Automated Adaptation for Mobile Computing Based on Mobile Agents

Thomas Springer and Alexander Schill

Dresden University of Technology, Department of Computer Science, Institute for System Architecture, Mommsenstr. 13, D-01062 Dresden, Germany, Tel.: +49 351 463 8261,
Fax.: +49 351 463 8251
{springet, schill}@rn.inf.tu-dresden.de

Abstract. In the last few years the market brought up a big variety of mobile devices with differing functionality. This development results in an increasing number of wireless subscribers and in the upcoming demand for a seamless access to information and services provided by the internet. To support arbitrary devices in a mobile environment we introduce an approach for adaptation at application level based on a partitioning model and abstract operations. Available techniques to support and improve mobile applications are integrated into a concept with generic and application specific components which adapt information according to the capabilities of mobile devices, network quality and user settings. We also describe a multimedia email application for the validation of our model. The implementation is based on mobile agents providing dynamic installation, flexible deployment and mobility support.

1 Introduction

One of today's visions in mobile computing is ubiquitous information access [1]. Information (e.g. email, fax and web pages) should be available anywhere at any time. During the last years the market brought up a big variety of mobile devices with differing functionality. This development results in an increasing amount of wireless subscribers and the desire to access information and services provided by the internet. Unfortunately most of the currently available applications and services for the internet expect high bandwidth connections and well equipped client devices. WAP [2] is one approach to overcome this problem but focuses on the characteristics of mobile phones and does not scale to the extended capabilities of a notebook. For a seamless integration of arbitrary mobile devices information has to be adapted to their resources (e.g. memory, display, software, network connections). Moreover users change between devices (e.g. from PC to PDA) but want to access and handle the same information. This requires applications which can react very flexibly to changes in their environment (e.g. changing client devices or network connections) and adapt to new devices, changing user requirements and individual user needs.

One of the most interesting applications (because it addresses basic user needs) is reading and writing email. Even mobile phones allow the access to email. But currently the download of messages via wireless networks or even via a standard modem is very unsatisfying. Email tools do not allow the selection of certain message

C. Linnhoff-Popien and H.-G. Hegering (Eds.): USM 2000, LNCS 1890, pp. 284-289, 2000.

content or the adaptation of attachments. Especially when only a slow connection is available needless data within a message (e.g. an audio attachment while the device is not audio enabled) should be filtered out automatically. Furthermore large attachments should not be downloaded or compressed before the transfer. Moreover the user should be able to determine what messages he wants to read (e.g. only the messages from certain senders). These examples point out three areas of adaptation: device capabilities, network connection and user preferences. Exceptions (e.g. a user needs to download a large attachment because it contains a map to a location he urgently searches for) should also be handled by the application.

In this paper we introduce a model for automated adaptation of information at application level based on mobile agents to address the requirements of mobile information access. While other approaches focus on special problems like filtering of information [7] or data reduction [6], the aim of our model is to integrate adaptation techniques (e.g. handling of frequent disconnections, varying network quality, low resources on the mobile device and mobility) into a generic approach supporting arbitrary devices. The adaptation is automatically, flexibly deployable and configurable. Section 2 describes an adaptation model with generic and application specific components. We also introduce a concept for the placement of the functionality with relocation of computation between client and server in mind. In section 3 we present a multimedia email application which was prototypically implemented to validate our model and contains most of the functionality described above. The implementation maps the abstract components for adaptation on mobile agents to enable the dynamic installation and placement of adaptation functionality. Section 4 concludes the paper and gives an outlook to future work.

2 Adaptation Model

The model introduces components at application level rather than a system based approach. The distinction of adaptation techniques shown in figure 1[1] was done using two criteria: placement of code and availability of information. For flexible configuration the components are intended to be loosely coupled, i.e. the generic components can be dynamically added and removed at runtime.

For the functionality contained in this partitioning model we have defined abstract operations which can be variably combined to sequences. Each operation gets input data and parameters and produces output data. The algorithms used for adaptation are not determined by the model. Rather the right functionality is selectable from a set of implementations. For instance the queuing operation can be implemented using different strategies (e.g. priorities). The following list of operations is not complete. We only describe the operations used for the implementation of the email application.

[1] Details can be found in [3].

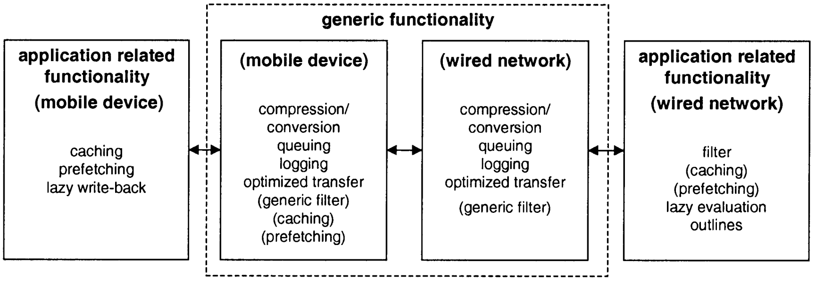

Fig. 1. Placement of adaptation functionality

1. The ***Selection*** operation checks or selects the input data according to a criterion (e.g. a filter or the content of a data object). One parameter can be the type of the input data to process different data types in different ways (e.g. for conversion).
2. The ***Replacement*** operation creates a new representation for the input data (e.g. a text description or an outline for an email attachment).
3. The ***Composition and Decomposition*** operations assemble and disassemble structured data types from and to a sequence of data objects.
4. The ***Compression and Conversion*** operation enables the adaptation of format and volume of multimedia data specific to the type. For instance a GIF image can be transformed to JPEG with a quality parameter for information loss. Furthermore the color depth and the image size can be reduced. The operation consists of encoding the data, conversions and decoding the data. Thereby the format for encoding and decoding may be different.
5. The ***Queuing*** operation facilitates the enqueuing and dequeuing of data objects according to the implemented strategy.
6. The ***Transfer*** operation provides a reliable data transfer optimized for special networks. For example multiple parallel send threads can be used for communication media with high delay times to better exploit the available bandwidth.
7. The ***Requirement Determination*** operation processes the information of the user and device profile and the actual network quality provided by the environment sensing service and creates a set containing the most restrictive requirements.

Figure 2 shows a possible combination of generic operations. The queuing component enqueues (1) the incoming data objects. If the transfer is possible, one element is dequeued (2). The compression and conversion operation adapts the volume and format of data according to the information provided by the requirement determination operation (3). The transfer operation initiates the dequeuing and controls the transmission of data over the network (4). The incoming data objects are handled according to their data type. Therefore the operations can be used in a generic manner independent from information at application level. The generic components are intended to be used for all applications on a mobile device. The next section describes a sample application and possible implementations of application related operations.

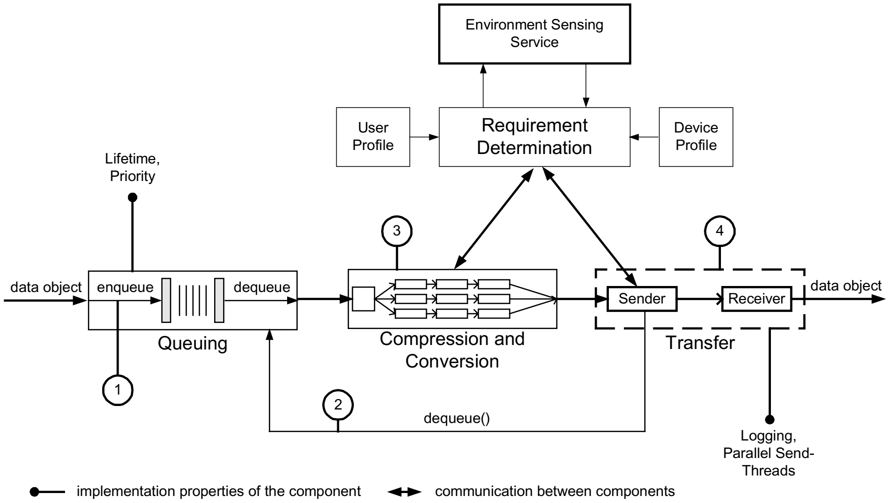
Environment Sensing Service
User Profile
Requirement Determination
Device Profile
Lifetime, Priority
3
4
data object
enqueue
dequeue
Queuing
Compression and Conversion
Sender
Receiver
data object
Transfer
1
2
dequeue()
Logging, Parallel Send-Threads
implementation properties of the component
communication between components

Fig. 2. Generic adaptation components

3 Agent Based Email Application

To validate our adaptation model we have implemented a multimedia email application using mobile agents as components. According to the partitioning model shown in figure 1 the application consists of an email tool (we use Netscape Messenger), two pairs of adaptation components and the mail server.

Configuration. As described in section 1 the application automatically filters and adapts the messages downloaded to the mobile device. The adaptation behavior is configurable. The application related component (email agent) sent to the mail server filters the messages before the transfer over the network. The user can specify two kinds of filters: email header filters (e.g. sender, receiver or subject) and filters for information in the body (e.g. body text or attachment data type and size). The header filters are used for a first selection of messages. Only messages matching at least one header filter are selected for download. If no header filter matches the message is not handled by the email agent but remains on the mail server. The attachment filters are applied before the transfer. Only data types specified in the filter are sent. Furthermore the attachment must not be larger than the threshold assigned to its data type. The generic components are configured independently. Currently the user can specify the data types for compression and the degree of the data loss (in percent). Further enhancements will be rules to convert data objects (e.g. a limited pixel size for images or a maximum number of colors). The user can also transfer a list of headers to the mobile device. What headers the list contains is configurable (e.g. all messages from the account, only new messages, all messages older than x days etc.) From this list the messages for download can be selected manually.

Application. The application is started on the mobile device. The email agent as well as one of the generic components migrate to hosts within the wired network. While the email agent moves to the mail server the generic component migrates to a host near the user and follows him when he moves. The email agent contains information about the mail account, filters and user preferences. At the mail server it asks for messages using the information about the email account. For filtering and transfer of messages the adaptation components shown in figure 3 are used. New messages are selected for download (1) by the evaluation of the headers. Selected messages are decomposed (2) to data objects which are divided (3) into the header, the body text and the attachments for separate handling. For the attachments the filters are applied (4). Attachments removed during the filtering are replaced (5) by another representation (currently a text description is added to the body text). The data objects of the messages are sent to the generic component within the wired network. Before the transfer over the last link to the mobile device data is automatically adapted to the quality of the connection, the resources on the mobile device and user preferences using the requirement determination operation (6). After the transfer to the mobile device the data objects are composed to a MIME message (7) which can be displayed by an arbitrary email tool and the user is notified.

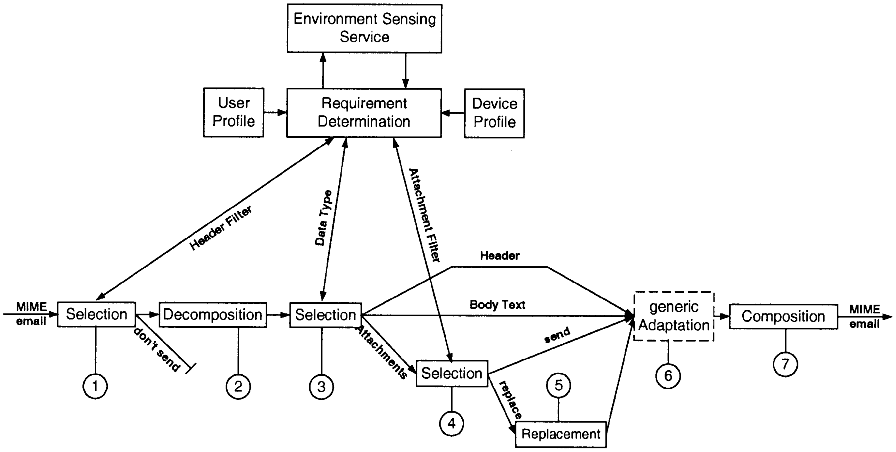

Fig. 3. Adaptation components of the email application

Performance. The implementation is based on Voyager by ObjectSpace [8] and the JDK 1.2 [9]. The communication between agents is based on JavaRMI. With the prototype we have done performance measurements. The test message contains a JPEG image (12204 bytes in high and 4439 bytes in low quality). The transmission of the message containing the high quality image between an Pentium Pro 200 MHz (64 MB RAM, WindowsNT) and a Thinkpad 760D (Pentium 166MHz, 64MB RAM, Windows95) connected via the Xircom Netwave wireless LAN took 809,6 ms with and 774,4 ms without the generic components. For the message containing the low quality image we measured 444,7 ms with and 378,3 ms without the generic components. While the lossy compression of the image (decoding and encoding) needs 135,6 ms in average altogether the transfer time for the message is significantly smaller. The indirection over the generic proxy takes also time but the benefit for disconnection handling and adaptation justifies this effort. For further details see [3].

4 Conclusion and Future Work

In this paper we described a model for automated and flexible adaptation of information according to the available network quality, the capabilities provided by the mobile device and user settings. While other approaches introduced static components and are focussing on special techniques (e.g. filtering of information [7], data reduction [6] or disconnected operations [5]) our concept integrates the techniques into one model and introduces mobile components taking user mobility into account. Because of the partitioning model shown in figure 1 the adaptation functionality can be distributed to where it can be used most efficiently. Moreover the generic components can be used by all applications on a mobile device. Because of the use of mobile agents adaptation components can be dynamically installed at runtime wherever they are needed. This is a big advantage compared to system based concepts with static components (see [4] and [5]). Furthermore the components are loosely coupled, enabling dynamic addition and removing of components (i.e. the generic components). Legacy software can be enhanced with adaptation functionality without changing the existing components. The additional functionality can be added using the proxy approach and client-site agents. Currently we work on the refinement and implementation of the operations described in the adaptation model. Moreover we want to implement further applications to prove the applicability our concepts.

References

1. Weiser, M.: Ubiquitous Computing; www.ubiq.com/hypertext/weiser/UbiHome.html
2. Wireless Application Protocol Forum: WAP White Paper, 1999
3. Schill, A.,Held, A., Ziegert, T., Springer, T.: A Partitioning Model for Applications in Mobile Environments; In Todd Papaioannou and Nelson Minar, editors, Proceedings. Mobile Agents in the Context of Competition and Cooperation (MAC3), a workshop at Autonomous Agents '99, pages 34-41, 1999
4. Schill, A., Kümmel, S., Springer, T., Ziegert, T.: Two Approaches for an Adaptive Multimedia Transfer Service for Mobile Environments; Computers & Graphics, Vol. 23, No. 5, 1999
5. James J. Kistler and M. Satyanarayanan: Disconnected Operation in the Coda File System; ACM Transactions on Computer Systems, 10(1), pp. 3-25, February 1992
6. Fox, A., Steven, D. Gribble, Brewer E.A., Amir, E.: Adapting to Network and Client Variability via On-Demand Dynamic Distillation; Proceedings of the Seventh International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 160-170, October 1-5, 1996
7. Bruce Zenel, Dan Duchamp: *A General Purpose Proxy Filtering Mechanism Applied to the Mobile Environment*; The Third Annual ACM/IEEE International Conference on Mobile Computing and Networking, 1997 pp. 248-259
8. www.objectspace.com
9. java.sun.com