Diplomarbeit

Automated Generation of JIAC AgentBeans from BPMN Diagrams

Fachbereich *Agententechnologien in betrieblichen Anwendungen und der Telekommunikation* (*AOT*)

Prof. Dr.-Ing. habil. Sahin Albayrak
Fakultät IV Elektrotechnik und Informatik
Technische Universität Berlin

Vorgelegt von: **Petrus Setiawan Tan**

Betreuer: Dipl.-Inform. Tobias Küster

Petrus Setiawan Tan
Matrikelnummer: 213933
Stettiner Str. 9
10243 Berlin

Die selbstständige und eigenhändige Anfertigung dieser Diplomarbeit versichere ich an Eides statt.

Ort, Datum	Petrus Setiawan Tan

Abstract

Zusammenfassung

Acknowledgement

Inhaltsverzeichnis

1 Introduction **1**

1.1 Motivation 1

1.2 Goals 2

2 Background **3**

2.1 JIAC(Java-based Intelligent Agent Componentware) 3

2.2 BPMN(Business Process Modelling Notation) 4

2.3 VSDT(Visual Service Design Tool) 5

2.4 JET(Java Emitter Templates) 6

3 Mapping BPMN to Jiac AgentBeans **7**

4 Implementation of the Mapping **9**

5 Fazit **11**

Literaturverzeichnis **13**

1. Introduction

In this chapter, we will start by introducing the motivation and the goals of this work.

1.1 Motivation

Over the past few years more and more software developers have been adopting the principle of *Model Driven Engineering*(MDE) where they no longer focus on writing programs but on creating a set of models which define the software. By modelling the software, the developer creates documents that provides an abstract view of the software system, independantly from the plattform or a specific programming language, making it understandable for non experts i.e. the stakeholders as well as applicable in different plattforms. These models will then be the basis for the implementation. A significant number of the so called CASE (Computer Aided Software Engineering) tools have been developed to support this methodology. Beside supporting the developers in creating and editing the models, most of these CASE tools are also equipped with transformation features that allows us to transform the model into text or even executeable Programs, thus increasing efficiency in the software development process. We can say that the real benefits of MDE lies in the transformation. By providing a set of rules in transforming the model into code, we can create standardized programs, accelerate development time and minimize faults in writing the code. By Providing a

An example of such tools is the ***Visual Service Design Tool (VSDT)***, developed by Tobias Küster in scope of his Diploma Thesis back in 2007, which provides the transformation of BPMN (Business Process Modelling Notation) to BPEL(*Business Process Execution Language*) and JIAC (Java-based Intelligent Agent Componentware) framework. In the scope of this work, a plugin to VSDT will be developed to enrich it's transformation feature with a code generator that will transform BPMN models into executeable Java Code, or JIAC AgentBeans to be more specific.

1.2 Goals

The main goal of this work is to develop an eclipse plugin as an extension to VSDT to enrich its transformation features with a new transformation from BPMN to Java Code or JIAC AgentBeans to be more specific. Because it is nearly impossible to put all implementation details into the model, and to anticipate the possibility, that the generated code will be edited manually, considerations has to be made, such that conflicts should not occur when the transformation is called to a code that has been edited manually.

2. Background

In this chapter, we will discuss the backgrounds of the transformation that should be developed. We will start with JIAC (the target of the transformation), BPMN (the model that is being used), followed by VSDT (the existing modeling and transformation framework to JIAC that should be extended), and JET (the technology that will help us to implement the transformation).

2.1 JIAC(Java-based Intelligent Agent Componentware)

JIAC [1, 3, 5, 10] is a Java-based agent architecture and framework that was developed to simplify the development of software agents. The framework supports the entire software development process of a software agent system, from the design, implementation, and deployment of the system.The JIAC framework provides features such as FIPA compliant communication, Believe-Desire-Intention (BDI) reasoning, strong migration, web-service connectivity and others. Further it provides high security (Common Criteria EAL3, certified by the Federal Office for Information Security of Germany, BSI) and advanced accounting mechanisms, making it suitable for the use in industrial and commercial applications. The JIAC framework comes with a runtime environment and a toolsuite for the creation of agents.

A typical JIAC Application consists of *AgentNodes*, *Agents*, and *AgentBeans* (see also Figure 1). An *AgentNode* is a Java VM providing the runtime infrastructure for agents, such as discovery services, white and yellow pages services, communication infrastructure. A JIAC application consists of one or more AgentNodes. Normally, there is one AgentNode per physical machine. The AgentNode comes ready-to-run, but can be adapted to the needs of the target environment and can also be extended by additional components, so-called AgentNodeBeans.
Each AgentNode may run several *Agents*. Agents provide services to other agents and comprise lifecycle, execution cycle and a memory. An agent can use infrastructure services in order to find other agents, to communicate to them and to use their services. Skills and abilities of the agent can be extended by so-called AgentBeans.

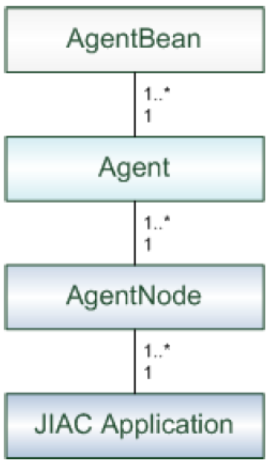

Abbildung 2.1: Jiac Basic concepts and their structural relationships [1]

AgentBeans is the mean to implement the functionality. They are plugged into agents and provide services (so-called Actions) to other agents. AgentBeans have a lifecycle.

2.2 BPMN(Business Process Modelling Notation)

BPMN [2] is a standard Notation for modelling business processes, initially published by the BPMI which is later adopted by the OMG(Object Management Group). A business process diagramm (as seen in Figure 2) can be compared to UML's activity diagram.

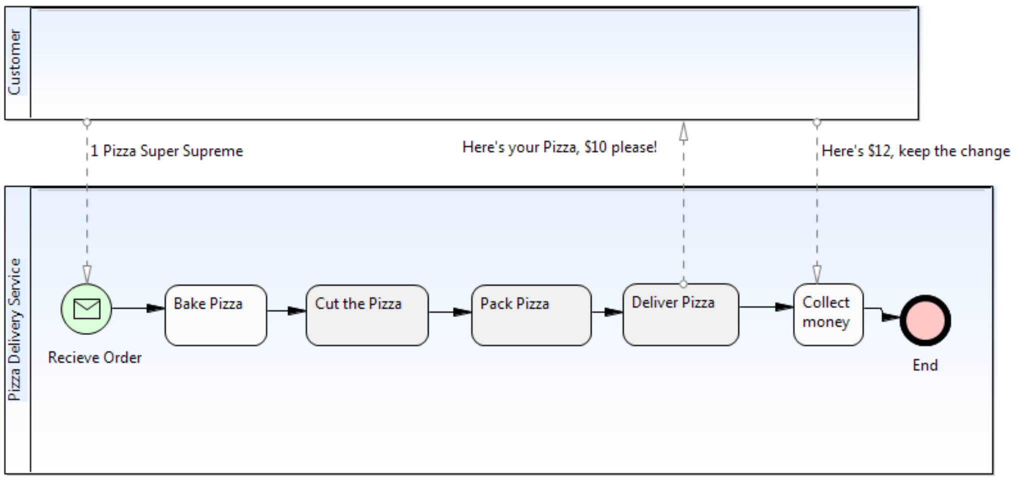

Abbildung 2.2: A simple Business Process Diagramm

BPMN was made to provide a notation that is understandable by all business users, creating a brige for the gap between the business process design and the process implementation. With the mapping of BPMN to Agents, we hope to be able to increase the spreading of the multi agent systems in the business world.

To provide a rough overview on how Agent technology can support the implementation of Business Processes let us take a look on this mapping example of BPMN elements to Agents. A Pool in a Business Process Diagram can represent an Agent, which are able to communicate with other agents (another pool) through messages (represented with the BPMN MessageFlows). Agents can react to Events. To fully implement the transformation a detailed mapping is needed and will be developed in the scope of this work.

2.3 VSDT(Visual Service Design Tool)

The VSDT [6, 8, 9, 10] is a well-equipped BPMN editor that is independent of any specific target language. Thus, while the usual transformation to BPEL, as well as other transformations, is included, the VSDT can easily be extended with additional export functionality targeting other languages. It was developed in early 2007 as a Diploma Thesis at the TU Berlin and since then it has been continuously extended. Since it's initial development, VSDT's transformation framework is designed to be extensible and reuseable. This allows the development of a new transformation to be easier. For this purpose the transformation process is subdivided into several stages:

1. *Validation*: Validate the input model.
2. *Normalisation*: Prepare the input model for transformation.
3. *Structure Mapping*: Convert the input model to a block-like structure.
4. *Element Mapping*: Perform the actual mapping, create target model.
5. *Clean Up*: Remove redundancies, improve readability, etc.

The Validation and Normalisation and Structure Mapping are mostly independent from the target language. In most cases the standard mapping provided for these stages are reuseable, which makes it possible to implement a new transformation by specifying the element mapping only. Figure 3 shows the UML Class Diagram of the transformation famework with the example transformation to BPEL.

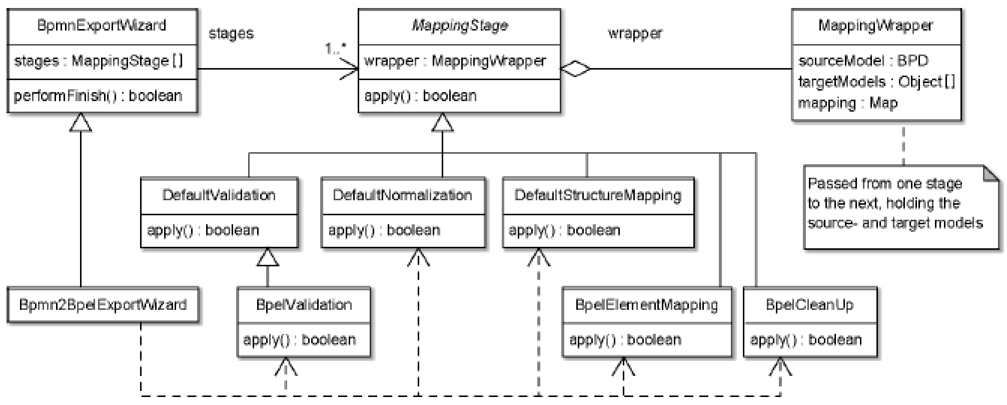

Abbildung 2.3: Essential classes of the transformation framework, including the BPEL case.[9]

2.4 JET(Java Emitter Templates)

JET[7] is a code generating framework, developed as a part of the Eclipse Modelling Project [4]. JET uses templates to transform the model into any kinds of code artifacts such as java, html, xml or even plain text.

JET templates uses a JSP-like syntax which makes it easy to write and understand. A set of JET templates is called transformation. It is possible to build this transformation with a main template which acts as a visitor and runs through the model, and this main template will then use other templates which handle a specific element of the model. For example, in UML to Java transformation you can have special templates that handles the package, class, variables and methods.

3. Mapping BPMN to Jiac AgentBeans

4. Implementation of the Mapping

5. Fazit

Literaturverzeichnis

[1] CC ACT, DAI-Labor, TU-Berlin. *JIAC - Java Intelligent Agent Componentware*, 06 2010. Version 5.1.0 Manual.

[2] Object Management Group. Bpmn specification v2.0. `http://www.omg.org/spec/BPMN/2.0/`.

[3] Benjamin Hirsch, Thomas Konnerth, and Axel Heer. Merging agents and services the jiac agent platform. In Amal El Fallah Seghrouchni, Jrgen Dix, Mehdi Dastani, and Rafael H. Bordini, editors, *Multi-Agent Programming:*, pages 159–185. Springer US, 2009. 10.1007/978-0-387-89299-3_5.

[4] Eclipse Modelling Project Homepage. `http://www.eclipse.org/modeling/`.

[5] JIAC Homepage. `http://jiac.de/?id=35`.

[6] VSDT Homepage. `http://jiac.de/?id=30`.

[7] JET. `http://www.eclipse.org/modeling/m2t/?project=jet#jet`.

[8] Tobias Küster. Development of a visual service design tool providing a mapping from BPMN to JIAC. Master's thesis, Technical University of Berlin, Faculty of Electrical Engineering and Computer Science, 2007.

[9] Tobias Küster and Axel Heßler. Towards transformations from BPMN to heterogeneous systems. In Massima Mecella and Jian Yang, editors, *BPM2008 Workshop Proceedings*, 2008.

[10] Tobias Küster, Marco Lützenberger, Axel Heßler, and Benjamin Hirsch. Integrating process modelling into multi-agent system engineering. In Michael Huhns, Ryszard Kowalczyk, Zakaria Maamar, Rainer Unland, and Bao Vo, editors, *Proceedings of the 5th Workshop of Service-Oriented Computing: Agents, Semantics, and Engineering (SOCASE) 2010*, 2010.