Institut für Parallele und Verteilte Systeme
Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Diplomarbeit Nr. 3169

Machine-Learning-basiertes Framework für eine Geschäftsprozesssimulation

Michael Panos

Studiengang:	Informatik
Prüfer:	Prof. Dr.-Ing. habil. Bernhard Mitschang
Betreuer:	M.Sc. Florian Niedermann
begonnen am:	04.04.2011
beendet am:	04.10.2011
CR-Klassifikation:	H.2.4, H.3.3, H.4.1

Inhaltsverzeichnis

Abkürzungsverzeichnis

API – Application Programming Interface

AWT – Abstract Window Toolkit

BIA – Business Impact Analysis

BPM – Business Process Management

BPMN – Business Process Modeling and Notation

CPN – Colored Petri Net

CWM - Common Warehouse Metamodel

DB - Datenbank

dBOP – deep Business Optimization Platform

dBOPSim - deep Business Optimization Platform Simulation

DOM – Document Object Model

DWH – Data Warehouse

GUI – Graphical User Interface

HTML – Hypertext Markup Language

HTTP – Hypertext Transfer Protocol

IBM – International Business Machines (Firmenname)

ID – kurz für identification

IT – Informationstechnologie

JAX-WS - Java API for XML Web Services

JDBC – Java Database Connectivity

OASIS – Organization for the Advancement of Structured Information Standards

OMG – Object Management Group

ProM – Process Mining (Tool)

SOA – Service-oriented Architecture

SOAP - Simple Object Access Protocol

UML - Unified Modeling Language

W3C – World Wide Web Consortium

WS-BPEL – Web Services Business Process Execution Language

WSDL - Web Services Definition Language

XML – eXtensible Markup Language

XOR – eXclusive OR (Logik-Element)

XPDL – XML Process Definition Language

Abbildungsverzeichnis

Tabellenverzeichnis

Listing-Verzeichnis

Kurzfassung

Geschäftsprozesse erfreuen sich immer größerer Beliebtheit in Unternehmen. Durch den Einsatz von Informationstechnologien kann ein Unternehmen seine Kosten senken und Erträge steigern. Dabei spielt die Geschäftsprozessoptimierung eine wichtige Rolle. Veränderungen der Prozesse können aber ein Risiko darstellen. Um das Risiko von Fehlern zu minimieren kann eine Simulation des Geschäftsprozess ausgeführt werden.

Auf Basis der deep Business Optimization Platform (dBOP) wurde ein Simulationsmodell konzipiert, das eine sehr realitätsnahe Simulation ausführt. Dabei verschafft man sich durch das integrierte Data Warehouse aus Prozessdaten und operativen Daten einen erheblichen Vorteil gegenüber herkömmlichen Simulationsmöglichkeiten.

In dieser Diplomarbeit wurde ein Framework entwickelt, das eine Geschäftsprozesssimulation ausüben soll, die mit Hilfe von Data Mining Algorithmen aufgebaut wird. Es wurde ein Konzept entwickelt, das aus dem vordefinierten dBOP-Modell ein Simulationsmodell aufbaut. Dieses Konzept wird mit seinen Phasen erläutert. Außerdem wurde eine prototypische Implementierung entwickelt.

Anschließend findet sich eine Evaluation der gewählten Geschäftsprozesssimulation auf Basis eines Beispiels. Es wird anhand dieses Testprozesses aufgezeigt, wie das Simulationsmodell entsteht und welche Besonderheiten man beachten muss. Anschließend wurden die Ergebnisse der Simulation analysiert und mit realen Daten und Simulationsdaten, ohne erweiterte Datenbasis, verglichen.

Abstract

Business processes are becoming more and more popular in the corporate environment. With the service of information technology enterprises can lower their cost and increase their income. Therefore business process optimization plays an important part. But changes in processes are involved with a certain risk. To reduce the risk of making mistakes there can be an execution of a simulation of a business process.

On top of the deep Business Optimization Platform (dBOP) we designed a simulation model that could execute a simulation very close to reality. Thereby the integrated Data Warehouse consisting of process and operational data give us a huge edge over conventional simulation models.

In this diploma thesis we designed a framework that would execute this business process simulation with the help of data mining algorithms. Therefore a concept is developed that builds a simulation model from the predefined dBOP model. This concept is described with all its phases. Additionally a prototype has been implemented.

After that an evaluation of the simulation model was made based on an example process. It is described by this example how the simulation model originates and how we have to treat the specifics. Following this, the results of the simulation are analyzed with real data and simulation data of a conventional simulation model.

1 Einführung

Die Informationstechnologie hält immer weiter Einzug in Unternehmen und ihrer Wertschöpfungskette. War es anfangs nur ein Mittel zum Zweck, haben Unternehmen heute erkannt, welches Potenzial in Computer-Systemen und ihrer richtigen Nutzung stecken. Ein gut automatisierter Prozess oder auch nur ein Web-Shop können die Erträge von Unternehmen steigern. Dabei spielt die Prozessoptimierung eine wichtige Rolle. (1)

Die richtige Gestaltung und Ausführung solcher Geschäftsprozesse ist ein Gebiet für sich. Um zu sehen, wie sich Geschäftsprozesse in gewissen Situationen verhalten, kann man dank der heutigen IT-Systeme Simulationen solcher Geschäftsprozesse ausführen. So lassen sich neu entwickelte Prozesse testen, bevor sie eingesetzt werden, oder mögliche Grenzwert-Szenarien durchspielen, bevor sie eintreten, um mögliche Risiken zu entdecken.

Im ersten Kapitel dieser Diplomarbeit wird zu kurz der Hintergrund des Geschäftsprozessmodells vorgestellt, der einer Simulation unterzogen wird. Anschließend wird neben der Aufgabenstellung, die allgemeinen Ziele und Inhalte dieser Diplomarbeit genauer beschrieben. Zum Abschluss der Einführung gibt es einen Überblick des Aufbaus der Diplomarbeit.

1.1 Hintergrund

Diese Diplomarbeit hat als Hintergrund, dass immer mehr dieser Geschäftsprozesse verbessert und optimiert werden müssen. Mit dem Ursprung des Business Process Reengineering (2) ist an der Universität Stuttgart das deep Businnes Optimization Platform (dBOP) Projekt entstanden. Um zu zeigen, dass dieses Projekt und sein Modell effizient arbeiten und relevante Ergebnisse liefern, soll ein Simulationsmodell erstellt werden.

Im dBOP Projekt werden nämlich nicht nur Prozessdaten gepflegt, sondern auch mit operativen Daten zusammengeführt (3). Diese Zusammenführung soll ein besseres Verständnis der Geschäftsprozesse liefern und somit besser optimiert werden können. Damit eine Optimierung, z.B. durch eine Automatisierung einer „Human-Task"-Aktivität im Prozess, gerechtfertigt werden könnte, wären eine Simulation der Automatisierung und ein Vergleich mit den schon vorhandenen realen Daten (vom manuellen Prozess) ein Beleg für eine Optimierung.

1.2 Aufgabenstellung

Auf Grundlage des dBOP Projekts, das in Kapitel 3 näher erläutert wird, soll ein Framework entwickelt werden, dass durch die integrierte Datenbasis ein Geschäftsprozesssimulationsmodell aufbauen soll und dieses dann auch ausführt. Dabei sollen Machine-Learning-basierte Verfahren eingesetzt werden, um mit Hilfe der Datenbasis eine nahezu realistische Simulation durchführen zu können.

Als Teilaufgaben wurden definiert:

- Simulationsmodell: Übernahme des dBOP Prozessmodells für die Simulation mit Definition der zu simulierenden Prozesseigenschaften
- Modellbildung: Zur Bildung des integrierten Simulationsmodells sollen Mining-Verfahren angewendet werden
- Modellanwendung: Das Simulationsmodell soll während der Geschäftsprozessausführung bereitgestellt werden können.
- Evaluation: Das bereitgestellte Simulationsmodell soll durch gewählte Merkmale evaluiert werden.

Für alle Teilaufgaben gilt: Es soll zuerst ein konzeptionelles Modell erarbeitet werden und in dieser Diplomarbeit dokumentiert werden. Anschließend sollen die entwickelten Konzepte implementiert werden.

1.3 Ziele und Inhalt

Das Ziel dieser Diplomarbeit ist somit die Entwicklung und Implementierung eines Frameworks, der eine Simulation eines speziell entwickelten Prozessmodells mit Hilfe von Data Mining gestützten Algorithmen ausführen soll. Dabei muss mit Bedacht an dem vorhandenen Prozessmodell ein Simulationsmodell erarbeitet werden, dass die Besonderheiten des Prozessmodells beachtet und eine realitätsnahe Simulation ergibt.

Die Simulation soll aufzeigen, dass das vorher erarbeitete Prozessmodell einen signifikanten Vorteil von herkömmlichen Prozessmodellen bietet. Durch die erweiterte Information im Prozessmodell sollte eine weitaus realitätsnähere Simulation möglich sein.

Außerdem soll diese Diplomarbeit zeigen, dass die analysierten Ergebnisse der zusammengeführten Daten eine (automatische) Optimierung zulässt und durch die Simulation belegt, dass die Optimierung ohne signifikante Fehler zum gleichen Ergebnis kommt.

1.4 Aufbau dieser Diplomarbeit

Nach dieser kleinen Einführung ist die Diplomarbeit folgendermaßen aufgebaut: In Kapitel 2 werden die Grundlagen für diese Diplomarbeit erläutert. Dabei werden nicht nur einzelne relevante Technologien vorgestellt, sondern auch allgemein gültige Definitionen, wie für das Geschäftsprozessmanagement. Anschließend werden in Kapitel 3 Vorarbeiten, wie das dBOP Projekt und Modell, und verwandte Projekte vorgestellt, die ein ähnliches Ziel verfolgten und auch für diese Arbeit interessant sind. In Kapitel 4 und 5 kommen wir dann zum eigentlichen Framework mit dem konzeptionellen Modell, sowie den Implementierungsvorstellungen dessen. Neben der genutzten Software werden die Lösungen der verschiedenen Teilaufgaben erläutert. Außerdem wird ein Testprozess simuliert und die Ergebnisse in Kapitel 6 dargestellt. Dabei werden die Ergebnisse des Frameworks analysiert und das gewählte Evaluationsmodell begründet. Im letzten Kapitel kommen wir zur kurzen Zusammenfassung der Arbeit und diskutieren Weiterentwicklungsmöglichkeiten.

2 Grundlagen

In diesem Kapitel werden die grundlegenden Definitionen und Technologien vorgestellt, die für diese Diplomarbeit relevant sind. Dabei wird im Laufe dieses Kapitels ein kleiner Einblick im Verlauf des Titels dieser Diplomarbeit („Machine-Learning-basiertes Framework für eine Geschäftsprozesssimulation") gegeben. Neben relevanten Data Mining Algorithmen werden Geschäftsprozesse und deren Beschreibung, sowie das Geschäftsprozessmanagement und die Simulation dieser erläutert. Das ganze wird von Web Services abgerundet, die im Zusammenhang mit Geschäftsprozessmanagement stehen und auch in dieser Diplomarbeit genutzt werden.

2.1 Geschäftsprozessmanagement (BPM)

Vor nicht allzu langer Zeit waren Unternehmen in Funktionen und Abteilungen aufgeteilt, die ihre Aufgaben in der Wertschöpfungskette unabhängig voneinander abarbeiteten. Das ist sowohl aus IT Sicht als auch aus der betriebswirtschaftlichen Sicht eines Unternehmens nicht von Vorteil. Durch die Trennung in Abteilungen kann es dazu kommen, dass jede Abteilung ihre eigenen IT Systeme anfordert, sodass in einem Unternehmen viele heterogene Systeme genutzt werden. Eine spätere Analyse von Prozessen und der Zusammenführung der Daten für eine Business Analyse ist somit nur schwer möglich. Einen solchen Gesamtüberblick über eine ganze Wertschöpfungskette und seine anschließende Verbesserung „kann zu Leistungs- und Qualitätssteigerungen und somit zu Wettbewerbsvorteilen führen." (4) Aus diesem Grund ist heutzutage die Prozess-orientierte Unternehmensgestaltung ein Muss für jedes größere Unternehmen. Dafür ist das Geschäftsprozessmanagement zuständig, dass „sich mit dem Dokumentieren, Gestalten und Verbessern von Geschäftsprozess und deren IT-technischen Unterstützung [befasst]." (4)

Das Dokumentieren und Gestalten geschieht im Normalfall mit Modellierungssprachen der IT, wie BPMN (5). Diese Modellierungssprachen werden in Kapitel 2.2 näher erläutert. Diese Modelle und ihre Analyse kann dann nicht nur „als Grundlage für die Prozessverbesserung (…) genutzt werden", sondern auch für die technische Implementierung durch die IT-Abteilung (4).

2.1.1 Betriebswirtschaftliche Sicht

Aus betriebswirtschaftlicher Sicht soll die prozess-orientierte Unternehmensgestaltung die Leistungen des Unternehmens in messbare Kennzahlen umwandeln. Diese Messung von „Kosten, Zeiten, Mengen, Ressourcen usw." ist für Entscheider im Unternehmen wichtig, um eben diese Entscheidungen zu treffen, die dem Unternehmen z.B. Kosten sparen und es somit wettbewerbsfähiger machen.

2.1.2 Technische Sicht (IT-Geschäftsprozessmanagement)

Aus technischer Sicht ist das (IT-)Geschäftsprozessmanagement dafür zuständig, die ausgearbeiteten Geschäftsprozesse mit technischen Hilfsmitteln zu unterstützen. Dabei sollen Prozesse weitestgehend automatisiert werden. In der IT wird deshalb auch oft von Workflow-Management gesprochen. Die Systeme, die diese Arbeitsabläufe darauf technisch umsetzen, werden Workflow-Management-Systeme genannt (6).

„Workflows sind ein verbreitetes Hilfsmittel zur Implementierung domänenspezifischer Geschäftsprozesse." (4) Sie bestehen wie Geschäftsprozesse aus Aktivitäten und ihrer Verknüpfung. Die Workflow-Management-Systeme sorgen dafür, dass die Aktivitäten in ihrer Reihenfolge und nach jeweiligen Bedingungen ausgeführt werden. Dabei stehen verschiedene Standards zur Implementierung bereit, wie WS-BPEL, was in Kapitel 2.2 näher erläutert wird.

Die einzelnen Aktivitäten in Workflows werden in zwei Klassen aufgeteilt. „Automation Workflows" sind automatisierte Aktivitäten, die „wenn überhaupt, externe Kommunikation über technische Schnittstellen zu

Fremdapplikation ausführen.", also insbesondere Services aufzurufen, siehe Kapitel 2.5 über Web Services. „Human-Centric Workflows" oder auch „Human Tasks" sind Aktivitäten, die einer Interaktion des Benutzers durch eine GUI bedürfen (4).

Um eins vorweg zu nehmen, das Ziel der Simulation wird später sein, diese Workflows bzw. die einzelnen Aktivitäten durch ein Simulation Service zu ersetzen.

2.1.3 Geschäftsprozessoptimierung

In (1) wird erwähnt, dass die Geschäftsprozessoptimierung helfen soll, „die bisherigen Kosten und Bearbeitungszeiten zu senken." Dabei sollen eben die Prozesse kürzere Wege gehen und dadurch auch an Qualität gewinnen. Für die Prozessoptimierung gibt es verschiedene Ansätze, die man verfolgen kann. Abbildung 2.1 zeigt diese:

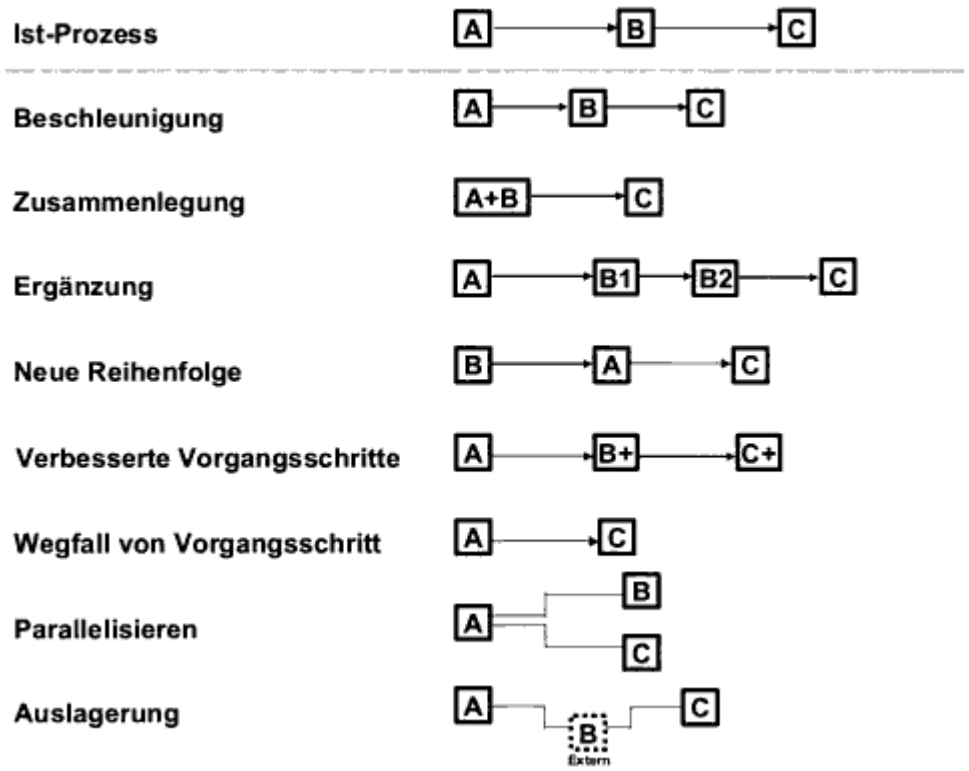

Abbildung 2.1: Ansätze zur Geschäftsprozessoptimierung (1)

2.2 Modellierung von Geschäftsprozessen

Für eine systematische Beschreibung von Geschäftsprozessen wurden in den letzten Jahren verschiedene Modelle und Sprachen entwickelt, die dies

umsetzen sollen. Dabei gibt es zwei Standards, die in der Unternehmenswelt hauptsächlich genutzt werden. BPMN ist eher für die graphische Modellierung und die Fachbereiche vorgesehen, die den Geschäftsprozess entwickeln, wobei BPEL für die Implementierung im IT Bereich genutzt wird.

2.2.1 BPMN

BPMN stand für Business Process Modeling Notation, seit BPMN 2.0 steht es laut Spezifikation für Business Process Model and Notation. Entwickelt wurde BPMN von der Object Management Group (OMG), ein Konsortium für die Entwicklung von IT Spezifikationen im Unternehmensumfeld. Neben BPMN entwickelte die OMG u.a. auch die UML (Unified Modeling Language) und das CWM (Common Warehouse Metamodel). Einen schnellen Überblick gibt das Poster der BPM-Offensive Berlin (7). Die genaue Spezifikation findet sich auf der Website (5).

Das Ziel von BPMN ist eine Beschreibung für Geschäftsprozesse für alle Fachbereiche eines Unternehmens bereitzustellen. Diese Beschreibung soll einfach und verständlich sein. Lange Zeit galt BPMN als die graphische Darstellung von ausführenden XML-basierten Prozess-Sprachen wie BPEL, was im nächsten Kapitel erläutert wird. Mit BPMN konnten die Fachbereiche ihre gewünschten Prozesse visualisieren und so für den entwickelten IT-Bereich verständlich machen. Ein Umwandeln von BPMN in eine technische Sprache ist möglich. XPDL (XML Process Definition Language (8)) kann ein vollständiges Mapping von BPMN anbieten, wohingegen ein BPEL-Mapping auf die Haupt-Elemente beschränkt ist.

Die graphische Modellierung von BPMN unterteilt sich laut (4) in vier Klassen. Dies sind die Ablaufelemente, Verbindungselemente, Schwimmbahnen und Artefakte. Abbildung 2.2 zeigt einige Elemente für die vier verschiedenen Klassen.

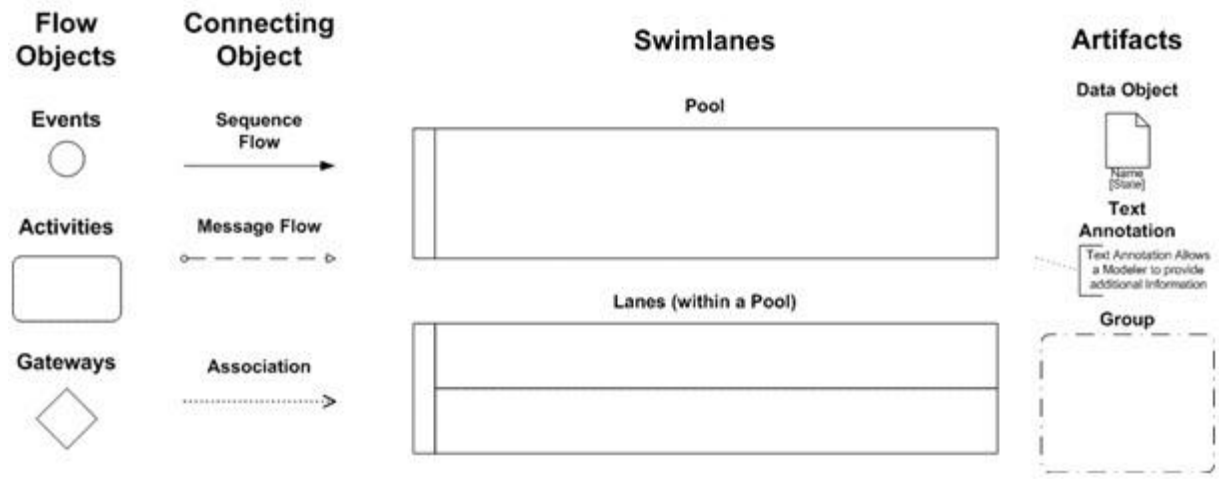

Abbildung 2.2: Haupt-Elemente der BPMN Modellierung (9)

Die Ablaufelemente sind Ereignisse, wie der Start oder das Ende eines Prozesses, Entscheidungen oder Gateways, die den Fluss des Prozesses gestalten können, und die Aktivitäten selbst, die die Teilaufgaben des Prozess beschreiben und lösen. In Abbildung 2.2 sind die Elemente unter „Flow Objects" zu sehen.

Verbindungselemente sind die Elemente, die z.B. die einzelnen Ablaufelemente miteinander verbindet. Auch Artefakte, die später erläutert werden, werden mit verschiedenen Verbindungselementen miteinander verbunden. Man unterscheidet zwischen „Sequence Flow" für den Ablauf des Prozesses oder auch „Message Flow", der für den Informationsaustausch von Artefakten zuständig ist.

Die Klasse der Schwimmbahnen enthält hauptsächlich zwei Elemente: „Pools" und „Lanes", auf Deutsch Schwimmbecken und Schwimmbahnen. Die Pools unterteilen den Prozess in Prozessbeteiligten, die Schwimmbahnen unterteilen die Pools nochmal, um z.B. interne Organisationseinheiten nochmals klarer abzugrenzen. In Abbildung 2.2 sieht man zwei Pools, einen mit nur einer Lane und einen mit zwei Lanes. Durch die gerade vorgestellten Verbindungselemente, können Prozesse über mehrere Pools und Lanes verbunden werden.

Als letztes, aber nicht unbedingt am unwichtigsten, kann man noch Artefakte in den Ablauf einfügen. Artefakte sind z.B. weitere Modellinformation, die die Prozesse noch genauer beschreiben und nicht in die anderen

Klassen eingeordnet werden. Dazu gehören Datenobjekte, die den Informationsaustausch im Prozess verdeutlichen können. Auch eine Gruppierung kann vorgenommen werden, um gewisse Bereiche besser abzugrenzen.

Abbildung 3.13 zeigt den später genutzten Testprozess als BPMN Modell.

2.2.2 WS-BPEL

BPEL steht für Business Process Execution Language, meist auch WS-BPEL, wobei das WS für Web Services steht (Beschreibung von Web Services in Kapitel 2.5). BPEL ist eine XML-basierte Sprache, die für die Implementierung von Workflows auf Workflow-Management-Systemen genutzt wird. Entwickelt von OASIS Standard findet sich auf deren Website auch die Spezifikation (10).

In BPEL werden Geschäftsprozesse und ihre zugehörigen Web Services definiert. Dabei werden meist graphische Editoren genutzt, um BPEL Dateien zu erstellen, da eine manuelle Erstellung sehr mühsam wäre. Dabei kann auch das vorgestellte BPMN Modell genutzt werden. Durch eine Transformation kann das ausgearbeitete BPMN Modell zu einer BPEL Datei umgewandelt werden.

Listing 2.1 zeigt die Grundstruktur einer BPEL Datei. Dabei werden unter <partnerLinks/> die eingebundenen Web Services aufgeführt. Unter <variables/> werden alle für den Geschäftsprozess benötigten Variablen und ihre Metadaten aufgezählt. Anschließend geschieht unter <sequence> der eigentliche Ablauf des Prozess. Dabei stehen verschiedene Aufrufe zur Verfügung. Unter <receive/> werden die Start-Variablen gefüllt. Ein Prozess beginnt mit einem Receive der Input-Parameter. <assign/> ist für die Zuordnung von Variablen, sehr wichtig für die richtige Zuordnung der <reply/>-Variablen, die hier an (externe) Services geleitet werden.

Der später eingesetzte Geschäftsprozess ist mit BPEL auf dem IBM Process Server (wird in Kapitel 5 zur Implementierung vorgestellt) schon vor implementiert und wird somit später für die Auswertung genutzt.

```
<?xml version="1.0" encoding="UTF-8"?>
<process name="Test">
   <partnerLinks/>
   <!-- Eingebundene Dienste -->
   <variables/>
   <!-- Workflow-Daten -->
   <sequence>
      <!-- Workflow-Definition-->
      <receive/>
      <!-- Dienstaufruf zur Variablenfüllung-->
      <assign/>
      <!-- Variablen-Übertrag (Verarbeitung)-->
      <reply/>
      <!-- Variable an Dienst übertragen -->
   </sequence>
</process>
```

Listing 2.1: WS-BPEL Grundstruktur (4)

2.3 Geschäftsprozesssimulationen

Nachdem nun Geschäftsprozessmanagement im Allgemeinen vorgestellt wurde, wird in diesem Kapitel die Simulation von Geschäftsprozessen erläutert. Dabei wird zuerst der Simulationsbegriff in verschiedenen Aspekten betrachtet um die Idee hinter einer Geschäftsprozesssimulation besser zu verstehen.

2.3.1 Simulation

Die Herkunft des Wortes Simulation kommt aus dem Lateinischen simulare und bedeutet: ähnlich machen, nachbilden; nachahmen. Die im Deutschen häufigste Bedeutung von simulieren ist das vortäuschen, wenn z.B. ein Fußballer eine Verletzung simuliert, um Spielzeit von der Uhr zu nehmen. Die zweite Bedeutung laut Duden ist auch die, die in dieser Diplomarbeit genutzt wird: „Sachverhalte, Vorgänge [mit technischen, naturwissenschaftlichen Mitteln] modellhaft zu Übungs-, Erkenntniszwecken nachbilden, wirklichkeitsgetreu nachahmen.“ (11) Ein gutes Beispiel ist die Simulation eines Raumflugs, um Erkenntnisse für die Raumfahrer zu schaffen, damit sie keine tödliche Fehler in der Realität begehen.

2.3.1.1 Computer-Simulation

In der Informatik ist die Simulation bekannt als Computer-Simulation. Durch den Aufstieg des Computers in den letzten Jahren sind Simulationen

mit Hilfe eben dieser immer wichtiger geworden. Dabei werden komplexe, reale Ereignisse in naturwissenschaftliche, bzw. mathematische Modelle umgewandelt. Die erstellten Modelle können mit Hilfe der Computer in Bruchteilen von Sekunden berechnet werden.

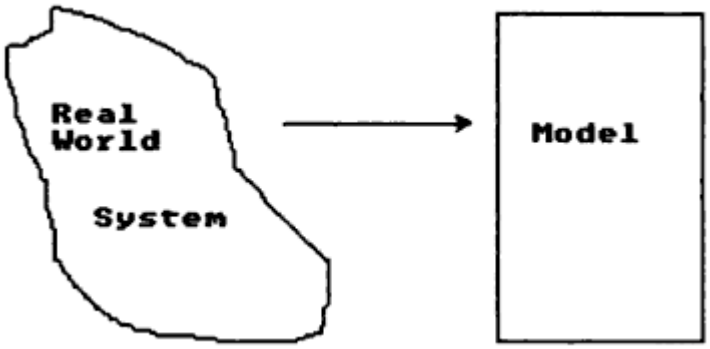

Abbildung 2.3: Modellerstellung für die Simulation (12)

„The main advantage in using simulation is the reduction of risk involved with implementing a new system or modifying an existing one." (12) Auf Deutsch: Der Hauptvorteil einer Simulation ist die Verminderung des Risikos mit der eine Implementierung eines neuen oder der Modifikation eines vorhandenen Systems verbunden ist. Die vorherige Ausführung des Simulationsmodells bevor das eigentlich reale Modell ausgeführt wird, gibt schon früh Erkenntnisse über Fehlplanungen und anderen Fehlern. Die frühe Erkennung von Fehlern durch eine Simulation senkt das Risiko eines Projekts drastisch. Abbildung 2.4 zeigt dies in einem Schaubild.

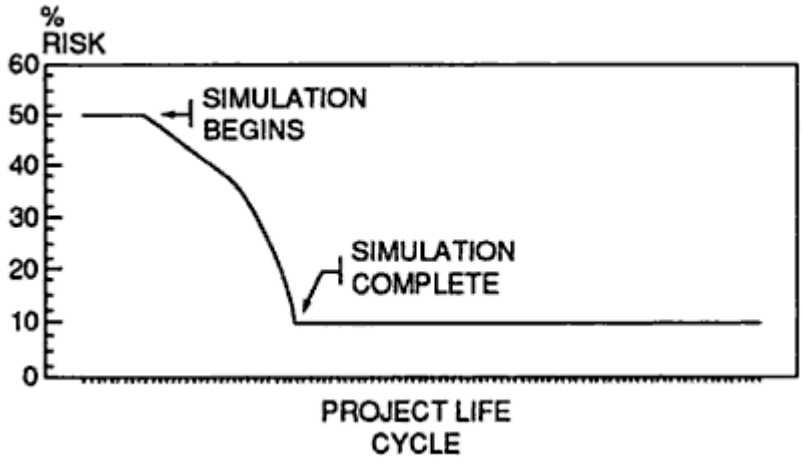

Abbildung 2.4: Beispiel einer Verminderung des Risikos durch eine Simulation (12)

2.3.1.2 Animation

Die Visualisierung der Simulation wird auch als Animation bezeichnet. Dabei wird meist nur ein einzelner Simulationsprozess ausgeführt um den Ablauf der Simulation zu zeigen. Somit wird zwischen Simulation, was mehrere Instanzen durchläuft, und Animation unterschieden (13).

2.3.2 Simulation von Geschäftsprozessen

In (13) wird die „zielgerichtete, experimentelle computergestützte Ausführung von Prozessmodellen […] als Geschäftsprozesssimulation bezeichnet." Die Vorteile einer Geschäftsprozesssimulation besteht aus der Gewinnung von Erkenntnissen zur Laufzeit, „ohne diese real ausführen zu müssen".

Für die Konstruktion der Simulation von Geschäftsprozessen sind verschiedene Modellierungen der Teilbausteine eines Geschäftsprozesses zu wählen. Neben der zu wählenden simulationsrelevanten Attribute sind auch Modellierungen von Wahrscheinlichkeitsverteilungen, Ablaufalternativen, Prozessinstanziierungen und Ressourcenverfügbarkeit zu wählen. Im Folgenden werden diese Modellierungen vorgestellt. Wie genau die zugrunde liegende Geschäftsprozessmodellierung für die Simulationsmodellierung zu benutzen ist, hängt von Fall zu Fall ab. Heutzutage werden aber die meisten Geschäftsprozessmodellierungen auch für eine mögliche Simulation vorbereitet. Kann die vorhandene Modellierung nicht übernommen werden, so kann man zumindest Teile entnehmen und so ein neues Modell erstellen. Dabei ist zu beachten, welche Teile man für die Simulation benutzt und welche nicht. Eine Auswertung wie sinnvoll eine Simulation ist für einzelne Aktivitäten, muss schon im Voraus geschehen (13).

2.3.2.1 Attributierung

Für eine Geschäftsprozesssimulation gilt es einige wichtige Attribute zu beachten. Dazu gehören als Hauptinformationen (13):

- „Verteilung der Bearbeitungsfunktion jeder Funktion" bzw. Aktivität
- Ressourcen, die die Aktivität benötigt

- Verzweigungsregeln, d.h. welche Wege werden im Prozessablauf, wann, warum und mit welcher Wahrscheinlichkeit gegangen
- Häufigkeit der Instanziierung

Natürlich können auch weitere Attribute genutzt werden. Diese Diplomarbeit wird aufzeigen, dass durch weitere hilfreiche Informationsgewinnung eine „genauere" Simulation möglich ist. Dabei ist natürlich die Genauigkeit auch von der Skalierung der benötigten Attribute wichtig, d.h. je detaillierter die Daten zur Verfügung stehen, desto genauer sollte auch die Simulation sein.

Abbildung 2.5 zeigt Beispiele für die Attributierung an den einzelnen Punkten im Geschäftsprozess. Neben den verschiedenen Verteilungen für die Bearbeitungszeiten ist auch eine Wahrscheinlichkeit für Gateways angegeben.

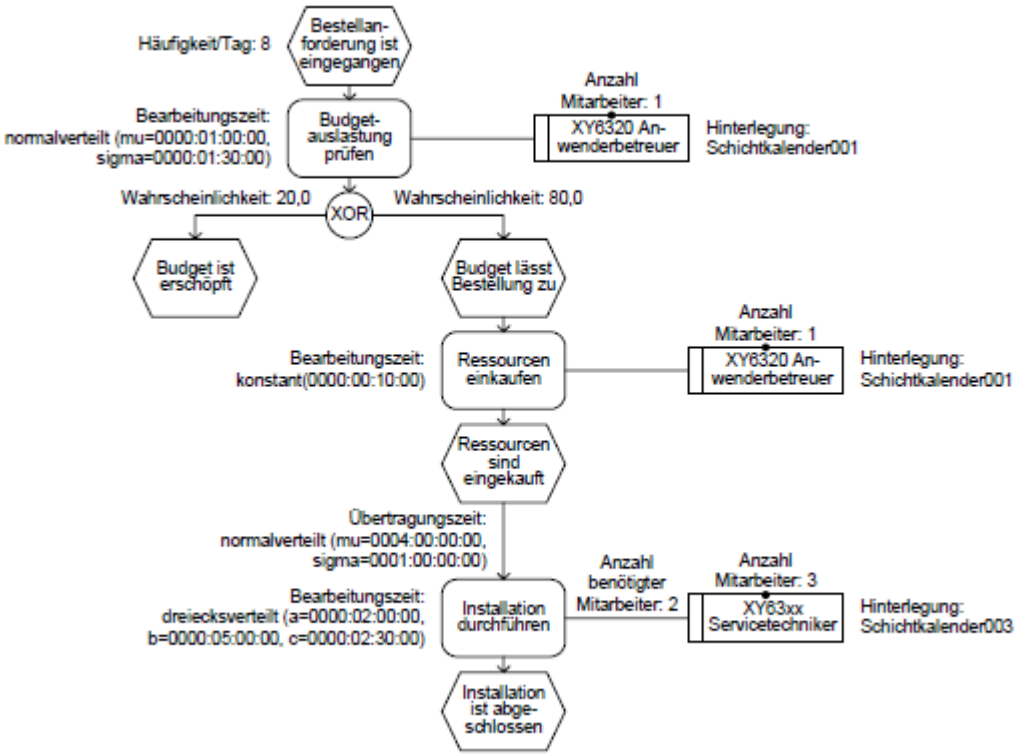

Abbildung 2.5: Beispiel für Attributierung in einer Geschäftsprozesssimulation (13)

2.3.2.2 Wahrscheinlichkeitsverteilungen

Wahrscheinlichkeitsverteilungen werden an einigen Stellen in der Geschäftsprozesssimulation angewendet. Dazu gehören neben Erstellung der zufälligen Input-Größen auch die der Bearbeitungszeit der Aktivitäten. Da-

bei können verschiedenste Wahrscheinlichkeitsverteilungen eingesetzt werden. Abbildung 2.6 zeigt die wichtigsten. Die Wahl der Verteilung hängt von den gegebenen Informationen ab. Stehen nur wenige Informationen zur Verfügung, so bedient man sich mit der Gleichverteilung und gibt nur Mindest- und Maximalwerte an und jeder Wert dazwischen kommt mit der gleichen Wahrscheinlichkeit vor. Hat man einen geeigneten Mittelwert zur Hand und kann die Streuung gut einschätzen, lassen sich die Normal-, die Lognormal-, sowie in seltenen Fällen die Dreiecksverteilung nutzen.

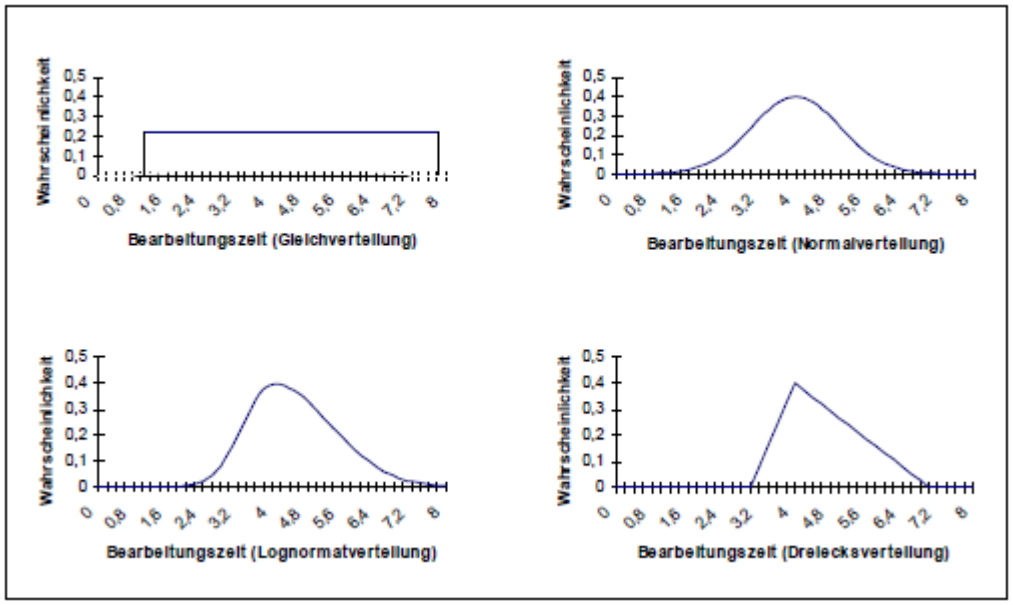

Abbildung 2.6: Beispiele für Wahrscheinlichkeitsverteilungen (13)

2.3.2.3 Ablaufalternativen

Nachdem nun Simulationen für einzelne Aktivitäten und Input-Größen modelliert wurden, müssen auch die Prozesse an sich simuliert werden. Dazu gehören die Ablaufalternativen, die „als exklusive oder inklusive O-der-Verzweigungen modelliert [sind und] (…) für die Simulation zusätzlich Angaben (…) gemacht werden [müssen].“ (13) Für diese Alternativen gibt es einfache und komplexere Lösungsmöglichkeiten. Eine einfache ist die konstante Wahrscheinlichkeit der Verzweigung, wenn der Ablauf „unabhängig vom bisherigen Prozessverlauf erfolgt.“ Die wahrscheinlich bessere

Möglichkeit ist die „Verzweigung auf der Basis von Attributwerten", die häufig in Geschäftsprozessen genutzt werden (13).

Abbildung 2.7 zeigt ein Beispiel einer Input-Größe mit einer Lognormalverteilung. Anschließend wird bei der XOR-Verzweigung auf Basis des Werts entschieden, welcher Pfad fortgeschritten wird und somit welche Aktivität ausgeführt wird.

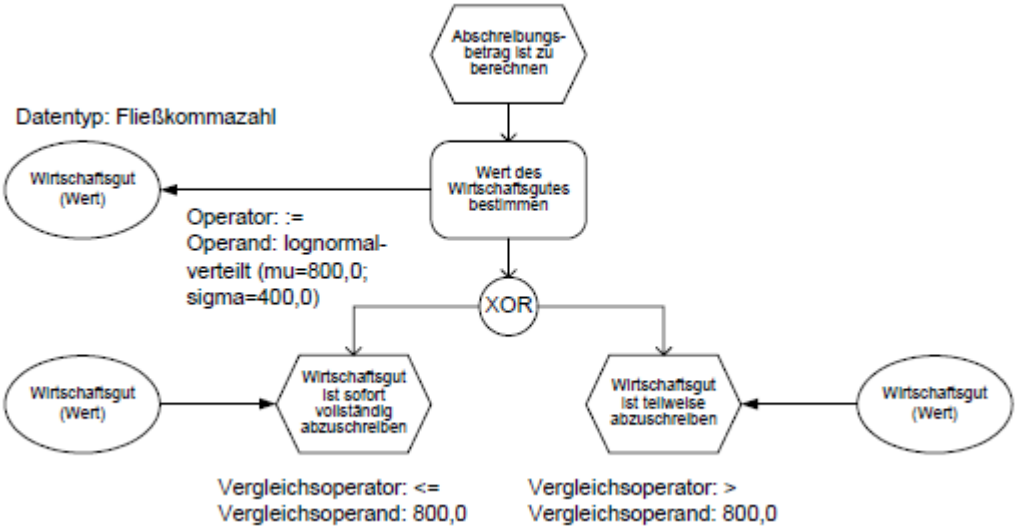

Abbildung 2.7: Beispiel für eine bedingte Verzweigung (13)

2.3.2.4 Prozessinstanziierung

Bei der Prozessinstanziierung geht es um die Häufigkeit des Prozessstars. Dabei kann der Prozessstart nach zwei Möglichkeiten instanziiert werden. Die einfachere Möglichkeit ist die Instanziierung nach einer konstanten Zeit. Hier wählt man ein Intervall, z.B. jede halbe Stunde, in der der Prozess mit seinen simulierten Größen gestartet und durchlaufen wird.

Als zweite Möglichkeit gibt es die stochastische Instanziierung. Dabei werden nach einer Wahrscheinlichkeitsverteilung (z.B. Gleichverteilung auf die Minuten) die Prozessstarts auf den Tag verteilt simuliert (13).

2.3.2.5 Ressourcenverfügbarkeit

Will man noch weiter ins Detail gehen, so muss man auch die verfügbaren Ressourcen in die Simulation mit einbeziehen. So kann eine Art „Schichtplan" eingeführt werden, um zu zeigen „in welchen Zeiträumen eine Ressource zur Verfügung steht." (13) Es muss angegeben werden,

wie viele Mitarbeiter zur Verfügung stehen und zu welchen Zeiten. Wenn dann ein Mitarbeiter gerade an einer Aktivität arbeitet, sollte diese Person dann auch keine weitere bearbeiten können. Dies sollte in der Simulation berücksichtigt werden.

2.3.3 Auswertung der Geschäftsprozesssimulation

Nach der mehrfachen Prozessinstanziierung können die Werte der Prozessdaten zusammengeführt und analysiert werden. Dabei sind folgende Erkenntnisse von großer Bedeutung:

- Aggregierte Auswertungen: Kennzahlen auf den Gesamtprozess bezogen, z.B. Anteil erfolgreich beendeter und nicht erfolgreich beendeter Prozesse.
- Instanzenbezogene Auswertungen: Detaillierte Kennzahlen für jeden Prozess und seine Durchlaufzeiten.
- Ressourcenbezogene Auswertungen: Kennzahlen für die Auslastung der jeweiligen Ressourcen, wie z.B. Mitarbeiter.

Die Daten sollten auch exportiert und für weitere Analysetools zugänglich gemacht werden können. Somit sollen auch weitere Analysen ermöglicht werden, wie z.B. Data Mining Analysen, die im nächsten Kapitel vorgestellt werden (13).

In der Auswertung kann geschaut werden, inwiefern die Simulation mit den erwarteten oder den realen Ergebnissen übereinstimmt. In (13) wurde in einem Fachbeispiel erwähnt, dass zwar die Simulation sinnlos war, da man keine besonderen Ergebnisse erlangt hatte, aber die Auseinandersetzung mit dem Prozess hat zum besseren Verständnis geführt und somit andere Verbesserungen herbeigeführt.

2.4 Data Mining

Data Mining gehört zu den neuesten aufsteigenden Technologien der Informatik. Die große Datenflut, die im Computer Zeitalter auf uns zukommt, bedarf an Techniken diese zu bändigen - dazu gehört Data Mining. Auch bekannt als „Machine Learning" oder „Pattern Recognition"

geht es um den Informationsgewinn aus sortierten und unsortierten Daten. Durch die Hilfe von Computer-Systemen können Strukturen nicht nur schneller erkannt werden, sondern auch andere gefunden werden, die mit dem bloßen Auge nicht zu sehen waren.

Die Informationsgewinnung im Data Mining geschieht durch verschiedene Algorithmen, die im Laufe der Jahre entstanden sind. Zu den bekanntesten gehört das Clustering-Verfahren oder das aus der Mathematik bekannte Regressionsverfahren. In diesem Kapitel werden die für diese Diplomarbeit relevanten Algorithmen vorgestellt. Dabei stand das Buch von (14) zur Hilfe.

2.4.1 Klassifikationsbäume

Das Ziel von Klassifikationsbäumen ist die Entscheidungsfindung eines Outputs mit Hilfe eines Baumes, der die verschiedenen Input Werte in Klassen aufteilt. Klassifikationsbäume sind auch als Entscheidungsbäume (engl. decision trees) bekannt. Dabei wird von der Wurzel eines Baumes, das einem Input-Wert entspricht, je nach Auswahl zum nächsten Knoten geführt, bis man zu einem Blattknoten gelangt, das dem Output bzw. der Ziel-Klasse entspricht.

Die Erstellung eines Klassifikationsbaumes für eine gegebene Datenmenge wird nach dem Divide-And-Conquer Prinzip durchgeführt. Dabei werden die Daten nach den Attributen aufgeteilt und anhand der aufgeteilten Klassen analysiert. Das Attribut, dass durch seine Aufteilung den größten Informationsgewinn zur Aufteilung der Zielklasse hervorbringt, wird vorrangig als Knotenpunkt genommen. Sind die Klassen vollständig aufgeteilt – oder ist auch die gewünschte Ziel-Tiefe des Baumes erreicht -, ist der Baum vollständig. Wenn nicht, werden die weiteren Teilbäume nach dem gleichen Prinzip nach Attributen aufgeteilt und ihr „Information Gain" neu beurteilt (15).

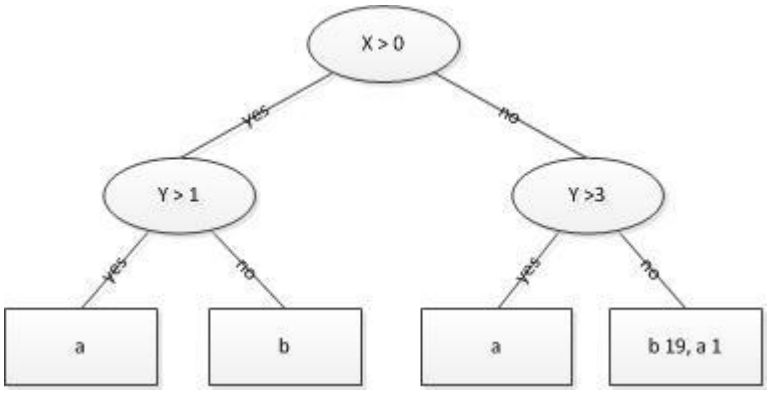

Abbildung 2.8: Ein Beispiel Klassifikationsbaum

In Abbildung 2.8 sehen wir einen so gebauten Klassifikationsbaum. Bei der Erstellung wurde als Zielklasse (a,b) angegeben und die Tiefe 2. Der linke Teilbaum wurde vollständig klassifiziert. Der rechte Teilbaum hat ein Blattknoten mit einem 95%-Ergebnis für b und ein 5% Ergebnis für a.

Für die Geschäftsprozesssimulation ist dieses Verfahren hilfreich um die Output Daten von diskreten Ergebnissen von Geschäftsprozessaktivitäten zu simulieren. Dabei stehen die Input-Daten der Aktivität als Attribute für die Aufteilung des Baumes zur Verfügung. Solche Entscheidungsbäume sind auch hilfreich für die Eliminierung von Human Tasks aus Geschäftsprozessen und die maschinelle Verarbeitung dieser Aktivitäten (16).

2.4.2 Lineare Regression

Werden keine diskreten Daten als Output Parameter einer Prozessaktivität erwartet, so muss eine andere Möglichkeit gefunden werden, die Output Parameter vorherzusagen. Eine Möglichkeit wäre, die Klassen von nicht-diskreten Werten für Klassifikationsbäume zu einer plausiblen Klasse zu berechnen. Die einfachere Möglichkeit ist die Verwendung eines anderen, für numerische Werte ausgelegtes Verfahren: der linearen Regression.

Das aus der Mathematik bekannte Verfahren kann als numerische Vorhersage genutzt werden. Dabei wird ein Output oder zu bestimmende Klasse durch die numerischen Attribute bestimmt. Jedem Attribut a_k wird durch den Data Mining Algorithmus ein Gewicht w_k zugewiesen, das be-

schreibt, wie „schwer" dieses Attribut den Output beschreibt (14). Als Formel sieht das Ganze dann so aus:

$$x = w_0 + w_1 a_1 + w_2 a_2 + \cdots + w_k a_k$$

Bei einer späteren Simulation eines Output Wertes können dann alle Informationen anhand der Attribute vorhergesagt werden. Ein sehr gutes Beispiel dafür ist die Simulation von Bearbeitungszeiten einer Aktivität. Durch „alte" Daten könnten Gewichte für die Attribute berechnet werden, und so eine sehr realistische Vorhersage für die Bearbeitungszeit gewonnen werden.

2.4.3 Training und Testen

In der Ausführung von Data Mining Algorithmen unterscheidet man in u.a. zwei Phasen: Training und Testen. Dabei soll durch diese Aufteilung eine sog. „error rate" berechnet werden. Die Trainingsphase ist die einfache Ausübung des gewünschten Data Mining Verfahrens auf die „alten" Daten. Dabei werden, wie in den vorherigen speziellen Beispielen, z.B. Klassen gebildet. Deswegen werden die Daten, die zur Erstellung der Klassifizierung genutzt werden auch „classifier data" oder „training data" genannt. Würde man jetzt die „error rate" nur auf Grundlage der genutzten Daten berechnen, wäre dieses Ergebnis zu abhängig, da man genau auf diese Daten die Klassifizierung durchgeführt hat. Um ein genaueres Ergebnis zu bekommen, benötigt man noch weitere „test data". Diese Daten sollen am besten unabhängig von den bisher genutzten Daten sein, um so ein genaueres Ergebnis der „error rate" zu erhalten (14).

Für diese Diplomarbeit sind diese zwei Phasen nicht nur für die Auswertung der Klassifizierungen wichtig, sondern auch für die spätere Evaluation der Simulationsergebnisse. Die Fehlerrate sollte anhand von Trainings- und Testphase für die Simulation bekannt sein, um die Relevanz der Simulation zu gewährleisten.

2.5 Web Services

Web Services haben sich in den letzten Jahren in der IT etabliert. Dabei sind Web Services durch die vorherrschende und auch gewollte Service-orientierten Architektur immer wichtiger geworden (17). Web Services werden später auch für die Implementierung der Simulation genutzt, um Prozesse zu starten und Aktivitäten zu ersetzen. In diesem Kapitel wird kurz neben SOA auch der Bezug zum Geschäftsprozessmanagement erläutert. Anschließend werden auch die Web Services Technologien vorgestellt, die in dieser Arbeit genutzt wurden.

2.5.1 SOA

SOA steht für „Service-oriented Architecture". Dabei ist das Hauptelement dieser Architektur, wie der Name schon sagt, der Service. Die wichtigste Eigenschaft des Services oder Dienst auf Deutsch ist „eine funktional klar abgegrenzte Ausführungslogik" (4). Dieser Service, der eigenständig arbeitet, kann immer wieder in verschiedenen Bereichen einer Implementierung aufgerufen werden. Dabei steht im Allgemeinen für jeden Service eine Dokumentation der Schnittstelle zu diesem Service zur Verfügung.

2.5.2 SOA und BPM

Was hat das nun mit dem Geschäftsprozessmanagement gemein? Laut (4) lassen sich die „Dienste einer Service-orientierten Architektur … mit den Aktivitäten einer Workflowdefintion identifizieren." Das wird auch im Laufe dieser Diplomarbeit aufgezeigt, wo einzelne Aktivitäten mit ihrem bisherigen Service-Aufruf durch einen Simulations-Aufruf ersetzt werden. Außerdem kann auch der ganze Geschäftsprozess als Service angesehen werden. Wie man in der Implementierung später sehen wird, wird für die Simulation der ganze Geschäftsprozess als Service ausgeführt.

2.5.3 WSDL

WSDL steht für Web Services Definition Language, wobei auch Description anstatt dem Definition genutzt wird. Selbst auf der HTML Webseite der W3C Organisation mit der WSDL Spezifikation steht zwar als Überschrift Web Services Definition Language, der Titel der Webseite ist aber Web Service Description Language. Die unterschiedlichen Namen ändern aber nichts an der Spezifikation der XML-basierten Sprache. WSDL ist eine beschreibende Sprache zur Verbindung von Web Services. Mit Hilfe von WSDL können z.B. Prozesse mit gegebenen Parametern aufgerufen werden. Dabei enthält die zugehörige .wsdl Datei die Schnittstellen Informationen. Neben der auszuführenden Operation enthält die Datei weitere Informationen über die genauen Datentypen. Aktuelle Version ist seit Juni 2007 WSDL 2.0. In dieser Diplomarbeit wird aber noch Version 1.1 verwendet, durch die gegeben Vorarbeiten, die in Kapitel 3 vorgestellt werden (18).

2.5.4 SOAP Web Services

SOAP steht für Simple Object Access Protocol und ist wie WSDL ein Standard der W3C Organisation. SOAP ist das Standardformat für die Nachrichten, die zwischen den Services ausgetauscht werden (4). Die Spezifikation ist wiederum auf der offiziellen Website verfügbar (19).

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Header>
  <n:alertcontrol xmlns:n="http://example.org/alertcontrol">
   <n:priority>1</n:priority>
   <n:expires>2001-06-22T14:00:00-05:00</n:expires>
  </n:alertcontrol>
 </env:Header>
 <env:Body>
  <m:alert xmlns:m="http://example.org/alert">
   <m:msg>Pick up Mary at school at 2pm</m:msg>
  </m:alert>
 </env:Body>
</env:Envelope>
```

Listing 2.2: Beispiel einer SOAPMessage (19)

Auch diese Sprache ist XML-basiert, wie man in Listing 2.2 sehen kann. Dabei enthält der Umschlag („envelope“) einen Kopf („header“) und einen Körper („body“), ähnlich wie bei HTML. Der „wichtigere“ Teil ist der body,

den hier werden die Parameter für die Operation, die aufgerufen wird, festgesetzt. Diese SOAPMessage muss nämlich so aufgebaut sein, wie es in der zugehörigen WSDL Datei dargestellt wurde.

In Abbildung 2.9 sieht man das Schema, wie dieses Protokoll funktioniert. Ein Client kann die XML-basierte SOAPMessage per HTTP an einen Server schicken. Nach außen ist von dem Server nur die WSDL Spezifikation bekannt. Der Server wird seinen Service ausführen und eine SOAPMessage zurück zum Client mit dem Ergebnis senden.

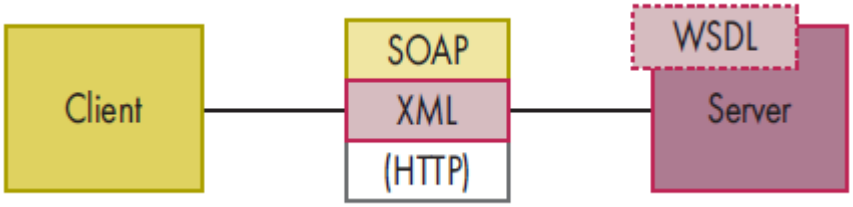

Abbildung 2.9: Schema der SOAP Web Services (17)

2.5.5 JAX-WS

„Java API for XML Web Services" ist wie der Name schon sagt eine Java Schnittstelle für XML-basierte Web Services. Mit JAX-WS können die vorgestellten Technologien von WSDL Interfaces und SOAPMessage einfach in Java umgesetzt werden. Diese Schnittstelle wird später genutzt, um die Simulation als Web-Service aufzurufen. Weitere Information und die Spezifikation auf der offiziellen Website (20).

3 Vorarbeit und verwandte Projekte

Es wurden schon einige Versuche in Richtung Geschäftsprozesssimulation gegangen. Dabei unterscheiden sich die Projekte in der Zielsetzung. Die Zielsetzung dieser Diplomarbeit wird in diesem Kapitel klarer. Es werden erst vorherige Projekte vorgestellt, die diese Diplomarbeit erst möglich gemacht haben. Darauf wird ein Projekt vorgestellt, dass die Geschäftsprozesssimulation als Hauptthema hatte. Als Abschluss wird noch ein Beispiel-Prozess vorgestellt, der aus den Vorarbeiten des dBOP Projekts entstanden ist.

3.1 Vorarbeiten zu dieser Diplomarbeit

Die vorgestellten Projekte wurden an der Universität Stuttgart durchgeführt. Dabei entstand aus der Business Impact Analysis (21) später die deep Business Optimization Platform (3). Der Hauptgedanke der Projekte ist die Zusammenführung von Prozessdaten und operativen Daten. Durch diese Zusammensetzung können Geschäftsprozesse besser analysiert werden und geben somit tiefere Einblicke in die Geschäftsprozesse. Die Optimierung dieser ist das Ziel durch diesen Informationsgewinn.

3.1.1 Business Impact Analysis

Um Geschäftsprozesse zu verbessern, bedarf es einer Analyse des bisherigen Prozesses. Dabei sollten alle relevanten Informationen zur Verfügung stehen. Leider stehen Geschäftsprozessen meist nur die Audit-Logs eines Prozess zur Verfügung, um eine Verbesserung durchzuführen. Weitere relevante Daten wären die des Data Warehouse eines Unternehmens mit seinen operativen Daten. Aber auch dies wären dann zwei unterschiedliche Analysen. Im Business Impact Analysis Projekt der Universität Stuttgart sollte nun genau diese beiden Quellen an Daten zusammengeführt werden. In Abbildung 3.1 ist die Idee der Business Impact Analysis zu sehen.

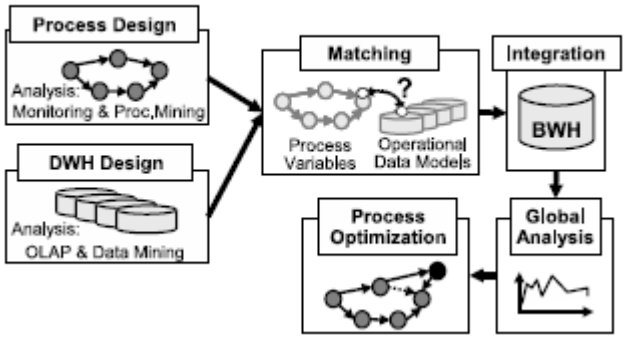

Abbildung 3.1: Überblick der Business Impact Analysis (22)

Zuerst werden aus dem Prozess- und Data-Warehouse-Design die zusammengehörenden Daten „gematcht". Dabei werden die Daten verknüpft, d.h. der aus den Prozessdaten bekannt Customer mit der ID = 2 wird mit seinem Namen und Alter aus dem DWH zusammengeknüpft. Diese neue zusammengesetzte Information wird nun als integriertes Business Warehouse gespeichert. Darauf lassen sich nun Analysen ausführen, wie die Simulation in dieser Diplomarbeit. Aus den Ergebnissen lassen sich dann hoffentlich auch Verbesserungen in den Geschäftsprozessen ableiten.

Um die operativen und Prozess Daten miteinander zu verbinden wurde der BIAEditor entwickelt, der anhand der Parameter des Prozesses und der Datenbank Attribute (semi-)automatisch verbindet und eine Meta-Verknüpfung aufbaut. Durch diese Informationen können die Daten dann später zusammengeführt werden. Der BIA Editor wird später in Kapitel 5.2.1 in der Implementierung vorgestellt.

3.1.2 deep Business Optimization Platform

Auf Grund der wichtigen Stellung, die Geschäftsprozesse in einem Unternehmen inzwischen inne halten, entwickelte sich an der Universität Stuttgart darauf auch die deep Business Optimization Platform. Die Problemstellung war, dass eine Prozessoptimierung bisher nur durch relativ wenig Daten begründet worden konnte und nur auf Grund der Analyse-Fähigkeiten von Mitarbeitern durchgeführt wurde. Die Optimierung von Geschäftsprozessen soll mit Hilfe einer größeren Datenmenge durch ein integriertes Data Warehouse geschehen. Die Zusammenführung von ope-

rativen und Prozess-Daten wurde schon durch das Business Impact Analysis Projekt im vorherigen Kapitel erläutert.

Die Optimierung soll aber nicht nur a-posteriori geschehen, sondern auch vor und während der Prozessausführung. Die a-priori Optimierung baut auf „Best Practices" auf, wobei die Optimierung während eines Geschäftsprozesses auf eine optimale Ressourcenverteilung strebt. Die Analyse nach der Ausführung baut dann am meisten auf das integrierte Warehouse aus. Informationen, die aus dem Prozess gesammelt wurden, werden nun genutzt, um den Geschäftsprozess zu verbessern, indem man ihn möglicherweise umstrukturiert. Alle diese vorgestellten Optimierung bauen aber auf vorherigen Geschäftsprozessen und den Informationen, die man gesammelt hat. „Best Practices", optimale Ressourcenverteilung und Umstrukturierung des Prozess brauchen Information, die als Vorlage in der Plattform zur Verfügung stehen.

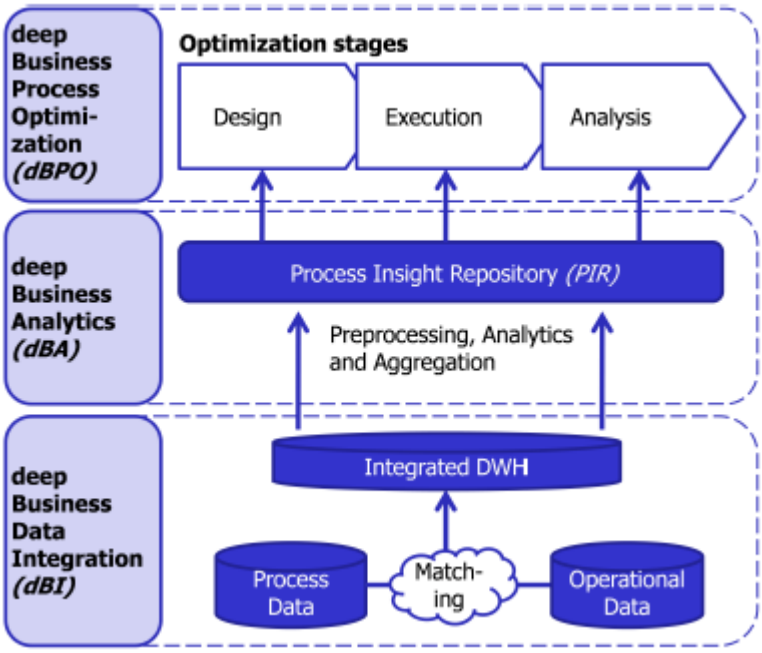

Abbildung 3.2: Layer der deep Business Optimization Platform (3)

Abbildung 3.2 zeigt den Aufbau der Plattform. Die untere Schicht zeigt die deep Business Data Integration, die durch die Business Impact Analysis gegeben ist. Anschließend wird das integrierte Data Warehouse in der

deep Business Analytics Schicht analysiert. Diese Informationen werden dann für die verschiedenen Optimierungsmöglichkeiten für die Geschäftsprozesse zur Verfügung gestellt.

3.1.2.1 deep Business Data Integration

Diese Schicht baut auf der Idee der Business Impact Analysis auf. Dabei ist das ausgearbeitete Matching der Daten die wichtigste Funktion. Die zugehörige Zusammenführung der Daten geschieht in drei Phasen, wie in Abbildung 3.3 dargestellt. Als erstes muss das Matching, wie die Prozessdaten und operativen Daten zusammengehören, bestimmt werden. Ist das geschehen, werden die Daten nach den Matching Regeln zusammengeführt. Am Ende werden die Daten in ein integriertes Data Warehouse konsolidiert. Näheres dazu in (16).

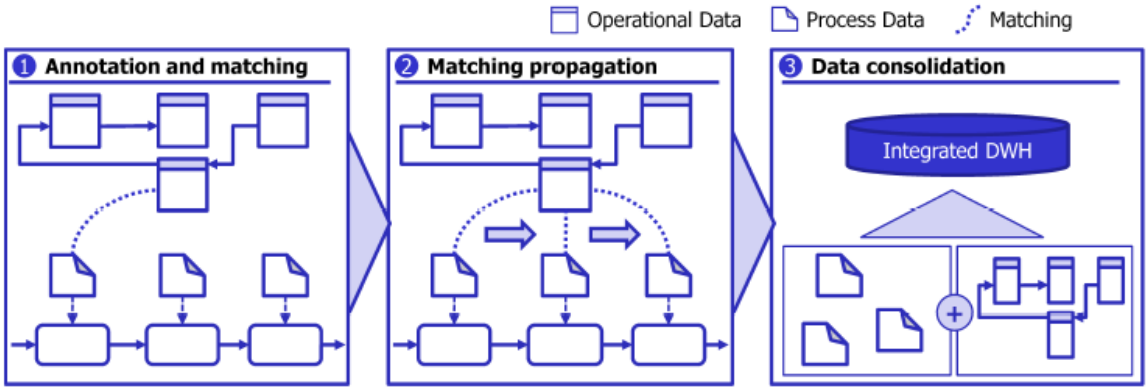

Abbildung 3.3: dBOP Data Integration Phasen (16)

3.1.2.2 deep Business Analytics

Aus Abbildung 3.2 bekannt ist die deep Business Analytics Schicht die Schicht für die Vorbereitung zur Optimierung von Geschäftsprozessen. In dieser Analyse sollen relevante Informationen aus dem integrierten Datenbestand gefiltert werden. Das geschieht in zwei Phasen, die auch aus dem Data Mining bekannt sind: Data Preprocessing und die eigentlich Ausführung der Analyse.

Beim Data Preprocessing werden die Daten aus dem „Integrated DWH" für die Analyse vorbereitet. Neben den im Data Mining benötigten Schrit-

te, wie dem Data Cleansing, sind weitere Preprocessing Schritte möglich und vielleicht auch nötig. Abbildung 3.4 zeigt die Möglichkeiten.

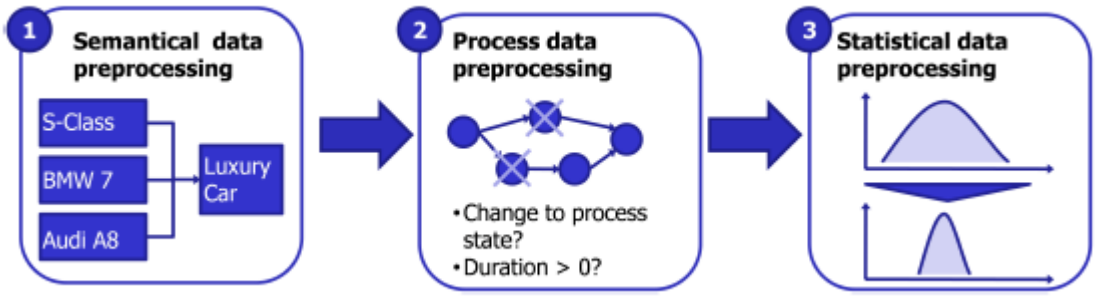

Abbildung 3.4: Data Preprocessing der deep Business Analytics Schicht (3)

Neben einer schematischen Gruppierung (Teil 1) ist auch z.B. eine Elimination von Aktivitäten im Prozess möglich (Teil 2), wenn diese keine Auswirkung auf den Output hat, und somit nicht relevant für das Data Mining Ergebnis ist. Darauf kann dann das bekannte Data Preprocessing mit z.B. Data Cleansing erfolgen (Teil 3).

In der Analyse Phase werden dann die vorbereitenden Daten analysiert. Dabei werden auch ganz einfache Daten, wie die Eckdaten (Min-, Max-, Mittelwert, u.a.) jeder Aktivität berechnet. Darüber hinaus werden auch Data Mining Algorithmen ausgeführt, wie die des vorgestellten Klassifikationsbaums. Abbildung 3.5 zeigt einen solchen Klassifikationsbaum eines dBOP Prozesses. Dabei können Knoten sowohl Prozessattribute als auch Attribute der operativen Daten sein.

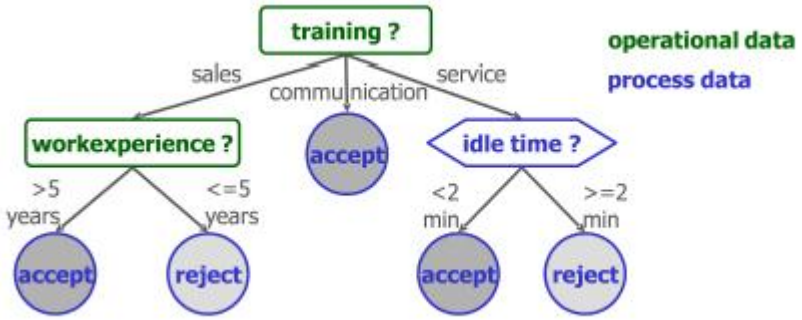

Abbildung 3.5: Klassifikationsbaum-Beispiel für den weiteren Ablauf nach Attributen (3)

3.1.2.3 deep Business Process Optimization

Das Ziel der deep Business Optimization Platform ist – wie der Name schon sagt – die Optimierung. Deshalb ist die dritte Schicht, die deep Business Process Optimization, die wohl wichtigste hinsichtlich der Verbesserung von Geschäftsprozessen. Dies kann mit Hilfe von „Optimization Patterns" geschehen, die im dBOP Projekt entwickelt wurden (23). Um diese zu nutzen, muss zu allererst eine Zielfunktion definiert werden.

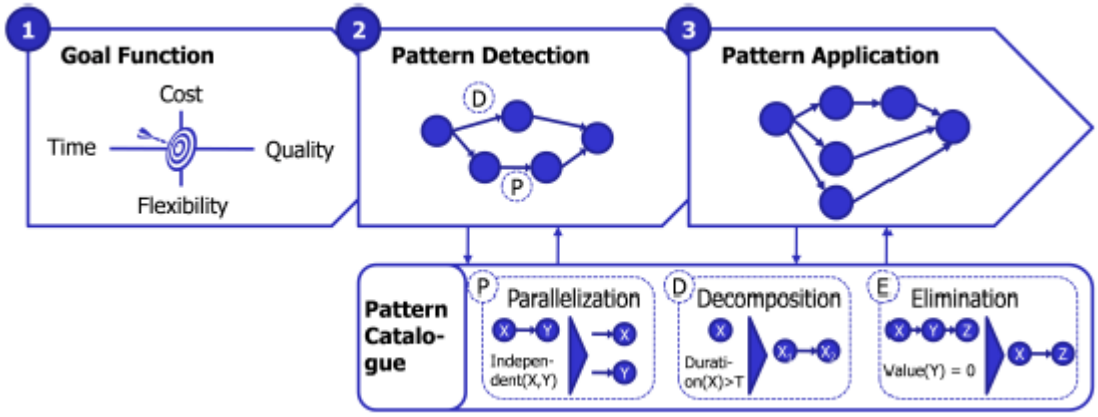

Abbildung 3.6: Übersicht der deep Business Process Optimization Schicht

Abbildung 3.6 zeigt die einzelnen Schritte, die für die Optimierung notwendig sind. Als erstes die Definition der Zielfunktion, mit Bestimmung der Wichtigkeit von Zeit, Kosten, Qualität und Flexibilität des Prozesses. Auf Grund der bestimmten Zielfunktion, wird darauf erkannt welche Pattern im gegebenen Prozess angewendet werden können. Sind diese dann ausgewählt, werden sie im letzten Schritt angewendet.

Die wichtigste Information dieser Schicht liegt im Pattern Katalog. Aus Abbildung 3.6 kann man die Pattern Parallelization, Decomposition und Elimination entnehmen. In (3) wird das „Automated Approval" Pattern genauer vorgestellt. Dieses Pattern hat als Ziel eine „Human Task" in einem Prozess, d.h. eine Aktivität, die von einem Mitarbeiter erst überprüft werden muss und somit Zeit und Geld kostet, in eine automatische umzuwandeln. Durch eine Automatisierung ist es möglich Zeit und Geld einzusparen. Im weiteren Verlauf wird ein Beispiel Prozess vorgestellt, in dem dieses Pattern eingesetzt werden könnte.

3.1.2.4 dBOP Prozessmodell

Das dBOP Prozessmodell ist ähnlich zu gewöhnlichen Prozessmodellen mit ein paar speziellen Unterschieden. Eine genaue Beschreibung findet man in (23). Der Kontrollfluss ist ziemlich ähnlich, nur die genaue Implementierung einer dBOP Aktivität ist spezieller. Eine dBOP Aktivität bezieht sich neben den üblichen Input und Output Daten und seiner Implementierung, auch einer Ressourcenverwaltung. Außerdem ist eine Aktivität immer als Service implementiert. Abbildung 3.7 zeigt eine Aktivität des dBOP Prozessmodells.

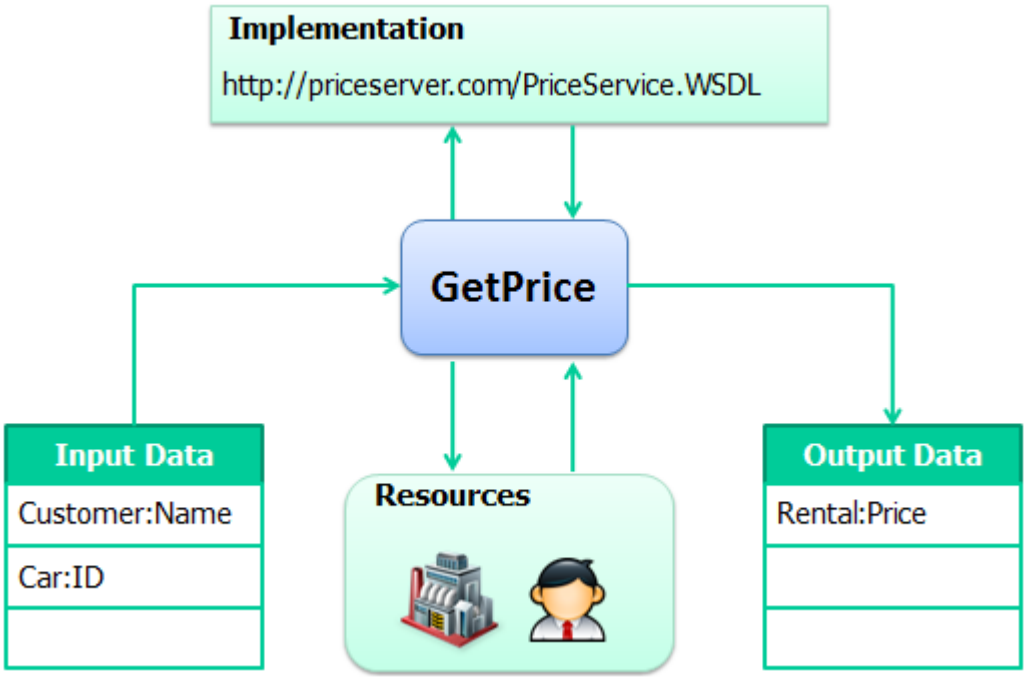

Abbildung 3.7: dBOP-Aktivität mit seinen Attributen

Ressourcen werden durch einen Typen, wie Maschinen oder Mitarbeiter, und seiner Rolle definiert. Durch diese Erweiterung von Ressourcen, bestehen verschieden Zugehörigkeiten im Prozessmodell. Neben den obligatorischen Kontrollfluss und Datenfluss ist nun auch ein Ressourcenfluss im Prozess zu beobachten.

Um das Ganze zu implementieren, bedarf es einer kleinen Modifikation in der BPEL Implementierung. Eine dBOP Aktivität kann nicht einfach eins zu eins in eine BPEL Aktivität transformiert werden, sondern muss aufgeteilt werden. Dabei wird vor und nach jedem BPEL Aufruf des eigentlichen Ak-

tivitäts-Services und seiner Implementierung, der „Resource Manager Service" aufgerufen, der die Ressourcen für die Aktivität bereitstellt. Abbildung 3.8 zeigt wie das Mapping schemenhaft auszusehen hat.

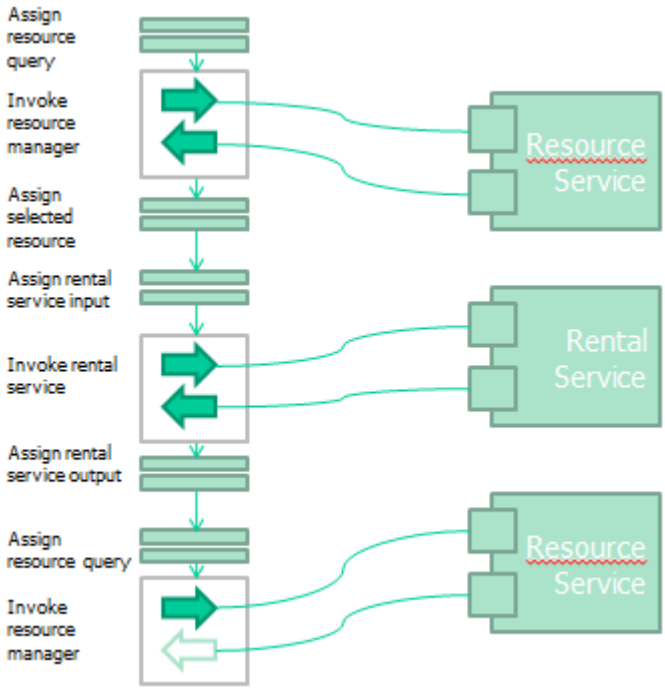

Abbildung 3.8: BPEL Implementierung einer dBOP Aktivität

Es sei schon einmal vorweggenommen, dass in dieser Diplomarbeit die Simulation für die Ressourcenverteilung nicht implementiert wurde. Dies kann aber später mit Hilfe des universellen Simulation Service auch erstellt werden.

3.1.3 Bezug zu dieser Diplomarbeit

Diese Diplomarbeit soll eine Geschäftsprozesssimulation ausführen, die auf der Analyse aus der deep Business Optimization Platform aufbaut. Dabei werden die Ergebnisse aus der Data Mining Analyse der Daten genutzt. Dies ermöglicht eine erweiterte Sicht auf die Daten und einen großen Informationsgewinn. Im vorgestellten dBOP Projekt mit dem Ziel der Optimierung kann mit Hilfe einer Simulation gezeigt werden, dass die automatisierten Optimierungen, wie das „Automated Approval" Pattern, zu einem schnelleren und trotzdem genauen Ergebnis kommen. Weiteres dazu in den weiteren Kapiteln.

3.2 Verwandte Projekte (Process Mining)

Diese Diplomarbeit ist nicht die erste wissenschaftliche Arbeit, die sich mit der Geschäftsprozesssimulation auseinandergesetzt hat. In diesem Kapitel wird ein weiteres Projekt vorgestellt, das eine ähnliche Thematik bearbeitet und somit auch einen Einfluss auf diese Arbeit hatte. Die Hauptargumente und relevanten Ideen werden aufgezeigt.

3.2.1 Überblick

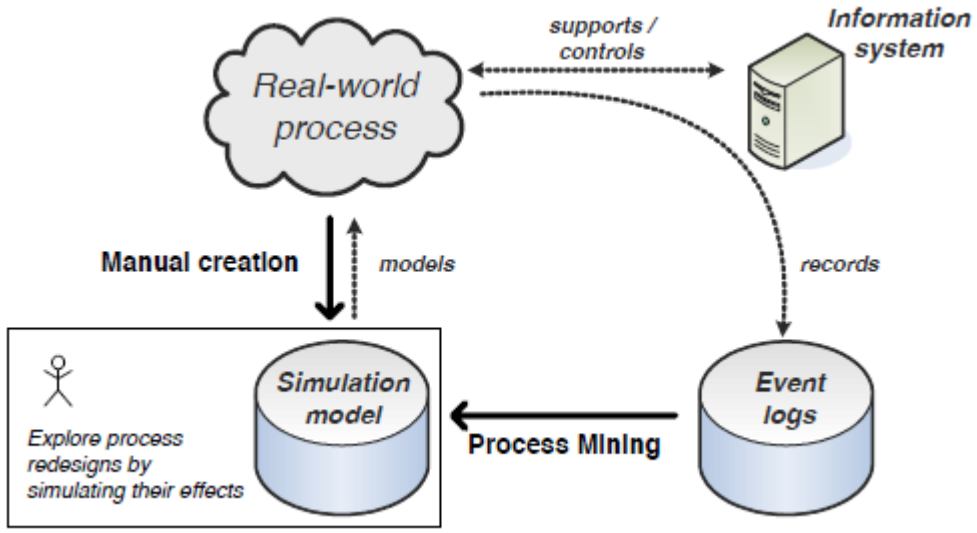

Abbildung 3.9: Schematisches Modell vom realen Prozess zur Simulation beim Process Mining (24)

Dieses Tool namens ProM (kurz für Process Mining) wurde an der Technischen Universität in Eindhoven, Niederlande entwickelt. Dabei werden Informationen ausschließlich aus den Prozess Logs gesammelt und der Prozess analysiert. Dabei muss der Prozess auch erst rekonstruiert werden („Control-flow Discovery"). Dies geschieht durch den Alpha-Algorithmus. Anschließend wird der rekonstruierte Prozess in ein Simulationsmodell umgewandelt. Dieses Modell ist ein Coloured Petri Net (CPN). Dieses Netz kann dann genutzt werden, um die Prozesse zu simulieren (24).

Das Tool wird als ProM[1] Framework zur Verfügung gestellt mit einer Export Funktion für die Nutzung mit der CPN Software namens CPN Tools[2].

[1] www.processmining.org

3.2.2 Process Mining Technicken

Bevor der Prozess in ein CPN umgewandelt werden kann, werden verschiedene Perspektiven auf den Prozess und seine Log Daten ausgeführt. Ein Überblick gibt die Abbildung 3.10.

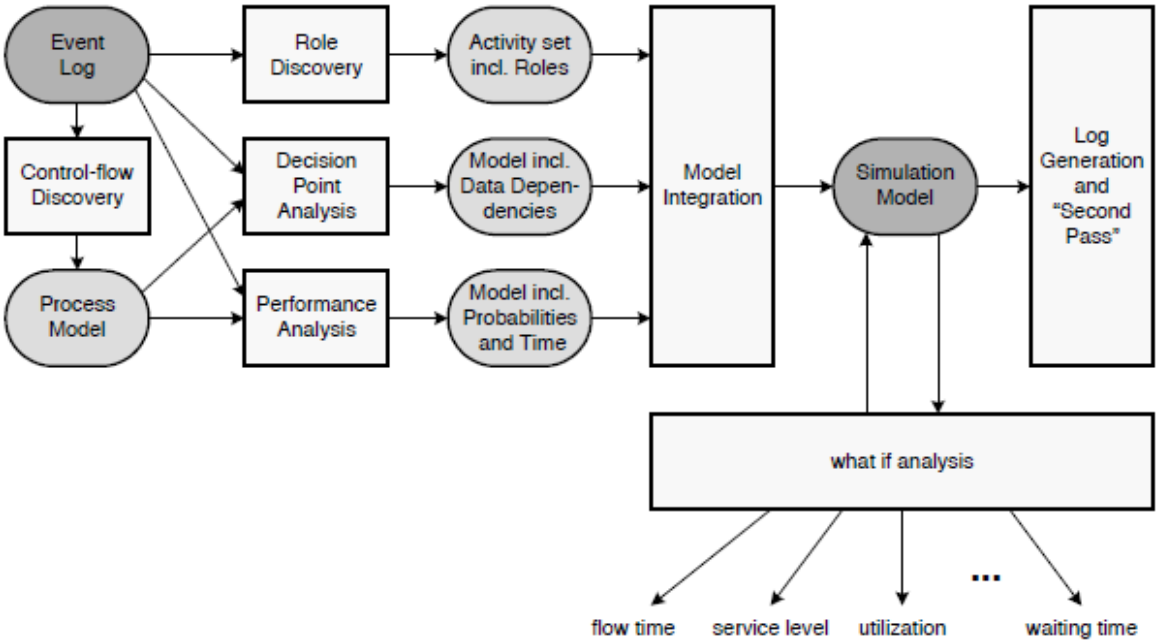

Abbildung 3.10: Überblick der Process Mining Techniken (24)

„Role Discovery" ist die Entdeckung von Rollen bei Ressourcen. Dabei werden die genutzten Ressourcen durch ihren Einsatz bei Aktivitäten analysiert. Je nach Einsatz werden die einzelnen Ressourcen in Klassen eingeteilt.

Bei „Decision Point Analysis" werden die Verzweigungen in Geschäftsprozessen analysiert. Dabei spielen die bedingten Verzweigungen eine wichtige Rolle. Es wird ein Klassifikationsbaum mit den Bedingungen, welcher Weg weiter ausgeführt wird, als Zielklassen aufgestellt. Alle Information, die bisher gesammelt wurden, können als Input-Parameter für die Klassifikation genutzt werden. Abbildung 3.11 zeigt Beispiel-Daten (a) und den zugehörigen Baum (b), der entstanden ist. „Aus diesem Entscheidungs-

[2] www.cpntools.org

baum können (…) jetzt (…) Regeln [abgeleitet werden.]" Diese Regeln können nun für die Aktivitäten zur Simulation eingesetzt werden.

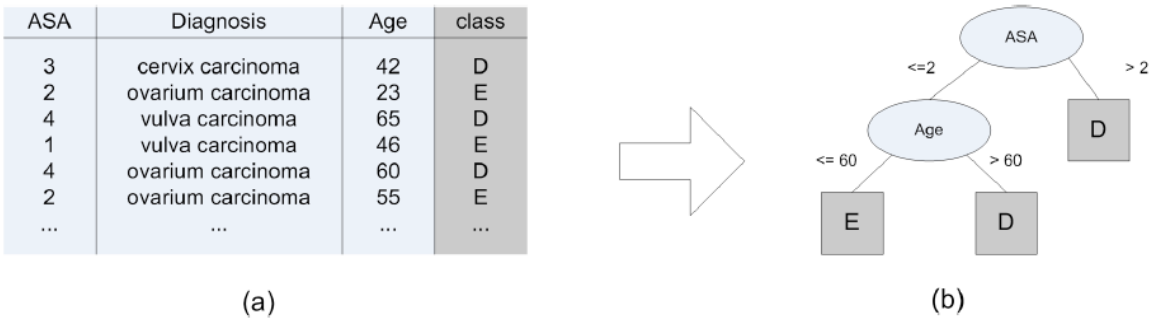

Abbildung 3.11: Decision Point Analysis (24)

Als vierte (auch Control-Flow Discovery gehört zu ihnen) Process Mining Technik wird Performance Analysis vorgestellt. Abbildung 3.12 zeigt die Performance Analyse eines ganzen Prozess.

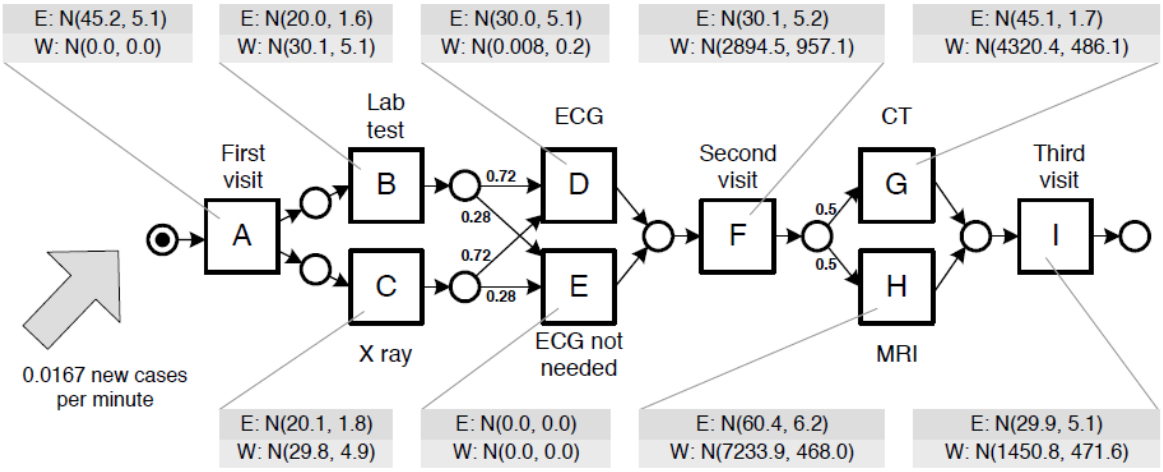

Abbildung 3.12: Performance Analysis (24)

Dabei werden folgende Daten gesammelt: Die Ausführungs- und Wartezeit jeder Aktivität kann aus dem vorhandenen Logs ausgelesen werden. Anschließend werden der Mittelwert und die Standardabweichung einer Normalverteilung darauf gebildet. Außerdem werden die zeitlichen Abstände für die Prozessinstanziierungen ausgelesen: In Abbildung 3.12 sind das 0,0167 Prozessinstanzen pro Minute. Um neben der „Decision Point

Analysis" noch einen Vergleichswert für die Ausführungen der Verzweigungen zu erhalten, werden für jede Verzweigung auch die Wahrscheinlichkeiten für jede weitere Kante gesammelt.

3.2.3 Simulations-Evaluation („Second Pass")

Darüber hinaus wurde eine Qualitätsevaluation durchgeführt, in dem die simulierten Ergebnisse mit den echten verglichen werden. Dabei wurde ein zweites Simulationsmodell mit Hilfe der simulierten Daten aufgestellt (deshalb „Second Pass"). Je mehr die Ergebnisse der Process Mining Technicken denen der ersten Ausführung gleichen, desto besser ist das Simulationsmodell.

3.3 Beispiel-Prozess

Im Laufe dieser Diplomarbeit soll ein (dBOP) Prozess simuliert werden. In diesem Kapitel wird dieser Prozess vorgestellt, der in dieser Diplomarbeit öfters erwähnt wird um die Konzepte näher zu bringen. Es handelt sich um ein Szenario für ein Autovermietungsunternehmen. Das Beispiel wurde auch schon in (3) vorgestellt.

3.3.1 Szenario des Prozess

Dieser Prozess handelt von dem Vertriebs-Prozess der Car Rental Company Ltd. Dieser Prozess wird auch später simuliert werden. Im Folgenden wird der Prozess mit Hilfe von BPMN vorgestellt und anschließend noch seine entsprechende BPEL Implementierung erläutert.

3.3.2 Prozessmodell

Der Prozess betrifft wie schon erläutert den Vertrieb im Unternehmen und beinhaltet somit den Kunden und das Büro als die zwei Beteiligten in diesem Prozess. Der Kunde betritt den Laden, wünscht ein bestimmtes Auto zu mieten und kann weitere (ihm vorgeschlagene) Extras wählen. Ist dies geschehen, wartet er auf das Ergebnis, ob er die Voraussetzungen für das Mieten des gewählten Auto erfüllt. Wenn ja, kann er den Vertrag unterschreiben und der Prozess endet für ihn; wenn nicht, endet der Prozess

für ihn ohne Abschluss eines Vertrags. Auf Seiten des Büros als Beteiligter wird dem Kunden beim Betreten ein Mitarbeiter zugewiesen. Hat der Kunde ein Auto gewählt, so werden vom Büro aus Extras vorgeschlagen, die für den Kunden und seine Wahl des Autos relevant sind. Sind die Extras gewählt, muss nun überprüft werden, ob der Kunde die Voraussetzungen für einen Vertragsabschluss erfüllt. Ist das der Fall wird ihm ein Vertrag aufgesetzt; wenn nicht, wird der Prozess abgebrochen. Abbildung 3.13 zeigt den Prozess in BPMN.

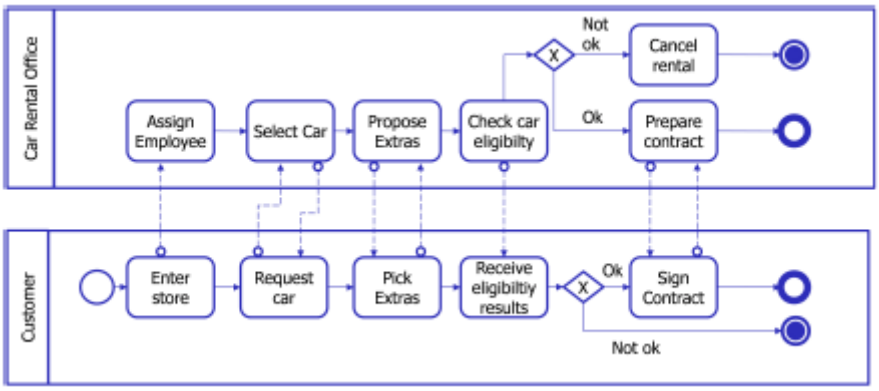

Abbildung 3.13: BPMN Modell des zu simulierenden Geschäftsprozesses (3)

3.3.3 BPEL Implementierung

Die Implementierung des Geschäftsprozesses geschieht in BPEL auf dem IBM Process Server (nähere Erläuterung im Kapitel zur Implementierung). Der Geschäftsprozess wurde im vorigen Kapitel in BPMN schematisch dargestellt. Die Implementierung dieses dBOP Prozesses geschieht wie im dBOP Kapitel vorgestellt mit einer Ressourcenverwaltung. Der Testprozess hat daher einen großen BPEL Aufbau aber nur wenige „wirkliche" Aktivitäten. Abbildung 3.14 zeigt einen Ausschnitt der eigentlichen Aufrufe der BPEL Datei.

Abbildung 3.14: Auszug der BPEL Version des Testprozess im BPEL Editor des WebSphere Integration Developer

Der Prozess hat eine Aufrufaktivität namens „receiveStart" und startet den Prozess mit dem Inhalt der customerID. Durch das Zusammenspiel von Prozessdaten und operativen Daten, kann später anhand der customerID die weiteren Daten aus dem Data Warehouse gelesen werden. Mit dem „CreateRental" Aufruf wird ein neuer Prozess mit dem zugewiesenen Mitarbeiter in die Datenbank geschrieben. Anschließend kann sich der Kunde ein Auto aussuchen („SelectCar"). Die Aktivität „Pick Extras" aus

dem BPMN Modell wurde vorerst nicht implementiert, da sie dem „CheckEligibility" sehr selten einen anderen Output ergeben würde.

Damit kommen wir zum wohl wichtigsten Aufruf für diese Diplomarbeit, dem „CheckEligibility". Mit diesem Aufruf bekommt ein Mitarbeiter alle Informationen zugesendet und kann auf Grund dieser entscheiden, ob ein Kunde „eligible" (berechtigt) ist oder nicht. Wenn ja, kommt ein Vertrag zu Stande („CreateContract") und der Gesamtpreis wird berechnet, wenn nicht, dann wird der Vermietungsantrag abgebrochen („CancelRental").

3.3.4 Aspekte der Simulation

Interessante Aspekte in diesem Prozess, die in einer Simulation untersuchten werden können und hilfreiche Erkenntnisse über den Prozess geben können, sind folgende:

- „Check Car Eligibilty": Die Überprüfung, ob der Kunde die Anforderung für das Mieten eines Autos erfüllt, geschieht durch einen Mitarbeiter („Human Task"). Eine Ersetzung dieser „Task" durch einen automatischen Anforderungscheck kann mit Hilfe einer Simulation begründet werden. Wenn die Simulation aufzeigt, dass diese Aktivität nun kürzer arbeitet, sich somit die Wartezeiten der anderen Aktivitäten verringert und der gegebene Prozess beschleunigt wird.
- Mitarbeiter-„Success": Zu jedem Prozess werden Mitarbeiter zugewiesen, die den Kunden im Laufe des Prozess beraten. Wie entscheidend ist es, was ein Mitarbeiter für ein Training hat, auf den Erfolg eines Vertragsabschluss, kann auch noch gesehen werden.
- Ressourcen-Management: Einer der speziellen dBOP Eigenschaften. Das Simulieren des Ressourcen-Management kann eventuelle Engpässe im Prozess aufzeigen. Dieser Aspekt wird in dieser Diplomarbeit aber nicht betrachtet.
- Ausführungs- und Wartezeit: Die Dauer jedes Prozess kann auschlaggebend für den Unternehmenserfolg sein. Deshalb sollten Geschäftsprozesse zügig abgearbeitet werden. Die Wartezeit Analyse in einem Prozess ist eine der interessanteren Analysen in einer Simulation, da

in einer Simulation Situationen, die im bestehenden Prozess auftreten könnten, ausgetestet werden können.

4 Konzepte des Simulations-Framework

In diesem Kapitel wird aufgezeigt mit welchen Mitteln und Wegen diese Simulation durchgeführt wird. Dabei wird zunächst das ganze Vorgehensmodell kurz vorgestellt, um anschließend die einzelnen Phasen detaillierter zu erläutern.

4.1 Motivation und Ziel

Für die Simulation „sollten sich die Daten aus Informationssystemen ermitteln lassen." (13) Da dies durch das dBOP Projekt gegeben ist (dBOP Data Integration), bietet sich eine Geschäftsprozesssimulation für einen dBOP Prozess an. Um diese These zu stützen wird am Ende bei der Evaluation aufgezeigt, dass detailliertere Daten eine bessere – im Sinne von realitätsnaher – Simulation bieten als nur Prozessdaten.

Die Simulation eines Geschäftsprozess wird durch den Austausch von Aktivitäten durchgeführt. Dabei werden die Aktivitäten des „normalen" Prozess durch neue Simulations-Aktivitäten ersetzt. Diese Aktivitäten sollen die gleichen Ergebnisse liefern. Dabei sollen die bisherig durchgeführten Geschäftsprozesse mit Hilfe von Data Mining Algorithmen analysiert werden. Wie das Ganze nun von Anfang bis Ende konzeptionell aussieht, ist das Ziel dieses Kapitels.

4.2 Vorgehensmodell

Um die Simulation durchführen zu können, müssen verschiedene Phasen durchlaufen werden um einen Prozess zu simulieren. Gestützt auf (13), die neun Phasen vorschlagen, aber je nach Zielsetzung verändert werden können, wurden hier Phasen für das Framework abgeleitet. Das gesamte Vorgehensmodell kann man Tabelle 4.1 entnehmen.

Phasen	Konzept	Implementierung
1. Planung und Analyse	Vorbereitende Maßnahmen für eine mögliche Simulation	BPAClient
2. Datendefinition und -erhebung	Vorbereitung für die Simulationskonstruktion und –ausführung	dBOPSim (Settings + Input Generator)
3. Simulationskonstruktion und –ausführung	Konstruktion der Modelle und anschließende Ausführung	Simulation Service
4. Bereitstellung der Ergebnisse	Ergebnisse der Simulation auslesen	dBOPSim (WEKA) + DB2

Tabelle 4.1: Vorgehensmodell aufgeteilt in Phasen

4.2.1 Planung und Analyse

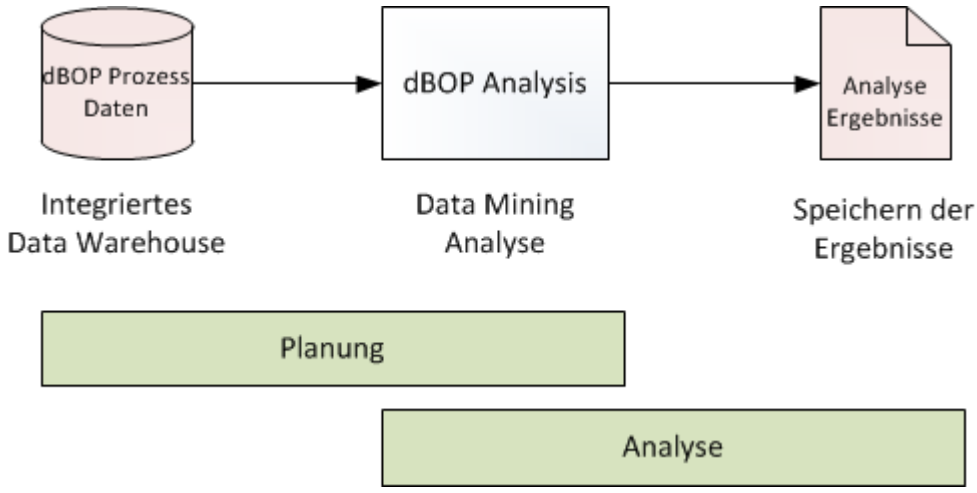

Abbildung 4.1: Schema der Planungs- und Analysephase

Grundsätzlich muss geschaut werden, ob eine Simulation überhaupt sinnvoll ist. Ob der Aufwand auch dem Nutzen entspricht. Ist dies der Fall so wird in der Analyse der zu simulierende Geschäftsprozess angeschaut. Dabei wird beachtet inwiefern der vorhandene Geschäftsprozess für die Simulationsbildung geeignet ist, welche Aktivitäten ersetzt werden müssen. Für die Planung und Analyse kann das bereits implementierte Insight

Repository des dBOP Projekts genutzt werden. Abbildung 4.1 zeigt die Phase schematisch.

4.2.2 Datendefinition und –erhebung

Der zu simulierende Prozess und die dazugehörigen Daten wurden analysiert. Nun müssen Vorbereitung für die Simulation getroffen werden: Datenbank-Tabellen für die Simulation müssen erstellt werden, damit man die Ergebnisse der Simulation speichern kann. Es wird auch festgelegt mit welcher Wahrscheinlichkeitsverteilung die Inputdaten simuliert werden. Außerdem können Elemente, die nicht unmittelbar für die Informationsgewinnung relevant sind, entfernt werden und somit wertvolle Ressourcen eingespart werden. Abbildung 4.2 zeigt das Schema.

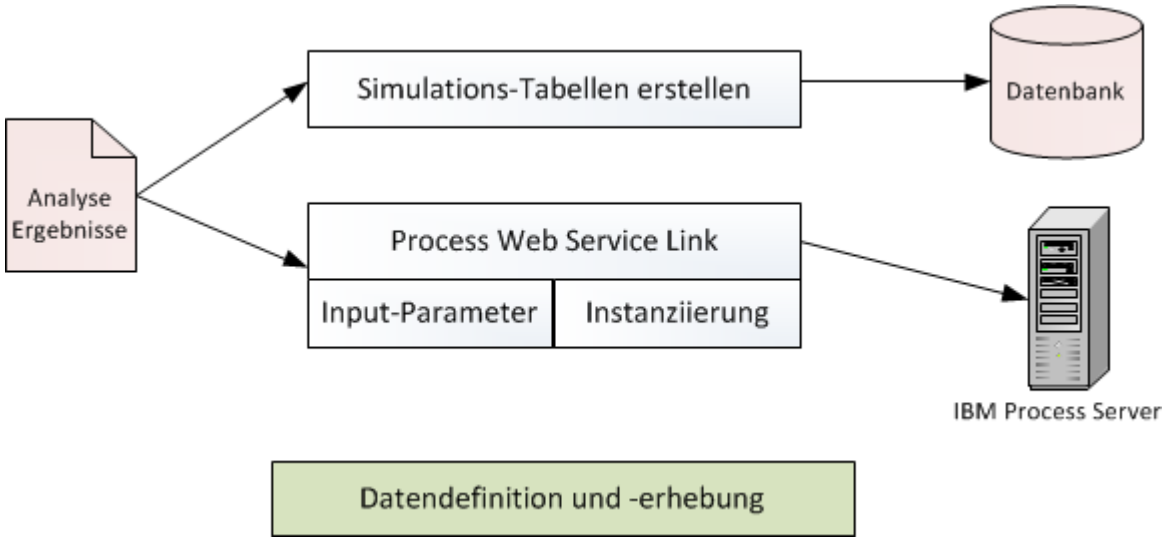

Abbildung 4.2: Schema der Datendefinition und –erhebung

4.2.3 Simulationskonstruktion und –durchführung

Die nachfolgende Phase ist eine Konstruktion des Simulationsmodells mit Hilfe der vorhergegangenen Phasen. Der Prozess bzw. die zu simulierenden Aktivitäten werden durch einen Simulations-Service ersetzt. Ist dies geschehen, wird die Simulation mit der ausgewählten Wahrscheinlichkeitsverteilung für Input-Daten ausgeführt. Abbildung 4.3 zeigt das Schema.

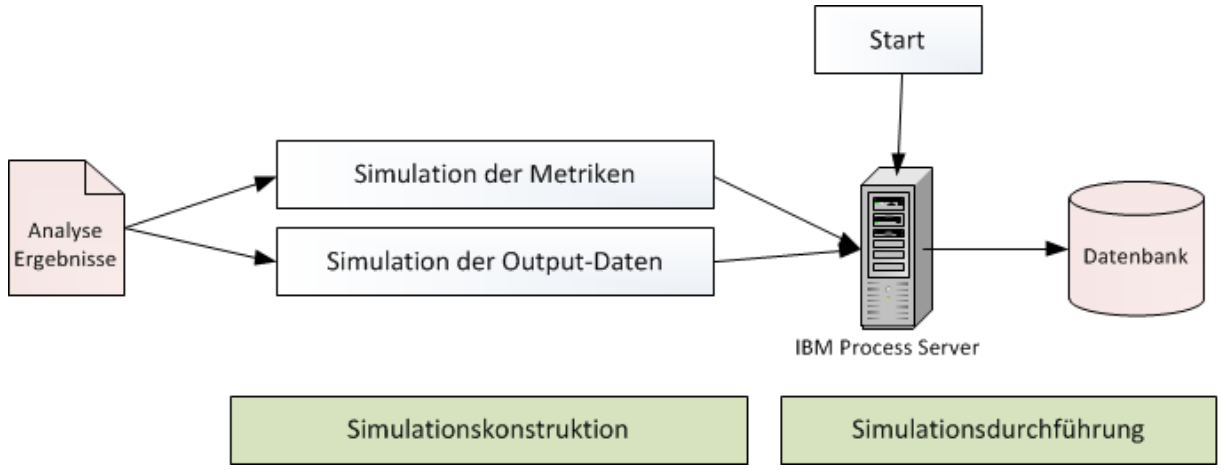

Abbildung 4.3: Schema der Simulationskonstruktion und -durchführung

4.2.4 Bereitstellung der Ergebnisse

Nachdem die Simulation mit gewählten Input-Größen durchlaufen ist, werden die Ergebnisse aus der Datenbank bereitgestellt. Diese Übersicht erlaubt „eine transparente Erläuterung der Ergebnisse." (13) Dabei sind vor allem die Ergebnisse der Output Parameter interessant. Außerdem wird auch eine Export-Funktion zur Verfügung gestellt, die erlaubt weitere Analysen mit den Ergebnissen über diese Diplomarbeit hinaus zu durchlaufen.

4.3 Planung und Analyse

Wie bereits vorgestellt, soll diese Diplomarbeit ein Simulationsmodell für das Prozessmodell des dBOP Projekts erarbeiten. Mit Hilfe der bereitgestellten Tools des Projekts können bereits durchgeführte Geschäftsprozesse analysiert werden. Dabei kann jede Aktivität für sich mit Hilfe der konsolidierten Daten von operativen und Prozess-Daten zu einem ausgesprochen guten Simulationsmodell gebaut werden.

Wie in Kapitel 4.2.1 schon erwähnt, wird ein Werkzeug für die Simulationskonstruktion und Ausführungsphase zur Verfügung gestellt werden. D.h. in der Analyse Phase müssen Informationen über den Prozess und seine Aktivitäten für die spätere Simulationskonstruktion gesammelt werden. Die Information die gesammelt werden müssen, sind folgende:

4.3.1 Analyse von einzelnen Aktivitäten

Aktivitäten haben Input-Daten und Output-Daten. Das Ziel ist nun mit Hilfe der Input-Daten eine Simulation zu erstellen, die auf Basis dieser realitätsnahe Output-Daten erstellt. Dabei müssen numerische Output-Daten am besten nach einem Attribut aus der Zusammenführung mit den operativen Daten klassifiziert werden. So lassen sich realitätsnähere Simulationen aufbauen. Ansonsten würde eine Simulation von numerischen Daten, eine willkürliche Verteilung der Output-Parameter ergeben und die Simulation wäre sehr realitätsfern..

4.3.1.1 Allgemeine Analyse von Outputs

Für alle Outputs gilt: Bestimmung durch die Input-Parameter und ihrer zugehörigen operativen Daten. Durch die Nutzung der operativen Daten ist ein realitätsnähere Simulation möglich, da durch die erweiterte Information eine Aktivität und ihr Output besser vorhergesagt werden kann. Die Data Mining Algorithmen dürfen dann nur diese für das Simulationsmodell nutzen. Alle anderen müssen entfernt werden. Im Grundlagen Kapitel wurden die Data Mining Methoden Lineare Regression und Klassifikationsbäume erläutert. Diese beiden werden für die unterschiedlichen Output-Datentypen angewendet. Die Ergebnisse werden der späteren Durchführung der Simulation in einer Datenbank bereitgestellt.

Aber nicht alle Output-Parameter benötigen eine Data Mining Analyse. Einige Parameter können auch immer konstante Werte oder aufsteigende Zahlenwerte zurückgeben (z.B. Primary Keys). Dabei unterscheidet man folgendermaßen:

- konstanter Wert (z.B. Output immer gleich 0)
- Output = Input (Der Input-Wert wird in dieser Aktivität nur weitergegeben)
- ID Wert: Primary Key, der in der Aktivität fortlaufend erstellt wurde.

Tabelle 4.2 zeigt die unterschiedlichen Möglichkeiten mit der ein Output simuliert werden kann.

Output-Ergebnis	Analyse	Implementierung
konstanter Wert	Nur ein Wert als Ergebnis	X = 1
Output = Input	Es ergibt Output = Input	X = Y
ID Wert (Primary Key)	Werte zu 100% unterschiedlich und aufeinanderfolgend	X = MAX(ID) + 1
Nominaler Output	Klassifikationsbaum Erstellung möglich	X = Klassifikationsbaum
Numerischer Output: Typ A	Lineare Regression	X =Lineare Regression
Numerischer Output: Typ B	Klassifikation nach weiterem Attribut des Outputs möglich	X_0 = Klassifikationsbaum X = get X where (Class = X_0)

Tabelle 4.2: Mögliche Simulationen von Werten

4.3.1.2 Analyse von nominalen Outputs

Die Konstruktion von nominalen Outputs wird mit Hilfe des Klassifikationsbaums erstellt. Dabei wird der Output durch die Input-Parameter bestimmt. Die bisherigen Instanzen werden dabei durch einen Klassifikations-Algorithmus analysiert und ergeben einen Baum, der zur Bestimmung des Outputs genutzt werden kann.

Kommt die Analyse zu keinem zufriedenstellenden Ergebnis – Fehlerrate zu hoch -, so kann folgendermaßen vorgegangen werden. Die Klassen können weiter aufgeteilt oder zusammengefasst werden. Dadurch wird vielleicht eine Klassenbestimmung genauer. Bei einer Zusammenfassung von Klassen, muss darauf eine Verteilung der Klassen in den zusammengefügten Klassen angegeben werden. Dies kann auf Grundlage der bisherigen Instanzen geschehen.

4.3.1.3 Analyse von numerischen Outputs

Um numerische Werte zu simulieren kann der Algorithmus für Lineare Regression genutzt werden (Typ A). Dabei ist dies nicht immer sinnvoll. Wenn die Analyse mit Linearer Regression zu einer zu hohen Fehlerquote führt oder aus einfachen Gründen keinen Sinn ergibt, so muss die Analyse erweitert werden.

Eine Erweiterung ist die Aufteilung der numerischen Werte in einen nominalen Wertebereich (Typ B). Durch die Gegebenheiten des dBOP Projekts können numerische Werte in Klassen von den operativen Daten aufgeteilt werden. Für die spätere Implementierung geschieht die Output-Bestimmung in zwei Schritten. Als erstes wird die nominale Klasse durch einen Klassifikationsbaum und anschließend wird per Zufall ein numerisches Element ausgewählt, dass der simulierten nominalen Klasse entspricht.

Als Beispiel kann das Autovermietungsbeispiel dienen. Dabei sucht sich der Kunde ein Auto aus (SelectCar Aktivität). Um dies zu simulieren, wäre eine lineare Regression auf die carID auf Grundlage des Kunden und seinen Attributen (average_income, etc.) nicht sinnvoll. Was aber Sinn machen könnte, wäre eine Simulation mit Hilfe eines Klassifikationsbaums für die Typen der Autos (Type der carID in der operativen Car-Tabelle). Eine Simulation auf Basis der Kunden-Attribute auf die Typen der Autos. Anschließend könnte per Zufall eine carID mit der zugehörigen Typ-Klasse ausgewählt werden. Wir werden später im Evaluationskapitel sehen, dass dies einen erheblichen Vorteil bringt.

4.3.1.4 Analyse der Metriken

Für die Metriken (execution time und waiting time) wird ein einfaches Simulationsmodell genutzt. Bei der Analyse werden der Mittelwert und die Standardabweichung gespeichert, mit der auch später die Zeiten konstruiert werden. Eine komplexere Simulation würde über den Rahmen dieser Diplomarbeit gehen. Das Ziel dieser Diplomarbeit ist auch zu zeigen, dass die Ersetzung von „Human Tasks" durch automatisierte Aktivitäten – wie

die der Simulation – zu realitätsnahen Ergebnissen kommt. Dafür wäre eine Simulation der Ausführungszeit nicht nötig.

4.4 Datendefinition und –erhebung

Wie schon im vorherigen Kapitel vorgestellt, besteht diese Phase aus der Vorbereitung der Simulation und der Auswahl der relevanten Input-Eigenschaften für die Simulation.

4.4.1 Vorbereitung der Simulation

Zunächst müssen für jede zu simulierende Aktivität Datenbank-Tabellen erstellt werden, die die Ergebnisse jeder Prozessinstanz speichern werden. Dabei bestehen diese Datenbank-Tabellen aus vier Spalten-Kategorien:

- Primary Key: Mit der ID der Ausführung
- Metriken: Execution Time und Waiting Time (2 Spalten)
- Input-Parameter: x Spalten für jeden Input
- Output-Parameter: y Spalten für jeden Output

4.4.2 Bestimmung des zu simulierenden Prozesses

Als erstes muss dem Framework der zu simulierende Prozess und seine Input-Operation mit den benötigten Attributen zur Verfügung gestellt werden. Dabei wird ein Web Service Port zur Verfügung gestellt, der den Simulations-Prozess aufrufen kann. Dabei sind auch die Input-Parameter (Attribute) des Gesamtprozess zu benennen.

4.4.3 Bestimmung der Simulation für die Input-Parameter

Ist dies geschehen können die verschiedenen Attribute bzw. Input-Parameter bestimmt werden. Um eine möglichst große Anzahl an Prozessdurchläufen zu simulieren, können die gewählten Input-Größen mit verschiedenen Wahrscheinlichkeitsverteilungen bestimmt werden. Dabei werden als Datentypen Integer, (Double) Float, Boolean und String Werte für die Simulation unterstützt.

Für die numerischen Datentypen kann auf die gleiche Weise die Input Parameter simuliert werden. Für boolean Werte ist die Sache noch einfa-

cher, da man nur zwei Werte simulieren muss. String Werte sollten eigentlich keine Auswirkung auf Prozesse haben, trotzdem wurde ein Simulationsmodell für diesen Datentyp zur Möglichkeit der späteren Nutzung herausgearbeitet.

4.4.3.1 Numerische Datentypen (Integer, Float)

Für Integer und Float Datentypen gibt es eine einfache Möglichkeit für die Simulation: eine Wahrscheinlichkeitsverteilung. Diese wurde im Grundlagen Kapitel zur Geschäftsprozesssimulation schon vorgestellt. Für diese Diplomarbeit genügen zwei Wahrscheinlichkeitsverteilungen Die erste ist die Gleichverteilung der simulierten Daten nach Min- und Max-Werten. Dabei wäre eine Simulation von Integer Werten zwischen 1 und 6 jeweils inklusive gleichzusetzen mit einem Würfelspiel. Die zweite Möglichkeit ist die einer Gaußschen Normalverteilung. Die benötigten Parameter hierfür sind der Mittelwert und die gewollte Standardabweichung als Angabe der Streuung des Mittelwerts.

4.4.3.2 Boolean

Die Boolean Simulation ist die einfachste, da sie auch nur zwei Werte annehmen kann. Deswegen wird bei der Simulation auch nur ein Parameter abgefragt und das ist, wie oft (in Prozent) der Wert true ergeben soll. Als Beispiel wäre z.B. das Feld „male?" als boolean, das bestimmt, ob diese Person männlich oder weiblich ist.

4.4.3.3 String

Ist ein String Parameter als Input verlangt, so können die Daten z.B. nicht durch eine Normalverteilung von Min- und Max-Werten simuliert werden. Das ergäbe keinen Sinn. Als Ersatz können Möglichkeiten von Werten für den Datentypen durch ein Semikolon getrennt eingegeben werden. Diese Namen werden als Parameter gleichverteilt übergeben. Möchte man einen Wert häufiger haben als andere, so gibt man einfach zweimal in der Liste der Möglichkeiten an. Eine Beispiel-Eingabe wäre: Mayer;Bauer;Mayer;Haußmann.

4.4.4 Bestimmung der Prozessinstanziierungen

Wie im Grundlagen Kapitel schon erläutert, gibt es zwei grundlegende Möglichkeiten die Abstände der Prozessinstanziierung zu bestimmen. Die erste ist mit einem konstanten Abstand der Instanziierungen. Die zweite ist eine stochastische und komplexere Instanziierung. In diesem Framework beschränkt sich die Instanziierung auf eine konstante und eine einfache stochastische Möglichkeit. Die stochastische Möglichkeit ist eine normalverteilte Instanziierung. Dabei wird die konstante Instanziierung mit einer Streuung ausgeführt. Dies soll mögliche Hoch- und Tiefdruck Zeiten in einem Prozess simulieren.

4.5 Simulationskonstruktion und –durchführung

Zeitlich gesehen geschieht der wichtige Teil der Konstruktion der Simulation schon vor der Datendefinition der Input-Parameter, nämlich bei der Analyse des Geschäftsprozess. Auf Grund der Bestimmung des Prozess und seinen zugehörigen Input-Parameter sollte aber erst jetzt die Simulation konstruiert werden, da auf Grund der Auswahl der Input Daten und der Prozessinstanziierung ein besseres Verständnis zum Geschäftsprozess und der gewünschten Simulation herrschen sollte. Nachdem dies nun klargestellt wurde, kommen wir zur eigentlichen Konstruktion der Simulation.

4.5.1 Simulationskonstruktion

Es wird ein Simulations-Service für die Simulationskonstruktion zur Verfügung gestellt. Dabei werden die zuvor gesammelten Informationen in dem Simulation Service implementiert. Dabei unterscheidet man zwischen der Implementierung der Metriken und der Output-Parameter.

4.5.1.1 Simulation der Metriken

Für die Simulation der Metriken werden die bereits gesammelten Informationen aus der Analysephase genutzt. Eine einfache Berechnung mit Hilfe eines „Randomizer" und dem ermittelten Mittelwert und der Stan-

dardabweichung wird genutzt. Dies würde ähnlich wie beim Process Mining gelöst.

4.5.1.2 Simulation der Output-Daten

Wie in der Analyse Phase schon vorgestellt, werden Data Mining Algorithmen auf die Output Daten ausgeführt. Nun sind diese auf Basis der operativen Daten geschehen, d.h. das auch in der Simulation eine Brücke zu den operativen Daten geschaffen werden muss. Die Implementierung muss somit eine Verbindung zu den operativen Daten herstellen und je nach Input Parameter die weiteren Informationen aus den operativen Daten erhalten, um den Output Wert bestimmen zu können. Anschließend werden die erstellten Analysen nach dem Schema aus Tabelle 4.2 implementiert. Näheres zur Implementierung dann in Kapitel 5.

4.5.2 Simulationsdurchführung (Simulations-Service)

Nachdem die ganzen Aktivitäten analysiert wurden und die Simulationen konstruiert wurden, werden die Aktivitäten durch den erstellten Simulations-Service ausgetauscht. Dabei muss auch beachtet werden, dass die verschiedenen Variablen im Prozess richtig zugeordnet werden. Durch den Austausch des Service kann es da zu Unstimmigkeiten kommen.

Der Simulations-Service erhält als Input-Daten auch die Input-Daten der eigentlichen Aktivitäten. Dazu kommt noch der Name der zu ersetzenden Aktivität, damit die richtige Operation ausgeführt wird und die Ergebnisse in die richtige Datenbank-Tabelle gelangen.

4.6 Bereitstellung der Ergebnisse

Die Bereitstellung der Ergebnisse geschieht tabellarisch. Dabei werden die Information zu jeder Aktivität aufgeteilt dargestellt. Die bei der Konstruktion vorgestellte Datenbank Tabelle für jede Aktivität wird während der Simulationsdurchführung gefüllt.

Anschließend kann jede Tabelle für sich abgefragt und analysiert werden. Außerdem steht eine Export-Funktion zur Verfügung um die Datenbank-

Tabellen zu extrahieren und weiterbearbeiten zu können. Natürlich können auch die Tabellen direkt abgefragt werden. Dies sollte aber mit Vorsicht geschehen, damit man die Ergebnisse nicht verfälscht.

Um einen Vergleich der Simulation mit den realen Daten zu erstellen, müssen die realen und simulierten Daten miteinander „gejoint" werden. Dies kann durch einen „Join" der beiden Tabellen (real und simuliert) auf Basis der Input-Parameter geschehen. Dabei müssen vielleicht auch die operativen Datenbank-Tabellen miteinander verknüpft werden, damit ein Vergleich auf Basis von Klassen geschehen werden muss und nicht auf numerische Daten.

5 Implementierung des Framework

Die Implementierung des Konzepts basiert auf vielen verschiedenen Bauteilen. Als erstes wird ein Überblick der Implementierung gegeben. Anschließend wernde schon implementierte Vorarbeiten und Daten vorgestellt, die zur Verfügung stehen, die die Implementierung des Frameworks erleichtern. Außerdem werden genutzte Software und Schnittstellen vorgestellt, die die Entwicklung des Frameworks ermöglicht haben, wie der WebSphere Integration Developer, und auf die das Framework aufbaut. Anschließend kommen wir zum eigentlichen Framework, dass den Arbeitstitel dBOPSim (kurz für: deep Business Optimization Platform Simulation) bekommen hat. Nach dem Aufbau der GUI werden die einzelnen implementierten Modellelemente von dBOPSim erläutert. Für die eigentliche Konstruktion des Simulation Services gibt es auch noch ein Kapitel.

5.1 Überblick der Implementierung

Um ein besseres Verständnis für die unterschiedlichen Teile des Simulations-Frameworks zu bekommen, zeigt Abbildung 5.1 das Zusammenspiel der unterschiedlichen Implementierungen für die Simulation. Dabei werden zuerst der Prozess und seine Daten im Allgemeinen mit Hilfe des BPAClient analysiert. Diese Analyse-Ergebnisse werden anschließend für die Erstellung des Simulation Service genutzt, der anschließend auf den IBM Process Server „deployt" wird. Außerdem werden in der Analyse die Parameter der Aktivitäts-Tabellen festgestellt, die mit Hilfe des dBOPSim Tools in der DB2 Datenbank erstellt werden. Anschließend kann der Prozess auch mit dem dBOPSim Tool auf dem IBM Process Server gestartet werden. Die Simulations-Ausführung schreibt während der Prozess-Ausführung die Simulations-Parameter in die DB2 Datenbank. Nach Abschluss der Simulation können die Daten aus der DB2 Datenbank ausgelesen werden.

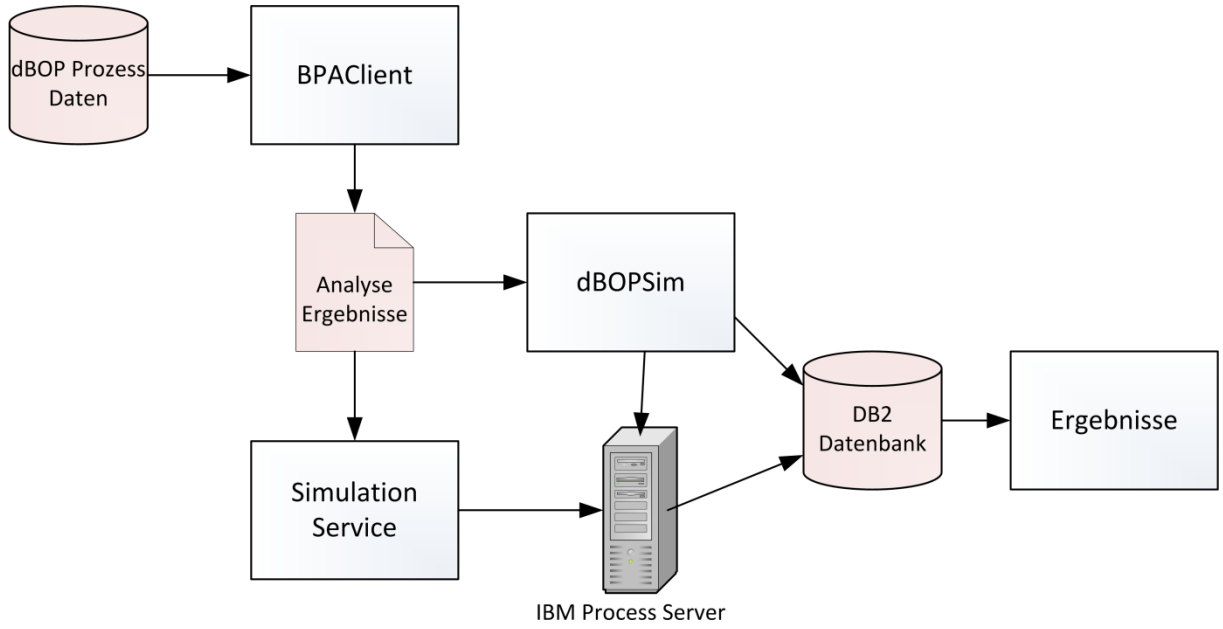

Abbildung 5.1: Überblick des Zusammenspiels der verschiedenen Implementierungen

5.2 Vorarbeiten für die Simulation

Für diese Diplomarbeit wurden schon viele Vorarbeiten geleistet, wie schon in Kapitel 3 vorgestellt. In diesem Kapitel werden die Implementierung der vorgestellten Projekte vorgestellt und inwiefern diese in dieser Diplomarbeit genutzt werden.

5.2.1 BIAEditor

Der BIAEditor ist das Matching Tool für operative und Prozessdaten. Wie schon in Kapitel 3.1.1 erwähnt, wurde der BIAEditor für die Business Impact Analysis erstellt. Die Matchings, die im BIAEditor vorgenommen werden, können per XML Datei extrahiert werden und so für eine Konsolidierung verwendet werden.

In dieser Diplomarbeit wird die zugehörige XML-Matching Datei des Testprozesses benötigt, um die relevanten Daten zu joinen und somit, die Data Mining Algorithmen durchzuführen. Abbildung 5.2 zeigt die Verknüpfung der Prozessvariablen mit den operativen Tabellen in der Datenbank.

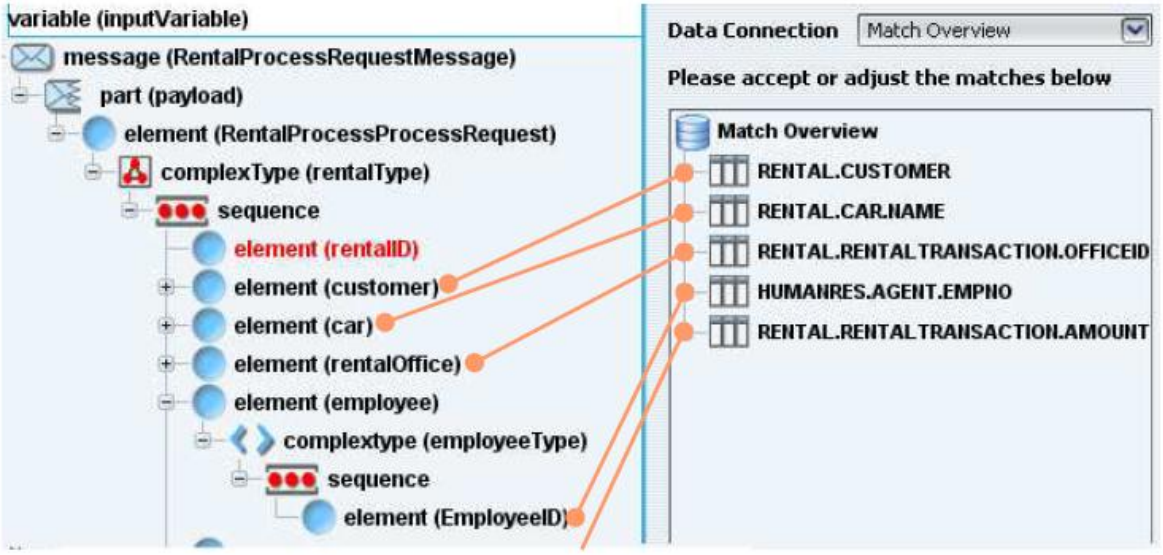

Abbildung 5.2: Matching View des BIAEditor (22)

So werden nun die Prozessvariablen mit weiteren Informationen aus der operativen Datenbank angereichert werden.

5.2.2 dBOP Editor

Der dBOP Editor ist das Haupt-Tool des dBOP Projekts. Hier können Prozesse nach dem dBOP Modell eingepflegt werden, die nach einer Optimierung verlangen. Ist dies geschehen, können die vorgestellten „Optimierungspattern" durchgeführt werden und Geschäftsprozesse verbessert werden. In dieser Diplomarbeit wurde er nicht genutzt, da der Testprozess aus Kapitel 3.3 schon implementiert war.

5.2.3 BPAClient

Im Rahmen des dBOP-Projekts wurde in (25) ein Client entwickelt, der den Process Insight Repository implementiert. Dabei werden anhand der bekannten Matchings aus dem BIA-Editor operative Daten und Prozess-Daten gejoint. Mit Hilfe des genutzten WEKA Tools (wird in Kapitel 5.3 näher erläutert) können Data Mining Algorithmen auf diese Daten angewendet werden und später ausgelesen werden, um Muster zu erkennen und somit Verbesserungen, die vorher unerkannt blieben, auszuführen.

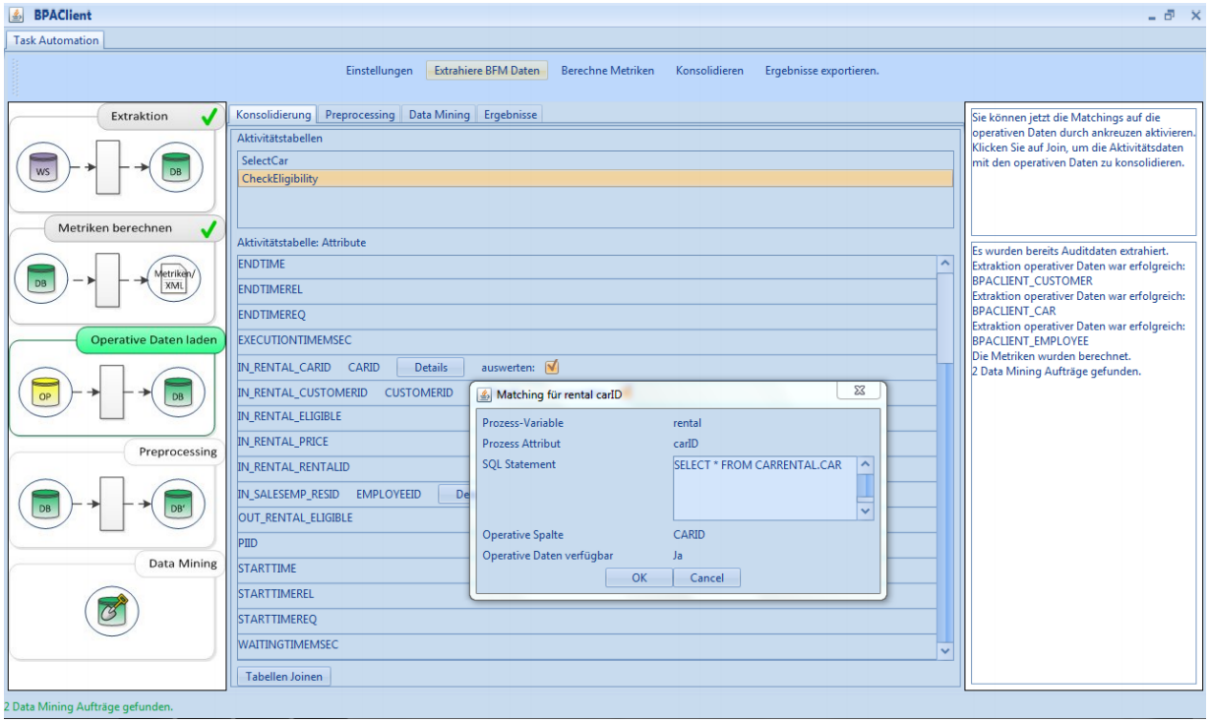

Abbildung 5.3: GUI des BPAClient (Konsolidierungs-Ansicht) (25)

Der BPAClient ist die Implementierung für die Erstellung des Insight Repository aus den operativen Daten. Als Inputs werden die operativen Daten, die Prozessdaten, sowie das XML-basierte Matching aus dem BIA-Editor benötigt, um den Repository zu erstellen. Ist dieser erstellt können die Metriken Berechnungen angeschaut und weitere Data Mining Analysen durchgeführt werden. Die Ergebnisse können exportiert werden und somit für weitere Analysen genutzt werden. In dieser Diplomarbeit wird dies als Input für die Simulationserstellung genutzt.

5.3 Genutzte Software und Schnittstellen

Das Framework wurde nicht auf einer grünen Wiese implementiert, sondern baut auf anderer Software auf. Als Programmierumgebung wurde der WebSphere Integration Developer von IBM genutzt. Als Datenbank wurde IBMs DB2 Windows Version genutzt. Für die Data Mining Algorithmen wurde die Open Source Software WEKA benutzt. Weiterhin wurden Quellcodes des dBOP Editor und des BPAClients benutzt, die im vorherigen Kapitel vorgestellt wurden. Als BPEL Engine für die Geschäftsprozessausführung wurde der Process Server von IBM genutzt.

5.3.1 WebSphere Integration Developer

Der WebSphere Integration Developer ist eine Eclipse-basierte Entwicklungsumgebung für die Erstellung von Integrationslösungen. Dabei steht die Service-oriented Architecture im Vordergrund. Die Entwicklung geschieht unter dem WebSphere Process Server um Services und Geschäftsprozesse miteinander zu verbinden. Dabei werden auch Java-Implementierungen und Möglichkeiten der manuellen Bearbeitung von Services bereitgestellt.

Dieses Tool wurde sowohl für die Java-Implementierung des dBOPSim Tools (Kapitel 5.5) genutzt, als auch für die Implementierung und Integration des Simulation Service (Kapitel 5.6).

5.3.2 IBM DB2

Die Datenbank, die für die das dBOP Projekt und auch die Simulation genutzt wird, ist IBMs Datenbank Software DB2. DB2 ist für alle gängigen Betriebssysteme erhältlich, sowie auch für IBMs Mainframe z/OS. Da die Entwicklung des Frameworks auf Java basiert, kann mit den JDBC Driver für DB2 die Datenbank einfach angesprochen werden.

Alle Informationen laufen auf diese Datenbank zusammen. Zwar geschieht die Prozessausführung und später auch die –simulation auf dem Process Server, aber am Ende jedes Prozesses, werden die Erkenntnisse durch den Simulation Service in die DB2 Datenbank geschrieben.

5.3.3 Workflow-Management-System

Als Workflow-Management System wurde die IBM WebSphere Software genutzt. Der WebSphere Application Server und der WebSphere Process Server werden für das Workflow-Management genutzt. Dabei wird später, wie bereits mehrmals erwähnt, ein eigener Simulation Service erstellt, der vom Geschäftsprozess anstatt des normalen Prozessaufruf aufgerufen wird.

Der IBM WebSphere Business Choreographer kann zur Administrierung des Workflow-Management-System genutzt werden, falls sich Probleme bei der Simulation bzw. der Prozessausführung darstellen.

5.3.4 WEKA

WEKA ist eine Data Mining Software der Machine Learning Group, entwickelt an der Universität von Waikato, Neuseeland. Diese in Java geschriebene Software ist eine Kollektion von verschiedenen Data Mining Algorithmen. Dabei können Daten nicht nur innerhalb des Tools genutzt werden, sondern die Algorithmen können auch als Service vom eigenen Code genutzt werden.

Im Framework werden die bereitgestellten Algorithmen genutzt um die Datenanalysen durchzuführen. Dies geschieht mit dem BPAClient der im vorherigen Kapitel vorgestellt wurde. Die analysierten Daten können betrachtet werden und für die Simulationskonstruktion später verwendet werden. Wie im Modell schon vorgestellt, werden die Ergebnisse der Analyse als Service implementiert.

Das Tool wurde schon im BPAClient integriert und dient der Analyse der Output Daten, wie im Konzept vorgestellt. Dort werden die einzelnen Algorithmen, wie Lineare Regression auf die Output Daten ausgeführt und die Modelle extrahiert. Außerdem wurde es für die Bereitstellung und Analyse der Simulations-Ergebnisse später auch in das Framework eingebaut. (26)

Abbildung 5.4 zeigt die graphische Oberfläche des Weka Explorer.

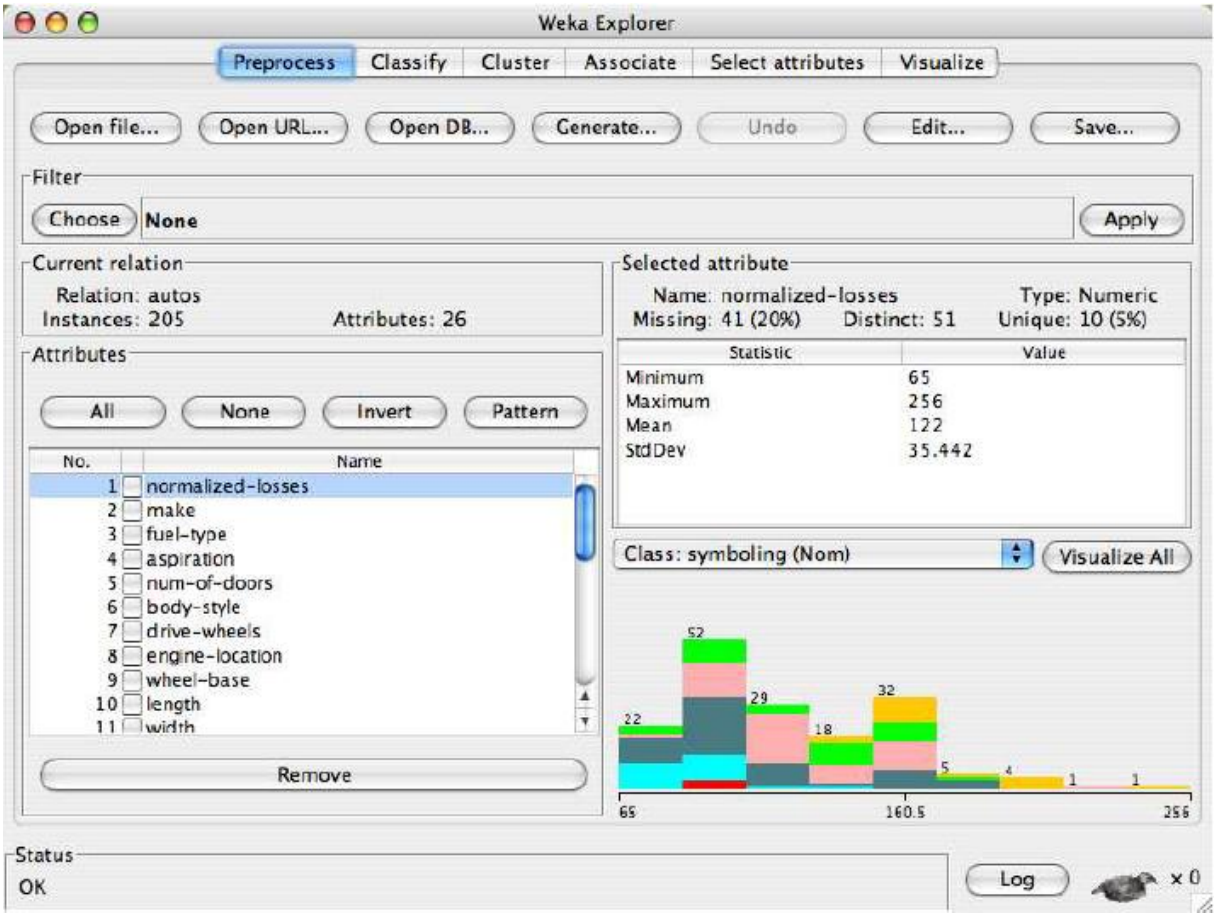

Abbildung 5.4: WEKA Explorer: Ansicht für das Preprocessing (26)

5.4 Planung und Analyse

Für die Planung und die Analyse werden die zur Verfügung stehenden Tools genutzt. Wurde ein Prozess ausgewählt, so wird er mit Hilfe des BPAClient analysiert. Jede Aktivität wird mit operativen Daten gejoint, um eine möglichst realitätsnahe Simulation zu erstellen.

Für jede Aktivität gelangt man in das Preprocessing Panel des BPAClient, ähnlich Abbildung 5.4. Man entfernt nun alle nicht relevanten Attribute, die für die Simulation keinen Sinn ergeben. Dazu gehören die Information, ob spätere Aktivitäten durchlaufen wurden, die Start- und Endzeiten und das (mehrfache) Vorkommen von IDs.

In der Analyse Phase werden jetzt die Information für die spätere Konstruktion gesammelt. Für jede Aktivität werden die Metriken und die Output-Daten analysiert. Für die Metriken ist dies vergleichsweise einfach. Im Preprocessing Panel wird auf das zugehörige EXECUTIONTIMEMSEC und

WAIITINGTIMEMSEC geklickt und die zugehörigen Mittelwerte und Standardabweichung ausgelesen. Abbildung 5.5 zeigt ein Beispiel. Der Mittelwert beträgt 413,14 ms und die Standardabweichung 312,935 ms.

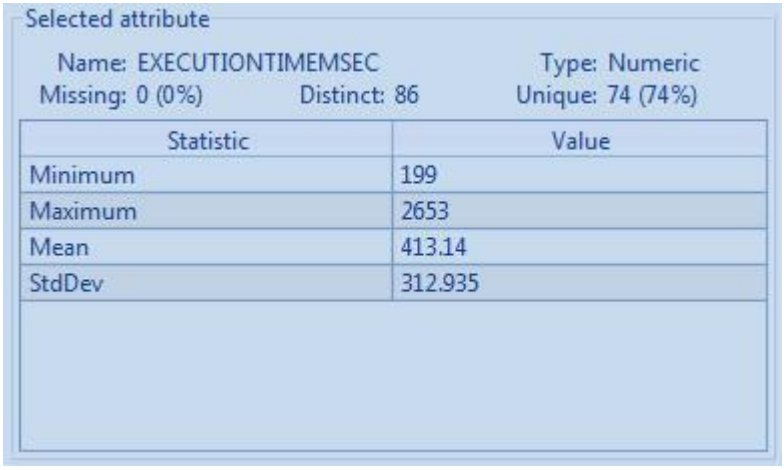

Abbildung 5.5: Eigenschaften der Ausführungszeit einer Aktivität

Die Analyse für die Output-Daten ist erheblich komplexer. Wie im Konzept-Kapitel schon erwähnt, geschieht dies mit Data Mining Algorithmen. Es kann aber auch vorkommen, dass Output-Werte keiner Data Mining Analyse bedürfen. Dies wurde in Kapitel 4.3 schon erläutert, wenn die Output-Werte z.B. nicht verändert werden. Das kann auch im Preprocessing Panel schon herausgeschaut werden.

Ist aber eine Data Mining Analyse nötig, so ist der nächste Schritt die Output-Daten z.B. mit dem „Classifier: Lineare Regression" für numerische Werte und dem „Classifier J48" (Klassifikationsbaum) für nominale Werte zu analysieren. Dies mag nicht gleich beim ersten Mal zum gewünschten Ergebnis führen, dass man wieder ein Schritt zurück muss, und im Preprocessing Panel, doch noch einige Attribute entfernen muss. Außerdem muss man beachten, ob nicht eine Klasse für numerische Werte auf Basis der operativen Daten geschehen kann. Als Beispiel aus dem Geschäftsprozessbeispiel ist das Simulieren der Auswahl einer carID. Durch eine Klassifizierung der carID zu Autotypen kann man leichter simulieren, welcher Kunde welche Autoklasse auswählt, anstatt welches Auto er explizit auswählt. In Kapitel 6 wird die Auswahl des Klassifikationsbaums zu diesem

Beispiel näher erläutert. Dort wird an zwei Beispielen die Analyse Phase noch gezielter betrachtet.

5.5 dBOPSim (deep Business Optimization Platform Simulation)

Kommen wir nun zum eigentlichen Programm, der die Simulation begleitet. Dieses Kapitel ist ähnlich wie das Konzept-Kapitel nach den Phasen aufgebaut. Als erstes gibt es einen Überblick über die einzelnen Phasen der Simulation anhand des GUI für die Simulation. Anschließend werden alle Phasen der Simulation einzeln erläutert und ihre Implementierung näher gebracht.

5.5.1 GUI

Die GUI wurde mit Hilfe der Java Klassen Swing und AWT umgesetzt. Dabei ist die GUI so aufgebaut, dass sie dem Vorgehensmodell nach mit ausgeführt werden kann. Außerdem wurde für ein besseres Aussehen die Substance[3] Look And Feel Erweiterung für Swing Komponenten genutzt. Für die genutzten Java GUI Komponenten kann man sich ein Bild in (27) machen.

5.5.2 Aufbau

Abbildung 5.6 zeigt den Aufbau der GUI nach den Klassen, die erstellt und genutzt wurden. Dafür wurde ein vereinfachtes UML-Modelldiagramm genutzt, um die wichtigen Inhalte der einzelnen Phasen zuzuordnen. Dabei sind die weißen Kästchen die genutzten Klassen der GUI, die roten beschreiben die Hauptaufgaben jeder Spalte und die blauen machen klar, zu welcher Phase die Komponenten gehören.

[3] http://java.net/projects/substance/

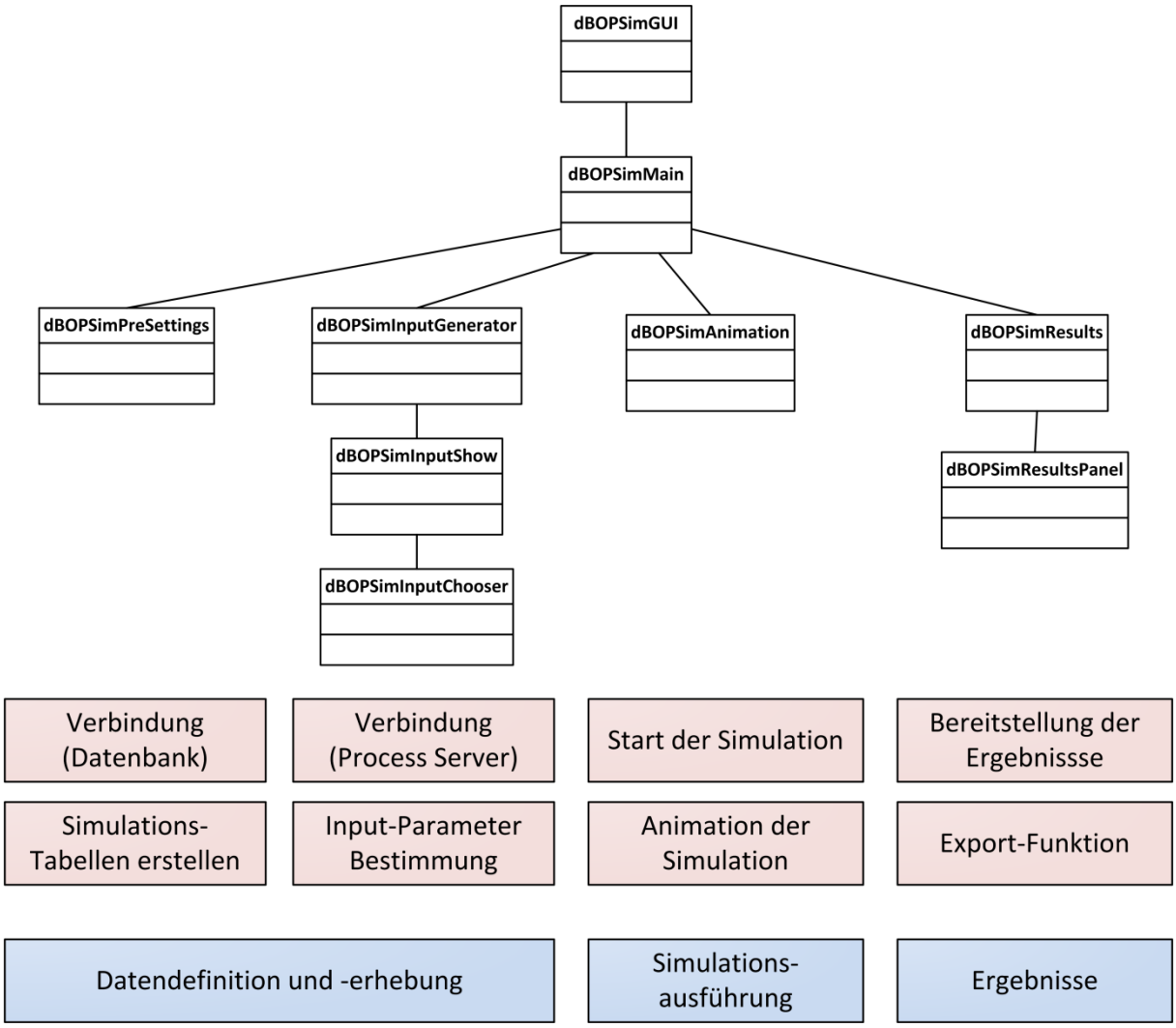

Abbildung 5.6: Aufteilung der Komponenten der dBOPSim GUI

Die GUI ist ein JFrame (dBOPSimGUI), das hauptsächlich aus einem JTabbedPane (dBOPSimMain) besteht, das wiederum die verschiedenen Phasen der Geschäftsprozesssimulation als Tabs besitzt. Abbildung 5.7 zeigt die vier Tabs, die die vier Spalten aus Abbildung 5.6 repräsentieren.

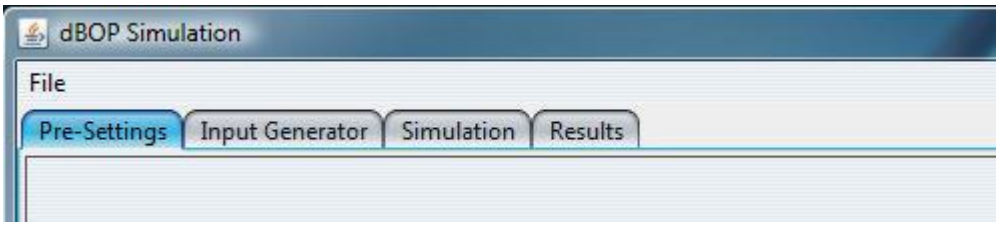

Abbildung 5.7: Tabs der GUI

Die Inhalte der Tabs sind JScrollPanes, damit die Inhalte auch bei Veränderung der Fenster-Größe durch Scrollen noch sichtbar sind. Im Pre-

Settings Tab wird die Verbindung zur Datenbank erstmals hergestellt und es besteht die Möglichkeit, die Simulations-Tabellen zu erstellen. Anschließend kann die Verbindung zum Web Service auf dem Process Server hergestellt werden und die benötigten Input-Parameter bestimmt werden. Im nächsten Tab - der Simulation - kann der User die Simulation mit den ausgewählten Parametern ausführen. Anschließend werden die Ergebnisse im Ergebnisse-Tab dargestellt. Eine genauere Erklärung der vier Spalten folgt nun.

5.5.3 Pre-Settings

Im Pre-Settings Tab (JScrollPane dBOPSimPreSettings) werden die DB2 Eigenschaften (DB-Name, DB-User, DB-Passwort) eingegeben, mit der das Programm die neuen Tabellen erstellen wird, die für die Simulation erforderlich sind. Die Verbindung geschieht mit Hilfe des JDBC Treibers, der eine Verbindung zu einer DB2 Datenbank so einfach wie möglich macht. Anschließend muss für jede simulierte Aktivität eine neue Tabelle erstellt werden, die die Attribute aus Kapitel 4.4.1 enthält. Für selectCar aus dem Beispiel-Prozess wäre dies folgende Tabelle:

```
CREATE TABLE SIM_SELECTCAR
(ID INT NOTNULL,
 EXECUTIONTIME INT,
 WAITINGTIME INT,
 IN_CARID INT,
 IN_CUSTOMERID INT,
 IN_ELIGIBLE VARCHAR(5),
 IN_PRICE DOUBLE,
 IN_RENTALID INT,
 OUT_CARID INT,
 OUT_PRICE INT,
 PRIMARY KEY (ID))
```

Listing 5.1: CREATE TABLE für eine zu simulierende Aktivität

Das ist Listing 5.1 vorgestellte CREATE TABLE kann dann in die große JTextArea des Tabs eingefügt werden und erstellt werden. Mit einem Klick auf „Create Tables!" wird die Datenbank Tabelle erstellt. Es müssen nun alle benötigten Datenbank-Tabellen für die Simulation so nacheinander erstellt werden, damit der Service später die Ergebnisse richtig eintragen kann. Abbildung 5.8 zeigt das Pre-Settings Panel.

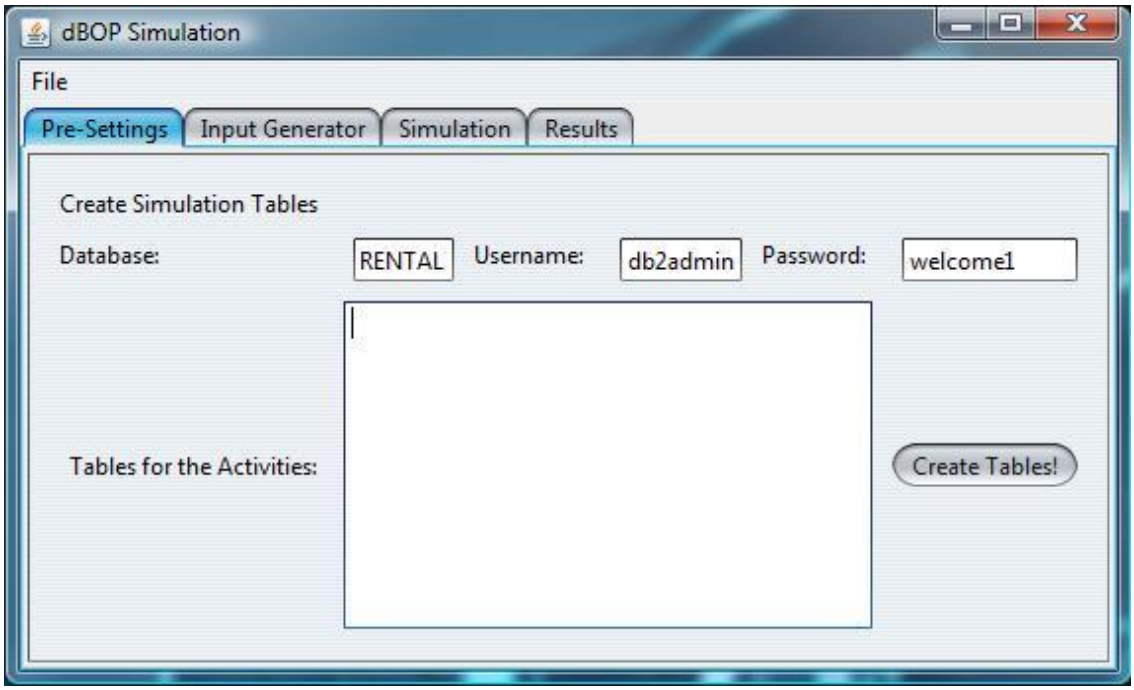

Abbildung 5.8: dBOPSim Pre-Settings Panel für die Erstellung der Simulationstabellen in die Datenbank

5.5.4 Input-Generator

Der Input Generator hat zwei Aufgaben. Als erstes wird der zugehörige WebService und die zugehörige WSDL Datei mit den Parameter der Simulation ausgesucht. Ist das geschehen, werden die Parameter geparst und man kann im Tool die Parameter-Auswahl vornehmen. Anschließend kann auch der zeitliche Abstand der Instanziierung des Prozesses in der Simulation ausgeführt werden. Alle Teile werden nun einzeln genauer erklärt.

5.5.4.1 Verbindung zum Web Service (dBOPSimInputGenerator)

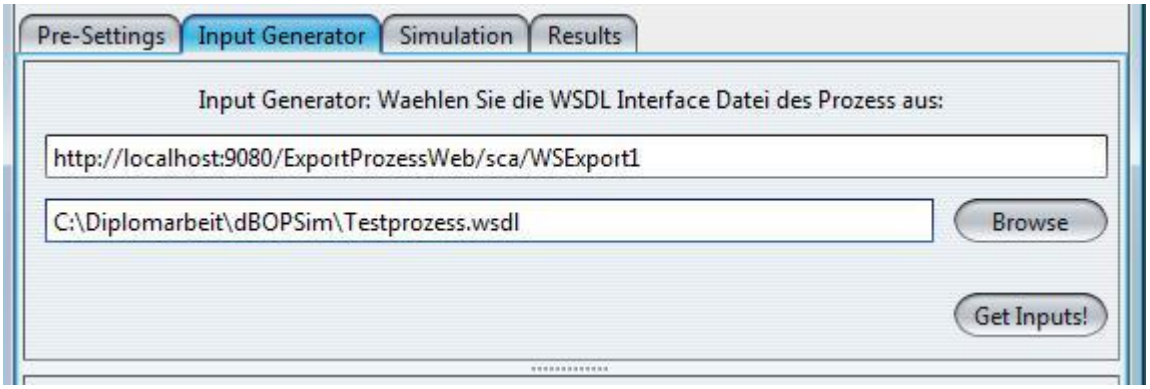

Abbildung 5.9: Auswahl des Web Services und der Schnittstelle

Abbildung 5.9 zeigt die Parameter für die Verbindung zum Prozess. Beide zu parsende Dateien sind WSDL Dateien, die XML-basiert sind. WSDL Dateien wurden bereits im Grundlagen Kapitel 2.5 über Web Services schon vorgestellt. Nun müssen diese Dateien ausgelesen werden, um den dynamischen Web Service Caller des Frameworks die richtigen Parameter zu übergeben, damit ein Aufruf des Geschäftsprozesses mit den simulierten Instanzen möglich ist.

5.5.4.2 Parsen des Web Services

Die WSDL Datei des Web Services kann mit Hilfe des Anhang „?wsdl" an der URL des Web Service aufgerufen werden. Für das „invoken" des Prozess über den Web Service sind verschiedene Parameter nötig. Dazu sind neben dem Namespace des Web Services, der Name, der Port sowie der Endpoint (die Aufruf-Operation des Gesamtprozess) nötig. Außerdem steht auch der Name des Input Parameters für die Operation in dieser Datei.

Die Datei wird mit Hilfe der javax.xml.parsers.* Klasse geparst und anschließend mit der org.w3c.dom.* Klasse in ein DOM-Dokument umgewandelt. Damit können Elemente nach Eigenschaften in XML Dokumenten durchsucht werden. Dadurch ist es einfach alle Information aus dem Dokument zu erhalten.

5.5.4.3 Parsen der Input Daten

Nachdem nun für den Web Service alle relevanten Daten bekannt sind, muss noch die genaue Beschreibung des Input Parameters herausgefunden werden. Die Inputs von Geschäftsprozessen sind oftmals sogenannte Data Objects (Business Objects im WebSphere Integration Developer). Diese beinhalten dann die eigentlichen Daten, wie Name als String, oder customerID als Integer. Die Beschreibung dieser Objekte kann auf unterschiedliche Weise geschehen. In dieser Diplomarbeit wurde die WSDL Beschreibung document/wrapped genutzt. Dabei wurden die Input Typen als <complex-type> mit einer <sequence> von <element> definiert.

```
<xsd:element name="TestprozessInput">
   <xsd:complexType>
      <xsd:sequence>
         <xsd:element name="customerID" type="xsd:integer" />
         ...
      </xsd:sequence>
   </xsd:complexType>
</xsd:element>
```

Listing 5.2: Beispiel eines Input Data Objects

Listing 5.2 zeigt den zu parsenden Teil der Interface Datei. Für jedes Element muss nur der Name und der Typ ausgelesen werden, damit die Auswahl für die Simulation geschehen kann.

5.5.4.4 Input-Parameter Auswahl (dBOPSimInputShow)

Nachdem nun alle relevanten Daten des Web Services und des Prozesses geparst wurden, werden als nächstes die Parameter für die Input-Daten Simulation gewählt. Dabei können je nach Datentyp verschiedene Simulationsverfahren gewählt werden. Abbildung 5.10 zeigt ein Beispiel Input-Objekt mit allen implementierten Simulationsdatentypen und seinen möglichen Parametereinstellungen. Die „customerID" wird von 1 bis 25 gleichverteilt aufgerufen. Dabei wird per Zufall auch ein „name" aus der eingegebenen Liste ausgewählt. Die Boolean-Variable „isMale" soll in 95% der Fälle wahr sein und die Größe („height") soll normalverteilt werden: Mittelwert 1.80, Standardabweichung 0.20.

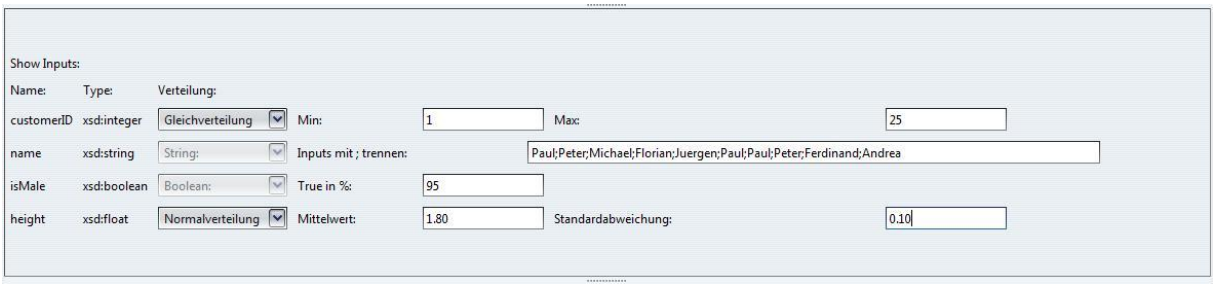

Abbildung 5.10: Input Auswahl der Parameter

5.5.4.5 Prozessinstanziierungen (dBOPSimInputChooser)

Als letzte Einstellung, bevor man den Prozess simulieren kann, werden noch die Parameter für die Häufigkeit und zeitliche Abstände der Instanziierung des Geschäftsprozesses benötigt. Dabei stehen folgende Parameter zur Auswahl: Anzahl der zu simulierende Instanzen und der zeitliche Ab-

stand (gleichverteilt oder mit einer Streuung). Abbildung 5.11 zeigt die Auswahl mit Beispiel-Eingaben.

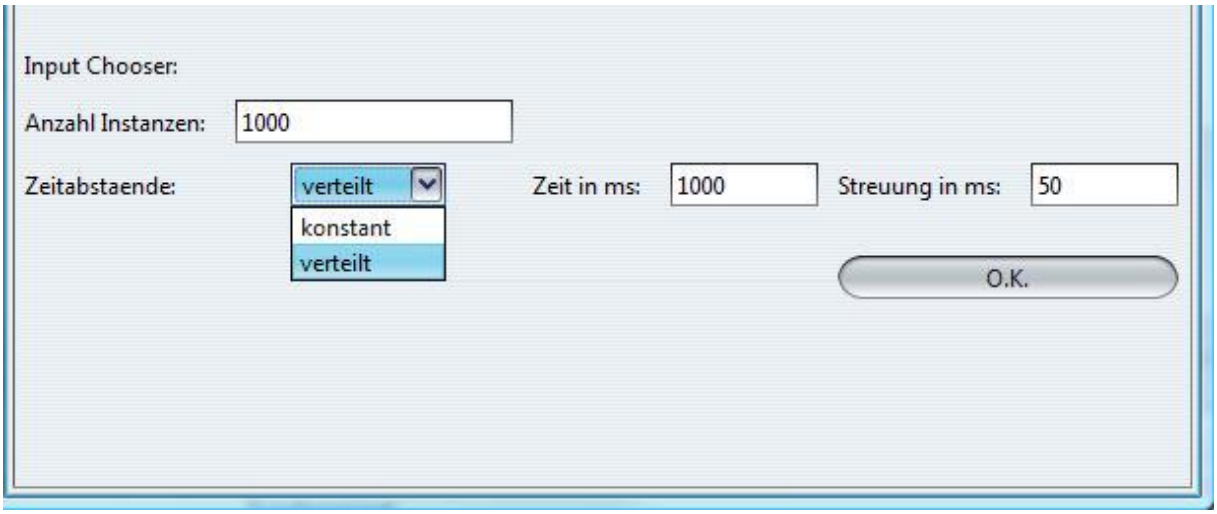

Abbildung 5.11: Auswahl der Prozessinstanziierung

5.5.5 Simulation (dBOPSimAnimation)

Die Simulation geschieht dann auf Basis der ausgearbeiteten Data Mining Ergebnisse und des neuen „Deployment" des Simulationsmodells. Dies wird im Simulation Service geschehen, der im nächsten Kapitel erläutert wird. Dann kommt man wieder zurück und kann die Simulation aus dem dBOPSim Tool starten.

Während die Simulation auf dem IBM Process Server läuft, steht im Simulation Tab des dBOPSim eine Konsole zur Verfügung, die anzeigt mit welchen Input-Daten der gerade simulierte Prozess ausgeführt wird.

5.5.6 Ergebnisse (dBOPSimResults)

Ist die Simulation vollständig gelaufen, gelangt man zum Ergebnis der Simulation. Hier werden die Ergebnisse der Simulation angezeigt und können auch aus der Datenbank exportiert werden. Als Ausgabe für die DB2 Tabellen wurde das WEKA Preprocessing Panel genutzt. Da kein JDBC Treiber für DB2 im WEKA vorinstalliert ist, wurde die Lösung vom BPAClient genutzt. In (25) wurde diese Lösung erläutert und implementiert. Das zugehörige Panel wurde in die Arbeit implementiert. So kann man die Ergebnisse der Simulation schon einmal direkt betrachten.

Die Betrachtung der Ergebnisse geschieht für jede Aktivität für sich. Durch Eingabe des Tabellen-Namens in der zusätzlich eingebauten Abfrage wird das WEKA Preprocessing Panel mit den Daten neu geladen. Außerdem enthält das Panel eine Export Funktion, mit der man die Daten der Simulation z.B. in eine .csv Datei extrahieren kann und möglicherweise weiterverwenden kann. Dazu einfach auf den Button „Save…" drücken. Abbildung 5.12 zeigt eine Beispiel-Ausgabe der Tabelle „SIM_SELECTCAR".

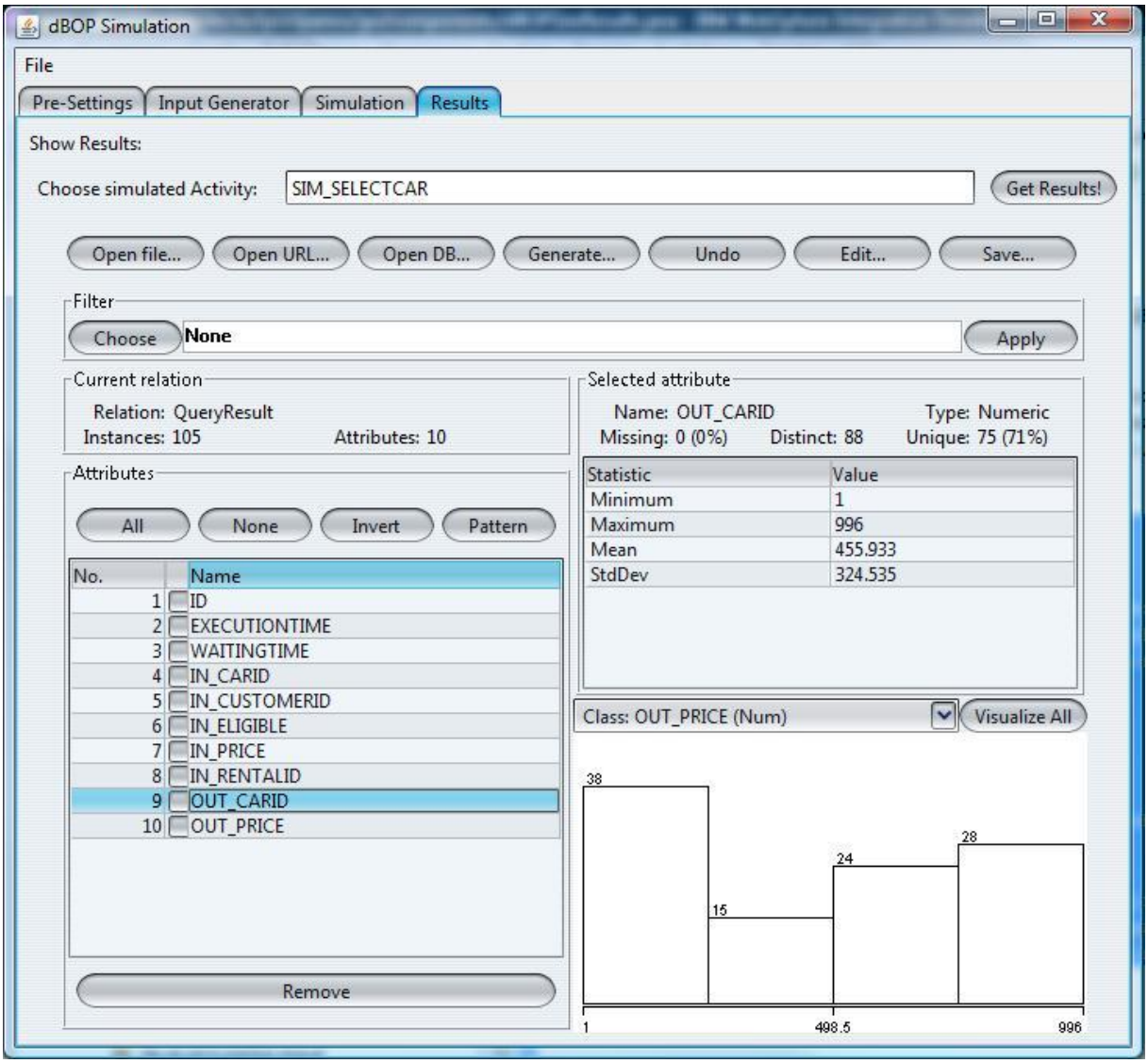

Abbildung 5.12: Bereitstellung der Ergebnisse im dBOPSim Tool

5.6 Simulation Service

Für die Simulation der einzelnen Aktivitäten wurde ein Simulations-Service implementiert, der die einzelnen Aktivitäten mit seinen Input- und Output-Daten simuliert. Dabei muss neben den zu simulierenden Input-

Daten das Simulationsmodell übergeben werden, damit die Output-Daten simuliert werden können.

Die Daten wurde in der Analyse Phase gesammelt und können nun im Service implementiert werden. Dabei kann die allgemeine Schnittstelle für den Service genutzt werden, die im Folgenden erläutert wird. Innerhalb des Service muss dann auch die Simulation für jede Aktivität implementiert werden. Es stehen Hilfsfunktion für die Simulation der Metriken und die Verbindung zur DB2 Datenbank zur Verfügung.

5.6.1 Interface Simulation Service

Der Simulation Service kann mit Hilfe eines allgemeinen Interface aufgerufen werden. Diese Schnittstelle besteht aus zwei speziellen Geschäftsobjekten, die im WebSphere Integration Developer erstellt wurden. Die zugehörige Operation „simulate" enthält die beiden Geschäftsobjekte als Input- und Output-Parameter. Für beide Geschäftsobjekte werden eine unbegrenzte Anzahl an anyType Datentypen zugelassen, die die einzelnen Parameter bei der realen Aktivitätsausführung ersetzen sollen. Das Geschäftsobjekt für den Input Parameter erhält zusätzlich ein Feld für den Namen der Aktivität, damit innerhalb des Prozess die richtige Simulation für die Aktivität ausgewählt wird. Abbildung 5.13 zeigt das Geschäftsobjekt für den Input aus dem WebSphere Integration Developer.

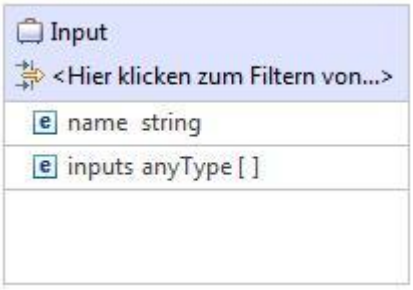

Abbildung 5.13: Geschäftsobjekt Input für den Simulation Service

5.6.2 Einbindung in die Geschäftsprozesse

Der Simulation Service ersetzt die bisherigen Aufrufe der zu simulierenden Aktivitäten. Dabei muss nicht nur der Aufruf umgeleitet werden, sondern auch die Input und Output-Parameter neu zugewiesen werden. In

BPEL muss man die <assign>-Parameter für den Service-Aufruf auf die neuen Parameter umtauschen. Dabei muss auch beachtet werden, dass neben den bisherigen Input-Daten auch der Name des Prozess zu den Inputs zugewiesen wird. Dieser wird benötigt, um die Input- und Output-Parameter für den Service zu bestimmen und die richtige Implementierung innerhalb des Services aufzurufen. Außerdem muss die Datenbank wissen in welche Tabelle die Daten später geschrieben werden sollen.

5.6.3 Java Implementierung (Hilfsfunktionen)

Für die Implementierung des Services wurde Java genutzt. Es wurden Hilfsfunktionen erstellt, die für die Simulation von allen Aktivitäten genutzt werden können. Zum Beispiel müssen bei jeder Simulation die Input- und Output-Parameter ausgelesen werden. Diese stehen als DataObject (commonj.sdo.DataObject) zur Verfügung. Damit diese ausgelesen werden können, gibt es die „getDataObject" Funktion.

Für die einfache Berechnung von Ausführungs- und Wartezeiten steht die „simulateDuration" Funktion zur Verfügung, die auf Basis einer Gaußschen Normalverteilung funktioniert. Die Mittelwerte und Standardabweichungen für die Metriken wurden für jede Aktivität schon herausgearbeitet. Diese beiden Werte werden als Parameter übergeben.

Außerdem ist eine „DBConnect" Funktion implementiert, die, wie der Name schon sagt, eine Verbindung zur DB2 Datenbank mit dem JDBC Treiber herstellt. Als Parameter werden der Datenbank-Name, -User und –Passwort benötigt. Diese Funktion wird benötigt, um die Ergebnisse in die Simulations-Aktivitätstabellen zu speichern. Dies ist wiederum nötig, um später die Ergebnisse auslesen zu können.

5.6.4 Service Implementierung (Aktivitäten)

Die Implementierung geschieht auf Basis des übergebenen Namens für die Aktivität. Dabei wird für jede benötigte Aktivität eine Funktion aufgerufen, die diese Aktivität simuliert. Nach Aufruf der Schnittstellen-

Operation „simulate" wird auf Grund des Namens, der mit übergeben wurde, die jeweilige Funktion mit den Input Daten aufgerufen.

Die aufgerufene Funktion wird wie folgt aufgebaut:

- Lesen der DataObjects für Input und Output
- Verbindung zur Datenbank herstellen
- Simulation der Metriken
- Simulation der Output-Parameter
- Ergebnisse in Datenbank schreiben

Die ersten drei Punkte können mit Hilfe der in Kapitel 5.6.3 vorgestellten Hilfsfunktionen implementiert werden. Für die zwei letzteren Punkte bedarf es einer konkreteren Implementierung.

5.6.5 Simulation der Output-Parameter

In der Analyse Phase wurden für jeden Output-Wert Modelle aufgestellt, die nun implementiert werden müssen. Dabei wird auch auf die operative DB2 Datenbank zugegriffen. Auf Basis der Modelle wird darauf die Java-Implementierung der Simulation durchgeführt. Ein genaues Beispiel wird in Kapitel 6 vorgestellt.

In Kapitel 4.2 haben wir in Tabelle 4.1 gesehen, dass unterschiedliche Möglichkeiten der Simulation vorherrschen. Dabei muss nicht jeder Output anhand eines Data Mining Modells implementiert werden. Die Simulation von Output-Parametern für konstante Werte und „Output-ist-gleich-Input" Werten ist trivial. Für die Implementierung der IDs, die keine signifikante Aussage machen, aber trotzdem zumindest „DISTINCT" sein müssen, behilft man sich der erstellten Datenbank-Tabelle, die die bisherigen IDs gespeichert hat. Durch eine einfache Abfrage, wie Listing 5.3, erhält man die letzte ID. Durch eine Erhöhung dieser ID um eins, hat man bereits die Simulation dieses neuen Werts erstellt.

```
SELECT MAX(ID) FROM SIM_ACTIVITYNAME
```

Listing 5.3: DB2 SQL für Abfrage der letzten ID

Die Analyse der weiteren Outputs ist dann schon komplexer. Als erstes muss jeder Data Mining Parameter, der in der Analyse zur Bestimmung

des Outputs genutzt wurde aus der Datenbank gelesen werden. Dazu gehören vor allem die operativen Daten, die nicht automatisch als Input-Parameter übergeben wurden. Dafür müssen dann DB2 SQL-Abfragen erstellt werden und in Hilfsvariablen gespeichert werden. Sind alle Parameter gegeben, können die eigentlichen Modelle implementiert werden.

Für die lineare Regression werden die Output-Parameter in einer einfachen Gleichung simuliert. Die Formel der linearen Regression in Java implementiert sieht man in Listing 5.4. Dabei wurden die zugehörigen Gewichte in der Datenanalyse analysiert. Ist der Output-Typ sogar ein ganzzahliger Typ, muss das Ergebnis noch umgewandelt werden.

```
output = weight0 + weight1 * parameter1 + … + weightx * parameterx
```

Listing 5.4: Implementierung der linearen Regression

Die Klassifikationsbäume werden mit Hilfe von geschachtelten if-Anweisungen implementiert. Dabei werden wiederum die erweiterten Input-Parameter aus der Datenbank benötigt. Die erweiterten Input-Parameter werden jetzt in den if-Anweisungen abgefragt. Wenn man das allgemeine Klassifikationsbaum Beispiel (Abbildung 2.8) aus dem Grundlangen Kapitel nimmt, so würde die Implementierung so aussehen:

```
if (X > 0) {
   if (Y > 1) {
      Output = "a";
   } else {
      Output = "b";
} else {
   if (Y > 3) {
      Output = "a";
   } else {
      Output = "b"; // einfache Variante
}
```

Listing 5.5: Code-Beispiel für die Implementierung eines Klassifikationsbaums

Wie im Baum dargestellt, ist der letzte Blattknoten 19-mal b und einmal a. Die einfache Variante ist den Wert, der öfter vorkommt, zu übergeben. Eine weitere Möglichkeit ist eine Wahrscheinlichkeit für den Output noch einzubauen. Dabei wird für jeden, nicht ganz klassifizierten Blattknoten eine Wahrscheinlichkeit aus dem Data Mining Modell für die möglichen

Outputs herausgelesen und implementiert. In Listing 5.5 wäre dies beim letzten Blattknoten eben zu 95% ein b und ansonsten ein a.

5.6.6 Speichern der Ergebnisse

Sind die Output Ergebnisse für die Input Variablen berechnet, werden sie, bevor die Outputs zurückgegeben werden, mit den Input Werten in die DB2 Datenbank geschrieben, um dann später analysiert werden zu können. Dabei werden die DB2 Tabellen genutzt, die vorher in der Pre-Settings-Phase erstellt wurden. Dabei muss auch auf die Datentypen geachtet werden, die bei der Erstellung der Datenbank gewählt wurden. Ein genaueres Beispiel folgt in Kapitel 6.

```
INSERT INTO SIM_ACTIVITYNAME (…)
```

Listing 5.6: DB2 SQL-Anweisung zur Speicherung der Ergebnisse

Um die Ergebnisse richtig in die Datenbank zu speichern, muss ein Mapping der Datentypen von Java auf DB2 geschehen. Dies muss auch schon in der Erstellung der Simulations-Tabellen beherzigt werden. Tabelle 5.1 zeigt das Mapping der möglichen Input-Parameter.

Java	DB2
int	INT
double	FLOAT
boolean	VARCHAR(5)
string	VARCHAR(255)

Tabelle 5.1: Mapping der Datentypen für die Simulation

Das interessanteste am Mapping ist die Wahl des boolean-Datentyps bei DB2 mit VARCHAR(5). Um die Analyse später einfach zu machen, wurde VARCHAR(5) dem INT mit 0 und 1 als Werten vorgezogen, da dies im WEKA Panel als nominalen Wert erkannt wird. Somit wären die Analysen darauf einfacher.

6 Beispiel, Ergebnisse und Evaluation

Nachdem nun das Modell und die Implementierung des Simulations-Frameworks vorgestellt wurden, soll in diesem Kapitel gezeigt werden, wie realitätsnah eine Simulation auf Grundlage des Simulationsmodells für einen dBOP-Prozess abschneidet. Außerdem wird in diesem konkreten Beispiel gezeigt, wie man aus dem dBOP Prozess zum Simulationsmodell gelangt.

Natürlich ist das Ergebnis eines einzigen Prozesses kein sehr guter Beweis für die Verbesserung auf Grund von konsolidierten Daten. Es zeigt aber schon einmal, dass dies für einen Prozess Bestand hat und es womöglich in die richtige Richtung geht. Wir werden aber sehen, dass allein auf Grundlage von mehr Informationen der Daten eine Klassifizierung von numerischen Daten in nominale Klassen einen erheblichen Vorteil bringt.

6.1 Planung und Analyse des Beispiels

Der Beispielprozess wurde schon in Kapitel 3.3 vorgestellt. Dabei wurden die einzelnen dBOP Aktivitäten vorgestellt. Wir wissen, dass wir später die Aktivitäten durch den Simulations Service ersetzen werden. Deshalb müssen in der Analyse Phase die Information für die spätere Konstruktion des Simulation Service gesammelt werden.

Das Simulationsmodell wurde auf Basis von 100 Prozessinstanzen herausgearbeitet. Dabei wurde jede Aktivität einzeln im BPAClient überprüft, sodass neben der Execution und Waiting Time, die durch die Metriken-Berechnung schon gegeben sind, auch die Outputdaten simuliert werden können. Für jeden Output wurde ein Simulationsmodell schon herausgearbeitet, das später bei der Konstruktion implementiert werden kann.

Nun werden die entstandenen Simulationsmodelle vorgestellt, die für die Evaluation ausgearbeitet wurden. Dabei wurden zwei Beispiele herausgezogen, die speziell für diese Simulation interessant sind. Dabei wurden ein

numerischer und ein nominaler Output mit Hilfe der operativen Daten als Klasse simuliert. Die wahrscheinlich größte Stärke des Simulationsmodells.

6.1.1 Modell-Analyse von SelectCar

Als erstes werden mit Hilfe des BPAClient, die Output Daten analysiert. Dabei sollen die Output Daten CARID (für die ID des ausgesuchten Autos) und PRICE (Preis für das Auto) simuliert werden. Da beide Werte numerisch sind, werden die Daten erst einmal durch den Algorithmus für Lineare Regression analysiert. Für PRICE ist das einfach, da der Output Wert immer 0 ergibt (der Preis wird an späterer Stelle berechnet), und es kann die Simulation für konstante Werte genutzt werden. Für CARID ist das schon komplexer. Eine lineare Regression kommt da nicht zu einem realitätsnahen Ergebnis, da wohl eine gleichverteilte Wahrscheinlichkeitsverteilung, nicht die Wünsche von Kunden realitätsnah simulieren wird. Deshalb sollte innerhalb der Konsolidierung geschaut werden, ob es weitere operative Daten für diesen Output Wert gibt, und somit eine Klassifizierung dieser möglich ist. Bei CARID ist das möglich. Die CARIDs können mit Hilfe der CAR Tabelle aus den operativen Daten nach Typ (TYPE) klassifiziert werden. Abbildung 6.1 zeigt die Konsolidierung von SelectCar. OUT_RENTAL_CARID wird mit operativen Daten analysiert und kann somit erfolgreich klassifiziert werden.

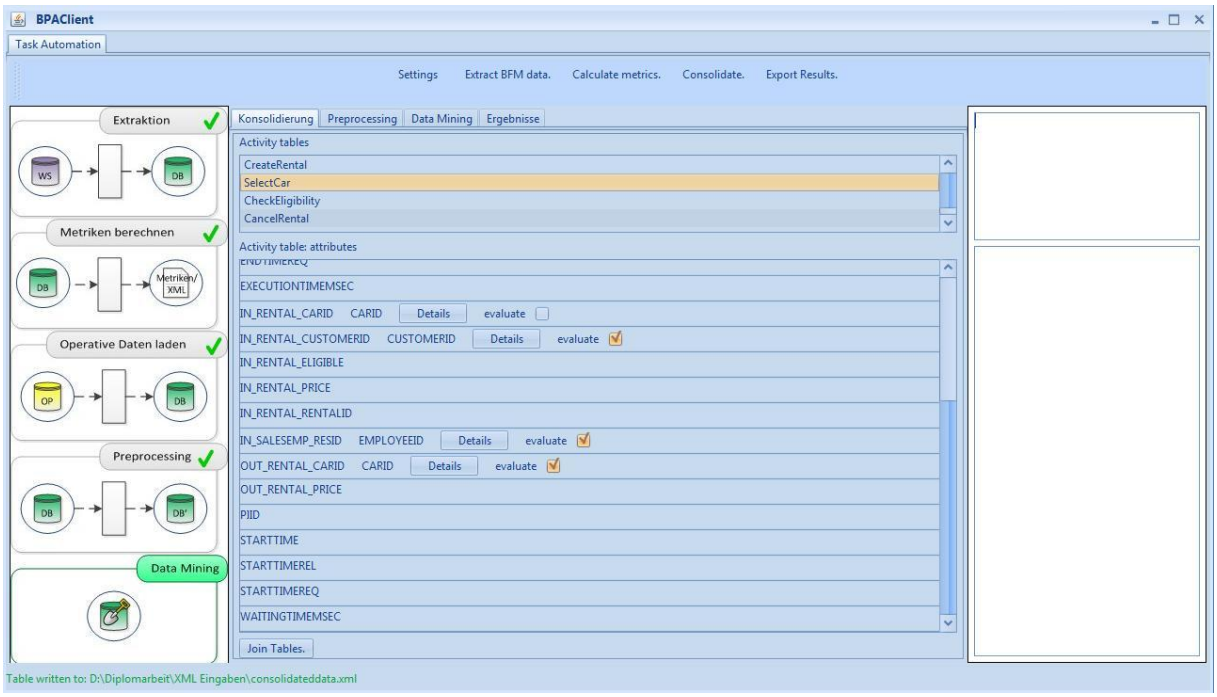

Abbildung 6.1: Screenshot des BPAClient bei der Analyse des Prozess für die Simulation

Außerdem werden auch die Input-Parameter vom Kunden (CUSTOMERID) und des Mitarbeiters (EMPLOYEEID) für die Analyse mit operativen Daten gejoint. Somit kann eine bessere Klassifizierung als ohne operative Daten geschehen.

In diesem konkreten Fall wäre eine Simulation ohne operative Daten zu ca. 25% genau (TYPE sind vier Klassen, bei eventueller Gleichverteilung der Autos). Mit den zusätzlichen operativen Daten der Input-Parameter sind die korrekt klassifizierten Autos bei 45% bei einer 80/20 Training- und Test-Menge. Abbildung 6.2 zeigt den Klassifikationsbaum.

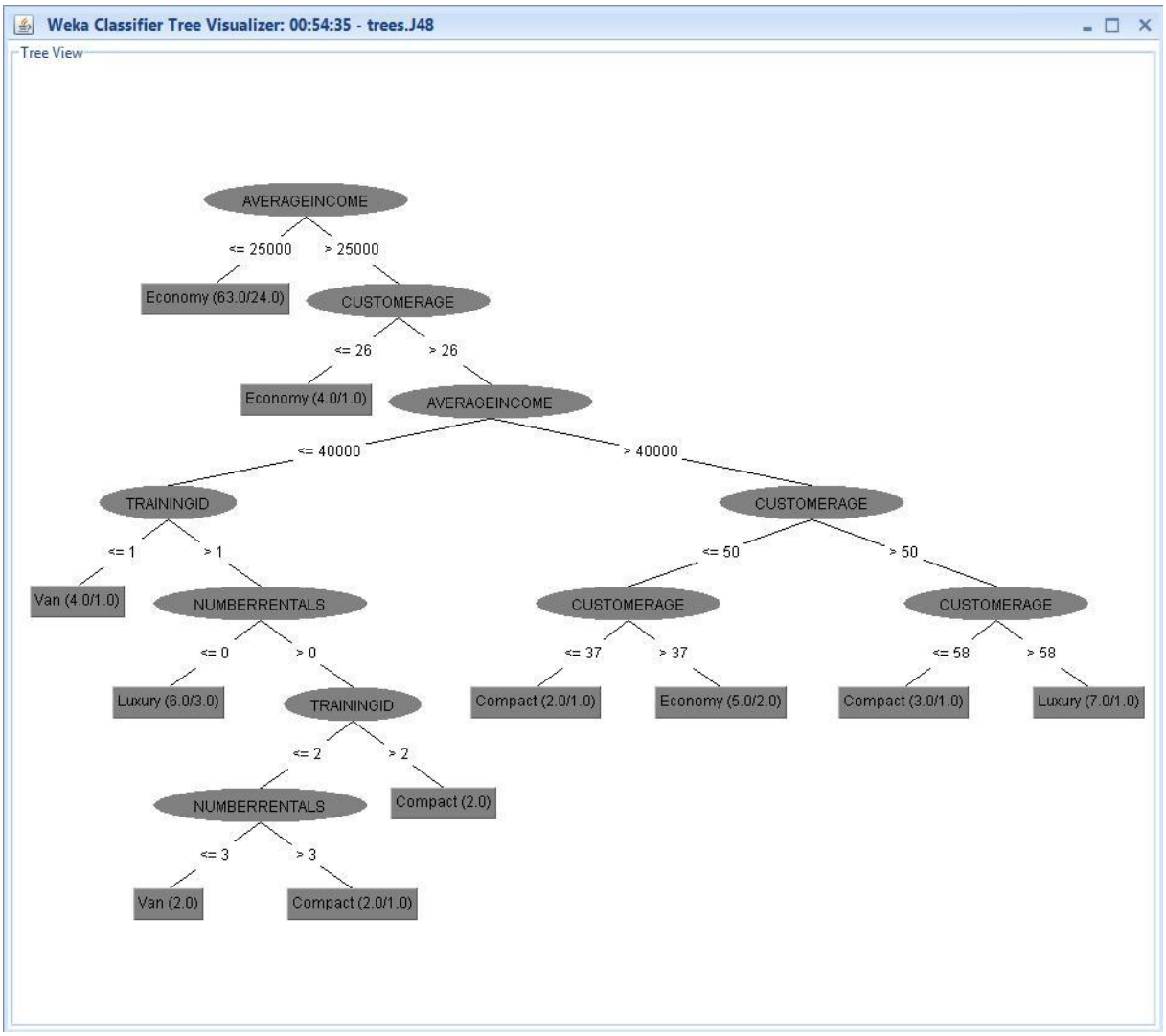

Abbildung 6.2: Klassifikationsbaum für Klasse Type in der SelectCar Aktivität

6.1.2 Modell-Analyse von CheckEligibility

Als weiteres Beispiel wird die Analyse und Modellerstellung von CheckEligibility erläutert. Dieses Beispiel ist ein klassisches Beispiel für die mögliche Ersetzung von „Human Tasks" in automatisierte Aktivitäten. Eine Simulation einer automatisierten Version könnte aufzeigen, dass die automatische Lösung ohne signifikante Nachteile zum selben Ergebnis kommt und dies selbstverständlich schneller ausführen kann. Abbildung 6.3 zeigt das Automatisierungsbeispiel aus dem dBOP Prozess. Die Simulation wird nicht in einen Pre-Check übergehen, wie es in der Abbildung dargestellt wird. Es wird die reale Aktivität durch den Simulation Service ersetzt. Ist es später einmal erwünscht, den Pre-Check zu simulieren, um eine Analyse der Verkürzung der Bearbeitungszeit durch den Pre-Check aufzuzeigen,

ist dies auch möglich. Dabei muss z.B. beachtet werden, dass Blattknoten, die nicht vollständig klassifiziert werden konnten, zur normalen CheckEligibility Aktivität geleitet werden.

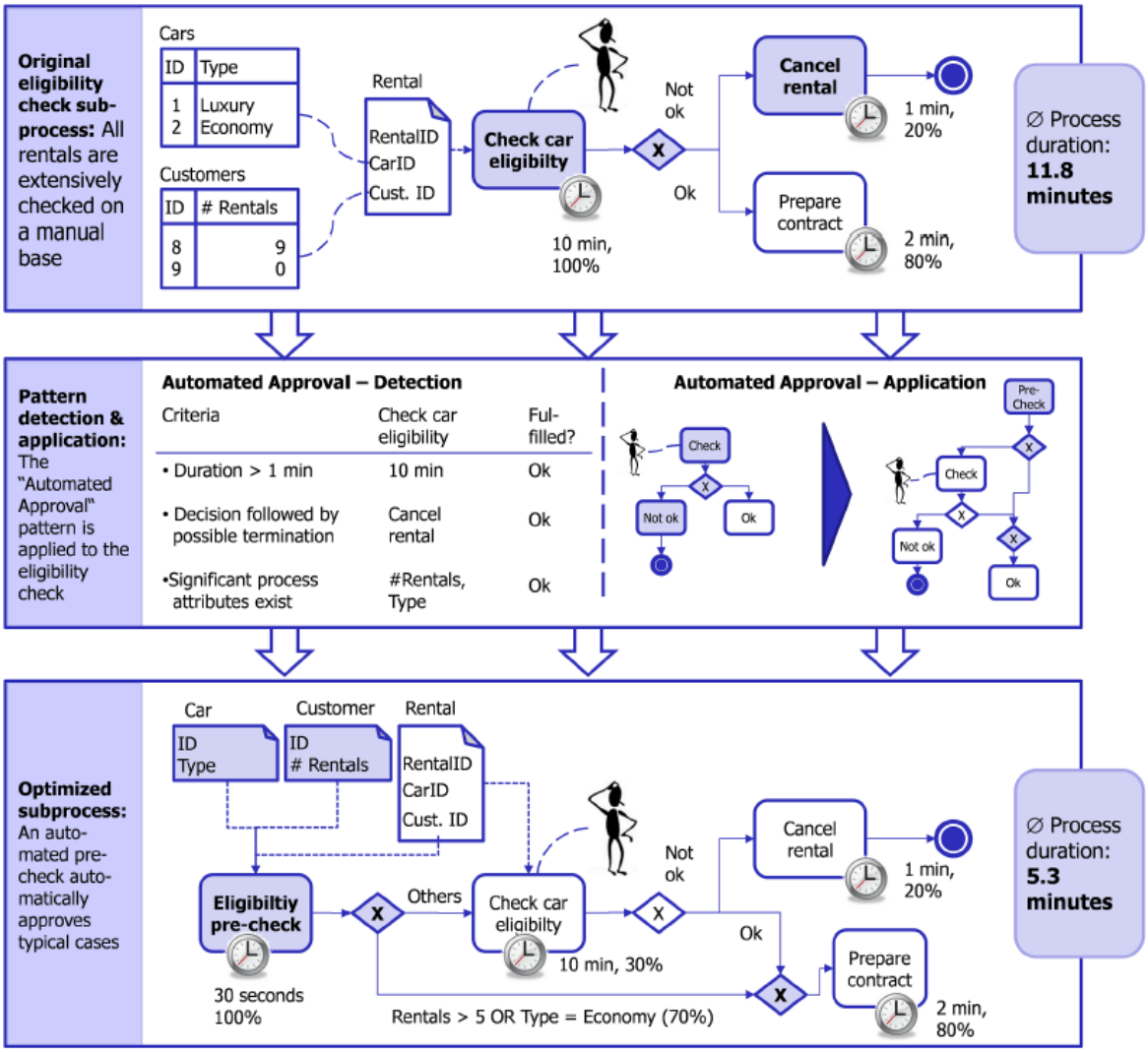

Abbildung 6.3: Beispiel des Automated Approval Pattern (3)

Der einzige Output-Parameter der CheckEligibility Aktivität ist die boolean-Variable eligible, die aussagt, ob der Kunde berechtigt ist das Automobil zu mieten. Da es sich um einen nominalen Wert handelt, wird wiederum ein Klassifikationsbaum aufgestellt. Abbildung 6.4 zeigt diesen aus dem BPAClient, der für die Analyse genutzt wurde.

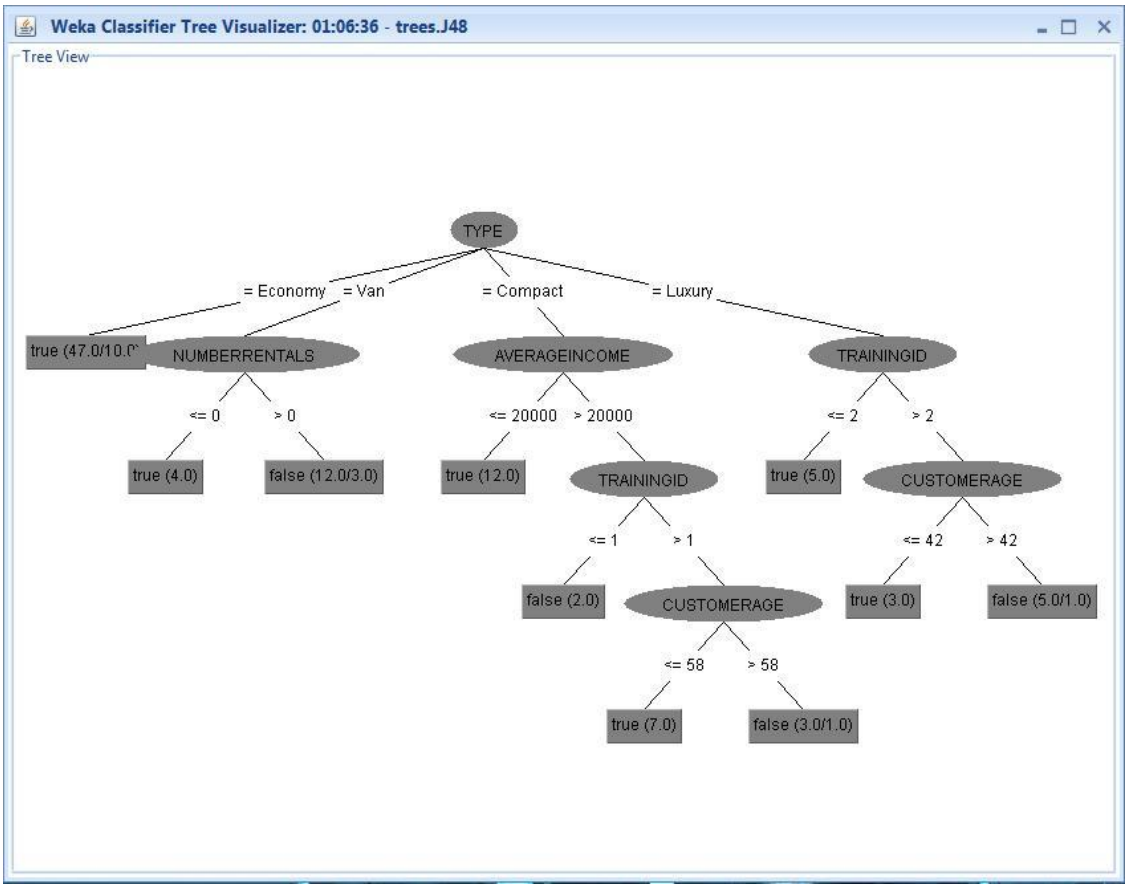

Abbildung 6.4: Data Mining Panel des BPAClient

6.2 Datendefintion und –erhebung des Beispiels

Nachdem nun diese und die weiteren Aktivitäten (CREATERENTAL, CANCELRENTAL, CREATE CONTRACT) analysiert wurden, kann man mit den Vorbereitungen für die Simulation beginnen. Als erstes muss mit Hilfe des dBOPSim Tools die Datenbank-Tabellen für die Simulation erstellt werden. Für die SelectCar Aktivität wurde dies schon in Kapitel 5.5.3 mit Listing 5.1 vorgestellt. Für CheckEligibility sieht das Beispiel so aus:

```
CREATE TABLE SIM_CHECKELIGIBILITY
(ID INT NOT NULL,
 EXECUTIONTIME INT,
 WAITINGTIME INT,
 IN_CARID INT,
 IN_CUSTOMERID INT,
 IN_ELIGIBLE VARCHAR(5),
 IN_PRICE DOUBLE,
 IN_RENTALID INT,
 IN_EMPLOYEEID INT,
 OUT_ELIGIBLE VARCHAR(5),
 PRIMARY KEY (ID));
```

Listing 6.1: Datenbank-Tabellen Erstellung für die Simulation

Man beachtet den Typ der OUT_ELIGIBLE Variable, der, wie in Kapitel 5 beschrieben, als VARCHAR(5) gespeichert wird.

Weiterhin muss die Verteilung der Input-Parameter für den Gesamtprozess bestimmt werden. Da der Prozess nur die customerID als Input-Parameter benötigt, ist die Auswahl recht einfach. Wenn man sich die Customer-Datenbank des Beispiels anschaut, sind dort 1000 Datensätze mit der ID 1 bis 1000 gespeichert. Da später die Simulationsergebnisse mit den realen Datensätzen verglichen werden soll, ist eine Gleichverteilung der customerID von 1 bis 1000 gewählt worden. Außerdem wurden die Prozessinstanzen mit keinem signifikanten zeitlichen Unterschied gestartet, da die dynamische Ressourcenverteilung nicht implementiert wurde und das Ziel dieser Analyse die realitätsnahe Simulation durch die konsolidierten Daten ist und nicht die Analyse der Bearbeitungszeit.

6.3 Simulationskonstruktion des Beispiels

Es wurden alle Vorbereitungen für die Simulation getroffen, nun muss die Simulation auch implementiert werden. Der zur Verfügung stehende Simulation Service wird für jede ersetzte Aktivität eine Funktion aufrufen, die nach dem gleichen Schema aufgebaut ist, wie in Kapitel 5.6.4 erläutert. Listing 6.2 zeigt das Beispiel der selectCar Simulation.

```
public DataObject selectCar(DataObject rental) {

   // Lesen der DataObjects für Input und Output
   DataObject outputR = getDataObject("selectCar", "rental", false);

   // Verbindung zur Datenbank herstellen
   Connection cn = DBConnect("Rental", "db2admin", "welcome1");
   Statement st;

   // Simulation der Metriken (Execution + Waiting Time)
   int execution = simulateDuration(413140,312935);
   int waiting = simulateDuration(60350,35314);

   // Simulation der Output-Parameter

      // Simulation for price = 0
      outputR.setFloat("price", 0.0F);

      // Simulation for carID
      int carID = 0;
      String type;
      int customerID = rental.getInt("customerID");
      int employeeID = rental.getInt("employeeID");
```

```
try {
 Statement stmt = cn.createStatement();
 // Get Information of operative Data
 ResultSet rs = stmt.executeQuery(
 "SELECT Occupation.AverageIncome, Customer.AGE AS CUSTOMERAGE,
 Customer.NUMBERRENTALS
 FROM CarRental.Occupation Occupation,
 CarRental.Customer Customer
 WHERE Occupation.OCCUPATIONID = Customer.OCCUPATIONID
 AND Customer.CustomerID = " + customerID);
 rs.next();
 int averageincome = rs.getInt(1);
 int customerage = rs.getInt(2);
 int numberrentals = rs.getInt(3);

 rs = stmt.executeQuery(
 "SELECT TRAININGID FROM CarRental.EmployeeTRAININGS
 WHERE employeeID = " + employeeID);
 rs.next();
 int trainingid = rs.getInt(1);

 // get Type with Classification Tree
 if (averageincome <= 25000) {
    type = "Economy";
 } else {
    if (customerage <= 26) {
        type = "Economy";
    } else {
        if (averageincome <= 40000) {
           if (trainingid <= 1) {
              type = "Van";
           } else {
              if (numberrentals <= 0) {
                 type = "Luxury";
              } else {
                 if (trainingid <= 2) {
                    if (numberrentals <= 3) {
                       type = "Van";
                    } else {
                       type = "Compact";
                    }
                 } else {
                    type = "Compact";
                 }
              }
           }
        } else {
           if (customerage <= 50) {
              if (customerage <= 37) {
                 type = "Compact";
              } else {
                 type = "Economy";
              }
           } else {
              if (customerage <= 58) {
                 type = "Compact";
              } else {
                 type = "Luxury";
              }
           }
        }
    }
 }

 // Type is chosen, now get a CarID of this type
 rs = stmt.executeQuery(
 "SELECT CarID, RAND() as IDX
 FROM CarRental.Car
```

```
            WHERE Type = '" + type +"'
            ORDER BY IDX FETCH FIRST 1 ROWS ONLY");
            rs.next();
            carID = rs.getInt(1);


            // WRITE RESULTS INTO DB2 DATABASE

            stmt.executeUpdate(
            "INSERT INTO SIM_SELECTCAR
            VALUES (" + rental.getInt("rentalID") + ", "
                  + execution + ", "
                  + waiting + ", "
                  + 0 + ", "
                  + rental.getInt("customerID") + ", "
                  + "'false'" + ", "
                  + "0.0" + ", "
                  + rental.getInt("rentalID") + ", "
                  + carID + ", "
                  + "0" + ")");
            stmt.close();
            cn.close();

            } catch (Exception e) {
                  e.printStackTrace();
            }

      // Simulation Ende der carID
      outputR.setInt("carID", carID);

      // Output-Parameter zurückgeben
      return outputR;
```

Listing 6.2: Simulation Funktion der selectCar Aktivität

Das Lesen der DataObjects, die Verbindung zur Datenbank und die Simulation der Metriken können mit den bereitgestellten Hilfsfunktion einfach implementiert werden. Die Simulation der Output-Parameter und das Schreiben der Ergebnisse in die Datenbank müssen bisher manuell geschehen.

In Listing 6.2 wird der größte Vorteil der Simulation auf den konsolidierten Daten vorgestellt. Die Simulation von numerischen Werten auf erweiterte Klassen durch das Matching auf operative Daten. Es werden zuerst alle relevanten Daten aus der Datenbank ausgelesen, die für die Simulation gebraucht werden. Anschließend wurde der Klassifikationsbaum für die Klasse in Form von if-Anweisungen implementiert, um die Klasse zu erhalten. Nun kann ein Auto per Zufall ausgewählt werden, das der Zielklasse entspricht.

6.4 Simulationsausführung des Beispiels

Es wurden nun 500 Prozessinstanzen ausgeführt, die später mit den realen Daten verglichen werden können. Eine weitere Erklärung bedarf es in diesem Abschnitt nicht.

6.5 Ergebnisse des Beispiels

Die interessanten Ergebnisse in diesem Beispiel zur Bestätigung des Simulationsmodells sind zwei Output-Parameter. Das erste ist die CarID bei der SelectCar Aktivität. Das zweite ist der boolean Wert der eligible Variable der CheckEligibility Aktivität. Beide Variablen erhalten eine realitätsnähere Simulation durch die Bereitstellung der operativen Daten. Erstmals können die Ergebnisse im dBOPSim betrachtet werden.

6.5.1 CarID (SelectCar)

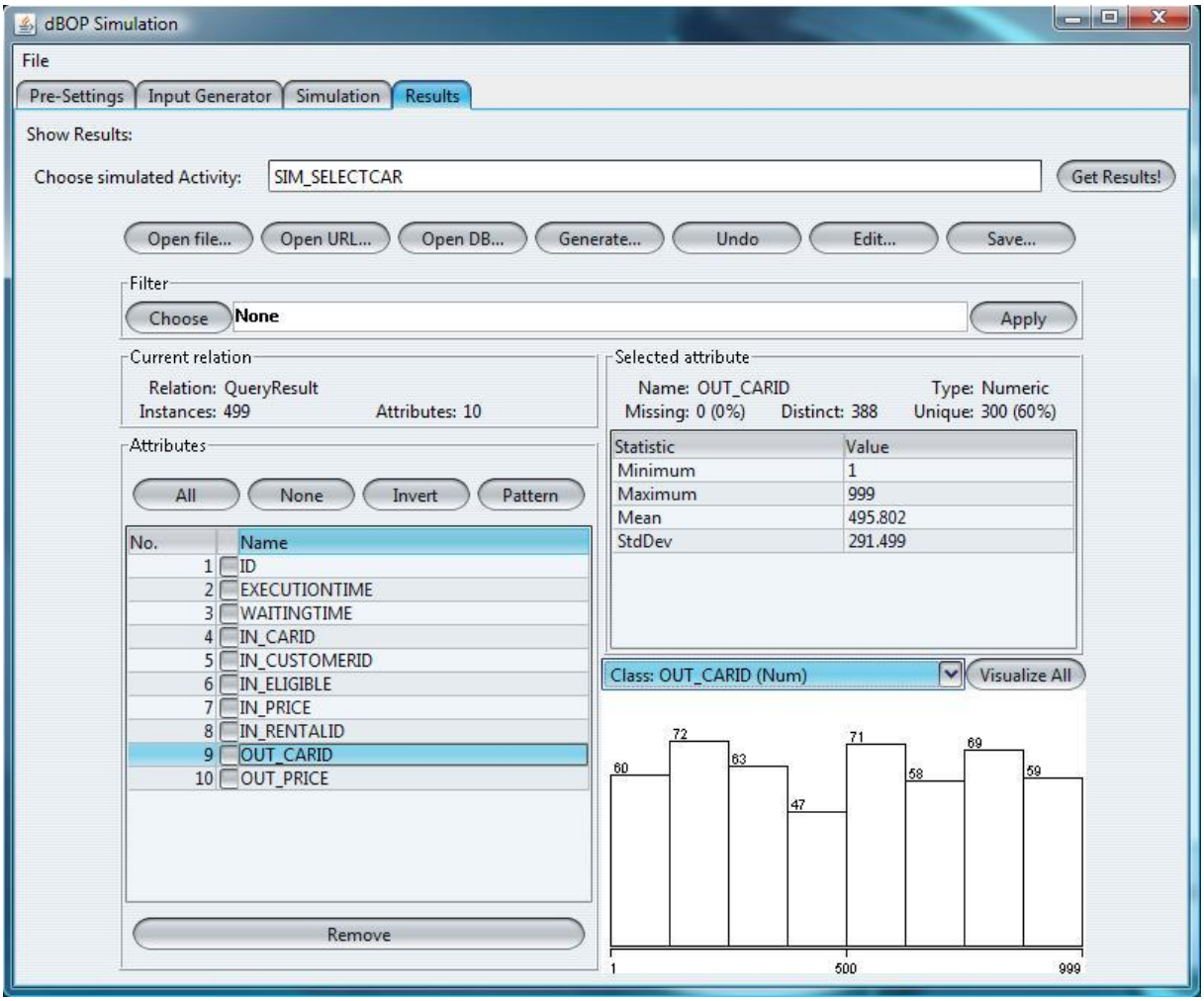

Abbildung 6.5: Ergebnis-Abfrage der Simulation für die SelectCar Aktivität

In Abbildung 6.5 kann man die Ergebnisse für die SelectCar Aktivität entnehmen. Die Ergebnisse für die carID allein sagen erst einmal nichts für die Realitätsnähe aus. Das kann dann später extrahiert werden und mit den operativen Daten mit verglichen werden. Was man aber schon sehen kann, ist das 499 Instanzen importiert worden sind und das diese ganz gut verteilt sind. Weitere Analysen folgen für dieses Beispiel noch.

6.5.2 Eligible (CheckEligibility)

Die Simulation der Eligible Variable, die dann später auch entscheidet welchen Zweig der Prozess weiter ausführt, ist auch ein Beispiel für die Optimierung eines Prozess durch die Automatisierung von manuellen Aktivitäten. Deshalb ist es wichtig, dass diese Simulation so genau wie mög-

lich ausgeführt wird. Abbildung 6.6 zeigt die Ergebnisse im Ergebnis-Panel.

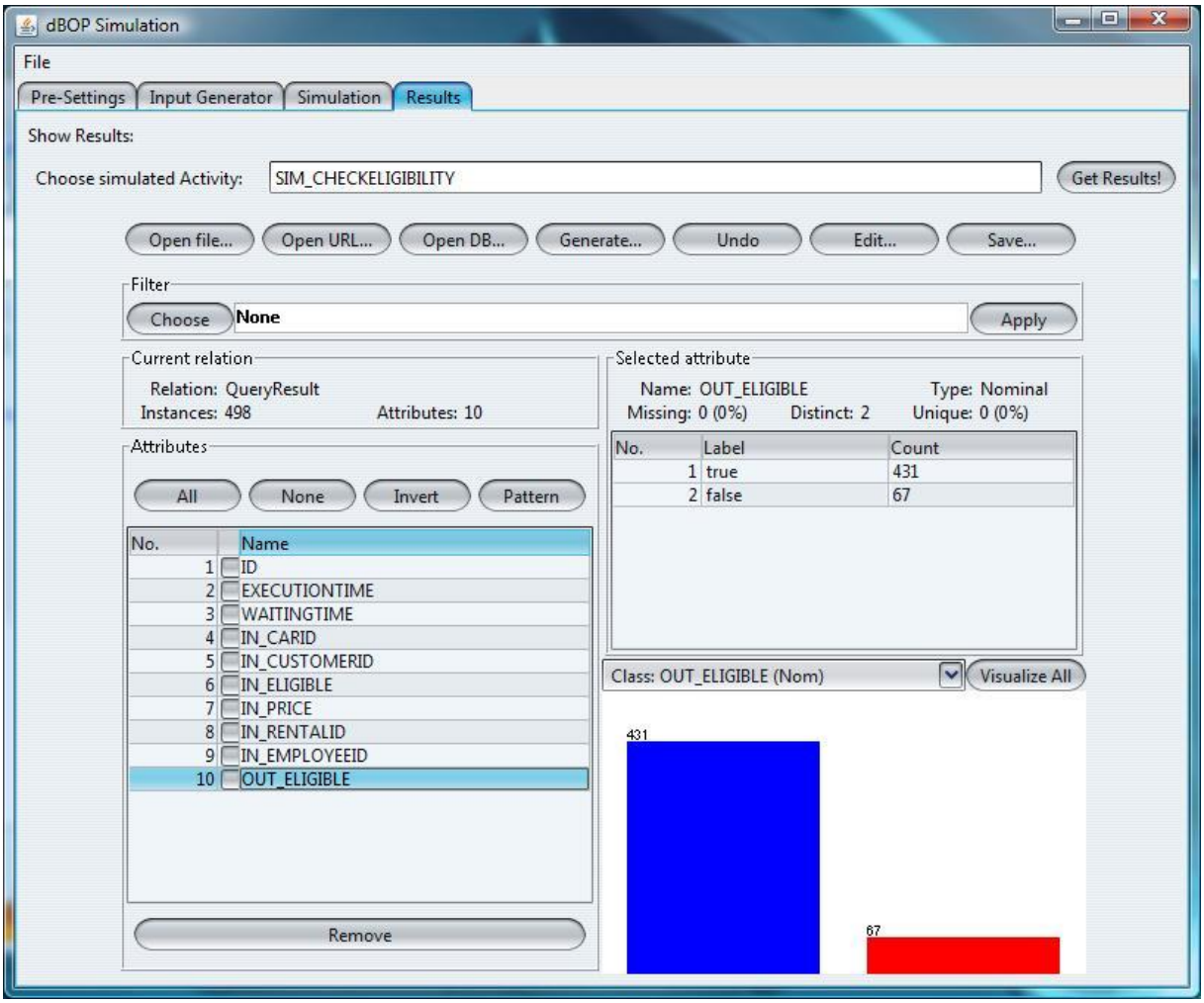

Abbildung 6.6: Ergebnis-Abfrage der Simulation für die CheckEligibility Aktivität

Von den 498 Prozessinstanzen, die extrahiert wurden, wurden 431 als „eligible" gewertet und 67 haben keine Freigabe für die Automietung bekommen. Das entspricht einer Erfolgsrate von 86,5%.

6.6 Vergleich der Simulation

Nun wird gezeigt, dass eine Geschäftsprozesssimulation auf Basis der deep Business Optimization Platform effizienter ist als ohne. Dazu werden nun Ergebnisse der Simulationen mit den realen Daten überprüft. Zuerst werden die simulierten Daten verglichen. Anschließend geschieht noch ein Vergleich zu einer möglichen Simulation ohne konsolidierte Daten.

Wie über das ganze Kapitel schon vorgestellt, sind die interessanten Simulationen die carID und eligible Variablen. Durch die 500 Instanzen, die ausgeführt wurden, wird jetzt eine SQL-Join Abfrage durchgeführt, die die simulierten Daten mit den realen Daten vergleicht.

6.6.1 Vergleich von carID (SelectCar)

Abbildung 6.7 (a) zeigt den Vergleich für die carID. Aus den 500 Prozessinstanzen konnten 45 reale Prozesse nachgeschlagen werden. Dabei wurden 14 von 15 Economy Autos richtig simuliert. Das entspricht einer „Success-Rate" von 93,3%. Für die restlichen Typ-Klassen sind das: Luxury (7/9 = 77,9%), Compact (3/12 = 25%) und Van (1/9 = 11,1%). Das entspricht einer Gesamt-Erfolgsrate von über 50%. Die relativ schlechten Ergebnisse für Compact- und Van-Typen lassen sich durch die kleine Menge an Analyse Daten und die einfache Implementierung der nicht voll klassifizierten Blattknoten erklären. Trotz der geringen Datenmenge für den Aufbau der Simulation und der nicht voll klassifizierten Blattknoten erreicht diese Simulation ein besseres Ergebnis als eine zufällige Simulation, wie man Abbildung 6.7 (b) entnehmen kann.

CUSTOMERID	REAL_TYPE	SIM_TYPE
742	Compact	Economy
968	Compact	Compact
361	Compact	Economy
884	Compact	Economy
349	Compact	Compact
459	Compact	Economy
869	Compact	Economy
968	Compact	Compact
213	Compact	Economy
280	Compact	Economy
415	Compact	Economy
415	Compact	Economy
123	Economy	Economy
854	Economy	Economy
396	Economy	Economy
396	Economy	Economy
324	Economy	Economy
101	Economy	Economy
150	Economy	Economy
260	Economy	Economy
436	Economy	Economy
10	Economy	Economy
200	Economy	Economy
591	Economy	Luxury
260	Economy	Economy
895	Economy	Economy
108	Economy	Economy
349	Luxury	Compact
969	Luxury	Luxury
556	Luxury	Luxury
200	Luxury	Economy
419	Luxury	Luxury
722	Luxury	Luxury
722	Luxury	Luxury
419	Luxury	Luxury
896	Luxury	Luxury
164	Van	Economy
386	Van	Compact
30	Van	Economy
21	Van	Luxury
894	Van	Economy
527	Van	Economy
386	Van	Van
304	Van	Economy
795	Van	Economy

(a)

SIM2_TYPE
Compact
Van
Economy
Economy
Luxury
Economy
Economy
Economy
Van
Van
Economy
Economy
Compact
Luxury
Compact
Economy
Compact
Economy
Luxury
Economy
Luxury
Luxury
Economy
Economy
Economy
Economy
Luxury
Compact
Economy
Economy
Economy
Compact
Van
Economy
Economy
Luxury
Compact
Economy
Economy
Economy
Economy
Economy
Luxury
Economy
Economy
Luxury

(b)

Abbildung 6.7: Vergleich der realen und simulierten Auto-Typen

Simuliert man nun die Daten ohne Informationen aus den operativen Daten, kann eine Simulation nur auf zufälligen Werten geschehen. Eine carID, die simuliert werden würde, würde in der Simulation nur eine Zahl bedeuten. Die Simulation wäre dann eine Gleichverteilung mit den Min-

und Max-Werten der Simulation für diese Aktivität. Abbildung 6.8 zeigt die Verteilung der Auto-Typen.

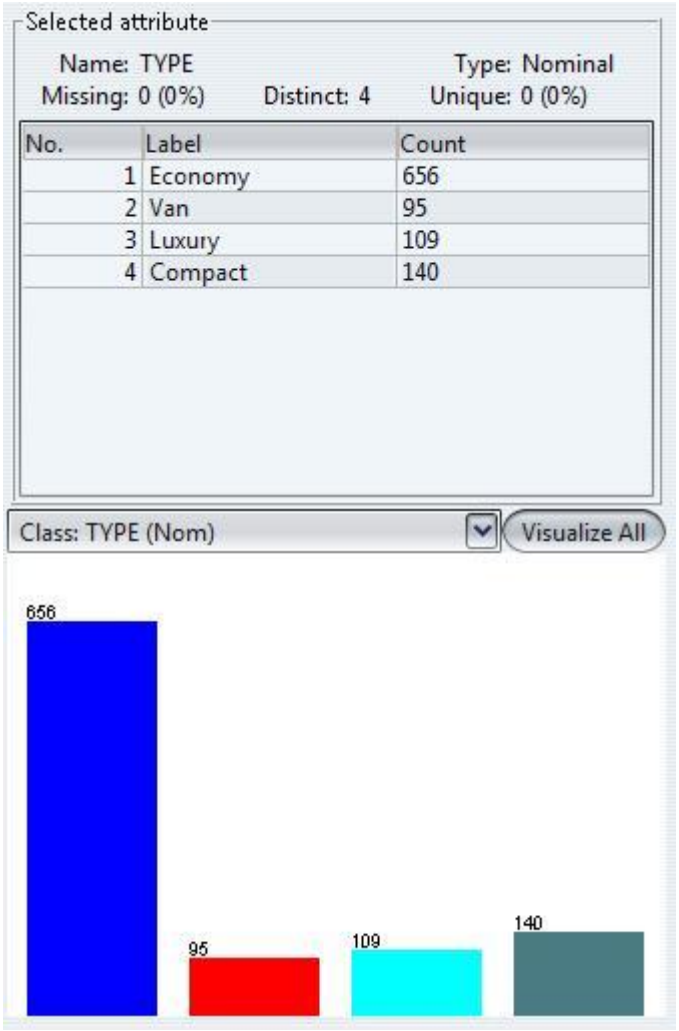

Abbildung 6.8: Verteilung der Auto-Typen

Das könnte aber zu einer realitätsfremden Simulation führen, wenn z.B. ein Kunde ein Luxus Auto auswählen würde, der das aus Geldgründen nicht machen würde. Abbildung 6.7 (b) zeigt wie so eine zufällige Verteilung aussehen könnte. In diesem Beispiel ist die Erfolgsrate nur bei 20% (9 von 45).

6.6.2 Vergleich von eligible (CheckEligibility)

Bei der CheckEligibility Simulation sieht es ähnlich aus. Abbildung 6.9 zeigt den Vergleich der realen und simulierten Daten.

CUSTOMERID	OUT_RENTAL_ELIGIBLE	OUT_ELIGIBLE
896	false	true
556	false	false
396	false	true
228	false	true
41	false	false
516	true	true
347	true	true
347	true	true
123	true	true
439	true	true
222	true	true
532	true	true
347	true	true
347	true	true
413	true	true
91	true	true
815	true	true
349	true	true
181	true	true
532	true	true
179	true	true
418	true	true
686	true	true
686	true	true

Abbildung 6.9: Ausschnitt des Vergleichs der eligible Variable

Wie die Abbildung zeigt, wurden reale „false" Werte fehl-simuliert und in der Simulation wurde „true" zurückgegeben. Trotzdem ist die „Success-Rate" bei diesem kleinen Beispiel bei 40%. Dafür ist die Erfolgsrate für „true" bei 100%. Eine zufällige Simulation ohne erweiterte Datenbasis könnte so etwas ohne eine Data Mining Analyse nicht realitätsnah simulieren.

Die einfachste Simulation mit geringer Datenbasis wäre die Simulation nach Verteilung, wie in Kapitel 2.3.2.4 Ablaufalternativen vorgestellt. Aus der realen Datenbasis geht ein „true" Wert von 73% hervor. Diese Simulation würde aber willkürlich zu den realen Daten ablaufen und würde im Durchschnitt eine zu hohe Fehlerrate für mögliche Autovermietungen zurückgeben.

Wie in (16) vorgestellt, sollten nur die voll klassifizierten Blattknoten automatisiert werden und Ergebnisse, die nicht vollständig klassifiziert wurden wiederum zu seiner manuellen Bearbeitung führen. Außerdem sollten

genügend Datensätze für die Bestimmung der Automatisierung zur Verfügung stehen.

6.7 Fazit

Wie wir sehen konnten, ist der Informationsgewinn durch das Matching gegeben. Weil auch der Output von Audit Daten abhängig ist von den operativen Daten war dies zu erwarten. Vor allem von Aktivitäten, in denen der Mensch den Aktivitätsausgang beeinflusst, werden mehr Information gebraucht, um sein Verhalten nachvollziehen und anschließend simulieren zu können. Abbildung 6.10 zeigt Faktoren für die Entscheidungsfindung bei „Human-Tasks".

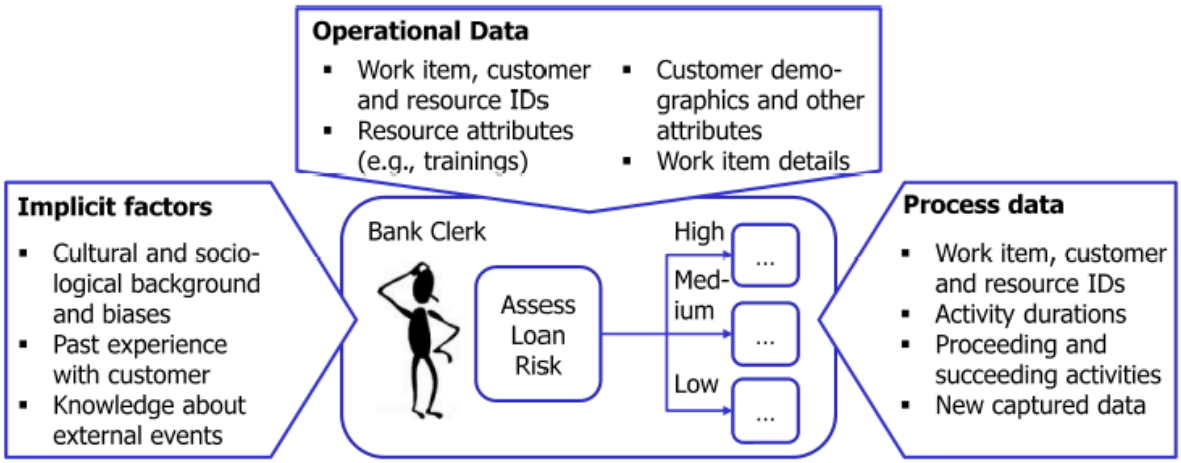

Abbildung 6.10: Faktoren für die Entscheidungsfindung (16)

Es muss auch darauf geachtet werden, dass nicht nur einzelne Aktivitäten möglichst realitätsnah simuliert werden, sondern alle Aktivitäten im Prozess. Eine Fehl-Simulation der Output-Parameter von vorherigen Simulationen kann wiederum zu falschen Ergebnissen der nachfolgenden Simulation führen. Wenn, wie im Beispiel, die Simulation der carID nicht nach Typen geschehen wäre, wäre die CheckEligibility Aktivität sehr realitätsfern simuliert, da die Input-Daten nicht mehr richtig klassifiziert wurden.

7 Zusammenfassung und Ausblick

Somit kommen wir zum Abschluss dieser Diplomarbeit. Als erstes wird nochmals eine kleine Zusammenfassung des Inhalts für den schnellen Überblick gegeben. Anschließend gebe ich einen kleinen Ausblick, was über diese Diplomarbeit hinaus, noch möglich und zu erwarten ist und vielleicht eine Untersuchung wert ist.

7.1 Zusammenfassung

In dieser Diplomarbeit wurden die Grundlagen von Simulation, Geschäftsprozessmanagement, Data Mining und Web Services erläutert. Mit diesen Grundkenntnissen wurde ein Framework entwickelt, das eine Geschäftsprozesssimulation ausüben soll, die auf Basis von Data Mining Algorithmen geschieht. Durch die bereits geleisteten Vorarbeiten der Business Impact Analysis (22) und der deep Business Optimization Platform (3) konnte eine Simulation auf einer erweiterten Datenbasis stattfinden, die eine sehr realitätsnahe Simulation überhaupt erst möglich macht. Außerdem wurde mit dem Process Mining Projekt (24) eine interessante ähnliche Arbeit vorgestellt, die sich schon mit dem Thema der Geschäftsprozesssimulation mit Hilfe von Data Mining Algorithmen auseinandergesetzt hat.

Darauf basierend wurde ein Konzept entwickelt, dass aus dem vordefinierten dBOP Modell ein Simulationsmodell aufbaut. Die vier Phasen waren die Planung und Analyse, die Datendefinition und –erhebung, die Simulationskonstruktion und –ausführung und als letztes die Bereitstellung der Ergebnisse. Anschließend wurde noch vorgestellt, wie dieses Konzept implementiert wurde. Mit Hilfe von vorherigen Arbeiten, wie dem BPAClient, und für wissenschaftliche Arbeiten bereitgestellte Software, wie der genutzten IBM Software und dem WEKA Tool, wurde dies realisiert. Dabei wurde nicht nur ein Simulation Service eingerichtet, der die einzelnen Ak-

tivitäten eines Prozess simulieren konnte, sondern auch ein Tool namens dBOPSim entwickelt, der die Prozesssimulation begleitet.

Der letzte Abschnitt ist die Evaluation der gewählten Geschäftsprozesssimulation auf Basis eines Beispiels. Es wird anhand eines Testprozesses aufgezeigt, wie das Simulationsmodell entsteht und welche Besonderheiten man beachten muss. Anschließend wurden die Ergebnisse der Simulation analysiert und mit realen Daten und Simulationsdaten ohne erweiterte Datenbasis verglichen.

7.2 Ausblick

Die Geschäftsprozesssimulation, die in dieser Diplomarbeit herausgearbeitet wurde, ist ein erster Prototyp zur Simulation auf Basis von erweiterten Informationen. Die am Ende bereitgestellten Ergebnisse könnten z.B. einer weiteren Analyse durchzogen werden, um etwaige Schwachstellen im Prozess zu erkennen. Gezieltere Input-Parameter Auswahl und dadurch andere Ergebnisse könnten noch miteinander verglichen werden.

Außerdem wurde in dieser Diplomarbeit kaum Wert auf die Ressourcenauswahl gelegt. Die Simulation für die Bereitstellung von Ressourcen könnte noch weitere Informationen hinsichtlich der Wartezeit von Aktivitäten ergeben. So könnte ein hohes Aufkommen an Prozessinstanzen zu einem Ressourcen-Engpass führen und so die Wartezeit enorm beeinflussen. Dies würde einer alternativen Simulation der „Waiting Time" entsprechen.

Die Simulationskonstruktion geschieht auch noch vorerst manuell, d.h. der Service könnte noch so erweitert werden, dass die gewünschte Simulation auf Basis der extrahierten Data Mining Ergebnisse aus dem Process Insight Repository automatisch und nicht mehr manuell implementiert wird. Dazu könnte die Data Mining Modell als Parameter übergeben werden anstatt direkt in den Simulation Service durch Regeln und if-Anweisungen implementiert zu werden.

Literaturverzeichnis

1. **Abts, Dietmar und Mülder, Wilhelm.** *Grundkurs Wirtschaftsinformatik: Eine kompakte und praxisorientierte Einführung.* Wiesbaden : Vieweg+Teubner, 2008.

2. **Hammer, Michael und Champy, James.** *Reengineering the Corporation: A Manifesto for Business Revolution.* New York : Harper Business Essentials, 2003.

3. **Niedermann, Florian, Radeschütz, Sylvia und Mitschang, Bernhard.** Deep Business Optimization: A Platform for Automated Process Optimization. *Business Process and Services Computing.* 2010.

4. **Becker, Jörg, Mathas, Christoph und Winkelmann, Alex.** *Geschäftsprozessmanagement.* Berlin, Heidelberg : Springer-Verlag, 2009.

5. **OMG.org.** Business Process Model and Notation 2.0. [Online] OMG.org. http://www.omg.org/spec/BPMN/2.0/.

6. **Leymann, Frank und Roller, Dieter.** *Production Workflow: Concepts and Techniques.* Englewood Cliffs : Prentice Hall, 1999.

7. **BPM-Offensive Berlin.** BPMN 2.0 - Business Process Model and Notation. *Poster.* [Online] BPM-Offensive Berlin. http://www.bpmb.de/images/BPMN2_0_Poster_DE.pdf.

8. **Workflow Management Coalition.** *Process Definition Interface - XML Process Definition Language.* s.l. : Workflow Management Coalition, 2008.

9. **OMG.org.** Business Process Management Initiative. *BPMN Graphical Elements.* [Online] OMG.org. http://www.omg.org/bpmn/Samples/Elements/Core_BPMN_Elements.htm .

10. **Jordan, Diane, et al., et al.** *Web Services Business Process Execution Language Version 2.0.* s.l. : OASIS Standard, 2007.

11. **Bibliographisches Institut GmbH.** Duden online. *Duden online.* [Online] Bibliographisches Institut GmbH. http://www.duden.de.

12. **McHaney, Roger.** *Computer simulation: a practical perspective.* San Diego, Kalifornien, USA : Academic Press, Inc., 1991.

13. **Becker, Jörg, Kugeler, Martin und Rosemann, Michael.** *Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung.* Berlin, Heidelberg [u.a.] : Springer, 2005.

14. **Witten, Ian H., Frank, Eibe und Hall, Mark A.** *Data Mining: Practical Machine Learning Tools and Techniques.* 2011.

15. **Quinlan, John Ross.** Induction of Decision Trees. *Machine Learning.* 1986, Bd. 1.

16. **Niedermann, Florian, et al., et al.** Automated Process Decision Making Based on Integrated Source Data. *Business Information Processing.* 2011, 87.

17. **Löwenstein, Bernhard und Kraeft, Oliver.** Einfach präzise - Entwickeln von Webservices mit JAX-WS und JAX-RS. *iX Magazin.* 2011, Bd. 8.

18. **W3C.** Web Services Description Language (WSDL) 1.1. [Online] http://www.w3.org/TR/wsdl.

19. —. SOAP Specifications. [Online] W3C. http://www.w3.org/TR/soap/.

20. **Java.net.** JAX-WS. [Online] http://jax-ws.java.net/.

21. **Radeschütz, Sylvia, Mitschang, Bernhard und Leymann, Frank.** Matching of Process Dataa and Operational Data for a Deep Business Analysis. *Enterprise Interoperability.* Part II, 2008.

22. **Radeschütz, Sylvia, Niedermann, Florian und Bischoff, Wolfgang.** BIAEditor: Matching Process and Operational Data for a Business Impact Analysis. *Proceedings of the 13th International Conference on Extending Database Technology.* 2010.

23. **Niedermann, Florian, Radeschütz, Sylvia und Mitschang, Bernhard.** Business Process Optimization using Formalized Patterns. *Business Information Processing.* 2011, 87.

24. **Rozinat, A., et al., et al.** *Discovering Simulation Models.* Eindhoven, Niederlande : s.n., 2009.

25. **Maier, Bernhard.** *Entwicklung eines Werkzeugs zur standardisierten Verarbeitung von Prozessdaten und operativen Daten.* Stuttgart : Universität Stuttgart, 2011.

26. **Hall, Mark, et al., et al.** The WEKA Data Mining Software: An Update. *SIGKDD Explorations.* 2009, Bd. 11, 1.

27. **Middendorf, Stefan, Singer, Reiner und Heid, Jörg.** *Java - Programmierhandbuch und Referenz für die Java-2-Plattform.* Heidelberg : dpunkt.Verlag, 2002.

Alle Links wurden zuletzt am 30. September 2011 verfolgt.

Erklärung

Hiermit versichere ich, diese Arbeit selbstständig verfasst und nur die angegebenen Quellen benutzt zu haben.

Nürtingen, den 4. Oktober 2011 _____

Michael Panos