


Tai-hoon Kim  
Hojjat Adeli  
Rosslin John Robles  
Maricel Balitanas (Eds.)

Communications in Computer and Information Science

199

# Advanced Communication and Networking

Third International Conference, ACN 2011  
Brno, Czech Republic, August 2011  
Proceedings

 Springer

Integrated Retrieval System for Rehabilitation Medical Equipment in Distributed DB Environments .....	209
<i>BokHee Jung, ChangKeun Lee, and SoonGohn Kim</i>	
Effective Method Tailoring in Construction of Medical Information System .....	215
<i>WonYoung Choi and SoonGohn Kim</i>	
A Study on the Access Control Module of Linux Secure Operating System .....	223
<i>JinSeok Park and SoonGohn Kim</i>	
An fMRI Study of Reading Different Word Form .....	229
<i>Hyo Woon Yoon and Ji-Hyang Lim</i>	
Intelligent Feature Selection by Bacterial Foraging Algorithm and Information Theory .....	238
<i>Jae Hoon Cho and Dong Hwa Kim</i>	
The Intelligent Video and Audio Recognition Black-Box System of the Elevator for the Disaster and Crime Prevention .....	245
<i>Woon-Yong Kim, Seok-Gyu Park, and Moon-Cheol Lim</i>	
Real-Time Intelligent Home Network Control System .....	253
<i>Yong-Soo Kim</i>	
LCN : Largest Common Neighbor Nodes Based Routing for Delay and Disruption Tolerant Mobile Networks .....	261
<i>Doo-Ok Seo, Gwang-Hyun Kim, and Dong-Ho Lee</i>	
A Perspective of Domestic Appstores Compared with Global Appstores .....	271
<i>Byungkook Jeon</i>	
A Design of Retrieval System for Presentation Documents Using Content-Based Image Retrieval .....	278
<i>Hongro Lee, Kwangnam Choi, Ki-Seok Choi, and Jae-Soo Kim</i>	
Data Quality Management Based on Data Profiling in E-Government Environments .....	286
<i>Youn-Gyou Kook, Joon Lee, Min-Woo Park, Ki-Seok Choi, Jae-Soo Kim, and Soung-Soo Shin</i>	
Design of Code Template for Automatic Code Generation of Heterogeneous Smartphone Application .....	292
<i>Woo Yeol Kim, Hyun Seung Son, and Robert Young Chul Kim</i>	
A Study on Test Case Generation Based on State Diagram in Modeling and Simulation Environment .....	298
<i>Woo Yeol Kim, Hyun Seung Son, and Robert Young Chul Kim</i>	



# Design of Code Template for Automatic Code Generation of Heterogeneous Smartphone Application

Woo Yeol Kim, Hyun Seung Son, and Robert Young Chul Kim

Dept. of CIC(Computer and Information Communication), Hongik University,  
Jochiwon, 339-701, Korea  
{john,son,bob}@selab.hongik.ac.kr

**Abstract.** Development of heterogeneous application becomes an issue recently as of the diversification of smartphone platform. It is possible to develop heterogeneous smartphone application when using the e-MDD (Embedded Model Driven Development). While e-MDD is divided into Model-to-Model and Model-to-Text transformations, we have been converting target independent model to the target dependent model using the Model-to-Model transformation in the past. This paper is regarding the creation of code for target dependent model mentioned in the above using the Model-to-Text transformation. Template-based approach has been used for the creation of code. Code template has been made for Windows Mobil and Android as an example of application, and the code was created using Aceleo.

**Keywords:** e-MDD(Embedded Model Driven Development), Code Template, Heterogeneous Development, Smartphone, Automatic Code Generation.

## 1 Introduction

The conventional platform type for smartphone was so limited. The platform has been diversified as the issue for smartphone is getting bigger recently. Typical examples are iPhone SDK [1] of Apple, Android SDK of Google [2], and Windows Phone7 SDK of Windows [3]. In addition, there are various kinds of smartphone platform [4]. As the platform types are diversified like this, the reuse of software has become a bigger issue. It is because there is a problem of developing the same smartphone application programs newly to apply to another platform.

It is possible to develop heterogeneous smartphone application when using e-MDD (Embedded Model Driven Development) [5]. e-MDD is divided into Model-to-Model transformation and Model-to-text transformation. For developing heterogeneous smartphone application in the past, we used Model-to-Model transformation to convert the target independent model to target dependent model [6,7,8]. This paper is

---

\* This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2011-0004203) and the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation.

regarding the code creation of previously composed target dependent model using the Model-to-Text transformation. There is a standard method of Model-to-Text transformation to OMG (Object Management Group) for the transformation from model to text (code). Model-to-Text provides the transformation language to convert the model, composed based on MOF (Meta Object Facility) [9], to the text. Template-based approach is used as the transformation language. It is used mainly for creating the code, which is put entirely in the hands of the user. This method requires composing template for the code creation. As an application example, code template has been composed for Windows Mobile and Android respectively, and the code was created with Acceleo. It has been possible to create 30% and 35% of the code for respective model as a result.

This paper is organized as follow. e-MDD is explained in Chapter 2 as a related work. Chapter 3 explains the template for the code generation. Chapter 4 shows the case study. Finally in Chapter 5, it mentions about the conclusion and future research.

## 2 Related Work

Model-to-Text approach is divided into visitor-based and template-based [10]. The very basic code creation method of visitor-based is the visitor mechanism consisted with the circulation of model's internal expression and the text writing on the text stream. Its example is Jamda, an object-oriented framework, providing the set of class that represents the UML model [11]. Jamda creates the API for handling the model and the code with the visitor mechanism (CodeWriters). Jamda does not support the MOF standard for the definition of new meta model, however, the element type of new model can execute the sub-classing of existing Java class.

Template based model text is being supported by most of MDD tools nowadays. Template is usually constituted with the object text including the meta-code splices that can execute the selection of code that is accessing the information in the source and its repeated expansion. LHS is used the execution logic that access the source, and RHS is the executable logic for the code selection and repeated expansion which is the combination of string patterns. Also, there is a definite separation of grammar in between the LHS and RHS. The example of its method is Acceleo provided by eclipse [12].

## 3 Design of Code Template

Code template describes the code type for each language, brings in the data from UML meta model, and creates the source code. In order to define the code template, it is necessary to compare and analyze the programming languages. The code template of this paper has been composed based on the class diagram. It has been divided into the class name, attribute, and method for each code, therefore, and the relations with the class have been analyzed. In the comparison and analysis result of the code, there are lots of similar parts with Java and C#. In order to compose and execute the code template, this paper has used Acceleo [12] based on eclipse that is complying with the Model-to-Text transformation standard for OMG.



<pre>[module generateCS ('http://www.eclipse.org/uml2/2.1.0/UML')/] [template public generate(c : Class)] [file (c.name.concat('.cs'), false)] public [if (c.isAbstract)] abstract[/if] class [c.name.toUpperFirst()] [for (superC : Class   c.superClass) before(' : ') separator(',')] [superC.name/][/for] [for (interf : Interface   c.getImplementedInterfaces()) before(':') separator(',')] [interf.name/][/for] { //association private : [for (p : Property   c.getAssociations().memberEnd] [p.type.name/] [p.name/]; [/for] public : [for (p : Property   c.getAssociations().memberEnd] void set[p.name.toUpperFirst()]([p.type.name/] in); [/for] //attribute private : [for (p : Property   c.attribute)] [p.type.name/] [p.name/]; [/for] public : [for (p : Property   c.attribute)] [p.type.name/] get[p.name.toUpperFirst()]() { return this.[p.name/]; } [/for] public : [for (o : Operation   c.ownedOperation)] [o.type.name/] [o.name/]() { // TODO should be implemented } [/for] } [/file] [/template]</pre>	<pre>[module generate ('http://www.eclipse.org/uml2/2.1.0/UML')/] [template public generate(c : Class)] [file (c.name.concat('.java'), false)] public [if (c.isAbstract)] abstract[/if] class [c.name.toUpperFirst()] [for (superC : Class   c.superClass) before(' extends ') separator(',')] [superC.name/][/for] [for (interf : Interface   c.getImplementedInterfaces()) before(' implements ') separator(',')] [interf.name/][/for] { //association [for (p : Property   c.getAssociations().memberEnd] private [p.type.name/] [p.name/]; [/for] [for (p : Property   c.getAssociations().memberEnd] public void set[p.name.toUpperFirst()] ([p.type.name/] in); [/for] //attribute [for (p : Property   c.attribute)] private [p.type.name/] [p.name/]; [/for] [for (p : Property   c.attribute)] public [p.type.name/] get[p.name.toUpperFirst()]() { return this.[p.name/]; } [/for] [for (o : Operation   c.ownedOperation)] public [o.type.name/] [o.name/]() { // TODO should be implemented } [/for] } [/file] [/template]</pre>
--	--

(a) Code Template of Windows Mobile (C#)

(b) Code Template of Android (Java)

Fig. 1. Proposed Code Templates

The code template for Windows Mobile Figure 1(a) is composed in the form of created code and additional data input. Observing the code template, it is divided into special commands using “[ ]” and the parts generated as it had been put in. For example, “[if (c.isAbstract)] abstract[/if]” means to create the abstract when the condition `c.isAbstract` is true as is shown in the figure. Also “[for (p: Property | c.attribute)][p.type.name/] [p.name/]; [/for]” means to execute the output of property type name and property name to the class by repeating all the property `p` in the attribute.

Code template for Android Figure 1(b) also is composed in the form of created code and input of additional data as same as Windows Mobile. Observing the code template, it is divided into the special commands using “[ ]” and the parts generated as it had been put in. As is in the figure, “class [c.name.toUpperFirst()/]” means to change the first character of the class name to the uppercase and to execute the output of the class text. If the class name is `cname`, its output is going to be `CName`.

The difference of code templates between Windows Mobile and Android is the difference of C# and Java used in each platform. The code template composed like this is finally manipulating the data and creating the code using the meta model of UML model.

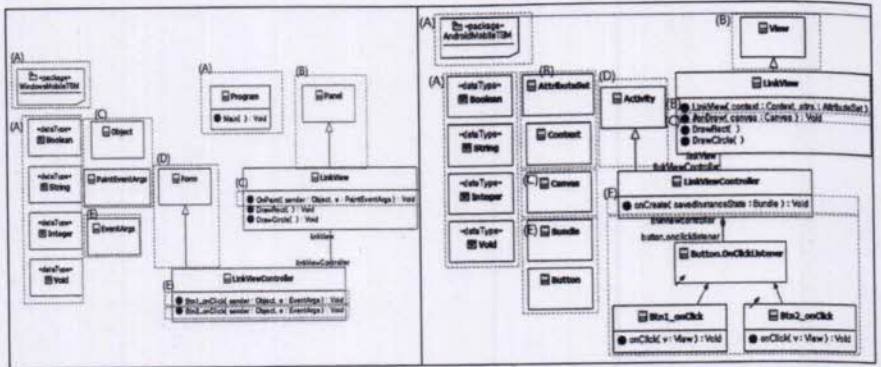
#### 4 Code Generation of Heterogeneous Smartphone Application Using Code Template

The final code created using the code template is shown in Figure 2. `LinkViewController` and `LinkView` classes have been created in the Windows Mobile application as is shown in Figure 2(a) as the input class diagram has 2 classes. As Windows Mobile inserts event handler inside the function for the button click, button handler has been added up in the internal function of `LinkViewController`. The development is completed with addition of function detail of corresponding functions based on the automatically generated code.

In the Android application of Figure 2(b), the classes of `LinkViewController`, `Btn1_onClick`, `Btn2_onClick`, and `LinkView` have been created as the input class diagram has 4 classes. Android application can be developed only with the function addition of corresponding functions based on the automatically generated code. Created `LinkViewController` class has been generated in the Java form of UML model's `onCreate`. Android is adding up event handler by using the anonymous class. Therefore, 2 buttons have been divided into the class. `onClick` of `Btn1_onClick` class is moved to the assigned page. `onClick` of `Btn2_onClick` class is a code that draws circle and rectangle alternately on the screen. The code that draws circle and rectangle on the screen has been composed up in the `LinkView` class.

It is not a perfect creation of code through UML design, however, about 30% of the code for Window Mobile application and about 35% of the code for Android application can be generated as of now. The design and embodiment procedure can be carried out more swiftly if the code creation rate can be increased.





```

class LinkViewController : Form {
    Linkview linkView;
    public LinkViewController() {
    }
    private void Btn1_onClick(object
sender, EventArgs e) {
    }
    private void Btn2_onClick(object
sender, EventArgs e) {
    }
}
class LinkView : Panel {
    private void OnPaint(object sender,
PaintEventArgs e) {
    }
    public void DrawRect(PaintEventArgs e)
{
    }
    public void DrawCircle(PaintEventArgs
e) {
    }
}
    
```

(a) Model of Windows Mobile

```

public class LinkViewController extends
Activity {
    public void onCreate(Bundle
savedInstanceState) {
    }
}
class Btn1_onClick extends
Button.OnClickListener{
    public void onClick(View v) {
    }
}
class Btn2_onClick extends
Button.OnClickListener{
    public void onClick(View v) {
    }
}
public class LinkView extends View {
    public LinkView(Context context,
AttributeSet attrs) {
    }
    protected void onDraw(Canvas canvas) {
    }
    public void DrawRect(Canvas canvas) {
    }
    public void DrawCircle(Canvas canvas) {
    }
}
    
```

(b) Model of Android

Fig. 2. Model-to-Text Transformation

## 5 Conclusion

This paper has created the code of previously composed target dependent model using the Model-to-Text transformation. Template-based approach has been used for the creation of code. While it requires composing the template for code generation, code template has been designed with the analysis of existing codes for Windows Mobile and Android. And for the execution of model transformation, template has been composed with the grammar of Acceleo tool. As an application example, code has been created by applying code template to the platforms of Windows Mobile and Android. As its result, it was possible to create 30% and 35% codes out of total codes for Windows Mobile and Android respectively.

The code template proposed can create skeleton code only as it is based on the class diagram. The quantity and quality of code generation are going to be enhanced in the future by composing the code templates applied with sequence and state diagram in addition to the class diagram. It will be possible to develop heterogeneous smartphone applications more swiftly and effectively when adding up other diagrams besides class diagram.

## References

1. Gang, D.: Touching the iPhone SDK 3.0. In: Insight (2009)
2. Seokhoon, K.: A Trend of Android Platform. The Korea Contents Association Semiannual 8(2), 45–49 (2010)
3. Wigley, A., Moth, D., Foot, P.: Microsoft Mobile Development Handbook. Microsoft Press, Redmond (2007)
4. Jegal, B.: Smartphone market and Mobile OS Trends. In: Semiconductor Insight (2010)
5. Son, H.S., Kim, W.Y., Kim, R.Y.C.: Semi-Automatic Software Development based on MDD for Heterogeneous Multi-Joint Robots. In: CA 2008(IEEE), vol. 2, pp. 93–98 (2008)
6. Kim, W.Y., Son, H.S., Kim, J.S., Kim, R.Y.C.: Development of Windows Mobile Application using Model Transformation Technique. Korea Computer Congress 16(11), 1091–1095 (2010)
7. Son, H.S., Kim, W.Y., Jang, W.S.: Robert Young Chul Kim: Development of Android Application using Model Transformation. In: KIISE & KIPS Joint Workshop on Software Engineering Technology 2010, vol. 8(1), pp. 64–67 (2010)
8. Kim, W.Y., Son, H.S., Yoo, J., Park, Y., Kim, R.Y.C.: A Study on Target Model Generation for Smartphone Applications using Model Transformation Technique. In: ICONI 2010, vol. 2, pp. 821–823 (2010)
9. OMG: MOF Model to Text Transformation Language, v1.0, OMG Available Specification (2008)
10. Czarnecki, K., Helsen, S.: Feature-Based Survey of Model Transformation Approaches. IBM Systems Journal 45(3), 621–664 (2006)
11. Jamba: The Java Model Driven Architecture 0.2 (2003), <http://sourceforge.net/projects/jamda/>
12. Obeo, Acceleo User Guide, <http://www.acceleo.org/>



## Communications in Computer and Information Science

The CCIS series is devoted to the publication of proceedings of computer science conferences. Its aim is to efficiently disseminate original research results in informatics in printed and electronic form. While the focus is on publication of peer-reviewed full papers presenting mature work, inclusion of reviewed short papers and abstracts reporting on work in progress is welcome, too. Besides globally relevant meetings with internationally representative program committees guaranteeing a strict peer-reviewing and paper selection process, conferences run by societies or of high regional or national relevance are considered for publication as well.

The topical scope of CCIS spans the entire spectrum of informatics ranging from foundational topics in the theory of computing to information and communications science and technology and a broad variety of interdisciplinary application fields.

Publication in CCIS is free of charge. No royalties are paid, however, CCIS volume editors receive 25 complimentary copies of the proceedings. CCIS proceedings can be published in time for distribution at conferences or as post-proceedings, as printed books and/or electronically as CDs; furthermore CCIS proceedings are included in the CCIS electronic book series hosted in the SpringerLink digital library.

The language of publication is exclusively English. Authors publishing in CCIS have to sign the Springer CCIS copyright transfer form, however, they are free to use their material published in CCIS for substantially changed, more elaborate subsequent publications elsewhere. For the preparation of the camera-ready papers/files, authors have to strictly adhere to the Springer CCIS Authors' Instructions and are strongly encouraged to use the CCIS LaTeX style files or templates.

Detailed information on CCIS can be found at [www.springer.com](http://www.springer.com)

ISSN 1865-0929

ISBN 978-3-642-23311-1



› [springer.com](http://springer.com)