ISSN: 1738-9984

International Journal of Software Engineering and Its Applications

IJSEIA

Vol.8, No.3, March, 2014

SCIENCE & ENGINEERING
RESEARCH SUPPORT SOCIETY

Table of Contents

Performance Analysis of Anti-collision Algorithm for Tag Identification Time Improvement **1**

Chang-Su Kim, Bong-Im Jang and Hoe-Kyung Jung

Performance Analysis of ORB Image Matching Based on Android **11**

Yu-Doo Kim, Jin-Tae Park, Il-Young Moon and Chang-Heon Oh

A Genetic Methodology for Object Evolution **21**

Enas Naffar and Said Ghoul

A Case Study of Quality Improvement for Water Resource Management System based on ISO/IEC 9126 **39**

Kidu Kim and R. YoungChul Kim

Model Transformation Rule for generating Automatic Database Schema of Business Process Framework **47**

Chae Yun Seo, Hyun Seung Son and R. Young Chul Kim

Enhancing Red Tide Image Recognition using Semantic Feature and Rotation of Algae Image Angle **55**

Sun Park, Myeong Soo Choi, Yeonwoo Lee and Seong Ro Lee

A Regression Test Selection Technique for SOA Based Applications **65**

Rajani Kanta Mohanty, Binod Kumar Pattanayak and Durga Prasad Mohapatra

International Journal of Software Engineering and Its Applications
Vol.8, No.3 (2014), pp.47-54
http://dx.doi.org/10.14257/ijseia.2014.8.3.05

Model Transformation Rule for generating Automatic Database Schema of Business Process Framework

Chae Yun Seo*, Hyun Seung Son and R. Young Chul Kim*

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

Abstract

The previous researches had developed business process frameworks for easily integrating the separating data processing and managing, and decision-support systems built by different times in different places, but never mentioned how to develop this complex business process structures, that is, six-layer architecture. This paper suggests how automatically to develop a whole database schema of business process framework. To do this, we apply with Model-To-Text transformation based on metamodel to automatically build the schema based business process model. This procedure is follows: 1) defining each meta-model of the entire structure and of database schema, and 2) also defining model transformation rules for it. With of model transformation rules of this procedure, we can automatically transform through meta-modeling of an integrated information system to the schema based model information table specification defined of the entire layer.

Keywords: *Model Driven Architecture (MDA), UML, Metamodel, Model Transformation, Business Process Framework (BPF), Query Language, BPSQL, Model-To-Text Transformation Language*

1. Introduction

An enterprise needs to provide with a business-integrated system framework to quickly change and adapt new business. It is to use the framework possible to easily integrate the system by different teams at different time in different places [2]. Most enterprises have a computer system to efficiently operate systematic information, and also to appropriately preserve/manage it, which consists of closed layer architecture [1]. The closed architecture is based on the layer mechanism, and directly access right under the layer. Seo and Kim [4] suggested and defined five layer structure based on the closed architecture. Seo[9] also suggested to reuse the existing software component for reducing development time and cost with mapping CBD(component based development) and BPM(Business process modeling)[2]. On the previous proposed business process framework based on closed architecture, we defined BPSQL(Business Process Structured Query Language)[7], and showed to retrieve and access information between each layer with the simple associated query statements. In this moment, we need to store each data, but manually develop the structure of each layer on BPF until now. With this database schema, we can create a whole business process framework. Therefore, we suggest how to generate the schema-based business process model with the complex business process framework based on a closed architecture. To do this, we study automatically to generate database schema with BPF modeling information, which 1) defines metamodel of the whole structure for Business process framework, 2) models our own target business based on the metamodel, 3) and also needs M2T (model-to text)transformation rules

ISSN: 1738-9984 IJSEIA

for automatic BPF schema creation. In this time, it is limited that this research represents Business process framework with just XMI data without UI and visualization

This paper describes as follows: Chapter 2 mentions related work, Chapter 3 defines metamodel of business process framework. Chapter 4 describes how to design BPF with a metamodel. And shows M2T transformation rules for creating automatic BPF schema, and conclusion.

2. Related Works

2.1. M2T (Model To Text Transformation Language)

MOF Model to Text Transformation Language (Mof2Text or MOFM2T) is an Object Management Group (OMG) specification for a model transformation language. Specifically, it can be used to express transformations that transform a model into text (M2T), for example a platform-specific model into source code or documentation. MOFM2T is one part of OMG's Model-driven architecture (MDA) and reuses many concepts of MOF, OMG's metamodeling architecture. Whereas MOFM2T is used for expressing M2T transformations, OMG's QVT is used for expressing M2M transformations [12].

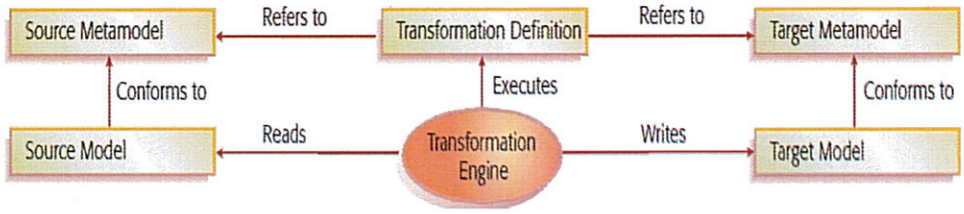

Figure 1. Basic Concepts of Model Transformation

We present two examples of model transformations: one that maps models to models, and another that maps models to code. Figure 1 gives an overview of the main concepts involved in model transformation, and shows the simple scenario of a transformation with one input (source) model and one output (target) model. Both models conform to their respective metamodels. A metamodel typically defines the abstract syntax of a modeling notation. A transformation is defined with respect to the metamodels. The definition is executed on concrete models by a transformation engine. In general, a transformation may have multiple source and target models. Furthermore, the source and target metamodels may be the same in some situations [11].

3. Business Process Framework

A BPF is comprised of five layers: business rules, business processes, services, components, and data modeling. A repository like DB tablization is present at each layer [5]. A BPF is a closed architecture where each layer is directly connected to the next layer.

The layered structure can quickly produce new services by reusing existing components when business demands exist. New businesses can be configured with this service [5]. The repository at each layer is tabulated, and a layer generates data query using BPSQL [6]. Finally, the required data are extracted. Figure 2 is Business Process Framework.

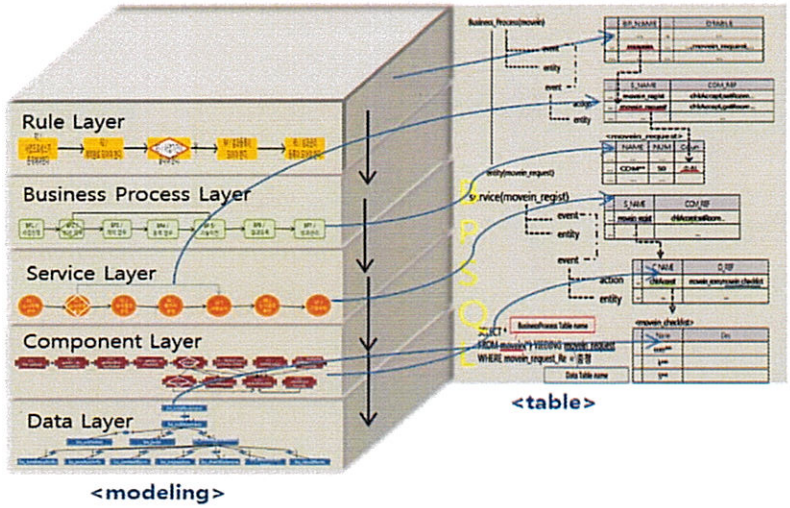

Figure 2. Business Process Framework

3.1. BPF Metamodel

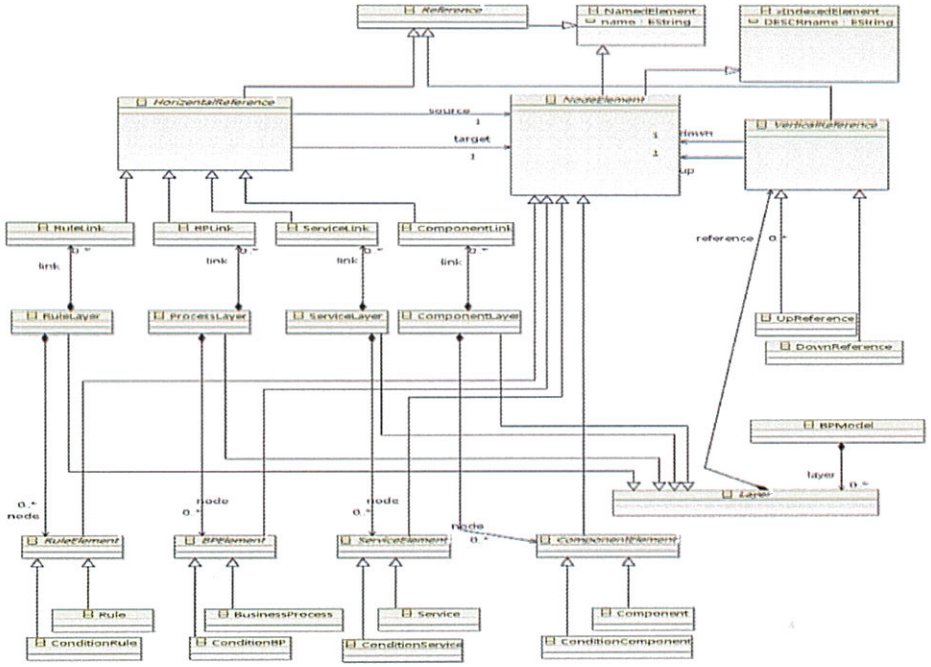

Figure 3. Business Process Framework Metamodel

A BPF metamodel defines the essential elements, grammar, and structure of UML metamodel which creates the business process framework model. For model transformation, Source Metamodel is designed based on BPF Metamodel. Figure 3 is the Business Process Framework Metamodel. It is a detailed description of the BPF metamodel as follows.

Top-level class is "BPmodel", and refers to the "layer" class. "Layer" class is inherited by RuleLayer, ProcessLayer, ServiceLayer, and ComponentLayer. Each layer from the Layer class within BPF inheritance can be modeled. From layer class in BPF, the "HorizentalReference" is modeled in the same layer and "VerticalReference" is modeled in the sub-layer. The "UpReference" of the "Vertical Reference" refers to the superlayer. And then "DownReference" refers to the sub-layer.

3.2. Modeling Integrated Information Management System (IIMS) based on BPF Metamodel

It shows to be modeled with integrated information management system (IISM) based on BPF metamodel. At the second step of transformation model, the appropriate Source Model produces the XMI data. Table1 is the XML data of the BPF metamodeling about Rule Layer, Process Layer, Service Layer, and Component Layer. Table1 just shows two ServiceLayer XMI data and the Process Layer.

Table 1. Process and Service's XMI data of BPF Metamodeling

```
<?xml version="1.0" encoding="UTF-8"?>
<bpmetamodel:BPModel xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:bpmetamodel="http://bpmetamodel/1.0">
<layer xsi:type="bpmetamodel:RuleLayer">
.....
</layer>
<layer xsi:type="bpmetamodel:ProcessLayer">
<reference xsi:type="bpmetamodel:DownReference" name="BP2" up="//@layer.0/@node.0"
down="//@layer.2/@node.0"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP1" DESCRname="Business
Application"/>
<node xsi:type="bpmetamodel:ConditionBP" name="BP2" DESCRname="Appropriate
Business"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP3" DESCRname="Contract
Business"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP4"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP5" DESCRname="Service contract"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP6" DESCRname="Result
registration"/>
<node xsi:type="bpmetamodel:BusinessProcess" name="BP7" DESCRname="Performance
management"/>
<link name="bp1" source="//@layer.1/@node.0" target="//@layer.1/@node.1"/>
<link name="bp2" source="//@layer.1/@node.1" target="//@layer.1/@node.2"/>
<link name="bp2'" source="//@layer.1/@node.1" target="//@layer.1/@node.4"/>
<link name="bp3" source="//@layer.1/@node.2" target="//@layer.1/@node.3"/>
<link name="bp4" source="//@layer.1/@node.3" target="//@layer.1/@node.4"/>
<link name="bp5" source="//@layer.1/@node.4" target="//@layer.1/@node.5"/>
<link name="bp6" source="//@layer.1/@node.5" target="//@layer.1/@node.6"/>
</layer>
<layer xsi:type="bpmetamodel:ServiceLayer">
<reference xsi:type="bpmetamodel:DownReference" name="S1" up="//@layer.1/@node.1"
down="//@layer.3/@node.0"/>
<reference xsi:type="bpmetamodel:DownReference" name="S2" down="//@layer.3/@node.9"/>
<reference xsi:type="bpmetamodel:DownReference" name="S3" down="//@layer.3/@node.10"/>
<reference xsi:type="bpmetamodel:DownReference" name="S4" down="//@layer.3/@node.1"/>
<reference xsi:type="bpmetamodel:DownReference" name="S5" down="//@layer.3/@node.2"/>
<reference xsi:type="bpmetamodel:DownReference" name="S6" down="//@layer.3/@node.4"/>
<reference xsi:type="bpmetamodel:DownReference" name="S7" down="//@layer.3/@node.8"/>
```

```
<node xsi:type="bpmetamodel:Service" name="S1" DESCRname="The Evaluator management"/>
<node xsi:type="bpmetamodel:ConditionService" name="S2" DESCRname="The Evaluator Assignment"/>
<node xsi:type="bpmetamodel:Service" name="S3" DESCRname="Input The Evaluate Result"/>
<node xsi:type="bpmetamodel:Service" name="S4" DESCRname="Create the Assessment"/>
<node xsi:type="bpmetamodel:Service" name="S5" DESCRname="Appropriate Business"/>
<node xsi:type="bpmetamodel:Service" name="S6" DESCRname="Check the The Evaluate Result"/>
<node xsi:type="bpmetamodel:Service" name="S7" DESCRname="Select the Company"/>
<link name="s1" source="//@layer.2/@node.0" target="//@layer.2/@node.1"/>
<link name="s2" source="//@layer.2/@node.1" target="//@layer.2/@node.2"/>
<link name="s2'" source="//@layer.2/@node.1" target="//@layer.2/@node.4"/>
<link name="s3" source="//@layer.2/@node.2" target="//@layer.2/@node.3"/>
<link name="s4" source="//@layer.2/@node.3" target="//@layer.2/@node.4"/>
<link name="s5" source="//@layer.2/@node.4" target="//@layer.2/@node.5"/>
<link name="s6" source="//@layer.2/@node.5" target="//@layer.2/@node.6"/>
</layer>
<layer xsi:type="bpmetamodel:ComponentLayer">
.....
</layer>
</bpmetamodel:BPModel>
```

XMI elements "<layer xsi:type="bpmetamodel:__Layer">" in each layer are separated. The DownReference is associated with the sub-layer, the UpReference is connected with the upper layer. <link> is connected within the equivalent layer, the source and target is distinguished as input and output respectively. <node> is the name of each layer, which describes the role of <node>.

3.3. Transformation Rule with Acceleo

Acceleo is an open-source code generator from the Eclipse Foundation that allows people to use a model-driven approach to building applications. It is an implementation of the "MOFM2T" standard, from the Object Management Group (OMG), for performing model-to-text transformation [12].

We transformed the XML data (which is obtained from the result value of Modeling) into SQL schema with a conversion tool, and use transformation rule with the Acceleo. Table 2 shows Transformation rule with Acceleo. We store SQL transformation with each models by the elements in XMI data.

Table 2. Transformation Rule with Acceleo

```
[comment encoding = UTF-8 /]
[module generate('http://www.eclipse.org/emf/2002/Ecore')]
[template public generateElement(aEPkage : EPackage)]
[comment @main/]
[file (aEPkage.name.concat('.sql'), false, 'UTF-8')]
[for ( eClassFier : EClassifier | aEPkage.eClassifiers)]
[if (eClassFier.eClass().name = 'EClass')]
[if ( eClassFier.oclAsType(EClass).name = 'RuleLayer' or
eClassFier.oclAsType(EClass).name = 'ProcessLayer' or
eClassFier.oclAsType(EClass).name = 'ServiceLayer' or
eClassFier.oclAsType(EClass).name = 'ComponentLayer')]
create table [eClassFier.oclAsType(EClass).name /]
(
[comment Attribute /]
[if (eClassFier.oclAsType(EClass).eStructuralFeatures->select(e | e.name='node')->first().eType.eClass().name = 'EClass')]
[generateAttribute( eClassFier.oclAsType(EClass).eStructuralFeatures->select(e | e.name='node')-
```

```
>first().eType.oclAsType(EClass) ) /]
    [/if]
    [comment link /]
    [if(eClassFier.oclAsType(EClass).eStructuralFeatures->select(e | e.name='link')->first()->size() = 1) ]
    ,[eClassFier.oclAsType(EClass).eStructuralFeatures->select(e    |    e.name='link')->first().name    /]
varchar(100) NULL
    [/if]
    [comment Layer /]
    [for ( eLayerClass : EClass | eClassFier.oclAsType(EClass).eSuperTypes) ]
    [for (eFeature:EStructuralFeature | eLayerClass.eStructuralFeatures) ]
    [if (eFeature.eClass().name = 'EReference')]
    [for ( vClass : EClass | eFeature.eType.siblings().oclAsType(EClass) ) ]
    [if (vClass.eSuperTypes->first().name = 'VerticalReference')]
    ,[vClass.name /] varchar(100) NULL
    [/if] [/for] [/if] [/for] [/for]
    )
    [/if] [/if] [/for] [/file] [/template]
    [template public generateAttribute(eClass : EClass)]
    [if (eClass.eStructuralFeatures->size() = 0) ]
    [for ( loopClass : EClass | eClass.eSuperTypes)][generateAttribute(loopClass) /][/for]
    [else]
    [for (eFeature:EStructuralFeature | eClass.eStructuralFeatures) ]
    [if (eFeature.eClass().name = 'EAttribute')]
    [if (eFeature.name = 'name')]
    [eFeature.name /] varchar(100) NOT NULL PRIMARY KEY
    [else]
    ,[eFeature.name /] varchar(100) NULL
    [/if] [/if] [/for] [/if] [/template]
```

Figure 4 shows the transformed SQL schema adapted with Acceleo Transformation rules.

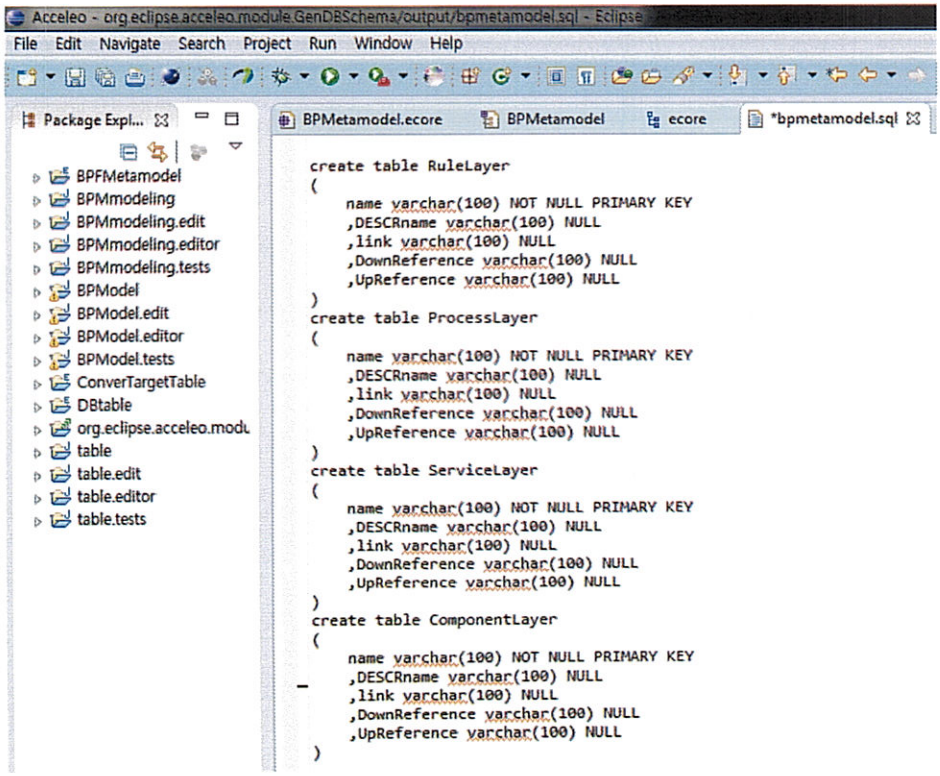

Figure 4. The Results of SQL Conversion showed with Applying a Transformation Rule

This approach automatically generates SQL schema structure with the modeled BPF structure is generated. All layer information is automatically stored in schema structure. Figure 5 shows the results obtained with the connection to the database. Figure 5 is BPF schematic table.

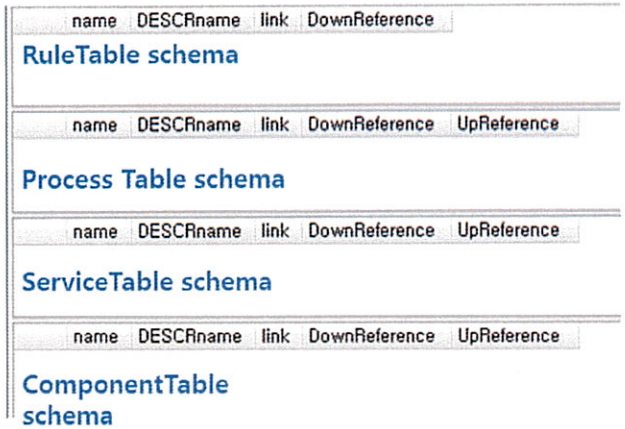

Figure 5. BPF Schematic Table

4. Conclusion

Our previous approaches do not focus on how to automatically develop the whole Business process framework. In this paper, nobody we suggest ModelToText transformation based on metamodel to automatically build the schema based business process model. This procedure is follows: 1) defining each meta-model of the entire structure and of database schema, and 2)also defining model transformation rules for it. With of model transformation rules of this procedure, we can automatically transform through meta-modeling of an integrated information system to the schema based model information table specification defined of the entire layer. Therefore, we can easily and automatically build the whole DB schematic with Model transformation technique.

Acknowledgements

This work was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS(Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"] and Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education (2013R1A1A2011601)

References

[1] C. Yun Seo, D. Woo Kim and R. Young Chul Kim, "Business Process Framework based on the Closed Architecture", Journal of the Korea Academia-Industrial cooperation Society, vol. 10, no. 8, **(2009)**, pp. 1939-1946.
[2] C. Yun Seo, D. Woo Kim and R. Young Chul Kim, "5-Layer Architecture for Efficient Business Process Modeling", IWIT, vol. 6, no. 1, **(2008)**, pp. 19-22.
[3] J. Martin, "Information Engineering", Prentice-Hall International, Inc., **(1990)**.

[4] C. Yun Seo, S. Young Moon, R. Young Chul Kim and B.-H. Ahn, "A Study on Modeling Efficient Business Process Framework: Mapping Business process Layer and Data Layer", The 1st Yellow Sea International Conference on ubiquitous Computing, Shandong Univ, China, vol. 1, no. 59, **(2011)**.
[5] R. S. Pressman, "Software Engineering A Practitioners' Approach", 3rd Ed, McGraw Hill, **(2004)**.
[6] P. Hedley Apperly, R. Hoffman and S. Latchem, "Service-And Component-Based Development".
[7] C. Yun Seo, S. Young Moon and R. Young Chul Kim, "A Study on Data Migration for BPSQL Query Language on Business Process Framwork(BPF)", KIISE, **(2011)**.
[8] W. Yeol Kim, H. Seung Son and R. Young Chul Kim, "Development of Windows Mobile Applications using Model Transformation Techiques", KIISE, **(2010)**.
[9] C.-Y. Seo and R. Young Chul Kim, "Development through Mapping CBD, Service Model onto BPM based on Business Process Framework", KIPS, **(2012)**.
[10] C.-Y. Seo and R. Young Chul Kim, "Design of Metamodel for 5 Layer Information on Business Process Framework", KIPS, **(2012)**.
[11] K. Czarnecki and S. Helsen, "Feature-based survey of model transformation approaches", **(2006)**.
[12] OMG, "MOF Model to Text Transformation Language", vol. 10, no. 1, **(2008)**.

Authors

Chae Yun Seo, received the B.S. and M.S degree in Software Engineering from Hongik University, Korea in 2005. He is currently a Ph.D. candidate in Hongik University and an adjunct professor in Yuhan College. His research interests are in the areas of metamodel, business process model, component, service, and framework.

Hyun Seung Son, received the B.S. and M.S. degree in Software Engineering from Hongik University, Korea in 2009. He is currently a Ph.D. candidate in Hongik University. His research interests are in the areas of Automation Tool Development in Embedded Software, Real Time Operation System Development, Metamodel design, Model Transformation, and Model Verification & Validation Method.

R. Young Chul Kim received the B.S. degree in Computer Science from Hongik University, Korea in 1985, and the Ph.D. degree in Software Engineering from the department of Computer Science, Illinois Institute of Technology (IIT), USA in 2000. He is currently a professor in Hongik University. His research interests are in the areas of Test Maturity Model, Embedded Software Development Methodology, Model Based Testing, Metamodel, Business Process Model, and User Behavior Analysis Methodology.