Eclipse Based Plug-ins for Restful Business Process and Code Generation

Supervisor: Li Ming, ZHU
Accessor: Yang, LIU
Hao Xiang, SHEN, z3160765,
Xiang Yu, WEN, z3156959
# Contents

## SECTION I  INTRODUCTION

<table>
<thead>
<tr>
<th>1.1 Motivation</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2 Thesis Goal</td>
<td>5</td>
</tr>
</tbody>
</table>

## SECTION II  RESEARCH

<table>
<thead>
<tr>
<th>2.1 Business Process</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2 Web Service and SOAP</td>
<td>7</td>
</tr>
<tr>
<td>2.2.1 SOAP Based Web Service</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2 Limitation</td>
<td>7</td>
</tr>
<tr>
<td>2.2.3 Tools Support for MDD</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Restful Web Service</td>
<td>10</td>
</tr>
<tr>
<td>2.3.1 What is REST?</td>
<td>10</td>
</tr>
<tr>
<td>2.3.2 Rest Principal</td>
<td>11</td>
</tr>
<tr>
<td>2.3.3 Restful Web Service</td>
<td>12</td>
</tr>
<tr>
<td>2.3.4 Lack of tool support</td>
<td>13</td>
</tr>
<tr>
<td>2.4 Resource Orientated Business Process</td>
<td>14</td>
</tr>
<tr>
<td>2.4.1 ROA</td>
<td>14</td>
</tr>
<tr>
<td>2.4.2 Business Process in Restful Way</td>
<td>15</td>
</tr>
</tbody>
</table>

## SECTION III  DEVELOPMENT

<table>
<thead>
<tr>
<th>3.1 Design</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.1 BPMN Modeler</td>
<td>17</td>
</tr>
<tr>
<td>3.1.2 Extension Points</td>
<td>17</td>
</tr>
<tr>
<td>3.1.3 Overview of Plug-ins</td>
<td>19</td>
</tr>
</tbody>
</table>
3.2 Implementation........................................................................................................... 22
  3.2.1 Knowledge Base and Choice of the Library...................................................... 22
  3.2.2 Add Annotation on Business Process.............................................................. 24
  3.2.3 Query the model ............................................................................................... 35
  3.2.4 An Eclipse Context View .................................................................................. 37
  3.2.5 An Code Generation Engine .............................................................................. 38
  3.2.6 An Eclipse perspective ..................................................................................... 42
  3.3 Evaluation .............................................................................................................. 44

SECTION IV FURTHER EXTENSION ............................................................................. 45
  4.1 Business Process Workflow Pattern Detector..................................................... 45
  4.2 Generation of Text-based File .............................................................................. 47
  4.3 Internationalization .............................................................................................. 48

SECTION V CONCLUSION ......................................................................................... 49

BIBLIOGRAPHY ............................................................................................................. 51

APPENDIX A HOW TO EXTENDS THE WORK .............................................................. 53

APPENDIX B INSTALLATION GUIDE .......................................................................... 59

APPENDIX C DEMONSTRATION .................................................................................. 60
SECTION I  
INTRODUCTION

1.1 Motivation

REST as an architecture style is increasing gaining the attention of both industry and research communities, but the lack of tools support for business process has been a major concern for applying REST in wider problem domains. SOAP as a competitor, has been used as a default implementation to web service, there are lots of MDD tools currently in the market which support business process composition and code generation toward SOAP-based web service. Since the technology has evolved, business process and MDD can be put together to help non-IT people such as business analyst to take a more important role in software development, we see an emerging need of a toolset for Restful business process development. Therefore, we conduct a research on a topic about Restful business process which is resource centric, and then design and develop a prototype MDD tool for applying REST principle in business process composition and code generation directly from the business process model.
1.2 Thesis Goal

First, research on the Restful business process composition, and the artifact needed on the business process model which can be used in code generation towards Restlet framework. Second, design a set of Eclipse plug-ins based on the existing Eclipse BPMN Modeler that can provide extra functionalities to be a prototype MDD tool, these functionalities includes:

- Annotate extra information on the business process diagram.
- Add process fragments to the “tasks” in the business process diagram.
- Using “Context View” to edit process fragment.
- Generate target code for Restlet framework using a code generation engine.
- Using “Bpmn Perspective” for business process development.
SECTION II RESEARCH

2.1 Business Process

Business process is a series of tasks designed to accomplish a particular outcome. In various industries, the outcome can be a product, a service or the certain state of an object.

A standard business process can be decomposed into several sub-processes, which have their own attributes, but also contribute to achieving the goal of the super-process. The outcome of a well designed business process is increased effectiveness and increase efficiency. The analysis of business processes typically includes the mapping of processes and sub-processes down to activity level. In the other words, all the activities and their relationship among each other in one business to achieve a certain goal can be well expressed in the business process. [18] This is the reason of why the business process is important and widely used in various industry areas.

Business processes can be modeled in a large number of techniques and methods. The Business Process Modeling Notation (BPMN) is one instance of these. The BPMN is a standardized graphical notation for drawing business process in a workflow. The primary goal of BPMN is to provide a standard notation that is readily understandable by all business stakeholders. These business stakeholders include the business analysts who create and refine the processes, the technical developers responsible for implementing the processes, and the business managers who monitor and manage the processes. Consequently BPMN is intended to serve as common language to bridge the communication gap that frequently occurs between business process design and implementation. [19]

In fact, our thesis is developed as an extension of the “SOA Tools BPMN Modeler” which is one of the BPMN plug-in of the Eclipse. Detail of this will be further discussed in the Development Section of this report.
2.2 Web Service and SOAP

2.2.1 SOAP Based Web Service

Simple Object Access Protocol (SOAP) is a protocol for exchanging XML-based messages over computer networks. SOAP forms the foundation layer of web services protocol stack providing a basic messaging framework upon which abstract layers and be built. It delivers the data needed for business process activities as an integrated service in XML message over the HTTP or SMTP protocol used on WEB between clients and servers.

As the most widely used as Web Service standard, SOAP appears several technical advantages relative to alternative technologies. First of all, by using SOAP over HTTP allows for easier communication through proxies and firewalls than previous remote execution technology. Secondly, it is versatile enough to allow for the use of different transport protocols [20]. (e.g., SMTP) There are a lot more, such as its platform independent, language independent, simple and easy extensible.

Here is a simple scenario of how SOAP procedures can be used. When a correctly formatted call sent to a web service enabled website with the data ranges needed for a search, the site could then return a formatted XML document with all the required results and associated data. These could then be integrated directly into a third-party site.

2.2.2 Limitation

Although in many business industries today, SOAP approach is widely used as a way of exposing their web services, it still has limitations towards business process used. Several major technical disadvantages are exposed below.

✧ **Complexity**

Whatever are SOAP, WSDL and WS-* stack, its technology is very complex. This imposes a high entry level for small to medium size businesses in various industries. For instance, landing industry, there are thousands of such parties are not able to afford such high cost on their web service.

✧ **Not Scalable**

The request and response in SOAP approach is normally stateful, the scalability of the system is a problem. The resource used on session with client reduces the process
capability. Also when it wants to achieve load balancing between different servers, using SOAP approach is much difficult.

✧ **Expose Business Activities Only**

Existing SOA in industries can expose individual business activities well with a set of SOA operations. However, it is far not enough to expose today’s business process. As business process is made up by sets of business activities, it needs the architecture exposes process with sets of operations with sequence and decision points which is not implemented by SOA so far.

✧ **Abuse WWW/HTTP principal**

The original WWW was not explicitly designed for process-centric business applications. It was designed for transporting data representations using URL identifiable resources. However, existing SOA only use WWW/HTTP as a tunneling protocol by encoding method semantics in them. It is designed as fine-grained distributed Remote Procedure Calls (RPC). This means the RPC’s parameters are directly encoded into URL or XML and use HTTP as a black-box transport protocol.

By doing this, many of the WWW/HTTP principles are abused and broken. Subsequently, it is harmful to the overall WWW ULS ecosystem health. Also, this decreases the process visibility and its interoperability.
2.2.3 Tools Support for MDD

In the market, some model driven development (MDD) tools bring the business process development and code generation together, for example, **ActiveVOS**.

![ActiveVOS, business process development MDD tool](image)

**FIGURE 1** ActiveVOS, business process development MDD tool

It is an all-in-one tool for modeling, designing, deploying and running business process. These tools are all SOAP-based, but we see the ability that a powerful MDD tool can bring to the developer.
2.3 Restful Web Service

2.3.1 What is REST?
REST is a style of software architecture for distributed hypermedia systems such as the World Wide Web. The terms “representational state transfer” and “REST” were introduced in 2000 in the doctoral dissertation of Roy Fielding [2], one of the principal authors of the Hypertext Transfer Protocol (HTTP) specification. He summaries the REST as

“REST enables intermediate processing by constraining messages to be self-descriptive: interaction is stateless between requests, standard methods and media types are used to indicate semantics and exchange information, and responses explicitly indicate cachability”

REST principles [3], which are a set of architectural constraints on top of the basic client-server architecture style, and is intended to evoke an image of how a well-designed web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use [4]. Any architecture which is compliant to the design principals of REST can be claimed as Restful.

According to Fielding [2], “software architecture is defined by a configuration of architectural elements – components, connectors, and data – constrained in their relationship in order to achieve a desired set of architectural properties.” REST as an architecture also has its architectural elements, they are listed following:

- **Data element**
  It is information transferred from a component, or received by a component, via a connector. In REST, they are: resource, resource identifier and the Representation of resource.

- **Components**
  It is an abstract unit of software instructions and internal states that provide a transformation of data via its interface. In REST, they are Origin server, gateway, proxy and user agent.
2.3.2 Rest Principal

- **Context free request - stateless**

  Stateless means all the request to server is self-descriptive, a request contain all the information to make the server to understand the request, put into another words, statelessness is the idea that any server for holding one resource should be able to handle any request from the clients, it does not have to be the same server as the one who earlier receive the request. In the REST, the servers do not hold any session information with the client.

- **Unified API**

  In order to manipulate the resource, components (the server and client) use HTTP verbs to communicate and exchange the representations of the resource. The most common verbs are: POST, PUT,
GET, DELETE, which aligns nicely with CRUD (CREATE, RETRIEVE, UPDATE, DELETE) from the database operation.

- **POST** a representation to create the representation of resource.
- **PUT** a representation to update the existing representation of resource.
- **GET** a representation by given URL.
- **DELETE** a representation by given URL.

- **Resource**
  Each piece of interested information can be exposed as the resource which is processed via the unified API, such as business process instances, tasks, and states.

- **Representation**
  Each resource can have multiple representations, such as in the format of XML, HTML, PDF, JSON, that is, use different formats to convey the same resource which is processable by different client.

- **URL**
  Resource is uniquely exposed as URL, there are no two resources with the same URL on the WWW. The relationship between resource URL, HTTP verbs and representation of resource is depicted on Figure 1.

### 2.3.3 Restful Web Service

According to the unique architecture and principals of Restful web service we discussed above, REST has its own advantage in the use of web service opposed to SOAP. Several major benefits are listed below.

- Support caching therefore largely improves the loading capability of the server.
- Stateless means better scalability of the server, server can handle more number of requests, and also it means we can perform loading balance between servers. Therefore if double the number of servers to handle the request, it also double the capability of the processing power, each of the server now can handle another part of requests which originally is sent to one server.
Provide long term compatibility due to the backward and forward compatibility of the HTTP protocol.

Compare to RPC, URL is human-friendly, easy to discovery, transmitted, scripted and bookmarked.

We have noticed that there are many successful web applications which are built using REST principals, for example, Flicker API. Amazon S3 provides an REST API which is much more popular than its SOAP one (85% of the usage is through the REST interface). But all these applications are focusing on data manipulation and exposure [5][6]. And SOAP-based approach has become the majority in most complex web applications. To model the complex business process by REST seems to be non-trivial, and that may due to the conceptual gap between REST and business process[3], where REST is more focus on the resource and its representation, and so-called resource oriented[7]. On the other hand, the Business process takes more consideration on its collaboration between tasks and message flow between different processes, and because of this gap, currently there are not existed any development tools to be able to model the Restful business process, and generate the code artifact for the developer. Therefore, in order to apply the REST principals in a wider problem domain, we need find a way to reduce the gap.

2.3.4 Lack of tool support
While the SOA has various mature Model Driven Development (MDD) tools, it is not the case for REST. However, there is evident proved REST is more and more popular, especially in the recent years. It is showed that having a mature MDD tools for REST is emergent.
2.4 Resource Orientated Business Process

2.4.1 ROA

Resources Oriented Architecture (ROA) is a specific set of guidelines of an implementation of the REST architecture. The ROA is a way of turning a problem into a Restful web service: an arrangement of URIs, HTTP, and XML that works like the rest of the web. It describes Restful architectures in general. There are several concepts and properties are included in the Resource Oriented Architecture refers to business process. Four of the most important features are listed below.

1. **Resources**
   A resource is anything important enough to be referenced as a thing in business process. Usually, a resource is something that can be stored on a computer and represented as a stream of bits. In business process, the resource can be the process, process case, task, task state and message.

2. **URL**
   The URL is the name and address of a resource. It is the most essential of a resource. A resource and its URL ought to have an intuitive correspondence. By definition, no two resources can be the same. However, a resource may have one or more URL. Every URL designates exactly one resource. If it designated more than one, it would not be a Universal Resource Identifier.

3. **Addressability and statelessness**
   Addressability and statelessness are two of the features of ROA. An application is addressable if it exposes the interesting aspects of its data set as resources. Since resources are exposed through URL, an addressable application exposes a URL for every piece of information it might conceivably server. Statelessness means that every HTTP request happens in complete isolation. In the business process example, when the client makes an HTTP request, it includes all information necessary for the server to fulfill that request. The server never relies on information from previous requests. If that information was important, the client would have sent it again in this request.

4. **The uniform interface**
   In ROA, there are only four basic operation methods are supported in HTTP. Those are GET, PUT, POST and DELETE. Detail example shows how the uniform interface is used in the business process will be exposed at the 2.4.2.
2.4.2 Business Process in Restful Way

The execution of business processes will finally affect on the business resource, it either creates some resource or transfers the state of resource from one to another, is this sound familiar? This is the principle of REST, we see an example that how we apply the REST principle on the business process, which is resource centric.

![Business Process Diagram](image)

**FIGURE 3** A simple business process diagram, all these tasks are dealing with job application.

There are three tasks in the process, one is create a job application then followed by submit the job application, and then review job application.

- **Resource**
  Obviously, all these tasks are manipulating the resource, in this case is the job application.

- **Unified interface**
  Then we apply the methods in the unified interface of HTTP, such as **GET**, **POST**, **PUT**, and **DELETE** to manipulate the job application.
  1. **GET**, to get the representation of the job application with {id}.
  2. **POST**, to do a state transfer of job application with {id} from “created” to “submitted”.
  3. **PUT**, to modify the job application with {id}.
  4. **DELETE**, to delete the job application with {id}.

We see that, by using nice URL rather than one like [http://www.nicta.com.au/soap](http://www.nicta.com.au/soap), which put all other information into a SOAP envelop wrapped by HTTP protocol, we can
use a simpler way to achieve the same like SOAP, and it gives meaningful URL, cachability, simplicity and scalability, most importantly, it aligns with HTTP itself perfectly, it is Restful.
SECTION III DEVELOPMENT

3.1 Design

3.1.1 BPMN Modeler
The Eclipse “SOA Tools BPMN Modeler” is initially designed by INTALIO [8], it allows business users to draw business process diagram, and has recently been donated to the Eclipse. Therefore it has been open-sourced, and this provides us an opportunity to extend the existing BPMN Modeler rather than redesign one with the similar base functionality – drawing the model, thus we can focus on extends its functionality rather than reinventing the same wheel. This BPMN Modeler is based on eclipse project EMF [12] and GMF [13], which EMF is used to design the meta-model object and GMF is used to map between these models to graphical representation.

3.1.2 Extension Points
The beauty of eclipse plug-in mechanism is that, instead of modifying the source code of the modeler, we can use extensions to extend the existing plug-ins, by this approach, not only result in a more loose coupled modular design but more flexible deployment of plug-in in a on-demand manner (people who has BPMN modeler already that need extra functionalities just only need to install corresponding plug-ins, no need to re-install the whole things). The following extension points either provided by eclipse itself or BPMN modeler are used:

- **org.eclipse.stp.bpmn.diagram.EAnnotationDecorator**
  By extending this extension point, we can add GUI representation to the annotation. See figure 4.

- **org.eclipse.ui.views.properties.tabbed.propertySections**
  By extending this extension point, we can add new property tab, see figure 5, 6.
org.eclipse.ui.views.properties.tabbed.propertyTabs

By extending this extension point, we can add content to the property tab. See figure 7.

![Property Tab]

FIGURE 7 The content of a property tab.

org.eclipse.ui.views

By extending this extension point, we can add new Eclipse view. See figure 8.

![Context View]

FIGURE 8 The context view which used to edit the process fragment.
By extending this extension point, we can add new eclipse perspective. See figure 9.

![Bpmn Perspective which put all related tools together for business process development.](image)

By extending this extension point, we can add new Action to the eclipse. See figure 10.

![New menu item and its action](image)

### 3.1.3 Overview of Plug-ins

Based on the BPMN modeler, we add the new functionalities to the existing plug-ins via providing another 5 plug-ins, the main reason for separating them to 5 plug-ins is, as mentioned in 3.1.2, a more modular design, following are the 4 plug-ins:

<table>
<thead>
<tr>
<th>Plug-in name</th>
<th>Qualified name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotation Plug-in</td>
<td>org.eclipse.stp.bpmn.nicta.extension.annotation</td>
<td>All annotation related classes, property tab, its content, decorator and internationalization.</td>
</tr>
<tr>
<td>Queryer Plug-in</td>
<td>org.eclipse.stp.bpmn.nicta.extension.codegen.query</td>
<td>Queryer used to query the diagram model, get the necessary information out of the model.</td>
</tr>
<tr>
<td>BPMN Perspective Plug-in</td>
<td>org.eclipse.stp.bpmn.nicta.extension.perspective</td>
<td>Defines a perspective that put all related tools together.</td>
</tr>
<tr>
<td>Context View</td>
<td>org.eclipse.stp.bpmn.nicta.extension.view.w.context</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Defines a context view that used to manipulate the process fragment.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TABLE 1** Plug-ins that add extra functionalities

The relationship between these plug-ins and the BPMN modeler is shown in the next figure.
FIGURE 11  The architecture design of plug-ins, they are dependent on the BPMN modeler, the code generation engine use the model queryer to query the business process diagram.
3.2 Implementation

3.2.1 Knowledge Base and Choice of the Library

3.2.1.1 SWT and JFACE

SWT [9] is an open source widget toolkit for use with the Java platform, it was originally developed by IBM and is now maintained by the Eclipse Foundation.

JFACE [10] brings model view controller (MVC) programming to the SWT, it is built on top of SWT, it is defined by the Eclipse project as “a UI toolkit that provides helper classes for developing UI features that can be tedious to implement.”

People know java probably heard of AWT and SWING, they are the default GUI implantation of java platform, but they may not heard of SWT and JFACE, they are the default GUI implantation of Eclipse platform, the reason for not using AWT/SWING for the GUI development are:

- It is slow, and not responsive.
- Its representation is not consistent with Eclipse.

On the other hand, SWT and JFACE bring the convenience to us, its well-designed API and various existing widget support, give us the better productivity and result in a much more professional GUI representation.

3.2.1.2 EMF and GMF

Although we do not redesign the meta-model of the BPMN Modeler and the graphical mapping between the meta-model to its representation, we still need to re-use the model object, therefore understanding of its meta-model of the BPMN Modeler is the basic requirement.

3.2.1.3 JET

We need to generate code from models, there are several alternatives available. First we can just write the output of a class which generating text into files, then these files as the generate code, but this approach is not scalable, which it put hard coded text into class, when the generated text become complex, it will be difficult to modify such class. What we need is a loose couple approach, that the model objects being pass to the translation engine, with a set of templates, then output files generated. The template is easier to modify, and by this approach, it can generate any text-based file, such as xml, source code, html, etc.
Freemarker [14] and JET [15] are both the choices, they are all template engine, which used to generate text output, we choose JET to be our code generation engine, because JET is a sub-project of EMF, also it does not need too much configuration to use. By the JET, people can use a JSP-like syntax (subset of JSP syntax), that makes it easy to write templates that express the code you want to generate.

### 3.2.1.4 EMF Model Query

We know that the BPMN Modeler is using EMF meta-model to describe the structured data, and the business process diagram drew by the BPMN Modeler is persisted by two xml files, we can query the xml file programmably, but these xml files are very complex and querying such files is not efficient, instead, it is much convenient to query the models via Model Query [16](MQ), when the BPMN Modeler editor open, it will build a Java memory models on top of the two xml files, these models is highly structured, and the MQ can query such models via Object Constraint Language(OCL) [17], using this analogy, now the diagram has became something like database, and we write OCL to query the models, just like we use SQL to query the database.

### 3.2.1.5 Restlet Framework

Although Java Servlet API can be used in a Restful way with careful reengineering and encapsulation of the existing API, but we consider the Servlet API is not born for REST, instead, we consider the Restlet Framework [11], it is an open source REST framework for the Java platform. The design principle of this framework is compliant with what Roy Fielding describes in his paper, and its API is very resource centric, also it supports URL template which is intensively used in Restful application. What more, it is a totally different API separated from the Servlet API, it can be ran in standalone mode or in a Servlet container with adapter, and its performance and scalability is comparable to the Servlet API. Our generated code will be put into a Java project with Restlet Framework dependency resolved, currently we use Restlet v1.1 release, and we make it ran in standalone mode.

The minimum dependency of Restlet framework is

- `org.restlet.jar`, the Restlet core API.
- `com.noelios.restlet.jar`, the Noelios [8] implantation of Restlet API.
- `org.restlet.ext.json_2.0.jar`, the JSON representation which is added through an extension to the Restlet framework.
- `org.simpleframework.jar`, `com.noelios.restlet.ext.simple_3.1.jar`, these jars provide ability that makes Restlet Framework ran in standalone mode.
3.2.2 Add Annotation on Business Process

The plug-in `org.eclipse.stp.bpmn.nicta.extension.annotation` is used to allowing the user to fill in the Restful annotation on various models in the BPMN diagram, in order to provide the information for the code generator. In another word, this plug-in develops the GUI part of the implementation on the BPMN diagram. The GUI can be divided into two parts, the annotation we show on the diagram and the annotation we show in the properties tag. Two parts are developed in two separated packages which are describing in the 3.2.3.1.

The detail annotations on the BPMN diagram is achieved by user input on the property tag of the `EObject` in the diagram when that particular object is selected. The plug-in supports the annotations add in the following five `EObject`, as only this five EObject participate in code generation. Each different property tag has corresponded different UI for user to input which will be detail discuss in Property Tag User Interface (3.2.3.2). The five different Restlet annotations are: (bracket inside is the EObject of BPMN Diagram)

1. Task Annotation (Activity)
2. State Annotation (Sequence Edge)
3. Role Annotation (Pool)
4. Role Annotation (Lane)
5. Process Annotation (BPMN Diagram)
3.2.2.1 Structure

In the implementation code structure, this plug-in includes three major packages.

![Package Explorer](image)

**FIGURE 12** All classes and packages of `org.eclipse.stp.bpmn.nicta.extension.annotation` plug-in

- **org.eclipse.stp.bpmn.nicta.extension.annotation**
  
  As other plug-in in this project, this package includes the activator class which controls the entire plug-in life cycle. Also, the other classes in this package are used as common parameter naming mapping, interact with plugin.xml and the Internationalization for the further extension development. The Internationalization class will be detail discuss in 4.3.

- **org.eclipse.stp.bpmn.nicta.extension.annotation.decorator**
  
  The classes include in this package are used to add the decorations on the basic BPMN diagram directly. There are two decorations added on the diagram which are the blue diamond indicator and the tooltip of it. The indicator on the EObject in the diagram indicates the Restlet annotation information is enabled. It is activated by the Enable check-box in the property tag which will be discussed later soon.

- **org.eclipse.stp.bpmn.nicta.extension.annotation.properties**
  
  The classes in this package are used to describe the property tag on the EObject which need to be annotated. Each property tag is described by two separate classes called `nameSection.java` and `nameTabFilter.java`. 
◆ **nameSection.java**

This class describes detail GUI of the property tag of named **EObject**.

◆ **nameTabFilter.java**

This class acts as a filter, so that the property tag of the particular **EObject** will be displayed only when user selected this **EObject** in the BPMN diagram. Otherwise, this tag will be hided.

### 3.2.2.2 Property Tag User Interface

There are five **EObject** can be annotated in the BPMN diagram. So there are five different property tag user interfaces. In this section, five different annotations UI will be detail discussed. However, the common function and field among these five interfaces will be discussed first.

#### 3.2.2.2.1 Common functions and fields

◇ **General Property Annotation Life Cycle**

When the user selected one of the **EObject** in the BPMN diagram, all the Tab Filter (**nameTagFilter.java**) will be active to check if the **EObject** is the one they want. If that is, the particular property tag will display as one of the property tag in the Properties view named *Restlet*.

At this stage, if the user clicks on the tag, all the annotation attributes will be initialized in the BPMN XMI file and ready for user to input details. All the details the user inputted will be save as attributes in the XMI file. When the user selects this particular **EObject**, the attributes will be recalled in the property annotation tag. All the annotation attributes in the XMI file are called at the later code generation stage by the query plug-in.
All the annotations are initialized as attributes in XMI, shown in the Annotations Tag.

Enable check box / Annotation Validation

When all the essential annotation of one EObject has been input by user, user can select if the input information is ready to involve in the code generation by clicking on the Enable check box which locates at the top left of the properties annotation tag.

Once the Enable check box is selected by user’s mouse event, the validation is activated. The validation will check all the essential fields (essential fields are indicated by "*" behind the tag name) in the property tag to ensure all of those have been filled by user. If any one of the essential field is empty, an error message will be shown beside the Enable check box says “Required field cannot be left blank”. Also, beside the empty field, a small error icon will indicate which field needs to be filled by user. In this case, the Enable check box still left empty.
FIGURE 15  The blue diamond on each EObject indicates the Restlet annotation is enabled by user

If all the essential fields have passed the validation, the Enable check box is selected successfully. Also, there is a small blue diamond will attach as a decoration on the down left corner of the EObject to indicate this EObject has been enable Restlet annotation.

❖ Supplementary Information
As this Restlet annotation is still the prototype, the annotation fields may not fully cover the essential information in REST. By the reason of this, at the bottom of each property tag, there is a Supplementary Information text field for user to supplement any necessary information to the Restlet.

FIGURE 16  Supplementary Information is implemented as text filed in the bottom of each properties annotation tag
3.2.2.2.2 Task (Activity) Annotation Properties Tag

The Task Annotation Properties Tag will be invoked in the properties field once the user selects one task in the BPMN diagram. In this Tag, five fields are included. They are Enable, Task Name, Representation Media Type, Allowed HTTP Method, and Supplementary Information.

- **Enable**
  
  As we discuss in the 3.2.3.2.1, Enable check box is included in every Annotation Properties Tag to show all the annotation is valid and enable to involve in the code generation if this check box is selected. At this stage, a blue diamond will be shown on the EObject in the BPMN diagram.

- **Task Name**
User needs to give a task name to identify the task. The name of the task is essential indicated by a “*” following the Task Name label. The task name in one BPMN diagram must be unique.

**Representation Media Type**

The Representation Media Type list lets user to select the available media type for this particular task to transfer via HTTP. User must select at least one representation media type in order to pass the validation.

For example, if the media type is selected as `TEXT_XML`. This lets web browser know it can render the activity as XML document.

![New Media Type](new_media_type.png)

**FIGURE 18** New Media Type input dialogue for user to input new media type

The media type list includes all the REST supported media types for user to select at this stage. We cannot ensure the list has included all the possible media types in the future. By the reason of this, an “Add Button” and a “Delete Button” are added. At the time when user cannot find out the proper media type in the list, they can click on the “Add Button” (Green Plus button). An input dialogue will pop out labeled “New Media Type” for user to input customize new media type. After it finished, the new media type will add in the unselect list for user to select. On the other hand, when the user want to delete the media type in the unselect list, he can simple select that media type, click on the “Delete Button” (Red Cross button). The media type will be deleted from the unselect list.

After the user has selected the media types to the selected list, they will be used in later code generation stage.

**Allowed HTTP Method**

There are only four methods for user to select as Allowed HTTP Method. They are Get, Post, Put, and Delete.

**Supplementary Information**

As we discuss in the 3.2.3.2.1, Supplementary Information text field is included in every Annotation Properties Tag to let the user input any supplementary information of the task or for future supplementary use.
3.2.2.2.3 State (Sequence Edge) Annotation Properties Tag

The State Annotation Properties Tag will be invoked in the properties field once the user selects one state (Sequence Edge) in the BPMN diagram. In this Tag, five fields are included. They are Enable, State Name, Representation Media Type, Allowed HTTP Method, and Supplementary Information.

The entire layout in State Annotation Properties Tag is similar to the task’s one. However, as a state in Restlet at this stage, the default Allowed HTTP Method is “Get” only.
3.2.2.2.4 Role (Pool) Annotation Properties Tag

The Role (Pool) Annotation Properties Tag will be invoked in the properties field once the user selects one role (Pool) in the BPMN diagram. In this Tag, three fields are included. They are Enable, Role Name, and Supplementary Information.

In this tag, only the Role name is essential for user to input. Actually, it should not have any annotation on the pool. The Role annotation should be annotated on the Lane in the BPMN diagram. However, if the pool only contains one lane, the lane will usually be omitted. For this case, the user can only annotated the Role on the pool. In the other word, if the pool contains at least one lane, this tag will be unavailable. Also, an error tag will be shown up asking the user to add the annotation on lane as following figure.

![Figure 21](image1.png)

**Figure 21** Full view of the Role Annotation Properties Tag

![Figure 22](image2.png)

**Figure 22** The error message shows up in the pool annotation tag when there is at least one lane in the pool. All the other fields in this tag are disabled and hided.
3.2.2.2.5 Role (Lane) Annotation Properties Tag

The Role (Lane) Annotation Properties Tag will be invoked in the properties field once the user selects one role (Lane) in the BPMN diagram. In this Tag, three fields are included. They are Enable, Role Name, and Supplementary Information which is exactly the same as the Pool Tag. Only the Role Name is essential for user to input.

One more feature to notice, if the lane is added after the user has annotated the Role information on the pool tag. All the information on the pool will be copied into lane tag. Also the blue diamond enable indicator will be displayed on the button left corner of lane instead of displayed on the pool. All the information on the pool will be removed and disable.
### 3.2.2.2.6 Process (BPMN Diagram) Annotation Properties Tag

The Process Annotation Properties Tag will be invoked in the properties field once the user selects the whole BPMN diagram (that is click on any of the blank part in the BPMN diagram). In this Tag, six fields are included. They are Enable, Process Name, Host, Process Representation Media Type, Case Representation Media Type, and Supplementary Information.

#### Host

![Full view of the Process Annotation Properties Tag](image_url)

**FIGURE 24** Full view of the Process Annotation Properties Tag
Host text box is only available in the Process Annotation Properties Tag. It indicated all the resources’ URL root in this process.

- **Representation Media Type**
  There are two representation media type list. One is used to describe the process resource, and another is used to describe the case resource.

### 3.2.3 Query the model

The plug-in `org.eclipse.stp.bpmn.nicta.extension.codegen.query` is used to query the models which represent the business process diagram. First, let us have a look that what level of abstraction of our queryer works on. As mentioned in 3.2.1.4, the business process diagram is persisted in the xml files, currently there are lots of technology existed can be used to query the xml document, such as **XPath** and **XQuery**, but these xml files are very complex and huge in size, also what we need is much more than just getting the value of several simple node, we need to traverse in between different models, see the figure, for example, determining the number of outgoing edges of A, and what is followed by that outgoing edge. A queryer that works on the object level will be much easier to handle, then a traverse in between them become method invocation such as

```java
// For demo
B = A.getOutgoingEdges()[0].getTarget()
C = A.getOutgoingEdges()[1].getTarget()
```

Fortunately, the EMF models have already been organized in this way. And the EMF sub-project called Model Query (MQ) is working on this level of abstraction, therefore we can use that facility to query the model, also the MQ supports the Object Constraint Language (OCL), we also can use the MQ to query the models with the help of OCL.

This plug-in has two main classes, `OclConditions` and `BpmnQueryer`.```
➢ Class **OclConditions**

This class is used together with **OclConditions.properties**, the property file used to define the OCL query conditions, we provide query condition via properties file rather than hard code it into class, therefore it is easier for people to change the query condition, and that can be done without re-compiling the plug-in.

➢ Class **BpmnQueryer**

This is the API exposed to other plug-ins, in another word, if you want to use the queryer functionality, you need to import this class, and use this class to query the business process model.

![Class diagram of BpmnQueryer](image)

FIGURE 27 Class diagram of BpmnQueryer

The get methods here serve as a convenient method to get the targeted business process models, you can supply any valid OCL String as argument to performQuery() method to query the model. (Note, we just use the very basic functionality of OCL, and OCL is much powerful than this, in ideal case, you can get what you want from the business process models by just supplying careful designed OCL without further filtering the result by programming code. We take another approach, because of our limitation on knowledge in OCL, we get a range of objects and filter what we need by code)

Let look at its usage by a simple example.

```java
// Example to use BpmnQueryer
// to print all the names of the Task, the business process diagram is in fig
// the result is A B C
Collection<Activity> activities = BpmnQueryer.getAllTasks();
```
For(Activity activity : activities) {
    System.out.print(activity.getName() + " ");
}

3.2.4 An Eclipse Context View
In the plug-in org.eclipse.stp.bpmn.nicta.extension.context.view, we define a context view which is used to manipulate the process fragments, in order to adding new view in Eclipse, first we have to extend the extension point, and in this case, it is the org.eclipse.ui.views. This view uses the JFACE library, it provides GUI programming in a Model-View-Controller way, its representation and its models are decoupled via different Providers. Let first look at the model objects of the view, see the figure, it is clear that a Root can contain arbitrary numbers of Task, and a Task can contain arbitrary numbers of Context, since the context view is indeed a tree viewer, the models in the tree view is also designed like a tree structure. The mapping between the models to their representations is shown in figure.

FIGURE 28 Class diagram of the model objects

FIGURE 29 Mapping between the models to their representations
The link between the models and their representations is via the Providers, there are two types of Provider, one is used to provide the content such as name of the node in tree, it is ContextViewContentProvider, another one is used to provide the label of the node, in this case, the label is the icon of the node, it is ContextViewLabelProvider, register these Providers in the createPartControl() method of the ContextView.

```java
// Register the Providers to the view
treeViewer.setContentProvider(contentProvider);
treeViewer.setLabelProvider(labelProvider);
```

The context view also has several view actions, add context, modify context name, and remove context, they are Action objects added to the tool bar manager.

```java
// Add actions to tool bar manager
IToolBarManager toolbarManager = getViwSite().getActionBars().getToolBarManager();
toolbarManager.add(addContextAction);
toolbarManager.add(changeContextNameAction);
toolbarManager.add(removeContextAction);
```

One important thing to be mentioned here is what stuff actually added to the set of resource, when a context has been added to the model? First, the new added context should be reflected on the GUI, therefore a Context object need to be created in memory, and let the GUI to refresh. Second, it needs to add the name of the context to the annotation, because we have to get what we have added to the model after restarting the Eclipse. Third, the content of the context is saved to an xml file, the file location is where the current editing project at. In similar case, when modifying the name of context and removing context, the corresponding memory object, annotation, and xml file need to be properly handled.

### 3.2.5 An Code Generation Engine
The idea of code generation engine is pretty similar to the Java Servlet Page, let us first take an example.
/\ // Class Book
public class Book {
    private String name;

    .....  

    public String getName() {
        return this.name;
    }
    .....  

} 

// The JSP code
Hi, the book name is <\%book.getName()\%>

What does the JET do for us is similar to the JSP translation process, we write the template, and give the translation engine a model object which contains the value we want it shown in the file.
The models which will be passed to the code generation engine is defined under package org.eclipse.stp.bpmn.nicta.extension.codegen.restlet.model, they are actually very similar to JavaBean, which hold value. The JetUtil is the encapsulation of the JET template engine, it contains methods to generate the content from templates using the models, and write the content to the target Eclipse project. There are two source of information which will be put into the generated file, one is from our business process model another is from a Code Generation Wizard, at the moment, the wizard is very simple which only have 3 pages, the first two is the same one of Java Project Create Wizard, the third page is used to supply the Author information, such as the name of Author and one’s company.
FIGURE 32  Code generation wizard, top left is the first page, top right is the second page, and bottom left is the third page.

We put all the templates under 
org.eclipse.stp.bpmn.nicta.extension.codegen.restlet/templates/ folder, modifying the templates is similar to modifying the JSP.

The Restlet framework dependency jars is under 
org.eclipse.stp.bpmn.nicta.extension.codegen.restlet/jars/ folder, these jar will be added to the newly created project by using the Code Generation Wizard. The code to add jars to a java project is in the afterJavaProjectCreationSetup() method of CodegenDetailsWizard.
// Add jars

.....

RestletCodegenPlugin.addJar(javaProject, RestletCodegenPlugin.PLUGIN_ID, "jars/com.noelios.restlet.jar");

.....

3.2.6 An Eclipse perspective

In the plug-in org.eclipse.stp.bpmn.nicta.extension.perspective, we define an Eclipse perspective which is used for business process development, it combines all the functionalities we developed together, user in this perspective can conduct business process drawing, annotation configuration, progress fragment definition, and code generation.
Defining a new perspective in Eclipse is very easy, in the class `BpmnPerspectiveFactory`, it creates the perspective layout. Place the views and editors somewhere on the layout by specifying its relative position to some component, and a ratio indicate the percentage of area the view or editor occupy.

// Place the context view
IFolderLayout topLeftDown = factory.createFolder("topLeftDown",
   IPageLayout.BOTTOM,
   0.5f,
   "topLeftUp");
topLeftDown.addView("org.eclipse.stp.bpmn.nicta.extension.view.context");

FIGURE 33  The Bpmn Perspective.
3.3 Evaluation

In order to make a clear demonstration to the reader who concerns this report and development, we have recorded a demonstration video. The demonstration video is about 9 minutes long. All the major functionalities we have implemented have shown up in that demonstration, you can find the demonstration on the DVD. The example is based on the Job application business process BPMN diagram.
SECTION IV  FURTHER EXTENSION

4.1 Business Process Workflow Pattern Detector

The workflow pattern can be used to describe the behavior of business process. There are existed totally 21 patterns, which 9 of them can be recognized in our plug-in, they are:

1. Sequence
   ![Sequence Diagram]

2. Parallel Split
   ![Parallel Split Diagram]

3. Synchronization
   ![Synchronization Diagram]

4. Exclusive Choice
   ![Exclusive Choice Diagram]

5. Simple Merge
   ![Simple Merge Diagram]

6. Multiple Choice
   ![Multiple Choice Diagram]

7. Synchronizing Merge
   ![Synchronizing Merge Diagram]
8. **Arbitrary Cycles**

![Diagram of arbitrary cycles]

9. **Implicit Termination**

![Diagram of implicit termination]

The pattern information is saved in each `ActivityModel`.

```java
// Workflow pattern
public class ActivityModel extends BaseModel {
    ....
    private WorkflowPattern workflowPattern = null;
    ....
}

In our generated code, we do not use any information from workflow pattern, but for further extension, it probably can be used for process fragment generation.
4.2 Generation of Text-based File

In the 3.2.6, we have looked how the code generation engine works, by giving models and templates, any text-based files can be generated. Say if you want to generate an xml file, then just write the xml template.

```
<% // Demo of another template
BpmnDiagramModel model = (BpmnDiagramModel)argument;
String processName = model.getProcessName();%
<task>

<name><%=processName%}}</name>

</task>
```

4.3 Internationalization

At the moment, we only developed the GUI in English, but we have architected in such a way that adding another GUI language is just as simple as writing a set of properties files. Let us take an example, if you want to add another language, say Chinese, then put a property file called `AnnotationPlugin18n_zh_CN.properties` under the same package of `AnnotationPlugin18n.properties`, in the `AnnotationPlugin18n_zh_CN.properties`, you write a set of key-value pairs which the keys are same with the `AnnotationPlugin18n.properties`, and value are the translated language in Unicode, then if your system locale is `zh_CN`, then the GUI will be shown in Chinese. Here is a list of such property files.

- `AnnotationPlugin18n.properties`
- `RestletCodeGenPlugin18n.properties`
- `ContextViewPlugin18n.properties`
SECTION V      CONCLUSION

In this report, it first discusses the SOAP-base web service, and its limitation. Then we discuss another approach to develop web service, i.e. applying the REST principal in business process, which is resource centric, also we see the emerging need of a toolset to support Restful business process development. Then we show the development of a set of Eclipse plug-ins to extend the existing functionality of the BPMN Modeler, including how we design and implement the plug-ins.
BIBLIOGRAPHY


APPENDIX A HOW TO EXTEND THE WORK

This section is for reader who want to extend the work we have done, reader who just want to install the plug-ins and use the software please go to Appendix B.

1. You need Eclipse 3.4 with EMF GMF etc dependency resolved, and JDK 6.
   There is a convenient way to install the Eclipse 3.4 and with dependent plug-ins installed.
   Get the eclipse package Eclipse Modeling Tools from http://www.eclipse.org (you can find on the DVD, eclipse-modeling-ganymede-SR1-incubation-win32.zip), in this package, it contains a collection of Eclipse Modeling Project components, including EMF, GMF etc, which is required to develop the BPMN Modeler, what more, it has all the source codes associated with EMF and GMF, therefore you can view the documentation easily when you develop the plug-ins.

   **Eclipse Modeling Tools (includes Incubating components) (300 MB)**
   This modeling package contains a collection of Eclipse Modeling Project components, including EMF elements. It includes a complete SDK, developer tools and source code. Note that the Mode components, as indicated by feature numbers less than 1.0.0 on the feature list. More...
   Downloads: 65,886

   ![The Eclipse package, Eclipse Modeling Tools](figure35)

2. Get the source code of the BPMN Modeler.
   You need to check out the SVN repository, therefore you need some tool to do that. Get the TortoiseSVN from http://tortoisessvn.tigris.org.
   Right click anywhere on your desktop, choose TortoiseSVN -> Repo-browser
FIGURE 36 The button of Repo-browser

Type http://dev.eclipse.org/svnroot/stp/org.eclipse.stp.bpmn/trunk, in the URL dialog, after you click on OK button, it will show something similar to figure.
Let us check out the plug-ins one by one, first right click on `org.eclipse.stp.bpmn`, then choose Checkout, specify the Checkout directory, say `c:\bpmn\org.eclipse.stp.bpmn`, then click OK button, do this for the

- `org.eclipse.stp.bpmn.diagram`
- `org.eclipse.stp.bpmn.validation`
- `org.eclipse.stp.bpmn.doc`
- `Org.eclipse.stp.bpmn.edit`

as well, also changes the corresponding checkout directory.

3. Import these source codes into your Eclipse workspace
FIGURE 38  The button of Import

Then choose General-> Existing Projects into Workspace, choose the root directory, say c:\bpmn\.
FIGURE 39 Import dialog

After the import done, you will see the Package Explorer view as, figure
4. We need to import our extension source code (source code can be found on the DVD, under folder source code). Similar to importing the BPMN Modeler source code, import the following plug-ins.

```java
org.eclipse.stp.bpmn.nicta.extension.annotation
org.eclipse.stp.bpmn.nicta.extension.codegen.query
org.eclipse.stp.bpmn.nicta.extension.codegen.restlet
org.eclipse.stp.bpmn.nicta.extension.perspective
org.eclipse.stp.bpmn.nicta.extension.view.context
```

After this done, your **Package Explorer** should be viewed as figure

5. The required plug-ins are all imported, and then you can begin extending the work.
APPENDIX B INSTALLATION GUIDE

1. Get the package Eclipse Modeling Tools from http://www.eclipse.org, we provide one copy on the DVD. (Alternatively, you can get the All-in-One package on the DVD, unzip it, then you can use it, then skip the step 2)

![Eclipse Modeling Tools](http://www.eclipse.org)

Eclipse Modeling Tools (includes Incubating components) (300 MB)
This modeling package contains a collection of Eclipse Modeling Project components, including EMFT elements. It includes a complete SDK, developer tools and source code. Note that the Mode components, as indicated by feature numbers less than 1.0.0 on the feature list More...

Downloads: 65,886

FIGURE 42 The Eclipse package, Eclipse Modeling Tools

2. Go the plugins folder of Eclipse, say F:\eclipse\plugins

Put the following jars into the folder (these jars can be found on the DVD)
- org.eclipse.stp.bpmn_1.0.0.200810192314.jar
- org.eclipse.stp.bpmn.diagram_1.0.0.200810192314.jar
- org.eclipse.stp.bpmn.edit_1.0.0.200810192314.jar
- org.eclipse.stp.bpmn.validation_1.0.1.200810192314.jar
- org.eclipse.stp.bpmn.nicta.extension.annotation_1.0.0.jar
- org.eclipse.stp.bpmn.nicta.extension.view.context_1.0.0.jar
- org.eclipse.stp.bpmn.nicta.extension.perspective_1.0.0.jar
- org.eclipse.stp.bpmn.nicta.extension.codegen.restlet_1.0.0.jar
- org.eclipse.stp.bpmn.nicta.extension.codegen.query_1.0.0.jar

3. Check the installation by Help -> About Eclipse Platform -> Plug-in Details

Check whether the plug-in installed or not, see figure

![Check the installation](http://www.eclipse.org)

FIGURE 43 Check the installation
APPENDIX C  DEMONSTRATION

Please see the demonstration on the CD.