A Ubiquitous Service Environment with Active Documents for Teamwork Support

Patrik Werle, Fredrik Kilander,
Martin Jonsson, Peter Lönnqvist and Carl Gustaf Jansson

The FUSE Research Group, KTH Center for Wireless Systems, KTH
DSV, Electrum 230, SE-164 40 Kista, Sweden
{werle,fk,martinj,peterl,calle}@dsv.su.se

Abstract. We present a ubiquitous service environment for teamwork, supported by Active Documents. The environment consists of a physically dressed conference room and a software architecture based on Java and Jini. At the application level, mobile agent technology provide Active Documents that utilize context information and distributed resources to support users. We also present a prototype and preliminary results from observations in a meeting scenario.

1 Introduction

This paper consists of three major parts. Each relates in its particular way to the issue of ubiquitous computing and an environment created for that purpose. The first part consists of a brief background followed by a conceptual description of the fuseONE environment. The second part details a prototype implementation, while the third part of the paper describes a pilot case study carried out in the realized prototype environment.

2 Background

A ubiquitous service environment, or USE, is a physical and logical space where users can collaborate supported by multiple services providing adequate computing and communication facilities. The physical space is dominated by the material manifestation of computing resources, such as computers, screens, keyboards, pointing devices, PDAs, sensors and so on.

The logical space is to a large extent shaped by the amount of communication available between the physical devices. None or very little communication tends to partition the logical space into cells which more or less equate the confines of a particular device. In this situation, each attempt to transfer data of effect an operation becomes hampered by the technicalities involved.

With extensive communication, the logical space is widened and unified, and works with the users to better reflect that what is physically close is also conceptually close. At the beautiful end of this dimension, the logical space extends beyond the physical confines of the real world and allows for communication and co-operation without drawing attention to the mechanics of each actual interchange.

However, the mere possibility to employ a dazzling range of far-reaching operations contains an inherent difficulty which grows with the number of added alternatives. Any broad expanse, be it the Gobi desert or a logical service space, needs to be navigated with knowledge and deliberation. A confusion created by too many options is just as hampering as too few.

The inescapable conclusion with respect to USEs is that a good USE should present its human participants with a well-prepared set of tools. The software in the USE should, as far as is practical, unobtrusively anticipate the needs and actions of its users and make the necessary facilities available. There are caveats lurking in this ambition, as many users exposed to animated paper-clips probably are ready to testify, but the pitfalls should in no way be regarded as a detriment, but as a challenge.

While the issue of communication and interoperability to some degree is negotiated with existing and new technologies, it leaves us at the level where we can establish many client-server interactions, but have a difficult time finding out which ones will be most beneficial in a given moment. As humans we want software and services to guide or choice, but how to guide the software? There is the problem of modalities, in which familiar content has to be presented on new devices with unexpected capabilities. There is the problem of choice and selection, in which there are several possible actions to take and each cannot be unambiguously evaluated. There is the problem of routing, addressing and access, in which the physical location of the intended recipient has to be ascertained. Software services have to deal with this distributed and heterogeneous hardware environment as well as with new ways of interaction. The challenge we have to tackle is how to design services that support users, and itself, in these kinds of hard- and software environments.

One important property of a USE is that it should not be limited to single users but work as a support for several people working together, thus making it an aid for collaborative work. Working together in modern organizations most often includes producing, consuming and modifying different kinds documents. Since most of the work processes are based on, or driven by, document flows it is critical for an organization that users and applications have access to documents [1] [2]. But the way documents are managed today seems to be a remnant of the times when users had continuous access to the organization's environment through established and working channels with only a few options to reach the systems. Once in this environment, they were connected. In a USE, where we are surrounded with a plenitude of communication tools and the possibilities to contact the organization’s computer environment by using a collection of heterogeneous hard- and software which could be unpredictably disconnected, the instant access and instant feedback on our actions have been lost. It is all too easy to be in the wrong place with the wrong people at the wrong time carrying the wrong documents.

We have to handle issues related to the large volume of documents originating from different sources, such as email, faxes, groupware and different desktop applications and the problem that documents may exist in several places: as a word

processor file on a personal computer, as an attachment or text-only version in an email, as a rendered copy in a fax etc. In addition, several instances of each format may be multiplied across the recipients' storage devices, in some cases truncated or converted to suit local software or user preferences. To further complicate the issue, presentation facilities in some devices may be inadequate for some properties of the document or the size of it is simply too expensive for a low-speed connection.

Another problem is that information, which other users would benefit from having access to, is either physically mobile in a mobile device or unavailable in a shutdown personal computer. Access to and use of documents are generally neither controlled nor automated. Just keeping track of what documents exist and to retrieve the latest version can be an overwhelming task, especially when considering mobile and remote users in a USE. The challenge is to make the right information available to the right people, at the right time, and at the right place.

3 fuseONE

At the department of Computer and Systems Sciences, KTH, we have dressed a small conference room with a few selected appointments, nowadays more often than not found in similar locations. The room, the equipment and the software used therein have been nicknamed ‘fuseONE’.

The physical space of fuseONE contains a full bandwidth videoconference system, a SmartBoard [3] combined projection screen and pointing device, a cordless keyboard and mouse, a set of iButton identification devices [4], a VGA projector and a server computer. When there is a meeting in fuseONE, most participants bring along further equipment in the form of laptop computers with 11 Mbit IEEE 802.11 compliant wireless LAN transceivers.

The logical space of fuseONE consists of the department systems on the local area network, the Internet, standard application software and our own applications and middleware. Our ambition with fuseONE has not been to replace standard software like web browsers and word processors, but to ease their deployment for the participating users. As it turns out, many practical issues revolve around communication and how to navigate focus of attention.

A gathering in fuseONE usually follows a similar pattern: the participants gather with their laptops around the conference table and insert their iButtons in the receptors scattered on the tabletop. The laptops display representations of the other participant’s, the fuseONE server computer and other software services. The projector and SmartBoard are activated and displays the desktop of the server computer with more of the fuseONE software.

As someone wants to display a document on the SmartBoard, a local drag and drop operation on the personal laptop transfers a copy of the document to the server computer where it is automatically launched. Interpersonal document exchange can be similarly arranged by dropping the document on the icon representing the receiver.

Depending on who is attending the meeting (as indicated by the iButtons present), an ‘Active Document’ may sense that a meeting is in progress and announce itself on one of the available displays. Files relevant to the group can be retrieved and saved by

interacting with the Active Document (assuming one is present for the project or task). The Active Document serves as a roaming repository for the group's efforts and is constantly watchful for changes to its payload or congregations in which it should participate.

One of the important properties of fuseONE is that the services available in it, such as Active Documents, should respond and react to dynamic, ad-hoc behavior from its human users. For instance, there is no need specify anywhere that a meeting of a certain group is under way. This event is instead inferred by components in the environment from the people present and their activities.

Our system design embraces the idea of graceful degradation when faced with a point failure. Modular components, general interfaces and redundancy are long-proven means to that end. For example, if the context system is not responding, there are other ways to find a user's document launcher. We want to build a USE that is not inherently dependent on specific software (such as in [5]) or specific hardware (such as in [6] [7]). Compared to [8] our goal is to, as mentioned above, design a more decentralized system capable of providing ad-hoc services in a more dynamic, distributed and heterogeneous environment.

4 fuseONE Services

We strive for providing fuseONE with a lot of different kinds of services, from simple reactive services and user-interface services to proactive more complex services. We also embrace the use of standard application programs like the whole host of software from Microsoft and other vendors. It would be impossible otherwise, since we believe that users would not accept their absence. Therefore, we want to operate on the unit of standard files and applications, but let our services and software provide additional actions which enhance the use and availability of the standard tools. There are issues raised by this, as how to smoothly involve tasks such as concurrent editing, application sharing and so on, without being tied down into a specific vendor or operating system.

Below are some services we have developed described, some of them simple which main purpose is to be exploited by other services and other that have a more complex and advanced behavior (i.e. Active Documents).

4.1 USE Context-sensitive Desktop

In the fuseONE environment, it is vital to have a visual handle on both human and machine participants as they appear in the logical space. This handle and the grips on it are provided by an application on the user's computer (commonly a laptop), intended to integrate with the native user interfaces.

The USE desktop presents a representation of the logical space and its participants, and familiar means of operation (click, drag and drop). To give someone a file from the local file system, simply drop the file icon on the receiver's icon. To wake up an Active Document, click on it. It should be emphasized that the desktop metaphor is

convenient because it is familiar and works well when interacting with the native system, but it is by no means uncontested as the best one.

As the number of visible service grow, the desktop view soon becomes cluttered and additional measures must be taken to prevent confusion and aid navigation among the icons. One of these measures is to make the USE desktop sensitive to the current context and thereby adjust the view that meets the user. In fuseONE, this is accomplished by hiding those desktop items which are not relevant to the current context. The task of estimating relevance is performed by a context inference system.

4.2 Context Inference

The availability of services in fuseONE can be expressed as the sum of its parts, but that quickly becomes an unwholesome endeavor as the number of components grows. To navigate the logical space in terms of services available to the users and certain applications, a special subsystem called 'Context Shadow' [9] has been introduced. Context Shadow makes it possible for humans and software services to ask questions about a persons current context, and specifically what services that are relevant to that context. In the system, services and sensors are organized in meaningful collections, creating a searchable topology of context information.

In fuseONE, the context inference system silently monitors users and their activities and maps the services available to each user. Properties of services and users, such as their location, project membership etc. can be used to e.g. find services near the user or near other users within a shared project. For example, a document application launcher may be running as registered to a particular user, in which case it makes sense to present a document addressed to that user on that particular service. The Context Shadow system is able to provide answers to questions such as: *"Is there a document application launcher available in the same room as Martin?"*

Location information is provided by user actions and the attributes and presence of certain software services. User actions include the manipulation of iButtons which explicitly announces their location and the detection of screen savers on stationary computers.

Unlike e.g. [10], the aim with Context Shadow is mainly to offer context information for other services, and not directly to users. Context Shadow is also application independent, which is not the case with systems such as [11].

The USE desktop application subscribes to information from the context inference system to filter out from its display those resources which are not pertinent to the immediate situation. The Active Documents also exploit the context inference system by monitoring the contents of a location, like the fuseONE conference room, rather than collecting information from separate iButton terminals.

4.3 Document Application Launcher

In order to receive a document most computers participating in fuseONE run a small service which accepts a file and then presents it, using the application associated with the document type. Both users and other services like Active Documents could use

the launcher. When it is operating, the user's personal environment is open to sudden disturbances, such as an application suddenly starting. We have found indications that ordinary social protocol cushions this when the parties are within talking range, as in the conference room setting. At larger distances, however, other modes of transfer, like email, appear to be preferred. A polite negotiation mechanism, like that of ICQ's document transfer [12] could easily be implemented but at the moment we have not done that.

4.4 Public Address Notifications

At the time of this writing, recent additions to fuseONE include a notification service in the form of a scrolling banner. A banner is displayed on the SmartBoard and user and other services can direct messages at it. The idea is that general announcements or a stream of updated background information which do not require immediate and disturbing notifications can be presented there. A small Active Document could for example use it to display its content.

In the spirit of the document application launcher, individual users are free to run their own notification services if they so desire.

4.5 Mobile Messenger Agent

The mobile messenger agent application is a small application which basically tries to hunt down the recipient and execute as close to it as possible. Originally envisioned as a kind of self-delivering information parcel, it was quickly realized that if wrapped correctly, the content of the messenger agent could be almost any program. The messenger agent thus has the ability to deliver secure content, extensive graphics or multimedia or any other form of messaging which requires local processing power or user interaction.

Our goal is also to make the Active Documents truly mobile and thereby provide them with the capability to execute near the users, e.g. for supporting disconnected operation.

In order to allow mobile code to move around a code execution server is needed. In fuseONE there is such a service which receives and runs objects. Given the security implications of such a service, it has not seen any general distribution so far, but it is obviously needed as part of the environment.

4.6 Active Documents

The notion of active documents has been used in several works. In [14] 'active document' refers to documents that perform activities as a result of users' manipulations on the documents, such as opening and scrolling. In [15] active documents react to changes in context information and accordingly adapt the information displayed. In [16] 'active document' denotes a new technique for modeling documents that includes physical, semantic, and functional properties. In the work presented in [17] 'active documents' is a document that has been enhanced

with the ability to autonomously create ‘adlets’ that contain metadata and that will advertise the document to other documents.

The work presented in [18] is, like the one presented in this paper, based on the idea that the documents themselves are active. The documents support coordination functionality by using ‘active properties’; they should be “situationally aware”. The activity is distributed to the documents, but the approach is based on a shared infrastructure where documents are structured according to their properties.

Our idea of Active Documents takes off from the agent-programming paradigm [13], and turn documents into autonomous mobile agents and by that give them some useful qualities. A document should, for example, be *aware of its content* and the intention with it, and be *aware of the context it is operating in,* e.g. its receivers (who, why, preferences about formats, physical surroundings, etc.). The documents are active in the sense that they are autonomous (act independently), reactive (react on changes in the environment), and proactive (have their own goals and plans).

The goal is to make a document capable of making decisions on its own and capable of performing different actions through its lifecycle, thereby supporting users. One example of this is that a document could decide, depending on how important its role is, if it should summarize itself and move to the user’s mobile phone, or if it should move to the user’s desktop computer and there wait for the user. Agent technology has also other advantages for programming services that should operate in a USE [13].

5 Prototype

We have implemented a prototype that illustrates some of our ideas. The prototype's main components are: Tipple – the USE desktop, FileStarter – the document application launcher and Ad – the Active Document.

The need to cover different platforms and operating systems, as well as mobility, has guided us towards the use of Java. The Jini extension to Java, in combination with remote method invocation and remote launch, has given us the ability to deploy and update a developing body of distributed software without constant reinstallation on each participating computer. One particular rewarding feature of Jini is the discovery mechanism whereby clients can locate and employ services without prior knowledge of host names and port numbers.

5.1 Distributed Software Deployment

The very nature of having multiple instances of client and server processes running simultaneously and in concert on eight different hosts is no easy feat. One of the most irritating problems is that of distributing updated versions of the client software. Users may not feel confident enough to do the upgrade themselves or they may not be able to do it in time, with the risk of unnecessary system brittleness introduced by conflicting versions of software.

FuseONE software uses a main repository of deployed software, located on an http server. All that clients need to install are a small bootstrap loader (written in Java)

which fetches the corresponding JAR file from the http server and executes it. In practice this process is condensed into small command scripts or desktop shortcuts.

The load taken off software maintenance with this scheme is very valuable, in that a new version of a program is installed in a single location and immediately available. There is a strong analogy to network file systems, except that the client computers do not have to mount any remote device, only know the URL to the http server, and the http server can (at least in theory) be on the other side of the planet.

Even so, the requirement to distribute the correct URL has been an obstacle at times. For example, our group successfully transported the fuseONE environment into another location some 700 kilometers away from our laboratory, using only laptops and a portable projector. This however, required that all participants mended their shortcuts and scripts to point to another http server, a procedure which took some time and effort.

Since one of the greatest benefits of Jini and Java is location independence (within the limits set up by multicast propagation outside the local area network), it appears natural to modify the bootstrap loader so that it can request the necessary and current invocation details from a Jini service in the network. With that achieved, the details of the distributed system installation ought to be completely transparent to the user.

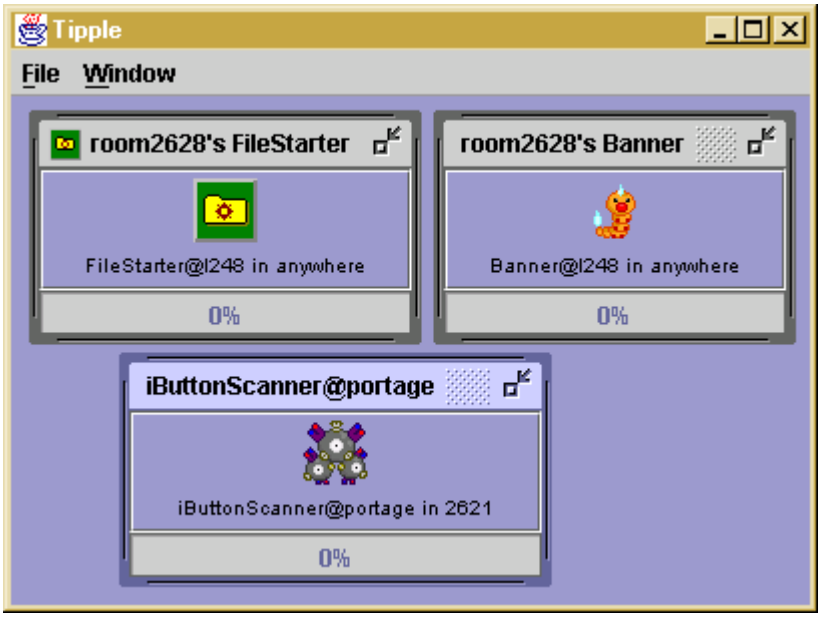

Fig. 1. The USE desktop prototype Tipple

5.2 Tipple – the USE Desktop

The prototype implementation of the USE desktop is called Tipple. Tipple is a graphical user interface that presents services for the user. A sample view is shown in

figure 1. Which services that are appropriate for the current context will be received from Context Shadow [9], the context information infrastructure. The idea is that local resources, both stationary and resources on nearby computers should be presented.

Tipple makes it possible for the user to click on and drop different items on the icons that represent the services. What will happen depends on the actual service's implemented behavior. The range of services depends on the current context.

The Tipple is basically a Jini service browser. It monitors the local offerings of Jini services and attempts to display any service which has registered under the Jini attributes Name and Location. If the service also implements the attribute ServiceType, an icon is retrieved and placed together with the text on the desktop.

Icons for services which implement our *putfile* interface allow drag and drop of files from the surrounding system desktop. These are also expected to respond to a mouse-click on their icon.

The Tipple retrieves the name of the current user from the command-line or from the system. Using that name as a key, it attempts to contact the Context Shadow system. If successful, the user is given the option to enable a filtered view. When the filtered view is enabled, the Tipple compares the Jini service IDs it has discovered with a list of service IDs obtained from the Context Shadow system. The intersection of the two sets is kept displayed while the remainder is hidden. The idea is that the list from the Context Shadow system contains only those services which are relevant to the user in the current location.

5.3 FileStarter – the Document Application Launcher

The Filestarter is a Jini service that accepts an arbitrary remote file and launches it. It does this by storing the file in a temporary directory and then applying the native operating system's start command. On Microsoft Windows this usually works quite well. On other platforms additional programming is needed to select the most appropriate application.

When the document is finished, the temporary file is deleted. However, if a permanent copy is needed there is often ample opportunity of making one while the application is waiting for user interaction.

As the FileStarter came into use, we discovered that we sometimes wanted to send a web page, i.e. not the HTML code itself, but actually a URL. We found two different ways to accomplish this. The first involved the transfer of a small HTML file which simply used the META element to request a reload of the actual target URL. The second strategy (between Ms Windows systems) was to construct a local shortcut to the target URL and then simply drag and drop the shortcut to the recipient.

The FileStarter is being run both by fuseONE conference room server computer and by individual users. Since the FileStarter is displayed by the Tipple service browser, the FileStarter has become the de facto indicator of another user. Without any explicit intent in that direction, it has come to serve as a rudimentary user avatar in fuseONE's logical space.

5.4 Ad – the Active Document

Ad is a service that, for example, stores and presents content files. The files could be of any type (e.g. Word files) and it can use other services to present its content. Ad uses the context information provided by the Context Shadow system, for instance who the project members are and their location. When Ad recognizes that two persons, members of the same project as the document itself, are in the same meeting room, the document assumes that there is a meeting going on. It then enters the room (for the moment only virtually) and tries to find a public resource available that it can present itself on, for instance a projector. Information about appropriate resources is received from Context Shadow. When the document enters the room, it also becomes part of the service collection that Context Shadow informs Tipple to present.

When a user 'clicks' on Ad's icon in Tipple, the document gets information about who the user is and tries to find a FileStarter, suitable for that user, to present itself on. The document then dynamically creates a list of its content, i.e. the files it has stored (Word documents, URLs, memos etc.). The document sends this list to the FileStarter with a reference to the computer where the document currently is executing. The document then starts an http server that gives access to the files.

When a user drops a file on Ad's icon, the file will be added (or updated) to the project's information structure, held by the document, and thereby the file will be available to all other project members.

One of the main ideas is that a document by *actively participating* at the meetings should be able to support the other (human) participants. From the system's point of view, a document is actually modeled (as far as possible) as a human meeting participant.

6 Case Study

A pilot study was performed in order to see if the environment built by the research group would be empowering for the users and to determine whether the fuseONE environment performed well in a user situation, to give input on what should be further developed in the environment. The goal of the study was also to be a test-bed for further studies, giving input on what methodology to use for studying environments of this kind and how to design usability studies in wireless, ubiquitous and distributed work environments. We summarize this as two major goals:

1. To examine if our efforts in developing services for such environments are somewhat on the right track.
2. To unconditionally examine the effect of wireless technology on local collaboration. What are the benefits and what new problems arise?

To achieve a real need for teamwork we chose to use the planning of a course at Department of Computer and Systems Sciences (DSV) as the setting for the study. A number of meetings (5) were scheduled and the participants chosen were those who lecture at this course or in any other way are responsible for parts of it. The goal of the meetings was to plan the coming course together. This is the usual way courses

are planned at this workplace and thus a natural situation to try to empower with new technologies.

We tried to evaluate how and if users are helped by the new technology in the fuseONE environment. We also wanted to look at the problems that might occur when using new artifacts and services in an environment like fuseONE. In our approach this emphasizes the need for an anthropological approach to the studying of the users in a natural setting.

Fig. 2. Study setting

6.1 Study Setting

Participating in this initial pilot study was the developer group themselves, with a few additions. Due to causes such as illness and other work the participatory frequency varied somewhat. The fewest number of participants was three and the highest seven. All participants were staff at DSV. Two were female and five male. All participants were to a very high degree familiar with computers although three were new to using laptop computers equipped with wireless local area network access in their daily work.

Each meeting was recorded to video and as some of the meeting participants were those performing the study, they also took notes. As this was a pilot study with the

goal to give input on how to design further studies and enhance the system the results should be looked upon just as observations.

All participants in the study were equipped with a laptop computer with wireless access to the network. Most of the participants had already access to that equipment and had been using it regularly in their daily work for a longer period. Only two of the participants were new to the technology. Each participant was also equipped with a personal iButton, a small memory device associated with the user. When entering the room the user pressed the iButton into a reader, thus identifying him/herself to the room. This identification, together with each user's personal software, enabled the room to distribute its services to each user.

On each users laptop computer were installed a standard package of commercial software for text editing, Internet browsing, e-mail etc. Each computer was running the fuseONE software.

For the users to be able to collaborate locally and distributed the room was set up with the following software: The SmartBoard software and the same commercial software as on the users laptop computers. To enable collaboration the fuseONE system was used. At one meeting one of the participants worked remotely using Microsoft NetMeeting to be able to see and talk with the participants in the room. He also used the fuseONE environment software remotely (Fig. 2).

6.2 Breakdowns

There were some breakdown occasions observed:

- Specific information is not directly accessible from a person's laptop, e.g.:
  "Oh that document resides on my other computer, or maybe I have it on paper somewhere"
  "My calendar is not accessible through this computer"
- Searching for information. On several occasions the meetings halted while someone was trying to find a document on a hard drive, or trying to remember the address to a certain web page.
- The different computer screens in the room (personal and public) displays different content (or different versions of the same content):
  CJ looks at the shared screen, which shows an old copy of a schedule, PW looks at a newer version on his own laptop.
  CJ: "…But this lecture about social navigation…"
  PW: "…There are two such lectures"
  CJ: "Hmm? …Two?"

There were also some breakdowns related to the fuseONE software identified originating from unpredicted system behavior. The fuseONE tools failed to send information to another computer. This was a fairly common problem that clearly disturbed the work process. Often several minutes were spent on finding other ways to send the information.

6.3 Results

Useful support
The ability to easily display personal information on a public display was commonly used, liked by the participants and seemed very useful. At some occasions a person would write something on his/her computer, not focusing on the meeting, then actively interrupting the meeting to use fuseONE software to display the document on the big screen. Another way this functionality was used was to provide information related to what someone in the meeting is currently talking about, e.g. a web page. This was often done without interrupting the person speaking.

Another interesting observation was that the observed meetings often continued from the point where the previous meeting ended. Support for this behavior was given by the ability to easily store and retrieve documents in preparation by using the Active Document software. Thus also more work was efficiently performed during the meetings, rather than the more common way of just using the meeting to set up goals and waypoints for the participants to be fulfilled until the next meeting. With the help from the new environment one participant could (and would) act as a note-taker or secretary, using the shared wireless keyboard to type on the SmartBoard screen. In this fashion all participants could cooperate on what should be appended to the document on the SmartBoard and use this as a shared focus of attention and at the same time use their own laptop computers to retrieve information relevant for the work in the room and add it to the shared focus (or each other) if needed. Participants could also, without losing focus on the work-process take care of personal work such as answering e-mail, take personal notes or other activities. The meeting room became an actual workspace while the participants still were able to stick to the meeting agenda.

Observations from a distributed scenario
At one of the meeting occasions we tried to extend the meeting to also include a person that was not present in the room. The person was present through the Microsoft NetMeeting videoconference application. The distributed participant was shown to the participants in fuseONE room on the same SmartBoard as mentioned above. The person not in the room could watch the group in the room through a small stationary camera turned to where the participants sat in the room. He could not follow if any of the participants went up to the SmartBoard to physically show anything on it or any other activities taking place outside the camera focus. The group used application-sharing software implemented in NetMeeting so that the remote person could see and manipulate the documents that were run on the shared screen in the meeting room. The person also had access to fuseONE software. The audio quality in this meeting was very poor. Since the remote person's picture sometimes was hidden behind different screens on the SmartBoard the issue of sound became even more important. Noted was also that the fact that the remote person could see the group as a whole was satisfying, although movement by participants in the room could not be followed. The known problem of gaze-direction by the remote person

was observed (the remote person looked at his screen and not into the camera) but not much discussed since, as mentioned above, sound became a more interesting issue.

One important observation was the need for the distributed person to have access to the information being displayed on the big screen in the meeting room. It was also preferable to share the same view of a document over NetMeeting, than to have to equal copies of a document. This way you could point to certain parts of the document while talking about it, without confusing the remote part. It also gives opportunities to co-write a document.

General observations

The support for sending information between each other during the meetings was not used to the extent we had expected. One factor that might have affected this behavior is that the service did not work reliably during the study, due to network problems.

A common situation was that the meeting chairperson has control over the big screen, i.e. that person used the keyboard and the mouse connected to the shared screen. The other participants were mainly giving oral suggestions of what should be written or in other ways manipulated on the shared screen. The other participants used fuseONE software to 'throw' documents to the screen, but in most of the observed occasions the meeting chairperson, acting as a moderator, took control of the document after it had arrived to the shared screen. In this setting the other participants have to ask for the mouse and keyboard to be able to manipulate the information on the shared screen. If it should be possible to use the laptops to interact with documents on the big screen this might change.

Over this series of meetings, most of the documents were created during the meetings, and not so much between meetings. Therefore the process of compiling information that was brought into the meetings was not examined in depth.

7 Discussion

We have investigated a ubiquitous service environment that integrates services and gives them and its human users the possibility to navigate and exploit resources by means of direct manipulation and by using context information. We have implemented a prototype with some services for a meeting scenario and a first case study has been performed. The fuseONE USE has, according to our early results, turned out to be useful.

The fuseONE environment in its current implementation seems to empower its users in several ways. The users are able to quickly and easily share information between each other and to provide a shared view of personal information. Also, more efficient and satisfying work can be done during a meeting situation. Participants can be remotely present and make full use of the fuseONE environment but there is a flagrant need for better tools for audio and video communication over computer networks for this to be really feasible and efficient. Breakdowns in the work-process occur, often due to that the environment does not support enough heterogeneous technological artifacts.

The fuseONE environment is in a constant process of development and already some of the technological causes of breakdowns have been taken care of. It has also been very rewarding to observe the emergence of unplanned utilitarian behavior among both users and software. For example, the FileStarter application in combination with the Tipple were invaluable during a workshop as they removed much of the cable switching which otherwise occurs between talks. On the same occasion, it was discovered that an Active Document could serve as a repository for the presentation files, giving some users the option to view the slides on their laptop computers.

It is important to take note of the fact that with a few steps towards a local collaborative computer environment we can see tendencies towards rather big changes in the work process of the users. The fact that the work *in* the room gets a higher weight and also is experienced as more fun when participants are technologically empowered through the fuseONE environment gives us the notion that we are on the right track in developing wireless, distributed and local tools for collaboration.

It is also notable that the fuseONE system has been, and continuously is, in almost daily use since the end of the abovementioned pilot study. The fuse group always uses it in their group work and meetings. The group members like the system and rely on it. It is most noticed when it due to some or other technicality is not working, a breakdown observation pointing at the fact that the systems level of transparency and "ubiquitousness" is high. It should thus be taken into account that the system functionality and HCI-aspects has not only been investigated during a specific study, but is continuously evaluated.

7.1 Future Work

Much work remains to be done. The fuseONE prototype, for instance will be dismounted and rebuilt in another location as fuseTWO. It stands to reason that the software will face a similar revision and extension. In particular we would like to incorporate support for security, add physical artifacts like PDAs and tangible interface prototypes, and extend the sensory apparatus by which the agents and inference mechanisms perceive the physical space of the service environment.

Security

There are of course a number of security issues involved when so many computer systems and programs are interacting as they do in fuseONE. For example, the ability to launch a program on another user's computer without prior warning or non-repudiation is not a comfortable concept. The traditional question of identification and authorization apply with full force and it is quite obvious that deployment on a larger scale requires a security layer to manage them.

A simple security scheme can be built around opaque tokens of certification. When a client is started it obtains such a token, using information required from a trusted facility. When the client makes demands on a remote service, the token is passed along with the request. The service inspects the token using the same or another trusted facility and is then able to reject or accept the client, as well as logging the

client's identity. The server may of course also be required to surrender a token to the client, to prevent server spoofing. The exchange tokens themselves must be one-time tokens, to prevent playback attacks. They should of course also be tamper-proof.

The trusted facility is most likely a piece of library code (or classes in Java) which performs the necessary user authentication and compiles the opaque tokens handed between clients and servers.

Basic authentication is the foundation for a security system. With the identity of another party ascertained, it becomes necessary to establish what that other party is allowed to do. A computer system, such as for example a Windows NT system, enforces a set of privileges and user rights among the resources it controls. Sometimes these rights are extended to other systems in the vicinity with the use of authentication servers. There is yet no complete intuition of how to model the security of a distributed system with many artifacts, like the fuseONE environment. For example, how public should a public display service be? Should presentation on it be available to all users or only the group of users who currently benefit from it?

Beyond laptops

The commercial range of small computing devices appears to expand each day. The promise of Bluetooth [19] has by now left most workers a trifle jaded of the hype although actual products recently have materialized. For many of the small and personal devices, like cell-phones and PDAs, there is no easy way to integrate them into the USE without special measures. Since many of them are not yet powerful enough to host a full Java virtual engine, one cannot continue to exploit Java's portability in that arena. Instead, native code must be produced, something which often takes too much time and requires both devotion and skill. Power requirements, communication speed and expensive access to the appropriate APIs also constitute limiting factors.

Still, what we would like to investigate further, is a set of lightweight devices which both physically and logically are available as public resources, but very quickly can be turned into personal interaction platforms. One practical example would be so called 'webpads', flat panel computers that lie scattered around the conference room, waiting to be picked up and used by anyone. In such a scenario, user identification must be smooth and transparent using electronic tags, fingerprint readers or some other convenient technology.

8 References

1. Sprague, R. H. Jr.: Electronic Document Management: Challenges and Opportunities for Information System Managers. MIS Quarterly (March, 1995)
2. Ranadivé, V.: Power of Now – How Winning Companies Sense and Respond to Change in Real Time. McGraw-Hill New York (1999)
3. SmartBoard is a product of SMART Technologies Inc. <http://www.smarttech.com>
4. iButton is a product of Dallas Semiconductor Corp. <http://www.ibutton.com>
5. Roseman, M., Greenberg, S.: TeamRooms: network places for collaboration. Proceedings of the ACM 1996 conference on Computer supported cooperative work. (1996) 325-333

6. Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D., George, J. F.: Electronic Meeting Systems to Support Group Work. Common. ACM 34, 7 (Jul. 1991) 40-61
7. Stefik, M., Foster, G., Bobrow, D. G., Kahn, K., Lanning, S., Suchman, L.: Beyond the chalkboard: computer support for collaboration and problem solving in meetings. Commun. ACM 30, 1 (Jan. 1987) 32-47
8. Fox, A., Johanson, B., Hanrahan, P.; Winograd, T.: Integrating information appliances into an interactive workspace. IEEE Computer Graphics and Applications, Vol. 20 Issue 3 (May-June 2000) 54 -65
9. Gustafsson, H., Jonsson, M.; Collaborative Services Using Local and Personal Facts. Proceedings of the PCC Workshop, Lund, (November 1999).
10. Caswell, D., Debaty, P.: Creating Web Representations for Places. Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC2K) Bristol, UK, (September 25-27, 2000) 114-126.
11. Dey, A.K., Understanding and Using Context, Personal and Ubiquitous Computing, Special issue on Situated Interaction and Ubiquitous Computing, 5(1), (2001)
12. ICQ is a product of ICQ Inc. <http://web.icq.com>
13. Jennings, N. R., Wooldridge, M.: Applications of Intelligent Agents. In Jennings, N. R., Wooldridge, M. (editors): Agent Technology: Foundations, Applications, and Markets. Springer-Verlag (1998)
14. Ahonen, H. et al: Intelligent Assembly of Structured Documents. Technical Report C-1996 40, University of Helsinki, Department of Computer Science (June 1996)
15. Voelker, G. M., Bershad, B. N.: Mobisaic: An Information System for a Mobile and Wireless Computing Environment. Workshop on Mobile Computing Systems and Applications (November 1994).
16. Sauvola, J., Kauniskangas, H.: Active multimedia documents. Proceedings of the Fifth International Conference on Document Analysis and Recognition 1999 (ICDAR '99). (1999) 21–24
17. Chang S-K., Znati, T.: Adlet: an active document abstraction for multimedia information fusion. Transactions on Knowledge and Data Engineering, IEEE, Volume: 13 Issue: 1 , (Jan./Feb. 2001) 112–123
18. LaMarca, A., Edwards, K. Dourish, P., Lamping, J., Smith, I., Thornton, J.: Taking the Work out of Workflow: Mechanisms for Document-Centered Collaboration. Proceedings of the European Conf. Computer-Supported Cooperative Work ECSCW'99. (1999)
19. Bluetooth wireless technology <http://www.bluetooth.com/>