Realization of XML as an essential supporting tool for the Object Oriented System Design

by A.U.Umagiliya and J.D.S.S.Gunathunge

1. Introduction

After the introduction of XML, it was rapidly spread to many areas and is widely used in many applications. UML also plays a major role in modern software development industry. It facilitates analysis and modeling of software systems UML class diagram can be identified as the core of the OOSD, because it identifies all the required functional units, their properties and behaviors and relationships among them. In practice, relational database schema is developed based on UML class design by applying specific Object to Relational Mapping techniques.

In distributed computing, XML is used as a communication medium to achieve language and platform independency, but still there is considerable dependency between database server and the software component. Also in distributed computing a presentation layer or GUI layer is composed as one unit including appearance of user interfaces, data validation and event handling. For example, when someone click a button, the corresponding action to be taken (e.g. what business method should be triggered, what data should be provided to that method, how to capture those data, how to validate those data) are tightly bound to that 'button' component and its container.

As the modern business world is focused on custom satisfaction, the following observations can be seen,

- Some systems need improvements of its user interfaces or implement entirely new interfaces without introducing modifications to the business layer or any other layer with effective manner.
- Some systems need to keep their existing interfaces, while introducing modifications to other layers.

For Achieve above goals significant amount of resources such as cost and time is needed. Therefore, it is important to develop tools to assist developers in resolving those problems.

The rest of this paper is structures as follows. Section 2 describes the related work. Our methodology is presented in Section 3.And Section 4 describes our case study and experimental results.

3. Methodology

Each class in the class diagram can be categorized into several stereotypes according to their role. These stereotypes are listed below:

1. boundary
2. web component
3. entity
4. other (roles that are not significant to our methodology)

Despite of the stereotype, all the information in every class is mapped into an intermediate XML Meta model so that it can be used for source code generation. Identifying of ‘entity’ classes is a crucial task, because database schema and other database related features are generated based on these classes.

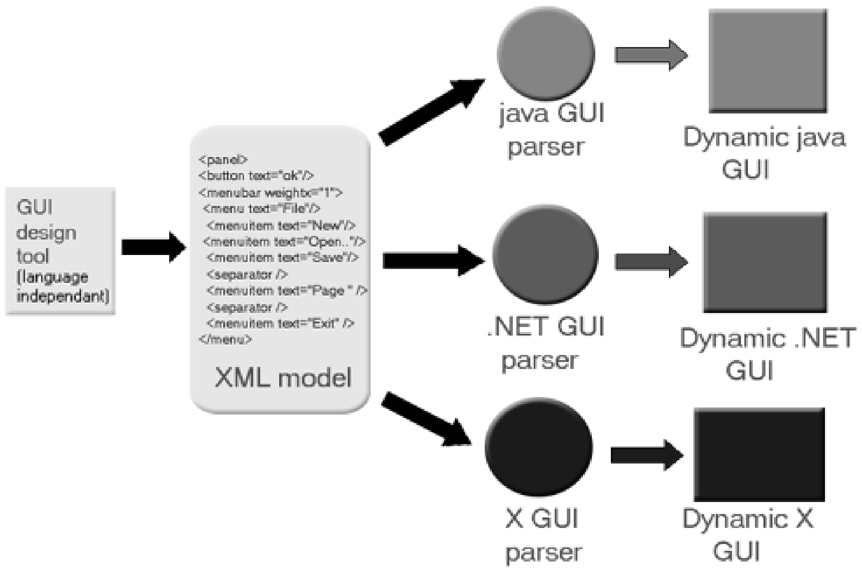
3.1 Language independent GUI design based on UML class model.

All the classes with stereotype 'boundary' are capable to have a GUI component. In this case, users are allowed to create GUI design for that specific class. They can use visual design tool to draw layouts, and add standard GUI component such as text boxes, lists, radio buttons, check boxes, menus, sliders according to their requirements. Further users can add the event handling codes. As an example, for a particular button, users can manage its layout, its properties and events, such as click, double click, mouse over and so forth. Still these activities are common for most of the available GUI design tools. One of the major differences of the proposed approach is that GUI model is converted into XML model instead of converting into a programming language code.

Most of the languages provide class libraries for GUI components, such as *javax.swing* for java programming language and *System.Windows.Forms* for .net platform.

It is possible to develop a XML based GUI parser for such programming language that takes above XML specification as an input and dynamically produce a GUI according to that language.

At run time, the GUI parser loads the appropriate GUI components from the specific GUI library of the given programming language.(e.g. java- javax.swing) The XML specification is used to decide which GUI component should be loaded and which layouts should be used. Further parser applies the properties and binds the associated events specified in the XML file (this procedure is similar to the java virtual machine that take the java byte code and run the program). Figure 3.2 illustrates this concept.



```
<panel>
<button text="ok"/>
<menubar weightx="1">
 <menu text="File"/>
  <menuitem text="New"/>
 <menuitem text="Open.."/>
  <menuitem text="Save"/>
  <separator />
  <menuitem text="Page " />
  <separator />
  <menuitem text="Exit" />
</menu>
```

Figure 3.2

If modifications are need to be done for the GUI, all the modifications are applied to the XML specification without dealing with any programming language code or other layers of the model.

Also, the same GUI specification can be reused with parsers which written in different programming languages. A GUI parser for HTML can also be developed that act as in previous scenario. But produce HTML and JavaScript codes as output. In this case particular boundary class should be run on a web server. For example as a JSP page as a servelet on tomcat server.

3.2 XML based stored procedures for all database access functionalities

The classes with ‘entity’ stereotype are mapped in to tables of database. In most cases, primary key for table can be determined using attribute provided in that particular ‘entity’ class. In some cases, relations among ‘entity’ classes are required to determine primary keys. Foreign keys are determined according to the relationship among classes and their multiplicity values.

In the proposed approach, Object to Relation Mapping is done through the intermediate XML model. First all the entity classes are transformed into a XML model. This XML model consists of tables, relationships, primary keys, foreign keys and so forth.

Additionally, it is easy to generate ANSI SQL schema or any vendor specific database schema based on this intermediate XML model.

It is possible to develop a set of XML based stored procedures in such away that the communication between database and software components is done through XML data. Tha nly using
above set of st

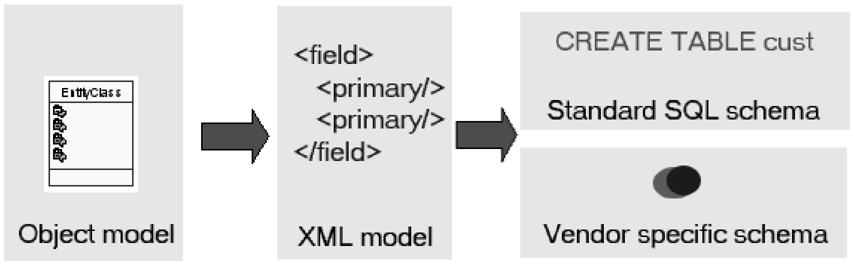


Figure 3.3

Based on the following study it is possible to reduce the number of required stored procedures for database activities significantly.

Basically most frequently used database access activities can be grouped into one of the following groups.

1. Update – update, delete, update operations

2. query – various SELECT queries

It is possible to use one stored procedure for all the update operations. According to the format of XML data, the stored procedure determines what should be the correct operation to perform (insert, deletion or update).

bookUpdate(IN xmlDocument)

As the "IN" parameter an XML document should be passed to this stored procedure. According to the "IN" parameter "xmlDocument", stored procedure decides what actions should be taken. The possible actions are, add new book to database, delete existing book from database or updates properties of existing book . "IN" parameter follows a similar XML structure as in figure 3.4

```
<insert>
   <book_id>…</book_id>
   <book_name>..</book_name>
   <author>…</author>
    .
    .
</insert>
```

In this case the book_id , book_name , author represents the name of the fields and their values to be inserted as a record.

When deleting a record from the database the XML has structure as in figure 3.5

```
<delete>
   <book_id>isbn-120</book_id>.
</delete>
```

In this case the all records with value 'isbn-120' in field book_id , are deleted. If the XML file contains several elements, records having given the field value are deleted.

When updating a record in the database XML get the structure as in Figure 3.6.

```
<update>
   <book_id>…</book_id>
   <book_name>..</book_name>
   <author>…</author>
    .
   .
  <update_key_value name="author" >Tennanbun.</update_key_value>
</update>
```

In this case book_id , book_name , author fields are updated in the records which have the author value ‘Tennanbun’.The ‘update_key_value’ is used to determine the which record to be updated. The ‘name’ attribute of the ‘update_key_value’ provide the specific field and the value of ‘update_key_value’ provide the value for select the records. In this example all the records which have the author field value ‘Tennanbun’, should be updated.

In querying it is needed to develop a separate stored procedure for simplicity. The most important part of SELECT statement is WHERE clause. Because it defines criteria for the selection, using following XML specification, a select query can be represented successfully.

bookQuery(IN xmlDocument)

The XML document takes structure showed in figure 3.7

```
<select>
  <book_name>Java</book_name>
 <author>Sun.</author>
</select>
```

Figure 3.7

The WHERE clause is constructed based on the elements and their values. In this example book_name author, construct “WHERE book_name=’Java’ AND author=’Sun’”.

After executing the query, the result set is converted into XML form and returns.

4. Case study and experimental results.

A RAD tool was developed to test the success of proposed method. This tool mainly consists of two diagramming tools. One for design UML class diagrams. Also it facilitates to add properties and behaviors. Other design tool facilitates the design of GUI interfaces according to the proposed methodology.

Once the user completes the class model, he/she can move to the database schema generation. The RAD tool facilitates to develop ANSI SQL schema for the database. Further it constructs the database and deploys set of XML stored procedures on database server.

Like other UML tools, this also provides source code generation facilities for many languages. Instead, this generates database utility classes to access above constructed stored procedures. Since developer no need much concern about SQL related things in the object model. Also this generate boundary classes that activate the GUI that accommodate the methods which accessed by the particular GUI (basically event handling methods)

XML based GUI parsers for java and .NET were developed. These parsers provide the correct results according to the XML specification.

Conclusion

The proposed language independent GUI generation method can address today's business oriented, GUI related problems in an effective manner. It is possible to develop GUI parsers for many languages.

XML based stored procedures provide higher degree of independency between database server and the software components. Further it supports existing of heterogeneous platforms and heterogeneous servers.

References

1. Bray, T., Paoli, J., and Sperberg-McQueen, C. M., Extensible Markup Language (XML) 1.0, W3C Recommendation, www.w3.org/TR/1998/REC-xml-19980210. (12 March 2006)
2. Cayenne project, Cayenne documentation. http://objectstyle.org/cayenne/index.html
3. Giles, R. D., Graphical Interface Development with Glade2 ,
4. http://www.kplug.org/glade_tutorial/glade2_tutorial/glade2_introduction.html
5. Wood, L., Document Object Model (DOM) Level 1 Specification, W3C Recommendation, www.w3.org/TR/REC-DOM-Level-1/.
6. Sun Microsystems, The JFC Swing Tutorial. http://www.sun.com/tutorial/books

7. Microsoft Press, Developing Windows-Based Applications with Microsoft Visual Basic .NET and Microsoft Visual C# .NET. Microsoft Press
8. Object Management Group(OMG) , UML Refernce Manual ,(http://www.uml.org)