KUNGL
TEKNISKA
HÖGSKOLAN

A 3G Convergence Strategy for Mobile Business Middleware Solutions

Applications and Implications

Master's Thesis

Fredrik Hacklin
fredrik@hacklin.com

Royal Institute of Technology
Faculty of Computer Science, Electrical Engineering and Engineering Physics
Department of Microelectronics and Information Technology
Stockholm, Sweden

Helsinki University of Technology
Department of Industrial Engineering and Management
Institute of Strategy and International Business
Helsinki, Finland

Smartner Information Systems Ltd.
Helsinki, Finland

Helsinki, 20th September 2001

Supervisors: Prof. Dr. Gerald Q. Maguire Jr.,
Prof. Dr. Erkko Autio
Instructor: M.Sc. Mika K Uusitalo

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY

smartner

This thesis is submitted to the Faculty of Computer Science, Electrical Engineering and Engineering Physics at the Royal Institute of Technology, *Kungliga Tekniska Högskolan (KTH)*, Stockholm, Sweden, as Master's Thesis in the area of Teleinformatics, *Examensarbete inom teleinformatik (2G1015)*, for partial fulfillment of requirements for the academic degree
Master of Science in Engineering,
M.Sc. (Eng.).
An electronic version of this thesis report as well as related material is available at
`http://www.hacklin.com/thesis`.
Typeset in LaTeX.
Version 5 (revised), 20th September 2001, Stockholm, Sweden.

Abstract

Mobile business solutions are one of the most attractive market segments of mobile information services. The third generation of mobile communication systems (3G) will be a significant step forward in the convergence of telecommunications and datacommunications industries. More specifically, the convergence of mobile technologies and the Internet allows compelling possibilities for future applications and solutions. However, most current mobile businesses and mobile application and solution providers are rather contributing to the process of convergence; many current ideas and solutions are based on the restrictions of existing mobile networks combined with Internet-based services. In the future, when mobile networks and the Internet have merged, it will no longer be possible to create revenue with these types of solutions.

One concrete solution is the mobile middleware concept, bridging the mobile technologies and Internet world. This Master's thesis studies the middleware concept for providing business applications in the light of 3G, making strategic recommendations to a provider of these kinds of services. A comprehensive discussion about the developments after 3G is introduced. Alternative solutions are presented and some strategic implications are introduced. The implications are motivated by an industry survey, carried out within this project. The topic of over-the-air data synchronization is discussed as an example for interim middleware. Mobile computing file system issues are seen as an interesting opportunity for business applications. The possibility of remote desktop screen access is studied, and measurements proving its feasability for hosted wireless application service provision are made. Emerging mobile Java technologies are discussed as an efficient platform for providing ubiquitous, device independent end-to-end solutions. As one of the recommended strategies, this thesis introduces the concept of hybrid thickness client applications as a feasible solution for migrating from current middleware solutions to an (uncertain) future of native, thick terminal applications, within a scope of two years. Based on this concept, a prototype for a 3G smartphone application was developed as an example. A set of possible strategic scenarios is presented and discussed. This thesis also discusses operator differentiation and business solutions in an all-IP based world.

3G networks and handset devices will introduce a large number of new applications and business opportunities, but such a change will also introduce new challenges and risks. The migration challenge is being illustrated in the case of Smartner, a mobile middleware solution provider focusing on business applications. As shown by this case, compared to current enabling solutions, a major shift in technologies is seen as needed, in order to maintain long-term success.

Keywords: 3G, mobile business, middleware, front-end, operator-hosted, strategy, convergence, migration, XML, hybrid thickness client application, SyncML, Wireless OS, UMTS, GPRS, EPOC, Symbian, value added services, operator differentiation, competitive advantage.

Sammanfattning

Mobila affärssystem bildar ett av de mest attraktiva marknadssegment inom mobila informationstjänster. Den tredje generationens mobila kommunikationssytem (3G) kommer att bli ett viktigt steg fram mot konvergensen mellan telekommunikations- och datakommunikationsindustrin. Särskilt konvergensen som äger rum mellan mobila teknologier och Internet erbjuder utmanande möjligheter för framtida applikationer och lösningar. De flesta nuvarande företag och tjänster inom mobilbranschen kan dock snarast betraktas som ett bidrag till denna konvergens. Många av de nuvarande idéerna och lösningarna är nämligen baserade på avgränsningar och problem som uppstår vid kombination av mobila system med Internet-baserade tjänster. I framtiden, när mobila nät har vuxit ihop med Internet till en symbios, kommer det inte längre att vara möjligt att förtjäna på detta slag av lösningar.

En konkret lösning är det mobila middleware-konceptet, som bildar en logisk koppling mellan mobila teknologier och Internet-världen. Detta examensarbete studerar middleware-konceptet från en 3G-orienterad synvinkel och framför strategiska råd för företag som erbjuder detta slag av tjänster. En detaljerad diskussion om utvecklingen efter 3G presenteras. Arbetet lägger fram alternativa lösningar och strategiska implikationer deriveras. Implikationerna är motiverade bl.a. av en intervjuundersökning som utfördes i samband med detta arbete. Temat trådlös datasynkronisering diskuteras som ett exempel för provisorisk middleware. Mobila filsystem införs som en intressant möjlighet för affärsapplikationer. Diverse möjligheter för fjärrkontroll av en arbetsplatsstation studeras och mätningar bevisar deras genomförbarhet för trådlösa applikationstjänster. Framträdande mobila Java-teknologier analyseras och presenteras som ett efficient underlag för plattformoberoende end-to-end-lösningar över lag. En av de rekommenderade strategierna är baserad på det hybrida klientkonceptet, vilket presenteras som en realistisk lösning för övergången från nuvarande middleware-system till en (osäker) framtid av nativa, tjocka terminalapplikationer. Den strategiska horisonten för detta är två år. Utgående från detta koncept utvecklades en prototyp som exempel för en sådan applikation. Arbetet definerar och diskuterar dessutom diverse strategiska scenarier. Slutligen nämns problematiken om operatörernas framtida differentieringsmöjligheter och rollen av affärssystem i en fullständigt IP-baserad värld.

3G nät och terminaler kommer att skapa ett stort antal nya användningar och affärsmöjligheter, men ändringen kommer också att medföra nya utmaningar och risker. Detta illustreras med hjälp av företaget Smartner som exempel för en leverantör av mobila middleware-lösningar för affärsanvändningar. Som demonstrerat i detta fall, anses i jämförelse med nuvarande applikationslösningar en signifikant teknologisk reorientering vara nödvändig, för att bevara ett långvarigt perspektiv.

Tiivistelmä

Langattomat yrityssovellukset ovat nykyään yksi kiinnostavimmista mobiilimarkkinoiden segmenteistä. Kolmannen sukupolven (3G) mobiilit viestintäjärjestelmät tulevat olemaan merkittävä askel kohti telekommunikaatio- ja dataliikennealojen yhdistymistä (ns. konvergenssia). Itse asiassa mobiiliteknologian ja Internetin lähentyminen mahdollistaa entistä hyödyllisempien mobiilisovellusten ja -ratkaisuiden rakentamisen tulevaisuudessa. Tällä hetkellä useat mobiiliyritykset ja mobiilisovellusten tuottajat ovat kuitenkin osana tätä yhdistymisprosessia. Monet nykyiset ideat ja ratkaisut ottavat nimittäin lähtökohdakseen rajoitukset, joita nykyiset tietoliikenneverkot asettavat yhdistyessään Internet-pohjaisiin palveluihin. Tulevaisuudessa, kun mobiiliverkot ja Internet ovat yhdistyneet, ei ole enää mahdollista ansaita rahaa tällaisten perinteisten ratkaisuiden avulla.

Yksi konkreettinen ratkaisumalli perustuu mobile middleware -käsitteeseen, joka liittää yhteen mobiiliteknologian ja Internetin. Tässä diplomityössä tutkitaan middleware-käsitettä yrityssovellusten tarjoamisessa erityisesti 3G-verkoissa, ja työssä esitellään strategisia suosituksia näiden sovelluspalveluiden tarjoajille. Työssä käydään perusteellisesti läpi kolmannen sukupolven jälkeistä kehitystä. Vaihtoehtoisia ratkaisuja esitellään, ja joitakin strategisia vaikutuksia tuodaan myös esille. Vaikutuksia perustellaan tuloksilla, joita tämän projektin osana tehty kysely paljasti. Tiedon langatonta synkronisointia tarkastellaan esimerkkinä tilapäisestä middlewaresta. Mobiileihin tiedostojärjestelmiin liittyvät asiat nähdään mielenkiintoisena mahdollisuutena yrityssovelluksille. Toimistojärjestelmien etäkäyttömahdollisuuksia on tutkittu ja niiden sopivuutta langattomaan sovellustarjontaan on mitattu. Kehittyviä mobiileja Java-teknologioita pidetään tehokkaana alustana, jonka avulla voidaan tarjota kaikkialla saatavilla olevia, päätelaiteriippumattomia ratkaisuja loppuasiakkaille. Yhtenä suositelluista strategioista tämä diplomityö esittelee yksinkertaisen päätelaitesovellusmallin, jonka avulla nykyisistä middleware-ratkaisuista voidaan siirtyä tulevaisuuden kehittyneempiin päätelaiteratkaisuihin kahden vuoden sisällä. Tähän konseptiin perustuen työssä on kehitetty esimerkki 3G-älypuhelimen sovelluksesta. Lisäksi esitellään ja arvioidaan mahdollisia strategisia skenaariovaihtoehtoja. Tämä diplomityö käsittelee myös operaattoreiden differointimahdollisuuksia ja yrityssovelluksia täysin IP-pohjaisissa verkoissa.

3G-verkot ja -päätelaitteet tuovat mukanaan laajan valikoiman uusia sovelluksia ja liiketoimintamahdollisuuksia, mutta tämä muutos merkitsee myös uusia haasteita ja riskejä. Tätä haastetta kuvataan tutkimuksen esimerkkiyrityksen Smartnerin tapauksessa, joka on yrityssovelluksiin fokusoitunut mobiilien middleware-ratkaisuiden tarjoaja. Tutkimus tuo esille, miten Smartnerin nykyiset sovellukset huomioon ottaen tarvitaan valtava teknologinen suunnanmuutos pitkäaikaisen perspektiivin säilyttämiseksi.

Zusammenfassung

Mobile Geschäftslösungen bilden eines der z.Zt. attraktivsten Marktsegmente im Bereich mobiler Informationsdienste. Die dritte Generation mobiler Kommunikationssysteme (3G) bedeutet einen wesentlichen Schritt in der Konvergenz zwischen der Telekommunikations- und Datenkommunikationsindustrie. Insbesondere die Konvergenz mobiler Technologien mit dem Internet erlaubt eine Vielzahl neuer Möglichkeiten für zukünftige Anwendungen und Lösungen. Das Problem der meisten heutigen Anbieter von mobilen Diensten und sonstigen Mobilfirmen besteht jedoch darin, dass sie eher zu dem Prozess der Konvergenz beitragen, als davon profitieren. Viele heutige Ideen und Lösungen solcher Firmen basieren auf Einschränkungen, die bei der Kombination von mobilen Netzwerken mit Internet-basierten Diensten entstehen. In Zukunft, wenn mobile Netze mit dem Internet in eine Symbiose verschmolzen sind, wird es nicht mehr Möglich sein, mit solchen Lösungen ertragreich zu sein.

Eine konkrete Lösung besteht im Konzept der mobilen Middleware, ein logisches Verbindungsstück zwischen Mobiltechnologien und dem Internet. Das Ziel dieser Diplomarbeit ist es, jenes Konzept in Anbetracht von 3G Anwendungen zu untersuchen, sowie davon ausgehend strategische Empfehlungen für solcher Art Dienste zu erzeugen. Das Thema wird mit einer umfassender Diskussion der 3G-Entwicklungen sowie alternativer Lösungen eingeführt. Die Arbeit benennt strategische Auswirkungen, welche zusätzlich anhand einer nebenbei durchgeführten Industrieumfrage belegt werden. Das Thema mobiler Datensynchronisierung wird als ein Beispiel zwischenzeitlicher Middleware herangeführt. Der Aspekt mobiler Dateiverwaltungssysteme wird als eine interessante Möglichkeit für mobile Geschäftslösungen vorgestellt. Die Möglichkeit eines entfernten Komplettzugriffes der Arbeitsplatzoberfläche wird studiert, und Messungen belegen dessen Realisierbarkeit in ausgegliederten, mobilen Anwendungsdiensten. Anwachsende Java-Technologien werden als eine effiziente Platform allgegenwärtiger, plattformunabhängiger End-zu-end-Lösungen diskutiert. Als ein Teil der empfohlenen Strategien präsentiert diese Arbeit das Konzept von hybriden Softwareklienten als eine sinnvolle Lösung für einen schrittweise Übergang von aktuellen Middlewarelösungen zu einer (ungewissen) Zukunft von eigenständigen Terminalapplikationen. Der strategische Rahmen dieses Konzeptes beträgt zwei Jahre. Als Beispiel hierfür dient ein Prototyp einer 3G-Applikation, der im Rahmen dieser Arbeit entwickelt wurde. Eine Anzahl strategischer Szenarien wird ausserdem präsentiert und diskutiert. Zusätzlich streift diese Arbeit das Thema von Dienstanbieterdifferenzierung und Lösungen im allgemeinen in einer vollkommen IP-basierten Netzwerkwelt.

3G-Netzwerke und Geräte werden einerseits eine breite Anzahl neuer Anwendungen und Geschäftsmöglichkeiten mit sich bringen, aber der Wechsel wird andererseits auch neue Herausforderungen und Risiken zu Tage bringen. Dieses wird anhand des Beispiels von Smartner illustriert, ein Anbieter von mobilen Middleware-Lösungen für Geschäftsanwendungen. Wie aus diesem Fall hervorgeht, verglichen mit aktuellen Lösungen mobiler Applikationen, wird ein signifikanter Richtungswechsel in Technologien erforderlich sein, um eine langzeitige Perspektive bewahren zu können.

Résumé

Les solutions mobiles pour le monde des affaires sont un des segments les plus attractifs du marché des services d'information mobiles. La troisième génération des systèmes de communications mobiles (3G) représentera un pas avancé significatif, illustrant la convergence de la télécommunication et de l'informatique. Plus spécifiquement, la convergence des technologies mobiles avec l'Internet permettra un grand développement d'applications et de solutions futures. En fait, la plupart des fournisseurs des services mobiles contribuent au processus de la convergence. Plusieurs idées et solutions actuelles sont basées sur les restrictions des réseaux mobiles existants combinés avec des services basés sur l'Internet. À l'avenir, quand les réseaux mobiles et l'Internet se seront développés ensemble, il ne sera plus possible de profiter économiquement de ce type de solutions.

Une proposition concrète s'avère être le concept de mobile middleware, combinant les technologies mobiles avec le monde d'Internet. Cette thèse étudie le concept de mobile middleware pour fournir des applications d'affaires dans les réseaux 3G, faisant des recommandations stratégiques à un fournisseur de ce genre de services. Une synthèse complète concernant ces développements d'après 3G est présentée. Des solutions alternatives sont exposées et quelques implications stratégiques sont introduites. Les implications sont motivées par une enquête d'industrie, effectuée dans le cadre de ce projet. Par exemple, la réalisation des systèmes de fichiers mobiles est abordée comme opportunité intéressante pour des applications d'affaires. Le concept de la synchronisation sans fil est présenté comme autre exemple pour le middleware temporaire. La possibilité d'accès au bureau à distance est étudiée, et des mesures prouvent la faisabilité pour les systèmes mobiles. Les technologies offertes de Java sont introduites comme plateforme efficace pour fournir des solutions end-to-end omniprésentes et indépendantes du dispositif. Une stratégie recommandée dans cette thèse est le concept des applications hybrides du client comme solution pour la migration (entre deux ans) des solutions actuelles du middleware vers un avenir (incertain) des applications natives. Sur la base de ce concept, un prototype pour une application de smartphone 3G est dévéloppé comme un exemple. Un ensemble des scénarios possibles avec des plans d'actions stratégiques est présenté et traité.

Les réseaux 3G et les dispositifs de communication introduisent un grand ensemble de nouvelles applications et occasions commerciales. Un tel changement impliquera tout de même de nouveaux défis et prises de risques. Pour illustrer ce projet, cette thèse présente le cas de Smartner. Comparé aux solutions actuelles, une variation significative des technologies est nécessaire, pour projeter une perspective sur le long terme.

Sommario

Le soluzioni mobili per il mondo affari sono uno dei segmenti più attraenti del mercato dei servizi d'informazione mobili. La terza generazione dei sistemi di comunicazione mobili (3G) rappresenterà un passo in avanti molto significativo nella convergenza delle telecomunicazioni e dell'informatica. Più specificamente, la convergenza di tecnologie mobili e Internet permetterà un grande sviluppo per le applicazioni e le soluzioni future. La maggior parte del mondo del commercio elettronico e delle applicazioni mobili stanno contribuendo infatti al processo di convergenza. Molte idee e soluzioni attuali sono vincolate dalle limitazioni sia delle reti mobili esistenti sia dei servizi Internet. In avvenire, quando le reti mobili e Internet si saranno ancora più sviluppate e integrate, tali idee e soluzioni non saranno più in grado di offrire guadagni.

Una proposta concreta descritta in questa tesi è il concetto del mobile middleware, che getta un ponte tra tecnologie mobili e mondo Internet. Questa tesi approfondisce l'utilizzo del middleware per fornire applicazioni professionali alla luce dei sistemi 3G. Una discussione completa circa gli sviluppi del 3G è inoltre inserita. Le soluzioni alternative sono presentate e alcune implicazioni strategiche sono introdotte. Le implicazioni sono motivate da un'indagine di mercato, effettuata all'interno di questo progetto. Ad esempio, le realizzazioni di filesystem mobili per sistemi di archiviazione moderni sono viste come occasione interessante per le applicazioni professionali. Il soggetto di sincronizzazione over-the-air è discusso come un altro esempio per il middleware provvisorio. Le tecnologie mobili offerte da Java sono discusse come piattaforma efficiente per fornire ubiquità e soluzioni end-to-end indipendenti dai dispositivi. Una delle strategie suggerite in questa tesi è il concetto di applicazioni ibride pesanti sul lato del cliente come soluzione per la migrazione (entro due anni) dalle soluzioni attuali del middleware verso un futuro (incerto) di applicazioni native. Sulla base di questo concetto, un prototipo per un'applicazione di smartphone 3G è stato sviluppato come esempio. Un insieme dei piani d'azione strategici possibili è presentato e discusso.

Le reti 3G ed i dispositivi ad esse collegati introdurranno molte nuove applicazioni e occasioni di affari. Un tal cambiamento, tuttavia, introdurrà anche nuove sfide e rischi. Come esempio per questo sogetto, questa tesi presenta il caso di Smartner. Confrontato con le soluzioni correnti, un profondo cambiamento nelle tecnologie è necessario, in una prospettiva di lunga durata.

Contents

List of Figures

List of Tables

Acknowledgements

I am indebted to a number of important contibutors, some of whom are detailed below. I am grateful to Prof. Gerald Q. *"Chip"* Maguire Jr., for taking his time to supervise this Master's thesis despite his sabbatical leave. Chip has furthermore been a reliable and excellent advisor during my three years of study in Stockholm. I am grateful to Prof. Erkko Autio, for his cooperation and uncomplicated supervision. Without Erkko this thesis could not have been realized in such a special constellation. Furthermore, I am grateful to Mika K Uusitalo, for supporting me in the definition of such an interesting topic, and for allowing me a profound insight into the Nordic mobile business arena. Mika did an excellent advisor job, shared his visions, and gave me highly skilled feedback.

I would like to thank the entire Smartner team for a very rewarding time and experience—both inside and outside office hours. For initially allowing this thesis project to take place, I am especially indebted to Antti Virkkunen, Ari Backholm, Robert Rasmus, Lauri Vuornos, and Jussi Räisänen. My research work found remarkable attention, reaction, and response within the team, which contributed to my thesis progress in an important way. I wish Smartner a prosperous future.

I would like to express my appreciation to Prof. Franco Davoli for inviting me to ICC'2001, which gave me a solid overview of related wireless and mobile technologies discussed in chapter 2. I am also grateful to Peter Jantunen, Timo Smura, Mikko Tervahauta, and Matias With, for helping me in initiating the strategy formulation for the Smartner case within the Strategic Management course (*see* section 4.7). For assisting me with the measurements in appendix A, I would like to thank my brother Johan. I am also grateful to Miikka Lindgren, Jouko Nuottila, and Kari Seittenranta at Nokia, for showing strong interest, excellent technical support and cooperation. I am indebted to David Megginson for providing his open source XML parser to the public.

I wish to thank all the contributors within the scope of my survey, namely Magnus Fransson, Dr. Eckhard Geulen, Arnthor Halldorsson, Esko Hannula, Jens Hartmann, Tom Henriksson, Johan Hjelm, Anders Huge, Prof. Randy H. Katz, Jani Kelloniemi, Prof. Lye Kin Mun, Dr. Norbert Niebert, Prof. Björn Pehrson, and Jaya Shankar.

I would like to thank Mathias Rönnblom and Roland Regamey for contributing to this thesis through their academic opposition. For additional reviewing support, I am grateful to Maria Malmström-Frigo, Daniel Frigo, Nora Malin, Dr. Piergiulio Maryni, and Aino.

Finally, I owe big thanks to my parents, for their support and especially remarkable interest during the entire duration of my studies. I also wish to thank my grandmother Mila for supporting me during the writing of this thesis.

Chapter 1

Introduction

1.1 Background

1.1.1 Mobile Applications

Mobile communication technologies seem to be exploding. Already today there are more mobile devices than wired phones worldwide [244]. Wireless networks cover large parts of the globe, and their capacity is escalating constantly. With the increasing capabilities of wireless terminal devices and their support of wireless data transmission, the set of possible applications and scenarios for mobile use reach new horizons, from of which telephony only might remain one element in the mass. The use of mobile devices combined with new business models is the great challenge. However, the form and shape of future applications and solutions is still unclear. What is a *mobile* application? Does it mean to carry around your laptop, continuously being online? Does it imply the execution of tiny applications on mobile phones with thumbnail sized screens? Or is there any combination of these two?

Applying mobility issues on business solutions results in one of the most compelling market segments within mobile communications. Business users are not only early adopters of new technologies in general, but building business solutions requires ideas and applications based on solid value propositions. Whereas consumer segments can be satisfied by the Internet and Mobility *hype* to a certain extent, businesses invest in new technologies when aiming at longterm competitive advantage [186]. Adopting new technologies alone does not provide any such concrete value [179]; new solutions should offer complements to *traditional* applications for doing business.

1.1.2 Convergence

The trend of telecommunications and datacommunications converging into a symbiosis has reached a peak in the fusion of mobile technologies with the Internet [76]. Many current mobile business ideas and solutions were created during the beginning of this evolution, and can therefore be seen as *a part of* the convergence. They are based on managing restrictions given by wireless and mobile technologies, when combining them with Internet applications (*see* table 1.1). It is obvious that such solutions will loose its function, once convergence is finalized, and it will be very difficult to make money based on these decreasing restrictions alone.

1.1.3 Middleware

One centric framework within the convergence between mobile technologies and the Internet is the concept of *mobile middleware*.[1]Middleware denotes that type of bridging mechanisms, as required by the convergence problem, for providing end-to-end solutions between front-end (e.g. terminal device) and back-end (e.g. corporate information server). Mobile middleware solutions are based on the restrictions of terminal devices and communication facilities. In a world of 3G and beyond, when technology has evolved a remarkable step in this convergence—wireless terminals allow effective stand-alone applications and mobile networks allow transparent Internet Protocol (IP) access—this type of middleware becomes obsolete [72].

1.1.4 Key Questions

For mobile businesses based on this middleware approach, a technological reorientation is absolutely necessary for surviving in a convergence world of 3G and beyond. Obviously, the aspect of locally running terminal applications has to be envisioned. This reorientation, however, arises many new questions. Do companies involved in this type of solutions have to skip current solutions over night, or is there any strategy for migration? Is there any use of current middleware in the context of 3G terminal applications?

1.1.5 Paradigm Shifts

An additional dimension is given by the uncertainty regarding the future technological platform. It is not said that the next generation will be dominated by current mobile operators and device vendors alone. The evolution of small, handheld computers and wireless local area connectivity might affect the dominance in a world of 3G and beyond. Operators might discover difficulties in providing differentiated,

[1] As the number of users owning a mobile phone is increasing, more and more businesses are interested in selling and distributing their services *to* mobile phone users and *via* mobile phones.

Mobile communications networks are not like any other media. They offer data services at low bandwidth only, with many different types of mobile network technology and mobile devices, and similarly fragmented network coverage. It takes a long time and a lot of effort to support all the mobile networks and devices that are on the market, thus the question is why not to find a supplier who already can provide this kind of solutions instead of reinventing the wheel?

Mobile Middleware is deigned to solve this problem—provided a fast and effective means of letting companies offer reasonably compelling mobile services without having to go through the learning and development curves that starting from scratch would entail. As its name suggests, middleware is logically located in-between server hosts and mobile phone clients and thereby connects the Internet and other server-based platforms to mobile phone users [16].

Mobile networks	**Internet**
Voice centric	Data centric
Circuit switched	Packet switched
Proprietary/closed environment	Open environment
Europe/Asia centric	USA centric
Working charging system	Difficult to charge contents and applications

Table 1.1: Challenges *during* convergence

hosted services, if proprietary mobile networks and wireless local area networks are being merged into an all-IP based access space. Future mobile applications might be provided by current major software vendors instead.

1.1.6 Smartner

An illustrating example is given by the 1999-founded company, *Smartner Information Systems*, based in Helsinki, Finland (*see* section 4.7). Smartner's commitment is on providing applications and solutions empowering mobile business services. These services are integrated into one common, generic Mobile Application Platform (MAP). They are targeted to small-and-medium-sized enterprises (SMEs) who prefer to outsource parts of their information technology (IT) solutions rather than to implement or purchase own, customized complete solutions. Smartner's service portfolio is marketed through operators, who in turn receive support in providing differentiated services and reaching corporate and business user segments.

As addressed above, Smartner's business idea and product strategy is an example for such a contributing part of the convergence. However, a real answer to convergence and especially perspectives for a world in 3G and beyond is currently missing. The challenge is to combine the middleware-platform oriented idea with 3G specific issues and to result in a strategy and roadmap for the next generation.

1.2 Problem Definition

A lot of research on the technological possibilites, variety of future mobile applications, and general market opportunities has been conducted [131, 65, 101, 181, 150, 41, 186, 153, 62, 84, 94, 198], but there exists relatively little discussion about these key questions and paradigm shifts. An analysis of 3G technology capabilities, combined with the convergence issue, and with the role of middleware concepts, is needed.

1.3 Objectives

The goal of this study was an initial definition of a strategic orientation towards 3G and convergence for middleware based mobile business solutions. Furthermore, the research objectives consisted of three main phases: *battlefield*, *solutions*, and *strategy*. The first objective was to build a solid understanding about Smartner, the surrounding business environment, and related technologies. Based on this background, the main contribution of this thesis should consist of describing a set of technology and business environment scenarios, as well as creating and analyzing opportunities.

Applying the opportunities on concrete applications, the second objective was based on a solution analysis, and consisted in discussing the feasability of a model for data synchronization within a middleware-oriented context. As data synchronization was seen as one possible next step in the migration towards 3G, a special focus was placed onto this issue. The aspect of middleware-oriented synchronization required a profound analysis, whereas the terminal application oriented solution was exemplified with a demonstrative implementation.

Finally, the definition and study of a set of business scenarios was envisioned. Based on the discussion and analysis of these, the final result should consist in recommendations in the scope of a two year strategy.

1.4 Research Methodology and Structure

The research was initiated by a literature study, covering technologies and business environment issues related to 3G and business applications [90]. Recognizing the importance of the middleware concept issue in the 3G context, large parts of the follow-up research was derived from and focused onto this topic. Current solutions were critically analyzed in the perspective of future obsolescence (*see* section 3.2). Future, middleware-independent approaches were studied (*see* section 3.7), and several ways for eventual benefit of middleware approaches for application centric solutions were examined. A survey was performed to get an impression about the industry's view on the future of mobile middleware (*see* appendix D). In section 3.6, a feasibility study was performed based on a simple experiment and measurements.

This thesis begins with a study of 3G related technologies in chapter 2. Applying the 3G background onto applications and solutions, a set of different approaches is examined in chapter 3, together with technical feasiblity recommendations. Chapter 4 discusses the implications of 3G and its applications on business models and is the basis for strategic imperatives and recommendations, which finally are summarized in chapter 5 (*see* figure 1.1). Topics discussed in chapters 3 and 4 are mainly targeted on Smartner, but should be regarded as applicable on similar mobile middleware solutions in general.

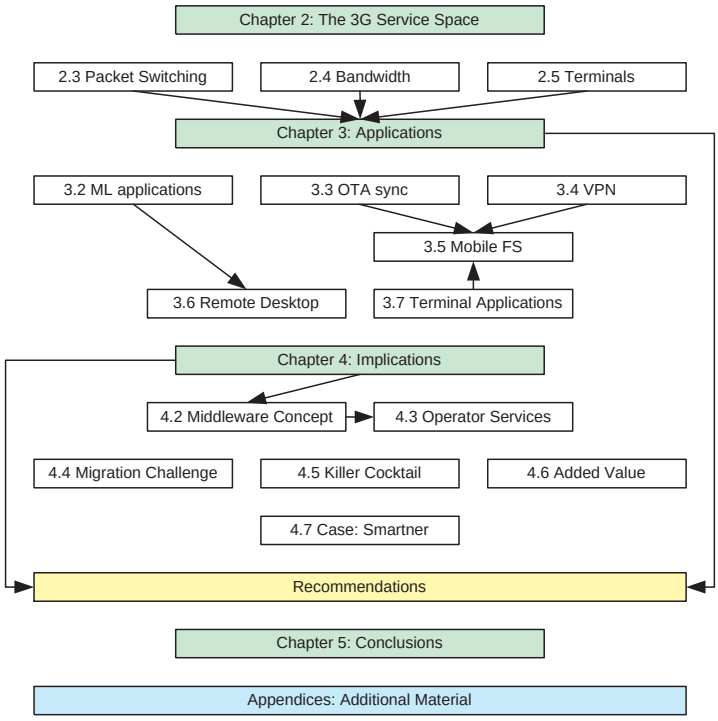

Figure 1.1: Research structure

1.5 Scope

The scope of this research is mainly technology oriented. Emerging wireless and mobile technologies were studied with an additional focus on alternative technologies. The range of potential 3G application technologies is presented in its full width, with an emphasis on a subset. Recognizing the issue of over-the-air synchronization (*see* section 3.3) as one key issue for upcoming mobile middleware solutions, the literature study was mainly focused onto this subject [90] and several issues were compared to and derived from it. Data synchronization was also addressed and illustrated from a theory background. However, expecting the emerging capabilities of future terminal devices to allow more advanced standalone applications, this aspect received high commitment (*see* section 3.7). Considering the ASP technology concept and business model as one key characteristics of current middleware based solutions (*see* section 3.2), a possibility for a future, similar technology concept was somewhat more deeply analyzed (i.e. with experiments and measurements).

The technology-based approach also defines the scope for the implication analysis and strategic recommendations (*see* chapter 4, *see* figure 1.1). However, the industry survey (*see* appendix D) contributes to the analysis independently from the technology-based study.

The scope can be summarized as current and future applications in combination with the migration challenge (from middleware-based solutions to wireless terminal applications). Special topics regarding the potential of 3G success in relation to the UMTS rollout will not further be addressed.[2]

1.6 Terminology

The term *mobile business* (also known as *m-business*) is commonly being referred to as the interchange of services, goods, and transactions on mobile devices. The interchanges can be performed between businesses and consumers (B2C), businesses and businesses (B2B), businesses and professionals (B2P), consumers and consumers (C2C), businesses and devices (B2D), devices and professionals (D2P), as well as devices and devices (D2D) [244].[3]

With this generic definition in mind, we chose to concentrate on the cases of B2B, B2P, and D2P. Recognizing mobile enterprise solutions as the currently most compelling focus of enabling mobile business solutions, we allow us to use the term *mobile business* when actually referring to such concrete enterprise and office solutions.

Unlike many publications in the analyst community, we carefully distiguish between the terms *wireless* and *mobile*. We use the definition of mobile respectively wireless technologies being two separate sets, with a common intersection, i.e. wireless *and* mobile technologies (*see* figure 1.2). This intersection is a *stronger* definition, implying additional issues such as roaming, hand-off, and movement to the wireless transmission technologies.[4]

[2] This thesis does not disclose any product strategy of Smartner; the research summarized by this document contributes to a foundation for deriving strategic decisions from, applied to a company like Smartner.

[3] Additionally, the term *mobile business* can refer to a company acting in the mobile business industry. For example, Smartner *is* a mobile business.

[4] An example for a mobile, but not wireless solution could be given by user sessions roaming between

When using the term *3G*, we denote the technological movement into the third generation of mobile communications, as well as the convergence between mobile communication and the Internet in general, rather than a specific technology (such as e.g. UMTS, section 2.4). By presenting alternative technologies fulfilling the same requirements as those ones characterizing 3G, we illustrate that 3G theoretically also can consist of something like e.g. WLAN (*see* section 2.6.2).

We use the term *component* when referring to a structural unit, such as a method, class, module, subsystem, binary encapsulated object with metadata, or framework, when we do not care to differentiate between these.

In our context, the term *shared data* refers to generic, common information databases, as well as common subsets or elements of such a database (e.g. distributed calendar, address registers, message archives, file directories, etc.).

By using the term *data*, we simply denote every abstract matter d, that can be represented by at least $d \in \mathbf{D} = \{0,1\}^*$.

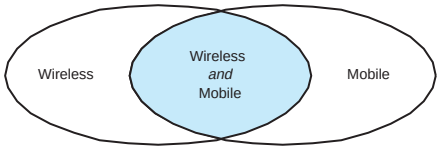

Figure 1.2: Distinguishing between wireless and mobile technologies

several client computers in the Internet, or even by personalized Web based solutions.

Chapter 2

The 3G Service Space

2.1 Introduction

The next generation of wireless and mobile technologies *is* the third generation (3G). There is a large diversity of parellel development trends in related technologies, and one might argue that there is no common way to summarize these trends. However, the 3G technology trends can be described by three fundamental dimensions, which together form the 3G technology space or the *3G service space* [194]:

Definition. The 3G service space can be identified by the space as spanned by the following set of abstract dimensions:

- Terminal capabilities,
- Bandwidth,
- Packet data service.

The parallel, mostly independent development of wireless terminal capabilities, wireless transmission bandwidth technologies, and wireless packet data functionality enable 3G solutions, services, and applications.[1] Although there might be some 3G technologies which only benefit from one of the three dimensions, most technologies are dependent on at least two of the three dimensions. The *real* 3G value results from an environment with increased terminal capabilities, increased packet data support, *and* increased bandwidth. For avoiding misleading use of the 3G terminology, we define it such that if a solution cannot be explained by and located in the 3G service space, it should not be considered as related to 3G (figure 2.1). This chapter will describe the main features and characteristics of the three dimensions which together form the 3G service space.

2.2 Evolution

A more technical definition of the terminolology used above can be made based on the historic development of mobile communication technologies. Still any precise

[1] Note that the number *three* in *3G* refers to the third generation, and not to the three dimensions of the 3G service space.

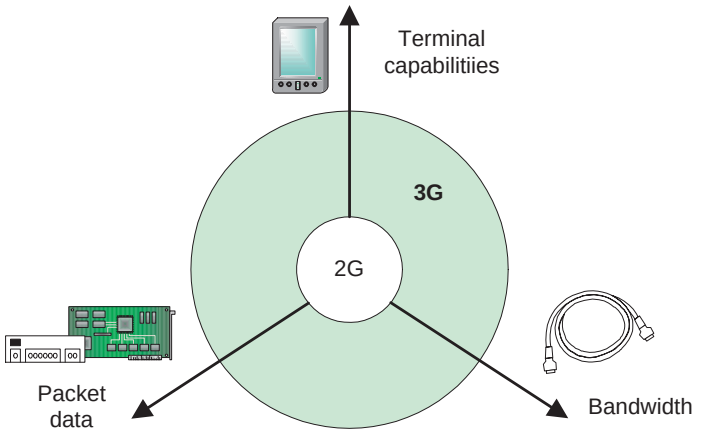

Figure 2.1: From 2G to 3G in three dimensions: the 3G service space

definition of different terms is not given, as their boundaries are not exact. Terminology ranging from *mobile internet* to *3G* sounds promising, but in most cases, these terms do not describe a concrete network technology, but rather a set of expected features. Terms like *GPRS*, *UMTS*, and *3G* are often used in contradictory ways, this might be because of the lack of any clear definition. This section will briefly describe the wireless network technology evolution in a short overview (*see* figures 2.2 and table 2.1), giving a minimal definition for terms used throughout this report [132, 41].

1G Technologies. Less often used than the following terms, 1G denotes the very first generation of common mobile communication networks connectable to the Public Switched Telephone Network (PSTN). These were analog cellular systems such as Advanced Mobile Phone System (AMPS) in the USA, Nordisk Mobiltelefon (NMT) in Scandinavia, or C-Netz in Germany. 1G technologies embodied the first realization of cellular concepts, including frequency reuse and handoffs.

2G Technologies. A widespread impact on personal life was caused by the second generation mobile communication networks, based on digital cellular systems such as Global System for Mobile Communication (GSM) in Europe and Asia, Digital AMPS (D-AMPS) in the USA, Code Division Multiple Access (CDMA) in USA and Japan, and Personal Digital Cellular (PDC) in Japan. Major functional enhancements of 2G technologies are voice coding, digital modulation, and forward error correction. Additional services like fax, data, messaging, and roaming between networks were provided. Especially in the GSM case, the successful Short-Message Systems (SMS) service has shown that voice traffic is not the only service users want. The standardization of the Wireless Application Protocol (WAP) brings the first phones with an integrated browser onto the market. These 2nd generation systems had such a wide impact due to the rapid reduction in costs *and* the perceived quality.

2.5G Technologies. One could say that 2.5G is a pragmatic solution, bridging 2G

with 3G. Technically, it provides improvements for most obvious 2G problems resulting in a tolerable solution. 2.5G is based on the introduction of packet-switched functionalities. Minor improvements were made regarding transmission speed, but idle time charges and pricing in general changed more or less drastically. GSM networks are currently being upgraded with High-Speed Circuit Switched Data (HSCSD) and *General Packet Radio System (GPRS)* in Europe and Asia (*see* section 2.3), while PDC networks are receiving high added value due to i-Mode in Japan (the same is envisioned with the WAP case in Europe). From the revolution intensity point of view, there are many industry opinions expecting 2.5G networks to represent the most remarkable change [220].

2.75G Technologies. Whereas 2.5G technologies introduce a set of packet-switched functionalities and minor changes of transmission speed only, 2.75G denotes 2.5G technologies with major improvements in transmission speed. GSM networks will be upgraded with Enhanced Data Rate for GSM Evolution (EDGE/ E-GPRS) technologies, enabling hypothetical maximums speeds of 384 kb/s. CDMA networks are being upgraded to the first versions of cdma2000 [41].

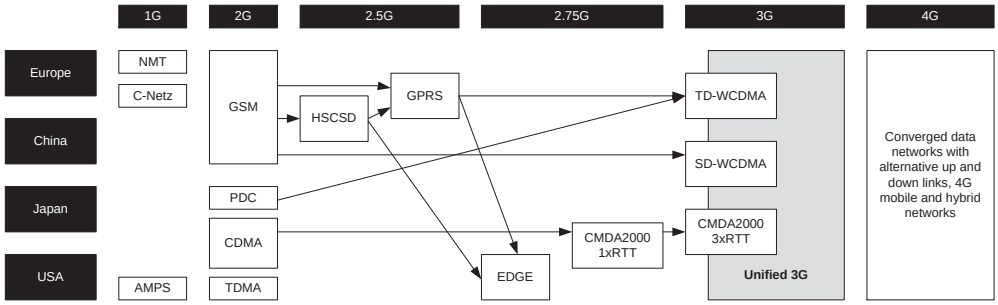

Figure 2.2: Wireless network technology evolution [41]

3G Technologies. Better system capacity and higher data transmission speed are the fundamentals of 3G *network* technologies, which were basically designed from scratch.[2] The International Mobile Telecommunications 2000 (IMT-2000) initiative aims to provide an effective solution for enabling these fundamental features. Multimedia services including audio, video, and images will determine the change of application appearance and will have an impact on mobile terminal devices. Another major difference from previous technologies is the fact, that the 3G initiative came from device manufacturers, not from operators. In 1996, the development was initiated by Nippon Telephone & Telegraph (NTT) and Ericsson; in 1997 the Telecommunications Industry Association (TIA) in the USA chose CDMA as a technology for 3G; in 1998 the European Telecommunications Standards Institute (ETSI) did the same thing; and finally, in 1998 Wideband CDMA (W-CDMA) and cdma2000 were adopted for the *Universal Mobile Telecommunications System (UMTS)* (*see* section 2.4). Fundamental changes are required on the terminal side, as the terminal is drifting away from

[2]Whereas these fundamentals characterize the 3G *networks*, note that the 3G service space is given by the additional emerging terminal capabilities (*see* section 2.1).

the classic telephone, and going towards smartphones, Personal Digital Assistands (PDA), and Pocket Computers. An obvious need for software platforms rises, and several *Wireless Operating System (Wireless OS)* approaches already exist.

4G Technologies. There are already rumours of technologies that will compete with UMTS [41]. One step in the development towards 4G technologies could be the separation of up and down link technologies. Feasibility tests made by AT&T, for a system with an EDGE up link and wideband Orthogonal Frequency Division Multiplexing (OFDM) down link, are examples of these possibilities. Another possible configuration could be the use of GPRS or UMTS technology for the up link, and Wireless LAN (WLAN) for the down link or even WLAN on both up and down link (*see* section 2.6.2). Downlink systems could also make use of existing digital broadcasting systems. The various 4G initiatives are very recent. Most of these set out to achieve the performance that 3G initially intended to provide, but the focus remains on the convergence between existing networks. The use of the *4G* term remains questionable and open for discussion, since no real revolution in convergence between telecommunication and datacommunication is expected after the third generation. On the other hand, deployments of all-IP based wireless networks might originate from operator-independent developments [187]. Simply increasing performance, speed, quality, etc.—as long as it can be described within the 3G service space dimensions, could still be regarded as development of 3G technologies.

Spectrum	Bandwidth	Systems
800 MHz	50 MHz	AMPS, IS-95, IS-136
900 MHz	50 MHz	GSM-900
1500 MHz	48 MHz	Japan PDC
1700 MHz	60 MHz	Korean PDS
1800 MHz	150 MHz	GSM-1800
1900 MHz	120 MHz	PCS
2100 MHz	155 MHz	3G

Table 2.1: Worldwide spectrum allocation for wireless technologies [132]

2.3 General Packet Radio Service

2.3.1 Introduction

As General Packet Radio Service (GPRS) is the most recent mobile technology being deployed, applications and terminals have to be developed for this technology, and new experiences and conclusions will result from its actual use (e.g. a second chance for disappointed WAP user community). Although there might have been problems in the recent deployments, GPRS has the clear advantage of easy upgrades on the operator side (*see* figure 2.3 and table 2.2; for a more detailed description of the components *see* section 2.4.3). We will not focus on the GPRS technology specification or definition in detail, even though it is an important underlying network layer for

technologies and solutions described in later sections. The following section will briefly describe the main features of GPRS, as necessary to understand the rest of the report.

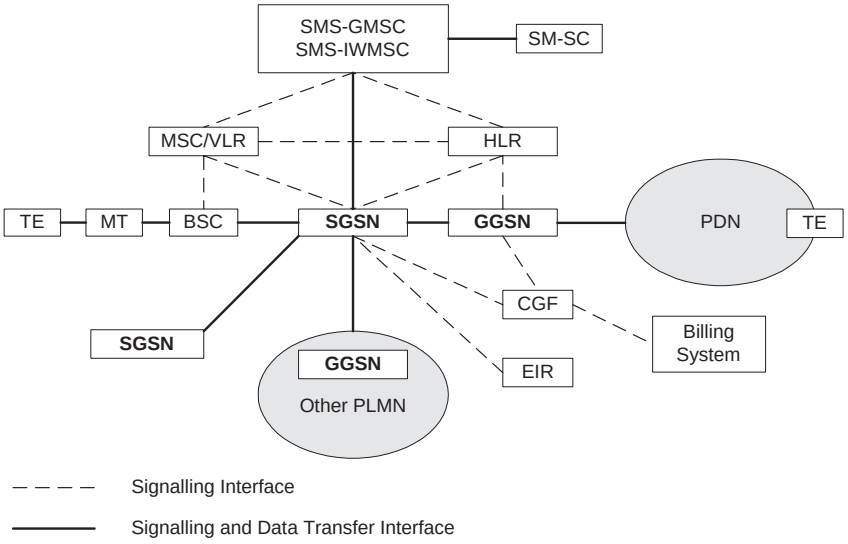

Figure 2.3: GPRS architecture overview [70]

Element	Abbrev.	Software	Hardware
Mobile Station	MS	Upgrade	Upgrade
Base Tranceiver Station	BTS	Upgrade	*No change*
Base Station Controller	BSC	Upgrade	PCU interface
Transcoder and Rate Adaptor Unit	TRAU	*No change*	*No change*
Mobile Switching Center	MSC/VLR	Upgrade	*No change*
Home Location Register	HLR	Upgrade	*No change*
Serving GPRS Support Node	SGSN	New	New
Gateway GPRS Support Node	GGSN	New	New

Table 2.2: Upgrading from GSM to GPRS [132]

2.3.2 Technology Improvements

Although based on the GSM network, GPRS is based on more flexible access to the existing radio channels. The new radio network is equipped with new resource allocation flexibility, working with 1-8 available time slots. Furthermore, several users can now share a single time slot, and the uplink and downlink time slots are separately available. All radio resources can be shared dynamically between speech and data, these resources can be allocated for instance as a function of traffic load and operator preference. Transmission rates range from 9 kb/s to 150 kb/s. Above all,

a much faster reservation of resources is possible, thus packet transmission starts within 0.5 s to 1 s [132, 54, 98, 117].

2.3.3 New Functionalities

The GPRS network provides a completely new set of functionalities, which make this network more packet-switched than traditional circuit-switched telephone networks. In fact, some of these functionalities are absolutely necessary for packet-switching.

Point-to-Point functionality. This functionality provides support for data transfer and unencrypted access. Additional important features are registration (associating mobile device identity with packet data protocols), authentication, and authorization. Admission control is possible on both radio and network levels, and message screening including filtering of unsolicited or erroneous datagrams is similar to Internet router functionality.

Packet routing functionality. Basic mechanisms for packet-switching were added. A *relay* is used by the Base Station Subsystem (BSS) to forward packets between the mobile station and Gateway Support Nodes (GSN). *Routing* adds the mechanisms for determining the packet destination based on datagram headers and for forwarding. *Address translation and mapping* allows mapping between GPRS network addresses and external data network address (e.g. IP). *Encapsulation and tunneling* (as well as *compression and encryption*) offer fundamental datagram multiplexing functionalities that allow establishing (secure) tunnels. Elementary *DNS functions* provide mechanisms for mapping logical GSN names to IP addresses and vice versa.

Logical link management. This functionality maintains the communication channel between a mobile station and GSM network. It involves three basic operations: Logical link establishment, Logical link maintenance, and Logical link release.

Radio resource management. This functionality provides mechanisms for allocating and maintaining radio communication paths. There are three basic components: *Um management* is the component determining the amount of required radio resources to be allocated, *Cell selection* allows the mobile station to select the optimal cell for radio, and *Um-tranx* is responsible for packet multiplexing, error detection and correction, and flow control [132].

2.3.4 Billing

In packet-switched networks, where the mobile station will be able to stay connected while idle, time-based pricing does not make sense anymore. Instead, the user would rather pay for the amount of data actually transmitted or simply pay a fixed service fee. In GPRS, the Serving GPRS Support Node (SGSN) can collect the following charging information for a mobile station, which makes GPRS highly suited for per packet (or even *flatrate*) pricing [132]:

- Location information (home or visited network),
- Amount of data transmitted (uplink and downlink),

- Statistics on the Quality of Service (QoS) profiles for the datagrams,
- Amount of time a Mobile Station (MS) is assigned a Packet Data Protocol (PDP) address,
- Amount of GPRS network activity (e.g. mobility management) dedicated to the mobile station.

2.4 Universal Mobile Telecommunications System

2.4.1 Introduction

With remarkably higher transmission rates, the Universal Mobile Telecommunications System (UMTS) delivers true universal multimedia coverage and nationwide roaming. More than that, UMTS offers greater spectrum efficiency and capacity compared to the current 2G and 2.5G networks. UMTS is intended to be a solution for managing increasing and converging demands for mobility, data, and multimedia. Due to the absence of global standardization in the early ages of wireless communication, there are today two major regional telecommunication standards dominating the global market, TDMA/CDMA developed by TIA in the USA and GSM developed by ETSI in Europe. Moving toward 3G wireless, there has been a rising need to develop more global and collaborative standards. There are two major recent partnerships projects created by the global wireless industry to address this issue [174]:

3rd Generation Partnership Project (3GPP). The 3GPP [1] is developing 3G standards for GSM based systems. The consortium includes ETSI (Europe), T1 (USA), ARIB/TTC (Japan), TTA (Korea) and CWTS (China). The North American part of the TDMA community is participating and contributing in 3GPP as ANSI-41 based TDMA systems evolve towards 3G architecture based on EDGE and GPRS.

3rd Generation Partnership Project 2 (3GPP2). The 3GPP2 [2] is developing 3G standards for IS-95 based CDMA systems. The consortium includes TIA (USA), ARIB/TTC, TTA and CWTS.

As the 3GPP represents the partnership project most relevant for European countries, we refer to UMTS as the technology based on the 3GPP specification during the rest of this section. There are two current specifications available [1]:

- the first phase, UMTS Release 99 (R'99),
- the second phase, UMTS Release 00 (R'00).

This section will briefly introduce the main technology changes, advantages, and features provided by the UMTS as of Release 99.

2.4.2 System Overview

The UMTS architecture can be divided into three major components: the User Equipment (UE), the radio access network, and the core network. The radio access network, also known under the name UMTS Terrestrial Radio Access Network (UTRAN), is the entity connecting the UE and the core network with each other. The UTRAN

can be subdivided into a set of interconnected Radio Network Subsystems (RNS), each of which consist of a Radio Network Controller (RNC) with an underlying set of Node Base Stations (BS) [99] (*see* figure 2.4).

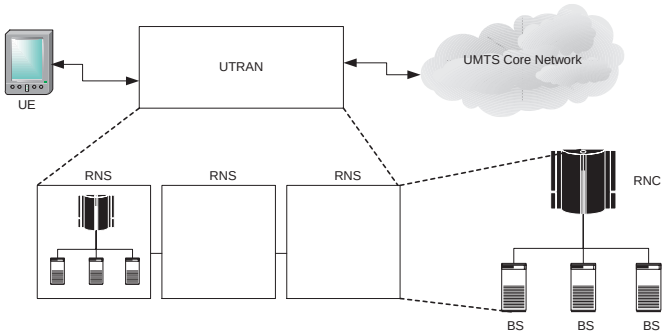

Figure 2.4: UMTS system and radio access network architecture [47]

A novelty in the UTRAN concept is the existence of two different, but complementary, radio access modes:

Frequency-Division Duplex (FDD). Suitable for symmetric traffic, this mode offers full mobility.

Time-Division Duplex (TDD). Suitable for asymmetric traffic such as web browsing, this mode offers only a limited mobility and therefore is more adapted for indoor environments.

These two modes offer high efficiency within one system whatever the conditions (wide area, urban, indoor coverage from outdoor, indoor, etc.).

2.4.3 Network Logic

The core network is based on two separate domains, one packet switched and one circuit switched, allowing integration and compatibility with 2G systems. Thus, the circuit switched domain is built on the current GSM system, whereas the packet switched domain can be regarded as an extension, as in GPRS (*see* section 2.3) and EDGE. The *circuit switched* core network domain consists of a Mobile Services Switching Centre (MSC), a Gateway MSC (GMSC) as well as Visited Location Register (VLR) and Home Location Register (HLR). The *packet switched* core network domain consists of two gateway support nodes, Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN).

The UMTS packet switched domain is highly similar to the logical architecture of the GPRS (*see* figure 2.3). Basically, only the UTRAN and the UE consisting of Terminal Equipment (TE) and Mobile Terminal (MT) have been added (*see* figure 2.5). One of the main changes in UMTS is an increased functionality shift from the core network nodes to the UTRAN, such as e.g. data compression and ciphering [47].

The functionality and logic of the UMTS network architecture components are described below [99]:[3]

[3]The components marked with the †-symbol are the same as in the GSM system.

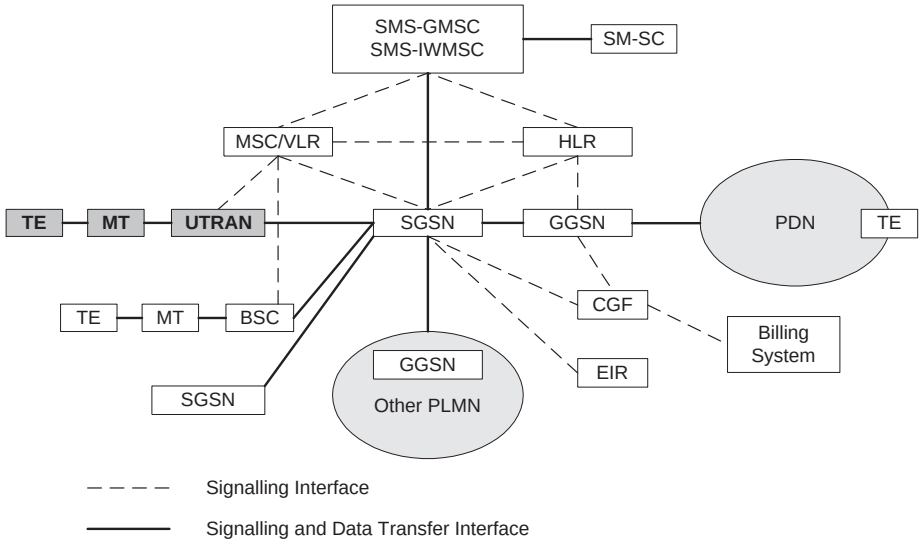

Figure 2.5: UMTS architecture overview [47]

SGSN. The SGSN is responsible for keeping track of the UEs individual location and performs security functions and access control. In the UMTS framework, the SGSN is connected to the UTRAN whereas in the GPRS system, it is connected to the GSM base station system.

GGSN. The GGSN is responsible for providing interworking with external packet switched networks, e.g. an IP network, and is connected over an IP-based backbone—the Public Land Mobile Network PLMN).

CGF. The Charging Gateway Functionality (CGF) monitors charging information from the SGSNs and GGSNs.

MSC/VLR. Depending on whether the system is based on GSM/GPRS or not, the SGSN can send location information to the MSC or VLR, respectively. The SGSN may furthermore receive paging requests from the MSC/VLR.

HLR.† The HLR contains individual GSM and UMTS subscriber-related information.

EIR.† The Equipment Identity Register (EIR) contains the terminal device equipment information. Equipment-related services and support can be implemented in the EIR.

SMS-GMSC/SMS-IWMSC.† The Short Message Service Gateway MSC (SMS-GMSC) and the Short Message Service Interworking MSC (SMS-IWMSC) provide support for SMS transission over the SGSN. This messaging is administered by the Short message Service Center (SM-SC).

2.4.4 Bandwidth

The main feature that clearly distinguishes the UMTS network from its 2.5G predecessors is the greater bandwidth and the resulting larger transmission rates. There are three main UMTS transmission classes (*see* table 2.3), determining the available bandwidth. With at least ten times higher transmission rates than in GSM, the UMTS allows interactive applications like video conferencing and bursty transmissions at a higher speed.

Class	Rate	Example
High-speed movement	144 Kb/s	in vehicles
Pedestrian	384 Kb/s	on campus
Stationary	2 Mb/s	in buildings

Table 2.3: UMTS transmission rate classes [132]

Although the UMTS bandwidth is relatively small compared to wireless local area network (*see* section 2.6.2) or other future mobile communication systems, the bandwidth development has raised fundamental research questions in physical and theoretical limits of wireless access. For cellular systems, capacities of 10-100 Mb/s might be reached, but the main constraints are still the number of users and high installation costs. The use of packet-switching for future user access multiplexing can be regarded as an efficient way to decrease costs [160].

2.4.5 Quality of Service

Another significant feature introduced by 3G mobile networks is the provisioning of Quality of Service (QoS) on an end-to-end basis. The 3G QoS idea is based on the goal of efficient use of resources in a packet-switched network. Whereas some users might not have any demands on current quality (e.g. when idle), others might need guaranteed quality (e.g. for interactive sessions). The 3G QoS control mechanisms are based on the ability to dynamically change the QoS parameters during a session, and allow interworking with current QoS schemes [132, 33].

The application level end-to-end service uses the bearer services of the underlying networks, partitioned into three segments: *Local Bearer Service*, *External Bearer Service*, and *3G Bearer Service*. The latter one is the relevant service for UMTS QoS provision, and again consists of two parts. The *Radio Access Bearer Interface* provides confidential transport of signalling and user data for moving users over the radio interface. The *Core Network Bearer Service* makes use of a 3G backbone network service providing layer 1 and layer 2 functionality (via ATM or IP).

Due to the restrictions and limitations of the air interface, the *QoS classes* defined for the mobile network are very different from fixed networks. In UMTS, four QoS classes have been defined based on delay sensivity (*see* table 2.4). In addition to the traffic classes, a single set of relevant QoS parameters have been defined [32], including maximum, minimum, and guaranteed bit rates; delivery order, maximum packet size, reliability, etc.

For any Internet application, the specification recommends how the 3G QoS is to be mapped to the Internet QoS definitions. These specifications define the appropriate attributes for integrated services (`IntServ`) and differentiated services (`DiffServ`).

`IntServ` may require flow establishment and aggregation control in the 3G packet core; `DiffServ` requires QoS profile conversion of packets. The service type parameters for both `IntServ` and `DiffServ` are being controlled by the applications, i.e. the terminal equimpent (TE). Existing IP QoS parameters are being mapped into 3G QoS at the border of the 3G core network (*see* figure 2.6). Current QoS mechanisms are designed based on the requirement of general but simple QoS attributes. More complex traffic models for estimating UMTS traffic exist, taking into account the possible combinations of service type, such as voice, data, and video with the set of different environments, such as private, public outdoor, or public indoor [143].

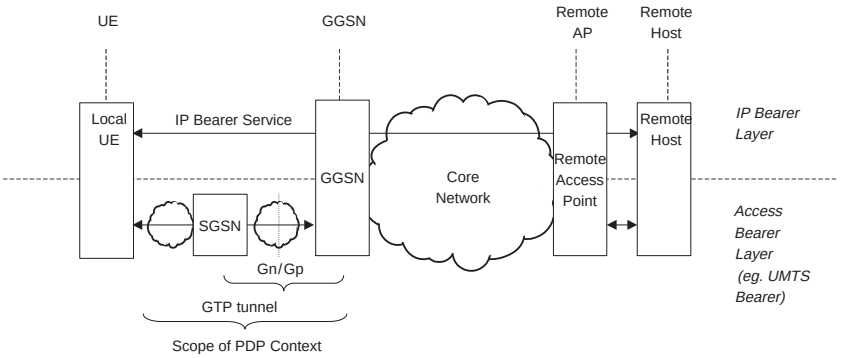

Figure 2.6: UMTS network architecture [33]

Technology trends show that UMTS is likely to be all-IP based, i.e. even telephony will be packet-switched [33, 53, 174].[4] The divergent approaches of 3GPP and 3GPP2 will need harmonization at some stage, if convergence toward IP based mobile telecommunication networks is to become a reality [45].

2.5 Wireless Operating Systems for 3G Handsets

2.5.1 Introduction

The terminal device evolution from simple phones, with simple numerical displays to small portable computers, Personal Digital Assistants (PDAs, with memory, pro-

[4]This is a remarkable *inversion*; first TCP/IP dial-in connections were established through a modem over the telephone line, now Voice over IP (VoIP) telephony is carried over the IP network.

Class	Requirements	Example
Conversational	very delay-sensitive	traditional voice; VoIP
Streaming	better channel coding; retransmission	one-way real time video/audio
Interactive	delay-insensitive	Telnet; interactive e-mail; WWW
Background	delay-insensitive	ftp; background e-mail

Table 2.4: UMTS QoS classes [132]

cessor and matrix display) is one of the main prerequisites for enabling mobile office access through wireless networks. One could segment the resulting set of mobile terminals based on three categories [153]: voice communication, Personal Information Management (PIM), and Internet access (*see* figure 2.7). According to this definition, the synergy of all the three categories—a *smartphone*—is a PDA with both types of communication facilities (i.e. Internet and voice). One could argue that intelligent middleware, such as voice controlled services and intelligent network agents could take over tasks like e-mail, PIM access, etc. But the mainstream trend is towards more advanced *applications*; the business user prefers his smartphone to allow access to existing computer desktop applications. Network access is required in order to provide consistency with information stored on corporate information systems (e.g. sales data, groupware plaftorm, etc.). However, this terminal device development does not imply a complete change of balance of intelligence, away from the networks, into the devices, the developments go hand in hand: advanced wireless applications require advanced middleware and networks. In fact, the combination of advanced wireless applications and smart middleware does not even necessarily need very high transmission speed. A pessimistic user could on the other hand argue that if mobile terminals are being equipped with greater processor speed, more memory, and with a more complex operating system, they will go through exactly the same development as the personal computer, ending up in a situation where the majority of system resources are being consumed by the operating system anyway. However, as the concept of a PDA as a software platform still is quite a new market, and there are alternatives to a Microsoft Windows dominated platform world: it is not yet clear that the same de facto standards as in desktop computing will evolve in this segment.

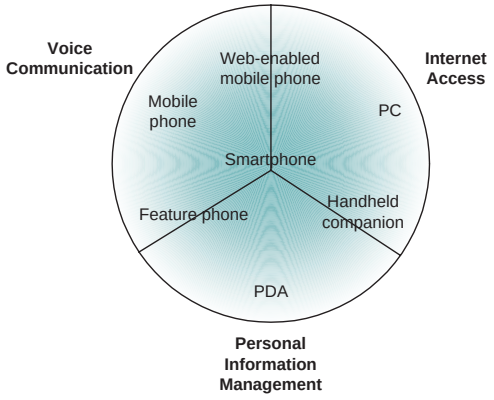

Figure 2.7: Mobile device classification [153]

2.5.2 Characteristics

A Wireless Operating System (Wireless OS) has some basic requirements which make it differ substantially from standard operating systems for personal computers. Unlike common desktop operating systems that need several minutes to boot, the wireless OS is required to provide *immediate access* to applications. A wireless network

does not help much, if the wireless OS does not launch immediately.[5] Another requirement is *modularity*, allowing a reduction of amount of software resident in the device. Given the fact that storage in the mobile device may be currently limited due to the device size,[6] the system should be able to remove and backup certain components and reload it from a server if necessary, in order to free up local storage. *Support for low-power CPUs* is also strongly recommended, based on the limited battery power available. Finally, the wireless OS should be designed to support an *On-Off capability*. If a mobile device is to be switched on and off (or put into standby mode for power save), then the wireless OS should be stored in a protected memory (such as a Flash ROM), which again implies the OS to be of very limited size [132].[7] However, this could be regarded as a minor feature, considering the vision of 3G handsets being *always on line*, i.e. even when the user has put his device away.

Among several approaches, there are three major players in the wireless OS market [132]. Given our focus on EPOC (*see* section 2.5.4) and Java (*see* section 2.5.7) in this thesis, the following sections briefly describe these approaches.

2.5.3 Windows Consumer Electronics

The *Windows Consumer Electronics (WinCE) OS* [170, 15], developed by Microsoft is a *"carefully designed subset of the Windows NT OS system"* [132].[8] It satisfies the wireless OS requirements previously described. Desktop Windows applications are not compatible with WinCE, but Microsoft offers a set of development tools for easy transformation of Windows applications to WinCE. WinCE is being supported by next-generation smart phone manufacturers such as Casio, Siemens, and Compaq.

2.5.4 Symbian Platform

Symbian is a company formed by Nokia, Ericsson, Motorola, and Psion. The Symbian platform is based on the *Electronic Pocket Communication (EPOC) OS* [24]. It is designed as the wireless OS platform for the coming mobile devices (*see* figure 2.8). EPOC OS was originally developed by Psion for its communicator, organizer, and subnotebook products.

The EPOC OS core consists of three components. The *Base component* represents a portable runtime system, kernel, file server, user library, and file server APIs. The *Engine support component* provides the APIs for data manipulation, an application architecture, resource files and utilities, the standard C library, and text tools. Finally, the *Graphics component* provides the high-level GUI framework including drawing and user interaction, fonts, printing, views, and text entry [132]. The EPOC OS is based on an object-oriented design, in combination with particularities that are a necessity for hand-held devices. Built within the kernel, the features of power management, memory management, event handling mechanisms, and multitasking are effectively supported [166].

[5] By using the term *immediate launch*, we refer to an instant *wake-up* of the system, without caring about the actual implementation (e.g. memory hibernation, low-power standby, etc.).

[6] As we all know, the constraint of size is instantly loosing importance.

[7] On the other hand, multi-megabyte Flash ROMs exist.

[8] The WinCE is also being referred to as *PocketPC*. However, PocketPC can also be interpreted as the entire device running with WinCE as operating system. We therefore care to differentiate between these terms.

One basic goal of Symbian is to define usage of EPOC on a higher level. The platform is supported by code libraries that hide the underlying communication protocols from application designers. For a heterogenous set of devices with Symbian support, software will be developed once requiring a little more effort, instead of writing the application for each communication technology. Although the EPOC OS is based on C++ as a programming language, a Java interpreter exists which runs on top of EPOC [166]. A more detailed discussion about application design under Symbian can be found in section 3.7.2.

2.5.5 Palm

The Palm OS [19] was developed by 3Com Palm Computing, and was aggressively designed to support wireless applications. Qualcomm has been working on a Palm OS based product called *pdQ*. Sony has licensed Palm OS to produce the next version of Palm OS for a more extended set of devices, including smart phones. Other vendors incorporating Palm OS into their devices are Handspring, IBM, Motorola, Nokia, and Symbol (*see* table 2.5).

Figure 2.8: The Nokia 9210 Communicator with EPOC OS

2.5.6 Linux

There are several minimized Linux distribution suitable to run on PDAs [42]. Two widespread Linux related operating systems are *pSOS* and *VxWorks* by WindRiver [199, 31, 28]. pSOS is a modular, high performance real-time operating system (RTOS)

designed for embedded microprocessors. It provides a complete multitasking environment based on a set of open system standards, such as network file system support. pSOS is designed for performance, reliability, and ease of use on a large number of hardware platforms. VxWorks is the operating system environment of WindRiver's Tornado II embedded development platform, and a widely adopted RTOS in the embedded industry.[9] VxWorks is flexible and scalable, and furthermore available on all popular platforms. *Tynux* is another Linux implementation offered by PalmPalm, which recently has introduced a Linux-powered mobile phone [41].

2.5.7 Java as a Platform

Another interesting *"operating system for small devices"* [140] is *JavaOS*. By now, Java has a long history as a net-capable programming language. Sun Microsystems' own spin off subsidiary, *JavaSoft*, announced JavaOS in 1996, a complete operating system with a minimal set of application and network services as well as a PDA-size memory footprint. Any subset of the traditional operating-system services (e.g. window management) can be included in a freely customized JavaOS into a memory space appropriate for an embedded processor. When JavaSoft designed JavaOS, it had to minimize the platform dependencies in order to produce platform-independent code. The smallest possible set of kernel services needed to support the virtual machine was developed first. Then, the virtual machine and the new minimalist kernel were used to implement all the other services, GUI support, networking, I/O drivers, and a file system. Finally, Sun checked its work to make sure it supported the full Java API. Applications written for JavaOS should run on any Java-enabled platform [140].

Although being more than a strictly OS approach, indications are that Java will become a very successful environment for mobile applications. Java is independent of the underlying OS, because a *Java Virtual Machine (JVM)* exists for a variety of platforms. Development of portable Java versions include *Personal Java* and *Java 2 Micro Edition (J2ME)*. These enable Java applications in cell phones, already this has received large support from e.g. NTT DoCoMo (i-appli), Nokia, and Motorola [93, 74, 96].

Assuming that Java will be supported by Palm, EPOC, and at least theoretically by WinCE,[10] Java could become a platform with huge support (*see* table 2.5). A more detailed discussion about Java as a software platform can be found in section 3.7.3.

Several approaches exist, combining applications running on a wireless OS with a distributed middleware framework, for providing applications like e-mail more efficiently [82]. Ideally, these applications can be developed using standard middleware components like CORBA to improve their quality and reduce their cost and cycle time.

[9] An *embedded system* is a set of devices used to control, monitor or assist the operation of equipment, machinery, or plant. *Embedded* reflects the fact that they are an integral part of the system.

[10] If not Microsoft themselves, someone will port the JVM to thw WinCE platform.

Manufacturer	Device	OS
Casio	Cassiopeia	CE
Compaq	Aero	CE
Ericsson	MC218	EPOC
Handspring	Visor	PalmOS
HP	Jornada	CE
IBM	WorkPad PC Companion	PalmOS
LG Telecom	CDMA mobiles, Korean market	J2ME
Motorola	New PDAs	PalmOS
Motorola	PageWriter	J2ME
Motorola	New mobile phones	EPOC
Nokia	New handheld devices	PalmOS
Nokia	New mobile phones	EPOC
Palm	Palm III, Palm V, Palm VII	PalmOS
Panasonic	New mobile phones	EPOC
Philips	Handheld devices	EPOC
Psion	Series 5mx, netBook	EPOC
Qualcomm	pdQ smartphone	PalmOS
RIM	New devices by end 2000	J2ME
Sony	New mobile phones series	PalmOS
Sony	New mobile phones	J2ME
Sony	New mobile phones	EPOC
Symbol	PPT 2700	CE
Symbol	SPT1500 and SPT1700 (VADs)	PalmOS

Table 2.5: Devices and operating systems [153]

2.6 Alternative and Complementary Technologies

2.6.1 Introduction

The convergence of the telecommunication and data communication industry not only implies mobile terminals and wireless networks to develop increased support for data. One could say that another approach toward this convergence comes from the computer industry rather than telecommunication operators. The *Iceberg* research project at UC Berkeley is an example for deploying integrated communications (telephony and data) on top of an Internet core network [228]. Furthermore, adding communication facilities to existing PDA and handheld PC platforms is obviously easier than adding operating systems to mobile phones [154]. The introduction of wireless OSs (*see* section 2.5) gives a taste of future market shifts, where software giants (e.g. Microsoft) will compete with terminal manufacturers (e.g. Nokia, Siemens, Ericsson, and Motorola).

Another aspect is brought up by alternatives to 3G technologies. It is not said that operator-hosted mobile communication services are the only right approach after all; the success of the Internet during the 1990's might have the same effect upon wireless. Open, free networks and multinetwork systems consisting of a set of different network technologies might become at least as popular. After all, the only thing users care about is the applications which they find useful.

> *"The real revoloution is not 3G, but its successor. The successor, however, will not be 4G, but the first generation of a new dynasty.*
>
> *The new dynasty will be based on less complex technologies and a new business culture in which the user has much more influence. You see it emerge in the licence-free 2.4 GHz band using IEEE 802.11b technology. Anyone can put up a base station and build a network."*
>
> *—Prof. Björn Pehrson, interview on 5.6.2001 in Stockholm*

2.6.2 Wireless Local Area Network

Wireless Local Area Network (also known as *Wireless LAN*, *WLAN*) systems have become increasingly popular after the initial availability of the first widely sold PCMCIA WLAN receivers (by e.g. Lucent Technologies) in 1997.[11] This WLAN network technology is also known as *IEEE 802.11*, which was standardized in the early 90's [58]. Under this standard, the data throughputs of these systems were 1.6 Mb/s and 11 Mb/s. Current research is underway to increase the access rate to the order of 25 Mb/s in the near term and up to 100 Mb/s in the long term [36], and enhancements for the IEEE 802.11 standard to improve WLAN support for multimedia applications has also been proposed [80]. The main advantages of WLAN—besides high bandwidth—include [80]:

- Cost effectiveness,
- Ease of installation,
- Flexibility,

[11] The first civillian WLAN products were available as early as 1988.

- Unlimited access to existing information infrastructures,
- Station mobility,
- Ubiquitious computing support.

In fact, WLANs can be quickly installed by nontechnical personnel, without network preplanning and even without a supporting backbone network. Depending on the network setup and environment (topology, interferences, etc.), wireless LAN access points (the base stations) have ranges, in the order of 150 m indoor and 300 m outdoors.

With these ranges, wireless LANs were originally intended for local, indoor wireless access to the fixed LAN. However, using the IEEE 802.11 standard allows seamless roaming between local LANs. Thus the interconnection of a set of wireless subnets, will result in a larger wireless access space, which has received commercial interest by network operators. For instance, the *Nokia Operator Wireless LAN* solution deploys a GSM SIM card for authenticating the user and identifying the home operator. Wireless LAN charges are then simply added through the GPRS Charging Gateway to the user's GSM invoice (*see* section 2.3). This solution has been developed to bring broadband access to the laptop in places like airports, convention centers, hotels and meeting rooms. With 11 Mb/s access built in, mobile laptop professionals can access their corporate network on the move [17].

The multiservice operator Jippii Group, based in Finland, currently offers WLAN access, *Jippii Freedom* in more than 100 locations in 50 Finnish cities.[12] The plan is to have numerous locations in business districts, hotels, airports, business centres, restaurants, multi-tenant office buildings and other locations with mobile users. The service will be rolled out in other countries as well [11].

Telia Mobile's *HomeRun* product in Sweden, allows registered users to power up their computers anywhere within a designated area and link directly to their home company networks. HomeRun areas exist across Sweden in places where business travellers are most likely to be waiting: airports, train stations, hotels, and conference centres. Security is an essential part of HomeRun. Together with different Virtual Private Networking (VPN) solutions (*see* section 3.4), it becomes safe for the business traveller to access the office LAN from an airport, hotel, etc. This also removes the problem of carrying around sensitive documents on the personal laptop. Telia introduced HomeRun in the end of 2000. It currently has around 50 access points, including hotels in Stockholm, Gothenburg, Arlanda airport, Central station in Stockholm, Malmö and Gothenburg. Before the end of year 2001, Telia Mobile plans to have more than 100 hotspots all over the country. Most users are employees of medium to large companies, who pay the subscription charge for them and work with Telia to design a consistent interface through HomeRun to their corporate intranets [26].

Another interesting, but compared to the previous two cases, completely different way of providing IEEE 802.11 wireless LAN access, is represented by the *StockholmOpen.Net* project. The idea is to allow everyone to put up wireless LAN stations e.g. in their window offering connectivity to bypassing users *and* providing a choice of Internet Service Providers (ISPs). This vision of and *open source* mobile network is being carried out by the Royal Institute of Technology (KTH), Stockholm in cooperation with the city of Stockholm. *"StockholmOpen.Net is an embryo of a city wide*

[12] One famous location is in downtown Helsinki, and more exactly, every spot where the TV tower is visible.

open access network with a freedom of choice of service operators" [23]. Together with a set of sponsors like Nokia, Ericsson, 3COM, etc., the project is looking for business and management models that could make this vision happen. Based on the fact that base stations are getting cheaper, and with the previous experience of the drastical Internet spread due to public access, the growth of the mobile internet is expected to shift power from network providers and operators to users [23].

Depending on the speed of current and future 3G network deployments, the importance of wireless LANs will be determined. Originally, wireless LAN technologies were not intended for e.g. countrywide coverage, but on the other hand, an increasing number of hotspots might turn wireless LAN into the mobile user's preferred access method for high speed wireless data (e.g. due to cost savings) [144, 91]. Although 3G technologies and wireless LAN belong to completely different categories, i.e. telecommunication versus computer communication, the computer side seems to have a greater chance to win the race. After all, laptops are getting smaller all the time, thus one future question could be why increase the mobile phones screensize all the time, if a small-screen laptop could replace the whole thing. This is why, as bizarre as it sounds today, one of Nokia's, Ericsson's etc. main future competitors will be computer software giants like Microsoft.

2.6.3 Bluetooth

Another popular approach for short range connectivity is *Bluetooth* [88, 192]. It has been recognized that the idea of a truly low-cost, low-power radio-based cable replacement is feasible. If all handset manufacturers adopted the same standard, then such a ubiquitous link would provide the basis for portable devices to communicate with, creating personal area networks (PANs) which have similar advantages to their office environment counterpart, the local area network. Bluetooth is an effort by a consortium of companies to design a royalty-free technology specification enabling this vision. The Bluetooth technology is based on an unlicensed, globally available radio band—the Industrial, Scientific, Medical (ISM) band, located around 2.45 GHz. The Bluetooth network is based on packet transmission with a maximum user rate over the asynchronous link at 723.2 kb/s.

An ideological characteristic that distinguishes Bluetooth from many other common wireless technologies is *Ad Hoc* radio connectivity, meaning peer-to-peer connections between all devices involved. Ad Hoc connectivity is a general concept which does not differentiate between terminal and base station. A mobile ad hoc network can be regarded as an autonomous sytems of mobile routers, moving randomly and organizing themselves. The topology of such a network may thus change rapidly and unpredictably. The main benefits of ad hoc architectures are self-reconfiguration and adaptability to highly variable mobile characteristics. These characteristics include power and transmission conditions, traffic distribution variations, and load balancing [81].

Due to its low transmission rates and short range (below 10 m), Bluetooth is not a real alternative to wireless LANs. But in multinetwork environments, i.e. in combination with other, *real* wireless technologies, Bluetooth can enable interesting applications. For instance, Ericsson offers a *Bluetooth WAP Server*, allowing customized Bluetooth applications (e.g. opening your garage door with a WAP phone) [128].

2.6.4 Home Radio Frequency

Unlike Wireless LAN and Bluetooth, the *Home Radio Frequency (HomeRF)* standard is a wireless low cost ad hoc network approach, designed and targeted especially for home usage. The vision behind HomeRF is to extend the reach of the PC and Internet throughout the home and yard, and connect the recources of the PC and Internet with legacy home applications such as telephony, audio entertainment, and home control systems. The HomeRF Working Group organization consists of approximately 100 members representing the bulk of PC, telecommunications, and consumer electronics industries (Motorola, Siemens, Intel, Proxim, etc.).

An interesting technology advantage of HomeRF is the resource sharing between data and voice services, and the definition of several different *node* types. The HomeRF working group has defined the *Shared Wireless Access Protocol (SWAP)*, supporting TCP/IP, standard voice telephony over the PSTN, as well as Voice over IP. The HomeRF network allows transmission rates up to 2 Mb/s, and is designed for a data range of 50 m.

Like Bluetooth, HomeRF can be regarded as a competitor to Wireless LAN, but the special focus on home environments makes it currently less relevant for mobile business usage [158, 100]. It has also seen little deployment so far.

2.6.5 High Performance Radio

The *High Performance Radio Local Area Network (HIPERLAN)*[13] is a high-speed Wireless LAN system operating in the 5 GHz band, defined by the European Telecommunications Standards Institute on Broadband Radio Access Networks (ETSI BRAN) [71]. The network will support multiple transmission modes, providing data rates up to 54 Mb/s. HIPERLAN will make integrated real-time multimedia applications, such as high-speed Internet access, and the distribution of high-definition audio and video streams, possible in a wireless environment. Within the scope of cellular 3G systems, HIPERLAN can be regarded as a complementary technology that can be used to provide users with high-data rate services in localized areas. This is especially attractive in hot-spot areas. Handovers between 3G cellular access networks and HIPERLAN access domains will be possible. Although HIPERLAN is not well suited for high mobility, its increased data transmission capacity is substantial. Whereas the maximum UMTS data rate of 2 Mb/s (*see* section 2.4) enables voice, data and moderate-quality video communications, the peak HIPERLAN siganling rate will enable high-resolution real-time motion video [67]. Instead of combining UMTS with HIPERLAN, a competitive alternative might be the combination of EDGE and HIPERLAN.

2.6.6 Intelligent Transport Systems

As a subset of a more advanced and complex general information and telecommunications network for users, roads, and vehicles, the Intelligent Transport System (ITS) is gaining importance. The ITS movement originates mainly from Japan, and will be able to contribute to solving problems such as traffic accidents and congestions, as well as provide multimedia services for drivers and passengers. The ITS consists of nine development areas: navigation and electronic toll collection (ETC) systems as

[13] We refer hereafter to HIPERLAN as the second version, i.e. HIPERLAN/2

well as safe driving assistance are likely to be the most interesting areas. In Japan, the first steps of ITS have already been developed using two pairs of the 5.8 GHz band. ITS is expected to become one of the most promising multimedia businesses [171, 8].

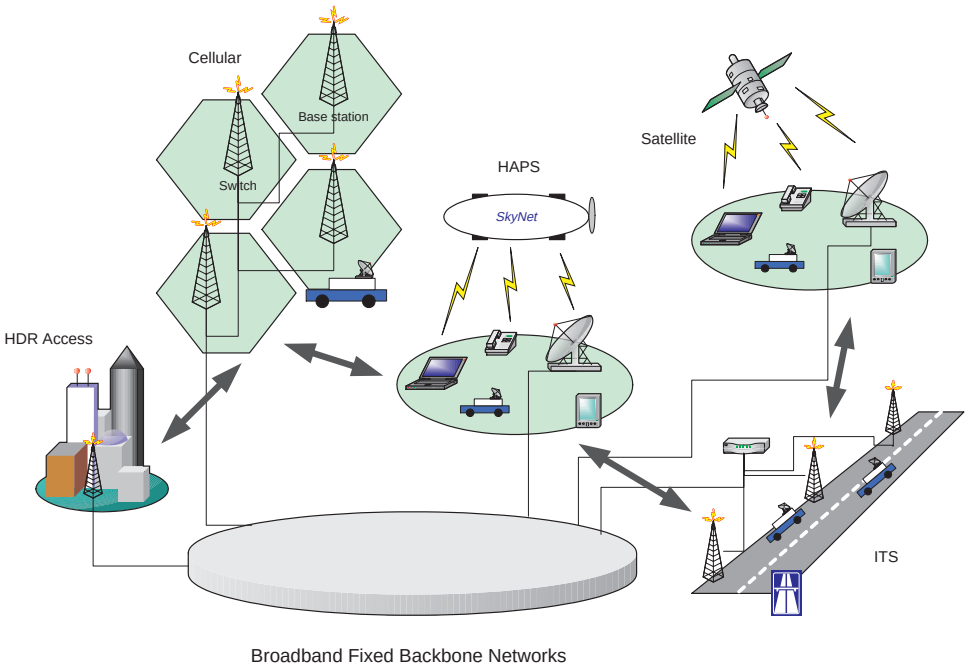

Figure 2.9: The concept of future multinetwork mobile communications [171]

2.6.7 Stratospheric Platform Radio

The concept of *High Altitude Stratospheric Platform Station (HAPS)* systems is another approach, based on the increasing demand for capacity for future generation applications. Being highly attractive for multimedia communications, the HAPS system has a potential to become a third communication infrastructure after terrestrial and satellite communication systems. A HAPS is located at about 20 km high in the stratosphere. A set of platform stations can form a network in the sky by interconnecting through optical communication links.

The user terminal communicates with the HAPS over a broadband access link. The user terminal can be fixed, or portable and mobile. The systems allows bit rates for fixed and portable terminals at 25 Mb/s, whereas fixed terminals with larger antennas can communicate with several hundred Mb/s. The use of dish antennas (e.g. a 5 cm dish antenna with 20 dBi gain for vehicles allows 144 kb/s) with high gain is feasible, since the frequency band is expected to be located in mm-wave lenghts. Recently, a 600 MHz bandwidth in the 48/47 GHz band has been allocated for the fixed services of high altitude stations.

Systems like HAPS have the capability to serve a large number of users, either in dense urban areas or over a wide geographical area. Attenuation by rain is restricted to the first few kilometres of the atmosphere, and since line-of-sight paths

can be readily obtained from HAPS. Thus considerably less infrastructure may be required to serve the same coverage area, when compared with terrestrial services. Another advantages of HAPS systems is the rapid deployment and the ability to be moved around to cover short term fluctuations in usage. The US company *Sky Station International* has proposed the use of 150 m long airships for offering this kind of broadband mobile communication [171, 241, 21].

2.6.8 Satellite Communications

Showing many similarities to the history of the Internet, satellite communications was first deployed in the 1960's and has its roots in military applications. Despite the commercial failure of the *Iridium*[14] satellite telephony system, satellite communications is still regarded as a feasible way to manage the continuous growth of worldwide Internet usage and mobile communications.[15] The industry is looking to satellite solutions for realiable, high-performance, and cost-effective technology for building networks and delivering services. After all, satellites can easiliy cover areas where land lines do not exist or cannot be installed [145].

There are two basic types of satellites, determined by the type of orbit they maintain. A *Geosynchronous Orbit (GEO)* satellite travels around the earth at an altitude of 35 800 km. This allows it to match the earth's rotation speed, so that it appears stationary to the fixed antenna on the ground. The signal footprint of a GEO satellite covers a large surface. The only bottleneck in GEO satellite communications is the physical two-way propagation delay[16] of approximately 540 ms, which has a negative effect especially on real-time communication. Research is focusing on enhancements to the TCP protocol for providing better throughputs and end-to-end performance over satellite links [141, 142], as well as on appropriate bandwidth allocation theories [51, 52].

The other type, *Low Earth Orbit (LEO)* satellites, orbit at an altitude ranging from a few hundred km to 1 500 km, circling the globe once every 90 min. Since a LEO satellite is visible to the fixed antenna on the ground for only a few minutes, the system must provide mechanisms for hand-off and signal routing between several satellites to maintain connectivity. Current research has presented efficient schemes for reducing connection dropping during hand-offs [119]. Although the physical two-way propagation delay ranges between 10 and 20 ms, the perceived, total transmission delay might increase due to inderstationary hand-off and routing. LEO satellites have small earth footprints, which makes a constellation of satellites necessary for ensuring coverage.

There are two main frequency bands. The *Ku* band operates between 10 GHz and 18 GHz, whereas the *Ka* band operates between 18 GHz and 31 GHz.[17] The bandwidth capacity of a satellite link is dependent on multiple factors. For instance *Intelsat* [9] offers links with up to 155 MHz using 72 MHz transponders. Satellite communication is extremely efficient, in terms of system setup, for delivering multimedia content to businesses and homes. On the other hand, compared to HAPS systems (*see* section 2.6.7), satellite communications systems—both LEO and GEO—the communication link between an earth station and a satellite is line-of-sight but

[14] The *Iridium* system uses a 66-satellite network system to provide global coverage [119, 10].

[15] Interestingly, Iridium remains in use by the U.S. government. Hence it will *not* be shut down.

[16] This value is calculated as the time required for the signal to travel 35 800 km into space and return.

[17] Note that the Federal Communications Commission (FCC) [5] no longer uses these alphabetic designators, only the frequency or frequency range is used.

suffers higher propagation loss, caused by the longer distance between satellite and ground stations [241].

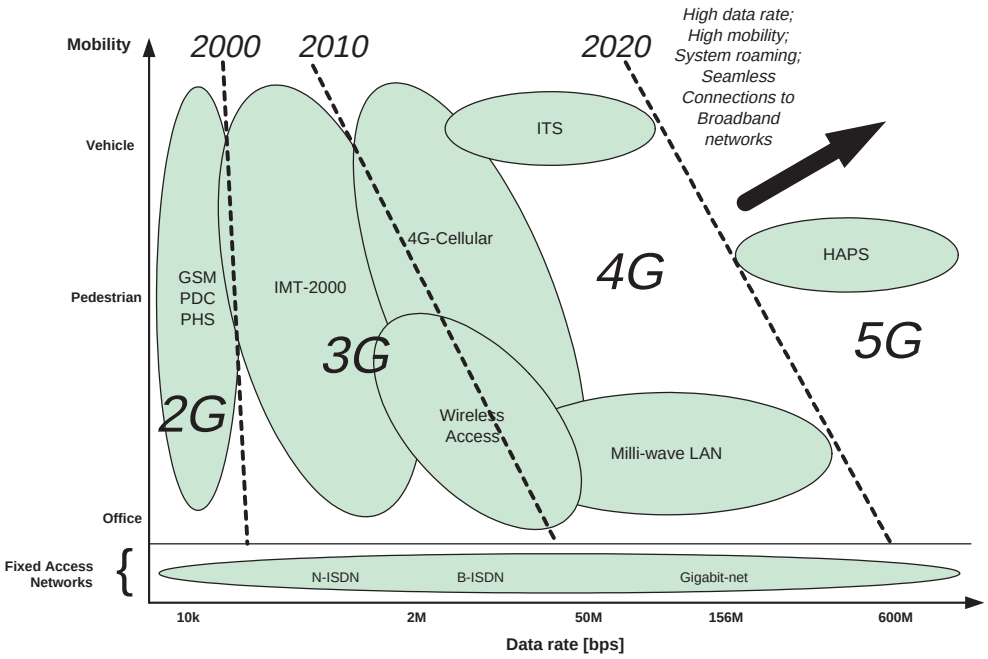

Figure 2.10: The future generations of mobile communications [171]

2.6.9 Multitier Networks

An obvious alternative can be observed in current research focusing on a vertical convergence of multiple networks. The integration of a set of networks, including WCDMA cellular systems, high speed wireless LANs, and home wireless networks will result in a seamless multitier network interface. Experiments and testbeds already exist [36], and the main challenge is integrating a heterogenous array of incompatible services, devices, and wireless technologies [85]. All access systems will be connected to a common, flexible, and seamless IP-based core network, where users will have a single identification (e.g. a string or a number) for all access technologies. A new media access system connects the core network to the appropriate access technology, and provides mobility management [87].

In this integration, the set of different access networks and technologies, ranging from fiber optic, cellular systems to HAPS and satellite systems, should be regarded as complementary rather than mutually competing options [160]. These multinetwork environments will be arranged in hierarchical levels, determining the trade-off between QoS and pricing, based on the mobilde device's current location [41]. Hierarchical levels could be defined as e.g. personal area network (PAN) zone, home area network (HAN) zone , WLAN zone, UMTS zone, GPRS zone, HAPS zone, satellite zone. For balancing between cost, QoS, and efficiency of complementary access strategies, operators must offer a variety of proprieraty solutions and find

their specific hybrid solutions. The provision of solutions for indivudual subscribers and customer segments will be a strategic imperative for operators in meeting their boundary conditions of customer amounts and revenue [69].

Another possible multinetwork hierarchy is given by table 2.6 [65, 131]. A commercial example of bridging different network technologies (although on a much higher logical level) by using the common Mobile IP protocol is presented by *Lifix Systems* [12].

Cell	**Example**
Wireless multimedia access	desktop; living room
Picocells	in-house
Microcells	city-centers; highways
Macrocells	suburban; regional; national
National and International Zones	countrywide
Global Information Village	worldwide

Table 2.6: Multinetwork hierarchy

2.7 Summary

The vision of 3G mobile communication is of multimedia terminals based on high bandwidth, whereas the so called 2.5G technologies enabled acceptable packet-switched systems and represent a significant step towards ubiquity, since connection establishment delays were drastically decreased.

The convergence of telecommunication and computer communication systems, i.e. the Internet and mobile communication, will intensify in the future. Applications in a world after the implementation of 3G can basically be regarded as similar to fixed Internet applications usage today. Future mobile communication systems continue the development towards increasing bandwidth and ubiquitous access (*see* figure 2.10), provided by seamless roaming between different systems. Another common denominator is packet switching, since IP seems to be an imperative for future systems. Location-dependent information and services might be a feature, which will be supported by even better future systems. The vision of future wireless and mobile communication systems could be described by the following key issues [135]:

- Responding to accelerated growth in the demand for broadband wireless connectivity,
- Provisioning of seamless services across a multitude of wireless systems and networks, from private to public, from indoor to wide area,
- Coping with the expected growth in Internet-based communications,
- Opening of new spectrum frontiers,
- Creating of new market opportunities.

This chapter presented a brief introduction to technologies associated with 3G, covering the three dimensions (increased bandwidth, introduction of packet switching, and increased terminal capabilities). The topics GPRS, UMTS and Wireless OS

were discussed as the main current developments in 3G technology. Going back to the model of the 3G Service Space (*see* section 2.1), we can now describe these technologies with our model, covering all the three dimensions defined (*see* figure 2.11). The dimension of *increased terminal capabilities*, addressed by the discussion of wireless operating systems, represents an enourmous step as mobile terminal and computer worlds merge together. The impact of this dimension is the most significant and certainly most interesting at this moment, being a threat as well as a challenge for current mobile middleware. This is one of the reasons for a special focus on this dimension throughout this thesis.

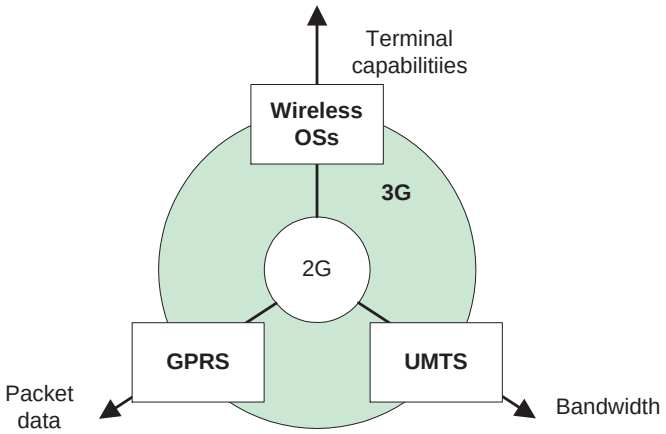

Figure 2.11: The three major branches of wireless and mobile technologies

Chapter 3

Applications

3.1 Introduction

What will mobile applications look like in 3G and beyond?—This is the key question addressed in this chapter. We will present and discuss a set of application technologies relevant for the scope of mobile business solutions, and illustrate the change of current middleware based applications to 3G applications. A set of concrete propositions for migrating from current technologies to future 3G solutions will be presented.

The convergence of application in 3G could be described by two—divergent—directions. All application technology candidates are balancing between the following two trade-offs:

- Mobility of information,
- Mobility of application computation.

The first dimension describes the balance between information being portable versus hosted. The second dimension describes the balance between using portable application computation versus the completely hosted approach. Both for information and application computation, the dimension of mobility has the advantage of immediate access and no dependency on the network response. On the other hand, the mobility of information has its constraints in mobile devices' storage and capacity, as well in security issues. That is why the mobility of representation sometimes is preferred, with the information remaining hosted. For application computation, mobility has its constraints in mobile device's computing power; any computing task will be solved in shorter time on the remote server host. The task now is to find an appropriate balance between these two divergent development directions. Full mobility of application computation with zero mobility of information (e.g. a PDA with Wireless OS and browser based access) does not seem very efficient, neither does full information mobility with zero application computation (e.g. a PDA with contacts database, but without e-mail client). Full mobility of both application computation and information brings us again back to laptops, which arises the question of the actual meaning of mobile devices.

Each section will be concluded by business opportunity recommendations for a mobile business middleware solutions provider, such as Smartner. Possibilities, feasibility, and alternatives for creating added value and competitive advantage will

be addressed by the recommendations, which will be followed up by a discussion of strategic implications in chapter 4.

3.2 Markup Language Applications

3.2.1 Introduction

The use of markup languages for implementing interactive applications has become very common since HTML. The original concept of markup languages in software applications was based on the definition of electronic documents with hyperlinks, hypertexts, for easier document browsing through selecting cross-references.[1] For many reasons, this concept seems to have become such a popular approach, that currently every application is developed based on this concept. One reason for this might be, that applications based on e.g. web servers enable easy deployment of hosted application services.

3.2.2 Wireless Application Protocol

The *Wireless Application Protocol (WAP)* was defined by the *WAP Forum*, founded in 1997 by Ericsson, Motorola, Nokia, and Phone.com [230]. Formerly, WAP attracted significant attention—many mobile operators joined the WAP Forum—for the following reason: wireless data services had not been successful despite mobile network operators expectations. The vision was to provide a better environment for integrating Internet content, and WAP was anticipated to significantly improve the wireless data market [132].

The protocol was designed in order to meet the requirements and constraints given by 2G mobile communication networks and devices. The protocol considered the following characteristics as intrinsic to mobile terminals [188, 230]:

- Low bandwidth
- Limited battery power
- Low QoS over wireless link
- Limited computing facilities (such as user interface, processor and memory)

At the moment, it is not possible to affect the local applications running on the end user's mobile terminal, as most mobile handsets are not programmable.[2] Given the availability of WAP as a common platform, a way of implementing any services is through a browser-style interface based on WML, using the WAP protocols, as this will be a baseline capability for a variety of mobile phone devices during the year 2002. In order to customize the WML output format at the operator's premises,

[1] Historically, the word markup has been used to describe annotation or other marks within a text intended to instruct a compositor or typist how a particular passage should be printed or laid out. Examples include wavy underlining to indicate boldface, special symbols for passages to be omitted or printed in a particular font and so forth. As the formatting and printing of texts was automated, the term was extended to cover all sorts of special markup codes inserted into electronic texts to govern formatting, printing, or other processing.

[2] This, however, is subject to change very soon, with emerging micro programming languages such as e.g. J2ME (*see* section 3.7.3).

device parameters can be passed in the HTTP header as tokens to describe the terminal's capability, with further negotiation for enhanced capability and content provision. MExE and Microsoft have recently introduced a set of parameters to describe and obtain network QoS [188].

The WAP framework provides a feasible high-level solution, allowing bearer indepencence and security. Concrete suggestions for improving the usability of WAP already exist [56]. However, the disapperance of the four fundamental constraints—which caused the need for WAP—will sooner or later force the WAP solution to become obsolete. Increasing bandwidth and QoS, longer battery life and increasing computing facilities (especially screensize) will improve the capability of mobile devices. It could be that WAP has a very short life, being superseded by HTML browsers, communicating over TCP/IP. WAP might thus become a legacy product within the next 3-5 years [188].[3]

It is increasingly important to provide the right content suited to each terminal's capabilities. The user is not satisfied, if the content is not compatible with his terminal device, and he cannot display it in a feasible manner. On the other hand, if the content is too simple and trivial, and the user's terminal could have displayed it in a more advanced style (with more colors, more text at once, etc.), then the end user is not satisfied either. Thus, terminal specific generation of content can be regarded as a prerequiseite for added value.

WAP introduces a microbrowser into the mobile phone, incorporating value-added services remotely by utilizing the intelligence in the existing Web server infrastructure. WAP is one of the key protocols which could be incorporated under the MExE standards umbrella (*see* section 3.8).

WAP data traffic shows strong similarity to WWW traffic: self similarity[4] and daily/weekly peridodicity. On the other hand, some characteristics differ drastically: browser-sessions are very short-lived and packet sizes are relatively small [127]. This is partly due to per minute charges and the limited memory for WML content.

Architecture

The architecture associated with WAP is a middleware approach, based on the *WAP Gateway*, interconnecting the mobile client with an existing Web server. This middleware approach helps overcome some of the issues due to limited bandwidth and high latency, and removes the need for the mobile terminal to implement a complete TCP protocol. The only modification required of the Web server is the delivery of content in the *Wireless Markup Language (WML)*. WML is an XML based format, defined for small hypertext and bitmap graphics documents [234]. Originally, application-level middleware was also included, allowing real-time filtering of HTML documents and conversion to WML (*see* figure 3.1). Although this feature has not been realized, it is part of the WAP 1.1 specification. This mechanism is only supported for HTML pages written in such a way as to provide meaningful output as WML. In addition, the use of frames and JavaScript might cause such a translation to fail [188]. The result is that a site *must* produce content in WML.

[3] In fact, many would argue that it already *is* obsolete.

[4] Traffic that is bursty on many or all time scales can be described statistically using the notion of self-similarity. Self-similarity is the property we associate with fractals—the object appears the same regardless of the scale at which it is viewed. In the case of stochastic objects like timeseries, self-similarity is used in the distributional sense: when viewed at varying scales, the object's distribution remains unchanged [64].

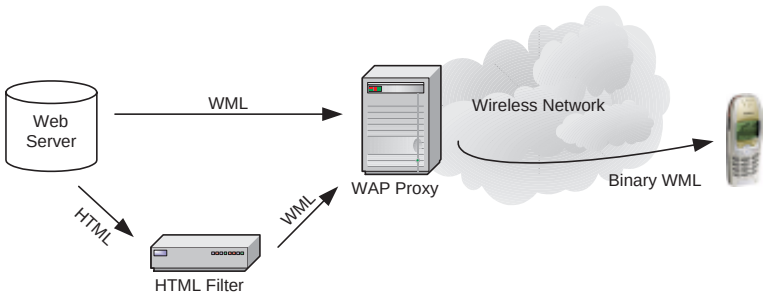

Figure 3.1: Two different ways for retrieving WAP content [188]

In reality though, basically all Web content has to be rewritten or manually transformed to WML. A reason for additional manual WML debugging are the terminal manufacturers different implementations of WAP browsers, resulting in a variance of supported features. If your WML page displays on a Nokia device, it does not necessarily have to do so on Motorola's terminal. This requires each server to produce device specific WML content, and generates an additional barrier to WAP services.

A hypertext only generates static pages and does not enable interactive end-user applications. For achieving such a functionality—in both HTML and WML—the additional use of the *Common Gateway Interface (CGI)*, the *PHP* language, or Java *Servlets* on the Web server is needed. Systems like these enable the dynamic creation of content, using content stored in a database.

Reference Model

The *Wireless Application Environment (WAE)* is an effort to standardize the WAP protocol on an industry-wide level. WAE specifies an application framework for a general variety of wireless devices, such as mobile phones, pagers, and PDAs. The WAE represents the application layer of WAP framework, incorporating underlying WAP related technologies into one ISO OSI conformant protocol stack, the *WAP Reference Model* (*see* figure 3.2).

Withing this stack, the WAE specifies the use of WML and WMLScript standards as well as acts as a container for applications, such as a browser. Futhermore, WAE defines a set of content formats, such as images, phone book records, calendar information, etc. Additionally, WAE also supports push technologies (*see* following paragraphs), allowing trusted application servers to send information directly to the application environment for processing [132]. The *Wireless Session Protocol (WSP)* is designed to utilize the transaction and datagram services. In combination with the *Wireless Transaction Protocol (WTP)*, the session/transaction layer perform the equivalent of an HTTP request agains a Web server. An optional security layer, the *Wireless Transport Layer Security (WTLS)*, is located above the transport layer, and preserves the transport service interfaces. The transport layer implements high-level datagrams and supports both the *Universal Datagram Protocol (UDP)* and the *Wireless Datagram Protocol (WDP)*. WDP supports connectionless reliable transport and bearer indepencence, allowing an arbitrary variety of underlying network technologies (both circuit switched and packet switched). Alltogether, the WSP provides transaction, session, and application management in a secure manner.

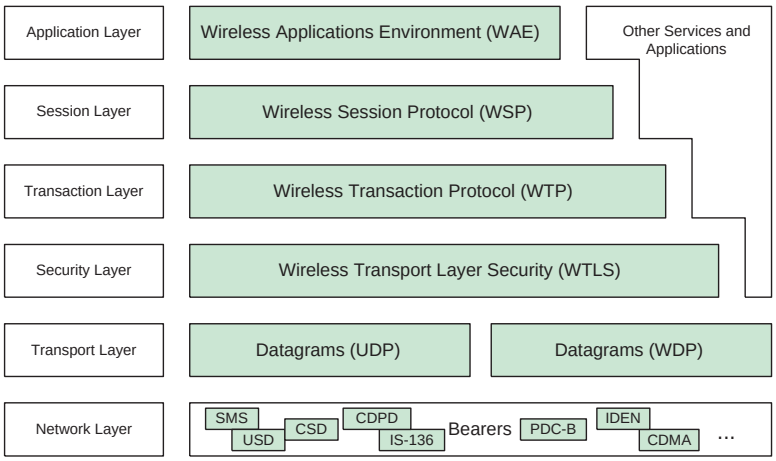

Figure 3.2: The WAP stack reference model [234]

The initial WAP standard can be regarded as incomplete, with key elements not yet standardized. The following paragraphs will briefly introduce improvements and enhancements of new WAP speficiations, namely WAP version 1.1, 1.2, and 2.0.

WAP 1.1 and WTA

Besides the requirement to implement the HTML filter (*see* above), the WAP 1.1 specification furthermore introduces the *Wireless Telephony Application (WTA)* server, allowing control of the voice channel using a data session.

WTA includes a client-side programming library and WTA server, allowing WAP sessions to control the voice channel. Voice calls can be placed, DTMF can be sent along the voice channel and conference calls can be initiated by the client. The WTA server generates WTA events which are interpreted by the WAP gateway, which sends the resulting WML to the WAP mobile phone. The WTA server controls the voice connections.

WTA currently is a challenging task to implement; vendors are unsure of customers' requirements for this functionality and find it challenging to implement Computer Telephony Integration (CTI) equipment without network operators' support. Another limitation is that current GSM networks only support one *either* voice or data channel at a time, and thus WTA will not become feasible before the availability of GPRS networks and terminals, supporting simultaneous voice and data transmission [188].

WAP 1.2 and Push

Realizing effective and usable mobile office applications, requires a transfer mode more advanced than the request-response[5] scheme typically used. Mobile data synchronization, for instance, does not make much sense as long as the user has to ini-

[5]This is also often referred to as *pull* technology

tiate the synchronization process himself by triggering an event.[6] It would be more valuable, if this data transfer happened autonomously, for instance as a background process, thus largely hidden from the user. As next generation mobile networks such as GPRS (*see* section 2.3) will let the mobile device be permanently connected to a packet-switched network, markup-language wireless applications requires capabilities for allowing the communication being initiated from outside the mobile station.[7] Mechanisms for proactive sending of information to mobile devices are called *pushes* (*see* figure 3.3), and the most common example for such a mechanism is SMS. Push allows applications to alert the user when time-sensitive information changes. Push mechanisms can also be useful for applications generating events, such as telephony applications, and emergency devices. Another aspect of push is multicasting. If the same information is requested by several users, the network can save network resources by broadcasting the information once instead of sending a copy to each subscriber.

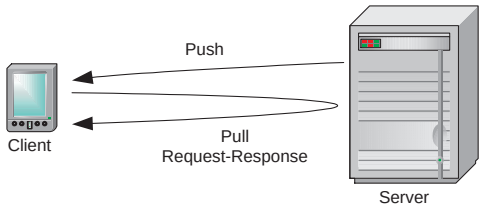

Figure 3.3: Push versus Pull technologies

As the WAP protocol needed an update to support push technologies, the WAP 1.2 specification contains a Push Architecture Framework [232], containing the definitions of a *Push Access Protocol (PAP)* as well as a *Push Over-the-Air (Push OTA)* Protocol.

The Push Access Protocol (PAP) PAP is designed to be tunneled through any common Internet protocol, but HTTP was chosen as an initially supported carrier. PAP carries XML-style entities and may be contained in a multipart document. A *Push Proxy Gateway (PPG)* connects the Push Initiator (on the Internet side) with the WAP client. It communicates over PAP via the Internet and over the Push Over-The-Air (Push OTA) Protocol to the WAP client. PAP supports the following operations [231]:

- Push Submission (Initiator to PPG)
- Result Notification (PPG to Initiator)
- Push Cancellation (Initiator to PPG)
- Status Query (Initiator to PPG)

[6] Obviously, a local clock or a remote packet could be the trigger instead, either of which is initiated by a user process.

[7] In locally running client applications, such as for instance e-mail clients, a background process for checking the arrival of new messages on the mail server every n minutes, makes request-response based communication sufficient. In markup-language applications such as WAP, the browser is a restriction which does allow these kind of background processes (since it is not implemented).

The Push Over-The-Air (Push OTA) Protocol The Push OTA Protocol is a *thin* protocol layer on top of the WSP. WSP sessions are used to deliver content. Each push WSP session is associated with only one Peer Address Quadruplet (one-to-one). Connection-oriented pushes require that an active WSP session is available, but a session cannot be created by the server. To solve the case where there is no active WSP session, the Push framework introduces a session Initiation Application in the client, that listens to session requests from the OTA servers and responds by setting up a WSP session for push purposes [233].

Nokia Activ Alert Nokia has recently released a beta version of the *Nokia Activ Alert* component, being a part of their *Nokia Activ Server 2.0* product package. Activ Alert is a Java API based package for providing WAP Push functionality. According to Nokia, it provides applications developers the world's first access to WAP Push.

As push functionalities will not be fully implemented before WAP 1.2, Activ Alert *Beta* is for development purposes, i.e. installing the Activ Alert component within the Activ Server framework on a Windows NT based server, and simulating pushes through Nokia's WAP Toolkit 2.1 as client. Nokia WAP Toolkit 2.1 listens for push messages. Activ Alert utilizes the Push Access Protocol (PAP) and an Activ Alert Manager is run as a remote client which accesses the Activ Alert WAP Push Gateway (PAP-gateway) functionality using Remote Method Invocation (RMI). Push messages can also be generated manually, on a command-line basis, determined by destination IP-address, MSISDN-address or user/group-address as well as the PAP-gateway (specified by `url:port`). There also exists a graphical user interface for generating pushes [165, 164].[8]

WAP 2.0 and XHTML

The recently released WAP 2.0 specification consists of a markup language migration from HTML and WML to Extensible *Hyper Text Markup Language (XHTML)*. This migration is supposed to support the convergence of WAP and fixed-Internet worlds and enable developers to build applications one time for rendering on multiple devices. The use of XHTML will also provide a common denominator between WAP and i-mode standards, providing possibilities for convergence between NTT DoCoMo's hugely successful platform and the WAP world.

Both WML and XHTML are derived from XML. The existing set of worldwide content WML can be accesed throught WAP 2.0, due to complete backward-compatibility. The use of XHTML will allow carriers to deploy new services via multiple channels (Web, WAP, PDA) while reducing the cost of maintaining those services. According to the WAP Forum, the goal is to have a one-application and one-markup-language environment for any screen size.

The new WAP implementation will also include TCP as a transport protocol, decreasing the importance of the WAP Gateway drastically (i.e. to the role of a dial-in point for circuit-switchen access). New features for developers and users were added, including an advanced-user interface and support for pop-up menus, color, graphics, animation, large-file downloading, and streaming media. Applications for both consumer and business segments will include graphic applications for wireless-chat products, a performance-management tool for scaleability prediction

[8] As long as WAP 1.2 mobile terminals supporting push are not on the market, the over-the-air feature obviously cannot be simulated within this framework yet.

and mapping applications. Other novelties include Bluetooth support and WAP-enabled remote-control frameworks for accessing information residing on desktop computers [209, 29].

3.2.3 Other Thin Markup Languages

There is a variety of other thin markup languages similar to WML. Most of them have XML or HTML as a common superset, and are based on restrictions to a minimal set of commands (*tags*) and supported features.

The *Compact Hypertext Markup Language (cHTML)* is another example of such a well-defined subset of the HTML 2.0 [48], HTML 3.2 [184], and HTML 4.0 [185] recommendations. cHTML was designed for small information appliances, and its development was initiated by the World Wide Web Consortium (W3C) as an alternative development to the trend of HTML expanding towards support for a richer multimedia document format. Supported features of cHTML are JPEG images, tables, image maps, background color and image, frames, stylesheets, and multiple character set support [236]. The latter one might be one of the reasons for why cHTML was chosen as the basis for the middleware in the *i-mode* service by NTT DoCoMo in Japan. Being a *clean* superset of HTML, cHTML pages can also be displayed by a normal web browser. As the difference between WAP 2.0 and cHTML is very small, there are already devices with dual-support for WAP and cHTML available, as well as gateways for converting cHTML to WML [136].

The *Handheld Device Markup Language (HDML)*) had a very similar goal to that of cHTML. However, the disadvantage of a special language approach like HDML compared to cHTML is that everything—contents, authoring tools, server software, client software, and textbooks—have to be prepared. For a product line ranging from high-end PDAs to low-end cellular phones, the consistent HTML-based approach makes more sense [236]. But as things turned out in Europe, the HDML standard was adapted into the WML [114]. Despite its origins in the U.S. market [41], HDML was the predecessor of WML.

3.2.4 Mobile Web Applications

In 1990, Tim Berners-Lee, a computer scientist, designed a hypertext library system for storing documents at CERN.[9] Today, this concept is more widely known under the name World Wide Web (WWW). The Hypertext Markup Language (HTML) was defined for supporting cross-references within a document, as well as between documents on different servers, by using world-wide unique identifiers.[10] The WWW was not designed for interactivity, it simply used the TCP protocol as an underlying layer for its HTTP protocol [77]. The HTTP specification has changed over time in several versions and updates, all basically extending the orginal specification with the addition of plug-ins such as applets, forms, animations, etc. Based on this background, one could argue that remote applications based on a WWW interface, such as reading and especially writing e-mail over hosted applications like Hotmail, are very artificial solutions. They more or less *abuse* the hypertext concept to act as something it was not originally intended for. But the fact is, that the web

[9] Centre Européen pour les Recherches Nucléares (CERN)
[10] This unique identifier is also known as *Uniform Resource Locator (URL)* [77].

browser is a world-wide common platform today, and that it would take an enourmous amount of time for any platform to become as famous. Another fact is, that these kinds of WWW based applications have worked to a certain extent so far, and developments such as the applet technology, represent more application oriented approaches. However, using the background of HTML, one could argue that using WAP for running online applications is an even worse solution, since the constraints of the HTTP based request-response type of interface have been narrowed down even more by the WAP specification. Obviously, WAP represents only a subset of the WWW, if even that, and should *not* be associated with the term *Wireless Internet*.

If—after all—the WAP concept is on its way to migrate *back* to the WWW concept, the question remains why not to completely switch over to HTML based browsers for future mobile phones and PDAs.[11] The only remaining argument for WAP, the screensize of mobile terminals, can be addressed by adaptation middleware. This middleware can run at an operator's premises (*see* figure 3.1), or can be a part of the client. Recently, many approaches for presenting web page content over small screen sizes have been introduced [50]. Furthermore, approaches for enabling more efficient user interfaces (especially for entering data) over limited web browsers exist [118].

3.2.5 Extensible Markup Language

The Extensible Markup Language (XML) is the superclass of all hypertext and markup languages [239]. Thus, every document existing in an XML derived format (such as HTML, WML, etc.) can also be represented by an XML document. The tree formed recursive data structure of XML[12], allows almost any abstract, generic structured data set or database to be represented by a single XML document. Given the fact that application developers and content providers do not have the time and resources to author content and deliver it in many different ways, to a variety of terminals with different capabilities, the separation between general, high-level content, and device specific generated content output is both feasible and important.

One approach for this type of application–middleware separation is through the usage of the *Extensible Stylesheet Language (XSL)* (*see* figure 3.4). Being a language for specifying stylesheets, XSL consists of two parts: a language for transforming XML documents and an XML vocabulary for specifying formatting semantics. An XSL stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary [240].

The content is once being written or automatically generated in XML at the server. When a document is requested, the XML document passes through an XSL processor, which acts as a filter. This filter contains a predefined set of *stylsheets*, allowing it to generates the terminal specific presentation of the content, e.g. a HTML or WML document, which then is being forwarded to the browser on the client [234]. Web

[11] There are already PDAs and smartphones on the market, which incorporate HTML support in a built-in web browser. The set-up and usage of such web browsers is much easier and seems more natural to users with experience with e.g. Netscape's Communicator or Microsoft's Internet Explorer. While working with the Nokia 9210 Communicator during the project (*see* section 3.7.8), enourmous difficulties in setting up the WAP client were experienced. The built-in web browser worked perfectly from the first day on (*see* figure 2.8). In addition, the WAP browser displayed content in a manner, much too simple for the $640 \cdot 200$ pixel color screen, thus it did not fully utilize the device's capabilities.

[12] Every XML document possesses one root, which refers to a set of nodes. Every node has the same characteristics as the root, i.e. it refers to a set of subnodes.

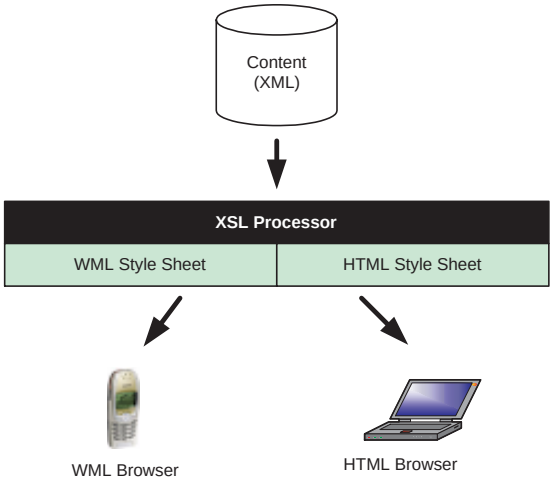

Figure 3.4: XML/XSL separate content and document generation [234]

content providers are switching over to complete XML use, avoiding the content to be written separately in different content formats [148].

The modeling of data in an XML conform manner is one of the key enablers for future applications. This is especially the case, when mobile applications will migrate to something else than markup language based solutions (*see* section 3.7). Consider locally running applications in different terminals on different wireless OSs, XML interpreters exist for almost any platform available, and this heterogenous set of terminals can easiliy be connected to existing middleware. Since XML is an open standard of the World Wide Web Consortium (W3C), there are lots of open-source XML parser implementations freely available.

3.2.6 Standard Generalized Markup Language

The *Standard Generalized Markup Language (SGML)* [102] is very similar to XML. The main differences are that SGML is defined by the International Organization for Standardization (ISO), whereas XML was defined by the World Wide Web Consortium (W3C) building upon the earlier SGML. Like XML, SGML is a metalanguage, that is, a means of formally describing a language, in this case, a markup language.

The SGML notation differs from the enclosing *tag* based notation in XML, which we will not evolve further in this document. In order to support the SGML standard in current applications, a transition for automated converting SGML to XML is defined by the World Wide Web Consortium (W3C) [235].

3.2.7 Recommendation

For thin client solutions, i.e. devices with limited memory space and processing power, markup language solutions offer a feasible way to implement mobile, interactive applications. However, devices that contain an implementation of an e.g. WAP-browser are in most cases not that limited anymore. Many such devices contain additionally built-in standalone applications (such as calendar, e-mail client,

etc.) which offer better usability and functionality than any remotely implemented WAP solution. As soon as this type of application platform becomes programmable (*see* section 3.7.3), the added value given by markup language solutions will become somewhat questionable. WAP is an intermin solution to the scarce use of radio bandwidth and the lack of independent screen and GUI.

In the WAP case, the original approach of *one* standardized markup language for platform independent mobile applications design has not proven its concept so far. Current WAP enabled devices on the market contain WAP implementations slightly differing from eachother, supporting different command subsets. The common standard approach, however, will probably improve with newer versions (i.e. WAP 1.2 and 2.0). However, future WAP implementations will integrate TCP, UDP, and HTML to some extent, thus making it more and more a repackaging of existing Internet protocols.

From the operator perspective, when scaling services, the WAP gateway is likely to become a bottleneck, constrained by processor saturation [188]. Another problem is the WAP access from abroad; data calls to the home dial-in point become expensive and roaming mechanisms are needed. However, the user would in such a case have to reconfigure his terminal every time when changing the country. WAP gateways and access points will have to be organized into complex, hierarchical clusters; multinational operators will have large advantages in offering such services.

GPRS devices supporting simultaneous data and voice sessions are expected at the end of 2001 [188]. Applications like WTA will provide more added value to WAP applications than current usage frameworks. However, the emerging packet-switched networks will in the near future allow *Voice-over-IP (VoIP)* applications, which might become more feasible and attractive for integrating data services with telephony [86, 195].

The upcoming features of WAP 1.2 alone will probably not bring any revolutionary changes; WAP Push (which is based on simple SMS messages containing an URL) will bring more interactivity to current applications. It is the combination of newer WAP implementations with GPRS that will transform WAP to a solid, more realistic mobile applications platform with an improved user experience. After all, GPRS will be somewhat spread during the end of 2001, and the screensizes will still be more or less the same, which indicates a market for several WAP solutions to become spread during 2002. After that, however, PDA devices with integrated communication facilities will replace WAP by wireless terminal applications. The MExE initiative (*see* section 3.8) is an interesting approach for linking WAP with a framework of future terminal applications.

From the application provider perspective, it will be feasible to keep the content generation and representation as separated as possible. The use of XML and XSL (as described in section 3.2.5) provides an open interface for future wireless applications, which even might not be markup language based at all (*see* sections 3.7.7 and 3.7.8). If any markup language is used, XML should be used as the core technology. Any commitment to device specific implementations will in the long term cause an extra amount of conversion work, which will be increasingly unnecessary.

Without questioning the technology and background of markup language based applications (such as WAP or Web), it still remains questionable whether they are needed at all. These kind of middleware approaches will probably sooner or later have to give way to emerging terminal application platforms, such as e.g. Java (*see* section 3.7.3). On the other hand, as long as web based solutions are being used on desktop PCs, it might still be probable that the end user prefers a similar solu-

tion on the mobile terminal as well [189]. Current PC oriented web based solutions are mainly being deployed due to their benefits on application distribution, application service provision (ASP), and outsourcing. However, these benefits can also be reached by other technology approaches (*see* sections 3.6 and 3.7.3), which only are a matter of adoption and standardization.

3.3 Over-the-Air Synchronization

3.3.1 Introduction

As an example for our discussion about the absolute need for higher transmission speed for wireless networks, over-the-air synchronization needs to be examined. A mobile office solution does not necessarily require an unlimited transmission speed, as long as the data is accessible in a feasible and suitable way. Going back to the 3G service space model (*see* section 2.1), the dimension of emerging terminal capabilities could enable this access method rather than the dimension of increasing bandwidth. However, *if* present wireless networks were free of charge, had infinite bandwidth, had very high quality of service, had universal coverage, and provided permanent connections, we would not need synchronization. But present wireless networks do not, and are unlikely to do so in the near future. The potential goal is to make information accessable from everywhere. Devices are becoming more and more capable, and they are also becoming more location-aware. Networks are providing more bandwidth, but the growth is small compared to the evolution of the mobile devices. Markets are becoming more efficient, which finally leads to the idea of customizing content.

The perception of data synchronization being about *importing* and *exporting* data is a historic view. Synchronization used to be from one application to another, and it worked only because one application was used at a time. Conflict resolution was rarely needed, if at all. Considering truly portable device, the situation is changing. The goal is to enable entry and access of data away from a fixed computer. Thus, data conflicts between the fixed computer and the mobile PDA are likely. This is where over-the-air synchronization begins.

The traditional synchronization landscape took form in 1994, when the focus was on importing and exporting data. In 1995, with the introduction of the Palm Pilot, the desktop became tied to the PDA and more or less everything changed. Around 1998, the Internet exploded and the idea of portals broke onto the scene. In the same year, the first 2G mobile phones allowed more sophisticated data storage and automated importing of address book information. The new synchronization landscape will focus on serving the exploding wireless device universe. The expectation is that there will be more than 1 billion wireless devices in use by the year 2003. Most of them will be able to store and edit *Personal Information Manager (PIM)* data. Wireless data synchronization has become an integrated part of the mobile network structure [214].

3.3.2 Consistency Theory

As over-the-air synchronization is a relatively recent issue, there is still a lot of theoretical research to be done. The real characteristics and implications of this type of wireless traffic is still mostly unkown. The search for mathematical theories and

models is on-going, but mostly has its roots in statistics. Obviously, when considering synchronization of business applications, such models have to be applied with care. Statistical (or probabilistic) reliability does not necessarily satisfy real usage. For example, a business user could argue that, when accessing PIM data, you want to be completely sure its the most current version of the data, and cannot accept a high probability of being the most current version. On the other hand, for example an error probability of 11^{-5}, would imply one half an hour wrong in almost a decade (assuming ≈50% fully scheduled year calendar with hourly events).[13]

Automated Synchronization and Database Freshness

A recent study at Stanford University has developed a model for analyzing sychronized database freshness [60]. The study identifies synchronization of a database as a probabilistic process, and defines policies for performing synchronization. Based on the elementary definition of *freshness* F of a local element e_i at time t

$$F(e_i, t) = \begin{cases} 1 & \text{if } e_i \text{ is up-to-date at time } t \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

and the freshness of a local database $S = \{e_1, e_2, \ldots, e_N\}$ at time t is

$$F(S, t) = \frac{1}{N} \sum_{i=1}^{N} F(e_i, t) \in [0, 1], \tag{3.2}$$

we can regard the problem as a *Poisson process* with probabilistic evolution of an element, which for a change rate λ results in the expectation value for the age A of a database at time $t \in (0, I)$ within a time interval I

$$E[A(e_i, t)] = \int_0^t (t - s)(\lambda e^{-\lambda s}) ds = t(1 - \frac{1 - e^{-\lambda t}}{\lambda t}). \tag{3.3}$$

This method has been used for analyzing web search engine data repositories and indexes, containing mirror copies of the web. To keep such repositories and indexes up-to-date, a *Web Crawler* has to revisit the web pages based on their probability of change. The *Google* search engine used this local repository at Stanford before it became a commercial product. Change-frequency-models are described, and the following set of dimensions could be relevant for wireless data synchronization [60]:

Synchronization frequency. It is necessary to know how frequently the local database should be synchronized. Obviously, the more often the synchronization is performed, the fresher the local database is. The assumption is made that N data elements are being synchronized per I time-units.

Resource allocation. After having decided how many elements to synchronize per unit interval, the decision about how often to synchronize each individual element has to be made. For instance, consider a database consisting of three elements e_1, e_2, and e_3. It is furthermore known that the elements change at the rates $\lambda_1 = 4$, $\lambda_2 = 3$, and $\lambda_3 = 2$ (which corresponds to the real-world example of for instance `personalprofile`, `calendar`, and `contacts`, which obviously change differently). There are two possible solutions, *uniform allocation policy* and *non-uniform allocation policy*, which decide when to initiate synchronizing at the same rate or, respectively, at different rates.

[13] $10^{-5} \cdot 12h \cdot 30 \cdot 12 \cdot 10 = 0.432h$.

Synchronization order. The decision still has to be made about the order in which the elements should be synchronized. There are policies like *fixed order*, synchronizing in a given, constant order, *random order*, synchronizing repeatedly but differently in each iteration, and finally *purely random order*, where an arbitrary element is selected at each synchronization point.

Synchronization points. Finally, a policy for selecting the appropriate time points for initiating synchronization has to be made up. In some cases, repeated synchronization is only needed in a certain time window, for a business user for instance, the synchronization does not have to be performed as often during the night as during business hours.

Given the fact that over-the-air synchronization need not be initiated by the user, a scheme for automating this process is desirable. This theoretical background could in combination with a cost model, be a basis for making decisions about the time interval for performing wireless data synchronization. The cost model should consider the cost of synchronization versus the cost of *no* synchronization (i.e. the cost of *not* being synchronized).

Subset and Superset Consistency

The synchronization of PIM data could be characterized as client side access to a *subset* of a corporate server database (the *superset*). The synchronization always concerns only the *relevant* superset of the server's database. But in some cases, determining the relevant superset is impossible, i.e. entered client data cannot automatically generate corresponding server data. This is the case when e.g. `lastname` and `firstname` from the superset becomes `displayname` in the subset. This type of mapping makes the synchronization procedure very complicated and sometimes even useless. Depending on the purpose of the requested data, this type of one-way synchronization is feasible, i.e. the data should not be updated in the client as it cannot be synchronized with the server without user interaction. Thus, the data structures for supersets and subsets have to be carefully modeled in the scope of their usage.

In order to describe a model for illustrating subset and superset synchronization, we present a small mathematical aside. We begin with the following definition:

Definition. For a given $m \times n$ matrix $\mathcal{M}$, with $n > m$ (where m is the number of rows and n the number of columns), we define:

(i) $\mathcal{M}$ is *δ-reducible*, $\Longleftrightarrow$:
Every row contains only one single element x, with $x \neq 0$.

(ii) $\mathcal{M}$ is *τ-reducible*, $\Longleftrightarrow$:
There exist $n - m$ empty columns, for which all elements x of that column satisfy $x = 0$.

(iii) $\mathcal{M}$ is *reduction irrelevant*, $\Longleftrightarrow$: $\mathcal{M}$ is not τ-reducible.

Obviously, if $\mathcal{M}$ is δ-reducible, it implies the existence of $n - m$ empty columns, which makes it τ-reducible as well. But if $\mathcal{M}$ is τ-reducible, it may additionally contain mulitple elements in one column. The characteristic of *δ-reducible* is thus stronger than *τ-reducible*.

Modeling the data set on the mobile client as a tuple $C = (c_1, c_2, \ldots, c_m)^T$ and the server database as a (larger) tuple $S = (s_1, s_2, \ldots, s_n)^T$, one could think of describing the synchronization as a linear (but not necessarily bijective) mapping. With the additional appropriate definition of the elementary operations $\bullet$, $\times$, and $+$, the linear model for synchronizing C_{old} and S into C_{new} will have the following structure:

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_m \end{pmatrix}_{new} = \Gamma \times \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ \vdots \\ s_n \end{pmatrix} + \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_m \end{pmatrix}_{old}, \tag{3.4}$$

for a given transition matrix

$$\Gamma = \begin{pmatrix} \psi_{1,1} & \psi_{1,2} & \psi_{1,3} & \cdots & \psi_{1,n} \\ \psi_{2,1} & \psi_{2,2} & \psi_{2,3} & \cdots & \psi_{2,n} \\ \psi_{3,1} & \psi_{3,2} & \psi_{3,3} & \cdots & \psi_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \psi_{m,1} & \psi_{m,2} & \psi_{m,3} & \cdots & \psi_{m,n} \end{pmatrix}. \tag{3.5}$$

In most cases of real-world subset synchronization, the transition matrix Γ is δ-reducible, where synchronization can be interpreted as a simple selection of elements from a superset into a subset. In some cases, elements of the subset C consist of a concatenation of elements of the superset S. Consider the previously mentioned example, where `lastname` and `firstname` from the superset become `displayname` in the subset. Using our definitions from above, this case could be described within our model as follows:[14]

$$\begin{aligned} +: \quad S \times S &\rightarrow C, \\ (\texttt{lastname}, \texttt{firstname}) &\mapsto \texttt{displayname}; \end{aligned} \tag{3.6}$$

$$\begin{aligned} \bullet: \quad S \times \texttt{Inv} &\rightarrow C, \\ (s, \psi_{i,j}) &\mapsto \psi_{i,j}(s) \qquad \forall i \forall j; \end{aligned} \tag{3.7}$$

$$\begin{aligned} \psi_{i,j}: \quad S &\rightarrow C, \\ s &\mapsto s, \\ \text{i.e. } \psi_{i,j} &\equiv id \qquad \text{for specific } i \text{ and } j. \end{aligned} \tag{3.8}$$

In this case (equations 3.6 to 3.8), Γ becomes either τ-reducible or reduction irrelevant. If $m \neq n$, Γ is not invertible, which means that the subset is not consistent without its superset.[15] If $m \neq n$ and Γ is δ-reducible, the superset vector S and Γ can be reduced to the relevant columns and elements only, where the reduced matrix Γ^*

[14] By writing `Inv`, we refer to the *set of all invertible functions*, i.e. $\psi_{i,j} \in \texttt{Inv}\ \forall i \forall j$ and $\exists \psi'$ such that $\psi' \bullet \psi \equiv id\ \forall \psi \in \texttt{Inv}$.

[15] This means, that a new element in C cannot create a complete corresponding element in S.

becomes invertible, i.e. $m = n$. In this case, elements from the subset can uniquely describe the relevant elements of the superset. If $m \neq n$ and $\mathbf{\Gamma}$ is τ-reducible, the superset vector S and $\mathbf{\Gamma}$ can be reduced to $m = n$ as well. Thus, the reduced $\mathbf{\Gamma}^*$ is quadratic, but it is still not invertible. Due to the concatenation of elements, the subset is still not consistent with its superset!

The *relative effort* of the synchronization could be defined based on the relative weight of the matrix:

$$\eta = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \sigma(\psi_{i,j})}{n \cdot m} \in [0,1], \text{ where} \tag{3.9}$$

$$\sigma(\psi) = \begin{cases} 1 & \text{if } \psi \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

This means, that for $\eta = 1$, the relative effort to synchronize S with C has reached its maximum, i.e. every element of S is undergoing an operation $\psi_{i,j}$ with every element of C. If $\mathbf{\Gamma}$ is δ-reducible, only m out of n elements are undergoing an operation $\psi_{i,j}$, thus $\eta = m$. If every column of $\mathbf{\Gamma}$ contains exactly one element x, such that $x \neq 0$, every element of the superset S is undergoing one operation $\psi_{i,j}$, and $\eta = n$. Finally, for $\eta = 0$, no synchronization takes places at all, S and C are mutually independent.

Results Based on this model, on the one hand, a carefully defined set of the mathematical operators involved, can provide a reliable functional mechanism when implementing this kind of subset/superset based synchronization. A matrix has the natural advantage of being relatively fast to compute [204]; the lowest complexity[16] of matrix multiplication reached is $O(n^{2.494})$ [44]. When scaling a solution based on this functional mechanism to a larger scale, the matrix multiplication could thus be a more efficient approach, than sending and receiving database queries and commands throughout the network. However, unless the transition matrix $\mathbf{\Gamma}$ is δ-reducible, the implementation of any other than one-way synchronization (dominated by the server) seems technically rather difficult. In our framework, $\mathbf{\Gamma}$ could be associated with the synchronization middleware, residing either at the operator, back-end server, or client.

On the other hand, one could argue that it would be a somewhat naïve approach to describe sychronization in one single formula.

Based on this approach, it still seems impossible to easily find a traffic model; the subset/superset model should be interpreted as purely an illustration. Since over-the-air synchronization still is a rarely new researched area, statistical models for estimating the generated traffic load would be desirable. Operators, who could offer synchronization as a new, value added service, should not need (and not be allowed to) care about the content, they should consider the rate of change for each data object. This information would surely be of interest to operators, who could develop models for pricing.

[16] $O(f) = \{g | \exists a_1 > 0 : \exists a_2 > 0 : \forall N \in \mathbf{Z}^+ : g(N) \leq a_1 \cdot f(N) + a_2\}$ [173].
The $O(f)$ notation in computer science is used for describing the complexity of a solution as a proportionality function f of a given problem size n. Consider the following example, a square football field with vertigo size n, what is the complexity for cutting the grass field? Since the problem size is depending on the area of the field, is it proportional to n^2. Thus, the problem can be solved in $O(n^2)$.

3.3.3 Platform Independent Data Formats

Every time two or more individuals communicate, *Personal Data Interchange (PDI)* occurs, either in a business or personal context, face-to-face, or across space and time. Interchanges of this kind frequently include the exchange of informal information, such as business cards, telephone numbers, addresses, dates, and times of appointments. Augmenting PDI with electronics and telecommunications can help ensure that information is quickly and reliably communicated, stored, organized, and easily located when needed.

The *Versit Consortium* developed a set of PDI technologies based on open specifications and interoperability agreements to help meet this need in order to allow you to communicate more easily, faster, and more accurately. Two main data format standards were defined, `vCard`, an electronic business card, and `vCalendar`, an electronic calendaring and scheduling exchange format.

Beginning in December 1996, the *Internet Mail Consortium (IMC)* initiated development and promotion of these two important format standards [110]. Today, they have been incorporated into many platforms and applications; even used for SMS messages based on *Smart Messaging* [163].

Electronic calendar entry: `vCalendar`

`vCalendar` represents a transport and platform-independent format for exchanging calendaring and scheduling information in an easy, automated, and consistent manner. Capturing information about event and *to-do* items that are normally used by applications such as PIMs and group schedulers, `vCalendar` allows programs to exchange data about events so that meetings can be scheduled with anyone who has a `vCalendar`-aware program. `vCalendar` is an open specification based on industry standards such as the x/Open and XAPIA *Calendaring and Scheduling API (CSA)* [172], the ISO 8601 international date and time standard [103] and the related MIME email standards [109, 105]. This allows features such as [110]:

- Effortlessly adding appointments and other events found on the Internet or Intranet into your PIM.
- Readily scheduling a meeting with a group of people using different calendaring and scheduling applications.
- Being able to easily move appointments and to-do tasks from email message attachments directly into your favorite personal productivity application.

To explain the advantage of a common calendar format versus living without it, IMC specfies a set of representative use cases in their specification on their web page [110]:

Using Email to Schedule a Meeting. You need to schedule a meeting with fellow employees from several divisions, customer representatives, and key vendors. The people you want to invite use a variety of scheduling applications or different operating systems. You broadcast an e-mail with a `vCalendar` attachment to all the invitees. The `vCalendar` attachment is processed by the application environment and can be automatically added as a new calendar entry into your agenda.

Project Management. You are a project leader responsible for an industry-wide project spanning a group of diverse companies. Your team uses a website on the Internet to distribute project information to the team members. Each company has its own calendar application standard and uses a different operating system. Using `vCalendar` in conjunction with the website, project members can effortlessly download individual events and to-do items.

Event Planning. Surfing along the Internetet you come across a website for an interesting trade show. As marketing coordinator of corporate events, you want to know more about this show and what it offers. The website has a calendar of events including information on trade show registration, travel planning, deadlines for booking either floor space or local hotel accommodations, individual seminar schedules, and other related activities. So you click on the associated `vCalendar` icon and, in a flash, receive a sequence of calendar events, as well as to-do or action items required to register or prepare for the show as either a participant or an exhibitor. Automatically, the information, in a flexible `vCalendar` format, is easily integrated into your calendaring and scheduling application. Now, to spread the news, you email the `vCalendar` object to colleagues who may also have an interest in this event.

Mobile Computing. You are at a business meeting with representatives from various companies. Every imaginable portable computing device is there: PDAs, hand-held organizers, laptops, and notebook PCs. Follow-on events and action items are scheduled during the meeting. The chairperson uses a PDA device or laptop with an infrared port to beam the information over to one representative's calendaring and scheduling application and it is, in turn, passed around to the other attendees. At this meeting, you have made contacts, set-up appointments or events and created your own action items. You accomplish this by using the `vCard` and `vCalendar` cross-platform standards as infrared transport formats. As a mobile user, this information is easily relayed to your colleagues and contacts over your company's network or Intranet via a dial-up connection.

Web Calendar Publishing. You manage the sales of entertainment tickets at a local concert hall. As part of a customer service strategy, your organization has set up a website that contains a calendar of events complete with a list of scheduled artists, performance dates, last minute cancellations or additional events. Your website also offers the option of downloading events, using the `vCalendar` format, so that web browsers can dynamically update their PIM-based calendars with current concert information. In addition, with the use of the `vCard` format, they can also download useful ticket information, as well as venue information and travel directions to the concert hall.

Electronic business card: `vCard`

The idea of `vCard` is to automate the exchange of personal information typically found on a traditional business card. The `vCard` format can be used in applications such as Internet mail, voice mail, Web browsers, telephony applications, call centers, video conferencing, PIMs, PDAs, pagers, fax, office equipment, and smart cards. `vCard` information goes way beyond simple text, and can include elements like pictures, company logos, Web addresses, etc. The `vCard` V2.1 specification from

the Internet Mail Consortium and the vCard V3.0 specification were approved as a proposed standard by the Internet Engineering Task Force (IETF). These specifications were developed in cooperation with leading producers of desktop software (PIMs, telephony products), hand-held organizers, Internet web clients, Email systems, on-line information and directory services, and other interested parties [106]. An example of a vCard source code is illustrated in figure 3.5. Main features are [110]:

- vCards carry vital directory information such as name, addresses (business, home, mailing, parcel), telephone numbers (home, business, fax, pager, cellular, ISDN, voice, data, video), email addresses, and Internet URLs.
- vCards can also have graphics and multimedia including photographs, company logos, audio clips (e.g name pronunciation).
- Geographic and time zone information in vCards let others know when to contact you.[17]
- vCards support multiple languages.
- The vCard specification is transport and operating system independent, hence you can have vCard-ready software on any computer.
- vCards are Internet friendly, standards based, and have wide industry support.

```
BEGIN:VCARD
   FN:Mr. John P. Smith, Jr.
   TITLE:General Manager
   ORG:XYZ Corp.;North American Division;Manufacturing
   ADR;POSTAL;WORK:;;P.O. Box 10010;AnyCity;AnyState;00000;U.S.A.
   LABEL;POSTAL;WORK;ENCODING=QUOTED-PRINTABLE:P.O. Box 10010=0D=0A=
    Anywhere, TN  37849=0D=0A=
    U.S.A.
   ADR;PARCEL;WORK:;133 Anywhere St.;Suite 360;AnyCity;AnyState;00000;U.S.A.
   LABEL;POSTAL;WORK;ENCODING=QUOTED-PRINTABLE:133 Anywhere St.=0D=0A=
    Anywhere, TN  37849=0D=0A=
    U.S.A.
   TEL;Work;VOICE;MESG;PREF:+1-234-456-7891 x56473
   TEL;Home:+1-234-456-7891
   TEL;Pager:+1-234-456-7891
   TEL;Cell:+1-234-456-7891
   TEL;Modem;FAX:+1-234-456-7891,,*3
   EMAIL;Internet:webmaster@anywhere.com
   URL:http://www.anywhere.com/mrh.vcf
   UID:http://www.anywhere.com/mrh.vcf
   TZ:-0500
   BDAY:1997-11-29
   REV:20010313T095443
   VERSION:2.1
END:VCARD
```

Figure 3.5: Example of a vCard object

Usage examples for vCard are [110]:

[17] This assumes you mostly are in the specified zone, which need not to be the case.

Infrared Exchange. Walking into a meeting and beaming `vCards` over infrared links between hand-held organizers, PDAs, and notebook PCs from any manufacturer. Within seconds, the participants have this vital information automatically stored in their favorite directory. Later it can be used to place a phone call, send a fax or Email, or even to initiate a video conference.

Internet Mail. Surfing the Internet you might suddenly run across a picture of some cool business site that is advertising on the Web. You click the `vCard` button and a `vCard` is returned to your client machine. It can be easily stored into your favorite address book or directory for later reference and includes all the vital directory information on that business. Another Internet scenario allows your `vCard` to be dragged over an Internet form (such as a registration or order form) and automatically populate it with the correct information. Electronic mail can carry your `vCard` as an attachment or embedded via MIME [111]. It can be automatically extracted by the recipient and placed into their desktop directory of choice.

Computer/Telephony Applications. Using a notebook PC with an Integrated Services Digital Network (ISDN) or Digital Simultaneous Voice and Data (DSVD) modem to browse product highlights on a company's homepage, you decide to place an order. Thus, you use your PC to dial the sales call center. When the sales person asks you to provide your shipping information, you use your PC to simultaneously send your `vCard` which enables the receiving application to quickly and accurately populate the order form with all your pertinent personal data.

Video and Data Conferencing. A video or data conference can start with an exchange of `vCards` which could be kept *on the table* by a `vCard` viewer window [106].

3.3.4 SyncML

SyncML stands for *synchronization markup language* and is a specification for a common data synchronization framework. Based on XML[18], SyncML describes a representation protocol, for synchronizing data on networked devices. Started by a core sponsor group from the telecom and handheld devices industry, this initiative aims to design a common synchronization system for use between mobile devices that are intermittently conected to the network, and network services that are continuously available on the network. Although SyncML can also be used for peer-to-peer synchronization, the main design goal is to handle the case where network services and the device store the data they are synchronizing in different formats and/or use different software systems [216].

Motivation The goal behind the SyncML initiative is to enable the rapid evolution of a marketplace for the deployment of mobile pervasive computing solutions, while reducing costs and increasing connectivity options. Targeting a large set of scenarios within the synchronization market, the approach aims to benefit all parties. The idea was to develop and exploit a framework for local synchronization between devices, wide area synchronization as well as mobile business applications. Furthermore,

[18] Extensible Markup Language (XML) [239]

the design focuses on synchronization of back end data sources (applicable to all previously mentioned models) and on synchronization as a platform (e.g. systems management) [214].

The Initiative The SyncML initiative was carried out in a joint engineering project, consisting of a core group of sponsors:

- Telefonaktiebolaget LM **Ericsson**,
- International Business Machines (**IBM**) Corporation,
- **Lotus** Development Corporation,
- **Matsushita** Communications Industrial Co. Ltd.,
- **Motorola** Inc.,
- **Nokia** Group,
- **Palm** Inc.,
- **Psion** PLC,
- **Starfish** Software Inc.[19]

Furthermore, the initiative is supported by over 400 companies. SyncML is regarded as a marketing initiative and has high acceptance in the analyst community [214].

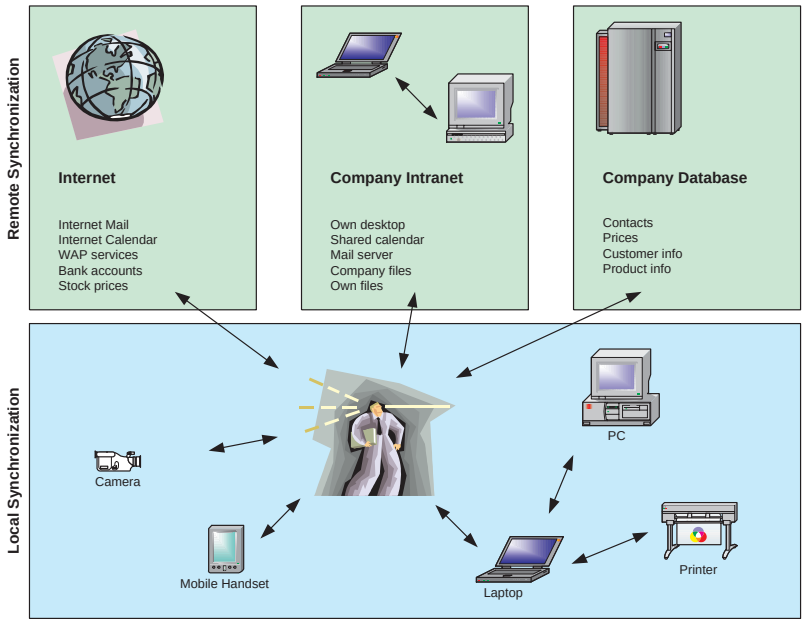

Figure 3.6: Scenarios for synchronizing with SyncML

[19]Starfish Software is a fully owned subsidiary of Motorola.

Overview

SyncML is based on XML technology and supports a variety of transport protocols (e.g. WSP/WAP, HTTP, OBEX[20]). Furthermore, support for arbitrary networked data and for existing open standards for object types is provided. As one of the main design goals, SyncML addresses the resource limitations of mobile devices.

The specification is based on three major parts:

- *Representation Protocol*, a common language for all parties,
- *Synchronization Protocol*, a common way of exchanging data for all parties,
- *Transport Bindings*, for implementing the data delivery.

The Framework

SyncML not only defines a format, but also a conceptual data synchronization framework and data synchronization protocol. As shown by the dotted line in figure 3.7, the framework scope consists of the SyncML representation protocol, as well as a conceptual SyncML Adapter and SyncML Interface. This SyncML Framework is useful for describing the particular system model associated with SyncML implementations. A common data synchronization protocol can be defined using the SyncML representation protocol. However, such a user defined protocol can be regarded rather as an application, and can thus be treated as outside the framework [216].

In figure 3.7, a networked service *A* provides data synchronization with other applications, in this case application *B*, on some networked device. Both service *A* and device *B* are connected over some common network transport, which could be HTTP [77]. Application *A* utilizes a data synchronization protocol, implemented as the *Sync Engine* process. The data synchronization protocol is manifested on the network by client applications accessing the *Sync Server* network resource. The Sync Server Agent manages the Sync Engine access to the network and communicates the data synchronization operations to/from the client application. These capabilities are provided through invocations of functions in the *SyncML I/F*, which is the Application Programming Interface (API) to the *SyncML Adapter*. The SyncML Adapter is the conceptual process that the originator and recipient of SyncML formatted objects utilize to communicate with each other. Futhermore, the SyncML Adapter is the framework entity that interfaces with the network transport, which is responsible for creating and maintaining a network connection between *A* and *B*. Application *B* utilizes a *Sync Client Agent* to access the network and it's SyncML Adapter through invocatons of functions in the SyncML I/F [216].

Representation Protocol

The SyncML representation protocol is the format specification for the SyncML framework. It is defined by a set of well-defined messages, which are used between the entities participating in a data synchronization operation. The messages are well-formed, but not necessarily valid, XML documents (the industry standard for text document markup [239], see section 3.2.5). One document consists of at least the XML body (which makes it *well-formed*), and optionally of the XML header (which

[20] Object Exchange Protocol (OBEX) [132] (*see* page 60)

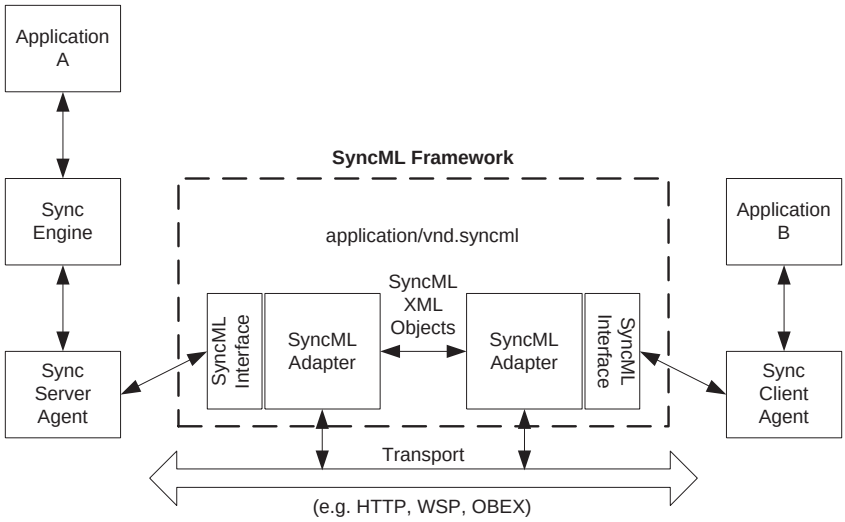

Figure 3.7: SyncML framework [216]

makes it *well-formed* and *valid*). The header specifies routing and versioning information, whereas the body contains one or more SyncML commands (*see* figure 3.8). The commands are defined by a set of *request* commands and a set of *response* commands. The representation not only provides space for a set of commands, but it also identifies a small set of common data formats. Hence, a SyncML document can also be interpreted as a Multipurpose Internet Mail Extension (MIME)[21] content type, for which a new MIME media type has been registered for: `application/vnd.syncml`. Thus, the representation is independent from the type of backend data store and decoupled from any synchronization object types. Furthermore, the design contains the concept of packaging. The SyncML package performs a set of data synchronization operations. This conceptual data sychronization *package* permits either a *batch* of multiple data synchronization operations put together in a single SyncML message or conveyed as separate SyncML messages, each containing a single data synchronization operation. SyncML messages are the body of the MIME entities [216].

The SyncML representation protocol supports the following basic types of data synchronization models:

Request/response command structure. The synchronization process is triggered by a request, either performed manually (user) or automatically (timer).

Blind push command structure. The sychronization process is performed without agreement with the user, mostly initiated by the server party.

A set of predefined data object content types is provided by the SyncML representation protocol. Defined according to the Augmented Backus-Naur Form (ABNF), standard objects recognized are *Arbitrary Database Object Filter*, *Contacts Media Object*

[21] *MIME* [109] is the Internet standard for multipurpose message contents. It provides a useful mechanism for differentiating between different content and document types.

Filter, *Calendar Media Object Filter*, and *Email media Object Filter*. The Calendar Media Object Filter is based on the `iCalendar` definition [107] whereas the Contacts Media Object Filter is based on the definition given by `vCard` (*see* section 3.3.3).

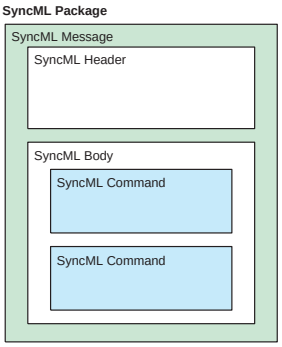

Figure 3.8: SyncML package DTD [214]

Document Type Definitions Three different *Document Type Definitions (DTDs)*[22] are defined by the representation protocol. Each DTD is identifiable as an XML name space and can be encoded in a tokenized, binary format defined by *WAP Binary XML (WBXML)*[23]. WBXML is already supported by WAP, thus allows immediate applications with the upcoming SyncML enabled WAP phones, before the end of this year, but other encoding schemes will be used on other platforms.

SyncML DTD. A small set of common object types and formats is included in the representation protocol: *contacts, calendar, to-do, journal*, and *e-mail* [213].

Meta Information DTD. Used to represent meta-information within the representation protocol; elements are: *format, type, version*, etc.

Device Information DTD. Used to represent device-information within the representation protocol; elements are: *client*, *server*, etc.

Data identifier mapping Suppose two sets of data, we call them data collection, *A* and *B* are to be synchronized with each other. SyncML does not require these two data collections to be homogenous (i.e. of the same structure). More concretely, both data identifiers and data formats can be different in both data collections. In order to use SyncML between *A* and *B*, the synchronizing applications would have to provide a mapping between data identifiers in both data collections. For instance, consider a *Globally Unique Identifier (GUID)*, pointing with 16 bytes to a document on the data sychronization server. The *Local Unique Identifier (LUID)*, identifying the corresponding version of the document on a mobile device, could be only two bytes long. Hence, synchronizing data between the mobile device and the data synchronization server, required a mapping from the smaller identifier of the mobile device

[22] The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements. A DTD can be declared inline in your XML document, or as an external reference. DTD is defined in XML 1.0 Specification [238].

[23] *WBXML* is a XML derivate for packaging binary data over WAP [237].

to the longer itentifier on the data synchronization server, and vice versa. SyncML provides the necessary mechanisms for this kind of identifier mapping.

Definition. In SyncML, a data collection, $A = \{a_1, a_2, ..., a_n\}$ with elements[24] $a_i \in \mathbf{D}$, and identifiers $1 \leq i \leq n$, has been *successfully synchronized* with a data collection, $B = \{b_1, b_2, ..., b_n\}$ with elements $b_j \in \mathbf{D}$, and identifiers $1 \leq j \leq n$, we use the notation $(A \cong B)$, $\Longleftrightarrow$:

There exists a bijective mapping $\varphi : A \rightarrow B, a_i \mapsto b_j$, such that $\forall i$:

$$\exists a_i \quad \Rightarrow \quad \exists b_{\varphi(i)} \wedge (b_{\varphi(i)} \equiv a_i) \tag{3.11}$$

$$\nexists a_i \quad \Rightarrow \quad \nexists b_{\varphi(i)} \tag{3.12}$$

$$(A \cong B) \quad \not\Rightarrow \quad (B \cong A) \tag{3.13}$$

In words, if a_i exists, then $b_{\varphi(i)}$ exists and $b_{\varphi(i)}$ is equivalent to a_i. If a_i does not exist, then $b_{\varphi(i)}$ does not exist. If A synchronized with B is true, this does not imply that B synchronized with A is also true. In other words, data synchronization using SyncML does not have to be reflexive. Furthermore, φ can obviously include a permutation on the tuple $A = (a_1, a_2, ..., a_n)$.

Data Operations Unlike ordinary synchronization principles, SyncML provides additional operations which can be practical, especially in mobile environments. The most inportant among these is the capability for refreshing the entire data stored on the client with the equivalent data on the server. This feature is necessary for cases when for instance the SyncML client and the server are no longer *in sync* due to power or hardware failure, if data for some reason gets corrupted or completely erased from memory, or if the user gets a new device and wants to have the same data on it.

Another important feature is based on distinguishing between *soft* and *hard* data deletion. Whereas a hard delete command completely deletes the specified data from the recipient's data store, a soft delete command just locally removes the specific data. Hard data deletion should additionally remove any association with the originator's synchronization data, soft delete does not do that. The idea behind a soft delete is based on the potentially limited storage resources in a mobile device. The data can be deleted to free-up storage for other, higher priority data. In other words, conceptually, soft deleted data remains in the set of synchronized data [216].

Security SyncML itself does not define any security mechanisms, but the framework was designed to allow integration of authentication, authorization, and inclusion of encrypted data into the existing package structure. Obviously, the underlying transport layer may be used for establishing a secure tunnel for exchanging SyncML packages. Additionally, the representation protocol provides at least two element type specifications, `Cred` and `Chal` for representing authentication information and schemes. Two coding schemes are supported, *Base64 Content-Transfer Encoding* [79] and the *MD5 Digest* scheme [193]. Base64 Content-Transfer Encoding is susceptible to the threat of network eavesdropping, however it is simple to implement. This

[24] This definition is made on a generic data space $\mathbf{D}$, which could be interpreted as the set of all bitstreams $\mathbf{D} = \{0,1\}^*$, or a more high-level set of data (e.g. strings, numbers, etc.). In each case, the equivalence $a \equiv b$ has to be defined carefully.

means that it should be used in combination with other security mechanisms in underlying layers. On the contrary, Keyed MD5 [146] allows authentication mechanisms, based on strings that are generated for each session [216].

Synchronization Protocol

The motivation behind the synchronization protocol is the insight that a representation protocol only is not enough for achieving interoperability. The main responsibility of the synchronization protocol is to synchronize multiple data types with multiple devices.

The major design principles of the synchronization protocol were [214]:

High and low latency networks. As the SyncML protocol should be able to run over basically any network, the communication flow should be designed to handle high or low network latency.

Sophisticated and non-sophisticated devices. A main goal of the SyncML initiative was the interoperability of all kinds of devices, the protocol should be able to handle simple cellular phones as well as advanced personal computers or servers.

Various persistent storage models. Synchronized data, especially mobile data can be regarded as volatile, therefore a reliable storage model has to be provided.

Different security requirements demanded. The protocol must be able to provide different security mechanisms for different devices and needs.

Various usage models. Data synchronization can be done in several different ways, the protocol must support these ways.

The protocol defines the roles of the client and the server within the framework, which is fundamental throughout the protocol specification. The *SyncML Client* is the device which contains a *sync client agent* and that in general sends its modifications to the server first. There are some special cases, where the server has the role of initiating synchronization. The client must also be able to receive responses from the SyncML server. Typically, the SyncML client is a mobile phone, a PC, or a PDA device. The *SyncML Server* is the device which contains a *sync server agent* and a *sync engine*. The role of the server in general is, to wait for the SyncML client to initiate synchronization by sending its modifications to the server. The server is responsible for performing the analysis and updating the persistent copy. In addition, the server may be able to initiate synchronization if unsolicited commands from the server to the client are supported by the transport protocol level. The SyncML server typically, runs on a server device or a PC.

Protocol Fundamentals One basic requirement of the protocol is that all devices involved keep track of their operations. Just as in databases, the SyncML devices are responsible for maintaning a *Change Log* containing information about the modifications associated with data items. Standard modifications include update, addition, and deletion. The protocol does not specify the format of the device specific change log, but requires any device to be able to specify which data items have changed when synchronization is initiated. If one device synchronizes with multiple devices,

the change log must be able to indicate all modifications in relation to a previous synchronization with each device.

Another fundamental requirement of the SyncML protocol is the usage of *Sync Anchors*, a special type of timestamp associated with database transactions. At every initialization of a new synchronization, there are two sync anchors to be exchanged and mutually compared, the `lastsync` and `nextsync` anchor, containing the timestamp of the previous and of the current synchronization, respectively. Based on the comparison of the `lastsync` anchors, the SyncML server can determine whether the database has changed internally since the last synchronization event, and if needed, initiate a complete synchronization (*see* page 59).

Because of the *different data identifiers* for client and server (*see* page 56), the protocol must be able to handle the mapping between these. Whereas the server database contains both a table containing the mapping between GUID and the data elements, and a table mapping between GUID and LUID, the client database only contains a table mapping between LUID and the data elements. The protocol provides mechanisms for speeding up this mapping by caching map operations, which may occur if the server has explicitly indicated that it does not require a response to its synchronization message. Cached items will then be transmitted at the beginning of a subsequent synchronization session.

As modifications to the same data items can occur on both client and server database, the protocol has to provide a mechanism for *Conflict Resolution*. In such a case, the SyncML client is notified about the conflicts, and some common policies for resolving can be used, as defined in the synchronization protocol.

Synchronization Types The protocol specification defines seven different synchronization types [214, 215], of which some are special cases of the previously defined ones:

Two-way sync. This is the normal case, in which the client and the server exchange information about modified data in these devices. The client sends its modifications first.

Slow sync. This is a special case of *two-way sync*, in which all items are compared with each other on a field-by-field basis. This means that the client sends all its data from its database to the server and the server does the synchronization analysis on a field-by-field basis. This provides a complete synchronization of the client with the server.

One-way sync from client. This is a synchronization type in which the client sends its modifications to the server, but the server does not send its modifications back to the client, i.e. only client modifications are exchanged. A database export is a special case of this synchronization type.

Refresh sync from client. This is a synchronization type in which the client exports all its data from a database to the server. The server is then expected to replace all data in the target database with the data sent by the client.

One-way sync from server. This is a synchronization type in which the client gets all modifications from the server, but the client does not send its modifications to the server. Thus, only server modifications are updated. A database import is as a special case of this synchronization type.

Refresh sync from server. This is a synchronization type in which the server sends all its data from a database to the client. The client is then expected to replace all data in the target database with the data sent by the server.

Server Alerted. This is a synchronization type in which the server alerts the client to perform synchronization by sending a notification from server to client. Thus, the server informs the client to start a specific type of synchronization with the server.

An Example The synchronization of a calendar entry can be exemplified in the message sequence chart (figure 3.9) and the corresponding SyncML contents of the first message are depicted in the following source code (figure 3.10). In this example, the authentication credentials are sent by the client, and the client has two items to be sent. These items are sent in separate messages, whereas the server has only one item to be sent. The data identifier mapping is done immediately.

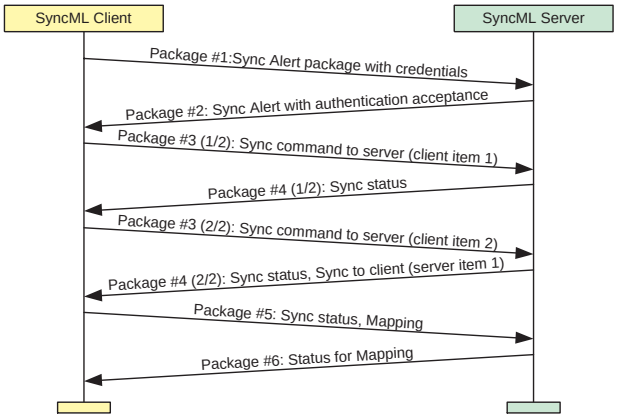

Figure 3.9: Example message flow [214]

Transport Bindings

By definition, the SyncML protocol and representation protocol are transport independent. However, three default transport bindings are defined, in order to support the most common transport media, and to exemplify implementation. Transport bindings specify how messages and responses are exchanged *over a specific medium*. The default bindings are summarized in the illustration below (figure 3.11), an example of the transport independent datagram multiplexing in the SyncML framework is depicted in the Protocol Data Unit (PDU) illustration (figure 3.12). Future bindings could be for instance email or message queries [214]. The default bindings are:

HTTP, the hypertext transfer protocol, for use over standard internet or intranet networks.

WSP, the wireless session protocol, for use over the Wireless Application Protocol (WAP).

```
<SyncML>
   <SyncHdr>
      <VerDTD>1.0</VerDTD><VerProto>SyncML/1.0</VerProto>
      <SessionID>1</SessionID><MsgID>1</MsgID>
      <Target><LocURI>http://www.syncml.org/sync</LocURI></Target>
      <Source><LocURI>IMEI:493005/10/059280/0</LocURI></Source>
      <Cred>
         <Meta><Type xmlns='syncml:metinf'>syncml:auth-basic</Type></Meta>
         <Data>dXNlcmlkOnBhc3N3b3Jk=</Data>
      </Cred>
   </SyncHdr>
   <SyncBody>
      <Alert> <!-- This is Sync Alert command for calendar. --></Alert>
      <Put> <!-- This is for sending client device info. --> </Put>
      <Get> <!-- This is for requesting server device info. --> </Get>
      <Final/>
   </SyncBody>
</SyncML>
```

Figure 3.10: XML code for first message in example [214]

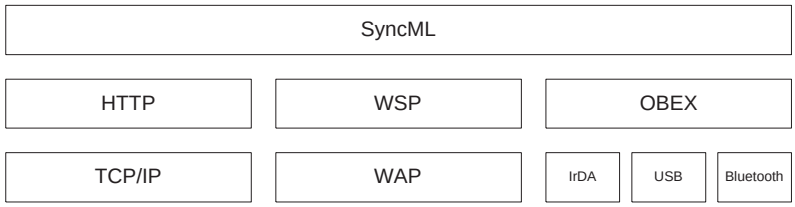

Figure 3.11: Default transport bindings [214]

OBEX, the object-exchange protocol [132], for use over e.g. IrDa (infrared link) or Bluetooth.

Summary

Problems Based on the change log requirement (*see* page 58), implementing the synchronization of e.g. email via SyncML between existing platforms (e.g. Microsoft Outlook, Lotus Notes) and mobile terminals may be difficult. The necessity of transaction logging, which has to exist for every server or client involved, is difficult to implement, since changes in the mail server have to be made, and source code for such commercial products is not readily available. On the other hand, simple mail protocols such as IMAP, support IDs for single messages, which can be used as SyncML object IDs. In similar systems, e.g. the *Outbox* or *Sent Items* folder can be interpreted as a certain kind of transaction log. However, it is somewhat difficult to integrate into existing systems and platforms who do not provide SyncML support.

The XML-based data representation can be viewed as more verbose than alternative binary representations. This is the reason why XML is often cited as unsuitable for low bandwidth network protocols, especially wireless systems. In most cases, SyncML uses shortened element and attribute names and only causes a constant *overhead* which should be neglectable in balance with larger data contents. Furthermore,on page 56 we described a way of binary XML encoding, which happens in

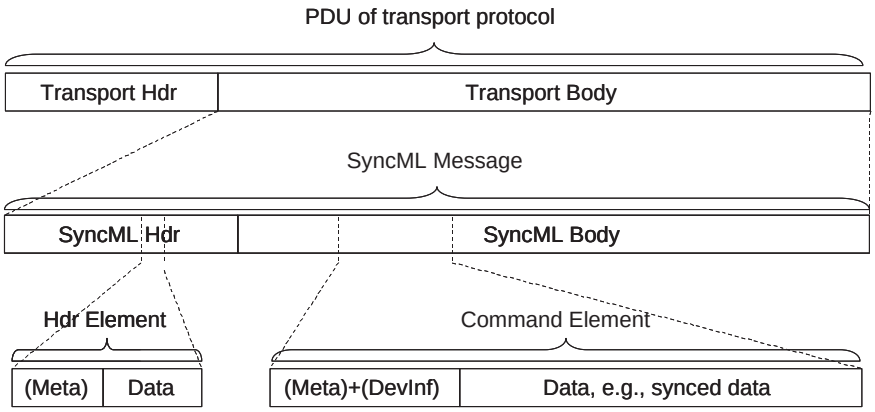

Figure 3.12: Transport bindings define the usage of the PDU [214]

a transparent manner to any SyncML application. However, additional link layer compression is still needed.

The SyncML Framework is still quite a new definition and very few publications or discussion material about SyncML are available. It remains to be proven by the first concrete applications to be in use.

Advantages The SyncML framework presents a robust but generic specification for mobile data synchronization. Sponsored by major companies from the telecom and handheld devices industry, it will have a solid supporting community. The derivation from the XML language makes it easy and effective to integrate into almost any framework.

3.3.5 Operator-Hosted Provision

For providing synchronization in the concept of operator-hosted services, a scalable solution is needed. Such an approach could be based on open standards, such as SyncML, which will be supported by a variety of devices released onto the market during the coming years. Especially for small and medium sized enterprises (SMEs), the access to an operator-hosted SyncML service could be interesting, since the setup of an own corporate SyncML server would cause high costs and installation effort when tailoring new services in combination with existing information servers (ISs).

One could think of integrating SyncML support into the existing operator-hosted mobile application portal (MAP), by letting the portal *speak* SyncML to the client terminals. Being an XML based markup language as well, generating SyncML output could be based on similar technologies like existing HTML and WML support. Instead of interacting with an own corporate SyncML server, all terminal devices supporting SyncML communicate with one operator-hosted MAP, which handles application-level multiplexing and demultiplexing of synchronization commands to and from the corresponding back-end corporate server (*see* figure 3.13). When adding SyncML support to the MAP, one could think of two successive integration phases:

SyncML translation. The MAP is being upgraded with a SyncML translating engine, implementing the SyncML data communication on the front-end side. The MAP itself as well as back-end middleware does not have to be changed, since the MAP communicates over proprietary protocols with the ISs.

End-to-end SyncML. Based on the observation that the existing MAP implementation is based on the XML language as communication standard, one could think of modifying all parts of the middleware to communicate in SyncML on an end-to-end basis. This would mean that a SyncML message, generated by the back-end middleware (connecting to the IS), could be forwarded without modification to the mobile handset.

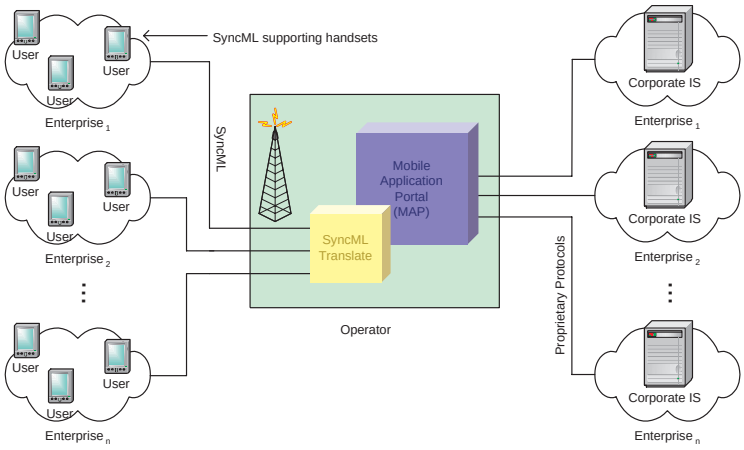

Figure 3.13: Concept for operator-hosted SyncML provision

This approach of designing SyncML support as a subservice of an existing business application portal, would enable a relatively easy and fast provision of SyncML in combination with existing services. Current terminal devices already contain local applications (such as calendar or e-mail clients), and the use of these local applications would be more preferred than markup language based solutions, *if* the corporate information repositories could be connected.

There are still some problems to consider when designing such an approach. One problem making synchronization services connecting to existing IS systems more difficult than markup language based solutions, is the need for back-end modifications. Obviously, for implementing any data synchronization support, the back-end IS has to generate transaction log information which the synchronization service can access and base its operations on (e.g. the *change log* information in section 3.3.4). This might be somewhat tricky to implement in closed IS systems (e.g. MS Outlook).

Another challenge would be to redefine the security and privacy concept for operator-hosted services; even though the firewall would be openend to a small extent in this kind of solution, a remarkable amount of personal information from multiple users from multiple companies will be tunneled through one single operator MAP. One idea to cope with this risk could be based on applying hashing mechanisms onto the front-end and back-end software. Instead of communicating

the data items in plain text, SyncML allows the encapsulation of binary data (*see* section 3.3.4), which could be the hashed data item.

3.3.6 Recommendation

The vision of having the mobile terminal automatically *in sync* with corporate infromation systems cannot be solved by one synchronization protocol alone (such as SyncML) [130]. The approach of synchronizing data between an information server and a simple, limited mobile device shows logical holes; limited storage space causes data entries to be simplified and truncated, which makes the support of bidirectional consistency difficult (*see* section 3.3.2).

On the other hand, the approach of synchronizing data between an information server and a more sophisticated terminal device (e.g. a PDA with remarkably higher storage and display facilities) does not seem too feasible as long as the user has to initiate and administrate the synchronization himself.[25] In these kinds of sophisticated terminal devices, one will be able to store files and directories, which arises the need for automated synchronization (*see* section 3.5). In future scenarios, where unlimited packet-based connectivity is guaranteed, this kind of synchronization might not be needed at all.

From the operator perspective, the provision of synchronization services could be integrated into existing portal systems (*see* section 3.3.5), allowing fast scaling of simple synchronization services based on common standards (such as SyncML). Automated synchronization of e.g. entire file systems would be rather complex to implement in a centralized system, and would arise severe questions in privacy issues. The operator could in both cases transfer the entire coprorate information system contents on the application level.

3.4 Virtual Private Networking

3.4.1 Introduction

Technologies enabling *Virtual Private Networking (VPN)* are the foundation for any fully IP-based mobile computing solutions for business use. Corporate back-end data nowadays mostly resides in private Intranets with strong firewalls. For making any IP-based Intranet application fully mobile, the Intranet information servers involved have to be made visible for mobile use. The use of corporate portals for accessing an Intranet application e.g. over a web interface requires this specific solution to be tailormade into a mobile business application. The client terminal cannot just be unplugged from the corporate LAN and reconnected through an ISP anywhere the world. For full indepencence of tailoring such new business application portals, VPN solutions are needed. Since most common LAN applications mostly are based on IP instead of other proprietary protocols, VPN is an interesting and cost effective approach for making the corporate Intranet mobile.

[25] The user decides himself when to initiate synchronization and is thus to a certain extent responsible for keeping track of his transactions and the data consistency.

3.4.2 Architecture

VPN is the general term for packet-based secure connectivity, using an Internet tunnel instead of a dedicated broadband link. VPN services can be divided into three categories [226]:

LAN Interconnect VPN services. This type of VPN service helps to interconnect local area networks located at multiple geographic areas over the shared network infrastructure. In most cases, this method is used to bridge multiple geographic locations of a multinational company into one logical corporate network. This provides a cost-effective replacement for the expensive dedicated links.

Dial-up VPN services. This type of VPN services provides access to the company's Intranet from any remote location throughout the Internet. The remote user connects to the nearest Remote Access Server (RAS),[26] in most cases through a *classic* circuit switched link). This is typically a local Point-of-Presence (PoP) of an Internet Service Provider (ISP) or the shared network infrastructure.

Extranet VPN services. Combining the architecture of LAN Interconnect VPN services and dial-in VPN services results in this type of VPN services. The idea of this type of service is to enable a specific community, e.g. external vendors, suppliers and customers, to access specific areas of the company's Intranet. The corporate Intranet contains a separate, specific area denoted as the *Demilitarized Zone (DMZ)*, which is open to the specific community.

Obviously, the second type of VPN services, the dial-up VPN services is the most relevant for our scope of mobile computing. One of these models is the *Layer 2 Tunneling Protocol (L2TP)*, in which the RAS automatically establishes a secure connection to a predefined location inside the company's Intranet, usually through a firewall enhanced with VPN capabilities. Followed by a successful authentication of the user, a *static* VPN connection provides transparent Intranet connectivity. The L2TP model is usually aimed at home-offices and mobile workforce who connect to a specific RAS.

Another model is given by the *Point-to-Point Tunneling Protocol (PPTP)*. This model is suitable for mobile users who may dial-in to any local ISP. After connectivity has been established to the ISP, the user himself initiates a connection to a VPN server located inside the remote company's Intranet. Like in the L2TP model, upon login success, the user has full Intranet connectivity. However, in contrast to the L2TP model, the RAS does not participate in the establishment of the VPN connection, and thus, no specific configuration of the RAS is needed for the PPTP model.

We will use the term *VPN* when referring to the dial-up type of VPN services in the remaining sections of this thesis.

Since the idea of the VPN approach is to use the *free* Internet for establishing the link instead of using dedicated physical links, a VPN implementation is based on transparent traffic encryption and decryption. This mechanism can be implemented on two logical layers [226]:

Network Layer VPN. On this layer, the VPN mechanism performs tunneling of each network layer datagram. A typical cryptography mechanism is given by *IPSec*,

[26] *Remote Access Server (RAS)* is the technical term for a dial-in modem pool.

resulting in an IP-based tunnel, but other network layer protocols can be used as well (e.g. AppleTalk or Novell's IPX).

Link Layer VPN. The VPN cryptography mechanism can be located on one logically lower level, if the common network is based on a switched link layer technology, such as Frame Relay (FR) or Asynchronous Transfer Mode (ATM).

However, the network layer mechanism is the most common VPN implementation, which often is being associated to IPSec over IP. Taking the IPSec solution as an example, such network layer VPNs support tow different technologies, both of which at least one implementation is needed for a complete VPN [242]:

Client implementation. A client implementation is inserted between the IP stack and the logical network drivers. This type of implementation is appropriate for use with common OSs (e.g. Windows 95, 98, and NT, Macintosh, Unix), since modification on the IP stack source code is not required. Windows 2000 has Microsoft's implementation of IPSec already built in. Since a client implementation is based on adding one layer into the protocol stack, it is being referred to as *bump in the stack (BITS)*.

Gateway implementation. A gateway implementation may be realized through either hardware or software. An example for a software implementation would be IPSec running on a firewall or a router; a hardware implementation would be IPSec running *in silicon*, i.e. in a black box. A hardware implementation is obviously much faster, since no overhead in the OS is caused. A software implementation is much less expensie to implement and deploy, and could even be implemented in high level programming languages, such as Java [133].

3.4.3 Recommendation

With the VPN concept, a key question on the meaning of business application mobility is addressed—the question whether future mobility will be based on portal-like application level solutions for remote access, or on ubiquitous and transparent secure all-IP access from laptop or smartphone clients [57].

Until now, VPN has not been interesting for wireless access, since datagram based encryption would have had unacceptably large impacts on the slow air interface (e.g. GSM). In combination with emerging wireless technologies, such as e.g. GPRS or WLAN, the VPN issue is a hot topic, which considering the increasing capabilities of wireless terminals, offers a set of new business opportunities. Alice Systems has recently released the first remote access client that combines VPN functionality and GPRS terminal connectivity. The product, *Alice Login*, is specifically designed for accessing the corporate Intranet and is compatible to most common used VPN gateways by supporting the IPSec security standard [38].

Internet-based QoS VPNs have become a feasible and economically interesting solution for deploying wide area corporate networks. However, the network management complexity of such frameworks has increased significantly. In order to solve the problem of automated and integrated network management, recent studies propose solutions allowing customers to administer own VPN tunnels and control their QoS parameters individually [55]. This issue of selling QoS services offers opportunities for operators.

On the other hand, with VPN based mobility solutions, the role of the operator will migrate to the sole responsibility of transmitting packets with satisfactory QoS,

which makes it difficult for any operator to participate in added value creation (*see* survey results in appendix D).

However, VPN technologies can be regarded as a prerequisite for applications and solutions presented in the remaining sections of this chapter.

3.5 Mobile Computing File Systems

3.5.1 Introduction

Whereas mobile data synchronization (*see* section 3.3) is mostly about wireless data transport issues and conflict resolution between client and server, one could say that this entire synchronization issue already is contained as an integral part of file systems with *disconnected operation*. Unlike usage scenarios envisioned by user-triggered over-the-air synchronization, the idea of such a file system is to provide data consistence between client and server—transparently to mobile applications and users [147]. As observed previously (*see* section 3.3.2), the issue of providing client cache data consistency autonomously in the background is not a trivial task. There are currently many on-going research projects, aiming to define such a middleware file system, without destroying the user experience.[27] The nature of such a file system for disconnected operation, allowing any collaborative data usage as a primary research goal, makes this issue highly relevant for mobile business applications.

Within this context, the existing research on mobile client-server computing can be categorized into three paradigms [114]:

Mobility Aware Adaptation. When accessing remote data, mobile clients can face rapid changes in network conditions and resource availability. A dynamic reaction is required by the mobile client-server system, in order to adjust the functionality of computation between mobile and stationary hosts. This allows the mobile application to continuously operate in such a dynamic environment. A general rule is that the computation of clients and servers has to be adaptive in response to the changes in mobile environments [122]. The range of mobility adaptation strategies is defined between *laissez-faire*, i.e. complete responsibility in the application, and *application-transparent* adaptation, i.e. complete responsibility in the system. Between these two extremes lies the set of *application-aware* adaptation techniques, i.e. the approach of collaborative adaptation between system *and* application [196, 197].

Extended Client-Server Model. The effect of mobility on the client-server computing model is another way to characterize computing in mobile environments. In a client-server information system, a server is any machine that holds a complete copy of one or more databases, and a client is able to access data residing on any server with which it can communicate. Such classic systems assume the location of client and server hosts as well as their interconnection to remain unchanged, resulting in a static partition of functionality. However, in a mobile environment, the disctinction between client and server may have to be temporariliy blurred [196]. The result is the *extended client-server model*, allowing a

[27] The user experience is destroyed, when the user notices that the current mobile data accessed is outdated, and when he has to initiate the synchronization manually. In order to keep user experience, there should thus not be assumed any failure tolerance.

client to emulate server functionality, and vice versa. The two extremes within this range are the *full client* and the *thin client* applications (*see* section 3.7.7). Between these two extremes, the *flexible client-server architecture* allows the dynamic relocation of application logic beteen client and server. One common example for this kind of dynamic relocation is contained in the concept of mobile objects (agents), representing application logic, that can travel through the network.

Mobile Data Access. The delivery of server data and the maintenance of client data consistency in a mobile environment is the main research goal for mobile data access. The efficiency of client data consistency approaches is the challenge, because of weak connectivity and resource constraints (such as local memory space) of mobile devices. Important issues in mobile data access are *delivery modes*, *data organization*, and *consistency requirements*.

As this section is focused on the provision of mobile computing file systems, obviously the last paradigm previously described is the most relevant. However, all the three paradigms represent an interesting reference model for mobile computing, which define essential design issues for wireless OSs. The following sections will briefly examine a set of mobile data access issues relevant for mobile computing file systems (*see* sections 3.5.2 to 3.5.7). There exists a set of on-going research projects and frameworks, aiming to define and implement candidates for such file systems. Some selected approaches which will be introduced in the latter part of this chapter (*see* sections 3.5.8 to 3.5.12).

3.5.2 Broadcast Disks

An obvious method for providing mobile data consistency is given by continuos transmission of data updates by the server. Taking this method one step further, when a server continously and repeatedly broadcasts the entire set of data to a client community, the broadcast channel becomes a *disk* [114]. Clients can listen to the broadcast disk channel and retrieve data as it goes by, and the broadcast disk can be organized as multiple databases of different sizes and speeds on the transmission medium [37].

The broadcast disk contains a multiplex of data from different disks onto the same broadcast channel. The multiplex can be weighted in order to separate between *fast disks* and *slow disks*. The fast disk data is repeated more often than the data of the slow disk. Using this technique efficiently, an opportunity is given to more closely match the broadcast to the statistic workload at the client. Assuming that the broadcasting server has an indication of the clients' access patterns, then *hot pages*, i.e. data that is more likely to be of interest to a larger part of the client community, can be transmitted faster, while *cold pages* can be transmitted with lower priority. The priority definition can be based on an arbitrariliy fine-grained hierarchy, allowing exact broadcast positioning between fast and slow disks [114].

Although the broadcast disk technologies currently do not seem that relevant for IP-based community connectivity (such as VPNs), it provides an effective framework for assymmetric communication, and might become interesting for e.g. satellite or HAPS enabled mobile communication. In additional combination with reliable cryptography methods, the broadcast disk technologies provide an improved performance for non-uniformly accessed wireless community data.

3.5.3 Automated Prefetching

The storage of nonlocal files into the client cache prior to disconnection is referred to as automated *prefetching* (or *hoarding*). Automated prefetching provides a useful solution to support disconnected operations, but the challenge lies in the determination of which files to select and cache locally. Possible approaches include the selection of the most recently referenced files, or an interactive prompt asking the user to participate at least peripherally in managing prefetching contents. The first solution might be wasteful of scarce cache memory space, while the latter requires more expertise and involvement that most users are willing to offer [114]. Another fully automated mechanism, *predicive prefetching* is based on the ability of the system to:

- observe user behavior,
- interprete semantic relationships between files,
- apply interpretation to aiding the user.

An example for such an automated predictive prefetching approach is given by the *SEER* system [126].

3.5.4 Traditional Client-Server Approaches

One idea for defining file system consistency models for mobile computing, is to examine existing models from traditional client-server computing, and to enrich them with mobility support. Traditional consistency methods in *non-mobile* client-server computing can be divided into two categories [114]:

Detection Approach. The clients send queries (pull messages) to the server in order to validate cached data.

Callback Approach. The server keeps track of the clients' locally cached data objects, and pushes invalidation messages directly to the specific client that has cached the data item to be updated.

None of these traditional approaches is really suitable for mobile environments with weak connectivity of clients. Frequently disconnected clients make the detection approach very ineffective to use, whereas the callback approach may cause high costs due to e.g. network latency and intermittent failures. If many data items are outdated after a long time of disconnection, the callback approach will cause a long queue of invalidation messages to be sent. The following two sections will introduce one example for deriving both approaches into solutions with mobility support.

3.5.5 Cache Coherence Granularity Variation

For describing the consistency of cached data, the definition of a more expressive measure than e.g. *valid* versus *outdated* is needed. Coherence granularity denotes the density of the spectrum between being valid and outdated.

The *Coda* file system (*see* section 3.5.8) provides an approach for coherence granularity based on the detection approach. In this system, clients can track the server state at multiple levels of granularity. A server maintains version stamps for its data

objects, as well as for its file volumes, i.e. collections of data objects. When an object is updated, the server increments the version stamp of the object and that of its superset volume. As blind prediction of disconnection, the Coda clients perform continuous caching of the volume version stamps. When the connection is restored after a network failure, the client sends queries containing its volume stamps for validation. If a volume stamp turns to be still valid, then each object cached from that volume is valid as well. If a volume stamp is not valid, each cached object contained in that volume has to be validated individually.

The method of cache coherence granularity variation thus increases the performance compared with the detection approach, as not necessarily every object has to be validated individually. In its worst-case, this method is as good as the detection approach.[28] Experiments and measurements from the Coda system confirm that the varied granularity of this approach dramatically improves the speed of cache validation [114, 156]. Although one could say that this solution is more or less derived from detection approach, it introduces a smart versioning mechanism.

3.5.6 Cache Invalidation

The cache invalidation method is based on callback approach, and can be implemented by e.g. utilizing wireless broadcast channels [114]. A server periodically broadcasts invalidation messages reporting the set of data items which have been updated. Instead of sending direct queries to the server, the clients listen to these invalidation messages over dedicated broadcast channels. The use of dedicated broadcast channels for invalidating cache items is in that sense very attractive for mobility, since the server does not need to know the location (e.g. current IP address) of its clients. Examples for existing broadcast channel cache invalidation algorithms are given by *Broadcasting Timestamps (TS)*, *Amnesic Terminals (AT)*, and *Signatures (SIG)* [43].

3.5.7 Distributed Shared Memory

In *distributed shared memory (DSM)* systems, processes share data transparently across physical boundaries. The data messaging, data faulting, location and movement is handled by the underlying system. While a DSM system implies performance tradeoffs, especially in loosely coupled distributed systems, there are several advantages of the shared programming model [161]:

- Shared memory application source code is usually shorter and easier to write and understand than equivalent message passing programs.
- Large or complex data structures may easily be communicated.
- Shared memory gives transparent process-to-process communication.
- Programming with shared memory is a well-understood problem.

As the physical boundaries, and especially the network latency still is the major performance factor in distributed systems, all DSM systems provide some sort of caching implementation for improving data access conditions in the disconneted

[28] The worst-case in the cache coherence granularity approach occurs, if *every* volume is outdated. In this case, every object has to be validated indivudually for all volumes.

state. Each system must decide whether or not to attempt to keep the data coherent, and, if so, what coherence strategy to use. DSM coherence strategies can be divided into three categories:

Strict Consistency. In a strict consistency strategy, a read command always returns the value written by the *most recent* write command. Obviously, for every read command, the remote repository has to be queried. Two main implementation approaches are broadcast disks (*see* section 3.5.2) and cache invalidation (*see* section 3.5.6).

Relaxed Consistency. In a relaxed, or *loosely* consistency strategy, the system enforces some form of weak consistency guarantees and the application (or compiler or user) can indicate synchronization points where consistency must be enforced.

Manual Synchronization. This strategy does not contain any automatic consistency mechanism, but provides the user with the facilities necessary to implement user level synchronization and consistency (*see* section 3.3).

There exists a large variety of academic and commercial DSM research projects, from of which relaxed consistency seems to be the most challening research topic. Permitting relaxed consistency, i.e. temporary inconsistencies, is a common method of increasing performance [243], and could be of relevant interest for mobile computing file systems. Memory is said to be loosely coherent if the value returned by a read operation is the value written by an update operation to the same object that *could* have immediately preceded the read operation in some legal schedule of the threads in execution [46].

3.5.8 Coda

Developed in a research project at the Carnegie Mellon University (CMU), the *Coda* file system [4, 196, 156] provides an example for mobility aware adaptation, namely application-transparent adaptation (*see* section 3.5.1). Coda is one of the pioneers providing a *file system proxy* as a core component of the file system, allowing existing applications to work without modifications [124, 114]. The Coda file system proxy supports functionality for three basic scenarios:

- Disconnected Operations,
- Weakly Connected Operations,
- Isolation-only Transactions.

Disconnected Operations

In the case of anticipated disconnected operations, a small collection of trusted Coda servers exports a location-transparent UNIX file name space to a larger collection of untrusted clients. On each client, a user-level process, *Venus*, manages a file cache on the local disk. Venus acts as a file system proxy and bears the brunt of disconnected operations. Venus operates in one of thre states:

Hoarding state. In the initial state, the server files are prefetched onto the mobile computer for local access.

Emulation state. Upon disconnection, Venus begins logging updates in a client modify log. In this state, Venus performs log optimizations to improve performance and reduce resource usage.

Reintegrating state. Upon reconnection, Venus enters the reintegrating state, where it synchronizes its cache with the servers, propagates updates from the client modify log, and returns to the hoarding state.

Policies for cache management are implemented based on a combination of a user provided file-priority list with the *Least Recently Updated (LRU)* information. If update conflicts occur during reintegration, the system ensures the detection and confinement of update conflicts and provides recovery mechanisms [114].

Weakly Connected Operations

Weak connectivity is supported by using object or entire volume callbacks for validating the mobile cache. Volume callback is *pessimistic* since invalidating a volume invalidates all the subobjects contained by that volume. The reason why this mechanism is suitable for weak connectivity is that the cache invalidation information communicated between client and server is reduced. The middleware supports automatic adaption, since the file system proxy can determine whether object or volume callbacks are best for a particular connection, based on factors such as cached data structures and changes in connectivity.

This approach of variable callback granularity attempts to minimize the cost of validation and invalidation and therefore provides effective support of operations for weakly connected clients [114].

Isolation-only Transactions

As disconnected operations may result in data inconsistency due to conflicting operations on multiple disconnected devices, the *Isolation-only Transaction (IOT)* concept is introduced. The idea of an IOT is to automatically detect read-write conflicts. Coda's file system proxy takes care of the IOT execution. Consistency guarantees depending on the system connectivity conditions is provided. However, it does not guarantee failure atomicity—it guarantees permanence only conditionally. After its termination, an IOT enters either the *committed* or the *pending* state, depending on the connectivity condition. Another state is *resolution*, where the conflict can be either automatically or manually resolved, before the new result will be committed to the server [114].

3.5.9 Odyssey

Originating from a research project at CMU as well, the *Odyssey* system [18] addresses an application-aware adaptation approach to deal with the diversity and concurrency of applications in mobile environments. The implementation is based on the support of system-coordinated type-specific operations. Concurrent client execution of diverse mobile applications is supported, whereas any data operations (read and write) always are performed on the remote server [114, 197].

The Odyssey data model supports a range of advanced formats, ranging from file servers, SQL servers, or Web servers, to video repositories, query-by-image-content databases, or geographical back-end information systems.

Application-aware adaptation is provided by the system's ability to monitor resource levels, notify applications of relevant changes, and enforce resource allocation decisions. The decision of how to adapt apropriately when notified, is left up to each application. Type-awareness is implemented via specialized code components called *wardens*, encapsulating the system-level interpretation to effectively manage a data type. An application-level component, the *viceroy*, is located on top of the wardens, which is type-independent and responsible for centralized resource management. The application-aware adaptation is implemented by a collaborative relationship between these components, consisting of the following two parts [114]:

Data-centric Relation. The relation between the viceroy and its wardens defines fidelity levels[29] for each data type and links them to the resource management components.

Action-centric Relation. The applications can control the selection of fidelity levels supported by the wardens.

Odyssey is integrated into NetBSD as a new VFS file system, and provides an API for application development [162]. Example applications for using the Odyssey API are given by a *Video player* and a *Web browser*. The latter one, for instance, implements a HTTP proxy making use of the fidelity levels for determining a compression level for distilling the remote contents (e.g. compressing images) [114]. Coda and Odyssey are fundamental parts of *Aura*, a new CMU project focusing on *distraction-free ubiquitous computing* [4].

3.5.10 Rover

The *Rover* toolkit is a research project at the Massachusetts Institute of Technology (MIT), providing a framework for developing mobile applications [115, 116]. This framework is based on a distributed object system on top of a client-server architecture. The Rover toolkit is based on the assumption that client applications typically run on mobile hosts, whereas server applications typically run on stationary hosts and maintain the long-term state of the system. Based on this scenario, the constructions of applications that allow useful operation in environments with changing computing resources and with intermittent connectivity (bandwidth variation, latency, and cost) is simplified. The support for this kind of flexibility is based on two ideas [115]:

Relocatable Dynamic Objects (RDO). An RDO is an object with a well-defined interface that can be dynamically loaded into a client computer from a server computer (or vice versa), in order to reduce client-server communication requirements. An RDO might be as simple as a calendar item with its associated operations or as complex as a module encapsulating part of an entire application (e.g. the GUI of the calendar tool). RDOs are suitable for accessing distributed and remote applications from the clients, but the clients can also use RDOs to export computation to servers.

[29] *Fidelity* is used to describe the degree to which client data matches the reference copy at the server. Fidelity has many dimensions, from of which one universal dimension is consistency. For instance in Video applications, fidelity could have two additional dimensions: frame rate and image quality of individual frames.

Queued Remote Procedure Call (QRPC). A QRPC is a communication system that permits applications to continue to make non-blocking remote procedure calls even when a connection is broken, i.e. messages are not lost due to timeouts, but instead queued into a logfile and retransmitted when the connection is restored.

Rover's mechanism for providing data consistency is based on two stages. First, the RDOs updated by the client applications are being propagated to the server using QRPC. Secondly, when the QRPC for an update arrives at the server, it is being passed to the appropriate server application. Using the RDO version vector maintained by the toolkit, the server checks whether the RDO has changes since it was imported by a mobile client. A built-in concurrency control allows conflicts to be resolved automatically. Additional support for detecting conflicts and passing them to the application for reconcilation are provided. The resulting RDO is sent back to the client [115].

There exists a set of mobility aware applications that have been developed using the Rover toolkit. Two application transparent solutions are [114]:

- *Rover NNTP proxy*,
- *Rover USENET reader proxy*,
- *Rover HTTP proxy*.

The latter one provides an interesting proxy for Web browsers, allowing the user of an existing browser to already request the next document while still waiting for the current one to be downloaded. This increases speed and usability of browsing the web.

Application aware solutions implemented using the toolkit are:

- *Rover Exmh*, an e-mail browser,
- *Rover Webcal*, a distributed calendar tool,
- *Rover Irolo*, a graphical rolodex tool,
- *Rover Stock Market Watcher*, a tool that obtains stock quotes.

Rover Exmh, for instance, uses three types of RDOs: mail messages, mail folders, and lists of mail folders. By using this level of granularity, many user requests can be handled locally without any network traffic. Upon start-up, the list of mail folders, the mail folders the user has recently visited, and the messages in the user's inbox folder are being prefetched. Another example solution, the Rover Webcal distributed calendar tool uses two types of RDOs: calendar items (appointments, daily to-do lists, and daily reminders) and calendars (lists of items). At this level of granularity, the client can fetch calendars and then prefetch items using a variety of strategies (e.g. plus or minus one week, a month at a time, etc.) [114].

3.5.11 Bayou

Another interesting research project, conducted at Xerox PARC, resulted in the *Bayou* system. The Bayou design focus is on collaborative applications in a mobile computing environment, containing portable machines with intermittent network connectivity [219, 114]. The Bayou system is a result for the exploration of mechanisms

that let mobile clients actively read and write shared data, such as appointment calendars, bibliographic databases, meeting notes, evolving design documents, news bulletin boards etc.

Application-specific mechanisms for detecting and resolving the update conflicts are provided, ensuring that replicas move towards eventual consistency, by defining a protocol by which the resolution of update conflicts is being stabilized. Bayou uses *dependency checks* as a part of its consistency management methods, as well as *per-write* conflict resolution based on client-provded merge procedures. In case of an eventual e.g. client originating conflict, consistency is guaranteed by the ability of rolling-back the effects of previously executed write operations. The order for roll-backs is given by a global serialization order.

In the Bayou file system, there exists a set of servers, *each* containing the complete database in *full* replication. The Bayou API is given as a basis for client applications interacting with servers. The Bayou API is implemented as a *client stub* bound with the application, and contains the underlying client-server protocol, which supports two atomic operations:

- `Read`, permits queries over a data collection,
- `Write`, allows insertion, modification, and deletion of a number of data items in a database.

A client and a server may be co-resident on a host, which could be the case on a laptop or a PDA running in isolation due to a temporary connectivity break (see *Extended Client-Server Model* in section 3.5.1). Access to one server—even if on the same physical host—is sufficient for a client to perform useful work. The client can send `Read` queries requesting the data held by that server and submit `Writes` to the server. Once the server has accepted a `Write`, the client has no further responsibility for that command, i.e. the client does not have to e.g. wait for the `Write` to propagate to other servers.

Bayou introduces a file system model for weakly consistent replication with free read and write access. Although the inititating `Read` and `Write` operations are performed at a single server, the clients do not have to depend on interaction with a single server. This is a remarkable advantage for mobile environments, where switching between servers often is desirable [114].

3.5.12 Personal File System

The *Personal File System (PFS)* [20] is a portable network file sharing system designed for mobile computers. PFS allows file servers to be mounted as network drives onto the laptop computer, supporting continuos access even when the computer is disconnected from the network. Modifications on client or server operating system are not necessary.

The framework is based on file servers on stationary hosts and mobile clients. Cache storage on the client is supported, and which dynamically adapts to the variation of network speeds and bandwidths up to disconnection. The PFS is implemented within the UNIX user layer, and communicates with client kernel through traditional NFS.

The name, *personal* file system, is based on the assumption that a single user uses one mobile machine at a time, and that an optimistic file consistency guarantee is acceptable. The main features supported by PFS are [218]:

Disconnection Support. Each PFS client possesses one cache storage. During disconnection, the local cache is being accessed instead of the remote network drive. When reconnecting, the cached copy is being re-integrated with the file server.

Automatic Adaption to Network Environments. PFS specifies a set of algorithms to synchronize files between server and clients. The file transfer performance is being measured by the PFS middleware, and the algorithm suitable to the current environment is automatically being chosen.

System Portability. PFS is implemented as a set of user layer programs and is built on top of the TCP/IP and NFS stack. Thus, for integrating PFS into other environments, kernel modificatons on client or server are not needed at all. PSF has been tested on FreeBSD-2.2.8R, BSD/OS 3.1, RedHat Linux 5.2, SunOS 4.1.3, and Solaris 2.7 [20].

3.5.13 Recommendation

Over-the-air synchronization is an interesting feature, but together with emerging terminal capabilities and methods to access corporate networks (even on a packet-level, *see* section 3.4), the next step would be to integrate synchronization into a transparent background service.

Existing mobile computing file systems are still somewhat emphasized on weakly connected laptops, and it would still be a challenge to integrate mobile computing file system functionality into a wireless OS. EPOC is an open platform approach, which would allow the implementation of such a background service. This approach, however, will certainly be more difficult to implement in other native wireless OSs (like e.g. Windows CE).

The focus on the disconnected state might have to be redefined. At least in theory, future mobile computing scenarios will be based on permanent network access (*always on*) provided by packet switching. On the other hand, for locally running terminal applications, it is not feasible to transmit all the time (e.g. when the contents is *only* a contacts database with relatively low modification rate).

However, these advances given by the 3G service space should be monitored closely and research should be directly adapted based on the parameters of the latest technology. This is important because some of the strong assumptions regarding the limitations of the wireless network or the mobile computer are being relaxed or nullified by newer technological developments.

Although mobile computing file systems will be problematic in allowing operator service differentiation (*see* section 4.3.2), this will become a feasible and highly valuable middleware approach. Weakly connected full clients, currently mostly represented by laptops, in the future partly replaced by PDAs, will together with the corporate network form a logical distributed system. Information and program objects (e.g. mobile agents [113], CORBA [82]) will be transfered between the network hosts, and middleware will be needed to enable transparency.

3.6 Remote Desktop Access

3.6.1 Introduction

Mobile business solutions serve to enable the user to work while on the move. From a user's point of view, this implies allowing the same access to the same applications as on the user's office computer desktop. One approach is to make the entire office computer screen available on the move.

3.6.2 An Old Idea

Application users from the X Window System[30] community should be familiar with this type of remote application access. Logging in from any X terminal and redirecting the application server's output to the local terminal has been a common way of remote application access for decades. When accessing a server remotely, the X Window System only sends commands and events to the local server, which in turn renders these to pixels.

3.6.3 Thin Client

Another possibility for forwarding an entire graphic screen or window over the network, is to send the entire pixel matrix every given time interval. This is obviously more costly and inefficient than the X approach. On the other hand, as during common desktop work (such as e.g. typing text or using a web browser) only some parts of the entire screen change, the use of appropriate graphic compression and reduction algorithms is feasible [190], and allows the client to be kept even *thinner*.[31] Even the common algorithms JPEG for still images and MPEG for moving images are suitable for this purpose [191].

3.6.4 Virtual Network Computing

An interesting platformwide solution is the *Virtual Network Computing (VNC)* approach [191], developed at the AT&T Research Laboratories Cambridge [27]. Remote application access allows access to centralized resources through simple, inexpensive networked devices acting as clients to more powerful server machines. In VNC, server machines supply not only applications and data, but also an entire desktop environment that can be accessed from any machine connected to the Internet. Independent of time and place of accessing a VNC desktop, its state and configuration (right down to the position of the mouse pointer) are exactly the same as when it was last accessed (*see* figure 3.14).

The main features of VNC include [27]:

- No state is stored at the viewer. If you leave the desk, and reconnect from a remote device, you can finish the sentence you were typing. Even if the client crashes or is disconnected, the remote session remains.

[30] The X Window System is a trademark of the Massachusetts Institute of Technology (MIT).

[31] Consider a provider of hosted remote applications, using efficient and opitmized pixel data instead of graphic commands for transmission. If now e.g. the remotely running application operating system is changed, only the pixel based output redirection has to be reimplemented. The same is valid for the thin client.

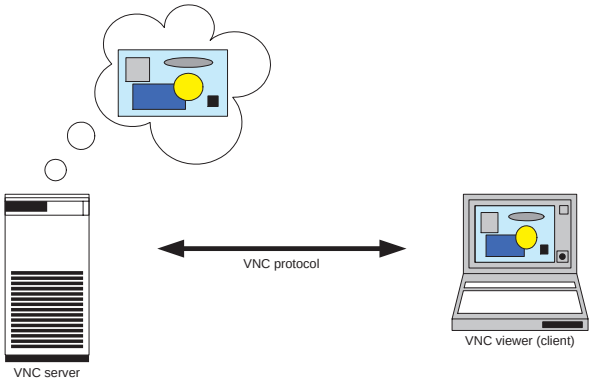

Figure 3.14: The VNC architecture

- VNC provides a thin client solution, which makes the solution interesting for PDA use. The Win32 viewer, for example, is about 150 kilobytes in size and can be run directly from a floppy. There is no installation needed on the client.
- The solution is platform-independent. Implementations exist for Windows, Linux, and Solaris, and the common VNC protocol allows arbitrary access across these platforms.[32] Furthermore, the simplicity of the VNC protocol makes it easy to port to new platforms. For example, a Java viewer is available, which can be run in any Java-capable browser.
- Multiuser-support is implemented. One desktop can be accessed by several users at a time, allowing e.g. remote group collaboration.
- The solution is free and open source. Both client and server parts are freely distributed via the web [27].

3.6.5 Mobility Potential

Since the VNC system is relevant for future wireless and mobile terminals, the compatibility to mobile communication has to be studied. If the capacity of the mobile network allows this kind of application, one could assume that remote desktop access technology could become a feasible foundation for mobile business solutions. In appendix A, VNC's applicability to 3G communication systems is being verified on the basis of simple data traffic load measurement series.

Conclusion

Obviously, one could argue that more measurement series should have been done in order to improve statistical stability. On the other hand, this small experiment was intented to give an overall picture about the data amounts being transmitted for clarifying whether systems like VNC could be considered as relevant for mobile solutions. Due to the nature of mobile and especially *handheld* devices, the screensize used for this measurement is unrealistic. However, assuming a wireless connected

[32] For instance, a desktop running on a Linux machine may be displayed on a PC.

PDA with a screen dimension of $640 \cdot 200$ pixels (such as for the *Nokia 9210*), we apply a best-case scale factor on our result, which is determined by

$$f_{scale} = \frac{640 \cdot 200}{1152 \cdot 864} = 12.86\% \tag{3.14}$$

This definition is obviously made under the assumption that the amount of data traffic is proportional to the screensize.[33] Without scaling, we saw an average bi-directional transmission speed of 22 KB/s, and with screen scale f_{scale}, the result is 3 KB/s (as in table A.1, last row, last two cells).

Now consider the capacity of 3G mobile networks, such as GPRS allowing bit rates from 9 kb/s to more than 150 kb/s per user [132] (from 1.125 KB/s to 18.75 KB/s respectively). Later, UMTS will allow 144 kb/s with high-speed movement, 384 kb/s for pedestrians, and 2 Mb/s for stationary access [132] (18 KB/s, 48 KB/s, and 250 KB/s respectively). Obviously, VNC's average transmission rates, especially the scaled values, do *not* seem unrealistic for wireless traffic at all. For a discrete value range with a width of five unit steps[34], compared in parallel with the data transmission rate range for GPRS and UMTS, a qualitative proof for the feasibility of VNC-type applications, under UMTS and especially the scaled version of VNC under GPRS and UMTS is given (*see* figure 3.15).

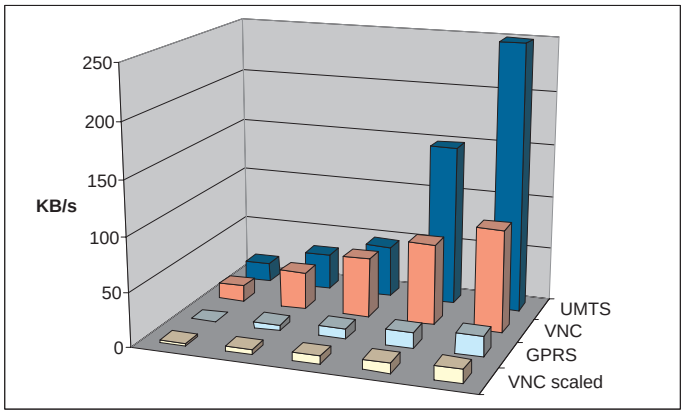

Figure 3.15: Transmission speed comparison

Furthermore, the drastic difference between uplink and downlink data transmission, makes this application suitable for assymetric systems. Another advantage of the VNC solution is its thin client characteristic; client devices need only limited CPU power just sufficient to simply display the graphic data. The AT&T Research group demonstrates the remote use of a X Windows desktop from a Windows CE PDA (although over wired LAN only) on their webpage [27]. Finally, the VNC protocol can also be run through the SSL protocol, thus security is guaranteed.

Other, similar thin client solutions include those built around the Citrix ICA protocol (e.g. Citrix's *Winframe* and *MetaFrame* [3, 159] and Insignia Solutions' *Ntrigue*),

[33] Although a change of color depth was not taken into consideration in the scaling, this is obviously still a very rough and over-optimistic approximization.

[34] The choice of five is based on the number of our previous testcases *with* activity.

Tarantella's *Enterprise 3* [25],[35] GraphOn's *RapidX* [7], and Microsoft's Windows-based *Terminal Server*[36]. The trade-off with these commercial systems (with the exception of Microsoft's version) is that, unlike in X Windows System, they use proprietary protocols without any open-source. Citrix's ICA protocol has become a popular mechanism for accessing PCs remotely, but it appears to be closely tied to the Microsoft Windows GUI, and is therefore not an ideal general purpose solution. Microsoft has developed its own protocol, *T.Share*, based on the ITU T.120 protocol [104], which already is being used in Microsoft's NetMeeting conferencing software product [191].

3.6.6 Recommendation

Although this seems an unelegant way of mobile communication, it is still the best way one could enable remote access to *any* application.[37] The client for such a solution can additionally be kept relatively thin (compared to an approach based on X Windows, *see* section 3.6.3). Thus, this is a clear alternative to carrying locally installed applications around with you all the time. After all, this approach is a hosted solution, meaning that the remotely accessed application is only restricted by the hosting server's own power and not the mobile device nor the communication link. A starting point for the development of mobile middleware is driven by the need to combine VNC technology with screensize and depth adaptation. One could also think of hosting a smaller-sized, or less colored (e.g. 8 bit) *virtual mobile desktop* on the corporate server, instead of accessing a fullscreen remote desktop.

From the perspective of providing hosted applications, this concept allow more powerful applications than any markup language based solution. Especially the use of complex markup languages on small screens seems as a clear overhead, compared to this alternative. Whether the remote desktop is being transmitted in pixels or in GUI commands, this thin client approach offers superior application integrity and transparency. One disadvantage could be in the difficulty to integrate device specific functionality (*see* section 3.7.7) into a remote desktop (e.g. placing calls from the contacts database), but we once again refer to VoIP solutions as alternative middleware solutions [86, 195].

3.7 Wireless Terminal Applications

3.7.1 Introduction

Currently the most interesting aspect of 3G mobile applications, is based on making use of the third dimension of the 3G service space, *terminal capabilities* (*see* section 2.1). Increasing wireless terminal capabilities finally allow the deployment of *real* applications, being locally executed in the terminal device, and using the communication facilities simply as a network interface. By developing wireless appli-

[35] In fact, the Nokia 9210 Communicator features support for the Tarantella Enterprise 3 software. Acting as a Mobile Application Portal (MAP), the software allows users to have remote access to all corporate applications without touching the servers they run on. This includes UNIX, Windows, Mainframe, AS/400 or web-based applications from any GSM-capable location [217].

[36] Also commonly known by the name *Hydra* [14].

[37] Note that the graphic screen of currently available WAP terminals with a monochrome dot matrix display of $84 \cdot 48$ pixels (such as e.g. Nokia 8310) can be represented by $\frac{84 \cdot 48}{8} = 504$ B, which renders the use of abstract markup languages for tiny devices like these completely questionable.

cations based on locally running terminal client software, an equal balance between increasing mobility of information and increasing mobility of application computation can be reached—more easily than with e.g. markup language based solutions (*see* section 3.2).

This section will introduce different approaches of making use of this dimension in mobile business applications. The novelty in these different approaches is based on the fact that they are based on the programmability of 3G handsets. Due to this preassumption, it should be clear that none of these application approaches would be possible without programmable wireless handsets!

The structural differences of these approaches, as well as their feasability are described in sections 3.7.4 to 3.7.7. Out of this set of structural application approaches, the concept of Hybrid Thickness Client Applications has been implemented for experimental purposes. Prior to that, a discussion about enabling technologies such as EPOC and Mobile Java, are given in sections 3.7.2 and 3.7.3.

3.7.2 Application Design with Symbian

The Symbian application design is suitable for client-server architectures, where many applications are clients that use resources of servers. A client application is a program that has a user interface, and server applications are programs that only can be accessed over well-defined interfaces from other programs. The client faces the user in an interactive way, whereas the server handles most of the procedural tasks and ensures a timely response to all the clients [166].

For developing client applications, the EPOC Symbian supports two general concepts, determining the fundamental type of the GUI:

Crystal Device Concept. A Crystal device specifies a wireless terminal with a horizontally positioned screen, i.e. a screen with higher width than height. Examples for Crystal devices are Psion's and Nokia's Communicator devices.

Quartz Device Concept. A Quartz device specifies a wireless terminal with a vertically positioned screen, i.e. a screen with higher height than width. Examples for Quartz devices are Palm-styled PDAs.

Within the Symbian software development framework, application development is currently supported in three programming languages [210]:

- *C++*, optimized for EPOC system development and high-performance application programming;
- *Java*, object-oriented language support for efficient development or deployment of applications, applets and APIs;
- *OPL*, a proprietary BASIC-like language with a long history in Symbian's and Psion's EPOC and SIBO software, for rapid application development.

For all these three alternatives, PC-based emulators for initial development and debugging are provided by Symbian. Before packaging and distributing the application, the program is transfered from the emulator to an EPOC device for final testing. In the case of using C++ as programming language, the application has to be rebuilt before transferring it to an EPOC device, due to the difference in the low-level instruction set. In the case of using OPL as the programming language, the application can even be written directly in the EPOC device [210].

In the future, Symbian version 6.1 will deliver GPRS-based packet data, WAP 1.2, and additional Bluetooth functionality [211]. For a more detailed description of the Symbian specific application build process, *see* appendix C.

3.7.3 Mobile Java

The porting of the Java Virtual Machine (JVM) onto a wide range of computing environments has evolved the Java language into a de facto standard for fast and effective application development with good quality. The reason is given by the increase of processing power and memory space in the recent histrory, which has led to less usage of machine level programming (assembler), and more usage of high-level object-oriented languages (e.g. C and C++). Additionally, the Java *bytecode* concept has showed that separate compilation of a high-level program code on different operating systems is not necessarily the only way. Today's computing power allows support for *Just In Time (JIT)* compilation or even runtime interpreters. The benefits are clear; given this platform-independent approach, a program can be written once and runned everywhere. The development benefits from shorter development cycles, and the user community benefits from a standardized and transparent application look-and-feel.

For mobile solutions, the *Java Platform 2 Enterprise Edition (J2EE)* has proven to enable efficient development of back-end and middleware solutions powering mobile applications. The emerging capabilities of wireless terminal devices allow more advanced and high-level terminal programming, but still have physical restrictions (memory size and processing speed) for making use of e.g. the *Java Platform 2 Standard Edition (J2SE)*. For enabling software development in the same programming language on the terminal devices as well, a *subset* version of the Java environment is needed, allowing it to be used with lower processing power and smaller memory footprint. Obviously, the use of the same or similar software components on back-end, middleware, and terminal applications, offers a feasible way for providing consistent end-to-end solutions. The following sections introduce current key technologies for building a mobile Java platform.

PersonalJava

In 1998, Sun Microsystems introduced three new technologies, as a reaction to the emergent market of limited resource devices: *PersonalJava*, *EmbeddedJava* and *JavaCard*. Each of these technologies was specified for a set of target devices, with a defined profile [152].

PersonalJava introduced the first Java environment optimized for consumer electronic devices with graphical user interface and communication capabilities, such as PDAs, webphones, or set-top boxes. The current PersonalJava 1.2 specification is based on a reduced virtual machine, which is a subset of the JVM 1.1, and an API, which is a subset of the JDK 1.1 API. PersonalJava technology introduces a variety of optimizations so that the memory usage can be dramatically reduced. In terms of the graphical interface, PersonalJava also brings huge changes, allowing adaptation to the target screen devices.

Based on the specification of the API definition, any target device must include network and graphic user interface features, allowing an implementation of PersonalJava. At the graphical level, Swing is not supported, leaving AWT as the only

choice. This decision is understandable due to the resource constraints on the target devices.

A *PersonalJava Emulation Environment (PJEE)* was developed in order to enable PersonalJava applications to be executed on a common PC, facilitating the development process.

JavaPhone

The *JavaPhone* API is an example of a vertical extension to the PersonalJava platform. It is a result of the cooperation between Sun and key leaders in the telecommunications industry. The API is designed to provide access to the functionality unique to client devices with telephony capabilities. The functional parts of the JavaPhone API include [207]:

- *JTAPI Core*, direct telephony control;
- *Network Datagram API*, datagram messaging;
- *AddressBook API*, access to address book information;
- *Calendar API*, access to calendar information;
- *User Profile API*, access to user information;
- *Power Management, Power Monitor APIs*, power management;
- *Installation API*, application installation mechanisms.

Based on this set of APIs, two profiles for meeting the needs of concrete scenarios are specified, the *Internet Screenphone Profile* as well as the *Wireless Profile*. In later sections, we are referring to the wireless profile (*see* figure 3.16.

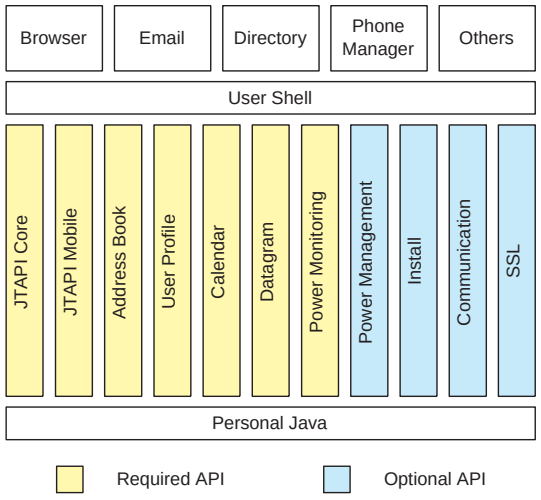

Figure 3.16: JavaPhone wireless profile [207]

J2ME

In 1999, during the JavaOne Conference, Sun regrouped its Java technologies based on the recognition that one size does not fit all. Sun regrouped its Java technologies into three editions: Java 2 Platform Enterprise Edition (J2EE), Java 2 Platform Standard Edition (J2SE), and *Java 2 Platform Micro Edition (J2ME)*.

With the J2ME creation, Sun grouped all the Java technologies, that are targeted to resource-constrained devices, into one common platform. J2ME meets generic needs for application mobility. The benefits of J2ME are targeted at device manufacturers, who build a diversity of information devices, service providers, who whish to deliver content over those devices, and content creators, who want to make compelling content for small, resource-constrained devices [206].

At a high level, J2ME is currently targeted at two configuration groups, representing two broad categories of products (*see* figure 3.17):

Connected Device Configuration (CDC). This category represents the set of shared, fixed, conneted information devices, such as TV set-top boxes, Internet TVs, Internet-enabled screenphones, high-end communicators, and automobile entertainments/navigation systems. The range of user interface capabilities in this category is large, and memory budgets in the range of 2 to 16 MB. These devices have access to persistent, high-bandwidth network connections, most often using TCP/IP. To ensure upward compatibility, the CDC is a superset ot the CLDC.

Connected Limited Device Configuration (CLDC). This category represents personal, mobile, connected information devices, such as cell phones, pagers and personal organizers. Compared to desktop computer systems, the user interfaces of these devices are very simple, and minimum memory budgets starting at about 128 kB. In this category, network connections are intermittent, with low bandwidth, and often *not* based on TCP/IP. The CLDC includes some new classes, not drawn from the superset J2SE APIs, designed specifically to fit the needs of small-footprint devices.

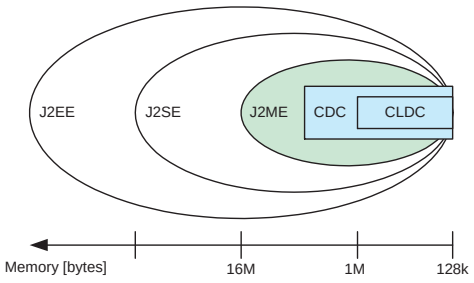

Figure 3.17: Memory footprint of Java subsets

Currently being closely related to CLDC, the *K Virtual Machine (KVM)*[38] technology is a compact, portable JVM specifically designed for small, resource-constrained

[38]The letter *K* in *KVM* stands for *kilo*, based on the idea of being able to measure its memory footprint in kilobytes.

devices [206]. The primary design goal for the KVM technology was to construct the smallest possible JVM that would run in a resource-constrained device with only a few hundred kilobytes of total memory, but still would maintain all the central aspects of the Java programming language. KVM is suitable for 16 bit/32 bit RISC/CISC microprocessors, with a minimum required memory space of 128 kB, including the virtual machine, the minimum set Java class libraries (specified by the configuration), and some heap space for running the applications. However, a more typical implementation requires a memory space of 256 kB, consisting of 128 kB heap space for the applications, 40 to 80 kB for the virtual machine, and the rest for configuration and profile class libraries.

The actual type of a KVM implementation may vary significantly. Some implementations include KVM technology on top of an existing native software stack to provide the ability to download and run Java content on the device. Other implementations use KVM technology at a lower level for also implementing system software and applications.

CLDC technology runs on top of KVM technology. However, over time it is expected that CLDC technology will run on other J2ME virtual machine implementations and that the KVM technology may support other new configuration definitions [206].

Whereas a *configuration* defines a platform for a *horizontal* category of devices, each with similar physical requirements, the J2ME specification also introduces a second key concept: A *profile* extends a configuration and addresses the specific demands of a certain *vertical* market segment.

For instance, together with the CLDC, the *Mobile Information Device Profile (MIDP)* API set, provides a complete J2ME application runtime environment targeted at mobile information devices, such as cellular phones and two-way pagers. The MIDP specification addresses issues such as user interface, persistence storage, networking, and application model. A standard runtime environment is provided, that allows new applications and services to be dynamically deployed on the end user devices. The MIDP specification has been developed by an expert group composed of over 20 companies representing the wireless industry.

Industry Support

PersonalJava and J2ME technologies are experiencing increasing support by the indusrty. In June 2001, NTT DoCoMo had together with LG Telecom and Nextel sold over 3 million Java technology-enabled wireless handsets in Japan since the product's launch in January the same year, and over 90 *i-appli* Java applications are avaible. Other operators around the world have similar plans to deploy or trial Java technology-based services and devices in 2001, including Cingular Interactive, Far EasTone, J-Phone, KDDI, Omnitel, One2One, SmarTone, Sprint PCS, Telefonica and Vodafone [208, 74, 96].

In addition to operator support, wireless device manufacturers, are either shipping or planning to ship devices integrated with Java technologies (*see* figure 3.18). These companies include Nokia, Motorola, Siemens, Samsung, LG Electronics, Panasonic, Fujitsu, NEC, Sony, Mitsubishi Electric, Sharp, Psion, RIM, and Inventec [208].

Figure 3.18: Motorola Accompli 008, Siemens SL45i, and i-appli.

3.7.4 Generic Applications

Generic applications for mobile business solutions could be defined as the set of applications, enabling a variety of business user segment related tasks and activities. Unlike in 2G middleware solutions, among these generic applications, the famous *killer applications* such as mobile access to e-mail and shared calendars [149, 40] can be regarded as already solved under 3G. Mobile device manufacturers already offer e-mail clients and calendar applications as standard built-in applications. Although these generic applications might not be implemented to access corporate data in a specific manner (e.g. firewall support, XML, etc.), it might still be very hard to convince the user of a 3G wireless handset to access his e-mails through a web browser based portal, instead of using a native e-mail client.[39] The development of competing terminal client software is one alternative, but the possibilities to challenge large device vendors' or software companies' e-mail or calendar solutions can be regarded as very low.

An opportunity for middleware would be to study the different communication interfaces and APIs provided by future de facto standard applications, and develop transparent end-to-end solutions bridging the client application with the corporate application through the mobile portal (*see* figure 3.19). The mobile portal is needed as the logical middleware part, to exchange information with entities, which would be unaccessible and unfeasible for the client alone (e.g. reading a corporate database and returning subset data in XML to the client).

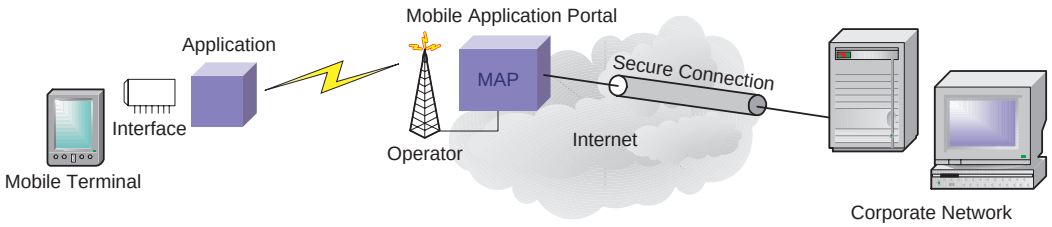

Figure 3.19: Bridging client interface with the corporate network

Another opportunity for developing generic mobile business applications could be to focus on *non-killer* applications, and to define and develop a unique de facto standard application suite. There exists a variety of research on ranking applications by end user needs, business segments, and enabling technology issues (e.g.

[39] However, there is a generation of users (the *Hotmail generation*), who is used to web-based e-mail clients—as they never have seen anything else. We assume that a user who has the experience from both web-based and standalone application e-mail client, will prefer the standalone version, given equal access to remote information (e.g. Intranet access) in both options.

QoS) [40]. For business solutions, the topic of *mobile collaboration* is an attractive area, and current office solutions do not actually provide efficient solutions for this issue yet.

3.7.5 Customized Applications

A *customized application* would be an approach, defining a client-server architecture based on the needs of a customer or industry specific segment, (*see* figure 3.20). However, this *vertical solution* would be difficult to define, but on the other hand probably easier to market (given the uniqueness of that specific solution). A starting point for defining an application for a business usage segment could be given by an intensive study of the needs of mobile work force in SMEs (e.g. engineering and construction, healthcare, transportation, cleaning services, etc.). Other, more generic vertical markets are: finance and professional services; petro-chemical, pharmaceutical, and manufacturing; energy and utilities; retail, leisure, and distribution; IT, communications, and media; construction and engineering; automotive and transport; and public sector.

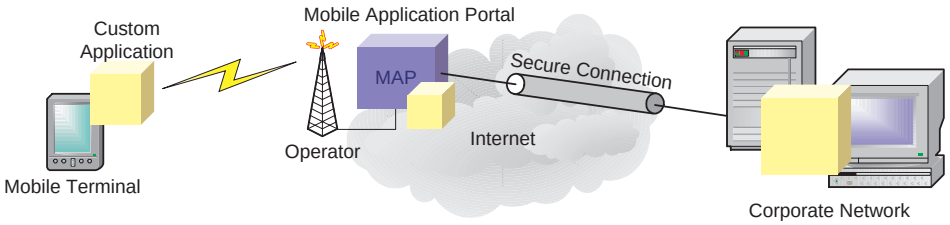

Figure 3.20: Defining a customized end-to-end application

Another opportunity would be the redefinition of a generic killer application, by applying the user needs of a specific business segment onto an (existing) application. The result would be a *customized killer application*, meeting customer and industry specific business user needs. For instance, an e-mail client might slightly differ in its implementation, if it would be based on e.g. consumer versus business user needs.[40] This *customized killer application* could be implemented as a client application connected to the mobile portal, and should thus provide added value to the specific customer segment, compared with its generic pendant, and result in competitive advantage in that segment [144]. However, the real end user need has to be carefully verified in advance.

3.7.6 Application Skeleton

The construction of a generic *application skeleton* would minimize the need to focus on any specific application or segment. Based on the idea that the amount and variety

[40] In the case of an e-mail client, the consumer might prefer a simple interface for easy messaging of text and multimedia information. Furthermore he would like his e-mail client to manage all his existing e-mail accounts on the web. The business user might also prefer a simple interface, but he would additionally prefer support for accessing existing shared mail and discussion folders within the corporate network. Integration of other groupware functionality such as e.g. shared meeting scheduling, or *request for review* support in e-mail messages might be other examples for business user needs.

of existing terminal applications for the evolving new software and OS platforms on wireless 3G handsets is very small, any tools for speeding up the software development should be feasible. For mobile business solutions, one could think of defining a generic library of classes, objects, interfaces, or components, on top of which the development of any new mobile business application would be much easier and faster, than developing it from scratch (*see* figure 3.21).

The additional definition of both client *and* server version of such an application skeleton, i.e. the library including corresponding component *pairs* for both client and server application components, would ease the provision of end-to-end services.[41]

The application skeleton should define these components applied for mobile solutions, taking constraints and advantages of mobile client-server computing into account.[42] In this way, such an application skeleton would distinguish itself from already existing developer classes and libraries. The modularity given by the Java programming language (*see* section 3.7.3) surely is an advantage in this approach. One could also think of orientating such a generic application skeleton towards even wider standardization approaches, such as MExE (*see* section 3.8). An example for the definition of such a Mobile API, although a more network oriented layer can be found in the *OnTheMove* project [75].

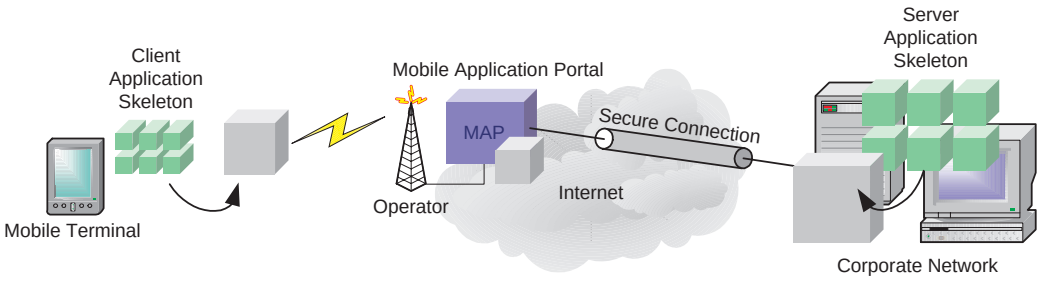

Figure 3.21: A client-server application skeleton

3.7.7 Hybrid Thickness Client Applications

Based on the definition given by Jing *et al.*,[43] the deployment of a *hybrid thickness client application* concept, based on a combination of the thin and the thick client mobile computing concept, has been considered. Our concept is a hybrid of a full local stand-alone application (thick) and a fully remotely executed application (thin). Un-

[41] The CORBA framework provides a similar approach [82], although it is somewhat more focused on distributing executable objects through the network (e.g. from client to back-end systems).

[42] For example, the skeleton could predefine a set of core messaging protocols, each based on different messaging types and QoS (e.g. interactive, background relay, bulk, etc.). When implementing any new application, these underlying messaging routines can be directly applied.

[43] An extreme case is called the *thin client* architecture, which offloads most application logic and functionality from clients to stationary servers. In this architecture, applications in stationary servers are usually mobile-aware and optimized for mobile client devices. The thin client architecture is especially suitable for dumb terminal or small PDA applications.

The other extreme case is the *full client* architecture. The full client architecture can emulate server functions on the client devices and, therefore, is able to minimize the uncertainty of connectivity [114].

like common thin client solutions (such as e.g. markup language based solutions[44]), a hybrid thickness client application based solution uses the local application for representation, and the remote application for generation of information. Compared to the WAP example, instead of using a thin browser for interpreting content in WML, we propose the usage of a hybrid thickness client application, which reads *raw* XML content, and implements the representation in a local Graphical User Interface (GUI). Thus, compared to the markup language based case, our concept implies a separation of content generation and representation between client and server (*see* figure 3.22).

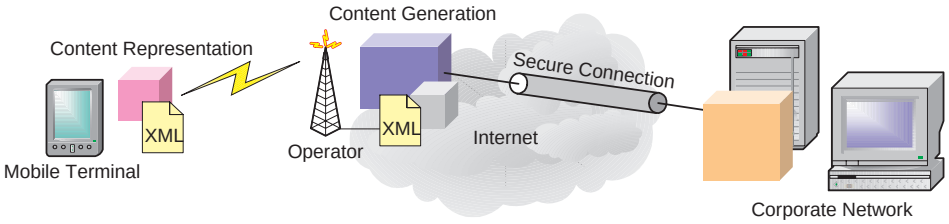

Figure 3.22: The hybrid thickness client application concept

The advantages of this concept are:

Integration. Compared to a fully remotely executed application, the hybrid thickness client application allows additional functionality by integrating device specific functionality, as e.g. provided by JavaPhone (*see* section 3.7.3).[45] The user can e.g. place a call or send messages directly from the hybrid thickness client application. Content and device functionality are not separated anymore, the separation of content generation and representation is accomplished instead (*see* figure 3.23).

Usability. The usage of the terminal GUI does not have to be interrupted by remote server response delays. A locally running GUI provides a higher degree of usability, especially when the GUI provides an API for constructing intuitive interfaces on mobile devices [227].

Reliability. Compared to the full client concept, and considering the relatively limited memory storage and processing speed capacities of upcoming 3G wireless handsets, the device implementation can devote its limited resources fully on the GUI.[46] Compared to thin client concepts, such as WAP, the testing and debugging of different browsers' compatibilities during development is not necessary. Furthermore, as the local client is not dependent on a remote GUI, our approach is more fault-tolerant to the uncertainty of connectivity.

3.7.8 Case: White Pages Application

Based on the hybrid thickness client application concept, an example application was implemented, extending a specific database solution into a mobile solution. As

[44] *see* section 3.2.

[45] This feature is essential, as long as telephony is carried out over a separate bearer. In the future, however, even phone calls might be based on IP.

[46] This concept is similar to an X-terminal.

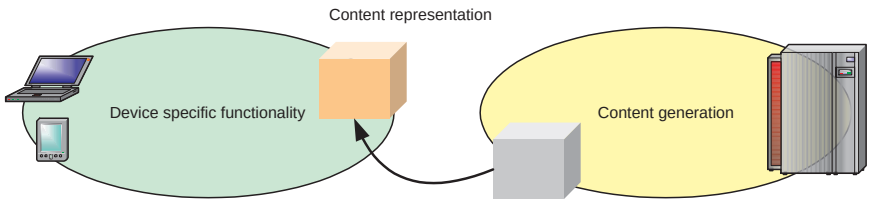

Figure 3.23: Migration of content representation towards devices

a specific database problem, the implementation of remote access to a corporate address book was chosen, also known as the *white pages*. The white pages application is based on a set of employee contact entries stored at corporate premises, which could be an SQL, Lotus Domino, Microsoft Exchange server, or even a simple spreadsheet file. Instead of having to manually copy or program the contacts into the mobile device's address book, the idea with *White Pages 1.0* (*see* figure 3.24) is to access the remote set of contacts instead. This example clearly illustrates the advantages of the hybrid thickness client application concept, since it clearly provides added value compared to a browser based database request application.

Features

The purpose with the implementation of this case was to demonstrate the benefits proposed by the hybrid thickness client application concept, and furthermore give an impression of what future mobile business applications could look like. As a platform for the implementation, the PersonalJava programming language on the *Nokia 9210 Communicator* with EPOC OS was used (*see* appendix B).

The implementation makes use of the separation between content generation and representation, as the data sent to the mobile client is being generated in XML form by the *Smartner Engine Platform* (*see* section 4.7.2), and the representation is implemented by the local GUI including the XML parser.

Content and device specific functionality are merged, as the user has the possibility to send text messages or to place calls directly from the application.[47] Additionally, he can copy selected entries from the remote contacts database to his local addressbook in the phone. The use of the device manufacturer's own implementation of the Symbian *Command Button Area (CBA)* specification [212], in the Nokia 9210 Communicator consisting of four vertically arranged buttons, is another example for the integration of device specific functionality (*see* figure B.12).

Higher reliability compared to markup language based approaches can be demonstrated by the fact that, if the connection to the remote server for some reason cannot be established, the application does not fail.[48] In this case, the application loads a

[47] In many currently available mobile phones, the content is still isolated from the application. In such cases, the usage can become very frustrating, e.g. when you browse a WML page that contains a phone number, which you want to call, but which cannot be extracted from the WML page. The user will have to write down the number onto a paper or memorize it, until he has closed the WAP browser. The option of extracting a number from the text, is quite a recent feature.

[48] Of course, neither does this application make sense if the network connection gets completely lost. But for temporary network problems or breaks, this application is better compared to a browser based system, where the session easily gets lost.

locally stored XML file, which contains the contacts retrieved during the last request from the remote server.[49]

Figure 3.24: The White Pages application

Results

As an outcome of this implementation project, some major lessons were learned. First of all—as expected—the use of Java as programming language resulted in a very slow application, especially as some graphic components had to be implemented manually.[50] Being used to the Swing API under Java 2, the restriction of using Java 1.1.8 and the Abstract Window Toolkit (AWT) for the GUI, was not satifying. A simple scrollable table had to be implemented manually, since no other predefined components for implementing this type of interface was found. On the other hand, some predefined AWT components included by the toolkit do not work correctly, which caused the application to crash at runtime.[51] Hence, these components were replaced by manually implemented components.

This minor failures definitely are due to the fact, that our Software Development Kit (SDK) was only a *beta* version at the time of the implementation. Tremendous difficulties in installing the environment and getting it *up and running* were discovered. Learning the SDK and its related components and APIs, especially getting used to the Java classes subset given by PersonalJava (*which classes to use and which not?*) was very time consuming. The implementation of the application thus took more than one month; with this experience, it would *now* definitely take much less time to implement a similar application from scratch.

The package build process, i.e. the process of going from source code to a runnable application, is somewhat complicated (*see* appendix C). As the emulator has to be restarted after terminating a Java application—such a restart took 1 minute 30 seconds in our environment—the testing of small changes in the source code was very unefficient. We hence decided in an early phase to develop the application in a manually defined *shell*, i.e. as a Java standalone application inserting one instance of our

[49] For a more detailed description of components and implementation, *see* appendix B.

[50] The manual implementation of GUI components cannot be as efficient as lower level, native implementation.

[51] These erroneous components were `java.awt.Choice` and `java.awt.CheckBox`.

application as an object. This seemed to be a very efficient alternative, and the real advantages of Java's object-orientation were experienced, when finally embedding the application into the emulator, and finally the real device (*see* appendix C).

Further Research

From an end-user perspective, the question arises why to use a third party application for managing contacts instead of the built-in native contacts directory. On the other hand, the built-in contacts directory does not offer possibilities for this XML based synchronization, neither does it offer APIs, or open source for mofifying the native application.[52]

Based on this observation, this application could be developed as a background process, a service, running autonomously in the background and automatically updating the locally stored contacts database. This updating could e.g. happen on a time interval basis, or could be initiated by the remote server—for example, when updates in the corporate database occur.[53]

The use of a more specific XML derivate, SyncML (*see* section 3.3.4) could be studied. In this case, even better integration with the client's supported features, i.e. even the client's built-in contacts directory application, could be reached.[54]

Being a smartphone with a wireless OS, the Nokia 9210 Communicator in its current version is still based on 2G communication facilities, i.e. GSM. Although IP connectivity can easily be established through the built-in dial-up networking stack, no packet switching could be tested. As the device does not support simultaneous use of the voice and the data channel,[55] calls can only be placed when the connection to the remote host was closed. Furthermore, since the terminal device cannot be *always online* (as it could with packet switching functionalities), the client application cannot listen to a specific port, and it is unfeasible to implement server initiated updating in this approach. 2.5G based push mechanisms (such as WAP push, *see* section 3.2.2), where server initiated communication starts with an SMS based notification, could have been possible to implement, but would not have been within the intended purpose of this project. With the future support of packet switching functionalities, one could even think of performing a *last-minute* update, i.e. always when a call to a specific person is placed, the existence of the latest version of the contact entry is verified. This could even further result in a *Voice-over-IP (VoIP)* based solution, where an IP based pipe between client device and corporate server is established (instead of a phone call), and the server places calls to the designated PSTN number [86].[56]

[52] It remains still questionable, if a device vendor, such as Nokia, would open its APIs entirely.

[53] This is where current front-end middleware solutions are likely to go in a terminal application dominated future, migrating from server front-end middleware to background processes in client terminals.

[54] Nokia is currently working on their integration of SyncML in the device.

[55] This feature will be supported by GPRS [139].

[56] This hosted voice service approach [195] would completely hide the existence of PSTN numbers to the end-user. People would be addressed by their firstnames and lastnames, instead of a numerical string. This might possibly be a feasible way for migration, too; people could easily get used to phone numbers disappearing in the future [120, 121]. After all, the user does not want to use the phone for dialing at all; the phone should serve the user for reaching people in the first case [157].

3.7.9 Recommendation

Standardized software platform approaches such as Symbian and Java (*see* sections 3.7.2 and 3.7.3) offer compelling possibilities for enabling interesting mobile applications and end-to-end solutions.

On the other hand, given the programmability of emerging mobile terminals, it will be difficult to compete about generic office applications with dominant software vendors. The customized application approach (*see* section 3.7.5) can meet a variety of concrete needs. Other emerging areas for developing customized wireless applications include sales and procurement (project management, automation of the sales and service forces, CRM, and communication management); supply chain management (inventory and logistics managent, insurance-claim notification), and service and support (process control, fleet management, automated meter reading, and asset management) [221].

Applying wireless terminal application technology to Smartner's current product portfolio, the current markup-language based *Office Extenders*(i.e. e-mail and calendar extender) could be transformed. Instead of developing new Web forms for the PDA built-in HTML browsers, a Java-based thin client could handle XML traffic and present the output in a locally running user interface (*see* section 3.7.7). As for the *Database Extender*, one could think of developing a generic database *toolkit*, which allows the customization of database forms and queries on the terminal and back-end side (*see* section 3.7.6).

Like in Smartner's case, the concept of a generic middleware platform for applicaton services has to be carefully studied. There are signals pointing towards a need of such a Mobile Application Platform (MAP) even in 3G and beyond. However, slight modifications to current concepts are mandatory, especially the extension of the concept to an end-to-end platform [189]. An end-to-end platform can be constructed based on an additional generic application skeleton on both front-end and back-end environments (*see* section 3.7.6).

A balance of the MAP into the terminal application direction can be implemented in form of *stub* applications based on the hybrid thickness client application concept (*see* section 3.7.7). Content representation and generation become separated. The generation remains at middleware or back-end premises (e.g. in XML), whereas the representation is performed in the client terminal. Tailored terminal applications result in much more robust end-to-end solutions. For platform-independent architectures, such as Java (*see* section 3.7.3), the effort of terminal-specific tailoring is neglectably small. The verification of the existence and correctness of Java classes used happens before runtime (during compilation), thus testing efforts can be minimized, and restricted to appearance, layout, and look-and-feel.

The hybrid thickness client application concept can be considered as a feasible way for migrating to 3G. Middleware is still needed, but the concept is being modifified based on convergence issues.

3.8 Mobile Execution Environment

3.8.1 Introduction

An interesting example for unifying the existing range of application types, ranging from markup language based solutions (*see* section 3.2) to wireless terminal appli-

cations (*see* section 3.7), is presented by the *Mobile Execution Environment (MExE)* approach (*see* figure 3.25). The MExE framework was created by ETSI and is now being maintained and developed by the 3GPP [34, 35]. In particular, the standardization of MExE was initiated based on results for defininig a Mobile API within the *OnTheMove* project [75].

The aim of the MExE approach is to provide a standardized application execution environment for mobile terminal in general, covering the range between small, light terminals (e.g. mobile phones) and more sophisticated devices (e.g. PDAs, PCs, etc.). The core of the execution environment is based on the ability to negotiate the MExE mobile terminal's supported capabilities with a MExE service provider. This allows applications to be developed independently of any mobile terminal platform [203, 132].

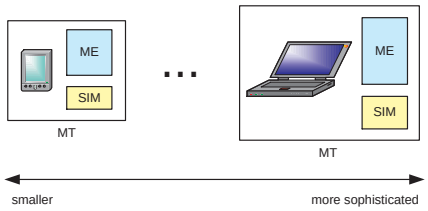

Figure 3.25: The MExE concept

3.8.2 The Concept

Insuring the portability of the variety of application types, across the broad spectrum of multi-vendor mobile terminals, a dynamic and open architecture within the *Mobile Equipment (ME)* and the *Subscriber Identity Module (SIM)* (together forming the *Mobile Terminal (MT)* is defined.[57] The ME and the SIM is taken as the basic assumption, upon which the terminal-independent client execution environment is defined [75].

A mobile network can be a transport bearer for the negotiation, interaction, and transfer of the MExE types of applications, applets and content. However, the network does not have to provide MExE support, instead, a *pipe* between client and another provider of MExE services can be established [203].

The MExE service provider will be available to offer new and interesting services. One example is given by location-dependent services, which can be provided by letting the service provider retrieve the current location information of a MT [75].

3.8.3 The Classmark Definition

To integrate the set of existing mobile application technologies under one common standards *umbrella*, the MExE standardization has begun with the standardization of three application classmarks. Each classmark represents a set of common terminal device features [203].

Classmark 1—WAP Environment. Classmark 1 MExE devices are based on WAP. They require limited input and output facilities on the client side only (e.g.

[57] In particular, this dynamic and open architecture means a common set of APIs and development tools.

as simple as a 3 lines by 15 characters display and a numeric keypad), and enable efficient information access even over low bandwidth connections (*see* section 3.2.2).

Classmark 2—PersonalJava Environment. Classmark 2 MExE devices enabled PersonalJava applications with the addition of the JavaPhone vertical extension (*see* section 3.7.3).

Classmark 3—J2ME Environment. Classmark 3 MExE devices support J2ME applications with CLDC as configuration and MIDP as profile (*see* section 3.7.3).

A typical MExE device only supports a subset of these three classmarks (e.g. one or two).

There are numerous members of the MExE Forum [13] involved in shaping the standard, and all the major and secondary handset manufacturers are present in the committee. It is likely that first MExE conform handsets will be announced during this year, and even Microsoft has shown its interest by proposing a new classmark based on the *Common Language Infrastructure (CLI)*, the $C^{\#}$, and the *ECMAScript language* [203].

3.8.4 Recommendation

Similarly to Smartner's concept, the MExE approach provides integration of different terminal types into one middleware platform. The MExE concept, however, takes this idea one step further: different terminal types are no longer being represented by differtent markup languages alone; the 3G dimension of locally running terminal applications is considered. MExE provides a feasible solution for integrating different types of *application services* into one common platform, and offers a solid way for migration: the MExE initiative supports the change of application from markup language based solutions towards terminal applications, at the same time based on an advanced application middleware platform. The concept enables possibilities for mobile operators to provide differentiated, value added services, and might become a significant standard for 3G applications.

3.9 Summary

One could discuss what is wasting more information and computation resources—markup language based solutions or heavy interpreter environments (such as JVM). For developing wireless terminal applications, machine oriented programming languages such as C++ and Assembler would be much more efficient, especially when device resources are limited (as they still are). But on the other hand, this would render the application development and debugging more difficult, considering the variety of different devices, OSs and versions. With the additional emerging computation and information storage capabilities of wireless terminals, Java is likely to become a common platform for enabling fast provision of new applications and end-to-end solutions.

Based on the observed trade-off between mobility of information versus mobility of application computation (*see* section 3.1), this chapter concludes by placing the various application technologies previously described into the field spanned by the

two characteristic dimensions (*see* figure 3.26). The need of a *diagonal,* balanced migration from thin client to thick client can be observed. The step between currently existing mobile application solutions and future 3G solutions can be illustrated by the *migration gap*.

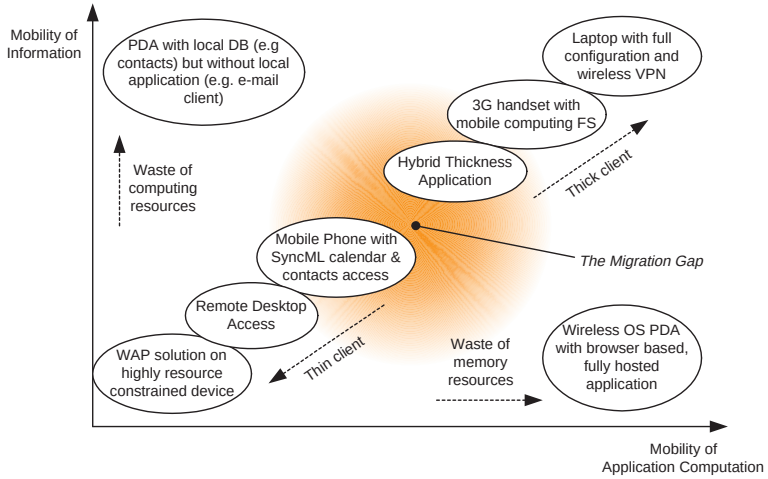

Figure 3.26: Information and application computation mobility balance

Chapter 4

Implications

"Don't focus on only delivering content to WAP phones, this is a strategy for the next 2 years. Focus on a pervasive wireless strategy providing content to many devices, this is a strategy for the next decade!"

—John Wright, Synamic Ltd., February 2000 [188]

4.1 Introduction

According to recent research in the area of 3G business opportunities, mobile operators are considered to become the *winners* of the emerging wireless data technologies [83], with cumulative total worldwide revenues from all 3G services expected to reach USD 322 billion by 2010 [225]. However, the question still is, what the new services offerings for achieving these revenue prospects will look like, and how currently available solutions can migrate in the right direction. It should be obvious that with emerging wireless terminal capabilities, increasing bandwidth, and packet-switched technology standards, the need for e.g. markup language based solutions (*see* section 3.2), especially those ones based on thin clients and limited communication possibilities (such as WAP), will decrease drastically.

In the new generation of mobile communication networks, the emerging wireless terminal can be considered as the most remarkable dimension. The additional ongoing convergence between the computer and telecommunication worlds implies radical changes in mobile services in comparison to the current way of thinking.

This chapter will examine and discuss the implications of the 3G service space and its applications on the current wireless industry, with a focus on middleware based solutions.

4.2 Mobile Middleware Concept Implications

4.2.1 Introduction

The current generation of mobile business solutions is based on an epoch, where mobile technologies are characterized by wireless phones, short messages, as well as very limited speed data transfer. The majority of current mobile businesses are

based on the initial idea of combining the growth of the Internet with emerging mobile communication facilities. The resulting first solutions included sending short messages from the Internet, as well as sending e-mail from a short message enabled mobile phone. Based on this background, the current business ideas, research, development, and knowledge are focused around the restrictions of mobile technologies, i.e. how to implement complicated solutions (e.g. e-mail) through simple devices (e.g. mobile phone with small screen, limited length messages, low data rates, and slow connection times). This area of solutions can be denoted as *middleware*, which can be characterized as the connection between mobile and computer communication.

Based on this, the question about the role of middleware with 3G and beyond arises, and one might ask if the entire concept will be needed at all, when mobile phones have converged with handheld computers, and mobile networks offer direct high speed packet-switched access to the Internet.

4.2.2 Balance between Terminal Application and Middleware

As increasing wireless terminal capabilities will bring wide opportunities for mobile applications development, it is an obvious consequence that the wireless terminal itself will take over some tasks which currently are being handled by middleware. One huge area of responsibility which probably will migrate entirely from middleware into the terminal in the future is the content representation, which will be integrated into the terminal application. In 3G, the balance between terminal applications and middleware can be summarized in one phrase: *The thicker the client, the thinner the middleware* (*see* figure 4.1).

Thus, the separation of content generation and representation (*see* figure 3.22) is an example for a strongly recommended way to take this balance into account.

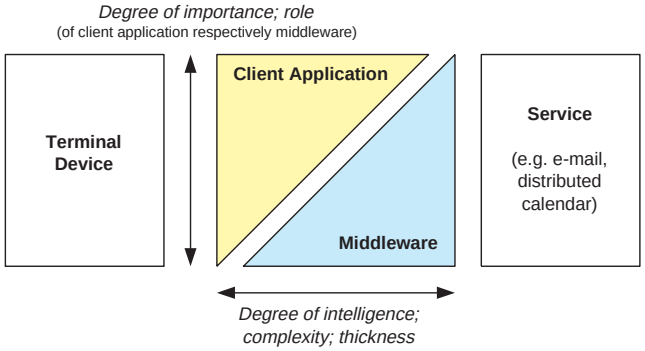

Figure 4.1: Balance between thin applications and thin middleware

4.2.3 Classes of Mobile Middleware

The set of mobile middleware solutions, can be roughly categorized into the following set of classes (*see* survey results in appendix D, and figure 4.2):

Enabling middleware. This class of middleware can be regarded as the most relevant for building mobile applications and solutions. It is wireless specific,

and implements high-level concepts such as location and subscriber profiles, service and device management, etc. This class can be further divided into *application-bound middleware*, such as content management and representation solutions, as well as *network-bound middleware*, which provide solutions that are difficult to reproduce outside the operator network.

Connectivity middleware. This class of middleware denotes the low-level components, which usually are located at operator premises, and therefore are impossible to implement elsewhere. Examples are gateways (e.g. for SMS or WAP) and performance enhancing proxies [183].

Front-end middleware. This generic class of middleware denotes solutions logically located near the client, such as e.g. content representation.

Back-end middleware. This generic class can be regarded as the opposite to front-end middleware, solving problems logically associated with back-end data (e.g. server access interfaces).

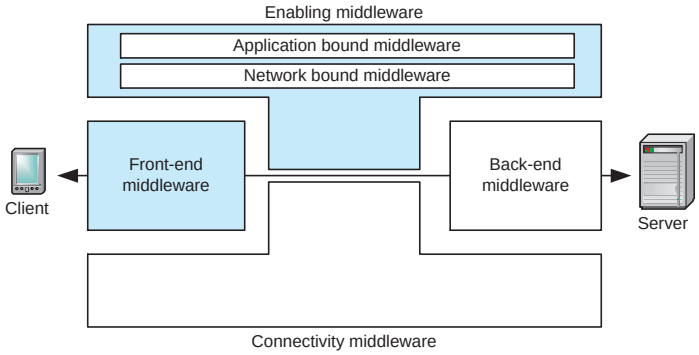

Figure 4.2: Different types of mobile middleware

4.2.4 Discussion

One could think of scenarios, where current types of front-end middleware enabling solutions will become completely obsolete under 3G, due to obvious substitutes. One fatal worst case for companies with a similar focus as Smartner currently (*see* section 4.7), would be a scenario where all mobile clients communicate over standardized content protocols such as XML or SyncML, and where representation and translation middleware would only would be a disturbing factor in the middle.

For end-to-end solutions, or more specifically, current *end-to-middle-to-end* solutions, the question is, whether substitute end-to-end solutions communicating e.g. over IP over GPRS directly, will eliminate middleware. Thus, the main requirements for end-to-end solutions and the greatest challenges to middleware can be addressed by pursuing a strategy that is based on the direct, standards-based integration of wireless channels to back-end systems. This approach enables provisioning of mobile services as a distinct, new channel for reaching customers, partners, and employees rather than as an extension of existing, e.g. web based, services.

On the other hand, there are some clear signs pointing towards middleware playing a clear role even in 3G and beyond (*see* section 4.4). One simple argument would be that a pocket terminal always will be smaller (in terms of screensize) than a desktop computer, hence a need for conversion. However, this type of front-end middleware, could migrate further to the client (application) side.

4.3 Operator Service Concept Implications

4.3.1 Introduction

Interpreting the mobile operator the provider of mobile middleware solutions and logics, the result is to question the entire middleware concept on an even higher level—the level of business models. Many mobile middleware solutions are targeted at operators, to enable the operator to increase average revenue per user (ARPU) by reselling value added service (VAS) offerings.

In Europe, over EUR 100 billion have been paid for UMTS licences, and remaining investments before the networks are built are esimated at around EUR 2 billion for each operator [41]. It will take some time for these mobile operators to reach the break-even point for their 3G investments, and already these investments are endangered by emerging alternative technologies (such as WLAN, *see* section 2.6.2).

Finally, investment in UMTS network technology is not the only 3G strategy for operators. The success of *i-mode* has given proof of the existence of superior revenue models for wireless data based on very simple mobile network technologies [49].

4.3.2 The Differentiation Dilemma

Until recently, mobile network operators have been primarily concernced with ensuring that their network has high population coverage and improving QoS.[1] Pricing of services is also an important issue for on-going development.[2] However, the operators and service providers cannot compete on price only. Value added services are envisioned, and mobile Internet access services (delivered e.g. using WAP Gateway) are thought to enable these alternative sources of revenue (*see* figure 4.3). This causes strong economic pressures on operators to transform from transport companies to service companies.[3] Operators will need a set of partnerships with the software industry, since it is likely that enablers like application software developers and wireless system integrators will capture a large part of the value in the race of service provisioning. Wireless datacommunication converging with Internet contents will be the basis of this transformation, which must be completed as a matter of urgency [188].[4] The variety of operators currently joining the WAP forum (*see* section 3.2.2) is an example of this urgency.[5]

[1] This is accomplished by monitoring dropped calls due to failed handover and other such metrics.

[2] An example for this movement is the explosion in pay-as-you-talk packages.

[3] If the operators fail to build up such services, the datacommunication model will dominate and simply *use* the operators as e.g. IP packet bearers. The question is, if there is anything to stop such a development in the longterm.

[4] On the other hand, if operators do provide VAS which *transform* content, then they have a legal liability for the content; If they do not, as a public carrier they are without such a liability.

[5] Operators challenges and difficulties have so far been mostly related to investment and timely infrastructure roll-out problems [61]. The aspect of difficulties in contributing to the value creation from the paradigm shift point of view has not been discussed enough yet.

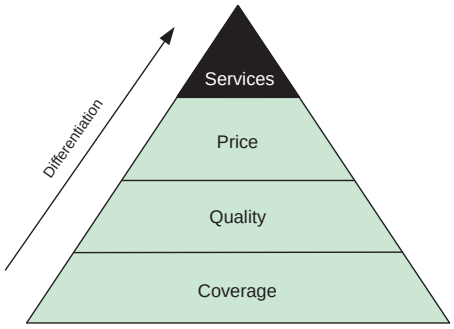

Figure 4.3: The need for operators to differentiate on services [188]

4.3.3 Critical Issues

In defining their 3G service portfolios for wireless data, mobile operators encounter several critical issues:

Obsolete Service Middleware. Certain front-end enabling middleware applications and solutions (e.g. hosted office services) adopted by operators might already be outdated and substituted by wireless terminal applications. For corporate solutions, the issue of opening the firewall to a huge telecommunications operator is somewhat tricky.[6]

Network Externalities. Especially compared to emerging alternative wireless technology solutions, mobile data services suffer from the following recursive problem [41]: If there are *no networks* ⇒ there are no services ⇒ there are no applications ⇒ there is no interest for specific terminals ⇒ there is no demand for the networks ⇒ there are *no networks*.[7]

Packet Carrier Only. An operator, ending up in providing an all-IP based network and transferring datagrams only, can be considered as a loser in the service differentiation race (*see* survey results in appendix D). Once the operator does not *understand* what is being transmitted, it is more or less too late to recover by differentiating on services *other* than QoS [155].

Wireless Customer Churn. Especially in the case of the operator becoming a packet data carrier, it will be difficult to keep long term wireless customers loyalty, as it will be easy to change to the cheapest service provider e.g. on a weekly or even packet by packet basis. Constantly losing and regaining customers is denoted as *churn* and can be regarded as a very harmful factor for creating reasonable ARPU.[8]

[6] On the other hand, as long as the operator hosted solutions implies the opening of a few ports only, the additional security risk cannot be regarded as more significant than e.g. a mail server. Efficient encryption algorithms can provide secure communication. The firewall issue is more of corporate, internal politics, than a technological problem.

[7] The term *network externalities* defines the economic effect of the amount, that one party is willing to pay for access to a network, being dependent on who or how many other parties are connected to it. This special case of the network externality effect, where the entire initialization is blocked, can also be denoted as *chicken-egg effect*.

[8] It is harmful unless the cost of regaining the customer is negligable.

Wireless Net Effect. In the convergence of mobile networks and the Internet, a tendency of migrating from proprietary systems to open standards-based platforms can be observed [205]. Even if a mobile network on the link layer might operate on e.g. WCDMA, it might be used through IP and TCP on the network and transport layer respectively, which makes it possible for everyone (not only the operator) to deploy new services. As occured with the Internet, the wireless *net* will not have any owner.[9]

4.4 The Migration Challenge

4.4.1 Introduction

In general, it is difficult for middleware developers to enter the application market and to reach any kind of platform dominance. One logical reason is their intermediate position in the mobile services value chain.[10] A total migration to applications of this kind can only be implemented in large companies or through intense partnering with 3G device vendors and terminal software providers.

As observed through the survey on the future of mobile middleware in a 3G context (*see* appendix D), the wireless industry and research communinity clearly points out that there still *will be* a considerable need for middleware solutions even in 3G and beyond. However, one clear tendency is, that the enabling middleware concepts might change somewhat from their current model.

4.4.2 Redefinition of Middleware Concept

Considering the wireless net effect (*see* section 4.3.3), the Internet itself is the best example for the need for application layer middleware. For instance, e-mail messages are not really transmitted on an *end-to-end* basis, mailservers are used *in the middle*. The question in the 3G middleware concept is, if such application layer middleware still has to be developed, or if such existing Internet middleware solutions can be directly applied. Alternatively, one starting point would be to ask, if there is any *missing* middleware.

An argument supporting the need for redefinition based on the wireless net effect on middleware in 3G, is given by the constraint of handsets' screensizes: As long as mobile pocket terminals have smaller screens than common desktop PCs, there will always be a need for middleware to connect and convert between mobile world and fixed world.

However, this type of front-end middleware, does not necessarily have to be located physically in the middle (e.g. at operator premises) anymore. A probable scenario is, that front-end middleware solutions will migrate into the terminal devices (*see* survey results in appendix D).

Finally, one aspect of future middleware is the tendency towards hidden, background functionality. For instance, a 3G handset user should not have to care about which SMS centre he sends his short message to in anticipation of conversion to an e-mail message—he should not see what happens, it should simply happen. One

[9] In general, any at least *two-way* communication creates a network effect drawing other users to the service [224].

[10] Another reason is, that providers of middleware for terminal devices, could be viewed as competitors by *their* customers.

quality of such end-to-end middleware is transparency, i.e. *good middleware is invisible middleware*.

One feasible approach for migrating to 3G could be denoted as *terminal application aware middleware*. Not simply adding *PDAs* as just another type of terminal devices to the existing set (e.g. WAP, SMS, WWW, etc.), but constructing the middleware in such a way, that it allows stepwise migration of application functionality and intelligence (not only content) into the terminal application (*see* section 3.7). This should be considered as an important 3G compatible approach. This approach could be summarized into one phrase: *Think terminal applications, while doing middleware.*

4.4.3 Critical Success Factors for Service Providers

For providers of wireless middleware based hosted services, critical success factors within the migration challenge are [202, 63, 205]:

Pricing Issues. Providers of middleware based hosted services have to apply billing based on usage and transactions.[11] Economies of scale come into play when the pricing model can be characterized by two factors: *high volume* and *low price*. This enhances their business case and increases revenue.

Differentiated Services. Providers have to offer a portfolio of differentiated and customized services with a clear value proposition. In the long term, this increases loyalty, decreases churn, and thus increases revenue.

Ensure Quality. The economies of scale come into play if and only if truly scalable software solutions are the basis for the system. Commitment to software technologies following these principles (e.g. J2EE) is essential. Furthermore, platform awareness, such as terminal application aware middleware (*see* section 4.4.2), are critical future compatibility issues.

Partnerships in the Value Chain. If providers of middleware based hosted services manage to establish *complementary* partnerships in the value chain and commit themselves to revenue sharing, they can turn out to be leaders in a community. This community could consist of international alliances, technologies, and services, ensuring the availability of required components needed to provide winning integrated end-to-end solutions (*see* figure 4.4).[12] In order to succeed in the service offering, the providers must hone their ability to identify potential partners quickly, determine the appropriate level of intimacy for the relationship, and secure the corresponding level of commitment [68]. Partnering is especially crucial for small, emerging companies within this industry [123].

[11] On the other hand, especially in the consumer segment, the user wants to perceive flatrate pricing.

[12] The Internet centric approach, as depicted in figure 4.4, is based on the idea of dividing each market segment into its subsegments (for simplicity a binary tree). In this model, a leadership position (e.g. platform dominance) can be illustrated by the control of one horizontal level. If an independence from and an effect on the underlying platform is desired, partnerships or *attacks* have to be directed to that horizontal level. If influence on the existing subset of platforms is desired, partnerships have to be directed in the other direction.

As an example, given the vertical level of wireless OSs, is if a provider can compete by its own wireless OS product, it can lead that level. If this does not seem reasonable, the alternative could be partnering with existing wireless OS manufacturers, thus moving the company's focus to the subset level of terminal applications for that OS. On the contrary, if an attack is directed to the underlying level, e.g. by influencing Nokia to move away from EPOC as a platform, the existing OS platform can be influenced.

Protect Investments. Commitments to 3G as a technology (e.g. UMTS investments) have to be protected by combining 3G technology with the wireless net effect. Providers should enable and evolve end-to-end solutions based on open standards.

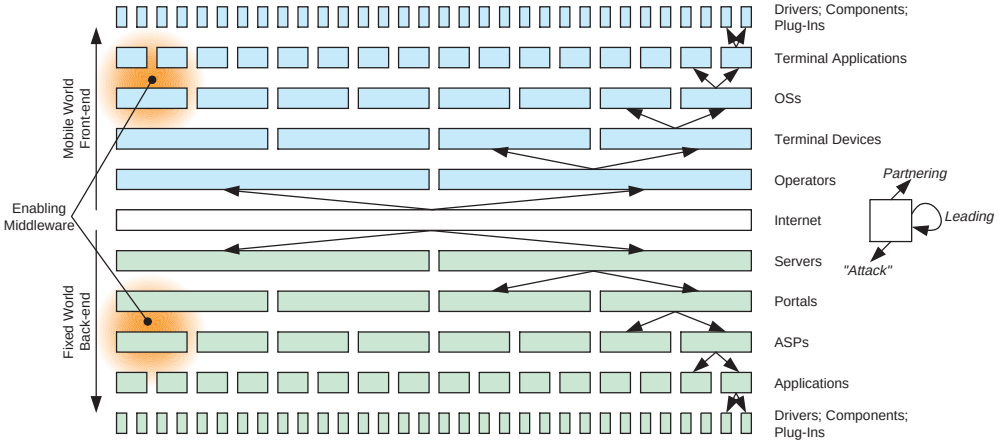

Figure 4.4: Partnering vs. leading in the value chain

4.5 Unleashing the Killer Cocktail

4.5.1 Introduction

As already addressed in section 3.9, the choice of the right type of application when converging from currently existing mobile application solutions to future 3G solutions, rises a lot of questions (i.e., the migration gap, *see* figure 3.26). The choice of an appropriate business segment is essential, where an hybrid between proprietary operator service and Internet based applications makes sense and provides added value to the customer.

It should be clear that a successful mobile business end-to-end solution cannot be based on a stand-alone application. Instead, a set of applications and influence at strategically positioned points is the right recipe. Therefore this approach is being denoted as the *killer cocktail* [78].

4.5.2 Value Chain Dominance

A business model oriented approach for finding opportunities for wireless services is based on analyzing the provision of end-to-end solutions from a value chain perspective. As it can be seen as very difficult to define a single value chain, covering and explaining the entire industry logic of mobile data services [59], recent studies have defined entire *value maps* attempting to cover the full complexity of mobile data services [144]. For mobile business solutions, the analysis of the *information* value chain from an operator perspective, might give an insight about potential dominance, partners, and rivals (*see* figure 4.5). Based on this approach, the application cocktail for

operator-hosted end-to-end solutions should be able to benefit from leading operators' areas of strength within the value chain.

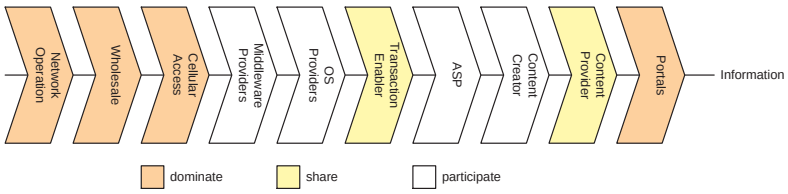

Figure 4.5: Information value chain for leading operators

4.5.3 Strategic Product Differentiation

For firms operating in uncertain markets, it is difficult to decide whether to focus on one core product, or maintain a variety of products [201, 97]. This is especially the case for small firms with scarce resources. With such a constellation, the management team must allocate resources to either creating a set of tailored products, or developing efficient scalability for producing a few products [151]. Product variety can be an important strategic variable—the product strategy must be located in an appropriate position between the following two trade-offs [201]:

High Product Variety. Variety becomes less valuable as the total number of products on the market increases. On the other hand, it increases in value as uncertainty renders accurate demand prediction difficult. In uncertain markets, product variety allows the firm to better meet the demands of heterogeneous consumers. Multiple product offerings can be rationally used to gather information about the distribution of consumer preferences. Selection amongst these offerings can provide an effective means of aligning the firm's product line with demand. Furthermore, multiple product offerings can increase sales [175] and allow the charging of higher prices for each product [176].

Lack of Product Variety. Above all, the lack of variety allows a vendor to dramatically take advantage of economies of scale. Lack of variety can be motivated by cost savings, since broad product lines can require more overhead [95], and coordination of the variety becomes increasingly difficult [129].

Based on this observation, the allocation of research and development (R&D) resources for enabling middleware providers has to be defined with care. It would be unwise to start developing terminal applications with full effort. However, the concept of remaining a provider of enabling hosted middleware might be outdated very soon. Therefore, the *integrated approach*, based on the definition of a generic mobile application portal (MAP) supporting terminal application awareness seems rational and reasonable (*see* section 4.7.4).

4.5.4 Strategic Orientation within the 3G Service Space

Successful applications for the mobile Internet will be ones that take advantage of the unique characteristics of the mobile environment [182]. Based on the definition

given in section 2.1, the 3G service space model could be applied to characterize applications from an added value perspective. Based on our model, the technological changes with 3G and beyond can be described with three dimensions. Thus, a very simple approach for determining the 3G added value (3GAV) and competitive advantage of a given mobile application solution, would be based on considering its extent in each of the three dimensions, i.e. bandwidth (BW), packet switching (PS), and terminal capabilities (TC).

Based on the n-dimensional linear space character of our abstraction,[13] we observe the extent of utilization within a discrete interval $[0, ..., 4]$, where zero means no extent at all, and four means full utilization. We determine values for BW, PS, and TC, and denote them by x_1, x_2, and x_3. For each dimension value x_i, we define a constant weight value a_i, which denotes the the importance of that dimension.[14] Now, the *Euclidean norm* can be used for calculating an indicative, nominal magnitude for 3GAV:

$$\texttt{3GAV} = \|a_i \cdot x_i\|_E = \sqrt{\sum_{i=1}^{n} (a_i \cdot x_i)^2} = \sqrt{\sum_{i=1}^{n} {x_i}^2} = \|x_i\|_E \tag{4.1}$$

This method provides a very simple approach for ranking application solutions by 3G added value. In our example (*see* table 4.1), we divided each value for 3GAV with the maximal possible value for the norm, resulting in a relative value.

$$\textit{Relative Size} = \frac{\texttt{3GAV}}{max\{\|x_i\|_E\}} = \frac{\texttt{3GAV}}{\sqrt{3 \cdot 4^2}} \tag{4.2}$$

Above all, such a ranking approach clearly illustrates that taking advantage of one 3G service space dimension *only*, is not the right approach for unleashing the killer cocktail.[15]

4.6 Competitive Advantage and Value Creation

4.6.1 Introduction

This section will address a set of crucial topics that should be considered when designing a technology roadmap for new technologies, such as 3G. A brief recommendation on how to build up a value proposition, and how *not* to do, will be presented.

4.6.2 Distorted Business Models

There exists a broad range of generic research on value creation in highly networked emerging markets, and generic business model definitions for value creations in information services have been attempted [39]. However, after several years of *Internet hype*, research has finally proven that adopting new technologies alone does not provide any concrete or even longterm competitive advantage. Instead, new technologies should be regarded as complements to traditional ways of competing [179, 169].

[13] The choice of a linear vector space motivated by simplicity.

[14] These weight values could e.g. be based on customer preferences. For simplicity, we use $a_i = 1 \ \forall i$, as well as equal ranges for $x_i \ \forall i$.

[15] An additional, fourth dimension for the extend of utilization will be given by positioning technologies and the possibility of building location dependent services.

These theories apply to mobile technologies as well, since mobility can make life easier in certain ways (e.g. efficient business solutions), but does not provide value *as is*.

Value and competitive advantage are very closely related factors. Only the creation of real value to the customers, brings longterm competitive advantage to the producer—as well as success. Externally observed facts and figures, such as e.g. *m-business value*, *e-value*, etc. are very speculative variables. As markets worldwide have shown during the last month, after all, even for mobile solution providers, the economic value consists of the gap between price and cost. Generating revenue, reducing expenses, or simply doing something useful by deploying mobile technologies, does not in itself give sufficient evidence of value having been created (e.g. selling ringing tones and operator logos).

4.6.3 Usage is Value

The only thing that creates value in mobile technologies is their *use* [244, 179]. Mobile solution providers might succeed for some time, independently from their proposed solutions being useful or not. During the growth and testing phase of emerging technologies and their new services, even erroneous business models and useless products might succeed. On the other hand, if the use of a new product does not cause cost savings and return on investment (ROI) in the long term, the amount of new product purchases will decrease, and the business model collapses.

Current 2G and 2.5G networks still represent large constraints in *real* value creation for mobile data services [59]. Migrating towards the convergence, a provider of mobile solutions must have a clear picture of how mobile solutions differ from their corresponding *classic* solutions, and design distinguished strategies accordingly. Within the context of mobile business solutions, 3G gives an excellent opportunity for this redefinition. A vision of the *mobile office* should be created based on possible future usage scenarios, which in turn are based on requirements of current standard business applications (*see* appendix E).

4.6.4 Tangible Benefits

The need for mobile business solutions is emerging in all areas, where connecting people, products, or equipment to the business system can yield a competitive advantage [221].

Examples of features that will make such mobile applications extremely popular are [188]:

- being able to connect to a corporate application in less time than it takes to log on from a laptop,
- being able to carry the device in a shirt pocket,
- being able to connect in most countries in the world without the need for a bag of adapters,
- being able to browse through e-mail.

This technology will allow individuals to become mobile and at the same time have access to shared corporate systems.

There are large opportunities for technology and service providers who can reduce complexity and cost. Especially in the mobile business enterprise solutions market, selecting the right lead customers and partners is crucial. However, both need to demonstrate tangible benefits with focused value propositions [144], otherwise the proprietary product idea will be encapsulated into something less useful.

4.7 Case: Smartner

4.7.1 Introduction

Smartner Information Systems [22, 200], founded in 1999, is a software and professional service company focusing on mobile business services. Smartner's current service offering includes competence and tools for operators and application service providers, who build and offer mobile services and solutions for enterprises. The sales of Smartner products happen through operators, who emphasize differentiation (*see* section 4.3.2), and thus seek to increase ARPU and decrease churn (*see* sections 4.3.3 and 4.4.3). The target market are small and medium-sized enterprises (SMEs). SMEs usually do not have resources to implement their own mobile Intranet platform, and cannot afford to buy a customized solution from system integrators, thus they have potential interest in purchasing such solutions from an operator as value added services.

4.7.2 Technology Solution

The solution proposed by Smartner is based on a mobile applications platform (MAP), the *Smartner Engine*, offering a scalable, J2EE technology based generic platform for easier deployment of new business applications. The platform supports independence of terminal devices by applying an internal separation of content generation and representation (*see* section 3.2.5). Currently supported terminal interfaces are WAP, WWW, and SMS (*see* figure 4.6).

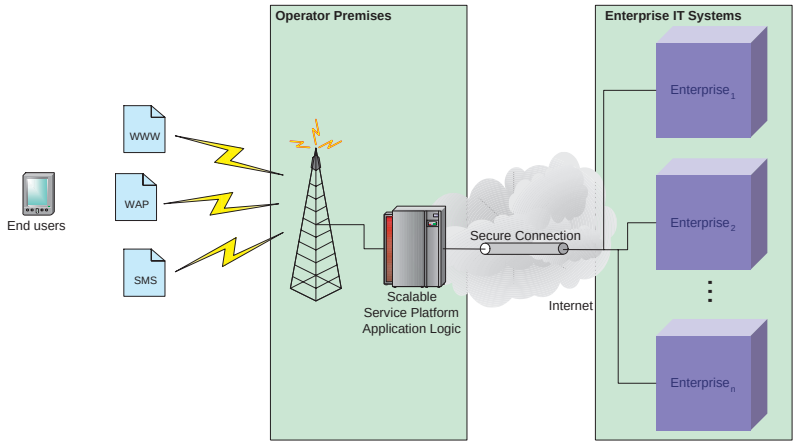

Figure 4.6: Smartner's concept

The Smartner Engine is partly located at the service provider (operator) and partly at an enterprise customer premises, and establishes a secure connection to the corporate Intranet. There exists a set of application *Extenders* already implemented by Smartner, such as e.g. *E-mail Extender*, *Calendar Extender*, and *Database Extender*, all of which establish access to existing enterprise applications.

4.7.3 The Migration Challenge

Being a developer of front-end enabling middleware, with a business idea based on the convergence between mobile communication and Internet technologies, Smartner has to redefine its solution concept in a 3G context on a very fundamental level—at the peril of the entire company's existence. The possible change in the role of the mobile middleware concept (*see* section 4.2), will directly affect Smartner's business model.

Based on the question whether Smartner's current knowledge, products, and development will be applicable under 3G, the following case analysis summarizes the challenge [112]:

Worst Case. Current mobile phones will evolve to mobile computing stations, with a complete OS and application set. All applications will run locally; integrated communication happens over open standards such as XML; there is no need for corporate application extending based on markup language solutions.

⇒ *There will not be any use of Smartner's current knowledge and solutions in 3G and beyond.*

Average Case. Mobile devices will become more complex and will allow more advanced, locally running applications. However, enabling middleware is still needed, since no platformwide format standard really dominates the scene, and end-to-end solutions suffer from different vendors' compatibility issues.

⇒ *There will be use of Smartner's current knowledge and solutions, if the company manages to develop new knowledge in the area terminal applications, and terminal application aware middleware.*

Best Case. The convergence as described previously never happens to that extent; mobile terminal devices remain very limited even in the future, and the current types of enabling middleware will be needed even in the future.

⇒ *There will be direct use of Smartner's current knowledge and solutions even in 3G and beyond.*

The *best case* seems very improbable, since there currently are not really signs of a stagnation of the convergence development. On the other hand, the *worst case* seems very unlike as well, based on heterogenous industry and technology developments (*see* survey results in appendix D). According to a currently very likely scenario, as described by the *average case*, Smartner should redefine its middleware concept in an application aware context (*see* section 4.7.4).

4.7.4 Redefinition of Middleware Concept

In Smartner's case, the definition of application aware middleware should start by questioning the *integrated approach*.

On the one hand, this approach, characterized by the standards-based integration of wireless channels to back-end systems, enables Smartner to fully leverage mobile technology as a distinct, generic channel for reaching enterprise systems, rather than extending applications one by one. On the other hand, it remains questionable and a dangerous assumption to integrate a 3G handset simply as another data output format in the platform.

The hybrid thickness client application concept could be a good starting point for redefining the platform. As 3G terminal applications will increasingly require separation of content generation and representation between middleware and terminal device, the platform should for instance support push technologies, and native XML content transfer in the future. The latter feature would imply the complete data set to be sent to the terminal client instead of small portions as a reaction to a request, and could allow a hybrid thickness client application to autonomously represent the data (*see* section 3.7.7).

Based on this concept, current solutions, such as the Office Extenders, could migrate to partly stand-alone applications, with the full data set remaining on the corporate server. In the case of the Database Extender, one could think of building a generic database end-to-end *skeleton*, allowing easy construction of customized database applications (*see* section 3.7.6).

In the longer term, integration of autonomous background data retrieval, such as proposed by synchronization concepts (*see* section 3.3) and mobile computing file system issues (*see* section 3.5) should be considered. Especially in this context, the migration of the middleware into the terminal device (*see* survey results in appendix D), i.e. the development of locally running background *drivers* should be taken into account.

4.7.5 Five Forces of Competition

The position of Smartner's competitive advantage based on current knowledge and solutions, can be summarized by applying the *five forces* model of competition onto the migration challenge context [177, 179].

Threat of substitute products. Mobile middleware will be replaced by locally running terminal applications.

Threat of new entrants. Software giants, such as Microsoft, are expanding their business into mobile middleware solutions and wireless terminal applications, and are competing on quality, price, as well as compatibility with existing products.

Bargaining power of customers. Mobile operators encounter financial difficulties, and as a reaction cut investments in subcontractor solutions. Instead of outsourcing, operators might allocate resources on internal development of such mobile middleware platforms. As observed previously (*see* section 4.6.3), it is the end user who decides upon mobile services, and thus Smartner's success. Compared to a huge operator, Smartner's possibilities to affect the end user are neglectably small.

Competitor rivalry. Currently, there are only a few rivals competing in the same segment as Smartner (SME through operators, *see* section 4.7.1). However, changes in technology solutions might also imply changes in segment, where new battlefields will be entered. Additionally, operators will probably become strong competitors in providing services [155].

Bargaining power of suppliers. Currently, Smartner is not really dependent upon suppliers and subcontractors. Development is mostly carried out internally, and open standard tools are used.

4.7.6 SWOT Analysis

For expressing the analysis of Smartner's current middleware position in combination with the migration challenge, we summarize Smartner's *3G technology* SWOT analysis[16] as in table 4.2.[17]

Entering the market for terminal applications is an unevitable action for a company like Smartner. Preparing itself for 3G technologies in an early stage can help increase the market share in a short time. The company already has an understanding of the need for mobile end users, thus the movement towards terminal solutions should not be a barrier, and does not have to be started from scratch. An optimal case would be to integrate their own solutions into device manufacturers' equipment in an early stage; this could be reached through effective partnering. Considering this issue, it will however be difficult to keep the platform independence for every single application.

The current application platform and product roadmap should definitely be kept. However, the additional preparation in 3G can bring Smartner an early mover advantage, which will pay off when the majority of 3G handsets hits the market.

4.8 Summary

Applying the six fundamental principles of strategic positioning to Smartner in the 3G application context, the implications can be summarized as follows [178]:

The right goal. All subgoals and roadmaps that are part of the product strategy should be in harmony with the main goal—superior long-term return on investment. In the case of Smartner, the main goal should correspond to fulfilling the needs of mobile business users in a competitive and fruitful manner. The technology aspect may not be a dominant factor in the provisioning of end-to-end solutions.

Value proposition. A company's strategy must enable to deliver a set of benefits, different from those that competitors offer. Smartner's value will be determined by a particular set of uses and customers—the use of business applications and solutions.

Distinctive value chain. The control of a distintive value chain or network would form a theoretically, superior prerequisite for reaching maximal competitive advantage. Partnership options have to be considered, but ones own vision

[16] *SWOT* stands for *Strengths*, *Weaknesses*, *Opportunities*, and *Threats*.
[17] For a more differentiated SWOT analysis, *see* appendix E.

	Extent of utilization				
	BW x_1	PS x_2	TC x_3	3GAV $\|x_i\|_E$	*Relative Size*
Application					
Local e-mail application with autonomous background synchronization and multimedia contents	4	4	4	6.9	100 %
Remote Desktop Access	4	4	3	6.4	92 %
Mobile Computing File System	3	4	4	6.4	92 %
Local e-mail application with IMAP and multimedia contents	4	0	4	5.7	82 %
Local e-mail application with autonomous background synchronization and text contents	2	4	2	4.9	71 %
Local e-mail application with IMAP and text contents	2	0	2	2.8	41 %
Hosted ML based e-mail application over GPRS or UMTS	2	0	1	2.2	32 %
⋮					⋮

Table 4.1: Using the 3G service space as a strategic measure

Strengths	**Weaknesses**
• Strong knowledge of software development, especially enabling middleware solutions • Experience in programming languages and techniques common in mobile technologies (e.g. J2EE) • Partnerships with device manufacturers • Vision of and experience concerning mobile user needs	• No real significant experience in the area of terminal application development • Strong knowledge of middleware alone is probably not sufficient for 3G
Opportunities	**Threats**
• Middleware knowledge and solutions can be reused in terminal applications (e.g. J2EE ⇒ J2ME) • Smartner gains an early mover advantage allowing a relatively big market share, and will be able to grow • Smartner's product, integrated with device manufacturers equipment	• Superiority of knowledge and solutions of device manufacturers compared to Smartner in the area of terminal applications • Software giants develop their de facto standard applications in *mobile versions*

Table 4.2: Smartner's general 3G technology SWOT

should remain. The advantage requires tasks to be solved in a different way than competitors do, or similar tasks solved in a different way. Imitating and downscaling existing solutions will obviously cause difficulties in creating an advantage.

Trade-offs. A company must abandon some product features, services, or activities in order to be unique with respect to other features. In the uncertain market of mobile technologies, radical changes might have to be made if the situation requires it. This should be considered with regard to the integrated approach (*see* section 4.5.3), as in the Smartner case (*see* section 4.7). The balance between middleware-centric and independent terminal application solutions is another example (*see* section 4.2.2).

Compatibility. One critical factor is how all the elements of a company's activities fit together. After all, the entire set of activities must be mutually reinforcing. This should be considered when migrating towards 3G, which should not be a sole responsibility of R&D activities.

Continuity. *Reinvention* or *reorientation* is usually a sign of poor strategic thinking. The decision on the commitment to 3G should be well-defined, considered, and communicated as a long term goal. A strategy is not a strategy if it is redefined too often.

Chapter 5

Conclusion

"The best way to have a good idea is to have a lot of ideas."

—Linus Pauling,
Nobel prize award chemistry 1954, peace 1962 [125]

5.1 Introduction

Paving the way for a convergence strategy, this study began by examining 3G technologies, as well as future and alternative approaches (chapter 2). We observed that the changes brought by these new technologies can be observed by the three dimensions of the 3G service space (*see* section 2.1), from which the dimension of increasing terminal capabilities was considered as the most significant.

Based on this foundation, we examined a variety of applications for building mobile solutions, and related them to the middleware issue (chapter 3). Current and future markup language based solutions were presented, and their feasability in advanced terminal devices was discussed (*see* section 3.2). Over-the-air synchronization, virtual private networking, and mobile computing file systems were introduced and presented and discussed as compelling possibilities for business applications (*see* sections 3.3 to 3.5). Remote desktop access was presented as an example for future hosted application services (*see* section 3.6). Wireless terminal applications were introduced partly as a completely new, stand-alone concept, partly as a valuable addition to middleware platforms (*see* section 3.7). Among a set of terminal application approaches, the hybrid thickness client application concept was discussed as a feasible way for migrating to 3G. The concept was illustrated by an example application implementation (*see* section 3.7.8, appendices B and C). Finally, the MExE approach was introduced as a feasible standardization approach among the variety of current and future mobile application technologies (*see* section 3.8).

Having studied and discussed 3G technologies and application possibilities, a set of implications on current solutions and business models was derived (chapter 4). The future of mobile middleware solutions and operator-hosted services in general was critically discussed (*see* sections 4.2 and 4.3). Challenges and strategies for creating valueable solutions and migrating to an application centric world were introduced and applied on the concrete case Smartner (*see* section 4.7). The recommendations of each application type (chapter 3) were summarized and finalized.

5.2 Key Findings

The emerging technologies of 3G and other future networks and terminals brings a broad variety of possibilities and challenges for application development. On the other hand, there also certain risks involved. Current solutions might loose their importance, and platform dominances might be affected by the computer communication and software industry. Mobile applications will migrate from thin client/thick server solutions to more advanced distributed systems and mobile computing solutions. Middleware will certainly exist in the near future, although the role of application-level enabling middleware is still unclear. The industry has clear goals and visions based on middleware supporting platform independent and across-vendor systems (*see* section 3.8 and appendix D).

Middleware will be characterized by its invisibilty. The less visible and the more autonomously the middleware acts in the background, the better its quality.

As developments of mobile applications converge towards wireless Internet access and IP-based solutions, mobile operators will discover difficulties in differentiating based on value added services (such as hosted application services, messaging etc.). As long as the operator cannot provide solutions that add substantial value for competing against IP-based applications, the operator discovers the risk of becoming a packet data carrier only.

On the other hand, there is one main industrial trend—mainly independent of any development of mobile technologies: outsourcing. Instead of tailoring own solutions, or purchasing customized installations from system integrators, especially smaller enterprises tend to invest in hosted, application service provider (ASP) solutions [222, 223]. The ASP concept for wireless solutions (WASP) might differ slightly, but the business model can be regarded as similar [73].

5.3 Strategic Imperatives

For middleware focused application and solution provider such as Smartner (*see* section 4.7), we conclude our study with the following set of primary strategic imperatives for migrating towards 3G and beyond:

Focus on wireless OSs. Allocate R&D resources on wireless OSs, future application platforms, and software development issues for mobile terminal applications (*see* section 2.5).

Understand role of middleware. Strive towards maintaining a clear and objective picture of the role and shape of mobile middleware in the future. Understand the implications of *front-end* and *enabling* middleware (*see* section 4.2.3) and consider alternative middleware solutions (e.g. SyncML, *see* section 3.3.4).

Profit from middleware technology. Seek 3G solutions that benefit from middleware knowledge and technology. Try to combine and adapt the middleware approach in an application-centric context,[1] but do not fully commit to middleware as the only way to do it. Depending on how future application markets develop, a quantum shift, i.e. an entire change of focus, might be necessary.

[1] For instance, the separation of content generation and representation (*see* figure 3.22), is strongly recommended.

Balance R&D towards applications. Prepare to balance the development of applications and solutions towards wireless teminal applications. This should be done in two separate modes, one middleware-centric and one middleware-independent approach.

Reallocate professional force appropriately. The migration towards 3G cannot be solved by R&D only. Especially considering wireless terminal applications, the study of vertical businesses solutions implies a need for industry knowledge and system integration into other products and solutions (*see* section 3.7.5). A considerable portion of the technology resources will have to be allocated onto external consulting services. Another crucial aspect is partnering (*see* sections 4.4, 4.4.3, and table 5.1).

Acquire and manage 3G knowledge. A permanent study of technological changes and advances in technology and business models is strongly recommended. The focus should not only be on mobile operator and device vendor dominated technologies, but also alternative approaches.

End-to-end solutions. Whether middleware will be the key for successful provisioning of mobile applications or not, the long term focus should be on creating competitive advantage by robust end-to-end solutions instead of singular parts of an application package.

Markets and partnerships with operators and device vendors. Allocate marketing and leads generation resources aggressively towards mobile operators and 3G handset vendors. Profile and distinguish as a serious enabler of applications and solutions. Consider partnerships within the value chain, supporting the provisioning of end-to-end solutions.

Aspect	Description
Concentrating resources	
Converging	Building consensus on strategic goals
Focusing	Specifying precise improvement goals
Targeting	Emphasizing high-value activities
Accumulating resources	
Learning	Fully using the brain of every employee
Borrowing	Accessing resources of partners
Complementing resources	
Blending	Combining skills in new ways
Balancing	Securing critical complementary assets
Conserving resources	
Recycling	Reusing skills and resources
Co-opting	Finding common cause with others
Protecting	Shielding resources from competitors
Recovering resources	
Expediting	Minimizing time to payback

Table 5.1: Achieving resource leverage [92]

5.4 Recommendation

The recommendations given throughout chapter 3 are focused and applied on one respective technology segment only, and therefore have to be summarized into a strategic entirety. Given the limited resources of a company like Smartner, an appropriate resource allocation leverage will have to be well defined. Early decisions about outsourcing and partnerships are crucial in order to avoid the risk of ignoring core competencies [92]. This thesis recommends any product roadmap to be strongly vision-oriented, i.e. any R&D steps and operational actions should be based on a clear vision definition. According to this approach, a concrete product roadmap with strategic development steps can be summarized into a table (*see* table 5.2).

5.5 Discussion

This thesis should represent an objective contribution to the decision making of a middleware-based mobile applications provider in building a strategy for 3G. We defined such a provider as a company *similar* to the Smartner case (*see* section 4.7), which especially refers to the size of the company, the operator-hosted provisioning concept, and the integrated approach (*see* section 4.5.3). Interpretations applied on other companies or solutions have to be handled with care.

As mentioned above, the reliability of this thesis is in accordance to the technology-based approach (*see* section 1.5). Considering the fact that the 3G initiative came from device manufacturers, not from operators, it is crucial to constantly monitor upcoming mobile devices and their new functionalities. As the adoption of mobile data services still is difficult to predict (what will be the killer cocktail?), future applications and enabling technologies are most probable to be dominated by the software industry, which allows the definition of a set of disjunctive scenarios (*see* appendix E).[2] However, if *none* of the envisioned scenarios turned out to become reality at least to a certain extent, the relevance of this thesis would remain questionable. Furthermore, if current middleware-based solutions turned out to represent dominant mobile applications even in the future, the contribution of this thesis would be neglectably small as well.

5.6 Further Research

A market oriented approach, based upon a profound study of prospective 3G market shares of wireless OSs, devices and platforms would be an interesting complement to this study, and could help estimate qualitative probabilities for the different dominance scenarios (appendix E). On the other hand, despite software and hardware manufacturers' dominances, the significance of end user preferences may be unpredictable.

From the technology point of view, completely different access methods (not necessarily wireless) and new areas of convergence should be studied [138]. This includes emerging new terminal technologies (such as Digital TV) for *mobile* (but not

[2] We believe in the goal of not predicting *the* future, but of imagining *a* future made possible by changes in technology, life style, work style, etc. [92]

Time	Technology Battlefield	Development Steps		
		⇐ **3. Operational**	⇐ **2. R&D Mgmt**	⇐ **1. Vision Creation**
2001	GPRS rollout	Start to broadcast clear corporate 3G vision	Develop SyncML support for current solutions	Define GPRS added value for WAP solutions (e.g. WAP push, integrated telephony, SyncML)
	WAP over GPRS rollout	Offer SyncML solutions	Support and *benefit* from new WAP 1.2 and GPRS features	Define shape and functionality of wireless terminal applications (e.g. Java hybrid clients)
2002	Variety of combined PocketPC- and EPOC-smartphones enters the market	Sell solutions based on new WAP 1.2 and GPRS feature value proposition; focus on vertical business segments and according partnerships	Incorporate wireless teminal applications into middleware framework (as far as possible, e.g. hybrid clients); Initiate new middleware solution types (e.g. remote desktop access solutions)	Define WAP 2.0 added value for new solutions
	First WAP 2.0 phones	Initiate partnerships with terminal manufacturers and wireless OS vendors; consider (partly) outsourcing of terminal applications development	Support and *benefit* from new WAP 2.0 features; consider new development areas (e.g. integrated VPN solutions)	Define implications of short range technologies (e.g. Bluetooth) and location-dependent services on mobile business solutions
	Java support for mass market (e.g. Nokia 50 million terminals)	Sell solutions based on new WAP 2.0 feature value proposition; offer value proposition based on Java end-to-end capabilities	Study application possibilities for Bluetooth and location-dependent information	Revised study of development of emerging terminal software platforms (e.g. dominance scenarios); define role of middleware and integration approach
	Widespread J2ME (e.g. Motorola support on *all* devices); Bluetooth as common platform	Offer J2ME applications in product portfolio (e.g. hybrid clients, both-side-stub solution)	Initiate development of full client terminal applications (e.g. mobile computing file systems)	Profound strategy revision (i.e. what is a *mobile business application*?)
2003	No. of wireless web users will exceed no. of wireline Internet users	Consider outsourcing and partnering in full client terminal development	Develop location-dependent business services	Consider convergence as completed (i.e. mobile Internet = wireline Internet)
	Location-dependent services will become part of everyday life	Offer location-dependent business application services	Develop (parts of) full client application suite	Consider other segments (e.g. consumers?)
	Worldwide Java adoption (e.g. Nokia 100 million terminals)	Offer full client applications and end-to-end solutions in product portfolio		
	Wireless OS handsets have become cheaper (i.e. not only high-end items)			⋮

Table 5.2: Strategic roadmap and development steps

necessarily *wireless*) access of business applications.[3] As for 3G and future applications, the usability issue for small, mobile handsets is an interesting field of research. Increasing terminal capabilities do not only imply higher screen resolutions, processing power, and memory size. The variety of sensory devices is increasing as well, and could be of interest for mobile applications, where common input devices (such as mouse pointer, `QWERTY`-keyboard, numerical pad, etc.) should not be taken for granted [157]. *Intelligent interfaces* and *wearable computing* are research areas with high potential in combination with future mobile devices.

Emerging positioning and location determination technologies of future networks will open the market for location-dependent services, introducing a new dimension of ubiquitous applications, where operators might have better possibilities to participate in value creation [134, 168]. Location-dependent computation and communication can be incorporated into end-to-end solutions for providing new, higher-level mobility related services. However, the successful deployment of any of these techniques requires a revised view of distributed systems for mobile computing [229]. These issues have to be studied with care [180].

5.7 Vision

Few companies have explicitly defined their corporate vision for technology [66]. Among technical and strategic product roadmap recommendations, this thesis can be regarded as a contribution to the technology vision.

The core ideology can be motivated by the vision of future mobile business applications, enabling unlimited access to office data and applications from any location in any situation. Mobile business solutions should enable working from outside the office without restrictions, i.e. in the same way as from the office desk. But this is not the case yet and there remains some work ahead. The R&D task should not only be to build new services, motivated by mobility hype (*see* section 4.6.2), the focus should remain on making life easier by facilitating mobility.

Some more concrete elements and usage scenarios determining the value of a next generation wireless services will be:

- The user has unlimited access to desktop data.
- Actions from a mobile device appear as if they originate from desktop.
- The mobile device contains an up-to-date subset of desktop data.
- The mobile device application is local, but still transparent to the desktop application (e.g. in the sense of usability).
- A programmable mobile device allows the (advanced) users to customize and personalize their mobile application suite, based on mobile device specific user interface features (e.g. command shortcuts, vibration signal patterns, etc.).

[3] The reader should notice that the use of the term *mobile* in this context is not contradictory to our initial definition (*see* section 1.6), where *mobile* was declared as a partly separate set from *wireless*.

5.8 Summary

Companies finding themselves in a similar situation as Smartner will have to enter the application market sooner or later anyway. The primary question in strategic product planning should be based on the focus on terminal applications. This should happen before being too late, and while still being one of the first.

Despite the current financial and other analyst community's pessimistic opinion on 3G (especially in association with UMTS investments), it should be clear that it in the long run will be hard to avoid any next generation. Anyone who has ever used a laptop with a 9600 bps GSM connection (*see* section 3.4) or tried to access unstable WAP solutions (*see* section 3.2.2), will—regardless of the breakthrough in mobile business solutions or not—sooner or later realize that *this* cannot be the end of the story.

Appendix A

Remote Desktop Traffic Analysis

A.1 Introduction

This chapter contains the analysis and results of the mobility potential measurement series explained in section 3.6. Any conclusions and further discussion are also mentioned in that section.

A.2 Analysis

Within the scope of mobile business solutions, the assumption of *standard* office applications was made for defining the test cases. The data traffic was measured for the following scenarios:

Reading mail. The user opens an e-mail message by double-clicking it. Subsequently, the message window is closed. This sequence is performed twice.

Typing text. The user constantly types text into a window for 1 minute. This sequence is performed twice, once with word wrap turned on, and once without word wrap.[1]

Surfing the web. The user enters an URL-address into the web browser. This was performed on four different websites sequentially, `www.kth.se`, `www.hut.fi`, `www.smartner.com`, and `www.w3.org`. After the last web page completely retrieved, the user scrolls down to the bottom of the page by using the cursor key. Finally, the page is scrolled up at once by pressing the page-up key.

Opening a fullsize window. The user opens a window and closes it. This sequence is performed twice.

Dragging a window. The user drags a window to the right, completely beneath its original position.[2] Then, the window is dragged back to the original position.

[1] *Word wrap* denotes the option of wrapping text lines when typing instead of continuing in one infinite single line. Most text editors allow selecting whether word wrap is used or not.

[2] This means that the distance Δx for dragging to the right is equal to the width x of the window; i.e. $\Delta x = x$.

No activity. The user is idle, i.e. no commands are being performed and the remote screen remains unchanged. The duration of this measurement is 700 seconds.[3]

As for the measurement configuration, both the VNC server and client are located on a Local Area Network (LAN) with 100 Mbit/s capacity. One user was accessing one server, and no other remarkable traffic was on-going during that time.[4] The traffic caused by the VNC application is being monitored at the server side by using a `tcpdump`-like application, where a filter only records the TCP events caused by the VNC server application.[5] Another filter allows the distinction between sent and received data to be made. Obviously amout of data sent is expected to be much higher than the data received from the client, since it contains the desktop screen data, whereas received data only contains events and commands sent from the user terminal. The screensize of the remote desktop is $1152 \cdot 864$ pixels with a color depth of 24 bits (true color), and all test cases are performed with fullscreen applications, except for *Typing text* with a window size of $463 \cdot 500$ pixels and *Dragging a window* with a window size of $472 \cdot 574$ pixels. The platform for performing the measurements is Windows 2000 on both client and server side.

A.3 Results

In total, 15635 lines of TCP dump were recorded, in a total duration of 700 seconds ($\approx$18 minutes). The traffic recordings have been plotted into distribution graphs (*see* figures A.1 to A.6), and the final results are summarized in table A.1. With the time on the x-axis and the amount of traffic shown on the y-axis, the specific events of each test case are clearly illustrated in the corresponding graphs.[6]

An interesting observation for the *Reading mail* test case is that closing a mail message seems to cause higher peaks in the traffic graph than opening a message.[7] The message window was opened somewhere around $t\approx$60 s and $t\approx$170 s, whereas it was closed at $t\approx$120 s and $t\approx$240 s (*see* figure A.1).

Clearly visible in the case *Typing text* is the difference between having word wrap turned on and off (*see* figure A.2). Obviously, typing text without word wrap (for $100 < t < 150$) causes the higher variation in graphics to be rendered (due to the single line constantly scrolling left) that typing text with word wrap (for $200 < t < 250$).

The plot for the *Surfing the Web* test case (*see* figure A.3) shows the expected behavior; the four given web sites were retrieved succesively at $t\approx$60 s, $t\approx$150 s, $t\approx$225 s, and $t\approx$275 s. The final web page was scrolled down at $t\approx$320 s and scrolled to the top of the page at $t\approx$380 s.

[3] From measuring traffic for no activity, we expect an impression about the weight of VNC's proprietary command traffic, i.e. polling, keep-alive messages, etc.

[4] We could therefore assume neither the network to be overloaded by our application, nor other applications to cause any capacity interference to our transmission. Thus, the measurement of traffic on the client or server is sufficient.

[5] We used the following VNC properties: Accept socket connections, automatic display number selection, foreground window polling, and console windows only polling were all enabled. The remaining options, i.e. disable remote keyboard and pointer, disable local keyboard and pointer, poll full screen, poll window under cursos, and poll on event received only, were all disabled.

[6] Note that the y-axis may vary in each graph.

[7] This is due to the underlying windows having to be repainted by the system. Although the data amount (the pixel area) is the same as when opening the mail window, the underlying window pixel data is already resident in the server's memory, and can be sent in a shorter time—which causes the high peak.

Again, an interesting observation for the *Opening a fullsize window* case is that closing a window causes higher peaks in the graph than opening a window. The window was opened at $t \approx 90$ *s* and $t \approx 225$ *s*, whereas it was closed at $t \approx 160$ *s* and $t \approx 275$ *s* (*see* figure A.4).

Two clear peaks in the plot are depicted in the *Dragging a window* test case (*see* figure A.5), where the largely increased traffic occurs at $t \approx 110$ *s* and $t \approx 170$ *s*. This reaction is obvious, since the real-time dragging of a window causes constant graphic changes.

Leaving the system in a state of *No activity* for a longer time, shows as expected a very low overall traffic load (*see* figure A.6). The three major peaks occuring within the entire test interval, may be explained by some application-layer specific keepalive messages, maintaining the session.

Within a given time interval Δt, the single data amounts of each TCP dump line (i.e. one packet) were accumulated for sent data and received data separately (*S* and *R* in *Traffic* column in table A.1). The choice of length and position of Δt was made based on the graphic result, assuring the data transmission peak to be caught. Thus, the respective average transmission speed on an interval with $m + n$ TCP packets can be calculated as

$$\texttt{speed}_S = \frac{\sum_{line=1}^{n} \texttt{data}_{line_S}}{\Delta t} = \frac{\texttt{traffic}_S}{\Delta t},$$

$$\texttt{speed}_R = \frac{\sum_{line=1}^{m} \texttt{data}_{line_R}}{\Delta t} = \frac{\texttt{traffic}_R}{\Delta t}$$

which is summarized in the column *Speed*. Values entitled with "Σ" denote sums, values entitled with "ø" denote average values.

Case	TCP packets			Δt [s]	Traffic [kB]			Speed [kB/s]			Scaled* [kB/s]
	S	R	Σ		S	R	Σ	S	R	Σ	
Reading mail											
1. open	75	67	142	10	67.7	0.3	68.0	6.8	0.0	6.8	0.9
1. close	693	700	1393	10	175.1	3.3	178.5	17.5	0.3	17.8	2.3
2. open	127	178	305	10	76.2	0.8	77.1	7.6	0.1	7.7	1.0
2. close	276	270	546	10	133.9	1.2	135.1	13.4	0.1	13.5	1.7
Typing text											
no word wrap	703	121	824	60	951.4	0.6	952.0	15.9	0.0	15.9	n.a.
word wrap	481	1760	2241	60	68.1	8.9	77.0	1.1	0.1	1.3	n.a.
Surfing the web											
`kth.se`	276	276	552	20	709.7	1.3	710.9	35.5	0.1	35.5	4.6
`hut.fi`	254	219	473	10	356.5	1.1	357.6	35.7	0.1	35.8	4.6
`smartner.com`	1033	1064	2097	30	376.4	5.0	381.5	12.5	0.2	12.7	1.6
`w3.org`	450	75	525	30	614.0	0.4	614.4	20.5	0.0	20.5	2.6
scroll down	507	530	1037	30	126.9	2.4	129.3	4.2	0.1	4.3	0.6
press page up	114	19	133	10	118.2	0.1	118.3	11.8	0.0	11.8	1.5
Opening a fullsize window											
1. open	63	18	81	10	75.9	0.1	76.0	7.6	0.0	7.6	1.0
1. close	308	306	614	10	947.1	1.5	948.6	94.7	0.1	94.9	12.2
2. open	138	91	229	10	96.4	0.5	96.8	9.6	0.0	9.7	1.2
2. close	130	114	244	10	920.4	0.5	920.9	92.0	0.1	92.1	11.8
Dragging a window											
right	773	931	1704	20	246.6	4.3	251.0	12.3	0.2	12.5	n.a.
left	979	1166	2145	20	358.2	5.5	363.7	17.9	0.3	18.2	n.a.
No activity											
	185	165	350	700	127.3	0.6	127.9	0.2	0.0	0.2	0.0
Total											
	7565	8070	15635	1070	344.5	2.0	346.6	21.9	0.1	22.0	2.9
	Σ	Σ	Σ	Σ	ø	ø	ø	ø	ø	ø	ø

* The scaling is explained in equation 3.14 on page 79.

Table A.1: The test case results

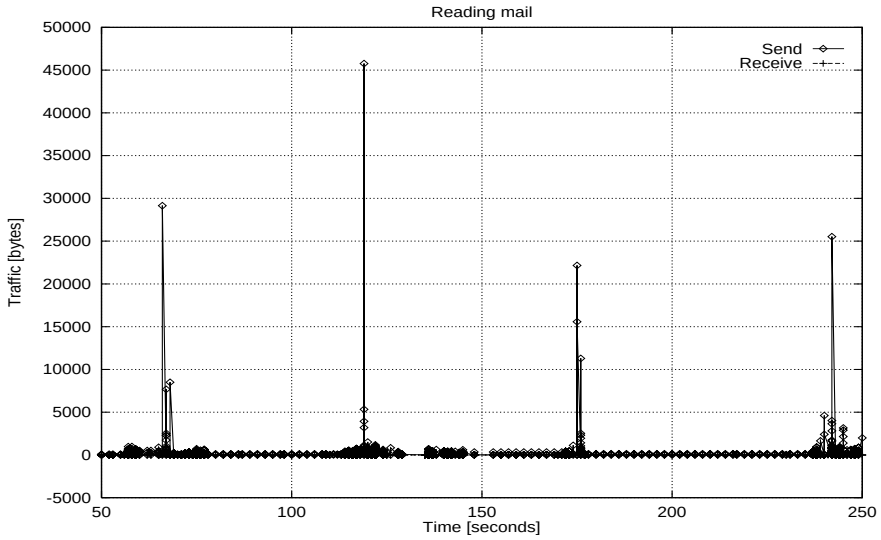

Figure A.1: Reading mail

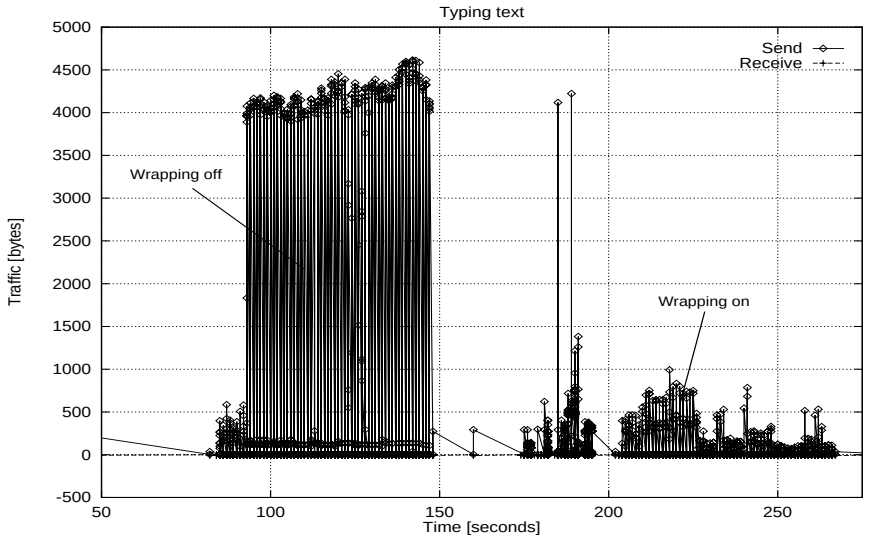

Figure A.2: Typing text

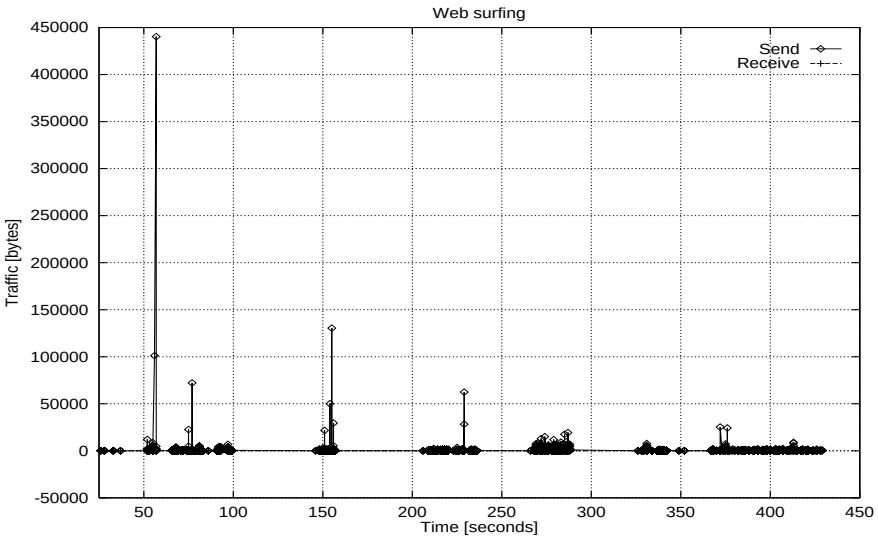

Figure A.3: Surfing the web

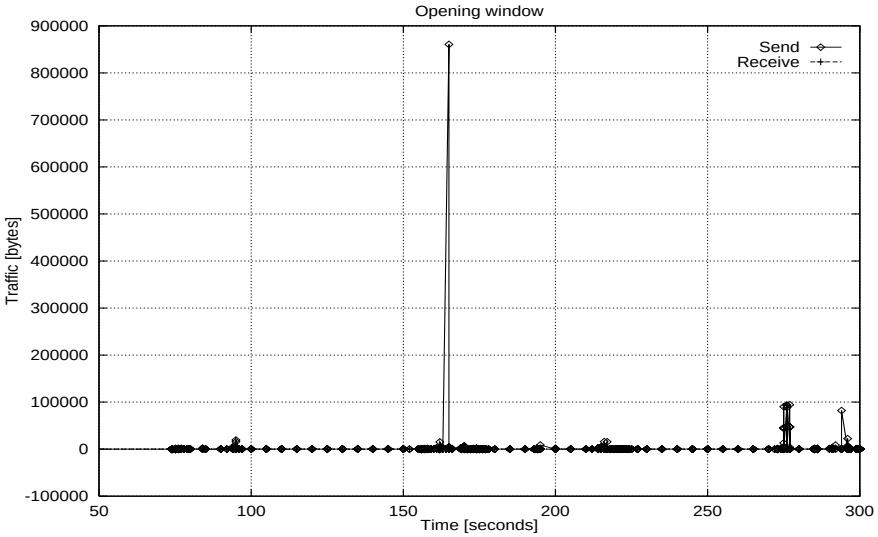

Figure A.4: Opening a fullsize window

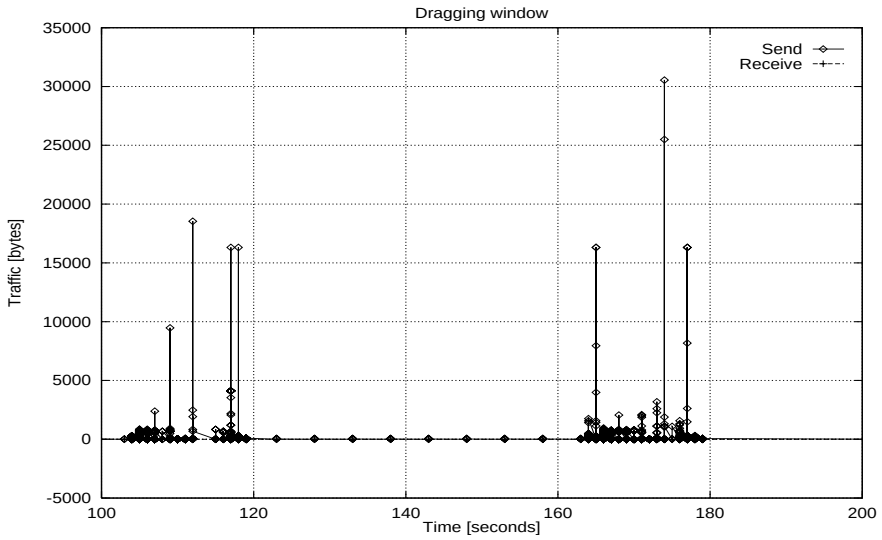

Figure A.5: Dragging a window

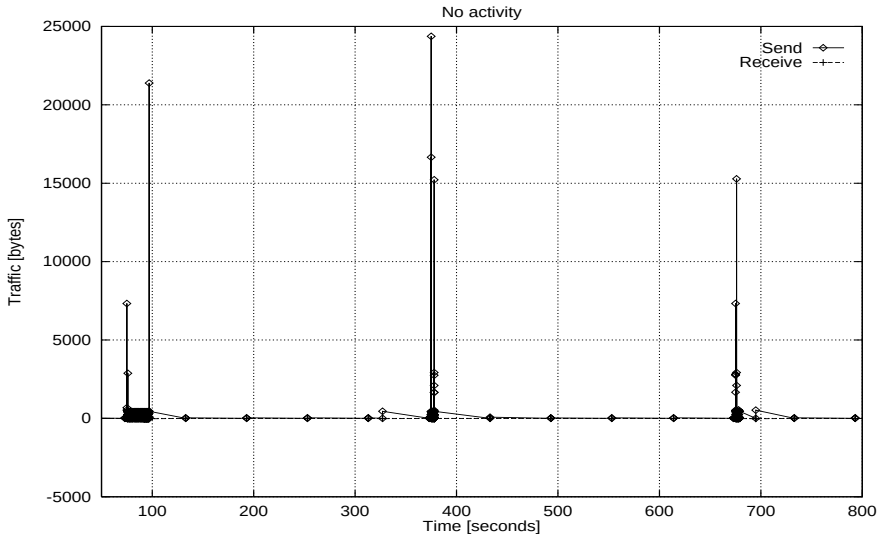

Figure A.6: No activity

Appendix B

White Pages Application

B.1 Architecture

B.1.1 Overview

The application was developed in *PersonalJava* and was compiled with *Java Software Development Kit (SDK) version 1.1.8*. Before developing the application with the Symbian SDK, a standalone application was developed in a native Java *shell* environment (*see* appendix C). The *JavaPhone* package as well as *Ælfred* [30] were used as external APIs.

The implemented components together forming the application can be arranged into six abstract categories (*see* figure B.1). The following section briefly introduced the object components belonging to the implementation. A more detailed description can be found in the `javadoc` generated API documentation.[1]

B.1.2 Components

There are three main components, belonging to the top-level *application* implementation:

EPOCstandalone. This class represents the EPOC application. It is an extension of the Symbian-specific `CFrame` class and contains a main-method as well as an implementation of the Symbian-specific `CBAListener`,[2] implementing the Nokia 9210 Communicator four-buttons menu. This component integrates the `ContactsPane` and the `StatusBar` and also calls external API specific methods declared in the *External API connection* class category.

Standalone and FourButtons. The class `Standalone` was the initial class for developing the application. It was kept thin, and the Symbian-specific menu was emulated by the `FourButtons` class.[3] Together, they formed the AWT shell for integrating subcomponents. Now, all subcomponents are embedded into `EPOCstandalone`.

[1] The API documentation is available at: `http://www.hacklin.com/thesis/wpages`
[2] The *Command Button Area (CBA)* is the four-button area on the right side of the display.
[3] After emulation in the shell, this class was replaced by the CBA.

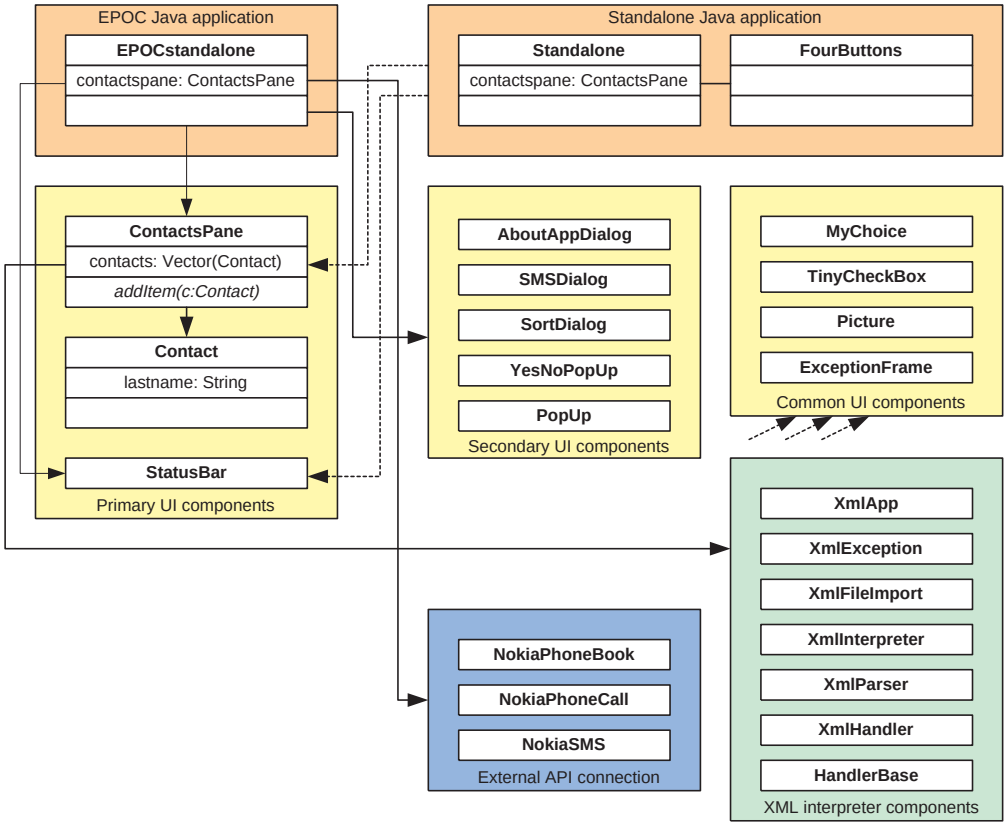

Figure B.1: Application architecture overview

The *Primary User Interface (UI) components* set contains three classes, representing the main user interface implementation and functionality:

Contact. This class represents a contact record, both conceptually (the contact record fields) and graphically (AWT components). In this implementation, a contact record consists of two different fields for phone numbers, demonstrating the difference between a corporate direct ID (DID)—for placing phone calls to—and the GSM number—for sending SMS messages to.[4]

ContactsPane. This class represents the set of contacts, both conceptually (the vector of contact records) and graphically (AWT components). This class furthermore implements methods for adding, removing, changing, and sorting record items. Most of the application logic and functionality is defined in this `ContactsPane`.

StatusBar. This class represents a graphic component, placed in the bottom of the application window, used for outputting a single line of text output.

The *Secondary UI components* set specifies separate windows, all of which are extensions of the `CFrame` class (*see* above). All the classes in this category are used

[4]For using more data fields, e.g. more phone numbers, the record format can be changed.

as dialogs, i.e. as windows prompting for user input. They are being called by `ContactsPane` and `EPOCstandalone`:

AboutAppDialog. This class displays the *about* dialog in a new window. This dialog is also displayed while the application is initializing; when the application is ready this window is being deleted.

PopUp. This class represents a generic *pop-up* window. While this window is active, the underlying application waits until the user has responded to the pop-up (e.g. *yes*, *no*, or *cancel*). By utilizing the `CBAListener` (i.e. implementing the Java interface), it also uses the four-button menu. This class is provided by the SDK.

SMSDialog. This is the pop-up dialog for entering and editing SMS messages. The class is based on, but not as an extension of the `PopUp` class.

SortDialog. This is the pop-up dialog for sorting the `ContactsPane` based on the given field. The class is based on, but not as an extension of the `PopUp` class.

YesNoPopUp. This is the pop-up dialog for a generic Yes-No-Cancel question The class is based on, but not extending the `PopUp` class. This class is abstract, and the key methods must be overridden. It must furthermore also implement the `CBAListener` as the `CBAHandler` is expecting to call-back to a `CBAListener`'s `cbaActionPerformed()` method when a CBA button is pushed.

The *Common UI components* are used by basically all classes implementing the GUI. All of them are implemented due to restrictions given by the SDK, i.e. due to components not available or not working in the used version of the SDK.

ExceptionFrame. This component represents a window for debugging purposes. It is used for catching exceptions in frameworks where standard console output does not exist.

Picture. A Panel containing an Image. This is a minimal implementation of a class well-known from Swing, i.e. a component containing and displaying a given image file, automatically sizing to that size given by the image file.

TinyCheckBox. This class implements a boolean value *check box*, it is simpler, and most importantly smaller (in terms of pixel width and height) than the the class `CheckBox` given by the SDK.

MyChoice. This class contains an implementation of a graphical *list of choices*, since the class `Choice` given by the SDK did not work in a reliable way.

The set of *XML interpreter components* contains the implementation of the XML interpreter, needed for integrating the external data into the application. It is based on a lightweight open source XML parser *Ælfred version 1.1* written by David Megginson:[5]

HandlerBase. This class is a base for all Ælfred handlers.

[5] *Ælfred* is provided by *Open Text Corp.* and written by David Megginson [30].

XmlApp. This class contains a base implementation for Ælfred based XML applications. The class fills in the basic interface, and provides an I/O infrastructure for simple applications and applets.

XmlException. This class implementats the exception class for reporting XML parsing errors.

XmlFileImport. This class contains a the XML parser application using Ælfred's event stream. This class is called by `ContactsPane` to import the contacts entries from the parsed XML file:

```
XmlFileImport xfi = new XmlFileImport(this);
Vector vec = xfi.getXmlContacts();

for (int i = 0; i < vec.size(); i++)
    blindAddItem((Contact) vec.elementAt(i));

updateLayout();
```

XmlHandler. This component implements the XML processing interface. Whenever an XML document is parsed, an object from a class that implements this interface must be provided in order to receive the parsing events.

XmlInterpreter. This class implements a sample application illustrating the event stream as provided by Ælfred.

XmlParser. This class implements the procedure for parsing XML documents and returning parse events to the application.

The XML file parsed by the application has the following format:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<contacts>
  <user>
    <lastname>
      Doe
    </lastname>
    <firstname>
      John
    </firstname>
    <title>
      CEO
    </title>
    <did>
      100
    </did>
    <msisdn>
      +35891234100
    </msisdn>
  </user>
</contacts>
```

Finally, there are three classes in the *External API connection* category, implementing the integration of external API usage into the application. These classes use the Java PIM and Java Telephony API[6] (both are part of Symbian's *JavaPhone* package) provided by the *Nokia Java SDK 0.3 Beta*.:

NokiaPhoneBook. This class provides access to the device's built-in contact database. It is mainly responsible for connecting the Nokia API with the application. It completely hides the API use from the application, thus the application can initiate phone book operations by simple commands, such as:

```
NokiaPhoneBook npb = new NokiaPhoneBook(this);

// [...]

npb.createContactCard(contact);
```

NokiaSMS. This class sends a text mesage. A message can be sent from the application with method calls, such as:

```
NokiaSMS nsms = new NokiaSMS(epocparent,
                             smstextarea.getText(),
                             smsnumber);
```

NokiaPhoneCall. This class represents the establishment of phone calls. A new phone call can be invoked by using the following method:

```
NokiaPhoneCall npc = new NokiaPhoneCall(this,
                                        callnumber);
```

B.2 User's Guide

B.2.1 Getting Started

The most convenient way to install the White Pages application on the Nokia 9210 Communicator is to make use of the *Symbian Install Script (SIS)*, provided in the package. The name of the installation file is `wPages.sis` (*see* figure B.2).

Figure B.2: Symbian Connect install script icon

When using the SIS installation, the terminal device must be connected to the serial port of the PC. When the SIS installation package is launched on the PC, the *Symbian Connect* application should automatically launch in the background, and

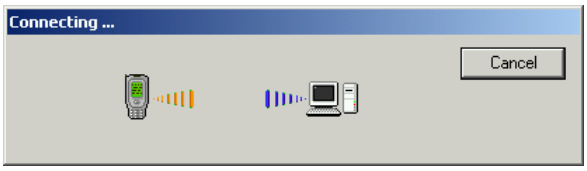

Figure B.3: Running the Symbian Connect install script

establish the connection to the terminal device. The installation script will automatically copy all necessary files and create folders on the terminal device (*see* figure B.3).

If the installation procedure succeeds, the *wPages* icon should appear the next time the *Extras* panel in the terminal device is entered (*see* figure B.4). The application can be launched by selecting the icon and either pressing the *Open* button in the Command Button Area (CBA) menu, or hitting the enter key.

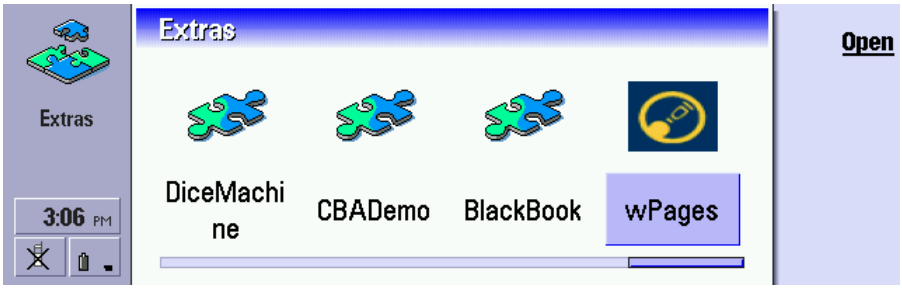

Figure B.4: Launching the application from the Extras menu

When the application has finished loading, a table with contacts will appear. These can be scrolled with the cursor keys and selected with the enter key. Pressing the *All* button will select the complete set of contacts.

B.2.2 Menu Overview

Although a mouse pointer is used in the terminal device emulator, the application is designed for use only through the keyboard and the four-button CBA menu. Except for some cases, the main view of the application does not change; the contacts list remains always visible, and all control happens through the CBA menu. The CBA menu has one top-level menu, from where three submenus are accessible. This menu structure can be viewed as a tree with one root, where the root represents the top-level menu, and every subroot represents a submenu. Every leaf of the tree represents a function (*see* figure B.12). After entering a submenu or function, the corresponding previous upper-level menu can be reached by pressing the last button (either *Cancel* or *Back*). The following sections will give a brief description of the usage of the menus and functions implemented in this version.

[6]These APIs are specified in the `javax.pim` and `javax.telephony` packages.

B.2.3 Placing Calls

Make sure the phone is turned on and a SIM card is inserted and registered with the network. Also make sure there are no other connections (voice or data) currently active. After selecting an entry, a phone call to the number given by the `Phone number` field will be established by pressing the *Call* button. If multiple contacts are selected, the application will place calls to all the selected contact numbers sequentially.[7] If the call establishment fails (e.g. busy or no answer), the attempted phone call can be re-established by pressing *Retry call*. An established call can be ended by pressing *End call*. By pressing the *Close* button, the application returns to the main menu (*see* figure B.5).

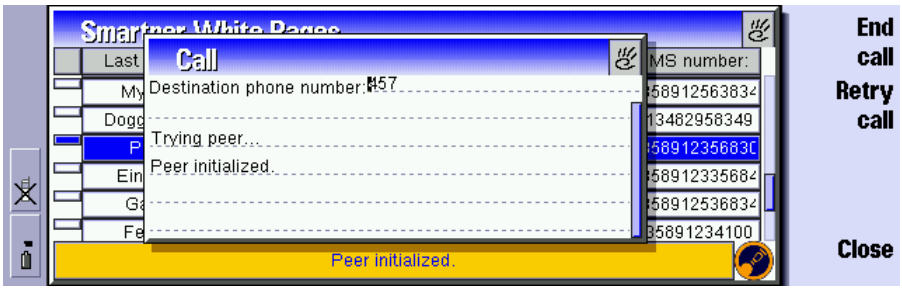

Figure B.5: Placing a phone call

B.2.4 Sending Messages

Make sure the phone device is turned on and a SIM card is inserted and registered to the network. After having selected an entry, press the *Write* button; this opens a new window for entering a text message.[8] Pressing the *Send* button sends the currently entered message to the number specified by the `SMS number` field. As the application uses the underlying API for sending the message, it will use the SMS settings specified by the terminal device selected (SMS centre etc.). By pressing the *Clear* button, the currently entered text is deleted. The deleted text can be recalled by pressing the *Undo* button. If selecting multiple contacts for sending messages to, the application will send the same message to to all the selected contact numbers sequentially. By pressing the *Cancel* button, the application returns to the main menu (*see* figure B.6).

B.2.5 Synchronization Menu

This submenu contains some functions for copying remote contacts entries to the device specific local contacts database.

[7] For future versions, one could think of using this feature for establishing conference calls with a number of parties. However, this API was not available in the JavaPhone API. Conference calls are supported by the GSM network.

[8] Note that in this version, only single SMS messages with 160 characters are supported.

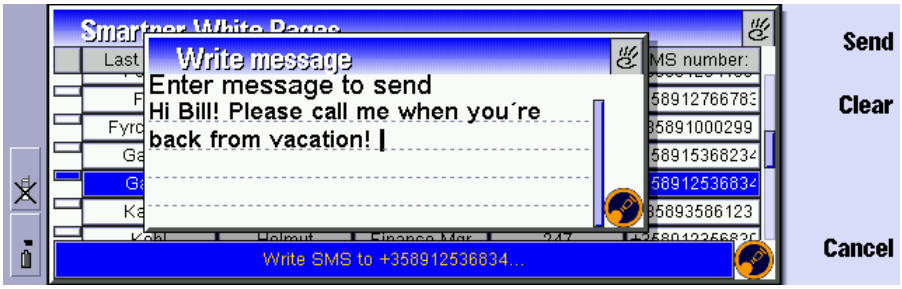

Figure B.6: Sending a message

Update online

Make sure the phone device is turned on and a SIM card is inserted and registered to the network. Also make sure there are no other connections (voice or data) currently active. Pressing the *Update online* button causes the application to attempt to establish a HTTP connection to the remote content representation server, in order to download the XML file (*see* figure B.7). As the application makes use of the underlying API, it uses the device specific TCP/IP support. If the settings are correct, a dial-up networking agent should establish IP connectivity. After having downloaded the XML file, it replaces the previous, locally stored XML file; this new file will be loaded the next time the application is launched.[9] The previous contacts table disappears for a few seconds, while the XML parser is interpreting the XML file. Finally, the new contact set is being displayed as sorted in the order specified by the *Sort* function (*see* section B.2.6).

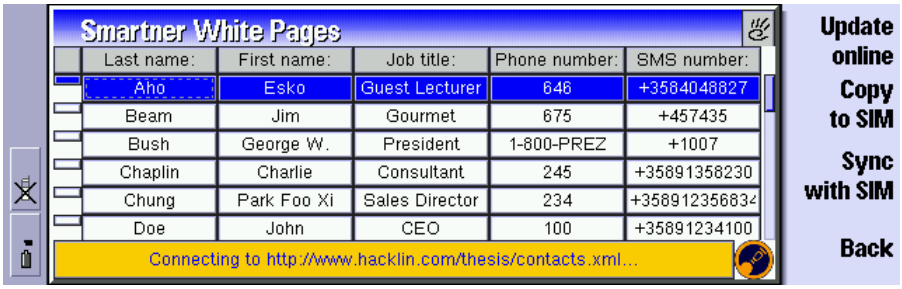

Figure B.7: Retrieving the remote set of contacts

Copy to SIM

After having selected one or multiple contacts, pressing the *Copy to SIM* button causes the selected contacts to be copied to the device specific local contacts database (*see* figure B.8 and B.9). If a contact with the same name already exists in the local database, a new, duplicate contact with the same name is being created.

[9] The previous version of the file can still be found as a backup file `contacts.xml.bak`.

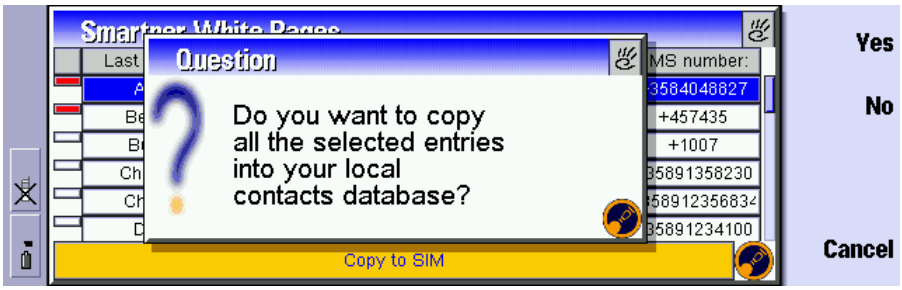

Figure B.8: Copying multiple contacts to the local device

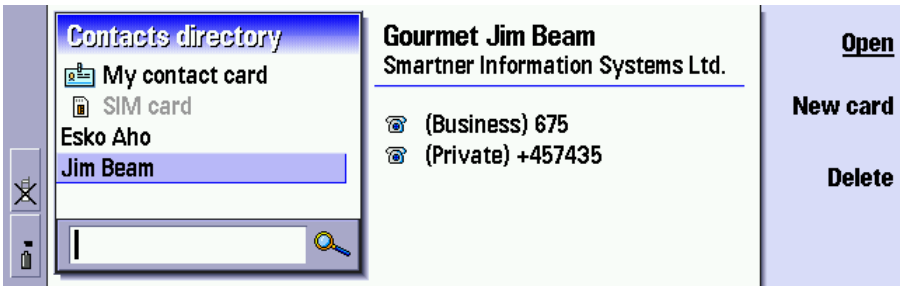

Figure B.9: The local device contacts after updating

Sync with SIM

Pressing the *Sync with SIM* button invokes the same functionality as pressing the *Copy to SIM* button, except that contacts entries existing in the local database with the same name are overwritten. This is a very simple synchronization method—a *server dominated one-way synchronization*. Due to lack of time, no further synchronization schemes were implemented. Instead of using `vCard` (*see* section 3.3.3) as a device specific database format standard, SyncML could be used instead.

B.2.6 Extras Menu

This submenu contains some extra, user interface specific functionality.

Sort

By pressing the *Sort* button, a dialog window is opened. Pressing the button *Toggle Field*, the current field for sorting the contacts list alternates. The application performs `quicksort` based on simple alphabetical ordering. After having reached the desired field for sorting, the dialog can be closed by pressing *Cancel* (*see* figure B.10).

Keys

For using the application completely without using an alphabetical keyboard, pressing the *Keys* function delivers three useful additional control keys (*see* figure B.11).

Figure B.10: Sorting the contacts

The *Up* button calls the same event as if the *cursor up* key was pressed. The *Down* button respectively calls the same event as if the *cursor down* key was pressed. The *Check* button checks the currently highlighted item, which causes the current CBA button to change label to *Uncheck*.

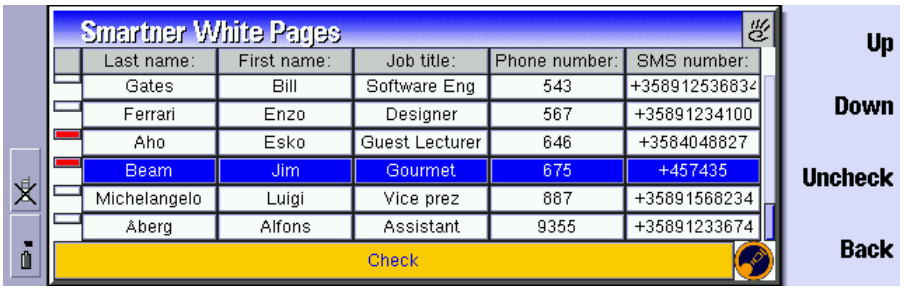

Figure B.11: Scrolling with CBA key control

About

By pressing the *About* button, the about window is being displayed (*see* figure 3.24). The window can be closed by pressing the *Close* button.

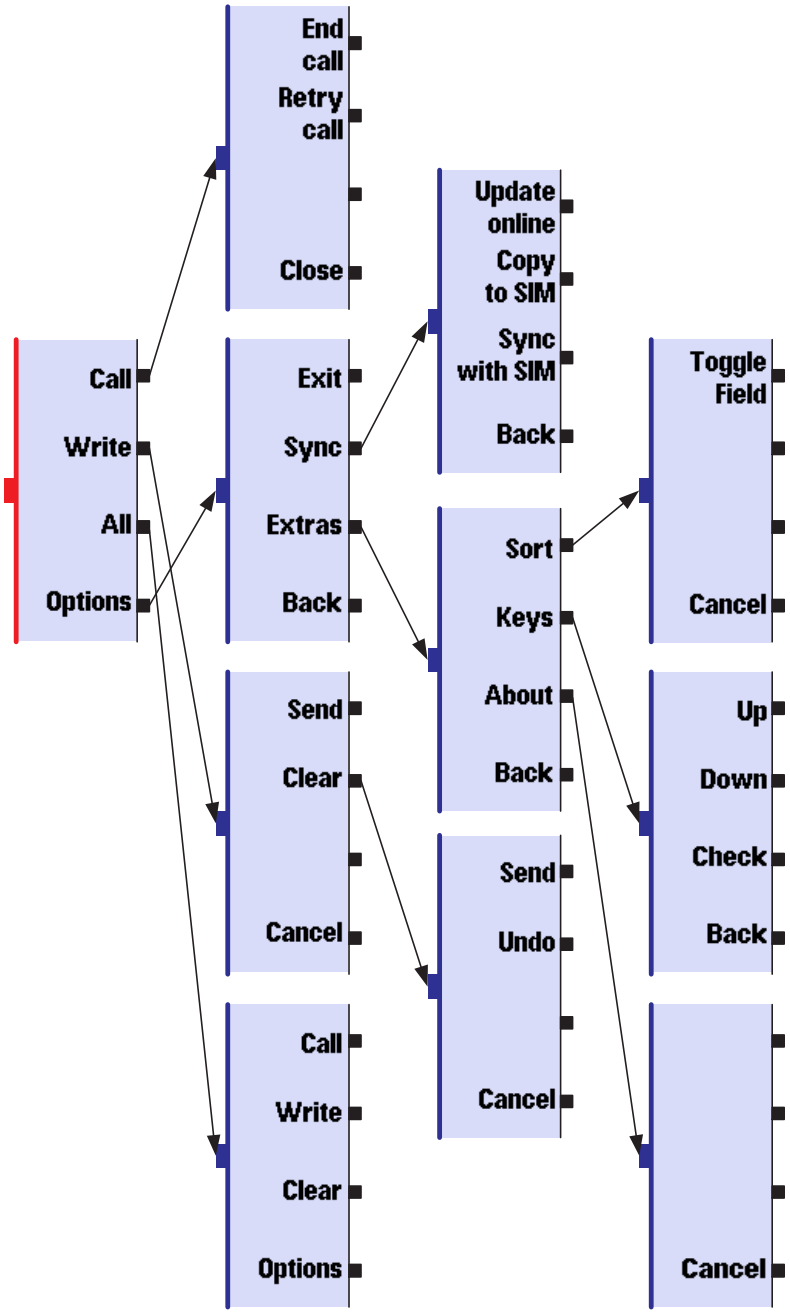

Figure B.12: The CBA menu structure

Appendix C

Symbian Specific Development Aspects

C.1 Application Development

As already mentioned in section 3.7.8, the Symbian SDK has a somewhat complicated build process, which makes the path from source code to running application too long, hence discouraging testing after every small change. Obviously, testing the application in the terminal device requires extra time for transferring the application into the device. However, the emulator is suitable for testing the application as long as GSM call or SMS message API functions need not be tested. Fortunately, the device specific phone book database does work in the emulator.

As with the *White Pages 1.0* application, it is thus recommended to build the application in a native Java environment for as long as possible, and profit from Java's platform independence. If the application is developed on a Windows platform, using the AWT packages with Java version 1.1.8, and the application window is dimensioned to $640 \cdot 200$ pixels, the look-and-feel when embedding it into the emulator does not change that much (*see* figures C.1 and C.2). Once the development shifted over to the emulator environment, the Java Virtual Machine (JVM) has to be restarted after running the application, which implies that the complete emulator to be restarted every time a change in the application is made. In our environment, restarting the emulator environment took about $1\frac{1}{2}$ minutes. This slows down the entire development process drastically, and can be very frustrating.

Choosing an approach for embedding a native Java application into the emulator as late as possible, it is recommended that you design the application using two layers of abstraction. The top level should define the standalone application as thin as possible, and integrate the second level component, which basically defines the *real* application. Thus, the real application component is hidden from the specific system environment.[1] The top level component will then be defined in two versions, i.e. one version for native Java, which should contain the `main()` method, and the

[1] The reader should notice that this approach is somewhat contradictory to the platform independence provided by the Java programming language. With somewhat more effort, it would be possible to redefine the CBA and other Symbian specific interfaces for the use in a native Java environment. In this case, no modifications in the source code would be necessary at all. However, this task would have been out of scope for this thesis.

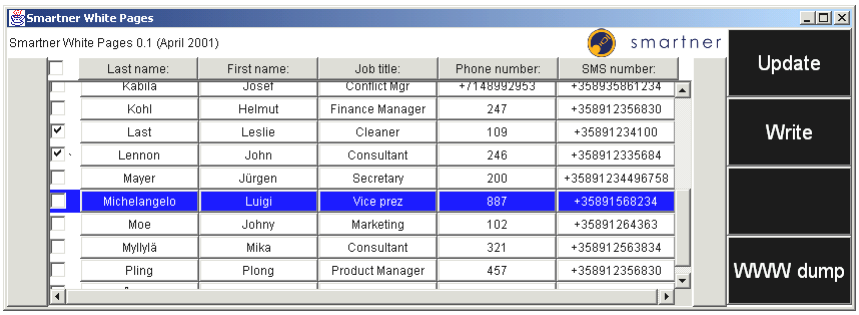

Figure C.1: Designing the standalone application

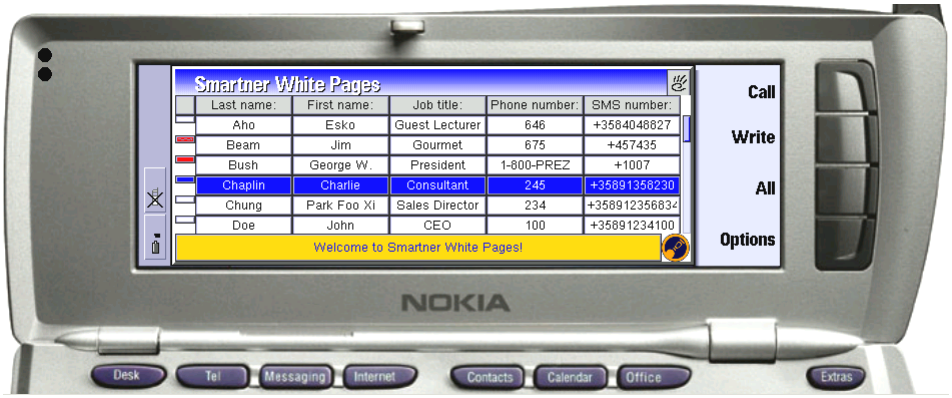

Figure C.2: The application after embedding into the emulator

Symbian component, which implements the `CBAListener`. Once these components are defined, transparency for further development is guaranteed by letting both top level components integrate the common second level component. This abstraction could be accomplished by e.g.:

$$\underbrace{\begin{array}{r}\texttt{NativeJavaApplication}\\ \texttt{SymbianApplication}\end{array}}_{\textit{top level abstraction}} \begin{array}{c}\textit{instantiates}\\ \longrightarrow\\ \textit{instantiates}\\ \longrightarrow\end{array} \underbrace{\begin{array}{l}\texttt{MainComponent}\\ \texttt{MainComponent}\end{array}}_{\textit{second level abstraction}} \tag{C.1}$$

In our case, the top level abstraction is provided by our implementations, `EPOC-standalone` and `Standalone`, whereas most of the application logic is represented by `ContactsPane` (*see* figure B.1).

C.2 Final Applicaton

When embedding the Java application into the emulator, the Symbian specific SDK tools have to be used for embedding the final application into the EPOC environ-

ment. This main tool for transforming the Java application to a EPOC standalone application is the `aifbuilder` (*see* figure C.3). For building a final installation package, the tool `makesis` is used.

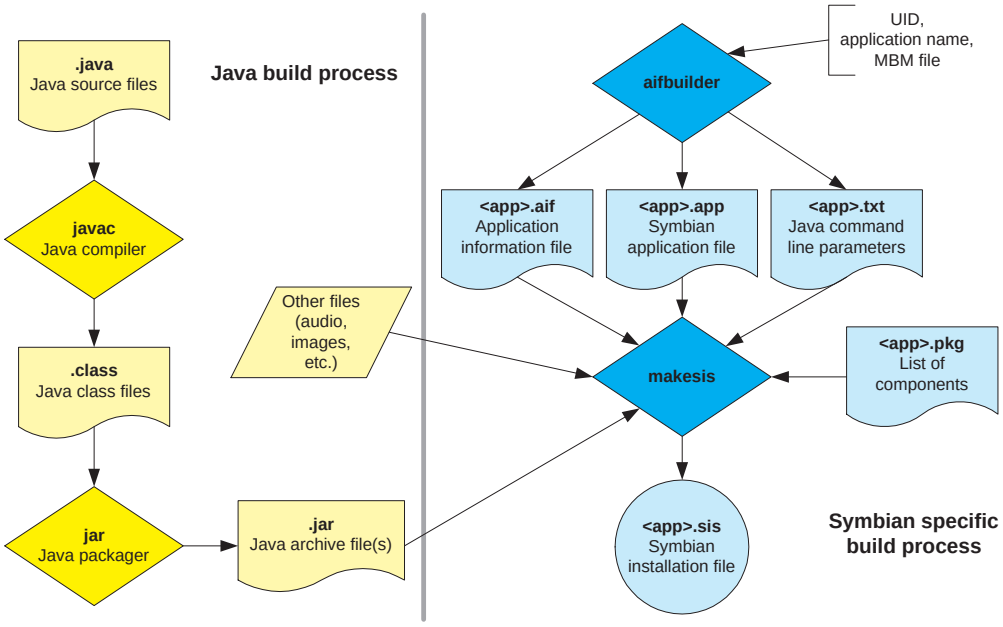

Figure C.3: From Java source code to Symbian application [167]

Once the Java source code is compiled into the set of `.class` files, a Java archive (`.jar` file) can also be built. However, this archiving step is not necessary; the application can be built based on the set of `.class` files.

C.2.1 `aifbuilder`

When embedding the Java standalone application into the EPOC emulator, the `aifbuilder` has to be used at least once for generating application related files. This is the step where the application name and unique identifier (UID) is registered to the EPOC desktop, program icons are assigned (`.mbm` multibitmap files), and documentation and language support is administered. Once the `aifbuilder` related files are generated, this step does not have to be repeated everytime the application is recompiled, unless substantial changes are made in the Java application (e.g. change of class names, etc.). The main outcomes of this step are [24]:

Application Information File (`.aif`). This file contains general representation information about the application, including its graphic representation for the OS, which can be bitmap icons in different sizes for different display modes. Furthermore it allows captions associated with various languages to be defined.

Symbian Application File (`.app`). This file defines the unique identification for the application and allows it to be detected so it is added to the Extras bar.

Java Command Line Parameters (`.txt`). This file contains the command line parameters for the Java Virtual Machine (JVM), i.e. it specifies which class to load initially (the class containing the `main()` method, and which archives to use (the classpath). The `.txt` file in our case is:

```
-cp .;cawt.jar
-Dcom.symbian.appName="EPOCstandalone" EPOCstandalone
```

The files generated during this step are already sufficient for testing the application in the emulator.

C.2.2 `makesis`

Before the final application is *shipped*, i.e. integrated into the terminal device, a simple way for installing this large set of small files and components into the Symbian device has to be performed. This packaging is accomplished by the `makesis` tool, which generates the installation executable file. The two file formats needed for this procedure are:

Package List (`.pkg`). This file contains the specification of *all* files to be integrated by `makesis`, i.e. application files as well as other files (audio, images, XML, etc.) needed by the Java application. The package list is preferably written manually, and contains the mapping of local files to the directory path on the destination Symbian terminal device:

```
#{"Smartner White Pages"},(0x10001104),1,0,1
; Only two files to install for the minimal application
"wPages.app"-"!:\system\apps\wPages\wPages.app"
"wPages.lst"-"!:\system\apps\wPages\wPages.lst"
"wPages.mbm"-"!:\system\apps\wPages\wPages.mbm"
"wPages.txt"-"!:\system\apps\wPages\wPages.txt"
"wPages.aif"-"!:\system\apps\wPages\wPages.aif"
"wPages.aifb"-"!:\system\apps\wPages\wPages.aifb"
# etc.
```

Symbian Installation Script (`.sis`). The `makesis` reads the `.pkg` and includes all files listed. The resulting file, the Symbian Installation Script is an executable file, which provides a standardized user interface for installing any Symbian application onto the Symbian device. The installation script makes use of Symbian Connect (*see* figure B.3), i.e. it also takes care of connecting to the Symbian device connected to the PC's serial port.

Appendix D

Qualitative Survey Results

D.1 Introduction

The aim of this survey was to get the industry's vision on the future of general front-end mobile middleware solutions in the context of 3G. The target group was a set of contacts from executives and skilled academics in the area of wireless and mobile research and development. The survey has no quantitative background, and was carried out over e-mail.[1,2]

D.2 Questions

(1) **How do you see the future of mobile middleware in the context of 3G?**

(2) **Consider the case of future mobile applications getting more application centric (running on PDAs, Smartphones), would not the worst case imply the need of middleware to become fully obsolete in the longterm?**

(3) **What will 3G mobile middleware look like, once the application running on the terminal communicates over IP with the network server, just using the operator *in the middle* as a bearer?**[3]

D.3 Answers

→ **(1)** *There will be a growing dependency on mobile middleware in 3G as we do not expect a single access network to dominate. Besides that, old networks (2G, 2.5G) will stay for a while. Therefore, we see the future access networks will consists of heterogeneous networks. There will be a need to unify the access to all these networks.*

The need for a middleware for services can also be justified. The traditional concept of building services for a single network is slowly moving towards an aggregated/unified service concept. This will be clearer when the telecom networks merge with the IP networks. When

[1]The reader should notice that the statements expressed below should be considered rather as the respondents' personal opinions than the respective company's vision.

[2]The survey results can be found online at `http://www.hacklin.com/thesis/survey` [89].

[3]In the original questionnaire, the word *bearer* was misleadingly misspelled as *barrier*. However, this led to an unintended provocation, which in turn resulted in interesting answers.

this happens, services will be built using similar components or building blocks. This justifies a middleware that unifies all the common components that are needed to build the services.

As far as devices are concern, I believe that there woudn't be a single device fits alls solution when it comes to accessing services in a mobile environment. Here again, we see the need for a middleware that would adapt the services to the device in the mobile environment.

→ **(2)** *There is a tradeoff when we speak of making the device less intelligent (just running applications). We can't perform that many complex functions (like mobility, security, QoS, mobile agent, etc.). I would agree that middleware would be obsolete if everyone would be just satisfied with what a PDA or smartphone can provide; that is, simple applications and functions. But that is not the case usually. (see my argument above about size of devices).*

→ **(3)** *There will still be some network functions that are still needed by the terminal to optimize the use of application in a mobile environment. Some of these include handoff and mobility issues, security, policy, etc. The middleware will try to unify these components on various networks.*

—Jaya Shankar
Assistant R&D Manager
Centre of Wireless Communications (CWC)
Singapore

→ **(1)** *If anywhere it has its future there because:*

- *3G is about IP (i.e. data rather than voice)*
- *more different devices are coming which need alignment*
- *much more feature rich service network including service realted billing and location APIs*

→ **(2)** *This is one (unlikely I guess) scenario. The exploding terminal (via BlueTooth) will mean that the various terminal components get less of an idea of the total service. Middleware could help here. Other trends of openenig up networks like active networking or JAIN mean that we will get distributed platforms end-to-end. Furthermore the location of the middleware could very well be the corporate server or the home PC (which has little to do if you are travelling). So I think middleware will come but "terminals"(i.e. network endpoints) will be important execution platforms for them. A lot will depend on interoperable execution environments which still seem to converge on Java I guess.*

→ **(3)** *The operator in the middle can do alot if he tries to cooperate rather than dictate. IN was a model which did not recognise smart terminals. Realising those the operator can do alot in hosting applications, providing both secure storage, processing and access to important data like billing, location, achievable QoS etc. But as said before the success will depend to my opinion whether the operator provides open middleware interfaces or not. "Eat or die"will not work - in the end the user has the choice.*

— Dr. Norbert Niebert
Manager Mobile Multimedia Networks Research Group
Ericsson Eurolab
Germany

→ **(1, 2, 3)** *My short answer to your question is that I believe we need application and support for those that allows application to be used from all sorts of access networks and devices. I would like to have my calendar available both from my PC at home as well as from my PDA and mobile phone.*

—Magnus Fransson
Ericsson Innovations
Sweden

→ **(1)** *Konvergensen mot Internet kommer att leda till att 3Gs systemdel absorberas i Internet, men det är en tvåvägsprocess, så Internet kommer också att påverkas av de mobila aspekterna.*

→ **(2)** *Nja, det beror på. Man kan ju mycket väl tänka sig att vi får se lösningar som gridware (där du kan delegera ut en del av bearbetningen till en annan processor). I så fall blir middleware centralt. Men det är en annan typ av middleware.*

→ **(3)** *Se ovan. Kortsiktigt kommer operatören naturligtvis att få hitta nya värden att tillföra, till exempel tillhandahålla caching, anpassning till användarens terminal och så vidare. På lång sikt tror jag vi kommer att se ett systemskifte som är lika stort som det vi har gått igenom de senaste åren.*

—Johan Hjelm
Nippon Ericsson
Japan

→ **(1)** *3G will extend the capabilities of mobile devices and applications. With 3G the heterogenity in communications will increase. Thus, I see a huge need for an intelligent and flexible approach in respect to an efficient mobile middleware. Btw. of course I am not talking about centralised approaches, but distributed solutions. One fancy approach is certainly the exploitation of mobile agent technology.*

→ **(2)** *Absolutely not, personally I feel that future applications have to be more user-centric. This means that these applications have to be ease to use, globally available and users expecting similiar interfaces on different devices. Exactly this is a great opportunity for an advanced mobile middleware.*

→ **(3)** *The middleware should take care about the quality of service, personal mobility management, content adaptation and privacy.*

The worst case scenario for operators looks exactly like that. Operators will only remain as bit pipes. However, operators, in particular mobile operators, have various chances to create new value-added application scenarios including financial transactions, content provision and inter-company relationsships.

—Jens Hartmann
Mobile Applications Laboratory
Ericsson Eurolab
Germany

→ **(1, 2, 3)** *Mobile middleware can be roughly divided in two: connectivity middleware and enabling middleware. Connectivity middleware consists of gateways (such as WAP gateway or SMS gateway) and performance enhancing proxies. Enabling middleware consists of wireless specific middleware such as location, subscriber profiles, service and device management, notification services, etc. Enabling middleware can be further divided into components that are network bound by nature and difficult to reproduce outside of the operator network, and application bound middleware (such as content management and manipulation solutions) that are not dependent on the network.*

It seems likely (but obviously not certain) that middleware components will increasingly appear also in the terminals, in addition to the network. This is a direct consequence of the enhanced computing capabilities of the terminal.

The role of enabling middleware is clearly increasing in the foreseeable future as applications grow in complexity and integration of different applications and services becomes increasingly important. The role of connectivity middleware is clearly changing in the context of 3G as wireless specific application protocols are less necessary than in 2G networks. Due to the specific characteristics of radio networks, performane enhancing proxies continue to be important.

—Esko Hannula
Director, Mobility Middleware
Nokia Networks
Finland

→ **(1)** *3G will deal with higher datarates and thus bigger datavolumes. There will be more access to data-pools and more "1 to many"communication than we see in 2.5G networks. In this respect there will definitely be a place for mobile middleware systems in 3G networks. The specifics of mobile networks will also not change with 3G (hand-over issues, ceised connections and the like) so that network support (i.e. middleware) will always be needed to fully exploit the systems.*

→ **(2)** *I do not at all believe that mobile applications will be more application- centric in future. The main application on a higher layer will always be information exchange. The applications around this info-stream will grow and be able to handle largely complex "on-site"processing. But the exchange of data will always be the inner value of most of the mobile applications and this implies an important place for the middleware architectures.*

→ **(3)** *The operator carrying bits and bytes w/o added value is definitely the looser in a game where differentiation is of growing importance. All winning operators are now trying to improve their position into something that adds intelligence IN the network supporting the end-to-end communications. As soon as the operator (owning the middleware part as well) does not "understand"any longer what is being transmitted, he cannot reasonably support the datafeed with middleware intelligence. Adaptive behaviour supported by middleware servers has a future and will improve the convenience with mobile terminals as much as higher bandwidth ever can.*

—Dr. Eckhard Geulen
Director of Strategy
Sapient
Germany

→ **(1, 2, 3)** *On the part of mobile middleware, I see its main use in ERP/SCM/other backoffice activity directly related to a specific company and its partners. These back-office systems will not be renewed as new technologies come along, rather the new technologies are adapted to these old systems. This is where EAI layers (such as MQ Series) play a significant role - they provide a path to the systems from any device, don't you think? On top of that, some kind of adapters (like AIRIS's) are needed, if those adapters are not provided by thick clients. As IP technology advances and mobile devices start using IP as their communication method to access more horizontal services (the internet...) than I don't really see a role for mobile middleware - the devices function as Internet terminals, where's the beef for additional mobile middleware?*

—Jani Kelloniemi
mCommerce Specialist
DareStep, a division of Cap Gemini Ernst & Young
Finland

→ **(1)** *Middleware will be the key to realize the "intelligent internet".*

→ **(2)** *The radio resource will contine to be the bottleneck, demanding middleware or "an intelligent edge"to create valuable user experiences. Content must be brought closer to the user applying intelligent caching and stored in memory to be there when the user initiates a request. Middleware will also be key to process in real time Location data, content and personalization.*

→ **(3)** *See above answer. Operators will continue to strife to avoid commoditization and provide "intelligent edge"that will give mobile operators a clear distinction and different value propostion to customers compared to the traditional "fat pipe"wireline internet.*

—Arnthor Halldorsson
Director, Advanced Services
Western Wireless
USA

→ **(1)** *I see a big need for Mobile middleware in 3G. The middleware will handle downloading of apps and information to the terminals. Since the terminals soon will have much more memory (since memory is getting cheaper and cheaper and infrastructure is not) will the middleware also be used for keeping the terminals updated with the latest and greatest.*

→ **(2)** *The applications will absolutely be executed on the terminals, I don't believe in a thin client with just a browser.*

→ **(3)** *I don't see the operator as a barrier. They have to offer their customers value added services otherwise they will end up as just a pipe provider. They have a lot of advantages since the "own"user info and i.e. the user location in the network. I believe the operator will add value and the future killer app will be Location based Multimedia Messaging.*

—Anders Huge
Development Manager
Intel Sweden

→ **(1)**

a. *Little future for much of it - it mostly just gets it the way*

b. *Middleware that helps – will be useful, but this middle wear is more related to providing funcitons to support either server or end-terminal functionality and is NOT inserted by the network operator as an attempt to control/maximize profit/intermediate*

→ **(2)** *Yes, most is obsolete now. Worse is that much of it actually reduces the performance and adds to delay.*

→ **(3)** *Barrier = bearer? – or was this a pun, since today the operator is often a barrier in the middle!*

The mobile middleware will run on the end-terminal or the the server and will provide functions to support user, device, terminal, .. mobility. (For example, enabling the user to move from one terminal to another while still maintaining their sessions, media streams, etc. Or enabling the user to redirect a media stream to a specific device for playout (such as a high quality audio system in the room where the user is).)

—Prof. Gerald Q. Maguire Jr.
Royal Institute of Technology
Department of Microelectronics and Information Technology
Sweden

→ **(1)** *3G is supposed to make Mobile Internet a reality. This would mean that very many if not most of the "Internet applications"would be available through mobile phones and other mobile client devices.*

We have seen application servers to become the platform of choice for the current Internet applications. Application server is the platform for running these multi-tier applications (especially for transactional applications) and provides several system level services and especially those required for scalability, ie. application servers are the middleware. BEA has phrased this as application server being the operating system for Internet.

So if my first comment about mobile Internet is true, middleware is the key ingredient for 3G applications. This fact can also be seen in the large number of new mobile platform offerings becoming available.

→ **(2)** *These will be multi-tiered applications with the client either only having a browser (=very thin client) or some part of the application also running on the client. Just using a browser clearly needs middleware, but I believe that we will see several applications running on both the client and the server (this is actually the traditional fat client approach that failed with PC's, because the software architecture couldn't support the requirements and especially the scalability needs. This is why middleware for distributed systems was developed in the first place.)*

Middleware provides the scalability and availability services for large applications. It is also about productivity, since you don't have to program infrastructure services and can concentrate on the application logic in stead. I see now reason why this requirement would disappear with distributed applications.

It is however typical that packaged applications use proprietary middleware which is not exposed. Best example currently is SAP R/3. Especially when the market is young it is typical that most of the applications will be based on proprietary solutions where middleware

services are embedded into the application code. Same happened with the current Internet, where companies like Amazon develop their applications by coding on the socket level.

Even if these applications would perform, the maintenance and management will be expensive and difficult. Especially when more and more new device types come available.

→ **(3)** *At least currently it looks like that the strongest contender for the programming model is Java2 EE for the server side and Java2 ME. Microsofts .NET initiative is basically a similar architecture, but is still only on the concept level.*

So these will be distributed layered applications. Then there is the other question: who will be owning and managing these layers. The operator will need to be able to offer value added services to be able to survive.

Some enterprises will probably want to own even the access servers to be able to control the whole application. For example Internet banks for security reasons.

On the other hand operators should be able to add value, because at least managing large number of connections is something they now how to do. It would therefore be natural for them to host a collection of services and provide easy access to their end customers.

It remains to be seen what is the business model that will sustain and who the players will be. Since mobile Internet is very much a consumer business, mobile entertainment will have a key role. It may strongly affect the business models and also the choice of middleware.

—Tom Henriksson
Global Account Manager
BEA Systems Finland

Appendix E

Scenarios for Future Wireless Office Usage

E.1 Introduction

In the world of mobile terminals, the race about operating systems is not decided yet. It is not said that requirements for usability will be the same as in desktop computing. Open standards such as EPOC or Java based systems might be more favourized by the users, compared to proprietary systems developed by major software vendors, such as Windows CE. On the other hand, some terminal device vendors already are convinced that Windows CE will be the one and only standard. In this appendix, a set of hypothetical scenarios for future wireless office use will be sketched.

We see the key issue based on a strategic balance between two directions:

$\mathcal{S}_1$: *Mobile usage will be desktop application-oriented.* Users prefer the same applications and usability on mobile devices as on desktop computers. $\Rightarrow$ Currently successful mobile phone vendors (e.g. Nokia, Ericsson, or Motorola) will experience difficulties in evolving their own OS platform (e.g. EPOC) and application suite. Microsoft will be dangerous and dominate with Windows even on the mobile applications front.

$\mathcal{S}_2$: *Mobile applications usability will be dramatically different.* People finally want a *different* OS, usability, and different applications on their mobile devices (e.g. you cannot start *double-clicking* and *drag-and-dropping* icons while walking on the street). Mobile usage might become browser oriented; users do not want locally stored applications (e.g. no trust, no security); devices cannot handle required data amount. People want Internet Explorer. $\Rightarrow$ Applications will be deployed through the WASP approach, partnering with operators can be fruitful, mobile portals will be evolved, etc.

Combining both directions, one hybrid scenario would be e.g. as follows:

$\mathcal{S}_1 \cap \mathcal{S}_2$: *People want current desktop applications with different usability.* Although this scenario somewhat contradicts to $\mathcal{S}_1$ (i.e., the mobile application environment being *transparent* to the corresponding desktop environment),

it could imply the success of e.g. pen-based Windows CE applications etc., where mobile usability has been taking into account to a certain extend, while designing the interface.

Similiar scenario considerations for the deployment of open standards versus proprietary wireless networks exist [137]. Applied on 3G, these could be described by the following opinions:

3G optimist. There are too many barriers and system constraints, that will significantly impede the practical deployent of open standards wireless networking (such as WLAN or ad hoc). The wireless net effect exists, but this movement will likely never occur significantly, compared to the deployment of operator owned proprietary 3G networks.

3G pessimist. Technological advances permit the deployment of large scale open standards, heterogenous networks, permitting wireless wide area Internet infrastructures with low cost, and little or no reliance on existing, fixed infrastructures.

3G pragmatist. This viewpoint falls somewhere between the first two opinions. Open source networking will be an enabling technology judged on its merits, filling the holes and extending the edges where proprietary forms of 3G wireless networks are either relatively costly or inflexible. Open source networks will deliver complementary connectivity supporting the evolution of ubiquitous computing.

E.2 EPOC Dominance Scenario

Characteristics

- 3G applications will denote dramatic differences in usability.
- Thin applications (e.g. browser based application access) will evolve in the short term.
- Hybrid Thickness applications will evolve in the long term (i.e., separation of content generation and representation).
- OS, hardware platform, and wireless bandwidth will allow remote desktop access (such as Tarantella, VNC, Citrix, etc.).

Implications

- Current mobile phone vendors will be able to establish widespread OS and software platforms.
- Focus on remote desktop access, SyncML, transparent synchorization, and mobile computing file system approaches.

E.3 Windows CE Dominance Scenario

Characteristics

- Successful mobile applications will be equivalent to successful desktop applications, with the addition of taking the wireless handsets' screensizes into account.
- Microsoft will develop its native applications for the Windows CE platform.
- EPOC will remain a pseudo-OS for managing contacts, writing notes, sending SMS messages, but will not be able to compete with Windows CE.

Implications

- It will be tremendously difficult to compete with generic office applications against Microsoft Office.
- The concept of *extending* an office application for mobile access does not make sense anymore, if the same application already exists in a mobile terminal application version.
- A possible focus could be based on customized applications (*see* section 3.7.5), where applications would be designed based on the requirements of a specific business segment.

E.4 Java Platform Dominance Scenario

Characteristics

- There will not be any significant dominance in OS and hardware platform; mobile terminals are based on vendor specific OSs.
- Java will dominate the software platforms.
- Common interface given by Java support, applications can run independently on OS and hardware platform.
- We will observe a trend towards *download and run* applications rather than *install once and keep*.

Implications

- Develop applications in J2ME and PersonalJava.

E.5 Laptop Dominance Scenario

Characteristics

- The mobile phone will not converge significantly towards Internet usage, the phone remains a device for talking only.

- The mobile phone will be even smaller, and eventually even loose its screen and keypad completely (e.g. earphone only).
- The mobile device will be used for connectivity, i.e. the connection establishment to the Internet/Intranet. The laptop will access the device thrgohu short range radio protocols, such as e.g. Bluetooth.

Implications

- Whereas the mobile phone does not converge significantly, the role of a laptop converges towards even smaller and lighter devices, with integrated wireless LAN access.
- The laptop is only being used for mobile business applications (which are equivalent to common desktop applications).
- VPN issues are becoming significant.

E.6 Evaluation

E.6.1 Scenario SWOT Analysis

The scenarios described previously are applied on the Smartner case (*see* section 4.7). The evaluation is based on a SWOT analysis for each scenario (*see* tables E.1 to E.4), followed by a mutual strategy map combining all scenarios with eachother (*see* section E.6.2). We recognize, that the EPOC dominance scenario and Java platform dominance scenario can be regarded as most favourable from Smartner's point of view, whereas the Windows CE dominance scenario will be critical, and the Laptop dominance scenario fatal.

E.6.2 Cross-Scenario Action Strategy

Based on the multiple scenario evaluation, we want to combine each dominance scenario mutually for evaluating strategies for migrating across them. In our compilation below, we define actions for migrating from the scenario committed to, to the emerging scenario (*see* table E.5). Cases where the emerging scenario appears to be the same as the scenario committed to are regarded as optimal cases and are not considered further here.

E.6.3 Roadmap Solution Concept in Each Scenario

Finally, we evaluate a set of Smartner's roadmap solution concepts in the light of each scenario. For each solution concept, we discuss general advantages and disadvantages, as well as study the feasability for each scenario (*see* figures E.1 to E.3).

Strengths	Weaknesses
• Strong Java programming knowledge • Well positioned in operator business • Some experience in mobile applications usability	• No real experience in mobile applications design • Difficulties in accessing vendor dominated platform
Opportunities	**Threats**
• Provision of customized, vertical end-to-end solutions • Usage of hybrid thickness client applications • Partnerships for participation in wireless OS issues (built-in services, drivers, etc.) • Provision of remote desktop access solutions	• Generic office solutions already implemented • Emerging middleware platforms (such as SyncML) already solved and integrated

Table E.1: SWOT for EPOC dominance scenario (s_1)

Strengths	Weaknesses
• Good overview of advantages and disadvantages of proprietary software systems	• Missing knowledge and experience in Windows applications development • Weak position in providing competitive office applications • Missing knowledge of any specific vertical business sectors
Opportunities	**Threats**
• Focus on and understand *real* customer needs, provide alternative solutions to Windows CE • Focus on customized, vertical solutions for competing. • Acquire knowledge in Windows CE and C++ development issues • Use Java based, light and thin end-to-end applications for competing • Partner with operators in competing against Windows CE dominance	• Windows CE takes over mobile applications market entirely • Mobile applications will be the same as fixed ones • Windows platform becomes a complete end-to-end platform • Current knowledge and experience partly irrelevant • Operators get attracted to Windows platform (also in back-end systems) rather than to Java and EPOC

Table E.2: SWOT for Windows CE dominance scenario (s_2)

Strengths	**Weaknesses**
• Good knowledge and experience in Java software development • Existing Java based software platform	• Little knowledge in mobile application design (PersonalJava, J2ME)
Opportunities	**Threats**
• PersonalJava, J2ME • Provision of end-to-end solutions • Extension of MAP concept, e.g. application skeleton • Provision of hybrid thickness client applications	• *Download an run* • Technical difficulties in constructing serious business applications • Java applets will have more gaming character

Table E.3: SWOT for Java platform dominance scenario (s_3)

Strengths	**Weaknesses**
• General knowledge in application design	• Any separated approach for mobile business applications questionable
Opportunities	**Threats**
• Reorientate on VPN issues • Consider voice-based business services (e.g. VoiceXML) • Define intelligent, transparent services • E-commerce solutions based on existing transaction infrastructure • Bluetooth business applications (e.g. for opening garage and office doors)	• Entire solution concept not needed • Current knowledge and experience completely irrelevant • Radical business redefinition mandatory

Table E.4: SWOT for Laptop dominance scenario (s_4)

	B $\rightarrow$ *			
A $\downarrow$	s_1	s_2	s_3	s_4
s_1	*Optimum*	**Resistance.** Apply EPOC based solutions on Windows CE. Try to compete by pushing simplicity and mobile aware usability issues.	**Subset knowledge.** Use EPOC Java subset for J2ME. Discard any C++ based applications approaches.	**Reorientation.** Focus on smart, transparent middleware services (e.g. voice-based). Repudiate any commitment to visual mobile application approaches.
s_2	**Full Benefit.** Apply Windows CE knowledge on EPOC. Build EPOC office applications and integrated solutions. Focus on Java, but benefit from potential C++ knowledge. Try to compete by pushing simplicity and mobile aware usability issues.	*Optimum*	**Partial Benefit.** Apply Windows CE knowledge on Java. Build end-to-end Java system solutions.	**Semi-Reorientation.** Apply Windows knowledge on fixed, laptop world. Focus on smart, transparent middleware services (e.g. voice-based). Repudiate any commitment to visual mobile application approaches.
s_3	**Transfer.** Explore Java application possibilities in EPOC. Try to compete by pushing simplicity and mobile aware usability issues.	**Resistantial Transfer.** Explore Java application possibilities in Windows CE. Try to compete by pushing simplicity and mobile aware usability issues. Develop hosted-application concept based on *download and run*.	*Optimum*	**Reorientation.** Focus on smart, transparent middleware services (e.g. voice-based). Repudiate any commitment to visual mobile application approaches.
s_4	**Remigrate.** Combine laptop issues (VPN, middleware) and intelligent, transparent service concepts with thin application client solutions.	**Remigrate.** Combine laptop issues (VPN, middleware) and intelligent, transparent service concepts with thin application client solutions. Benefit from potential Windows platform knowledge.	**Remigrate.** Combine laptop issues (VPN, middleware) and intelligent, transparent service concepts with thin application client solutions.	*Optimum*

* **A**: Commitment scenario

B: Emerged scenario

s_1: EPOC dominance scenario

s_2: Windows CE dominance scenario

s_3: Java platform dominance scenario

s_4: Laptop dominance scenario

Table E.5: Cross-scenario action strategy plan

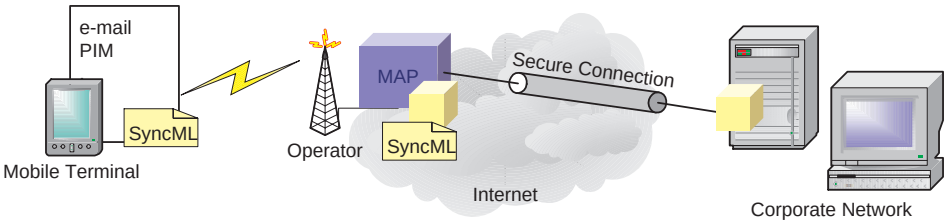

Figure E.1: Synchronization roadmap concept (R_1)

Synchronization Roadmap Concept

Advantages

- Open standard access to vendor specific terminal devices is supported.
- The concept provides an interesing outsourcing possibility of synchronization services for SMEs.
- The integration to Smartner's existing platform feasible, e.g. the online translation from XML to SyncML would be possible.

Disadvantages

- The concept would basically *repackage* existing services.
- It will be rather difficult to add own solutions (e.g. SyncML already exists).
- The corporate intranet has to be widely opened to the operator for autonomous access of the synchronization engine.

Scenario feasability

- The concept is highly feasible for any device vendor dominance, where little terminal application programmability is given; i.e. the EPOC dominance scenario (s_1).
- The concept is somewhat feasible with the Windows CE dominance scenario (s_2), but any approaches based on open standards will be difficult.
- The concept is unsuitable for the Java platform dominance scenario (s_3) and Laptop dominance scenario (s_4).

Both-side-stub Roadmap Concept

Advantages

- This approach offers good support for the provision of robust, reliable, and valuable end-to-end solutions.

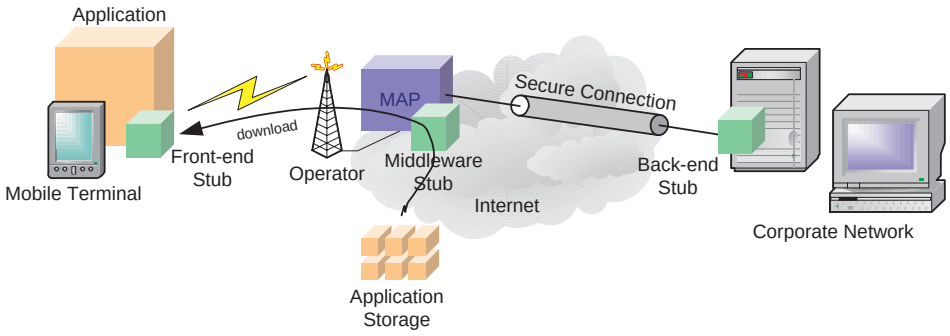

Figure E.2: Both-side-stub roadmap concept (R_2 and R_3)

- In and end-to-end solution, the applications can be based on own, proprietary protocols, i.e. any open standards formats do not necessarily have to be used.
- One important value will be based on usability and experience.
- Development and especially testing of new applications would be eased.
- An additional application compoment storage (R_3) could allow operators to host application clients and distribute them to terminal devices.

Disadvantages

- Realizing such a framework on other platforms than Java (J2ME and J2EE) seems rather difficult.
- The emerging Java client solutions might not be serious enough for complex business applications (i.e. applets have gaming character).

Scenario feasability

- The concept is highly feasible for the Java platform dominance scenario (s_3) and the EPOC dominance scenario (s_1) (use Java subset platform in EPOC).
- The concept will be difficult for the Windows CE dominance scenario (s_2).
- For the Laptop dominance scenario (s_4), this concept does not make sense.

VPN Roadmap Concept

Advantages

- VPN offers possibilities for operator to differentiate on packet traffic to business customers.

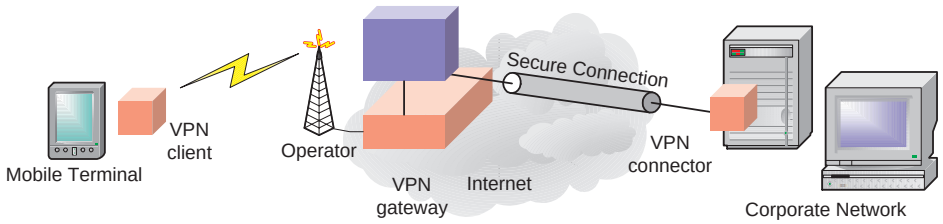

Figure E.3: VPN roadmap concept (R_4)

Disadvantages

- The migration will be difficult for Smartner.
- Smartner does not have any real knowledge about VPN issues.
- VPN technology involves network and link layer; these are new areas which will be difficult to combine with the existing application level middleware.

Scenario feasability

- This concept is highly feasible for the Laptop dominance scenario (s_4).
- It is somehwhat feasible for the Windows CE dominance scenario (s_2), considering the need to interconnect existing Windows applications.
- It is however unsuitable for the EPOC dominance scenario (s_1) and the Java platform dominance scenario (s_3).

Appendix F

Abbreviations and Acronyms

1G	*First generation of mobile communications*
2.5G	*see section 2.2*
2.75G	*see section 2.2*
2G	*Second generation of mobile communications*
3G	*Third generation of mobile communications*
4G	*Fourth generation of mobile communications*
ACM	*Association for Computing Machinery*
AMPS	*Advanced Mobile Phone System*
ANSI	*American National Standards Institute*
API	*Application Programming Interface*
ARIB	*Association of Radio Industries and Business*
ARPU	*Average Revenue Per User*
ASP	*Application Service Provider*
ATM	*Asynchronous Transfer Mode*
BS	*Base Station*
BSC	*Base Station Controller*
BTS	*Base Tranceiver Station*
CBA	*Command Button Area*
CDC	*Connected Device Configuration*
CDMA	*Code Division Multiple Access*
cdma2000	*Code Division Multiple Access*
CGF	*Charging Gateway Functionality*
CGI	*Common Gateway Interface*
cHTML	*Compact HTML*
CISC	*Complex Istruction Set Computer*
CLDC	*Connected Limited Device Configuration*
CLI	*Common Language Infrastructure*
CORBA	*Common Object Request Broker Architecture*
CPU	*Central Processing Unit*
CRM	*Customer Relationship Management*
CTI	*Computer Telephony Integration*
CWTS	*China Wireless Telecommunication Standard Group*
D-AMPS	*Digital AMPS*
DMZ	*Demilitarized Zone*
DSM	*Distributed Shared Memory*
DSVD	*Digital Simultaneous Voice and Data*
E-GPRS	*Enhanced GPRS*
EDGE	*Enhanced Data rates for Global Evolution*
ETC	*Electronic Toll Collection*
ETSI	*European Telecommunications Standards Institute*
FCC	*Federal Communications Commission*
FDD	*Frequency Division Duplex*
FDMA	*Frequency Division Mulitple Access*
FR	*Frame Relay*
GEO	*Geosynchronous Orbit*
GGSN	*Gateway GPRS Support Node*
GHz	*Gigahertz*
GMSC	*Gateway MSC*
GPRS	*General Packet Radio Service*

GSM	*Global System for Mobile communication*
GUI	*Graphical User Interface*
HAN	*Home Area Network*
HAPS	*High Altitude Stratospheric Platform Station*
HDML	*Handheld Device Markup Language*
HIPERLAN	*High Performance Radio Local Area Network*
HLR	*Home Location Register*
HSCSD	*High Speed Circuit Switched Data*
Hz	*Hertz*
I/O	*Input/Output*
IEEE	*Institute of Electrical and Electronics Engineers*
IEEE 802.11b	*WLAN*
IETF	*Internet Engineering Task Force*
IMAP	*Internet Message Access Protocol*
IMT-2000	*International Mobile Telecommunications*
ISDN	*Integrated Services Digital Network*
ISM	*Industrial, Scientific, Medical*
ISP	*Internet Service Provider*
ITS	*Intelligent Transport System*
ITU	*International Telecommunication Union*
J2EE	*Java 2 Enterprise Edition*
J2ME	*Java 2 Micro Edition*
J2SE	*Java 2 Standard Edition*
JPEG	*Joint Photographic Experts Group*
kb	*kilobits, 1000 bits*
kb/s	*kilobits per second*
kbps	*kilobits per second*
KB	*kilobytes, 1024 bytes*
KB/s	*kilobytes per second*
KBps	*kilobytes per second*
KVM	*K Virtual Machine*
L2TP	*Layer 2 Tunneling Protocol*
LAN	*Local Area Network*
LEO	*Low Earth Orbit*
ME	*Mobile Equipment*
MExE	*Mobile Execution Environment*
MIDP	*Mobile Information Device Profile*
MIT	*Massachusetts Institute of Technology*
MPEG	*Moving Picture Expert Group*
MS	*Mobile Station*
MSC	*Mobile Switching Center*
MSISDN	*Mobile Subscriber ISDN*
MT	*Mobile Terminal*
NMT	*Nordisk Mobiltelefoni*
NTT	*Nippon Telephone & Telegraph*
OOP	*Object-Oriented Programming*
OPL	*Optimization Programming Language*
OS	*Operating System*
OTA	*Over-the-Air*
PAN	*Personal Area Network*
PAP	*Push Access Protocol*
PCMCIA	*Personal Computer Memory Card International Association*
PCU	*Packet Control Unit*
PDA	*Personal Digital Assistant*
PDC	*Personal Digital Cellular*
PDI	*Personal Data Interchange*
PDS	*Passive Double Star transmission system*
PHP	*Hypertext Preprocessor*
PHS	*Personal Handyphone System*
PIM	*Personal Information Management*
PJEE	*PersonalJava Emulation Environment*
PLMN	*Public Land Mobile Network*
PSTN	*Public Switched Telephony Network*
PoP	*Point-of-Presence*
QRPC	*Queued Remote Procedure Call*
RAS	*Remote Access Server*
RDO	*Relocatable Dynamic Objects*
RFC	*Request for Comments*
RISC	*Reduced Instruction Set Computer*
RNC	*Radio Network Controller*
RNS	*Radio Network Subsystem*
ROI	*Return On Investment*
RTOS	*Real-time Operating System*
SGSN	*Serving GPRS Support Node*
SIBO	*Sixteen Bit Organiser*
SIM	*Subscriber Identity Module*
SME	*Small and Medium-sized Enterprise*
SMS	*Short Message System*
SMS-GMSC	*SMS Gateway Mobile Switching Center*
SMS-IWMSC	*SMS Interworking Mobile Switching Center*
SQL	*Structured Query Language*
SSH	*Secure Shell*
SSL	*Secure Socket Layer*
SWAP	*Shared Wireless Access Protocol*

SWOT	*Strengths, Weaknesses, Opportunities, Threats*
T1	*Stanards Committee T1 Telecommunications*
TDD	*Time Division Duplex*
TDMA	*Time Division Multiple Access*
TE	*Terminal Equipment*
TIA	*Telecommunications Industry Association*
TRAU	*Transcoder and Rate Adaptor Unit*
TTA	*Telecommunications Technology Association*
TTC	*Telecommunication Technology Committee*
UMTS	*Universal Mobile Telecommunications System*
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*
UTRAN	*UMTS Terrestrial Radio Access Network*
VLR	*Visiting Location Register*
VNC	*Virtual Network Computing*
VPN	*Virtual Private Networking*
VoIP	*Voice-over-IP*
W-CDMA	*Wideband CDMA*
WAE	*Wireless Application Environment*
WAP	*Wireless Application Protocol*
WASP	*Wireless ASP*
WDP	*Wireless Datagram Protocol*
WLAN	*Wireless LAN*
WSP	*Wireless Session Protocol*
WTA	*Wireless Telephony Application Server*
WTLS	*Wireless Transport Layer Security*
WTP	*Wireless Transaction Protocol*
WWW	*World Wide Web*
X	*X Window System*

Bibliography

[1] *3rd Generation Partnership Project*. [online] `http://www.3gpp.org` (referred June 2001).

[2] *3rd Generation Partnership Project 2*. [online] `http://www.3gpp2.org` (referred June 2001).

[3] *Citrix Systems*. [online] `http://www.citrix.com/products/server` (referred June 2001).

[4] *Coda File System*. [online] `http://www.coda.cs.cmu.edu/` (referred July 2001).

[5] *Federal Communications Commission (FCC)*. [online] `http://www.fcc.gov` (referred Aug. 2001).

[6] *Forum Nokia*. [online] `http://www.forum.nokia.com` (referred Mar. 2001).

[7] *Graphon Corporation*. [online] `http://www.graphon.com` (referred June 2001).

[8] *Intelligent Transport System (ITS) International*. [online] `http://www.itsinternational.com` (referred June 2001).

[9] *Intelsat*. [online] `http://www.intelsat.com` (referred June 2001).

[10] *Iridium*. [online] `http://www.iridium.com` (referred June 2001).

[11] *Jippii Freedom*. [online] `http://www.jippiigroup.com/wlan.shtml` (referred June 2001).

[12] *Lifix Systems*. [online] `http://www.lifix.fi` (referred June 2001).

[13] *MExE Forum*. [online] `http://www.mexeforum.org` (referred June 2001).

[14] *Microsoft Terminal Server*. [online] `http://www.microsoft.com/ntserver/terminalserver` (referred June 2001).

[15] *Microsoft Windows CE*. [online] `http://www.microsoft.com/pocketpc` (referred June 2001).

[16] *MobileMiddleware.com*. [online] `http://www.mobilemiddleware.com/what_is.htm` (referred Apr. 2001).

[17] *Nokia Operator Wireless LAN*. [online] `http://www.nokia.com/networks/wireless_lan/hl_solution.html` (referred June 2001).

[18] *Odyssey Product Information*. [online] `http://www.genmagic.com/agents/odyssey.html` (referred July 2001).

[19] *Palm OS*. [online] `http://www.palmos.com` (referred June 2001).

[20] *Personal File System (PFS)*. [online] `http://www.spa.is.uec.ac.jp/~tate/pfs/` (referred July 2001).

[21] *Sky Station International*. [online] `http://www.skystation.com` (referred June 2001).

[22] *Smartner Information Systems Ltd.* [online] `http://www.smartner.com` (referred June 2001).

[23] *StockholmOpen.Net*. [online] `http://www.StockholmOpen.net` (referred June 2001).

[24] *Symbian Developer Network*. [online] `http://www.symbiandevnet.com` (referred June 2001).

[25] *Tarantella*. [online] `http://www.tarantella.com` (referred June 2001).

[26] *Telia HomeRun*. [online] `http://www.homerun.telia.com` (referred June 2001).

[27] *Virtual Network Computing (VNC)*. [online] `http://www.uk.research.att.com/vnc` (referred June 2001).

[28] *WindRiver*. [online] `http://www.wrs.com` (referred June 2001).

[29] *Wireless Application Protocol (WAP) Forum*. [online] `http://www.wapforum.org` (referred June 2001).

[30] *XML/SGML Solutions: Ælfred*. [online] `http://www.opentext.com/services/content_management_services/xml_sgml_solutions.html` (referred July 2001).

[31] VxWorks, Version 5.2. *IEEE Design & Test of Computers*, 12(3):104, 1995.

[32] 3rd Generation Partnership Project (3GPP). QoS Concept and Architecture. TR 23.907 1.6.0, Technical Report, Technical Specification Group Services and System Aspects, Sept. 1999. [online] `http://www.3gpp.org/ftp/tsg_ran/WG3_Iu/TSGR3_08/Docs/Pdfs/R3-99e36.PDF`.

[33] 3rd Generation Partnership Project (3GPP). Architecture Principles for Release 2000. TR 23.821 V1.0.1, Technical Report, Technical Specification Group Services and System Aspects, July 2000. [online] `http://www.3gpp.org/ftp/Specs/2000-06/R2000/23_series/23821-101.zip`.

[34] 3rd Generation Partnership Project (3GPP). Mobile Execution Environment (MExE), Service Description, Stage 1, Release 4. TS 22.057 V4.0.0, Technical Report, Technical Specification Group Terminals, Oct. 2000. [online] `http://www.3gpp.org/ftp/Specs/2000-09/Rel-4/22_series/22057-400.zip`.

[35] 3rd Generation Partnership Project (3GPP). Mobile Execution Environment (MExE), Functional Description, Stage 2, Release 4. TS 23.057 V4.1.0, Technical Specification Group Terminals, Mar. 2001. [online] `http://www.3gpp.org/ftp/Specs/2001-03/Rel-4/23_series/23057-410.zip`.

[36] B. Aazhang and J. Cavallaro. Multitier wireless communications. *Wireless Personal Communications*, 17(2-3):323–330, June 2001.

[37] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: Data management for asymmetric communication environments. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD) 1995, San Jose, CA, USA*, pages 199–210, May 1995.

[38] Alice Systems AB. Alice Login 1.1. Technical Description, WL01-113, R3, July 2001. [online] `http://www.alicesystems.com`.

[39] R. Amit and C. Zott. Value creation in e-business. *Strategic Management Journal*, 22:493–520, 2001.

[40] S. Andhare, A. Borelli, V. Eriksson, F. Hacklin, and S. Kansal. Mobile Internet: The Markets, The Users, The Future. Global Project Coordination Team Report, Stanford University, Royal Institute of Technology, Stockholm School of Economics, Ericsson Packet-Switched Systems, June 2000.

[41] E. Autio, D. Bout, G. Golub, J. Häyrinen, B. Kohlenbach, S. Laitinen, F. Müller-Veerse, and S. Singh. *UMTS Report—An Investment Perspective*. Durlacher Research Ltd, London, Eqvitec Partners, Helsinki, Helsinki University of Technology, Helsinki, Mar. 2001.

[42] B. Ball. Building a wireless network with Linux. *ACM Linux Journal*, (73), 2000.

[43] D. Barbara and T. Imielinski. Sleepers and workaholics: Caching strategies in mobile environments. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD) 1994, Minneapolis, MN, USA*, May 1994.

[44] S. Basse. *Computer Algorithms: Introduction to Design and Analysis*. Addison-Wesley Publishing Company, 1978.

[45] K. Basu, A. T. Campbell, and A. Joseph. IP-based Mobile Telecommunications Networks. *IEEE Personal Communications*, pages 8–9, Aug. 2000.

[46] J. K. Bennett, J. B. Carter, and W. Zwaenopoel. Munin: distributed shared memory based on type-specific memory coherence. In *Proceedings of the 2nd ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming, SIGPLAN Notices, Seattle, WA, USA*, volume 25, pages 168–176, Mar. 1990.

[47] P. Bergström. Packet Channel Optimisation Strategies in UMTS. Master's thesis, Royal Institute of Technology, Stockholm, June 2001.

[48] T. Berners-Lee and D. Connolly. The HyperText Markup Language (HTML) Specification Version 2.0. RFC 1866, Nov. 1995.

[49] C. Bickers. The way of the mobile warrior. *Far Eastern Economic Review*, 164(24):46–48, June 2001.

[50] S. Björk, L. E. Holmquist, J. Redström, I. Bretan, R. Danielsson, J. Karlgren, and K. Franzén. WEST: a Web browser for small terminals. In *Proceedings of the 12th annual ACM symposium on User interface software and technology, Asheville, NC, USA*, pages 187–196, Nov. 1999.

[51] R. Bolla, F. Davoli, and M. Marchese. A bandwidth allocation strategy for multimedia traffic in a satellite network. In *IEEE Global Telecommunications Conference (GLOBECOM) 2000, San Francisco, CA, USA*, volume 2, pages 1130–1134, 2000.

[52] R. Bolla, F. Davoli, and M. Marchese. Adaptive bandwidth allocation methods in the satellite environment. In *IEEE International Conference on Communications (ICC) 2001, Helsinki, Finland*, volume 10, pages 3183–3190, 2001.

[53] L. Bos and S. Leroy. Toward an all-IP-based UMTS system architecture. *IEEE Network*, 15(1):36–45, Jan./Feb. 2001.

[54] G. Brasche and B. Walke. Concepts, Services, and Protocols, of the New GSM Phase 2+ General Packet Radio Service. *IEEE Communications Magazine*, 35(8):94–104, Aug. 1997.

[55] T. Braun, M. Guenter, and I. Khalil. Management of Quality of Service Enabled VPNs. *IEEE Communications Magazine*, 39(5):90–98, May 2001.

[56] G. Buchanan, S. Farrant, M. Jones, H. Thimbleby, G. Marsden, and M. Pazzani. Improving mobile internet usability. In *Proceedings of the Tenth International Conference on World Wide Web, Hong Kong, China*, pages 673–680, Oct. 2001.

[57] L. Cappelletti. Mobile computing: beyond laptops. In *Proceedings of the ACM 15th Annual International Conference on Computer Documentation (SIGDOC), Snowbird, UT, USA*, pages 23–26, Oct. 1997.

[58] J. Cheah. A proposed access method for the wireless lan standard. In *Proceedings of the Third IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Boston, MA, USA*, pages 145–148, 1992.

[59] M. E. Ching, A. Ahmad, W. Choi, and A. Shutze. Wireless equipment industry: What's in the air for 2001? Report, Global Securities Research & Economics Group, Merill Lynch, Apr. 2001.

[60] J. Cho and H. Garcia-Molina. Synchronizing a database to improve freshness. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD) 2000, Dallas, TX, USA*, May 2000.

[61] Commerzbank Securities. European Mobile Network Operators: Where to now? Report, Pan European Telecommunications, June 2001.

[62] I. R. Corden. 3G evolution: Real drivers and real solutions. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 355–357, Mar. 2001.

[63] J. Costello. Partnering: The Essential 3G Challenge. Technical report, Telecom Media Networks, Mar. 2001. Cap Gemini Ernst & Young.

[64] M. E. Crovella and A. Bestavros. Explaining World Wide Web Traffic Self-Similarity. Technical Report TR-95-015, Boston University, Computer Science Department, Oct. 1995.

[65] Y. Depeursinge. Applications of the future communications. *Wireless Personal Communications*, 17(2-3):331–335, June 2001.

[66] J.-P. Deschamps. Aligning technology beliefs and values: The starting-point of a technology vision. *Insights, Alliance for Technology-Based Enterprise*, (8), July 2001.

[67] A. Doufexi, M. Butler, S. Armour, P. Karlsson, A. Nix, and D. Bull. Simulated performance of the HIPERLAN/2 physical layer with real and statistical channels. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 407–411, Mar. 2001.

[68] L. Downes and C. Mui. *Unleashing the killer app: digital strategies for market dominance*. Harvard Business School Press, Boston, MA, USA, 1998.

[69] C. Drewes, W. Aicher, and J. Hausner. The wireless art and the wired force of subscriber access. *IEEE Communications Magazine*, 39(5):118–124, May 2001.

[70] European Telecommunications Standards Institute (ETSI). General Packet Radio Service (GPRS), Service description, Stage 2. *Digital cellular telecommunications system (Phase 2+)*. ETSI EN 301 344 V7.4.1 (2000-09).

[71] European Telecommunications Standards Institute (ETSI). HIPERLAN Type 2. System overview, Broadband Radio Access Networks (BRAN). ETSI TR 101 683 V1.1.1 (2000-08-02).

[72] N. Evans. Be Careful Choosing Wireless Middleware. *InternetWeek*, 84:23, Feb. 2001.

[73] N. Evans. The big challenge for wireless ASPs. *InternetWeek*, page 15, Apr. 2001.

[74] X. Facon and A. Dean. WirelessJAVA. An Overview of the Conference, Mar. 2001. [online] `http://www.microjava.com`.

[75] A. Fasbender, F. Reichert, E. Geulen, J. Hjelm, and T. Wierlemann. Any network, any terminal, anywhere. *IEEE Personal Communications*, 6(2):22–30, Apr. 1999.

[76] J. C. Ferreira. Will service convergence ever happen? A service based method for evolution planning. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 351–354, Mar. 2001.

[77] R. Fielding, C. Gettys, and J. Mogul. Hypertext Transfer Protocol (HTTP) Version 1.1. RFC 2616, June 1999. [online] `http://www.w3.org/Protocols/rfc2616/rfc2616.html` (referred Mar. 2001).

[78] S. Foucart. Des services aux contours encore flous. *Le Monde Interactif*, 15th January 2001.

[79] N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies. Nov. 1996.

[80] A. Ganz and A. Phonphoem. Robust Superpoll with Chaining Protocol for IEEE 802.11 Wireless LANs in Support of Multimedia Applications. *Wireless Networks*, 7(1):65–73, Jan./Feb. 2001.

[81] S. Giordano and W. W. Lu. Challenges in mobile ad hoc networking. *IEEE Communications Magazine*, page 129, June 2001.

[82] A. Gokhale and D. Schmidt. Techniques for optimizing CORBA middleware for distributed embedded systems. In *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '99), New York, NY, USA*, volume 2, pages 513–521, Mar. 1999.

[83] Goldman Sachs. Wireless Data. Issues & Outlook 2000. Report, Global Investment Research, Jan. 2000.

[84] Goldman Sachs. The Impact of 3G Mobile Phone Services: Telecom Services. Report, Global Equity Research, June 2001.

[85] D. Goodman. The wireless internet: Promises and challenges. *IEEE Computer*, 33(7):36–41, 2000.

[86] L. Greenstein. Transporting voice traffic over packet networks. *International Journal of Network Management*, 8(4):227–234, 1998.

[87] M. Haardt and W. Mohr. The complete solution for third-generation wireless communications: Two modes on air, one winning strategy. *IEEE Personal Communications*, pages 18–24, Dec. 2000.

[88] J. Haartsen. The Bluetooth radio system. *IEEE Personal Communications*, 7(1):28–36, Feb. 2000.

[89] F. Hacklin. Industry survey on the future of mobile middleware. Survey results. [online] `http://www.hacklin.com/thesis/survey` (referred July 2001).

[90] F. Hacklin. Towards the ubiquitous office vision. With focus on mobile data synchronization. Literature study for Master's thesis, Mar. 2001. [online] `http://www.hacklin.com/thesis/literaturestudy.pdf`.

[91] W. L. Hahn. Worldwide telecommunications equipment overview, 1999 through 2004. Report, Gartner Group, Mar. 2001.

[92] G. Hamel and C. K. Prahalad. *Competing for the Future*. Harvard Business School Press, Boston, MA, USA, 1994.

[93] D. Hardin. Crafting a Java virtual machine in silicon. *IEEE Instrumentation & Measurement Magazine*, 4(1):54–56, Mar. 2001.

[94] A. S. Hatloy. Strategies and scenarios for wireless information services. Master's thesis, Massachusetts Institute of Technology, June 2000.

[95] R. Hayes and S. Wheelright. *Restoring Our Competitive Edge*. John Wiley & Sons, New York, NY, USA, 1984.

[96] J. J. Heiss. Big in Japan: NTT DoCoMo Hits a Wireless Milestone with i-appli Service. Press Release, Mar. 2001. [online] `http://java.sun.com/features/2001/03/docomo.html?frontpage-headlinesfeatures`.

[97] M. Hitt, R. Ireland, and R. Hoskisson. *Strategic Management. Competitiveness and Globalization*. South-Western College Publishing, 3 edition, 1999.

[98] S. Hoff, M. Meyer, and J. Sachs. Analysis of the General Packet Radio Service (GPRS) of GSM as access to the Internet. In *IEEE International Conference on Universal Personal Communications (ICUPC '98), Florence, Italy*, volume 1, pages 415–419, 1998.

[99] H. Holma and A. Toskala. *WCDMA for UMTS Radio Access for Third Generation Mobile Communications*. John Wiley & Sons, New York, NY, USA, 2 edition, 2001.

[100] Home RF Working Group. Wireless Networking Choices for the Broadband Internet Home. White Paper. [online] `http://www.homerf.org/data/tech/homerfbroadband_whitepaper.pdf` (referred June 2001).

[101] H. Huomo. Applications are the drivers in the mobile information society. *Wireless Personal Communications*, 17(2-3):337–341, June 2001.

[102] International Organization for Standardization. Standard Generalized Markup Language (SGML). ISO 8879, Information processing, Text and office systems, 1986.

[103] International Organization for Standardization. Data elements and interchange formats. ISO 8601:2000, Technical committee 154, Dec. 2000. [online] `http://www.iso.ch`.

[104] International Telecommunication Union (ITU). Data protocols for multimedia conferencing. ITU-T Recommendation T.120, July 1996.

[105] Internet Mail Consortium (IMC). An overview of vCalendar. [online] `http://www.imc.org/pdi/vcaloverview.html` (referred Mar. 2001).

[106] Internet Mail Consortium (IMC). An overview of vCard. [online] `http://www.imc.org/pdi/vcardoverview.html` (referred Mar. 2001).

[107] Internet Mail Consortium (IMC). Internet Calendaring and Scheduling Core Object Specification (iCalendar). RFC 2445. [online] `http://www.imc.org/rfc2445` (referred Mar. 2001).

[108] Internet Mail Consortium (IMC). MIME Content-Type for Directory Information. RFC 2425. [online] `http://www.imc.org/rfc2425` (referred Mar. 2001).

[109] Internet Mail Consortium (IMC). Multipurpose Internet Mail Extensions (MIME). RFC 2045-2049. *See* [108, 111].

[110] Internet Mail Consortium (IMC). Personal Data Interchange (PDI). Specification. [online] `http://www.imc.org/pdi` (referred Mar. 2001).

[111] Internet Mail Consortium (IMC). vCard MIME Directory Profile. RFC 2426. [online] `http://www.imc.org/rfc2426` (referred Mar. 2001).

[112] P. Jantunen, T. Smura, M. Tervahauta, M. With, and F. Hacklin. 3G strategia mobiileihin yrityssovelluksiin. Course assignment (unpublished), TU-91.129, Helsinki University of Technology, Apr. 2001.

[113] N. Jefferies, J. Irvine, A. Munro, and K. Moessner. Managing increasingly software dependent mobile systems. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 247–251, Mar. 2001.

[114] J. Jing, A. S. Helal, and A. Elmagarmid. Client-server computing in mobile environments. *ACM Computing Surveys*, 31(2):117–157, 1999.

[115] A. D. Joseph and M. F. Kaashoek. Building reliable mobile-aware applications using the Rover toolkit. *Wireless Networks*, 3(5):405–419, 1997.

[116] A. D. Joseph, J. A. Tauber, and M. F. Kaashoek. Mobile computing with the Rover toolkit. *IEEE Transactions on Computers*, 46(3):337–352, Mar. 1997.

[117] R. Kalden, I. Meirick, and M. Meyer. Wireless Internet access based on GPRS. *IEEE Personal Communications*, 7(2):8–18, Apr. 2000.

[118] O. Kaljuvee, O. Buyukkokten, H. Garcia-Molina, and A. Paepcke. Efficient Web form entry on PDAs. In *Proceedings of the Tenth International Conference on World Wide Web, Hong Kong, China*, pages 663–672, Oct. 2001.

[119] S. Kalyanasundaram, E. K. P. Chong, and N. B. Shroff. An Efficient Scheme to Reduce Handoff Dropping in LEO Satellite Systems. *Wireless Networks*, 7(1):75–85, Jan./Feb. 2001.

[120] T. Kanter and H. Gustafsson. VoIP in context-aware communication spaces. In *International Symposium on Handheld and Ubiquitous Computing (HUC'99), Karlsruhe, Germany*, pages 365–367, Sept. 1999.

[121] T. Kanter and C. Olrog. VoIP in applications for wireless access. In *Proceedings of the 10th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN'99), Sydney, Australia*, pages 44–47, Nov. 1999.

[122] R. Katz. Adaptation and mobility in wireless information systems. *IEEE Personal Communications*, 1(1):6–17, Q1 1994.

[123] V. Kauppinen. Partnering as a part of the internationalization of a new technology-based firm. Master's thesis, Helsinki University of Technology, Apr. 2001.

[124] J. J. Kistler and M. Satyanarayanan. Disconnected operation in the Coda File System. *ACM Transactions on Computer Systems*, 10(1):3–25, Feb. 1992.

[125] T. Kubr, D. Ilar, and H. Marchesi. *Planen, gründen, wachsen. Mit dem professionellen Businessplan zum Erfolg*. McKinsey & Company Zürich, Wirtschaftsverlag Carl Ueberreuter, 1999.

[126] G. H. Kuenning and G. J. Popek. Automated hoarding for mobile computers. *ACM Operating System Review*, 31(5):264–275, 1997.

[127] T. Kunz, T. Barry, J. P. Black, and H. M. Mahoney. WAP traffic: description and comparison to WWW traffic. *Proceedings of the 3rd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 11–19, Aug. 2000.

[128] L. M. Ericsson AB. Ericsson unveils first embedded WAP-over-Bluetooth server. Press Release, Dec. 2000. [online] `http://www.ericsson.com/infocenter/news/embeddedwap.html`.

[129] P. Lawrence and J. Lorsch. Organization and Environment: Managing Differentiation and Integration. Report, Graduate School of Business Administration, Harvard University, Boston, MA, USA, 1967.

[130] U. Leufvén. Synchronization in a wireless world. Can SyncML handle the task? Master's thesis, Royal Institute of Technology, Stockholm, June 2001.

[131] J. Lilleberg and R. Prasad. Research challenges for 3G and paving the way for emerging new generations. *Wireless Personal Communications*, 17(2-3):355–362, June 2001.

[132] Y. B. Lin and I. Chlamtac. *Wireless and Mobile Network Architectures*. John Wiley & Sons, New York, NY, USA, 2001.

[133] Y. Liu. IP based VPN application server using Java. In *IEEE International Conference on Communications (ICC) 2000, New Orleans, LA, USA*, volume 2, pages 872–876, 2000.

[134] P. Lorenz and H. Tobiet. Location-dependent and Value Added Services (VAS) for Mobile Communications. In *1st European Conference on Universal Multiservice Networks, ECUMN*, pages 204–209, 2000.

[135] W. W. Lu. Third-generation wireless mobile communications and beyond. *IEEE Personal Communications*, page 5, Dec. 2000.

[136] L. Luna. Browser wars. *Telephony*, 240(5):74–76, Jan. 2001.

[137] J. P. Macker, V. D. Park, and M. S. Corson. Mobile and wireless internet services: Putting the pieces together. *IEEE Communications Magazine*, pages 148–155, June 2001.

[138] G. Q. Maguire. Personal computing and communication: The virtual convergence of telecom and datacom for new services. Presentation slides, XIIIth International Symposium on Services and Local Access (ISSLS), Stockholm, Sweden, June 2000.

[139] M. Mahdavi and R. Tafazolli. Analysis of Integrated Voice and Data for GPRS. In *First International Conference on 3G Mobile Communication Technologies, London, UK*, Mar. 2000.

[140] S. Mann. JavaOS: An operating system for small devices. *ACM Dr. Dobbs Journal*, (5), Jan. 1996. Article 6.

[141] M. Marchese. Study and performance evaluation of TCP modifications and tuning over satellite links. In *IEEE International Conference on Communications (ICC) 2000, New Orleans, LA, USA*, volume 1, pages 129–133, 2000.

[142] M. Marchese. Proposal of a modified version of the slow start algorithm to improve TCP performance over large delay satellite channels. In *IEEE International Conference on Communications (ICC) 2001, Helsinki, Finland*, volume 10, pages 3145–3149, 2001.

[143] J. G. Markoulidakis, G. L. Lyberopoulos, and M. E. Anagnostou. Traffic model for third generation cellular mobile telecommunication systems. *Wireless Networks*, 4(5):389–400, 1998.

[144] McKinsey & Company. Observations on wireless industry trends and implications for start-ups. Presentation slides, May 2001.

[145] C. Metz. IP-over-satellite: Internet connectivity blasts off. *IEEE Internet Computing*, 4(4):84–89, July-Aug. 2000.

[146] P. Metzger and W. Simpson. IP Authentication using Keyed MD5. *Request for Comments (RFC)*, 1828, Aug. 1995. [online] `http://www.ietf.org/rfc/rfc1828.txt` (referred Aug. 2001).

[147] N. Mirghafori and A. Fontaine. A design for file access in a mobile environment. In *Proceedings of the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA, USA*, pages 57–62, Dec. 1994.

[148] V. Möllerberg and M. Duvell. WAP-utveckling för webbföretag. Master's thesis, Royal Institute of Technology, Stockholm, Mar. 2000.

[149] Mobile Lifestreams. Yes 2 3G. White Paper, Feb. 2001. [online] `http://www.mobile3G.com`.

[150] W. Mohr. Development of mobile communications system beyond third generation. *Wireless Personal Communications*, 17(2-3):191–207, June 2001.

[151] S. Moorthy. Market segmentation, self-selection and product line design. *Marketing Science*, 3:288–307, 1984.

[152] P. Moreira. From PersonalJava to J2ME: Some Introductory Ideas, May 2001. [online] `http://www.microjava.com`.

[153] Morgan Stanley Dean Witter. Wireless Internet Report: Boxing Clever, Sept. 2000.

[154] Morgan Stanley Dean Witter. Industry Outlook: Development of the PDA Market. Presentation slides, Aug. 2001.

[155] D. S. Morrison. Mobile cast their net. *Business 2.0*, May 2001.

[156] L. Mummert and M. Satyanarayanan. Large granularity cache coherence in the Coda file system. In *Proceedings of the USENIX Summer Conference, Boston, MA, USA*, 1994.

[157] N. Negroponte. *Being digital*. Alfred A. Knopf, New York, 1995.

[158] K. Negus, A. Stephens, and J. Lansford. HomeRF: wireless networking for the connected home. *IEEE Personal Communications*, 7(1):20–27, Feb. 2000.

[159] N. Nilsson. Application services via the Internet. Master's thesis, Royal Institute of Technology, Stockholm, Mar. 2000.

[160] O. Nilsson. Fundamental limits and possibilities for future telecommunications. *IEEE Communications Magazine*, 39(5):164–167, May 2001.

[161] W. Nitzberg and V. Lo. Distributed shared memory: a survey of issues and algorithms. *IEEE Computer*, 22(4):52–60, Aug. 1991.

[162] B. Noble, M. Price, and M. Satyanarayanan. A programming interface for application-aware adaptation in mobile computing. In *Proceedings of the 2nd USENIX Symposium on Mobile and Location-Independent Computing, Ann Arbor, MI, USA*, 1995.

[163] Nokia Group. Smart messaging. Specification, June 1997. *See* [6].

[164] Nokia Group. Nokia Activ Alert Beta. User's guide, 2000. *See* [6].

[165] Nokia Group. Nokia Activ Server 2.0 Product Options. Product data sheet, Oct. 2000. *See* [6].

[166] Nokia Group. The Symbian Platform. White Paper, 2000. *See* [6].

[167] Nokia Group. How to develop Java Applications for the Nokia 9210 Communicator. Documentation, Version 1.0, Jan. 2001. *See* [6].

[168] Nokia Group. Mobile location services. White Paper, Apr. 2001.

[169] K. A. Nordström and J. Ridderstråle. *Funky Business—Talent makes capital dance*. Bookhouse Publishing, Stockholm, 1999.

[170] R. O'Hara. Microsoft Windows CE: a new handheld computing platform. In *Proceedings of the 1997 ACM symposium on Applied computing, San Jose, CA, USA*, pages 295–296, Mar. 1997.

[171] S. Ohmori, Y. Yamao, and N. Nakajima. The future generations of mobile communications based on broadband access methods. *Wireless Personal Communications*, 17(2-3):175–190, June 2001.

[172] Open Group. Calendaring and Scheduling API (XCS). Specification, Technical Standard, Apr. 1995.

[173] T. Ottmann and P. Widmayer. *Algorithmen und Datenstrukturen*. Spektrum Verlag, Heidelberg, Berlin, Oxford, 3 edition, 1996.

[174] G. Patel and S. Dennett. The 3GPP and 3GPP2 Movements Towards an all-IP Mobile Network. *IEEE Personal Communications*, 7:62–66, Aug. 2000.

[175] J. Perloff and S. Salop. Equilibrium with product differentiation. *Review of Economic Studies*, 52:107–120, 1985.

[176] A. Pigou. *The Economics of Welfare*. Macmillan, London, 1920.

[177] M. E. Porter. *Competitive Strategy*. Free Press, 1980.

[178] M. E. Porter. What is Strategy? *Harvard Business Review*, 11-12:61–78, Nov./Dec. 1996.

[179] M. E. Porter. Strategy and the Internet. *Harvard Business Review*, 3:63–78, Mar. 2001.

[180] S. Poslad, H. Laamanen, R. Malaka, A. Nick, P. Buckle, and A. Zipl. CRUMPET: Creation of user-friendly mobile services personalised for tourism. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 28–32, Mar. 2001.

[181] R. Prasad and M. Ruggieri. Future strategy for the new millenium wireless world. *Wireless Personal Communications*, 17(2-3):149–153, June 2001.

[182] PricewaterhouseCoopers. Applications for the mobile Internet. Emerging platforms. Enabling software technologies. Technology forecast: 2001-2003, Technology Centre, Menlo Park, CA, May 2001.

[183] K. E. E. Raatikainen. Middleware solution for all IP networks. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 335–340, Mar. 2001.

[184] D. Raggett. Hypertext Markup Language (HTML) 3.2 Reference Specification. W3C Recommendation, Jan. 1997.

[185] D. Raggett, A. L. Hors, and I. Jacobs. Hypertext Markup Language (HTML) 4.0 Reference Specification. W3C Recommendation, Dec. 1997.

[186] K. Railsback. Strategy: Building a wireless infrastructure. *InfoWorld*, 23(19):67, May 2001.

[187] Y. Raivio. 4G—hype or reality? In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 346–350, Mar. 2001.

[188] D. Ralph and H. Aghvami. Wireless application protocol overview. *Wireless Communications and Mobile Computing*, 1(2):125–140, Apr.-Jun. 2001.

[189] D. Ralph and C. Shephard. Services via mobility portals. In *Second International Conference on 3G Mobile Communication Technologies, London, UK*, pages 38–43, Mar. 2001.

[190] T. Richardson, F. Bennett, G. Mapp, and A. Hopper. A ubiquitous, personalized computing environment for all: Teleporting in an X Window System Environment. *IEEE Personal Communications*, 1(3):6, 1994.

[191] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, Jan./Feb. 1998.

[192] M. Riezenman. The rebirth of radio. *IEEE Spectrum*, 38(1):62–64, Jan. 2001.

[193] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, Internet Engineering Task Force (IETF), Apr. 1992.

[194] J. Räsänen. From messaging to 3G: Enabling the mobile access. Presentation at CeBIT 2001. FirstHop, Mar. 2001.

[195] J. Ryan. Hosted Voice Services: The Next Big Opportunity for ASPs. White Paper, The Technology Guide Series, The Applied Technologies Group, 2001. [online] `http://www.techpaper.com` (referred July 2001).

[196] M. Satyanarayanan. Accessing information on demand at any location. mobile information access. *IEEE Personal Communications*, 3(1):26–33, 1996.

[197] M. Satyanarayanan, B. Noble, P. Kumar, and M. Price. Application-aware adaptationfor mobile computing. *ACM Operating System Review*, 29(1):52–55, Jan. 1995.

[198] H. Shahinfar. Strategic implications of complementary wireless communication services in the consumer market, for manufacturers of handheld computing devices. Master's thesis, Massachusetts Institute of Technology, May 1999.

[199] K. Shin, D. Kandlur, D. Kiskis, P. Dodd, H. Rosenberg, and A. Indiresan. A distributed real-time operating system. *IEEE Software*, 9(5):58–68, Sept. 1992.

[200] Smartner Information Systems Ltd. Operator-Hosted Business Services for the SME sector. White Paper, Helsinki, June 2001. *See* [22].

[201] O. Sorenson. Letting the market work for you: An evolutionary perspective on product strategy. *Strategic Management Journal*, 21:577–592, 2000.

[202] P. Steemers. Critical success factors for wireless application service providers. *Global Wireless Developments*, 1, Mar. 2001. Cap Gemini Ernst & Young.

[203] G. Stone. Mobile Execution Environment (MExE). White Paper, MExE Forum, Dec. 2000. *See* [13].

[204] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.*, 13:354–356, 1969.

[205] Sun Microsystems. Enabling The Wireless Net Effect. Technical report, Oct. 2000.

[206] Sun Microsystems. Java 2 Platform Micro Edition (J2ME) Technology for Creating Mobile Devices. White Paper, May 2000.

[207] Sun Microsystems. JavaPhone API. White Paper, May 2001. [online] `http://java.sun.com/products/javaphone`.

[208] Sun Microsystems. Millions of Java technology-enabled wireless handsets to be deployed in 2001. Press Release, June 2001.

[209] N. Swartz. WAP 2.0. *Wireless Review*, 18(10):12–15, May 2001.

[210] Symbian Developer Network. EPOC Overview: Summary. Technical Report Revision 1.0(007), 1999.

[211] Symbian Developer Network. Generic Technology Overview. Technical report, 2000.

[212] Symbian Developer Network. GUI System Design. Technical Paper, Nov. 2000. *See* [24].

[213] SyncML Initiative. SyncML DTD. Specification. [online] `http://www.syncml.org/docs/syncml_represent_v10_20001207.dtd` (referred Mar. 2001).

[214] SyncML Initiative. Presentations from SyncML Supporter Summit, Dublin, Oct. 2000. [online] `http://www.syncml.org/supporters`.

[215] SyncML Initiative. SyncML Protocol, Version 1.0. Specification, Dec. 2000. [online] `http://www.syncml.org/docs/syncml_protocol_v10_20001207.pdf`.

[216] SyncML Initiative. SyncML Representation Protocol, Version 1.0. Specification, Dec. 2000. [online] `http://www.syncml.org/docs/syncml_represent_v10_20001207.pdf`.

[217] Tarantella. Tarantella Technology Gives Nokia 9210 Communicator Users Fast Access to Corporate Applications. Press Release, Mar. 2001. [online] `http://tarantella.sco.com/9210`.

[218] T. Tateoka, K. Uehara, H. Sunahara, and F. Teraoka. PFS: A File System Dynamically Adaptive to Various Networking Environments. *Computer Software*, 15(2):62–81, 1998. JSSST (in Japanese).

[219] D. B. Terry, M. M. Theimer, K. Petersen, A. J. Demers, M. J. Spreitzer, and C. H. Hauser. Managing update conflicts in Bayou, a weakly connected replicated storage system. *ACM Operating System Review*, 29(5):172–182, Dec. 1995.

[220] The Boston Consulting Group. Mobile Commerce: Winning the On-Air Consumer. Report, Nov. 2000.

[221] The Boston Consulting Group. Achieving Strategic Advantage in European business-to-business e-commerce: the Nordic Perspective. Report, May 2001.

[222] A. C. Trembly. Outsourcing IT in insurance is gaining favor. *National Underwriter*, 105(26):43–44, June 2001.

[223] G. Tyler. The ASP's bite can be fatal. *Management Services*, 45(5):22–24, 2001.

[224] UBS Warburg. European Technology Sector: Down and Nearly Out. Report, Global Equity Research, Sept. 2001.

[225] Universal Mobile Telecommunications System (UMTS) Forum. The UMTS Third Generation Market. Phase II: Structuring the Service Revenue Opportunities. Report No. 13, Apr. 2001.

[226] R. Venkateswaran. Virtual private networks. *IEEE Potentials*, 20(1):11–15, Feb./Mar. 2001.

[227] K. Väänänen-Vainio-Mattila and S. Ruuska. Design: Designing mobile phones and communicators for consumer needs at Nokia. *ACM Interactions*, 6(5):23–26, 1999.

[228] H. J. Wang, B. Raman, C.-N. Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J. S. Shih, L. Subramanian, B. Y. Zhao, A. D. Joseph, and R. H. Katz. ICEBERG: An Internet Core Network Architecture for Integrated Communications. *IEEE Personal Communications*, pages 10–19, Aug. 2000.

[229] G. S. Welling. *Designing adaptive environment-aware applications for mobile computing*. PhD thesis, The State University of New Jersey, Oct. 1999.

[230] Wireless Application Protocol (WAP) Forum. Wireless Application Protocol Architecture. Specification, Apr. 1998. *See* [29].

[231] Wireless Application Protocol (WAP) Forum. WAP Push Access Protocol. Specification, Nov. 1999. *See* [29].

[232] Wireless Application Protocol (WAP) Forum. *WAP Push Architectural Overview*, Nov. 1999. *See* [29].

[233] Wireless Application Protocol (WAP) Forum. WAP Push OTA. Specification, Nov. 1999. *See* [29].

[234] Wireless Application Protocol (WAP) Forum. Wireless application protocol. White Paper, June 2000. [online] `http://www.wapforum.org/what/WAP_white_pages.pdf`.

[235] World Wide Web Consortium (W3C). Comparison of SGML and XML. Note, Dec. 1997. [online] `http://www.w3.org/TR/NOTE-sgml-xml-971215`.

[236] World Wide Web Consortium (W3C). Compact HTML for Small Information Appliances. Note, Feb. 1998. [online] `http://www.w3.org/TR/1998/NOTE-compactHTML-19980209`.

[237] World Wide Web Consortium (W3C). WAP Binary XML Content Format. Note, June 1999. [online] `http://www.w3.org/TR/wbxml`.

[238] World Wide Web Consortium (W3C). Document Type Definition (DTD), Extensible Markup Language (XML) 1.0. Recommendation, second edition, Oct. 2000. *See* [239].

[239] World Wide Web Consortium (W3C). Extensible Markup Language (XML) 1.0. Recommendation, Second Edition, Oct. 2000. [online] `http://www.w3.org/T/REC-xml`.

[240] World Wide Web Consortium (W3C). Extensible Stylesheet Language (XSL) Version 1.0. Candidate Recommendation, Nov. 2000. [online] `http://www.w3.org/TR/2000/CR-xsl-20001121/`.

[241] G. Wu, R. Miura, and Y. Hase. A broadband wireless access system using stratospheric platforms. In *IEEE Global Telecommunications Conference (GLOBECOM) 2000, San Francisco, CA, USA*, volume 1, pages 225–230, Dec. 2000.

[242] R. Younglove. Virtual private networks—how they work. *Computing & Control Engineering Journal*, 11(6):260–262, Dec. 2000.

[243] Y. Zhou, L. Iftode, J. P. Sing, K. Li, B. R. Toonen, I. Schoinas, M. D. Hill, and D. A. Wood. Relaxed consistency and coherence granularity in DSM systems: a performance evaluation. In *Proceedings of the 6th ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming, Las Vegas, NV, USA*, pages 193–205, 1997.

[244] J. Zobel. *Mobile Business und M-Commerce. Die Märkte der Zukunft erobern*. Carl Hanser Verlag, München, Wien, Feb. 2001.