Mobility Management and Roaming with Mobile Agents

Do Van Thanh[1], Sverre Steensen[2], and Jan A. Audestad[3]

[1] Ericsson Norway Applied Research Center, P.O box 34, N-1361 Billingstad, Norway, Tel: + 47 66 84 12 00
etodvt@eto.ericsson.se
[2] University of Oslo - Unik, P.O box 70, N-2007 Kjeller, Norway, Tel: + 47 63 81 45 70
sverrest@ifi.uio.no
[3] Telenor AS, P.O box 6701, N-0130 Oslo, Norway, Tel: + 47 22 77 99 52
Jan.Audestad@telenor.no

Abstract. In this paper we propose that the mobile agent concept can be used in the design and implementation of ideal user mobility. The paper starts with a discussion of the traditional mobility concepts; terminal and personal mobility. The concept of ideal user mobility is introduced which can be seen as an extension of these two concepts better suited for the needs of the mobile user. Then, the fundamental mechanisms for mobility support are described and how they can be implemented by mobile agents. Strategically mobile agents which can combine all these mechanisms in a flexible way is then discussed. The paper concludes with a summary of why mobile agents are well suited to provide mobility support and some suggestions for future studies.

Background:
The work described in this paper is a result from the mobility part of the TELEcom REsearch Program TELEREP, carried out at the Ericsson Norway Applied Research Center (NorARC) in collaboration with the University of Oslo, Center of Technology at Kjeller Unik [1]. The program has been established to obtain practical experience with Distributed Object Computing methods such as ODP/TINA [2], [3], [4], CORBA[5], mobile code in the implementation of TMN and IN functionality, and to study how mobility and security can be provided in the DPE environment.

1 Introduction

With the growth of the computer industry computers are showing up everywhere. Today most people interact with a number of different computers daily, at their job, at home or when travelling. All these computer users know that taking the mobile computer on the road or accessing a remote computer is something completely different than doing work from their normal environment. In other

C.G. Omidyar (Ed.): MWCN 2000, LNCS 1818, pp. 123–137, 2000.

words, the situation for the mobile users are far from being ideal. Problems like missing applications, unfamiliar user interfaces and unavailable services are normal. A user changing terminals and network access points is most often facing a frustrating and time consuming task to access services that are normally used.

There has been much optimism surrounding the concept of mobile agents. Mobile agents like the users move around in heterogeneous networks. With a *mobile agent* we mean a computer program that acts autonomously on behalf of a person or organization, and is not bound to the system where it began execution, as defined by the MASIF standard [6].

It is the intention of this paper to combine the user's need for mobility with mobile agent technology witch has great promise and built in mobility concepts that has not yet been realized in large commercial applications. In order to do so we will first examine general mobility concepts, before the fundamental mobility mechanisms used to implement these mechanisms are presented. Then the concept of strategically mobile agents, and how each of the mobility mechanisms can be supported by agents, are discussed.

2 Mobility Concepts

This section will present the concept of terminal and personal mobility, which can be achieved independently of the technology used for the implementation. However, implementing these mobility concepts are not enough to provide user friendly mobility services. We therefore propose the concept of ideal user mobility to extend terminal and personal mobility.

Terminal Mobility
Terminal mobility is the ability to provide service to a terminal that moves around, or is connected to a variety of network access points (NAPs). To be able to access the network the terminal has to be connected to a NAP. To provide network access there must exist an association between the terminal and the NAP. This association has to be dynamic to allow for terminal mobility. The NAP has physical characteristics that restrict the kind of terminals that can use the NAP and the type of services that can be provided. In telecommunication, terminal mobility has been supported by keeping the user's current address continuously updated at a database located at the user's home domain, as is the case for Mobile IP and GSM. This allows for terminal mobility, but the services that the users can access have been very limited. The computing industry has had more trouble supporting terminal mobility. On a large part the management of addresses has been left to the users themselves, which limits the usefulness of terminal mobility.

Personal Mobility
TINA-C defines personal mobility as the ability to let the user access services from any terminal at any location [7]. ETSI takes this definition a little further

and states that access to a terminal is based on a personal telecommunication identifier, and the network must provide services according to the user's service profile [8]. We will use the ETSI definition in this paper. To be able to provide services to a user connected to a terminal on a network there must exist an association between the user and the terminal. This association must be dynamic in order to ensure personal mobility. A user might have a profile. This profile records the user's preferences and there must exist some compatibility between this profile and the capabilities of the terminal. For instance if the user requires a video session, but the terminal can only offer voice capabilities or services, some compromise must be made, or the terminal must reject the request for service. What personal mobility really ensures is that the user can access a set of basic services across the network from a number of different terminals connected to a variety of NAPs. However, these basic services are not enough for most users. They also need to have access to their customized applications, data and profile. The term personal mobility does not ensure this and it is up to the users themselves to expand the services offered.

Ideal User Mobility
Even when terminal and personal mobility are combined the mobile user might still not be satisfied. We propose the concept of ideal user mobility which incorporates the concepts of terminal and personal mobility while adding transparency on top of these concepts and extending the number of applications, data and profile that can be accessed. Ideally the different computing environments should adapt to the users instead of the users having to adapt to them. *Ideal user mobility* can then be seen as the ability to provide the same applications, data and profile transparently to the user anywhere, anytime and with the same look and feel. The concept of ideal user mobility used in this paper has some analogies to the UMTS's Virtual Home Environment (VHE) concept. It is defined as a *system concept for personalized service portability across network boundaries and between terminals* [9], [10]. However, the VHE concept focuses only on service while our concept of ideal user mobility comprises of services/applications, data and profile.

Some alterations to ideal user mobility can be accepted if the terminal used to access the network clearly does not have the capabilities needed to provide the requested services. By capabilities we mean limitations in memory, processing power, display capabilities etc. If this is the case a negotiation between the user and the terminal should decide what services can be offered. For instance if a user accesses the computer network through a cellular telephone, the obvious limitations in display and computing capabilities restrict the available applications that can be accessed, and the degree of ideal user mobility that can be achieved. Such an alteration will be more acceptable to the user because of the obvious limitations of the terminal.

3 Fundamental Mobility Mechanisms

When the user is away from the home domain at a visiting domain there are several alternatives to enable access to the applications, data and profile. After studying systems offering mobility like GSM, UPT, Mobile IP and Telnet, the fundamental mechanisms used to provide mobility are classified into two categories:

Establishment of a Channel from the Visiting to the Home Domain
This is a fairly simple approach which resembles how traditional telephony works. A channel is established between the visiting and home domain and data is transported over the channel. This can possibly give the user access to synchronously and asynchronously communication applications, computational applications, user data, and the profile stored at the home domain.

1. *Continuously open channel:* A continuously open channel is a channel that after it has been established, remains open whether it is used or not. The user is then guaranteed to have a certain amount of bandwidth available at all times, whether the channel is used or not. This approach is adopted by for instance Telnet, where a channel is kept open after the two Network Virtual Terminals have negotiated the available services. The channel is not closed until the user requests it, or some network failure occurs. This approach is well suited if the user needs quick access to his home environment, and a fairly stable connection can be established between the two domains. This mechanism is best suited for synchronous communication applications. The disadvantages of this approach are that it can be very cost inefficient to keep an open channel between the home and visiting domain. In addition security restrictions on both sides can restrict the environment that can be accessed through the open channel. If the channel has low capacity for data transmission (which is often the case), the open channel approach can be too slow, especially if the user requires fast interaction with applications on his home domain.
2. *Discontinuously open channel:* Instead of keeping a channel continuously open, a channel can be established only when the user is about to send or receive data. After the final bit of the transmission has been sent the channel closes. Actually such a discontinuously open channel is only a virtual channel, and transmission is usually based on "best effort" delivery of data packages. Mobile IP employ this approach. The use of a discontinuously open channel is well suited if the user only needs access to his home domain for limited periods, or if he wants to be notified and receive incoming data when it is sent to his home domain. This approach is best suited to support asynchronously communication applications. This approach faces problems from low quality network connection, and possibly slow interaction with home applications. By its nature the discontinuously open channel approach is not well suited to support synchronous communication applications since transmission is normally based on best effort.

Duplication of Applications, Data, and Profile
This approach has been extensively used by both the telecommunication and computing industry. Traditionally applications, data files and profile have been moved between computers on storage mediums before they were installed on a new computer. This was a tedious task requiring extensive knowledge from the users. The duplication mechanism can be split into two:

1. *Pre-duplication:* With pre-duplication we mean that the applications, user data and profile are copied and installed at the visiting domain before the user arrives. Pre-duplication can again be split into two:
   (a) *Static pre-duplication:* When applications, data and profile are copied to the visiting domain before the user arrives and remains after the user has logged off, we have static pre-duplication. In GSM the Basic service and some of the supplementary services are statically pre-duplicated. These services are standardized and are identical on all hosts. Advantages of this approach are that the applications that are duplicated this way, will be quickly available to the users when they register at the visiting domain. The users can also rely on these particular services being available almost everywhere. This approach is best fitted for supporting applications that are used by a broad range of users, and are large and complicated. Disadvantages of this approach are that they are not very well suited for giving access to user data or profile. Nobody wants to have their files spread all over the network, and inconsistency between all these files could become a large problem. Standardization is often required to achieve effective static pre-duplication, but this process is normally time consuming and will increase a new service's time to market. Changing existing services is difficult since applications are distributed on many different hosts.
   (b) *Dynamic pre-duplication:* If the user specifies where he is going and when, his applications, user data and profile can be transported to that domain ahead of him. When the user leaves the visiting domain the applications, data and profile are destroyed or transported back to his home domain. We do not know of any systems currently supporting this form of duplication. Advantages of this approach are that the applications, data and profile are transported ahead of the user. This ensures a quick initialization process when the user logs in. When logged in the user can interact fast and locally with his application. This approach is best suited for potentially large applications that the user needs access to but will not wait for while it downloads. A disadvantage of this approach is that it requires security mechanisms to be in place so that the user's data and profile will not be spread around in the network.

2. *Dynamic duplication:* Dynamic duplication takes place after the user has registered at the remote host. It is a process where applications, user data and profile are copied from the user's home domain to the visiting domain. This form of duplication requires that there is some sort of platform interoperability between the two domains, and that there can be allocated enough

resources at the visiting domain. GSM uses this approach to move the user profile from the home to the visiting domain. Advantages of this approach are that it might be an excellent alternative if the user is going to access the network over a long period. It is then very cost efficient, while providing fast interaction with applications and data. Dynamic duplication is well suited for computational applications. Disadvantages of this approach are that it might be impractical to use if the amount of data to be transfered is very large. It would then require extensive transmission which can be a time consuming process. It does not support communication applications well.

It is not possible to conclude that any one of these mechanisms are superior to the others. They all have their advantages and disadvantages and are well suited for some situations and applications but not for others. Obviously a combination of these mechanisms will offer the best solution. This combination should be flexible adapting to the user's needs and situation, in order to provide ideal user mobility. Any concept that permits a dynamic and flexible combination of these mechanisms will be suitable for the realization of ideal user mobility.

4 Mobile Agents Supporting Mobility

So far very little research has been done on how mobile agents can support mobile users. Most mobile agent applications suggested have been based on fixed users sending out mobile agents to accomplish tasks like information searching and electronic shopping. Gray et al. have suggested that mobile agents can offer advantages to the mobile user [11]. They focus on mobile users that are partially connected through often unreliable network connections with laptops or personal digital assistants (PDAs). In our terminology they are investigating ways to support terminal mobility and their focus has been on how to make efficient use of network resources. Through experiments they demonstrated that mobile agents can leave laptops and return despite disconnection and reconnection at different IP-addresses. The purpose of this paper is to see how mobile agents can implement the mobility mechanisms discussed in the previous section. Before these implementations are presented the most central agents are presented briefly:

Domain Agent(DA)
At every domain supporting our implementation there should exist one DA. It is created before any other agents can populate the domain, and is a stationary agent remaining at the same host for its lifetime. Its main responsibility is to manage the domain's resources and enforce security restrictions.

User Agent (UA)
The UA is the user's main representative in the system. It could be created when the user subscribes to the system, or by the user through an agent generator program. The UA is responsible for providing services to the user based on the user's specifications. It needs to communicate with the user, the DA and

the other agents and should therefore have advanced communication abilities. When it migrates it clones itself so a copy is always left on the home domain. This ensures that the UA is not lost or destroyed completely if things go wrong during migration.

User Data Agent (UDA)
The UDA is responsible for collecting the user's data files according to the UA's specifications. These files can be spread at the user's home domain and are copied and collected when the UDA is sent for. The reason for only dispatching copies of the files instead of the original data files are two fold: First, if something goes wrong during transfer the data can be permanently lost. Secondly if only a copy is sent, it would be unnecessary to return data files that are unchanged. A better solution is just to destroy those files remotely. The UA therefore only contains the logic necessary to collect those files, not the files themselves.

User Application Agent (UAA)
The UAA is like the UDA created by the user or UA, and remains de-activated until the UA retracts it. The UAA contains logic or program code, making it able to collect the necessary files to provide the applications specified by the UA.

User Profile Agent (UPA)
The UPA is created by the user or UA, and as the other task agents remains de-activated until it is retracted by the UA. The UPA is responsible for collecting the customizations that the user has done to the applications and the subscribed services the UA has asked for. There would be no point in bringing the user's entire profile over if only some services and applications are needed. The UPA should therefore collect that part of the profile that is best adapted to that particular domain.

Mobile Agents Supporting Dynamic Duplication
We will now present how mobile agents can be used to implement dynamic duplication. Our basic model can of course be extended to offer more sophisticated services but we believe that this model presents the core concepts. We have presented this model in figure 1. To get a better understanding on how dynamic duplication can be implemented we will go through the model based on the numbers in the figure.

1. When a user logs in at a terminal connected to the network the DA presents him with a graphical user interface (GUI). At the GUI the user should provide his user identifier and password. In order to support a transparent login scheme the user ID needs to be valid at all hosts supporting the agent system. The identifier can be a combination of the user's domain address and a user identifier, or an independent identifier not based on the user's home domain's address. When the DA knows the user's home domain it retracts the user's UA.

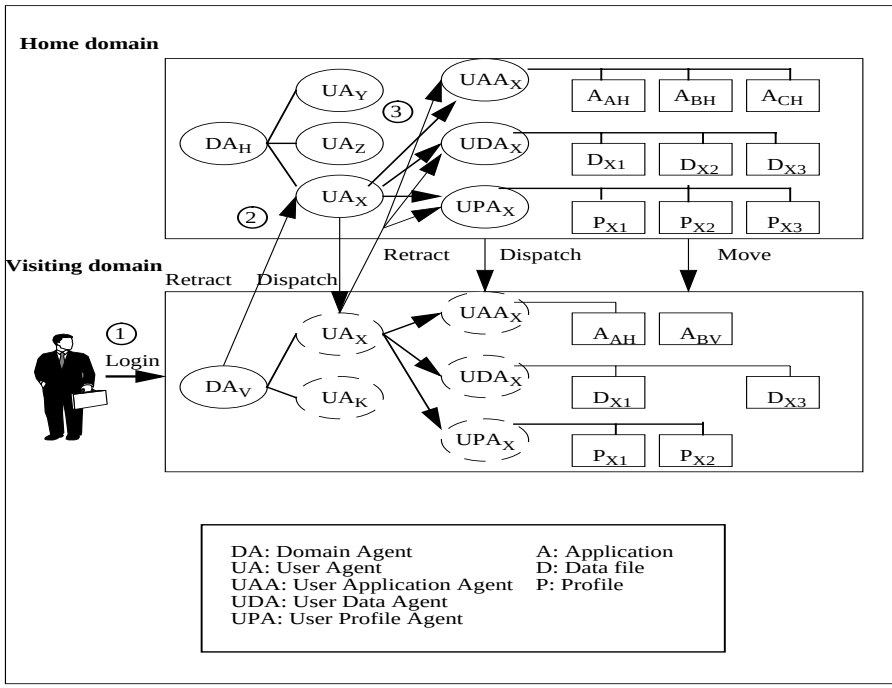

Fig. 1. Mobile agents supporting dynamic duplication

2. When the UA receives the retract call it should migrate to where the migration call originated. After arriving at the remote host it registers with the DA. At this point the quality of the user's session should be negotiated including access rights to resources, allocated memory, CPU cycles and physical memory. The UA then presents these restrictions to the user along with a GUI. From the GUI the user decides what applications, data and profile he needs. The user can decide directly by naming what to bring explicitly, or more indirectly based on criteria such as: The most recently used files, files on certain formats, files belonging to certain projects etc. As we can see from figure 1, application B is already present at the visiting domain so there is no need to bring over that application. The user does not need application C, so it is not brought over. When the user has finished specifying what is needed, the UA is responsible for retracting the specified agents.
3. Agents receiving the retract call collect the specified files which might possibly be spread around the user's home domain. The reason for not collecting all the files, when the agents are initialized, is that the file's content might change in the meantime. These files are loaded into the agent's memory before the agents dispatch to the visiting host. When arriving at the visiting host, the agents write their data to files and distribute them in a way so that the user can have easy access to them. The applications, data and profile are now available to the user.

When the user decides to log off he does so by clicking on the log off button on the GUI presented by the DA. The DA then sends a dispose call to the UA, which sends dispose messages to the other agents. When receiving the dispose call the agents examine the files they brought over to see if they have changed. If they have, they are transported back to the user's home domain by the appropriate agent. If no changes have occurred the agents are garbage collected at the visiting domain. After the application, data and profile agents have been disposed the UA is disposed and garbage collected.

The dynamic duplication mechanism gives the user access to his computing applications, data and profile. The user can get the communication applications duplicated as well, but it would only be useful for outgoing communication. Incoming communication would still be routed to the user's home domain and duplicating the applications will not be sufficient to receive communication. However, it takes time to dynamically duplicate which may be frustrating to the user.

Mobile Agents Supporting Dynamic Pre-duplication
In the dynamic duplication process the user has to wait while the environment is being initialized. This process could possibly take a long time, depending on the quality of the communication channel and the size of the data being transported. If the user knows where he is going and at what time he will be arriving at a certain domain, the pre-duplication mechanism can be employed. His agents can be sent ahead of him initializing his environment and preparing the domain for his arrival. When the user then logs in, his environment will be initialized quickly. We have not illustrated the design of this scenario since it is based on the dynamic duplication model.

When the user is at his home domain knowing that he will need to access a visiting domain later, he can have his agents migrate before him. In order to achieve this he needs to specify the visiting domain's address and when he will be there. This could be done each time the user travels, or the UA could have access to the user's time manager and follow that timetable. When the user is deciding on what applications, data and profile to bring, there should at least exist two options:

1. The user can specify exactly what applications, data and profile he wants access to. The UA then migrates to the host and negotiates with the DA to achieve the best possible environment for the user. After the negotiation has taken place the UA retracts the other agents. The rest of the process is similar to dynamic duplication.
2. Instead of the user having to specify exactly what to bring over, the UA can be responsible for negotiating the best possible environment based on a pre-defined strategy. The agents are then retracted in the same way as in dynamic duplication.

When the user arrives and logs in, his environment is presented. If the user somehow does not show up, the UA should have a timer that times out resulting in

garbage collecting of all files brought over. This would prevent against the user's files being spread around the network. As with dynamic duplication changes to the files are transported back to the user's home domain by the appropriate agents.

As we have seen pre-duplication relieves the user of the time consuming initialization process. On the other hand it also removes some of the flexibility and adaptiveness of the dynamic duplication process, since the user must specify where and when he will be visiting the remote domain. This approach could be extended further if the user has certain patterns of movements. As with dynamic duplication it does not support access to communication applications satisfactorily.

Mobile Agents Supporting Static Pre-duplication
Static pre-duplication is the process of copying and installing applications, data and profile to a new domain and let them remain there after the user has logged off. As discussed under network management this is an often mentioned application area for mobile agents. The mobile agents could be used to install for instance certain applications at nodes in a network or between different domains. However, static pre-duplication can for all practical purposes only be used to pre-duplicate applications. Having user data and profile statically pre-duplicated will limit the user's privacy, and it will be very difficult to keep the data and profile consistent between nodes. Static pre-duplication can be based on the dynamic pre-duplication mechanism with the only difference that what the agents brought over remains at the domain after the user has logged off. After the agent has installed the applications at the domain it could return to its home domain or travel to other hosts, with the same applications.

Mobile Agents Supporting a Continuously Open Channel
A system based on mobile agents can also be used in establishing a continuously open channel between the home domain and the visiting domain. We have presented an overall model for this design in figure 2. We will now go through the numbered steps:

1. The initialization phase is identical to that of dynamic duplication. The user logs in at a remote host and with his normal user identifier and password.
2. The DA at the visiting domain then retracts the UA from the visiting domain. When the UA arrives it presents a GUI to the user with the option *Establish a continuously open channel*. The user then selects this option.
3. The UA then negotiates with the DA or possibly a Network Agent (NA). The NA is responsible for ensuring a certain quality of service (QoS) on the network connection. This can be defined through parameters such as throughput, transit delays, security of communication, priority etc. After the negotiation the UA is allowed by the DA to establish a connection to the user's home domain with the negotiated QoS. The UA located at the user's home domain is notified about the resources available at the visiting

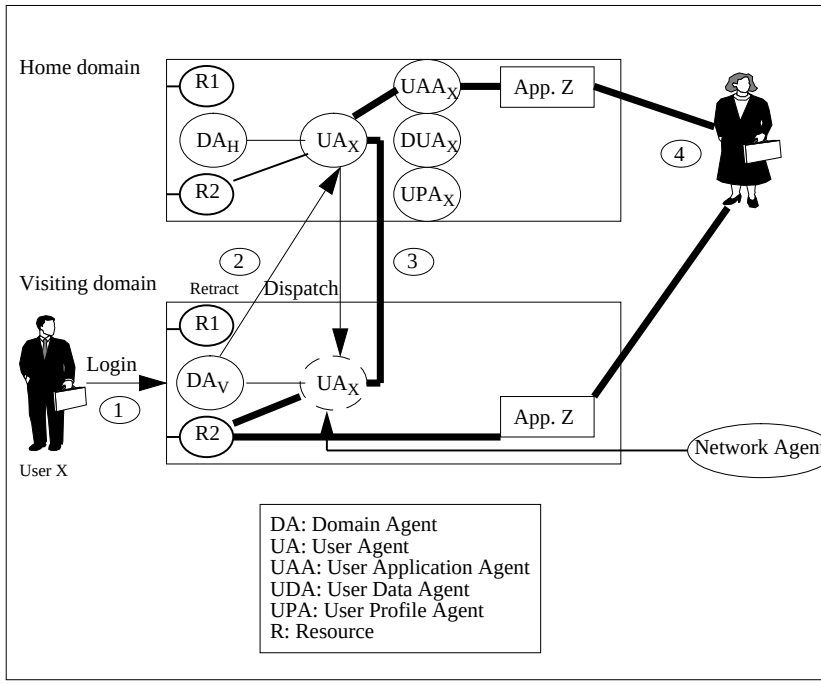

Fig. 2. Mobile agents supporting a continuously open channel

domain. This ensures that if someone wants to communicate with the user by for instance video conferencing, and those capabilities do not exist at the visiting domain, the request for communication could be denied or some compromise should be made. The channel is then established with the original UA left at the user's home domain and the migrated UA functioning as endpoints. The UA then, rather than the user itself, is responsible for maintaining the connection. If for instance the connection goes down, the UA should re-establish the connection transparent to the user. Also if the connection is characterized by bursty network traffic, the UA should be able to dynamically reserve more network resources to accommodate that traffic by negotiation with the NA.

4. If someone wants to communicate with the user the user's normal home address will be used. When the request reaches the user's home domain it is received by the appropriate application which forwards the communication to the UAA. It again contacts the UA. When receiving the request for communication the UA should consider the following:

   (a) If the user is logged in at his home location and the resources needed are available, communication is routed to the appropriate resource and communication can be engaged.

(b) If the user is not logged in at the home domain and the UA has no current address of the user, synchronous communication should be denied while asynchronous communication can be accepted.
(c) If the UA has the current address of the user and there is established a continuously open channel between the home and visiting domain, the UA should at least have two options:
   i. It could return the user's current address. In doing so a direct channel can be established between the two parties, circumventing the user's home domain. For this to be possible there has to exist necessary applications and resources at the visiting domain.
   ii. The UA at the home domain could also establish a channel on behalf of the user between the requester and the home domain. This is similar to the approach taken by Mobile IP, but here the triangle routing depends on an open channel instead of a discontinuous one. This approach requires that there are appropriate resources at the visiting domain. However, since the communication goes through the appropriate application at the home domain, similar applications does not have to exist at the visiting domain.

The UA on the home domain can then be seen as responsible for forwarding communication intended for the user sent to his home domain. The UA at the visiting domain is responsible for sorting the incoming communication according to what application type it belongs to and the resources it needs (R). For instance if the user is talking to someone on an IP telephone and he wants to establish a text based *talk* session with someone else, the UA will be responsible for splitting incoming data to its appropriate resource. In figure 2 we see that the User Agent routes the incoming data to resource *R2*.

This approach gives the user access to his computational and communication applications but in a much slower fashion than if it was accessed locally. Continuous interaction can then become a tedious and frustrating process. This approach can give the user access to his applications, data and profile but at the cost of keeping an open channel and slow interaction with applications at the user's home domain.

Mobile Agents Supporting a Discontinuously Open Channel
A discontinuously open channel can also be supported by mobile agents. This approach is based on the way mobile IP works, but while mobile IP only provides the user with incoming data, this will in addition provide the user with access to his home environment. We have not illustrated this mechanism because it closely resembles the continuously open channel approach.

When the user arrives at the remote host he retracts his user agent as usual. When the UA arrives it presents a GUI to the user and a discontinuously open channel is decided on. The UA then sends a message back to the UA at the home domain notifying it of its choice. The user can then start working at the

host. When computational applications, data or profile are needed he indicates it to the UA who opens a channel to the home domain. This channel is shut down when the user decides to do so, or if the connection has been inactive over a longer period. If someone wants to communicate with the user, either synchronously or asynchronously, a channel is automatically established so the communication can proceed. When the user decides to log off, the UA is garbage collected.

As for the open channel approach, the user has access to computational and communication applications, data and profile. But the same problems of delays can be experienced due to remote access. However, the cost of keeping an open channel is reduced.

Combining All the Fundamental Mobility Mechanisms
We have seen that the mobile agent concept is capable of supporting the five fundamental mobility mechanisms. However, the strongest feature of mobile agents is the ability to provide a combination of all five mechanisms. A user might need:

- One or more synchronous communication applications.
- One or more asynchronous communication applications.
- One or more computational application.
- Data and profile.

In such a case all the fundamental mobility mechanisms should be employed to provide the best possible solution for the user. To illustrate how these mechanisms can be combined we consider a user moving to a visiting domain:

- Standard computational applications may have been statically pre-duplicated. The UA discovers this when arriving at the visiting host and there is no need to dynamically duplicate those applications.
- Synchronous communication applications can be moved or left at the user's home domain depending on resources present at the visiting domain and the quality of a possible continuously open channel.
- Asynchronously communication applications can also be left at the home domain and can be accessed through a discontinuously open channel.

5 Strategically Mobile Agents

As we have illustrated in the previous sections the different mobility mechanisms have their strengths and weaknesses and combining them would obviously provide best mobility support for the users. However, what constitutes such a combination for providing *ideal user mobility* is very dependent on the situation and the user's preferences. It is therefore essential to device a strategy that combines these mechanisms, in order for the agents to provide ideal user mobility. We define strategically mobile agents as agents optimizing their performance on tasks according to some strategy. This strategy can consists of:

Decision heuristics: Decision heuristics are the decision rules the agent constitutes when deciding on what mechanisms to employ. These rules are specified by the agent's owner in combination with general knowledge about distributed computing and must be decided on before the agent migrates.

Task knowledge: It is not enough to only have static decision rules on how to mix the mobility mechanisms. This would result in very little flexibility for the users. Information about the specific tasks that the user wishes to accomplish is needed to provide a flexible strategy.

Environment knowledge: The last piece of information needed to fully exploit strategic mobility is to have knowledge about the surrounding environment. No matter how good a strategy is, it will not make sense if it is not adopted to the environment. The environment poses certain restrictions on the execution of mobile agents. These restrictions can be split into *static* and *dynamic* restrictions. The *static restrictions* are initial restrictions put on mobile agents before they are allowed to execute. These restrictions are valid as long as the agent is executing at the host. *Dynamic restrictions* on the other hand change over time, according to what happens in the surrounding environment.

6 Conclusion

We believe that mobile agents, in addition to supporting terminal and personal mobility, can provide ideal user mobility. Although we have not found any research on this topic we believe that agent technology provides a useful framework that can be applied in this respect. The strongest arguments for employing agent technology to support ideal user mobility are:

- **Built in support for mobility:** Mobile agents are by their very nature mobile and all agent platforms have built in support for security although limited, and easy to use primitives for migration. Like other distributed technologies mobile agent technology hides the specific details of the underlying network. In contrast to other technologies that also hide the location of the objects, mobile agents explicitly use the location awareness to offer better services.
- **Encapsulation of code and data:** A requirement for ideal user mobility is that the user's applications (code), data (data), profile (code and data) are provided transparent to the user independent of the user's location. Mobile agents can support this requirement since they encapsulate both code and data. This is a core characteristic of mobile agent technology and is a strong argument for using it.
- **Powerful abstraction:** The agent abstraction is powerful to use for the designers and programmers, but also for the end users. Different tasks can be delegated to appropriate agents that are responsible for carrying them out. By explicitly exploiting location awareness which is a criterion for strate-

gic mobility, it can represent a better abstraction than traditional object-oriented methodology.
- **Defined models:** When mobile agent technology matures it is likely that the fundamental models we have discussed in previous chapters will be better defined and offer an established framework to the developers. Functionality like security mechanisms, naming schemes for resources, host and agents addressing, will make it a well suited technology for a heterogeneous distributed environment. When these models are in place it will ease the development of applications built on mobile agents

Mobile agents seem to be a promising concept that has the flexibility and the adaptiveness required to meet the demands from the mobile users. However, in order to be really usable, the mobile agent concept relies on the availability of technologies such as Java chips, Jini etc. which can equalize the introduced overhead. In addition, there are several issues that also need to be resolved. One crucial issue is security. It is necessary to protect the visiting site from malicious agents and reciprocally, the agent from pirate sites. The user must also be protected so that private files are not spread around the network. Last but not least, since the mobile agent concept is still new, standards are still lacking despite the effort from OMG and FIPA. This can lead to incompatibility between the different platforms and lead to slower acceptance of mobile agent technology in general.

7 References

[1] Pål Spilling. The telecom research program telerep at Unik. UNIK Aarsrapport 1996,1996.
[2] TINA-C. Overall Concepts and Principles of TINA, February 1995.
[3] ITU-TS. Basic Reference Model of Open Distributed Processing - Part 1 Overview and guide to use the Reference Model. Rec.X901(ISO/IEC 10746-1).
[4] ITU-TS. Basic Reference Model of Open Distributed Processing - Part 2 Descriptive Model. Rec.X901(ISO/IEC 10746-2)
[5] OMG. The common Object Request Broker. Architecture and specification - Rev 2.2 February 99.
[6] Dejan Milojičić and others. MASIF: The OMG Mobile Agent System Interoperability Facility. Mobility Processes, Computers and Agents
[7] TINA-C Glossary of Terms, Version 2.0, 1997
[8] ETSI, Global Multimedia Mobility: A Standarization framework for Multimedia Mobility in the Information Society, 1996
[9] 3GPP. Universal Mobile Telecommunications Systems (UMTS); Service Aspects; Virtual Home Environment (VHE) UMTS 22.70 version 3.0.0.
[10] 3GPP. Universal Mobile Telecommunications System (UMTS); Provision of Services in UMTS - The Virtual Home Environment (highlighting release 99 requirements UMTS 22.21 version 1.0.
[11] Robert S. Gray and others (1996). Mobile Agents for Mobile Computing. Technical report, Dartmouth College.