Context Oriented Application – Concept and Implementation

Grzegorz Siewruk [1,2], Marek Średniawa [1], Sebastian Grabowski [3], Jarosław Legierski [3]

(1) Orange ul. Skargi 56 03-516 Warsaw, Poland
Email: Grzegorz.Siewruk@orange.pl

(2) Warsaw University of Technology Faculty of Electronics and Information Technology
ul. Nowowiejska 15/19 00-665 Warsaw, Poland
Email: mareks@tele.pw.edu.pl

(3) Orange Labs, ul. Obrzeżna 7, 02-691 Warsaw, Poland
Email: sebastian.grabowski@orange.com, jaroslaw.legierski@orange.com

ABSTRACT: The paper presents an idea of a context-aware application, which collects context data from many different sources, stores them in a dedicated database and makes use of it to support flexible scenarios for end users. Using open APIs it integrates different types of context information provided by: Unified Communication system, APIs exposed by communication service providers and information from Machine to Machine (M2M) framework. Methods for recording and unifying different types of context data are proposed and their performance is compared with results for the most popular database structures. A context-aware contact list application for a mobile phone user is presented as an example illustrating the main ideas of the paper. This paper is extended version of publication "Integration of context information from different sources: Unified Communication, Telco 2.0 and M2M" which was presented during Computer Science and Information Systems (FedCSIS 2013).

KEYWORDS: context information, context of communication, API, Unified Communications, M2M.

1. Introduction

Nowadays, value of information about context of communication is growing and having more impact on next generation of context aware services. Companies such as Google, Twitter or Facebook which are in possession of large databases which contain mostly context information [1] became major players in the ICT market. Information about users and their activities is the base of social networks' owners' business model and has started to be monetized, e.g. mainly in personalized focused advertising.

Availability and usage of precise context information has a very big potential for enterprises and constitutes a catalyst for innovative ICT systems and applications [2]. From a mobile phone user's point of view, participation in social networks implies a change of a lifestyle. People want to "stay in touch" which means that they have to be reachable via phone independently of their location. Most of the businessmen (but not only) can't imagine working without multiple phone numbers: a personal number to contact family, an office phone number to contact coworkers and a business number to communicate with customers.

The idea of easy contact for business users is one of the principles of Unified Communications – a very popular communication trend in enterprise area. Typically it is implemented with:

- One number service – functionality allows the user to define and manage preferred communication devices via voice and video. Despite of using different terminals, subscriber is identified by one universal telephone number.
- Integrated application – single application supporting in a unified way many different communication channels: voice, video, e-mail, voice mail, Instant Messaging and web collaboration.

Efficient usage of Unified Communication systems and their features requires interaction with the end system user. Usually a user has to pay attention to and consciously set his or her status in UC application (ready, out of office, on holiday etc.), set and activate auto response in corporate e-mail system or set an active phone (office phone, mobile phone, softphone etc.).

From the end user perspective, these processes preferably should be automated to ease him/her of manual setting of an active phone. All information required for Unified Communication system can be acquired by sensors or software components taking advantage of context information.

The idea of automatic information exchange between devices and sensors is similar to the smart home automation concept, and M2M systems are a very promising source of context information e.g. for potential commercial systems. Vision of database which is meant to store information which are coming from smart home solution (e.g. electric energy meter), mobile network operator along with internet service provider is very attractive. Number of solutions which could make use of database like this is impossible to count.

2. Background

Main purpose of this paper is to propose a method of storing information about context of users which came from number of different sources (Unified Communication systems, Mobile Network Operator API and Machine to Machine framework) and integrate them in shared database. Context information mentioned before has different structure depend on source and thus method of storage those data must vary. Source like UC system provide information in asynchronous way while on the contrary the MNO API is synchronously sending requests to the server which is exposing the API. Database which contains such information will be heavily loaded so proper way of gathering and next processing data is crucial for the solution efficiency.

2.1. Existing solutions

Many context-aware computing paradigms have been developed in the recent years. For example, Cisco context-aware Healthcare solution, which is a tool for monitoring and simplifying business operations in hospitals [3]. However the most popular context-aware solution is currently the Google Ads Gadget, which displays context based personalized commercials to users browsing a specific website [4].

Cisco solution [3] uses RFID [5] (Radio Frequency Identification) technology for real time tracking of patients, medical personnel and hospital assets, what is crucial for providing the best quality health care services and optimization of workflow and medical staff management. The uniqueness of the approach stems from the fact that Cisco health care solution treats equipment and machines, such as X-ray or wheelchairs, as objects which are considered as the source of the context information. Context aware systems also bring new features into the survey conducting area [6], [7], [8], [9]. In this field, gathering context data and taking advantage of data collected by mobile devices, such as smart phones or tablets, allows developers to invent useful mechanisms which in turn allow to specify the exact target group of a survey. A solution presented in A survey on context-aware systems [6] uses a variety of many flexible sources, including widgets, wireless sensors and middleware infrastructure, which is somewhat similar to the mechanism presented in this paper.

2.2. Unified Communication

Unified Communication [10], [11] systems constitute an important element of private communication networks dedicated to enterprises. Many vendors (e.g.: Avaya, Cisco, IBM, Microsoft, Oracle and Siemens) offer comprehensive UC solutions. An example of such system developed by

Unify (formerly Siemens Enterprise Communications) is OpenScape UC application.

UC system offers many features, e.g. standardization of communications services, support for user mobility, communication medium neutrality and support for any type of equipment (type of terminal). Possibility of virtualization of resources and services and their exposition by using LAN, WAN or Internet networks is of utmost importance for business users. Openscape UC (Fig. 1) offers VoIP (Voice over IP) services (implemented in the OpenScape Voice subsystem), video communication between employees (OpenScape Video) and integration of e-mail and voice in a single message box (OpenScape Messaging). Integration of external applications and systems with OpenScape system is possible due to availability of Application Programming Interfaces (APIs), implemented in the UC system. Its APIs are exposed as Web Services and are offered using a Service-oriented architecture model (SOA) and SOAP Protocol (Simple Object Application Protocol). By taking advantage of UC open APIs, external developers can create their own specific application and UC system extensions described in [11] and [13].

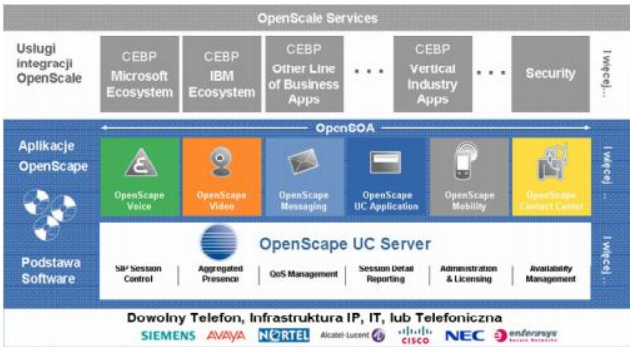

Fig. 1. OpenScape UC system architecture [11]

Openscape UC system have been equipped with large set of Application Programming Interfaces exposed as Web Services in SOA model. Some of UC API used in context based application are described in Tab. 1.

Presented in this paper context application requires implement a large part of the API function offered by Openscape UC. Openscape platform supplier besides the API in Web Services form, offers WS description in WSDL (Web Services Description Language) and Software Development Kit (SDK). All SDK's functions are secured and an account on the UC platform with administration privileges is required for API calls (for example for check status of all users). SDK for Openscape UC is offered in two version form MS .Net Framework (written in C#) and for Java. Context application implements SDK functions in MS .Net variant.

2.3. Communication Service Providers APIs

Telco 2.0 [14] is an idea based on exposition of Communication Service Providers APIs for external developers and third party companies on the Web which provide access for functionality like getting GPS location. Even if this method is not as precise as GPS module built in the device it still allow developer to be able to track user even if he does not have signal from satellite (being inside of building). From the technical point of view, telecom operator's APIs are exposed using a dedicated entity – a Service Delivery Platform (SDP). An SDP is located between the operating network and the Web. Its southbound interfaces are connected to the operating network and integrate it with enablers offering telecommunications capabilities (e.g.: SMS, MMS, USSD, voice call, etc.). The northbound interface exposes open APIs as Web Services on the Web. The SDP is also connected to an Operations Support Systems (OSS) responsible for: maintenance, inventory, provisioning and fault management as well as Business Support Systems –BSS (responsible for order handling, billing, payments etc.). There are two ways of exposing Telco 2.0 APIs – either via SOAP [16] interfaces or by using REST architectural style [15].

Both SOAP and REST approach have advantages. SOAP web service require less plumbing code then REST one and among with standards like JAX-WS [28] or AXIS [29] is really easy to create client solution. However RESTful web services are completely stateless and provides good catching infrastructure, which is good for synchronous calls which are used in the location module. That is why RESTful architecture is being used in the presented solution.

Telecommunications Service Delivery Platforms (SDP) usually allow access to their capabilities in the service-oriented architecture (Parlay X specification [17]) or resource oriented Web Services (RESTful) model compliant with the OneAPI specification [18]. By using open APIs, it is possible to develop new applications and services merging IT and the Internet world with telecommunication area [13], [19], [20], [21], [22].

The Tab. 2 shows basic telco APIs set exposed by Communication Service Providers based on One API version 2 standard [18]. It should be noticed that only two APIs from basic set are implemented in the context application.

The GSM Association (GSMA) from a business point of view divides APIs into the following categories [24]: user experience improved, enriched functionality and business model enabling. Main context based API – mobile terminal location and is categorized as an enriched functionality enabler. Second API implemented in context application - customer authentication is defined by GSMA as very important API which besides user experience improvement have large impact on business model. Tab. 3 describes API dedicated for RCS (Rich Communication Suite). RCS is client application for next generation

communication environment for Communication Service Providers based on packet transmission in IP network (VoIP). Main communication VoIP system for telco operators IMS (IP Multimedia Subsystem) is equivalent to dedicated for private networks "softswitch" - OpenScape Voice mentioned in chapter 2.2.

Tab. 1. Implemented Openscape UC Application Programming Interfaces [12]

API function	Description	Implemented in context application
Logon/logoff	This function allow logon and logoff UC user	Yes
Contacts handing	Allows a user to create, find and manage contacts	No
Presence	Allow get or set the presence status of the subscriber. API offers two main capabilities: set presence status, presence status notification	Yes
Eventing	This service provides the eventing mechanisms. Application can subscribe for specific events that are provided by the various services, e. g. subscribe to presence change event or call notification.	Yes
SetPreffered Device	Device Management expose following functions: set preferred device for incoming call, provide user's device lists, add, delete and modify device or device list	Yes
Makecall	This API implements set of call control functions: click to dial, click to drop, click to answer, click to conference, call status information	Yes
VoiceConference	This API is used to create and manage conferences: scheduled conferences and ad-hoc voice conferences	No
TellmeWhen	Allows a user to request to be informed when a certain event occurs, e.g. when the subscriber terminates the connection or change of their status	No

Tab. 2. Implemented Communication Service Providers APIs Application Programming Interfaces

API function	Description	Implemented in context application
Authentication API	Service for authenticating API users	Yes
Send/Receive SMS	This function allows sending and receiving SMS messages	No
Send/Receive MMS	Allows sending and receiving MMS messages	No
Send/Receive USSD[1]	Allows sending and receiving USSD (ang. Unstructured Supplementary Service Data) messages	No
Terminal Location	The Location API allows to geolocate a mobile device	Yes
Voice Call Control	Allows to establish and manage calls between two telephones	No
Payment API	Allows external applications to charge, reserve or refund an amount to an CSP customer's account	No
Data Connection Profile	Service for retrieving the user's profile data connection attributes (GPRS, HSPA, etc.)	No
Device Capability	Allow retrieving devices capability by brand/model, and to get device data	No

2.4. M2M approach

The Machine to machine (M2M) concept refers to technologies that allow both the wireless and wireline systems to communicate with other devices with the same capabilities. M2M uses devices, such as e.g. sensors or meters, to capture events or current state indicators (temperature, inventory level, etc.), which are then relayed via a network (wireless, wireline or hybrid) to an application (software program), that translates them into meaningful information (for example, items need to be restocked) [25]. Such communication was originally implemented using a remote network of machines which relayed

[1] USSD API is not part of One API specification, but because their usability [23] have been included in the Tab. 2.

information back to a central hub for analysis, which would then be rerouted into a system like a personal computer.

Tab. 3. Telco APIs categories from business point of view [24]

Operator capability \Application impact	User experience improve	Enriched functionality	Business model enabling
Messaging (SMS, MMS)		X	
Voice calls		X	
RCS (IP messaging, video share, file share)		X	
Payment / charge to mobile bill	X		X
Customer authentication	X		X
Customer profile attributes	X	X	
Terminal Location	X	X	

3. System architecture

Solution which is presented in this paper is integrating data which is being gathered from three different sources (Fig 2). First source is Unified Communication system. SDK provided by Unify allows to build service which store every event which reflect user activity like changing of user status (e.g. busy to available) or changing status of phone line. SDK is also implementing method which could establish the connection with selected number. Functionality like this gives the opportunity to implement "Click to call" button.

Next source is Mobile Network Operator API allow to collect context information related to user terminal location by using API described in *Tab 2*. The last source is M2M framework which could generate endless number of information like knowledge about presence, identification of threads or simply create a user profile based on his habits.

This chapter presents the main modules comprising the system collecting context information from different sources.

3.1. Database

The data base is the most important part of context application. It plays the role of the data repository collecting context records from three mentioned above sources. The database has to be: scalable, easy to access and fast. Meeting

these requirements is easy in case of small data-bases but must be relaxed in case of large systems handling thousands or millions of records.

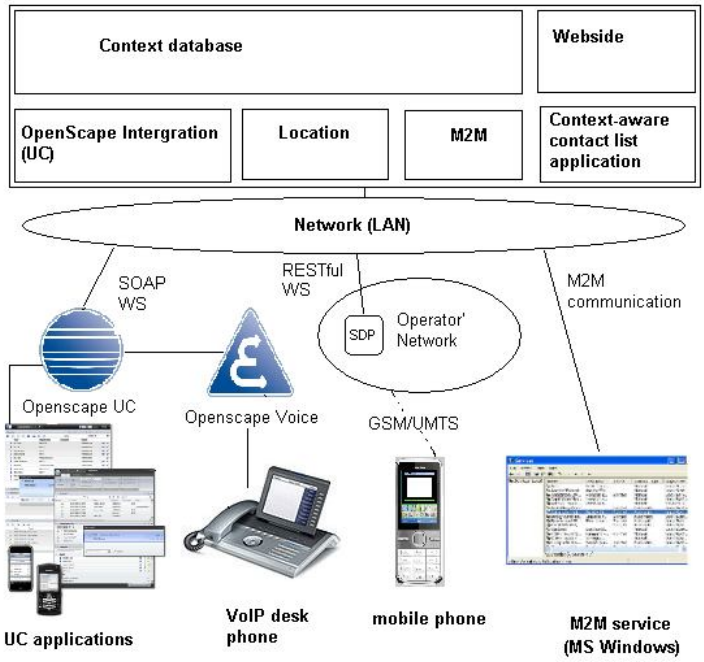

Fig. 2. Architecture of system

The implemented MySQL database uses a relational data model and specifies four main entities (Fig 3):

- User – entity stores information about the user: login information for OpenScape management portal, name, surname, Openscape VoIP terminal and mobile terminal numbers as well as the home address.
- Context_data – stores context information: timestamp and context data set which is JSON form of context information (for example latitude and longitude).
- Context_type (Fig 4) – a table which contains information about the context data source.
- Param_type – a table which assists in detection of the type of parameters coded in context data set (JSON).

The presented database schema guaranties to be scalable – e.g. there is no need to add new table when the system is integrated with a new context information source in the future. Data sets are easy to access because of the usage of JSON [26] format, which is supported by many programming environments.

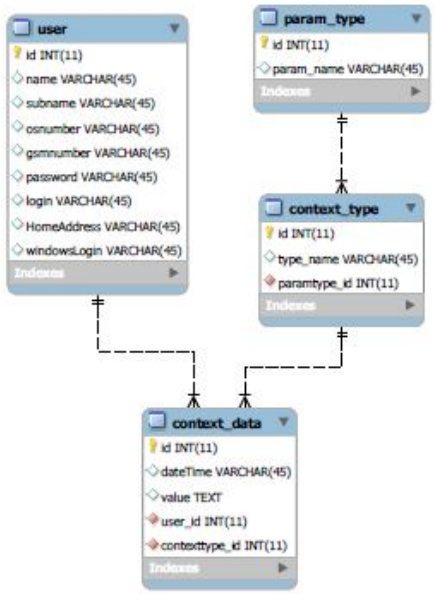

Fig. 3. ER diagram of database

id	name	paramtype_id
1	openstageMediaStat	4
2	openstagestat	5
3	gsmlocation	1
6	systemlogin	6

Fig. 4. Context_type entity

3.2. Location module

The location module is responsible for collection of information about a mobile phone's geolocation. The location data is collected using terminal location API calls, based on those exposed in the Internet by Communication Service Provider using a dedicated Service Delivery Platform (SDP). The response received from Terminal Location API contains approximated latitude and longitude of the mobile terminal. The location information is stored in the database using pooling method. The measurements are repeated every minute. Unfortunately, the pooling method is not very efficient in terms of performance, but currently there aren't any other methods available in Poland for small developers and external companies that could be exposed on the Web by Communication Service Provider in the Telco 2.0 model.

3.3. OpenScape integration module

Unify - the Openscape UC supplier, provides a high level Software Development Kit for programmers, which allows to implement a methods responsible for integration with a UC system. Its main role considered here is implementation of a method, which captures events in relation to the change of media (terminal) status or user status in UC system. There are three possible states of media status: 0. UNKNOWN (when the observing user is not authorized to see the recipient status) 1. AVALIABLE (when there is no call on the phone line) and 2. UNAVAILABLE (when the user is already busy in a call). The events, like e.g. change of the status are captured and stored in the database with a timestamp and an appropriate JSON value.

The second parameter correlated with the user activity by the presence API is the user status. There are 6 possible states of user status: 1 - Do not disturb 2 - Be right back 3 - Unavailable 4 - Busy 5 - In meeting 10 – Available. This parameter is stored in the database – when an event (triggered by the change of the status) appears. The user status information is stored (in the JSON format) with a timestamp in the database.

Another method implemented in this module is the click to call feature. This method requires user login and password information to recognize an active phone from which the call originated (defined in the system OpenScape UC as a logged user terminal with activated One Number Service function).

3.4. M2M module

This application was implemented in Windows Service form which starts with the user's login and ends with the system user's logout. The module collects information about the user logged on to the workstation. The name of the user is sent to the main system and is stored in the database, using timestamp in JSON format.

3.5. Module Website – end user application

Module Website – the graphical user interface, was created using ASP.NET framework. The module contains 4 subpages: login.aspx, contactlist.aspx, adduser.aspx and about.aspx. Only a successfully authorized user is able to use the portal. This module implements login and password; the ones required by portal are the login and password from OpenScape UC system. If the logged user is also the administrator, he or she is able to manage the system's administration panel, e.g. add or remove users to the system, (Fig. 5) define work address and home address.

Fig. 5. Module Website - add user form

To add a user, all fields are required, with the condition of setting a correct login and password (user credential must be the same as defined in OpenScape UC platform). The ending user application of the website is contactlist.aspx which contains all contacts, context data and the click to call button (Fig. 6).

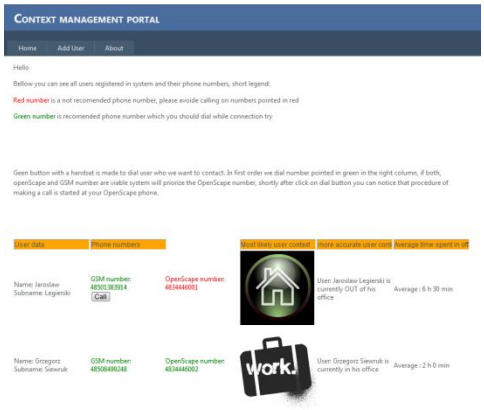

Fig. 6. Context-aware contact list application

4. Simple context based decision algorithm

This chapter, based on the contact list application features, presents a simple context-based decision algorithm. The first column of this end user' web application (Fig. 6) contains contact data, such as the name and surname from the database. The next column named "Most likely user context" is a field based on information acquired from the Communication service Providers location API.

The application uses Google Map API to recognize the home address and work address based on geolocation (longitude and latitude), which is recognized by using the distance between the user's current location (the latest database record) – and simultaneously between the current location and home location. If the distances calculated above are less than 1000m, the algorithm determines that the user is in the defined location (work or home).

The next column - "More accurate user context" - contains information from the data collected from M2M application. If the user is already logged on to the workstation, the algorithm decides that the user is in office and vice versa: if the user is not logged on to system, the result is interpreted as "out of office".

The column "Average time spent in office" is based on the data received from M2M application. The application measures the time from the timestamp between login and logout, and calculates an average value.

The last column named "Phone numbers" contains all phone numbers defined for a specific user. If the phone number is underlined in red – the number is not recommended to dial, when it is in green – the number is recommended for use.

There are several conditions that must be met for the number to be underlined in red: for the context information from OpenScape UC – AggregatedMediaStatus attribute must have value Unavailable; userStatus – In Meeting, Unavailable, Busy or Don't disturb; however, if AggregatedMediaStatus is Available but the user is out of office, the number will also be underlined in red.

Finally, by clicking the "call" button, the user can dial the number which is underlined in green. When all numbers are highlighted in green, the number allocated to Openscape UC system (office number) has higher priority.

5. Information integrity and security

According to [30] Database leaks with personal information is third most likely thread in cyber space. Along with fact that this kind of attack is one of the most common one gives us appropriate reason about considering setting up proper security methods.

In the previous sections we have described what could be done with properly processed context information. Under those conditions we can categorize this kind of data as critical in term of system security. There is no need to explain what harm could have be done if only unauthorized person get access database like this.

5.1. Context Conflicts

There is possibility that context information which are obtained from number of different sources may conflict. Situation like this can happen for

example when subscribed user by mistake leaves his mobile phone at home. In this case M2M module has discovered user as logged in and present in office however Telco 2.0 module is pointing that user is being at home.

It is extremely important to deal with this kind of situations properly because any inconsistency has impact on reliability of presented solution. For example in the scenario described above, source which can authenticate user is selected as stronger source and other conflicted information will be ignored (however user will be informed about context conflict - e.g. that there is something wrong with his devices). In order to be able to use M2M module, user has to be successfully logged to his computer by providing correct password or plugging his USB token with certificate, which in integration with the authorization mechanisms provided in the Windows Domain (verification based on Active Directory database) implies that this context source has a higher priority. Submitted solution is efficient but using semantic algorithms should be considered.

5.2. Information Security

As it is described in the previous chapters presented system is processing sensitive information, wherefore there must be implemented methods which prevent to loss them.

There are several mechanisms to perform these tasks both the application side and server side. Below is list of basic requirements to provide security for this kind of environment.

Application:

- Enforce users to use strong passwords (Upper-case letters, special signs, numbers),
- Enforce users to change their passwords every 30 days,
- User roles which give the certain user possibility to perform only allowed operations.

Server:

- Antivirus software up to date,
- Configured firewall,
- Database calls can be made only from the specific IP addresses (website).

In Context-aware contact list application requirements for strong passwords and regular change of password is possible to achieve only if administrator of OpenScape system implement such policy in OpenScape panel (password for application and OpenScape is the same).

Application is immune to the most common SQL- injection and Cross Site Scripting (XSS) attacks.

6. Measurements

6.1. Performance tests result

In order to verify the SQL database structure presented in chapter III, some performance tests had been run. The proposed database scheme was compared with a classical database structure based on 5 independent tables (user, gsmlocationcontext, openscapeaggmediacontext, openscapestatuscontext, m2mcontext) and filled with context information.

The first test presents time needed to load all the context datasets to the memory. In order to get all the information about a specific user, 4 SELECT statements were executed, whereas getting all the possible information in the presented solution required only one SELECT statement. The results are presented in Tab. 4.

Tab. 4. SQL query response time (4 context sources)

Select all context			
Context-aware contact list database		Standard context database	
Table length [bytes]	SELECT duration [s]	Table length [bytes]	SELECT duration [s]
500	0,00501125	500	0,047842
1000	0,01011150	1000	0,055079
1500	0,01409725	1500	0,206002
2000	0,02308850	2000	0,179314
2500	0,02743775	2500	0,275252

The results presented above show dependency of the execution time on the table length. The proposed database structure is more efficient because it decreases the number of SQL Select statements (Tab. 5 and 6). The execution of one SELECT statement is faster than the execution of four SQL Select statements.

In the next test, the performance of a database based on the information from only one context source was measured. The SQL queries used during the test are presented in Tab. 7 and 8 respectively. The results of the measurement are presented in Tab. 9.

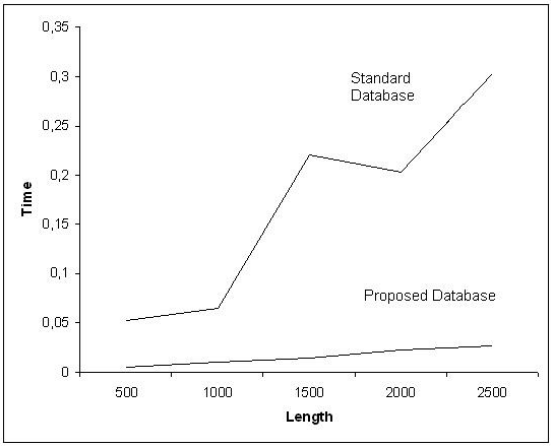

Fig. 7. SQL query response time. (4 context sources)

Tab. 5. SQL query executed for the proposed DB scheme (4 context sources)

SELECT * from contextdata where user_id =1

Tab. 6. SQL query executed based on standard solution. (4 context sources)

SELECT * from gsmlocation where user_id=1; SELECT * from openscapestatus where user_id=1; SELECT * from openscapeaggmediastatus where user_id=1; SELECT * from m2mcontext where user_id=1

Tab. 7. SQL query executed for the proposed DB scheme (1 context source)

SELECT * from contextdata where user_id =1 and cotnexttype=1

Tab. 8. SQL query executed for the standard solution (1 context source)

SELECT * from gsmlocation where user_id=1

Tab. 9. SQL query execution time (1 context data source)

Select all context			
Context-aware contactlist database		Standard context database	
Table length [bytes]	SELECT duration [s]	Table length [bytes]	SELECT duration [s]
500	0,011354	500	0,019467
1000	0,020775	1000	0,02147375
1500	0,02485975	1500	0,04349075
2000	0,0322245	2000	0,05073
2500	0,04947975	2500	0,05930075

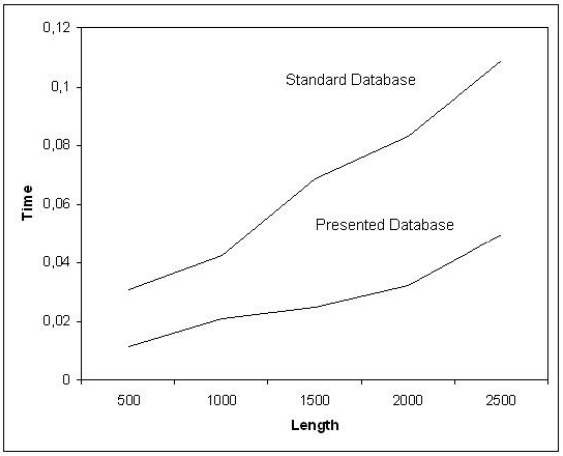

Fig. 8. SQL query response time (1 context data source)

Using the proposed context notation in the JSON format results in the execution time of SQL SELECT function being shorter than in the case of the standard notation. It must be pointed out that in the presented solution, the table always has 4 columns and this value is independent from the number of parameters represented in the context information (context data source).

6.2. Functionality test results

Fig. 9 and 10 present the results of functionality tests. The maps present the location of selected points where the service was invoked, and show the location received using the Telco 2.0 location API as well as the potential location recognized based on the user's activities in the Unified Communication system:

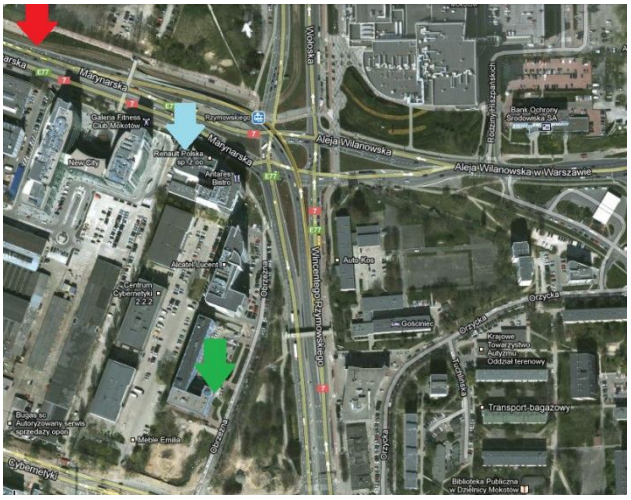

Fig. 9. Sample result – "user out of office"

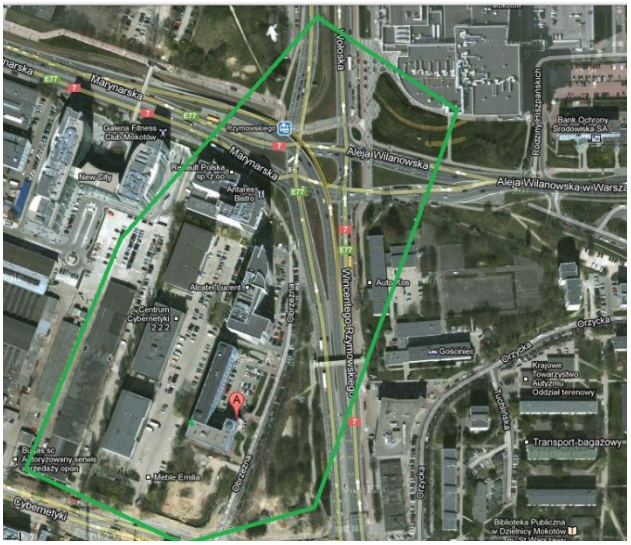

Fig. 10. "user in the office area " – approximate range

- Blue arrow - real location of the user.
- Green arrow - location recognized based on the information from UC system.
- Red arrow - location recognized based on the mobile network.

For the presentation of the test results, Google maps application was employed. Fig. 9 shows the case in which the user forgot to change his or her UC status, and left the office. Based on the mobile terminal location services, the presented context aware system can, in this case, change the status of the user in the UC system automatically from "in the office" to "out of office". Fig. 10 presents the approximate range within which the user is considered to be in his or her office. When the user passes the border line highlighted in green, he or she can be recognized by the context application as being out of office.

7. Summary

The architecture of the context aware application presented in this paper is characterized by an easy access to the context data sets stored in the database. The context information is stored using the JSON format, which is very popular in the IT world, and is characterized by the lower redundancy in comparison to other standard data description – XML (Extensible Markup Language).

The main challenge was development of the structure of the database which had to meet many criteria described in the system architecture (chapter II). Further work on the described system should be focused on the usage of NoSQL databases as data repository. The presented solution is, first of all: flexible, ready and easy to extend. The most important aspect is the possibility to add another source of context just by inserting a record into the database with a new context data definition. The functionality test results presented in chapter 6 show the location error in mobile networks using Terminal Location API and their impact on the context aware application functionality. This error can be minimized using more accurate location algorithms by communication service providers (e.g. based on WiFi hybrid algorithms [31]) or the implementation of other location methods e.g. based on installed in the phones GPS receiver.

The prototype of the context aware application has been developed under the Orange Labs Open Middleware 2.0 Community program [27] as a part of Grzegorz Siewruk' B.Sc. Thesis.

References

[1] A.C, WEAVER, B. B. MORRISON, *Social Networking, Computer*, Volume: 41, Issue: 2, 2008

[2] C. PETTEY AND H. STEVENS, *Gartner Says Context-Aware Computing Will Provide Significant Competitive Advantage*, Gartner Press Releases, 2010. [Online]. Available: http://www.gartner.com/it/page.jsp?id=1190313 [20.05.2013]

[3] *The Cisco Location-Aware Healthcare Solution*, http://www.cisco.com/web/strategy/docs/healthcare/CLA_HealthcareSolution.pdf. Cisco Systems, Inc. 2007 [20.05.2013]

[4] MAGNUS YANG, *Introducing Google Gadget Ads*, Google Inside AdSense Blog http://adsense.blogspot.com/2007/09/introducing-google-gadget-ads.html [20.05.2013]

[5] STEPHEN A. WEIS, *RFID (Radio Frequency Identification): Principles and Applications*, MIT CSAIL http://www.eecs.harvard.edu/cs199r/readings/rfid-article.pdf [20.05.2013]

[6] MATTHIAS BALDAUF, *Int. J. Ad Hoc and Ubiquitous Computing*, *A survey on context-aware systems*, Vol. 2, No. 4, 2007 pp.263-277,

[7] B.CHIEN, H. TSAI, Y.HSUEH, " *CADBA: A Context-aware Architecture based on Context Database for Mobile Computing*" Autonomic and Trusted Computing, 2009. UIC-ATC '09

[8] A.SRINIVASAN, G. IRWIN, *"Communication the Message: Translating Task Into Queries in a Database Context"*, IEEE transactions on professional communication vol 49, no.2 june 2006

[9] BAUER, M.; KOVACS, E.; SCHÜLKE, A.; Ito, N.; Criminisi, C.; Goix, L.-W.; Valla, M., *"The Context API in the OMA Next Generation Service Interface,"* Intelligence in Next Generation Networks (ICIN), 2010 14th International Conference on, vol., no., pp.1,5, 11-14 Oct. 2010

[10] *Siemens to Reveal New LifeWorks Vision and Showcase Advanced IPBased applications and Services*, SUPERCOMM 2003

[11] D. BOGUSZ, P. KORBEL, J. LEGIERSKI, *Integracja systemów Unified Communications z platformami usługowymi operatorów*, KSTiT 2011 conference materials, Przegląd Telekomunikacyjny 8-9/2011

[12] OpenScape UC Application SOAP SDK Programming Guide

[13] BOGUSZ, D.; LEGIERSKI, J.; PODZIEWSKI, A.; LITWINIUK, K., *"Telco 2.0 for UC — An example of integration telecommunications service provider's SDP with enterprise UC system,"* Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on, vol., no., pp.603,606, 9-12 Sept. 2012

[14] PORTAL STL PARTNERS: http://www.telco2.net [10.12.2013]

[15] L. RICHARDSON, SAM RUBY, David Heinemeier Hansson, *RESTful Web Services*, O'Reilly, 2007

[16] *SOAP specification W3C*, http://www.w3.org/TR/soap/ [10.12.2013]

[17] *Open Service Access (OSA); Parlay X web services*, 3 GPP, http://www.3gpp.org/ftp/Specs/html-info/29199-01.htm, [10.12.2013]

[18] *GSMA OneAPI* http://www.gsma.com/oneapi/ [10.12.2013]

[19] LITWINIUK, K.; CZARNECKI, T.; GRABOWSKI, S.; LEGIERSKI, J., *"BusStop — Telco 2.0 application supporting public transport in agglomerations*," Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on, vol., no., pp.649,653, 9-12 Sept. 2012

[20] PODZIEWSKI, A.; LITWINIUK, K.; LEGIERSKI, J., *"Emergency button — A Telco 2.0 application in the e-health environment,"* Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on, vol., no., pp.663,677, 9-12 Sept. 2012

[21] WAWRZYNIAK, P.; KORBEL, P.; BOROWSKA-TERKA, A., *"Student information delivery platform using telecommunications open middleware APIs,"* Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on, vol., no., pp.871,874, 8-11 Sept. 2013

[22] TRUSIEWICZ, P.; WITAN, M.; KUZIA, M., *"Mobile Payment System - Telco 2.0 application dedicated for payments,"* Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on, vol., no., pp.859,864, 8-11 Sept. 2013

[23] BOGUSZ D., SIEWRUK G., LEGIERSKI J., KUNICKI J.S.,*USSD communication channel as alternative to XML SOAP in mobile Unified Communication applications*, Federated Conference on Computer Science and Information Systems. Place: Krakow. September 8-11, 2013

[24] GSMA *OneAPI Business rationale for cross-operator network APIs*, GSMA February 2013

[25] *M2M: The Internet of 50 Billion Devices*", WinWin Magazine, January 2010

[26] CROCKFORD, DOUGLAS (May 28, 2009). *"Introducing JSON"*. json.org. Retrieved July 3, 2009.

[27] *Open Middleware 2.0 Community portal* – http://www.openmiddleware.pl [20.05.2013]

[28] VINOSKI, S *"Scripting JAX-WS [JavaScript],"* Internet Computing, IEEE, vol. 10, issue. 3, pp.91-94, May-June 2006

[29] NAKAMURA, M.; GRAD. SCH OF ENG., KOBE UNIV., KOBE; FUKOUOKA, Y.; IGAKI, H.; MATSUMOTO, M-I.; *"Implementing Multi-Vendor Home Network System with Vendor-Neutral Services and Dynamic Service Bining"* Services Computing, vol. 2, issue., pp.275-282, 7-11 July 2008

[30] FUNDACJA BEZPIECZNA CYBERPRZESTRZEŃ *"Największe zagrożenia dla bezpieczeństwa w Internecie w roku 2013"* http://cybsecurity.org/pdfy/FBC_Predictions_RAPORT_2013.pdf

[31] KORBEL, P.; WAWRZYNIAK, P.; GRABOWSKI, S.; KRASINSKA, D., *"LocFusion API - Programming interface for accurate multi-source mobile terminal positioning,"* Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on, vol., no., pp.819,823, 8-11 Sept. 2013

[32] SIEWRUK, G.; ŚREDNIAWA, M.; GRABOWSKI, S.; LEGIERSKI, J., "*Integration of context information from different sources: Unified Communication, Telco 2.0 and M2M*," Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on, vol., no., pp.851,858, 8-11 Sept. 2013

Aplikacja wykorzystująca kontekst komunikacji – koncepcja i implementacja

STRESZCZENIE: W publikacji przedstawiono ideę aplikacji, która gromadzi i przetwarza dane kontekstowe pochodzące z wielu różnych źródeł. Informacje związane z kontekstem komunikacji przechowywane są w specjalnej bazie danych oraz udostępniane bezpośrednio użytkownikom końcowym w postaci funkcji komunikacyjnych. Aplikacja, wykorzystując otwarte API dokonuje integracji różnego rodzaju informacji kontekstowej udostępnianej przez systemy: Unified Communication, platformy usługowe operatora telefonii komórkowej oraz system M2M. W publikacji zaproponowano sposób przechowywana różnych typów danych kontekstowych w relacyjnej bazie danych (postaci dedykowanej struktury tablic), przeprowadzono testy wydajnościowe oraz porównano ich wyniki z wynikami dla tradycyjnych struktur danych. Przykładem praktycznym ilustrujący główne założenia jest aplikacja kontekstowa – zdefiniowana jako system dla końcowego użytkownika firmowego systemu telekomunikacyjnego. Publikacja jest rozszerzoną, wersją artykułu "Integration of context information from different sources: Unified Communication, Telco 2.0 and M2M", zaprezentowanego na konferencji Computer Science and Information Systems (FedCSIS 2013).

SŁOWA KLUCZOWE: informacja kontekstowa, kontekst komunikacji, API, Unified Communications, M2M,.

Praca wpłynęła do redakcji: 17.01.2014 r.