Progressive Data Synchronization Model for Mobile Devices

Mehdi Adda, University of Quebec at Rimouski, Canada

ABSTRACT

Mobile data synchronization is an important technique used to replicate or synchronize data between a mobile client and a remote server. It helps overcome unstable wireless networks and support the disconnected operation. The current state of mobile data synchronization is most successful at transparently synchronizing all data. Generally, once a synchronization feature enabled, data is automatically synchronized without offering a fine grained and customizable synchronization policy. Furthermore, the context information, such as localization, date, device overload and resource consumption, type and quality of the connection, etc., are not equally taken into account or not considered at all. One solution is to develop a synchronization model that rely on a fine grained and flexible synchronization policy and where context information is considered as first-class citizen. This article puts forward a new model of data synchronization in mobile devices based on progressive data access schema.

Keywords: Cache, Data, Data Synchronization, Databases, Mobile Applications, Mobile Data Synchronization

1. INTRODUCTION

Mobile devices such as smartphones and tablets are becoming increasingly popular. This popularity may be indorsed to the following factors: i) Mobile devices are reaching computing performances that are comparable to those of desktop computers of five years back. ii) Communication and wireless networks are becoming more capable to insure decent Internet bandwidth to/from mobile devices. For instance, in 2002, the top-of-the-line Palm m515 PDA was equipped with a 33-MHz Motorola Dragonball VZ processor and 16 MB of RAM storage (Microsoft, 2002). Compared to that, the Samsung Galaxy Nexus smartphone is equipped with a 1.2Ghz dual core processor, 1GB of RAM and 4.65” HD (1280x720) screen. As we can see, the CPU speed increased by more than 36 times and the RAM increased by more than 60 times. Add to this, the Galaxy Nexus is 4G capable and may offer network speeds of up to 21 Mbps (Google, 2011).

This popularity is also reflected by the number of mobile applications downloaded. For instance, mobile app stores downloads have reached 8.2 billion in 2010 and are expected to reach 185 billion by 2014 (Gartner, 2011).
However, mobile applications inherit the limitations mobile devices in general and smartphones in particular have. One of those limitations is related to occasional disconnection that users may experience due to wireless networks unstable connectivity. In mobile application development, data synchronization is a mechanism by which data may be replicated or synchronized between a mobile client and a remote server. Whenever a disconnection is experienced, a local data source, a database or a client cache, must be synchronized with the remote data source (database, web service, etc.). Another point in favor of data synchronization is performance; putting data close to the presentation tier of a mobile application decreases latency.

We believe that a successful data synchronization model has to answer questions such as: Is it possible to synchronize only part of the data in a particular context (ex. at a given date, or when doing a specific task)? Are users allowed to participate in building such policies? For example, when executing a mobile application in a given context (date, time, type of connectivity, applications running, etc.), is it possible to trigger a data synchronization operation based on a set of constraints derived from this context?

So far, and to the best of our knowledge, we have not found a (mobile) data synchronization model that positively answers all those questions. In this paper, we propose a data synchronization model that is flexible and may be used to express sophisticated data synchronization policies based on hierarchical data synchronization priorities and context constraints.

Updates in a data synchronization model may be performed based on two different models: traditional update model and optimistic update model (Tancred, Jaakko, & Sasu, 2009). With the traditional update model, conflicts and ambiguities are avoided by coordinating updates between the different devices before the execution. With the optimistic update model, no coordination is required between the different devices to update the data. In the model we propose, conflicts and ambiguities are dealt with after they are detected. We opted for the optimistic update model to ensure that data may be updated on a device that is disconnected from a network. To deal with conflicts and ambiguities, a conflict detection strategy is designed.

This rest of this paper is organized as follows. First, we present setting and notation in Section 2. In Section 3, we present the big picture of the synchronization model followed by the synchronization framework theory in Section 4. Next, we discuss some conflicts that may be encountered in the model in Section 5. Then, we present a progressive synchronization engine and a domain-specific language in Section 6. After that, we discuss related work in Section 7. Finally, we conclude the paper in Section 8.

2. SETTING AND NOTATION

In our setting, we refer to a domain as a finite or infinite collection of values of the same type/category. For example, we may represent the domain of an attribute age as a collection of numbers that belong to the interval [0..120]. Unlike a domain that may be finite, a universe is infinite by definition.

The basic domain types we are using are: i) the domain of strings denoted by $S_n$, ii) the domain of alphanumerical values denoted by $A_n$. The universe of all domains is represented by $U_d$. An attribute is a symbolic concept to which values, drawn from a domain that belongs to $U_d$, may be assigned. The domain of an attribute $X$ is denoted by $[X]$. For instance, an attribute name takes its values from the domain $S_n$ and thus: $[name] = S_n$.

3. SYNCHRONIZATION MODEL: THE BIG PICTURE

The synchronization model we present in this paper is based on the following assumptions:

- The synchronization is managed by the mobile device which is assumed to have
Quality of Experience Models for Multimedia Streaming
www.igi-global.com/chapter/quality-experience-models-multimedia-streaming/62968?camid=4v1a