Research and Implementation of J2EE Web Application Code Automatic Generation System Based on UML Model

Maochun Zheng

Computer Science and Technology Department, Ocean University of China,

Qingdao 266000, China

zmc1125@foxmail.com

Abstract. In recent years, the demand of data management systems for various businesses is growing, resulting in an increasingly large system size and amount of codes which costs more time and manpower. Therefore, in order to improve the efficiency of software development, the development of an automatic code generator is imperative. Through study of previous code generating theories, it is obvious that a few issues such as "automatic generation", "operation visualization", "and multi-tables association" remain unresolved. Firstly, through the research of MDA theory, this paper puts forward an automatic code generating project based on the combination of UML data modeling, HTML5 visual controls, templates file definitions and template engines and finally validate the use and functionality of the code generator through a specific test.

Keywords: Automatic code generation, UML, Visualization, Customizable templates, Metadata.

1. Introduction

In today's world with highly developed software engineering, most software developers still use manual coding. The development cycle of this traditional software development methods are longer, and there is a lot of duplication of effort. In this case, issues such as delay of large software projects, cost overruns plans and performance below expectations usually appear. These issues compel software developers to think how to reduce software development and maintenance costs and improve development efficiency [1].

Then in the traditional development approaches, these problems are always difficult to solve perfectly. This requires a new way of software developing that can avoid low level duplication of development and achieve standardized production of soft wares [2], in order to help businesses cope with the rapidly changing market environment. Automatic code generation technology solves this problem to a certain degree.

The authors of this paper run an in-depth research on code generation technology used in the current software development technology put forward a solution making use of visual UML modeling for automatic code generation and achieve an automatic code generator based on combination of Free marker, Spring and Hibernate.

2. Research on status of Automatic code generation technology

In recent years, domestic and foreign researchers have proposed a number of ideas and solutions on the research and implementation of automatic code generation technology [3]. Currently, the common solutions of code generation programs are based on metadata-driven, design patterns [4], UML models, XML and templates. The backgrounds these programs get involved with vary, but they all have a certain academic or practical value as well as a certain influence on the development of automatic code generation technology.

Among which, code generation scheme based on the UML models can generate directly multilayer system, when generating codes, and retain the relationship embodied in the model hierarchy. However, the program uses XML for the description of modeling information, resulting that business model is not easy to manage and therefore operation cannot be visualized. Template-based code generation scheme has characteristics such as relatively easy to achieve, code templates that are easy to maintain and flexibility and strong and easy to follow expansion [5]. But it lacks corresponding template rules when handling multiple tables.

3. Solutions of status of Automatic code generation technology

Automatic code generation techniques presented in this paper is based on UML visual modeling and template techniques. The overall flow of the system is shown in Figure 1.

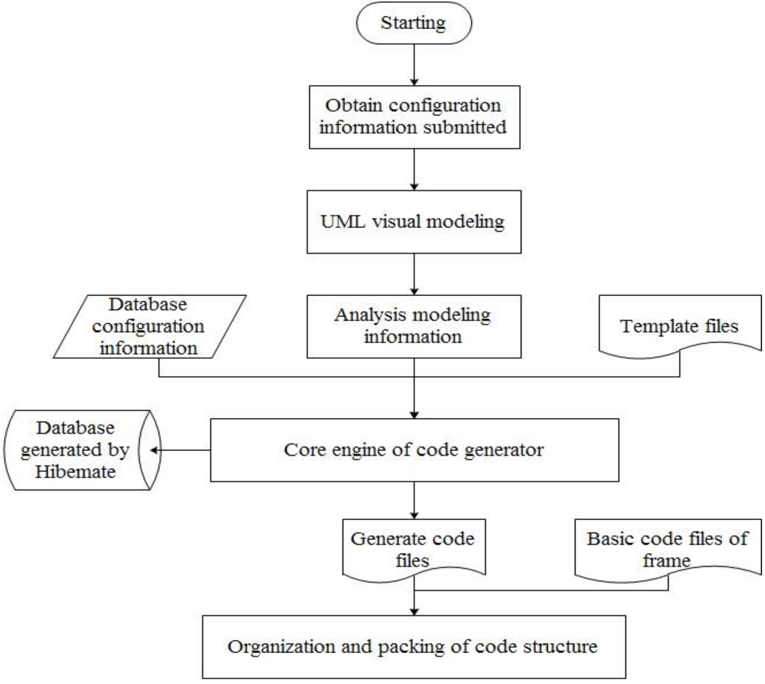

Starting
Obtain configuration information submitted
UML visual modeling
Database configuration information
Analysis modeling information
Template files
Database generated by Hibemate
Core engine of code generator
Generate code files
Basic code files of frame
Organization and packing of code structure

Figure 1. The design process of code generator

Users abstract universal framework of all template layers according to the maturity framework and input these templates into the code generator. Users will conduct model design of the associated business according to the actual needs of projects. Backend code generators generate objects which are used to encapsulate metadata based on user-designed business models. Handler will combine the imported data objects with templates and use Free marker template engine to generate project code, database files generated by Hibernate. After code generation, prompts the user to download the code, the user-generated code and mature business integration framework underlying code, while input file into the target database to complete the initialization project system.

The current code generation technology can only deal with ordinary adding, deleting, modifying, and searching business logics. This program focuses on solving business process of relationship between data tables, developing relevant template rules and improves the efficiency and formativeness of the code written by automatically generating code.

4. Key technologies of Automatic code generation technology

4.1 Business models design.

This paper proposes a set of mechanism for business models based on database design by studying UMl data modeling as well as combining with the design principle of database. This mechanism is divided into three levels (as shown in figure 2): Project, Dyna Class and Dyna Attribute. Project refers to basic environmental information that is required for the project to establish the basic objectives. After the establishment of the project information, UML modeling on the project starts which include dynamic class and static class data modeling. Business objects are described through these attributes, and the generator combines with logic modules based on these attributes to generate a database table file and specific business handling code.

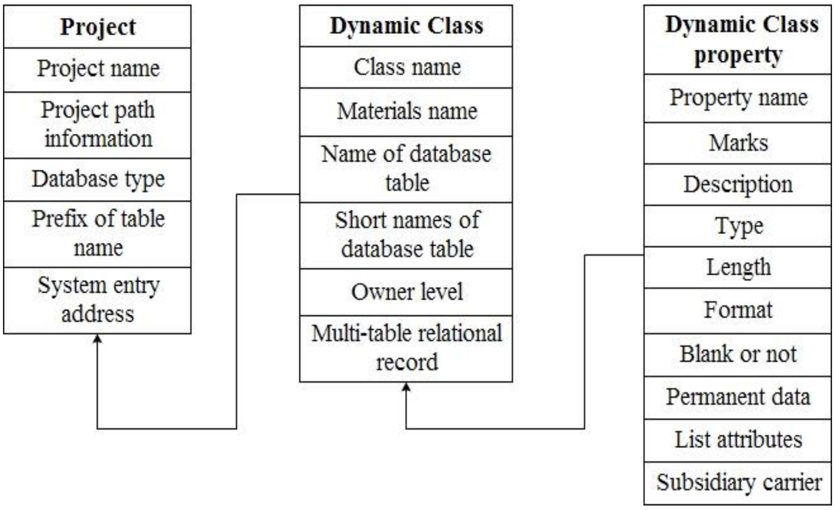


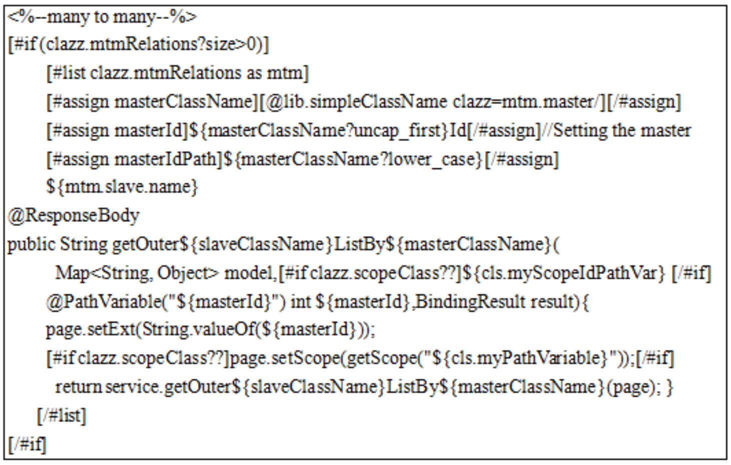
Figure 2. Business model mechanism designs

4.2 Business model relationships treatment.

In the process of code generating, the association between business objects costs more effort, but currently most automatic code generators cannot handle multi-table associated, so this paper proposes a multi-table associated with handling mechanism.

Multi-table association refers to one to one, one to many, many to one and many to many. By studying the multi-table association database as well as query methods, many too many and one to one relationships will be expressed using the combination of two-way association mapping. In this builder, set mtmRelations in List <MT Relation> type for Dyna class to mark bidirectional association and achieve association mapping. Among which, MT Relation’s set properties: Master class, master class properties, controlled class, controlled class attributes, and associated logo (one to one or many to many). For one to many and many to one relationship, means to control transfer and mapping the column are used to achieve unidirectional association mapping. Dynamic class (Dyna Class) set of property will be Scope Class, which is the controlling side of one to many relationships. Class attribute (Dyna Attribute) sets the Boolean type attribute is Ref Class indicate whether it is a list of many, indicates that the attribute points to a specific dynamic class, set the type of property Dyna Class ref Class, indicate the specific dynamic class points.

Free marker template technique is simple with lightweight and excellent performance [6]. Therefore this paper adapts free marker as the template language. Template example will be shown in figure 3, which shows that, code generation templates include fixed code and variable code. $ {} is the interpolation part of free marker, thus completing the combination of data model and template files [7] by the definition of interpolation. In order to deal with the characteristics of the business models, the template takes full advantage of the logic syntax and operators built-in free marker, thereby easily generate personalized configuration code in accordance with requirements.

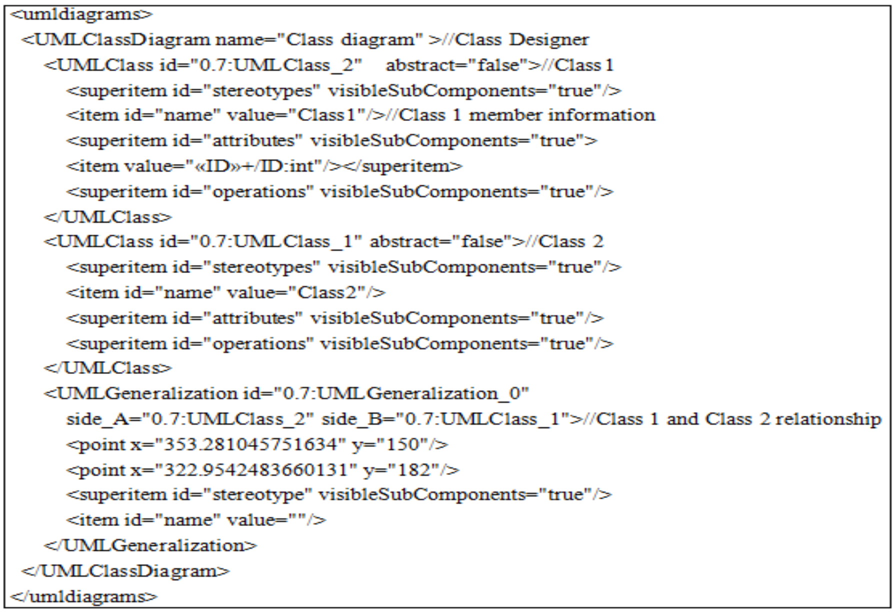


```
<%--many to many--%>
[#if(clazz.mtmRelations?size>0)]
    [#list clazz.mtmRelations as mtm]
    [#assign masterClassName][@lib.simpleClassName clazz=mtm.master/][/#assign]
    [#assign masterId]${masterClassName?uncap_first}Id[/#assign]//Setting the master
    [#assign masterIdPath]${masterClassName?lower_case}[/#assign]
    ${mtm.slave.name}
@ResponseBody
public String getOuter${slaveClassName}ListBy${masterClassName}(
     Map<String, Object> model,[#if clazz.scopeClass??]${cls.myScopeIdPathVar} [/#if]
    @PathVariable("${masterId}") int ${masterId},BindingResult result){
    page.setExt(String.valueOf(${masterId}));
    [#if clazz.scopeClass??]page.setScope(getScope("${cls.myPathVariable}"));[/#if]
     return service.getOuter${slaveClassName}ListBy${masterClassName}(page); }
   [/#list]
[/#if]
```

Figure 3. Sample code of template files

4.3 UML visual modeling technology.

JsUML2 library is a HTML5 / JavaScript database that supports UML2 diagrams and provides Web. Developers can easily visually edit UML models in their own websites. By studying JsMUL2 modeling database, according to JsMUL2 rule, this paper use xml (Figure 4) as the interaction channel between visual model and database model, thus completing the combination between it and the business model design mechanisms. Users can switch the view, not only to edit on the surface of the UML modeling the database, which are free conversions between UML modeling and database design interfacial pages.

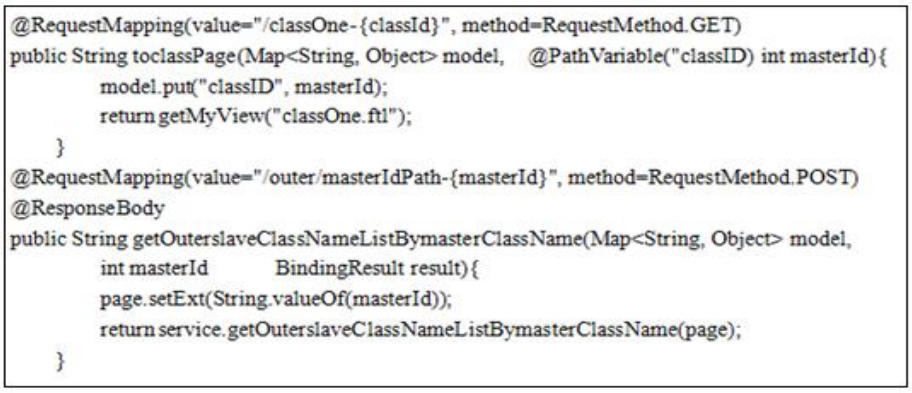


```
<umldiagrams>
 <UMLClassDiagram name="Class diagram" >//Class Designer
   <UMLClass id="0.7:UMLClass_2"    abstract="false">//Class 1
     <superitem id="stereotypes" visibleSubComponents="true"/>
     <item id="name" value="Class1"/>//Class 1 member information
     <superitem id="attributes" visibleSubComponents="true">
     <item value="«ID»+/ID:int"/></superitem>
     <superitem id="operations" visibleSubComponents="true"/>
   </UMLClass>
   <UMLClass id="0.7:UMLClass_1" abstract="false">//Class 2
     <superitem id="stereotypes" visibleSubComponents="true"/>
     <item id="name" value="Class2"/>
     <superitem id="attributes" visibleSubComponents="true"/>
     <superitem id="operations" visibleSubComponents="true"/>
   </UMLClass>
   <UMLGeneralization id="0.7:UMLGeneralization_0"
     side_A="0.7:UMLClass_2" side_B="0.7:UMLClass_1">//Class 1 and Class 2 relationship
     <point x="353.281045751634" y="150"/>
     <point x="322.9542483660131" y="182"/>
     <superitem id="stereotype" visibleSubComponents="true"/>
     <item id="name" value=""/>
   </UMLGeneralization>
 </UMLClassDiagram>
</umldiagrams>
```

Figure 4. Sample codes in UML view of the XML schema

5. Experiment results

In a running program, input the above XML examples and templates file as import, and the test results obtained are directly generated codes related to templates as output, as shown in figure 5. Watching from the output results, there are relevant properties and related functions, the system achieve the purpose of the original design from the aspect of functions.



```
@RequestMapping(value="/classOne-{classId}", method=RequestMethod.GET)
public String toclassPage(Map<String, Object> model,    @PathVariable("classID) int masterId){
            model.put("classID", masterId);
            return getMyView("classOne.ftl");
      }
@RequestMapping(value="/outer/masterIdPath-{masterId}", method=RequestMethod.POST)
@ResponseBody
public String getOuterslaveClassNameListBymasterClassName(Map<String, Object> model,
            int masterId          BindingResult result){
            page.setExt(String.valueOf(masterId));
            return service.getOuterslaveClassNameListBymasterClassName(page);
      }
```

Figure 5. Samples of code generation

6. Conclusion

This paper presents a solution to realize Java EE code generator which function relatively well, which not only proposes solutions for multi-table related issues, but also realizes UML-based visual design. Of course, there is a large gap between the systems generated by code generator in this paper away from mature enterprise-class applications, such as the lack of IDE's support for code level, the generated code cannot be directly imported by Eclipse and other IDE. On the basis of this paper, IDE-based projects and integration of automatically codes deploy, you can achieve a more comprehensive information system software solutions for automatic code generation.

References

[1] Dong Yuming. Research and Application of Code Generation Technology in Development of Management Information System [D]. Jilin University: Jilin University, 2012.

[2] Anitha Rani Marneni. Automatic generation of object-oriented class implementations from behavioral specifications [D].The University of Texas at San Antonio, 2010.

[3] Abid Mehmood, Dayang N.A. Jawawi. Aspect-oriented model-driven code generation: A systematic mapping study [J]. Information and Software Technology, 2013, 552:

[4] Hamad I. Alsawalqah, Sungwon Kang, Jihyun Lee.A method to optimize the scope of a software product platform based on end-user features[J].The Journal of Systems & Software, 2014, 98.

[5] Guan Taiyang. Research on automatic code generation based on templates [D]. University of Electronic Science and Technology, 2007.

[6] Yang Xingtao, Su Guiping, Wang Ruifang and Wang Xiaofang. Research and implementation of domain-specific modeling and code generation [J]. Computer System, 2009, 04:100-103.

[7] John Grundy, John Hosking. Guest editor’s introduction: special issue on innovative automated software engineering tools [J]. Automated Software Engineering, 2013, 20(2).