UNITE – an Agent-Oriented Teamwork Environment

Dr. Michael Zapf, Rolf Reinema, Ruben Wolf, and Sven Türpe

Fraunhofer Institute for Secure Telecooperation (SIT), Darmstadt, Germany

Abstract. Providing seamless, fluid interaction between team members, contractors, consultants from any world-wide location stimulates virtual organisations. The agent-based team work environment UNITE offers team members intuitive and ubiquitous access to each other, and to information and resources of their project, secure and transparent to their physical workplaces and their own tools. Thus, team focus will remain wholly on the project. In this article we will provide informations about the underlying architectural model, the application of agents in particular, and show how we integrated external tools.

1 Introduction

The UNITE (Ubiquitous and Integrated Teamwork Environment) project acknowledges the growing need for computerised collaboration environments supporting teams working on specific projects, in particular when team members are located at different and, over time, changing places. It is essential that such environments allow team members to switch between different projects, make it easy for them to preserve and restore the work context of any given project, allows them to reserve and tailor physical workplaces as team members move and change places, and last but not least be sufficiently secured.

UNITE aims to do research and development on co-operative workplaces and their creation from a unified co-operation platform. This platform is the key system component which provides the facilities for devices, components, and networks to fully interact, despite the possible original inherent heterogenity, and which takes care that a uniform and ubiquitous view is presented to all team members regardless of their physical location.

The overall objectives of the UNITE project are as follows:

1. Identify, document, and verify the overall requirements of future project oriented team work and develop for it a co-operative workplace.
2. Define a suitable architecture of the co-operation platform, based on the above paradigm.
3. Validate the paradigm of a co-operative workplace and the architecture of the co-operation platform. To this end, a prototype is developed and evaluated by means of realistic trials involving real users and teams.

This article is structured as follows. In section 2, we give some information about the underlying paradigms which led to the architecture of the UNITE platform, described in section 3. This section will also present some basics of the agent system AMETAS which serves as the base for implementing the core platform. Section 4 provides some more in-depth information of the implementation issues, and finally section 5 offers our conclusions.

2 Requirements and Paradigm

The UNITE Platform is an innovative approach to merge the world of virtual team work with real-life team work. With projects spanning multiple institutions, we identified a clear need of defining the idea of teamwork in a distributed environment.

2.1 Distributed Teamwork

In many locations, workplaces are no longer allocated to a specific person but are flexibly assigned as required. There is an increased focus on team work world-wide, and societies without a previously well-developed teamwork ethic are joining this trend. For this reason, in addition to the emergence of mobile and tele-working forms of occupation, offices will continue to gain in importance as meeting points.

The overall concept and architecture of the office is still pretty traditional and does not consider how to accommodate the needs of modern organizational structures of work (such as networked structures, projects, and virtual organizations), to address increased employees' responsibility and diversity of tasks, and to fully reap the benefits of new technologies. The only changes in the working environment that have occurred have resulted from the need of telecommuters for home offices.

However, to appropriately implement new methods of work, a holistic environment is needed which can fully support these methods effectively while giving people at work the resources that they need to perform their duties, and convenient means for interacting with each other.

Traditional, work-sharing organisations are not suitable for reacting dynamically, flexibly, quickly and economically enough for dealing adequately with permanently changing customer demands. New paradigms of work organization ask for room structures and furnishings which meet the requirements of teams and which contain dynamically re-configurable teamwork environments, individual workplaces, conference rooms, and communication islands. There are many other important factors which have to be considered in developing future workspaces like

- driving the transition to virtual team-based structures,
- managing virtual project-teams, or
- dynamic sharing of resources, such as e.g. office spaces.

With respect to the physical workspaces, intelligent buildings with flexible building structures must be considered. Last but not least, social impacts have to be carefully investigated, for example the increasing demands on private time and space, the blurring of boundaries between private and professional zones, and integration and coordination of distributed work across boundaries.

One of the objectives of the UNITE project is to define a new paradigm, namely the *co-operative workplace*, aiming at supporting communication and collaboration in distributed project teams and to put it into practice by developing a *co-operation platform*.

2.2 Work Contexts

UNITE focuses on the idea of supporting team work in distributed teams. People working in teams may participate in different projects. They usually organize their way of working by filtering E-Mail messages, putting them in different mail folders, using various document repositories and so on. Behind this organization strategy lies the idea of sets of resources that are associated to a project.

The virtual workspace is where co-operation among project team members occur as a whole. The physical workspace is where individual or group work physically takes place. Hence, there is exactly one virtual workspace per project while physical workspaces may be assigned to different projects at different times.

Conceptually, each project has a virtual workspace of its own, and anyone who is a member of several projects would go from one virtual workspace to another as he or she goes to work from one project to another. The co-operative workplace integrates physical and virtual workspaces under a unified user interface. Physical and virtual workspaces are combined under a common metaphor, so that one can act equally in either one and make a seamless transition from one to the other.

Entering a context means that a user wants to have access to the respective resources, filtering out those that are connected to other projects. This means that instead of remembering the mailbox to use, the calender to access, the number to dial, the document repository to utilize, the user should be presented the correct set just by selecting the project.

While in a team context, users move among the team resources in a virtual way and are accessible by other team members. If a user wants to communicate with another user, they may create an ad-hoc meeting. This meeting is defined by the participating people and by the resources that are required. One example could be a phone conference, a simple text chat session, or advanced groupware tools like application sharing.

2.3 Agent-Orientation

Since the mid-nineties of the last century, the new paradigm of agent-oriented software development received an ever-growing interest for application on various scenarios. The definition of an agent is not as concise as other definitions because agenthood does not only relate to implementing special components called agents but also to design the architecture in a certain way. Different research groups, especially from the area of artificial intelligence on one side and distributed systems on the other side, assessed specific properties of agents to be most important, thereby coming to diverging specifications.

In UNITE, we apply autonomous agents for several purposes:

1. *Entity modelling*: Users, teams, and resources are represented by agents.
2. *Distribution handling*: The distributed nature of the platform requires mechanisms for information transport from one network location to another.
3. *Remote execution*: In order to negotiate communication parameters, not only parameters need to be transported, but also code to be executed remotely.
4. *Autonomy*: The agents shall decide how to contact their respective real-world counterpart; for example, the agents representing users may decide to send an SMS notification whenever special events happened during the user's absence.

3 The UNITE Architecture

At first we give a short view on the agent system AMETAS which is the base of our platform implementation. After that, some details on the platform architecture are presented.

3.1 The Agent System AMETAS

AMETAS (Asynchronous Message Transfer Agent System) [6] is a platform for the development of autonomous and mobile agents. AMETAS was designed and implemented at the University of Frankfurt/Main, Germany.

In the agent system, three kinds of components are defined: *Agents*, *user adapters*, and *services*. All three kinds of components share the same message exchange mechanism and are called *Place Users*. In order to send a message to another Place User, the message must be created by the sender and submitted to the infrastructure (the *place*) which then deposits the message in the recipient's mailbox. The recipient has to retrieve the message by itself; it may be notified that a new message has arrived. Note that it is up to the recipient whether and when to retrieve a message.

Places are the execution environments for Place Users. Each network node which is part of the agent environment should run an own place. Mobile agents may then decide to migrate to these places in order to access the specific services or to meet other agents.

AMETAS abstracts from the details of external facilities by introducing the services which provide access to the outside world. On the other hand, agents are not allowed to access these resources by themselves. Thus, the only way to communicate with an agent or to have the agent change its environment is by means of asynchronous messages. Services may translate these messages to database accesses, file system accesses, socket communication and so on.

Individual users may be coupled to the agent system using the so-called *user adapters*. A user adapter may consist of a graphical user interface which translates user actions to messages. In the other direction, messages from other Place Users are translated to graphical output. From an agent's point of view, a user adapter looks like any other agent, so the user interaction is modelled similar to any other inter-agent interaction.

Place Users are strictly separated from each other by the infrastructure. Each Place User is driven by one or more separate threads in the associated threadgroup. Furthermore, separate class loaders are utilized. It is not possible to pass references between Place Users so that the decoupling is ensured.

Despite the autonomy, agents of one application are usually designed to cooperate, similarly to a team of individual humans working together.

3.2 The UNITE Platform as an Agent-Based System

UNITE employs an inherent feature of the AMETAS platform to support virtual teams. A virtual team is formed temporarily out of members of several separate organisations. Each participating organisation remains independent, and may participate in multiple virtual teams. Besides participating in virtual teams each organisation still has its private projects and infrastructure. This separation may most easily be provided by introducing several kinds of AMETAS places.

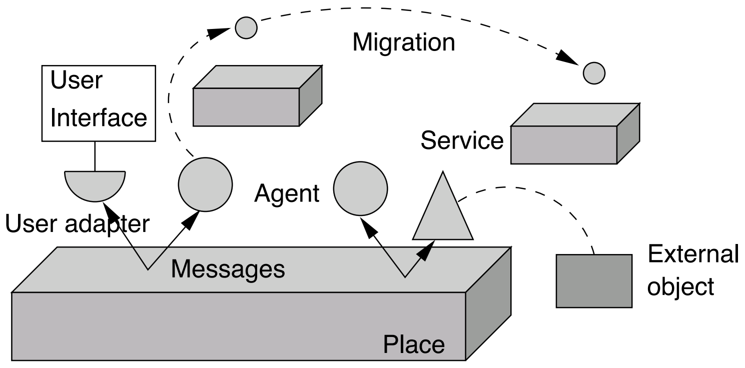


Fig. 1. AMETAS structure

Places AMETAS provides places as the execution environment for agents and agent migration as a communications mechanism between them. Places are independent of each other and all settings and security policies are defined locally. Thus, inherently, AMETAS supports co-operative operations of the infrastructure with a clear separation of ownership and responsibility.

Within UNITE, three types of places are distinguished: *homebases*, *team places*, and *room places*. On homebases, users and their personal tools, services, and data are represented. A team place represents one particular team; distinct teams use distinct team places. Thus, the independence of places is maintained within the UNITE platform. Homebases are independent of team places, and team places are independent of each other. There are no relationships between homebases, or between team places. Relationships between homebases and team places are established dynamically as users start or cease to work in teams.

Generally, a homebases represents a set of users and can thus be used to represent an organization. A team place, being an independent entity, can be run by an organization participating in the respective project or by an independent third party.

Place Users The AMETAS places provide the architectural base for the UNITE platform. Each of these places has specific Place Users running on them.

Homebase At the homebase, Personal Agents are maintained of all users who take this homebase as their UNITE login access point. The Personal Agents are launched automatically when starting the place, and they continue running after the user logged out again.

Furthermore, to allow access, a specific service (described below) is provided to allow users to log in with their web browser. This so-called *UI service* gets the user actions, translates them to messages, and directs them to the respective Personal Agent. Vice versa, messages coming back from the Personal Agent as a result of the action are translated into a suitable output frame to be displayed by the user's browser.

Auxiliary services may be found at the homebase which provide access to features like the user's private calendar, or the own document repository. As described above, these services are required to bridge the abstract world of AMETAS services with the concrete service implementations.

Team place Similar to homebases hosting the Personal Agents, the team place hosts Team Agent as well as the so-called Personal Representatives which are sent by the respective Personal Agents when the user enters a project context.

This team agent is responsible for granting access of Personal Representatives which have migrated to this team place as a consequence of users entering this team context. On the team place, similar auxiliary services may be found which allow to access the team address book, the team calender and so on.

Again, there is a UI service which allows users to directly contact the team place. During our design it showed that having a separate web access point into the team is preferred to handling everything by one single access point.

Room place The third kind of place hosts the so-called room agents and device agents. These kinds of agents provide interact with a room and facilities management infrastructure which allows to allocate rooms and facilities for real meetings which happen in the course of virtual meetings. The current version of the UNITE platform, however, does not yet support rooms and devices management, but this feature has been selected to be important for future extensions.

3.3 Capabilities and preferences

One of the key features of the UNITE platform is to allow users to log in from various locations, not only from the standard office workplace. Furthermore, different devices may be used for accessing.

Many of the tools used for synchronous communication depend on all participants using the same or at least a compatible client program. However, some of these tools are probably not available at the client's side, or they are not executable because of the different capabilities of the execution environment.

Therefore, UNITE defines the notion of *capabilities* and *preferences*. Capabilities are related to the set of possible tools which are available at the client's side, while preferences are related to the user's desires which tool should be used while communicating with him. While the user may have constant preferences (e.g. preferring to use text chat instead of phone communication), the capabilities change over time and for each login incident.

3.4 Role of the mobile agents

The set of capabilities is managed by the Personal Agent. When the user joins a team, these data must be propagated to the team place. This is done by sending the mobile agent called *Personal Representative* to the team.

When a communication session shall be established between two team members, the corresponding PRs compare the capabilities and preferences of each participant and try to find out which collaboration scenario is most appropriate. Furthermore, changes in the personal profile that are related to a special team may be synchronized with the information held by the team agent.

In order to access resources at remote places, the team agent sends simple agents (called *messengers*) to the remote places. These mobile agents may either just deliver a message (thus implementing a remote message routing) or execute a multi-step task remotely, for instance, to allocate rooms and ensure the availability of devices.

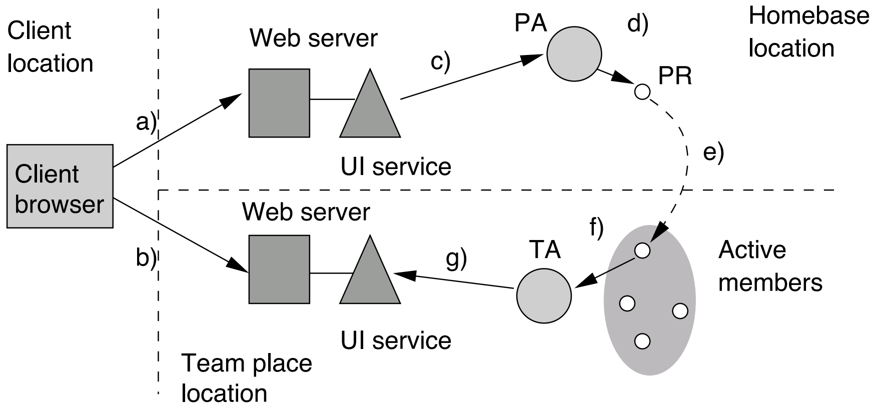


Fig. 2. User sending the PR into a team

Figure 2 illustrates what happens when the user enters a team context. At first the user's browser contacts the home web server – and also the team server. But the team server will not respond until the following action is complete. The UI service at the Homebase looks up the associated Personal Agent (PA), creates a message, and submits it. The PA instructs the Personal Representative (PR) to wander to the Team place and provides it with the user's current preferences and capabilities.

After the PR has arrived at the Team place (where PRs from other persons may be present), it informs the Team Agent which sends an update to the local UI service. Now the UI service enables the team web server to complete the request by sending a reply to the user's browser.

3.5 Communication and Collaboration Tools

UNITE specifically addresses the easy-to-use provision of communication channels (synchronous as well as asynchronous) between the people involved in the collaborative work processes, at any time and at any place. UNITE provides a unified communication and collaboration middleware, independent of specific protocols, networks, media types or languages so that the virtual office will be flexible and work over various disciplines, will use any available communication channel and enable media rich real-time interaction. Users may use different networks and protocols but still join in the same communicative session. Such hybrid communication services will support the ability to establish connections between end-points that use different communication protocols as well as blend different media streams.

Since collaborative processes often span multiple sites, the support of multi-party distributed communication is mandatory. By providing a unified way of communication for all types of real-time interaction, it is possible to develop applications for the virtual office with the ability to focus on the application domain rather than on the communication protocols and infrastructure. The UNITE supported user will be able to locate another UNITE-supported user and contact him without considering which method of communication to use.

The communication/collaboration middleware services is responsible for the following main tasks:

– Connect between one or more UNITE users
– Conference between several UNITE parties
– Establish communication with people that are not UNITE users

Communication and collaboration between team members relies on a communication and collaboration layer, which itself accesses several communication and collaboration servers and services and has the capability of integrating existing software packages. The software chosen for the platform is given below; but in general, the platform permits other packages to be used as well.

4 Implementation details

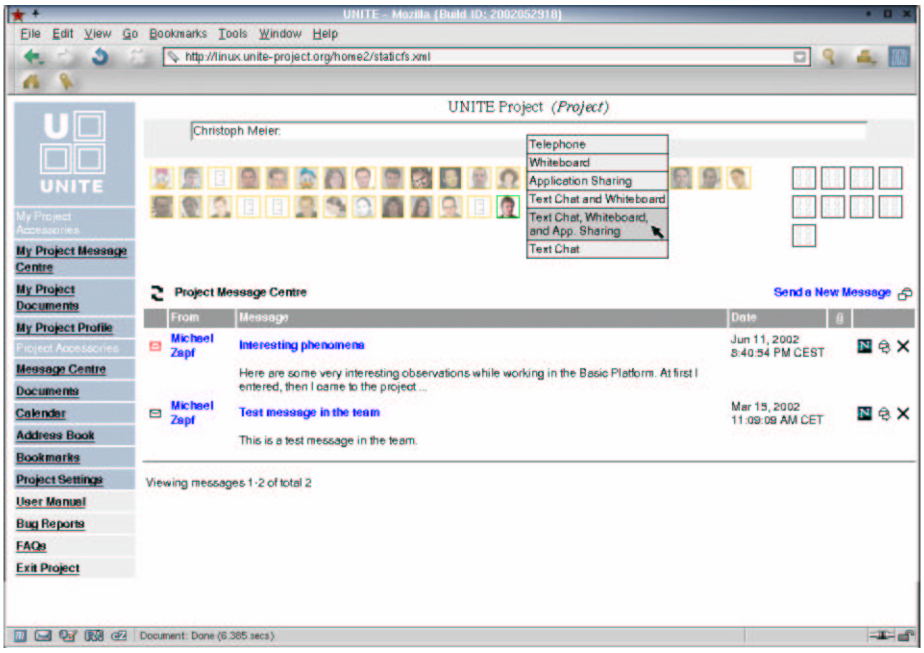
In this section, we give some further information about implementation issues. These refer to the implementation of the Basic Platform, the first prototype of the UNITE platform. The Enhanced Platform which has entered the implemention phase with contain these implementation issues as well.

4.1 User Interface

The UNITE user interfaces are fully based on Web technology. Although Web browsers as a UI platform have quite some limitations compared to a fully featured GUI platform, its great advantage is the support of ubiquitous access. The user interface consists of a navigation bar at the left side which allows access to the respective features, and a detail view in the rest of the screen. Figure 3 shows a sample screenshot of the current *UNITE Basic Platform*.

Target users of the UNITE platform are expected to be highly mobile, accessing their work-related resources from a variety of places. Web-based access to e-mail messages gained quite some popularity for just this reason. Each AMETAS place in the UNITE platform provides its own user interface if desired. Thus, from the user's perspective, UNITE appears as a network of web sites.

The homebase serves as the entry point; single sign-on is supported throughout the sites a user has access to. Although there is a strict separation of user interfaces belonging to different work contexts in a technical sense, integration is possible in the user's



Christoph Meier:

From	Message	Date
Michael Zapf	Interesting phenomena Here are some very interesting observations while working in the Basic Platform. At first I entered, then I came to the project ...	Jun 11, 2002 8:40:54 PM CEST
Michael Zapf	Test message in the team This is a test message in the team.	Mar 15, 2002 11:09:09 AM CET

Viewing messages 1-2 of total 2

Fig. 3. UNITE Project Portal

perception. In particular, elements of the homebases user interface, that is, personal tools and services, can be embedded in any team user interface, provided the Personal Representative carries URLs of them to the team place. In contrast, Room Places do not offer similar user interfaces; they are only accessible via the agents that migrate to them.

From the agent system perspective, the user interface is regarded as an external entity interacting with Place Users inside. This enables the UNITE platform to support other user interface technologies besides plain Web interfaces, e.g. Java applets continuously interacting with platform components or dedicated client software components.

The user interfaces are implemented in three layers. The gateway between the inner platform and the outer world is the UIService. It provides a call-level API to an AMETAS place of the UNITE platform. Its methods can be called remotely through an RMI-like scheme. Method calls from outside generally lead to one or more messages being sent to place users inside. Return values are retrieved from response messages sent back to the UIService.

Besides this translation between the communication paradigms of remote method invocation (outside) and asynchronous messaging (inside the platform), the UIService also performs the task of a firewall, and user session management. What messages can be sent into the platform is controlled there and message contents can be controlled as well. An authenticated session ID is required for all API operations except those supporting login. This session ID serves as a reference to the user ID.

On top of UIService's API there is the server-side presentation layer. It consists of servlets and *eXtensible Server Pages* (XSP) [4] which are accessed through HTTP and in turn access the platform API. XSP produce an abstract XML representation of the user interface, which is then translated into client-specific formats like HTML for standard Web browsers or WML for handheld WAP devices. Such client-side components form the third layer of the user interface, accessing the server side through HTTP. This protocol is also the preferred method of attaching client-side applets or applications to the platform since it is available virtually everywhere, even behind many firewalls.

4.2 Service Adapters

Since a key feature of the UNITE platform is integration of exisiting tools and services, agents have to interact with such external entities. This is achieved through service adapters. AMETAS offers the concept of services for interfacing with the outer world [5].

A service is a Place User that can, if possessing appropriate permissions, perform arbitrary actions outside AMETAS, e.g. open network connections or access files. It is therefore natural to represent external tools and services by AMETAS services inside a place. Besides mere representation, AMETAS services perform role-based access control for the service they represent, single sign-on, and abstraction.

Though there are several different implementations of each kind of tool, e.g. a synchronous collaboration service or a calendar tool, there is only one abstract AMETAS service interface for each type. A lower level of abstraction is buried inside the AMETAS services. There, similar services are encapsulated under unified interfaces at the Java class level as far as possible. The difference between these two levels of abstraction is that Place Users, and thus AMETAS services in particular, can be exchanged without any code modifications and even at runtime, while class-level abstraction is helpful mainly in the implementation phase.

4.3 Communication, Collaboration, and Messaging

The communication and collaboration features of the UNITE platform are accessed from other AMETAS place users via so called accessor services. These services, which are running on each team place, are implemented as AMETAS services and provide an API for accessing the underlying communication and collaboration middleware.

The communication and collaboration services use telephony (PSTN) and IP-based networks. The following types of interactions are currently supported on these lines (some also in combinations):

- PSTN and mobile phone connections (based on IBM DirectTalk using JTAPI).
- Voice over IP connections (based on Lotus Sametime using Sametime API).
- Chat connections (based on Lotus Sametime using Sametime API).
- Application sharing meetings (based on Lotus Sametime using Sametime API).
- White Board meetings (based on Lotus Sametime using Sametime API).
- Unified messaging of e-mail and voice-mail with access to e-mail and voice-mail (based on IBM DirectTalk using Java Mail API and JTAPI).

The communication and collaboration middleware is composed of several communication and collaboration servers and services. The API exposes the functionality without getting into the details of implementation, standards, protocols etc. The implementation itself is aiming to use as many as possible standards. In order to have more than one option for implementation we chose to use standards that are implemented under the communication/collaboration API:

- Java Telephony API (JTAPI) [2] for telephony and collaboration services. The Java Telephony Service Provider Interface (JTSPI) part of the JTAPI implementation is specific to Sametime. It interacts with the Sametime server in order to send messages to the Sametime client on the user's PC.
- JavaMail [3] for mail server. It provides the ability to view the current contents of a user's mailbox and do some primitive operations on this mailbox. The generic mail client (i.e. Netscape mail client) will run on the user's PC. There is no interaction between the Java Mail stack and the mail client. They both interact with the mail server independently. Currently, the protocols SMTP, POP3 and IMAP are supported.

The communication and collaboration layer is designed as a stateless mechanism that assumes that the agents that are managing the system save the state of entities in the system in a consistent way. The API is composed from a set of classes that are mostly Java Beans, which are not dependant on the calling layers or the implementation layers. Users can interact with each other or with a server (i.e. an e-mail server) that holds asynchronous messages received from others (or can send messages to others).

4.4 Security Issues

A variety of features and properties supports platform security. The very use of the AMETAS platform as the implementation basis provides means to ensure integrity of mobile code and communications security. The user interface gateway, the UIService, controls user access. It ensures only authenticated users can access platform functionality, and the user can be identified whenever accessing the platform.

An inherent security feature is the inability to login without the personal agent or representative being present. In order to log in to a homebase under some user ID, a Personal Agent for that user ID has to be present there. Otherwise there is no way to get an authenticated session. In order to enter a team place, a personal representative of the user has to be present there, and the team agent has to acknowledge this particular instance of a personal representative.

Access control is performed as a local function of each place. This is an implication of the distributed nature of the platform, and of distributed responsibility and ownership. On the place, access control is performed by each place user individually.

The UNITE platform does not address the danger of hostile hosts and hostile agents. One one hand, the AMETAS infrastructure ensures that only agents which were signed by a trustful signer in the agent system may enter any place in the system. On the other hand, we believe that there is some degree of responsibility of the maintainers of such an agent environment to check whether the participants are trustful or not.

5 Conclusions and Future Work

Co-operative Workplaces are environments that facilitate and support collaborative work. They provide for easy communication between co-workers and for easy co-ordination (for example by way of supporting peripheral awareness) of their individual activities. In addition, they provide for collaborative work among co-workers. The ultimate goal of UNITE is to strive for a comprehensive Co-operation Platform, which is based on basic building blocks and services provided by heterogeneous tools. Such a co-operative workplace develops and dissolves with the progress of a project, is ubiquitous, i.e. accessible from virtually anywhere, and is integrated with respect to different contexts of work, virtual and physical workspaces, and established and emerging tools. Furthermore, it allows for awareness with respect to the progress of project related work, the current activities of other project members, and the availability of other project members and allows for adaptation and personalization not only through a personalized portal but also through the arrangement of tools and resources, and opportunities to adapt the look and feel of the workspace.

The current state of the project is that the *Basic Platform* (the user interface of which has been shown in this article) is completed and that it passed the user trials. We are currently implementing the *Enhanced Platform* which realizes nearly all architectural issues introduced in this article. However, as stated, some features will probably stay for future extension, like a full-featured interaction with rooms and facilities management.

6 Acknowledgement

This project is funded by the EU under the Information Society Technologies (IST) Programme (Project No. IST-2000-25436). The UNITE consortium is led by IBM France (F), for project management, and the Fraunhofer Institute SIT (D), for technical direction. The Fraunhofer IAO Institute (D) and Coventry University (UK) are co-operating on user requirements elicitation and user evaluation, with the primary user input throughout coming from consultancy company Pentascope (NL). Together, Fraunhofer-SIT, IBM Haifa Research Laboratory (IL), ADETTI (P), and Steria (F) are developing the platform architecture and prototype. For further details on UNITE see also: www.unite-project.org This paper was supported by the IST UNITE project; however, it represents the view of the authors only.

References

1. Sven Türpe, Rolf Reinema (eds.): *UNITE Platform Architecture – Basic Platform*. UNITE consortium, 2001. http://www.unite-project.org/public/results/UNITE_Architecture_Basic.pdf
2. *Java Telephony API*. http://java.sun.com/products/jtapi/
3. *Java Mail API*. http://java.sun.com/products/javamail/
4. The Apache Software Foundation: *Cocoon 2 User Documentation – XSP*. http://xml.apache.org/cocoon/userdocs/xsp/index.html
5. M. Zapf, K. Herrmann, K. Geihs: *Decentralized SNMP Management with Mobile Agents*. In: Sloman, Mazumdar, Lupu: Integrated Network Management VI (IM'99), IEEE/IFIP
6. M. Zapf, K. Herrmann: *AMETAS web site*. http://www.ametas.de/