Data Collection Architecture for Field Research in Heterogeneous Computational Environments

Henrique P. de F. Filho, Maristela Holanda, Bernardo Macêdo, Renata Nunes, Paulo Brener
Department of Computer Science
University of Brasilia
Brasilia, Brazil
henripff@gmail.com, mholanda@cic.unb.br, bernardo-macedo@hotmail.com, cmn.renata@gmail.com, paulo_brener_12@hotmail.com

Henrique Llacer Roig
Institute of Geoscience
University of Brasilia
Brasilia, Brazil
roig@unb.br

Abstract—**The computational environment based on wireless communication made it possible to access information anywhere and at anytime, which is favorable for data collection in field research. Currently, most architecture for data collection are designed for a specific purpose, using specific technologies that are limited regarding data, network, synchronization and device type. This paper presents an architecture for field research data collection that works in heterogeneous computational environments and supports vector geographic data. This architecture was validated with a case study conducted at the Institute of Geosciences (IG) of the University of Brasilia (UNB).**

Keywords-Data collection; field research; architecture; heterogeneous computational environments; vector geographic data

I. INTRODUCTION

The technological advances of the last decades associated with wireless technology communication are intended to allow access to information anywhere and at anytime, making a favorable environment for the development of computational systems, which are aiming at collecting data in field research [16].

One of the challenges in this environment is to assure the consistent state of the database involved in the system. There are databases in mobile devices and the central database [21].

Nowadays, most of the data collecting architectures work in homogeneous computational environments [16][18][21]. In other words, they use specific technologies concerning, for instance, data, network, synchronization, and device types, and they fulfill a specific data gathering need.

In this context, this paper proposes an architecture for collecting data in a heterogeneous computational environment where information stored on each mobile device is replicated and correctly integrated into a central database. The proposed architecture not only collects conventional data, but also vector geographic data, because the demand for this type of data is high in geology and geoscience research, and spatial remote sensing.

This paper is organized as follows: Section II presents other work related to this research study; Section III deals with the main aspects of a system for mobile data collection; Section IV presents the proposed data collection architecture; Section V presents the case study; Section VI presents the conclusions.

II. RELATED WORK

A software suite that can be used for data collection in field research inserted in a mobile computational environment is available on the market. These software have architecture for data collection that caters to diverse contexts of field research, however, they are tied to specific technologies, which is a limitation of their use. Nokia Data Gathering solution and AuditMagic are examples of these software applications [3][16][18].

Nokia Data Gathering is a system that allows the collection of data on mobile devices and the transmission of results to real-time analysis, in accordance with the access to a network data communication. The system allows the creation and delivery of questionnaires for mobile phones and integration into a database using a pre-existing cellular network common [16][18]. With regards to this solution, a problem was identified: it is a closed architecture tool that only works for Nokia cell phone devices.

AuditMatic has a set of solutions for developing instruments to collect field data associated analysis tools. However, this tool is also a package deal where you cannot make changes according to the needs of each research [3][16].

Most architectures for data collection in field research proposed by researchers meet only a single context of data collection, namely, the context in which the research is embedded. The goal of this paper is to propose an architecture that meets the needs of data collection of this research. An example is the architecture for collection and dissemination of weather data in the state of Piauí proposed by researchers from EMBRAPA (Brazilian Agricultural Research Corporation) and the Faculty of Technological Education of Teresina [21].

Another example is the system architecture for data collection in a mobile computing environment where Internet access is intermittent, which was validated in the research of Rural School Transportation, developed by the Centro

ISBN: 978-1-61208-326-1

Interdisciplinar de Estudos em Transportes (CEFTRU/UNB) in partnership with the Fundo Nacional de Desenvolvimento da Educação (FNDE), whose objective is to evaluate the reality of school transport in rural Brazil. [16] This architecture has mechanisms for fault tolerance; however, it does not support geographic data and uses a specific technology that meets only the context of the collection in question.

As previously stated, these architectures serve only these context specific collections using specifics technologies, specifics synchronization protocols that best fit the types of data to be synchronized, whether architectures can be used in other context collections or not.

The architectures proposed by researchers, which have the same objective as the architecture proposed in this paper, do not support the collection of geographic data.

Ji [23] presents an architecture that aims to collect georeferenced data. The communication between client and server is done via web Services using the technology of wireless networks.

III. MOBILE DATA COLLECTING SYSTEM

The Mobile Data Collection System (MDCS) allows the collection of remote geographic locations and the transmission of data to central locations - data storing repositories through a wireless network. It is a combination of a client application running on mobile devices, wireless network infrastructure and remotely accessible database servers [11].

Most of these systems share the same principles and guidelines for remote data collection. As shown in Figure 1, the process begins by developing the application with a form, which contains a set of questions to collect relevant data. Data collectors use these forms on the mobile device to collect real data in the field. It is possible to upload the data to a database in the central server [11].

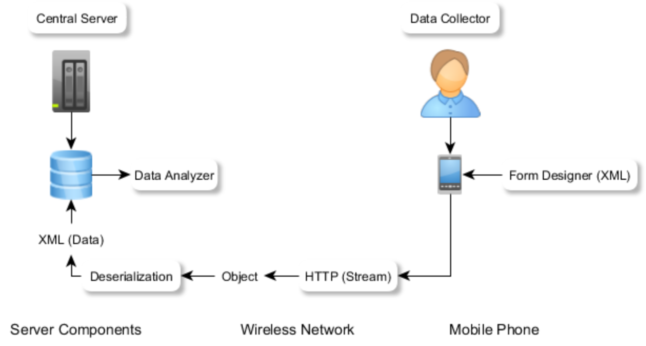


Figure 1. Flow diagram of a MDCS. Adapted from [11].

One challenge in MDCS is the data synchronization. In mobile environments, synchronization may be defined as the act of establishing equivalence between collections of clients and server databases [4][6]. For this synchronization to occur data replication is necessary.

Replication is the process in which transactions executed in a database are propagated synchronously or asynchronously to one or more databases in a serial manner, i.e., this means that all transactions are replicated in the same order in which they were requested [20]. If this does not happen, there may be inconsistencies between the mobile unit and the server in a mobile environment. Spreading asynchronously refers to a way of storing and sending replication, i.e., the operations to be replicated from mobile units can be stored in a local database until they are propagated to the server at the time of synchronization, since sometimes the connection is impossible [20].

The replication may be partial or total. In partial replication only part of the data from a database is replicated to other databases, while in full replication all data is replicated from one database to another [8]. With full replication, at least one copy is available, and the reliability is increased, since the user does not depend solely on the data available at a single location. In cases where a fault occurs in the system, data replication is requested by the user [14].

A synchronization protocol defines the workflow for communication over a section of data synchronization when the mobile device is connected to the fixed network. It should support the identification of records, protocol commands common to the local database and network synchronization, and be able to support the identification and resolution of synchronization conflicts [17].

IV. THE PROPOSED ARCHITECTURE

The goal of an architecture for data collection in field research that works on heterogeneous computational environments is to disassociate the specificities of each collection with the used technologies, providing an architecture that can be used in various contexts for data collection. Subsequently, in this study, for a data collection architecture to work in several different computational environments this architecture must take two aspects into account: interoperability and flexibility.

Interoperability is required so that architecture is not dependent on specific technology, but covering the possibility of using several technologies. Flexibility is required in two directions: the first to meet the diverse dimensions that data collection in the field can have, i.e., fulfill the data collection involving one or more institutions. A specific data collection can store the collected data in a single database located in a specific place or multiple databases located in several different places. The second is to make the necessary changes needed to pass the architecture to suit the contexts of data collection different, i.e., that the architecture is susceptible to changes - it can be configurable and reconfigurable.

An architecture for data collection in field research that supports vector geographic data must use a *Data Base Management System* (DBMS) with support for spatial data in both collection devices as the servers involved in collecting.

Abstractly, the proposed architecture consists of three stages: the collection field, synchronization and storage, as shown in Figure 2. Each of these stages is described as follows.

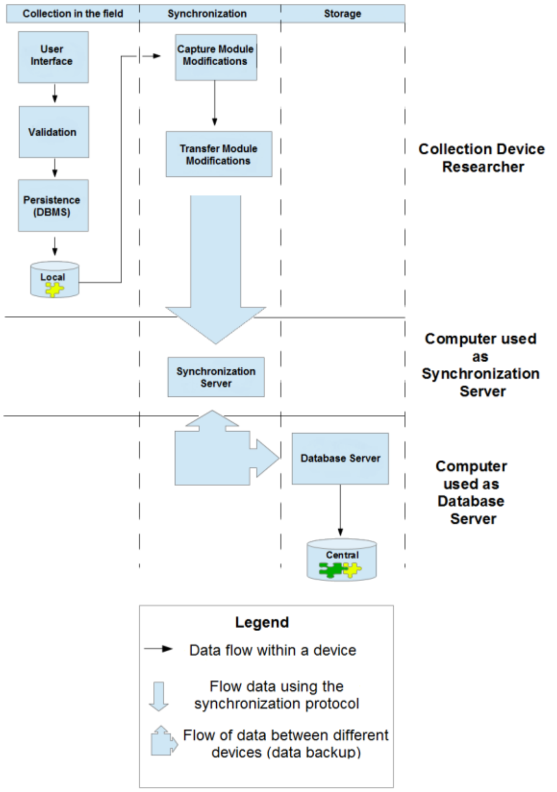

Collection in the field
Synchronization
Storage
User Interface
Validation
Persistence (DBMS)
Local
Capture Module Modifications
Transfer Module Modifications
Collection Device Researcher
Computer used as Synchronization Server
Synchronization Server
Database Server
Computer used as Database Server
Central
Legend
Data flow within a device
Flow data using the synchronization protocol
Flow of data between different devices (data backup)

Figure 2. Abstract architecture of the heterogeneous data collection system.

Data collection in the field occurs through a computer system in the researchers' mobile devices. This computer system must have a user interface, a method of data validation and a persistence layer. The user interface is responsible for entering data into the system, the validation method is responsible for validating the data entered so that relevant data are not forgotten or recorded incorrectly and the persistence layer is responsible for storing the data in the database mobile devices. The persistence layer must necessarily store the data in a DBMS that supports spatial data.

After collecting the data, there is a synchronization step, where the collected data is synchronized with the synchronization server. The objective of synchronization is to generate replicas, through a full replication of the collected information, to the server that synchronizes data with the database server involved in the core collection. This synchronization is specified in the next section. Finally, data servers are stored in the core data set involved in the collection via data backup.

The possibility of having more than one server central database is taken into consideration in this architecture, given the possible dimensions of data collection, contributing to the architecture's characteristic flexibility. When we have more than one server database replication, server data synchronization occurs for these servers, creating data redundancy, which prevents the architecture from having a single point of failure.

The synchronization protocol is used in architecture that will lay down the interoperability and flexibility of the architecture (flexibility towards the possibility of change, to adapt to contexts of different data collection), so the protocol for the architecture should be characterized as interoperability, and should be a free protocol, i.e., not proprietary protocol.

A. *Synchronization Stage*

The proposed synchronization is divided into two stages: capturing *Structured Query Language* (SQL) [8] statements and the transfer of these instructions to the synchronization server. The capture module modifications are responsible for capturing the SQL database from the mobile device in the event of any change in the database. The transfer module modification plays a client role in a client-server architecture, synchronizing the changes to the SQL statement capture module with the synchronization server.

1) Capture Module Modifications

The capture of these new SQL statements is by searching for data that have a flag with value 1. This flag determines if the data has been synchronized or not. If it is set as 1, then the data has not been synchronized yet; if it is set as 0, then the data has been synchronized. This search can occur in all tables of the database or only one table in the database that concentrates all the changes that have occurred; see Figure 3. The flag only changes to 0 when all modifications are finalized, i. e., when i <=n, this way if the connection is lost during this process the flag does not change, and all modification are done when the mobile unit recovers the connection.

2) Transfer Modifications Module

After the capture of new instructions, the modifications transfer module makes the transfer of these instructions to the server synchronization [13]. This transfer occurs according to the synchronization protocol chosen for the architecture, which, as previously mentioned, and should be a protocol that has the characteristic of interoperability and is not proprietary; see Figure 3.

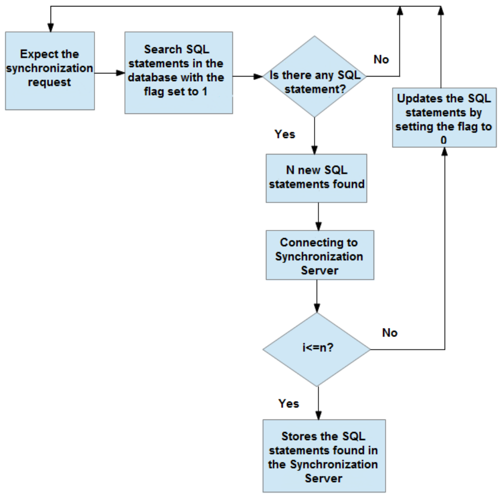

Expect the synchronization request
Search SQL statements in the database with the flag set to 1
Is there any SQL statement?
No
Yes
N new SQL statements found
Connecting to Synchronization Server
i<=n?
No
Yes
Stores the SQL statements found in the Synchronization Server
Updates the SQL statements by setting the flag to 0

Figure 3. Flow diagram of the SQL statements.Adapted from [13].

The flow diagram of these modules shows that when a connection error occurs while transferring changes, the variable n (the variable in the conditional structure of the flow diagram) will have the value of the SQL statements that were not transferred. Thus, when the internet connection is reinstated the transfer continues from where it stopped without causing data inconsistency.

V. CASE STUDY

A. Field Data Collection Stage

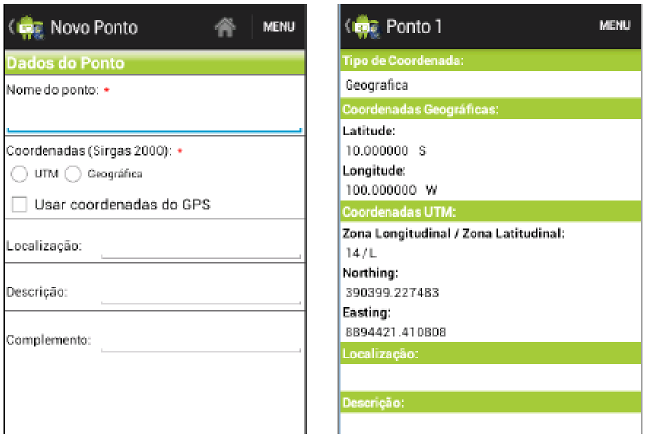
In the first stage of the architecture, for data collection in the field, a mobile application called RockDroid was developed using a methodology of data collection based on remote sensing spatial research undertaken by the Institute of Geosciences of the University of Brasilia (UNB). The purpose of the application was to facilitate the collection, storage and data synchronization using, Smartphones and Tablets, which are Mobile computational resources easily accessible to members of the collection team.

The application was developed for the Android platform, version 2.3 or later, and focused on using only libraries' open source components for easy maintenance and availability. The decision was made to use the Android platform, primarily because it is an open source operating system, which is consistent with the aim of the work, and also because the operating system is present on most Smartphones and Tablets, representing a share of 59.5% of the market in the first quarter of 2013 [5]. It also assists the developer in supporting devices of different sizes, shapes and specifications. Using some good programming practices you can develop an application that can be run on most Android devices, leaving the care system operative, resizing the visual components [7].

The RockDroid is responsible for storing information collected about geographical points, the rocks contained therein and a brief description of the specimens and structures of each rock. It provides forms for researchers to fill out with the data collected, and has screens to display information retrieved from the database, as well as displaying options to edit and delete records. The software also provides some mechanisms for validating the data entered to ensure the integrity and consistency of the database. Part of the persistence is implemented through a relational database created from SQLite [15], a small management system relational database, commonly used in embedded systems and does not require a large processing capacity [1].

Although the proposed architecture supports the collection of vector geographic data, it was used in applying the DBMS SQLite instead of SpatiaLite due to the requirements of the application, because the only vector geographic data types that the RockDroid collects are points by latitude and longitude, and so it is not necessary to use the SpatiaLite. However, the application also supports the DBMS SpatiaLite.

The user interface is responsible for displaying forms so that the user can insert or update information and display screens with data retrieved from the database. Another way to view the data was retrieved through a map on which the user can view the data geographically distributed. Another feature of the map is to display the current location of the device functionality that could be implemented through the mechanisms provided by the *Global Positioning System* (GPS) location, the cell phone networks and the Internet. Figures 4 and 5 show some images of the user interface application developed; the data are not real, but were inserted for demonstration only.



Novo Ponto

Dados do Ponto

Nome do ponto: *

Coordenadas (Sirgas 2000): *

○ UTM ○ Geográfica

□ Usar coordenadas do GPS

Localização:

Descrição:

Complemento:

Ponto 1

Tipo de Coordenada:

Geografica

Coordenadas Geográficas:

Latitude:

10.000000 S

Longitude:

100.000000 W

Coordenadas UTM:

Zona Longitudinal / Zona Latitudinal:

14 / L

Northing:

390399.227483

Easting:

8894421.410808

Localização:

Descrição:

Figure 4. RockDroid application screens.

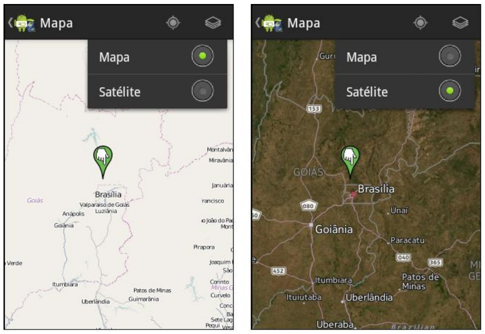


Figure 5. Data displayed on a map on the RockDroid application.

Data validation, before being entered into the database, is essential as it ensures the consistency and integrity of the information stored.

The script to create the database was embedded into the application. Thus, when installed on a mobile device, the application will create a local database. If the application is uninstalled, the database will be deleted as well. It is important to note that the database will only be created if it does not exist at the time the application starts. If there is already a database for RockDroid, it will be kept intact.

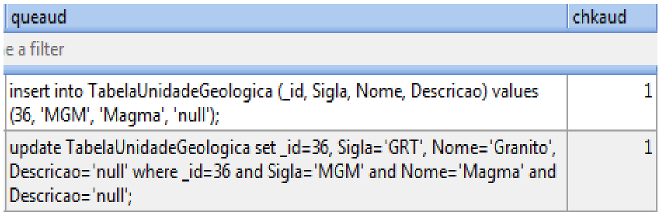
B. Synchronization Stage

In the synchronization stage of the architecture, the capture module modifications were developed based on the creation of triggers that are fired when there is an insert operation, update, or delete the database. These triggers aim to store all the changes in the database in one table, facilitating the search for such modifications. All changes are stored in the audit table that has the following columns: date and time of the operation (timestamp of the operation,

ISBN: 978-1-61208-326-1

dthaud), the database where the action occurred (banaud), the table where the action occurred (tabaud), the type of operation that occurred (tacaud), the query executed during the action (queaud), and a flag that indicates whether the record has been synchronized or not (chkaud), as explained in the previous section. Figure 6 shows a section of the table with some audit records stored. Initially this flag is set to 1 for all changes because none of them had been synchronized with the synchronization server yet.

The SyncML protocol [12] was chosen for use in the proposed architecture because this protocol meets the prerequisites that the proposed architecture requires, i.e., promotes interoperability of data synchronization and is a non proprietary protocol. It is a standard protocol for data synchronization, regardless of platform, device and network. It is based on Extended Markup Language (XML) technology and maintained by Open Mobile Alliance (OMA), an alliance of several companies to create a common protocol for data synchronization [9][10].



queaud	chkaud
e a filter	
insert into TabelaUnidadeGeologica (_id, Sigla, Nome, Descricao) values (36, 'MGM', 'Magma', 'null');	1
update TabelaUnidadeGeologica set _id=36, Sigla='GRT', Nome='Granito', Descricao='null' where _id=36 and Sigla='MGM' and Nome='Magma' and Descricao='null';	1

Figure 6. A piece of the table with two audit records stored.

In the proposed synchronization, we used the so-called one-way synchronization from the client to the server, where the changes in the database of mobile researchers (SyncML clients) are synchronized with the server synchronization (SyncML) [9][10]. This type of synchronization was chosen because in a data collection it is necessary to have all the data collected only in the server, due to collection devices having limited computational resources.

The transfer module of modifications has been developed in Java using the library server Sync4j Funambol Data Sync Server (Funambol). The Funambol is a synchronization server that implements the SyncML protocol and was used to implement the synchronization server architecture [2]. This module is nothing more than a SyncML client to query the audit table, capturing the changes in the database, and transfers the changes occurring to the synchronization server. After synchronization, the flags of the changes are synchronized to 0 unset.

C. *Storage Stage*

The last stage, i.e., the storage stage, was developed through replication (backup) server data synchronization (Funambol) to the server database, which was developed based on the PostgreSQL DBMS with spatial extension PostGIS.

D. *Results*

Three tests were performed on a heterogeneous computing environment. The three tests performed were:

- Check if synchronization occurs properly and in a timely fashion;
- Check if, when a disconnection of the internet in the middle of a sync occurs, the sync continues where it left off or resynchronizes after the reestablishment of the connection ;
- Check if the synchronization of multiple mobile devices simultaneously occurs correctly (we used five mobile devices for testing).

Figure 7 shows the computing environment in which the proposed architecture was tested.

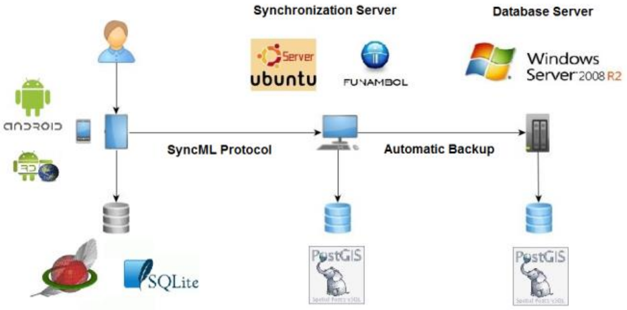


Figure 7. The heterogeneous computing environment in which the proposed architecture was tested.

Phase of data collection in the field:

- Device used for collecting data: tablet and smartphone;
- Operating System: Android;
- Computer system: geographic information system (RockDroid);
- Supported data types: conventional and geographic data;
- System manager database: SQlite and SpatiaLite.

Synchronization Phase:

- Type of computer used as synchronization server: desktop;
- Operating System: Linux Ubuntu Server 12;
- System manager database: PostGIS;
- Supported data types: conventional and geographic data;
- Protocol synchronization: SyncML (Funambol).

Storage phase:

- Type of computer used as server database: Minicomputer;
- Operating System: Windows Server 8 R2;
- System manager database: PostGIS;
- Types of data stored in the server database: conventional and geographic data.

The synchronization was tested in different types of networks (wireless, wired) and in different states of the network (Internet with volume data traffic 600 megabits per second, 100 megabits per second and 10 megabits per second), and in every case both the collection and the synchronization were successful and presented acceptable times, respectively 1 to 3, 4 to 8 and 8 to 10 seconds; the time synchronization varied according to the state of the network.

Tests were conducted off the Internet in the middle of a synchronization and after the reestablishment of the connection, synchronization restarted again. Both the capture module modifications and the transfer module modifications followed the diagram shown in the previous section, so there was fault tolerance.

The last test was to synchronize multiple mobile devices simultaneously. This test was intended to simulate the reality of the collection, because normally all researchers, after collection, synchronize their data simultaneously to the server. Five mobile devices were synchronized at the same time and no error occurred when synchronizing.

VI. CONCLUSION AND FUTURE WORK

The existing data collection architectures satisfy a specific cause, using specific technologies that are limited regarding the type of data, type of networks, synchronization type, device type, and so on. For each different data collection context, an architecture is proposed for data collection with different budget limitations, computational environments and available technology within the company or organization.

The proposed architecture for this work was an architecture of data collection for field research in heterogeneous computational environments that supports vector geographic data where information stored on each mobile device are replicated and correctly integrated into a central database. To reach this goal, the architecture had to obey two aspects: interoperability and flexibility. This architecture complies with the aspect of interoperability because it uses different mobile devices, operational systems and wireless network properties. It also complies with the aspect of flexibility, by replicating data amount where there is an intermediate server and a free synchronization protocol, which supports replication to many databases. In addition, the architecture uses some safety mechanisms such as fault tolerance, full replication data and authentication.

With this architecture, it is no longer necessary to plan an architecture whenever you make a data collection in the field. Researchers may simply use the proposed architecture instead, which in addition to functioning in heterogeneous environments and support vector geographic data, is low cost and high quality.

Further testing of these applications is will be carried out in the field. These tests will be conducted with the support of the Institute of Geosciences, University of Brasília and concerns the collection of geological data in several cities of Brazil.

REFERENCES

[1] About SQLite, http://www.sqlite.org/about.html. [retrieved: Feb., 2014]

[2] A. L. B. Alonso, C. Oliveira, L. Fedalto, F. Vilas Boas, T. L. G. Assis, and C. S. Hara, "A synchronization experience of relational databases using SyncML.", In: . Proc.: Escola Regional de Banco de Dados (ERBD), Brazil, Apr. 2010, pp. 1-4.

[3] AuditMatic, http://www.auditmatic.com. [retrieved: Feb., 2014]

[4] B. R. Badrinath, and S. H. Phatak, "Bounded locking for optimistic concurrency control," Department of Computer Science, University Rutgers, New Jersey, EUA, 1995.

[5] Canalys, http://www.canalys.com/newsroom/smart-mobile-device-shipments-exceed-300-million-q1-2013. [retrieved: Feb., 2014].

[6] D. L. Costa and F. Franquini, "Synchronization protocols in wireless environment," Department of Computer Science of Federal University of Santa Catarina, Master These, Brazil, 2004

[7] A. Developers, http://developer.android.com /guide/practices/screens_support.html.[retrieved: feb., 2014]

[8] R. Elmasri, and S. B. Navathe "Fundamental of Database System," Addison Wesley, USA, 2011.

[9] Ericsson, IBM, Lotus, Matsushita, Communications Industrial Co. Ltd., Motorola, Nokia, Openwave, Palm, Psion, Starfish Software, Symbian, and others "Building an industry-wide mobile data synchronization protocol," SyncML White Paper, 2000.

[10] Ericsson, IBM, Lotus, Matsushita, Communications Industrial Co. Ltd., Motorola, Nokia, Openwave, Palm, Psion, Starfish Software, Symbian, and others "SyncML sync protocol," SyncML White Paper, 2002.

[11] S. Gejibo, F. Mancini, K. A. Mughal, R. A. B. Valvik, and J. Klungsøyr "Secure data storage for mobile data collection systems,". Proc: IEEE HealthCom conference, 2012, pp. 498-501.

[12] U. Hansmann, R. Mettala, A. Purakayastha, P. Thompson, and P. Kahn "SyncML: synchronizing and managing your mobile data," Prentice Hall, 2002.

[13] M. I. Hossain, and M. M. Ali "SQL query based data synchronization in heterogeneous database environment," International Conference on Computer Communication and Informatics, Coimbatore, India, Jan. 2012, pp. 1-5.DOI: 10.1109/ICCCI.2012.6158818

[14] G. C. Ito, M. Ferreira and N. Sant'Ana, "Mobile Computing: characteristics about data management ", Instituto Nacional de Pesquisas espaciais – INPE, 2003.

[15] R. Lecheta "Aprenda a criar aplicações para dispositivos móveis com o Android SDK," NOVATEC, São Paulo, SP, Brasil, 2009.

[16] J. Magalhães, M. Holanda, and R. Chaim "Architecture for data collection in mobile computing with intermittent internet access," 6ª Conferência Ibérica de Sistemas e Tecnologias de Informação (CISTI 11), Chaves, Portugal, AISTI Press,2011, pp. 1-6.

[17] E. C. Manganelli, and J. Romani "Data synchronization protocols in wireless environments: a case study," Department of Computer Science, Master These, Federal University of Santa Catarina, SC, Brazil, 2004.

[18] Nokia Data Gathering, http://www.nokia.com/corporateresponsibility/society/nokia-data gathering/english. [retrieved: Feb., 2014]

[19] M. Rennhackkamp "Mobile Databases Tetherless Computational Liberates End Users But Complicates the Enterprise," DBMS online, 1997.

[20] A. J. S. Silva, A. S. de Andrade Júnior, and F. R. Marin "Data Collection and dissemination Architecture of climate data in the state of Piauí," Revista de Tecnologia de Fortaleza, vol. 29, Brazil, 2008.

[21] M. Ji, Y. Sun, F. Jin, T. Jiang, J. Wang, and X. Yao "Research and Development of Field Data Collecting Synchronously System of Mining Area," IEEE Internacional Geoscience and Remote Sensing Symposium, Hawaii, USA, IEEE Press, 2010. pp. 3948 – 3951. DOI: 10.1109/IGARSS.2010.5650825.