1

Introduction

1.1 Mobile Middleware

Mobile devices are increasingly dependent on good software and ultimately good user experience. In order to be able to rapidly develop good software in this operating environment that is very different from desktop computing, a lot of support services are needed. Typically, these are provided in the *middleware layer*, which is lower than applications, but above the operating system and basic TCP/IP protocol stack. Thus middleware provides a level of indirection and transparency for applications. By providing commonly used services using standardized or well-known interfaces, application developers save time, and ultimately cost, when developing their products. To consider some examples, websites and mashups, industrial systems, banking systems, and stock market systems rely extensively on middleware.

This development time and cost has traditionally been high for mobile applications and services, because the environment is more challenging than the typical fixed network environment. Namely, the wireless and mobile environment is not reliable, has long latency, small bandwidth, and there are many different terminal types. Thus one specific implementation is not necessarily usable on all mobile terminals and phones on the market. This motivates the development of mobile middleware solutions that abstract many of the issues pertaining to the operating environment for the developers and that supports adaptability to the current devices and operating environment.

Mobile middleware has evolved by leaps and bounds during recent years. From the client system viewpoint, developments such as the *Java 2 Micro Edition (Java ME)*, have enabled the creation of custom software for phones and other small devices that can be deployed at runtime. In addition to Java technologies, browser-based technologies have also evolved, and both thin browsers and rich browsers are available. A more recent advance has been the introduction of mobile web servers, thus allowing people to provide resources on their mobile devices. Today's systems are not very flexible in terms of the operating environment and usage context. Indeed, it is expected that forthcoming mobile applications are more asynchronous in nature, *context-aware*, and cope better with the different usage environments available.

Mobile Middleware Sasu Tarkoma

1.2 Mobile Applications and Services

We characterize mobile applications and services by dividing them into four generations. This allows us to inspect the evolution of the mobile application landscape and outline some of the significant trends.

Early mobile phones in the 1980s did not support applications at all, but provided only the basic voice services. The first generation mobile applications and services, introduced around 1991, were restricted by technology. The two key enablers for applications were mobile data and the *Short Message Service (SMS)*. Indeed, the role of SMS has been paramount in the path towards mobile services. The iMode has also been instrumental in Japan for mobile applications, combining low-cost data and messaging.

The second generation of mobile applications has been supported by built-in browsers, such as the *Wireless Application Protocol (WAP)* browsers, and more recently light-weight web browsers, such as the ones on available on Nokia series 60 devices and the Apple iPhone. The second generation introduced a new messaging service around 2001 that supports multimedia content messages, namely the *Multimedia Messaging Service (MMS)*. MMS allows users to compose messages that include images, audio, and video content.

The third generation applications are supported by a more sophisticated environment than just basic messaging and data services, and limited browsing, as in the case of the previous generations. The third generation applications and services are built on top of a platform that offers various services, such as location support, content adaptation, storage, and caching, to name a few well-known support services. We are now witnessing the emergence of such third generation applications and example platforms include Series 60, J2ME, Android, and iPhone. Indeed, these new devices are able to support middleware and more complex applications as *smartphones*.

A web browser that has been adapted or created for mobile devices is typically called a *microbrowser* if the offered feature set is considerably limited due to device constraints. The feature set needs to be limited due to the processing power and display capabilities of various devices. Since 2006, an increasing number of mobile devices have appeared on the market that are capable of supporting mobile web browsers with advanced features, such as CSS 2.1, JavaScript, and *Asynchronous Javascript (AJAX)*. Indeed, the previously two separate worlds of wireless telecommunications and the Internet can be seen to be on a converging evolutionary paths.

The fourth generation of applications has not yet arrived and we briefly sketch the expected properties of these applications in the light of recent proposals in research and standardization communities. The fourth generation is expected to be adaptive not only in terms of application behaviour and content, but also regarding the networking stack and air interface. Always-on connectivity, multi-mode communications, mesh networking, and adaptive use network interfaces and physical communication medium can be said to be important parts of future mobile computing devices [1].

The following list summarizes the evolution of mobile applications and services with approximate dates for the generations:

- *1st (1990–1999).* Text messages (SMS) and mobile data. Speeds up to tens of Kbps.
- *2nd (1999–2003).* Limited browsers, WAP, iMode, and MMS. Speeds up to 144 Kbps.

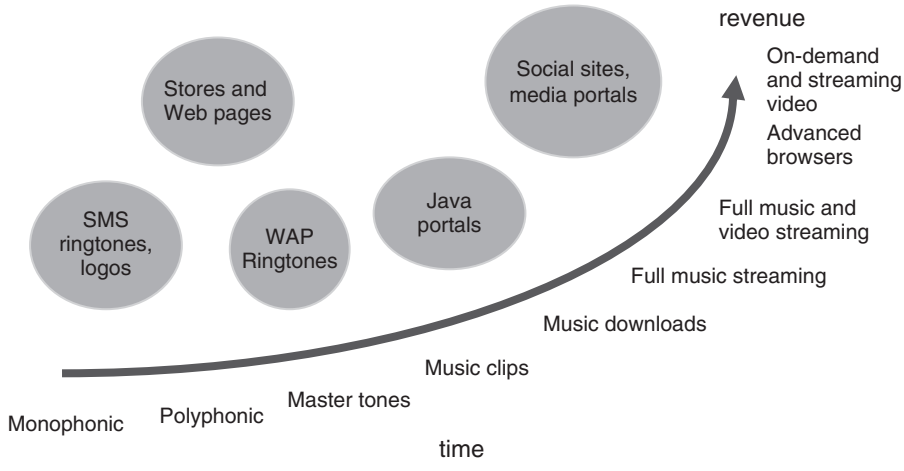

Figure 1.1 Evolution of mobile content

- *3rd (2003–2008).* Mobile platforms, middleware services. Series 60, J2ME, Android, iPhone. Speeds up to several Mbps.
- *4th (2008–).* Adaptive services, user interfaces, and protocols. Context-awareness, always-on connectivity. Speeds up to hundreds of Mbps.

Figure 1.1 illustrates the evolution of mobile content. Early mobile content was based on short text messages, namely SMS, ringtones, and logos. The early ringtones were monophonic and purchasable using SMS messages. Later monophonic ringtones were replaced with polyphonic tones and then music clips. More recent developments include storing more and more audio and video on the mobile devices and sharing it with other devices. This is typically supported using portals and other web services. The current trend is towards full music and video streaming over the Internet, and Video-on-Demand with the emergence of massively popular sites such as YouTube.

1.3 Middleware Services

Figure 1.2 illustrates the layered nature of the communications stack of today's computing devices. The network layer, namely the *Internet Protocol (IP)*, acts as the common technology for interoperability. Indeed, the TCP/IP protocol is said to have an hourglass shape due to this reliance on IP. In this book, we make the assumption that middleware and applications are built on top of the TCP/IP stack. Given the current dominant role of IP in networking and its emerging role in telecommunications systems, this appears to be a very reasonable assumption.

The four-layer TCP/IP has gained a somewhat fuzzy layer called middleware, which consists of various support services and APIs for higher level applications. Middleware typically includes services such as messaging and *Remote Procedure Call (RPC)* facilities, resource discovery, transactions, security, directory, and storage services.

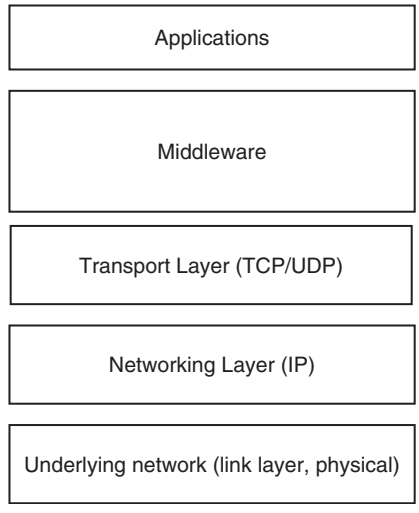

Figure 1.2 Layered network architecture

The basic communications middleware typically includes the following four services:

- *Messaging service*, which allows entities to send and receive one-way messages. The messaging service is a crucial part of a subset of middleware, called *message-oriented middleware*. This service is used to realize more complex interactions, for example the other three services. UDP can be seen as the primitive message service provided by the TCP/IP stack.
- (RPC), which allows the invocation of a remote procedure as if it were local. This requires the marshalling and unmarshalling of the request parameters and return values.
- *Remote Method Invocation (RMI)*, which allows the call of a remote method as if it were local. RMI is the Java way of doing RPC and requires the use of distributed garbage collection to remove unnecessary state in the distributed system.
- *Event service*, which allows entities to receive asynchronous notifications. A notification is the result of some observable event, and results in a possible state change in the listener. Thus, events are a basic building block for adaptive distributed systems.

We observe that the mobile environment, due to its intermittent connectivity and reachability challenges, presents significant challenges for data synchronization. Given that mobile devices have increasing amounts of volatile and non-volatile storage, it becomes challenging to keep data items synchronized over the networks. Therefore, data synchronization service can be seen as a basic service that must be included in a mobile middleware solution.

From the server-side viewpoint, we are currently witnessing the convergence of telecommunications technologies, such as the *Session Initiation Protocol (SIP)*, used in *Beyond 3G networks (B3G)*, and websites and services. Middleware plays a crucial role in this converged communications environment. This server-side part of mobile middleware facilitates the development, deployment, and execution of services for mobile devices.

1.4 Transparencies

Middleware provides various transparencies for higher layers that are not typically supported by the lower layers. These transparencies include location, transport, operating system, programming language, and failure transparencies. In the following, we briefly summarize the transparencies typically provided by middleware.

Location transparency abstracts network attachment points of communicating entities from applications. This allows applications to communicate with each other without knowing the locations of the destinations. A typical example of this is the host name, which abstracts the IP address, the topological location of an entity in the Internet, from the application.

Transport protocol transparency means that a given interaction can utilize any transport protocol suitable for the task. *Transparency of the operating system and programming language* means that external data representation is used to describe interactions and any data associated with them. Language and operating system independent description of both procedures and data types is needed. External data representation is crucial for supporting software for multiple operating environments and ensuring communications between them.

For example, web servers have been implemented in many different environments ranging from small embedded devices and mobile phones to large-scale clusters. The *Hypertext Transfer Protocol (HTTP)* protocol standardized by the *Internet Engineering Task Force (IETF)* ensures ubiquitous access to web resources. Hence, standards play a very important role in data communications. HTTP has become the de facto protocol for middleware and application communications due to its central role in web access, simple specification, and firewall friendliness. In addition to HTTP, the *Extensible Markup Language (XML)* standardized by the *World Wide Web Consortium (W3C)* has become the universally accepted data interchange format.

1.5 Mobile Environment

The mobile environment is radically different from the traditional fixed-network environment, which typically has low latency, high bandwidth, and reliable communications. The mobile environment is subject to disconnections and reachability problems. Moreover, mobile devices typically have multiple interfaces for communication, limited battery, limited processing capability, and limited memory.

The capabilities and limitations of the device dictates the constraints on how the user is able to access the services and what kind of content is provided for the user. The capabilities may be divided into two categories: *computational capabilities* and *interface capabilities*. The computational characteristics define what kind of resources the client device has in terms of providing services. Some devices are only capable of acting as an interface to the services running in the network node (e.g. web browsers), whereas more capable devices may run services or parts of the services by themselves. The interface capabilities dictate the characteristics of how the service is displayed to the end-user. For instance, a mobile phone has limited ways of showing high-end images and video, whereas a tablet computer is less limited in this respect.

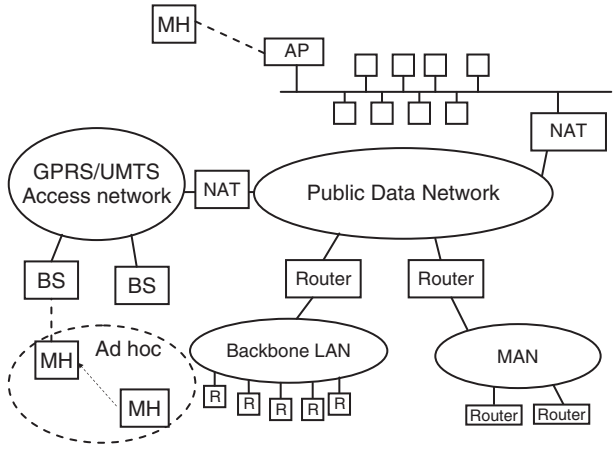

Figure 1.3 The mobile and wireless environment

Figure 1.3 presents an overview of the mobile and wireless communication environment. The environment consists of a number of distinct communications scenarios, including:

- fixed-network communications;
- communications with a wireless access point; and
- wireless ad hoc communications.

We define the central elements of the environment as follows: *MH* denotes the Mobile Host, *BS* is a wireless Base Station that provides connectivity to MHs, an Access Network is a network that provides connectivity for MHs and connects a set of BSes. *AP* denotes an Access Point, which provides connectivity for MHs. We refer to BSes as part of a mobile access network, such as *Global System for Mobile (GSM)*, *General Packet Radio Service (GPRS)*, or *Universal Mobile Telecommunications System (UMTS)*, and an AP as part of a wireless *Local Area Network (LAN)*. Different access networks and other networks are reachable over the Internet backbone. In the figure, *MAN* denotes Metropolitan Area Network and *R* a single router.

We briefly summarize the state of the art wireless technologies that have been deployed today. The phenomenally successful 2G digital cellular networks have been extended with 2.5G data services such as GPRS and EDGE, which are now being widely used in Europe and Asia. After a slow start, 3G networks and services are now available with the data rates from hundreds of kilobytes to several megabits per second. A significant trend has been toward all IP access networks. Indeed, it is expected that Beyond-3G systems will be based on IP technologies.

In addition to cellular technologies, 802.11 *Wireless LAN (WLAN)* has proved to be hugely popular in the residential market and in the enterprise sector, despite security concerns. WLAN offers typically 11Mbit/s and 54Mbit/s data rates. *Worldwide Interoperability for Microwave Access (WiMax)* is another example of wireless LAN technology. WiMax is an industry forum that develops point-to-point wireless access standards for

home and mobile users. WiMax offers much larger bandwidth than 802.11 of up to 70 Mbit/s.

Now, given the popularity of WLAN access points and the cost of building a cellular data access, some companies are building WLAN access networks on a peer-to-peer basis. These mesh networks are currently an active topic in research and development, and may well be part of future wireless access solutions.

In addition to long-range technologies, *radio frequency identification (RFID)* and *Wireless Personal Area Network (WPAN)* are examples of short-range systems. An RFID tag is a small integrated circuit that can respond to a signal with some data. RFIDs are extensively used to track goods and people. RFIDs can be passive and only react to external interrogating signals, or they can be active and send beacons. Passive RFIDs do not require a battery whereas active ones do. WPAN is another example of short-range wireless communications. WPAN is a collection of short-range, low-power technologies, including infrared, Bluetooth, and ultra-wideband (UWB) technologies.

1.6 Context-awareness

Context information is important for applications that need to adapt to different situations [2]. To be context-aware, a system must gather contextual information from the surrounding environment. This information can then be used to support adaptability and ultimately better user experience.

There are many ways to define context and context-awareness. A. Dey states context is any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a use and an application, including the user and applications themselves. Primary context types include location, identity, time and activity that characterize the situation of a particular entity.

1.7 Mobility

Wireless devices are not stationary and may move from one network to another. This kind of *terminal mobility* requires special protocol support both on the mobile device, and in the infrastructure. Another form of mobility, called *user mobility*, happens when a user relocates and switches devices. The user should be able to use the same services irrespective of the time, location, and device. This form of user mobility also requires support from the distributed system. The notion of mobility can be taken further and considered as *logical mobility*. Logical mobility pertains to changes in user interests, context, and other attributes defined in a multi-dimensional space that does not necessarily have a one-to-one mapping with the physical world. We shall return to these different notions of mobility later in this book.

Typically networks are separated by firewalls and *Network Address Translation (NAT)* devices. NAT devices support the separation of network address spaces by allowing private IP addresses which are then mapped to one or more public IP addresses. Only public IP addresses are globally routable. This means that if a communication path has one or more NAT devices, the end points of a communication may not necessarily be able to communicate with each other. Indeed, *NAT traversal* has become an active area of

standardization at IETF. The goal of NAT traversal is to detect the presence of NATs and then utilize different techniques depending on the environment to ensure end-to-end connectivity. We shall return to the issue of reachability in the following chapter.

1.8 Example Use Case

The Information Society and Technology Advisory Group's (ISTAG)[1] future ambient intelligent scenarios illustrate future ubiquitous environments.

To give an example, we consider the first scenario, namely 'Maria – The Road Warrior'. In this scenario the user, Maria, is on a business trip and arrives at a foreign country. Maria is carrying a personalized communications device, the *P-Com*, which automates various tasks during her travels. For example, from the airport to the hotel: her visa is automatically checked at the immigration, her car rental is arranged beforehand, the P-Com recognizes and personalizes the rental car for Maria. Real-time traffic instructions are provided by the car and P-Com during the drive to the hotel. Finally, the hotel room facilities are customized for Maria by interactions between the P-Com and the hotel room computing infrastructure.

All of these above tasks require the knowledge of the user preferences and the computing and networking infrastructure at a given time and in a particular location. The networking environment is heterogeneous and may consist of various technologies, such as Wireless LAN, infrared, Bluetooth, 2G and 3G, Wibree, and WiMax. WLAN, 2G, 3G, and WiMax are examples of long-range technologies, whereas Bluetooth and infrared are short-range technologies.

For example, at the airport there may be WLAN hotspots available, and at the immigration there could be local Bluetooth coverage. During the walk from the airport arrival hall and at the car garage a 3G network is available, and once Maria enters the car, there could be a Bluetooth network specific to the car. During the drive to the hotel, traffic information is delivered using a 3G network. At the hotel, the hotel room may contain a WLAN hotspot to control room facilities and to access the Internet. In order to make all the different network infrastructures transparent to Maria, the P-Com can make use of the wireless network and location information, and based on these, change seamlessly from one network to another.

The scenario points out several notable characteristics of the ubiquitous environment including the presence of multiple and possibly overlapping wireless networks, seamless roaming between the access networks, different kinds of client terminals and network elements, location awareness, context awareness, data and content centric communications, and personalization.

We also observe that the interactions are user-centric and the goal of the distributed system is to support the user and anticipate user requirements. The interactions are also inherently context-dependent and content-centric. Maria is not interested in knowing which Web server is providing certain information, she as a user is only concerned that she receives the relevant content, for example driving instructions.

Compelling usage scenarios can also be found in the Wireless World Research Forum's Book of Visions [3, 4].

[1] ISTAG is the advisory body to the European Commission in the area of ICT.

1.9 Requirements for Mobile Computing

Mobile computing consists of many facets. As we have discussed already, the wireless and dynamic nature of the environment presents many challenges for application and service developers [5].

Mobile computing can be seen as overlapping with several related areas of computing, namely pervasive and ubiquitous computing. They are based on Mark Weiser's vision of ubiquitous computing [6]. The basic setting of pervasive and ubiquitous computing revolves around interactions between distributed sensors and other devices with computational capability. The devices are embedded and seamlessly integrated with the physical environment. One way to distinguish these areas is to consider the goal of mobile computing to offer content to users 'anytime, anywhere', and the goal of pervasive and ubiquitous computing to assist users 'all the time, everywhere'. Therefore the former is a more reactive and the latter a more proactive approach.

We briefly outline the different users of middleware:

- *End user.* The goal of middleware is not to directly interact with the end users, but rather support the applications and services that are visible to the users. This means that middleware should provide sufficient APIs and mechanisms to cope with different kinds of failures and faults, and in general support enhanced usage experience.
- *Device Manufacturers.* Device manufacturers use middleware in order to provide extended features that interface with device drivers.
- *Internet Service Providers.* Internet service providers utilize middleware to monitor and administer the network.
- *Platform Providers.* Platform providers develop middleware platforms that integrate with different operating systems.
- *Application Service Providers.* Application service providers utilize middleware in order to facilitate application development and deployment in a scalable and secure manner.

In this section, we briefly consider the important facets of mobile computing and highlight the central requirements for mobile middleware. We consider the following five non-functional requirements as key elements of mobile computing, namely:

- accessibility;
- reachability;
- adaptability;
- trustworthiness; and
- universality.

These non-functional requirements do not stand alone, but rather they are intertwined. Accessibility means that resources are available and accessible for end users irrespective of the current location and where the resource is located. This has many implications for the protocol stack, middleware, and applications. To be able to ensure accessibility, external data representation is needed. Furthermore, the desired resources must be discovered and located in the network before any access can happen.

Reachability is needed to ensure that resources are available in any location. In order for resources to be available for global access on mobile devices, they need to accessible

and reachable. Reachability cannot be taken for granted in today's heterogeneous and dynamic environment depicted in Figure 1.3. As mentioned in this chapter, firewalls and NAT devices pose grave connectivity challenges. Moreover, mobility complicates reachability and requires special solutions in order to ensure the reachability of mobile devices.

Adaptability is a crucial requirement for this environment, because the environment is subject to changes, for example due to physical, logical mobility, or some other change in the environment. The nature of data access is inherently related to adaptability. A mobile device may not have the processing capability to handle all incoming data, or the data may not be presentable to a user without first adapting it to the current display and mode of user interaction. Adaptability implies that the mobile entity, either alone or assisted by some other entities, can monitor these changes in the environment, and then react accordingly and reconfigure the system. Therefore, context-awareness is an essential part of mobile computing. Adaptability also implies reflectivity, namely the ability of the applications to query and modify different parameters and context attributes at runtime.

From the security point of view, the mobile environment poses a number of challenges that must be addressed by the applications, middleware, and the protocol stack. We are faced with questions concerning the authenticity and integrity of data being accessed, and the authenticity of a network entity. There are many ways that malicious entities can disrupt, eavesdrop, and hinder data communications. Well-known examples of attacks include *man-in-the-middle attack*, *impersonation*, *data injection*, and various *Denial of Service (DoS)* attacks. Protocols that are engineered for the mobile environment to support the special requirements of the environment, for example mobility and multi-homing, must not introduce new security problems.

Trust is a challenging notion for distributed systems. An entity trusts another entity if it believes that the other entity follows a contract which is shared by the parties. Before any trust in external entities can be established, the applications must be able to trust the middleware and protocol stack. The notion of a contract is evident in the typical communications setting in which a user pays for a network operator for the services rendered, for example Internet data access. The notion of trust becomes complicated when there are no explicit contracts between the entities, which is typical of decentralized communication environments. Therefore, various security techniques are needed to build and maintain trust.

Security features can be realized on different layers of the protocol stack. Solutions exist for the link layer, network layer, transport layer, middleware, and applications. One current challenge in security is that these different security solutions are independent of each other, and applications do not have mechanisms to determine whether or not a lower layer security solution is being used. In practice, this means that multiple security solutions are used at the same time. The goal of mobile middleware is to provide security services for applications that enable end-to-end and end-to-middle security. These security solutions can then be used by applications to determine the trustworthiness of entities and data items.

Universal data access is one of the key reasons for the success of the Internet. Resources on the Internet are accessible from anywhere, at anytime, using standard protocols and formats. Therefore, universality is also a crucial requirement for mobile computing. This implies that standardized protocols and formats are used to discover and access resources.

Convergent communications, in which telecommunication solutions, such as SIP, and web protocols are used seamlessly, is an example of this requirement in action. Indeed, although a number of dedicated protocols have been developed for mobile data access, the current trend is to utilize web protocols also in the mobile environment. This allows mobile entities to be joined directly as part of the Internet ecosystem rather than a new ecosystem for mobile users being created.

Scalability is implied by the notion of universality. A network system needs to attain critical mass in order to sustain the service ecosystem. Therefore massive scalability is needed before universality can be achieved. Scalability can be perceived in many ways, for example the number of nodes that are supported by the network and the maximum number of messages that can go through a system. Currently there are over three billion mobile phones in the market and the expectation is that the five billion mark will be passed in recent years. The network therefore must be able to support these five billion, or more, devices and able to provide the requested services according to agreements with users. This ensures a crucial role for middleware to be able to support this requirement for scalability both in the device systems and on the server-side.

1.10 Mobile Platforms

Mobile middleware aims to support the development, deployment, and execution of distributed applications in the heterogeneous and dynamic mobile environment. The goals for mobile middleware include adaptability support, fault-tolerance, heterogeneity, scalability, and context-awareness.

The presented non-functional requirements pose significant challenges for software. The industry solution to these challenges has been to create middleware *platforms*. A platform collects frequently used services and APIs under a coherent unified framework. In this book, we consider state of the art mobile platforms including Java ME, *Open Mobile Alliance (OMA)*, Symbian, .NET, Android, and Apple's iPhone. These platforms have many common elements and they follow similar design principles and patterns. In the following chapter, we focus on these platforms and then analyze their commonalities and differences.

1.11 Organization of the Book

In this chapter, we examined the motivation for mobile middleware and considered some important properties of the wireless and mobile environment. The need for supporting easy service development, deployment, and execution has resulted in a fuzzy layer of middleware solutions that are above the operating system, but below the end-user applications. This book aims to present a synthesis of mobile middleware and how it is motivated and positioned with the current Internet protocol stack. We also investigate mobile applications and services and consider their requirements towards the middleware and the network.

Chapter 2 presents key mobile middleware architectures and platforms. We start with a brief overview of the current TCP/IP protocol suite and some of the current networking challenges. Then the key components of middleware architectures and platforms are covered, including objects, components, services and communication mechanisms. We take the Java platform and a recently proposed SPICE platform as examples of large platforms.

The mobile platforms that are examined include Java ME, iPhone, Symbian and Series 60, BREW, WAP, .NET Compact Framework, NoTA, Maemo, and Android. We highlight some of the similarities and differences between the platforms.

Chapter 3 presents a set of important support technologies that are the building block for more complex middleware solutions. We start by giving a brief summary of the SIP architecture and protocol, which is a fundamental protocol for current IP-based voice services, and it is becoming an enabler for other types of services as well. After examining SIP and the IMS system, we continue with web services. Web services and issues pertaining to web service integration are crucial for current and forthcoming mobile services. We consider the different facets of web services and compare them with a more lightweight approach called REST. Other considered technologies include SQLite and OpenGL. Then we focus on service discovery techniques, which are needed to realize information accessibility. We consider UPnP, Jini, SLP, and ZeroConf. One of the aims of mobile middleware is to support mobile terminals and users. The SIP architecture has intrinsic support for different kinds of mobility. In addition to SIP, we consider Mobile IP and Host Identity Protocol for host mobility and Wireless CORBA for object mobility. The advanced topics discussed in this chapter include overlay networks, context-awareness, service composition, security and trust, and charging and billing. Finally, we draw some of the discussed topics together by presenting an example of middleware platfom.

Chapter 4 presents a number of principles and patterns pertinent for middleware and mobile middleware. We start by discussing different distributed system design principles, including Internet, Web, SOA, Security, and mobile principles. After this, a number of crucial architectural patterns are examined. The patterns are presented briefly and the goals and liabilities are summarized. After the architectural patterns, some general patterns are considered before presenting a family of patterns for mobile computing. We focus on patterns pertaining to mobile communications and synchronization.

Chapter 5 presents an overview of relevant standards and motivates interoperability between specifications and implementations. We discuss standardization process in general and examine the key standards organizations from the mobile middleware and computing viewpoint. Key wireless communication standards and middleware standards are briefly mentioned. Some interesting emerging standards are also considered.

In Chapter 6 we consider mobile messaging, which is one of the core middleware services in today's systems. The examined solutions include the web services protocol stack, namely the SOAP protocol, REST, and *Java Message Service (JMS)*. We discuss techniques to optimize messaging for the mobile environment and pay particular attention to push technologies. Message push solutions are needed in order to realize universal data access.

We continue the discussion on asynchronous communication in Chapter 7, which focuses on the *publish/subscribe (pub/sub)* paradigm. Pub/sub is an emerging solution for the mobile and pervasive environment, in which the communication is based on current demand and supply of data. We examine the commonly used information dissemination topologies and consider how middleware can support information delivery based on supply and demand of data. The relevant standards include SIP event package, JMS, and the *Data Distribution Service (DDS)*. We also discuss issues pertaining to mobile data subscribers and publishers, and consider some advanced topics in this area.

Chapter 8 examines a number of data synchronization techniques especially for the mobile environment. We start with an overview of data synchronization, and then proceed to the SyncML standard, which is implemented on most current high-end mobile phones. We also consider other systems including Coda and Unison.

Chapter 9 focuses on security issues and we start with an overview of basic security principles and then consider cryptographic operations and public key infrastructures. This chapter considers security solutions needed on multiple layers of the protocol stack, including link, network, transport, and application layers. We briefly introduce the core security solutions for 3G and Beyond-3G networks, namely AAA, RADIUS, and Diameter. And then move on to the use of security tokens in mobile service access, and how security tokens can be used to realize single-sign on solutions. Finally, towards the end of the chapter, we consider web services security, and the security implications of downloaded active objects and code.

Chapter 10 presents a number of applications and service case studies. We start this chapter with a general overview of mobile applications and services, and compare the Internet and mobile application development processes. We then continue to examine several important use cases and application domains, including Mobile Web Server, mobile advertisement, push email, video delivery, mobile widgets, and airline services. We discuss how the mobile patterns described in Chapter 4 are used in these examples.

Finally, we conclude the book in Chapter 11 with a discussion and outlook for mobile middleware.

Bibliography

[1] Raatikainen, K., Christensen, H. B. and Nakajima, T. 2002 Application requirements for middleware for mobile and pervasive systems. *ACM SIGMOBILE Mobile Computing and Communications Review* **6**(4), 16–24.

[2] Dey, A. K. 2001 Understanding and using context. *Personal Ubiquitous Comput* **5**(1), 4–7.

[3] (ed. David, K.) 2008 *Technologies for the Wireless Future: Wireless World Research Forum (WWRF)* vol. **3**. Wiley.

[4] (ed. Tafazolli, R.) 2006 *Technologies for the Wireless Future: Wireless World Research Forum (WWRF)* vol. **2**. John Wiley & Sons Ltd.

[5] Satyanarayanan, M. 1996 Fundamental challenges in mobile computing *PODC '96: Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing*, pp. 1–7. ACM, New York, NY, USA.

[6] Weiser, M. 1993 Ubiquitous computing. *Computer* **26**(10), 71–72.