Wireless Personal Communications **4:** 163–184, 1997.

Location-Independent Access in Mobile Systems*

ZYGMUNT J. HAAS
Cornell University, School of Electrical Engineering, Ithaca, N.Y. 14853, U.S.A. (e-mail: haas@ee.cornell.edu)

Abstract. In this paper, we report on the design of a protocol for transparent, location-independent access in mobile systems. The protocol, termed by us *Mobile Client/Server Protocol* (MCSP) is an implementation of a communication layer on a mobile host and a stationary machine within the wireline network, and provides middleware functionality. The MCSP supports communication services through the client/server paradigm to transaction-oriented applications distributed between the mobile and a set of service and information brokers. To support the mobile and the wireless environment, a surrogate process is created within the fixed network to perform operations on behalf of the mobile within the fixed network. The MCSP then loosely couples the communicating entities, thus reducing the overhead associated with handoffs and disconnects, so characteristic of the mobile environment. Of particular interest is the application of the MCSP to location-based services, for example, the Intelligent Vehicle Highway Systems [1].

Key words: Location Independent Access, Indirect Interaction, Surrogate Process, Mobile Communications, Mobile Computing, Mobility Management, Client–Server, User Profile, Location-Based Access.

1. Introduction and Motivation

The enormous explosion in the research, development, and application of wireless communication makes integration of the wireless networking feature into computing products quite promising. This has lead to the introduction of the relatively new field of *Mobile Computing.* Undoubtedly, interest in Mobile Computing was boosted by the technological progress in miniaturization of electronics, by the focus in the technical literature on wireless communication, and by the future vision of PCS[1] that advocates ubiquitous telecommunication access anywhere, at any time, with anyone, and in any format.

Mobile Computing introduces a drastically novel paradigm into the future computing environment. The following three features characterize the mobile computing environment [2, 3]:

1. portability and continuous access: a user is constantly plugged-in wherever s/he goes,
2. preservation of the logical environment or transparency to mobility: e.g., provision of network services invariant to users' locations, such as location-independent access
3. new class of services, especially tailored to the mobile and wireless environment (e.g., person- and service-locator services).

Thus, of primary concern is the support of network services to the mobile and wirelessly-interconnected machines, so that, while users are in motion, the user interfaces and the applications' performance remain essentially unchanged, as compared with the wired implementations. To achieve this, one needs to cope with two major factors that distinguish the

* Portions reprinted, with permission, from Proc. ICC'96, Dallas, TX, June 23–27, 1996 ©1996 IEEE and from Proc. Globecom'95, Singapore, Nov. 13–17, 1995 ©1995 IEEE.

[1] Personal Communications Services.

mobile from the fixed environment: the shortage of computing and communication resources and the effects of mobility. In this paper, we address the second issue only[2]; more specifically, we propose a protocol that preserves a user's access to network resources in the face of mobility. In other words, our protocol supports location-independent access.

Mobility affects the communication process through the potential need to constantly change the bindings of the communicating entities. For example, while changing his/her location, a user may want to access *local* servers to retrieve some information, while preserving the bindings with other remote servers. This results in dynamic (and possibly frequent) changes in the set of communicating servers. If the binding between the mobile computer and the servers is tight, rebinding to local servers requires closing and opening many session-level "connections." Thus complex protocol synchronization is needed that may carry too high performance costs. Also, it is predicted that disconnections will be a frequent event in the future wireless environment,[3] increasing the protocol synchronization needs of every level of the protocol stack. In these regards, in this work, we are interested in ways in which a mobile environment can best provide network services to a mobile client, alleviating some of this synchronization burden.

In order to efficiently support network services to mobile clients, we propose here the Mobile Client/Server Protocol (MCSP) – an implementation of a distributed-communication protocol layer between the mobile, its base-station, and a set of service/information brokers, based on the client/server paradigm. More specifically, the MCSP is based on a **surrogate process** that "lives" within the wireline network and that performs bindings on behalf of the mobile application between the mobile application and a server. Existence of this surrogate process allows faster, more optimal, and more efficient binding process and reduces the required communication load over the wireless link.

This paper is organized as follows: The next three subsections define the terminology used throughout this paper, describe the related previous work, and outline our network model. Next, the Mobile Client/Server Protocol is introduced. In Section 3, various scenarios of the MCSP operation are explained, including the behavior of the MCSP during special operation conditions, such as handoffs and transmission errors. Finally, some conclusive remarks are made in Section 4.

1.1. NOMENCLATURE

- MH – Mobile Host: A machine capable of mobility. Also called a *mobile*.
- CH – Corresponding Host: A machine communicating with a Mobile Host. Corresponding Host can be a stationary or a mobile machine.
- HN – Home Network: The administrative network of a mobile machine. The network portion of a machine's IP address equals the IP address of its Home Network.
- VN – Visiting Network: A network, other than the mobile's Home Network, to which the mobile is connected on a temporary basis.
- HA – Home Agent: A process residing on the Mobile Host's Home Network which is capable of capturing traffic addressed to the Mobile Host, while the Mobile Host is away from the Home Network.

[2] Readers interested in discussion on the first factor are referred to [4, 5].

[3] Especially as the number of users increases and the operation of wireless systems shifts into the higher-frequency spectrum.

- FA – Foreign Agent: A process resident on a visiting (wireline) network and which communicates with the Mobile Host and its Home Agent.
- COA – Care-Of-Address: An IP address on the Visiting Network through which packets are forwarded to the Mobile Host.
- POA – Point-Of-Attachement: An adaptor on the wireline network through, which a Mobile Host can communicate with the wireline network.
- Handoff – A procedure of replacing a binding between a MH and a network entity with a new binding between the MH and another network entity.
- Server – A machine performing some specialized operation (service) for other machines (clients).

1.2. Related Work

The basis for the MCSP is the use of the FA as an agent performing operation on behalf of the MH in the fixed network. The case for indirect operation for mobile clients has been described in [6]. In that work, the entity performing this indirect operation is the Mobile Support Station,[4] which is a gateway between the wireless and the fixed networks. In the MCSP, the indirection can be supported by any entity capable of hosting the FA function. Interestingly, [6] also pointed out that mobility needs to be supported on every layer of the OSI model. Although we agree with this claim, our MCSP, by creating an additional layer between the application and the transport layers, hides some of the mobility issues from the application layer. Thus, the MCSP reduces the need to deal with mobility in the application layer.

An example of the indirect interaction in the mobile environment is the Indirect TCP (I-TCP) described in [8, 9]. In those works, an intermediary, Mobility Support Router (MSR), establishes two TCP connections, one with the MH and one with the CH. It then performs the transport layer functionalities separately on each one of the two connections, forwarding the data between the MH and the CH. The MSR offloads most of the mobility management from the MH, similarly to what is done in our MCSP.

In [10], a novel communication model, the *Dissemination Model* has been proposed. In this model, there is no strict binding between the communicating entities. Rather, a sender of data maintains channels over which information is disseminated to receivers. A receiver of data, in general, does not establish any type of connection with the sender.[5] Our work, although does not directly use the Dissemination Model, does rely on the "loose binding" approach between the source of information (i.e., the server) and its destination, the MH. Furthermore, our MCSP could be relatively easily extended to support the Dissemination Model.

In the MCSP, the "loose binding" is resolved by the FA, which, acting on behalf of the MH, determines which is the most appropriate server to execute the MH's request. Resolving a binding by an agent was implemented before in several systems; for example, in the Cygnus work described in [11].

[4] This is also consistent with the Columbia Mobile IP protocol [7].

[5] Although, for some operations, such as authorization, billing, and security, some binding between the source of the data and its receivers may be required.

1.3. The Network Model

Our fixed network model is the Internet; a collection of networks interconnected by gateways and communicating through the TCP/IP protocol suite. Services to clients are provided through a set of servers (sometimes also referred to as *brokers*). Each server is dedicated to a specific service. Examples of servers include: file server (for public storage of files), authorization server (to verify users' access permissions), various data-base servers (such as telephone numbers data-base, e-mail addresses data-base, IP addresses of other servers, data-base of mobiles' locations), printing servers, weather servers, stock-market quote servers, road congestion condition servers, map servers, and game server. Note that login to a machine is also covered under this model; every machine is just another (private) server on the network.

Each client in our model can be either a stationary host or a mobile machine. Of course, the case of mobile machines is the more interesting one. Examples of mobile machines include laptops, notebooks, tablets, or PDA-s. A mobile may be characterized by its associated *profile*, stored somewhere in the network. The profile describes the preferences of the mobile in communication and information processing. For example, a mobile may prefer to request WWW services through a specific WWW server, or the mobile may prefer requested images to be encoded with a specific encoding scheme. We envision that it is the responsibility of the HA to maintain its mobiles' profiles, although, one may argue that a profile should be stored on the mobile itself.

In the physical layer, mobile machines connect to the fixed network through Point-Of-Attachements (POA), which are generally referred to as *base stations*. Base-stations maintain wireless links[6] to mobile clients. It is assumed that mobiles communicate with the base-stations through a multiple-access scheme (e.g., [12] or [13]) and that the bandwidth of the wireline network links is considerably larger than that of the wireless links. The wireless links are point-to-point links. We exclude here the possibility of mobile peer-to-peer interconnection.

Each base-station has some area, termed *coverage area*; mobiles currently located within the coverage area of a base-station maintain connectivity with this base-station. Thus the wireless network forms a cellular infrastructure. In a local-area environment, for example, the cells (sometimes called pico-cells in a local environment) are on the order of offices, aisles, and labs. Each of these pico-cells is served by a transmitter/receiver pair, referred to as *wireless transceiver*. Transceivers are controlled by base-stations.[7] The wireless network can be LAN, MAN, or WAN; examples of wireless LANs include *Rangelan*(TM) or *Wavelan*(TM), while the wide-area wireless interconnection can be provided through the CDPD protocol or through the existing wireless data networks, such as *Ram Mobile Data* or *Ardis*.

Mobiles can "roam" throughout the coverage of the wireless network. While changing its physical location, the mobile's POA also changes to support adequate wireless (e.g., radio) connectivity (e.g., sufficient RF signal strength). Such changes of the POA are referred to as physical-level handoffs or handovers.

While a mobile changes its location in a wireless network, it may move into a "dead area," where there is no wireless coverage at all. Thus, a fundamental feature of our network model is that a MH may temporarily loose its connectivity to the wired network. The duration of

[6] Mobile machines that frequently change location by plugging into different networks are also considered to create mobile networks. In this work, we mainly address the wireless, rather than the 'plug-in' networks. However, our work is also applicable to the 'plug-in' environment.

[7] In general, a base-station may serve several transceivers. Furthermore, the transceivers need not to be physically attached or even in the proximity of their base-station.

this temporary disconnection is a function of a lot of parameters, some of which may be unpredictable. It can last from several seconds in the wireless medium to hours or even days, as in plug-in connectivity. Protocols designed for the mobile environment must take into the account that the connectivity to the MH may be lost for any length of time and that when the connectivity is restored, the MH may not be associated with the same base-station as it was before the disconnection. Yet, this change of association should be *transparent* to the mobile user.

Since our work is targeted at issues above the network layer, we are not concerned here with, what are generally referred to as "lower-layers" issues. For instance, since the physical-level handoff operation is performed at the MAC layer, the manner in which the physical handoff is achieved is of no concern to us. Of course, the effect of the handoff, resulting in changing of the mobile's POA to the wireline network, is of primary importance to us.

At the network layer, as the mobiles migrate between POA, the IP layer must be modified to allow routing to their new locations. There have been a number of proposals for mobile IP. We assume here that a scheme closely related to the IETF Working Group proposal [14] is implemented. Some minor modifications[8] to our protocol may be required if another mobile IP protocol is used. In what follows, we briefly describe the main features of the mobile IP used in our work. For additional explanation, the reader is referred to [14].

Four entities participate in the mobile IP: the Mobile Host (MH), the Corresponding Host (CH), the Home Agent (HA), and the Foreign Agent (FA). The MH maintains his HN IP address; i.e., there is no assignment of temporary IP addresses. Upon migration from the MH's Home Network, the MH searches for a local FA on the Visiting Network (VN). This process is referred to as *Foreign Agent Discovery*. Upon identifying a FA, the MH performs a *Registration* procedure, at the end of which the MH identity is added to the visitor's list of the FA and the identity of the FA (actually the Care-Of-Address) is registered with the HA.

A MH sends out its IP packets using the traditional IP procedures. Other CH send packets to a MH by addressing the packets with the regular IP address of the MH. When the packets are delivered to the HN, the HA is responsible for capturing the packets in place of the absent MH. The HA tunnels the captured packets (through IP-in-IP, for example) to the FA (using the COA). The FA locally delivers the packets to the MH. As the MH roams from one network to another, the Foreign Agent Discovery and the Registration procedures are repeated.

The challenge that the MCSP addresses is the provision of network services in a mobile environment, characterized by frequent changes of clients' Points-Of-Attachement and by unpredictable disconnections. In order to concentrate on the main features of the MCSP, we had to exclude or inadequately address some quite interesting and important issues. One of them is advanced security issues (i.e., privacy, information integrity, etc.) of the proposed protocol. Also, network management functions that are not directly related to the client-server connectivity are beyond of scope of this paper.

2. The *Mobile Client/Server Protocol* (MCSP)

MCSP is a protocol layer residing on MH-s and on FA-s and is positioned between the Application and the Transport layers (see Figure 1). MCSP addresses two main problems, characteristic of a mobile environment: the frequent change of bindings between mobiles and

[8] Since our work on MCSP is above the transport layer, the implementation details of the network layer are mostly immaterial.

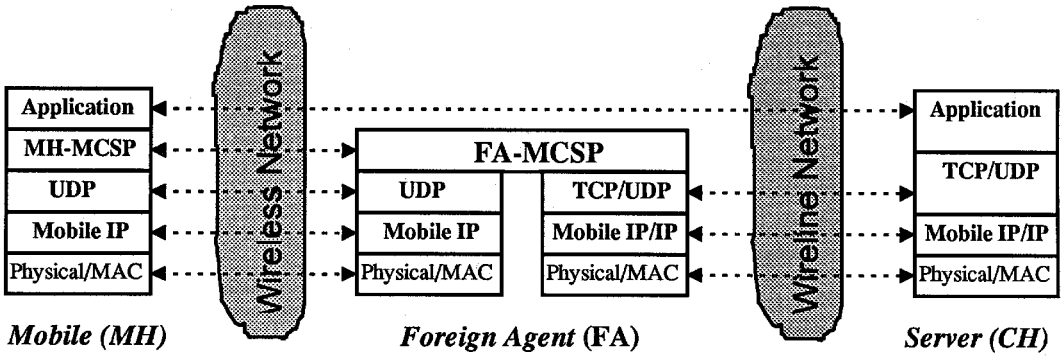

Figure 1. The position of the MCSP.

servers in response to mobile users' mobility, and the abnormal conditions caused by mobility (such as disconnections), which need to be processed differently in a mobile network.

The purpose of the MCSP is to hide the effect of the above two problems. This is done by having the FA act as an intermediary (surrogate or agent) on behalf of the mobiles in its coverage. In the intermediary function, the FA has access to both the fixed network and the mobiles, allowing hiding of some of the mobility effects from the mobile applications and supporting transparent and location-independent access to network servers.

From the programming point of view, the MCSP is a set of Application Programming Interfaces (API-s), which allows abstraction of information-access functions, effectively hiding the mobility effects from the programmer. The MCSP provides a shell under which a user may program or use his applications without having to worry about the mobility effects. Thus, MCSP provides *Middleware* functionality.

In MCSP, a binding between a mobile and a server may be either *loose binding* – to provide location-based services, or *strict binding* – to provide location-independent access. Loose binding is used, for example, in transactions initiated specifically with a local server (i.e., in the vicinity of the mobile's current location). An example of strict binding is a telnet session to a specific host. The choice of strict vs. loose binding depends on the specific operation that needs to be performed.

We claim that, because of the possibly frequent changes in binding and the different way the abnormal conditions are handled, connectionless protocols [15], rather than connection-oriented protocols, are better suited for mobile communication. In particular, the transactional communication model [16] is best suited for such an environment. The transactional communication model advocates use of packetized data, based on the request-response paradigm. (Although we concentrate on the request-response communication model, the design of the MCSP, as presented here, also supports server callbacks.)

To elaborate, because of the possibly frequent changes in the bindings between a mobile and the set of available servers, setup procedures between communicating entities may be too time consuming. In fact, in the local network characterized by small, micro-environments, the mean time of a mobile within an environment may be on the same order of magnitude as the time length of the setup and the close procedures. This results in limiting the throughput to some fraction of the available bandwidth, because long periods of time are wasted on the setup procedures. Furthermore, the signaling overhead on the wireline network due to the

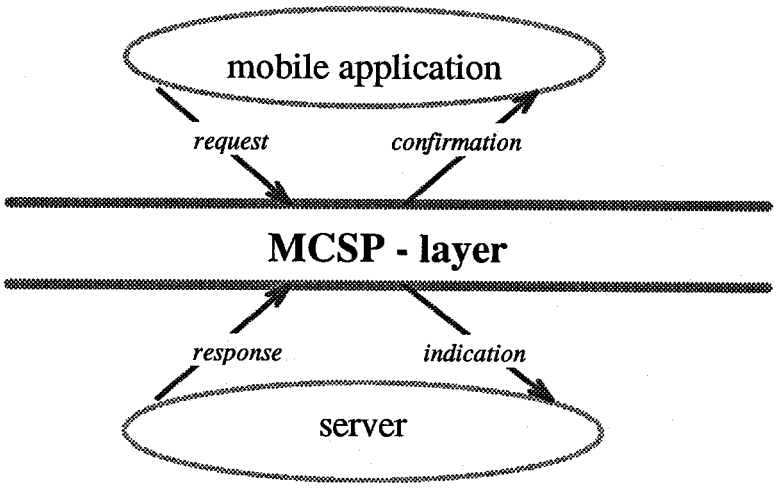

Figure 2. Messages in MCSP.

connection-oriented protocols can be prohibitively large. Consequently, we design the MCSP as a connectionless protocol.

There are four major entities that take part in the execution of our protocol:

- a mobile application, executed on the MH,
- the server(s) – the entity(ies) responsible for execution of requests,
- the yellow-page server (YP-server) – a data-base containing the IP addresses of all the servers accessible within a specific domain
- the MCSP, which consists of two parts: the mobile-executed part (MH-MCSP) and the Foreign-Agent-executed counterpart (FA-MCSP). The FA-MCSP provides the service-to-server[9] mapping and request-clearing functionality.

One basic feature of our protocol is that it hides the server identity from the user[10]; in other words, the user does not know where the server that processes its request is. It is the responsibility of the FA to map and to forward the requests to the appropriate network addresses. The choice of a server is based on the availability, the profile of the originating mobile, the traffic and load conditions, and the cost (e.g., billing) considerations. Furthermore, mobiles never communicate directly with servers, only with their current FA.

The list of potential servers is maintained by the YP-server. Servers periodically advertise their existence and their availability in their local environment to the YP-server(s) in that environment.

There are four kinds of messages[11] in the MCSP (see Figure 2):

- messages from the mobile application to the MCSP layer (MH-MCSP), called *requests*,
- messages from the MCSP layer (FA-MCSP) to servers, called *indications*,
- messages from the servers to the MCSP layer (MH-MCSP), called *responses*, and
- messages from the MCSP (MH-MCSP) layer to the mobile application, called *confirmations*.

[9] Logical service and physical server.

[10] Throughout this work we use interchangeably the terms user and application.

[11] We follow here the ITU terminology.

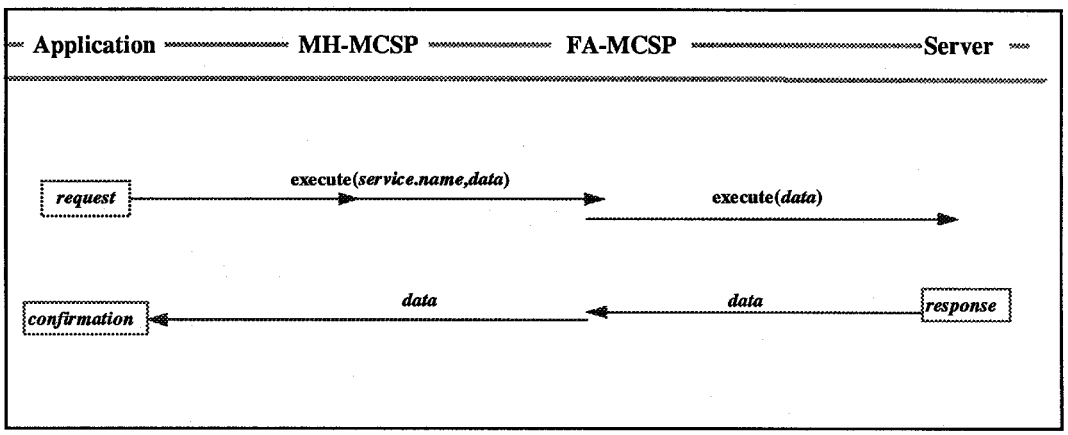

Figure 3. Simplified exchange of messages in MCSP.

The basic operation of the MCSP is as follows: the MCSP receives a request from the mobile application and processes (inserts or replaces) some of the control information to create an indication, which is sent to the appropriate server(s). The processing of the control information is done to support location-independent operations based on the mobile's profile. A copy of the mobile's profile temporarily resides in the FA-MCSP. Servers respond to inquiries by sending responses to the MCSP (i.e., to the MH-MCSP through the FA currently serving the mobile). The MH-MCSP passes on the responses, which are now termed confirmations, to the mobile application, possibly after removing some control information. Note that responses are sent from the server to the mobile's IP address (and not directly to the FA); i.e., responses are delivered to the MH-MCSP through the mobile IP protocol without the FA-MCSP intervention (but with the assistance of the FA, according to the mobile IP protocol). Thus the FA is used for communication from the MH to the server(s) only (i.e., request/indication). This is a crucial point in the design of the MCSP, since the server does not know the IP address of the FA currently serving the MH. Figure 3 depicts the basic flow of messages between the MH, FA, and a server.

Both the application and the MH-MCSP can initiate requests. A request initiated by the application is, in general, a solicitation to perform some server operation, with a confirmation returned to the application. Requests initiated by the MH-MCSP are mostly supervisory in nature and result in responses returned to the MH-MCSP. As there is no elementary difference between the requests originated at the applications and requests originated at the MH-MCSP, we do not differentiate between these two types of requests in our discussion.[12]

Messages between the MH-MCSP and FA-MCSP (i.e., between the two parts of the MCSP) are connectionless,[13] while the messages between a FA and servers may be either connectionless or connection-oriented, depending on the servers' implementation. Confirmations are always connectionless.

After receiving a new request from the MH-MCSP and after determining the identity of the server that is scheduled to process the request, the FA-MCSP returns to the MH-MCSP the IP address of the server that will serve the request. The IP address of the server is referred to here as the request *handle*. The basic exchange of messages for execution of a *service.name*

[12] In figures, only the application-originated requests are marked as such.

[13] Using UDP.

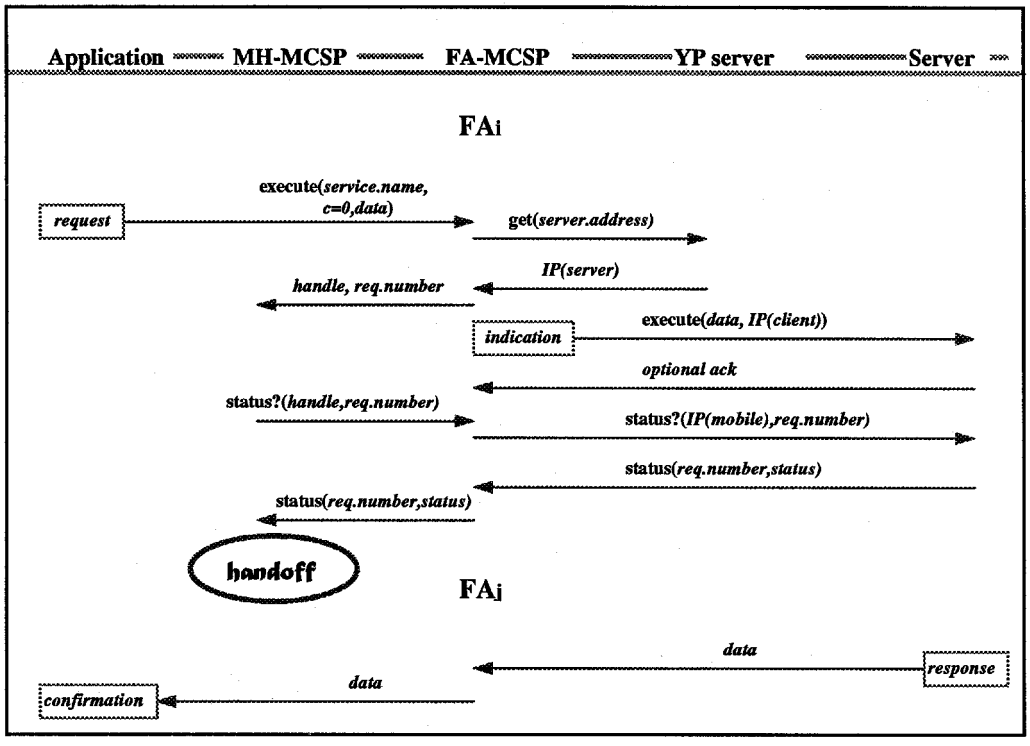

Figure 4. Basic exchange of messages in the MCSP protocol.

request (**execute(***service.name,c=0,data***)**) is depicted in Figure 4. The YP-server is queried for the IP addresses of potential servers that might execute the request (**get(***server.address***)**). After reception of the IP addresses and after choosing one of the servers, a handle is returned to the MH-MCSP. The handle also acknowledges that the request was received and will be processed. Then the request is sent to the server as indication. Later, when the MH-MCSP requests the status of the request (**status?(***handle,req.number***)**), it uses the handle to access the server. The server responds to the status query (**status(***req.number,status***)**) using the MH-s IP address. When the server is ready with the response to the initial request, it sends it directly to the mobile's IP address. The response (*data*) will be delivered to the application by the MH-MCSP through the currently serving FA, transparently to any handoff procedure that might have taken place in the meantime.

Of particular interest is the processing of the control information by the FA. This processing accommodates several functions:

- server selection, based on the mobile's profile and the current status of the servers
- handle abnormal server conditions; e.g., service not available
- authenticate the mobile to the server; e.g., for access permissions or billing purposes
- perform segmentation/reassembly of messages
- perform complex operations; e.g., forwarding, chaining, referal, multicast, search
- facilitate handoffs and disconnections

We now proceed with the description of the request and the confirmation formats, starting with the request shown in Figure 5. The request consists of the following fields:

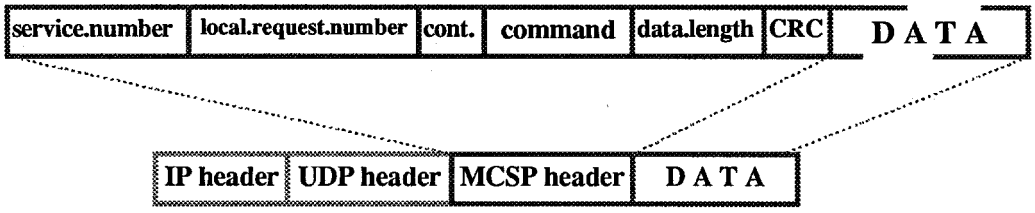

Figure 5. The request format.

- *service.number* (4 Bytes): i.e., service number
- *local.request.number* (3 Bytes): the number of this request based on a local counter
- *continuation* (2 Bytes): an indication whether this is a continuation of a previous request segment
- *command* (2 Byte): the command to be executed
- *data.length* (3 Bytes): the length of the data field (in Bytes), if any
- *CRC* (2 Bytes): Cyclic Redundancy Code for error detection/correction
- *data* (0–16,777,216 Bytes): the data associated with the command

The *service.number* field is a 4-byte number representing the service/server name. It is either a service name, in the case that the FA is to determine the physical server to execute the request (i.e., loose binding), or a server name (the IP address), when binding to a specific server is required (i.e., strict binding). In the former case, the FA-MCSP maps the *service.number* to an IP address of a physical server. The most significant bit of the *command* field indicates whether the *service.number* is service name or server name. The *local.request.number* is the value of the request counter at the mobile and is 3-bytes long. A mobile can, thus, have as many as $2^{24} = 16,777,216$ outstanding requests. *local.request.number* has only local significance. To globally identify a request, *request.number* is used, which is composed of the concatenation of two fields: *source.address* and *local.request.number*, where the *source.address* is the mobile's IP address. The *command* field indicates the requested server operation. The *data.length* indicates the number of bytes in the data field; the minimum and the maximum total lengths of the request are 16 Bytes and 16,777,232 Bytes, respectively. The *CRC* is computed based on all the fields of the request message, except that *data* field.

We envision that the vast majority of the requests can be accommodated by the above format. However, there may be requests which require to convey more than 16 MBytes of data. Our solution is to repeat the above format a number of times; each repetition is called a *segment*. All the segments of the same request bear the same *local.request.number*. The *continuation* field, indicating that a request spans several segments, is actually a sequence number: if it equals zero, this is the first and the only segment of the request; if it equals one, it is the first segment of the request; if it equals two, it is the second segment of the request, etc. The continuation field of the last segment is a negative number. For instance, if a request is composed of seven segments, the continuation fields of the seven segments are: 1, 2, 3, 4, 5, 6, -7. A request can consists of at most 32,767 segments and can convey at most $2^{39} \approx 0.55$ TBytes of data. Note that even though the segments of the same request are logically related, they do not need to be transmitted or received back-to-back, nor even in sequence, as long as the first segment is received first by the FA-MCSP. The segments are delivered to the server, whose responsibility is to ensure end-to-end reliability. In other words, it is the servers' responsibility to request retransmission of missing (e.g., erroneous)

segments. This is achieved in the MCSP by the server sending a response, indicating an error in processing the request. The error number (the *status* field in the confirmation format in Figure 7) indicates that some of the segments were missing and the numbers of the missing segments are included in the *data* field.

In a multi-segmented request, the handle is sent in response to the first segment, identified by the *continuation* field being equal to 0 or 1. If a first segment of a request that hits the FA-MCSP has *continuation* field other than 0 or 1 and the *service.number* needs address translation, the segment is ignored. If the first segment with the *continuation* field other than 0 or 1 has actual server IP address, the segment is ordinarily forwarded to this server.[14] Segments transmitted by the MH-MCSP after receipt of the handle (i.e., segments with *continuation* $>$ 1) contain the actual server IP address in the *service.number* field. This allows continuation of the multiple-segment message in the case of a handoff occurring in the middle of the segments transmission. An example of the exchange of messages in the case of a 4-segment long request is shown in Figure 6.

The structure of a confirmation message (i.e., from FA-MCSP to the mobile application) is shown in Figure 7 and consists of the following fields:

- *source.address* (4 Bytes): i.e., server's IP address
- *request.number* (3 Bytes): the number of the original request that generated this response
- *status* (2 Bytes): indicates an error in processing the request
- *continuation* (2 Bytes): indication whether this is a continuation of a previous response segment
- *data.length* (3 Bytes): the length of the data field (in Bytes), if any
- *CRC* (2 Bytes): Cyclic Redundancy Code for error detection/correction (optional)
- *data* (0–16,777,216 Bytes): the data associated with the response

Note that the MH-MCSP returns to the application in the confirmation message the server's IP address. This is to facilitate further communication between the application and the **actual** location of the request execution. Note that the original request handle may not point to the actual execution location of the request, since the request may have been forwarded or chained from the original server to another server. However, the *source.address* always points to the *last* entity that executed the request. For example, if the application finds the confirmation data in error, the application can request retransmission of the corresponding response from the last server.[15]

The *status* field allows to communicate to the MH-MCSP any abnormal conditions encountered during processing of the request. For instance, if not all the segments of a multi-segmented request were received at the server, this will be communicated to the MH-MCSP through this field. The MH-MCSP can then retransmit only the missing segments. If no error occurred during the request processing, the *status* field is set to 0. The significance of other fields in the confirmation format are the same as the corresponding fields in the request format.

The communication between the FA-MCSP and the servers is carried over a fixed network and may be connectionless or connection-oriented, depending on the implementation of a particular server. In our implementation, we used only UDP protocol between the FA and the servers. Furthermore, we allowed some degree of servers' mobility,[16] including frequent

[14] This is the case when a MH underwent handoff during the transmission of a multi-segmented request.

[15] This request needs to be addressed to the last server, since the response can be cached in the last server only.

[16] I.e., the servers could also reside on a mobile machine, not just the clients.

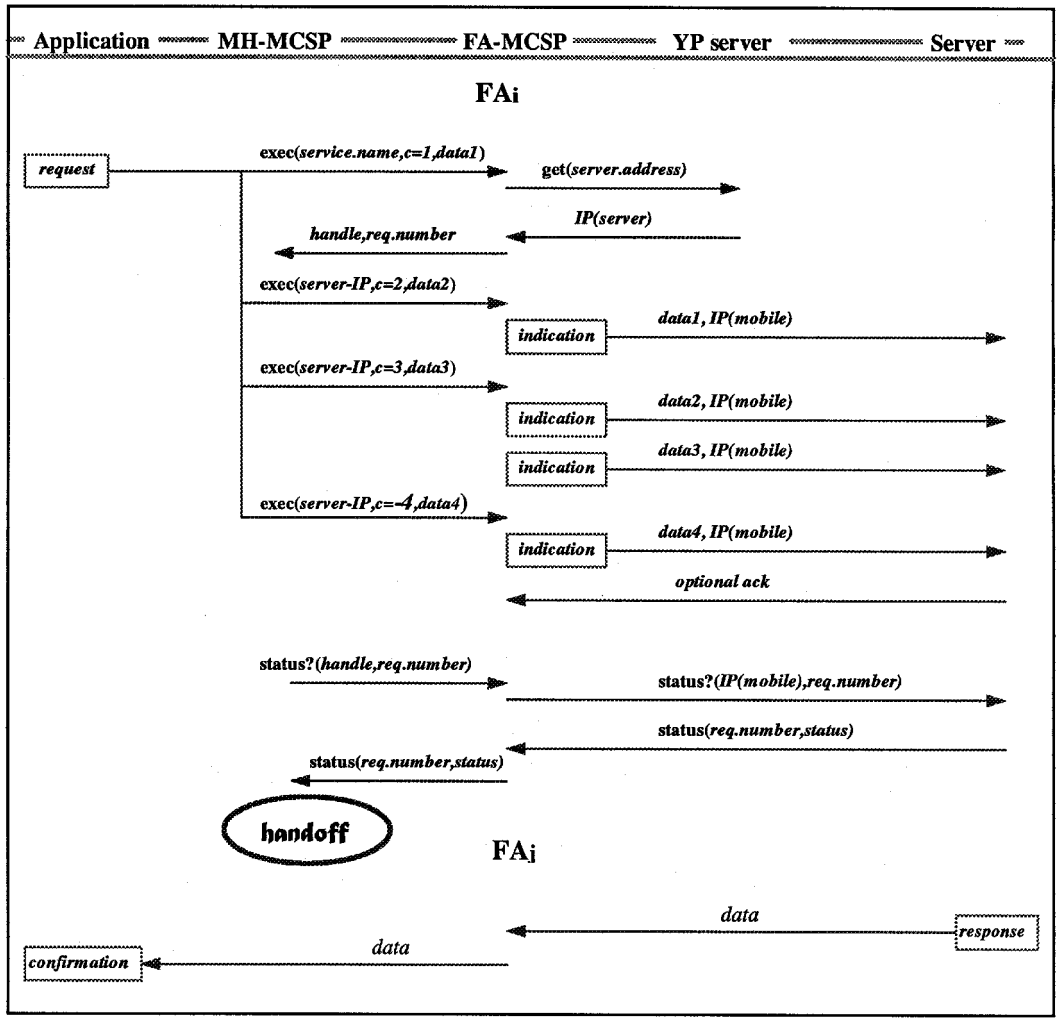

Figure 6. Exchange of messages in the case of a segmented request.

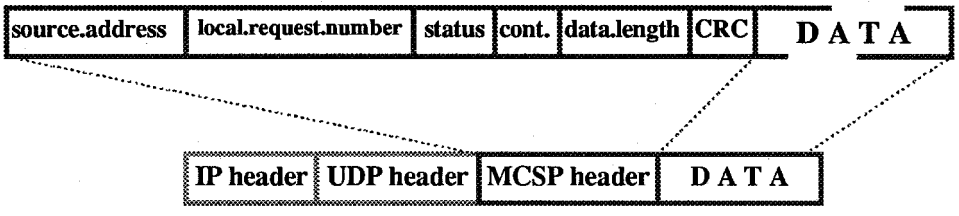

Figure 7. The confirmation format.

disconnections. Thus, similar argumentation to the MH-MCSP/FA-MCSP communication being based on connectionless approach can also apply to the communication between the FA and the servers.

In the next section, we discuss some specific scenarios of the MCSP operation to further demonstrate the design and the capabilities of the protocol.

3. Scenarios of Operation

3.1. INITIALIZATION

When the communication stack on a MH is initialized, when a MH reconnects after being disconnected, or when the MH-MCSP on the mobile detects that the mobile has been handed-off to a new network (see Section 3.2), a registration procedure of the MH with the new FA is executed. Such a registration procedure, which is part of the mobile IP protocol, establishes bindings between the MH and the FA and between the MH and the HA for the purpose of routing the MH-s packets through the COA.

In the MCSP, it is assumed that the mobile IP registration is performed transparently to the MCSP layer. However, since additional exchange of information between the FA and the mobile is required, such exchange (termed here initialization) is performed on the MCSP layer. The initialization procedure includes identification, authentication procedure of the MH-MCSP to the new FA-MCSP, and supply of the MH-s profile to the new FA-MCSP.

The authentication is Kerberos-like [17] and consists of MH-MCSP providing the FA-MCSP with a Kerberos certificate. To keep this discussion short, we go only through the major steps of Kerberos authentication, to show how these apply to the specific mobile situation. To obtain the certificate, the FA-MCSP sends a certificate request to the Global Certifier (GC), with the MH identification (e.g., its IP address).[17] The certificate is generated by the GC and contains the ticket and the authenticator portions. The ticket is encrypted with the MH private key, while the authenticator is encrypted with the FA private key. Both the ticket and the authenticator contain the session key drawn by the GC. The certificate is returned to the MH-MCSP, which decrypts the ticket using the private MH's key, extracting the session key. The MH-MCSP composes a new ticket that contains the time stamp, the checksum, and the subsession key and encrypts the new ticket with the previously obtained session key. This new ticket, together with the original authenticator are sent to the FA-MCSP. The FA-MCSP decrypts the authenticator with the FA-s private key, obtains the session key, decrypts the ticket, and calculates the checksum. If the locally calculated checksum matches the checksum in the ticket, the mobile is authenticated. In this case, the FA-MCSP sends the mobile acknowledgement (*authenticated.*). The subsession key is used for any future exchange of information that need to be kept private.[18] In the case that the certificate checksum does not match the locally calculated checksum, the MH-MCSP is notified (*authentication.failed*) and no further action is taken by the FA-MCSP. It is the MH-MCSP responsibility to reinitialize. The authentication procedure is depicted in Figure 8.

After successful authentication, the MH-MCSP transmits the mobile's profile (more typically the location of the profile) to the FA-MCSP of the new FA. The profile is cached on the FA-MCSP as part of the mobile's record.

Similarly to the mobile IP protocol, there is no reverse procedure to initialization in the MCSP; i.e., the cache entry automatically expires the negotiated time-out after the initialization.[19] Expired entries are purged when space is needed for a new initialization. A mobile needs to reinitialize itself with the FA before its initialization expires.

The initialization and the reinitialization are shown in Figure 8.

---

[17] It is assumed that there exists a GC entity that issues keys to the mobiles, as well as issues certificates for authentication purposes. The description of the operation of this global entity is beyond the scope of this paper.

[18] For simplicity and clarity, we omit the encryption from the time diagrams in the rest of the paper.

[19] Alternatively, a specific initialization time-out period (*initialization.lifetime*) can be defined.

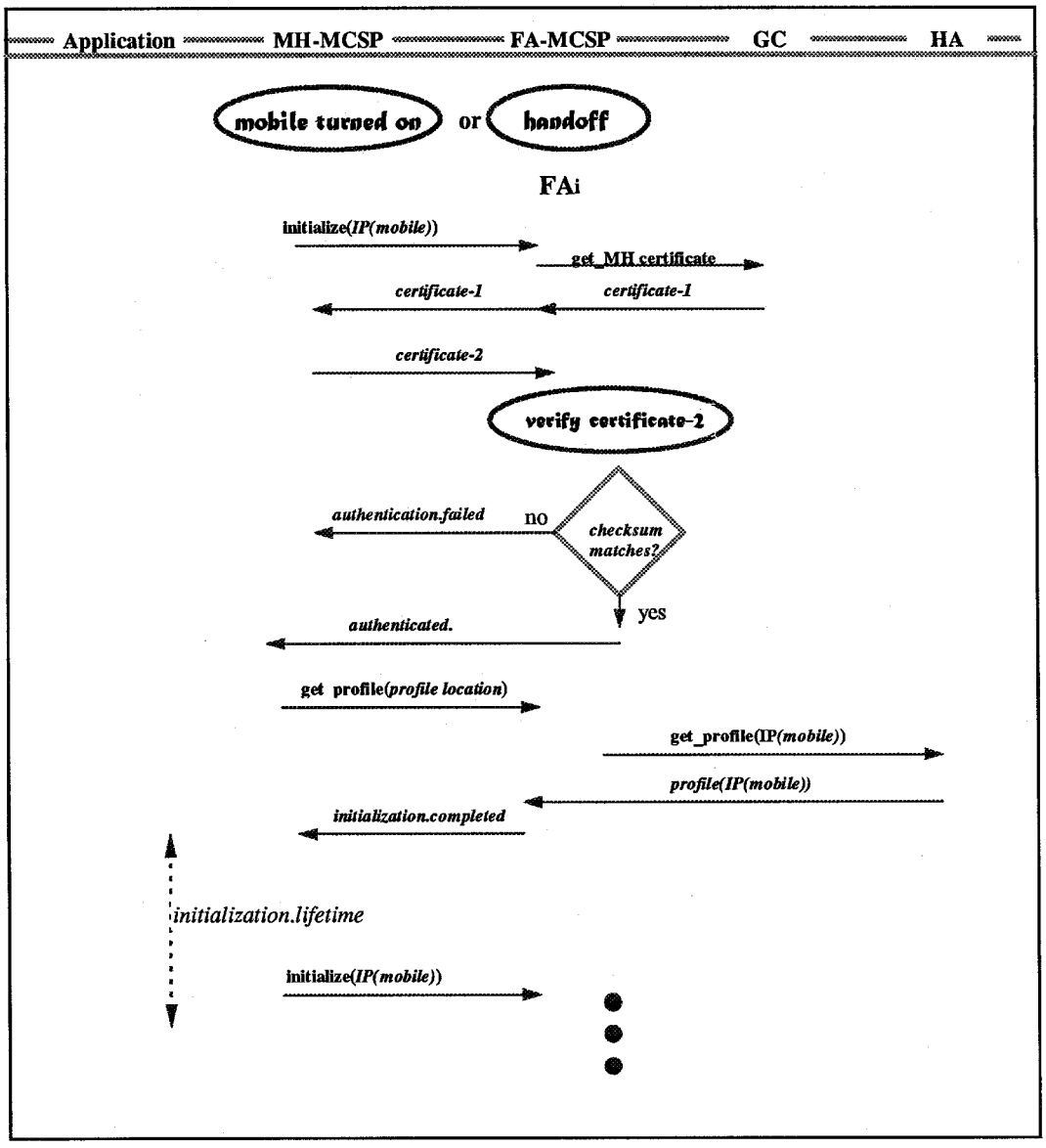

Figure 8. Mobile initialization and reinitialization with a new FA.

3.2. HANDOFFS

Handoff can occur at any time, unsynchronized with the MH's transmission or reception. There are several ways in which indication can be given to the MH-MCSP about the MH being handed-off to a new FA: the lower layers (i.e., mobile IP) may provide this indication to the MH-MCSP, the MH-MCSP on the mobile may periodically poll the FA-MCSP for its identity (through the "who are you" request), or the FA-MCSP may periodically broadcast packets indicating the identity of the served MH-s. It is immaterial to the operation of the

MCSP which is the actually implemented scheme.[20] If neither such mechanisms is possible, our protocol will still perform correctly, but messages lost during the handoff process may need to be retransmitted through an end-to-end retransmission procedure. We assume here that the MH-MCSP receives indications of a handoff in some way when the handoff procedure starts and when it is completed and we show here how a handoff operation affects the state of the MCSP. We consider the cases when handoff is performed:

1. in the idle state; after the initialization process is completed and when there are no outstanding or being-transmitted requests,
2. in the middle of the initialization process,
3. while there are outstanding requests,
4. in the middle of a request transmission.

Handoff, while the mobile is in the idle state, requires a new initialization process with the new FA-MCSP. There is no other implication on the MCSP. Handoff during the initialization process may result in incompletion of the initialization process when the communication between the FA-MCSP and the MH-MCSP is terminated before the *initialization.completed* message. Otherwise, the initialization process is completed. In either case, since the mobile will start a new initialization process, there is no further implication on the MCSP.[21] Handoffs during the initialization process are depicted in Figure 9.

If there are outstanding requests, the responses will be directed to the mobile by the virtue of the mobile IP routing scheme. The mobile can still reference the requests through the use of the handles to the requests. This situation is shown in Figures 6 and 11.

If a handoff occurs during the transmission of a request, the outcome depends on whether the mobile received the handle to the request or not prior to the handoff. If the mobile received the handle, then it can be sure that the request was received by the FA. If the mobile did not receive the handle (even through the request may have been sent to a server), the mobile will retransmit the request through the new FA after the handoff is completed. In the case that the request was already sent by the previous FA to a server, the response from that server will be discarded, as the responding server IP address will be different from the request handle. Only the response to the request that originated through the new FA will be accepted. The cases of handoff with outstanding requests are shown in Figure 10.

3.3. Long Requests

When the requests are longer than approximately 16 MBytes or when, because of other considerations, there is a limitation on the packet size, each request needs to be divided into several segments. Each segment is sent independently from the mobile to the FA. The question arises what happens when a handoff occurs between transmission of the segments; in other words, if the request is quite long, the system should not discard the already received segments. (Note that the previous FA may have already forwarded the first segments to the server.) The rule here is the same as in the single-segment message: if the handle is received by the MH-MCSP, the long request can be completed through the new FA (of course with the *service.number* equal to the server IP address from the original handle). The remaining segments are coerced to be delivered through the new FA to the original server, since the MH-MCSP uses the server's IP address from the original handle. If the handle was never

[20] In our prototype, we used the periodic polling scheme.

[21] Recall that an initialization will expire by itself after *initialization-lifetime*.

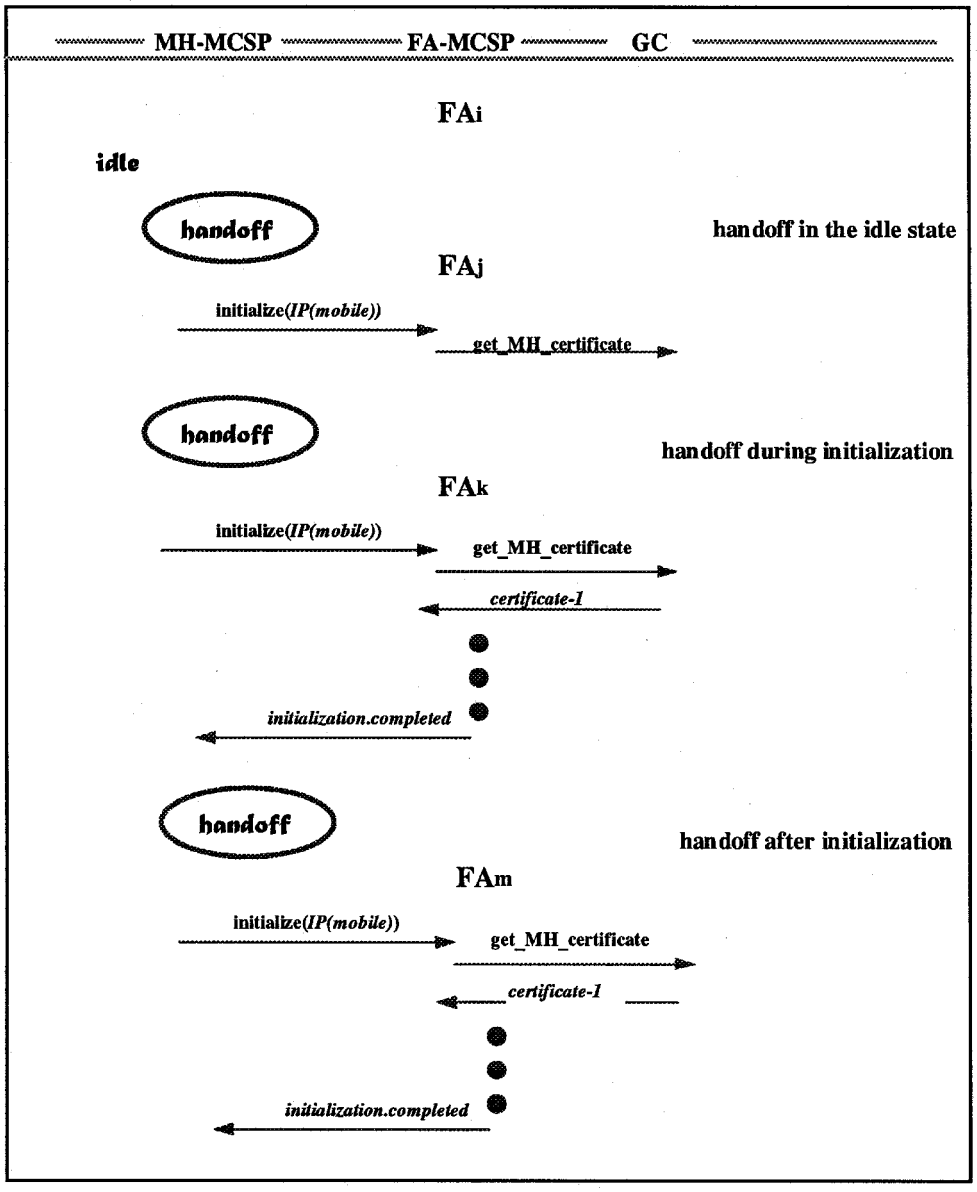

Figure 9. Handoffs during initialization.

received before the handoff is completed, the MH-MCSP needs to retransmit the request from the beginning. Partial segments received by the server, as in the case of handoff prior to the handle being delivered to the MH-MCSP, will be discarded at the server after *request.time-out*. The case of handoffs during a multi-segmented request is demonstrated in Figure 11.

Note that if a handoff occurs in the middle of a segment transmission, it can potentially destroy the transmission. Such a segment will be considered lost and will be retransmitted. If this segment was, in fact, not lost, the duplicate will be discarded by the server.

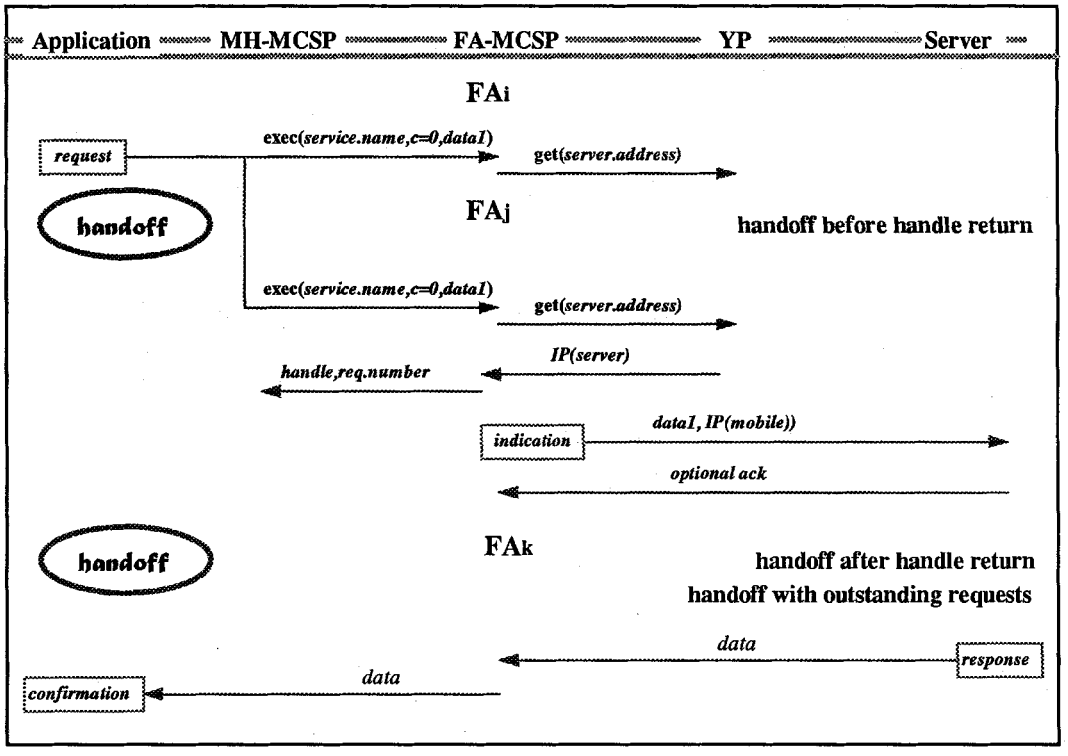

Figure 10. Handoffs with outstanding requests.

3.4. LONG RESPONSES: MESSAGE SEGMENTATION

Similarly to the long-requests case, if the response consists of a large amount of data, it may be necessary to break the response into multiple segments. The sequencing of segments in this case is performed as in the long-requests case; i.e., a single-segment response carries sequence number of 0 and the sequence numbers of a k-segment response are $1, 2, \ldots, -k$.

When handoff occurs during reception of a multi-segmented response, there is the possibility that some segments will be lost. This can happen, since the mobile IP does not support deregistration at the old FA. Thus, as the mobile leaves the coverage of the current FA but has not yet completed the registration with the new FA-MCSP, the HA will continue to tunnel MH's packets to the old FA, which will sent the packets on the VN for the MH's reception. However, because the MH has already left the coverage of the old FA, the packets will be simply lost. The sequence of events is demonstrated in Figure 12.

This phenomenon of packet loss during FA rebinding is basic to every mobile IP protocol. In fact this phenomenon is responsible for the anomalous behavior of TCP on top of mobile IP [18, 19]. Furthermore, as IP does not guarantee delivery, this phenomenon does not violate any protocol definition. Consequently, it is up to the transport/application layers to ensure reliability of packet delivery. In the MCSP the reliability is supported by the MCSP sequencing mechanism coupled with retransmission protocols (Section 3.5). In some cases, retransmission of lost segments may be facilitated by the inclusion of the replying server IP address in the

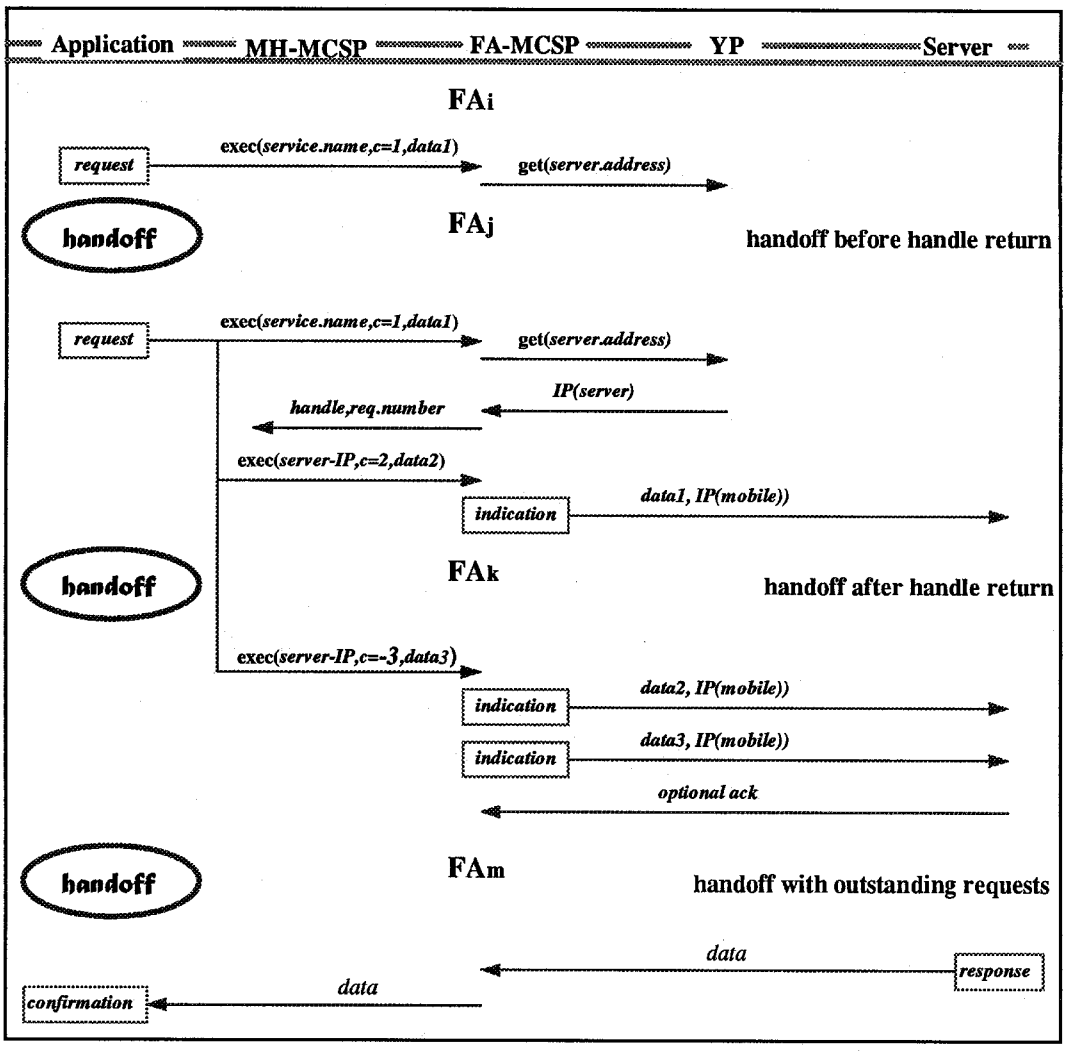

Figure 11. Handoffs between segmented request.

confirmation messages. Figure 12 depicts lost confirmation and its retransmission by issuing the *ror*[22] request.

3.5. TRANSMISSION ERRORS

Both ends of the MCSP periodically handshake through the use of the "are-you-alive" command (*aya*) to determine the status of the connection between them. The periodicity depends on the span of the network and the previous history of the link. The actual handshaking policy is not discussed here.

Although the handshaking may reduce the chances of transmission during disconnection period, it may not totally eliminate the probability of packet loss due to disconnection. Thus

[22] *ror* = retransmit old response.

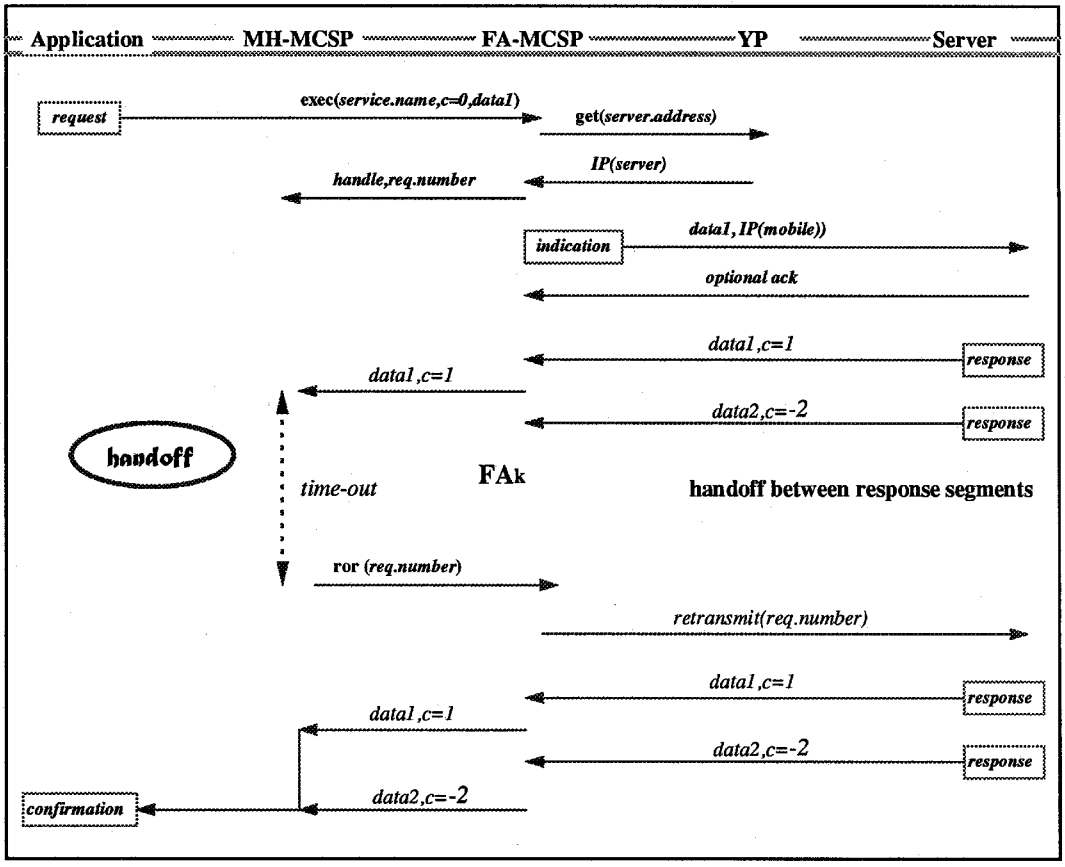

Figure 12. Handoffs between a segmented response.

the retransmission procedure is used to cope with disconnections, in addition to the traditional case of transmission errors.

Transmission errors, detected through erroneous *CRC*, are in most cases corrected through retransmission. To facilitate retransmission, a time-out period, *retx.time-out*, is defined. The *retx.time-out* is adjustable based on the anticipated time to performed the required operation. Thus, for example, the *retx.time-out* for the authentication operation is considerably longer than for the status request command.

First, we address retransmission due to errors in confirmations of outstanding requests. The MH-MCSP maintains a list of all outstanding requests with their corresponding *retx.time-outs*. Upon expiration of the particular *retx.time-out*, the MH-MCSP takes an appropriate action, depending on the status of the connection, the history of the connection, and the anticipated reception. Thus, if a connection is down (i.e., no response to the *aya* commands), all retransmissions are delayed until the connection is up again. If the connection was down at least once during the lifetime of the request, the MH-MCSP assumes that the confirmation is lost and will immediately ask for retransmission of the response (using the *ror* command with the server's address from the handle), rather than ask for the status of the request. Finally, if the confirmation is multi-segmented, and the MH-MCSP received at least one of the segments,

the retransmission-request (the *ror* command) will be addressed to the server's IP address, extracted from the first segment of the confirmation.

A response to the *ror* command can be either the retransmitted confirmation or an indication that there is no confirmation-on-file. The case when there is no confirmation-on-file can occur if the server had never received the request (or all the request segments, in the case of a multi-segmented request) or if the server has already erased the confirmation. In the case when there is no confirmation-on-file, the MH-MCSP will retransmit the request from the beginning.

A similar retransmission procedure is used for the initial transmission of a request. If no handle is received during the appropriate *retx.time-out*, the request is assumed lost and is retransmitted. Multi-segmented requests are treated similarly and the *retx.time-out* starts from the first segment.

If there is an error in the request data (or in the data of a segment), the server may either request retransmission of the request from the MH-MCSP or ignore the request all together. The actual implementation depends on the server type and on the implementation strategy.

Finally, when errors occur in the initialization procedure, a similar retransmission procedure is used.

4. Concluding Remarks

We have described a novel communication protocol for mobile environments. The protocol, termed by us Mobile Client Server Protocol, resides on top of the transport layer (mainly UDP) and provides services to the application layer. Thus MCSP is middleware.

The purpose of the MCSP is to mask some of the mobility effects from the mobile user and the mobile application. To accomplish this, the MCSP requires implementation of mobility extension to IP, such as the IETF IP mobility support, on which we have based our design described in this paper.

The MCSP allows a mobile application to initiate queries without the application having the need to learn about the environment (i.e., the network) that it is currently visiting. This is accomplished by the FA acting within the wireline network as an intermediary (agent or surrogate) on behalf of the MH and by the idea of the handle, which is returned to the MH upon determination of the server that is chosen to execute the MH's request. The handle contains the IP address of the server. Thus the handle can be used in future communication between the MH and the server, even after the MH is handed-off to another FA. Furthermore, the MCSP supports disconnections, which are so characteristic of the mobile environment.

We have implemented a partial set of the MCSP functionality as part of the *MobiNet* experimentation ([4] and [5]). Because of space limitation, we will report on our implementation experience in future publications.

Of special interest is the applicability of the MCSP to location-based services. Since the MCSP supports location-independent access, it can be used to provide location-based services by having the FA select the servers that provide services relevant to the current location of the mobile. An example of such an environment is the Intelligent Vehicle Highway Systems [1], in which, for example, users may request information about weather and traffic conditions, location of stores and services (emergency services, in particular), or local maps and route calculations.

We expect that, as the mobile networks proliferate, location-independent access will become of major concern to application and service providers. Thus, protocols like the introduced-here MCSP are prime candidates to support communication in such environments.

References

1. "Concepts and progress in IVHS," a session at the PIMRC'95 Conference, Toronto, Canada, September 27–29, 1995.
2. G.H. Forman and J. Zahorjan, "The challenges of mobile computing," IEEE COMPUTER, April 1994.
3. T. Imielinski and B.R. Badrinath, "Mobile wireless computing: Solutions and challenges in data management," Rutgers University, Department of Computer Science, Technical Report.
4. Z.J. Haas, "The progressive execution technique for mobile systems," in Proc. Milcom'95, San Diego, CA, November 5-8, 1995.
5. Z.J. Haas, "On the design of a mobile system," in Proc. Mobidata Workshop, Rutgers University, Piscataway, NJ, Nov. 1, 1994.
6. B.R. Badrinath, A. Bakre, T. Imielinski, and R. Marantz, "Handling mobile clients: A case for indirect interaction," in Proc. 4th Workshop on Workstation Operating Systems, October 1993.
7. J. Ioannidis, D. Duchamp, and G.Q. Maguire, Jr., "IP-based protocols for mobile internetworking," Columbia University, Department of Computer Science, Technical Report, 1992.
8. A. Bakre and B.R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," in Proc. 15th International Conference on Distributed Computing Systems, Vancouver, Canada, June 1995, pp. 136–143.
9. A. Bakre and B.R. Badrinath, "Handoff and system support for indirect TCP/IP," in Proc. Second Usenix Symposium on Mobile and Location-Dependent Computing, Ann Arbor, Michigan, April 1995, pp. 11–24.
10. D. Cheriton, "Dissemination oriented communication systems," Stanford University, Computer Science Department, Technical Report, 1992.
11. R. Chang and R. Ravishankar, "A service based acquisition mechanism for the client/server model in Cygnus," in Proc. 11th ICDCS 1991, pp. 90–98.
12. S. Nanda, D.J. Goodman, and U. Timor, "Performance of PRMA: A packet voice protocol for cellular systems," IEEE Trans. on Veh. Tech., 1991, pp. 584–598.
13. Z. Haas and C.-L. I, "On handoffs in packetized wireless systems," in Proc. Globecom'93, Houston, TX, 29 Nov.–2 Dec. 1993.
14. C. Perkins (ed.), "IP mobility support," Internet Engineering Task Force, Internet Draft, draft-ietf-mobileip-protocol-15.txt, 9 Feb. 1996.
15. A. Tanenbaum, Computer Networks, Englewood Cliffs, NJ: Prentice-Hall. 1981.
16. D.R. Cheriton, "VMTP: A transport protocol for the next generation of communication systems," in Proc. ACM SIGCOMM'86, August 5–7, 1986.
17. B.C. Neuman and T. Tso, "Kerberos: An authentication service for computer networks," IEEE Communications Magazine, Vol. 32, No. 9, 1994, pp. 33–38.
18. R. Cáceres and L. Iftode, "The effects of mobility on reliable transport protocols," Matsushita International Technology Labs, Technical Report MITL-TR-73-93, 1993.
19. P. Manzoni, D. Ghosal, and G. Serazzi, "Impact of mobility on TCP/IP: An integrated performance study," IEEE Journal on Selected Areas in Communications, issue on Mobile Computing Networks, 1995.

Zygmunt Haas received his B.Sc. in EE in 1979 and M.Sc. in EE in 1985. In 1988, he earned his Ph.D. from Stanford University and subsequently joined AT&T Bell Laboratories in Holmdel NJ as a Member of Technical Staff in the Network Research Department. There he pursued research on wireless communications, mobility management, fast protocols, optical networks, and optical switching. From September 1994 till July 1995, Dr. Haas worked for the

AT&T Wireless Center of Excellence in Whippany NJ, where he investigated various aspects of wireless and mobile networking, concentrating on TCP/IP networks. As of August 1995, he joined the faculty of the School of Electrical Engineering at Cornell University as Associate Professor. Dr. Haas is an author of numerous technical papers and holds twelve patents in the fields of high-speed networking, wireless networks, and optical switching. He has organized several Workshops, delivered tutorials at major IEEE conferences, and serves as editor of several journals. He was a guest editor of two IEEE JSAC issues ("Gigabit Networks" and "Mobile Computing Networks"). Dr. Haas is a Senior Member of IEEE and a voting member of ACM. His interests include: mobile and wireless communication and networks, personal communication service and high-speed communication and protocols.