BROKER OF DYNAMIC AND STRUCTURED INFORMATION
FOR MOBILE DEVICES

PEDRO MIGUEL PEREIRA LOPES

ABSTRACT. Why did we create countries to join a immense group of people? Why did the mobile phone, PDA, laptop and smart-phone concepts got merged? Why is there a need to unify? The simple answer would be: minimize the clutter, maximize the efficiency, and focus on the substance of the problem. Data however, is a paradigmatic subject which will be studied in depth throughout this paper: when unified and transformed into information, the once siloed and of little interest data, reveals it’s real value when combined with other pieces of data. Various methods, concepts and examples were analyzed, in order to develop a structure where various sets of data could be efficiently acquired and unified into pieces of relevant and useful information. A great part of this work was developed at M-Insight Technologies (an technology based company, dedicated to multimedia content’s edition and distribution, of high graphic value, in mobile devices), and applied in it’s main product.

1. Introduction

Having such a large amount of in data available, there is an opportunity to gather it and unify it, so that a greater value can be extrapolated from these data. Mashups are a good example of this growing trend: executive editor David Berlind says mashups are the fastest growing ecosystem on the Web and that by 2007, there will be 10 new mashups per day[1]. The focus of this work however, is another form of data gathering and unifying, which is the information broker. An information broker functions just like mashups, the difference being that the first is majorally applied outside the web sites context, and the second is connoted as an information merging mechanism for web sites.

The reason why the information broker was chosen as a major focus for this work, was due to a challenge proposed by the M-Insight Technologies company: this being to build a system where various contents could be inserted automatically on a structured database (latterly to be used on various applications and mobile devices). This challenge led to others, since there was a need to support this automatic system with a comprehensive edition process, where contents could be inserted manually (this is, by specialized personnel), supporting the automatic content retrieval and insertion on to the structured contents data base.

This work is inserted in M-Insight’s main product project. The project consists on the gathering, distribution and presentation of touristic information for mobile devices. Consider this example: a tourist arrives to Lisbon and wishes to know more about this city. He uses the product stated above on his mobile device and acquires information of: what places to visit, guided tours, lodging, restaurants and other touristic information.
To sustain the above product’s concept, the supporting system should allow a user to be able, through technology and applications, to obtain personalized information from the heterogeneity of inputs and through its contextualization. That is, regardless of the device and consequently the technology, the client has to see what information (searched according to the client’s user profile, its target device and location) was transmitted.

Having a structure where the client device can interact with the server and obtain personalized contents, arises the need to produce these contents and to systemize its gathering, editing and management. Two major systems will be used to meet this need:

- **Manual System** - Creation, modification and maintenance of contents will be made. The core information will be placed by specialized personnel (e.g. tourism professionals) and all the maintenance (e.g. content approval) will be made by people who are authorized to do so. Typically, the core information is obtained through web sites, books, people and organizations, being later inserted into the global system by the appropriate means.

- **Automatic System** - A broker will be the centerpiece of this system, using sets of sources that will be used to collect data, in order to obtain relevant information (contents) for the global system.

The union between the Manual System and the Automatic one will be designated as **Content Edition and Management**, and the process that will be used produce contents, will be named as **Edition Process** (this process will make use of these two systems).

This paper will present the problematic of data unification and transformation to information, and a data unification and treatment solution will be made to unravel a specific problem of a business enterprise product.

### 2. How to Unify Information

The following sub-sections take different two names but serve the same purpose: data unification. Whilst **data integration** is a broad area that implies all methods of data unification, the **mashups** are connoted as a means to unify data for the web.

#### 2.1. Data Integration

Data integration is the process of combining data residing at different sources and providing the user with a unified view of these data [2]. It has been the focus of extensive theoretical work and numerous open problems remain to be solved. In management practice, data integration is frequently called Enterprise Information Integration.

The problem of combining heterogeneous data sources under a single query interface is not a new one. The rapid adoption of databases after the 1960s naturally led to the need to share or merge existing repositories. This merging can be done at several levels in the database architecture[3]. There are two major approaches to tackle this problem:

- **Data Warehousing** - differed style of integrating data. All the data is collected into a single data base, which is then used for all the queries for information.

- **Wrapper Mediator Method** - immediate style of integrating data. The information queries are done through a mediator that is linked with various
wrappers. There is no need for a data base that unifies all the data, because all the data is gathered from the wrappers (which obey to a mediated schema).

**Data Warehousing.** A data warehouse is a repository of an organization’s electronically stored data designed to facilitate reporting and analysis. An efficient data warehouse is able to retrieve and analyze data, to extract, transform and load data, and to manage dictionary data – so that it can be used for business intelligence.[4]

This classic definition of the data warehouse focuses on data storage. However, the means to retrieve and analyze data, to extract, transform and load data, and to manage the data dictionary are also considered essential components of a data warehousing system. Many references to data warehousing use this broader context. Thus, an expanded definition for data warehousing includes business intelligence tools, tools to extract, transform, and load data into the repository, and tools to manage and retrieve metadata.

Here data from several sources are extracted, transformed, and loaded (ETL) into source and can be queried with a single schema. This can be perceived architecturally as a tightly coupled approach because the data reside together in a single repository at query time. Problems with tight coupling can arise with the "freshness" of data, for example when an original data source is updated, but the warehouse still contains the older data and the ETL process needs to be executed again. It is also difficult to construct data warehouses when you only have a query interface to the data sources and no access to the full data. This problem frequently arises when integrating several commercial query services like travel or classified advertisement web applications.

**Wrapper Mediator Method.** Using this method, the external resources are considered as materialized views over a virtual mediated schema, resulting in "virtual data integration". This virtual schema is called the mediated schema. For each data source (such as the crime database and weather website) a wrapper (or adapter) is made tailored. These wrappers simply transform the local query results (those returned by the respective websites or databases) into an easily processed form for the data integration solution. When a query is made to the mediated schema, the data integration solution transforms this query into appropriate queries over the respective data sources. Finally, the results of these queries are combined into the answer to the first query.

This method can also be called as view based query answering because we can consider each of the data sources to be a view over the (nonexistent) mediated schema. Formally such an approach is called Local As View (LAV), where Local refers to the local sources/databases. An alternate model of integration is one where the mediated schema is designed to be a view over the sources. This approach called Global As View (GAV) - where Global refers to the global (mediated) schema - is often used due to the simplicity involved in answering queries issued over the mediated schema. However, the obvious drawback is the need to rewrite the view for mediated schema whenever a new source is to be integrated and/or an existing source changes its schema.

2.2. **Mashup.** A good insight as to what makes a mashup is to look at the etymology of the term: it was borrowed from the pop music scene, where a mashup is a new song that is mixed from the vocal and instrumental tracks from two different
source songs (usually belonging to different genres). Like these "bastard pop" songs, a mashup is an unusual or innovative composition of content (often from unrelated data sources), made for human (rather than computerized) consumption.

In technology, a mashup is a web application that combines data from more than one source into a single integrated tool; an example is the use of cartographic data from Google Maps to add location information to real-estate data, thereby creating a new and distinct web service that was not originally provided by either source.

Many people are experimenting with mashups using Amazon, eBay, Flickr, Google, Microsoft, Yahoo and YouTube APIs, which has led to the creation of various mashup editors like Yahoo Pipes, Google Mashup Editor, Microsoft Popfly, JackBe’s Prest, IBM Sharable Code, Serena Software and Kapow Robosuite.

Executive editor David Berlind says mashups are the fastest growing ecosystem on the Web and that by 2007, there will be 10 new mashups per day.

3. Information Broker Implementation

3.1. Supporting Structure. The theory behind an information broker was addressed in the previous part, but one must first understand the structure in which the broker will be based upon. Since this work will be implemented in a corporate context, in which the global vision was already defined, one must first study what is the globally defined structure and where does this work will be placed.

Along this work, a screen terminology will be used to characterize certain types of screens (produced by this supporting structure) that will appear on the client’s mobile device:

- **L0** - The root screen, which is used as an "index" of the region’s information. Clicking on any of these "index items" will lead to a L1 or Lx1 screen.
- **L1** - Listing of items. Clicking on one of these items will lead to a Lx1 screen.
- **Lx1** - Basic description of an item. Through this screen, one can access the Lx2 screen.
- **Lx2** - Detailed description of an item.

To speed up the production process, the **macro template** concept was created. A macro template can be thought as a template screen that serves as a layout to produce - in a fast and efficient way - a substantial number of similar screens.

The global architecture overview in which all the work will be developed is presented in Figure 1 whose players are:

- **Client Device**
  - **Resources** - Used by the content interpreter and obtained through the server.
    - **XML** - Typically, these files will hold the screen’s structure and properties.
    - **Multimedia Files** - Images, videos and sounds referenced by the XML; therefore used on the screen’s presentation.
    - **Other** - For example, configuration files used by the content interpreter.
  - **Content Interpreter** - Interprets and processes the resources and uses the presentation module to expose this information to the user.
Figure 1. Global Architecture Overview

- **Presentation Module** - Presents screens and other information to the user and manages user interaction.
- **Mobile Services Provider** - TMN, Vodafone or Optimus are examples of such. This will be the bridge between the server and the client’s mobile device.
- **Server**
  - **Requests Manager** - Receives and processes the client’s requests and sends the proper resources back to the client. The translation from the structured contents in the data base to XML is done here.
- **Content Management**
  - **Edition Module** - The creation, maintenance and management of structured contents will be made in this module. It will serve as a bridge that links the information broker with the data base as well.
  - **Information Broker** - Does automatic information retrieval of information using various sources and forwards this structured information to the data base (through the edition module).
- **Data Base** - Contains structured content information, multimedia files and other types of information.
- **Information Sources** - These sources can be the ones used directly by the information broker (e.g. Web Services and Web Feeds) or other kind of
sources (e.g. information obtained on-site, books, tourism agencies) which are used by the content editors.

The **Information Broker** and **Edition Module** (which is closely related with the first) will be the main focus of this work.

### 3.2. Edition Module.

#### 3.2.1. Architecture.

The Edition Module is composed by (Figure 2):

- **MI-Editor** - edition of micro templates, macro templates, screens, content’s characteristics and approval management.
- **MI-Manager** - management of editors, entities and regions.
- **Content Management Libraries** - set of libraries that manage content’s data and their persistence in the data base.

![Figure 2. MI-Edition Module Architecture](image)

The MI-Editor/MI-Manager is an important separation of concepts: while MI-Editor serves the edition part of the whole production process, this is, screen edition and maintenance, the MI-Manager manages properties that are used in various parts of the production line (website, screens and payments). Therefore, this disassociation increases the simplification, concept and security of the two applications.

#### 3.2.2. MI-Editor.

MI-Editor is a C#.NET developed Windows application. It’s main purpose is to aid the screen/macro template/micro template development and deployment and all editors, reviewers, managers and specialized technicians can access this application (although there are certain restrictions for each role). This application will mainly communicate with a data base.

The GUI development of this application was largely facilitated due to the usage of the Infragistics NetAdvantage libraries for .NET. These libraries are a comprehensive suite of ASP.NET and Windows Forms controls, components, and tools for the .NET platform. Thanks to these libraries, a great amount of development time was cut.

The major features of this application are:

- **Content edition**
- **Screen/macro template) edition and approval**
- **Micro Template edition**
- **Fast screen edition**
• **Fast screen preview** - using a phone emulator (Sun Java (TM) Wireless Toolkit 2.5.2 for CLDC)
• **Resources synchronization** - copies the screen’s resources from a source folder to the emulator’s folder and/or the server’s folder.
• **Import/export of screens/macro templates/micro templates** - persistence of data into a .DAT file

3.2.3. **MI-Manager.** The main purpose of this application is to help manage content properties that are only editable by managers, this is, only managers can enter this application. These properties are: regions, entities and editors.

Just like MI-Editor, the MI-Manager application is an C# .NET developed Windows application, using the Infragistics NetAdvantage libraries for .NET.

3.2.4. **Content Management Libraries.** The Content Management Libraries were developed on C# .NET and are composed by:

• **MI-Business Edition** - Manages all of the content’s persistence and offers a comprehensive structure to handle contents.
• **MI-Windows Components** - Library for GUI components and classes

3.3. **Information Broker Structure.**

3.3.1. **Architecture.** The chosen architecture is one based on the wrapper mediator method, given that there is no a need to cross different types of data or to make inferences between data. The sole purpose was to retrieve updated data and process rapidly. Without a doubt, the wrapper mediator method was the best choice for this information broker’s architecture.

The **Main Actors.** The real power belies on the system’s simplicity and the distribution of tasks throughout the various components. In Figure 3 are represented the main actors of the Information Broker:

• **Broker Engine** - the mediator. Acts as an orchestrator of the different broker service implementations that are attached to it. It calls the broker service implementations, receives their data, and sends to the edition module, in order to persist this information.
• **Broker Service Interface** - the schema. All the broker service implementations must link to this interface. This interface is very simple and basically obliges its implementations to have a getDataSet() method.
• **Broker Service Implementation** - the wrapper. DLL files that will be consumed by the information broker. These implementations are responsible for the translation of the outside source information to a "DataSet".\(^1\)
• **Information Sources** - the data sources, although in this case they can be considered information sources, since the data gathered is in a useful and structured form. E.g. Weather Underground’s RSS feeds\(^8\), Weather Bug’s RPC Web Services\(^9\) and Eventful’s REST XML Web Services\(^10\).

---

\(^1\) The DataSet is a memory-resident object that can contain one or more tables and relationships between tables. This memory-resident object and its child objects make up the disconnected data source that is the centerpiece of the ADO.NET architecture. Figure ?? presents an overview of the DataSet object model.\(^7\)
Since the broker service implementations only have to link to the broker’s interface and don’t know how the broker’s core works, they can easily be developed by third party organizations, broadening the usage possibilities of these components.

4. Evaluation

All of the above chapters merge here. Results and measures derived from the work made above are listed in this chapter.

4.1. Demonstration: Updating a screen with weather information. The developed structure and applications described above are used in this small demonstration, which aims to give a simple overview of the whole production process.

Figure 3 presents a very simplified demonstration on how to create a L0 Screen (L0 Lisbon) with an overlay of the Lisbon’s current weather conditions. The two major pieces of developed work (the MI-Editor and MI-Information Broker) are featured in this illustration, where their most significant functionalities are portrayed (macro template/screen edition by the MI-Editor and content update by the MI-Information Broker).

4.2. Screen Production Efficiency. Table 1 presents the measurements obtained for Lx1 screen production times.
TABLE 1. Screen Production Time

<table>
<thead>
<tr>
<th>Screen Creation (min)</th>
<th>Screen Editing (min)</th>
<th>Producing 150 Screens (min)</th>
<th>Producing 1000 Screens (min)</th>
<th>Producing 1600 Screens (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using MI-Editor with Macro Templates</td>
<td>4</td>
<td>0.15</td>
<td>605</td>
<td>4037</td>
</tr>
<tr>
<td>Using MI-Editor without Macro Templates</td>
<td>74</td>
<td>0.95</td>
<td>11135</td>
<td>74237</td>
</tr>
<tr>
<td>Using Database Editor</td>
<td>48</td>
<td>1.75</td>
<td>7265</td>
<td>48437</td>
</tr>
</tbody>
</table>

Clearly, the MI-Editor with Macro Template wins over the other two options, due to its low screen creation and maintenance time.

Note the MI-Editor without Macro Templates results. The time consumed by this option is considerably higher that the others because every screen must be created from scratch, this is, every content must be placed individually in the MI-Editor’s personalized mode, which consumes a great deal of time.

The tendency over time, will be the increasing demand of an higher number of screens produced, and the MI-Editor with Macro Template solution that was presented throughout this document proved out to be a good choice, as demonstrated above.

5. Conclusions

The developed information broker became a key piece of a company’s edition process and increased the value of the information that a client could receive on his mobile device. However, to reach a point where the broker could be implemented,

1Screen Creation - Creation a screen from scratch (using or not using a macro template)
2Screen Editing - Editing a single parameter of the screen and saving it.
3Screen Producing - The formula used to calculate the time consumed to produce X screens was: (Screen Creation Time)*X + X*(1/4)*(Editing Screen Time). This meaning that there is a supposition that one quarter of the screens will be edited. The chosen target values were 150, 1000 and 1600, because 1600 is the number of projected screens to be made until the production phase. The 150 target was chosen because there were measurements of the time consumed to make 150 screen, and it was relatively close with the estimates made in this work. The 1000 target was chosen because it is a mid-range target (considering the 150-1600 range).
4MI-Editor with Macro Templates - Creation and editing of screen using macro templates, this is, fast edition.
5MI-Editor without Macro Templates - Creation and editing of screen without the help of macro templates, this is, using only the personalized edition.
6Database Editor - Creation and editing of screen using only the Microsoft SQL Server Management Studio and some scripts that help the creation and edition of screens in the database.
a whole basis had to be laid and developed upon (edition process, MI-Editor, MI-Manager, general concepts), so there could be a basis where the broker could work on.

The information broker and data unification and transformation into information are not new concepts, so the added value of this work is the implementation of these concepts into a specific enterprise solution. Based on a proved concept (the wrapper mediator method), the Information Broker was developed based on a well known basilar structure and implemented to solve a specific problem on a (pragmatic) enterprise context.

The work developed on MI-Editor, MI-Manager, MI-Information Broker and general edition process concepts are currently being used as part of a whole production process on a company. By the time of this work’s publication, the MI-Editor was used by two tourism professionals to create, edit and approve 300+ screens. Also the MI-Manager is being used to manage parts of the edition process, and the MI-Information Broker is up and running, delivering updated weather information to some screens.

References

[33] C. Commons, “Creative commons attribution-share alike 2.5 generic.” Website. [http://creativecommons.org/licenses/by-sa/2.5/]
[41] T. Primrose and E. Adler, “America online and time warner will merge to create world’s first internet-age media and communications company.” TimerWarner.com - Website, 2000. [http://www.timewarner.com/corp/newsroom/pr/0,20812,687602,00.html]

INSTITUTO SUPERIOR TECNICO - UTL, AVENIDA PROFESSOR CAVACO SILVA, 2744-016 PORTO SALVO, PORTUGAL

E-mail address: pedro.lopes@tagus.ist.utl.pt
Figure 4. Demo 3: Updating a screen with weather information