Gerador de Aplicações Móveis

Summary

In the last few years we've been witnessing a significant technological advance to the level of the mobile systems. This is due to the evolution of two factors. On one hand, the features present on mobile devices which have considerably developed its autonomy as well as its capacity of processing. On the other hand, wireless communications are getting faster and more secure every day. Today, the world is more predisposed to contact with mobile environments than it was a few years ago. Nowadays it is common to see people everywhere working with mobile devices, such as computers or PDA, enabled with wireless connections to the Internet. Mobile communications have grown considerably.

These factors make organizations consider this mobile environment as an opportunity to make their business more agile. Many of them already offer services of technological basis to their collaborators and even their clients. However, it should be considered that many of the solutions presented by these organizations have a difficult integration and evolution. As such, we should reckon that many of the existing applications are property of the organizations that developed them, having to create from root many components like connectivity management, security, synchronization, interfaces, etc. Besides, these applications are of difficult evolution and integration with other solutions found in the market. Thence, there is the necessity of evolving to a standard platform of normalization that facilitates the development of applications and endows them with a set of noteworthy functionalities of modern and fit applications and within recent paradigms of programming.

Important factors to be considered are the inherent constraints to mobile devices that incapacitate these terminals to be constantly connected to the network. Thus being, it will be necessary to enable the platform of normalization with data management mechanisms that make possible for them to be accessed and handled when the devices are disconnected and synchronized with the server, when connectivity is available.

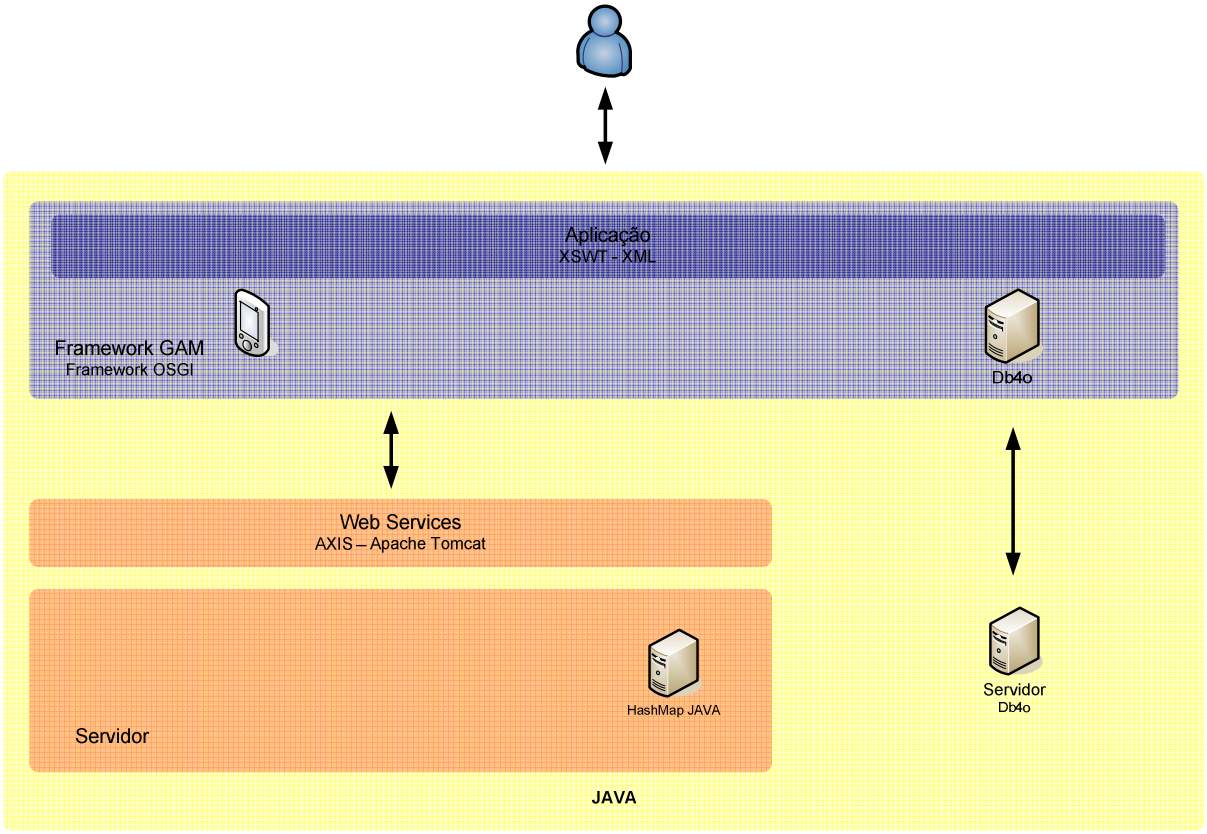
The work carried out within the context of this master's degree dissertation follows a course final thesis – 2005/2006 – designated "Mobile Application Generator".

Previous Work

The work developed in the previous project went directly in the way of the normalization referred to early on, consisting in the creation of a standard platform, built under the OSGI Framework, to speed up and simplify the process of implementation, distribution and maintenance of applications for mobile environments. Namely, graphical interfaces, communication capacities and connectivity management of the terminal where they are lodged, security of data kept persistently and changed with external entities and data synchronization with a server. Therefore, the OSGI Framework ends up being the basis where the GAM Framework was implemented.

Focusing ourselves in the matters of data synchronization, the main objective of this project, the previous project contemplates the synchronization between two databases (Db4o) - client, server. This synchronization is made by request of the user, through a service made available by the Framework to the application. The actions taken by the user in the application are kept as pending operations, if connectivity isn't present, and can later be synchronized with a HashMap table, developed for demonstrative purposes, through the invocation of definite services such as webservices. By "operations" we should takes all the transactions initiated in the application.

The following figure represents the environment developed for communication with the exterior, as well as some used technologies:



Architecture for data synchronization (previous work)

In accordance with this figure, we can see that the GAM Framework manages a local database by application. Communications with the exterior typically occur in two distinct ways: by invocation of webservices and/or synchronization of the local database with the one of the server. As for the technologies, it uses Db4o databases and a HasMap Java for the persistence of data, language XSWT, in the scope of the library used, SWT - "Standard Widget Toolkit", for drawing of the graphical interfaces for the applications and Axis from the Tomcat Project for the creation and implementation of the webservices.

Objectives

The main objective of this work was the development of a Framework, denominated GAM Framework, under the OSGI Framework, capable of speeding up the development, management and maintenance of applications in mobile devices, namely PDA. Continuing with the developed work, some objectives

were drafted in the area of data consistency, such as the improvement of the management of its access or their synchronization between fixed and mobile environment. This dissertation underwent in an initial phase, an investigation of the existing constraints in mobile devices that could affect the consistency and coherence of data, a study of the developed framework and a listing of problems found in the platform made in the previous work. Afterwards, a new architecture of standard data synchronization was designed and implemented under the platform of normalization, and finally, an application was implemented that would display the synchronization mechanisms created.

After concluding the study on what is a mobile device and its main characteristic of mobility, the data access and synchronization mechanisms between the mobile and fixed systems present in the GAM Framework were analyzed, so it could serve as a model for the expected functionalities and the existing gaps in it. After concluding the analysis process of the data consistency, it was necessary to define and to implement a new architecture of synchronization that allows the devices to keep operating in a situation of disconnection. In this sense, when the connection is reestablished, the operations are synchronized with the server and the local database is automatically updated. To perform this architecture an environment was developed, able to serve as a support for the mechanisms previously identified. Finally, an application that could show this particular architecture was implemented.

Work carried out

After the analysis of the developed component of synchronization, we verified that it didn't resolve some important problems, such as:

- Db4o database defined as *server*, in the previous work, ends up not having server functions (it operates totally independent from the rest of the system)

- Db4o database defined as *server* is not in accordance with the legated systems found in the organizations. Normally, appear systems of high scalability in this position. (Oracle, SQL Server, MySQL...);

- Pending operations are not kept in a persistent way;

- It doesn't exist a *fixed* central server that keeps all the information and guarantees its coherence and consistency for use in the mobile environment;

- Automatic mechanisms of synchronization of the mobile database with the server do not exist;

- It doesn't exist a notion of groups of chained operations. The operations are seen independently of the others, without the concept of "transaction" as a set of operations.

Requirements

After identifying the problems, the requirements of the architecture of synchronization to be developed were listed. Hereafter, the architecture would have to be generic and function with any type of application. The low costs of development would have to be maintained, following the work under the platform previously accomplished. As for the synchronization itself, it was to be expected that the architecture to develop would act under the local database and the server, in case the device is connected.

In the mobile device, through any application, the actions of the user should be, in any case, reflected in the local database, so that the user can continue to execute his tasks even when disconnected. In that case, the operations in question would be kept for later in a persistent way, thus the connections were available, to synchronize with the server.

It would be also necessary to enable the Framework with other requirements, such as, the user having total control over the pending operations, being able to visualize them or even delete them with the respective reestablishment of the local database. If by any reason the pending operations fail, the user should be notified of it.

Another subject that it was emphasized in the coordination meetings was the possibility to exist formalized actions in group, instead of being simply atomic. In this case, the user could interact with the system, formalizing a number of chained operations that would only be definitively concluded if all could be fulfilled.

As for the local database synchronization, it would be interesting for the user to be able to synchronize it with the server database whenever he wanted, or being configured to synchronize from time to time.

Solutions found

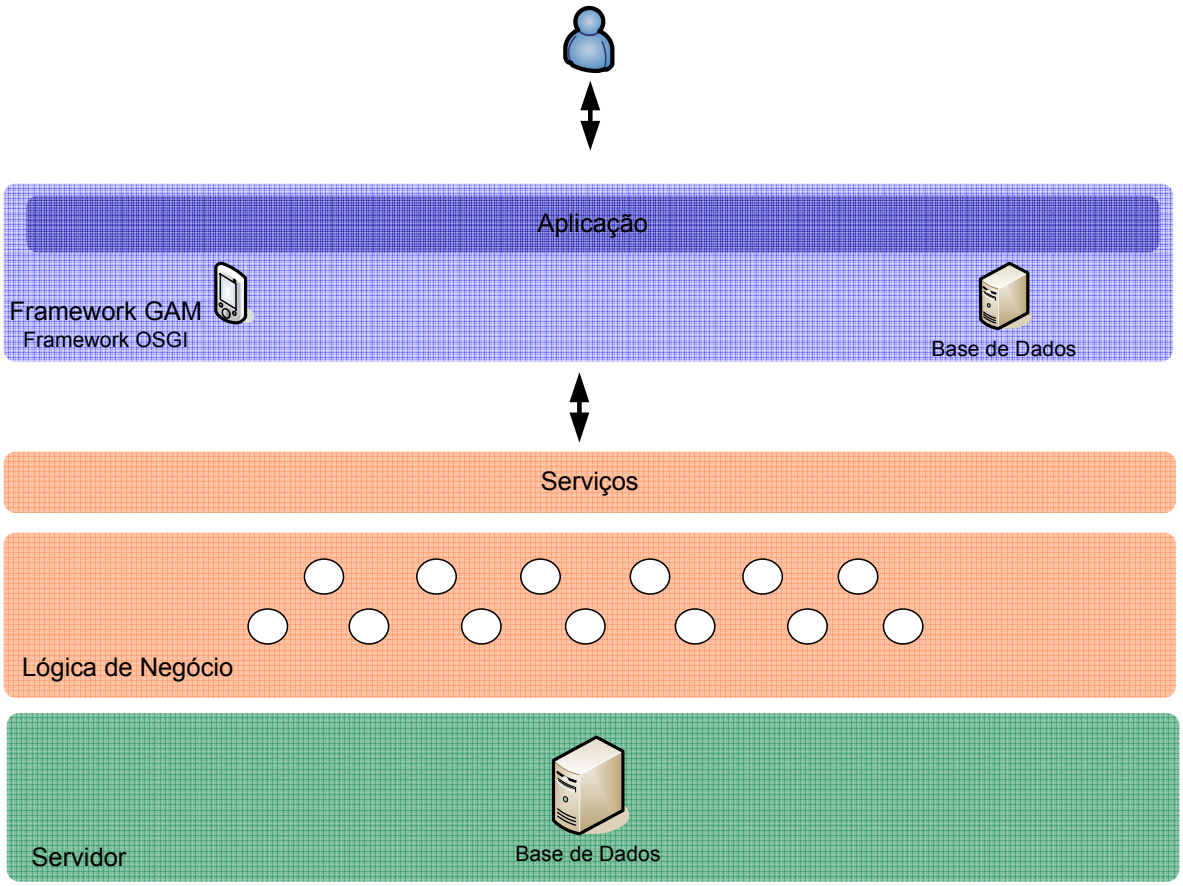
In the sight of the presented requirements, it was necessary to develop a series of solutions that would allow resolving them.

Mobile System Database

It was necessary to migrate the developed platform to an environment closer to the legated systems of the organizations. The Db4o database created as a "server" was substituted by the database management system MySQL. This system was responsible for consolidating all the information and guaranteeing its consistency and coherence. The actions taken by the user through the application (operations) had been redefined to invoke services that read/write in the central database (server).

Flexible architecture Client-Server

One of the most important transformations implemented was the definition and implementation of architecture of support to the communications called "flexible architecture client-server". In this type of architecture, the Application/Framework behaves as a *client* responsible for executing its operations in the server, but also as an *autonomous client* when completely disconnected from the network. In that way, it would be expectable for the Framework to be able to manage a local database by application, so that the device could operate when disconnected from the network. This management goes through methods of reading, updating and writing. To allow the database to be constantly updated, a single database defined as server was created and some automatic mechanisms of synchronization, such as, one thread of synchronization that can be configured by the application and some mechanisms that the Framework uses to update the necessary objects in the local database as the operations are executed in the server. Thus, the updates of objects on the local database are easily executed, using the technique of replication of the objects from the server to the local environment, keeping the environment with coherent and consistent data.



Architecture developed for communication with the "server"

Pending Operations

As for the pending operations, the concept of “*critical*” and "*non-critical*” services was created, to distinguish between a service that needs an immediate answer or can be kept and be later synchronized with the server. The operations had to be modified and kept in a persistent way, using the tool for persistence of local data based on objects - Db4o. Through services that the Framework makes available to the Application, it was given total control to the user to see/delete all pending operations or its report, in case of failure in the execution.

Transactions

This concept comes from the necessity of creating a system that would allow to “chain" operations that presented themselves as totally independent. With this mechanism, it is possible to invoke various types of operations, to create dependence between them. In this way, any user will be able to place two operations of transference in the system that will only be realized together. That is, if one of them fails the other won’t be carried out.