Peter Layne

SOFTWARE ARCHITECTURES FOR MOBILE COMPUTING

Pro graduate thesis in
Information Systems
14.11.1999

University of Jyväskylä
Department of Computer Science and Information Systems
Finland

Preface

There are many people who have contributed directly and indirectly towards the completion of this work. When the author started out on the journey towards the attainment of a Master's degree in Information Systems Science at the University of Jyväskylä, the road ahead was fraught with ambiguity, hesitation, and difficulty. These people made that road treavellable. So, without further ado, I would first like to thank my family, both in Barbados and in Finland, for giving their unwavering encouragement and support throughout my studies. More especially to my partner Irina, and to our two lovely and precious daughters, Naomi, who was only 18 months old when I started my studies, and Nadel who is now 5 months old when I am about to complete those studies, for affording me the time to spend all those weeks, months, and indeed, years away from our home in Helsinki, with only weekend visits.

In addition, there have been several people from the University of Jyväskylä who helped to ease the burden studying in a foreign language. Foremost among the people I would like to thank from the university is Mirja Tervo for being the kind and wonderful person that she is. I would also like to thank my supervisor, Professor Jari Veijalainen for the guidance, counsel, and support that he provided throughout this work, and on earlier occasions. Lastly, my deepest gratitude to my fellow Nokian colleagues, Timo Ketara and Jorma Virkkunen from Nokia Mobile Phones, for prove-reading the product of my rambling mind, and for their kind and insightful comments that left me with the endearing notion that perhaps there is something here after all.

Master's Thesis in Computer Science and Information Systems
Department of Computer Science and Information Systems
University of Jyväskylä

Author: Peter Layne
Supervisor: Professor Jari Veijalainen
Topic: Software architecture for mobile computing
Discipline: Information Systems
Date of completion: 14.11.1999
Number of pages: textpages 98, references 8

KEYWORDS: Mobile computing, Ubiquitous Computing, Networking infrastructure, Global connectivity, WAP architecture, Bluetooth, Software architecture, Miniaturisation, Hybrid communicator

Abstract

The concept of mobile computing may be described as a computing and technological alchemy that enables people to have access to computing and informational resources irrespective of temporal and spatial circumstances through such devices as laptops and Personal Digital Assistants (PDA). However, the act of simply having a portable computing device available for use does not in itself constitute mobile, or ubiquitous computing. To capture the true essence of these terms, the situation mandates that some external network is traversed in the process of accessing the required information and services.

The cumulative result of recent and ongoing developments in both the hardware and software domains, in concert with tangible application needs, has given mobile computing the explosive growth that it has experienced in the last decade. However, there have also been developments in other areas that have contributed to this growth. One other such development has been advances in computer networking infrastructure, which has made global connectivity a reality over the wireless medium. These advances have included the launch of the Wireless Application Protocol (WAP) and Bluetooth technologies, as well as the growth of the Internet as a global communication bus.

The subject of this thesis is software architectures for mobile computing. To this extent, the purpose of the research is to examine and discuss the high-level software features of wireless computing. These issues are critical to the development of effective solutions for specific problems pertaining to accessing information over the wireless medium. In particular, the focus will be on accessing the remote network using miniaturised computing devices that are constrained by their lack of computational resources in terms of processing power, memory, and physical characteristics. These devices may be categorised as hybrid communicator devices, and they represent one of the latest developments in the ongoing technological trend towards miniaturisation in mobile computing hardware.

TABLE OF CONTENTS

1.INTRODUCTION

To provide an indication of the stage at which mobile computing is currently, Scott McNealy, Chairman and Chief Executive Officer of SUN Microsystems, recently in a BBC interview from the World Economic Forum described his vision of the future of mobile computing. In doing so, he indicated the pervasive and progressive direction in which mobile technology is currently heading. In his futuristic vision, Mr McNealy foresees a person being equipped only with a cellular telephone, through which they would have access to the Internet. A further step of simply inserting a credit card or bank card would give that person access to their bank accounts, to credit, enable them to purchase a plane ticket, or even a ticket to an ice-hockey (Jokerit) match [BBC99]. Indeed, part of this vision has already been implemented in Finland by a number of local banks. Customers are able to use a GSM cellular telephone running a WAP [Wireless Application Protocol] application to transact banking operations [Dis99d, MER99].

Although the services mentioned here are grouped under the umbrella of mobile computing, in the last five years there has arisen a noticeable chasm in emerging mobile systems based on the characteristics of the mobile device. Early mobile architectures such as Coda [MES95], and DIANA [ACD94] were developed with an underlying philosophy of attempting to mimic traditional desktop or wired systems. These systems can be categorised as file and workflow systems as their usage is usually aimed at accessing common file and database systems to enact high-powered transactions that are executed on devices which have relatively high storage and processor capacity. In stark contrast, many of the modern mobile architectures, such as WAP are categorised as Personal Communication Systems (PCS) [PRW98]. As the term "Personal" implies, such systems are concerned with allowing the user access private information such as bank accounts. Moreover, architectures for PCSs are intended for handheld devices whose trademark characteristic is their dearth of computational resources.

SUN is currently working with a number of device manufacturers with the objective of using Java and other SUN technology to embed computing capability in the most ubiquitous of devices. For instance, imagine a person passing a ring worn on their finger across a car door. This action unlocks the door, the seat is adjusted to suit that person, the radio is automatically tuned to that person's favourite station, the Geographical Information System (GIS) is activated and the current location is indicated on the screen. All of this is as a result of the miniature computer that is embedded in the

person's ring that causes the car to instantly recognise the user and their preferences [SUN99]. Obviously, there are serious security issues in this scenario that have not been considered, but, nevertheless, are assumed to be in place.

At the same time Nokia and a diverse group of other industry leaders are steadily developing and nurturing their own vision of the future for mobile computing through such creations as WAP and Bluetooth. The WAP Forum is a conglomerate organisation that has been created by these industry leaders to spearhead the realisation of their vision. The Wireless Application Protocol specifications are some of the many important achievements that have been attained thus far by the Forum. The goals of WAP, according to published documents are to promote industry-wide specifications for technologies that are useful in developing applications and services that operate in the mobile domain [WAP98].

The objectives of the WAP Forum, as articulated in their published documents are as follows [WAP98]:

- to bring Internet content and advanced data services to digital cellular phones and other wireless terminals.
- to create a global wireless protocol specification that will work across differing wireless network technologies.
- to enable the creation of content and applications that scale across a very wide range of bearer networks and device types.
- to embrace and extend existing standards and technology wherever possible.

These are bold and progressive objectives, and their attainment will generate, and indeed is currently generating, a tremendous and practical impact on all aspects of mobile computing from hardware to software design. To this end, a number of mechanisms have been created in order to render the objectives into concrete reality. The WAP architecture, to be discussed in detail at a later stage, is the primary foundation on which the implementation the WAP specifications will be effected.

In addition to WAP developments, there are also other technological innovations such as the Universal Mobile Telecommunications System (UMTS), and Bluetooth, which are highly pertinent

to the future of mobile computing. The UMTS is the third generation digital cellular standard as developed by ETSI (European Telecommunications Standards Institute), and according to the UMTS Forum, UMTS will take the personal communications user into the information society of the 21st century. It will deliver advanced information directly to the people, and provide them with access to new and innovative services [UMTS97]. The implications of this vision are that all telecommunications networks will be integrated, and that users will have access to data rates of 2 megabits per second in some instances. The UMTS is especially promising for multimedia data, which according to Holley and Costello is what will fuel the future growth of the mobile market [HoC98].

However, despite the obvious strides that mobile technology has made over the recent years, there are a number of factors that are curtailing the development of mobile computing from attaining the stature of ubiquity. One such restrictive factor lies in the fact that ultimately mobile computing is dependent on software to propel it to new heights of usability. However, in their Masters Thesis Rossi and Sillander noted that the entire field of software engineering has been engulfed in a state of crisis for the last thirty years [RS98]. This suggests that the development of mobile applications has not escaped this domain pervasive state of crisis. Moreover, it is a commonly accepted principle among software specialists, and indeed, all who are knowledgeable of systems development activities, that a software product to a high degree relies on the software process that created that product [Hum89, HBL94].

Another formidable challenge to the development of mobile computing can be termed as environmental deficiencies. More specifically, these deficiencies are a direct manifestation of the inherent characteristics of the mobile hosts, as well as of the wireless data transmission medium. These problems have been acknowledged by all researchers who have ventured into the mobile computing domain. The following is a concise, but not exhaustive, list of the constraints affecting the mobile computing domain that was advanced in [PiSa97].

Characteristics of wireless medium [PiSa97]

- Fixed but low bandwidth
- Frequent disconnections
- Unpredictable disconnections
- Financially expensive
- Broadcast is supported by cell
- Unreliable
- Asymmetry

Characteristics of Mobile hosts [PiSa97]

- Limited memory
- Limited computational power
- Small screen
- Applications integrated in device interface
- Limited battery life
- Relatively unreliable
- Frequent location updates

A cursory glance at the above list inevitably leads to the conclusion, as was noted by Welling and Badrinath, that mobile hosts are afflicted with a dearth of computing resources in their tasks [WeB95]. Indeed, it is for these very reasons that Katz went to the extent as to use the term "adaptive communication" because of the need of the mobile system to deliver information in the face of a constantly changing environment [Kat95].

The purpose of the preceding discussion was to provide a brief synopsis of the prevailing sentiment in the mobile computing domain – the tremendous market potential, as well as the current research trends, whose road ahead is filled with bright and exciting possibilities, but at the same time, is fraught with challenges. It is against this backdrop that the topic of this thesis, *software architectures for mobile computing*, is broached. In this vein, it is important to recall the dictum of Heineman et al that a software product is only as good as the process that produced [HBC94] it. For the purposes of this thesis, these words are rearranged somewhat to reflect the supposition that a mobile application is only as good as the underlying architecture upon which it is built. It is through the development of efficient and well-designed software that the dazzling potential of mobile computing can reach fruition.

Furthermore, despite the far-reaching impact of such developments as WAP, and UMTS, the deep-seated environmental and resource challenges remain for the moment implacable. However, the design of mobile software is entirely within the grasp and imagination of the builders of such systems. Mobile computing software must, through the architectural composition, introduce effective techniques for dealing with the information acquisition, resource management [ImBa92], and data distribution [ImBa93] policies that can temper the rigors of the mobile domain.

1.1 MOBILE DEVICES

It has been stated that mobile hosts, regardless of any impact of future technological advances, will always suffer from limited computing resources [PiB94]. However, when considered today this statement must be judged with some degree of scepticism, since the definition of what constitutes a mobile host in the context of resource deficiencies is becoming increasingly blurred. Nowadays it is a common occurrence to find laptop computers coming equipped with memory in the gigabyte ranges, desktop comparable processors, and excellent graphic display units. Clearly, such devices do not suffer from any scarcity of computing resources when compared with their desktop counterparts.

As a result of the confusion surrounding the demarcation boundaries of mobile devices, a clear need has arisen for the formulation of some taxonomy of these devices. To this end, Lawton has presented a limited categorisation of handheld digital wireless devices into Smart Phones and Communicators [Law99]. However, Veijalainen has advanced a more comprehensive taxonomy of mobile devices [Vei99]. According to this latter approach, mobile devices may be grouped into one of four categories, which are the following:

- Portable computers / laptops
- Communicators devices such as the Nokia 9110, and other PDA devices
- WAP phones
- Cellular phones

Table 1 provides an indication of the resource levels of the mobile devices, together with traditional desktop terminals.

TABLE 1.1: MOBILE DEVICES

Device Type	Computing power	Memory	Storage	Weight	Display size Energy requirements
Workstation Desktop PC [SUNa99]	Very High	Gigabytes	Gigabytes, Terabytes	<200 kg	15 - 21 inch None 1600*1200 Pix
Laptop Computer [MSL99]	High	>32 and <200 Mb	Gigabytes	<4 kg	<35 cm Battery – usage is <15 hours 1024x768 Pix
Communicator [Kon98]	Low to Medium	<20 Mb	<100 Mb	<1 kg	<20 cm 8 – 72 hours per recharge 640x240 Pix
Nokia 9110 [Hybrid] [Noka99]	Low to Medium	8 Mb	4 Mb memory card	253 g	<5 cm 3 – 6 hours 640x200 Pix
WAP Phone [Nokb99]	Low	<4 Mb	Same	141 g	2 – 5 hours
Voice Phone	Very Low	< 1 Mb	Same	50 – 100 g	50 – 80 hours – depends on the model

The complexities that are manifest in the software architecture of mobile software are a direct result of the technological trend towards smaller and more portable devices, which in turn are characterised by less computing resources. It is against the backdrop of the restrictions that are imposed by miniaturisation that the importance of software architecture assumes its importance. This is because an application must implement communication and data management strategies such as location of applications and data, when to transmit information, and where and when to utilise caching throughout the communication medium, among other issues. Decisions made in light of these issues must be fully cognisant of the level of resources that are available.

As is evident from Table 1.1, laptop computers share a considerable common ground with their desktop counterparts in terms of computing resources such as memory and computing power. Consequently, the computational and data management strategies that are embedded in the underlying software architecture of applications for such devices will be radically different to those for other mobile device categories. For example, applications that can be designed for laptop

environments are able to utilise large-scale caching and other strategies that can utilise the vast memory resources that are available on these devices. The laptop environment typically supports application groups such as workflow and file systems, such as DocMan [BuB96], and the Coda File System [MES95], which are high processor and memory consuming systems. In addition, the communication environment is also affected to the extent that memory rich devices are able to mask the impact of disconnections easier than other devices by their ability to carry on working off-line during periods of network disturbance.

In the taxonomy of mobile devices advanced by Veijalainen, four classes of mobile terminals were proposed [Vei99]. However, it is proposed here that this taxonomy should be extended to accommodate hybrid communicator devices such as the Nokia Communicator, since the type of device that the Nokia communicator typifies is more than a WAP phone. However, at the same time the Nokia Communicator does not enter the realms of the PDA echelon when compared to terminals such as the Hewlett Packard 620LX [Kon98]. The rationale behind the proposal for a separate category is further founded on the view that the software methodology behind applications developed for PDAs, WAP phones, and hybrids must be different due to the diverse physical characteristics of these devices. Each of these device classes are endowed with significantly differing levels of memory and processing resources, which play a crucial role both in decisions pertaining to the data management strategy employed by applications as well as to the mobile operating system that is chosen to run the hardware.

Indeed, in presenting the mobile operating systems that currently competing for the mobile device market, Lawton acknowledged the need for a distinct hybrid communicator category of mobile devices [Law99]. According to Lawton, hand held digital devices are grouped into smart telephones and communicators. Smart telephones are voice centric devices with information functions, such as Nokia's 7110 WAP phone. This device offers voice capability in addition to fax, SMS (Short Message service), web access, as well as a range of other applications. In contrast, communicators are information centric devices that carry voice capability. Such devices clearly fit into a hybrid family. The major difference between a genuine communicator and a hybrid communicator is the latter's dual capability in being able to conduct telephony as well data communication. The effect of this dualism is manifested in hybrid communicators having significantly fewer resources than full-fledged communicators.

Despite the debilitations that arise from the minuscule size of mobile devices, in some instances the methods that are employed in loading applications exacerbates these deficiencies. For instance, the Nokia Communicator already suffers from lack of memory, and a minimised keypad. However, at the same time, a separate key is allocated to each application that is supported by this terminal. This situation is inefficient in two respects. Firstly, on the physical plane, these keys can be utilised more efficiently to provide added expressiveness and functionality in the communicative context. Secondly, when the applications are "hardwired" into the device, it restricts the ability to incorporate additional services and applications at a later stage in a transparent fashion without inducing confusion.

Consequently, it is proposed that there is a need for the development of a method that will dynamically and transparently load applications that are associated to a user into the mobile terminal. The value of such a model lies in the independence that is imparted to devices. In that, the future application and service requirements of the user will become extensible features that can be modified independently of terminal features.

1.2 HYPOTHESIS

In the preceding introduction a general overview of mobile computing, as well as the current technological trends was sketched. This section serves to elucidate the intent of the research thrust of this thesis in greater detail. Accordingly, in exploring the subject of *software architectures for mobile computing, the intent is to develop an understanding of the role of application architecture in the wireless environment based on the requirements of emerging applications, and to develop solutions to pertinent data management problems therein. In addition, a practical software architecture that can be used in building applications for the mobile domain in terms of the Personal Communication System will be proposed.* In performing these tasks, special emphasis will be placed on the processing of continuous data, such as multimedia data, in the mobile domain. The definition of continuous data is any streaming data, which can include text, streaming audio and video content, as well as geographical and image data.

Thus, the research focus of the thesis is two-fold, although the two streams are inextricably bound up in each other. In that, it is only through a comprehensive appreciation of the characteristics of the mobile environment, that the researcher is fully equipped to proceed to the next step of actually developing a practical software methodology.

In terms of developing an appreciation of the factors that influence architectural considerations for mobile applications, a number of sub-issues must be considered. Those sub-issues that will be featured in this in the task of elucidating the main theme are the following:

- What are the environmental and technological challenges that confront mobile computing, and how do they influence architectural considerations? For example, in the Nokia communicator the loading of applications is physically embedded in the device so that a separate key is associated with each application. The question here is how to incorporate extensibility in the software architecture so as to separate the hardware features from the user's current and future applications needs.
- What are the available mobile system architecture models, and how should they be considered in the task of constructing a mobile application?
- What are the data management policies that should be included in design, for example, caching, replication, and the loading of applets and WMLScripts (Wireless Markup Language)? In that, should the mobile devices resemble network computers and load the applications from the server, and perhaps the data as well?
- What are the data management concerns in terms of processing continuous data, and how can solutions be integrated into the overall architecture?
- How should resource management be handled in the mobile environment, especially in terms of continuous data transmission, and how can QoS (Quality of Service) parameters be implemented in the underlying architecture? The aim is to provide applications with up-to-date and relevant information on their environment that empowers them to make intelligent decisions bandwidth, power, and other limitations.

The conclusions, as well as the discernible application development trends that result from the preceding theoretical and conceptual discussion will be used as input in the task of proposing a practical software architecture. To this end, *it is envisioned that the proposed architecture will serve applications that are intended to execute on hand-held hybrid communicator devices, such as the Nokia 9110 Communicator. In addition, the architecture will facilitate the processing of continuous data.*

1.3 WHAT IS SOFTWARE ARCHITECTURE

As stated earlier, the purpose of this thesis is to explore the role of software architecture in the mobile domain, and to propose a practical example of an architecture that can be applied in solving a particular breed of problems. However, a critical question that must be posed before starting on that long and winding road must be "what is a software architecture"? The purpose of this section is to shed some light on this subject.

According to Pressman, software architecture refers to two important features of a software application. These two features may be stated as the hierarchical structure of the procedural components, and secondly, the structure of the data [Pre94]. Thus, the concept of software architecture is intimately linked to the concepts of simplification and abstraction. To further grind down the definition of software architecture, Mellor and Johnson takes it one level higher in an attempt to grasp the quintessential essence of the term. By their definition, software architecture, in its purest form, is a domain specific software methodology that is closely matched with a specific kind of problem [MeJ97].

The definition of software architect adopted by [MeJ97] implies that given the implicit nature of a problem, the system developer's first task is to find the most suitable software architecture that can be applied in expressing the problem in some lower level format. Thus, once the architecture has been selected, it provides the developer with a set of high level abstractions to express the application. If this modus operandi is viewed from another perspective, it can be argued that the systems developer regresses from the architecture to the problem specification.

Work on architectural styles focus less on the specific problems, and more on the solution. Ideally, an architectural style can be applied to a broad range of problems [MeJ97]. Consequently, an architecture is a representation of the global structure of a system as a composition of interacting parts.

The basic notions that are included in the above stated high level definitions of what constitutes software architecture will remain steadfast in the vanguard of this research agenda. However, in the process of discussing and evolving a unique architecture, the exercise must be steeped in a richer framework of practicality. To this extent, this thesis will define software architecture in terms of the

three basic issues that were advocated Monroe et al. The first of these basic issues is system structure, which characterises a system's structure in terms of high level computational elements, and their interactions [MKMG97]. Thus, the intent will not be focused on the requirements of any specific problem, or on the specifics of implementation details.

The second basic issue of architecture is that they provide rich abstractions for interaction between architectural components, which are often depicted as interconnecting lines. Such abstractions also provide a rich and informative vocabulary for system designers, for example, the remote procedure call paradigm, and the client-server paradigm. The third basic element to be used to reflect the practical properties of software architecture can be termed global properties. In that, the intent of architecture is to describe overall system behaviour. Thus, the types of problems that are focused on relate to the fundamental issues as in, for example, end-to-end data rates, and latencies [MKMG97].

The definition of software architecture as advanced in [MKMG97] is the definition that will be adhered to throughout this thesis in developing the central theme of mobile software architecture. This definition has been successfully applied in a number of other works, which presented unique and well-structured mobile software architectures. For example, the treatment of software architectures, such as DIANA (Display Independence Asynchronous Network Access), and L_2IMBO were presented in light of the three qualities of software architecture as defined in [MKMG97]. This methodology was highly beneficial in highlighting the main goal of the DIANA architecture, which is to provide display and network independence by separating the user interface logic and the communication logic from the processing logic of each application [ACD94]. In L^2IMBO, the emphasis is placed on collating and managing quality of service [QoS] information from a wide range of sources for presentation to higher layers, thus enabling feedback and control throughout the architecture [DFWB98].

Having added an appreciation of the theoretical and practical concepts that underpin software architecture, the stage is now set, the lights are focused, and the actors are in their respective places. All that remains is the unravelling of the plot.

1.4 IMPORTANCE OF TOPIC

In spite of being afforded the prestigious title of "computing paradigm of the future", the road ahead for mobile computing in providing and guaranteeing a stable and efficient computing solution for a

diverse user community is not completely clear. There is still a dire need for further research in a range of sub areas to uncover new techniques that can be applied to the challenges that accrue from the mobile arena. For example, Katz called for deeper research in the ubiquitous wireless network that supports mobile devices [Kat95]. Subsequent to this call being issued, there have been several valuable research initiatives in this direction, such as [FPLV98, PRW98, and SB98].

In a similar manner, the software infrastructural or architecture aspect of mobile computing is also in need of further research and development. This need does not arise only out of the intrinsic relevance and importance of developing efficient software architectures, but in addition, this need for research is further heightened by the recent and ongoing technological development within the mobile domain. The appearance of such developments as WAP, and UMTS has injected additional urgency in the need for new approaches and techniques for confronting data management, as well as a host of other important issues. In light of the above, the primary importance of this thesis lies in its intended focus on structural concepts that are independent of the logic and semantics of specific applications, but which can be moulded to accommodate implementation of current and emerging technologies. Ultimately, the realisation of these generalised software approaches and techniques will assist in strengthening the mobile computing paradigm.

Furthermore, the increasing demand for streaming data applications, such as multimedia applications, on the mobile platform will inevitably result in the building of heavier and more resource consuming applications. In the author's opinion this development inevitably introduces questions pertaining to levels of guaranteed service that must be provided by mobile applications. In addition, object oriented technologies, such as Java are generating a significant impact on the design of mobile applications. All of these developments must be reflected in the software architecture of future mobile applications.

Therefore, developing an understanding and appreciation of software architectures that are pertinent to mobile computing, and of the valuable role that they play in the entire mobile computing schema, will undoubtedly contribute towards the development of richer applications. Such architectures are an essential medium for confronting the data management complexities in the highly distributed environment, as well as accommodating the influx of new technologies.

1.5 PREVIOUS RESEARCH

Research on mobile computing originating from both academia and industry has witnessed a dramatic increase in the last five years. A testament to this rise in research interest has been the establishment of a number of dedicated journals, such as MONET (Mobile Networking and Applications, Mobile Networks), as well as Andrew Seybold's Mobile Communications Outlook. There is also a tremendous wealth of information covering this domain that is located on the WWW. For example, Aline Baggio's [AB99] provides an excellent repository of bibliographic information covering all aspects of mobile computing.

A common feature of all published research findings on mobile computing is the general acceptance of the multitude of environmental and device ailments that afflict the domain. These ailments include insufficient bandwidth, predictable and unpredictable disconnection, and small resource deficient mobile devices [AloK93, FoZ94, Kat95, PiSa97, SaP98, Bahl98].

In fact, it has been the drive to alleviate these ailments that has been the primary focus of academic and industrial research over the last five years. For example, [BaP97] on supporting mobile clients in accessing client server databases, and [Ray94] on the issues involved in implementing mobile computing over the ATM network. Despite the valuable contributions that these and other similar research initiatives have presented, there has nevertheless been a lack of attention given to the development of software architectures for mobile computing that are infrastructural in character. The value of such structural and high-level approaches is evident in the independence that such architectures retain of the different applications and their semantics, while simultaneously permitting customisation of individual applications.

In this vein, [SaP98] has presented an insightful taxonomy of mobile system architectures. These computational models may be grouped into mobile client-server models and mobile agent models. The client server models may be further sub-divided into the client–agent-server model, the client-intercept-server model, and the peer-to-peer model. These generic formulations largely comprise the building blocks, or meta-meta level resources that more concrete software architectures are built on.

Although, as previously stated, research on providing software architectures has not been the subject of extensive research efforts, there have been some highly commendable research endeavours in this domain, which have resulted in imaginative and workable architectures. Such endeavours have included the previously mentioned DIANA [ACD94], and the L^2IMBO architectures [DFWB98].

The QoS issues that are raised in the L^2IMBO approach represent features that are becoming increasingly common place in mobile computing. For example, Kassler and Schulthess adopted the approach of introducing QoS at both the application and the transport levels in order to handle short-term fluctuations on the wireless link [KaS99]. It is certain that QoS issues will become more important as continuous data, and all the complications that are introduced by routing such data over the wireless medium, becomes more pronounced in the mobile computing domain. QoS guarantees are essential requirements for continuous data due to the time critical nature of the data exchange. Such QoS guarantees will include minimum bandwidth reservation, as well as power and processing thresholds, and must be given due consideration in protocol composition of the mobile system.

In addition, there are several on-going industry-led standardisation initiatives that will play a crucial role in the future of mobile computing, and which will impact significantly on software architectural issues. Chief among these standardisation initiatives is the WAP Forum's WAP stack specifications, and development environment that consists of WMLScript, and WML. In addition, the emergence of Symbian's EPOC OS for ROM based computing will provide the stability and support that is needed to drive the mission critical mobile applications of the future.

All of the above work have some bearing on issues pertaining to the type of structural concepts that will be discussed in this thesis. They serve as a starting point in task of developing and ultimately proposing a novel software architecture that can be used in building robust and reliable applications. Another interesting feature pertaining to existing software architectures for the mobile domain is that the great majority of them were developed over five years ago. Given the pace of technological advancement in the mobile computing world, this time period represents an eternity. Consequently, architectures such as DIANA and L^2IMBO were developed without thought afforded to such crucial developments as UMTS, or the now pervasive stampede to funnel multimedia over the wireless

link. This work of this thesis will be among the pioneering efforts that will explore these new topics.

1.6 RESEARCH METHODOLOGY

The research strategy that will be applied in deriving answers to the research questions posed in this thesis is two-fold in character. In that, there are both theoretical and constructive aspects that are involved. To this end, a number of research methodologies were considered in an attempt to identify a research methodology that would afford maximum effect in the tasks of developing and presenting the argued case in the richest and clearest possible manner. For example, Jenkins [Jen85] has developed an insightful taxonomy of research methodologies that can be applied to the IS (Information Systems) field. However, from the standpoint of this thesis, Jenkins' research schema is seen as being too restrictive, as it compartmentalises and isolates the various research strategies.

Therefore, a more pragmatic and cohesive methodological approach will be adopted in deriving and elucidating the aims of this thesis. To this end, an extensive literature review will provide the basis for developing an appreciation of the prevailing issues within the mobile computing domain. This review will encompass literature gathered from traditional printed sources, as in industry and academic Journals, as well as information that is drawn from the WWW. The knowledge that is gleaned from these sources will then be used to develop and construct a theoretical software architecture.

1.7 LIMITATIONS OF RESEARCH

As was stated earlier in the research methodology section, this thesis will enact a version of the systems development methodology. The intent of the research is not to focus on implementation issues such as the coding of any specific application. Rather the intent lies in developing a software methodology that defines how the structure of a particular set of mobile applications should evolve. To this end, this methodology will focus on issues such as how these applications handle their data, how and where the computing is actually performed, and the how results are made available to the mobile device.

In terms of proposing a practical software architecture the emphasis will be on the first two stages of the systems development systems stage of the research methodology that is being applied. Those

two stages are constructing the conceptual framework, and developing the system architecture. In this vein, the motivation behind the proposed architecture will exhibit a predisposition towards emerging technologies such as WAP and UMTS, and to the processing of multimedia data on the wireless platform.

1.8 CONTRIBUTION TO KNOWLEDGE

The results of this will research will extend the knowledge base of the mobile computing domain in its treatment of issues such as computing models, and data and resource management strategies. Furthermore, these issues will be discussed in light of emerging technologies and standards such as WAP, and UMTS. To this extent, this research can be of value to telecommunication industry players such as Nokia, who are heavily engaged in developing cutting edge mobile computing solutions.

Moreover, the results of this research, particularly the proposed software architecture, can also be utilised by software developers of mobile computing solutions. A clear and definitive taxonomy of viable software architectures that are pertinent to current as well as emerging mobile computing trends can help these organisations to more aptly design and build applications that are mindful of both technological trends as well as the restrictions of the mobile domain.

1.9 THESIS OUTLINE

The structure of this work will follow structured and logical flow, starting from a discussion and analysis of the limitations of mobile devices and wireless data transmission, through to the development of a solution software architecture. This chapter provided some insights into the problem domain, and the specific problems that are to be addressed. In addition, chapter 1 provided definitions of important concepts in the context of this thesis, such as software architecture. Chapter 2 is a natural extension of the introductory chapter. Chapter 2 presents a general overview of the challenges that affect and limit mobile computing. In chapter 3, the mobile system architectures are discussed in light what strategies at this level are useful in addressing the highlighted challenges, and in catering to the needs of applications.

The system architectures are followed in chapter 4 by a discussion of the WAP and Bluetooth specifications. These technologies are highly important developments that signal the maturity of

mobile computing. In chapter 5, the generic layers of the mobile software system are discussed, and the JMAM middleware is developed. Chapter 6 expands on what was started in the previous chapter to present the entire JMAM software architecture.

Chapter 7 concludes the work with a brief look into the future technological trends in mobile computing. These technological developments are mainly focused on network technologies that serve to increase the offered bandwidth. This is followed by a discussion of the main findings and conclusion.

2.CHALLENGES OF MOBILE COMPUTING

As can be witnessed from Table 1.1, the resource levels of laptop computers and handheld are quite divergent. When this resource divergence is accepted, the next step is to appreciate that such divergence does influence the software architectural makeup that underlies the applications that run on these devices.

The second implement, which meshes with the actual hardware device to bring the concept of mobile computing into existence, is the wireless medium. However, unlike the situation that prevails in the device domain where many of the challenges are mitigated across the scope of different devices, the challenges relating to the wireless medium remain implacable across the entire spectrum of mobile computing. In addition, new challenges are being introduced into the mobile computing alchemy because of the increasing requirement by mobile applications to process multimedia data.

This chapter examines the technical challenges that are faced by mobile computing, which arise from the characteristics of the mobile hosts and the wireless medium, as well as the newer challenges that are bred as a result of the processing of multimedia data. An understanding of the issues to be discussed here is critical, as these are the issues that must be confronted in the design of mobile systems. The goal of the designer of such systems is to discern a software architecture that exhibits strengths in the areas that are crucial to the implicit philosophy behind the application, and which are reflected in the explicit requirements. The challenges facing mobile computing can be broadly categorised into communication, mobility, and portability associated challenges.

2.1 WIRELESS COMMUNICATION

Network failure is one of the great concerns that afflict mobile communication. This section presents the most perplexing of these communication problems, which include network disconnection, and bandwidth starvation and variability.

2.1.1 Disconnection

Disconnection can occur in the mobile environment due to network failure, or as a result of the (communication) c-autonomous nature of the mobile device. A computer system is c-autonomous within the environment if it cannot be compelled by external systems to start a communication

session, continue that session, or be prohibited from initiating communication with an external system. In addition, mobile devices must contend with lower bandwidth, higher error rates, and more spurious disconnections that are caused by the frailty of the wireless link. This environment forces the designer of mobile systems to grapple with the vexing dilemma of spending resources on the network trying to prevent disconnections, or alternatively on spending those resources on enabling the system to cope with disconnections more gracefully by working around them where possible.

It may be advanced that the more autonomous a mobile device is, the better it is able to withstand the trauma of disconnection. For example, applications that are built on the DIANA architecture reduce communications by running entirely locally on the mobile device when the chances of network disconnection are great [ACD94]. Thus, in environments that are beset by frequent disconnections it is highly advantageous that the mobile device is able to operate as a stand-alone device as opposed to as a mobile terminal. However, this presupposes some minimum levels of resources that the mobile device must be equipped with, which cannot necessarily be met by many of the handheld devices that are the focus of this work.

In some cases, round-trip latency and short disconnections can be disguised by operating asynchronously [FoZ94]. This operation mode is counter to the synchronous remote procedure call paradigm in which the client waits for a reply after each request. In contrast, in asynchronous operation a client can send multiple requests before asking for any acknowledgement. In addition, prefetching and lazy write-back are also viable strategies that have been developed to decouple the act of communicating communication from the actual time a program consumes or produces data. Such strategies permit the mobile device to make progress during network disconnections. In terms of multimedia data, this would imply that it would be more advantageous for the mobile terminal to operate in asynchronous mode, since the delays involved in invoking the reliability standards in synchronous mode could be harmful to the time sensitive nature of such data during transmission.

In situations where the mobile terminal is accessing a remote database the problem of disconnection becomes more heightened. This is because the traditional commit protocols are not amenable to the unpredictable disconnections that are typical of the mobile environment. The simplest, and a widely used, commit protocol is the basic two-phase commit protocol (B2PC), and its derivative, the optimistic two-phase commit protocol (O2PC) [LeC98]. There are two main deficiencies involved

in using these commit protocols in the mobile environment. In the first place, if the mobile client commences a transaction, and is shortly after disconnected, the locks that were placed on data items at sites involved in that transaction will cause such data items to be inaccessible to other users. Secondly, communication failures can violate the transaction atomicity rule. Thus, in B2PC, the adopted solution is to abort the entire transaction. A high number of such aborts can have a debilitating impact on a system's throughput and fault tolerance [BoDe96].

The DocMan document management system [BuB96] provides a good example of a system that handles network disconnection. It achieves this by allowing the user to download important information onto the mobile device. During those periods when the user is disconnected from the network, work is allowed to progress, and any changes that are made to documents can be reconciled with the master repository when network connection is re-established. Here the problems of blocking and site inaccessibility do not arise. This strategy would appear justify the finding that in distributed systems less than 1% of all writes are followed by a write by a different user [KiS92]. However, this statistic has yet to be verified in a mobile environment.

2.1.2 Low Bandwidth

As with disconnection, mobile computing designs are required to be more concerned about bandwidth consumption and constraints than designs for stationary computing because wireless networks deliver considerably lower bandwidth than stationary networks. Although the situation is changing rapidly – even as this work is progressing, Sonera is implementing HSCSD (High Speed Circuit Switched Data) – typical wireless networks achieve only a meagre 9 – 14 Kbps for application data [Dis99c]. Moreover, since the available bandwidth is divided among the users sharing a cell, the effective bandwidth per user is even lower. Indeed, for radio transmissions the error rate is so high that the effective bandwidth is limited to less than 10 Kbps [PiSa97]. Accordingly, in mobile computing bandwidth is very much a premium resource and this is further exacerbated by the financial costs of the little bandwidth that is available.

2.1.3 Bandwidth Variability

Wireless networks vary considerably on the degree of bandwidth and reliability that they provide to mobile devices [Kat95]. However, this is largely dependent on the type of network under consideration. For example, in GSM networks, the issue of bandwidth variability does not arise as current standards are able to guarantee 9.6 or 14.4 bps. On the other hand, the issue of reliability

comes to the fore, since due to the cell dependent nature of the communication, not all terminals are necessarily able to start a communication session.

In terms of networks that are based on Ethernet principles [Sta94] the variability in the amount of available bandwidth is caused mainly by the number of users who are simultaneously sharing the resources of a cell, as well as the interaction of the radio signal with the environment. The fundamental difference between bandwidth starvation and bandwidth variability is that the former is an inherent and largely static characteristic of the mobile domain, whereas the latter is characterised by its unpredictable occurrence.

To effectively deal with the constraint of bandwidth variation as described in the latter instance, the situation calls for mobile systems to be adaptive in their outlook. The term adaptivity entails that systems should be able to detect and to adapt to situations in which there is either an abundance of, or a lack of bandwidth, while maintaining delay sensitive communications. In this vein, [WeBa97] have posited the view that applications should be totally aware of changes in the environment, and should assume sole responsibility for implementing appropriate actions. An opposing extreme of that position entails the system bearing the complete burden of detecting and counteracting all environment changes [NoS95]. However, the most effective adaptation strategy is one that strikes a balance between these polarities. Such a compromise approach would enable individual applications to determine how best to adapt, but at the same time allow the system to centrally monitor the resources, and ensure that they are used effectively [Nos95].

Table 2.1 provides some insightful information on how mobile communication mediums compare with a number of other communication technologies. This table serves to portray the depth of the bandwidth starvation and variability problems that afflict mobile connections. It also serves to indicate the impact that 3rd generation technologies will have on the future of wireless communications.

TABLE 2.1: CONNECTION MEDIUMS

Network Type	Bandwidth	Coverage	Availability	Usage Cost
Fixed LAN (Sta94)	10 M – Gbps	Local	High	Low
Satellite (NoH95)	9.6 Kbps- 3 Mbps	National	Medium	High
GSM				
SMS [Ojan98, Dis99c]	9.6 Bps	International	High	Low
Circuit Switched Data [Ojan98, Dis99c]	14.4 Kbps	Regional	Medium	High
HSCDS [Pere98, Dis99c]	57.6 Kbps	Regional	High	Medium
GPRS [Eggl99, Dis99c]	115 Kbps	Regional	High	Low
EDGE [Dis99c, Pere98]	48 Kbps	Regional	Medium	Low
UMTS				
UMTS (WCDMA) [Pere98, Ojan98]	144 Kbps – 2 Mbps	Global	High	Low
Wireless Broadband				
Wireless LANs [Pere98]	1 – 25 Mbps	N/A	N/A	N/A
Wireless ATM [Pere98]	25 - 155 Mbps	N/A	N/A	N/A

2.1.4 Security Risks

Security risks are higher in mobile computing precisely because it is easier to connect to the wireless link. Therefore, the security of wireless communication can be compromised much more easily than wired communication, and this risk is especially higher if the transmission range stretches over a wide area [Kat95]. This situation heightens the requirement for mobile computing designs to include suitable security measures that would detect and prevent the system from being violated by unauthorised parties.

Secure communication over insecure wireless channels is one of the main benefits of digital networks. As a result of the binary nature of the data, strategies such as encryption and interleaving

can be more readily and effectively introduced into the communication framework, than in an analogue system. The security advantages of digital network is borne out by the reality that up until the present point in time, which represents a period in excess of 10 years, there have been no reported incidents of the GSM encryption algorithms having been violated.

2.1.5 Mobile Multimedia Communication

The term multimedia refers to two or more media – audio, video, or text - that are presented together. As Pekkola noted, all multimedia products can be grouped into one of the following three categories [Pek98]:

- Multimedia conferencing. These systems allow geographically separated users to be able to communicate via audio and video. Some applications also offer means of data collaboration tools such as a shared whiteboards or other common workspaces. Examples of these systems include collaborative virtual environments (CVEs), multi-point videoconference systems and shared workspace applications such as BSCW and the AltaVista Forum.
- Media-on-demand. This is used to solicit some kind of media from the server. For example, if users want to see television-news on their computer, they order news from the media server, which broadcasts the news back to them.
- Multimedia mail. This is like ordinary mail, but it is also consists of some elements of multimedia files which come as attachments. These multimedia files include, for example, MPEG or AVI video or even interactive multimedia applications such as interactive WWW pages. In addition, ordinary files that are transferred with multimedia files through the network can be bundled into this class.

The increasing requirement for multimedia data over the wireless communication link has conjured up additional technical difficulties in mobile data transmission [ACIM95]. The technical requirements that must be achieved for high quality multimedia connections are difficult to meet in the mobile domain. This is because, unlike traditional data transmissions, a critical requirement of real-time data, as is video, and especially audio, is that they demand some minimum network bandwidth and packet loss rate [Pek98]. Furthermore, mobile multimedia communication suffers from the time varying error characteristics of the wireless medium, as well as the power limitations of mobile devices [Bahl98].

Table 2.2 depicts the data categories and the network requirements for each category using a rough but informative scale.

TABLE 2.2: COMPARISON OF NETWORK REQUIREMENTS FOR DIFFERENT DATA

Challenges	**Traditional Data**	**Multimedia Conferencing**	**Media-on-demand**			**Email /Multimediamail**
			Audio	Video	Text	
Bandwidth Requirements [ACIM95, Bah98, PRW98]	No minimum	Minimum level required	Minimum level required			No minimum
			Highest	High	Low	
Round Trip Time (ACIM95, AFZ95, RW98, RaSt98])	Can vary	Small	Small - BUT From server to client is constant – can vary in opposite direction			Can vary
			N/A	N/A	N/A	
Packet Loss Rate [Bah98, RaSt98]	Can vary	Small	Small			Can vary
			None	Small	Can vary	
Retransmission Need [Pek98]	Can retransmit	Only data	No need			Can retransmit
			No	No	Yes	
Disconnection [Pek98]	Can work offline	No	No			Can compose offline

As noted earlier, mobile communication suffers from severe bandwidth restrictions and higher than average bit error rates. It follows from these technical limitations that the response or round trip time is a critical problem that degrades the performance of mobile applications [SaP98]. In terms of multimedia data transmission, as Table 2.2 shows, these limitations serve to aggravate the impact that real-time data transmission constraints impose on network connections.

There are several strategies that can be applied to the challenges posed by mobile multimedia communication. Such strategies include bandwidth reservation, stream synchronisation, buffering

and prefetching. However, the use of any of these strategies must be carefully balanced against the resource limitations of the mobile device. These strategies will be discussed in depth at a later stage.

2.2 MOBILITY

The ability to change locations while still connected creates confusion in IP networks. In that, some data that is considered static for stationary computers must be reconfigured to accommodate the dynamism of mobile computing. For example, although a stationary computer can be configured to statically to seek out the nearest server, a mobile computer requires some independent mechanism that will help it determine which server is most appropriate. The dynamism of mobility reinforces the importance, and indeed, the necessity of ensuring that the concept of adaptivity is ably reflected in all aspects of the mobile paradigm. There are three particularly difficult problems that result from mobility [FoZ94]. These problems are addressed below.

2.2.1 Address Migration

When a mobile terminal moves while connected to the network, its current address needs to be mapped to a new access point. This modus operandi introduces severe difficulties for traditional network addressing policies, since those network protocols were not designed to handle clients whose addresses relocate dynamically over time. Once an address for a host name is known to a system, it is typically cached and given a distance expiration date. For example, the Internet Protocol (IP) assigns each host a fixed IP address network address that identifies the network to which that host is connected. Any data that are addressed to this host contains the location information that is included in the IP address. Consequently, if a mobile host relocates to a new network, the current IP address will not reflect the new point of attachment. In this situation, if traditional routing protocols are used, the mobile host will lose data packets that are addressed to the outdated IP address [Kat95, LDG96].

There have been a number of attempts at developing effective strategies for coping with dynamic addressing [Kat95, FoZ94]. The Mobile IP scheme, as proposed by the IETF (Internet Engineering Task Force), is one of these mobile IP strategies. Mobile IP is an enhancement to IP that allows a mobile host to roam freely on the Internet, while maintaining the same IP address. The Mobile IP architecture achieves this goal by defining two new entities, which are referred to as the Home Agent (HA), and the Foreign Agent (FA). These agents work in close co-operation with each other to allow the mobile host to relocate without changing its IP address [LDG96].

Mobile IP operates in such a way that each mobile host is associated with a unique home network, as is identified in its permanent IP address. The normal IP routing procedure always delivers data packets that are addressed to the mobile host to this network. However, when the mobile host is roaming in another network, a specially designated host on this home network, which is referred to as the home agent, is responsible for intercepting and forwarding those data packets to the absent mobile host [LDG96, Perk98, YWT98].

There are problems associated with the IETF Mobile IP standard, such as the long latency that is involved due to the extended routing of data packets through a third entity. In addition, there is the overhead associated with the exchange of messages between the HA and the FA. Whenever the mobile host moves to a new point of attachment these exchange of messages are mandatory to inform the HA of where to direct packets. However, these inefficiencies are being actively addressed by the IETF, and IP version 6 has corrected many of these deficiencies [Perk98]. Furthermore, there are research efforts, such as to be found in [YWT98] where a new scheme is proposed to make Mobile IP more efficient in terms of the highlighted criticisms.

2.2.2 Location Dependent Information

Location dependent information implies the ability of the mobile device to quickly identify and adapt to its current location. In order to achieve this goal, the mobile terminal must acknowledge the amount of resources that are available within its current domain. This is not an easy task due to the high level of dynamism that is involved in mobile communications, which give rise to exorbitant search costs that fan the growth of the overall system load [FoZ94]. The impact of location dependent information on mobile computing further serves to reinforce the concept of adaptivity in the mobile paradigm. In order to successfully counter this characteristic designers must forge out a suitable balance between meeting the demands for this type of information, and in eking out every possible slice of bandwidth for user data.

According to Pitoura and Samaras [PiSa97], there is a still a need for the development of efficient data structures, algorithms, and query execution plans for representing, managing and querying the location of mobile elements.

2.2.3 Communication Heterogeneity

One of the definitive features of the mobile paradigm is the random relocation of the mobile terminal. Consequently, even when the mobile terminal expends significant communication resources in locating and establishing a connection with the nearest server, the validity of that connection over time becomes outdated. The act of simply entering an adjacent cell can result in a new communication pattern that can at times bear no relation to the actual physical movement that occurred. In such instances, the longer network path means that the signal must traverse more network intermediaries, which results in longer latencies that can lead to disconnection [AloK93]. These series of chain events also consume more network capacity, although the level of bandwidth that exists between the mobile terminal and the server may not suffer greatly.

The problem of communication heterogeneity is also evident in terms of the topology and morphology aspects of the current location of the mobile terminal. For instance, while outdoors a mobile device may have to rely on low bandwidth. In contrast, while indoors, the mobile device may have the opportunity to avail itself of ample bandwidth. The concerns that are raised by the diversity in the range of environments that a mobile device might have to content with again highlights the general issue of adaptivity in the mobile paradigm. Each step of the communication cycle calls for the mobile terminal to assess its situation based on multiple and diverse informational sources, and based on these inputs, formulate a work-around strategy.

2.3 PORTABILITY

The portability of the mobile device is another definitive feature of the mobile computing paradigm. In that, the design of mobile computing devices is founded on the concept that such devices should be small, light in weight, durable, and capable of long battery life [Foz94, Kat95]. Each of these critical limitations that result from portability will be examined. However, a pertinent consideration that must be borne in mind here is the degree of relativism that has been injected into the term portability due to recent technological advances. For instance, both laptop computers and the Nokia Communicator are termed mobile devices. However, in terms of resource levels, there are now few differences between a modern laptop and its desktop counterpart. This is not to suggest that laptops and handheld devices share nothing in common because in terms of the communication environment they both suffer from the same debilitations.

2.3.1 Power Consumption

The Nokia Communicator 9110 has a battery performance of 3 – 6 hours of talk/data time (NoKa99). However, heavy communication sessions will most likely place such users at the lower end of the performance scale. The power consumption of the dynamic components of a mobile terminal is proportional to *CV'F, where C* is the capacitance of the wires, V is the voltage swing, and F is the clock frequency [FoZ94]. This representation suggests that there are three ways to save on power consumption. Firstly, the capacitance can be reduced by developing more efficient chip technology. Secondly, the voltage can be reduced by redesigning chips to operate at lower voltages. Finally, the clock frequency can be reduced, whereby power savings can be realised at the expense of computational power.

Power can also be conserved by efficient operation of the device. In that, power management software can be utilised to power down individual components when they are idle, as for example, by running down the hard disk or turning off screen lighting when these components have been idle for some seconds. Moreover, Alonso and Korth have investigated the problem of power from the application perspective [AloK93]. In this vein, they have suggested that one of the critical tasks facing designers is to develop efficient query optimisation algorithms that will select plans based on their energy consumption patterns. In addition, it has been shown that cellular transmission typically requires between ten, and as much as thirty-six, times as much power as reception [Agr98, CSAK98]. Thus, a strategy of minimising the time the receiver is in an active mode can also significantly save power.

2.3.2 Risks to Data

Risks to data that is stored on mobile devices represents a very real concern for mobile users. In one occasion involving an acquaintance of the author, a car was reversed (accidentally) over a Nokia 9110 Communicator. In this case, there was no backup copy of the device's data, and as a result, several man-hours were consumed in a partially successful effort to recover the data from the broken device.

Consequently, as the above incident reveals, the act of making computers portable heightens the risk of physical damage. Furthermore, other risks attached to portability include unauthorised access, loss, and theft of the mobile terminal, which can lead to breaches of privacy or total loss of data. However, these risks can be reduced by minimising the essential data that is kept in the

permanent storage of the mobile terminal. In addition, in order to guard against unauthorised disclosure of information data stored on disks and removable memory cards can be encrypted [Foz94, Alok93]. Information loss can be minimised by designing systems that prompt users to take backup copies of local data on a regular basis. Systems such as the CODA file system solve data security problems by enabling all newly produced data to be immediately copied to a secure remote server [MES95]. This type of solution is taken to the extreme by the network computer concept whereby there is no need for the mobile user to be concerned with data integrity issues, since all data is stored on the Net.

2.3.3 Limited Storage Capacity

There are several other challenges that are involved in the mobile computing paradigm, the solutions for which must be embedded in the design of mobile systems. One such challenge is the limited storage capacity that is available to mobile devices. However, the critical question here is whether storage capacity is of any consequence to mobile devices given the market trends towards the network computer, and visibly towards smaller and lighter devices. According to a recent survey the weight of a modern mobile terminal averaged 368 grams, and the dimensions were 171 x 88 x 29 mm [Kon98]. These statistics carry with them powerful implications for data management issues relating to applications that are designed for such devices. For instance, given the lack of storage capacity, it is apparent that the applications must be loaded onto the device from the server, and also that all data should reside on the server.

2.3.4 Miniature Devices

The size constraints of portable devices require that these devices are equipped with a small user interface in terms of the display, as well as the input devices. Desktop windowing environments may be easily accommodated by modern laptop computers, which are currently equipped with displays that are comparable with desktop computers in terms of resolution and the number of colours supported. However, for smaller devices, such as hybrid communicators, viewing information on the screen still presents significant difficulties, even though the Nokia has improved the lighting and increased the size of the screen for the 9110 Communicator

In addition, hybrid devices also encounter problems with user interaction with the input devices. Most often a fine balance has to be made between the small working area that is available, and maintaining a healthy degree user friendliness. For instance, on the 9110 Communicator there are individual keys to represent each of the applications that this terminal supports, such as Internet,

SMS, Fax, and Contacts, among others. This policy has the advantage that it is highly user friendly, and requires minimal time for a new user to become familiar with the terminal's operation.

However, there are two major disadvantages that result from the policy of having individual keys for each application. In the first instance, even though the Web is rapidly assuming the role of the omnipotent communication medium, the entire domain of telecommunications, and the services that are being devised by operators, are evolving at an even more rapid pace. In this vein, the policy of embedding applications and services in the terminal's physical interface restricts the freedom of both users in subscribing to new services, and operators in developing new and challenging products. The second disadvantage relates to interface sub-optimisation. In that, given the already minuscule size of the keypad and screen, the strategy of allocating a separate key on an application basis exacerbates the problem.

2.4 Overview

The preceding discussion provided an overview of the challenges that are endemic of mobile computing. The remainder of this thesis will focus on how solutions to those challenges are reflected in the design of mobile systems. To this extent the tone will assume a more practical outlook as reference will be made to a number of currently existing architectures, the purpose of which, is to afford the reader an opportunity to grasp a practical view of the issues that are being discussed. The specific architectures that will serve this purpose are WAP (Wireless Application Protocol), In addition, the JMAM (Jyväskylä Mobile Architecture for Multimedia Data) architecture, which is one of the primary aims of the thesis, will enter onto the stage for the first time.

In the next chapter, a number of system architectural models that support mobility and wireless computing will be discussed. These models have been the generic building blocks of the mobile architectures that have appeared thus far. Furthermore, the generic nature of these computational models has given them a high level of flexibility that permits these models to remain the building blocks of emerging software architectures.

3 MOBILE SYSTEM ARCHITECTURES

A system architecture represents the underlying data communication paradigm upon which any application is built. In mobile computing, the system architecture that is employed in developing applications must cope with the special characteristics and limitations of the wireless environment, while simultaneously providing efficient access to both existing and new applications.

In the previous chapter where the environmental challenges facing mobile computing were discussed, it was emphasised that one of the chief goals of mobile systems is the attainment of a high degree of adaptivity. This requirement for adaptivity arises primarily because the inherent dynamism of the mobile environment, which is induced by the communication autonomous (c-autonomous) nature of mobile communications. A computer system is c-autonomous within the environment if it cannot be compelled by external systems to start a communication session, continuing that session, or be prohibited from initiating communication with an external system. There are two basic types of c-autonomous interaction patterns that are distinguished by Veijalainen [Vei90].

In the first instance, potential c-autonomy is an inherent element of mobile computing since the mobile device is free to initiate a communication session with another system. Furthermore, the mobile device is freely able to decide whether to continue that session or to end it. The decision to exercise this c-autonomous prerogative and prematurely end a session may arise from the user, or from external factors such as loss of the radio signal, or because of cell blocking. In this latter instance, potential c-autonomy would become effective c-autonomy, since the other system would still be interested in continuing the session [Vei90].

Accordingly, the second form of c-autonomy - effective c-autonomy - refers to the total or practical inability of asynchronous communication between the two systems due to incompatible availability criteria. What this definition implies, is that c-autonomy becomes effective in relation to an external system when the mobile device tries to initiate a communication session, or to continue an active session, at a time which is not suitable for the external system. Likewise, the mobile device can refuse start, or end a session when the external system is interested in communicating with the mobile device. However, the effect of effective c-autonomy can be eliminated by situating a non-c-autonomous third party between the two entities that will act as a communication buffer [Vei90].

Consequently, a critical feature of mobile systems is their aptitude in capturing and implementing adaptivity. The system architecture must be able to support c-autonomy to the extent that messages that are intended for a mobile terminal, which were sent during the time when that terminal was not connected to the network, are stored. These messages must stored by the network so that they are recoverable by the terminable when reconnection to the network is subsequently re-established. This system may be referred to as one-way c-autonomy, as opposed to two-way c-autonomy whereby the network can totally disregard the mobile terminal when there is no current connection [Vie99]. The critical challenge that is involved in achieving one-way c-autonomy is in providing the network with the ability to distinguish c-autonomous disconnection as normal, as opposed to an network error situation that requires continuos attempts at retransmission.

Adaptivity and c-autonomy are thus critical elements that must be embedded in a mobile communication system at every level. This is with regard to the underlying software architecture paradigm associated with the mobile application. For instance, adaptivity can be achieved by redefining the partition of duties between the client and the server. Thus, during periods of disconnection, the mobile client may be permitted to work autonomously, whereas during periods of strong connectivity, the client may depend more heavily on the fixed network [SaP98].

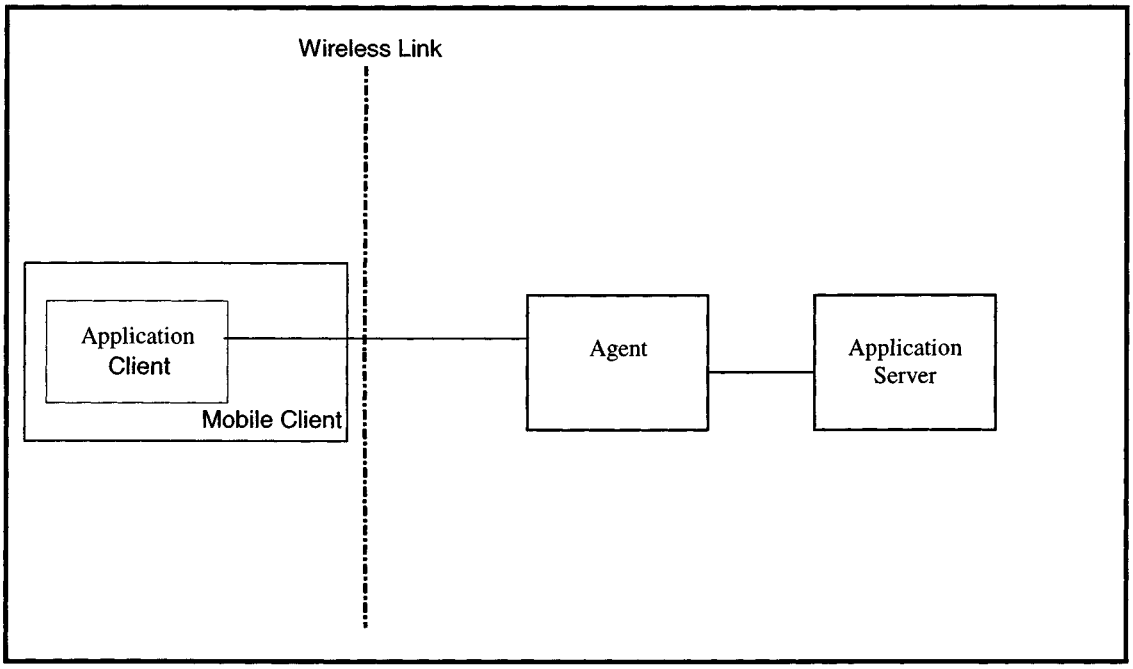
From the above discussion, it is evident that there are a myriad of ways in which system architectural models influence the operation of mobile computing systems. Moreover, the choice of one system architecture over another is fundamentally influenced by the characteristics of the mobile device, the limitations of the wireless medium, and thirdly, by the challenges that are posed by mobility. This remainder of this chapter will be focused on presenting the two system architectural models and their variations. These two models are referred to as the mobile client-server model, and the mobile-agent model.

3.1 MOBILE CLIENT-SERVER MODEL

The mobile client-server architecture has its roots in the well-known client-server paradigm that was developed for fixed computing systems. However, given the diversity of the computing environment between fixed and mobile systems, the client-server model has undergone a significant transformation in order to conform to the rigors of the mobile domain. The mobile implementation of the client-server architecture is referred to as the client-agent-server model [SaP98] to reflect the introduction of an agent between the mobile client and the fixed server. The purpose of this additional mediator into the paradigm is to provide relief for the resource stricken clients by accepting part of the computational burden.

As Figure 3.1 depicts, the client-agent-server architecture displays a horizontal orientation. In principle, the mobile terminal is permitted to assume the role of server, and vice versa. However, for practical reasons pertaining to resource constraints, the role of the mobile terminal is normally restricted to that of client.

FIGURE 3.1: CLIENT-AGENT-SERVER ARCHITECTURE



The role of the agent in the client-agent-server architecture is the critical factor that prevents the harmful effects of c-autonomous communication between the mobile and the fixed network. The agent stores and forwards messages that are exchanged between the two entities, and introduces

asymmetry in the interaction pattern. In that, the server is given most of the computing and communication burden.

The client-agent-server architecture also provides facilities for dealing with the limited battery life of the mobile client. Indeed, the same strategy used to preserve the client's scarce battery resources is useful in countering weak connectivity. Weak connectivity can be described as a degraded form of service that results from slow or over-burdened networks. In such networks, connection can be lost for short periods of time [PiSa97]. The client-agent-server architecture is able to counter these erroneous situations by optimising the asymmetric burden of traffic that occurs between the client and the server. In addition, the power consumption pattern of generating traffic as opposed to passively receiving transmissions also provides opportunity for minimising energy consumption.

Thus, the client can submit a request to the agent, after which it enters a doze mode, and is reawakened when the agent is able to deliver the response. In terms of weak connectivity, the agent can employ optimisation techniques that minimise the size of the data to be transmitted to the client depending on the type of data and on the specific application. The agent can also manipulate the data before transmission to the client by altering the transmission order so that the most important data is transferred first. In addition, the agent can perform data specific compression that tailors the content to the specific constraints of the client, or lump together multiple replies [PRWS98].

The characteristics of the client-agent-server architecture make it an appropriate computational model for lightweight clients, such as hybrid communicators and PDAs. This is because the burden of computational responsibility is shifted from the mobile client onto the shoulders of the agent, which can be likened to a workhorse. Architectural models such as WAP [WAP99a] are based on the client-agent-server architectural model.

A powerful functionality of the agent methodology is the extensibility of the roles that agents can be programmed to play. In that, a mobile terminal can have the functionality of several diverse clients. This means that there will be as many agents as there are applications associated with the particular terminal. For instance, an agent may be designed to provide a unique service such as web browsing [HSL98], or for database access [Ora99, Ara99]. Any traffic that is associated with this application must be communicated through the service-specific agent. This case can be symmetrically

generalised by having a service-specific agent servicing multiple clients, as is the position in WebExpress [HSL98].

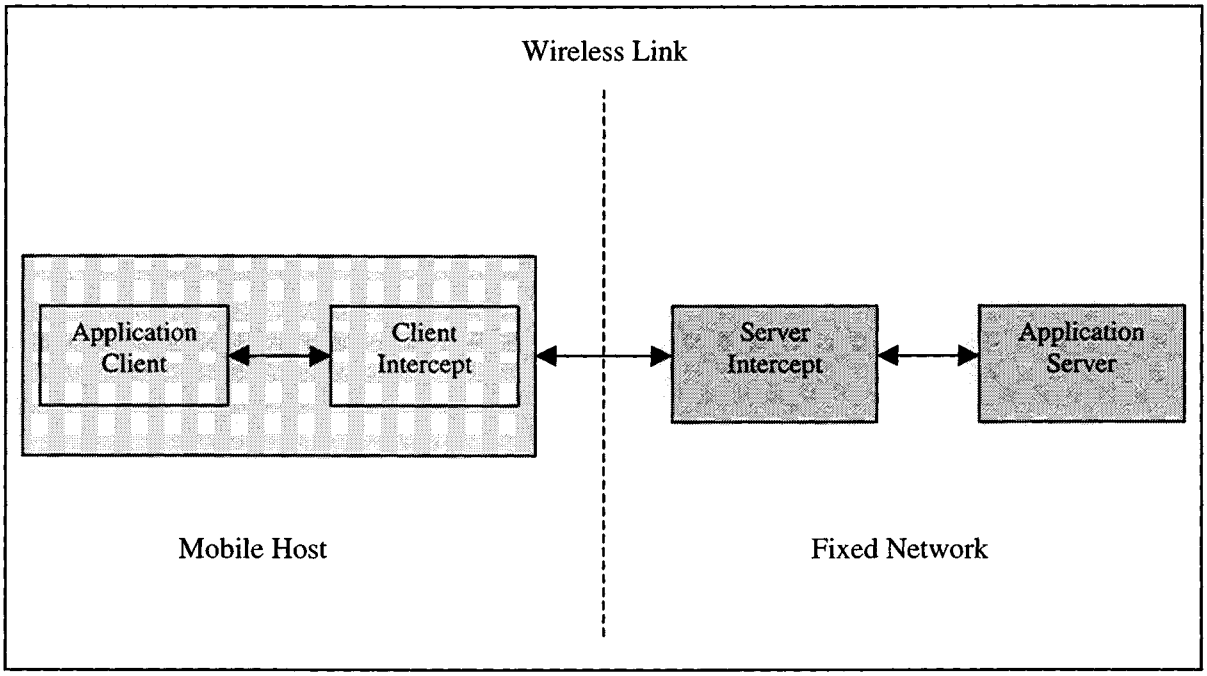
The role of agents can become more complex when a service is associated with multiple agents, and this group of service-specific agents serves multiple clients. In this case, any client request associated with the service can be serviced by any available agent in the group. This flexibility, however, entails the need of a more complex management function. A service-specific meta-agent is required to co-ordinate clients' requests and the service-specific group of agents. Thus, the meta-agent can be viewed as a service-specific agent serving multiple clients with the other agents playing only a supporting role. In Oracle, the role of the meta-agent is played by an agent management facility called the Message Gateway [Ora99], and in the WAP architecture, this role is performed by the Gateway [WAP99a].

Despite the benefits of the client-agent-server model in minimising the debilitating impact of many of the wireless challenges, the model is still fallible in terms of disconnections. In that, the model is unable to sustain computational activity on the client side when the client is disconnected from the network, although the server may continue operation. To enable the client to deal with disconnection transparently requires additional complexity to the lightweight client application, which given the client's memory and processing power limitations, might not be reasonable. Furthermore, the agent can only optimise data transmission over the wireless link from the fixed network to the mobile client, and not in the opposite direction [SaP98, HSL99].

3.1.1 Client-Intercept-Server Architecture

The client-agent-server architecture is a highly extensible architecture model, and a number of developments have made in order to extend this architecture to allow it to adapt in instances where the client becomes disconnected. To this end, Samaras and Pitoura, as well as Housel et al, have proposed the deployment of a client-side agent on the mobile device. The purpose of this client is to operate in conjunction with the agent on the wireline network. This model has been termed the client-intercept-server model, and its operation is depicted in Figure 3.2.

FIGURE 3.2: CLIENT-INTERCEPT-SERVER ARCHITECTURE



The client-intercept-server architecture is also founded on the asymmetric communication pattern between the mobile terminal and the server. To this extent, the mediating agents in this architecture also function to protect the fixed network from the c-autonomous character of the mobile terminal. The client-side agent intercepts client's request and together with the server-side agent performs optimisations to reduce data transmission over the wireless link, improve data availability and sustain uninterrupted computing at the mobile terminal. In this architecture, the client side intercept is perceived as the local server proxy that is co-resident with the client. Similarly, the server-side intercept appears as the local client proxy that resides on the fixed network with the server.

The client is able to submit an encapsulated request through the client-side agent to the server. After this request is submitted the client can enter a passive mode, and wait for the server to indicate when the task is completed. Furthermore, as in the case of the client, the server-side agent can also encapsulate the response before it is sent on to the client-side agent, which decompresses the response before it is viewable by the client. [PiSa97, HSL97].

The client-intercept-server model provides transparent operation between the client and server, which infers that this dual agent approach can be extended across the full range of applications. Moreover, there are several benefits, which this model brings to the operation of the agent paradigm. Firstly, the communication protocol that links the two agents can be adapted to provide

effective data reduction and protocol optimisation without limiting the functionality or interoperability of the client. Secondly, the co-operation of the two agents allows for more efficient optimisations of the wireless link for the benefit of different applications. Thirdly, application specific optimisations can be more effectively realised by the agent pair. Finally, the existence of two agents, as opposed to one, means that a higher degree of adaptivity can be achieved, since the two agents can divide the workload based on environmental conditions.

The model offers flexibility in handling disconnections. For instance, a cache may be maintained at the client-side to allow it to work autonomously. The cache can be used to satisfy the client's requirements for data during disconnection. Items that are not cached may be queued at the client-side agent to be served to the server agent upon reconnection. Similarly, requests to the client can be queued at the server-side agent and transferred to the client upon reconnection. Weak connectivity can also be handled in a variety of ways. For example, relocating computation from the client-side agent to the server-side agent or vice versa can minimise the effect of weak connectivity, and background prefetching to the client-side agent can reduce communication during weak connectivity [HSL98].

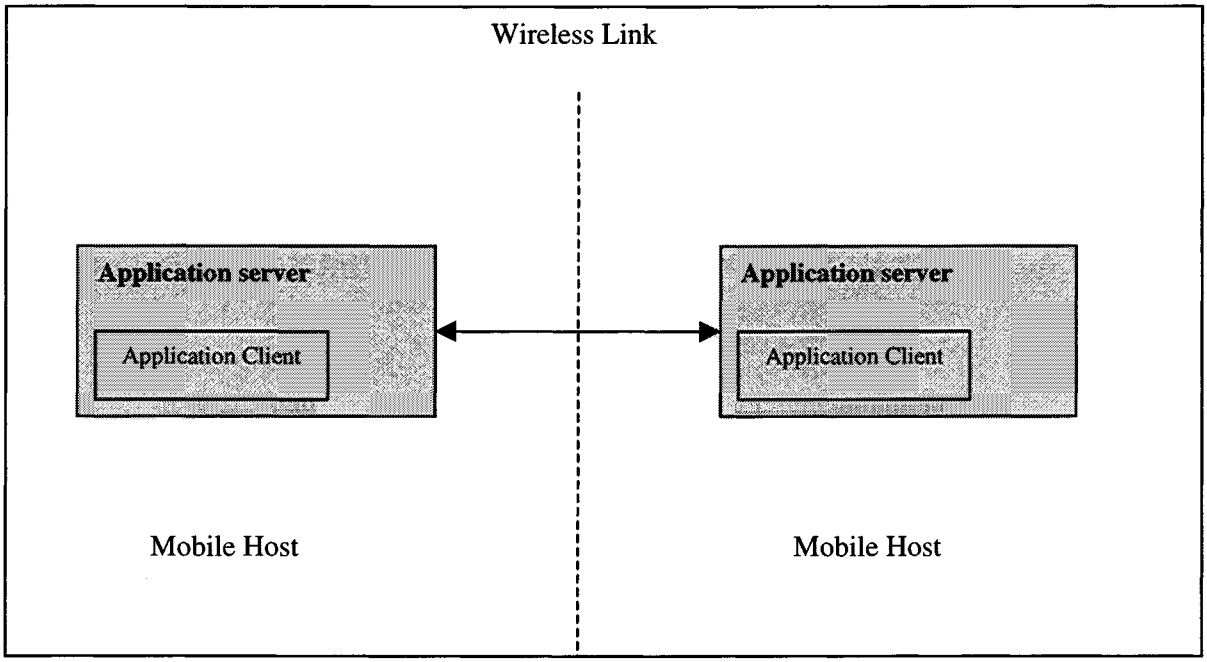
The intercept model provides high support for upward compatibility so that legacy and existing applications can be executed without alteration. This is because the agent pair shields them from the limitations of mobility and the wireless media. However, a weakness of this model is that every application requires development work at both the server and the client sides. A mitigating factor here is that there is no need to develop a pair of agents for every instance of an application. Instead, since the functionality and optimisations performed by the agent pair is generic enough, it is only required to develop a different pair of agents per application type, for example, file, database, or web application categories [SaP97, PiSa97].

The idea of employing proxy pairs has been gaining some popularity of late. Such an approach is employed by WebExpress [HSL98], which is an IBM system for optimising web browsing for routine and repetitive commercial applications in a wireless environment. In this system, the client-side agent is called client-side intercept (CSI), while the server-side agent is called server-side intercept (SSI).

3.1.2 Peer-to-Peer Architecture

The peer-to-peer architecture is another variation of the horizontally oriented client-server family of distributed architectures. In a peer-to-peer distributed architecture, there is no distinction between clients and servers. Theoretically, each site has the full functionality of both a client and a server, so that mobile hosts have are made equal partners in the distributed processing. Consequently, this architecture is only applicable to heavyweight devices, such as laptop computers, which are endowed with a significant level of computing resources. Figure 3.3 provides a representation of this architecture.

FIGURE 3.3: PEER-TO-PEER ARCHITECTURE



In contrast to the previously discussed system architectures, which aim for an asymmetrical communication pattern between the mobile terminal and the server, the communication pattern of the peer-to-peer architecture is entirely symmetrical. Thus, the types of applications that are suited to the peer-to-peer model are high resource consuming applications, such as DocMan [BuB96], which demands at least 8 MB of RAM memory, and either a Windows NT or Windows 3.1 operating system. DocMan is distributed document management system that allows users to make simultaneous updates to documents. When these updates are made, they are visible to each participant since there is a direct connection existing between the two (or more) interacting parties. In a pure client-server system, this opportunity to trade updates online would not be highly wasteful, since all interactions would need to be funnelled through the remote server.

3.2 MOBILE AGENT ARCHITECTURE

The mobile agent architecture represents an alternative system architecture. It is a highly versatile and powerful computational paradigm, in which the functionality of the agent can be programmed to suit the particular needs of the application, as well as the needs of the client. The mobile agent concept is an extension of the Remote Procedure Call communication technique, whereby the client sends an encapsulated message or procedure to a server for execution [PiSa97, CGH95]. After its submission, the mobile agent proceeds autonomously and independently of the sending client. Consequently, a mobile agent is armed with its own protocol stack, authentication information, and data. Thus, in contrast to the horizontal structure of the client-server architecture, a mobile agent is a vertically structured, self-contained, executable object [Vie99].

The mobile agent paradigm is environmentally and architecturally fashioned in many regards such that agents complement the nature of mobile and distributed computing. The domain of network computing, and especially the WWW environment, is most often a highly heterogeneous environment that contains a wide assortment of hardware and software systems. To function effectively, and to be able to provide powerful benefits in such diverse environments, a primary feature of mobile agents is their ability to provide a host independent execution environment [CGH95].

This host independence characteristic serves as starting point on which the concept of openness in the agent paradigm is built. The agent itself is a program that is divided into three parts, which may be defined as the agent passport, the table of contents, and the executable contents. This generic structure must, however, be reinforced by a system for creating and maintaining vocabularies. One of the major benefits of agents is that they allow domain specific knowledge to be applied to problems. To this end, systems such as the Knowledge Query and Manipulation Language (KQML), and the Knowledge Sharing Effort approaches can be used effectively to support named vocabularies. These knowledge sharing systems qualify vocabulary terms, and help to reduce confusion when agents are interacting with other entities in multiple vocabularies [CGH95, KiZi97].

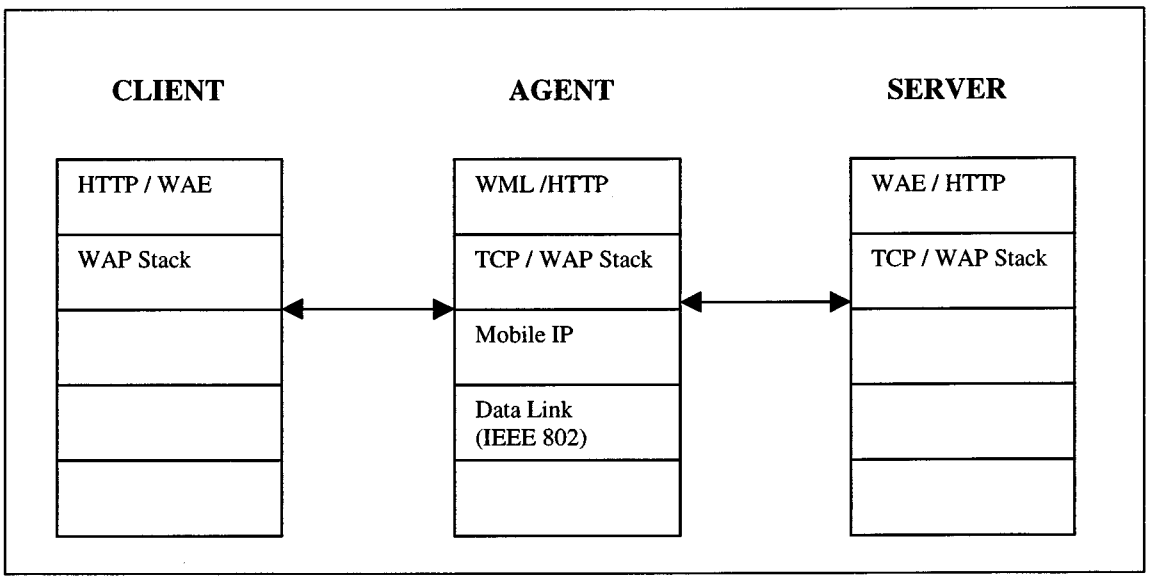
In view of the autonomy that an agent possesses to roam among foreign systems and networks in pursuant of its task, the need to embed adequate security mechanisms to ensure privacy of

information and prevention of viruses becomes more heightened. To this end, it is vital that security mechanisms such as digital signatures and trusted hardware are employed so that agents can be adequately encapsulated and protected during their execution, and screened before acceptance.

In addition, the ability to develop mobile agents that can be loaded onto lightweight devices points to the need for a lightweight language in the mobile agent paradigm that is suited for expressing a summarised version of the agent's tasks. Furthermore, once an agent has been delivered to the network this summary of tasks has to be expanded and executed in a semantically and intellectually richer environment. In recent years there have emerged a range of object based script languages that are capable of fulfilling this role, such as COBRA, ILU, and Java [JaKe98].

Figure 3.4 provides an overview of the vertical and self-contained structure of a mobile agent. However, the diagram also reveals that fundamentally, the agent architecture must rely on some horizontal platform for both its initialisation and conclusion when the results are filtered back to the originating client [Vei99]. In this schema, the client launches the agent into the network, after which its proceeds independently to make connections to the necessary data sources in the network. Thus, the agent architecture also introduces an asymmetric interaction pattern between the mobile client and the fixed network.

FIGURE 3.4: AGENT ARCHITECTURE [VEI99]



In the process of performing the specified task, the mobile agent has the autonomy to roam among foreign servers to gather information, parent new child agents, and interact with other agents. Upon

completion of the specified task, the mobile agent delivers the results to the sending client or to another server. There has been significant interest of late in the use of mobile agents. One of the chief reasons for this occurrence is the fact that the mobile agent architecture can be successfully and beneficially married with the mobile client-agent-server architecture, and its variations where there is need for an agent. For instance, [PSP99] has recently advanced the approach of employing Java programmable mobile agents, referred to as Aglets, between a client and server to provide database connectivity, processing and communication over the Internet. In both of these instances, hybrid architectures that include elements of horizontal and vertical system architectures are used effectively [TDB97].

There are two principal reasons behind the popularity of the mobile agent architecture. In the first instance, mobile agents provide an efficient, asynchronous method for searching for information or services in rapidly evolving networks, since they are launched into the unstructured network to roam around to gather information. Secondly, mobile agents support intermittent connectivity, slow networks, and lightweight devices. This second property makes the use of mobile agents in mobile computing very attractive [CGH95, PiSa97, PSP99], since they fit in well with the general trend towards smaller and lighter devices.

According to [PSP99, PiSa97], an important functionality of agents is their ability to be programmed with intelligence. However, in addition to intelligence, there are a number of other critical features, which heighten the value of mobile agents. These features may be stated as:

- the ability of an agent to interact and co-operate with other agents
- agent autonomy, in the sense that agents' execution proceeds with little or no intervention with the sending client
- agent interoperability, that is, agents can be executed in diverse platforms
- agent reactiveness, the ability of an agent to respond to external events
- agent mobility, which refers to the ability to roam among a set of networked servers

Consequently, there are a myriad of factors that render the mobile agent paradigm highly effective in dealing with several important mobile computing challenges. In the first instance, the agent architecture supports disconnected operation. Thus, a mobile client can launch an agent while connected to the network, and then disconnect for whatever reason. During this period of

disconnection, the agent proceeds independently to accomplish the delegated task, and when the task is completed, the agent waits until reconnection to submit the result to the mobile client. In addition, weak connectivity, and reduced energy consumption, is supported by the model since the overall communication traffic through the wireless link is reduced. In that, depending on the size of the agent, it can be dispatched from the terminal, or from the network if its size would incur high communication costs to move it from the terminal to the network. Conversely, only the result is delivered from the network back to the client. Furthermore, by letting mobile hosts submit agents, the burden of computation is shifted from the resource-poor mobile hosts to the fixed network.

The development of Java has provided a significant boost to the use of mobile agents over the past few years. This is mainly because Java provides a powerful environment that enhances the autonomy and portability of agents. Java is built on a computational model that allows a program to be seamlessly distributed among a collection of heterogeneous processors by allowing concurrent thread of control to execute on top of a virtual portable, distributed virtual machine. In addition, Java includes adequate security mechanisms that can guard against attacks on roaming agents by other malicious agents [TDB97, JaKe98,PSP99, BPS99].

3.3 OVERVIEW OF SYSTEM ARCHITECTURES

Table 3.1 [PiSa97] provides a rich and insightful synopsis of the strengths and weaknesses of the mobile system architectures that have been presented.

TABLE 3.1: STRENGTHS AND WEAKNESSES OF MOBILE SYSTEM ARCHITECTURES [PISA97]

	Client-Agent	**Client-Intercept**	**Peer to Peer**	**Mobile Agents**
Disconnection	Partly	YES	Partly	Partly
Weak Connectivity	Partly	YES	Partly	Partly
Adaptivity	Medium	HIGH	NO	Medium to High
Type of Client	Light	Heavy	Heavy	Light
Functionality divisions	High	Low-High	Low-High	Low-High
Mobility	Not aware	Aware–not aware	Aware-not aware	Inherent
Upward compatibility	Server side only	High	High	Server side only

The functionality division heading refers to a model's ability to repartition the computational duties between the client and server. The client-agent architecture provides a high degree relief for the mobile client in off-loading the majority of the processing to the server. In contrast, the other architectures provide a flexible scale in defining the balance of processing. The upward compatibility heading refers to the ability to accommodate various types of application models that are based on old as well as emerging standards.

As was noted earlier, it is possible and highly beneficial in some instances to merge the client-agent architecture with the mobile agent architecture to build hybrid systems. Indeed, the agent entity as is implemented in the client-agent model can be viewed as a type of fixed agent. In addition, as Figure 3.4 depicts, the agent model relies on some form of horizontal architecture for its implementation. In light of these observations, the principle aim is to develop a suitable architecture that conforms to the stated goals of this thesis. The first of these goals is to provide for the processing of multimedia data over the wireless medium. The second objective is to provide hardware and service developers with a flexible and efficient methodology whereby the physical interface of the terminal can be separated from the services provided to a particular user.

In proceeding towards the attainment of the outlined goals, Table 3.2 provides an indication of the strengths and weaknesses of the hybrid architectures. From this tabular view, the architecture that combines the client-intercept and the mobile agent system architectures appears very appealing.

TABLE 3.2: STRENGTHS AND WEAKNESSES OF HYBRID SYSTEM ARCHITECTURES [PISA97]

	Client-Agent Mobile agents	**Client-Intercept Mobile agents**	**Peer to Peer Mobile agents**
Disconnection	Partly	YES	Partly
Weak Connectivity	Partly	YES	Partly
Adaptivity	Medium	HIGH	NO
Type of Client	Light	Heavy	Heavy
Functionality divisions	High	Low-High	Low-High
Mobility	Inherent	Inherent	Inherent
Upward compatibility	Server side only	High	High

However, there is a critical deficiency in the Client-Intercept hybrid architecture that severely challenges its applicability to hybrid communicator devices. In that, this hybrid supports heavy weight devices due to the requirement that a client-side intercept for every application class must be developed, and be permanently resident at the client. The use of the client-Intercept hybrid also has an impact on computational activity at the client, since the client-side intercept must share a significant part of the computational burden.

Consequently, the most appropriate architecture that is suited to achieving the outlined aims of this thesis is the client-agent-server architecture. In the first instance, a profile manager, which is accessible by the agent, can be implemented at the client. This strategy can be used effectively to detach the terminal's physical interface from the applications and services that are available to the terminal. This profile would contain the user's access rights to the services that are provided by an operator. Moreover, this profile can be verified during the terminal's authentication procedure, and it can be dynamically and seamlessly updated from a central service center location to reflect changes to the user's subscribed services at any point in time.

A deeper discussion of the hybrid client-agent-server model as it pertains to the problems posed by this thesis will be undertaken in the following chapters. In the next chapter, the Wireless Application Protocol (WAP) architecture, and Bluetooth technology will be discussed. These are two of the most powerful innovations in the mobile computing domain regarding the type of devices that are the focus of this thesis.

4 MOBILE COMPUTING MODELS

The increasing popularity, or indeed necessity, for high-speed access to the Internet, intranets, and other IP delivered services is the force that is driving the need for ubiquitous connectivity. In recent years, the explosive growth of the Internet has massaged and nurtured in an increasing number of users, a hunger for information. This hunger is applicable both at the individual level, as well as at the corporate level, where corporations are now deeming access to internal and external information sources, a matter of competitive advantage [Earl89, Laud95]. Moreover, according to a recent Nokia publication, in 1998 73% of European corporations were using some manner of mobile data solution, and 91% of those who were not using mobile data solutions, indicated that they would start in 1999. In addition, 60% of mobile data users in the United States said that they would rather use a mobile phone for mobile data rather than a computer [WAP99a].

The emergence of WAP has been the single most pioneering development in the mobile world that has contributed towards shortening the existing gap between providing Internet access and other value-added-services to small devices, and the facilitating communication architecture. The significance of WAP is that it provides a common standard for the way in which information is delivered over wireless networks to resource deficient devices, such as hybrid communicator devices. Furthermore, a second development that will complement WAP, and generate a positive impact on wireless connectivity and computing is Bluetooth technology. Bluetooth technology will facilitate the seamless and invisible communication between mobile devices, which are both equipped with the Bluetooth chip, and the supporting software.

In the following chapter, the significance of WAP and Bluetooth to mobile architecture issues will be discussed in detail.

4.1 WAP (WIRELESS APPLICATION PROTOCOL)

WAP is the result of a strategic alliance that was founded by a number of major players in the telecommunications industry, including Nokia, Motorola, Ericsson and Phone Com. However, WAP technology has grown in popularity to the extent that currently the membership includes terminal and infrastructure manufacturers, operators, carriers, service providers, software houses and content providers [Dis99b]. Recently, the Microsoft Corporation has also joined the WAP

Forum. The most likely reason for this development would appear to be that Microsoft has failed in its bid to develop to a competing technology that would be strong enough to dominate the market.

The origins and grounding philosophy of WAP are rooted in the sweeping changes that have occurred in the last decade in people's lifestyles in terms of how they use and access information. The Internet has been a primary player in expanding the notion of information usage out of the confines of the corporate sphere, and in bringing it into the homes and lives of every individual who desires information. Thus, the world is now an information society, in which access to information is demanded by people both in their professional lives, as well as in their private lives, and in many instances, these two spheres are difficult to distinguish. The result can be described as a kind of information personalisation whereby the user is the primary player, and information, be it the user's bank balance, or inventory levels, is indistinguishable and subsidiary in terms of the computer hardware or the manner of access.

As the general perception of information evolves, so too must the perceptions of the implementers and providers of technology. Consequently, as information becomes personalised, so too must the devices used to access this information. There are four basic qualities that these new devices must possess in order to serve the needs of the new information society. Firstly, the device must be personal, in the sense of being under the control of the user to choose when, where and how the device is used. Secondly, the device must be mobile to facilitate anywhere, anytime communication, while being able to fit into the user's pocket. Thirdly, the mobile device must be trusted, and fourthly it must be usable. Usable in this sense currently means that the device must be able to support text images and voice [WAP99a, Dis99b]. For instance, the new Nokia 7110 media phone has been designed to enable easy access to Internet content. The 7110 phone has a large graphics display, as well as a number of new and innovative features for ease of operation and text input [Dis99e]. However, it must be noted that in order for the 7110 to become a truly powerful facilitator of the information society, the display would require some additional enlargement.

4.1.1 Motivation for WAP

It is against the above outlined backdrop that WAP comes to the forefront of the stage. Standing juxtaposed to each other are the miniature terminals with their peculiar characteristics, the constraints of the wireless communication medium, and the Internet, with WAP planted stoutly in the center performing the ceremonial rites. WAP is fully competent to support this role since the

WAP specification is compatible with existing Internet standards, and is based on a set of standard communication protocols that have been optimised for the unique characteristics of the wireless environment. WAP is a communications and application development environment that can be built on top of any OS. The primary advantage of WAP is that it was conceived with the aim of confronting, and adapting to, the constraints of the wireless environment in terms of compression of data, long latency, and limited bandwidth. In addition, WAP also focuses on the limitations which typify mobile devices, such as less powerful CPUs, less memory capacity, restricted power consumption, small displays, and different input devices [Dis99b, WAP98, WAE98].

A crucial element in the motivation for the development of WAP also included the deficiencies of the traditional web model. These deficiencies were centered partly on the fact that the traditional web browsers are too heavy for small mobile devices, which are equipped with small graphic displays, and limited data input functionality. The problem in this regard is that HTML browsers are not efficient in separating content and presentation functions. To solve this problem, the WAP specification defines a microbrowser as part of the Wireless Application Environment (WAE), which is a thin client that operates with a small memory allocation [WAE98]. When combined with the use of proxy technology and compression in the network interface, this results in significantly reduced processing and power consumption at the mobile device.

In addition to the deficiencies of the traditional browsers for small mobile devices, problems with the traditional web model also arise in the area of communications traffic. In that, the TCP request-response transaction, as well as the HTTP header information both contain high quantities of redundant header information, which is not appropriate for the wireless environment. Furthermore, the TCP/IP stack is not appropriate for voice traffic, since there is no end-to-end time service guarantee. Table 4.1 shows the results of a comparison test between information transmitted from a desktop browser running HTTP 1.0, and from a WAP browser

TABLE 4.1 – TYPICAL COMMUNICATION SESSION – 3 REQUESTS, 3 RESPONSES [WAP99]

HTTP/TCP/IP	WSP/WTP/UDP
17 packets	7 packets
65% overhead	14% overhead

Figure 4.1 highlights the amount of bandwidth that can be saved in the WAP environment through use of compression techniques in the WAP stack. The WAP protocol uses less than half the number of packets that the standard HTTP/TCP/IP stack uses to deliver the same content. This improvement is essential in order to maximise use of the limited wireless bandwidth [WAP99].

The global objectives of the WAP Forum, as articulated in their published documents are as follows [WAP99]:

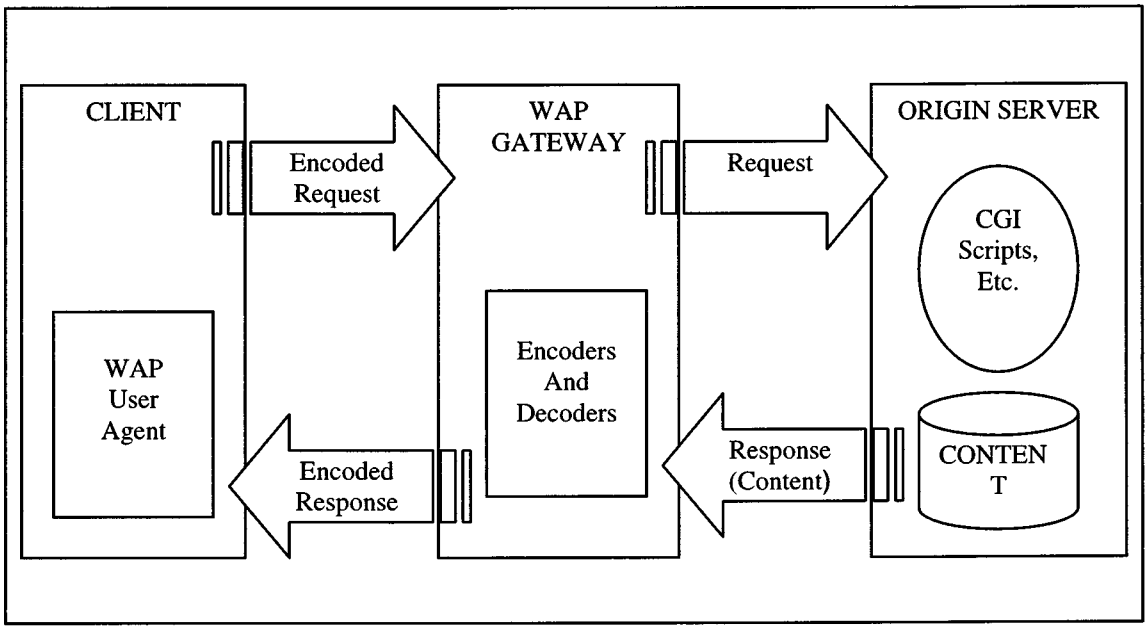
- to bring Internet content and advanced data services to digital cellular phones and other wireless terminals.
- to create a global wireless protocol specification that will work across differing wireless network technologies.
- to enable the creation of content and applications that scale across a very wide range of bearer networks and device types.
- to embrace and extend existing standards and technology wherever possible.

The following section will deal with how these global objectives are borne out in the WAP architecture model.

4.2 WAP ARCHITECTURE

The WAP architectural model is based on the horizontally structured client-agent-server computational model. Figure 4.1 provides a graphical representation of the WAP system elements and their interaction.

FIGURE 4.1 – WAP PROGRAMMING MODEL



As Figure 4.1 depicts, the mobile client submits all its requests to the WAP gateway, which modifies the requests before passing them on to the origin server for execution. The server's response is compressed in a similar manner before it is forwarded on to the client. In this way, the WAP computational model offloads all of the intensive processing activities from the resource deficient clients to the WAP gateway on the fixed network. To this end, the WAP gateway has two central roles, which may be described as [WAP98]:

- A protocol gateway – In fulfilling the role of the protocol gateway, the WAP proxy translates requests from the WAP protocol stack to the WWW protocol stack, and vice versa.
- Content encoding and decoding – In this role the proxy compresses WAP content into a format that reduces that the size of data that must be transferred over the wireless interface. When the messages are on the fixed network, they must be decompressed into a semantically richer format for execution.

Figure 4.1 is based on the minimal three-node WAP configuration consisting of the WAP client, WAP proxy, and Web server. Furthermore, in Figure 4.1, if the Web server contains WAP content, this content can be retrieved directly by the WAP proxy. However, if the web server provides HTML content then a filter is used to translate the HTML content into WAP content. It is also possible to create an origin server that includes the WAP proxy functionality. This type of WAP configuration can be used to facilitate end-to-end security solutions, or applications that require

better access control, or a guaranteed response time, such as those applications that involve the processing of continuous data.

The picture that emerges of the WAP architecture is that it is a robust and flexible computational model that can be developed and configured by an operator to provide a wide range of services to mobile users. Moreover, the WAP architecture is highly extensible in terms of its ability to accommodate forthcoming 3rd generation technologies without the need for extensive maintenance. In that, once applications are developed within the WAP environment, any future introduction of 3rd generation standards will merely increase the network transmission capacity without affecting the overlying applications. This robust and flexible character results from the fact that the WAP architecture has incorporated a set of standard components that facilitate communication between mobile terminals and network servers. These standard components include the following [WAP99a]:

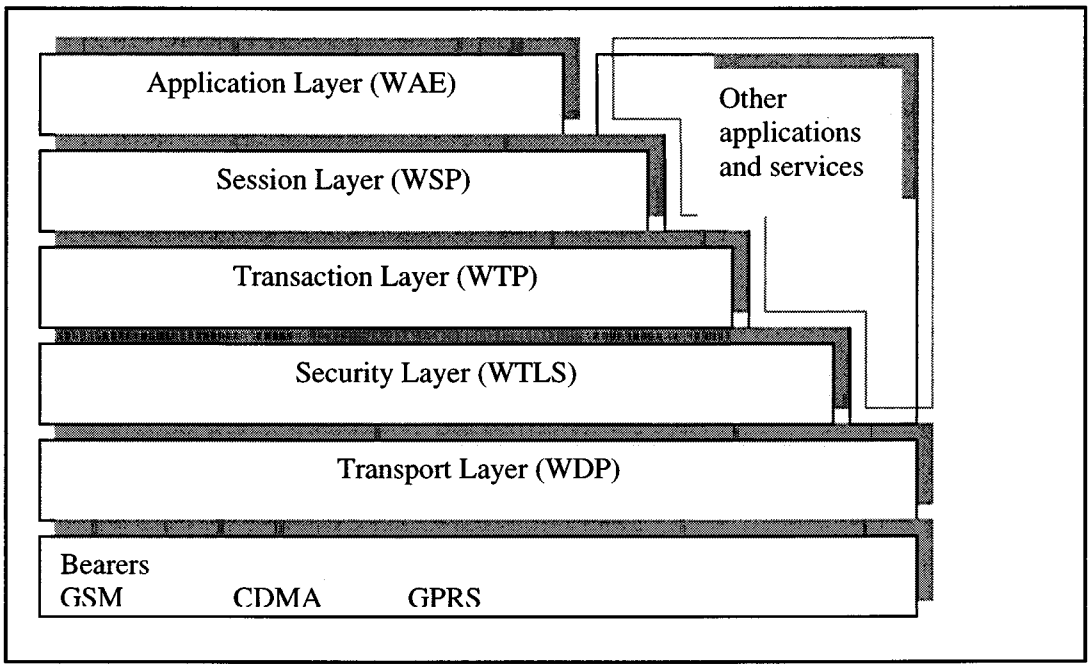
- A standard naming model – WWW-standard URLs are used to identify WAP content on origin servers. WWW-standard URLs are used to identify local resources in a device, e.g. call control functions.
- Content typing – All WAP content is given a specific type consistent with WWW typing. This allows WAP user agents to correctly process the content based on its type.
- Standard content formats – WAP content formats are based on WWW technology and include display markup, calendar information, electronic business card objects, images and scripting language.
- Standard communication protocols – WAP communication protocols enable the communication of browser requests from the mobile terminal to the network web server.

These standard components enable all WAP content and applications to be specified in the internationally established WWW content format.

4.3 COMPONENTS OF THE WAP ARCHITECTURE

The WAP architecture provides a scalable and extensible environment for application development for mobile devices. This is achieved through a layered design of the entire protocol stack, in which each layer is accessible by the layers above, as well as by other services and applications. Figure 5.2 depicts the WAP protocol stack.

FIGURE 4.2 – WAP ARCHITECTURE



As is visible from Figure 4.2, the WAP architecture enables other services and applications to utilise the features of the WAP stack. This is highly beneficial since it allows the WAP architecture to support a range of applications and services, which are not currently specified by WAP. The support for these other applications and services is achieved through a set of well-defined interfaces, which enable these external entities to access the session, transaction, security, and transport layers directly.

The various layers of the WAP stack will be discussed more fully in the following sections. The application layer will be discussed next. This layer is definitive of the entire architecture in how applications are partitioned among the system elements, and in how these applications are able to access underlying network resources and environments.

4.3.1 The Wireless Application Environment (WAE)

The WAE layer is analogous to the generic middleware layer of the mobile software system. Its purpose is for the provision of a general-purpose application environment that is firmly rooted in WWW technologies and philosophies. To this end, one of the primary objectives of the WAE effort is to establish an interoperable environment that will allow operators and service providers to build applications and services that are equipped to reach a wide variety of wireless platforms in an efficient and useful manner.

The major components of the WAE are:

1. WAE User Agents:

These agents are software that reside at the client, and provide specific functionality such as display content to the end-user. User agents, such as browsers, are integrated into the WAP architecture. They interpret network content that is referenced by URLs. The WAE includes user agents for the two primary standard contents, which are encoded Wireless Markup Language (WML) and compiled Wireless Markup Language Script (WMLScript) [WAE98].

2. Content Generators:

These are applications or services, such as CGI scripts, on origin servers that produce standard content formats in response to requests from user agents in the mobile terminal. The WAE does not specify any standard content generators, but expects that there will be many such generators running on typical HTTP origin servers commonly used in the WWW today [WAE98].

3. Standard Content Encoding:

A set of well-defined content encoding mechanisms and formats that allow a WAE user agent, such as a browser, to conveniently navigate web content. The standard content encoding includes compressed encoding for WML bytecode, WMLScript, standard image formats, a multi-part container format and adopted business and calendar data formats [WAE98].

Figure 4.3 provides on overview of the WAE architecture.

FIGURE 4.3 – LOGICAL PARTITIONING OF WAE IN CLIENT.

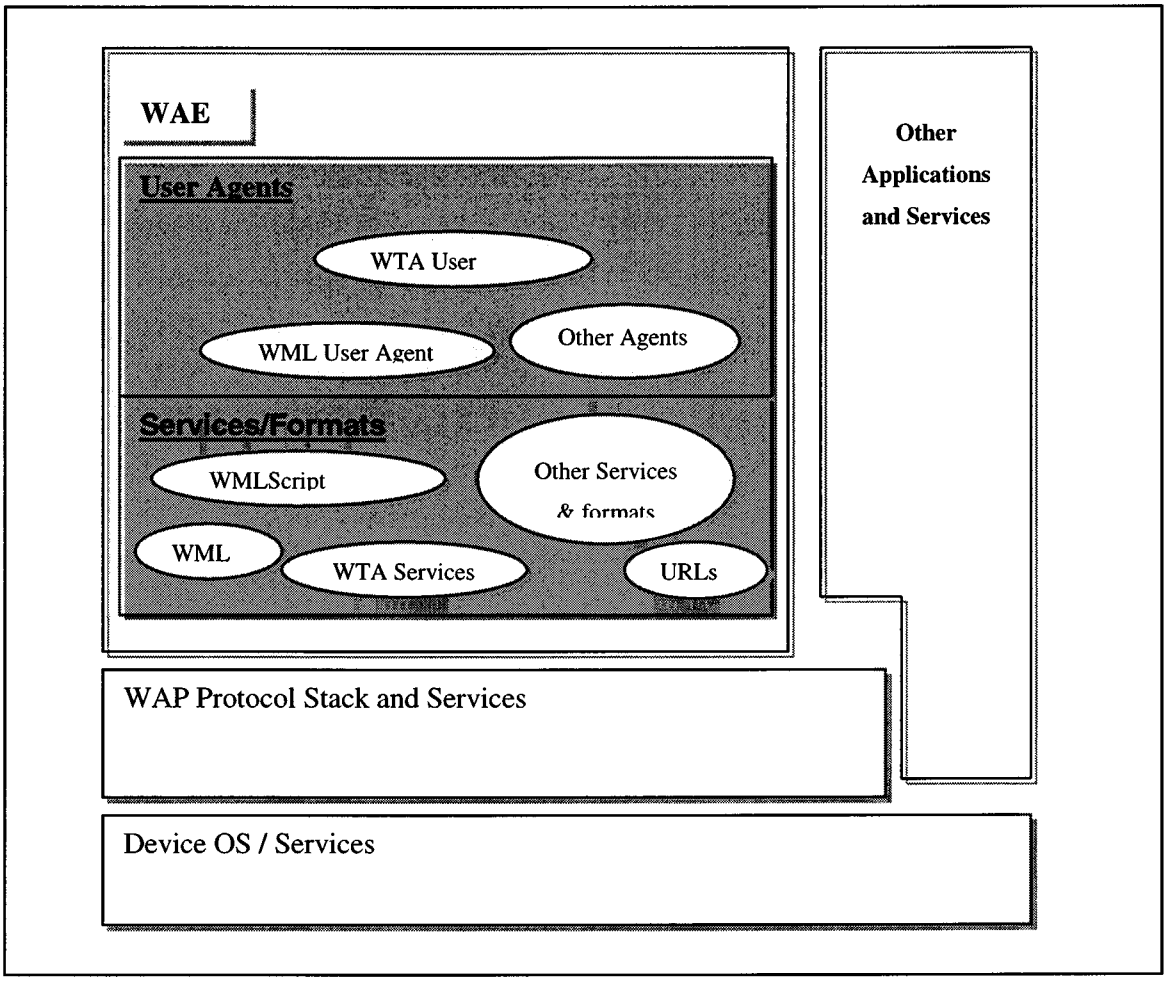


Figure 4.3 shows that the WAE is composed of two logical layers, which are the user agents layer, and the services and formats layer. The WAE by default separates services from user agents, and assumes an environment with multiple user agents. However, this logical view is not intended to prescribe an implementation model. Thus, for example, a designer may choose to combine all the services into a single agent. Alternatively, there is also the option to distribute all the services among several user agents [WAE98].

4.3.1.1 WAE User Agents

The Wireless Markup Language (WML) user agent is the only fundamental user agent of the WAE. However, the WAE is not limited to this single WML agent. The WAE allows the integration of domain specific user agents with varying architectures and environments. In particular, a Wireless Telephony Application (WTA) user agent has been specified as an extension to the WAE specification for the mobile telephony environments. The WTA extensions allow application

developers to access and interact with mobile telephone features, such as call control, as well as applications on the telephone, such as phonebook and calendar applications.

The WAE does not specify any other user agents apart from the WML agent. This is in keeping with the openness of the WAP philosophy. The features and capabilities of user agents are left entirely to the system designers. Instead, the WAE only defines fundamental services and formats that are needed to ensure interoperability among implementations. This services and formats layer is discussed in the next section [WAE98].

4.3.1.2 WAE Services and Formats

As is depicted in Figure 4.3, WML and WMLScript form part of the services and formats layer. These two programming languages are the major elements through which knowledge and intelligence are expressed and represented in the WAP model. WML is a tag-based language that is optimised for specifying presentation and user interaction on limited capability devices such as hybrid communicators, and other mobile devices. Some of the major features of WML include support for text and images, various forms of user input, and a number of navigation mechanisms that allow roaming the history stack. In addition, there is international support in the form of the document character set, Man Machine Interface (MMI) independence, narrow band optimisation, and state and context management (for a discussion of these features see [WAE98]).

WMLScript is the second major element of the services and formats layer. WMLScript is a lightweight procedural scripting language that is based on a subset of the JavaScript scripting language. It enhances the standard browsing and presentation facilities of WML with behavioural capabilities, supports more advanced user interaction behaviour, adds intelligence to the client, provides a convenient mechanism to access the device and its peripherals, and reduces the need for round-trips to the origin server (see [WAE98] for a fuller discussion).

Another important element of the services and format layer is the WAE content formats. The WAE includes a set of specified content formats that facilitate interoperable data exchange. The method of data exchange depends on the data and the targeted WAE user agent. The two most important formats defined in the WAE are the encoded WML, and the WMLScript bytecode formats. These two encoding formats allow transmission of WML and WMLScript more efficient, as well as minimise the computational efforts that are required of the client.

There are a number of other formats for data types that can be adopted in the WAE. In terms of continuous data, particularly visual data types, a highly valuable feature is the support for multiple choices of pixel depth. In addition, there is support for colourspace tables, very low CPU and RAM decoding and presentation demands, as well as availability of common tools and other developer support. Other formats that are supported include multipart messages that include more than one data type, and user agent specific formats [WAE98].

4.3.2 Wireless Session Protocol (WSP)

The Wireless Session Protocol (WSP) provides the application layer of WAP with a consistent interface for two session services. The first is a connection-oriented service that operates above the transaction layer protocol (WTP). The second service is a connectionless service that operates a secure or non-secure datagram service (WDP). The Wireless Session Protocols currently consist of services suited for browsing applications (WSP/B). WSP/B provides the following functionality:

- HTTP/1.1 functionality and semantics in a compact over-the-air encoding scheme.
- Long-lived session state.
- Session suspend and resume with session migration.
- A common facility for reliable and unreliable data push.
- Protocol feature negotiation.

The protocols in the WSP family are optimised for low-bandwidth bearer networks with relatively long latency. WSP/B is designed to allow a WAP proxy to connect a WSP/B client to a standard HTTP server (See [WSP98] for a fuller discussion).

4.3.3 Wireless Transaction Protocol (WTP)

The Wireless Transaction Protocol (WTP) runs on top of a datagram service and provides as a light-weight transaction-oriented protocol that is suitable for implementation in "thin" clients (mobile stations). WTP operates efficiently over secure or non-secure wireless datagram networks and provides the following features:

- Three classes of transaction service:
  - Unreliable one-way requests.
  - Reliable one-way requests.
  - Reliable two-way request-reply transactions.
- Optional user-to-user reliability - WTP user triggers the confirmation of each received message.

- Optional out-of-band data on acknowledgements.
- PDU concatenation and delayed acknowledgement to reduce the number of messages sent.
- Asynchronous transactions.

(See [WTP98] for a fuller discussion)

4.3.4 Wireless Transport Layer Security (WTLS)

WTLS is a security protocol based upon the industry-standard Transport Layer Security (TLS) protocol, formerly known as Secure Sockets Layer (SSL). WTLS is intended for use with the WAP transport protocols and has been optimised for use over narrow-band communication channels. WTLS provides the following features:

- Data integrity – WTLS contains facilities to ensure that data sent between the terminal and an application server is unchanged and uncorrupted.
- Privacy – WTLS contains facilities to ensures that data transmitted between the terminal and an application server is private and cannot be understood by any intermediate parties that may have intercepted the data stream.
- Authentication – WTLS contains facilities to establish the authenticity of the terminal and application server.
- Denial-of-service protection – WTLS contains facilities for detecting and rejecting data that is replayed or not successfully verified. WTLS makes many typical denial-of-service attacks harder to accomplish and protects the upper protocol layers.

In terms of data integrity, authentication, and denial-of-service effective security is implemented by dividing the security into two phases. In the first instance, security is implemented between the terminal and the WAP gateway, and secondly, between the gateway and the external server over a TCP connection. This might seems redundant and superfluous since there encryption is already implemented in the GSM protocol. However, being an open specification Wap security should not be tied to the security scheme of any particular network.

WTLS may also be used for secure communication between terminals, for example, for authentication of electronic business card exchange. Applications are able to selectively enable or disable WTLS features depending on their security requirements and the characteristics of the underlying network. For example, privacy may be disabled on networks already providing this service at a lower layer). (See [WTLS98] for a fuller discussion)

4.3.5 Wireless Datagram Protocol (WDP)

The Transport layer protocol in the WAP architecture is referred to as the Wireless Datagram Protocol (WDP). The WDP layer operates above the data capable bearer services supported by the various network types. As a general transport service, WDP offers a consistent service to the upper layer protocols of WAP, and communicates transparently over one of the available bearer services.

Since the WDP protocols provide a common interface to the upper layer protocols, the Security, Session, and Application layers are able to function independently of the underlying wireless network. This is accomplished by adapting the transport layer to specific features of the underlying bearer. By keeping the transport layer interface and the basic features consistent, global interoperability can be achieved using mediating gateways. (See [WDP98] for a fuller discussion).

5.3.6 Bearers

The WAP protocols are designed to operate over a variety of bearer services, including short message, circuit-switched data, and packet data. The bearers offer differing levels of quality of service with respect to throughput, error rate, and delays. The WAP protocols are designed to compensate for, or tolerate these varying levels of service.

Since the WDP layer provides the convergence between the bearer service and the rest of the WAP stack, the WDP specification [WDP98] lists the bearers that are supported, and the techniques used to allow WAP protocols to run over each bearer. The list of supported bearers will change over time with new bearers being added as the wireless market evolves.

4.3.7 Other Services and Applications

The WAP layered architecture enables other services and applications to utilise the features of the WAP stack through a set of well-defined interfaces. External applications may access the session, transaction, security and transport layers directly. This allows the WAP stack to be used for applications and services not currently specified by WAP, but deemed to be valuable for the wireless market. For example, applications, such as electronic mail, calendar, phone book, notepad, and electronic commerce, or services, such as white and yellow pages, may be developed to use the WAP protocols.

4.3.8 WAP Overview

The preceding discussion highlighted WAP to be a modular structured system caters for the three highly critical features of extensibility, adaptivity, and abstraction. The goal of extensibility is featured in the ability of WAP to accommodate many of the currently available bearer services, and is reinforced in the ability to effortlessly incorporate 3rd generation network technologies as they emerge. Adaptivity is evident in the underlying design goal to optimise the interaction between lightweight mobile clients and fixed networks in light of the limitations faced by those devices, as well as transmitting data over the air.

The concept of abstraction is visible in the openness of the WAP specification. In specifying the WAP components, there are very few prescriptive specifications. For example, WML is specified in a way that allows presentation on a wide variety of devices. At the same time, however, the specification allows vendors to incorporate their own Man Machine Interface (MMI). Thus, WML does not specify how implementations request input from the user. Rather, WML specifies the intent in an abstract manner. This level of abstraction allows WML to be implemented on a wide range of input devices and mechanisms [WAE98].

Bluetooth technology represents the second powerful catalyst towards enabling hybrid communicators, and other device categories to usher in the new information society. In the following section, Bluetooth technology will be discussed in terms of its impact on architectural issues.

4.4 BLUETOOTH TECHNOLOGY

The Bluetooth special interest group was formed in February 1998 by mobile telephony and computing leaders Ericsson, IBM, Nokia, and Toshiba. Like the WAP specification, Bluetooth represents another powerful open specification that is geared towards the advancement of wireless computing of data and voice traffic. According to Seybold, Bluetooth has succeeded in merging communications and computing technologies [Seyb98]. The Bluetooth specification facilitates seamless voice and data communication by means of short radio links, which can allow mobile users to connect to a wide range of devices easily and efficiently without the need for cables, or other physical connections [HNIJA99]. As such, the Bluetooth technology is based on a low cost short radio link that is built into a 9 x 9 mm microchip, which facilitates protected and ad hoc

connections for mobile and stationary communications environments [Blue99]. Consequently, in [HNIJA99] the type of computing that Bluetooth implements is referred to as hidden computing or unconscious connectivity.

A key characteristic of Bluetooth technology that differentiates it from other wireless technologies is that it combines usability models based on the functions that are provided by different devices. The importance of this device independence and functionality characteristic is that although each device functions separately in terms of its intended functionality, when they are brought in close proximity of each other these devices are able to connect to provide a useful and efficient combined function, as in the case of a phone and a PDA combining to enable remote network access. Thus, Bluetooth conforms to the standard in terms of extensibility, abstraction, transparency, and adaptivity that was set by the WAP specification. A fuller description of the Bluetooth specification is provided in the next section.

4.4.1 Bluetooth Specification

The Bluetooth specification defines the requirements that ensure interoperable communication between Bluetooth devices from different manufacturers. Figure 4.4 provides an overview of the Bluetooth application development framework [HNIJA99].

FIGURE 4.4 – BLUETOOTH APPLICATION FRAMEWORK

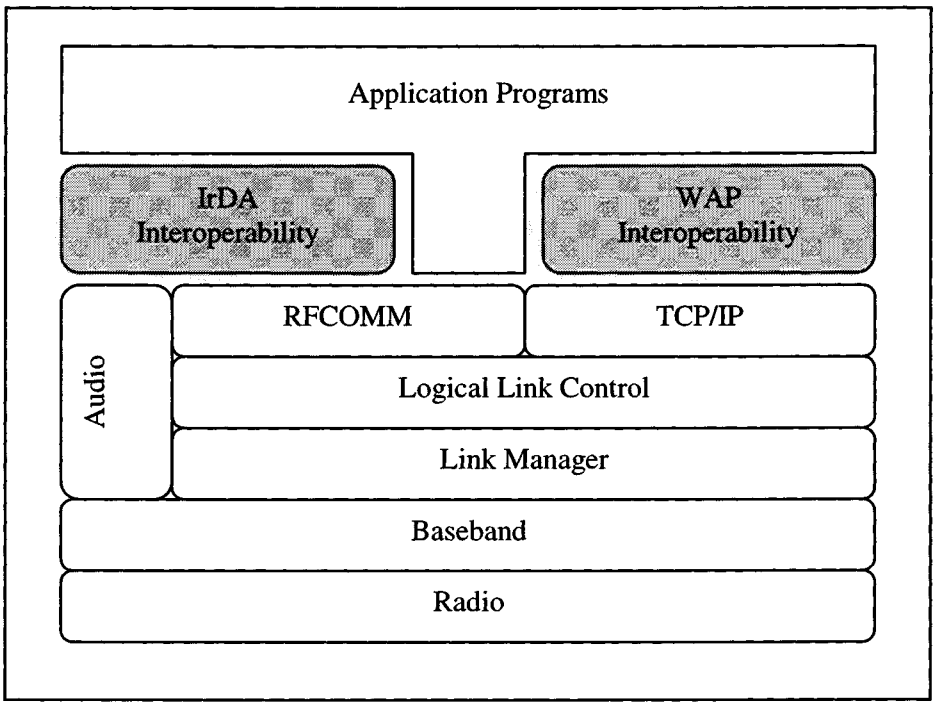


Figure 4.4 outlines the Bluetooth application framework in the context of the radio and protocol stack. The Radio layer handles the sending and receiving of modulated bitstreams, whereas the Baseband layer defines the timing, framing, packets, and flow control on the link. The Link Manager is responsible for managing the connection states, enforcing fairness among slaves, power management, and other management tasks. The Logical Link Control handles multiplexing of higher level protocols, segmentation and reassembly of large packets, and device discovery. Audio data is mapped directly to the Baseband layer, while audio control is layered above the logical Link Control layer [HNIJA99].

The RFCOMM (Radio Frequency Communication) and network level protocols provide different communication abstractions. RFCOMM provides serial cable emulation by using a subset of the ETSI GSM 07.10 standard. Other parts of the Bluetooth deal with the issue of interoperability with other protocols, and protocol stacks. For instance, in order to define TCP/IP over Bluetooth requires that bridging, address resolution, and multicast/broadcast mappings issues to be solved. In addition, in order to fuel the acceleration of the number of wireless specific applications, the Bluetooth Special Interest group is contemplating interoperability with higher layer IrDA (Infrared Data Association) and WAP protocol stacks [HNIJA99].

Given the motivation behind the development of Bluetooth technology, as well as the stated objectives which it hopes to achieve, it is obvious that Bluetooth will have a tremendous impact on the future of wireless computing. The primary characteristic of Bluetooth lies in its ability to provide abstract services that are delivered to the mobile terminal in a totally transparent manner. Such types of abstract services, which available to the entire range of applications running on a device must be implemented in the underlying mobile system in a manner that seamlessly blends them with the existing services. Thus, the task of integrating Bluetooth into the mobile software architecture will affect the application and middleware layers, and introduce a new set of protocols into the mobile software architecture.

4.5 SUMMARY

Since the WAP Forum was first created it has been successful in attracting the support of a large and diverse base that is made up of all the major players in the telecommunications and computing industries. Indeed, it is reasonable to conclude that WAP is currently the most prominent standard in terms of providing network access to small, pocket-sized terminals.

It has been estimated that in three to five years 20 – 50 percent of all Internet nodes will be wireless mobile terminals [FRGH99]. This vision is entirely possible given that mobile devices are getting smaller and more powerful, the growth in new wireless networks, and the growth of the Internet as the global information medium. New packet-switched networks and 3rd generation technologies will enable higher data rates, and will allow terminals to stay connected to the network for longer periods of time. At the same time, Bluetooth technology will offer the opportunity to interconnect mobile devices in a flexible manner.

From the discussion provided in this chapter it has to be concluded that both Bluetooth and WAP technologies are equipped to take the lead in any future developments in wireless computing. Their openness of specification will allow these technologies to be customised and optimised to suit the emerging data and application needs in the highly volatile wireless environment.

The ability of pocket-sized terminals to access multimedia content over the wireless link will be one of the next significant development in mobile computing. Given the extensibility, adaptivity, and abstraction qualities that are included in WAP, this technology will provide a powerful environment for the wireless multimedia vision to become a reality. It was for these reasons that the Jyväskylä

Mobile Architecture for Multimedia data (JMAM) system will be developed in the WAP environment.

In the next chapter, mobile software will be discussed. The purpose of this chapter is to provide context for the software level at which the communications and data management strategies that are involved in mobile computing are best accommodated.

5 MOBILE SOFTWARE

As was shown in Table 1.1, there is a substantial resource chasm between heavyweight mobile devices, such as laptop computers, and their lighter counterparts, such as hybrid communicators. Lightweight mobile terminals are characterised by three predominant restrictions, which may be stated as:

- Low memory capacity.
- Slow processors.
- Limited power availability.

These fundamental limitations play a powerful role in the design of mobile software systems. This chapter will analyse these limitations, and provide strategies on how the software architecture of a mobile system may be designed to support these inherent restrictions.

In terms of memory, it is of utmost importance in the design of a mobile system that the memory requirements of the algorithms that are executed at the mobile terminal are not high consumers of memory. This memory restriction is equally applicable to system software as well as to data applications that are loaded at the terminal. Accordingly, the memory requirements of applications, and of the OS are critical considerations that feature in decisions regarding the location of computational activity. In these situations, the memory limitations are to be prudently balanced against the communication cost of transferring the data over the wireless medium for processing at another location. Indeed, even in the event that the mobile terminal has the memory resources to load an application applet, this task still requires that presence of the P-interpreter, which itself demands a significant portion of memory [Vei99].

There are also several data management issues that arise as a result of the memory deficiency of the mobile terminal [ImB93, PiB94, Pib95, EJF95]. For example, for applications that require the caching of preliminary results as a prerequisite for proceeding, a strategy must be devised as to where these results are to be temporarily located. Furthermore, this temporarily location must be cost efficient in terms of the communications overhead of transporting data, and must also respect quality of service thresholds regarding time sensitive data. This latter quality of service demand is particularly important to continuous data where multiple streams of delay sensitive data have to be synchronised, and in some instances, prioritised to provide understandable viewing, or playback.

The slow speed of the processor of the mobile device is another characteristic that has a critical bearing on the design of mobile software systems. The impact of this deficiency imposes a severe restriction on the complexity of algorithms that can be executed at the mobile terminal in light of the general trade-off between the time and space complexity of algorithms [Vei99]. In that, the more space that is available in which to encode a problem, the less time is needed by the algorithm to solve that problem, and vice versa. The critical problem that is faced by hybrid communicator devices is that both time (processor speed) and space (memory) is limited.

The processor deficiency can be cured to some extent by increasing the speed of the processor. However, this approach introduces the problem of scalability. In this vein, Veijalainen has proposed that a more efficient approach is to reduce the complexity of the algorithms that are executed at the mobile terminal [Vei99]. This proposal has a critical bearing on the processing of continuous data in mobile networks, such as GSM, where the bandwidth is allocated by the underlying network. In such networks, the allocated bandwidth is currently so limited (see Table 2.1) that the processing of continuous data types involves heavy compression and decompression techniques at both ends of the connection.

Consequently, in light of the reduction in complexity heuristic, it is important that compression/decompression algorithms that are implemented at the mobile terminal are conscious of the processor, memory, and energy limitations that are present in the mobile terminal. In this regard, the standard proposed in [ACKR98], although specifically aimed at conserving the energy supplies of the mobile terminal, presents a viable strategy. This strategy involves decreasing the number of transmitted bits over the wireless link during the transmission of a video stream. In addition, the MPEG-4 encoding and decoding standard provides an adaptable representation scheme that accommodates very low bitrate applications [MPEG99]. The benefits of implementing the MPEG-4 standard in applications designed for lightweight mobile devices include the following:

- High compression performance. This refers to the efficient time complexity of the algorithm in view of the limited space in which it is executed.
- Flexibility implementing the encoding and decoding complexity, such that for example, different spatial resolutions, temporal resolutions, and quality can enable a flexible compromise between quality, performance, and cost.
- Object based coding functionality that allows for interaction with audio-visual objects. This provides the basis for new interactive applications in the mobile environment.

- Face animation parameters can be used to reduce bandwidth consumption for real-time communication applications. This is particularly important for mobile conferencing applications.

The third characteristic of mobile terminals is limited energy resources, which arises from the finite life span of the onboard battery. This limitation can be confronted through strategies that are embedded at the hardware level, as well as at the OS (operating system) and higher application levels. In terms of power conservation at the hardware level, strategies have typically focused on optimising power usage in the CPU, Transmitter and the Receiver, since these are the major consumers of battery power in mobile terminals. The types of strategies that have been employed here have included variable clock speed CPUs, flash memory, and disk shutdowns [YDS95, Agr98].

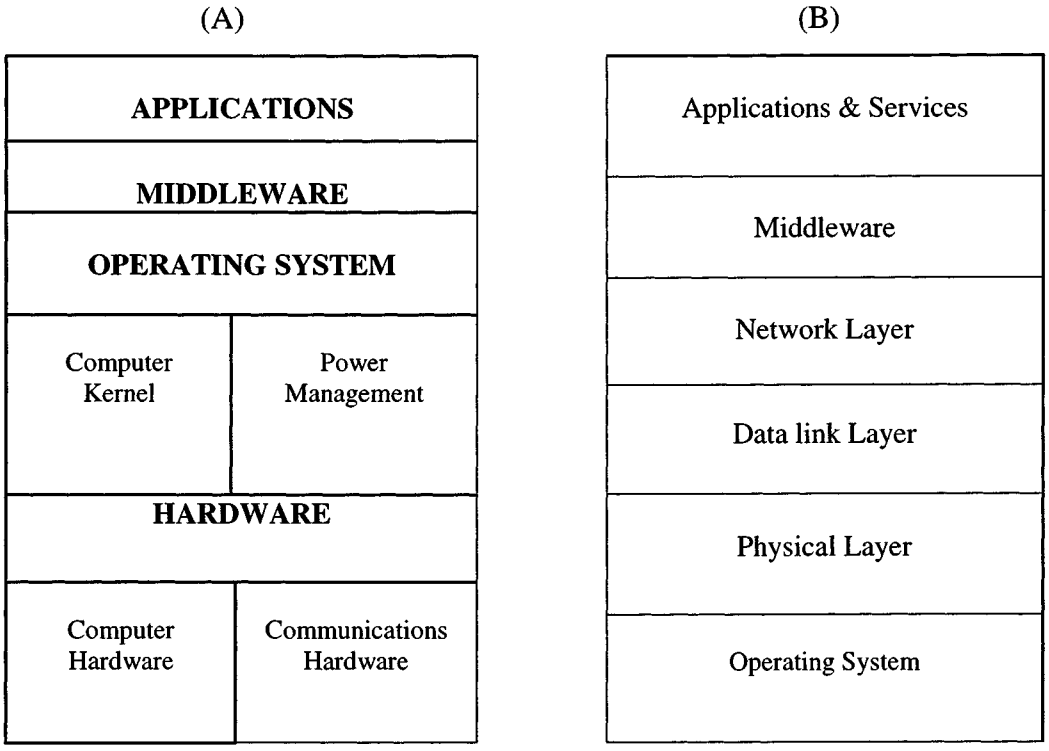
At the OS level, Weiser et al have proposed a highly useful strategy whereby on a terminal with a variable speed processor, the OS is able to reduce the power consumption by efficiently scheduling jobs [WWDS94]. In addition, power conservation techniques have also been approached from the perspective of the Medium Access Control (MAC) level. Such MAC level conservation strategies are founded mainly on the characteristics of the radio device of the mobile terminal. In that, of the three possible transmitter states of transmit, receive and standby, most power is consumed in the transmit state, and the least in the standby state [AFZ95 and Agr98]. Based on these power consumption patterns, a MAC protocol stack can be designed to exploit energy saving opportunities through such means as the elimination of collisions, and allowing the radio transmitter to enter standby mode whenever there is no activity [Agr98].

At the application level, energy saving can be achieved by restricting algorithms at the mobile terminal which demand heavy usage of the intensive energy consuming peripheral devices, such as the CPU and the memory [Agr98]. However, in general, the goal of reducing the energy consumption at the mobile terminal cannot be the sole optimising criteria of a mobile system, since this tends to introduce a number of inconsistencies. For example, if the memory requirements of an algorithm is so large that it cannot be executed at the terminal this would require the use of the transmitter to transfer the data to the server, and later the receiver to retrieve the results of the computation.

Indeed, at a more general level in terms of the three mobile terminal characteristic that have been discussed, none of them can assume precedence over each other in formulating an overriding optimising criteria when designing a mobile system. The reason for this dichotomy is that they all have an indirect impact on each other, such that, to maintain memory optimisation as an overriding system design criterion, the use of compressed data could be employed. However, this approach would have a negative impact on system speed and energy consumption, since it would require more CPU cycles to decompress the data [Vei99]. From this, it would appear that sending compressed data is more economical in terms of energy than receiving compressed data, which must then be decompressed. However, this position has not been scientifically established thus far. Thus, the responsibility of achieving the highly complex adaptivity that is essential in a mobile computing system is a responsibility that must be shouldered by the entire software structure.

Figure 5.1 provides a representation of the software architecture for two mobile systems.

FIGURE 5.1: MOBILE SOFTWARE SYSTEMS [LAW99, AGR98]



What is being compared in Figure 5.1 (A) and (B) is the generic software structure for a lightweight mobile device (A), and that of a heavyweight mobile device (B). As is depicted in Figure 5.1, the software architecture for lightweight devices demonstrates a predisposition in favour of the resource limitations of such devices.

In Figure 5.1 (A), the hardware level is composed of the computer hardware and communications hardware, and occupies the lowest level of a mobile system. The communications hardware refers to modem and radio based transmission devices. The next highest layer in a mobile system is the OS layer, which is composed of the computer kernel, power management, and the real-time kernel.

The middleware layer lies above the OS, and from a software architectural perspective is one of the most complex and interesting aspects of the mobile system. The main purpose of middleware is to facilitate the development and execution of mobile aware applications by providing a common underlying platform. Furthermore, according to Kreller et al, another key function of middleware is to accommodate the smooth transition from old standards and technologies to newer versions, such as HSCDS, and GPRS, and UMTS [KPM98].

The application layer occupies the pinnacle of the mobile system. In a mobile system this layer could be responsible for allocating computational activities between the fixed and the mobile hosts. Based on the memory, processor, power, and bandwidth limitations the application layer could selectively partition the processes between the interacting elements. In this case, most of the resource intensive activities could be offloaded to the server, while the mobile terminal plays the role of an intelligent terminal in acquiring and interpreting the results of the computation [Agr98]. However, in a system such as JMAM which is meant to run environmentally independent applications, the function of the application layer must be adjusted. In JMAM the role of the application layer will be that of providing an execution environment for applications.

In the following sections, the OS, middleware, and application layers will be discussed in deeper detail, since these are the aspects of the mobile system that are an integral part of the software architecture.

5.1 OPERATING SYSTEM LAYER

As in most other aspects of mobile software, the memory, processor, and power deficiencies of mobile terminals are significant factors that impinge on, and constrain the mobile OS. The mobile OS, as is revealed in Figure 5.1, is composed of the computer kernel, the power management facility, and the real-time kernel. The computer kernel is the entity that manages access to physical resources, such as the CPU and disk space. The power management facility is responsible for

reducing power consumption, and is an important asset in the effort to prolong the operating span of mobile terminals. To this end, strategies such as cutting the energy supply to significant power consumption devices as the disk, transmitter, and CPU during idle periods are viable power conservation approaches that can be pursued at this level [ACKR98]

However, power conservation techniques that are based on the design of CPU circuitry have been shown to be more adept than those that involve shutting down the CPU, and other peripheries, during periods of inactivity. The power consumption of the dynamic components of a mobile terminal is proportional to *CV'F, where C* is the capacitance of the wires, V is the voltage swing, and F is the clock frequency [FoZ94]. This representation suggests that there are three ways to save on power consumption. Firstly, the capacitance can be reduced by developing more efficient chip technology. Secondly, the voltage can be reduced by redesigning chips to operate at lower voltages. Finally, the clock frequency can be reduced, whereby power savings can be realised at the expense of computational power [WWDS94, YDS95]. The role of the OS in this energy saving matrix is based on its job-scheduling ability. The scheduler is able to balance the system load in a manner whereby a feasible and energy efficient schedule of CPU cycles is allocated for each incoming job [YDS95].

The responsibility of the final entity in the OS, the real-time kernel, is that of managing the communications link. This kernel is needed if the OS is required to perform computing as well as communication tasks. However, this functionality adds considerable complexity and memory overhead to the mobile OS. At the same time, real-time ability is the most significant market driver in the battle among competing OSs for the mobile device market [Law99]. In the course performing this role, the real-time kernel can be utilised as a valuable source of QoS information to higher layers in the mobile software system.

Having gathered an overview of the mobile OS, and of what are its primary functions, the next step in the process is to consider the currently available mobile OSs.

TABLE 5.1 – MOBILE OPERATING SYSTEMS

QUALITIES	**GEOS [Geos99]**	**Windows CE [Micr99]**	**OS-9 [Os99]**	**Palm OS [Palm99]**	**Symbian EPOC [Sym99]**
MEMORY REQUIREMENTS	500 Kb	2 Mb	68 – 308 Kb	1 Mb	1 Mb
INSTALLATION SCALABILITY	NO	YES	YES	NO	YES
REAL-TIME	YES	POOR	YES	NO	YES
NETWORK	GSM	CDMA	CDMA	N/A	GSM
APPLICATION RANGE	WIDE	WIDE	LIMITED	LIMITED	WIDE
NETWORK CAPABILITY	YES	YES	NO	NO	YES
SPEED – Time taken to complete a submitted request	FAST	SLOW	FAST	FAST	FAST
SPECIAL FEATURE	Minimises data loss and fast reboot	Wide support from device and application vendors	High performance for limited range of functions	Simple and light	Wide support in Europe, small with good fault protection capability

The comparison of mobile OSs in Table 5.1 shows that such an OS generally requires a memory capacity of 68 Kb – 2 Mb. To place Table 5.1 in a device context, the Nokia 9110 Communicator provides an excellent case study. The 9110 Communicator has a memory capacity of 8 Mb of internal memory of which 2 Mb is reserved for user data. In addition, there is a card slot for an additional 4 Mb of user data [Dis98b]. Thus, it is clear that the GEOS and EPOC OSs offer this device type the optimum platform, and especially EPOC, since the installation can be scaled to suit application needs, or memory limitations.

Other important features of the mobile OS include the range of applications that can be supported by the OS, as well as its ability to be connected to a PC, or indeed, to a network. In both these

regards, Windows CE has the lead on other OSs for two reasons. Firstly, Windows CE is the largest OS, and thus it incorporates a broad range of support many applications and devices from a multitude of vendors. In the second instance, Windows CE is a product of the Microsoft Corporation, and is thus able to run many of this overwhelming software developer's other products. Indeed, according to a study of mobile OSs performed by Lawton, it was noted that despite its deficiencies, the Windows CE OS might become the enterprise mobile OS standard simply because of these reasons [Law99]. However, an important aspect of this discussion is that the EPOC and GEOS OSs are also competent to support a wide range of applications.

A further important mobile OS criterion is the system platform that is targeted by the OS. Although it is possible for an OS to be adapted to either the CDMA or the GSM platform, it is highly important that the EPOC and GEOS OSs were designed specifically for GSM radio networks. Indeed, the GEOS OS is currently the OS that runs the Nokia 9110 hybrid Communicator. However, the EPOC OS is the creation of a co-operative venture between several major players in the GSM market. It is a mobile OS that is optimised for low-powered handheld devices, and for propelling long running, mission-critical application [Sym99, Dis99b].

It is certain that EPOC will replace the GEOS OS in the near future as the hybrid communicator class OS [Dis99b]. Accordingly, in light of the impact the this OS will have on the hybrid communicator device category, it will be discussed in greater detail in the following section.

5.1.1 Symbian's EPOC Mobile Operating System

Symbian LTD is the result of a joint venture between Ericsson, Motorola, Nokia, and Psion. The primary goal of this strategic alliance is to develop a GSM based OS to compete in the mobile OS market [Law99, Dis99b]. One of the most notable results of the Symbian venture has been the EPOC OS, which is a third generation mobile OS for ROM-based computing. The main benefits of EPOC are that it is a fully multi-tasking, 32-bit OS that supports a pen-based GUI (Graphical User Interface), and incorporates full networking capability.

Furthermore, EPOC is component structured, and so can be scaled from relatively large configurations for a fully functional handheld computer, to small configurations for embedded applications. In these respects, the EPOC design represents radical departure from the design of previous OSs, which enables it to achieve a much tighter integration of networking and data

management. Moreover, its component-based portable design allows it to address a much wider range of applications than handheld computing [Sym99].

5.1.1.1 Multi-Platform Real-Time Microkernel

The majority of EPOC is platform-independent. At the core of EPOC is the E32 OS, which provides a user library, microkernel and hardware abstraction layer. Only a very small part of E32 is hardware dependent. The microkernel has been ported to a number of platforms, including a development environment running on a PC, using the Microsoft Windows environment for system services, and using Microsoft Visual C++ 4.0 IDE for management of large-scale programming projects and sophisticated debugging [Sym99].

EPOC also includes a comprehensive pre-emptive multi-tasking and integrated power management facility. A robust process model uses an MMU (Mobile Multi-tasking Unit) to give each process a separate address space. Within each process, one or more threads represent independent units of execution. However, a single-process configuration without an MMU is available, which further reduces ROM size and RAM requirements. The ROM size requirement of this OS depends on the product requirements. For instance, a typical high-end PDA configuration including Word Processor, Spreadsheet, Email, World Time, and such other services, would be around 4Mb of ARM 32-bit code. In contrast, an embedded mobile phone application would be around 1Mb of ARM 32 bit code [Sym99]. ARM 32-bit code is a development environment for the Advanced RISC Machines (ARM) family of processors [ARM99].

EPOC's microkernel is fully re-entrant, with very low interrupt and thread latency. Standard user-mode threads have a maximum latency in the order of tens of milliseconds. Real-time threads, running in privileged mode, have a maximum interrupt latency of a hundred microseconds. This enables EPOC to run real-time communications software without using a second processor [Sym99].

5.1.1.2 Stability

EPOC was designed to boot once, and thereafter run without interruption. Thus, it is possible to run applications for long periods - months or years - without restarting. When a machine is switched off, it may be switched on again, with all applications instantly in the state they were in at switch-off. This presents an entirely natural interface to the user of a handheld device. Consequently, EPOC creates an excellent and stable environment for the mission-critical handheld applications

that are needed in the business world. As a result, EPOC has significant advantages in these areas over other platforms, whose heritage is the desktop environment [Dis99a].

One example of EPOC's approach to mission-critical software is exception handling. When a program encounters an exception, such as out-of-memory, it should recover without any loss of data and, if the exception occurred midway through an operation that had begun to allocate other resources, these resources should be freed immediately. These requirements are addressed with an exception handling mechanism, a clean-up stack, and the type classification system used for all C++ classes. Consequently, EPOC-based handheld computers do not need to be regularly re-booted, and applications can survive out-of-memory conditions without restarting or losing data [Dis99a].

5.1.1.3 Wired & wireless network computing

EPOC supports the full range of network computing activity, with support for serial and infrared hardware, a BSD-compatible sockets programming interface, protocol stacks for TCP/IP, SLIP/PPP, IrCOMM, IrDA, and file transfer protocols including X-, Y- and Z-modem. Also included is Psion's proprietary PLP protocol, which is used on EPOC16 to support the industry-leading PsiWin desktop integration product. These protocols support applications such as web browsing, e-mail and terminal emulation [Sym99].

5.1.1.4 Data Management

The EPOC OS incorporates the VFAT filing system, which supports both DOS compatibility, and long filenames. EPOC's major building block for storing persistent data is the stream store. A stream interface is provided, which may be supported by any class. A class supporting a stream interface can support being stored in a stream store (usually, a file), sent over a communications link, or copied to a clipboard or to an undo buffer. Reading from the stream produces the opposite effect. Stream stores may store complex networks of streams, which are used to support deferred loading, embedded objects, and a relational database. An SQL-compatible relational database is provided, with full support for multiple tables, indexes, rich data types, and transactions with commit and rollback capabilities [Sym99].

5.1.1.5 Compatibility and Flexibility

EPOC machines are fully supportive of office applications, personal information management and communication using e-mail, fax and SMS. There are capabilities to print directly from an EPOC machine, to exchange data with other EPOC machine, as well as standards-compliant devices and

phones, using infrared. There is even freedom to install applications that have been downloaded from websites, without using a PC [Sym99].

For many purposes, however, PC connectivity and synchronisation with PC - or corporate-based data are important. PCs also provide a convenient backup medium. EPOC Connect integrates with Windows Explorer to provide these and other functions. EPOC Connect is an application, which runs on PCs under Microsoft Windows NT 4.0, Windows 95 or Windows 98. EPOC Connect is a framework providing connection to EPOC machines, file management, data synchronisation, backup/restore, and utility applications [Sym99].

5.1.1.6 Summary of EPOC

The preceding discussion on EPOC highlighted it to be a powerful, but yet lean, operating environment, which has enormous potential to accommodate the future developments of mobile computing. Among the current and future developments that EPOC will have a positive impact on are WAP, wireless Java, and Bluetooth technologies [Dis99b]. Furthermore, this OS will be a powerful complement to UMTS when this third-generation telecommunication standard is finally implemented in 2002.

The superior position of EPOC as the premier OS among its competitors is further reinforced by its international outlook. However, at the same time, there is a strong association to European technology and standards, which does not diminish its internationalism. These facts, together with its score in the various yardsticks that were applied in Table 4.1, will give EPOC a powerful role in the miniature device OS market.

5.2 MIDDLEWARE LAYER

Middleware occupies the position between the application layer and the OS layer in the mobile software system. As such, this software element is situated in a unique intermediary locale, which is evident both in structural terms, as well as in terms of its designated role. Middleware can be conceptualised as the software, which does not directly handle application protocol needs, but functions in a generic manner in providing intermediate network services and execution environments to applications [Per98]. Thus, middleware plays a central and brokerage role in the interaction between applications and the external environment, as well as with other internal system resources. This factor makes middleware one of the most interesting and challenging components of the mobile software system.

There is a specified range of services that are suited for inclusion in middleware. However, in terms of hybrid communicator devices, the functionality that is implemented at the middleware has to be restricted in view of the resource limitations. Some of these intermediary network and environment services that are prominent candidates for inclusion in the middleware layer, are the following:

1. Service management
2. Data Management
3. Proxies and Agents
4. Communication resilience
5. Continuous data synchronisation

Each of these candidates will be discussed in the following sections.

5.2.1 Service Management

The purpose of service management in the context of the JMAM architecture is to store the user's network service profile, or the set of network services that the user has subscribed to and is authorised to access. The advantage of such a feature is that the service, or application profile is independent of, and de-coupled from the features of any particular mobile device. In this system, the application and network services are closely tied to the user. The middleware layer provides a stable and accessible environment for the type of information that is stored in the user's service profile. It is accessible to both the users, as well as to the network operator through the API for online updating.

However, despite the benefits that are visible from implementing a service manager, the concept can appear susceptible to the criticism of being superfluous. This criticism can arise out the fact that the web interface is widely regarded as a universal access to network information. However, the Internet and the associated value added services is progressing at an almost revolutionary pace. Given the current growth and development rate, it is entirely possible that there will be situations in which operators will want to, or indeed need to, develop specialist services for market niches [Dis98a]. In these, as well as many other instances, the freedom that is imparted to both operator, and user, by the service manager will be a valuable commodity.

5.2.2 Data Management

Data management functionality at the middleware layer is important in terms of enabling the development of environmentally independent applications. The placement of data management functionality at this layer is also beneficial for disconnected operation in terms of data hoarding, and for the attainment of data consistency at the terminal and at the server after reconnection. As was noted earlier, the partitioning of applications between the terminal and the server is normally defined in the application. However, in order to reduce the complexity of developing applications for the terminal, where activities must be partitioned for every application, a more effective strategy would be to implement this generic role at the middleware layer.

The purpose of disconnected operation is to facilitate operation by the terminal even though the connection to the fixed network has been terminated. In situations where a network disconnection is anticipated by the terminal through monitoring of the wireless link, the terminal is required to instigate proceedings that will minimise the effect of the network interruption as best as its onboard resources will allow. There are three states, which interact to make disconnected operation possible. The states may be termed data hoarding, operation, and reintegration [PiB94, PiSa97, LeC98].

In the data hoarding phase the necessary data items that are needed for autonomous operation are downloaded to the mobile host from the fixed network. The type of data items that are downloaded can be files, relations or Html pages, depending on the category of the user application. A critical issue in this phase revolves around the determination of which items are to be hoarded. In this regard, two approaches have evolved. The first is to let the user to play an active role in specifying which objects are to be hoarded, whereas the second approach favours the use of previously accessed data by the application as a guide to predicting future data needs [PiSa97].

In the disconnected operation phase, the mobile host is able to utilise the hoarded data items so as to continue operation. Any requests for items that have not been hoarded are rejected, and may be queued for later servicing when the host has been re-connected to the network. When such re-connection occurs, these queued requests are drained from the host's log, and serviced appropriately. The reintegration phase of the disconnected operation process also becomes active after reconnection to the network. In this phase, any data that was updated at the host during disconnection must be merged with the network data, and all conflicts resolved in a manner that assures the integrity of the data [PiSa97]. In this regard, the work of the Transaction Management Support for Co-operative Applications (TransCoop), which involved the development of semantically oriented schemes to facilitate the merging and reconciliation of hoarded data with data at the server, offers tremendous insights into what is being proposed in this work [BLV97].

The ability of a mobile terminal to work autonomously during a network disconnection has a critical and direct bearing on the available memory resources at the terminal. Consequently, since the focus of this thesis, and of the JMAM architecture, is on minuscule devices that are memory deficient, the use of disconnected operation must be severely curtailed. For instance, there can be sufficient memory resources to hoard the email and other messages such as fax or SMS, and small office files for use during disconnection. However, hybrid communicator devices cannot accept the hoarding of multimedia and other bulky data objects due to the high memory capacity that they require. Thus, disconnected operation must be approached on an application basis, whereby the system is aware of the current application class to the extent that it recognises, among others, email, fax, and continuous data application classes.

However, there will be opportunity to significantly extend this hoarding strategy in the near future. The capacity roadmap for the MultiMediaCard (MMU) extended memory facility that is currently used in Nokia's 9110 Communicator has recently been published. According to this roadmap, there will be cards available in the year 2000 with capacities of 32 and 62 Mb, and by the year 2001 this capacity is expected to increase to 256 Mb [Sjos99]. Given the statistic that 1 Mb of memory can hold 500 pages of unformatted text, the realisation of the capacity roadmap will permit hoarding of large text files, and later, even the hoarding of larger multimedia and other graphic data.

5.2.3 Proxies and Agents

The use of proxy and agent technology is mainly to facilitate access to diverse network environments by weak clients. Proxies are standardised network components that are used to perform specialised services such as protocol translation, or to provide access to specialised network environments, such as multimedia servers [Per98, Agr98]. In contrast, agents are purely software modules that are dispatched from some source terminal to roam among a set of network servers until their specified task is accomplished [CGH95, PiB95].

The use of proxies, and to lesser extent agents, is already an essential aspect of mobile computing, and they are implemented in some manner in all of the system architectures. Many benefits can be derived from the use of proxies and agents, which are all rooted in the opportunity to off-load many of the intensive computational activities from the mobile host to the proxy and agent. Some of the major benefits that proxies and agents offer are:

- Power savings
- Reduced memory requirement
- Access to specialised environments
- Reduced mobile platform maintenance
- Performance improvements
- Disconnected operation

In order for itinerant technology to function, the software environment at every node in the mobile system must be compatible. This compatibility requirement introduces some complexity in the task of managing and controlling a mobile system, especially in terms of carrying out software upgrades. In that, any changes to the software environment will have to be reflected at the agent or proxy, as well as at the mobile host. However, the task of managing and controlling the mobile system can be simplified by implementing the itinerant environment at the middleware layer. Since the mobile host has scarce memory resources, only a skeleton representative of the itinerant is situated at the middleware layer of mobile host's software architecture. Here, the task of configuring and managing this skeleton structure can be performed easily and efficiently through the API.

5.2.4 Communication Resilience

Failure recovery is another element that can be readily implemented at the middleware layer. This service is covered to some extent by the EPOC OS, in which on of the main driving forces is for the

provision of safeguards to eliminate data loss (see Section 5.1.1), and to some extent by the increasing availability of packet services. However, in wireless communication two critical features that affect the quality of data exchange between mobile and fixed hosts are disconnection, and high latency depending on the quality of the radio signal [WeB96, KDHR97, Kat95]. These two problems are generic across the entire range of mobile applications and services, and as such, any proposed solution should be implemented at the middleware layer, which provides the common execution environment for all applications.

In terms of disconnection, the problem that is of interests here is how to enable the terminal to continue functioning without detecting a break in the link. The proposed solution is to introduce some degree of resilience into the wireless connection that would complement EPOC's fault resilience. This solution entails the use of an agent running on the mobile terminal that operates in conjunction with a proxy on the fixed network. At the same the lower network layers are responsible for detecting a break in the connection, or for implementing a break. This information is the channelled upwards to the upper layers.

The purpose of the agent is to monitor the bit stream that is being exchanged between the mobile and the fixed network. In the event that the lower layers announce a break in the connection, the agent will be able to confirm whether all of the data has been successfully exchanged. If a premature break did occur, the agent, together with the proxy, initiates a link restoration that continues the transmission from the point where the break occurred. This can be a highly valuable service in terms of continuous data, since otherwise, the entire bulky transmission would have to be restarted.

5.2.5 Continuous Data synchronisation

The ability to either encode or decode continuous data streams requires intensive processor and memory resources. Given the lack of such resources on the part of hybrid communicator devices, this functionality must be implemented with care at the mobile terminal. To combat the dual problems of low bandwidth and high random bit error rates each traffic class in the data stream must be suitably formatted and compressed before being transmitted [Bahl98, ACKRS98]. In that, the multimedia stream is compressed and split into orthogonal sub-streams, which are given different priorities. These sub-streams are then packaged and classified appropriately.

These critical and intensive tasks can be adequately performed at the middleware layer. In addition to transforming the multimedia data into a suitable state for transmission over the wireless medium, there is also the task monitoring the current QoS situation in order to synchronise the incoming multi-media data streams to render effective playback. There is a high degree of adaptivity that is required that is involved in this trading process. For example, when the link conditions are unstable the video stream can be de-emphasised, while the audio stream is given precedence, or video frames can be carefully discarded, while at the same time maintaining tolerable video quality [Agr98, ACKRS98].

5.2.6 Middleware Component Overview

The previous sections highlighted the middleware layer to be one of the most interesting and complex building blocks of the mobile system. In this regard, there are many other functions that can be appropriately situated in this layer apart from those discussed above. For instance, QoS and failure recovery functions are prime candidates for inclusion in middleware.

However, in general, a strict and isolated approach cannot be taken in rationalising the elements that should go into the middleware. A vivid example of this approach is the case of adaptivity. Adaptivity is perhaps the single most pervasive concept that permeates mobile computing, and in order for this to be achieved, there must be in place a reliable and steady stream of QoS statistics, which can be used as a basis for appropriate adaptation measures. Thus, many of the specific features that help implement adaptivity are embedded in this general heading. Furthermore, the quality and detail of the facilities that are implemented to achieve the required level of system adaptivity is dependent on the level of resources that are available at the mobile terminal.

5.3 JMAM MIDDLEWARE

The JMAM system architecture is a distributed system environment that is implemented at mobile terminal, as well as at the network access node. The following discussion is focused on the generic components of JMAM.

The grounding concepts that JMAM is built around may be described as awareness, adaptation, abstraction and extensibility.

5.3.1 Awareness

Mobile terminal characteristics, as well as the authorised network services are stored in profiles, which are managed by the service manager. In addition, the system is constantly kept aware of the communication environment through a steady stream of QoS statistics that are provided by the adaptivity manager.

5.3.2 Adaptation

A QoS facility is to be situated under the system adaptivity manager. The purpose of this facility is to provide a steady and reliable stream of QoS statistics on the operating and communications environment. This information will be of use to the communications elements to inform of the need to adapt their behaviour to suit current conditions. A main benefit of implementing adaptivity at the middleware layer is that all adaptation is transparent to applications.

5.3.3 Abstraction

JMAM will provide high level abstractions for a number of functions. These will include alerting when a message is delivered to the terminal, and reconnection in the event of a broken network connection.

5.3.4 Extensibility

In terms of third generation network standards, extensibility is inherent in this system since these technologies only serve to increase the level of bandwidth. This increase in allocated bandwidth will have no effect on the architecture on the overlying applications. Moreover, the inclusion of a service manager that stores the user's authorised network services, which can be verified and updated automatically, also introduces some extensibility in terms of the network operator's ability to develop and distribute specialised services.

5.3.5 Middleware Overview

FIGURE 5.3: JMAM MIDDLEWARE

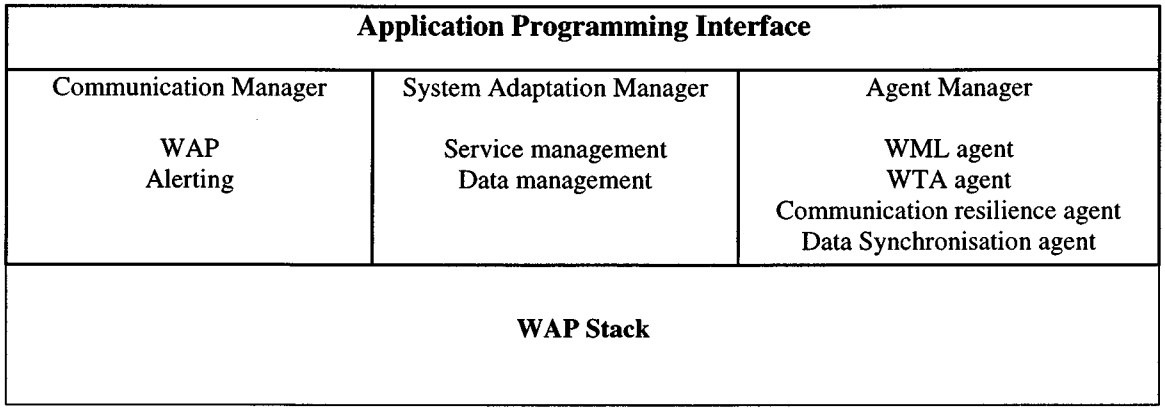
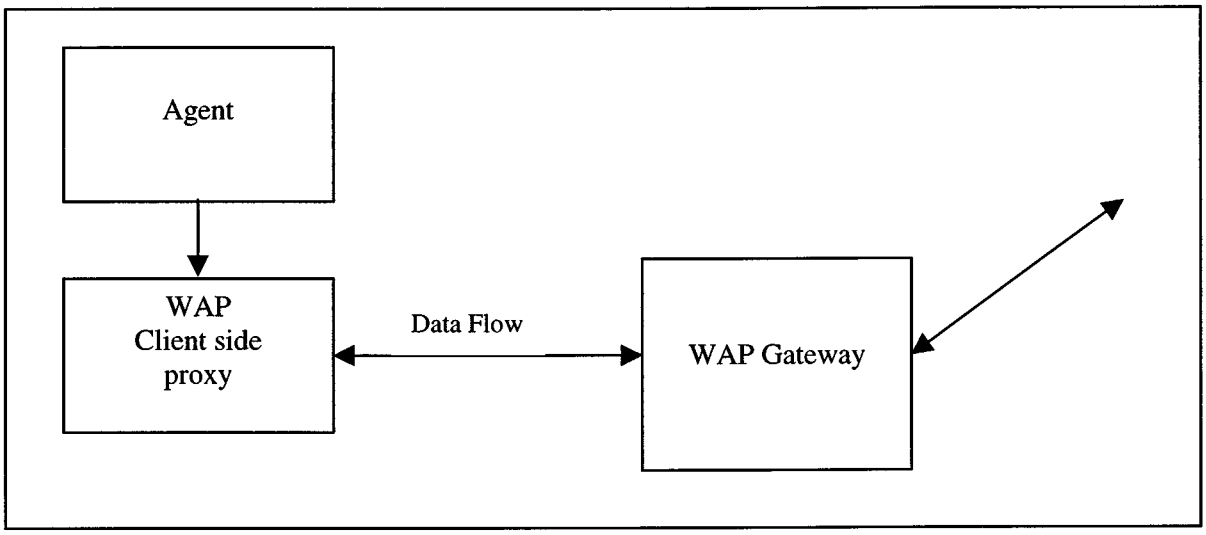


Figure 5.3 depicts the overall architecture of the JMAM middleware architecture. This middleware will utilise the multi-tasking, power management, and exception handling services that are provided by the EPOC OS. This combination will The functionality of this middleware layer combined with the services of the underlying EPOC OS points to a robust and lean mobile software architecture that caters to future application needs, and that is mindful of the technological driven nature of mobile computing.

5.3.5.1 Communication Manager

The Communication Manager of the JMAM architecture supports the WAP communication environment, alerting, messaging, and limited disconnection. The major component here is the WAP proxy, which supports Internet browsing and other advanced data services for weak devices over low-bandwidth network connections. For instance, when the Internet agent requests HTTP pages from the remote server, the WAP proxy intercepts this call, and forwards an encoded request to the WAP gateway. Figure 5.4 provides a graphical representation of various data flows that are involved in a communication session.

FIGURE 5.4: COMMUNICATIONS MANAGER

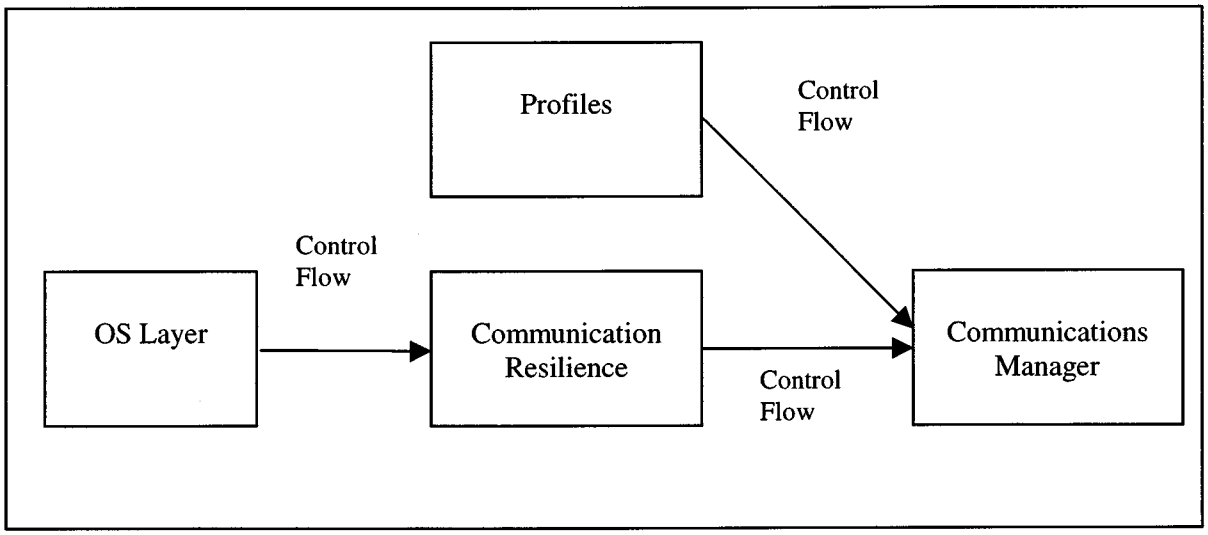


5.3.5.2 System adaptivity Manager

The System Adaptivity Manager is responsible for the provision of optimised and personalised mobile services. The services profile contains the services that the user is authorised to access, and allows the user to remotely subscribe to additional services, and the network operator to update this profile from the server.

The data management facility is responsible for making policy decisions relating to the hoarding of data during disconnection, and on the partitioning of applications between the server and the terminal. Figure 5.5 shows that interactions that are involved in the System Adaptivity Manager.

FIGURE 5.5: SYSTEM ADAPTIVITY MANAGER



5.3.5.3 Agent Manager

The Agent Manager is responsible for the various application group agents that use the JMAM services.

5.4 APPLICATION LAYER

The objective of application layer in the mobile system is to partition the tasks between the mobile and fixed hosts [Agr98, Vei99]. Thus, the aim of this layer is to implement communication asymmetry, whereby the more intensive tasks are allocated to the fixed network, while the less complex tasks can remain at the mobile terminal. The mobile terminal can then fulfil the role of an intelligent node for receiving and display the data. Table 5.2 provides a proposed heuristic along which the partitioning process can be implemented. The primary aim of this approach is to acknowledge the space, processor, and power limitations that are characteristic of the hybrid communicator device category.

TABLE 5.2: LOGICAL PARTITIONING OF JMAM APPLICATION LAYER

CLIENT	*SERVER/AGENT*
Agent	Data synchronisation
Communications	Data hoarding
Disconnection	Buffering
GUI	Filtering
Compression / Decompression	Program / Applets / Interpreter
Data synchronisation	Preliminary program data
	Compression / decompression

In addition to the partitioning of the application between the fixed and mobile hosts, another equally critical task that must be performed simultaneously at the application layer is rationalisation of the communication between the mobile and fixed hosts. There are two aspects to this rationalisation process, which is important in terms of communication reliability as well as communication cost.

Firstly, the amount of data that is transferred over the wireless medium must be minimal, and secondly, the number of times that the communications link is set up and closed during the course of an application's execution should be minimal. The importance of rationalising the communication can be appreciated when viewed in the context that the wireless link is an unreliable

medium, and is subjected to high latency. Accordingly, the longer the communication endures over this medium the higher is the probability that errors will ensue. In JMAM, communication rationalisation is partly achieved through the use of the agent computational paradigm.

5.5 SUMMARY

The task of designing a mobile system is largely to do with adapting to constraints. Consequently, it is crucial that these constraints are borne by, and shared by every layer in the mobile system. One of the main benefits of the JMAM architecture is the balance that is achieved between the constrains that are imposed by the resource deficient terminals, and the level of services that are supported by the system. Another benefit that was strived for was a high degree of extensibility in being competent to integrate new technologies into the architecture at minimal cost.

The functionality of the middleware layer that is being proposed here is ambitious in it scope. However, when considered in light of the pace of technological advancement in the hybrid communicator product range, as well as the demands of current and emerging applications, this middleware is highly pertinent and realistic. Indeed, it has recently come to light that Nokia and Palm have formed a strategic development and licensing alliance to develop, among other things, a pen-based hybrid communicator [Sjos99]. This is an extremely important development for the handheld market as it brings together the market leaders of the PDA, and the mobile phone product segments. The promise that this alliance holds is a greater and more powerful range of applications on hybrid devices, and greater interoperability among operating platforms – in this case Symbian and Palm OS.

In the following chapter, a comprehensive overview of the JMAM mobile system will be presented in light of the issues that were raised in previous chapters.

6. THE JMAM MOBILE SYSTEM

The purpose of this chapter is to consolidate the theoretical efforts of the previous chapters, and to present them in the context of the JMAM mobile system. To this end, solutions to the pertinent data management, and other challenges that were raised in chapter 2 will be developed. This discussion will be followed by a comprehensive presentation of a WAP implementation of JMAM that considers the software issues that were presented in chapters 4 and 5.

6.1 SOLUTIONS TO WIRELESS CHALLENGES

In chapter 2, network disconnection, low bandwidth, and bandwidth variability were highlighted as the challenges that afflicted communication over the wireless medium. The following sections will discuss solutions to the first two of these challenges. In GSM networks, the network allocates the bandwidth so the question of bandwidth variability is not relevant. However, in future third generation networks, there will be opportunity for the network to vary the quantity of bandwidth that is allocated to a terminal by limiting the number of slots on the time spectrum [Nokia98, Nokia99].

In addition, a number of other challenges were raised, including data management, power and memory limitations. These issues will also be addressed in the following sections.

6.1.1 Disconnected Operation

The problems that are superimposed on mobile computing as a result of network disconnection cannot be cured very easily since they are an inherent feature of the environment. The only recourse lies in designing mobile systems that are able to cope with the disconnections as they come. For handheld devices, the currently available strategies that limit the effects of disconnections impose constraints that run against the grain of these devices in terms of the level of resources that are available. This is because the processes that are involved in disconnected operation, such as data hoarding, cannot be accommodated by the hybrid communicator family of devices. As was noted in chapter 5 (see section 5.2.4), JMAM will allow minimum disconnected operation at the mobile terminal that will be decided on an application basis.

The other major problem concerning network disconnection is the locking of data items that were held by a client application when the disconnection occurred. In this regard, an effective strategy

would be to instate a deadline in the transaction. This is the approach adopted in RCP (Real-time Commit Protocol) and TACP (Time Atomic Commit Protocol), whereby if this deadline expires, the transaction triggers an exception routine that releases locked data items [YHC96]. On the other hand, the problem of unnecessary aborts due to transient disconnections can be reduced by instituting a cautious commit protocol. An excellent example of such a cautious protocol is the P2PC (Prudent 2-phase Commit Protocol) protocol. This protocol was developed with the aim of providing interacting entities in a transaction with a second opportunity to deliver an acknowledgement before the entire transaction is aborted. This strategy could prevent aborts in mobile situations where many of the disconnections are transient [BoDe96, Kat95].

6.1.2 Low Bandwidth

As was noted in earlier (see Table 2.1) GSM networks current offer data rates of 9.6 and 14.4 Kbits. However, the communications asymmetry [AFZ95] that is typical of mobile computing environments, by which simple client requests generate much larger responses from the server, can be used as the basis for coping with bandwidth scarcity. The strategy herein is to group several short-lived requests together and transmit them all at once [FoZ94]. Indeed, this is at the heart of the forthcoming packet based services, such as GPRS (General Packet Radio Service), which represents a significant growth point in the evolution of GSM towards a full fledged packetised 3rd generation network [Pere98].

The problem of low bandwidth is effectively handled WAP. As was noted in chapter 4, the WAP specification includes the use of coding and decoding mechanisms that minimise the amount of data that is exchanged over the wireless medium. The WAP protocol stack is also optimised for traffic over low bandwidth networks that are subject to long latency [WAP98, WAE98].

6.1.3 Continuous Data Transmission

The main characteristic of traditional data transmission is connectivity of data. In that, the flow of data does not necessarily have to be continuous, and the data packets can be transmitted when bandwidth availability permits movement. In cases of faults or lost packets, requests for transmission of those lost or damaged datagrams are communicated to the data source, and importantly, any retransmissions that are deemed necessary are not considered as time-dependent due to the nature of the file transfer [Sta94].

In stark contrast, the communication pattern associated with continuous data transmission is particularly demanding in terms of delay constraints, since, for example, the end-to-end delay for multimedia data has to meet playback deadlines. A further problematic feature is that continuous data can be composed of audio, video, textual, and geographical streams, and as such, can contain priority constraints for the various data sub-streams. In a typical multimedia stream consisting of audio and video, it is quite evident that when faced with a lack of bandwidth, the video can be de-emphasised, while the audio stream is given precedence so as to render an effective playback. If a general purpose transmission protocol is being employed that is unable to consider information content, such a protocol will be unable to make intelligent decisions on prioritising the component data streams so as to provide a comprehensible display, or playback of information [ACIM95].

However, the situation becomes slightly more complex when, for instance, geographic data is the subject of transmission. Here, the question is what sub-streams are to be de-emphasised, and which ones should be given priority. For example, a typical map is made up of vectors, text, morphology, topography as well as a number of other data classes. The decision as to which data sub-types can be emphasised or de-emphasised can have a closer bearing on the user's immediate circumstances that typical audio-video data. For instance, if a person is lost then road, building and text data can be more important. In another situation, topography and morphology might be more important.

Thus, it is highly important that constraints with respect to continuous data sub-streams are defined, and moreover, that these constraints are respected. This is essential for the perceived quality of the playback, as well as to the overall understanding of the information that is being transmitted. In this regard, a serious drawback that can manifest itself here depends on the transmission protocol that is being used. In JMAM the task of disassembling and synchronisation of continuous data streams will be implemented at the middleware layer (see section 5.2.7).

6.1.4 Data Management

The hybrid communicator category of devices that are the focus of this work are small portable devices that are faced by limited power and memory resources. These observations suggest that mobile devices should employ passive listening strategy in communications with the server. In this schema the server acts as the data and application warehouse, and performs all high power and memory related functions in pushing or broadcasting the data to the doorstep of the terminal, whose only role is to await the arrival of the service and read the data.

Accordingly, the data management scheme that is to be employed is based on a hybrid data delivery scheme similar to that proposed in [AFZ97]. In this data management approach, the intent is to combine the best elements of the pull and push data delivery mechanisms in a unified and optimal manner. The pull data model is based on the request/response paradigm, whereby the client sends a request to the server, which then provides the requested information synchronously or asynchronously. In contrast, the push model is based on the publish/subscribe/distribute paradigm, whereby the client subscribes to the desired information that is offered by the server. After this subscription, the desired information is pushed to the client on a specified regular basis [AFZ97, SuTa97, Mar99]

The hybrid data delivery strategy is founded on two main principles, which are inherent to mobile computing. In the first instance, communication autonomy must be taken into consideration in the mobile environment, and secondly on bandwidth conservation and maximisation. Communication autonomy is handled in a way that allows that server to optimise the broadcast data delivery strategy by placing data that is intended for disconnected clients into a queue. When the client reconnects to the network, these queued items may be delivered automatically or by the specific request of the client.

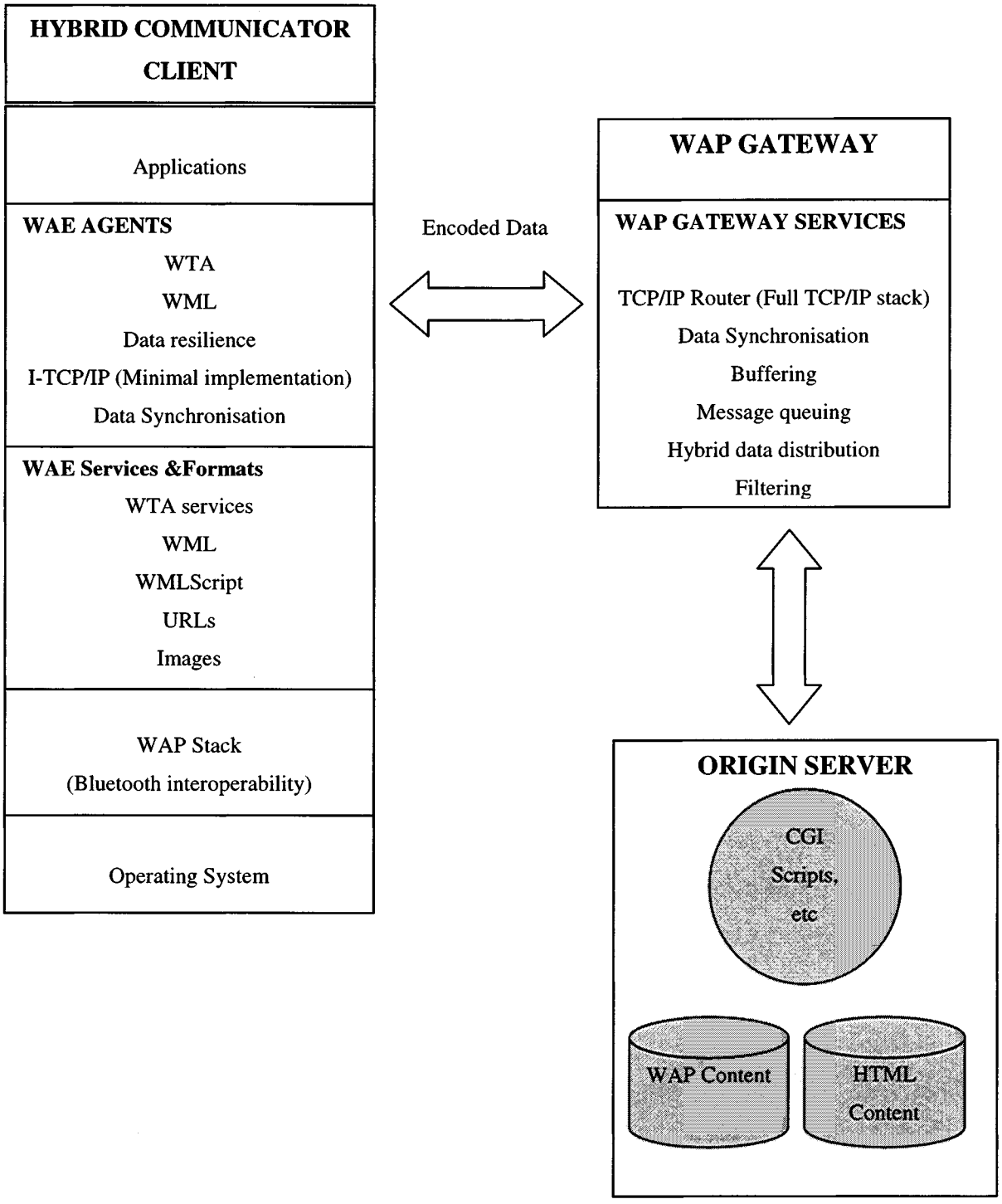
Bandwidth conservation and maximisation is handled by configuring the hybrid scheme in such a manner that the data which is stored at the server is divided into two categories. One data category is made up of broadcast items, which are demanded frequently by a high proportion of clients. Data that fits into this category are items which are not updated frequently, such as who has pole position for the coming Formula 1 race, or multimedia clips of an interview with the driver. The second category is made up of data items, which are subjected to a higher update pattern, and as such are delivered only by specific request to individual clients.

6.2 JMAM LOGICAL ARCHITECTURE

The following Figure 6.1 illustrates the logical architecture of the JMAM software architecture. The system is implemented in a WAP environment. However, Figure 6.1 is a simplified illustration of the architecture, and so it neglects some critical aspects of the WAP specification, but which are not needed in this context. For example, the content encoding and decoding mechanisms at both client

and gateway are not depicted in this logical architecture, but nevertheless, are an integral and critical part of the entire implementation.

FIGURE 6.1: JMAM LOGICAL ARCHITECTURE



6.3 SUMMARY

The principle of the software architecture, as depicted in Figure 6.1, is to balance the oftentimes conflicting forces of the limitations of hybrid communicator devices, as discussed in earlier chapters, against the current need for continuous data applications on such devices. A further challenge in this respect is that these continuous data applications are expected to function in a reliable and credible manner. For example, a typical situation that could arise in the present information society age could involve a person needing to download a map, review multimedia news clips, or access and enact a query on a remote database. All of these tasks will need to be accomplished on the pocket-sized device with which this user is equipped

These application requirements are some of the main driving forces in the diffusion of mobile computing. Accordingly, ever effort must be made to address the data management, and other challenges that are posed, through bold and ambitious software architectures. In terms of the size of the JMAM middleware, it is estimated that the various agents would need approximately 500 kb of memory.

The following chapter will conclude this work, and provide some insights into technological trends that will affect the future of mobile computing. These technological trends are mainly heading towards the provision of higher bandwidth, and in GSM include such developments as High Speed Circuit Switched data (HSCSD), General Packet Radio Service, Enhanced Data Rates for GSM, and ultimately Universal Mobile Telecommunications System.

7 CONCLUSION

The focus of the work conducted in the preceding chapters was on examining and analysing software architecture for mobile computing in light of hybrid communicator type devices, and in doing so, demonstrated a strong disposition towards GSM technology. There was also a constructive aspect of the work that was concerned with the development of a practical software architecture, which was referred to as the JMAM (Jyväskylä Mobile Architecture for Multimedia Data). These tasks were infused with a significant degree of complexity, which was centered mainly around the resource deficiency of the hardware platform, as well as the problems of wireless data transmission. Furthermore, the task was made more complex by the requirement to process continuous data, such as multimedia and geographic data.

In the process of unravelling the research agenda of the work, the aim was to develop a logical and structured presentation of the problem domain, and to evolve pertinent solutions to the specified research problem. Thus, the work commenced in the introductory chapter with a general outline of the mobile computing domain, which was subsequently distilled into the specific research problem of software architecture. In addition, definitions were also provided of important terms. For instance, the term software architecture was discussed at length, and was placed within the context of the research agenda. Software architecture was defined to be a representation of the global structure of a system, which could be depicted as a composition of the interacting parts. Furthermore, the role of software architecture is related more to the development of broad solutions than to specific problems.

In order to comprehensively tackle the broad research agenda it was necessary to dissect it into a number of sub-issues. The first of these sub-issues was concerned with the environmental and technological challenges that restrict the scope and applicability of mobile computing. These challenges to mobile computing were the focus of chapter 2, and were identified as stemming from wireless communication, mobility, and portability. However, in the main, the purpose of chapter 2 was not intended to provide a forum for solutions to those identified challenges. That task was left until a later stage, except in those challenges which did not have a direct bearing on the research problem.

The second sub-problem was concerned with the impact of mobile system architectures on software architecture, and the criteria used in selecting a particular system architecture to form the logical architecture of interaction between a mobile client a the fixed network. This problem was treated in chapter 3, where the mobile client-server model and all of its variations were scrutinised and discussed in detail. Other mobile system architectures were also evaluated. These other system architectures included the mobile agent architecture, as well as the hybrid architectures, which represent attempts to combine the healthier aspects of the two principal system architectures to form more robust system platforms.

The treatment of the mobile system architectures revealed that the mobile client-agent-server architecture was most suited for optimising the interaction pattern involving the hybrid communicator device category that was being catered for in this work. This is because the mobile client-agent-server architecture is highly applicable to lightweight devices. Moreover, it provides a robust and flexible platform for confronting such issues as continuous data through compression/decompression possibilities, limited disconnected operation, and the complexity that is introduced by the c-autonomous nature of mobile devices.

The treatment of WAP and Bluetooth in chapter 4 provided the opportunity to focus on new technologies that offer powerful solutions to many of the ailments that are involved in the mobile interaction pattern between weak clients and fixed networks. WAP is an open technology that provides a common standard for the way in which information is delivered over wireless networks to weak devices. To achieve this end, the WAP logical architecture is founded on the client-agent-server paradigm. However, the WAP architecture is a flexible platform that is capable of incorporating other communication approaches, including hybrid approaches.

Like WAP, Bluetooth represents another open technological standard that allows for the seamless and invisible connection of devices, which both contain the Bluetooth chip, and the necessary supporting software. An interesting and positive development between these two technologies is the planned merging of the protocol stacks that would allow Bluetooth to operate over the WAP stack. This development will be a significant step towards the simplification and reduction of the protocol configuration.

A major focus of this thesis was on identifying pertinent data management solutions that pertain to both traditional as well as continuous data. These questions were the subject of chapters 5 and 6. The focus of chapter 5 was on the structure of mobile software, and here it was revealed that the mobile middleware layer plays a major role in implementing data management strategies. Thus, at this layer services such as application dependent disconnected operation, compression and decompression techniques, and continuous data synchronisation were implemented in the JMAM middleware. These services were feasible data management services in the context of the weak hardware platform that was being catered for in this thesis.

In terms of the management of continuous data, it was suggested that a feasible strategy would be to define synchronisation parameters for each individual sub-stream. This strategy is based on a heuristic that is aimed at providing the maximum quality viewing or playback of a transmission involving continuous data. Thus, for example, in a multimedia data transmission, it would be possible to configure the system such that the audio sub-stream is given precedence to video sub-stream when the link conditions are unfavourable. Likewise, for geographic, there would be parameters that define which data sub-streams are afforded priority depending on link conditions, and the user's needs.

In addition, chapter 5 presented a solution that would de-couple the terminal's physical interface from that of the available network services. This was one of the main goals of this thesis, and the strategy that was posited was the implementation of a service profile at the middleware layer. This profile is intended to contain user specific information about authorised network services. A major benefit of this approach is that operators will be more empowered to develop a fuller range of services that can be targeted at specific user group. Moreover, by locating this service management agent at the middleware, it was intended this would make the profile more amenable to online update by the operator.

Chapter 6 was the forum for collating the findings of the earlier work into a unified software architecture solution for hybrid communicator devices. The system was developed in a WAP environment, and the communication architecture was founded on the client-agent-server paradigm. Chapter 6 also refined and extended the data management approach. Here, it was advanced that the global data management strategy would be based on a hybrid model of both the pull and push data management strategies. In light the deficiencies of the hardware platform, as well of the wireless

medium, this hybrid model offers the optimum approach. In this approach, data can be pushed to the clients on a regular basis. However, in instances where the data is seldomly requested by a majority of clients, then individual clients can opt to pull this data. Such a system results in maximum use of the available bandwidth.

The implications of the preceding work for mobile computing are becoming more relevant with each passing day. The market for handheld devices that can carry voice traffic as well as data traffic is steadily increasing. This market is growing on an international scale, and companies such as Nokia are actively increasing and fine-tuning their market offerings of these dual capability devices, through such products as the Nokia 7110, and the 9110 communicator [Par99].

7.1 FUTURE TRENDS

It is estimated that in the longer run the resource gaps that currently exists between hybrid communicator devices and their data centric PDA counterparts will eventually cease to exist. Indeed, the first step in lessening this gap has already been taken by the partnership between Nokia and Palm, whereby Nokia intends to implement the Palm OS on a new range of pen-based devices. However, in the meantime there is a crucial need to develop software architectures, such as the one proposed herein, that are capable of providing efficient interaction between the mobile client and the fixed network.

In addition, other technological developments are currently in progress, which are aimed at providing higher bandwidth services. In terms of GSM technology, such developments include High Speed Circuit Switched data (HSCSD), General Packet Radio Service, Enhanced Data Rates for GSM, and ultimately Universal Mobile Telecommunications System. However, the act of increasing the level of resources at the mobile terminal, or of increasing the level of bandwidth, are developments that will have no impact on the software architecture of the underlying applications.

The software structure that has been presented in this thesis has been an ambitious one in light of the limitations of the range of devices that it has been designed to accommodate. However, the main focus of this thesis is on the future, and what the future has in store is faster access rates, more robust devices in terms of memory and processor power, and significantly, multimedia, geographic and other bulky application demands. Indeed, given the rate of development in the

telecommunications and computing domains, the future as anticipated by this work is actually not that far in the offing.

REFERENCES

[AB99] Aline Baggio's web page on mobile computing links at URL address: http://www..sor.inria.fr/~aline/mobile/(February 1999).

[ACD94] Ahmad T. et al, The DIANA Approach to Mobile Computing in MOBIDATA (Interactive Journal of Mobile Computing) Vol. 1, No. 1, 1994.

[ACIM95] Alonso R. et al, Managing video data in a Mobile Environment, SIGMOD Record, Vol. 24, No. 4, 1995, pp 28 – 33.

[ACKR98] Agrawal P. et al , Battery Power Sensitive Video Processing in Wireless Networks in 9th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 1998, Vol. 1, pp 116 – 120.

[AFZ95] Swarup A. et al, Balancing Push and Pull for Data Broadcast, SIGMOD Record (ACM Special Interest Group on Management of Data), 26(2), 1997 pp. 183 – 194. 1997

[Agr98] Agrawal P., Energy Efficient Protocols for Wireless Systems in 9th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 1998, Vol. 2, pp 564 – 569.

[AloK93] Alonso R. and Korth H. F., Database System Issues in Nomadic Computing, SIGMON Record, 1998, pp 388 – 392.

[Ara99] The Ara Platform for Mobile Agents, 1999, at URL address: http://www.uni-kl.de/AG-Nehmer/Projekte/Ara/index_e.html

[ARM99] ARM Company Milestones at URL Address: http://www.arm.com/coInfo/CoBackground/milestones.html

[BaBa95] Bakre A., Badrinath B. R., I-TCP: Indirect TCP for Mobile Hosts, Proceedings of the 15th International Conference on Distributed Computing Systems, 1995, Pp 136 – 143.

[Bad96] Badrinath B. R., Designing Distributed Algorithms for Mobile Computing Networks, Computer Communications, Vol. 19, No. 4, 1996.

[Bahl98] Bahl P., Supporting Digital Video in a managed Wireless Network, IEEE Communications Magazine, Vol. 36 6, June 1998, pp 94 – 102.

[Bah98] ARMAP – An Energy Conserving Protocol for Wireless Multimedia Communications, in 9th IEEE International Symposium on Personal Indoor Mobile and Radio Communications (PIMRC), Vol. 2, Pp 575 – 580, 1998.

[BaP97] Badrinath B. R., Phatak S., Database Server Organisation for handling Mobile Clients, Department of Computer science, Rutgers University, 1997.

[BBC99] British Broadcasting Corporation, Interview with Scott McNealy from the World Economic Forum in Geneva, Switzerland, January 1999.

[Blue99] Bluetooth Product Specifications at URL address: http://www.bluetooth.com/document/default.asp?page=overview

[BLV97] De By R. A., Klas K., Veijalainen J., Transaction Management Support for Co-operative Applications, Kluwer Academic Publishers, Boston, 1997.

[BoDe96] Boutros B. S., Desai B. C., A Two-Phase Commit Protocol and its Performance in Proceedings of the 7th International Conference on Database and Expert Systems Applications, 1996, Pp 100 – 105.

[BPS99] Barelos D., Pitoura E. And Samaras G., Mobile Agent Procedures: Metacomputing in Java, in Proceedings of the 19th International Conference on Distributed Computing Systems Workshops on Electronic Commerce and Web-based Applications/Middleware, 1999, Pp 90 – 95.

[BuB96] Busbach U. and Bäcker A., DocMan: A Document Management System for Co-operation Support, Hawaii international Conference on Systems Sciences, 1996.

[CGH95] Chess D. et al, Itinerant Agents for Mobile Computing, Journal IEEE personal Communications, Vol. 2, No. 5, October 1995, pp 34 – 49.

[CSAK98] Chen J-C. Et al, A Comparison of MAC Protocols for Wireless Local Networks Based on Battery Power Consumption, in 17th Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM), Vol. 1, Pp 150 – 157, 1998.

[Dis98a] Mobile Value-added Services: A Revenue Bonanza for Operators in DISCOVERY - Nokia's Telecommunications Magazine, Vol. 48, Fourth Quarter 1998, pp 26 – 33.

[Dis98b] The Nokia 9110 Communicator in DISCOVERY – Nokia's Telecommunications Magazine, Vol. 47, Second/Third Quarter 1998, pp 54 – 60.

[Dis99b] A New Epoch Dawns for Wireless SYMBIAN in DISCOVERY – Nokia's Telecommunications Magazine, Vol. 49, June 1999, pp 24 – 27.

[Dis99c] Solving the Wireless Data Puzzle with Nokia Total Connectivity in DISCOVERY – Nokia's Telecommunications Magazine, Vol. 49, June 1999, pp 12 – 14.

[Dis99d] Visa, Nokia and MeritaNordbanken group to Pilot Mobile Payment in DISCOVERY – Nokia's Telecommunications Magazine, Vol. 49, June 1999, pp 23.

[Dis99e] Nokia 7110 Introduces World's First Media Phone in DISCOVERY – Nokia's Telecommunications Magazine, Vol. 49, June 1999, pp 20 – 21.

[DFWB98] Davies N. et al, L^2imbo: A Distributed Systems Platform for Mobile Computing, Mobile Networks and Applications, No. 3, 1998, pp 143 – 156.

[DuH95] Dunham M. H. and Helal A., Mobile Computing and Databases: Anything New?, SIGMON Record, Vol. 24, No. 4, 1995, pp 5 – 9.

[Earl89] Earl M. J., Management Strategies for Information Technology, Prentice Hall Europe, 1989.

[Eggl99] Eggletor S., The Mobile Business in Telecommunications, August 1999, Pp 88 – 89.

[EJF95] Elmagarmid A. et al, Wireless Client/Server Computing for Personal Information Services and Applications, SIGMOD Record, Vol. 24, No. 4, 1995, pp 16 – 21.

[FoZ94]] Forman G., Zahorjan J., The Challenges of Mobile Computing, IEEE Computer, Vol. 27, No. 4, April 1994, pp 38 – 47.

[FPLV98] Francis J. C. et al, Project EXODUS: Experimental Mobile Multimedia Deployment in ATM Networks, Mobile Networks and Applications 3, 1998, pp 61 – 72.

[FRGH99] Fasbender A. et al, Any network, Any Terminal, Anywhere, IEEE Personal Communications, Vol. 62, April 1999, pp 22 – 30.

[Geos99] GEOS Operating System Specifications at URL address: http://www.geoworks.com/os/architecture.html

[HaAg97] Haas Z. J., Agrawal P., Mobile-TCP: An Asymmetric Transport Protocol Design for Mobile Systems, Proceedings of the IEEE International Conference on Communications (ICC), 1997, Vol. 2, Pp 1054 – 1058.

[HBC94] Heineman, G. T. et al, Emerging technologies that support a software process life cycle, IBM Systems Journal 33 (3), 1994, pp 501 – 529.

[HNIJA99] Haartsen J. et al, Bluetooth: Visions, Goals, and Architecture, Mobile Computing and Communications Review, Vol. 2, Number 4, 1999, pp 38 – 45.

[HoC98] Holley K., Costello T., The Evolution of GSM Data Towards UMTS, GSM World Congress, France, February 1998, and GSM Data Today, 1998.

[HSL98] Housel B., Samaras G., Lindquist B., WebExpress: A Client/Intercept based system for optimising Web browsing in a wireless environment, Mobile Networks and Applications, No. 3, 1998, pp 419 – 431.

[Hum89] Humphrey, W., Managing the Software Process, Addison-Wesley Publishing Company, 1989.

[ImBa92] Imielinski T., Badrinath B. R., Replication and Mobility, Second Workshop on the Management of replicated Data, 1992, Pp 9 - 12.

[ImBa93] Imielinski T., Badrinath B. R., Data Management for Mobile Computing, SIGMON Record, Vol. 22, No. 1, 1993, pp 15 – 20.

[IMM98] Iera A., Marano S., Molinaro A., A Layered Protocol Architecture for Multi-media Wireless-PCS Networks, Mobile Networks and Applications, Vol. 3, 1998, pp 73 – 87.

[JaKe98] Jagannathan S. and Kelsey R., On the Interaction between Mobile Processes and Objects, in Proceedings of the 7th Heterogeneous Computing Workshop, 1998, Pp 163 – 170.

[Jen85] Jenkins A. M., Research Methodologies and MIS Research, Research Methods in Information Systems, Elsevier Science Publishers B.V, 1985.

[Jer97] Jerome M., Airwar, Byte Magazine, August 1997.

[KaS99] kassler A., Schulthess P., An End-to-End Quality of service Management Architecture for Wireless ATM Networks in Proceedings of the 32nd Hawaii international Conference on System sciences, 1999 pp 10.

[Kat95] Katz R., Adaptation and Mobility in Wireless Information Systems, IEEE Personal Communications Magazine, Vol. 1, No. 1, 1995, pp 6 – 17.

[KDHR97] Keller A. M., et al, Zippering: Managing Intermittent Connectivity in DIANA, Mobile Networks and Applications, 1997, Vol. 2, Pp 357 – 364.

[KiS92] Kistler K. K. and Satyanarayanan M., Disconnected Operation in the Coda File System, ACM Transactions on Computer Systems, 10 (1) 1992, pp 3 – 25.

[KiZi97] Kiniry J. and Zimmerman D., A Hands-On Look at Java Mobile Agents, in IEEE Internet Computing, July-August 1997, Vol. 1, Issue 4, Pp 21 – 30.

[KPM98] Kreller B. et al, UMTS: A Middleware Architecture and Mobile API Approach in IEEE Personal Communications, April 1998, Vol. 52,.pp 32 – 38.

[Kon98] könönen I., et al, PC:N Pienet Perilliset, Mikro PC, No. 12, August 1998, pp 40 – 49.

[Law99] Lawton G., Vendors Battle Over Mobile OS Market, IEEE Computer, February 1999 Issue, pp 13 – 15.

[Lau94] Laudon K. C., Laudon J., Management Information Systems, Macmillan College Publishing Company, Inc., 1994.

[LDG96] Lancki B., Dixit A., Gupta V., Mobile-IP: Supporting Transparent Migration on the Internet in Linux Journal, August 1996.

[LeC98] Lee K. S., Chin Y. H., A New Replication Strategy for Unforeseeable Disconnection under Agent-Based Mobile Computing System, Proceedings of Parallel and Distributed Systems, 1998, Pp 164 – 171.

[LiH98] Lim J. B., Hurson A. R., Data Duplication and Consistency in a Mobile, Multi-Database Environment, Proceedings of the Parallel and Distributed Systems, 1998, Pp 50 – 58.

[Mar99] Martin-Flatin J. P., Push Vs Pull in Web-Based Network Management, Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management, 1999, Pp 3 – 18.

[May95] Mayer J., Wireless Stretches LANs, Computer Design, February 1995.

[Micr99] Microsoft Windows CE Specifications at URL address: http://msdn.microsoft.com/cetools/platform/development.as

[MeJ97] Mellor S. J. and Johnson R., Why Explore Object Methods, Patterns, and Architectures?, IEEE Software, , January 1997, Vol. 14 1, pp 27 – 29.

[MES95] Mummert L. B., Ebling M. and Satyanarayanan M., Exploiting weak Connectivity for Mobile File Access, Proceedings of the 15th ACM Symposium on Operating Systems Principles, December 1995.

[MER99] MeritaNordbanken joins the WAP Forum to develop banking services for mobile phones in Merita News at URL address: http://www.merita.fi/e/Merita/sijoita/uutta/990705.STM

[MKMG97] Monroe R. T. et al, Architectural Styles, Design Patterns, and Objects,m IEEE Software, January 1997, pp 43 – 52.

[MSL99] Mustonen J., Sulanto M., and Lihavainen S., Pienikokoiset ja Hyvätapaiset, Mikro PC, No. 4, March 1999, pp 38 – 43.

[MPEG99] International Organisation for Standardisation, Coding of Moving Pictures and Audio, ISO/IEC JTC1/SC29/WG11 MPEG99/N2724, March 1999 at URL address: http://drogo.cselt.stet.it/mpeg/

[NoKa99] Nokia 9110 Communicator Technical Specifications for 9110 Communicator, , March 1999 at URL address: http://www.forum.nokia.com/products/communicators/9110/9110specs.html.

[NoKb99] Nokia Technical Specifications for 7110, , March 1999 at URL address: http://www.nokia.com/phones/7110/phone/specifications.html

[Nokia98] Nokia Networks White Paper, High Speed Circuit Switched Data, Nokia's vision for a service platform supporting circuit switched applications, at URL address: http://www.nokia.com

[Nokia99] Nokia Networks White Paper, Wireless Data Evolution, Nokia's vision for wireless data in GSM networks today and tomorrow, at URL address: http://www.nokia.com

[NoS95] Noble B. D. and Satyanarayanan M., A Research Status report on Adaptation for Mobile Data Access, SIGMOD Record, Vol. 24, No. 4, 1995, pp 10 – 15.

[Nor97] Norman J. T., Of Speeds and Internet Access, Andrew Seybold's Outlook on Communications and Computing, Vol. 15, No. 6, 1997, pp 33 – 37.

[Ojan98] Ojanperä T., Perspectives on 3G Development, in 9th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Vol. 1, Pp 405 – 409, 1998.

[Ora99] Oracle Mobile Agents White Paper at URL address, 1999: http://www.oracle.com/mobile/olite/html/omawp.html

[Os99] Microwave Os-9 Operating System Specifications at URL address, 1999: http://www.microwave.com/ProductsServices/DataSheets/enhanced_os-9_for_s.html

[Palm99] Palm Operating System Specifications at URL address http://oasis.palm.com/devzone/

[Par99] Parkes S., Mobiles Made Simple, Newsweek Magazine, September 1999.

[Pere98] Pereira J. M., Third Generation: An Unified Architecture to Provide Mobile Multimedia, in 9th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), 1998.

[Pek98] Pekkola S., Aspects of Combining Live-Video and Virtual Reality, Masters' Thesis, University of Jyväskylä, Department of Mathematics, 1998.

[Per98] Perkins C. E., Mobile Networking in the Internet, Mobile Networks and Applications, Vol. 3, 1998, pp 319 – 334.

[Perk98] Perkins C. E., Mobile Networking through Mobile IP, IEEE Internet Computing, January – February 1998, Vol. 2, Pp 58 – 69.

[PiB94] Pitoura E., Bhargava B., Building information Systems for Mobile Environments in proceedings of the 3rd International Conference on Information and Knowledge Management, Gaithersburg, MD, November 1994, pp 371 – 378.

[PiB95] Pitoura E., Bhargava B., A Framework for Providing Consistent and Recoverable Agent-Based Access to Heterogeneous Mobile Databases, SIGMOD Record, Vol. 24 No. 3, September 1995, pp 44 – 49.

[PiSa97] Pitoura E., Samaras G., Data Management for Mobile Computing, Kluwer Academic Publishers, 1997.

[Pre94] Pressman R. S., Software Engineering: A Practitioner's Approach, McGraw-Hill Book Company, third Edition, European Adaptation, 1994.

[PRW98] La Porta T. F. et al, Experiences with Network based User Agents for Mobile Applications, Mobile Networks and Applications 3, 1998, pp 123 – 141.

[PSP99] Papastavrou S., Samaras G., Pitoura E., Mobile Agents for WWW Distributed Database Access, Proceedings of the 15th International Conference on Data Engineering, March 1999, (ICDE 99), pp 228 - 237.

[Ray98] Raychaudhuri D., Current Topics in Wireless & Mobile ATM Networks: QoS control, IP support and Legacy Service Integration in Proceedings of The 9th IEEE International Symposium on Personal, Indoor and Mobile Communications, 1998, Vol. 1, pp 38 - 44.

[RaSt98] Ramanathan R. and Steenstrup M., Hierarchically-Organised, Multihop Mobile Networks for Quality-of-Service Support, in Mobile Networks and Applications, Vol. 3, Pp 101 – 119, 1998.

[RS98] Rossi S. and Sillander T., Four Fundamental Software Process Modelling Principles: The case of Nokia Telecommunications, Masters' Thesis, University of Jyväskylä, Department of Computer Science and Information Systems, 1998.

[SaP98] Samaras G., Pitoura E., Computational Models for Wireless and Mobile Environments, Technical report No. 98-9, University of Ioannina, march 1998.

[SB98] Storz W., Beling G., Transmitting time critical data over heterogeneous sub-networks using standardised protocols, Mobile Networks and Applications 2, 1997, pp 243 – 249.

[Seyb98] Seybold, A., The Convergence of Communications and Computing, Andrew Seybold's Outlook, May 1998, at URL address: http://www.outlook.com/articles/may98article1.html

[Sjos99] Sjöström M., Massamuistit: Flash-Korttiin Kunnon Kopiosuojaus, in Mikro PC No. 16, 28 October 1999, Pp 10.

[Sta94] Stallings W., Data and Computer Communications, fourth Edition, Macmillan Publishing Company, 1994.

[SuTa97] Su Chi-Jiun, Tassiulas L., Broadcast Scheduling for Information Distribution, Proceedings of the 16th Annual Joint Conference of the IEEE Computer and

Communications Societies (Driving the Information Revolution) 1997, Vol. 1, Pp 109 – 117.

[SUN99] Sun Microsystems's web page, February 1999, at URL address: http://www.sun.com/sunergy/untoldstory.html

[SUNa99] Sun Ultra Sparc Workstation Product Specifications at URL address: http://www.sun.com/desktop/products/index.html#ultrasparcworkstations

[SUNb99] Sun Workstation Product Specifications at URL address: http://www.sun.com/nc/whitepapers/javastation/javast_ch4.html

[Sym99] EPOC Operating System Specifications at URL address: http://www.symbian.com/epoc/papers/e5oall/e5oall.html

[TDB97] Tarau P., Dahl V. and De Bosschere K., A Logic Programming Infrastructure for Remote Execution, Mobile Code and Agents, in Proceedings of the 6th IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1997, Pp 106 – 111.

[UMTS97] UMTS Forum – A regulatory Framework for UMTS, February 1999- at URL address: http://umts-forum.org/

[Vei89] Viejalainen J., Transaction Concepts in Autonomous Database Environments (Ph.D. thesis), GMD-Bericht No. 183, Oldenbourg Verlag, Munich, Germany, April 1990.

[Vei99] Veijalainen J., Mobile Computing Course slides (TJT L60) held during the spring semester at the University of Jyväskylä at URL address: http://www.cs.jyu.fi/~veijalai/mobilecomputing99/mc99.html.

[WAP98] WAP Architecture– Wireless Application Protocol Architecture Specification, Version 30, April 1998, at URL address: http://www.wapforum.org/.

[WAP99] WAP White Paper - Wireless Internet Today, June 1999, at URL: http://www.wapforum.com/what/WAP_white_pages.pdf

[WAP99a] Wireless Application Protocol – The Corporate Perspective, White Paper, March 1999, at URL address: URL:http://www.nokia.com/corporate/wap/future.html

[WAE98] WAP WAE– Wireless Application Protocol Wireless Application Environment Overview, Version 30, April 1998, at URL address: http://www.wapforum.org/.

[WDP98] WAP WDP– Wireless Datagram Protocol, Version 30, April 1998, at URL address: http://www.wapforum.org/.

[WeB95] Welling G., Badrinath B. R., Mobjects: Programming Support for Environment Directed Application Policies in Mobile Computing, ECOOP'95 Workshop on Mobility and Replication, August 1995.

[WeBa97] Welling G., Badrinath B. R., A Framework for Environment Aware Mobile Applications in Proceedings of the 17th International Conference on Data communications Systems, May 1997.

[WSP98] Wireless Session Protocol – Wireless Session Protocol Specification, April 1998, at URL address: http://www.wapforum.org/.

[WTLS98] Wireless Transport Layer Security Specification, April 1998, at URL address: http://www.wapforum.org/.

[WTP98] Wireless Transaction Protocol – Wireless Transaction Protocol Specification, April 1998, at URL address: http://www.wapforum.org/.

[WWDS94] Weiser M. et al, Scheduling for Reduced CPU Energy in Proceedings of Symposium on Operating Systems Design and Implementation, 1994, pp 13 – 23.

[YDS95] Yao, F., Demers A., Shenker S., A Scheduling Model for Reduced CPU Energy, Foundation of Computer Science, Proceedings of 36th Annual Symposium, 1995, pp 374 – 382.

[YHC96] Yoon Y., Han M., Cho J., Real-Time Commit Protocol for Distributed Database Systems in Proceedings of the 2nd IEEE International Conference on Engineering of Complex Computer Systems, 1996, Pp 221 – 225.

[YWT98] Yang C., Wu K., Tseng C., Support an Efficient Connection for Mobile IP, Proceedings of the Ninth International Workshop on Database and Expert Systems Applications, 1998, Pp 514 – 519.