Automatic Creation of Mobile Web Applications from Design Models

Pablo Martín Vera[1], Claudia Pons[2], Carina González González[3], Rocío Andrea Rodríguez[1], Daniel Alberto Giulianelli[1]

[1] National University of La Matanza
GIDFIS - Research, Development and Training Group on Software Innovation
Department of Engeniering and Technological Research
San Justo, Buenos Aires, Argentina
{pvera, rrodriguez, dgiulian}@ing.unlam.edu.ar

[2] National University of La Plata
LIFIA – Research and Education Laboratory on Advance Computing
La Plata, Buenos Aires, Argentina
cpons@lifia.info.unlp.edu.ar

[3] La Laguna University
Department of Systems Engineering and Automation,
Architecture and Computer Technology
La Laguna, España
cjgonza@ull.es

Abstract. This paper shows a methodology and a support tool for automatically creating the source code of a mobile web application from design models. A designer only needs performing two models: Data model and User Interface with Navigation Model. All models are based on UML extended with stereotypes and tagged values allowing behavior configuration. Models extensions contain all necessary information for creating the complete source code of an application. A support tool was build easiness the process of modeling and configuring with the proposed methodology. This support tool has an example implementation allowing the generation of an ASP.NET MVC 4 mobile web application.

Keywords: MDD, Models, Mobile Web Applications, UML, Components

1 Introduction

Developing Software by Making Models and deriving source code is a tendency started several years ago. These techniques can be found with several names MDD (Model Driven Development), MDA (Model Driven Architecture), MDSE (Model Driven Software Engineering). All these techniques has something in common, they use models and transformations to generate source code. A transformation is a process that takes as input a model and generates a target model or source code. For example

OMG MDA Approach [1] uses different kinds of models with different types of abstraction levels, starting from Platform Independent Models (PIM) to Platform Specific Models (PSM). From PIM to PSM there are automatic or semi-automatic transformations. And the final goal of these techniques is to automate the generation of the application source code, making designers focusing on the models rather in the coding process. But most of existing techniques are difficult to use, requires to do a really complex process detailing models and configuring transformations in order to obtain useful code, and most of them only can generate part of that code.

In the other hand the use of components in the software development process is a very well establish technique started several years ago [2]. Components are pre-defined pieces of software with a very well establish purpose. Components are excellent for re-use and they are reliable because once they were tested they are re-use without modification. Most of modern programming frameworks include pre-defined components for easiness the development process. For example login component or some UI components like grids, carrousels, etc.

Our proposal is the use of components as modeling pieces, adding to pre-established components some configuration capabilities in order to be adapted to the system being modeled.

2 Targeting mobile applications

This methodology is proposed for modeling mobile web applications for the following reasons:

- **Reduced User Interface**: Screens sizes of mobile devices are small, beyond their resolution. Therefore the user interface shown to the user must be simple, comfortable and suitable for the way of using it. As this methodology is based on UI components, each one will represent a screen that will be shown to the user. Being the user interface necessarily simple, makes the component configuration direct without having to specify complex control layouts. Configuration is focused on the information to be shown and the web pages will be generated with a standard layout optimized for mobile devices visualization.
- **Simple and Intuitive Navigation System:** Because of having a reduced screen, navigation system should also be minimized. Navigation options were added to each component. Later when creating the application source code, navigation bar will be shown on the top of the page following W3C (World Wide Web Consortium) guidelines about mobile web sites [3].
- **Taking advance of special characteristics**: some elements were added to take advance of capabilities on mobile devices, like SMS and Call links, also allowing the use geo-location taking advance of the GPS present on most devices.

Nevertheless this methodology can also be applied to some desktop web applications where complex screen layouts are not required like standard intranet systems or a configuration backend.

3 Component Based Hypermedia Design Method (CBHDM)

CBHDM is a design methodology for designing mobile web application and generating it´s source code using extended UML Models.

CBHDM requires only two models: the data model and the user interface model (which includes navigation modeling). The data model is based on the UML Class Diagram and the interface model uses UML component diagrams. Both diagrams are extended with stereotypes and tagged values generating a conservative extension of UML. This allows creating a CBHDM model with any existing modeling tool, but in order to easiness the process a custom tool was specifically design to support the methodology. This tool also performs all necessary transformations to create the final source code.

The methodology starts from the UML Class diagram where the system entities are defined using some stereotypes that will be used on the final transformation to facilitate the process of generating the database script, and for identifying and describing entities on the system. Later a transformation tool uses the class diagram to automatically generate a Component Diagram that the designer will modify and complete with the desire behavior of the interface. The last step consists in using the transformation tool for a second time with all models as input. The result will be a database script and a fully functional application source code. Figure 1 shows the different stages of the methodology and also remarks the steps that requires user participation.

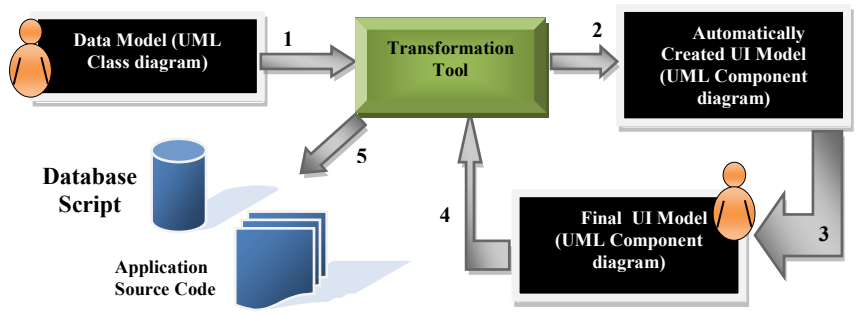

Fig. 1. CBHDM Methodology steps

4 Data Model

Data model is based on the UML class diagram where each class will represent a database table. This diagram is extended with:

(A) **Stereotypes for special properties:** like the numeric unique identifier of the table and the textual descriptor which gives a human readable identifier of the database register.

(B) **Enumeration Classes:** In some applications it will be necessary to identify those classes representing limited values, like for example the different states of an invoice, a list of allowed actions, etc. Those classes must be distinguished from other classes because they will have a special treatment when generating the source code. In order to identify those classes the UML stereotype `<enumeration>` will be used.

(C) **Special Data Types:** In order to improve user experience when using the application, especially from a mobile device, two special data types will be used:

- `phoneNumber` which is a string representing a telephone number allowing to render a link to automatically make a phone call or send a SMS to that number;
- `address`: this data type will allow using the GPS device available on advance mobile phones and take the exact location where the user is located. If the GPS is not available or is temporally out of service, it will be possible to enter the address like a string. Also this data type will allow using the GPS to navigate to that location.

Support tool allows adding classes and enumerations (Figure 2, left) easiness the process of configuring the special properties (Figure 2, right).

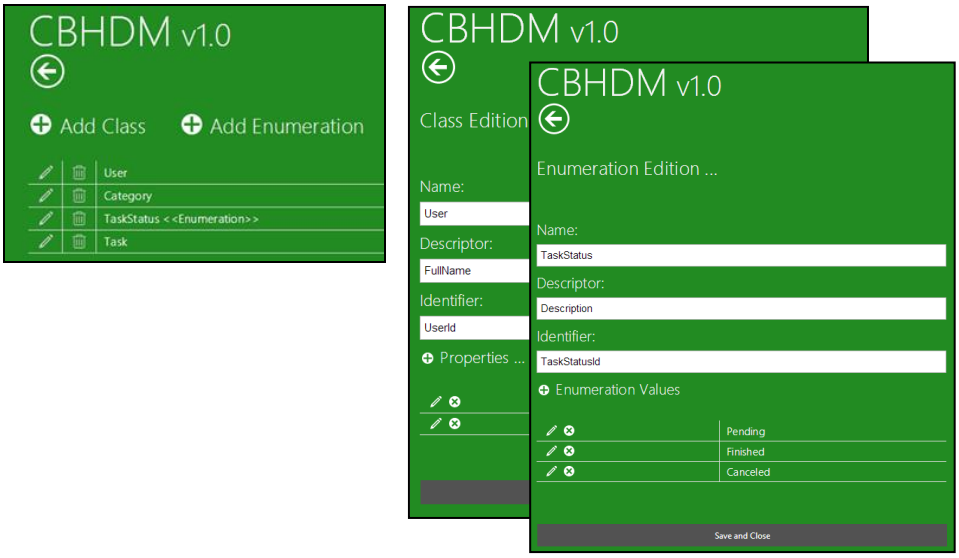

Fig. 2: Screenshots of the tool support for Data Model

After configuring enumeration and classes, the system automatically generates a preliminary version of the UI Model, using "Generate Components" option, but also with the data model, a database script can be generate (see Figure 3).

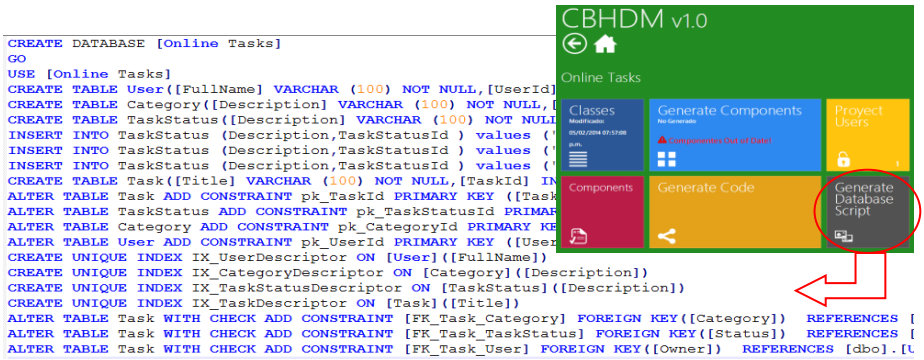

Fig. 3. Generating Database Script from UI Model

5 User Interface Model

The user interface will be described by using a set of pre-defined components types. The components types are defined by stereotypes and each type will have some tagged values in order to configure them.

Pre-defined components are: `Login`, `List`, `Search`, `Menu`, `CRUD`, and `UpdateView`.

From data model, a preliminary version of the component diagram is automatically created. For each class (not for enumerations) a list and a crud component are created.

A main menu is also created pointing to list components. Figure 4 shows automatically created components from data model of Figure 2.

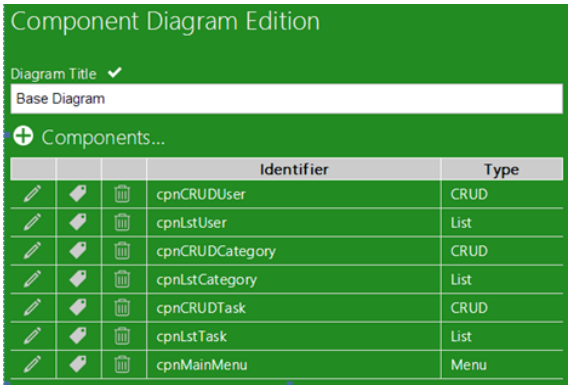

Fig. 4. Automatically created UI components from Data Model

Each component is configured using tagged values. These tagged values gives the personalization options for adapting components to the application being modeling.

Al automatic created components are preconfigured. For example, a List component shows a column with the descriptor of the class being displayed and has an action link to edit each item. In the Navigation bar a link is created to add new objects and another link allows going back to the home page. (See Figure 6 left).
The complete component configuration with all allowed tagged values can be found in [4].

Available components are:

- `Login`: a component to authenticate the user on the system. It configuration allows defining to the properties of the class to perform the authentication (See Figure 5 left). The component can also be configured in order to allow the user to be selected instead of having to be typed (Figure 5 right). This feature is useful for mobile devices with few users, minimizing text input, which is a complex tax especially on small devices with reduced keyboards.

Fig.5. Pages automatically created from Login Components

- `List`: a component that shows a list of information to the user. For example Figure 6 on left shows a List component showing the categories list. This list is configured to show only the category description. But the list can also be configured to show more than one property, for example the list of figure 6 on the right shows a task list displaying, the task title, the category of that task and an additional information line with the description. All this features can be configured in the components using tagged values.

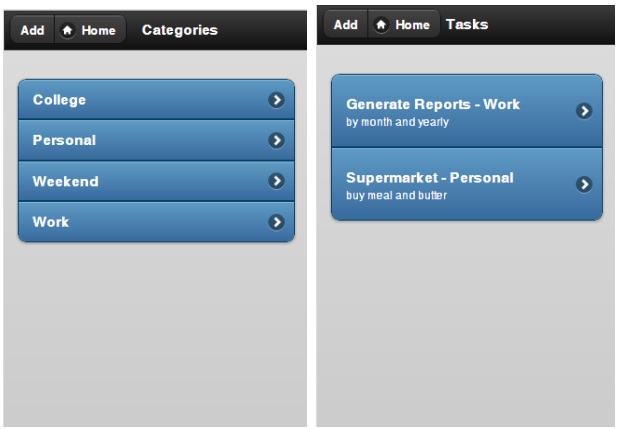

Fig. 6. Pages automatically created from List Components

- `Search`: like `List` is a component to show information to the user but it also allows filtering the information. It uses the same tagged values of List adding an extra tag called `SearchFilters`. This tag allows configuring which properties could be filter and the type of the filter that will be applied (free text, single selection, multiple selection, data range, and yes/no selector). Figure 7 shows several filters to search a task.

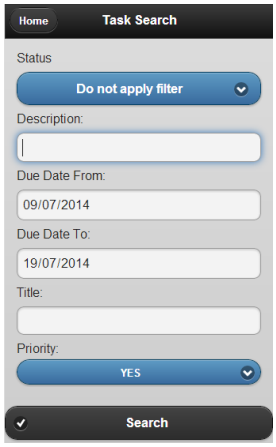

Fig. 7. Page automatically created from a search component

- `Menu`: this component shows a menu to the user. In order to configure the menu options a set of links are configured pointing to others components.

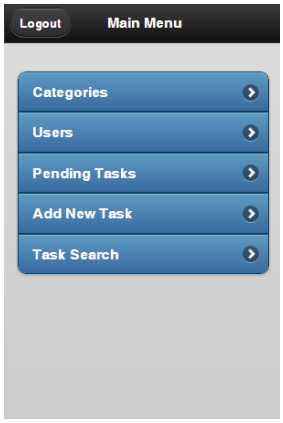

Fig. 8. Page automatically created from Main Menu component

- `CRUD`: is a component to create, show, update or delete a record of the data base. By default it asks the user to fill all properties, but it has a powerful configuration allowing:
    - Defining default values when creating or updating a record

- Avoid writing some field of the record when creating or updating data.
- Creating an additional record of another table when creating or updating data.
- Defining optional Properties

Figure 9 shows a `CRUD` component for creating a new Task. Some values are set by default so, they are not asked to the user, for example newly created task are always in pending status.

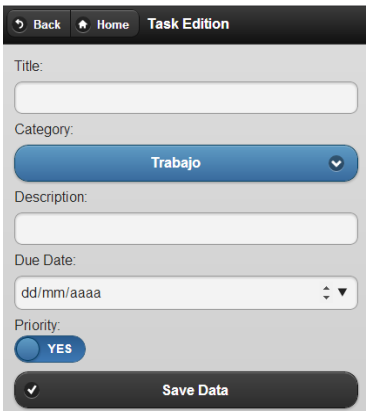

Fig. 9. Page automatically created from CRUD component

- `UpdateView`: this component is an update operation with special characteristics. It´s possible that a record needs to be updated partially where some properties must be editable and others doesn´t. It´s especially suitable for records that are partially filled when moving from different states on the system. For example Figure 10 shows an update view component for finishing a Task. When finishing a task, status will be changed to "Finished" and only some comments can be added by the user. This component is also configure to show in read only mode the due date, title and the category.

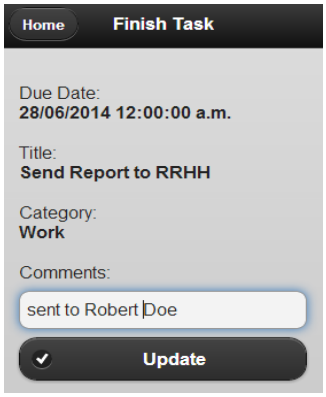

Fig. 10. Page automatically created from Update View component

CBHDM also includes some characteristics like the use of complex functions for data retrieval and creating views for different roles, these characteristics can be found in [5].

6 Related Work

Several methodologies based on MDA are oriented to the development of web applications including UI modelling like for example:

- UWE (UML based Web Engineering) is based on several UML diagrams, with stereotypes and automatic transformations in order to generate a web application. It requires doing a big number of diagrams and to apply automatic transformations for example from content to navigation. Is possible to automatically generate nodes (navigation units) which will be connected by links but the user must do a big work to check if each automatically generated node is really needed for the correct navigation on the system, and must delete all unwanted nodes (which in some cases force to restructure the start diagram) [6]
- OOHDM: is a modeling methodology consisting in four activities: conceptual modeling, navigation modeling, abstract interface design and implementation [7] It models interface using ADV (Abstract Data Views) [8] where different objects are grouped to created a screen mockup.
- IFML (Interaction Flow Modelling Language): It was born on 2013 as an OMG (Object Management Group) standard. It replaces WEBML (Web Modeling Language). It supports modeling of desktop and mobile web applications [9]. Is limited in several aspects and difficult for the designer when defining some particularities like class relations with foreign keys and enumeration classes.

This three methodologies models user interface by creating diagrams with containers, controls and data references. Is like creating a mockup for each screen which is complex and usually requires more work than really building the interface in the target development environment. Nevertheless some of them allow generating interfaces source code from the models, but not the whole application source code.

7 Conclusions and Future Work

CBHDM is a modelling methodology that allows modeling systems with all necessary information for automatic code generation. With the help of the tool, designer doesn´t need to remember how to configure components because tool guides the whole process.

The tool includes transformations from data model to components and from components to code.

By just doing data diagram and configuring auto created components designers can have the complete application source code and database script created in few minutes, without writing a single line of code.

As future work when generating the source code, user will be able to choose between different templates allowing to generated code on different languages or with different front ends, for example in XHTML for low performance mobile phones.

References

1. OMG: MDA Guide Version 1.0.1. (2003), http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf
2. Heineman G.T., Councill W.T.: Component-Based Software Engineering: Putting the Pieces Together. Addison-Wesley Professional, Reading. (2001)
3. W3C: Mobile Web Best Practices 1.0. (2008), http://www.w3.org/TR/mobile-bp/
4. Vera P., Pons C. Gonzales C, Giulianelli D., Rodriguez R.: MDA based Hypermedia Modeling Methodology using reusable components, XVIII Congreso Argentino de Ciencias de la Computación (2012)
5. Vera P., Pons C. Gonzales C, Giulianelli D., Rodriguez R.: Modeling Complex Mobile Web Applications from UI Components – Adding Different Roles and complex Database Design, XIX Congreso Argentino de Ciencias de la Computación (2013)
6. LMU – Ludwig-Maximilians-Universität München.: UWE – UML-based Web Engineering. Institute for Informatics. Research Unit of Programming and Software Engineering (2012) http://uwe.pst.ifi.lmu.de/index.html
7. Schwabe D. y Rossi G.: An object oriented approach to Web-based applications design. Theor. Pract. Object Syst. Volume 4, Issue 4, pp 207-225. (1998)
8. Cowan D. and Lucena C.: Abstract Data Views: An Interface Specification Concept to Enhance Design for Reuse. IEEE Trans. Softw. Eng. 21, 3, pp. 229-243. (1995).
9. OMG: Interaction Flow Modeling Language (IFML) (2014) http://www.omg.org/spec/IFML/