MTrainSchedule: Combining Web Services and Data Caching on Mobile Devices

Iulia Ion   Alexandru Caracaş   Hagen Höpfner*

April 2, 2007

Abstract. Despite recent improvements in wireless network protocols and the manufacturing of better, more performant mobile devices, developing mobile applications and services still poses major challenges. Mobile devices have scarce computing capabilities, limited storage, short battery lifetime, and slow, expensive, and unreliable wireless communication. To alleviate these problems, we propose (1) to move the computationally intensive operations from the mobile client to an external *Web Service* and (2) to implement efficient *data caching* on the mobile device to minimize network traffic. In this article, we present these two key design aspects in the context of a Java development of a mobile service that offers train schedule information. The results show better response time and reduced network traffic.

1 Introduction and Related Work

The enhancement in the capabilities of mobile devices together with the increasing user adoption rate provide a solid basis for the development and deployment of innovative mobile applications and services. However, despite these improvements, mobile devices are still far behind personal computers in terms of scarce computing capabilities, limited storage, short battery lifetime, and slow, expensive, and unreliable wireless communication.

Despite the definition of the 3rd Generation (3G) network protocols (e.g. UMTS) which promise higher transmission rates, 3G networks are expensive and not widely deployed. Wireless connectivity is by its inherent nature unreliable due to reflection, refraction, signal attenuation and so on. Furthermore, mobile devices have limited storage, computational power, and battery lifetime. To alleviate these problems, we propose a Web Service based architecture to distribute computational intensive operations, and client-side data caching to reduce network traffic. As a direct benefit, the battery power consumption of the mobile or embedded device decreases.

In this article, we present the Java implementation of a mobile application that gives train schedule information and we emphasize how we put in practice the above mentioned design principles. The user enters the departure station and the application displays the next departing trains and their destinations.

Other projects aim to provide similar services. *Deutsche Bahn* offers personal schedules between two specific stations[1]. However, the user needs to download and install one application for each start and destination pair. Once installed, the application never updates the data nor checks for data validity.

Another approach is by *Actuan Mobile* who offers the complete dynamic schedule for the New Jersey Transit rail system. The application[2] is offered both as a mobile browser solution and Java application with extensive features. However, no data caching is used; the schedule information is obtained directly by querying on-line servers for each request.

Most of the mobile devices on the market (mobile phones, smart phones, PDAs, etc) run an implementation of the Java 2 MicroEdition (J2ME). J2ME contains several profiles, but the most widely used is the Mobile Information Device Profile (MIDP). For these reasons and for portability considerations, we implemented the *MTrainSchedule* as an MIDP application (i.e. MIDlet).

In the following section, we present theoretical considerations of the two key design aspects: data caching and Web Services, as well as general practical concerns regarding Java implementation on mobile devices. Section 3 presents the MTrainSchedule application and emphasizes the design principles and implementation decisions. Section 4 evaluates the approach while Section 5 concludes and proposes further research directions.

2 Theoretical and Practical Considerations

The usage of a service oriented architecture offers significant advantages in terms of loose coupling of services, flexibility, and adaptability to changes. The JSR 172 optional package, J2ME Web Services Specification [Ellis and Young, 2003] for MIDP, defines the API for accessing remote SOAP/XML based Web Services. Method invocation follows the synchronous request-response model of client-server interaction. The JSR 172 stub and runtime transparently handle the encoding and decoding of method calls and parameters, object serialization, as well as sending the request and receiving the response. However, despite the introduction of this standard, Web Services for mobile clients are still scarcely discussed in literature and implementations for mobile phones are nascent.

General, theoretical considerations to data caching have been throughly discussed in literature. Based on the classification of replication, hoarding and caching in [Höpfner et al., 2005], one

*Supervisor of the project.
[1]`http://persoenlicherfahrplan.bahn.de`
[2]`http://www.actuanmobile.com/documentation/train_schedule_manual-online-version-v0_1.pdf`

must consider major caching issues such as coherency, replacement and look-up. [Lee et al., 1999] introduces semantic caching and [Gray et al., 1996] presents synchronization techniques.

To cache data on a mobile device, one must use the persistent storage offered by MIDP: the Record Management Store (RMS) [Ghosh, 2002, Giguere, 2004]. Record Stores are uniquely identified by their names and contain a number of records accessible by their integer index. We use RMS to persistently store cached application data. The specific indexing and storing formats are described in the next section, together with the system's architecture.

3 MTrainSchedule: Architecture and Implementation

The scheduled train departures can normally be checked online from public transport websites. However, accessing these services from a mobile phone remains expensive and inefficient, especially when the user repeatedly enters the same query. Alternatively, we propose a mobile application that stores the schedule and reuses the information during the next sessions, thus minimizing network traffic.

Besides the usual HTML interface, the Karlsruher Verkehrsverbund Website (KVV[3]) also offers, for each station, raw data files containing limited information on all the departing trains; we used these plain text files as information source.

Figure 1 presents the main components of the system: (1) the MIDlet application running on the device, denoted *MTrainSchedule*, (2) the remote *Web Service* which answers the MIDlet request, and (3) the *KVV Website* which offers up-to-date schedule information.

The MTrainSchedule MIDlet queries the Web Service through the *WS stub* to retrieve the information on the train departing from the given station. The reply from the Web Service contains basic details on the next trains such as type (e.g. regional train, InterCity Express, etc), departure time, and final destination. The Web Service is responsible of data consistency, updates and proper format. Concretely, the Web Service gathers and parses the latest raw train schedule data from the KVV Website.

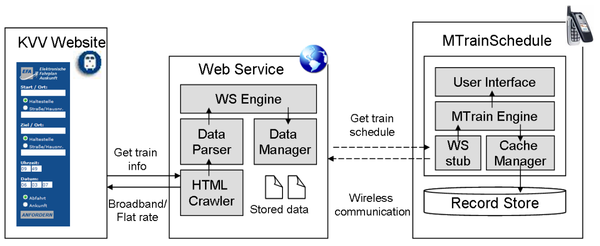

Figure 1: Train schedule application architecture

The *Cache Manager* on the client side is responsible for caching and retrieving data from the persistent storage of the Record Store. If the user queries for the next trains for a given station, the *MTrain Engine* first checks whether the information is already present in the cache. If so, the next departing trains are displayed to the user, without generating any network traffic. Otherwise, the MIDlet queries the Web Service for the latest schedule for the given station and caches the received response.

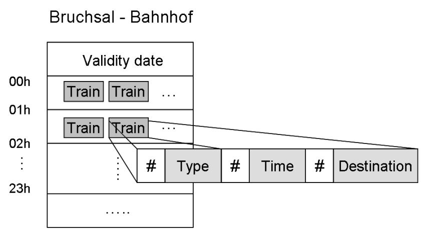

Figure 2: Record Store: Structuring and indexing train schedule information

Figure 2 depicts the index data structure used for caching the train schedule information for one departing train station. The data structure is known and understood by the application - hence, semantic caching techniques are not necessary. Furthermore, synchronization techniques are not required because clients do not modify the provided train schedule. Each train station is saved in a separate Record Store and uniquely identified by the name of the departing station. The first entry contains validity information. The next 24 entries (one for every hour) contain the train partitioned by departing time. For instance, when the application needs to retrieve trains leaving at 14:02h, only the 16th entry in approrpiate the Record Store is read. To minimize storage, train information is binary serialized and contains the train type, departure time and destination.

4 Evaluation

The performance tests presented in this section have been made on a desktop computer using the J2ME Wireless Toolkit [Sun, 2006]. This emulator offers an execution environment similar to the one on mobile devices and allows adjusting the virtual machine emulator speed and network throughput.

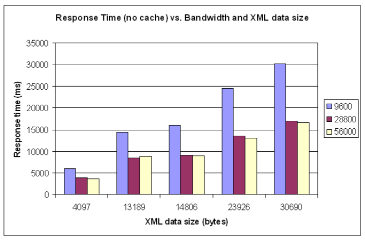

Figure 3: Evaluation – Data size vs. response time

Figure 3 illustrates the response time for answering a query without cache against variable data size. In this case, all data is retrieved from the server.

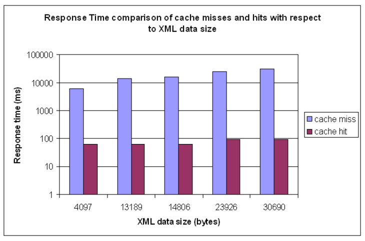

Figure 4: Evaluation – Cache hit vs. Cache miss

[3]http://www.kvv.de/kvv/

Figure 4 compares the response time of a cache miss with respect to a cache hit. Answering a query locally is much faster compared to retrieving the data over the network. The influence of the size of requested data is significantly amortized if the data can be provided from local cache.

5 Conclusions and Future Work

In this article, we presented a service oriented train schedule application using data caching for MIDP enabled mobile device. We discussed the architecture and implementation, illustrated the caching issues and presented first evaluation results. This article is only a first step in terms of caching implementations for mobile clients. The results show that Web Services and data caching on mobile phones are indeed possible and have a huge potential to support the integration of mobile devices into distributed information systems.

Additional extensions of the work should consider allowing the user to set trade-offs of cached data versus data accessed remotely. In this regard, we want to investigate and implement dynamic caching strategies that depend on query history and user preferences, and are location and time-aware.

Remark: This paper is an improved version of the paper [Caracaş et al., 2007] that has been presented at the Studierendenprogramm of the 12th BTW-conference in Aachen. The authors would like to thank Mihaela Ion for her help with the implementation of the prototyp.

References

[Caracaş et al., 2007] Caracaş, A., Ion, I., and Ion, M. (2007). Web Services and Data Caching for Java Mobile Clients. In *Proceedings of the Studierendenprogramm at the 12th GI-conference on Database Systems in Business, Technology and Web, Aachen, March 6, 2007*, volume 1/2007 of *Technical Reports*, pages 10–12, Bruchsal, Germany. School of Information Technology, International University in Germany.

[Ellis and Young, 2003] Ellis, J. and Young, M. (2003). *J2ME Web Services 1.0*. Sun Microsystem, Inc., Santa Clara, CA, USA, final draft edition. Available at http://jcp.org/aboutJava/communityprocess/final/jsr172/index.html.

[Ghosh, 2002] Ghosh, S. (2002). J2ME record management store – Add data storage capacities to your MIDlet apps. Online article at IBM developerWorks. Available at http://www-128.ibm.com/developerworks/library/wi-rms/.

[Giguere, 2004] Giguere, E. (2004). Databases and MIDP, Part 1: Understanding the Record Management System. Online article at Sun Developer Network. Available at http://developers.sun.com/techtopics/mobility/midp/articles/databaserms/.

[Gray et al., 1996] Gray, J., Helland, P., O'Neil, P. E., and Shasha, D. (1996). The Dangers of Replication and a Solution. In Jagadish, H. V. and Mumick, I. S., editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, volume 25 of *SIGMOD Record*, pages 173–182, New York, NY, USA. ACM Press.

[Höpfner et al., 2005] Höpfner, H., Türker, C., and König-Ries, B. (2005). *Mobile Datenbanken und Informationssysteme — Konzepte und Techniken*. dpunkt.verlag, Heidelberg, Germany. in German.

[Lee et al., 1999] Lee, K. C. K., Leong, H. V., and Si, A. (1999). Semantic query caching in a mobile environment. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(2):28–36.

[Sun, 2006] Sun (2006). J2ME Wireless Toolkit. Available at http://java.sun.com/products/sjwtoolkit/.

Iulia Ion was born in Romania and recently obtained her M.Sc. in Computer Science from the International University in Germany. She is currently working on security of mobile and embedded systems at the CREATE-NET research institute, Trento, Italy. For her Master thesis she extended the J2ME architecture to support fine-grained security policy and runtime enforcement. Previous collaborations include the XenoServers public computing platform at Computer Laboratory, Cambridge, UK, and development of a Memory Enhancement application with the Neurobiology Laboratory, Université de Provence, Marseille, France.

Alexandru Caracaş was born at the Black Sea in Constanta, Romania. He earned his M.Sc. in Computer Science in April 2007 from the International University in Germany. Before his Master studies he spent two years as a Software Developer for SAP's search engine TREX part of the NetWeaver platform. Since March 2007, he is working at the IBM Zurich Research Labs where he will continue with his PhD studies. His present field of work includes automated asset and relationships discovery within the IDD project. His general areas of interests are in computer networks and security, distributed systems and Grid computing as well as technologies for mobile devices.

Dr. Hagen Höpfner is Assistant Professor for Databases and Information Systems at the International University in Germany (Bruchsal, Germany). He received his Diplom in Computer Science and his Ph.D. in Computer Science in 2005 from the University of Magdeburg, Germany. His research interests include database theory, transaction processing as well as mobility aspects of databases and information systems in general. He is co-author of the German textbook on "Mobile Databases and Information Systems" and works as a freelance author for various German and international magazines.

Alexandru Caracas
IBM Research GmbH
Zürich Research Laboratory
Säumerstrasse 4
8803 Rüschlikon, Switzerland
xan@zurich.ibm.com

Iulia Ion
CREATE-NET
Via Solteri 38
38100 Trento, Italy
iulia.ion@create-net.org

Dr.-Ing. Hagen Höpfner
International University in Germany
Campus 3
76646 Bruchsal, Germany
hoepfner@acm.org