Consuming Web Services on Android Mobile Platform for Finding Parking Lots

Isak Shabani, Besmir Sejdiu, Fatushe Jasharaj
Department of Computer Engineering
Faculty of Electrical and Computer Engineering
University of Prishtina
Prishtina, Republic of Kosovo

***Abstract*—Many web applications over the last decade are built using Web services based on Simple Object Access Protocol (SOAP), because these Web services are the best choice for web applications and mobile applications in general. Researches and the results of them show how architectures and the systems primarily designed for use on desktop such as Web services calls with SOAP messaging, now are possible to be used on mobile platforms such as Android.**

The purpose of this paper is the study of Android mobile platform, more precisely the ability of this platform for consuming Web services and exploring existing alternatives for consuming Web services from this platform. People use their vehicles every day for transport and this of course leads to a constant demand for finding a parking lot. In this paper is proposed the system, named as *MyParking* through which it is aimed to facilitate users finding a parking lot for their vehicle depending on their current location. *MyParking* consists of three modules: Android client, administration and Web services.

Keywords—Web application; Web services; Android platform; Mobile devices; MyParking

I. INTRODUCTION

Smart mobile devices are increasingly popular the last few years and the total number of mobile devices sold globally is constantly growing. At the same time, the computing power of these devices is growing at an incredible pace and for several years has already reached the power of desktop personal computers. Since many of the limitations of previous generations of mobile devices such as limited memory and persistent storage capacity, limited CPU power, as well as limited and intermittent internet connection and bandwidth are not prevalent anymore, these devices nowadays can be used for advanced applications [1].

Along with the popularity of mobile devices and their growth are created a range of platforms and applications programming environments for them. Different platforms of operating systems like Symbian OS, PalmOS, J2ME, Blackberry, Windows Mobile, iOS and Android, are currently used by equipment vendors in their mobile devices. All these platforms require specific programming language or dialect specific to the implementation of applications. Basically, the only option that is independent from the platform is Java ME for virtual machines which exist in most recent platforms. Unfortunately Java ME is quite old and most of the existing restrictions, are used in its design. This makes Java ME somewhat outdated today but it is the only technology-independent platform. Android supports a set of Java APIs, it uses Java as programming language, it has a broad support of adaption, it has built in components for the graphic of user's interface and a set of key applications available built from third-party developers. A range of service platforms were recently built based on Service Oriented Architecture with SOAP messaging protocol. By creating mobile clients for these platforms it is possible a greater support of system using [2].

Since majority of commercial applications are not pure mobile applications but they rather use mobile clients in distributed and complex software systems, there exists an emergency need for at least implementation of the independent server by distributed application platforms. Web services often are used to provide such an implementation. As majority of clients mobile applications require more subtle interactions and in this way they are often implemented as Representational State Transfer (RESTful) Web services, semantically rich interfaces of commercial applications often use Web services based in SOAP. Hence, in order to be able to use mobile devices of recent time in business processes, it`s important the support of SOAP Web services in platforms of mobile devices. The support for SOAP Web services is not perfect in platforms mentioned above nowadays but there exists other ways to use such services [1]. Whenever an application offers a type of interface that can be called programmatically from another application by sending commands through the Hyper Text Transfer Protocol (HTTP), is said to be an example of Web services [3].

Applications in mobile devices need to communicate with others system components by consuming Web services. Therefore the aim of this project is to explain how is possible to be consumed these services in Android platform, respectively the consumption of SOAP Web services.

II. BACKGROUND AND RELATED WORK

One of the usual functions that is required in mobile applications is to call a Web service to draws the data. This process includes searching of Web service with parameters of getting response. There are two different types of Web services: SOAP and RESTful.

- *SOAP services* usually have a defined contract that is signed or is followed with all structures of data, service

methods and not only them. This contract is written in Web Services Description Language (WSDL) and is published for costumers that use Web services. Also these types of services mainly use EXtensible Markup Language (XML) for requirements and data response.

- *RESTful services* are more ad-hoc than SOAP services because they don't use WSDL and they are based on the standards preliminarily established as XML and HTTP. These types of services are free to restore the data in every possible format, and the communication between them is easier.

A. Consuming RESTful Web services on Android

RESTful web services are simple, scalable, easy to use, attuned to the philosophy of the Web, and able to handle a wide variety of clients [4]. In technical level, Web services can be implemented in Android. Before the implementation particular client of Web services should take in consideration that mobile devices are limited by bandwidth of network and power which is based on the battery. There a lot of headers and layers of SOAP elements in the XML load. So the usage of SOAP services unlike the usage of RESTful client Web services in Android devices is more costly, as for the developer, so for the user. Further, Android SDK offers support for consumption of RESTful Web services by offering libraries/packages in form of HTTP client.

B. Consuming SOAP Web services on Android

There is a considerable number of Web services based on SOAP that are consumed by mobile applications. Especially in the world of enterprises, applications in mobile devices need to communicate with components of other systems by consuming Web services. Android doesn't offer native support for consumption of Web services, but exist a useful library called kSOAP2 which permits Android applications that in an easy and efficient way to consume Web services based on SOAP [5]. This library is third-party library distributed as free source, optimized for Android [6].

In the proposed system MyParking, consume of Web services in Android is realized through kSOAP2. In fact, kSOAP2 [7] is only a project that simplifies usage of SOAP in Android. This library encompasses details of basic layer of transport, offers different mechanisms for (de) serialization of different messages and facilitates handling of SOAP defects. Libraries should be added in project in order to be used. This library is based in SOAP architecture and there is no need to generate any proxy/stub to call Web service methods [6].

There exist different applications for finding parking lots in different countries.

S. Srikanth et.al, [9] proposed a Smart Parking (SPARK) Management System which provides advanced features like remote parking monitoring, automated guidance & parking reservation mechanism. Though prototype system, they proposed the architecture which satisfies the car parking management system requirement.

S. Khang et.al, [10] proposed a parking system in which driver comes to know about the space availability in the parking lot with the help of SMS service. Driver can resend SMS in order to request new space if the previous one is filled. Driver can find nearest space for parking using wireless mobile based car parking system. Results, shows that the system efficiently allocates the slots and utilizes the full parking space.

G. Yan et.al, [11] proposed NOTICE based parking system. In this parking system, drivers can check and reserve the slot for parking. For security purposes encryption/decryption techniques are used. Simulation results are highly efficient.

In Kosovo, actually doesn't exist any application, in any mobile platform for finding parking lots, therefore the proposed system in this paper will find a wide application.

III. System Design

MyParking is an Android application that helps users to find parking lots depending on their location. The main purpose of this application is to offer to users facilities to use application which helps them to find parking places depending on their location. Except MyParking module for clients, there exists also administration module and Web services for communication between client module and the server as well as Web services for communication between parking lots and the server. In the context of this paper client application is an application that is executed in Android mobile platform and which accesses the SOAP Web services server.

A. System Architecture

System consists from administration module and client's module. Administration module is developed in Microsoft .NET Framework 4.0/ASP.NET platform and as programming language is used *C#*. For developing client application on Android platform is used *Java* programming language and other components which are needed to develop Android applications such as *Eclipse* with *ADT plugin* and *Android SDK*. For exchanging the data between client and server are used *SOAP Web services* developed in *ASP.NET 4.0* platform.

System architecture as is shown in Fig. 1 consists of three main parts: Android client, the server and parking lots. In Android client is made registration of client that uses the application, search of parking by proximity, city and address and also the visualization of the data. In server is made managing of parking lots, cities and clients, where from parking lot are sent parking details such as number of free places, prices, the total number of places and identification code of parking lot.

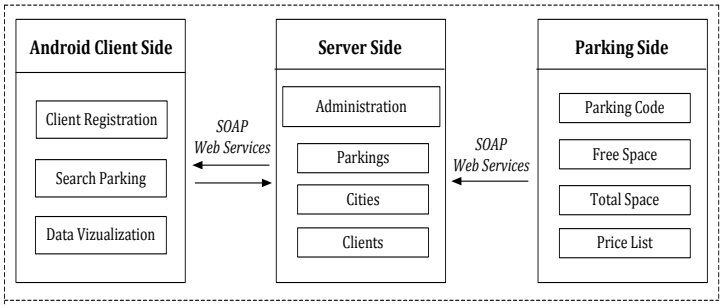


Fig. 1. System architecture

B. Network architecture

In Fig. 2 is presented system network architecture that consists of database server, the server in which is published Web application of administration module, the server in which are published Web services that are used from parkings for sending free places and the server in which are published Web services that are consumed from Android devices (clients) for data visualization about parking places that consist: location, number of free places, the total number of places, price and address.

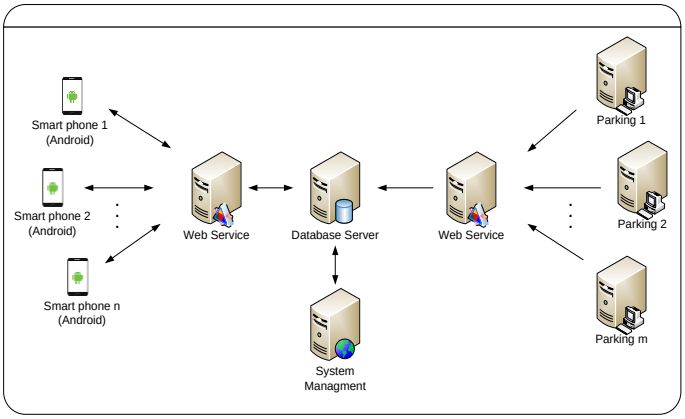


Fig. 2. System network architecture

IV. Database Design

System contains server database which is in MS SQL Server platform and the client database SQLite. In the Table I is presented the list of table's databases of the server and their description.

TABLE I. Description of the Database Server Tables

#	Table Name	Description
1	Parking	Contains information of parking places
2	Cites	Contains the list of the cities
3	Clients	Contains the clients which use the application
4	ClientVisits	Contains the client visits
5	Administration Users	Contains the list of administration module users
6	SyncData	The table which contains information about data syncronization between the client and the serve

Table II presents the list of the database tables of the client and their description, whereas the diagram of database is presented in Fig. 3.

TABLE II. Description of the Client Database Tables

#	Table Name	Description
1	Parking	Contains informations about parking places which are obtained from server
2	Cities	Containsthe list of the cities which are obtained from server
3	ClientInfo	Contains informations of the client that use the application in his device.
4	SyncData	Table which contains informaions about syncronization of database between the client and the server

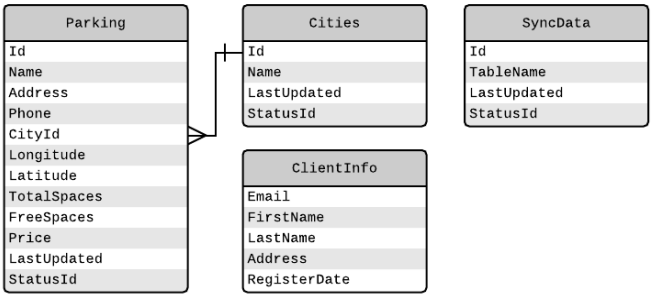


Fig. 3. Diagram of the server database - SQLite

V. Data Syncronization Through Web Services

Communication between *Parkings* and the *Server* is done through *SOAP Web services* [8], the flow of the data is only in one direction, Parking - Server. These Web services consist of two web methods:

- PostParkingFreeSpace
- PostParkingData

PostParkingFreeSpace – consists of two parameters: *parkingCode* (parking code that identifies which parking is sending data) and *freeSpace* (number of free places that actually are in the parking). This web method is called by parkings every time when in parking changes the number of free places.

PostParkingData – consists of three parameters: *parkingCode* (parking code that identifies, which parking is sending data). *totalSpace* (total number of places which are in parking). This Web method is called rarely from parkings, only in cases when the parking changes the total number of places or changes the prices.

Communication between *Android client* and the *Server* is also made by SOAP Web services *Android client* sends and receives data from the server.

Two web methods are developed for sending data in server:

- PostClient
- PostClientVisit

PostClient – through this web method client's data are sent in server in the case of registration. The sent data are: e-mail, first name, last name and registration date.

PostClientVisit – web method is called from Android client when the user opens the Android application, to send in the server the information that the application is used. The data which are sent are: e-mail, date, Android version of client's device and also his actual geographical position (latitude and longitude).

To take the data from the server are developed three web methods:

- ListSyncTables
- ListCities

- ListParkings

Web method *ListSyncTables* lists tables for synchronization, such as the name of the table, and the date of the last update (insert/update) of that table:

TABLE III. LIST OF THE DATA THAT WEB METHOD LISTSYNCTABLES RESTORES

Table Name	Last Updated
Cities	2014-09-30 22:32:50.963
Parkings	2014-10-02 14:00:58.850

Based on this list is easy to understand that which of the tables had changes of the data by comparing the last update with the date which is in SyncTables table in SQLite (in Android device). If the last update which is in the SQLite table is older than the date of the list which restores *ListSyncTables* web method for the specific table, for example Parkings, then is understood that in that table were some changes of the data and the synchronization of the data in that table is needed.

Web method *List Cities* – as input parameter accepts the date which is taken from the table of client synchronizations (where the name of the table is "*Cities*") which can be found in SQLite and restores the list of cities from the server - only in those cities where are added/modified after this date of input parameter. After wards these cities are added/modified in the table *Cities* of the client database SQLite. Synchronization of the cities is accomplished in this way. Similarly functions also parking synchronization by using *ListParkings* Web method.

VI. THE INTERFACE

A. Client application MyParking

Client application MyParking is designed by user's view. User friendly design helps users to achieve their aims. Efforts and aim has been that design must be very simple and understandable for the users. Client applications forms are designed in XML and business logic is written in Java. Google Maps API is used to make easier for the user to find the parkings in the nearest distance with his current position. Users won't need to try a lot to understand the functionality and navigation in the application. Following are presented forms and main characteristics of client application:

- Registration Form
- Home page
- The search of parking lots by proximity
- The search of parking lots by city
- The search of parking lots by address
- The parking lot details form

1) Registration form

Registration form appears to the user only once, after application is installed. Through this form user is registered by giving data such as: first name, last name and e-mail. These data are automatically sent to server through Web services. The Fig. 4 shows the registration form.

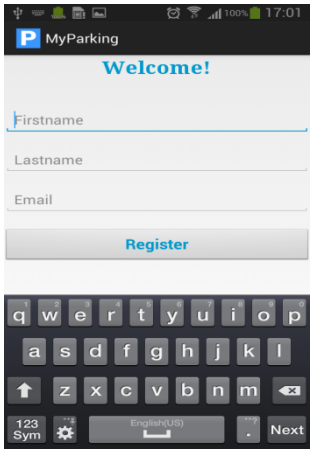


Fig. 4. Registration form

2) Home page

The Home page form is displayed after user`s registration. If the user is registered earlier, this form opens as application`s starting form. At the top of this form is displayed user`s current location on the map, and five nearest parkings if any, while at the bottom are buttons for advanced search (search by proximity, city and address) and the button to update data. This form is shown in Fig. 5.

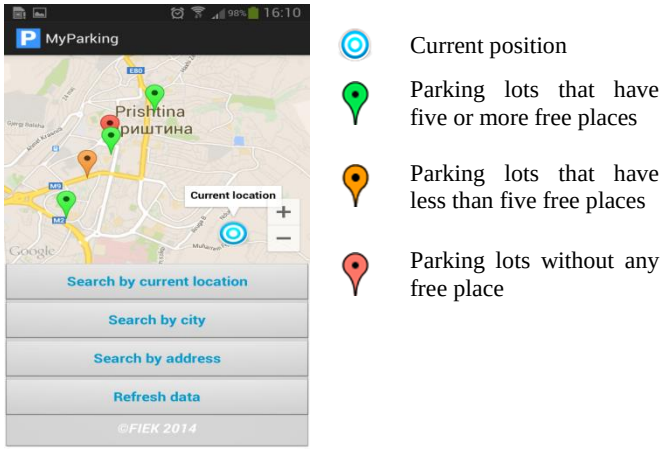
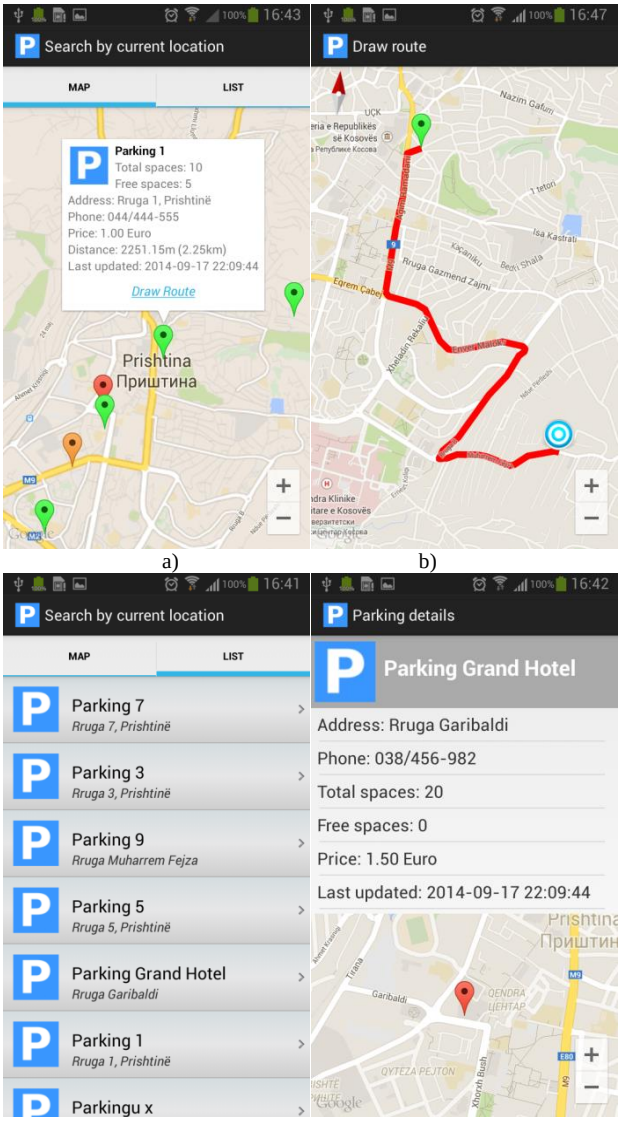


Fig. 5. User's Home page

3) The search of parking lots by proximity

Search for parking lots by proximity is enabled by clicking 'Search by proximity' button from *Home page*. Through this form user is able to see nearest parking lots in visual form in the map or in the list form. Fig. 6 a) shows parking lots close to the current position. By clicking on the particular parking lot, appears the window that contains information about that parking as well as 'Show the path' link through which opens the form as shown in Fig. 6 b) which shows the path form current position to the selected parking lot. The search for parking lots by proximity appears also in the list form as shown in Fig. 6 c), where parking lots are displayed sorted according to proximity. By clicking on particular parking lot opens the parking lot details form as in Fig. 6 d).



a) b)

c) d)

Fig. 6. Search according to proximity: a) parking lots map, b) the path from current position to the selected parking lot, c) a list of parking lots, d) details of parking lot

4) The search of parking lots by city

In this form are listed all cities of Kosovo. To see all parking lots of one city the certain city is selected and the form that contains parking lots opens in visual way on the map and the list form. The Fig. 7 on the left shows the search of parking lots by cities, while Fig. 7 to the right shows on the map parking lots of selected city, which also can be displayed on the list form, similar to the Fig. 6c and 6d.

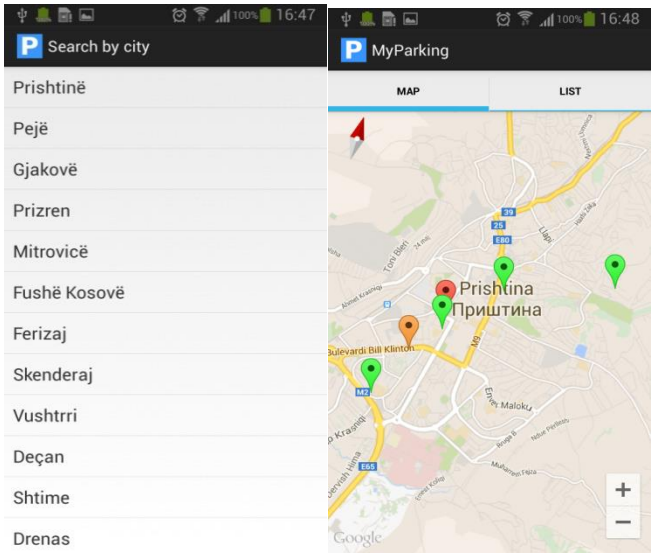


Fig. 7. Search by city form

5) The search of parking lots by address

Through this form the user is allowed to search for parking lots by address. The Fig. 8 shows the form of such a search.

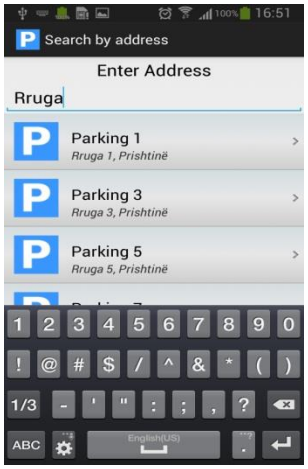


Fig. 8. Search by address form

6) The parking lot details form

Once the parking lot from any of the forms above is selected, than details of parking lot such as: parking lot name, the address, city, phone, total number of parking places, the number of free spaces and the date (timestamp) when the information about parking lot are lastly taken (update date) are displayed. At the bottom of this form appears the parking lot on the map. The Fig. 9 shows the details of parking lot form.

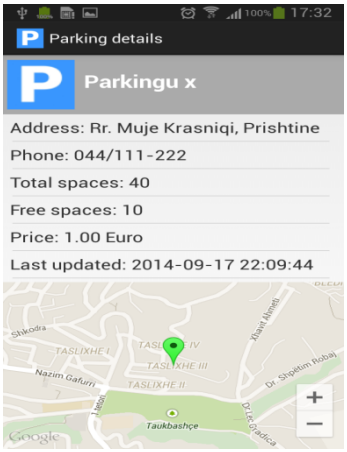


Fig. 9. The form of parking lot details

B. *The administration module*

Management (recording/editing) of parking lots and cities is possible through this module. Through this module also is possible the displaying of customers that use client application.

To login in the administration module is necessary that in advance, the user (administrator) to log in the system by giving username and the password. After the user is logged on the system, a form as in the Fig. 10 opens in which appear parking lots and their details, on the entire territory of Kosovo.

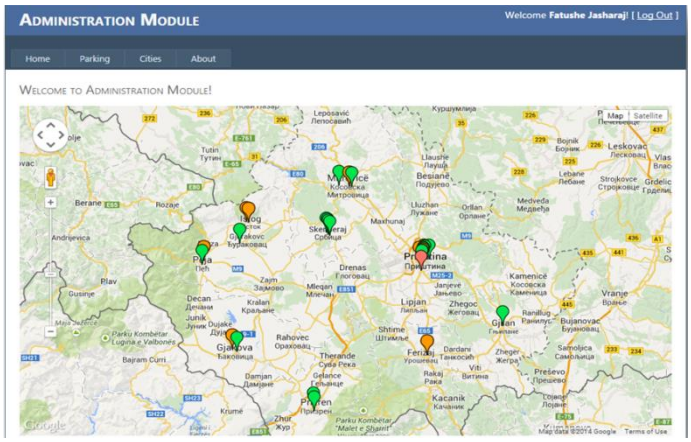


Fig. 10. The map of parking lots in the administration module

Via the form in Fig. 11, administrator registers parking lots and their relevant details, in which case a 5 digit code is generated that identifies the parking lot. When the parking lot sends data on the server about the number of free spaces, also sends this code to identify himself.

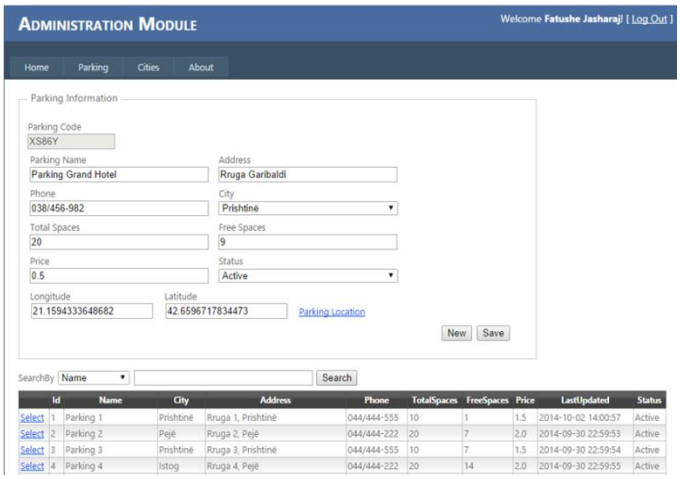


Fig. 11. The form of parking lots management

VII. Conclusion and Future Work

The processing capabilities of mobile devices have increased enormously in the recent years. This makes it possible to build more complex applications that target mobile devices. Recent searches and studies show how architectures and systems primarily designed for use on desktop such as Web services calls with SOAP messaging, now are available to be used on mobile platforms such as Android. With the help of faster and available mobile networks, direct access on Web services with SOAP messaging is definitely possible on Android.

Achieved results in this paper are successful adaptations of the solution, respectively of kSOAP2 library by client side for Web services on Android. It is also important to mention that kSOAP2 works fine when it`s imported into Android, despite the fact that some doubts are raised from developers about the necessity of rewriting some classes of APIs.

Based on the obtained results from our simulation study, we conclude that as from the server side as well as the client side, data synchronization is quick, optimal and effective.

Also, from the obtained results, we conclude that the proposed system can alleviate traffic congestion and reduce the amount of traffic volume caused by searching for parking.

In the future we will try to work and also explore other possible alternatives for consuming Web services on the Android platform and their support for this platform and other mobile platforms.

References

[1] C. Kleiner and Th. Schneider, “Securing SOAP Web Services for Mobile Devices on Different Platforms,” MMS 2011: Mobile und ubiquitäre Informationssysteme. Proceedings der 6. Konferenz, 28. Februar 2011 in Kaiserslautern, Deutschland. GI 2011 LNI ISBN 978-3-88579-279-6, vol. 185, pp. 25-38, 2011.

[2] J. Knutsen, “Web Service Clients on Mobile Android Devices, A Study on Architectural Alternatives and Client Performance,” p. 19, June 2009.

[3] U. Cei and P. Lucidi, “Alfresco 3 Web Services Build Alfresco applications using Web Services, WebScripts, and CMIS”, Packt Publishing, ISBN 978-1-849511-52-0, p. 8, 2010.

[4] L.Richardson and S. Ruby, “RESTful Web Services”, Published by O’Reilly Media, Inc., ISBN: 0-596-52925-0, p. 299-314, 2007.

[5] Th. Weerasinghe, Introduction to working with kSOAP2 in Android, thiranjith.com. [Accessible on October 20, 2014]

[6] P. Pocatilu, "Developing Mobile Learning Applications for Android using Web Services, Informatica Economica Journal, Vol. 14, No. 3/2010, pp. 106-115, 2010.

[7] J. Bertram and C. Kleiner, "Secure Web Service Clients on Mobile Devices", MobiWIS 2012: The 9th International Conference on Mobile Web Information Systems, vol. 10, pp. 696-704, 2012.

[8] I. Shabani, B. Çiço and A. Dika, "Solving Problems in Software Applications through Data Synchronization in Case of Absence of the Network" IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 3, January 2012.

[9] S. Srikanth, P. Pramod, K. Dileep, S. Tapas, U. Patil, N. Babu, "Design & Implementation of a Prototype Smart Parking (SPARK) System using Wireless Sensor Networks", International Conference on Advanced Information Networking & Applications workshops, 978-0-7695-3639-2/09, IEE, 2009.

[10] S. Khang, T. Hong, T. Chin and Sh. Wang, "Wireless Mobile-Based Shopping Mall Car Parking System (WMCPS)", Services Computing Conference (APSCC), IEEE Asia-Pacific, pp. 573-577, 2010.

[11] G. Yan, S. Olariu, M.C. Weigle and M. Abuelela, "SmartParking: A Secure and Intelligent Parking System Using NOTICE", Intelligent Transportation Systems, ITSC 2008, 11th International IEEE Conference on, pp. 569-574, 2008.

[12]