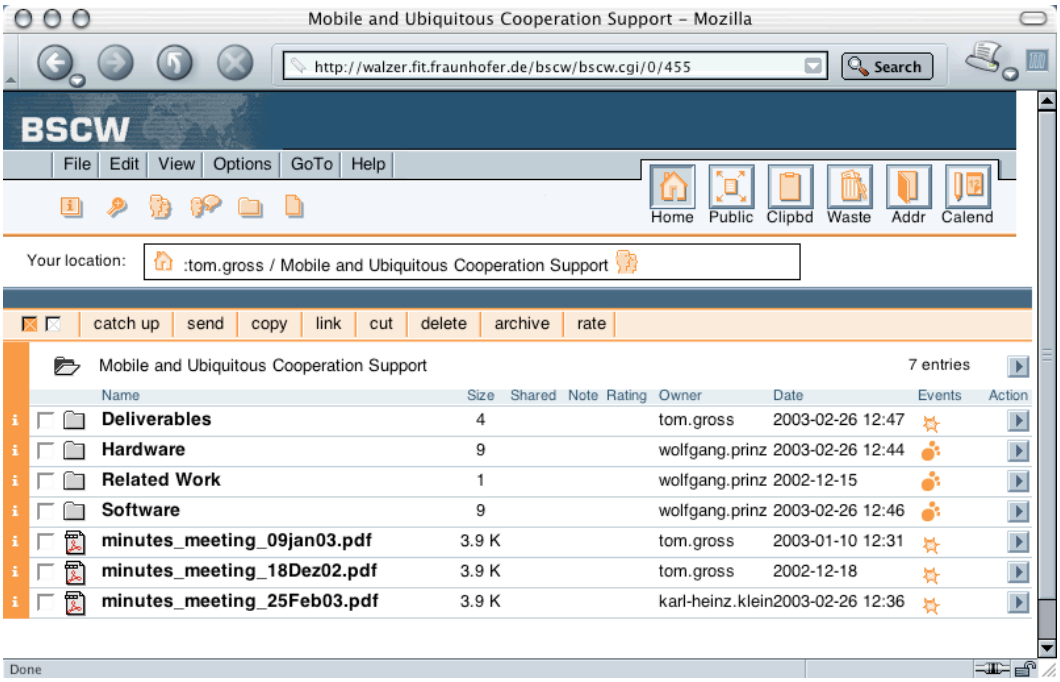

Figure 6. Screenshot of a BSCW workspace (including folders and documents).

Figure 7. Screenshot of the members of this BSCW workspace.

When a new workspace is created, the BSCW server send 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 periodically (in general, in a 24 hour interval) to query the BSCW server for information of all workspaces it knows (the folders and documents, the members, etc.). The BSCW server sends back for each workspace a list of properties of the respective workspace in XML. Figure 8 shows an excerpt of this list (for simplification we only include part of the folder and document list as well as the user list).

```
<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 8: Workspace property list from BSCW server (excerpt).

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

Examination of the Model

In order to find an expressive description for the event-context mapping, we have analysed events that were produced by user operations over a period of 18 months. For a specific (real) user whom we have selected as a test user, the ENI server 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 BSCW-usage and a comparison of the user's workspaces with that of others, we can consider this user as a representative user.

Each event has the structure of the example event shown in Figure 4. The user uses the BSCW server to support cooperation processes in 11 different projects or tasks. The workspaces contained between 25 and approx. 1200 different objects, but most contained more than 100 objects. The number of members for a workspace ranged between 8 to 12 (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 11 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 7 out of 11 contexts contain similar membership list (i.e., the list of members differed only in 2 or 3 members). These typically were 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 a 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 3 out of the 11 contexts contain the same members. Two of the three contexts a 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 lower 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.

Some lessons learned

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 unanimously. 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.) belongs 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. However, a very detailed context description is problematic for dynamically changing context data, since it requires continuously updated context description. 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.

Context sharing

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 which format and by which media the events should be presented, and when they 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 is a means 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 want to accept asymmetry.

Related Work

The AREA system offers similar group awareness support to our system. Situations are described as relationships among objects. 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).

As opposed to the ENI system the AREA system is based on an object-relationship model that is application specific and requires detailed specifications. The modelling of contexts with attribute-value pairs in ENI allows a simple and easy adaptation. Furthermore, any number of new attributes can be added. Therefore, one event can be matched with any number of contexts. In AREA events are caused by actions within the relationship model. In ENI situations relate to actions of users—that is, the system analyses the actions of users and tries to identify the context in which a user is in, instead of requiring a pre-specification of object and event relations.

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 pre-defined contextor has to be selected. Consequently, the model offers a better quality of event information (e.g., write report, instead of simply open report), but requires the users to explicitly specify the respective contextor. As opposed to ENI Atmosphere is based on a detailed and static description of relationships among artefacts, objects, and actions.

Besides the above-mentioned event-based models, spatial models have been presented. In spatial models awareness between objects—persons or artefacts—is calculated by means of the distance between them in a medium. Objects are surrounded by the aura, which can be seen as an area in which objects can be

perceived. At the same time all objects have a focus; that is, an area which they can capture. Mutual overlaps between auras and foci determine mutual awareness of objects (Benford and Fahlén, 1993) and can be seen as spatial context. Spatial models are, in general, only used for synchronous group awareness, because presence of several users at the same time is necessary. The group awareness is mainly calculated based on the distance between objects and does not respect individual interests of users.

The AETHER model can be seen as an extension of the spatial model; the concepts of aura, focus and medium are also used, although in a slightly modified manner. The AETHER model defines the relations between objects with a semantical network (Sandor et al., 1997). The Model of Modulated Awareness (MoMA) is based on 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). Both models, AETHER and MoMA are rather sophisticated, but have the disadvantage that their setup is complicated and that adaptation is hardly possible.

ELVIN (Fitzpatrick et al., 1999) and NSTP (Patterson et al., 1996) are event notification systems similar to our prior version of ENI—that is, they are client/server infrastructures that capture events and present events regardless of the context of origin and the users' context of work.

In the area of knowledge management complex ontologies are used to model the contexts of information seekers in order to improve search results. Whereas these ontologies provide very detailed and adequate models, they are hard to adapt and, therefore, too rigid for the support of dynamic group processes (Gross and Klemke, 2002).

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 WWW—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. 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.

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 persons 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.

References

Abiteboul, Serge, Peter Bunemann, and Dan Suciu (1999): Data on the Web - From Relational to Semistructured Data and XMLMorgan Kaufmann.

Agostini, A., G. de 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, Wolfgang (1999). WWW Based Collaboration with the BSCW System SOFSEM'99, Milovy, Czech Republic. Springer Lecture Notes in Computer Science 1725, pp. 66-78.

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 (TOCHI), vol. 6, no. 6, pp. 95-132.

Benford, Steve and Lennart Fahlén (1993). A Spatial Model of Interaction in Large Virtual Environments In G. d. Michelis, C. Simone, and K. Schmidt (eds.): Third European Conference on Computer Supported Cooperative Work - ECSCW ´93, Milan. Kluwer, pp. 109-124.

Cadiz, Jonathan, Gina Venolia, Gavin Jancke, and Anoop Gupta (2002). Designing and Deploying an Information Awareness Interface Conference on Computer Supported Cooperative Work, CSCW 2002, New Orleans. ACM Press.

Dourish, Paul and V. Bellotti (1992). Awareness and Coordination in Shared Workspaces In J. Turner and R. Kraut (eds.): CSCW ´92 - Sharing Perspectives, Toronto, Canada. ACM Press, 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. Proceedings of the Conference on Human Factors in Computing Systems - CHI'99, Philadelphia, PE. ACM press, pp. 72 -79.

Fitzpatrick, Geraldine, Tim Mansfield, Simon Kaplan, David Arnold, Ted Phelps, and Bill Segall (1999). Augmenting the Workaday World with Elvin In S. Bødker, M. Kyng, and K. Schmidt (eds.): ECSCW'99: Sixth Conference on Computer Supported Cooperative Work, Copenhagen. Kluwer Academic Publishers, pp. 431-450.

Fitzpatrick, Geraldine, Sara Parsowith, Bill Segall, and Simon Kaplan (1998). Tickertape: Awareness in a Single Line CHI'98: ACM SIGCHI Conference on Human Factors in Computing, Los Angeles, CA. ACM Press.

Fuchs, Ludwin (1999). AREA: A cross-application notification service for groupware In S. Bødker, M. Kyng, and K. Schmidt (eds.): ECSCW'99: Sixth Conference on Computer Supported Cooperative Work, Copenhagen. Kluwer Academic Publishers, pp. 61-80.

Gross, Tom and Roland Klemke (2002). Context Modelling for Information Retrieval - Requirements and Approaches In P. Isaias (ed.): Proceedings of the IADIS International Conference - WWW/Internet 2002, Lisbon, Portugal. IADIS Press, pp. 247-254.

Gutwin, Carl and Saul Greenberg (1998). Design for lndividuals, Design for Groups: Tradeoffs between Power and Workspace Awareness In S. Poltrock and J. Grudin (eds.): CSCW '98 Computer Supported Cooperative Work, Seattle, WA. ACM Press NY, pp. 207-216.

Gutwin, Carl, Mark Roseman, and Saul Greenberg (1996). A Usability Study of Awareness Widgets in a Shared Workspace Groupware System In M. S. Ackermann (ed.): CSCW'96: Conference on Computer Supported Cooperative Work, Boston, MA. ACM Press, pp. 258-267.

Heath, C. and P. Luff (1991). Collaborative Activity and Technological Design: Task Coordination in London Underground Control Rooms In L. Bannon, M. Robinson, and K. Schmidt (eds.): Third European Conference on Computer Supported Cooperative Work, Amsterdam. Kluwer, pp. 65-80.

Pankoke-Babatz, Uta and Anja Syri (1997). Collaborative Workspaces for Time Deferred Electronic Cooperation In S. Hayne and W. Prinz (eds.): GROUP'97: International ACM SIGGROUP Conference on Supporting Group Work, Phoenix, AZ. ACM Press, pp. 187-196.

Patterson, John F., Mark Day, and Jakov Kucan (1996). Notification Servers for Synchronous Groupware In M. S. Ackermann (ed.): Conference on Computer Supported Cooperative Work (CSCW'96), Boston, MA. ACM Press, pp. 122-129.

Prinz, Wolfgang (1999). NESSIE: An Awareness Environment for Cooperative Settings In S. Bødker, M. Kyng, and K. Schmidt (eds.): ECSCW'99: Sixth Conference on Computer Supported Cooperative Work, Copenhagen. Kluwer Academic Publishers, pp. 391-410.

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

Sandor, Ovidiu, Christian Bogdan, and John Bowers (1997). Aether: An Awareness Engine for CSCW In H. Hughes, W. Prinz, T. Rodden, and K. Schmidt (eds.): ECSCW'97: Fifth European Conference on Computer Supported Cooperative Work, Lancaster, UK. Kluwer Academic Publishers, pp. 221-236.

Schmidt, Kjeld (2002): The Problem with Awareness: Introductory Remarks on Awareness in CSCW. Computer Supported Cooperative Work: The Journal of Collaborative Computing (Kluwer Academic Publ., Dordrecht), vol. 11, no. 3-4, pp. 285-298.

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

Sohlenkamp, Markus, Wolfgang Prinz, and Ludwin Fuchs (2000): PoliAwac - Design and Evaluation of an Awareness Enhanced Groupware Client. AI and Society - Special Issue on CSCW, vol. 14, no. 1, pp. 31-47.