Abstract

The ongoing European ACTS project *OnTheMove* provides support services for distributed mobile multimedia applications. The project defines, implements, and demonstrates a mobile middleware called a Mobile Application Support Environment (MASE) which is based on UMTS concepts. The mobile application programming interface (mobile API) of MASE, which will be submitted for standardization, allows common access to the underlying operating systems and network infrastructure, and facilitates the development of new, mobile-aware, multimedia applications.

UMTS: A Middleware Architecture and Mobile API Approach

Birgit Kreller, Siemens AG
Anthony Sang-Bum Park and Jens Meggers, Aachen University of Technology
Gunnar Forsgren, Ericsson Radio Systems AB
Ernö Kovacs and Michael Rosinus, Sony International (Europe) GMBH

Any information at any time, at any place, in any form. This promise of mobile multimedia will be realized through third-generation mobile communication networks, which will offer high-bit-rate data services, guaranteed on-demand bandwidth, and low delays. The European Telecommunication Standards Institute (ETSI) is working on the Universal Mobile Telecommunications System (UMTS) [1, 2], which belongs to the family of similar or compatible standards developed within the International Telecommunication Union (ITU) called International Mobile Telecommunication in the year 2000 (IMT-2000) [3].

Today, mobile users already utilize a wide variety of mobile terminals ranging from simple mobile phones and personal digital assistants (PDA) to high-end multimedia notebooks. UMTS and the Mobile Broadband System (MBS) will offer suitable bandwidth and global connectivity to enable true mobile multimedia. As an early contribution to the UMTS service specifications, and in order to provide a smooth evolution path from second- to third-generation communication systems, the Advanced Communications Technologies and Services (ACTS) project *OnTheMove* has developed a mobile middleware system called the Mobile Application Support Environment (MASE). Along with the MASE, an application programming interface (API) has been defined that allows applications to access the MASE components. This API, called the *mobile API*, will be submitted for standardization. The purpose of the MASE middleware is to ease the development of mobile-aware applications by providing a common underlying platform. Furthermore, the middleware approach enables a smooth transition from current wireless networks, such as Global System for Mobile Communications (GSM) and Digital European Cordless Telecommunications (DECT) to future UMTS networks. Instead of directly accessing the operating system, mobile-aware applications make use of the *mobile* API and benefit from simple access to MASE services, which hide the complexity of heterogeneous networks and operating systems from the applications. Thus, MASE simplifies the development of mobile-aware applications and frees them from the complex processing caused by additional needs when accessing heterogeneous networks and running on mobile devices. Furthermore, MASE eases the evolution of multimedia applications toward UTMS [4] and enables legacy applications to benefit from a subset of its functionality.

In this article we focus on the services provided by MASE and their relation to UMTS. We demonstrate the interworking between different parts of MASE through a typical mobile-aware application, the CityGuide. The CityGuide uses geographical information provided by the MASE Location Manager module to display a map of the current geographical surroundings of the mobile user. Several different layers of interesting places (e.g., public transportation, administration buildings, accommodation, restaurants, and much more) are shown on the map and are linked to corresponding Web pages.

We will first elaborate on the generic MASE architecture. We will outline a possible scenario for the ongoing UMTS service definitions and then explain, step by step, the MASE components accessed by the CityGuide implementation.

MASE

The Mobile Application Support Environment is a distributed system that runs on both the mobile device and the so-called mobility gateway. The latter acts as a mediator between the wireless and fixed network infrastructures. It works as an agent for mobile clients which are typically connected over unreliable wireless access networks with low bandwidth. MASE enables access to the UMTS adaptation layer (UAL), which provides applications and middleware components unified access to all possible underlying networks. An additional general support layer provides the functionality required for distributed systems. On top of both layers several manager components are installed, providing different dedicated services. Figure 1 shows the overall MASE architecture with its corresponding building blocks. All the components shown have been implemented.

MASE is built around the concepts of awareness, adapta-

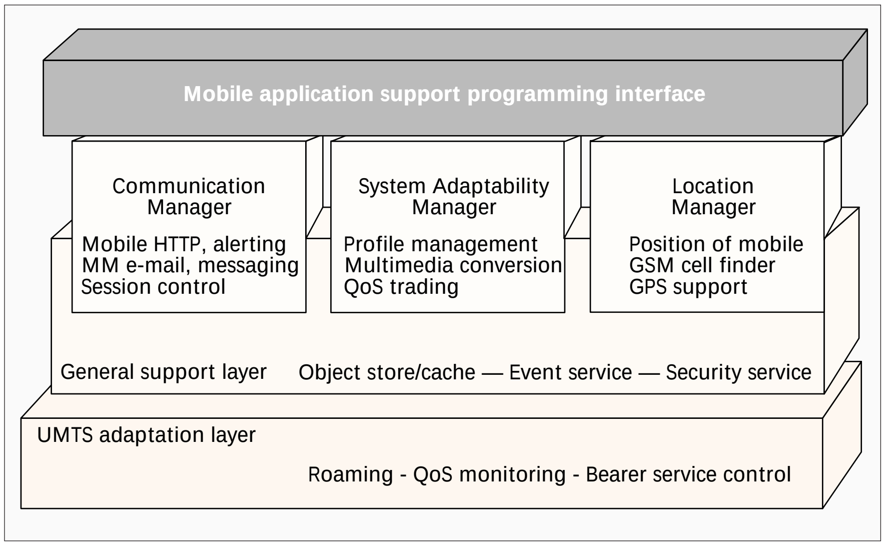

Figure 1. *Overall architecture of MASE.*

tion, and abstraction. MASE applications are aware of the current network quality of service (QoS) through sophisticated monitoring and management facilities which are provided by the UAL. The UAL hides network specific details by selecting the appropriate bearer service and protocol stack according to the requested QoS. The general functional features of the UAL are:

- Selecting appropriate transport protocols and configuring them for efficient use (e.g., adjusting packet size and timers to the bearer service parameters)
- Selecting and configuring an appropriate bearer service
- Managing roaming between different networks and bearer services as well as bearer service switching

Details of the UAL have been published elsewhere (e.g., [5]). The General support layer implements object storing and caching as well as event handling and security services. These services will also not be described in detail here.

Awareness — End terminal characteristics and user preferences are stored in profiles. They are managed by the Profile Manager, a part of the System Adaptability Manager (SAM), and are available on demand at all nodes of the system.

Adaptation — Profile information and monitored QoS are used by the MASE communication facilities to adapt their usage to the current QoS situation and user requirements. This adaptation is transparent to the application.

Abstraction — The MASE provides high-level abstractions, for example, an alerting function or location management. An alert is an abstraction of an important short message which has to be sent to the user. Depending on the current network situation, the alert manager maps an alert to different network services. If the user is connected to the network over TCP/IP, the message will be delivered directly to an alert server on the mobile device messaging service, or as a GSM short message service (SMS) notification if no such connection is available. The final version of MASE will deal with adaptation to varying degrees of QoS, robustness in the face of disconnected links, roaming between different operators and network types, personalized information filtering, and location-aware applications using various location trackers.

MASE also integrates legacy communication applications and improves communication over wireless networks. In this way, the MASE takes up the challenges of mobile multimedia and prepares for UMTS.

The CityGuide Application Scenario

The CityGuide is part of a set of mobile-aware applications (Fig. 2). It is a typical mobile user application providing access to a map of the surroundings of the mobile user. This map displays several information layers, such as hotels, restaurants, automated teller machines, bus stops, and phone booths. These information sources are linked to Web pages to allow instant access to further information about a particular location.

The CityGuide runs as a Java applet within a Web browser and provides access to maps describing the current surroundings. This application uses the *mobile* API to ask the MASE Location Manager for the actual coordinates (longitude and latitude) of the user. This information will be checked against the coordinates stored in the server. The most suitable map and the associated geographical objects will be loaded using HTTP. Since the browser is a legacy application and normal HTTP is not well suited to mobile communication, the MASE integrates these calls using an HTTP proxy system. In the following we describe the MASE components participating in this process.

Location Manager

The Location Manager (LM) helps users navigate in new environments. It enables applications and other MASE components to determine the parameters of the current geographical position of a mobile device as well as the accuracy of these values. This is another example of the abstractions provided by MASE because the geographical information is accessible through a simple uniform API and independent of the mechanism used by the LM to gather location data. A subset of the LM API calls is shown in Table 1.

The current LM implementation supports the Global Positioning System (GPS), a satellite-based radio navigation system developed and operated by the U.S. Department of Defense [6], and uses WaveLAN (a wireless LAN device from Lucent Technology [7]) cell identifier information. GPS provides latitude and longitude coordinates, velocity, and the user's moving direction with an accuracy of about 10–300 m.

Method	Description
getAccuracy	Returns the position error of the device
getLastDateOfUpdate	Returns information about last date of update of GPS location information
getLastTimeOfUpdate	Returns information about last time of update of GPS location information
getLatitude	Returns the latitude in WGS84 format
getLongitude	Returns the longitude in WGS84 format
getVelocity	Returns the speed of the device

Table 1. *A subset of the Location Manager API.*

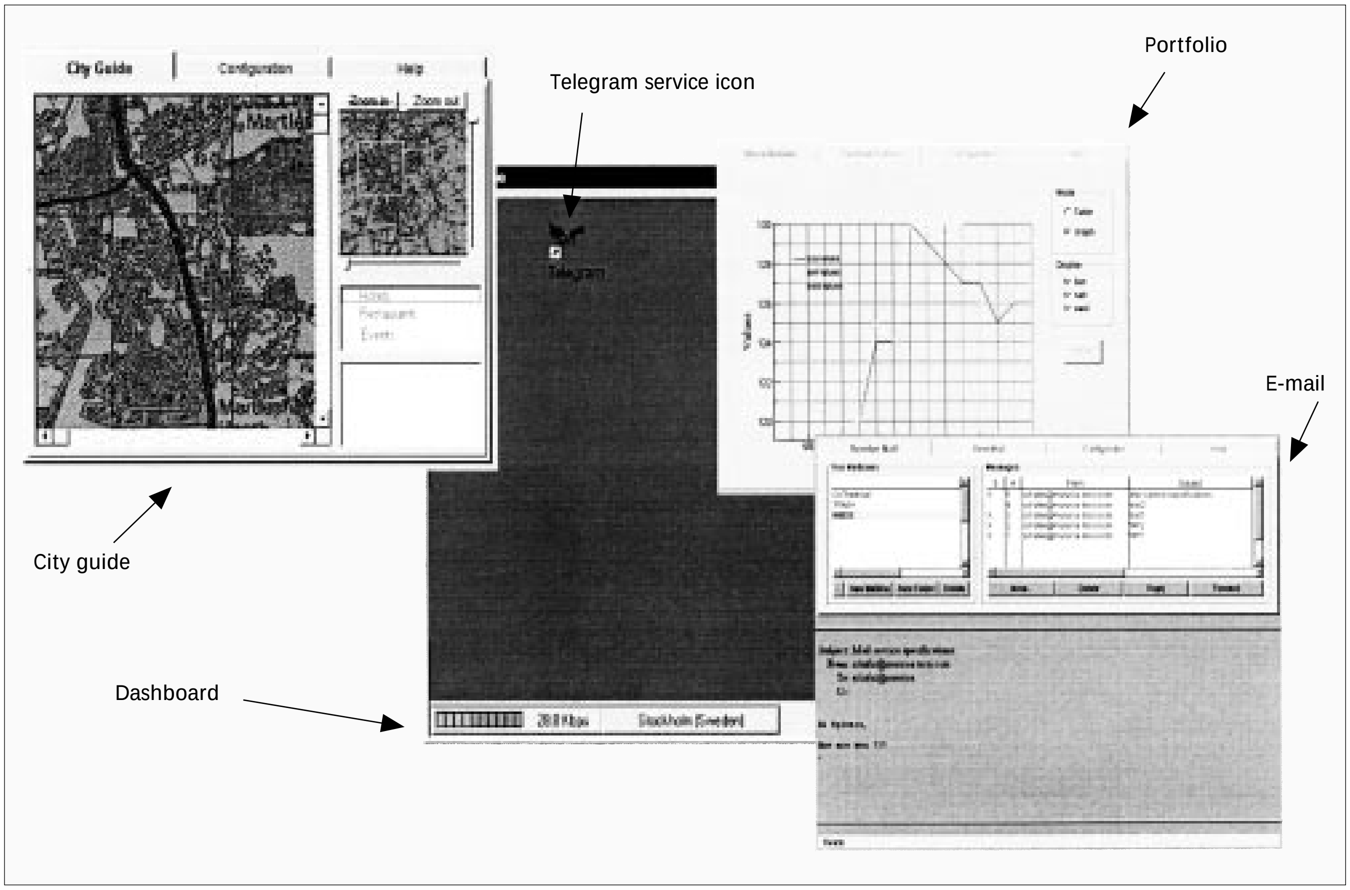

Figure 2. *OnTheMove desktop and mobile-aware applications.*

In wireless communication systems cell identifiers can help determine the position of the mobile device. The accuracy depends on the network cell size and varies from about 30 m (wireless LAN) up to a few kilometers using GSM. A table lookup maps these cell identifiers to the physical position of the terminal. A central LM process communicates with the available location information sources. The retrieved values are made available to MASE-aware applications and other MASE components by stub interfaces to the central process. This architecture is shown in Fig. 3.

Communication Manager

The Communication Manager (CM) of the MASE architecture supports HTTP and e-mail communication, HTTP prefetching, alerting, messaging, and disconnected operations. In the following we will focus on the HTTP proxy. The HTTP proxy system improves the performance and usability of HTTP over low-bandwidth network connections (cellular, radio LAN, modem). Requested multimedia objects can be processed/converted by MASE to match the current network bandwidth, terminal characteristics (display capability, storage capacity, etc.), and user requirements. This adaptation is performed at the mobility gateway before transmitting the data to the mobile terminal. The distribution of the HTTP proxy functionality is shown in Fig. 4.

Multiple users simultaneously access Web-based information using the HTTP protocol. Each mobile terminal runs a client-side HTTP proxy (CSP) that requests Web objects from the server-side HTTP proxy (SSP). SSP can serve multiple client connections in parallel, with each connection being handled by a separate connection handler thread. Each such handler communicates with a peer process on the CSP over a multiplexed logical communication session. Multiple sessions run on top of a single TCP/IP connection between a mobile terminal and the mobility gateway. The multiplexing scheme allows data transfer for all logical sessions from one terminal to be transmitted in parallel over a single full-duplex TCP/IP connection.

In a typical scenario, an HTTP client on the mobile terminal (such as the CityGuide application) will request HTML pages and related graphic objects from a remote HTTP server. The HTTP Proxy system intercepts these calls and communicates with the SAM to adapt the requested objects to the current QoS and user requirements. The application (in this case the browser) is configured to use the CSP as an HTTP proxy server.

As shown in Fig. 4, for each HTTP request the client will

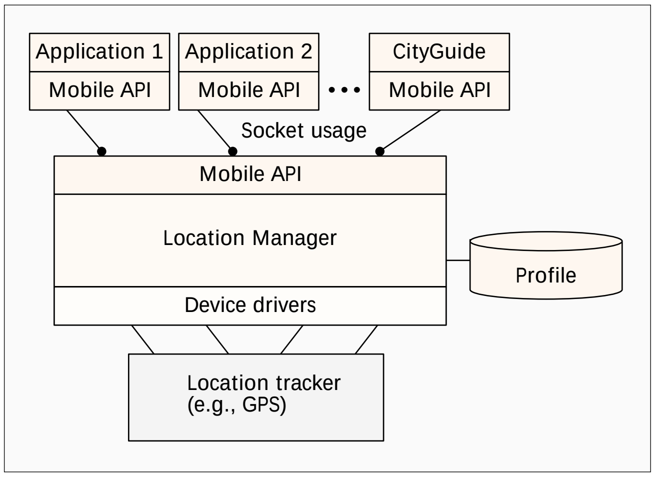

Figure 3. *Location Manager architecture.*

establish a TCP/IP connection to the CSP ①. The CSP checks whether the requested object is available locally by calling the `checkCache()` methods of the SAM ②. See Table 2 for the description of the SAM API. If the SAM indicates local availability, the object will be returned by the CSP to the requesting client.

Otherwise, the CSP will create a logical communication session to the SSP side and will pass the HTTP request to the SSP ③. Another `checkCache()` call ④ checks whether the requested object is available on the Mobility Gateway. If not, the object is fetched from the HTTP server ⑤ and inserted into the local cache using the `createMMO()` method. The SSP now calls the `trade()` method to adapt the multimedia object. This trading process is described in the next section. The SAM creates a variant of the original object and inserts it into the family of related objects managed by the local cache. The variant together with instructions for the post trading phase (e.g., which decompression method to use) is returned and transferred to the CSP. Here a post trading phase is initiated and the resulting object passed to the requesting client.

Method	Description
trade()	Performs trading
postTrade()	Performs required processing steps on the receiver side (e.g., decompression)
checkCache()	Searches cache for a local available MMObject
createMMO()	Creates an initial MMObject
insertVariant()	Inserts an MMObject variant into the family of related objects
removeVariant()	Removes variant from the family

Table 2. *A subset of the QoS trader API.*

user.image.reductionAllowed user.image.maxTransmissionSize user.image.maxWaitTime user.image.resolutionReduction	True 20,000 bytes 5 s Yes
system.terminal.display system.terminal.memory system.terminal.diskSize	Black and white 2 MB 20 MB
network.currentBearer network.currentBearer.expThroughput	GSM 9600 b/s

Table 3. *Profile values.*

System Adaptability Manager

The System Adaptability Manager (SAM) is responsible for the provision of optimized and personalized mobile services. User-, network-, and terminal-specific QoS parameters are managed in profiles handled by the Profile Manager. These profiles are used to compute the best adaptation of the MASE communication services, taking into account the current network and end system QoS parameters as well as to the user's personal preferences. The SAM has several adaptation possibilities:

- It can make a choice between different available networks based on the available bandwidth, bandwidth guarantees, and cost.
- It can compress, convert, transcode, or reduce the multimedia objects prior to transmission.

For example, an image is supposed to be transmitted to a terminal with a black and white screen. In this case color information can be eliminated by the mobility gateway. Other reasons for adaptation can arise, for example, from the available network bandwidth and the cost involved. These decisions are made by the QoS trader within the SAM.

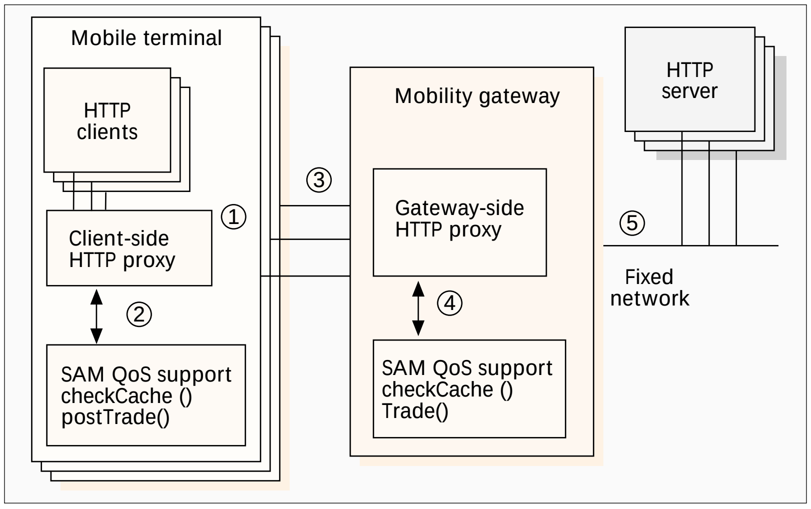

Figure 4. *The partitioning of the HTTP proxy.*

Profile Manager

The terminal characteristics of the mobile device are stored in a *terminal profile*. The network characteristics of the mobile device's current (wireless) connections are stored in the *network profile*. The network profile is constantly updated by the UAL. User-specific preferences are stored in a *user profile*. A profile generally contains a hierarchical tree of properties (name/value pairs), each describing a certain property. Profiles are replicated on demand and stored persistently throughout the MASE system.

A MASE-aware application can access the profile values at either the mobile device or the mobility gateway. Consistency can be enforced independently at every host and for each node to reduce the communication overhead. Table 3 shows examples of profile values for the user, system and network tree.

QoS Trader

The QoS trader adapts the MASE communication services according to the user's preferences and the current terminal/network situation as reflected in the profile. Instead of transmitting multimedia objects directly to the mobile client, the HTTP proxy calls the QoS Trader to perform a trading process. This process consists of the steps illustrated in Fig. 5.

During the *QoS gathering* phase the QoS trader accesses the current terminal and network QoS which are stored in the profile. The profile also contains the user's preferences, which are later used to compute filters and preferences for the planning process. It further examines the properties of the current multimedia object, and then enters the *planning* phase, during which it decides about the actions to perform on the current multimedia object. This process generates a "new" plan from a knowledge base (*plan generation*) and the gathered QoS parameters.

A plan consists of one or more compression, conversion or reduction steps. During the *plan* *filtering* phase the static properties of the new plan are matched against the user preferences and terminal requirements. Small devices, for instance, might have implemented only one or two decompression methods. The plan filtering phase will only select plans suitable for this particular device.

Subsequently, the QoS trader predicts the prospective outcome of the current plan using knowledge provided by the multimedia conversion (MMC) routines. MMC works *online*. It offers lookup functions that enable it to estimate conversion time, and execute conversions if appropriate. MMC provides some general-purpose methods for image manipulation, such as conversion of images to other image formats by means of scaling, graining, and color reductions.

Two methods that support the QoS prediction and filtering phases of the QoS trader, by estimating the required conversion time and the size of the reduced objects, are important for the SAM, which checks whether varying the reduction R of an object x can fulfill Eq. 1.

$$T_{MaxWait} > T_{Seek}(x_{org}) + T_{Reduce}(x_{org}, R) + T_{Trans}(x_{red}, B) \quad (1)$$

$T_{Trans}(x,B)$ is the transmission time of an object x at bandwidth B (b/s), $T_{Reduce}(x,R)$ is the estimated reduction time of the original object x at reduction R. $T_{Seek}(x)$ is the time used to estimate the reduction time of an object x to the requested file size (planning process). $T_{MaxWait}$ (stored in the profile under `user.image.maxWaitTime`) is a user preference parameter which defines the maximal time the mobile client wants to wait for an image. If $T_{MaxWait} > T_{Trans}(x_{org}, B)$, there is no need to reduce the file size, and the SAM transmits the original image.

A set of suitable conversion commands have been selected for our purpose. For images we use scaling, reducing colors, dithering, and converting to black and white to reduce image sizes. Some formats like JPEG allow conversion by scaling and reducing the overall quality. All those commands are "lossy"; they reduce the quality of an image and hence reduce the file size. To obtain a table of relative values for conversion predictions we have measured a set of images with all available conversion commands.

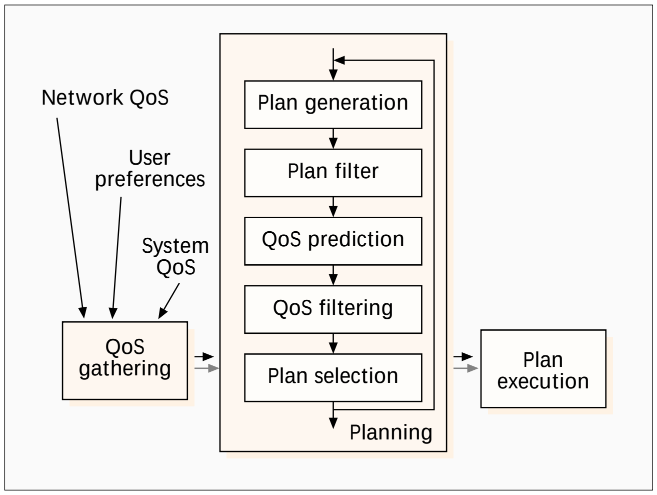

Figure 5. *The QoS trading process.*

Using this knowledge the trader can estimate the QoS parameters resulting from converting and transmitting the converted multimedia object (e.g., the overall processing time). These predicted QoS parameters are matched against the filters derived from the profile setting. During the *plan selection* phase a preference index is computed for each plan which has passed the filters. The used preference function is selected according to a user-defined criterion (e.g., the smallest resulting object size, the shortest time required for converting and transmitting the result, the best quality remaining).

When the planning is finished, the best plan with the highest preference index is executed. The resulting object is either returned to the application (local trading) or stored locally and will be marshaled for transmission.

Method	Description
compress	Compresses a file
decompress	Decompresses a file
estimateReductionTime	Returns estimated time to perform a desired reduction
reduceFileSize	Reduces file size by the desired reduction
convert	Converts an image into a given type
colorRedQuant	Reduces the number/depth of colors
colorRedDepth	Reduces the depth of color
colorRedDither	Reduces the number of shades per color.
getHeigth	Returns the height of an image
getMaxVal	Returns the max value of shades per color
getSize	Returns the file size in bytes
getType	Returns the image type as String
getWidth	Returns the width of the image
isCompressed	Returns true if the image is compressed
isImage	Returns true if the instance is an image

Table 4. *Subset of multimedia object conversion API.*

Example — The *plan generation* phase generated the plan "SCF(0.5); TRA; SCF(2)," which defines a scaling operation by the factor of 0.5, the transfer of the image, and the rescaling of the image to the original size during the `postTrading()` call on the mobile terminal. During the *plan filtering* phase this plan is compared with user preferences stored in the profile. For example, the parameter user.image.resolutionReduction defines whether or not the user accepts resolution reductions (scaling). If the user does not allow scaling, the filter derived from that value will prevent the above plan to be selected.

During the *QoS prediction* phase, the time required to execute the above plan will be computed by predicting the time required for the conversion SCF (0.5), for the transmission of the reduced image, and for the reconversion (SCF (2)) on the mobile device. The time and resulting image size for both conversions are computed by using average results derived from former conversions which were initially performed offline. Furthermore, the computing power of the mobile device is taken into account by using relative computing indices from the terminal profile for this device. In addition, the time to transfer the original image and a user-specific quality factor are computed.

During the *QoS filtering* phase, these results are used in Eq. 1 to check whether the user's

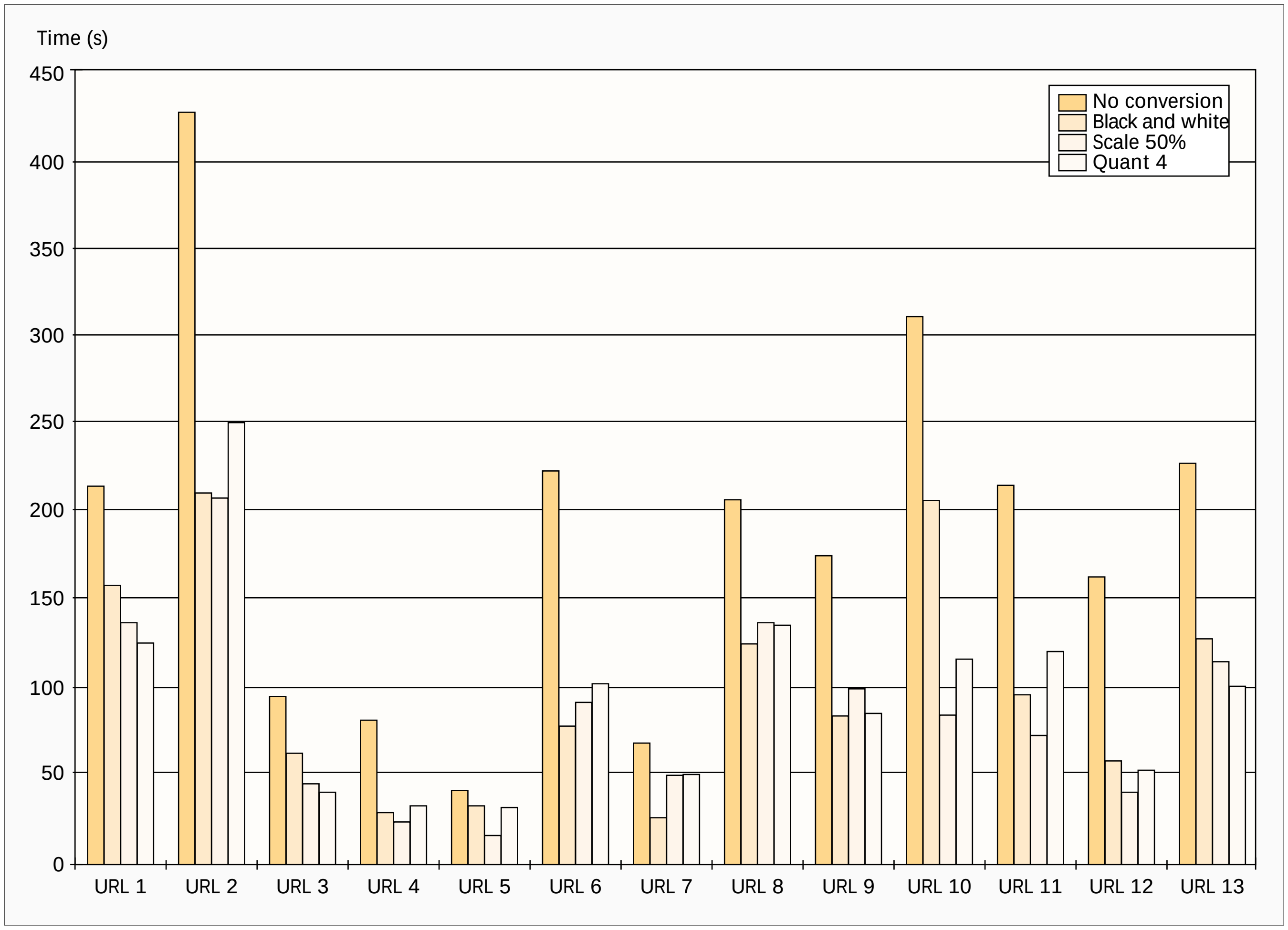

Figure 6. *Measurements of a MASE-supported browser over GSM.*

requirement `maxTransmissionTime` is fulfilled and whether the reduction will result in a faster transmission. If the plan is still valid, a preference index will be computed. Several preference functions are possible, ranging from a selection based purely on the processing time to a mixture of processing time and the quality of the reduced image. Finally, the plan with the best preference index will be executed.

Figure 6 shows the achieved measurements over the actual cellular system, GSM, that has data transport capabilities of up to 9600 b/s. The results show the advantage of the QoS trading and multimedia conversion. Depending on the automatically chosen conversion method the interaction of CM, SAM, and UAL achieve up to 70 percent acceleration of the HTTP transmission time. Table 4 represents a small subset of the SAM API that application programmers may use to realize conversions offline, or even during online connections.

Future Work

Future work will deal with the downsizing of the current architecture and implementation to fit the requirements of small mobile devices (e.g., PDAs and PICs), which usually have only limited memory and CPU power. The implementation of the Location Manager will be extended by supporting more input devices, such as an interface to a DECT system which holds information concerning the actual interworking unit. Further development will be done on a Session Manager that provides a higher abstraction of connections than TCP/IP does. The Session Manager controls the ongoing communication sessions and allows scheduling of different MASE operations like prefetching and caching. Results of the project's field trials will be used to improve the current status.

Conclusions

The data services of future third-generation mobile telecommunication systems play a critical role in facilitating new and innovative mobile-aware applications and services. UMTS will define a set of services enabling seamless roaming between different networks, QoS monitoring, and bandwidth on demand. In this article we have introduced the OnTheMove approach, which employs middleware to support the special needs of mobile-aware applications. This middleware not only allows the development of mobile-aware applications in an easy way, it also shields today's application developer from the ongoing developments toward UMTS. The MASE mobile middleware will pave the road to future mobile telecommunication systems.

Acknowledgments

The authors would like to acknowledge all contributing partners of the OnTheMove project [8] which is sponsored partially by the European Commission in the ACTS program under contract AC034. The project participants are Ericsson Radio Systems AB, T-Mobil GmbH, Ericsson Eurolab GmbH, Siemens AG, Rheinisch-Westfälische Technische Hochschule Aachen (RWTH), IBM France, Tecsi, BT, Bonnier Business Press, Royal Institute of Technology (KTH), Sony International (Europe) GmbH, Burda Com Media Solutions GmbH, Centre for Wireless Communications (CWC), and Iona.

References

[1] EC DG XIII/B (1/3/96), "UMTS Task Force Final Report," Brussels,Belgium, ACTS InfoWin, http://www.infowin.org/ACTS/.

[2] J. Schwarz da Silva *et al.*, "Evolution Towards UMTS," ACTS InfoWin,http://www.infowin.org/ACTS/IENM/CONCERTATION/MOBILITY/umts0.htm.

[3] M. H. Callendar, Ed., *IEEE Pers. Commun.*, Special Issue on International Mobile Telecommunications-2000: Standards Efforts of the ITU, vol. 4, no. 4, Aug. 1997.

[4] J. Meggers and A. S. Park, "Mobile Middleware: Additional Functionality to Cover Wireless Terminals," *Proc. 3rd Int'l. Wkshp. Mobile Multimedia Commun.* (MoMuC-3), Princeton, NJ, Sept. 1996; D. J. Goodman and D. Raychaudhuri, Eds., *Mobile Multimedia Communications*, Plenum, 1997, pp. 151–57.

[5] J. Meggers, A. S. Park, and R. Ludwig, "Roaming between GSM and Wireless LAN," *ACTS Mobile Commun. Summit*, Granada, Spain, Nov. 1996, pp. 828–34.

[6] U.S. Coast Guard Navigation center, "GPS, DGPS, LORAN, OMEGA, LNM," http://www.navcen.uscg.mil/gps/gps.htm.

[7] WaveLAN Wireless Computing, http://www.wavelan.com.

[8] OnTheMove home page, http://www.sics.se/~onthemove.

Biographies

BIRGIT KRELLER (birgit.kreller@mchp.siemens.de) received her Dipl.-Inform. (Master's in computer science) from the University of Magdeburg, Germany, in 1996 on the subject of mobile agents for load balancing in large telecommunication systems. She joined the Siemens Corporate Research and Development Department in March 1996, where she is working in the ACTS project OnTheMove. Her current research interests include mobile computing architectures, wireless networks, geographical positioning systems for mobile devices, and mobile agents.

ANTHONY SANG-BUM PARK received his Dipl.-Inform. from Aachen University of Technology (RWTH), Germany, in 1995, previously studying at the University of Koblenz. From 1989 to 1992 he was with Philips Communication Industry AG in quality control, and with Parsytec Computer GmbH focusing on massive parallel computing. Since 1995 he has been a researcher and Ph.D. candidate at RWTH, Department of Computer Science, responsible for agent technology research projects and working in ACTS projects. Research topics are mobile computing and personal multimedia communications. Activities concerning distributed systems and middleware architectures are mainly in the area of mobile agent technology.

JENS MEGGERS received his Dipl.-Inform. in computer science from RWTH, Germany, in 1995. He joined the Department of Computer Science of the same university as a Ph.D. candidate in computer science in 1995. He participates in various internal and external research activities, including the ACTS project OnTheMove. His main research focus lies in QoS supporting network and transport protocols for mobile multimedia communications.

GUNNAR FORSGREN received his B.S.E.E degree from Härnösand Gymnasium, Sweden, in 1978 and has been with Ericsson in various R&D positions since 1980. His research interests include information/communication services on wireless devices and their interaction with agent services.

ERNÖ KOVACS received his Dipl.-Inform.from the University of Kaiserslautern, Germany in 1991. During 1986–1990 he worked at IBM's European Networking Centre (ENC) in various research projects concerning multimedia e-mail, multimedia documents, and distributed hypermedia systems. From 1991 to 1996 he worked at the Institute of Parallel and Distributed High-Performance Systems (IPVR) of the University of Stuttgart. He conducted several projects in the area of middleware for distributed systems. In 1997 he joined Sony's Research and Development Department in Stuttgart and worked in the ACTS project OnTheMove. His current research interests include mobile multimedia, quality-of-service trading, and mobile agent systems.

MICHAEL ROSINUS received his Dipl.-Inform. at the Universität des Saarlandes, Germany, in 1996 and worked at the German Research Center for Artificial Intelligence (DFKI) in the area of intelligent agents. In May 1996 he joined the Sony Research and Development Department where he is working in the OnTheMove project. His current research interests include mobile and intelligent agents, multimedia data processing, and wireless networks.