MOBILE DEVICE LOCATION INFORMATION ACQUISITION FRAMEWORK FOR DEVELOPMENT OF LOCATION INFORMATION WEB APPLICATIONS

Design and Architecture

Andrej Dolmac, Stephan Haslinger
tisco GmbH, Tigergasse 21/13, 1080 Wien, Austria

Schahram Dustdar
Distributed Systems Group, Institute of Information Systems, Vienna University of Technology
Argentinierstrasse 8, 1040 Vienna, Austria

Keywords: Mobile device location information, Location information based services, Location information acquisition.

Abstract: Mobile device location information based services are one of the key drivers in Telecommunication market today. The current development of mobile device location based service solutions is focusing on a development of a particular services, but there is nearly no effort to build a framework that does not focus on a particular location information based service, but instead, provides developers with a set of tools that would enable them easier and faster development of new services based on mobile device location information. Within the EUREKA project MyMobileWeb, framework was implement for acquisition of location information from mobile devices. Framework architecture enables obtaining location information from various mobile devices and is not bound to any special device type or capability. Furthermore the architecture can be used not only to obtain location information, but also to obtain any other information from mobile device, such as e.g. battery level.

1 INTRODUCTION

Mobile device location information based services are one of the key drivers in Telecommunication market today. New solutions, that would extend the usage of mobile device location information for providing value added services, are required.

The current development of mobile device location based service solutions is constrained within several independent projects. The main focus in those projects is put on a development of a particular service. Usually output of such projects is a standalone software, that can not be reused for other different purposes. More effort should be invested in building a platform or a framework that does not focus on a particular location based solution, but instead, provides service developers with a set of tools and libraries that would enable them easier and faster development of new services based on mobile device location information.

One of the major problems and most time consuming steps in development of location based services is implementation of software that acquires the location information from the device itself. Since there is a broad spectrum of different mobile devices on the market, the code for acquisition of location information from the device and its delivery to the server side must often be rewritten many times in order to cover more than one device type. So current mobile device location information solutions usually target only limited set of devices.

This paper presents design and architecture of a framework whose main purpose is acquisition of location information from different types of mobile devices. Framework as such is not focusing on a implementation of a particular service based on a specific device and location information type. Instead, idea of this framework is to provide service developers with a set of tools for acquiring different location information types from various devices without specifying how and for what purpose this information will be used for. Further on, main focus of the framework

Dolmac A., Haslinger S. and Dustdar S. (2009).
MOBILE DEVICE LOCATION INFORMATION ACQUISITION FRAMEWORK FOR DEVELOPMENT OF LOCATION INFORMATION WEB APPLICATIONS - Design and Architecture.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Software Agents and Internet Computing*, pages 74-81
DOI: 10.5220/0001952200740081

is development of Web based solutions. Since at the moment one of the most mature frameworks targeting development of Web based solutions for mobile devices is MyMobileWeb project (MyMobileWebProject, 2008), described architecture is integrated within already existing MyMobileWeb framework in order to provide single uniform solution for development of Web based location information services for mobile devices.

2 PROBLEMS

One of the main steps in development of location based services is acquisition of location information from mobile device. This step involves development of device specific code targeting particular device. The resulting code is usually not portable to other devices since it is dependent on underlaying device hardware, operating system and device driver libraries provided by device vendor. Resulting solutions are usually targeting only restricted and small set of devices available on the market. In order to extend the set of devices that support such service, same functionality must be rewritten multiple times for different devices from different device vendors. This makes development of location based solutions time consuming process which slows down development of new services. So first problem that has to be addressed by the framework design and architecture is covering as many mobile device types with as much reusable code as possible. Second, since described framework is targeting development of Web based services and it is intended as extension to MyMobileWeb solution, communication channel between location information acquisition code running on the device and back end server must be established. Through this communication channel back end server side must be able to trigger acquisition of location information on the mobile device. Through the same communication channel code running on the device must deliver the location information back to the server side.

3 REQUIREMENTS

On one side, MyMobileWeb is based on Web technology for which the client side software is the Web browser installed on the user mobile device. On the other hand, in order to acquire location information from a mobile device, a dedicated software must be run on the device itself. The requirements imposed by this on the overall framework architecture are:

- Server side triggering of location information retrieval from the mobile device must be done through a Web browser as a client side software.
- Communication channel must be established between the server side and the location information acquisition framework running on the mobile device for delivery of location information.

Additionally, there are different sorts of location information types (like GPS position, Cell ID, NFC ID, etc.). Therefore the server side must have a possibility to specify which location information type is required for a particular service. Since there are many different mobile device types on the market with different capabilities for acquisition of location information, there also must be a possibility of detecting which types of location information can be acquired on a particular mobile device. This information must be delivered to the server side application prior to actual acquisition of location information. So the following requirements must be satisfied by the architecture as well:

- Detection of device capabilities regarding location information types.
- Notifying server side about device capabilities.
- Possibility of specifying required location information type by the server side.

Further on, there are many different types of location information and different ways of collecting it from different devices. It is not possible to cover all of it with one monolithic application so architecture must be designed in a way that makes it easily extendible with additional modules. Intention of additional modules is to easily add support for additional location information types and additional location information acquisition methods, as well as providing the possibility to reimplement already existing modules for different device types. This means that an additional architectural requirement is:

- Modular design that is easily extendible.

Finally, it can be useful to acquire location information without this action being triggered from server side and then use it (deliver it to the server side) later on. The final requirement is:

- Possibility to trigger location information acquisition not only by the server side but by the user as well.

3.1 Triggering Location Information Acquisition from the Server Side Application

If the user is browsing Web pages that are not offering any location information services it does not make any sense that the acquisition framework is running all the time on his mobile device and blocking its resources. The acquisition framework should be started only when the user browses a Web page that really offers some location based services. The location information should be acquired exactly at the moment when the server side is ready to process the received information. Mechanism that enables the server side to trigger location information acquisition on the user's mobile phone must be designed. Since the client software on the mobile device is a Web browser, the triggering mechanism must be implemented through means provided by the browser itself.

3.2 Channel for Delivery of Location Information to the Server Side Application

Once the location information acquisition has been triggered and location information has been gathered on the mobile device, a communication channel between the acquisition framework on the mobile device and the server side application must be established in order to deliver acquired data. This can be implemented through the means provided by the Web browser on the users device, or by establishing a direct connection between the acquisition framework and the server side.

3.3 Detection of Device Capabilities

Detection of device capabilities does not only consist of finding out what sort of device it is and deciding which location information can be retrieved from it. It also involves detection of the version of the acquisition framework installed on the device. Based on this information the server side can trigger the desired location information acquisition method or it can inform the user that for the particular service he must install additional software (or simply informing him that the service can not be used).

3.4 Specification of Desired Location Information by the Server

Once information about the device capabilities has been delivered to the server side, the server must inform the acquisition framework about the type of location information it would like to receive (Cell ID, GPS position, NFC ID, etc.) and the acquisition framework must be able to gather the desired location information accordingly.

3.5 Modular Design

There are some common tasks that must be performed by the acquisition framework regardless of the location information type that is used. Those are:

- Detection of device capabilities.
- Communication with the server (reporting device capabilities and delivering acquired data).

Those functions can be grouped together in one common module. On the other hand the actual implementation of the location information acquisition is not only dependent on the location information type, but also on the device type, the underlying operating system and hardware. Therefore the implementation of this functionality should be separated into different modules for each location information type and, in some cases, even for different device types.

3.6 Triggering Location Information Acquisition by the User

It can be useful to give the user the possibility to acquire the location information on his own, when not connected to the server side (offline mode). It should also be possible to deliver the acquired data to the server side later on. For this acquisition framework should provide user interface through which the user can see what type of location information he can get from his mobile device, select the desired type and trigger the acquisition. The collected data must be stored locally on the device until the user visits the Web site that can use stored information.

4 POSSIBLE SOLUTIONS

During design phase two main implementation possibilities were considered and both of them, together with their advantages and disadvantages will be presented in the following sections. Also, a third possibility, which is a hybrid combination of the first two, will be shown.

4.1 Web Browser Extensions based Architecture

This architecture is based on extending the mobile Web browser with a set of plug-ins, through which location information would be acquired from the mobile device and delivered to the server side. The concept of this architecture is presented in Figure 1.

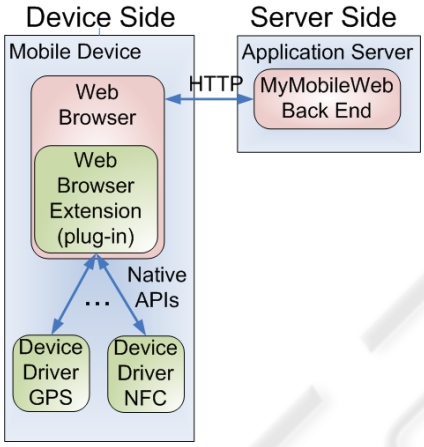

Figure 1: Web Browser Extensions Based Architecture.

Web browser on the mobile device is extended with an additional plug-in that is capable of communicating directly with the device drivers through native APIs in order to fetch location information and send it back to the servers side.

Advantages of this architecture:

- Since MyMobileWeb is a Web based technology, this architecture simplifies integration of location based services into the rest of the MyMobileWeb framework. The easiest way to deliver location information from a mobile device to the server side is to extend the browser capabilities in a way that it is able to acquire location information from the mobile device. Doing this through a plug-in mechanism (provided by most mobile Web browsers on the market) is an obvious and straightforward way.
- This architecture reduces problems with synchronization of the browser and the back end server, since the browser itself is responsible for location information acquisition (synchronisation problems between Web browser and the acquisition framework are more noticeable in the Java2ME Based Architecture, which will be explained in detail in the following chapter).
- MyMobileWeb incorporates AJAX capabilities through usage of JavaScript and this architectural approach makes it easier to implement a JavaScript event mechanism (creating additional events when location information is acquired and available for usage on the server side). This approach eases the usage of location information during the Web site development phase for developers that use the MyMobileWeb framework.

There are some disadvantages of this approach:

- The first big disadvantage is the fact that developed source code is Web browser dependent. There are a couple of different mobile Web browsers on the market today and different extensions would be required for each of them. Whenever a new Web browser appears on the market a completely new implementation from scratch would be needed.
- Most of the Web browsers are implemented as device native applications (OperaMini is one exception) so plug-in development is dependent upon the underlying operating system.
- Reasons mentioned above imply that this architecture is not time resilient. Whenever some Web browser or operating system disappears from the market the source code for this browser or system becomes obsolete.

4.2 Java2ME based Architecture

This architecture is relying on Java2ME technology. In this solution a standalone program is installed on a mobile device and runs as a server that provides location information whenever it receives requests through a standard localhost socket connection. Figure 2 presents this concept.

In this case the Web browser does not need to be extended and does not have the capability to acquire the location information from a mobile device on its own. Instead, there is a separate, standalone software (Java2ME Server Midlet) installed on the device that is in charge for location information acquisition. Whenever the server side requests location information, browser forwards the request to the Server Midlet, which in turn collects the location information and sends it back to the server.

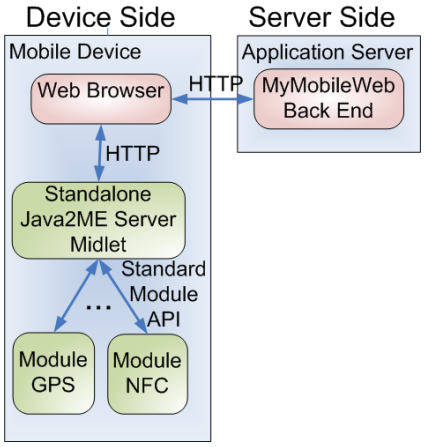

Figure 2: Java2ME Based Architecture.

The differences from the architecture presented in the previous section are:

- The implementation is completely independent of the Web browser used on the mobile device. The only interaction between the Web browser and the standalone Server Midlet is done through classical TCP/IP socket connection through the usage of HTTP protocol. This removes necessity to support different Web browsers individually.
- Java2ME technology ensures operating system independence and source code that is interoperable between different devices.
- This approach is more time resilient than the first one since it's not dependent on the currently existing Web browsers. It will integrate seamlessly with every new Web browser that could appear on the market.
- An additional advantage when using a standalone concept is that the architecture can be extended easily for a off-line mode of work, when fetching of location information is not triggered by the server side but by the user instead. The off-line mode can be supported by adding GUI that is presented to the user in the case that the Server Midlet is started manually (not triggered by the server side). This provides the possibility of collecting location information at any time. Collected information can then be used with any Web site providing location information services.

Although this architecture has some obvious advantages over the previous one, it also introduces some disadvantages and problems that are not present in the first approach:

- This approach complicates data exchange with the back end server. Since the acquisition framework runs as a standalone application on the mobile device, there must be some mechanism implemented to establish a communication channel between the server side and the acquisition framework through a Web browser (this communication channel will be used for triggering a location acquisition from the server side and for delivery of the location information back to the MyMobileweb server).
- In order to request and receive location information the Web browser must connect to the acquisition framework running on the device through a standard TCP/IP socket connection with usage of HTTP protocol for data exchange. This imposes possible time out problems, since in some cases location acquisition can be a time consuming process. If the communication between the Web browser and acquisition framework is synchronous it can happen in certain cases that the session times out if location acquisition process takes too long.
- If the communication between the Web browser and the acquisition framework on the device is asynchronous, which means that the Web browser sends a request to the acquisition framework on the device and gets redirected back to the server side Web page before the location information has been acquired, there is a problem of how and when to deliver the location information to the server side.
- In case of an asynchronous communication, architecture should be designed in a way that it enables the implementation of some sort of event based mechanisms on the server side in order to make usage of location information easier during the Web site development process.
- This architectural approach, although providing big overall benefits, introduces some additional problems. Although none of those problems is technically unsolvable, and none of them represents major drawbacks they should be considered during implementation phase and solved accordingly.

4.3 Hybrid Java2ME and Web Browser Extensions Architecture

This architectural solution is a combination of the architectures described previously. The idea is to combine advantages of both approaches into a single solution in order to minimize problems that both architectures inherently have. The high level idea of this approach is presented on Figure 3.

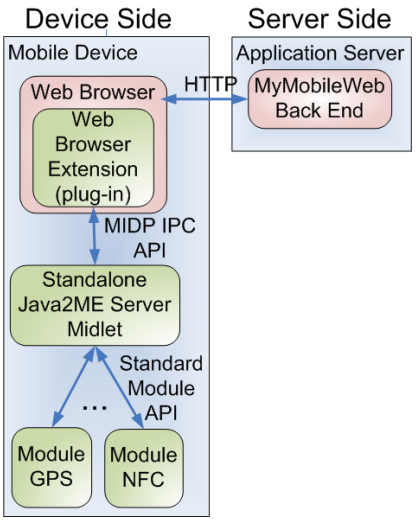

Figure 3: Hybrid Architecture.

This architecture is just a hybrid composition of the previous two. The idea is to try to take the best from both and combine it into a single solution.

This approach has the following advantages:

- Easier implementation of data exchange and synchronization between server and client side through Web browser extensions.
- Easier implementation of Event Based Development mechanisms through Web browser extensions.
- Retaining most of the Web browser and underlying operating system independence through implementing most of the functionality in Java2ME technology and keeping only a small functional parts as a Web browser extensions (implementing 'thin' Web browser extensions with a small code base and using them only as a 'glue' between MyMobileWeb server side and the acquisition framework implemented in Java2ME technology).
- Retaining the off-line mode of work through the Server Midlet.

Disadvantages that still remain in this approach:

- Although 'thin', Web browser extensions are still used. This implies a certain amount of dependency to both, the Web browser and the underlying operating system.
- Heavy prototypes. A lot of time and effort is needed in order to develop a prototype, since it requires the implementation of the whole Java2ME Based Architecture in addition to the 'thin' Web browser extensions for a couple of browsers. Of all considered architectures, this one is the most effort consuming.

5 THE CHOSEN ARCHITECTURE

As a result of different architectural considerations, conclusion was made that Java2ME Based Architecture is the most appropriate one for mobile device location acquisition framework implementation. This decision was made based on following:

- Although introducing some additional problems, the Java2ME Based Architecture has big advantages over the Web Browser Extensions based Architecture in terms of independence. It does not rely on specific browser or underlaying operating system features.
- The implementation of Hybrid Java2ME and Web Browser Extensions Architecture although having certain advantages, is too time and effort consuming for initial framework implementation.
- After Java2ME Based Architecture implementation is finished, it can be upgraded to the Hybrid version. During the implementation phase special attention should be made in order to make this upgrade as easy as possible. Upgrading the architecture should be possible only by adding implementations of 'thin' Web Browser Extensions on top of the already existing implementation.

6 RELATED WORK AND PROJECTS

At the moment there is W3C initiative to specify high level API for acquisition of location information from mobile devices (Popescu, 2008) within the Geolocation API Specification. The Geolocation API Specification defines a high-level interface to the location information associated with the hosting device, such as latitude and longitude. The API itself is agnostic of the underlying location information sources. Common sources of location information include Global Positioning System (GPS) and location inferred from network signals such as IP address, RFID, WiFi and Bluetooth MAC addresses, and GSM/CDMA cell IDs (Popescu, 2008). But this specification is focusing only on the standardization of the high level scripting language APIs for accessing mobile device location, without concerning about low level location acquisition mechanisms. Though, this W3C specification can be used later on, when upgrading from Java2ME Architecture to Hybrid Architecture. In this scenario Java2ME acquisition framework will be used as underlaying layer for low device level location in-

formation acquisition, and 'thin' Web browser extensions will on one side communicate with the Java2ME Server Midlet, and on another side provide high level scripting language APIs specified by Geolocation API Specification.

There is a project by Google, called Google Gears (Google, 2008), that is dealing with development of a framework for the mobile device location information acquisition. Within this project Google implemented a framework that complies with W3C Geolocation API Specification. As a result, Google released a set of Web browser plug-ins that provide high level JavaScript API compliant with W3C Geolocation API. The problem is that at this time Google did not provide any means of extending the functionality implemented by them with additional location information acquisition modules developed by third parties. Only location information types and only devices supported by Google can be used for location based service development. At the moment Google supports Cell ID, WiFi and GPS location information acquisition from mobile devices and only on a limited device set. Additional problem is that Google implementation is Web browser dependent (there is a different set of plug-ins for different Web browsers) so this solution can only be used with Web browsers supported by Google. At the moment this project is being developed and maintained entirely by Google and it's not publicly opened. Extending the existing functionality with support for new location information types and/or support for different device types must be done through cooperation with Google and can not be done by third parties on their own.

Another similar project from Mozilla exists. Project name is Geode (MozillaLabs, 2008) and its main focus is extending the Mozilla Firefox Web browser with location information acquisition abilities. Similar to Google, Mozilla is focusing on development of a browser plug-in that would provide high level Java Script API compliant with W3C Geolocation API Specification. Since the project is in a early phase, there is only one Firefox plug-in developed, that support location information acquisition based on Skyhooks Loki technology (SkyhookWireless, 2008) which uses WiFi to determine location. Similar to Goolge Gears, this project is developed internally by Mozilla, and the current implementation supports only Firefox Web browser and a limited set of devices.

There is one additional project which main focus are not location based services but is worth while mentioning here. This is WebVM project by Aplix Corporation (AplixCorporation, 2008). The scope of this project is to develop a set of Web browser plug-ins that would provide a connection between the Web application environment (Web browser) and the Java runtime that is present on mobile device. Idea is to allow the Web developers to deploy a Java library along with the Web application so the code in the Web application can make calls to the Java library. The reason this project is mentioned here is because its outcome can be used for upgrading the location information acquisition framework from Java2Me architecture into full Hybrid architecture.

As it can be seen, there are couple of project dealing with development of frameworks for location based service development. But, more or less, all of them focus only on providing high level scripting APIs for location information acquisition. None of those projects is targeting development of a low device level framework for location information acquisition that could be easily extended by third parties. All of the mentioned projects are based on extending the Web browser capabilities and are Web browser dependent where one of the main ideas of framework described here is to avoid any dependencies on a particular browser. Besides this, one important aspect of described framework design is to make it modular, make specifications and reference implementations for module development and give them public in order to enable third parties to develop modules for location information acquisition on their own.

7 FRAMEWORK DESIGN AND IMPLEMENTATION EVALUATION

In order to evaluate design and implementation quality, couple of measurements were considered.

Since one major requirement imposed on the framework design and implementation is coverage of different devices, the main measurement for the quality of implemented solution is portability of produced libraries to different devices without necessity for code recompilation. Final framework implementation, once compiled for a particular mobile device vendor Java runtime environment, should work on all phones running this Java runtime environment implementation (of course, modules dedicated for acquisition of particular location information type should work only on devices that support it on a hardware level).

Second, since JavaMobile runtime environments differ from one device vendor to another, some times developed code must be recompiled in order to work on a device provided by different vendor. Usually

only code recompilation is required in order for the library to work on a different device and no code changes are necessary. Another measure for implementation quality is amount of devices covered by only recompiling the developed code. Developed code should work on all devices supporting JavaMe runtime environment (starting from MIDP 2.0 version of JavaME standard and above).

Third, one idea of this concept is to be independent of particular Web browser implementations, meaning that it should work the same way (without extending the Web browser through plug-ins) regardless of the browser available on the mobile device.

8 CONCLUSIONS AND FURTHER WORK

At the moment there are not many standard tools that would ease development of location information based services when location information acquisition from mobile device is concerned. There are server side solutions that are focusing on development Web pages targeted for mobile devices, but none of this solutions is addressing problems when development of code that needs to acquire some information from mobile device is concerned.

Within this paper problems encountered while developing mobile device location information based services were shown. Requirements imposed on a framework for location information acquisition that would run on a mobile device side were discussed and possible architectural solutions were presented. Each architectural solution was discussed together with its advantages and disadvantages.

Amongst considered architectures Java2ME Based Architecture was chosen as the most suitable one for the framework implementation and the reasons for this decision were presented.

Although the initial purpose of Java2ME Based Architecture is to be used for the acquisition of location based information from mobile devices, it must be noted here that this is a general purpose solution that can be easily extended. Any other sort of information that can be collected from the mobile phone can easily be acquired through the same architecture. This is an important notice in respect to the idea of retrieving contextual information from a mobile phone within the MyMobileWeb project. With this architecture it would be e.g. possible even to fetch the battery level of a phone when needed. Just a plug-in for fetching such information would have to be deployed on the phone.

For the time being only prototypes that prove the concept of the Java2ME architecture have been developed. But within the scope of MyMobileWeb project full featured mobile device location acquisition framework will be implemented and integrated within the rest of MyMobileWeb infrastructure.

ACKNOWLEDGEMENTS

This work is supported in part by the European Social Fund and Madrid's Regional Government under their Research Personnel Training programs. Furthermore this work is supported by the Austrian Funding Authorities under their Basic Founding Program.

REFERENCES

AplixCorporation (2008). Webvm whitepaper. Technical report. Available at http://wiki.webvm.net/webvm/whitepaper/.

Google (2008). Google gears. Technical report. Available at http://code.google.com/apis/gears/.

MozillaLabs (2008). Introducing geode. Technical report. Available at http://labs.mozilla.com/2008/10/introducing-geode/.

MyMobileWebProject (2008). Mymobileweb. Technical report. Available at http://mymobileweb.morfeo-project.org/ mymobileweb.

Popescu, A. (2008). Geolocation api specification. Technical report. Available at http://dev.w3.org/geo/api/spec-source.html.

SkyhookWireless (2008). Getting started with the javascript api. Technical report. Available at http://loki.com/developers/documentation.