A Novel Synchronization between SQL Server and Mobile Client Databases

Mohanad Faeq Ali, Norharyati Harum and Nur Azman Abu

INTRODUCTION

The advancement in the area of computing and mobile technology have led to the occurrence of a new computing atmosphere and different categories on small mobile devices such as smart phones, Personal Digital Assistants (PDA), Handheld PCs (HPC) and Pocket PCs. Mobile devices have rapidly evolved from simple devices that merely make and receive calls and sends messages to larger and more sophisticated devices that can also be used as tools to manage and store personal data. As various network technologies and enterprise applications are progressively being connected with such devices, the data management, processing as well as manipulation of enterprise business information can be handy and obtainable using these mobile devices. As a result, more business models that are solely dependent on mobile technologies begin to emerge.

Synchronization is an activity of transferring data between two entities. According to [1], it is either transferring data from server to client or vice versa. Synchronization consist of two operations which are upload and download. An upload operation is executed first and followed by download operation to get equivalence data for both interactive databases[2]. Synchronization methods depend on types of database management system (DBMS) supports such as trigger, timestamp, meta-stored procedure which bring varieties of synchronization result performance. Relational Database management system (RDBMS) also known as SQL database are the developers’ choice in late 1990’s as the emergence of webs application development to handle simple structured data like text and numeric value. However in this 21st century, clients tend to operate the system via mobile devices. At the same time, an emergence of media has introduced new database management system: NoSQL, Schema less database that able to handle unstructured data better which can handle large size of media data using high processing power to perform synchronization.

According to [3], the emergence of advance applications that extend its functionality and requirements make the SQL database unable to fully fulfil the requirements in distributed environment. Besides, implementation of SQL database type alone will not be able to perform well in a distributed system nor provide availability of data [4]. A simple SQL database does not scale out well in a distributed system. Hence, SQL database need to operate together with NoSQL database, which according [5], NoSQL database provide simpler structure and faster database management in distributed environment and bring high synchronization performance to server and mobile device client as well. This mismatch calls for database synchronization between mobile database and server-side database. Based on these, varieties of data synchronization solutions are provided to solve this problem.

According to [6], more than 80% of interactive business data nowadays are in unstructured data form. Unstructured data consists of various formats in the form of image, video, document etc. These unstructured data provide burdensome to synchronize compared to structure data in mobile environment that have limitations in terms of processing power, network

Keywords: SQL, NoSQL, Mobile Database, Synchronization
bandwidth and battery capacity [7]. Hence, a new image synchronization approach needs to be efficient in server and mobile client environment by reducing the burden of computing capacity in mobile client side. As NoSQL database characteristics are able to lower the processing time [3] which make it simpler, faster [5] and schema-less, we strongly believe that by implementing NoSQL database in client side could help in reducing clients’ mobile devices computing resources and synchronization process time.

However, many synchronization approaches either in server and client side are dependent only on proprietary implementation of DBMS which is vendor specific. In other words, only mobile and server side database that have identical type of database structure, framework and programming language are able to synchronize without restrictions and conflicts [1]. Different type of vendors, devices operating system and different database management, make synchronization process are problematic as having different framework, data model and synchronization method as well. In addition, every new release of DBMS, operating system or even new version of existing DBMS, there is unclear gateways such as framework and data model that are provided by anyone to support synchronization process among heterogeneous databases resulting from difficulties in reconciliation among them [8].

Sometimes, even programmers need to modify existing applications for synchronization process for reconciliation of dissimilar databases [9]. Reconciliation of data synchronization method of these dissimilar databases could be performed by manual query programming between SQL and NoSQL language [10]. Thus, the idea of an establishment of connectivity among different databases connection is possible.

**NOSQL DATABASE**

Web 2.0 applications store their attributes in numerous tables in terms of text, comment, image, video and source code. In order to support easy schema evaluation, underlying databases have to be flexible. Since web applications are very agile and NoSQL offers that flexibility [11]. SQL database is not able to add and remove features when it comes to system unavailability. In addition, NoSQL manages to handle massive data set by indexing arbitrarily and at the same time by enabling a large amount of concurrent user request [12]. According to [13], in order to support mobility, low bandwidth variability and heterogeneous networks and security, the new database processing schemes need to be introduced. Traditional database schemes are not suitable to cater for an effective synchronization process.

As stated by [14], NoSQL databases are able to extend its performance and scalability compared to SQL by abandoning atomicity, consistent, durable, isolated (ACID) properties. This statement also supported by [15] in which he properly explained that NoSQL has been introduced to achieve more availability and scalability for which SQL is unable to achieve as well as NoSQL, especially in handling large data. [16] pointed out that SQL databases are unable to provide enough computational and storage allocation when handling massive data. NoSQL is becoming popular approach in handling the data as its enhance scalability and practicing non-relation data stores and also some of the database like Cassandra and Voldemort had proven that NoSQL able to produce cost effectiveness approach when handling the data in databases by using distributed key-value stores [17].

In conclusion, this paper focuses on implementing NoSQL database on mobile client sides in order to reduce the computing and memory resources of mobile device by adopting NoSQL characteristics in a simpler schema-less deployment. The synchronization process to SQL server side will be followed by a newly proposed sync gateway. Thus, a connection can be established between a dissimilar databases structure for both database to synchronize. This proposed framework eliminates previous synchronization methods between SQL server and SQL client like timestamp, triggered and meta-stored procedure by introducing general technique for heterogeneous databases to synchronize successfully.

**DATABASE CONNECTION APPROACH**

Suitable development toolkits used to create mobile applications and database connection are needed here in order to create communication links between server and mobile client. For example, [18] approach to use Android Cloud to Device Messaging (C2DM) is already deprecated on October 2015 hence new enhancement or migration of project needs to use another toolkits like Google Cloud Messaging (GCM). Empirical study has been done to study the suitable tools used to implement mobile application and database connection.

Most of the researchers are using Java language to develop mobile applications and connecting to the database. [7], [19-20], use android platform and most of them use JDBC to connect to the database. According to [18], HTTP is a popular approach in connecting the database as it is common practice and popularly used by developer as it follows popular Representational State Transfer (REST) API architecture. REST API architecture can easily create a simple and extensible web-service. HTTP has been implemented by [19], [21].

From the study above, it is shown that SQL databases are being used on both server and client databases. MySQL, Microsoft SQL server and SQLite have been selected as the databases to store and transfer the data. Unlike [18], they implement two databases, SQLLite as SQL database to interact with android client, while Big Table as NoSQL database to serve Web clients by using Google App Engine’s data store as mediator. [22] uses 3 types of data-bases in distributed environment consisting of MySQL database on server side and SQLite and XML database in mobile client side. XML database is used by [22] to create a generic data model for image synchronization between heterogeneous databases.
Besides, the authors in [22] use JSON language as the intermediary compound to execute transaction process of structured and semi-structured data between heterogeneous databases. Above all, recent researches such as [18] and [22] show a clear pattern towards reconciliation of different type of databases to adapt the heterogeneity environment coming from variety of vendors, DBMS, and operating system nowadays.

**METHODOLOGY**

This framework will execute synchronously, which allocate one specific time to this synchronization thread on the mobile processor. The mobile processor will delay all other processes until the synchronization process is completed. Server side consists of RDBMS which is MySQL DB that carries SQL data model behaviour, which is table form database type whereas mobile client side consists of NoSQL Couch base Lite DB which stores data in schema-less JSON Native Document database. There will be typical components to represent the two types of database characteristics as shown in figure 1 below.

From figure 1 above, synchronization process of images needs wireless internet connectivity. Volley function in android library is used in sync gateway to create connection between these two data-bases. Base64 is used to encode binary files such as images within scripts, to avoid depending on external files. Hence, the images are able to be saved in databases in encoded forms thus make it lighter and faster to perform image transaction. At the same time, reducing the time for apps and server have to load will naturally make those apps faster. Base-64 primary purpose is to make it efficient to store large binary data in a document. Binary data stored in JSON has to be encoded into base64 as a string, which inflates its size by 33% [24]. Sync Gateway for dissimilar databases and image synchronization framework from NoSQL client DB to SQL server DB framework will be discussed in next section.

**Sync Gateway**

A sync gateway for dissimilar databases type will focus on the compatibility of communication protocol between SQL and NoSQL database. This sync gateway consists of several class functions. The class function will manage image data structure, insertion and deletion of images and also information for databases connectivity and error handling. Figure 2 below shows components in proposed sync gateway between MySQL – Couch base lite DB.

From figure 2, Volley function is used to pass its request objects in parameter. The parameter consists of 3 elements which are POST method, server URL, and Listener to do error checking while synchronizing the image to server. A request to sync client DB with server DB will be put in Requested queue. Request queue will handle multiple requests, where when one request is added; it is picked up by the cache thread and tree will be placed on network queue.

This volley function manages the dispatching parsed response back to the main thread. Thus, the first request in request queue will perform POST transaction, parses the response on the worker thread, write the respond to cache and post the parsed response back to the main thread for delivery to server. This volley requires internet permission in client android manifest. Images information which consists of ID, name, size, and base-64 encoded image content will put in POST parameter upon delivery data to server through the pipeline.

Mobile Sync function components consist of two major functions that carry insertion and deletion of images activities in client’s side. An image is either capture using phone camera or from image gallery. An image is then given its own image ID number and encoded to string using base64-encode To String function before saving into couchbase lite DB. Get Document function is used to retrieve Couchbase’s JSON document that stores encoded image. HashMap is used to map keys to values which implements hash function to compute an image index into an array of buckets or slots. This method means an image ID is an index key that will be mapped into their base64 encoded image string values.

An image in client database is deleted by following several procedures. Firstly, get document (ID) function will retrieve
image ID that is selected by client to delete it from client database. Then, function delete from server will be executed that passes the image name in POST parameter located in Volley function. An image name will be parsed into PHP file in the host server that will delete the image in the server that have the same image name in POST parameter.

An Error Handler component has major responsibility in tracing error in android java classes. The error in Volley functions while establishing connection to server will be traced in the log cat from an android studio. Besides, any errors while inserting new images and deleting existing images in client DB will be notified to client through log cat and android 'toast' in mobile device’s screen. This error will track synchronization activities either the functions executed in normal manner or having problem.

Synchronization from Client to Server

This image synchronization framework goes in a one way direction, from client to server. An android app is developed to retrieve images from camera or SD card and save them to the client NoSQL database before synchronizing to the SQL server. Then, an image status, either already synchronized or not to the server, will be set based on its image status in client DB. “True” for already synchronized to server and ‘False’ whenever it is not synchronized yet. Only images with ‘False’ status will be synchronized to server once apps sync button is pressed. This framework will track synchronization activities either in POST parameter.

A mobile client sends a request to synchronize with server. Once request is received, any images with ‘false’ status will be transferred to the server and store the encoded image in SQL row and column (table) in Server Database. To view an image, PHP file is called to execute Base64_decode function to encoded images. Once the synchronization is completed, notification will be sent out to client devices. Android Logcat will play major roles as it will check images status sync and any deletion of images occurs inside the client database. Meaning, an image ID of any deleted image and image ‘False’ status will be parsed into POST parameter. Any failure of image synchronization to server, images sync status will not be updated to be ‘True’ and images ID of deleted images will be traced in error logcat tray and images deletion resynchronization request will be put back on request queue volley function.

CONCLUSION

Mobile computing has proven to be a fruitful area of concentration for researchers especially in the field of distributed databases and data management. This paper presents the sync gateway to establish connection between heterogeneous databases structures in order to make image synchronization from NoSQL client DB to SQL server DB successfully executed. The proposed framework makes use of JSON language which enables an image transaction in client DB and parses the image data to PHP file in server for synchronization process. Thus, JSON language can solve the problem of limitation of NoSQL client DB for those who are unable to use general SQL query languages to send an image data. From this research, JSON language can intermediate compound of NoSQL DB to send the images data to SQL server without any conflicts. This framework solution provides an alternative for client database to store image schemalessly which is in Document Type database instead of SQL table form. An installation of NoSQL database in client side will help speed up the development phase of database as it requires no schema definition. In addition, by using horizontally indexing scalability in NoSQL client DB, it will not only reduce the workload of mobile devices which does not need to use SQL query language to check the image ID synchronization status in the SQL table data column by column, but also sort the images by their upload status.

ACKNOWLEDGEMENT

The authors would like to thank the UTeM Zamalah Scheme. This research is supported by Universiti Teknikal Malaysia Melaka (UTeM) under UTeM Zamalah Scheme.
REFERENCES:


