A First Draft of a Model-driven Method for Designing Graphical User Interfaces of Rich Internet Applications

Francisco J. Martínez-Ruiz[1], Jaime Muñoz Arteaga[2], Jean Vanderdonckt[3]
and Juan M. González-Calleros[3]
[1]*Universidad Autónoma de Zacatecas. Departamento de Ingeniería en Computación, Avenida Ramón López Velarde No. 801, Zacatecas, Zac., 98060. México. fjmr1@yahoo.com.mx*
[2]*Universidad Autónoma de Aguascalientes. Centro de Ciencias Básicas Aguascalientes, México. jmunozar@correo.uaa.mx*
[3]*Université catholique de Louvain, Belgian Lab. of Computer-Human Interaction {vanderdonckt, gonzalez}@isys.ucl.ac.be*

Abstract

The design and development of Graphical User Interfaces for Rich Internet applications are well known difficult tasks with current tools. The designers must be aware of the computing platform, the user's characteristics (education, social background, among others) and the environment within users must interact with the application. We present a method to design theses type of User Interfaces that is model-based and applies an iterative series of XSLT transformations to translate the abstract modeled interface into a Final User Interface that is coded in a specific platform. In order to avoid the proprietary engines dependency for designing tasks. UsiXML is used to model all the levels. Several model based technologies have been proposed and in this study we review a XML-compliant User Interface Description language: XAML.

1. Introduction

The Development of Web User Interfaces remains an empirical exercise; furthermore the available tools are specialized in manual design. Today a model-based design of interactive software applications is an approach which is gaining more acceptance because the increasing number of applications to be build and the growing number of conditions that applications should fulfill e.g., Web applications are made for a wide spectrum of users from highly qualified to novice and disabled ones. That's why their development must be ruled by usability criteria and ergonomic guidelines to assure their quality. Besides that, nowadays Web applications User Interfaces are complex and their design isn't trivial. Their design requires a model [1].

These new Web applications that are emerging are called Rich Internet Applications (RIAs). RIAs are Web applications that transfer most of the load of processing the user interface to the Web client while the predominant part of data (from control and maintaining to business data) remains on the application server. A standard RIA architecture is shown in Figure 1.

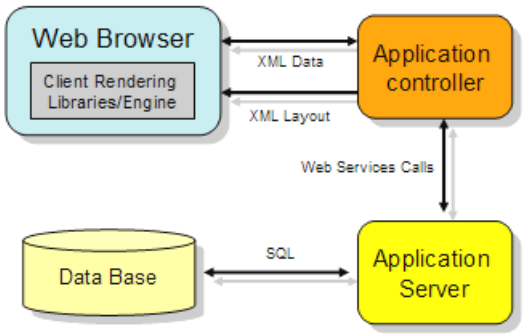

Fig. 1.Typical architecture of a RIA application.

Building a model based application requires a framework to define the design steps needed for describe our computer system, including the features: Multi-level abstraction, Modality independence, among others [2]. The Cameleon Reference framework [3] expresses these features to describe an application.

This framework structures the development process within four levels of abstraction: Task and concepts, Abstract User Interface (AUI), Concrete User Interface (CUI) and Final User Interface (FUI), as shown in Fig.

2, the arrows pointing to a lower position in a hierarchy represent reification steps (forward engineering) from abstract to a real world interface. Meanwhile, arrows pointing to upper positions reflect the process of inference abstract descriptions from the run-time code (reverse engineering) [3].

It's required a User Interface Description Language (UIDL) [4] to denote a UI at any level of abstraction. One of theses languages is UsiXML (UsiXML which stands for User Interface eXtensible Markup Language). This language incorporates the four abstraction levels of Fig. 2 as described in [5].

UsiXML describes the UI for multiple contexts e.g., Character User Interfaces, Graphical and Multimodal ones in a form that maintains design independent from specific platforms [2].

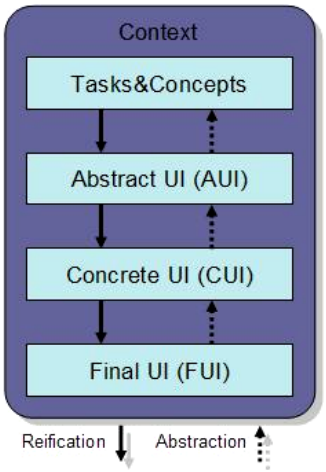

Figure 2. The Cameleon Reference framework.

2 Problem Description

In this section we are going to discuss about the problems in the design of Web User Interfaces. First, typical web applications are built from back to front leaving to final stages the development of User Interfaces, this approach is changing, especially from the new multimedia overloaded RIA User Interfaces (RIAUI).

Second, RIAUIs construction blocks include a plethora of gadgets not available in typical desktop applications so it's needed a new method to deal with these applications [15]. The "explosion" of new gadgets caused by the cognitive overload [17] product of massive information quantities those Web services like RSS [17] and popular portals (*www.yahoo.com*, *google.com*, among others) supply to users is huge, so new mechanisms to present information are been created as part of RIAIUs. Furthermore, developer teams has to face this problem after embracing a technology, if you must build (or migrate) an application but in another technology is mandatory recoding most of the project even with RIAs where as part of the core of these technologies resides a XML-compliant Description Language.

These languages have a common objective: model the User Interface, The most popular options to design and develop a RIA application (at FUI level) are: XUL[8] openlaszlo [9], Flash&Flex from MacroMedia [10] and one proposed by Microsoft: XAML [11][12].

After the literature review, we recognize the XSLT transformation schema [1, 2, 4 and 13] as one of the most promising efforts and we would like to analyze it in our study.

3 Method Outline

In this paper we proposed a novel approach to model RIAUIs which includes the complete software development life cycle besides repositories of the constitutive elements of RIAUIs that include a plethora of new gadgets not available in typical desktop applications, so a new method is required to deal with this new type of applications.

Our RIAUI development cycle is progressively refined from the Computing Independent Models (CIM) as defined by OMG [14] to the concrete models: Platform Specific models. Now we are going to describe the proposed method that is conformed by four steps (as shown in Fig. 3).

In the first step, we define the task and domain model also the relationship between them (Figure 3, step 1). This process must be accomplished by a software engineer without worrying about the graphical output of the interface.

We must define what is going to do the user as a sequence of tasks. The task & domain model are represented using the CTT diagrams [18] a popular task model representation in the field of HCI.

The second step produces a UI definition independent of any interaction modality (e.g., graphical, vocal, tactile among others) to deliver this Abstract UI definition (AUI) we have to transform the RIAUI task&domain model to an abstract model (Figure 3, step 2), the process involves an clustering process (Figure 5) and the use of IDEALXML tool [20] to produce this level UI.

In the third step, we transform AUI definitions to Concrete Individual Objects (CIOs) these are native widgets sets present in popular graphical toolkits in this important step is needed to include ergonomic criteria [19] to identify the most suitable element to be

settled in the Final User Interface (FUI).We use XSLT transformations to translate the CUI objects onto platform/language specific elements.

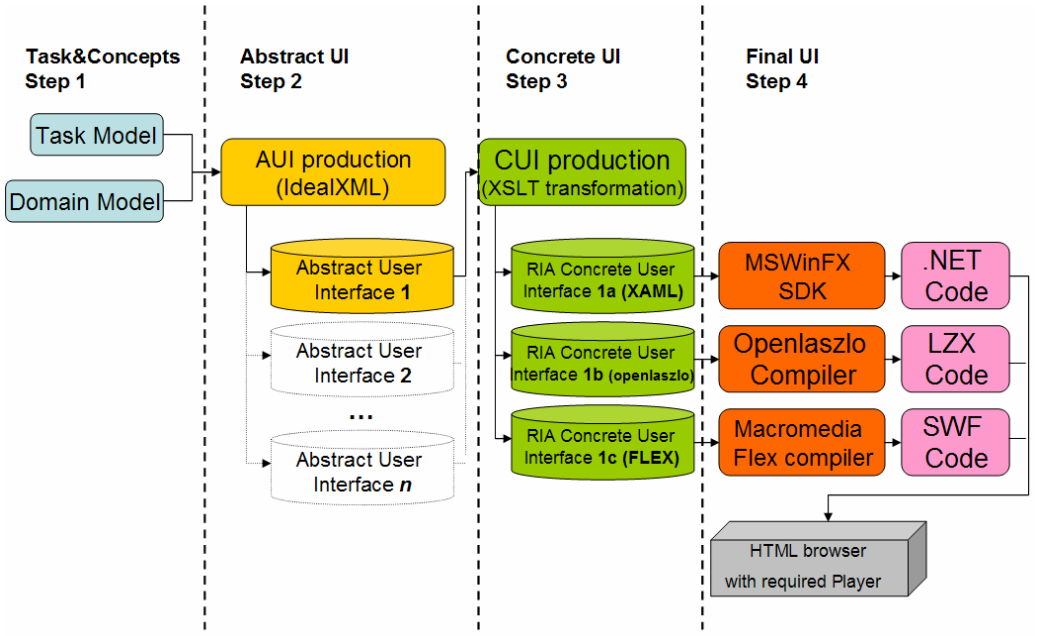

Figure3. Outline of the method for designing Graphical RIAUIs.

An important feature of the method is its capability to redirect the target FUI e.g., in the figure 3, step 3, as example we present three target transformations: XAML [11], FLEX [10] or Openlaszlo [9].

Finally, in the last step are produced operational UIs that are executed, compiled or interpreted on a particular platform (e.g., .NET, LZX, SWF among others). The code that is obtained is translated by the associated toolkit compiler (see Fig 3, step 4).

In order to support the platform independent features we are going to use UsiXML construction and settling of the rules needed to construct an XSLT specification to transform a source GUI defined in UsiXML in the Concrete User Interface level into a XAML implementation (as a working example).

The next section explains only the XLST transformation of the CUI to FUI and only the adding contact AIC for space restriction reasons.

The proposed schema results in the translation of the UI basic elements: windows, buttons, and textboxes, among others. In order to do so, Xpath expressions [7] were written to get access to the UsiXML nodes that match the XSLT rules of substitution.

This work would be the initial step for modeling RIA applications with UsiXML and second, to provide a platform independent way to design RIA applications without being attached to a proprietary solution.

The architecture of this transformation application is shown in figure 4. It's a typical XML to XML transformation schema; in Fig. 4.a the input document is specified in CUI layer of UsiXML that is a concrete version of the elements defined in the abstract level besides layout and navigation behavior [3]. In the fig. 6 is shown an excerpt of the XLS transformation proposed document.

The widgets are recognizable UI elements still not attached to a particular toolkit. That targeting to a specific toolkit is settled in the FUI (Fig. 4.d) which is the production of UI code to be compiled or interpreted, here in XAML (figure 7).

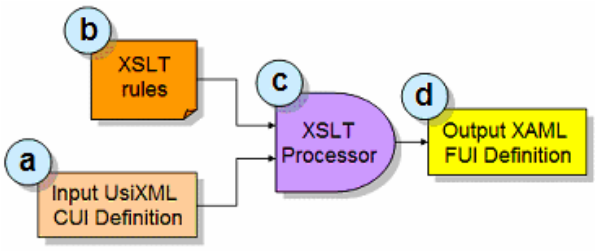

Figure 4. XLST scheme.

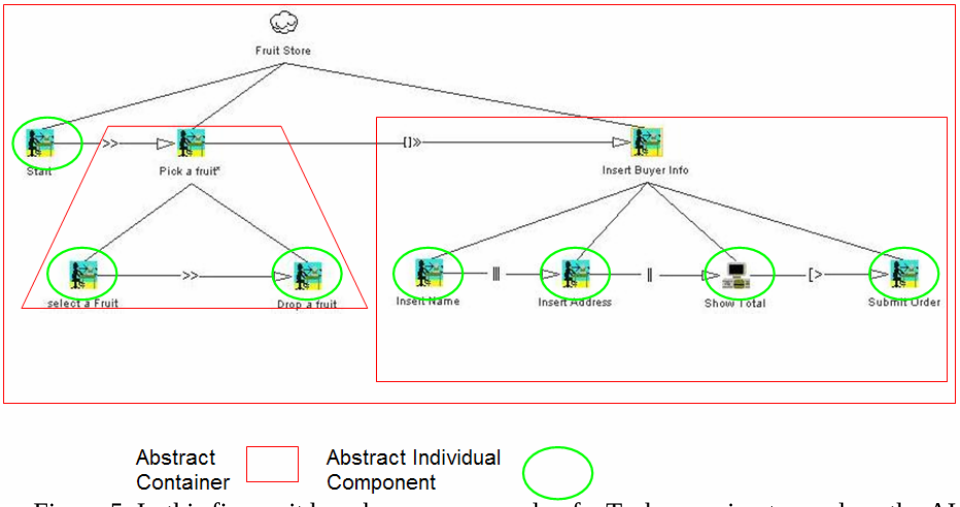

Figure 5. In this figure, it has shown an example of a Task grouping to produce the AUI.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:wf="http://schemas.microsoft.com/2003/xaml/" version="1.0">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="*|/">
   <wf:UserControl Name="WebForm1" ClientSize="200, 200"
xmlns="http://schemas.microsoft.com/2003/xaml/" xmlns:def="Definition"
xmlns:wf="wf" def:Class="XamlonApplication8.WebForm1"
def:CodeBehind="WebForm1.xaml.cs">
            <xsl:apply-templates select="/cuiModel/window"/>
            </wf:UserControl>
   </xsl:template>
      <xsl:template match="window">
      <wf:UserControl.Controls>
         <xsl:apply-templates select="/cuiModel/window/box/inputText"/>
         <xsl:apply-templates select="/cuiModel/window/box/button"/>
         <xsl:apply-templates select="/cuiModel/window/box/outputText"/>
         <xsl:apply-templates select="/cuiModel/window/box/slider"/>
      </wf:UserControl.Controls>
   </xsl:template>

   <xsl:template match="inputText">
      <wf:TextBox Text="{@defaultContent}" TabIndex="1"
Name="{@name}"/>
   </xsl:template>
   <xsl:template match="button">
      <wf:Button Text="{@defaultContent}" TabIndex="1"
...
```

Fig 6. Excerpt of the XSL transformation document.

```
<wf:UserControl xmlns:wf="wf"
xmlns="http://schemas.microsoft.com/2003/xaml/"
xmlns:def="Definition" Name="WebForm1" ClientSize="200, 200"
def:Class="XamlonApplication8.WebForm1"
def:CodeBehind="WebForm1.xaml.cs">
      <wf:UserControl.Controls
      xmlns:wf="http://schemas.microsoft.com/2003/xaml/"><wf:Te
      xtBox Text="" TabIndex="1"
      Name="input_text_component_9"/><wf:TextBox Text=""
      TabIndex="1"
      Name="input_text_component_11"/><wf:TextBox
      Text="0.00" TabIndex="1"
      Name="input_text_component_13"/><wf:Button
      Text="Submit order" TabIndex="1"
      Name="button_component_14"/><wf:Label Text="Name"
      TabIndex="1" Name="output_text_component_8"/><wf:Label
      Text="Address" TabIndex="1"
      Name="output_text_component_10"/><wf:Label Text="Total
      to Pay:" TabIndex="1" Name="output_text_component_12"/>
      </wf:UserControl.Controls>
</wf:UserControl>
```

Figure 7. An example of a XAML resulting file as produced by the XSLT.

5 Conclusions and Directions for Future Research

In this paper we have presented a novel method for designing Graphical User Interfaces of RIAs. The process can produce an automatic generated UI that follows the principles of the Model Driven Engineering. Beginning with the recompilation of the user requirements an abstract model of the UI is created and using a iterative series of XSLT transformations the model is successively translated to more concrete model to finally arrive to a Platform Specific model (PSM) which can be used in a browser or treated by a appropriate platform compiler in order to deliver a solution specific code.

Besides that, our approach automates the translation of UIs defined in UsiXML to XAML documents with XSL transformations in our preliminary evaluation we have had satisfactory results.

Right now we are compiling a repository of all the UI gadgets (components) defined in XAML for complete the XSLT translation sheet in a java implemented prototype called RIAXML but in this early stage of development, some interesting features are still non included e.g., how to deal with alternatives in the widgets selection and define the most suitable.

Acknowledgments

We would like to thank the reviewers for their pertinent comments, The University of Zacatecas (UAZ), The University of Aguascalientes (UAA) and the BHCI lab members (especially to Josefina Guerrero, Juan Gonzalez and Adrian Stanciulescu) at the UCL-IAG department for their support in our scientific stage.

References

[1] Miller, J., and Mukerji, J., MDA Guide Version 1.0.1, 2003, Object Management Group, Inc.

[2] Bouillon, L., Limbourg, Q., Vanderdonckt, J., Michotte, B., Reverse Engineering of Web Pages based on Derivations and Transformations, Proc. of 3rd Latin American Web Congress LA-Web'2005 (Buenos Aires, October 31-November 2, 2005), IEEE Computer Society Press, Los Alamitos, 2005, pp. 3-13.

[3] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon, and J. Vanderdonckt, A Unifying Reference Framework for Multi-Target User Interfaces, Interacting with Comp., Vol. 15, No. 3, June 2003, pp. 289-308.

[4] Luyten, K., Coninx, K, and Abrams, M., Integrating UIML, Task and Dialogs with Layout Patterns for Multi-Device User Interface Design. The 11th International Conference on Human-Computer Interaction, Las Vegas, Nevada, USA, July 22-27, 2005.

[5] Limbourg, Q., Vanderdonckt, J., Michotte, B., Bouillon, L., and Lopez, V. UsiXML: a Language Supporting Multi-Path Development of User Interfaces, Proc. of 9th IFIP Working Conference on Engineering for Human-Computer Interaction jointly with 11th Int. Workshop on Design, Specification, and Verification of Interactive Systems EHCI-DSVIS'2004 (Hamburg, July 11-13, 2004). Lecture Notes in Computer Science, Vol. 3425, Springer-Verlag, Berlin, 2005, pp. 207-228.

[6] O'Rourke, C., A Look at Rich Internet Applications. Oracle Magazine. July - August 2004.

[7] Berglund, A., Boag, S., Chamberlin, D., Fernández, M. F., Kay, M., Robie J. and J. Siméon, XML Path Language (XPath) 2.0. http://www.w3.org/TR/2005/WD-xpath20-20050404/#id-references, (W3C, March 15th, 2006).

[8] XML User Interface Language (XUL) 1.0, http://www.mozilla.org/projects/xul/xul.html (Mozilla Foundation, March 20th , 2006).

[9] Openlaszlo, http://www.openlaszlo.org/ (Laszlo Systems, Inc., March 20th , 2006).

[10] FLEX. http://www.macromedia.com/software/flex/ (Adobe Systems Incorporated, March 20th ,2006).

[11]XAML. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnintlong/ html/ longhornch01.asp (Microsoft, March 20th, 2006).

[12] WinFX SDK. http://msdn.microsoft.com/winfx/ (Microsoft, March 10th, 2006).

[13] Gerber, A., Lawley, M., Raymond, K., Steel, J., and Wood, A., "Transformation: The Missing Link of MDA".

[14] OMG, http://www.omg.org , March 10th, 2006

[15] Preciado, J.C., Linaje, M., Sanchez, F., Comai, S. Necessity of methodologies to model Rich Internet Applications. Seventh IEEE International Symposium on Web Site Evolution. pp. 7-13, 2005.

[16] Quiroga, Luz M., Crosby,Martha E., Iding, Marie K. Reducing Cognitive Load. Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04), 2004.

[17] Zhou Ying ,Chen Xin ,Wang Chen. A Self-Organizing Search Engine for RSS Syndicated Web Contents. 22nd International Conference on Data Engineering Workshops (ICDEW'06), 2006.

[18] Paternò, Fabio.Towards a UML for Interactive Systems Engineering for Human-Computer Interaction: 8th IFIP International Conference, EHCI 2001, Canada.

[19] Vanderdonckt, Jean. Règles ergonomiques de sélection d'objets interactifs pour une information simple, Actes du 6ème Colloque Ergonomie et Informatique Avancée (Biarritz, 4-6 novembre 1998), M.-F. Barthet (éd.), ESTIA/ILS, Bidart, 1998, pp. 250–259.

[20] GrafiXML. http://www.usixml.org/index.php?view=page&idpage=10 (Home of the USer Interface eXtensible Markup Language, March 20th ,2006).