Institut für Architektur von Anwendungssystemen

Universität Stuttgart
Universitätsstraße 38
D - 70569 Stuttgart

Studienarbeit Nr. 2217

Architektur eines Workflow-Frameworks zur graphischen Erstellung und Ausführung von Simulationsexperimenten

Karolina Vukojevic

Studiengang:	Informatik
Prüfer:	Prof. Dr. Dimka Karastoyanova
Betreuer:	Dipl. Inf. Katharina Görlach
begonnen am:	03.03.2009
beendet am:	02.09.2009
CR-Klassifikation:	H.4.1, H.5.2, K.1

Inhaltsverzeichnis

1 Einführung 4
2 Verwendete Technologien 5
2.1. Web Services 5
2.2. BPEL 5
2.3. DUNE 5
2.4. Eclipse 6
2.4.1. Eclipse Workbench 6
2.4.2. Extension Points und Extensions 7
2.5. EMF 8
2.6. BPEL Designer 9
3 Architektur des SimTech Workflow Tools 10
3.1. Konzept 10
3.2. Umsetzung in Eclipse 10
3.3. Implementierung 13
4 SimTech Service Catalog 15
4.1. Konzept 15
4.1.1. Ausgangsprozess 15
4.1.2. Aufbau einer SimTech Aktivität 16
4.1.3. SimTech Aktivitäten für DUNE 17
4.1.4. Prozess mit SimTech Aktivitäten 19
4.1.5. Integration in den BPEL Designer 20
4.2. Implementierung 21
4.2.1. Erweiterung des EMF Modells 22
4.2.2. Serialisierer und Deserialisierer 22
4.2.3. Grafische Darstellung, UIObjectFactory 23
4.2.4. Adapter 24
4.2.5. Erweiterung der Palette 24
4.2.6. Ausspezifizieren der SimTech Aktivitäten 25
4.2.7. Transformation in BPEL Prozesse ohne Extension Activities 26
4.2.8. Zusammenfassung aller Erweiterungen des BPEL Designers 26
5 Zusammenfassung und Ausblick 28
6 Referenzen 29

Abbildungsverzeichnis

Abbildung 1: Eclipse Workbench 7
Abbildung 2: Extension Point und Extension 8
Abbildung 3: EMF 8
Abbildung 4: MVC mit GEF 9
Abbildung 5: SimTech Modeling Perspective 11
Abbildung 6: SimTech Runtime Perspective 12
Abbildung 7: SimTech Analysis Perspective 13
Abbildung 8: Klassendiagramm - Perspektiven und Views 14
Abbildung 9: Simulationsprozess "Tinte in Wasser" 16
Abbildung 10: Mapping auf SimTech Aktivität "DUNESimulationStartUp 17
Abbildung 11: Mapping auf SimTech Aktivität "DUNEBuildGrid" 18
Abbildung 12: Mapping auf SimTech Aktivität "DUNESimulationRun" 18

Abbildung 13: Mapping auf SimTech Aktivität "DUNESimulationShutDown" 19
Abbildung 14: Prozess ohne (links) und mit SimTech Aktivitäten (rechts)..... 20
Abbildung 15: Palette mit Kategorie “SimTech Service Catalog” 21
Abbildung 16: SimTech spezifische Icons 21
Abbildung 17: EMF Ecore Modell für die SimTech Aktivitäten 22
Abbildung 18: Erweiterungen des BPEL Designers 27

Verzeichnis der Listings

Listing 1: Methode "createInstance()" 23
Listing 2: Erweiterung der BPEL Designer Palette 24
Listing 3: Ausspezifizieren der SimTech Aktivitäten im Serialisierer 25
Listing 4: Entfernen von Extension Activity 26

Abkürzungsverzeichnis

BPEL	WS-Business Process Execution Language
DOM	Document Object Model
DUNE	Distributed and Unified Numerics Environment
EMF	Eclipse Modeling Framework
GUI	Graphical User Interface
IT	Informationstechnologie
MVC	Model View Controller
WS	Web Service
WSDL	Web Service Description Language

1 Einführung

Als Workflow bezeichnet man üblicherweise Geschäftsprozesse, die in einer IT Umgebung ausgeführt werden [LR00]. Workflow Technologien sind aber nicht nur auf die Beschreibung und Ausführung von Geschäftsprozessen beschränkt. Ein weiteres Anwendungsgebiet findet sich in den Bereichen der Wissenschaft, in denen computergestützte Simulationen zur Anwendung kommen. Die Ausführung einer solchen Simulation kann in vielen Fällen als komplexer Prozess beschrieben werden. Die Anwendung von Workflow Technologien kann in wissenschaftlichen Anwendungsbereichen helfen, diese Komplexität besser zu beherrschen und die Ausführung von computergestützten Simulationen zu vereinfachen.
Im Rahmen des Exzellenzclusters Simulation Technology[1] (SimTech) der Universität Stuttgart wird ein Workflow Framework zum Modellieren und Ausführen von Simulations-Workflows entwickelt. Ziel dieser Arbeit ist es, eine Architektur für das Frontend des SimTech Workflow Frameworks zu entwickeln. Dem Benutzer soll dieses Frontend in Form eines Tools zur Verfügung gestellt werden, welches als *SimTech Workflow Tool* bezeichnet wird. Mit diesem Tool kann der Benutzer Simulationsexperimente modellieren, ausführen, überwachen und analysieren.
Das SimTech Workflow Tool soll unter anderem eine Komponente *Service Catalog* enthalten, welche dem Benutzer simulationsspezifische Aktivitäten zur Modellierung bereitstellt. Die Komponente soll prototypisch implementiert und in ein bereits bestehendes Workflow Modellierungstool, den *Eclipse BPEL Designer*[2], eingebunden werden.
In Kapitel 2 wird ein Überblick der in dieser Arbeit verwendeten Technologien gegeben. In Kapitel 3 werden der Entwurf und die Implementierung des SimTech Workflow Tools beschrieben. In Kapitel 4 wird eine Komponente dieses Tools, der SimTech Service Catalog, entworfen und prototypisch umgesetzt. Abschließend wird in Kapitel 5 die gesamte Arbeit noch einmal zusammengefasst.

[1] http://www.iaas.uni-stuttgart.de/forschung/projects/simtech/
[2] http://www.eclipse.org/bpel/

2 Verwendete Technologien

2.1. Web Services

Web Services sind Anwendungen, die über Netzwerk Protokolle benutzbar sind. Sie haben eine „always on“ Semantik, das heißt, sie sind immer erreichbar. Web Services basieren auf Web Standards, wie beispielsweise *HTTP*, *XML* oder *SOAP.* Die Implementierung eines Web Services kann beliebig sein. Ein Web Service beschreibt nur die Schnittstelle, über die die Implementierung angesprochen werden kann.
Web Service Description Language (WSDL) ist eine XML basierte Sprache zur Beschreibung von Web Services [WSDL07]. In WSDL werden sowohl die abstrakten Operationen als auch konkrete Formate und Protokolle zur Verwendung dieser Operationen definiert.

2.2. BPEL

Web Service Business Process Execution Language (WS-BPEL) ist eine XML basierte Sprache zur Beschreibung von Geschäftsprozessen [BPEL07]. BPEL Prozesse kommunizieren ausschließlich mit Webservices. Ein BPEL Prozess kann selbst wieder als Web Service zur Verfügung gestellt werden.
Zur Modellierung von BPEL Prozessen werden Aktivitäten verwendet. BPEL definiert zwei grundlegende Arten von Aktivitäten, Basis Aktivitäten und strukturierte Aktivitäten. Basis Aktivitäten sind atomare Aktivitäten. Strukturierte Aktivitäten enthalten andere Aktivitäten und erlauben so den Aufbau komplexer Prozessstrukturen.
Die für diese Arbeit interessanten Basis Aktivitäten sind *Invoke* und *Assign.* Mit der Invoke Aktivität wird der Aufruf eines Web Services dargestellt. Dabei werden Variablen für Eingabe- und Ausgabedaten angegeben. Zusätzlich wird spezifiziert, welche Operation von welchem Web Service aufgerufen wird. Mit der Assign Aktivität können Zuweisungen zwischen Variablen realisiert werden. Dazu werden innerhalb einer Assign Aktivität Kopieranweisungen zwischen Variablen oder Teilen von Variablen spezifiziert. Über die Angabe von XPath Ausdrücken sind auch komplexe Zuweisungen möglich.
Die für diese Arbeit interessanten strukturierten Aktivitäten sind *Scope* und *Sequence*. Der Scope erlaubt die Kapselung mehrerer Aktivitäten. Innerhalb eines Scopes können lokale Variablen deklariert werden. Zusätzlich können für einen Scope Ereignis- und Fehlerbehandlung definiert werden. Die Aktivität Sequence enthält Aktivitäten, die nacheinander ausgeführt werden sollen.

2.3. DUNE

Distributed and Unified Numerics Environment[3] (DUNE) ist ein unter Open Source Lizenz entwickeltes Simulationsframework, welches unter anderem an der Universität Stuttgart mit entwickelt wird. Das Framework ist modular aufgebaut und basiert auf C++ und Templateprogrammierung.

[3] http://www.dune-project.org/

In der Diplomarbeit von Jens Rutschmann [Rut09] wird gezeigt, wie DUNE um eine Web Service Schnittstelle erweitert werden kann. Implementiert wird die Web Service Schnittstelle *WSI_Dune* exemplarisch für eine Beispielsimulation „Tinte in Wasser".
In dieser Beispielsimulation wird berechnet, wie sich Tinte in einem Wasserbehälter über die Zeit betrachtet verteilt. Dazu wird zunächst der Ausgangsdatensatz in ein Gitter geladen. Dieser Datensatz beschreibt die Ausgangssituation für dieses Experiment. Die Simulation kann beispielsweise zu dem Zeitpunkt beginnen, an dem die Tinte in das Wasser gegeben wird. Der Datensatz kann aber genauso einen Zustand beschreiben, in dem sich die Tinte bereits zu einem gewissen Grad im Wasser ausgebreitet hat. Nachdem die Ausgangsdaten geladen wurden, wird auf diesem Datensatz die eigentliche Simulation durchgeführt. In einer Schleife wird der zeitliche Verlauf der Verbreitung der Tinte im Wasser berechnet. Jeder Schleifendurchlauf entspricht einer festen Zeitspanne im Experiment. Nach jedem Schleifendurchlauf entspricht der Datensatz, auf dem die Berechnung durchgeführt wird, dem aktuellen Zustand der Verbreitung der Tinte in Wasser. Die Simulation kann nach einer festen Anzahl von Schleifendurchläufen oder abhängig von beliebigen weiteren Kriterien beendet werden.

2.4. Eclipse

Eclipse[4] ist eine in Java entwickelte Anwendung, die durch ihre modulare Struktur erweiterbar und anpassbar ist. Die Module, aus denen Eclipse besteht und durch die es erweitert werden kann, bezeichnet man als *Plugin*. Der Kern von Eclipse, die sogenannte Eclipse Plattform, kann als Basis zur Entwicklung von beliebigen Anwendungen dienen [IBM06]. Der in dieser Arbeit verwendete BPEL Designer ist als Eclipse Plugin implementiert. Dementsprechend werden die in dieser Arbeit entwickelten Komponenten auch in Java implementiert. Im Folgenden werden einzelne Komponenten und Technologien der Eclipse Plattform näher beschrieben.

2.4.1. Eclipse Workbench

Die Benutzeroberfläche in Eclipse wird als *Workbench*[5] bezeichnet. Die Workbench besteht aus Menüs, Toolbars, Statusleiste, einem Editierbereich, Editoren und Views. Editoren sind spezialisiert auf jeweils ein Dateiformat. Wird ein Dokument vom Benutzer geöffnet, so kann das Dokument im Editierbereich mit dem passenden Editor bearbeitet werden. In *Views* werden üblicherweise Navigationshilfen dargestellt oder Eigenschaften des gerade aktiven Editors.
Eine bestimmte Anordnung von Views und dem Editierbereich bezeichnet man in Eclipse als *Perspective*. Über die Toolbar *Perspective* kann der Benutzer zwischen verschiedenen vordefinierten Perspektiven wechseln. Er kann zusätzlich eigene Perspektiven definieren und abspeichern. In Abbildung 1 wird eine Eclipse Workbench dargestellt. Die mit (1) markierten Bereiche sind Views, der mit (2) markierte Bereich ist der Editierbereich.

[4] http://www.eclipse.org/
[5] http://help.eclipse.org/ganymede/index.jsp, Unterpunkt Workbench User Guide

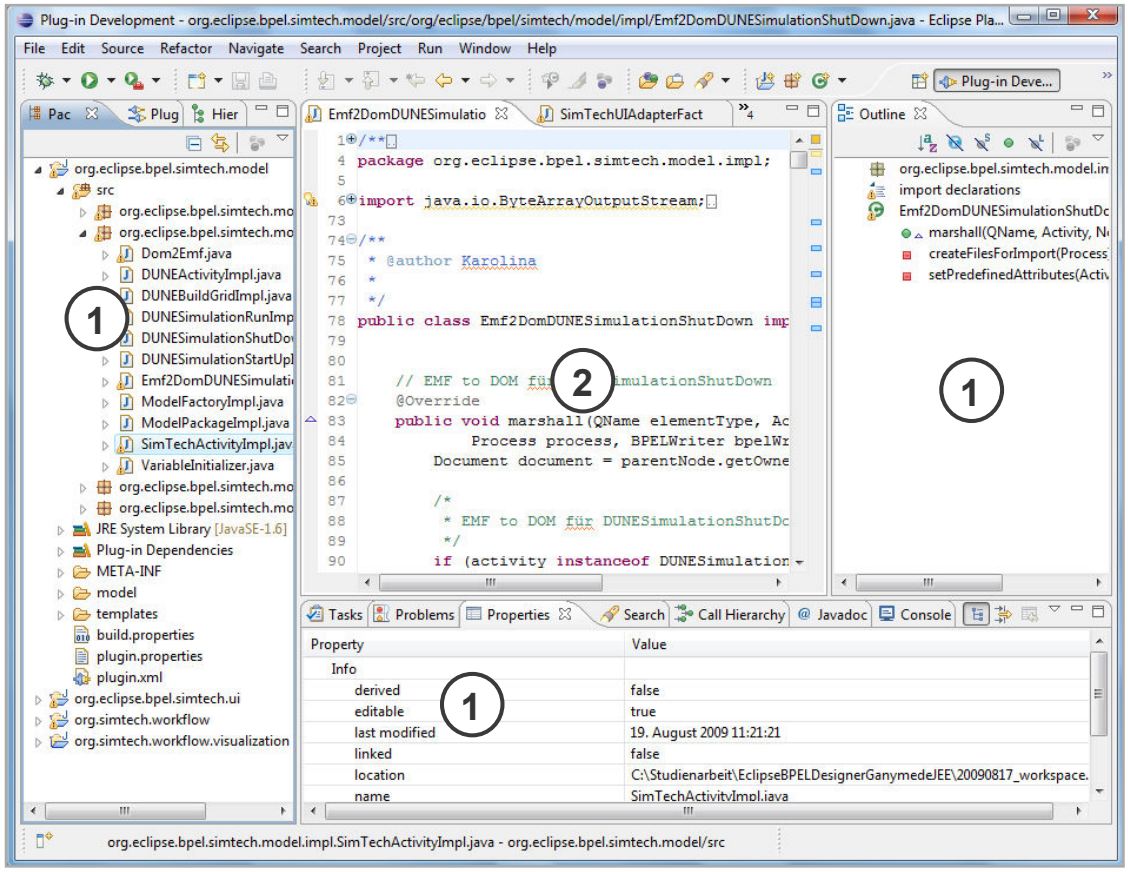


Abbildung 1: Eclipse Workbench

2.4.2. Extension Points und Extensions

Eclipse bietet einen Mechanismus, mit dem ein bestehendes Plugin durch neue Funktionalitäten erweitert werden kann. Dabei wird für das Plugin ein so genannter *Extension Point*[6] definiert. Der Extension Point gibt vor, wo und wie dieses Plugin erweitert werden kann. Andere Plugins können dieses Plugin nun erweitern, indem sie passende *Extensions*[7] deklarieren.

Ein Plugin wird durch die Datei plugin.xml beschrieben, welche im Hauptverzeichnis jedes Plugin Projektes liegt. In dieser Datei werden sowohl Extension Points als auch Extensions deklariert. In der Abbildung 2 werden zwei Plugins dargestellt. Plugin A bietet einen Extension Point an, Plugin B erweitert diesen durch eine entsprechende Extension. In der Deklaration des Extension Points des Plugins A werden eine eindeutige Id, ein Name und ein XML Schema angegeben. In diesem Schema wird beschrieben, wie eine gültige Extension zu diesem Extension Point aussieht. Die Deklaration der Extension im Plugin B bezieht sich auf die Id des Extension Point vom Plugin A und folgt dem dort angegebenen Schema.

[6] http://help.eclipse.org/ganymede/index.jsp, Unterpunkt Platform Plug-in Developer Guide
[7] http://help.eclipse.org/ganymede/index.jsp, Unterpunkt Platform Plug-in Developer Guide

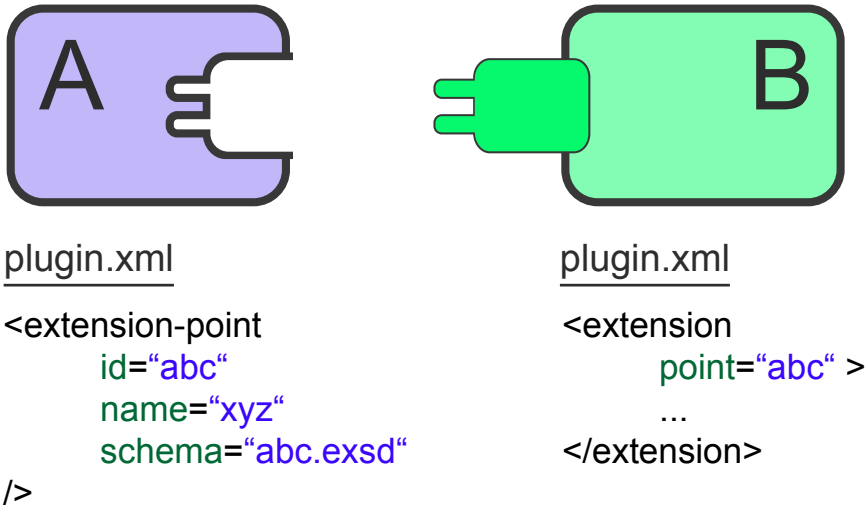


Abbildung 2: Extension Point und Extension

2.5. EMF

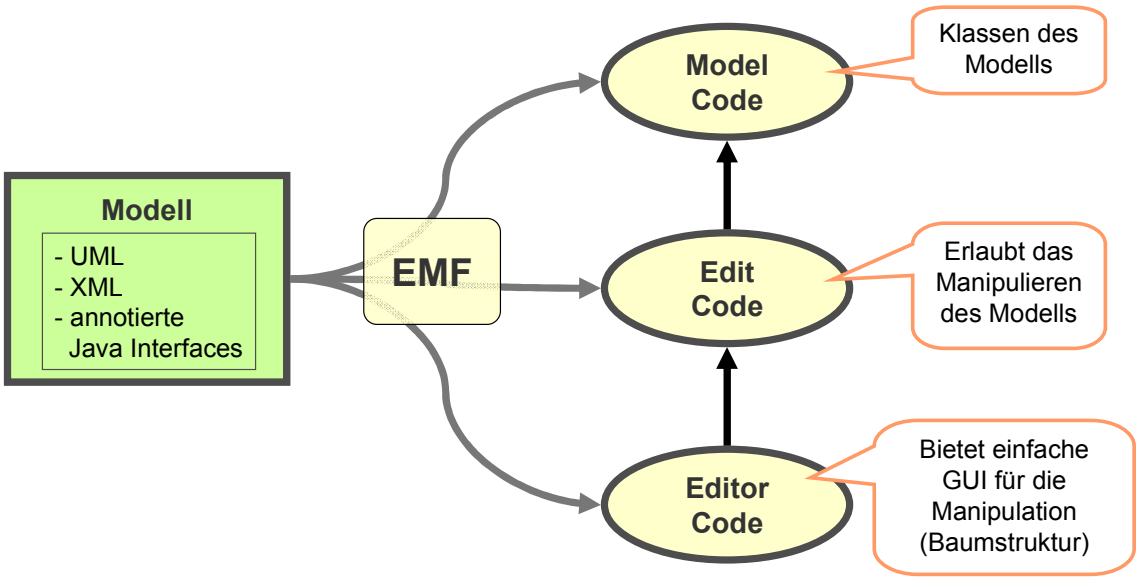


Abbildung 3: EMF

Das *Eclipse Modeling Framework*[8] (EMF) ist ein Java Framework, welches die Entwicklung von Java Anwendungen auf der Basis von strukturierten Modellen unterstützt. In Abbildung 3 ist die Funktionsweise von EMF schematisch dargestellt. Das Ausgangsmodell kann in verschiedenen Formaten vorliegen. Es können Modelle eingelesen werden, die als UML, XML oder annotierte Java Interfaces vorliegen. Das eingelesene Modell wird in ein EMF Modell konvertiert. Dieses Modell kann parametrisiert und angepasst werden. Aus dem EMF Modell wird anschließend Java Code generiert [MDG04].

[8] http://www.eclipse.org/modeling/emf/

2.6. BPEL Designer

Der *Eclipse BPEL Designer* ist eine Anwendung zur graphischen Erstellung von BPEL Prozessen. Der BPEL Designer ist als Eclipse Plugin implementiert. Seine Architektur folgt dem *Model View Controller* (MVC) Prinzip. Das dem BPEL Designer zugrunde liegende Modell bildet den BPEL Standard ab und wurde mit EMF umgesetzt.
Die Editor GUI ist auf Basis des *Graphical Editor Frameworks*[9] (GEF) umgesetzt. GEF ist Teil der Eclipse Plattform und unterstützt die Entwicklung von graphischen Anwendungen nach dem MVC Prinzip [MDG04]. In Abbildung 4 wird das MVC Prinzip in Bezug auf GEF dargestellt. Der Controller wird in GEF durch sogenannte *Editparts* realisiert. Die View wird in GEF mit Draw2D, einem GUI Toolkit der Eclipse Plattform, aufgebaut. Als Modell wird im Zusammenhang mit GEF oft ein EMF Modell verwendet. GEF erlaubt jedoch die Verwendung beliebiger Modelle.

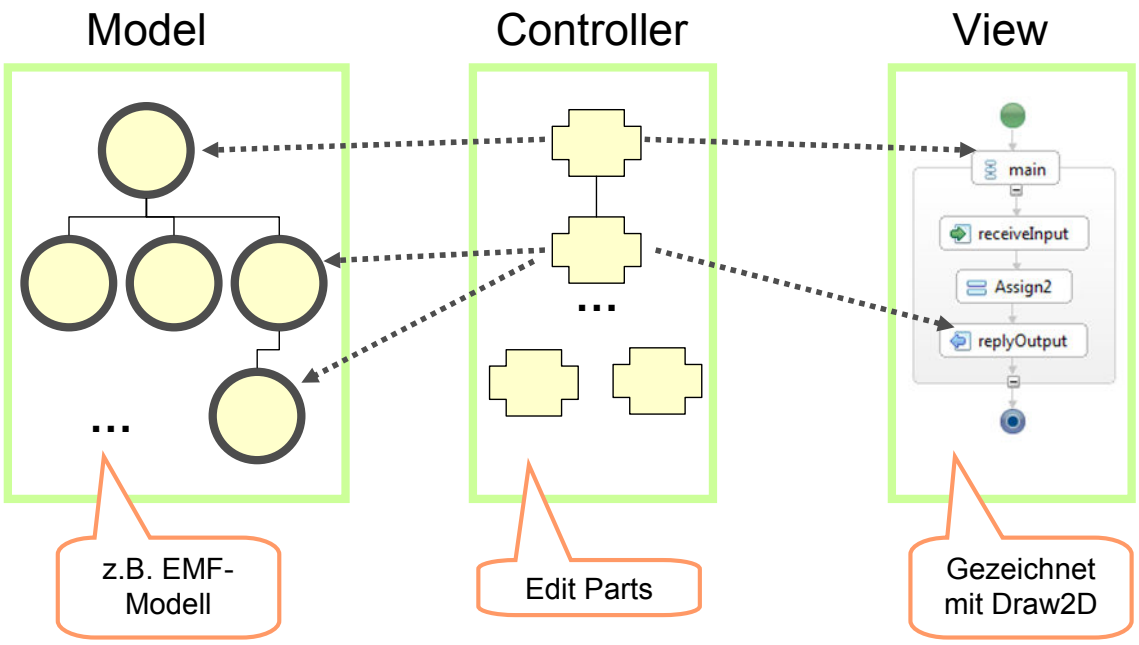


Abbildung 4: MVC mit GEF

[9] http://www.eclipse.org/gef/

3 Architektur des SimTech Workflow Tools

In den klassischen Anwendungsfeldern der Workflow Technologie, den *Business Workflows,* gibt es viele Anwendungsszenarien, in denen die Modellierung, die Ausführung und Überwachung, sowie die Analyse von Workflows voneinander getrennt sind. Jede dieser Phasen kann von unterschiedlichen Personen mit unterschiedlichen Tools durchgeführt werden. Simulations-Workflows unterscheiden sich in diesem Aspekt grundlegend von Business Workflows. Im Bereich der Simulations-Workflows ist es ein übliches Anwendungsszenario, dass die Workflows von der gleichen Person modelliert, ausgeführt, überwacht und analysiert werden.
Das SimTech Workflow Tool soll den Benutzer über den gesamten Lebenszyklus eines Simulations-Workflows hinweg unterstützen. Konkret resultiert daraus die Anforderung, dass das Tool die Komponenten *Modeler*, *Service Catalog*, *Monitoring* und *Result Display* enthalten soll.
Die Umsetzung erfolgt auf Basis der Eclipse Plattform. Als Modeler Komponente wird der Eclispe BPEL Designer[10] verwendet. Die Komponente Service Catalog wird im Rahmen dieser Arbeit prototypisch implementiert. Die Komponente Monitoring wird in [Hau09] entworfen und umgesetzt. Die Komponente Result Display wird im weiteren Verlauf des SimTech Projektes entwickelt. In dieser Arbeit werden aber bereits Schnittstellen definiert, über welche die genannten Komponenten in das SimTech Workflow Tool eingebunden werden können.

3.1. Konzept

Im Lebenszyklus eines Simulationsexperimentes werden je nach Phase unterschiedliche Komponenten benötigt. Die erste Phase ist die „Modellierung“. In dieser Phase wird mit der Komponente „Modeler“ der Simulations-Workflow aufgebaut. Zusätzlich wird in dieser Phase die Komponente „Service Catalog“ benötigt, um die dort definierten simulationsspezifischen Aktivitäten bei der Modellierung verwenden zu können.
Ist die Modellierung abgeschlossen, wird der Workflow ausgeführt und während dessen überwacht. Über die Komponente „Monitoring“ kann der aktuelle Ausführungszustand erfasst werden. Die Komponente „Result Display“ kann in dieser Phase bereits Zwischenergebnisse darstellen.
Nach der Ausführung des Workflows werden die Ergebnisse analysiert. Dies geschieht mit Hilfe der Komponente „Result Display“. Über die Komponente „Monitoring“ kann weiterhin die Ausführungshistorie des Workflows nachvollzogen werden, falls diese für die Analyse der Ergebnisse relevant ist.

3.2. Umsetzung in Eclipse

Die Komponente Modeler ist ein Eclipse Editor, alle weiteren Komponenten werden als Views dargestellt. Um die verschiedenen Phasen im Lebenszyklus eines Simulations-Workflows zu unterstützen, bietet das SimTech Workflow Tool mehrere Perspektiven. Jede Perspektive definiert, welche Komponenten in der jeweiligen Phase angezeigt werden.

[10] http://www.eclipse.org/bpel/

Die *SimTech Modeling Perspective* bietet alle Informationen, die der Benutzer für die Modellierung eines Simulations-Prozesses benötigt. In dieser Phase sind der Modeler und der Simtech Service Catalog die einzigen aktiven Komponenten.
In Abbildung 5 wird diese Perspektive dargestellt. In der Mitte des Fensters befindet sich der Editierbereich, in dem ein Prozess modelliert werden kann (2). Wenn ein BPEL-Simulationsprozess geöffnet wird, so wird hier das Prozessmodell graphisch dargestellt. Neben dem Prozessmodell werden alle verwendeten Variablen, Partner Links usw. im sogenannten *Process Tray* (3) dargestellt. Die View *Palette* (4) bietet klassische BPEL-Aktivitäten. In der Kategorie *SimTech Service Catalog* (5) befinden sich zudem ausspezifizierte SimTech Aktivitäten. Alle Aktivitäten der Palette können für die Modellierung verwendet werden. Die View *Ressourcen Navigator* (1) stellt die Prozesse und alle anderen Dokumente des Arbeitsordners (Workspace) in einer Ordnerstruktur dar. Die View *Outline* (6) bietet die gleichen Daten wie der Process Tray innerhalb des Editierbereichs, nur in einer anderen Darstellung. Die View *Properties* (7) zeigt die Eigenschaften eines gerade ausgewählten Elementes im Editierbereich bzw. der Outline View. Im rechten oberen Bereich des Fensters kann der Benutzer zwischen den Perspektiven wechseln (8).

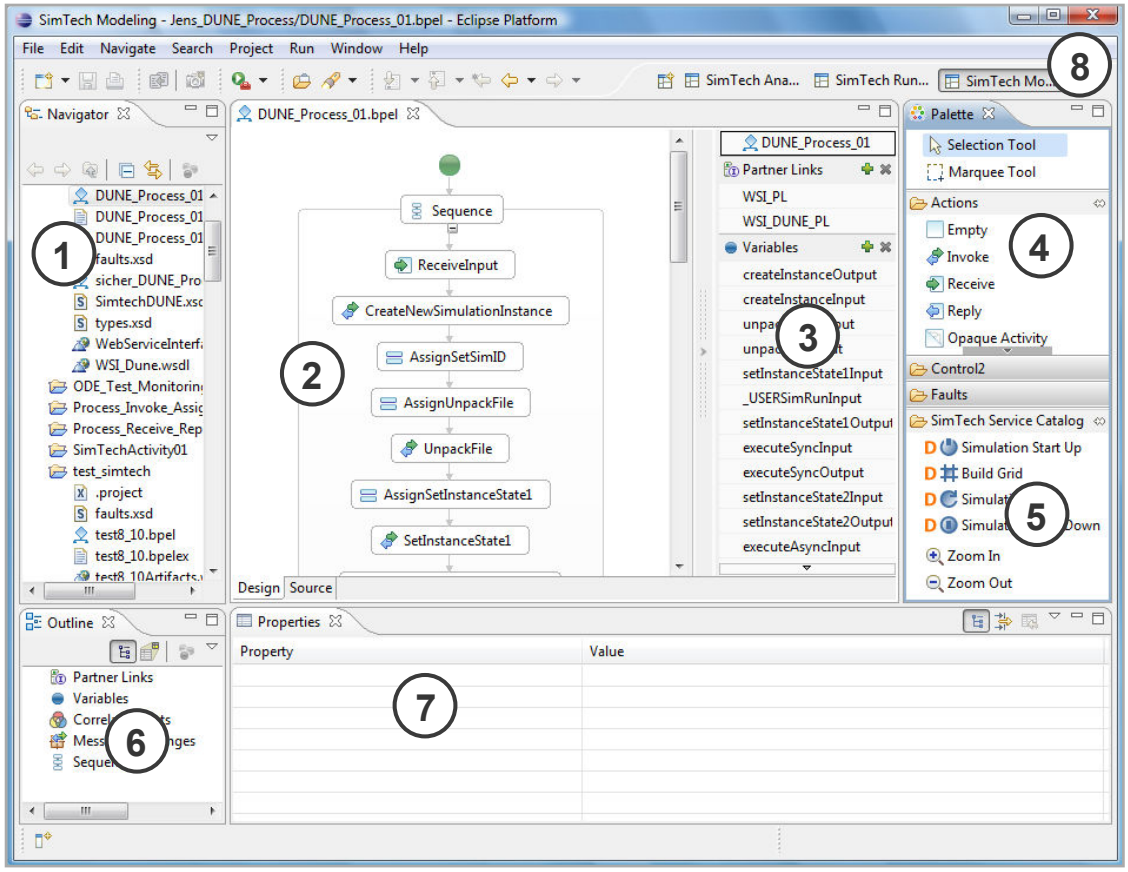


Abbildung 5: SimTech Modeling Perspective

Die *SimTech Runtime Perspective* unterstützt den Benutzer bei der Ausführung und Überwachung von Simulationsexperimenten. Der Fokus dieser Perspektive liegt auf den Komponenten Monitoring und Result Display. Die Modeler Komponente wird aber auch weiterhin dargestellt, damit der Benutzer die Informationen aus dem Monitoring mit dem Prozessmodell abgleichen kann.

Abbildung 6 zeigt diese Perspektive. Die View *Monitoring* (1) gibt Auskunft über den aktuellen Zustand der Ausführung. Die View *Result Display* (2) stellt die Zwischenergebnisse und Endergebnisse des Simulationsexperimentes grafisch dar.

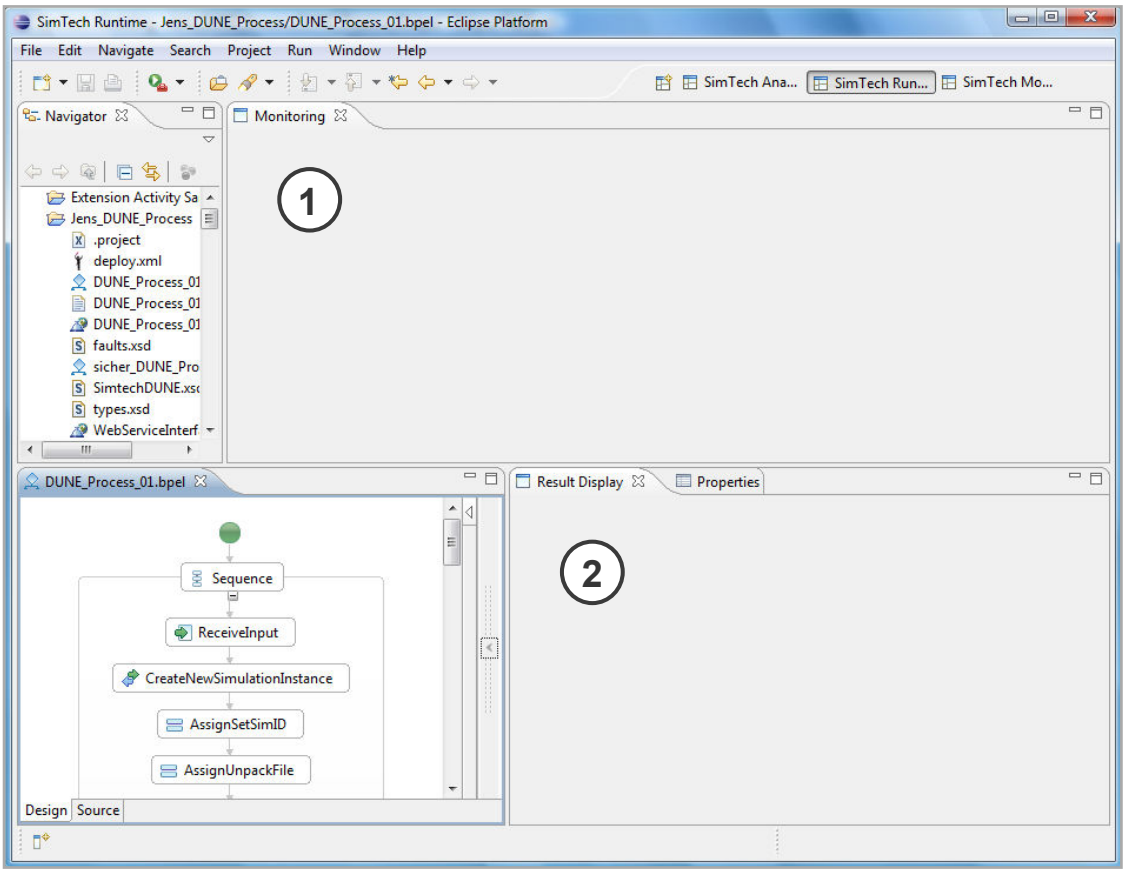


Abbildung 6: SimTech Runtime Perspective

In der *SimTech Analysis Perspective* können die Ergebnisse eines Simulations-Workflows analysiert werden Dies geschieht hauptsächlich in der Komponente Result Display. Die Komponente Monitoring bietet noch zusätzlich Informationen über die Ausführungshistorie des Workflows. In Abbildung 7 wird diese Perspektive dargestellt. Die View Result Display (1) ist im oberen Bereich des Fensters angeordnet und bietet viel Platz für die Analyse. Die View Monitoring (2) befindet sich direkt darunter.

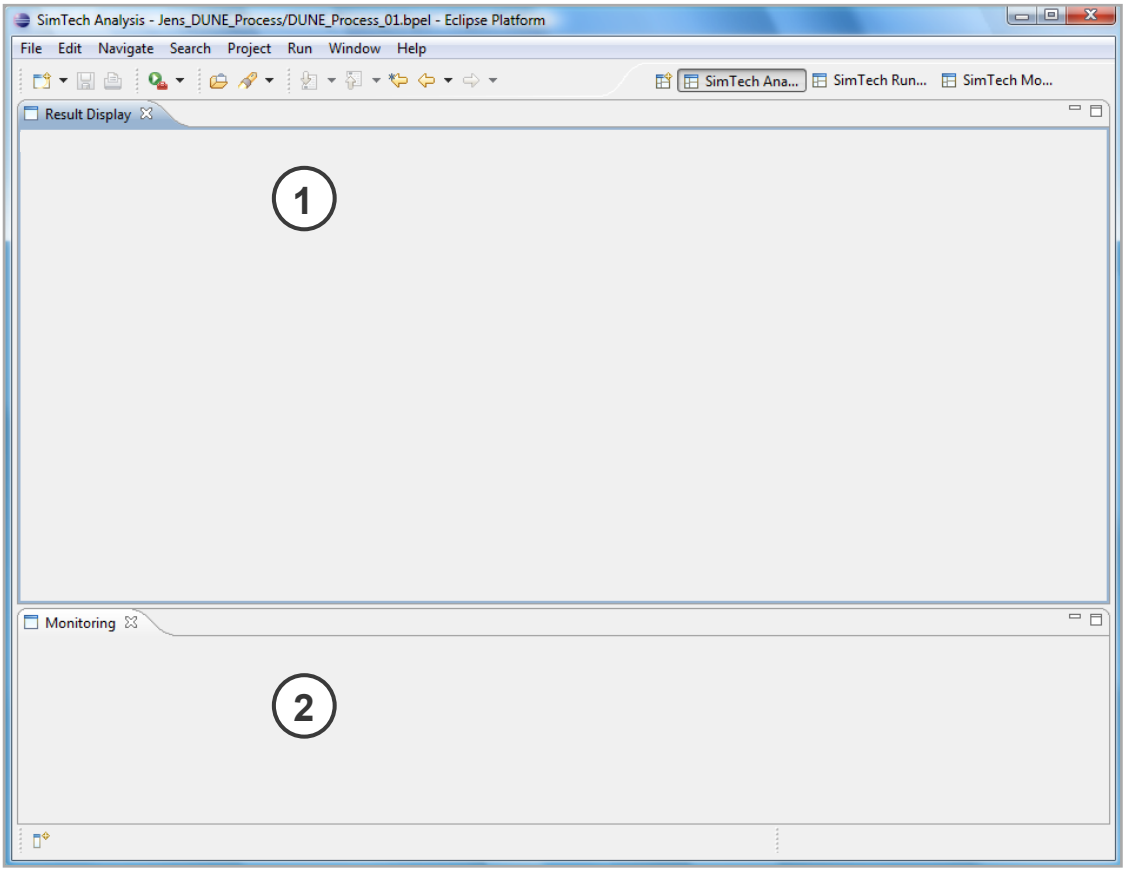


Abbildung 7: SimTech Analysis Perspective

3.3. Implementierung

Die Architektur des SimTech Workflow Tools wird durch das Plugin *org.simtech.workflow* implementiert. Eclipse bietet für die Entwicklung von Perspektiven den Extension Point *org.eclipse.ui.perspectives.* Die Klassen *ModelingPerspective, RuntimePerspective* und *AnalysisPerspective,* die als Extension für diesen Extension Point in der plugin.xml des Projektes registriert werden, müssen dabei jeweils das Interface *IPerspectiveFactory* implementieren. In der Methode createInitialLayout() wird das Layout der Perspektive angelegt, in dem die entsprechenden Views und der Editierbereich positioniert werden und die Größe der einzelnen Elemente festgelegt wird. Die so entwickelten Perspektiven *SimTech Modeling*, *SimTech Runtime und SimTech Analysis* verwenden Eclipse Standard Views wie Ressource Navigator, Outline, Property und Palette, sowie die eigens entwickelten Views Monitoring und Result Display.
Für die Entwicklung von Views bietet Eclipse den Extension Point *org.eclipse.ui.views*. Die Klassen *MonitoringView* und *VisualizationView* werden als Extension für den Extension Point in der plugin.xml des Projektes registriert. Diese Klassen erben von der abstrakten Klasse *ViewPart* und überschreiben die Methode *createPartControl(Composite parent).* In Abbildung 8 wird ein Überblick über die Implementierung der Perspektiven und Views gegeben.
Die von den Klassen MonitoringView und VisualizationView implementierten Views Monitoring und Result Display sollen Platzhalter sein für ein Monitoring-Tool und ein Visualisierungs-Tool, deren Umsetzung nicht Gegenstand dieser Arbeit ist, aber in näherer Zukunft geplant ist. Trotzdem soll in dieser Arbeit eine Schnittstelle definiert werden, über die die genannten

Komponenten Monitoring und Result Display in das SimTech Workflow Tool eingebunden werden können. Die Schnittstelle wird über die beiden Extension Points *org.simtech.workflow.views.monitoring* und org. *simtech.workflow.views.visualization* definiert.
Beim Instanziieren der Platzhalter Views wird über die *Eclipse Extension Registry* abgefragt, ob sich Extensions für die oben genannten Extension Points registriert haben. Wenn dies der Fall ist, dann werden die in den Extensions deklarierten Views ebenfalls instanziiert und mit den Plathalter Views verknüpft. Ab diesem Zeitpunkt werden alle Aufrufe an die Platzhalter Views weiter geleitet an diese Views. Das SimTech Plugin behält damit weiterhin die Kontrolle über die Positionierung und Größe der Views in den Perspektiven, trotzdem kann von außen Funktionalität implementiert werden, so wie es auch über den Eclipse Extension Point *org.eclipse.ui.views* möglich gewesen wäre.

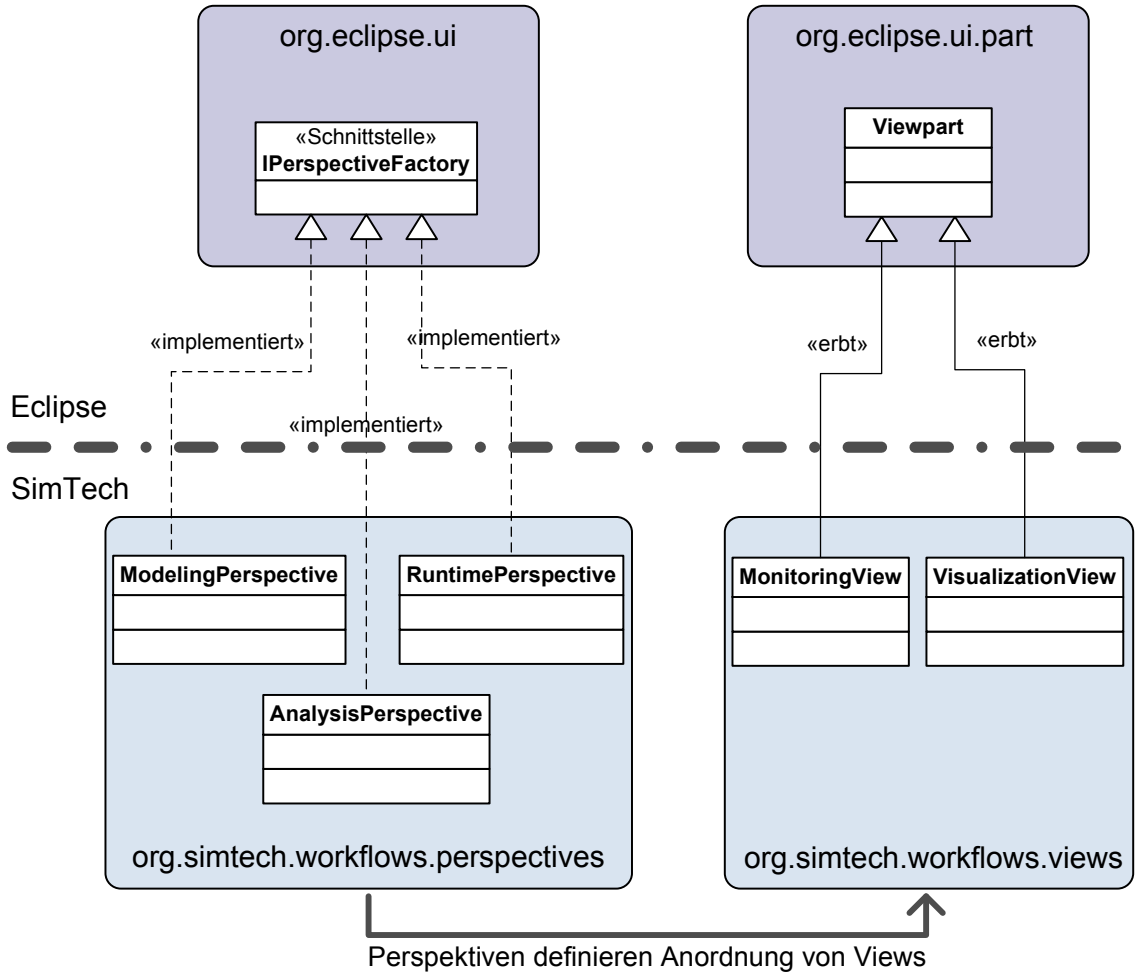


Abbildung 8: Klassendiagramm - Perspektiven und Views

4 SimTech Service Catalog

Ein weiterer Teil dieser Arbeit ist die Entwicklung einer Komponente *SimTech Service Catalog.* Dieser Service Catalog stellt dem Benutzer simulationsspezifische Aktivitäten zur Modellierung zur Verfügung. Diese im Folgenden als *SimTech Aktivitäten* bezeichneten Aktivitäten repräsentieren ausspezifizierte BPEL Prozessfragmente, welche im Prozess aber als einzelne Aktivitäten vom Typ „SimTech Aktivität“ dargestellt werden. Die SimTech Aktivitäten besitzen ein SimTech spezifisches Erscheinungsbild, welches sie von den BPEL Aktivitäten unterscheidet. Die Verwendung der SimTech Aktivitäten erspart es dem Benutzer, Prozessfragmente zu modellieren und auszuspezifizieren, die immer wieder in gleicher oder ähnlicher Form in vielen Simulationsexperimenten vorkommen.
Die Komponente wird prototypisch implementiert und in ein bereits bestehendes Workflow Modellierungstool, den Eclispe BPEL Designer, eingebunden. Es werden SimTech-Aktivitäten definiert um die in [Rut09] vorgestellte DUNE Simulation „Tinte in Wasser“ modellieren zu können.

4.1. Konzept

Beim Erstellen eines Prozesses kann man zwischen den Tätigkeiten „Modellieren“ und „Ausspezifizieren“ unterscheiden. Beim Modellieren wird die Struktur des Prozesses aufgebaut, indem Aktivitäten erzeugt und angeordnet werden. Das Ausspezifizieren beinhaltet das Setzen von Attributen der erzeugten Aktivitäten sowie das Anlegen von Variablen, Partnerlinks und Imports.
In Kapitel 4.1.1 wird ein konkreter DUNE Simulationsprozess vorgestellt und gezeigt, wie dieser Prozess in BPEL modelliert und ausspezifiziert werden kann. In Kapitel 4.1.2 wird das allgemeine Konzept und der Aufbau der SimTech Aktivität vorgestellt, bevor in Kapitel 4.1.3 die konkreten SimTech Aktivitäten für den vorgestellten DUNE Prozess entwickelt werden. In Kapitel 4.1.4 wird dann gezeigt, wie dieser Prozess mit Hilfe dieser SimTech Aktivitäten aufgebaut werden kann, und welche Vorteile die Verwendung der SimTech Aktivitäten bietet. Kapitel 4.1.5 gibt einen kurzen Überblick, wie die SimTech Aktivitäten in den Eclipse BPEL Designer integriert werden.

4.1.1. Ausgangsprozess

Zunächst wird die DUNE Simulation „Tinte in Wasser“ mit BPEL Aktivitäten modelliert. Das Ausspezifizieren dieses Prozesses erfolgt in mehreren Schritten, die im Folgenden beschrieben werden.
Um den DUNE Web Service benutzen zu können, werden zuerst die entsprechenden WSDL Dateien importiert. In einer weiteren WSDL Datei werden zusätzliche prozessspezifische Datentypen definiert. Anschließend werden alle benötigten Partnerlinks im BPEL Prozess angelegt.
Für jede Invoke Aktivität wird spezifiziert, welcher Partnerlink und welche Operation verwendet werden. Als weiteres wird für jede Invoke Aktivität eine globale Input- und eine globale Outputvariable definiert und der jeweiligen Aktivität zugeordnet.
In jeder Assign Aktivität wird angegeben, aus welchem Part einer Quellvariable jeweils Daten kopiert werden und in welchen Part der Zielvariable sie eingefügt werden. Vor jedem dieser Kopiervorgänge muss noch die Zielvariable (Input Variable der Invoke Aktivität) initialisiert werden.

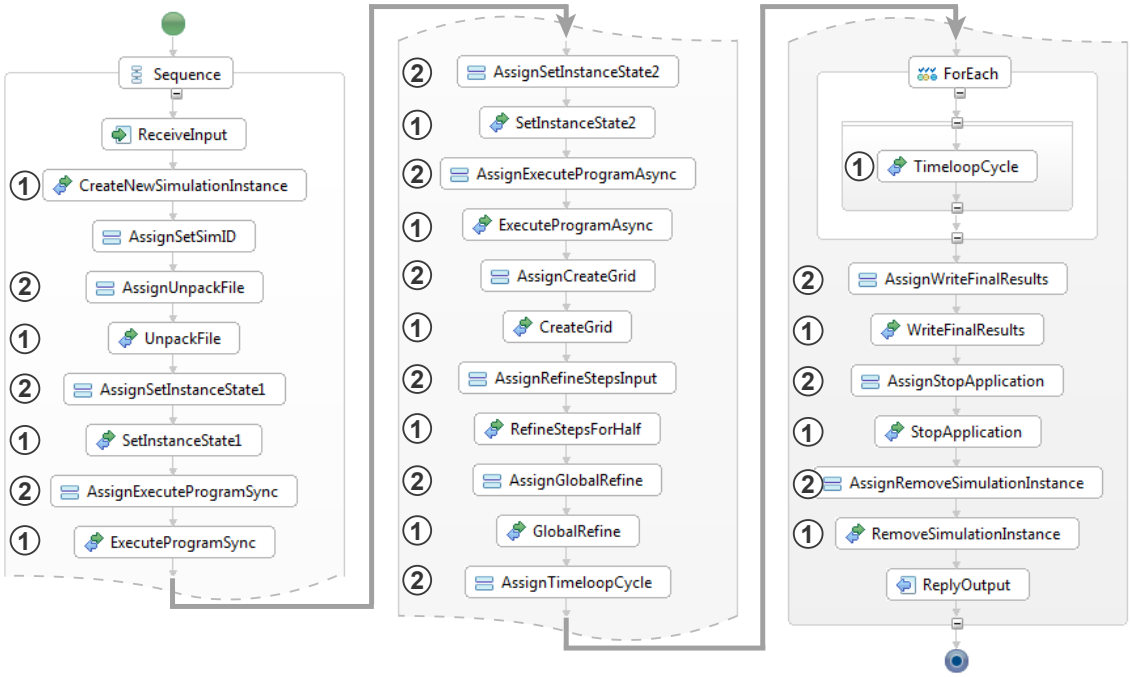


Abbildung 9: Simulationsprozess "Tinte in Wasser"

Abbildung 9 zeigt den entstandenen Simulations-Prozess. Die mit (1) markierten Aktivitäten stellen Invoke Aktivitäten dar. Jede dieser Invoke Aktivitäten ruft eine Operation der DUNE Web Service Schnittstelle auf. Die mit (2) markierten Aktivitäten sind Assign Aktivitäten. Sie kopieren die jeweils benötigten Daten in die Inputvariable der nachfolgenden Invoke Aktivität.

4.1.2. Aufbau einer SimTech Aktivität

Jede SimTech Aktivität hat genau ein Kindelement. Das Kindelement ist eine Scope Aktivität. Innerhalb dieses Scopes wird das Prozessfragment modelliert, welches das Verhalten der SimTech Aktivität beschreibt. Innerhalb eines Scopes ist es möglich, Variablen und Partnerlinktypes zu definieren. Dadurch lassen sich mögliche Konflikte mit bereits bestehenden Variablen im Prozess vermeiden. Der Scope unterstützt somit die Kapselung der SimTech Aktivität und verhindert Nebeneffekte. Außerdem kann man auf diesem Weg für jede SimTech Aktivität spezifische Fault-, Termination- und Compensationhandler definieren.
Beim Einfügen einer SimTech Aktvität in ein Prozessmodell werden alle für diese Aktivität benötigten Daten programmatisch erzeugt. Im Einzelnen bedeutet dies, dass alle benötigten WSDL Dateien importiert und alle globalen Variablen im Prozess definiert werden. Zudem werden alle benötigten PartnerLinks und lokale Variablen im Scope definiert. Als weiteres werden alle Assign und Invoke Aktivitäten ausspezifiziert. Den Invoke Aktivitäten werden die entsprechenden Input und Output Variablen mit dem passendem Typ zugeordnet und für jede Invoke Aktivität der PartnerLink und die Operation gesetzt. Für jede Assign Aktivität werden die entsprechenden Copy-Befehle erzeugt und die notwendigen Initialisierungen der Variablen umgesetzt.
Für den Beispielprozess „Tinte in Wasser“ gibt es nur zwei globale Variablen, *_StDSimId* und *_USERSimRunInput*. In der Variablen _StDSimId wird die Identifikationsnummer der jeweiligen Simulationsinstanz gespeichert. Alle SimTech Aktivitäten arbeiten mit dieser Identifikationsnummer und können über die globale Variable _StDSimId darauf zugreifen.
Für die SimTech Aktivität *DUNESimulationRun* wird die globale Variable *_USERSimRunInput* erzeugt. Diese Variable erwartet als einzige Benutzereingaben.

4.1.3. SimTech Aktivitäten für DUNE

Im Folgenden wird ein Mapping von BPEL Aktivitiäten auf SimTech Aktivitäten für DUNE definiert. BPEL Aktivitäten, die inhaltlich zusammengehören, werden als eine SimTech Aktivität dargestellt.

In Abbildung 10 wird das Mapping der BPEL Aktivitäten *CreateNewSimulationInstance, AssignSetSimID, AssignUnpackFile, UnpackFile, AssignSetInstanceState1, SetInstanceState1, AssignExecuteProgramSync, ExecuteProgramSync, AssignSetInstanceState2, SetInstanceState2, AssignExecuteProgramAsync und ExecuteProgramAsync* auf die SimTech Aktivität *DUNESimulationStartUp* gezeigt. In diesem Prozessfragment wird eine Simulationsinstanz vorbereitet und gestartet. Da das DUNE Framework auf Templateprogrammierung basiert, wird der entsprechende Quellcode vor jeder Ausführung neu gebaut und installiert.

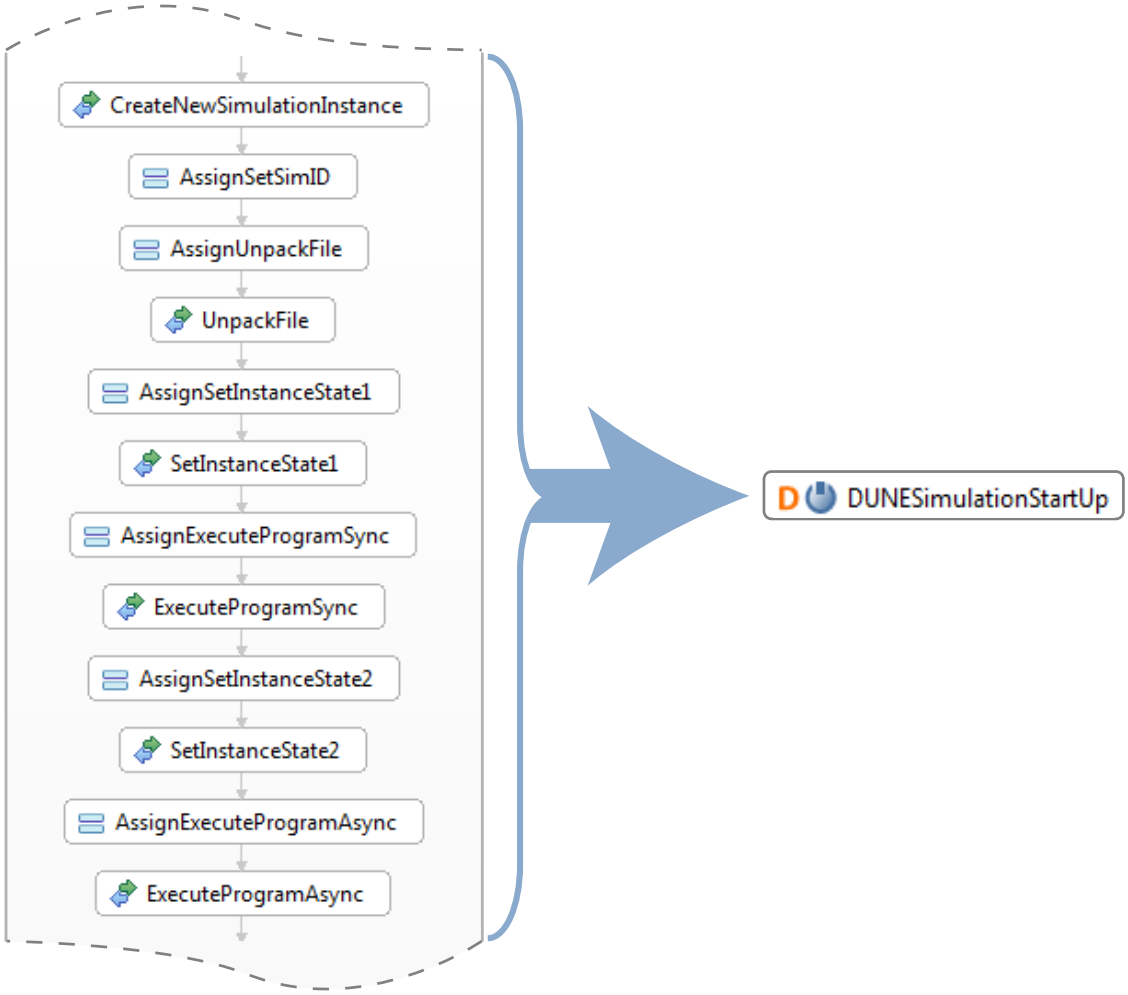


Abbildung 10: Mapping auf SimTech Aktivität "DUNESimulationStartUp

In Abbildung 11 wird das Mapping der BPEL Aktivitäten *AssignCreateGrid, CreateGrid, AssignRefineStepsInput, RefineStepsForHalf, AssignGlobalRefine* und *GlobalRefine* auf die SimTech Aktivität *DUNEBuildGrid* dargestellt. In diesen Schritten wird das Gitter, auf welchem die spätere Simulation berechnet wird, erzeugt und vorbereitet.

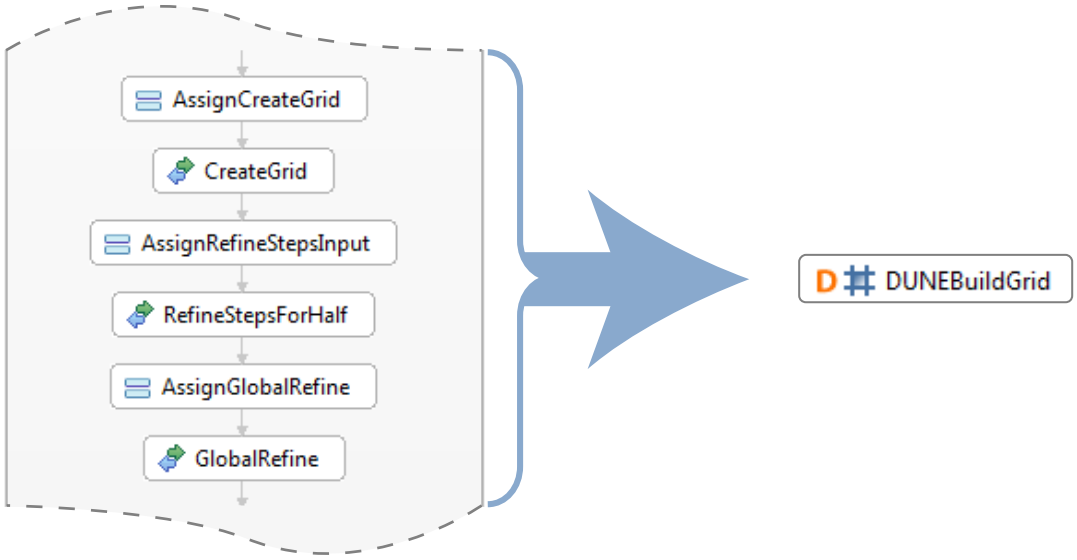


Abbildung 11: Mapping auf SimTech Aktivität "DUNEBuildGrid"

In Abbildung 12 wird das Mapping der BPEL-Aktivitäten *AssignTimeloopCycle, ForEach, AssignWriteFinalResults* und *WriteFinalResults* auf die SimTech Aktivität *DUNESimulationRun* beschrieben. In diesem Prozessfragment erfolgen die eigentliche Berechnung der Simulation und das Speichern der Ergebnisse.
Für die SimTech Aktivität *DUNESimulationRun* wird die globale Variable *_USERSimRunInput* erzeugt. Hier muss der Benutzer angeben, wie viele Simulationsschritte bei einem Aufruf des Web Services berechnet werden sollen. Dieser Wert wird der entsprechenden Operation des Web Services als Parameter übergeben. Als weiteres muss der Benutzer festlegen, wie oft diese Operation insgesamt aufgerufen werden soll. Dieser Wert beeinflusst die Laufvariable der ForEach Aktivität.

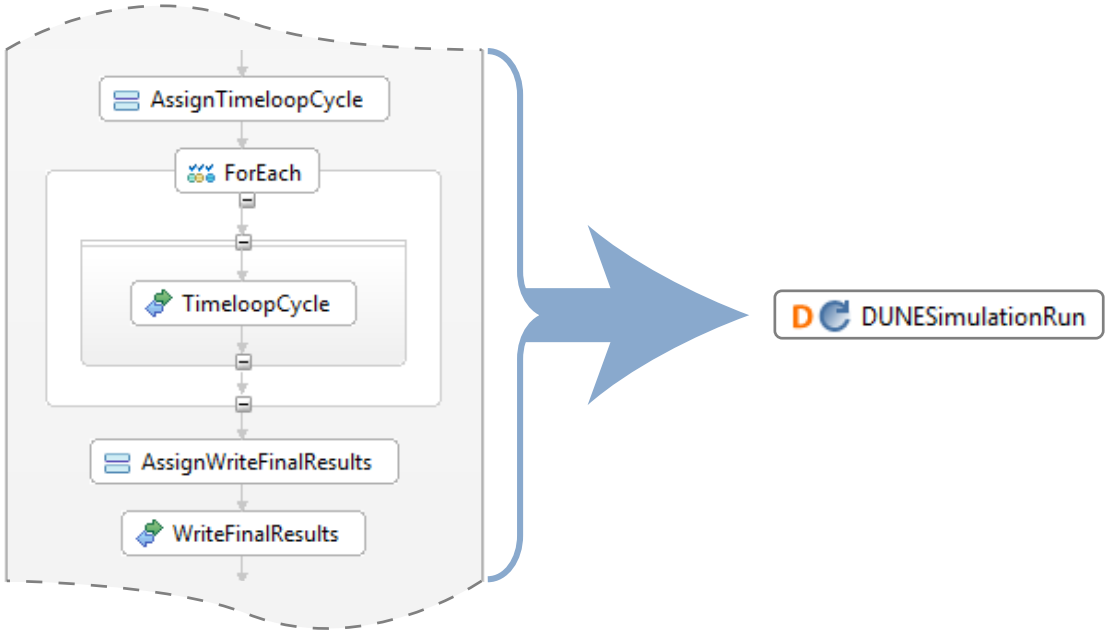


Abbildung 12: Mapping auf SimTech Aktivität "DUNESimulationRun"

In Abbildung 13 wird das Mapping der BPEL Aktivitäten *AssignStopApplication, StopApplication, AssignRemoveSimulationInstance* und *RemoveSimulationInstance* auf die SimTech Aktivität *DUNESimulationShutDown* gezeigt. In diesem Prozessfragment wird die Simulationsinstanz gestoppt und gelöscht.

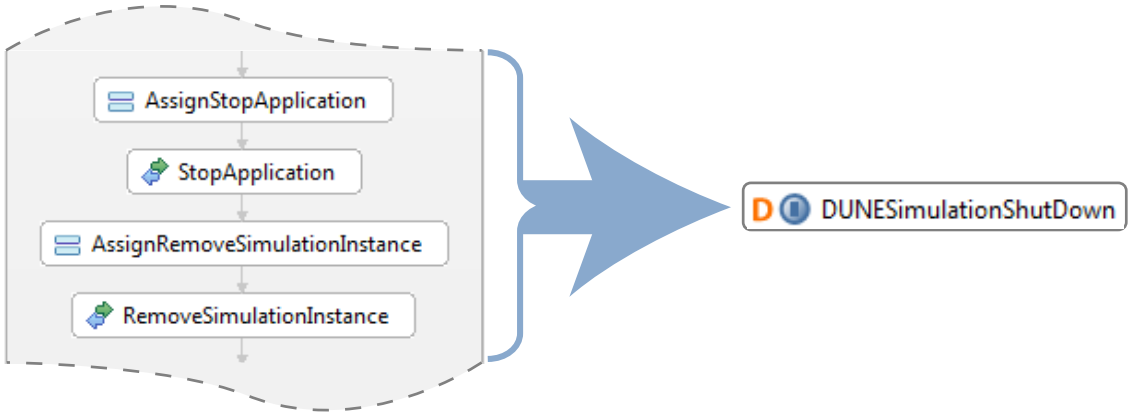


Abbildung 13: Mapping auf SimTech Aktivität "DUNESimulationShutDown"

4.1.4. Prozess mit SimTech Aktivitäten

Mit Hilfe der neu entwickelten ausspezifizierten SimTech Aktivitäten *DUNESimulationStartUp*, *DUNEBuildGrid*, *DUNESimulationRun* und *DUNESimulationShutDown* kann der Benutzer in nur wenigen Schritten einen Simulationsprozess „Tinte in Wasser“ modellieren. Dem Benutzer werden das Modellieren und das Ausspezifizieren der einzelnen BPEL Prozessfragmente abgenommen. Im Vergleich zum Ausgangsprozess erscheint der mit Hilfe der SimTech Aktivitäten modellierter Simulationsprozess nun übersichtlich und leicht verständlich. In Abbildung 14 werden beide Modellierungsvarianten gegenüber gestellt. Links sieht man den Ausgangsprozess ohne SimTech Aktivitäten, rechts den gleichen Prozess mit SimTech Aktivitäten. Zwischen den Prozessen wird mit Pfeilen das Mapping der Prozessfragmente auf die jeweilige SimTech Aktivität abgebildet. Im neuen Prozess ist es jedoch weiterhin möglich, die Struktur der einzelnen SimTech Aktivitäten anzusehen und gegebenenfalls anzupassen. Dazu können die SimTech Aktivitäten, wie am Beispiel der Aktivität DUNESimulationShutDown dargestellt, auf und zu geklappt werden.

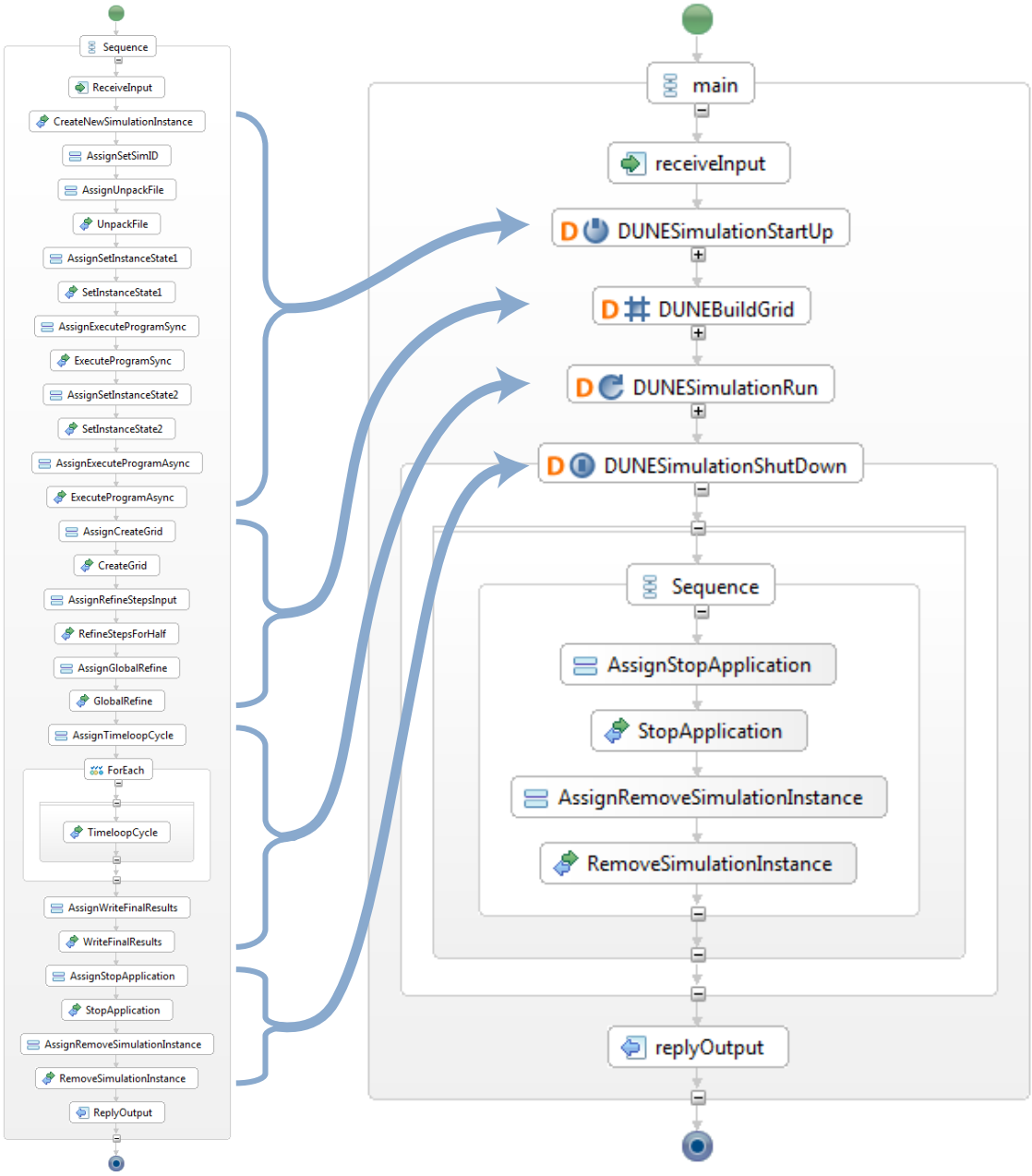


Abbildung 14: Prozess ohne (links) und mit SimTech Aktivitäten (rechts)

4.1.5. Integration in den BPEL Designer

Der Benutzer kann die SimTech Aktivitäten genau so benutzen wie die normalen BPEL Aktivitäten. Die Palette des BPEL Designers wird dafür um eine neue Kategorie SimTech Service Catalog erweitert, welche die einzelnen SimTech Aktivitäten zur Verfügung stellt. Abbildung 15 zeigt die BPEL Designer Palette mit der geöffneten SimTech Kategorie. Die Kategorien *Actions, Control2* und *Faults* enthalten Standard BPEL Aktivitäten. Wählt der Benutzer nun eine SimTech Aktivität in der Palette aus und klickt darauf hin in das Modellierfenster des BPEL Designers, so wird hier eine entsprechende SimTech Aktivität eingefügt.

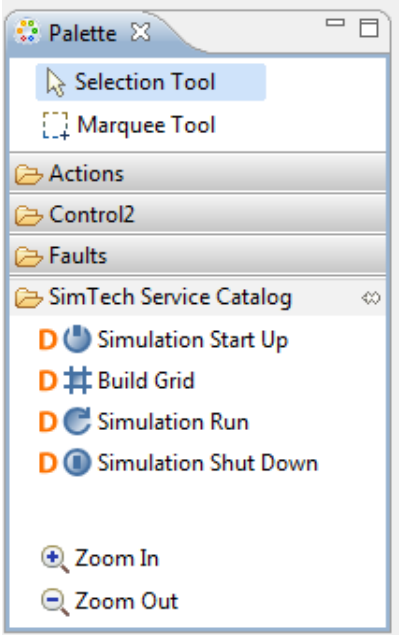


Abbildung 15: Palette mit Kategorie “SimTech Service Catalog”

Für jede SimTech Aktivität wird ein SimTech spezifisches Icon entwickelt. Wie in Abbildung 16 zu sehen ist, besteht ein SimTech Icon aus zwei Symbolen. Das erste Symbol beschreibt das Simulationsframework. Das zweite Symbol beschreibt den Aktivitätstyp. In der Beispielsimulation „Tinte in Wasser“ werden nur SimTech Aktivitäten für das DUNE Simulationsframework entwickelt. In Zukunft sollen aber auch SimTech Aktivitäten für andere Frameworks wie beispielsweise ChemShell entwickelt werden.

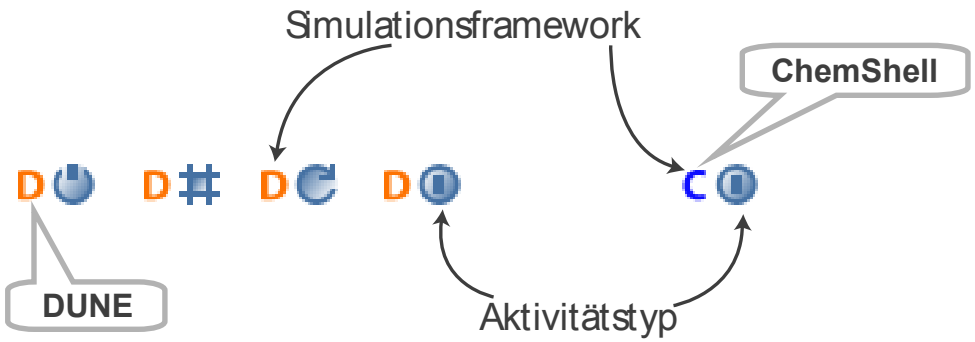


Abbildung 16: SimTech spezifische Icons

4.2. Implementierung

Ziel dieser Arbeit ist es, den BPEL Designer um SimTech Aktivitäten zu erweitern. Die Erweiterung wird in Form von Plugins vorgenommen. Der Quellcode des BPEL Designers wird nicht verändert.
Da der BPEL Designer den BPEL Standard implementiert, bietet er explizite Erweiterungsmöglichkeiten nur dort, wo es auch der Standard definiert. Der BPEL Standard definiert als Erweiterungsmöglichkeit eine sogenannte *Extension Activity* [BPEL07]. Diese Aktivität kann bei der Modellierung wie eine ganz normale BPEL Aktivität verwendet werden. Der Inhalt und die Semantik einer Extension Activity ist nicht Teil des Standards, sondern kann

beliebig definiert werden. Der BPEL Designer ist so erweiterbar, dass er mit Extension Activities umgehen kann.
Der Zweck von Extension Activites ist es, BPEL um zusätzliche Funktionalitäten zu erweitern. Die vorgestellten SimTech Aktivitäten erweitern BPEL nicht. Sie repräsentieren standardkonforme BPEL Prozessfragmente. Das Konstrukt der Extension Activity wird in dieser Arbeit dennoch verwendet, da dies die einzige Möglichkeit ist, den BPEL Designer von außen ohne Eingriffe in den Quellcode in der gewünschten Art und Weise zu erweitern.
In Kapitel 4.2.7 wird beschrieben, wie Prozesse mit SimTech Aktivitäten in äquivalente Prozesse ohne Extension Activities transformiert werden können. Die so erzeugten Prozesse enthalten keine zweckentfremdeten Extension Activities mehr und sind auf jeder BPEL Workflow Engine problemlos ausführbar.
Um den BPEL Designer auf diesem Weg um neue Aktivitäts-Typen zu erweitern, muss an verschiedenen Stellen neue Funktionalität von außen hinzugefügt werden. Im Folgenden werden die einzelnen Schritte beschrieben, die nötig sind, um den BPEL Designer um SimTech Aktivitäten zu erweitern [HIM06].

4.2.1. Erweiterung des EMF Modells

Der BPEL Designer basiert auf EMF. Alle BPEL Aktivitäten wurden als EMF Modellelemente definiert. Daraus folgt, dass auch die SimTech Aktivitäten in einem EMF Modell definiert werden. Das SimTech Modell ist in Abbildung 17 dargestellt. Um spätere Erweiterungen des Modells zu unterstützen, ist es hierarchisch aufgebaut.
Eine *SimTechActivity* erbt von der BPEL *ExtensionActivity*. Sie hat genau ein Kindelement vom Typ Scope. Da das SimTech Workflow Tool in Zukunft neben DUNE auch andere Simulationssysteme unterstützen soll, wird eine *DUNEActivity* eingeführt. Diese kapselt alle DUNE Aktivitäten und erbt von SimTechActivity.
Aus dem so aufgebauten Modell wird anschließend EMF Model Code und EMF Edit Code generiert. Im Folgenden wird dieser Code noch erweitert und modifiziert.

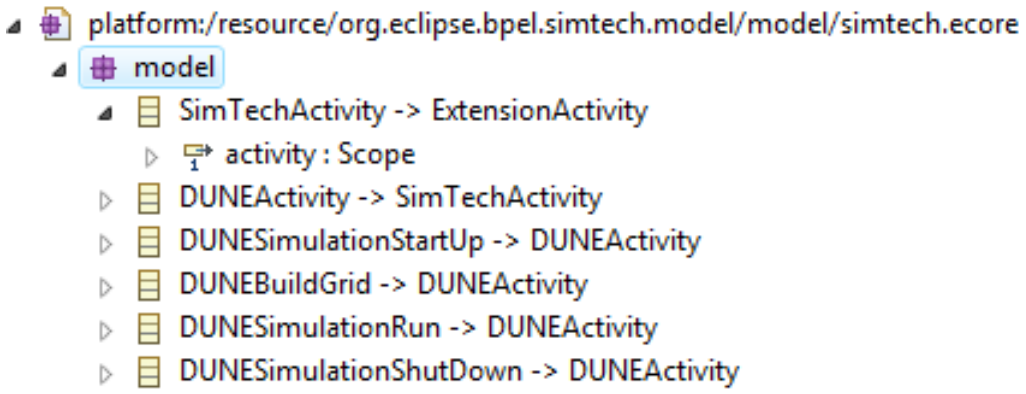


Abbildung 17: EMF Ecore Modell für die SimTech Aktivitäten

4.2.2. Serialisierer und Deserialisierer

Wird im BPEL Designer eine BPEL Datei geöffnet, so wird der Dateiinhalt zunächst einmal als DOM[11] Baum eingelesen. Im nächsten Schritt wird aus dem DOM Baum eine entsprechende

[11] Document Object Model (DOM) definiert eine allgemeine Schnittstelle zur Manipulation von Dokumenten in einer Baumstruktur.
http://www.w3.org/DOM/

EMF Modell Instanz erzeugt. Der so eingelesene Prozess kann dann über die GUI des BPEL Designers bearbeitet werden.
Modifikationen des BPEL Prozesses über die GUI werden direkt in die EMF Modellinstanz übertragen. Änderungen in der EMF Modellinstanz werden automatisch in den DOM Baum weiter propagiert.
Wird ein BPEL Prozess abgespeichert, so wird der DOM Baum serialisiert und als XML Datei abgespeichert. Die EMF Modellinstanz wird zum Speichern nicht benötigt, da EMF Modellinstanz und DOM Baum immer konsistent gehalten werden.
Das Umwandeln von EMF zu DOM wird im BPEL Designer Serialisierung genannt, das Umwandeln von DOM nach EMF heißt dementsprechend Deserialisierung. Damit diese Umwandlungen auch in den SimTech Aktivitäten funktionieren, müssen eigene Serialisierer und Deserialisierer implementiert und beim BPELDesigner registriert werden.

4.2.3. Grafische Darstellung, UIObjectFactory

Neben dem Modell muss auch die grafische Repräsentation des Modells erweitert werden. Dafür bietet der BPEL Designer einen Extension Point *org.eclipse.bpel.ui.uiObjectFactories*, über den sich zusätzliche UIObjectFactories registrieren können. Für die SimTech Aktivitäten wird eine eigene UIObjectFactory implementiert und registriert.
Neben Methoden zur graphischen Repräsentation, wie zum Beispiel *getImage*() oder *getLabel*(), wird die UIObjectFactory auch benutzt, um neue Instanzen von Modellelementen zu erzeugen. An diesem Punkt setzt die Erzeugung der komplexen Inhalte der SimTech Aktivitäten an. Wie in Listing 1 zu sehen ist, wird in der Methode *createInstance*() nicht nur eine entsprechende Instanz des Modelelements zurückgegeben. Die erzeugte Instanz wird an dieser Stelle durch die im Mapping vordefinierten Kindelemente erweitert.

```
@Override
public EObject createInstance() {
    EObject result = super.createInstance();
    if (result instanceof DUNESimulationShutDown) {
        DUNESimulationShutDown duneSimShutDown = (DUNESimulationShutDown) result;

        Scope myScope = BPELFactory.eINSTANCE.createScope();
        Sequence mySequence = BPELFactory.eINSTANCE.createSequence();
        Assign myAssignStopApplication = BPELFactory.eINSTANCE.createAssign();
        Invoke myStopApplication = BPELFactory.eINSTANCE.createInvoke();
        Assign myAssignRemoveSimulationInstance = BPELFactory.eINSTANCE.createAssign();
        Invoke myRemoveSimulationInstance = BPELFactory.eINSTANCE.createInvoke();

        mySequence.getActivities().add(myAssignStopApplication);
        mySequence.getActivities().add(myStopApplication);
        mySequence.getActivities().add(myAssignRemoveSimulationInstance);
        mySequence.getActivities().add(myRemoveSimulationInstance);
        myScope.setActivity(mySequence);

        duneSimShutDown.setActivity(myScope);
    }

    return result;

}
```

Listing 1: Methode "createInstance()"

4.2.4. Adapter

Für jede SimTech Aktivität wird ein *Adapter* implementiert. Der BPEL Designer benötigt diesen Adapter, um auf das Kindelement der SimTech Aktivität zuzugreifen und um den SimTech *EditPart* zu erzeugen. Der EditPart stellt die Verbindung zwischen der Instanz des Modellelements und der grafischen Repräsentation her.
Als weiteres wird eine *SimTechUIAdapterFactory* erstellt, die von der von EMF generierten *ModelAdapterFactory* erbt. Methoden der ModelAdapterFactory werden so überschrieben, dass die für die SimTech Aktivitäten neu implementierten Adapter erzeugt werden können. Die SimTechUIAdapterFactory wird um weitere Adapter erweitert. Der BPEL Designer benötigt von jeder Aktivität einen Adapter für den Zugriff auf die sogenannte *Namespace Map*. Da die von EMF generierten SimTech Aktivitäten einen solchen Adapter nicht besitzen, wird er der SimTechUIAdapterFactory hinzugefügt. Der BPEL Designer bietet dazu eine Standardimplementierung dieses Adapters ohne spezifische Funktionalität an, welche für die SimTech Aktivitäten verwendet wird.
Abschließend wird die neu erstellte SimTechUIAdapterFactory beim BPEL Designer registriert. Die Registrierung erfolgt in der *Activator* Klasse des Plugins. Der Activator kontrolliert den Lebenszyklus des Plugins. Er wird beim Start des Plugins erzeugt.

4.2.5. Erweiterung der Palette

```
@Override
public void contributeItems(PaletteRoot paletteRoot) {
    PaletteCategory category = new PaletteCategory("SimTech Service Catalog");
    category.setCategoryId("SimTechServiceCatalog");

    category.add(new BPELCreationToolEntry(
        "Simulation Start Up",
        "Creates a new SimTech Activity DUNE Simulation Start Up",
        new SimTechUIObjectFactory(ModelPackage.eINSTANCE.getDUNESimulationStartUp())));

    category.add(new BPELCreationToolEntry(
        "Build Grid",
        "Creates a new SimTech Activity DUNE Build Grid",
        new SimTechUIObjectFactory(ModelPackage.eINSTANCE.getDUNEBuildGrid())));

    ...

    paletteRoot.add(category);
}
```

Listing 2: Erweiterung der BPEL Designer Palette

In den vorhergehenden Schritten wurde der BPEL Designer so erweitert, dass er die SimTech Aktivitäten „kennt“ und mit ihnen umgehen kann. Damit der Benutzer diese Aktivitäten bei der Modellierung verwenden kann, wird die Palette des BPEL Designers um entsprechende Einträge erweitert. Der BPEL Designer bietet dafür den Extension Point *org.eclipse.bpel.common.ui.paletteAdditions* an. Über diesen Extension Point wird eine Klasse *SimTechPaletteProvider* registriert, in welcher die Palette programmatisch erweitert wird. Für die SimTech spezifischen Einträge wird zunächst eine Kategorie *SimTech Service Catalog* angelegt. Anschließend wird für jede SimTech Aktivität ein Paletteneintrag erzeugt, mit welchem die entsprechenden Aktivitäten im Modell erzeugt werden können. Die Zuordnung eines Paletteneintrags zu einer bestimmten SimTech Aktivität erfolgt dadurch, dass dem Paletteneintrag eine Instanz der SimTechUIObjectFactory für das entsprechende Modellelement zugeordnet wird. In Listing 2 ist dieses Vorgehen beispielhaft dargestellt.

4.2.6. Ausspezifizieren der SimTech Aktivitäten

In Kapitel 4.2.3 wird beschrieben, wie in der SimTechUIObjectFactory Instanzen der SimTech Aktivitäten erzeugt werden. Dabei werden die vordefinierten Prozessfragmente innerhalb der SimTech Aktivitäten ebenfalls mit angelegt, aber nicht ausspezifiziert.
Zum vollständigen Ausspezifizieren der SimTech Aktivitäten ist der Zugriff auf den umgebenden Prozess notwendig. Bei der Erzeugung aller SimTech Aktivitäten muss es möglich sein, globale Variablen im Prozess anzulegen oder darauf zuzugreifen. Bei der Ausspezifizierung der Invoke Aktivitäten wird Zugriff auf die DUNE WSDL Definitionen benötigt. Diese Definitionen können nur auf Prozessebene importiert werden, der Zugriff ist nur über den Prozess möglich.
In der SimTechUIObjectFactory werden die SimTech Aktivitäten erzeugt, sie sind aber noch keinem Prozess zugeordnet. Daraus folgt, dass das Ausspezifizieren der Aktivitäten an dieser Stelle nicht geschehen kann. Möglich ist das Ausspezifizieren dagegen im Serialisierer der entsprechenden SimTech Aktivität. Der Serialisierer wird im Lebenszyklus einer Aktivität genau einmal aktiviert. In dem Moment, in dem eine Aktivität in ein Prozessmodell eingefügt wird, wird der Serialisierer aufgerufen, um eine DOM Repräsentation der Aktivität zu erzeugen. Ab diesem Zeitpunkt wird der Serialisierer für diese konkrete Aktivität nicht mehr benötigt. Die EMF Modellinstanz und der DOM Baum werden ständig synchron gehalten. Dabei wird der DOM Baum angepasst, aber nicht mehr neu erzeugt. Dieses Verhalten deckt sich mit den Anforderungen an das Ausspezifizieren der SimTech Aktivitäten. Dieses soll genau einmal beim Einfügen in das Prozessmodell geschehen.

```
private void setPredefinedAttributes(Activity activity, Node parentNode, Process process) {
    ...

    //1
    Import wsdlImportWSI = BPELFactory.eINSTANCE.createImport();
    wsdlImportWSI.setImportType(WSDLConstants.WSDL_NAMESPACE_URI);
    wsdlImportWSI.setLocation("WebServiceInterface.wsdl");
    wsdlImportWSI.setNamespace("http://wsi.simtech.de/ws/");
    process.getImports().add(wsdlImportWSI);

    //2
    Message simIDMessageWSI = (Message)BPELUtils.lookup(
        activity,
        new QName("http://wsi.simtech.de/ws/","SimIDMessage"),
        null,
        WSDLUtil.WSDL_MESSAGE);

    //3
    Variable stopApplicationInput = BPELFactory.eINSTANCE.createVariable();
    stopApplicationInput.setName("stopApplicationInput");
    stopApplicationInput.setMessageType(simIDMessageWSI);

    //4
    myScope.getVariables().getChildren().add(stopApplicationInput);

    ...
}
```

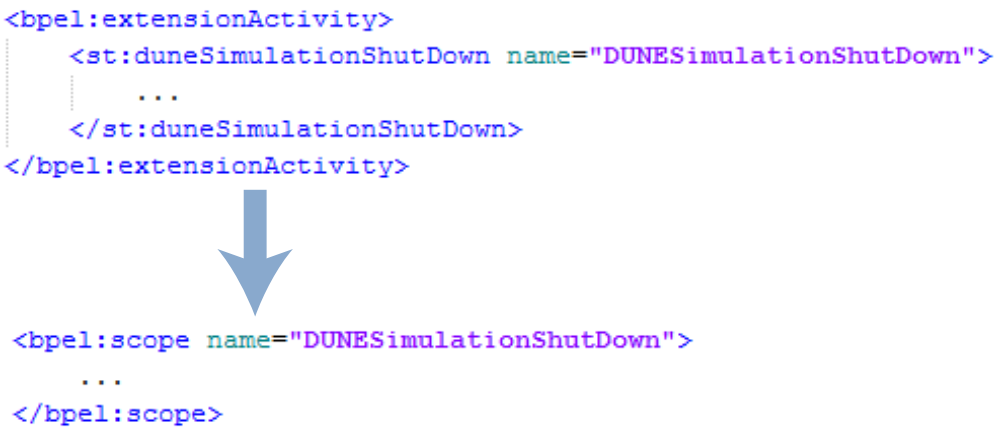
Listing 3: Ausspezifizieren der SimTech Aktivitäten im Serialisierer

In Listing 3 wird an einem kleinen Beispiel gezeigt, wie das programmatische Ausspezifizieren einer SimTech Aktivität abläuft. In Abschnitt 1 wird ein Import einer WSDL Datei erzeugt und dem Prozess hinzugefügt. Instanzen von BPEL Modellelementen werden immer über die BPELFactory erzeugt. In Abschnitt 2 wird aus der soeben importierten WSDL Datei ein Message Datentyp eingelesen. Die Klasse BPELUtils bietet für das Abfragen von Elementen aus WSDL Dateien die Methode lookup() an. In Abschnitt 3 wird eine Variable angelegt. Dieser

Variable wird der aus der WSDL Datei abgefragte Message Datentyp zugewiesen. Im letzten Abschnitt wird die erzeugte Variable dem Scope der aktuellen SimTech Aktivität zugewiesen. Dieses Beispiel verdeutlicht die Notwendigkeit des Zugriffs auf den Prozess.

4.2.7. Transformation in BPEL Prozesse ohne Extension Activities

Die SimTech Aktivitäten werden als BPEL Extension Activities umgesetzt. Die Begründung dieses Vorgehens wurde in Kapitel 4.2 erläutert. Im Folgenden wird beschrieben, wie BPEL Prozesse, die SimTech Aktivitäten enthalten, in äquivalente BPEL Prozesse ohne Extension Activities umgewandelt werden können.



Listing 4: Entfernen von Extension Activity

In Listing 4 wird dargestellt, wie die Umwandlung einer SimTech Aktivität in eine BPEL Aktivität funktioniert. Die Extension Activity und die darin enthaltene SimTech Aktivität werden durch einen Scope ersetzt. Dabei wird das Kindelement der SimTech Aktivität in den Scope übernommen. Sollten in der Extension Activity Sources oder Targets definiert sein, so werden diese ebenfalls in den Scope übernommen. Der neu erzeugte Scope bekommt den Namen der SimTech Aktivität. Damit ist ein klarer Bezug zur ursprünglichen SimTech Aktivität gegeben, die Transformation bleibt für den Benutzer nachvollziehbar.

4.2.8. Zusammenfassung aller Erweiterungen des BPEL Designers

In Abbildung 18 werden noch einmal alle Erweiterungen des BPEL Designers dargestellt, die bei der Umsetzung der SimTech Aktivitäten vorgenommen wurden. Für jede SimTech Aktivität wird ein Serialisierer (EMF2DOM) implementiert. Der Deserialisierer (DOM2EMF) ist für alle SimTech Aktivitäten der Gleiche. Serialisierer und Deserialisierer werden über die BPELExtensionRegistry beim BPEL Designer registriert. Für die SimTech Aktivitäten werden zusätzlich Adapter sowie die SimTechUIAdapterFactory implementiert. Diese wird in der AdapterFactory des BPEL Designers registriert.

Die restlichen Erweiterungen werden über Extension Points des BPEL Designers vorgenommen. Über den Extension Point org.*eclipse.bpel.ui.uiObjectFactories* wird die SimTechUIObjectFactory in den BPEL Designer eingebunden. Die Palette des BPEL Designers wird über den Extension Point *org.eclipse.bpel.common.ui.paletteAditions* um zusätzliche SimTech Einträge erweitert.

Die Modellierung und die Ausspezifizierung des Inhalts der SimTech Aktivitäten erfolgt in zwei Schritten. In der SimTechUIObjectFactory wird beim Erzeugen der SimTech Aktivität zusätzlich

das darin enthaltene Prozessfragment erzeugt. Die Ausspezifizierung erfolgt dann im Serialisierer der entsprechenden SimTech Aktivität.

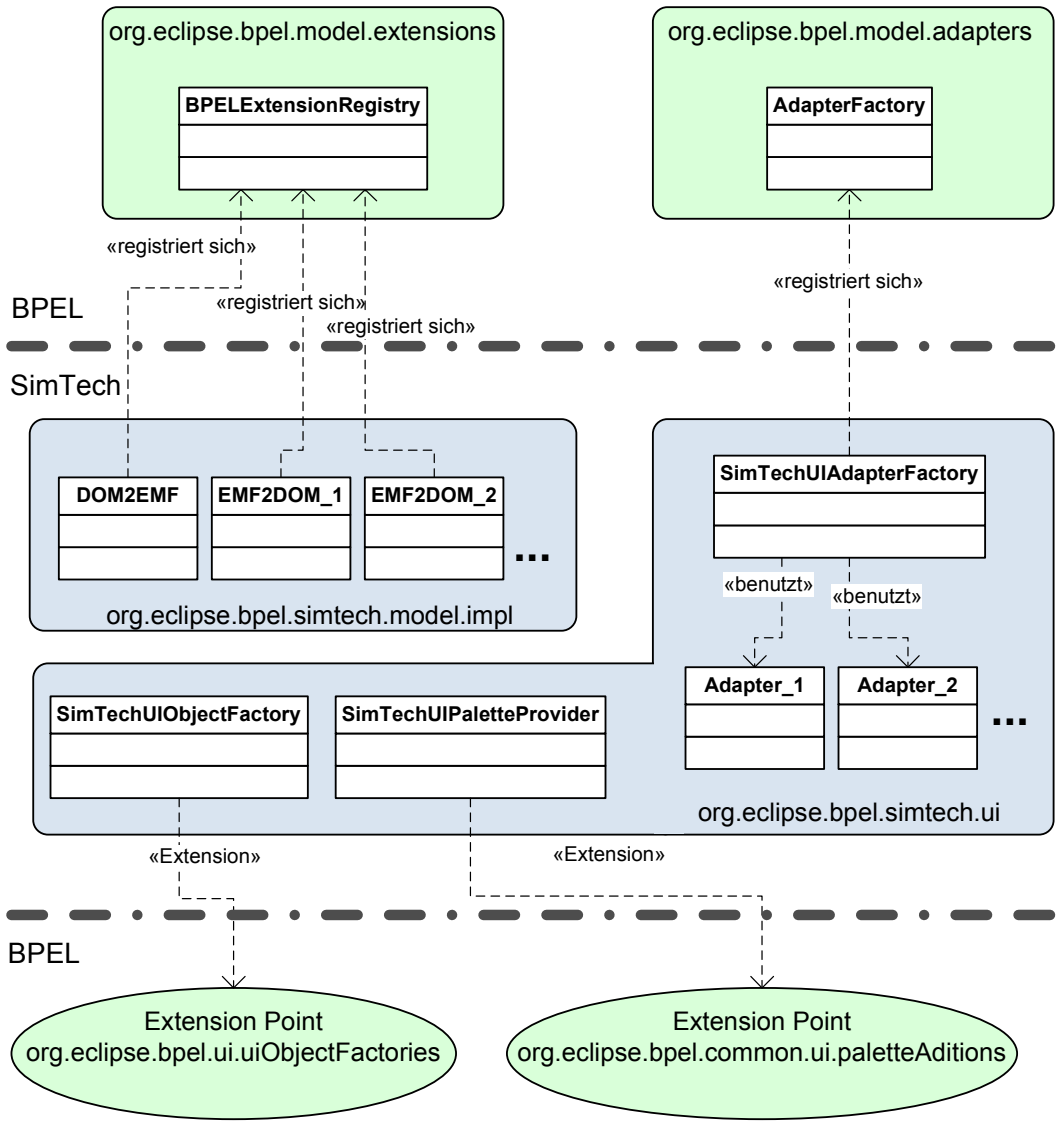


Abbildung 18: Erweiterungen des BPEL Designers

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde zunächst eine Architektur für das SimTech Workflow Tool entwickelt. Das SimTech Workflow Tool wurde auf der Basis von Eclipse aufgebaut und besteht aus den Komponenten Modeler, Service Catalog, Monitoring und Result Display. Als Modeler Komponente wird der Eclipse BPEL Designer benutzt. Für die Komponenten Monitoring und Result Display wurden Extension Points definiert, über welche diese Komponenten zu einem späteren Zeitpunkt in das SimTech Workflow Tool integriert werden können. Für die unterschiedlichen Phasen im Lebenszyklus eines Simulationsexperimentes wurden Eclipse Perspektiven entworfen und implementiert. Diese Perspektiven passen die graphische Oberfläche des Tools an die Anforderungen der entsprechenden Phasen an.
Im zweiten Teil dieser Arbeit wurde die Komponente Service Catalog prototypisch entwickelt. Anhand eines gegebenen Simulationsprozesses wurden SimTech Aktivitäten aufgebaut. Anschließend wurde der BPEL Designer so erweitert, dass er die Verwendung von SimTech Aktivitäten unterstützt. Der SimTech Service Catalog wurde als Erweiterung der BPEL Designer Palette implementiert. Über diese erweiterte Palette können die SimTech Aktivitäten zur Modellierung von Simulationsprozessen benutzt werden.
Das SimTech Workflow Tool befindet sich noch in der Entwicklung. Die Komponente Monitoring wird parallel zu dieser Arbeit entwickelt, ist aber noch nicht in das Tool integriert. Die Komponente Result Display wird in näherer Zukunft entwickelt werden und kann dann, ebenso wie das Monitoring, über die in dieser Arbeit definierten Extension Points in das Tool eingebunden werden.
Die in dieser Arbeit entwickelten SimTech Aktivitäten beschränken sich auf einen Beispielprozess aus dem DUNE Framework. Die Entwicklung von SimTech Aktivitäten für weitere Simulations-Frameworks ist analog zu dem vorgestellten Vorgehen möglich. Zudem wurde allgemein gezeigt, wie SimTech Aktivitäten über die Palette in den Eclipse BPEL Designer eingebunden werden können. Die Implementierung wurde an den entsprechenden Stellen so gestaltet, dass eine spätere Erweiterung problemlos möglich ist.

6 Referenzen

[BPEL07]	Organization for the Advancement of Structured Information Standards (OASIS): Web Services Business Process Execution Language Version 2.0, OASIS Standard, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html, April 2007
[Hau09]	Haupt, Florian: Monitoring von workflowbasierten DUNE Simulationen. Institut für Architektur von Anwendungssystemen der Universität Stuttgart, Studienarbeit, 2009
[HIM06]	Höhensteiger, Benjamin; Illiger, Michael; Moser, Simon: Extending the Eclipse BPEL Designer with custom Activities. http://www.eclipse.org/bpel/users.php, November 2006
[IBM06]	International Business Machines Corp. (IBM): Eclipse Platform Technical Overview. http://eclipse.org/articles/, April 2006
[LR00]	Leymann, F.; Roller, D.: Production Workflow. Concepts and Techniques. Prentice Hall Inc., 2000
[MDG04]	Moore, Bill; Dean, David; Gerber, Anna; Wagenknecht, Gunnar; Vanderheyden, Philippe: Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework. http://www.redbooks.ibm.com/redbooks, Februar 2004
[Rut09]	Rutschmann, Jens: Generisches Webservice-Interface um Simulationsanwendungen in BPEL-Prozesse einzubinden. Institut für Architektur von Anwendungssystemen der Universität Stuttgart, Diplomarbeit, 2009
[WSDL07]	World Wide Web Consortium (W3C): Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. http://www.w3.org/TR/wsdl20/, Juni 2007

Alle in dieser Arbeit aufgeführten Weblinks wurden das letzte Mal am 31.08.2009 geprüft.

Erklärung

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

Stuttgart, 31.08.2009