

Modular Mobile Application Development Framework for Resource Constrained Devices (May 2005)

Poornima Weerasekara, Sasakthi S. Abeysinghe, Member, *IEEE*

Abstract—The rapid growth in global mobile phone usage has created a great demand for feature rich mobile applications. However, the resource limitations in mobile devices and the slow, unreliable nature of mobile networks are two major hindrances to the provision of such services. This paper presents a generic framework for mobile application development, which provides guidelines to develop an application as a collection of independently executable segments. Further, an execution mechanism is proposed which enables these code segments to be downloaded “on demand”. A network infrastructure using 3G mobile technologies is used to mitigate the problem of low bandwidth that may hinder the timely transfer of segments. The authors conclude that the concept of “segmented application development” and “code-on-demand” discussed in this paper could change the nature and quality of applications available for resource constrained mobile devices.

Index Terms— Mobile communication, Distributed computing

I. INTRODUCTION

WHILE today’s mobile applications seek to provide an unprecedented level of functionality, the limited memory, processing power and storage available in mobile devices have been a hindrance to the development of more sophisticated and larger applications. The core problem lies with the monolithic structure of current mobile applications that consume a considerable amount device storage and memory [1].

The unbalanced use of network bandwidth is another problem. While 3G promises broadband speeds on wireless internet, practical implementations fall short of expectations. Hence mobile applications do not make optimal use of the available bandwidth when code and data are being downloaded together at once, prior to execution [2].

The authors investigate the possibility of adopting the concepts of “segmented application development” and “code-on-demand” to increase the execution flexibility of large applications, while reducing their memory signature [6] and to enable the optimal utilization of available

bandwidth.

II. LIMITATIONS OF CURRENT DEVELOPMENT STRATEGIES

There is no “definite” solution at present to address the problems discussed above. The following models represent certain industry “best practices” and heuristics that are being used to moderate these issues, with mixed results.

A. Using Lightweight Libraries

For instance, the Java Connected Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP) standard libraries are designed from the ground up as lightweight components [9], [11], [12]. However lightweight libraries provided by vendors often use proprietary APIs. Although this leads to having a smaller memory footprint and better performance, it requires extra developer training and results in less portable applications [3].

B. Reducing the Application Footprint

Developers could minimise both the storage and runtime footprints of the application by:

- 1) *Optimizing the packaging process*: i.e., attempting to only include the classes actually used by the application. This can be done manually for smaller libraries or with the aid of automatic tools incorporated into certain IDEs (for example, the IBM WebSphere Studio Device Developer).
- 2) *Partitioning the application*: e.g., The MIDP runtime loads classes only as needed. Therefore the application can be partitioned into separate parts to reduce the memory and storage requirements [11]. For MIDP applications, the MIDlet suite can contain several relatively independent MIDlets.
- 3) *Using shared libraries*: These further reduce the overall memory and storage space required, since the library no longer needs to be duplicated and packaged with each application [10].

C. Minimizing the garbage collection

The garbage collector on mobile devices must run more often due to the limited memory available [14]. However, this takes up precious CPU cycles and slows down all other application processes [15]. The only rule of thumb to mitigate this problem is to minimize object creation and quickly dispose of objects that are no longer in use [13].

Manuscript received May 30, 2005.

P. Weerasekara is with the Informatics Institute of Technology, Sri Lanka and the Manchester Metropolitan University, United Kingdom. (phone: +94-11-2698519; e-mail: poornimaw@gmail.com).

S.S. Abeysinghe, is with Informatics Institute of Technology, Sri Lanka and the Manchester Metropolitan University, United Kingdom. (phone: +94-77-3104818; e-mail: sasakthi@gmail.com).

III. DESIGNING THE GENERIC FRAMEWORK

The main design goals of the Modular Mobile Application Development Framework are reusability, extensibility and enhanced performance. The component-based design of the framework is discussed below.

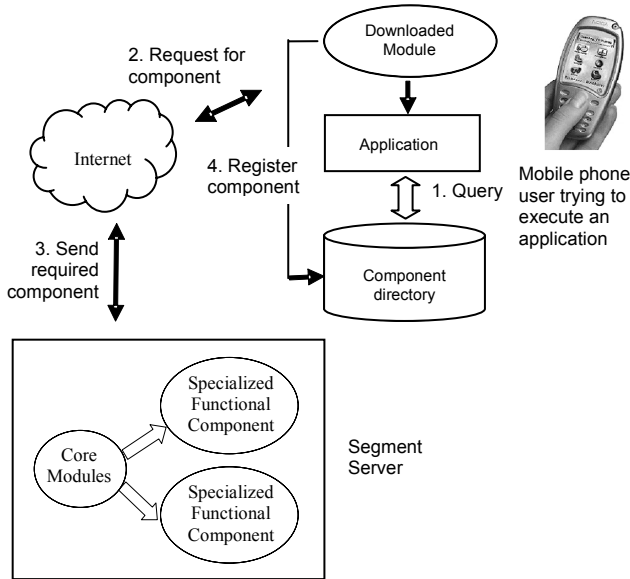


Fig. 1. Overview of the framework

A. Application segment:

The concept of an application “segment” provides the basic design and development unit for a modularized mobile application based on the proposed framework [7]. It provides guidelines that enable developers to group related functionalities into a segment and define relationships between segments that make up the overall application [1]. A “functionality” can be defined as a single well-defined task in an application. A segment consists of two parts.

- 1) *Segment interface*: This is where the meta-data of the segment is defined. It includes information such as the segment id, input and output specification, vendor and versioning information, its resource requirements (such as memory utilization) and its functional dependencies.
- 2) *Segment implementation*: This refers to the code which implements the functionality. Segments are hosted on segment servers and can be discarded from the run-time once its execution has completed.

B. The segment locating and loading mechanism:

If a user needs to download and execute an application he/she has to first access the central application repository, which lists all the applications available. The user initially downloads the “application manager”, which contains the relevant information about the server that holds the application segments [6].

When a particular segment needs to be downloaded, the application manager first checks whether the segment could be executed with the resources currently available in the device [10]. Then it checks whether the segment has been previously downloaded, and if so, whether it exists in the segment cache. If the segment does not exist in the cache, the application manager forms a secure connection with the relevant server and downloads the required segment.

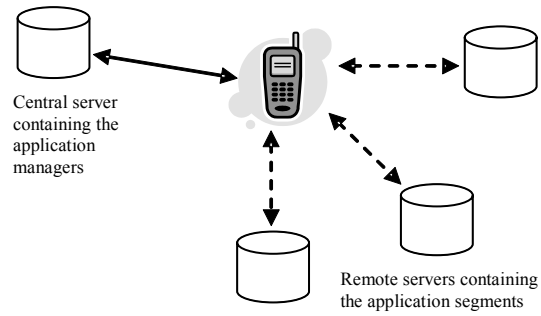


Fig. 2. Segment locating and loading mechanism

C. The segment execution mechanism:

The runtime environment to execute a downloaded application is created by the application manager. A segment manager is assigned to each segment dynamically during run time. This monitors the execution progress of the segment and informs the application manager of any errors that may occur during execution. The application manager decides on the best method to rectify any error. Just before a segment completes execution, the segment manager alerts the application manager so that it could start downloading the other related segments based on the information provided by the segment interface or it may request for a user input.

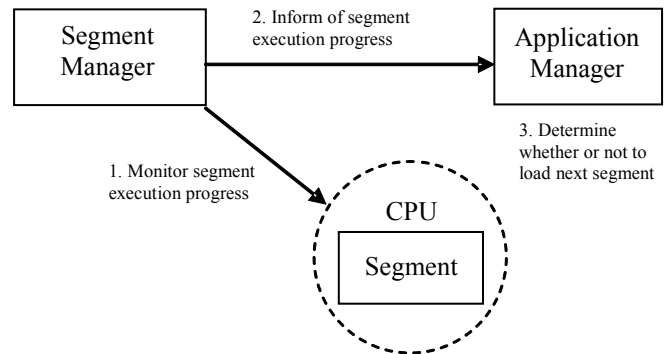


Fig. 3. Segment execution mechanism

D. Segment Caching & Discarding:

The segment caching and discarding mechanism will not vary from application to application. Therefore these functionalities are provided as concrete helper classes which can be directly re-used by the developers. Caching frequently used segments minimizes the number of requests to the server [1]. The ranking algorithm for cached segments is as follows:

- 1) If a segment is downloaded for the first time and has completed execution, then
 - a) If the cache is full, discard segment with the lowest rank.
 - b) Insert segment into cache.
- 2) Every time the item is accessed increase its rank by one.
- 3) Check whether any segments that are related to the segment being currently executed are residing in the cache. If so increase their rank. (This is done because a related segment has a higher probability of being requested next, and thus should not be discarded).

IV. DISCUSSION OF RESULTS

A distributed VoIP Application was developed as a proof-of-concept of the proposed framework. This application aimed to channel mobile phone calls via VoIP instead of the traditional GSM network [8]. The segmented development approach increases the flexibility of the application to adopt a wide range audio/video standards, security and Quality of Service (QoS) implementations.

The experimental results obtained through the implementation of the Modular Mobile Application Development Framework could be discussed from the following perspectives.

A. Memory utilization using framework:

The framework minimizes memory utilization by enabling only required application segments to reside in the device memory at the time of execution. This frees more memory for runtime use which could be utilised for application code and data downloaded from servers as required [15]. Further, better memory usage leads to higher application efficiency and performance resulting in an improved end-user experience [5].

B. Bandwidth utilization using framework

Only the core modules such as the application manager need to be downloaded prior to execution. This minimizes network congestion. Further, download on demand offers better network utilization and improves efficiency [1]. This results in a balanced sharing of bandwidth between data and code.

C. Enhanced end-user experience:

The proposed framework helps to enhance the end user experience by enabling sophisticated applications to run on resource constrained devices. Further, the lower memory utilization and reduced network download times lead to faster execution of applications [4]. Finally, it also results in lower costs for users as only the required segments are downloaded using billed bandwidth.

V. FUTURE ENHANCEMENTS

The proposed framework could also be used as a basis for formulating solutions in related problem domains, such as in mobile-grid computing and developing context-aware applications. For example, the framework could be enhanced by incorporating the following suggestions.

1) Better static and dynamic module-usage profiling techniques can be used to fine-tune the ranking algorithm used to determine which cached segments need to be discarded. This would help minimise the module re-load frequency.

2) The framework proposed can be used as the foundation for developing "context aware" applications. Since segments are brought in at run-time to compose the required application it enables them to dynamically adapt to the client device. For example, mobile devices can be used to provide the user with context related information, such as flight information within an airport or a product catalog inside a super market [6]. In order to serve such purposes the device should be able to download and execute relevant application code for each context [2].

3) The framework could be enhanced to provide ubiquitous integration across heterogeneous mobile devices and different mobile development platforms [6].

4) The concept of "code on demand" described in this paper and the proposed framework for designing mobile applications as a collection of independently executable segments can be enhanced to provide mobile grid-computing solutions [5]. Although this is a research area in its infancy, studies show that a group of mobile users can be grouped into an ad-hoc network and each mobile device could act as a processing point in a distributed processing environment, which can be used to execute complex applications.

It is expected that the enhancements mentioned above would serve as a reference point to others with similar research interests.

VI. CONCLUSION

This paper discusses an alternative approach to mobile application development, which results in lightweight and flexible execution modules that can be run in resource constrained devices. A generic framework based on the concepts of "segmented application development" and "code-on-demand" was introduced to reduce the memory usage and network bandwidth utilization, which are the two most valuable resources in a mobile computing environment. Based on the results observed by the authors it could be concluded that the Modular Mobile Application Development Framework enables large, complex, resource consuming applications to be executed in resource constrained mobile devices.

ACKNOWLEDGMENT

The authors wish to acknowledge the support and supervision given by Prof. Alfred Perera, Head of the Department of Computing, Informatics Institute of Technology, and all faculty members and colleagues at the institute who supported and contributed to the successful completion of this research and related documentation.

REFERENCES

- [1] Y. Chow, W. Zhu, C. Wang and F.C.M. Lau, State-On-Demand Execution for Adaptive Component-based Mobile Agent Systems. The University of Hong Kong, 2004.
- [2] I. Athanasias, L. Popa, C. Raiciu, R. Pandey and R. Teodorescu, Using Code Collection to Support Large Applications on Mobile Devices Politehnica University Bucharest, University of California, Davis, Lehman Brothers, New York, 2004.
- [3] H. Baldwin, The challenges of mobile development. <<http://asia.cnet.com/enterprise/infrastructure/printfriendly.htm?AT=39087822-39006409t-39000220c>>, 2002.
- [4] UMTS, 3G and UMTS Technology. <<http://www.umtsworld.com/technology/technology.htm>>, 2003.
- [5] Y. Huang and N. Venkatasubramanian, Supporting Mobile Multimedia Services with Intermittently Available Grid Resources. <<http://mapgrid.ics.uci.edu/HiPC2003.pdf>>, 2004.
- [6] J. Karvonen and J. Warsta, Mobile multimedia services development: value chain perspective, Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia. University of Oulu. <<http://portal.acm.org/citation.cfm?id=1052380.1052404>>, 2004.
- [7] M. E. Fayad, R. E. Johnson and D. C. Schmidt, Building Application Frameworks: Object-Oriented Foundations of Framework Design. John Wiley & Sons, 1999.

- [8] M. Mallick, *Mobile and Wireless Design Essentials*. John Wiley & Sons, 2003.
- [9] Sun Developer Network, *Connected Limited Device Configuration (CLDC)* <<http://java.sun.com/products/cldc/>>, 2005.
- [10] V. Lee, H. Schneider and R. Schell, *Mobile Applications: Architecture, Design, and Development*. Prentice Hall, 2004.
- [11] S. Venkateswaran, *Java Programming for Wireless devices using J2ME/CLDC/MIDP*. California Software Labs, USA, 1998.
- [12] Sun Developer Network, *Mobile Information Device Profile (MIDP)* <<http://java.sun.com/products/midp/>>, 2005.
- [13] M. J. Yuan, *Enterprise J2ME: Developing Mobile Java Applications*. Prentice Hall, 2003.
- [14] A. Muir and R. Bialach, *Reduce, Reuse and Recycle: Reusing Objects*. <<http://www.microjava.com/articles/techtalk/recycle2>>, 2002.
- [15] E. Giguere, *Mobile Information Device Profile for Java 2 Micro Edition: Professional Developer's Guide*. John Wiley & Sons, 2001, ch 3.

Poornima Weerasekara was born in Colombo, Sri Lanka in 1985. Weerasekara is currently a final year undergraduate student following a BSc. (Hons) in Information Systems at the Informatics Institute of Technology, Colombo, Sri Lanka.

She completed her internship at Virtusa, a global solutions provider, in 2004. Her current research interests include Mobile Application Development Strategies and Future Wireless Internet.

Sasakthi S. Abeysinghe (M'04) became a Member (M) of IEEE in 2004. He was born in Colombo, Sri Lanka in 1980. Mr. Abeysinghe earned his BSc. (Hons) in Information Systems at the Manchester Metropolitan University, UK in 2004.

He is employed at IFS R&D International as a Software Engineer and at the Informatics Institute of Technology, Sri Lanka as a visiting lecturer. His paper on Three-Dimensional Motion Tracking using Stereo Vision has been accepted for publication on the 11th Annual Transactions of the IEE, Sri Lanka (2004). His research interests are in computer vision, wireless communication and data engineering.

Mr. Abeysinghe holds professional memberships of the IEE and the BCS.