System Support for Mobile Distributed Applications

A. Schill, B.Bellmann, W. Böhmak, S. Kümmel

Dresden University of Technology, Faculty of Computer Science, 01062 Dresden
e-Mail: schill@ibc.inf.tu-dresden.de

Abstract

With the widespread use of distributed systems on one hand and the rapid deployment of mobile computing and communication infrastructure on the other, it becomes important to link both technologies together. This paper first outlines new problems arising from distributed mobile computing and then presents a software support architecture and system for mobile applications. We then discuss a system model for structuring mobile applications, a station software infrastructure for managing resource access in dynamic mobile environments, and a description technique for specifying behavioural aspects of mobile applications. The implementation is based on the remote procedure call of the OSF Distributed Computing Environment and on Microsoft RPC. First experiences with our prototype are reported and directions for future development are outlined.

1. Introduction

Distributed applications have become commonplace in many application areas such as office automation or computer integrated manufacturing. Typically, such applications are based on the client/server model and are using Remote Procedure Call (RPC) as their basic communication mechanism. One of the important industry standards in this area is the Distributed Computing Environment (DCE) of the Open Software Foundation (OSF) /DCE93/. In addition to RPC, it offers directory services for naming and resource management, security services for encryption, authentication and authorization, and several other features. However, like most other related approaches (such as OMG´s CORBA /OMG92/), it is dedicated towards a conventional network infrastructure with static hosts and links.

In this paper, we describe a support architecture for mobile distributed applications. From the hardware point of view, this includes mobile hosts (i.e. notebooks, notepads and similar equipment), wireless LANs, cellular WANs, but also conventional networks, PCs and workstations. In particular, mobile hosts can migrate between subnetworks and can be dynamically attached and detached to/from networks. This enables various new kinds of cooperative diostributed applications, including support for mobile service engineers, mobile office staff, or mobile traffic control systems. However, several new application-level problems arise from that, too (see also /IMB94/ for a general discussion):

- *Dynamic configuration:* Mobile hosts dynamically move between various subnetworks and organizational domains. Therefore, frequent changes of the overall system structure result. Although IP addressing transparency has already been provided by approaches like the virtual internet protocol (VIP; /TET93/, /MYS93/), dynamic configuration changes have´to be considered explicitly by application-level management facilities, for example during access to RPC servers /DAS94, SPT94/.
- *Resource access:* Due to dynamic configuration changes, the availability and quality of resources that are available to a mobile host vary frequently. A major goal of appropriate system support is to nevertheless provide as much resource access transparency as possible.
- *Quality of communication mechanisms:* In mobile environments, the quality of communication facilities also varies significantly /MEW93, DPB94/. Namely, throughput, delay and costs of cellular WANs like GSM /RAH93/, wireless LANs and conventional networks are very different. The system support software should incorporate these aspects as a base for decisions concerning application-level interaction paths and resource access policies /ATD93/. However, at the application-level, widely accepted and established communication mechanisms, namely RPC, should be used.
- *Disconnection and caching:* A mobile host can be disconnected temporarily. Therefore, new requirements concerning cache management arise in order to enable continuous usage of application programs. First solutions are offered by advanced file systems like CODA /HUH93, SKM93, KIS92/.

The major contributions of this paper with respect to these requirements are:

- *System model:* We present a system model that explicitly represents organizational domains of a

mobile infrastructure. In particular, dynamic structural aspects, quality of service of communication paths and management of application-level resources are addressed. Based on this model, generic resource management policies for mobile environments are presented, extending former approaches such as /STW93/ for customizing mobile applications.

- *Mobile station structure:* A detailed software architecture for mobile stations has been developed as a major part of the system model. It provides components and facilities for achieving the desired level of transparency concerning resource access in mobile distributed applications. Moreover, it offers an interface to use the advanced services of our infrastructure at the application level. All remote interactions between system components are based on DCE RPC and Microsoft RPC facilities, i.e. on industry standards.
- *Application description and management:* Decisions concerning resource access are triggered by application behaviour. As a base for that, a formal model of applications is provided based on extended state machines. It is also shown how this is used at runtime and how it is integrated with the overall architecture.
- *Implementation:* A prototype implementation of the approach has been completed under OSF/1 and Windows NT in C and C++. Experiences and first results are reported.

The paper is organized as follows. Section 2 presents our system model and a corresponding example application. In section 3, the details of our station structure with its components and resource management mechanisms are discussed. Section 4 presents the application description based on an example. Section 5 discusses our implementation experiences and section 6 concludes with an outlook to future work.

2. System Model

2.1.Basic Structure

As a base for mobile application support, a distributed environment is divided into logical management areas called domains. An example of the resulting system model is shown in fig. 1. It comprises various domains that may represent departments, organizational units or customers of a company with mobile staff, for example. Each domain can consist of different subnets and includes a number of fixed and mobile stations. However, it is important to note that a domain is a logical rather than a physical construct; therefore, it can also include widely dispersed subnets or stations if they are considered to be organizationally close (although we will typically observe a strong correlation between logical and physical structure). Each domain is controlled by a domain manager. This component manages an information base with

- all stations that are currently members of the domain,
- the application-level services and resources of the domain that are offered via RPC,
- an abstract representation of the network topology that belongs to the domain,
- and an adaptable list of peer domains where frequent inter-domain interactions exist

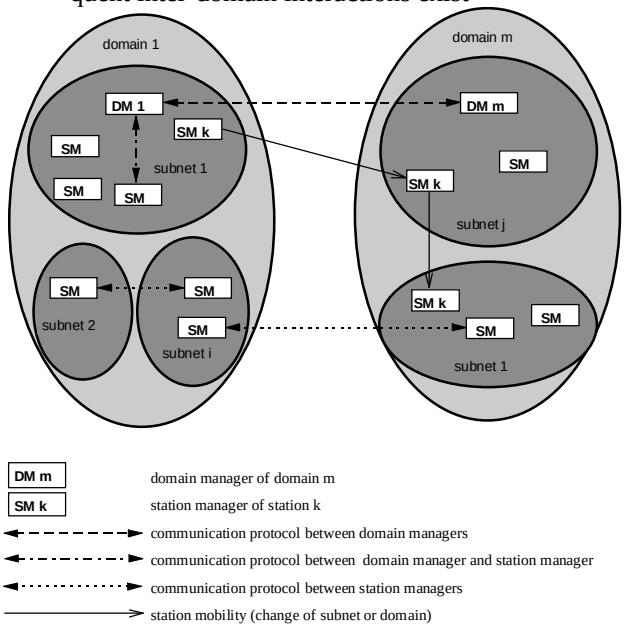

Fig. 1: Domain model

Each station comprises a station manager that is responsible for station-specific management tasks including planning of remote communication and resource access, caching of information, and interactions with the domain manager. (Fixed) stations provide services and resources to the distributed environment. This functionality is detailed in the following section. A station is a member of at most one domain. Although stations usually interact directly with each other via RPC at the application level (i.e. without involvement of the domain manager), communication with the domain manager is required for various specific tasks.

The interactions between station and domain managers are based on a domain access protocol (DAP). It enables stations to dynamically enter and leave domains and to locate services and resources. When it enters a domain, a station registers itself with the domain manager and transfers station description information to it. Upon leaving a domain, the domain manager deletes the station from its list of domain

members and adapts the corresponding resource information.

The domain managers themselves interact via a domain interchange protocol (DIP). It implements the exchange of information about member stations between peer domains. The information exchange is done periodically provided that there have been any changes. This enables remote domain managers to keep hints about stations within other peer domains. The list of peer domains of a given domain can be adapted dynamically in order to add or remove stations from the information exchange policy. All DAP and DIP interactions are based on DCE RPC and Microsoft RPC.

2.2.Example

Fig. 2 shows an example of the domain model. A mobile service engineer is working for a company that has various clients. While he is usually a member of the main domain of his company, he can also access remote domains via wireless networks when he is on the way to a client. Moreover, he can become a member of a client domain when being connected to a subnet of the client, for example. This leads to the communication paths shown in the figure.

The following scenarios illustrate some basic processing functionality in the domain environment:

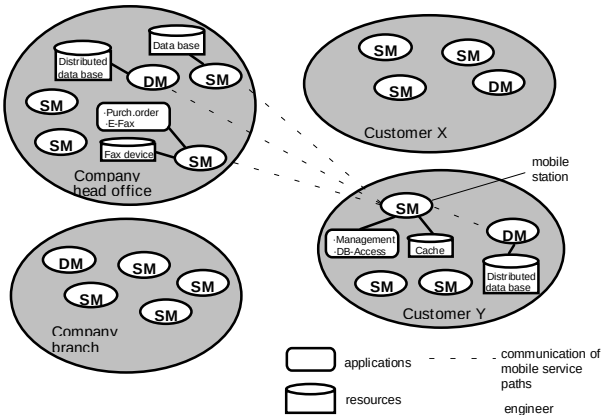

Fig. 2: Example domains

- *Remote access to foreign domain:* In the client environment, a machine failed. Due to high availability requirements, the service engineer accesses the client domain already via a wireless network (e.g. based on GSM data communication facilities) while on the way to the client. Resource access and bandwidth constraints have to be considered explicitly.
- *Remote access to home domain:* When integrated dynamically into the client domain, the service engineer unexpectedly has to access technical documents on a file server of his home domain. This is done via a traditional WAN (like DATEX-P or ISDN) with dynamic network access (via PCMCIA) in the client domain. The documents may be transferred back after completion in order to be processed in the home domain by a workflow application.
- *Caching:* The service engineer will usually also make use of dedicated caching techniques before changing domains. For example, service documents and information about the client will be cached in advance in order to reduce accesses to the home domain via public networks. This requires a priori information about required resources and about domain-specific resource availability.

In the following section, these resource access mechanisms are imbedded into a more general policy.

2.3. Generic Resource Management

In order to generalize the mechanisms discussed above, we first present a classification of typical characteristics of resources used by an application:

- *State:* A resource can be stateless (SL, such as printer services) or stateful (SF). In the latter case, we further distinct between stateful static resources (SFS) that are only modified by a single application (like individual service data of a service engineer), and stateful dynamic resources (SFD) that are modified concurrently over some period of time (like a global product repository used by service engineers).
- *Emulation:* In case of resources of type SF, it can further be classified whether a resource can be emulated (ER - emulatable resource). In this case, a user can be provided with acceptable limited resource functionality by a cache-based emulation. A typical example is a local e-mail emulation in disconnected mode or a program for performing product analyses without access to a global service database.
- *Amount of data:* Finally, the actual amount of data associated with a given resource has to be considered at a coarse-grained level. Several generic categories for describing resource data amounts are introduced that are a base for our heuristic decisions.

Based on this information, various policies are being provided by the system in case of announced disconnection with expected reconnection in a given target domain; examples have already been discussed above and are generalized now:

- *Full caching (FC):* The mobile host generates a cached copy of resource data of type SFS but can also select a subset of these data in an application-specific way (in the case of an unexpected disconnection, this is of course not possible unless ex-

plicit advance specifications of data to be cached exist and are evaluated (not implemented by our approach yet; see /SKM93/ for details)).

- *Remote access (RA):* With resources of type SFD, remote access to the home domain is usually required. For deciding about such access, the topology information of the domain manager and the resource data amount (to be transferred to the destination domain) have to be considered.
- *Use of resources in destination domain (RD):* For resources of type SL, a mapping onto similar resources in the destination domain is possible. The required resource information is provided by the domain manager, subject to typical access control of DCE.
- *Resource emulation (RE):* For resources of type ER, full caching with subsequent resource emulation in the destination domain is possible. This is similar to the full caching case, but typically the resulting functionality is limited.

Kind of resource Kind of connectivity	SL	SFS	SFD	ER
Disconnection	RD (printer)	FC (personal service data)	FC (with manual reintegration)	FC / RE (e-mail)
Low bandwidth, high cost	RD	FC	RA (service database)	FC / RE
Medium bandwidth	RD / RA	FC	RA	RA
LAN environment	RA (evaluation program)	RA	RA	RA (e-mail with remote access)

Fig. 3: Resource management decisions

The table in figure 3 summarizes the basic decision policies within our approach based on the above classification and gives further hints to examples. The following sections present our detailed implementation architecture that addresses resource management in such a generic way as a platform for mobile application support. A key component is the station management that is detailed in the next section. Moreover, we also present our technique for describing applications at an abstract level in order to provide the required a priori information for making the above decisions about communication and resource management at runtime.

3. Station Management

3.1. Basic Concepts

Station management enables the applications on a given (fixed or mobile) station to make use of our mobile support environment. A major goal is to provide a relatively system- and application-independent generic support that is operational on a large number of platforms and for a large number of applications. Therefore, the station manager offers techniques that support mobile computing and communication in a generic way rather that implementing specific mobile application services (such as mobile e-mail as a self-contained service).

An application accesses the station management functionality via so-called subsystems (see fig. 3). They provide the required mapping onto system-specific underlying architectures. At the moment, we consider the following kinds of implementation-level subsystems:

- *Extension of standard C-lib:* Standard functions of the C-lib such as fopen(), fclose(), read(), write() etc. are encapsulated and are extended by mobile functionality; this includes operations in disconnected mode and operations on substitute files in foreign domains, for example. Applications that are linked with the extended library do not have to consider mobility aspects, i.e. existing conventional applications can be enhanced with mobility features this way.
- *Extension of operating system functions:* Kernel functions are extended by new features considering mobility, especially in the file system area. This solution is similar to the former one but is implemented at the lower level within the file system. Both OSF/1 and Windows/NT, the operating systems used by us, offer the required functionality.
- *Special APIs:* For new applications that are explicitly aware of mobility, new APIs to explicitly use mobility features are offered. This way, the components and functionality of the station manager can be directly used by an application. Optionally, the subsystem can also provide additional functionality, for example for user or resource management.

Based on the subsystem approach, the station manager itself is highly portable and is decoupled from application- and system-specific integration problems. The only prerequisite is the existence of DCE-conformant RPCs and threads. The interactions between the subsystems and the station manager are based on local

RPCs that are highly efficient under Windows/NT, for example. This way, the implementation is independent from specific interprocess communication mechanisms. The components of the station manager communicate via internal APIs and are operating as a single process that comprises multiple threads.

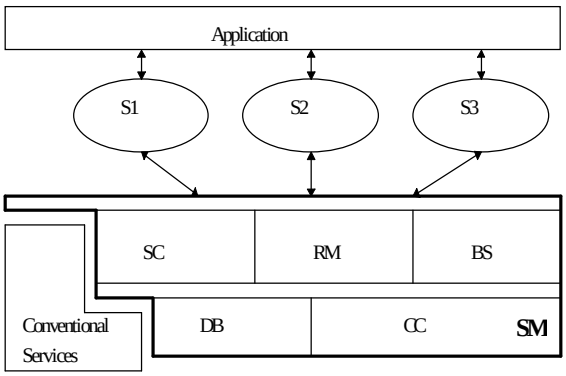

Fig.3: Structure of the station manager

3.2. Architectural Components

The station manager is structured internally according to fig. 3. It is based on conventional operating system and RPC services. The different components offer the following functionality:

- *Resource manager (RM):* This component manages resources and services of the station, makes decisions about resource access, and handles resource access requests from other stations. Decisions are made about existing alternatives for specific resource access requests; examples are access of a cached file after disconnection or access of an alternative or substitute RPC server in case of bandwidth or cost restrictions. The RM uses the base services (BS) and the database (DB).
- *Support component (SC):* The support component implements the alternatives that are selected by the RM. This mainly comprises advance caching of data with the corresponding cache access mechanisms. Moreover, access to alternative and substitute RPC servers is also handled by SC based on existing DCE services. SC uses services of BS and DB.
- *Base services (BS) and Connection Component (CC)* The base services implement the communication with the domain management and are also used by the other components. The connection component presents the RPC interface to the domain manager and to other station managers.
- *Data Base (DB):* This component manages all local information that is required by the other components. This includes meta-information about the current domain, the resources and services of the station and also access mechanisms to operating-system-specific information.

The next section describes the cooperation between these components for dedicated application support.

3.3. Application Support by Station Management

The following scenario explains the interaction between the application, the subsystems, the station manager and the conventional services (see fig. 4).

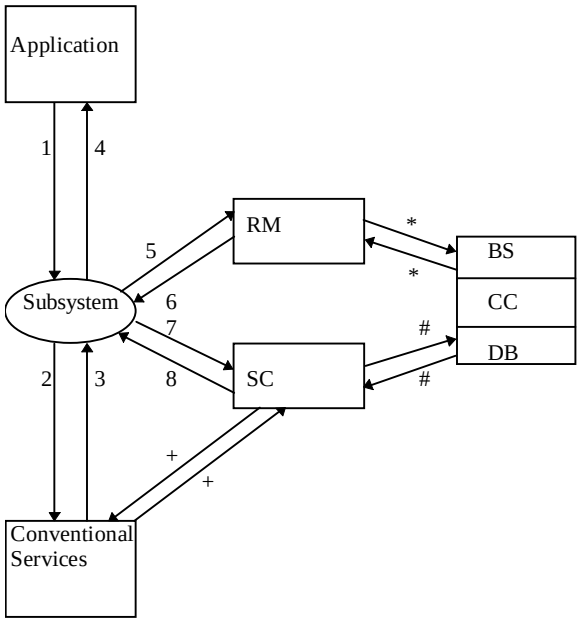

Fig.4: Application support

1,2 The application accesses conventional services via a subsystem. In case of services with mobility aspects, additional functionality is offered by the subsystem. An example is write access to a remote file.

3 Service access (e.g. file access) is affected by a network disconnection. A timeout is signalled at the subsystem level. It can be handled directly by the subsystem or can optionally be forwarded to the application.

4 The application receives a timeout and can react upon it (optional).

5 The subsystem contacts the RM to ask for possible reactions upon the occurred event (3). The RM communicates with BS, CC and DB (*) in order to evaluate the possible alternatives.

6 The RM transfers the calculated alternatives to the subsystem. In our file access example, this can be cancellation of file access, emulation by writing into a local buffer or operation on a locally cached copy. In the latter case, caching must have been initiated in advance as discussed above.

7 The subsystem (or the mobility-aware application) evaluates the proposal of the RM and selects

an alternative. It directs the support component (SC) to offer the selected service or to provide the required service support functionality.

8 The SC offers the basic system support. It uses the BS, CC and DB components (#). The actual service implementation is based on conventional services (+).

3.4. Dedicated RPC Communication Support

Our approach especially supports RPC communication and RPC-based resource access in mobile environments by various alternatives in case of disconnections or communication problems. This functionality is provided as a mobile DCE RPC extension (see fig. 5) and is explained below. Usually, a client has access to an original server. In case of a longer-term quality reduction of a connection, for example in the context of a domain change, the following possibilities can be offered:

- *Access to alternative server:* In this simple case, the existing RPC binding is modified in order to access an alternative server with similar functionality.
- *Access to substitute server with later reintegration:* The RPC binding is modified in order to access a substitute server that emulates the basic functionality of the original server. It acts as an intermediate layer for masking the communication problem and is typically able to cache information and invocations that it later forwards to the original server after reconnection. An example is a file server operating in disconnected mode.
- *Access to substitute server with alternative server:* In this case, the substitute server has access to an alternative server with reduced functionality. The substitute server provides the missing functionality and augments the alternative server this way in order to offer the required level of transparency to the client. An example is a file server interface that provides specific files for read access in addition to files offered by the alternative server.

If the client is completely disconnected from alternative servers and from the original server for a longer period, the replacement server can only emulate the desired functionality; an example is the usage of a local spreadsheet program without access to a global data repository. The implementation of the replacement server is of course application-specific, but the management of replacement and alternative servers can be performed in a generic way.

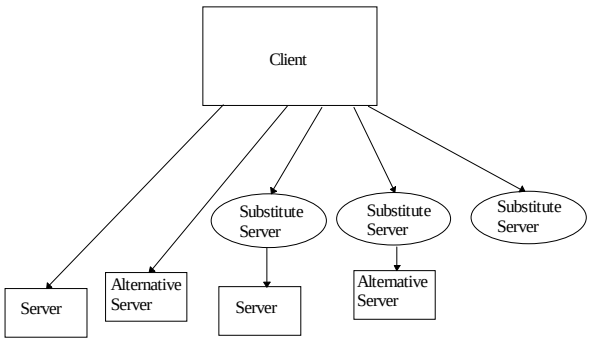

Fig. 5: Functionality of mobile DCE RPC extension

4. Application Description

4.1. Basic Concepts

For making the discussed decisions about resource and service access, the system needs information about the behaviour of the application. As a base, the application is divided into processing phases with associated specifications of service and resource requirements and other necessary information. These data (summarized in fig. 6) are part of the station manager´s data base and are used by the resource manager and the subsystems.

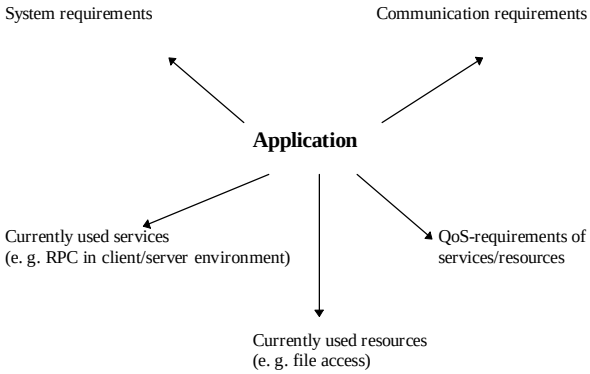

Fig. 6: Information within the application description

As a base for this description, services and resources are classified accordingly. A service description includes the service type, a unique service identifier, the quality of service requirements concerning the system and communication environment, and (optionally) a list of selected servers. A resource description includes a state model and selected performance characteristics of the resource.

4.2. Behaviour Model

The application behaviour is described by a hierarchical state model. This enables a stepwise decomposition and refinement of a given behaviour specification. An application can enter multiple states simultaneously,

for example to model concurrent operations. It is also possible to specify several non-connected automata within an overall application description, for example to model independent processing phases with non-deterministic transitions. A dummy state represents phases of an application that are not relevant with respect to mobility aspects (such as lengthy local processing phases). For each state, an attributed specification of the expected service and resource accesses is given at the type level. This can be instantiated during application setup (for example, by instantiating file types with concrete files). Fig.7 shows an example of a behaviour description for a file access phase.

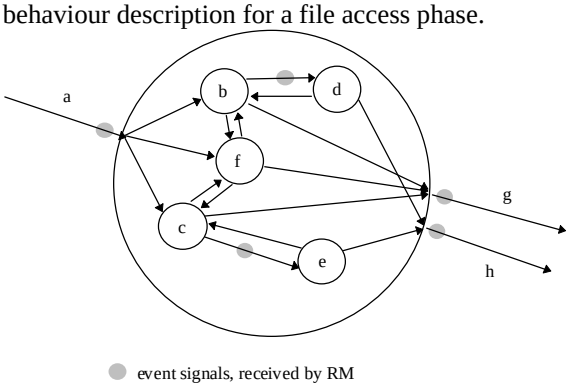

Fig. 7: Description of file access

a	File open
b/c	File read/write
d/e	File not accessible for read/write (support component required)
f	Currently no access
g	File closed
h	File access problem could not be handled by system support

Each state description comprises the following information:

- State identifier
- Integration into state hierarchy
- Entry points for the given state (e.g. fopen)
- Exit points (e.g. fclose)
- Services used in this state (e.g. remote read/write access)
- Required resources (file itself)
- Possible mobility-specific events (service / resource not accessible or not available with required quality (e.g. throughput of file access)

The behaviour description guides the station components during their interaction concerning resource management.

5. Implementation and Experiences

The architecture is currently under implementation under the Windows/NT operating system on 486 and Pentium notebooks and PCs. A preliminary prototype has already been completed under OSF/1 on DEC Alpha 3000 AXP 300; it implements the basic domain architecture and a simple version of the domain access and interchange protocols. All application-level communication is based on DCE RPC and Microsoft RPC. The DCE directory service is used for managing the static part of the domain manager´s information. DCE Threads provide the required concurrency within the station and domain managers. For the protocols between the components, namely the Domain Access Protocol (DAP) and the Domain Interaction Protocol (DIP), first performance measurements have been completed (DEC Alpha 3000 AXP 300, DCE V1.1, TCP/IP, mean value over 10000 interactions); typical roundtrip invocation times were 15 ms, including the exchange of station and resource lists of about 4 kbyte. The example application discussed initially has been implemented for validating the basic concepts.

Major experiences with this rapid prototype implementation are as follows: (1) Even based on a small example, it became obvious that complete transparency of distribution (as addressed by conventional client/server solutions) is not possible nor desirable anymore in mobile environments. (2) However, it is important to represent the network-related information at a rather high level as a base for semi-automatic decisions concerning resource access (rather than using low-level network management information). (3) The use of established industry standards, namely DCE, is crucial for achieving good acceptance of application developers and end users. (4) The use of RPC has significantly simplified the integration of workstation and PC platforms based on the interoperability of DCE and Microsoft RPC using NDR (Network Data Representation). (5) The classification and explicit description of application behaviour and resource characteristics is a major prerequisite for semi-automatic support for mobile applications concerning resource management. We expect to come up with more experiences after completion of our current work on further applications.

6. Summary

The paper described the architecture of a software support platform for mobile distributed applications. It is based on a domain model for structuring an overall mobile environment into organizational domains that also present units of resource and service management. All stations are equipped with a specific support architecture that is accessible via subsystems. This enables existing conventional as well as new mobility-aware applications to take advantage of the support architecture.

The station management approach provides the required functionality to handle domain change, disconnection and communication problems in a uniform way. This includes a decision-making process by the resource manager concerning the required reaction in case of a given event. Moreover, the necessary basic functionality, for example caching, is offered by the support component in cooperation with the data base and other components. A specific aspect of the offered support is mobile RPC; this enables a largely transparent access to alternative or replacement RPC servers.
Future implementation work will include validation of the prototype by various other examples, refinement of the decision mechanisms and development of generic subsystems and services for a wider variety of mobile applications.

References

/ATD93/ Athan, A., Duchamp, D.: Agent-Mediated Message Passing for Constrained Environments; USENIX Mobile & Location Independent Computing Symposium, Cambridge, MA, Aug. 1993, pp. 103-107

/DAS94/ Dasgupta, P.: Resource Location in Very Large Networks; IEEE Computer Society First International Workshop on Services in Distributed and Networked Environments (SDNE‘94), Prague, June 27-28, 1994, pp. 156-163

/DCE93/ Distributed Computing Environment - An Overview; Open Software Foundation, 1993

/DPB94/Davies, N., Pink, S., Blair, G. S.: Services to Support Distributed Applications in a Mobile Environment; IEEE Computer Society First International Workshop on Services in Distributed and Networked Environments (SDNE‘94), Prague, June 27-28, 1994, pp. 84-89

/HUH93/ Huston, L., Honeyman, P.: Disconnected Operations for AFS; USENIX Mobile & Location Independent Computing Symposium, Cambridge, MA, Aug. 1993, pp. 1-10

/IMB94/ Imielinski, T., Badrinath, B.: Mobile Wireless Computing: Challenges in Data Management; Comm. ACM, Vol. 37, No. 10, Oct. 1994, pp. 18-28

/KIS92/ Kistler, J., Satyanarayanan, M.: Disconnected Operation in the Coda File System; ACM Trans. on Computer Systems, Vol. 10, No. 1, Feb. 1992

/MEW93 /Mello, J., Wayner, P.: Wireless Mobile Telecommunications; Byte, Feb. 1993, pp. 147-154

/MYS93/ Myles, A., Skellern, D.: Comparison of Mobile Host Protocols for IP; Internetworking - Practice and Experience, Vol. 4, 1993, pp. 175-194

/OMG92/ Object Management Group: The Common Object Request Broker: Architecture and Specification; OMG, 1992

/RAH93/ Rahnema, M.: Overview of the GSM System and Protocol Architecture; IEEE Communications Magazine, April 1993, pp. 92-100

/SKM93/ Satyanarayanan, M., Kistler, J., Mummert, L., Ebling, M., Kumar, P., Lu, Q.: Experience with Disconnected Operation in a Mobile Computing Environment; USENIX Mobile & Location Independent Computing Symposium, Cambridge, MA, Aug. 1993, pp. 11-28

/SPT94/ Spreitzer, M., Theimer, M.: Architectural Considerations for Scalable, Secure, Mobile Computing with Location Information; The 14th International Conference on Distributed Computing Systems, IEEE Computer Society, Poznan, June 21-24, 1994,
pp. 29-38

/STW93/ Schilit, B., Theimer, M., Welch, B.: Customizing Mobile Applications; USENIX Mobile & Location Independent Computing Symposium, Cambridge, MA, Aug. 1993, pp. 11-28

/TET93/ Teraoka, F., Tokoro, M.: Host Migration Transparency in IP Networks: The VIP Approach; ACM Sigcomm Computer Comm. Review, Jan. 1993, pp. 45-65