J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

A Systematic Design Automation Method for RDA-based .NET Component with MDA☆

Deuk Kyu Kum[1*]

ABSTRACT

Recent Enterprise System has component driven real-time distributed architecture (RDA) and this kind of architecture should performed with satisfying strict constraints on life cycle of object and response time such as synchronization, transaction and so on. Microsoft's .NET platform supports RDA and is able to implement services including before mentioned time restriction and security service by only specifying attribute code and maximizing advantages of OMG's Model Driven Architecture (MDA). In this study, a method to automatically generate an extended model of essential elements in an enterprise-system-based RDA as well as the platform specific model (PSM) for Microsoft's .NET platform are proposed. To realize these ideas, the functionalities that should be considered in enterprise system development are specified and defined in a meta-model and an extended UML profile. In addition, after defining the UML profile for .NET specification, these are developed and applied as plug-ins of the open source MDA tool, and extended models are automatically generated using this tool. Accordingly, by using the proposed specification technology, the profile and tools can easily and quickly generate a reusable extended model even without detailed coding-level information about the functionalities considered in the .NET platform and RDA.

☞ keyword : Model Driven Architecture (MDA), Unified Modeling Language (UML) Profile, Microsoft .NET, Real-time Distributed Architecture (RDA), Attribute Code

1. Introduction

In recent years, business environments have become more complex, and therefore, enterprises need to be capable of flexible and agile response. In this context, a real-time distributed architecture (RDA) is widely accepted as an architecture that supports enterprise systems [1]. In this type of architecture, time constraints and security services, such as object pooling, synchronization services, and transaction services for object life cycle and response time, are essential [2]. Initial studies of OMG's Model Driven Architecture (MDA) mainly focused on supporting various conversion formats and source code generation for various development languages. Recently, a method has been developed to generate an extended model by applying the elements necessary for model extension from the basic model. However, it was limited to specific fields or specific tools such as embedded software, and it lacked model extensibility for the abovementioned time constraints.

Microsoft's .NET platform supports RDA. It can be used to implement components that support services such as time constraints and security by simply specifying an attribute code in the source code. Therefore, the benefits of MDA can be maximized by using the .NET attribute code and Unified Modeling Language (UML) profile mechanism. However, compared to the meta-model and UML profile in Enterprise Java Beans [3], .NET is not yet established. Only a few studies have focused on the platform-specific model (PSM) specification technique for .NET.

In this study, the functional elements of various services that should be considered for developing enterprise systems are specified and defined as a meta-model and an extended UML profile. In addition, after defining the UML profile for PSM specifications for .NET, these are developed and applied as plug-ins of StarUML [4], an open source MDA platform, and the design model is generated automatically. Therefore, even if we do not know detailed coding-level information about the functions to be considered in the .NET platform and the RDA using the proposed specification technique, profile, and tool, the roles of each element of the

1 Dept. of Information and Communication Engineering, Yuhan University, Bucheon, Gyeonggi, 14780, Korea.
* Corresponding author (dkkum1@yuhan.ac.kr)
[Received 30 December 2018, Reviewed 23 January 2019, Accepted 25 February 2019]
☆ This paper is an extended version of the paper presented and published in APIC-IST 2018 conference.

model must be defined. A reusable expansion model can be generated easily and quickly if an attribute value can be established according to the role. Because the proposed profile is an extension of OMG's UML profile and MDA standard, it can be reused in several Meta Object Facility (MOF) [5]-compliant UML and MDA tools to increase the productivity, scalability, portability, and maintainability of the design model.

The remainder of this paper is organized as follows. Chapter 2 describes related research. Chapter 3 defines RDA-based service elements and their meta-model. Chapter 4 defines the UML profile to which the specified service is applied, and it describes the model design process, presentation method, and transformation process using the defined profile. Chapter 5 describes case studies. Finally, Chapter 6 presents the conclusions of this study.

2. Related Works

2.1 UML Profile for EDOC

OMG proposed a UML Profile for Enterprise Distributed Object Computing (EDOC) for a distributed computing environment to express events, processing, entities, and patterns in the component architecture. However, it is difficult to express functions such as time constraints and security services such as object pooling, synchronization service, and transaction service. The main components of the UML Profile for EDOC are as follows.

First, Enterprise Collaboration Architecture (ECA) [6] is used for developing an EDOC system with a modeling framework. Second, UML Profile for Patterns [7] can express business function object patterns using UML package notation. Third, UML Profile for ECA provides specifications for entities, events, and business. Forth, the UML Profile for MOF specifies the mapping between UML and MOF. Finally, the UML Profile for Relationships specifies a standard for relationships in business models. Table 1 shows the element mapping for entities.

(Table 1) Mapping in UML Profile for ECA elementt

Meta-model Element	UML Profile Element	UML Base Class
Data Manager	Data Manager	Class
Entity Data	Entity Data	Class
Entity	Entity	Class
Entity Role	Entity Role	Class
Class Key	Key	Class
Key Element	Key Element	Attribute
Key Attribute	Key Attribute	Attribute
Foreign Key	Foreign Key	Attribute
Data Manager	Data Manager	Class

2.2 Wang's Model to Model Transformation [8]

Wang proposed methods for automating model-to-model mapping and transformation to support model-based system engineering. He defined model transformation rules for both semantics and syntax and described the mapping and transformation between models as a meta-model-based transformation process. In addition, the proposed tool can automate model generation and testing using extended model design capabilities with defined rules. However, the meta-model for the proposed technique and extended model generation does not comply with OMG's MOF standard, and thus, it lacks portability and interoperability. The detailed specification elements and supporting platform of PSM are not mentioned.

2.3 .NET Attribute code

In the .NET platform, some specific form may be defined in advance and be used to control the run time motion; this is called the "Attribute" code. The attribute class provides convenient methods for testing the applicable characteristics and access to designate user. All attributes are directly or indirectly derived from the attribute, and the characteristics may be applied to all subject elements [9]. Figure 1 shows an example of specifying attributes for transactions, object pooling, Component Object Model (COM), and so on using attribute codes in a minimal .NET class code. Attributes can also specify security, synchronization, timely activation, and other factors.

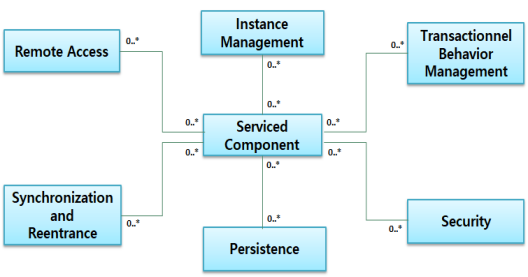
```
…
   [ComVisible(true)]
   [Transaction(TransactionOption.Supported)]
   [ObjectPooling(true,
      MinPoolSize=5,MaxPoolSize=10,
      CreationTimeout=5000)]

   public class MyComponent : ServicedComponent
   {
            public MyComponent()
            {
            …
            }
…
```

(Figure 1) .NET Attribute code format

3. RDA Service Elements

We define a meta-model for functional elements that are considered essential for developing an RDA-based enterprise system. The application architecture is determined by the application requirements. In general, high-end enterprise applications require high scalability and stability for the basic user experience; therefore, an RDA is selected for this purpose. These architectures inevitably use functions such as synchronization, transaction, object pooling, and security services depending on the object's life cycle and response time [10].

A Serviced Component supports an RDA-based service, and the service elements mainly include remote access, instance management, transaction management, synchronization, persistence, and security services, as shown in Figure 2.



(Figure 2) RDA-Service meta-model

Remote access is a service that invokes an instance of a locally detached component to request the necessary functions. Instance management is a service that manages instance activation/deactivation, pooling, and lifecycle. The class that wants to be a component supporting RDA-based service can use Serviced Component class as the base class. Table 2 summarizes these service elements. In this study, it is called RDA-Service.

(Table 2) Main RDA-Service elements

Element Name	Specification Notation	UML Base Class	Remark
Transaction support	«Transaction»	Class, Component	
Transaction complete	«AutoComplete»	Method	
Security	{Security = "role name", SetEveryoneAccess = value}	Class	OCL
Synchronization	«Synchronization»	Class	
Object pooling	{Object Pooling = true, MinPoolSize = value, MaxPoolSize = value, CreationTimeout = value}	Class	OCL
Object life cycle	{Just In Time Activation}	Class	OCL
Queue use	«Queued»	Class, Component	

4. RDA-.NET Profile

We propose a UML profile that supports RDA-Service specifications based on the meta-model defined above and a UML profile for .NET/C#. This profile assumes a .NET/C#-related UML profile and adds the RDA-Service elements defined in this study. In this study, it is called RDA-.NET profile.

4.1 RDA-Service Specification Elements

In Table 3, transaction support expresses the generated code as having a transaction attribute as a component. A class or component specified as a «Transaction» stereotype will have the transaction attribute to be supported when participating in the transaction as needed. The transaction termination is specified by the «AutoComplete» method, and therefore, the generated code automatically determines whether to commit or cancel the entire transaction including

the component depending on whether the error occurred at the end of the method. It is the role that can be performed.

Security uses the {Security = "role name", SetEveryoneAccess = value} tagged value, and the role name specifies the policy name (role name) that sets security-related policies and roles. The SetEveryoneAccess property is a setting for control permissions for all user groups. If the property is set to true, all user groups are automatically added to the role. Object pooling refers to a mechanism that activates an object in a pool rather than creating a new one when the client requests the component by pooling an instance of the required component in advance.

Just-In-Time (JIT) activation refers to deactivating an object immediately when the component completes its work, even if the client maintains a reference to the component. When the client invokes the second method in this component, it instantaneously activates an instance of a new component. In other words, when Just-In-Time activation is used, the components used by the client are different each time the method is invoked. Timely activation helps ensure transaction accuracy, especially consistency and isolation.

(Table 3) UML profile element for RDA-service specification

Element Name	Specification Notation	UML Base Class	Remark
Transaction support	«Transaction»	Class, Component	
Transaction complete	«AutoComplete»	Method	
Security	{Security = "role name", SetEveryoneAccess = value}	Class	OCL
Synchronization	«Synchronization»	Class	
Object pooling	{Object Pooling = true, MinPoolSize = value, MaxPoolSize = value, CreationTimeout = value}	Class	OCL
Object life cycle	{Just In Time Activation}	Class	OCL
Queue use	«Queued»	Class, Component	

4.2 Element for .NET/C# Specifications of RDA-.NET Profile

We propose a meta-model that defines the essential elements, syntax, and structure of each PSM element for .NET. The PSM meta-model defines the PSM model and a design model element to be transformed when the PSM model is generated. The PSM meta-model defines PSM meta-model elements, types, and meta-classes for each PSM model element and describes the constraints. The PSM meta-model classifies the items to be expressed in PSM by classifying them as shown in Table 4. Elements common to all layers, such as "C# Operator," are described in the common layer.

Table 5 defines the components of the UML profile for the .NET / C # specification in the RDA-.NET profile. These profile elements are intended to extend the PSM meta-model elements for .NET to generate PSM models for .NET / C #.

4.3 Component development process applying RDA-.NET profile

We developed a prototype tool based on StarUML to make the proposed technique more practical and easy to use. StarUML can objectize modules to access most programs such as UML meta-model and application object, expose the API to the outside, and easily develop plug-ins. In this chapter, we propose the component development process applying the RDA-.NET Profile and StarUML plug-ins, as shown in Figure 3. In the requirements analysis phase, the requirements specification is analyzed to derive functional and nonfunctional requirements.

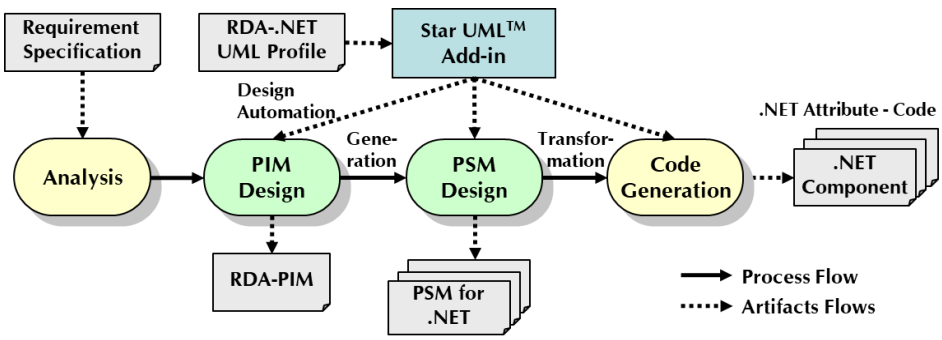
First, the RDA-.NET Profile is created in the XML/XMI format as defined above, and it is developed using StarUML plug-ins for use. In the next PIM design phase, the service element based on a real-time distributed architecture is designed using plug-ins. In this paper, platform independent design using RDA-.NET Profile is defined as "RDA-PIM." In the PSM design stage, the rough RDA-PIM design is used to create a detailed PSM design for .NET that considers the .NET platform using the specification elements provided by the profile. Finally, in the code generation step, the PSM is generated using RDA-.NET Profile plug-ins, and the .NET

(Table 4) PSM meta-model element for .NET

Layer	Meta-Model Element	Type	Definition	Meta Class
Presentation	ASPX	Stereotype	Indicates user interface	UML Class
Business	.NET Assembly	Stereotype	A unit of reusable and deployable .NET components	UML Component
Persistent	Data Type	Stereotype	Dataset class in which information is stored	UML Class
Common	C# Operator	Stereotype	Indicates C# operator	UML Operation

(Table 5) UML profile element for .NET/C# specification

Element Name	Specification Notation	Description	Base Classes
DotNet Assembly	<<DotNetAssembly>>	C# by compilation result of source file that .NET assembly	UML Component
CSharp SourceFile	<<CSharpSourceFile>	C# source file that implementation code comes	UML Component
CSharp Delegate	<<CSharpDelegate>>	C# Delegate	UML Class
CSharp Struct	<<CSharpStruct>>	C# Struct type	UML Class
CSharp Event	<<CSharpEvent>>	When define C# event object	UML Operation
CSharp Property	<<CSharpProperty>>	Display that is C# Property that express attribute of class or structure	UML Operation
CSharp Indexer	<<CSharpIndexer>>	Display that is Indexer that can approach class or structure with general arrangement in C#	UML Operation
ASPX	<<ASPX>>	Display that is client side web page by .NET web page extension life	UML Class
Data Type	<<DataType>>	By objective that information is stored, is corresponded in mapping table to database	UML Class



(Figure 3) Component development process applying the RDA-.NET profile

component artifact including the .Net attribute code is generated. This study focuses on the scope of PIM and PSM design except code generation.

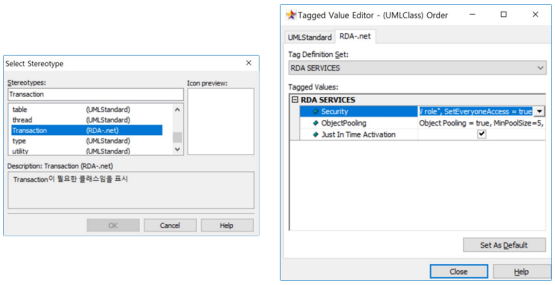
5. Case Study

We applied the rental/reservation management function to Best Car Corporation's (BCC) sales (car rental) management system. It is based on StarUML, .NET/C #, and XML/XMI.

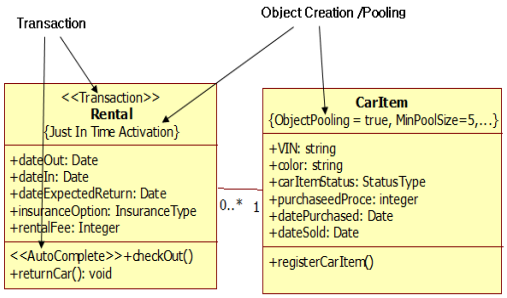
5.1 RDA-PIM design

After doing necessary work such as creating the icon file and registry registration file corresponding to the RDA-.NET profile defined above, a profile is added to the new project through StarUML's profile manager. After including the RDA-.NET profile, we designed the PIM for the system to be developed. At this time, the RDA-Service specification defined above is generated in the required class, component, or method by using the developed plug-ins.

Figure 4 shows the stereotype of the RDA-.NET profile and the content of the tag definition item by using the extension attribute editor of the plug-ins. The set values are automatically generated in the form of stereotype, tagged value, OCL, etc.



(Figure 4) Setting values of elements defined in RDA-.NET profile



(Figure 5) Example of generated RDA-PIM

Figure 5 shows that the "Rental" class supports transaction attributes and the "checkOut" method is created to support the "AutoComplete" attribute for transaction termination and "Just In Time Activation." In the "CarItem" class, property values for object pooling are generated in the OCL format.

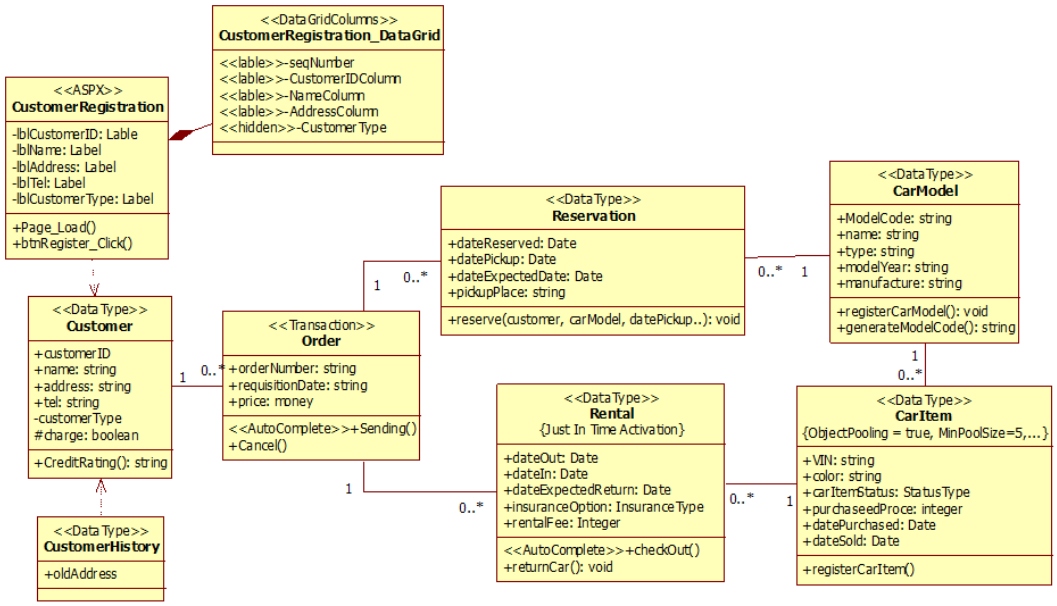
5.2 Generation of PSM for .NET

After generating the RDA-PIM, we generate the PSM for .NET using the items from the .NET/C# specification of RDA-.NET Profile. For this purpose, we define the transformation rules and constraints necessary for transforming PIM to PSM. The QVT for transformation between the MOF-based models established by OMG consists of the language for model generation, language for model query, and transformation definition language for describing transformation rule [11]. Figure 6 shows the definitions used when mapping a PIM model to a PSM for .NET based on QVT. The RDA-Service element defined in the RDA-.NET Profile is a .NET attribute class, and the .NET/C# element is PSM for .NET and is converted into model definition items. Source/target conditions are written using OCL Boolean expressions.

Figure 7 shows the result of modeling RDA-PIM and generating PSM for .NET through the add-in that implements the transformation rule. The example in Figure 7 shows that the RDA-Service element defined in the RDA-.NET profile and the PSM meta-model definition elements for .NET are generated as a class diagram for the rental/reservation management module of BCC's sales management system.

```
  Transformation     RDAServiceToAttributeClass
(UMLProfile, .NET) {
  source umlProfileElement : UMLProfile :: RDAService;
  target attributeClass : .NET :: AttributeClass;
  unidirectional;
  mapping
        umlProfileElement.name     <~>
              .NETAttributeClass.name
    umlProfileElement.parameter     <~>
              .NETAttributeClass.parameter
    } ...
  S := GetStringTaggedValue(AClass, 'RDA-.NET',
   'RDA SERVICES', 'transaction');
   if S     <> '' then begin
     FWriter.WriteLine('[Transaction(%s)]',     [S]);
     PropAdded     := True;
   end
   else     if AClass.StereotypeName = 'Transaction' then
begin
     FWriter.WriteLine('[Transaction(TransactionOption.S
upported)]');
     PropAdded     := True;
   end; ...
```

(Figure 6) Realization of transformation rule using transformation definition language



(Figure 7) Rental/reservation administration class diagram (PSM)

6. Conclusions

In this study, we propose an RDA-.NET profile for RDA-based services such as transaction, security, synchronization service, and object pooling that are essential for enterprise applications as well as extension models for .NET platform. In addition, it was constructed to apply it to the StarUML open source modeling platform; plug-ins that was implemented and applied the defined meta-model, transformation rule, etc.; and generated RDA-PIM and PSM for .NET.

The RDA-.NET profile is an XMI-based XML document that can be added or modified easily, and the plug-ins can be applied easily using an external API. In addition, the RDA-.NET profile supports OMG's UML Profile function and conforms to MOF, and therefore, it can be used with MOF-compliant UML tools and MDA tools. Therefore, we can use the proposed method to easily and quickly generate reusable extension models even if we do not know low-level information about the functions to be considered in the .NET platform and RDA, and it can increase the productivity, scalability, portability, and maintainability of the design model.

References

[1] H. N. Sad, T. Noria, "A Novel Approach for Integrating Security in Business Rules Modeling Using Agents and an Encryption Algorithm," Journal of Information Processing Systems, Vol. 12, No. 4, pp. 688-710, 2016. http://dx.doi.org/10.3745/jips.03.0056

[2] M. Thirumaran and G. G. Brendha, "Incremental stages of a semantic framework for automating the changes on long term composed services," Human-centric Computing and Information Sciences, Vol. 6, No. 1, pp. 1-26, 2016. http://dx.doi.org/10.1186/s13673-016-0067-0

[3] OMG. Meta-model and UML Profile for Java and EJB Specification. February 2004. Version 1.0, formal/ 04-02-02. An Adopted Specification of the Object Management Group, Inc.

[4] Open source UML/MDA platform.

https://sourceforge.net/projects/staruml/
[5] MOF Model to Text Transformation Language RFP, OMG document ad/04-04-07. https://www.omg.org/mof/
[6] OMG. UML Profile for Enterprise Collaboration Architecture (ECA) V1.0, 2004. https://www.omg.org/news/meetings/workshops/Web_Services_USA_Manual/08-5_Casanave.pdf
[7] OMG. UML Profile for Patterns V1.0, 2004. https://www.omg.org/spec/category/uml-profile
[8] T. Wang, S. Truptil, F. Benaben, "An automatic model-to-model mapping and transformation methodology to serve model-based systems engineering," Information Systems and e-Business Management, Vol. 14, pp. 1–14, 2016. http://dx.doi.org/10.1007/s10257-016-0321-z
[9] J. Lowy, "COM and .NET Component Services", p. 384, O'Reilly, Boston, 2001.
[10] D. S. Platt, "Understanding COM+", Microsoft Press, Washington D.C., 1999.
[11] MOF 2.0 Query/Views/Transformations SPEC, OMG document ad/2002-04-10. https://www.omg.org/spec/QVT/About-QVT/

◐ 저 자 소 개 ◑

Deuk Kyu Kum

Deuk Kyu Kum is currently a professor in Dept. of Information and Communication Engineering, Yuhan University, Bucheon, Korea. He received the M.S. and Ph.D. degrees in Computer Science and Engineering from Soongsil University, Korea, in 2005, 2012, respectively. His research interests include big data analysis technology, internet and mobile computing, and cloud computing.
E-mail : dkkum1@yuhan.ac.kr