Mobile Host Tracking and Resource Discovery

Aline Baggio & Ian Piumarta

INRIA, SOR Group
B.P. 105, 78153 Le Chesnay Cedex, FRANCE

E-mail: *Aline.Baggio@inria.fr, Ian.Piumarta@inria.fr*
URL: *http://www-sor.inria.fr/*

Abstract

In *mobile environments*, as computers move to unknown networks, they need to discover new services, applications, and other network resources. Since the performance characteristics of such environments are often poor (due mainly to wireless communications and the restricted power of machines), mobile hosts require access to the nearest equivalent of some resource. On the other hand, services and applications located on the fixed part of the network may need to be aware of mobile host locations, in order to redirect messages, replies, references, files, displays, and so on.

To achieve such *resource discovery* and *mobile host tracking*, a possible solution is to use a mechanism that supports remote object references — *Stub-Scion Pair Chains* — extended to allow host movement, disconnection and reconnection. Given this mechanism, we are able to install on both the fixed and mobile parts of the network some dedicated objects that allow the management of resource accesses and locations of mobile hosts, via tables and chains of references.

1. Introduction

Mobile environments have particular characteristics (hardware, communication media, etc.) that differ significantly from fixed network architectures [Forman 94]. They require techniques to ensure a complete independence of movement (disconnectability, location independence, *resource discovery* in newly visited networks, and some degree of execution transparency). Resource discovery poses two difficult problems: discovering which resources are available or appropriate for a mobile host at some given location, and the management of the locations of (and, implicitly, the communication with) these resources. In this paper we concentrate mainly on the second of these problems, and present briefly a way to manage resources (services or applications) in the presence of mobility.

In such an environment we assume that our machines are portable computers (i.e. *mobile hosts* rather than personal assistants, pagers or mobile phones), that they have reasonable capabilities (such as processor power, memory or storage), and remote access and execution mechanisms (such as *Remote Procedure Call*) so that fixed stations can run any applications which are unavailable on mobile hosts.

We also assume that there are fixed stations which can communicate with both the fixed (through a wired interface) and mobile (through a wireless interface) parts of the network, called *support stations,* and that a support station permanently responsible for a mobile host (keeping primary copies of its files, email, etc.) is called the mobile hosts' *home support station*.

This paper is divided into five main parts. Section 2 presents our object designation mechanism. Section 3 presents the model used to achieve *mobile host tracking* and resource discovery. Section 4 describes the resource discovery mechanism. Section 5 illustrates the mechanism with the example of discovering the nearest printer service. Lastly Section 6 explains the mobile host location management.

2. Basic referencing technique

To access objects stored on mobile and fixed hosts we need a remote referencing mechanism; in this work we are using *Stub-Scion Pair Chains* (SSP Chains) [Shapiro 92]. The main goal of SSP Chains is to provide transparent referencing of remote objects in a distributed system, as a way to resolve the classical problem of object designation between remote hosts.

Figure 1 shows the basic principles of SSP Chains. In this example we have two spaces (Space 1 and Space 2), containing objects (A, B and C) and references to local (&C) or remote (&B) objects (in the SSP Chain meaning, a space corresponds to a process, so we can have several SSP Chain spaces on the same host).

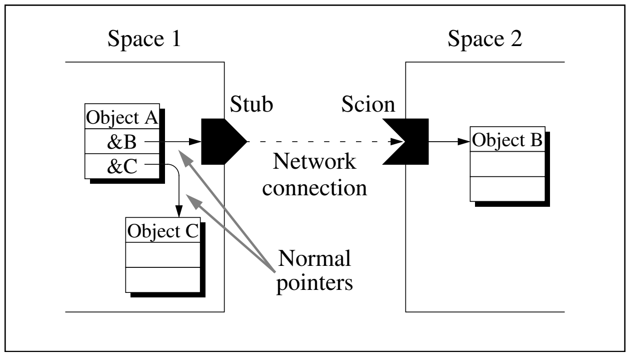

Figure 1: Stub-Scion Pair Chains mechanisms [Piumarta 95]

A *stub* represents an exiting reference to a remote object, and a *scion* an entering reference coming from a remote host to a local object. In Figure 1, the stub represents the exiting reference from Space 1 to object B of Space 2. The scion is the entering reference from object A of Space 1 to local object B.

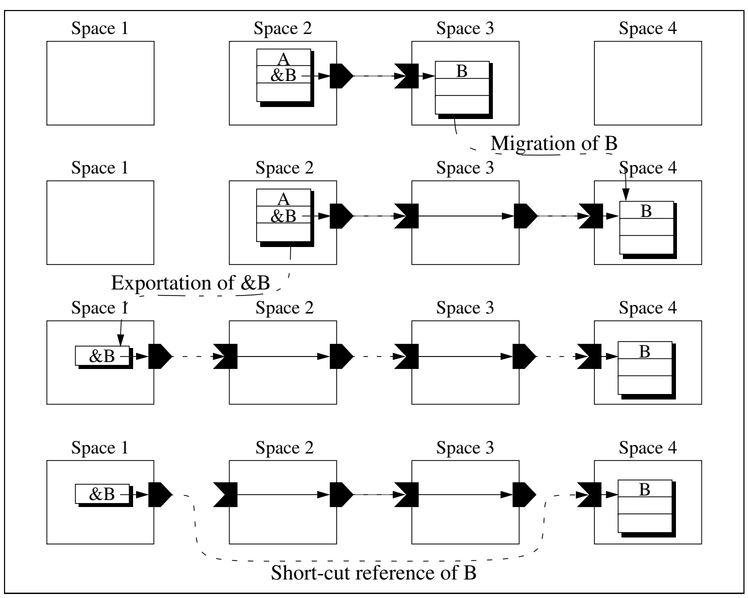

Figure 2: Extension and short-cutting of SSP Chains

Stubs and scions support Remote Procedure Call, i.e. are in charge of marshalling and unmarshalling of calls and replies, transparently from the applications' point of view.

In addition, stubs implement exactly the same interface as the object they replace. So for an object holding a reference it is usually not necessary to care whether the reference is to a local, or transparently to a remote, object.

SSP Chains support chain *extension* and their subsequent *short-cutting* [Shapiro 92] (Figure 2). Extension corresponds to adding stub(s) or scion(s) to either end of a chain. It is achieved by migration of an object or by exporting a reference to another space (from a support station to a mobile host, for example). Short-cutting corresponds to the creation of a new stub and scion pair to replace an indirect chain with a direct one.

Finally, SSP Chains can use a *binding protocol* to achieve an explicit linking of references. Only after binding are we able to transmit data between two spaces. This binding allows us to dynamically finish the establishment of the connection (such as creation of appropriate kinds of marshallers and unmarshallers), specify communication parameters (such as transport protocol, quality of service, actions to execute in case of host failure), and supports flexible binding (e.g. access to nearest equivalent objects or services). Flexible binding also supports the unbinding of references when the conditions which existed at the initial time of binding no longer hold — for example, during the movement of a mobile host between support stations.

3. The Support Station Model

Using a wireless medium, a mobile host can only directly communicate with its current support station (or stations, if several wireless mediums are used concurrently — radio-frequency and infrared for example). This creates a (transparent) indirection in the communications between a mobile host and a fixed station, as illustrated on the left of Figure 3.

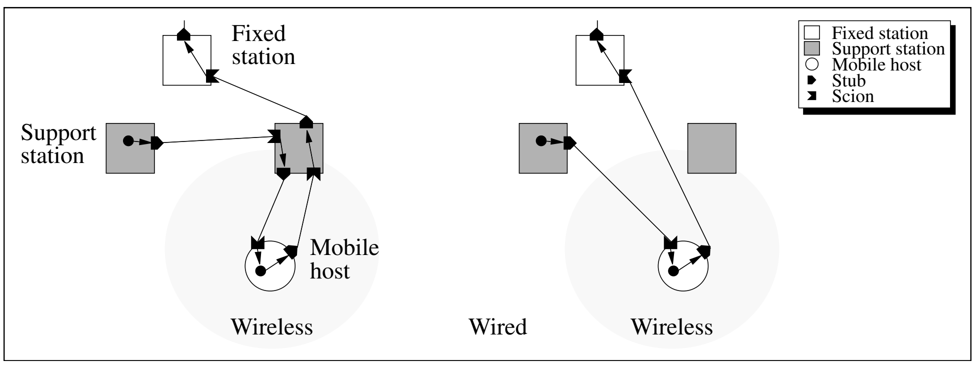

Figure 3: The support station model

Since we choose to manage mobility at the system level, so that we can take advantage of SSP Chains mechanisms, we need to preserve the indirection through the support station, despite the normal behaviour of SSP Chains which would be to short-cut the indirection (illustrated on the right side of Figure 3).

To prevent this unwanted short-cutting we add a special object called *gateway* on the support station; one gateway for each connected mobile host. Messages exchanged between the mobile host and some remote location are modified so that references to objects on the mobile host appear (to any outside process) to originate from the gateway, and (symetrically) any references from outside processes appear (to the mobile host) also to originate from the gateway.

This mechanism for preventing short-cutting has the important charateristic that all references to, or from, the mobile host pass through a single gateway object. We use this fact to good advantage in our reconnection protocols.

As shown Figure 4, the gateway resides in its own SSP Chain space, redirecting the messages according to their source scion. It eventually creates new pairs if these messages contain some references to remote objects, so forcing all SSP Chains to/from a mobile host to cross the gateway space. Moreover, as the mobile host moves, the gateway object(s) located on the support station simply follow it — this has the effect of redirecting all the SSP Chains that go throught the gateway object to the mobile host's new location. (Our protocols actually support two modes of "migration" for the gateways: "lazy" in which all the chains are extended towards a new gateway and later short-cut as messages traverse them, and "eager" in which the object effectively migrates, shortcutting all the chains as part of the reconnection process.).

This mechanism allows us to solve the indirection problem without making changes in the SSP Chain protocol, retaining the simplicity and transparency of SSP Chains. Also the gateway's functionality is kept as simple as possible, since it does not concern itself at all with mobility (our disconnection and reconnection protocols — which together manage mobility — are the responsibility of a different object, and are beyond the scope of this paper).

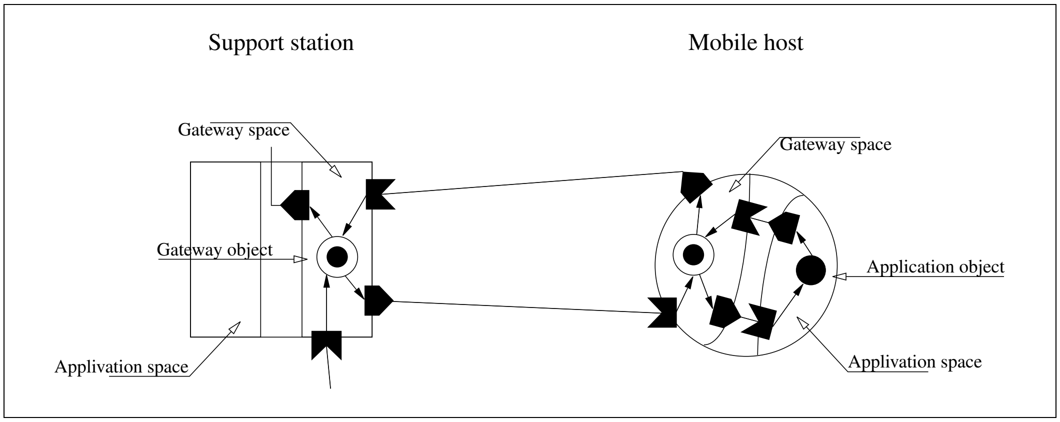

Figure 4: The gateway object

Another advantage of the gateway is that it implicitly multiplexes chains between a mobile host and its support station, using only a single physical connection for all chains.

Because of the particular needs of our protocols, we also install gateways on the mobile hosts. We therefore create a symetric structure with gateway objects both on support stations and mobile hosts. On each mobile host we do not strictly need to use a gateway object, since the necessary preservation of indirections is performed by the gateway on the support station. The use of a gateway on the mobile host does however make the management of reconnection significantly simpler. The gateway object works in such a way as to "repair" references to objects that migrate away from a mobile host, avoiding the possibility of "loops" (crossing the gateway, "looping" in the mobile, and then emerging back out of the gateway) in references between objects residing only on the fixed part of the network.

In the following figures, to improve clarity, we will represent the gateway space as a thick line wrapping the different hosts. However, it should be kept in mind that these lines actually hide a gateway object, residing in its own space, with an extra stub and scion for each reference which crosses it.

4. Resource discovery mechanisms

Armed with an extension to the SSP Chains mechanism which enables us to cope with mobile hosts (and the disconnection and reconnection of chains, which is not described here), we can create another mechanism above this which is dedicated to resource discovery.

Mobile hosts, as well as fixed stations, need access to network resources (file systems, name servers, printers, mail servers, etc.). Nowadays, portables or fixed stations are configured to use the resources of their own network (*home network*). In mobile computing, this method implies remote access through a wireless medium that already has very poor performance, and an increased risk of disconnection.

Mobile computing implies the management of localities which change as a mobile host moves. A typical example might be a mobile host connecting to the nearest server, which might cease to be the nearest server at any point during the subsequent dialog due to the host moving.

To avoid these numerous inconveniences, connections currently in use need to be transferred to local servers when appropriate. Using a flexible location (and communication) management protocol in association with a resource discovery protocol it is possible to access the most appropriate resource provider, and thus to achieve better performance transparency in addition to location transparency [Bhagwat 94].

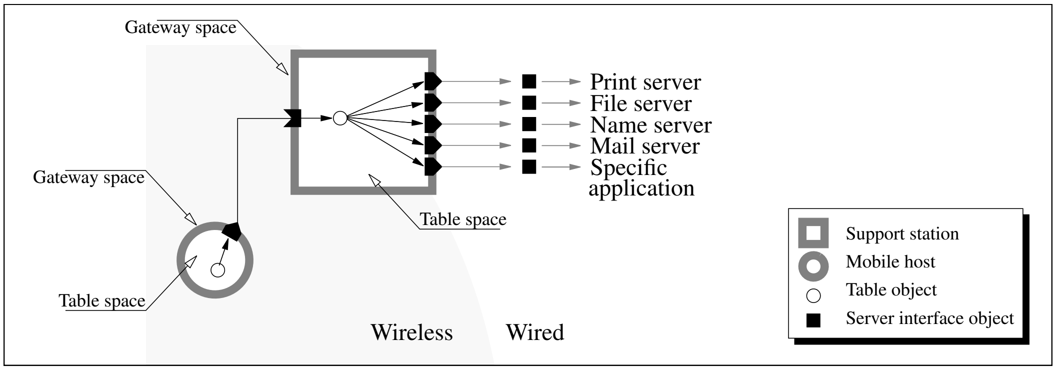

Figure 5: Resource discovery management on support station

To discover resources in a new network we choose to use a mechanism similar to UNIX resource retrieval, but which is able to discover dynamically resources in unknown networks. The mechanism therefore uses resource tables on both support stations and mobile hosts, which are modified *dynamically* according to service availability.

In our model each support station owns a two-part *resource table* object, which contains "pointers" to the available network services and applications that a mobile host might need. Table objects ensure the matching of service names with real network servers and maintain a set of references to these servers. Since these services are mostly remote, we use the previously described extended SSP Chains for referencing services on their fixed station, as shown Figure 5.

On a support station, one part of the table corresponds to the available services and mobile hosts currently connected on this support station. The other part contains pointers towards the mobile hosts that the support station is permanently "in charge" of.

On a mobile host, the resource table object is used as an access point to the support station's table object. One part contains the resources of the home network that must always be accessible. The other part is used for resources currently in use in the local network, or on some nearby network if the service is not locally available.

Now that we have outlined the principles of our resource discovery protocol, we can move onto the question of their application. This is the topic of the next section.

5. Resource discovery scenario

The simple example of use which we give in this section is the management of access to a printer service.

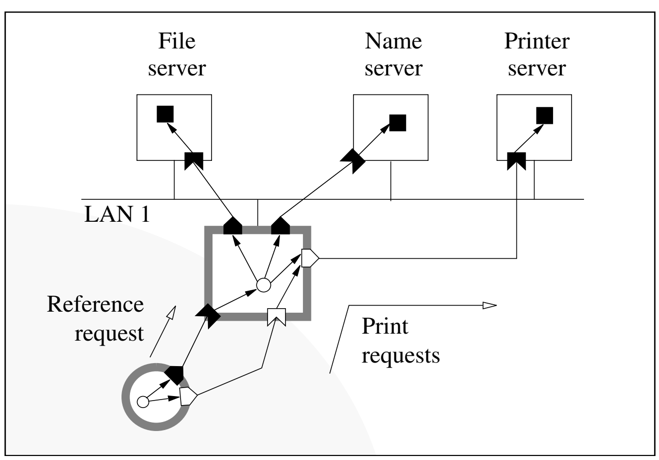

Figure 6: Indirect and direct accesses to print service

In Figure 6, the mobile host has just connected to a support station. Consequently it receives a reference to the table object of this support station, which allows it to retrieve and use the local network services.

When a local printer is needed, a request is sent to the support station resource table to obtain the reference of such a server. The support station resource table returns a print server reference back to the mobile host, creating a new SSP Chain (the white chain in Figure 6). Print requests can now be sent directly to the local print server, using the print service reference stored in the mobile host's resource table.

Now we consider what happens when the mobile host disconnects (electively) from LAN 1, and later reconnects in a new network LAN 2. The SSP Chain rebinding protocol will cause a renegotiation of the connection to the print service. The appropriate outcomes are to either reconnect to the same print service, or to connect to a different (possibly closer) print service.

Figure 7 shows how we can re-establish the link with the previous print server (on LAN1) if we wich to ensure continuity of the service. If an equivalent service is found to be locally available, and is considered preferable, then the link is redirected as a side-effect of rebinding. Figure 7 also shows how the previous remote link (white chain) can be redirected to the new printer (black chain).

This automatic redirection can improve the performance of otherwise remote access to services. But we can imagine cases where the user explicitly requires that the link to the previous printer be kept, or even redirected to a third printer in another remote network. We can also imagine cases where the network refuses redirection due to security or load considerations. The SSP Chain-based mechanism allows such a flexible use of binding, although a full discussion is beyond the scope of this very short paper.

More complex behaviours are possible using the SSP Chain binding protocol. For example it might be appropriate to instantiate a "proxy" server on the support station which acts as a "buffer" between the mobile host and other processes on the fixed part of the network. Such proxies might process data in some way — for example by reducing the resolution of bitmap images coming from a Web server to limit the amount of communication with the mobile host. Other possible uses are mobile file caches which maintain their state as they migrate to the support station on which the mobile host is currently connected after moving.

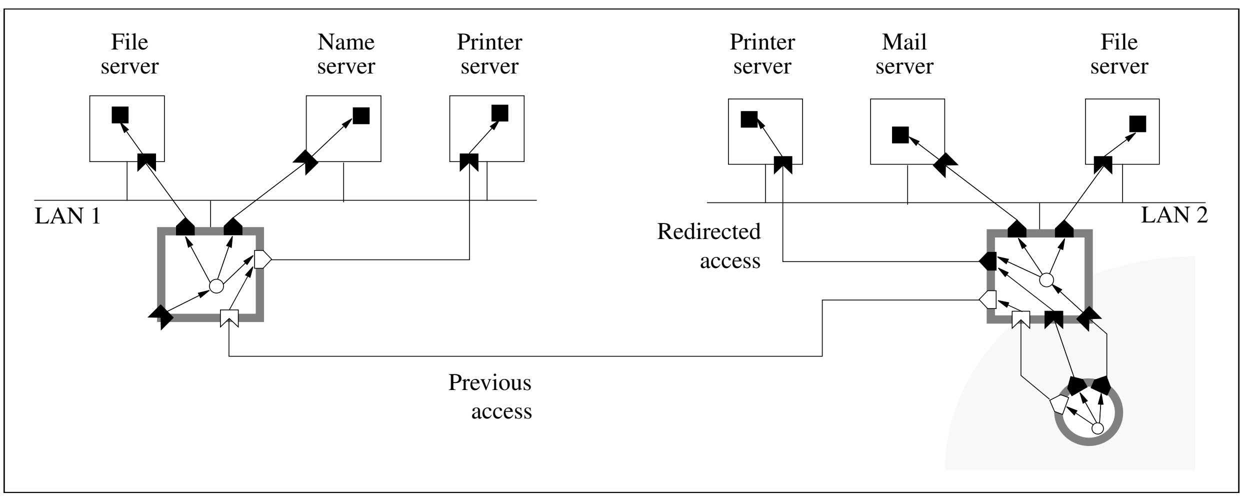

Figure 7: Reconnection and redirection of the link to server

6. Mobile host location management

A useful side effect of mechanisms described previously is that we are able to "track" a mobile host.

Mobile hosts move, run applications, use network services (print, naming, file services, etc.), send messages and requests, and so on. They might also act as "servers" themselves, for example when deployed in the field to gater data.

Considering these activities, mobile host location is indispensable simply to direct the replies to previous messages, or new requests for services on the mobile host, towards the mobile's current location.

In order to manage mobile host location we transparently use resource tables and support station gateways. As we said in Section 4, a home support station always has references to its mobile hosts, in the form of SSP Chains kept in the home station's resource tables.

When a mobile host moves, it extends the SSP Chain coming from its home support station which points towards its resource table object; i.e. it inserts a new stub-scion pair in the chain, between its previous and current support station (this is part of the disconnection/reconnection protocol, which interacts with the gateway objects).

When the mobile host's location is first needed after a move (i.e. a change of support station) the indirect chain through the old support station will be short-cut. Using these permanent access chains, other hosts are able to track each mobile host and directly send their messages to the correct location along the shortest path.

Therefore in such an environment the home support station always knows the current location of a mobile host, or the last known location if the host is currently disconnected. Using the gateway objects we can also determine if a mobile host is connected or disconnected, simply by knowing if the mobile's gateway object is reachable from the support station's gateway.

7. Conclusion and perspectives

Given a protocol that permits remote references to objects — such as SSP Chains — and additional mechanisms to support mobility, it is possible to manage accesses to resources and track mobile host location in a manner that is transparent to applications.

Our model proposes the use of an object (the table object), on both support stations and mobile hosts, to associate service names with references to the appropriate servers.

Another object (the gateway object), located on both mobile hosts and support stations, is in charge of maintaining an indirection in the SSP Chains to/from a mobile host, and to cooperate during the reconnection of such SSP Chains. On the fixed part of the network, a reference to a gateway object can be used to locate the associated mobile host. The gateway object understands a kind of "where are you" message. Sending this message implicitly short-cuts the chain (if necessary) before returning the location of the mobile host. In effect the location "is" the reference to the gateway object — but we can imagine that an actual support station address would be useful in some circumstances.

This way of finding (and using) resources and mobile hosts in a mobile environment is somewhat centralised. However, this is not problematical since all wireless communications to mobile hosts must implicitly pass through a support station, and the support stations are a convenient and logical place to store the table objects (if needed, it is always possible to shift both table and gateway objects to another station, further in the fixed network). In addition, our mechanism is only "weakly" centralised: the point of centralisation migrates between support stations so as to remain close to the associated mobile host.

The management of location and resource access is simplified by using internal mechanisms of SSP Chains such as short-cutting and flexible binding, and the implementation of a first "mobile SSP Chain" prototype is currently well underway. However, any realistic system based on these mechanisms would have to address other important questions. One such question is that of fault tolerance: How can we recover when some space (which is in reality a unix process) involved in the support of the mobile machine crashes? Some work has been done on designing algorithms which reconstruct such lost state, but its suitability will remain unknown until an actual implementation is undertaken.

Acknowledgments

Thanks to Marc Shapiro for his help and numerous comments on this work, and to Guillaume Pierre for commenting on an earlier version of this article.

References

[Badrinath 94] B. R. Badrinath, Arup Acharya, Tomasz Imielinski: Structuring Distributed Algorithms for Mobile Hosts, *14th International Conference on Distributed Computing Systems*, Poznan, Poland, June 1994.

[Forman 94] Georges Forman, John Zahorjan: The Challenges of Mobile Computing, *IEEE Computer*, April 1994, p. 39-47.

[Piumarta 95] Ian Piumarta: SSP Chains - from Mobile Objects to Mobile Computing, *ECOOP'95 Workshop on Mobility and Replication*, August 1995.

[Shapiro 92] Marc Shapiro, Peter Dickman, David Plainfossé: *SSP Chains: Robust, Distributed References Supporting Acyclic Garbage Collection*, INRIA report 1799, November 1992.