Institut für Architektur von Anwendungssystemen

Universität Stuttgart
Universitätsstraße 38
D - 70569 Stuttgart

Studienarbeit Nr. 2218

Monitoring von workflowbasierten DUNE Simulationen

Florian Haupt

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

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BPEL	Business Process Execution Language
DUNE	Distributed and Unified Numerics Environment
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IAAS	Institut für Architektur von Anwendungssystemen
JMS	Java Message Service
MOM	Message Oriented Middleware
OASIS	Organization for the Advancement of Structured Information Standards
ODE	Orchestration Director Engine
POI	Point of Interest
SOAP	SOAP[1]
SWT	Standard Widget Toolkit
WS	Web Service
WSDL	Web Service Description Language

[1] SOAP stand ursprünglich für „Simple Object Access Protocol“. Seit SOAP Version 1.2 wird SOAP nicht mehr als Akronym sondern als Name verwendet.
http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

1 Einleitung

Der Ursprung der *Workflow Technologie* liegt in der Automatisierung von Geschäftsprozessen. In diesem Bereich werden die entwickelten Technologien auch heute noch überwiegend eingesetzt. Workflow Technologien sind jedoch nicht auf diesen einen Anwendungsbereich beschränkt. Ein Beispiel für ein weiteres Einsatzgebiet ist die Wissenschaft.
Neben Theorie und Experiment hat sich der Bereich der Simulation und der Berechnung als drittes Standbein der Forschung etabliert [TDG+07]. Die Durchführung von computergestützten Berechnungen und Simulationen lässt sich ebenfalls als Prozess beschreiben. Daten müssen bewegt und bearbeitet werden, die Ausführung von Programmen oder Programmfunktionen muss koordiniert werden. In vielen Fällen handelt es sich dabei um Prozesse, welche in gleicher oder ähnlicher Form regelmäßig wiederholt werden. Diese Ausgangsbedingungen haben dazu geführt, dass Technologien aus dem Bereich der Automatisierung von Geschäftsprozessen in die Wissenschaftswelt übertragen wurden. Analog zum Begriff des *Business Workflow* spricht man in diesem Anwendungskontext auch von *Scientific Workflow*.
Im Rahmen eines Forschungsprojektes des Stuttgarter Exzellenzclusters *Simulation Technology (SimTech*[2]*)* werden bestehende Simulationssysteme so adaptiert, dass Simulationsexperimente mit Hilfe von Workflow Technologien modelliert, ausgeführt, überwacht und ausgewertet werden können. Ein Prototyp eines entsprechenden Softwaresystems befindet sich momentan in der Entwicklung. Das Ziel dieser Arbeit ist der Entwurf und die prototypische Implementierung einer Monitoring Komponente für diesen Prototypen.
Eine wesentliche Anforderung an die Monitoring Komponente rührt daher, dass die ausgeführten Prozesse aus Sicht des Benutzers nicht dem entsprechen, was er mit der Modellierungskomponente modelliert hat. Die Modellierungskomponente stellt dem Benutzer spezielle Aktivitäten zur Verfügung, welche komplexe Prozessfragmente repräsentieren. Der Benutzer sieht diese Aktivitäten als atomar an. Sie stellen jedoch nicht eine einzelne Aktivität im Prozessmodell dar, sondern im allgemeinen Fall eine komplexe Menge von Aktivitäten. Die Ausführung dieser Prozessfragmente soll durch die Monitoring Komponente als die Ausführung einer atomaren Aktivität dargestellt werden.

[2] http://www.simtech.uni-stuttgart.de/

2 Ausgangssituation und Aufgabestellung

In den folgenden Abschnitten wird das Umfeld, in das sich diese Arbeit einordnet, näher vorgestellt. Anschließend wird die Aufgabenstellung, die dieser Arbeit zugrunde liegt, nochmals präzisiert dargestellt.

2.1. Der Exzellenzcluster SimTech

Der Cluster *Simulation Technology (SimTech)* der Universität Stuttgart bündelt die Forschungsarbeit von Instituten unterschiedlicher Fachbereiche im Bereich der Simulationstechnologien. Der Cluster wurde im Rahmen der Exzellenzinitiative des Bundes und der Länder mit dem Exzellenzsiegel ausgezeichnet.
Das *Institut für Architektur von Anwendungssystemen (IAAS*[3]*)* der Universität Stuttgart ist an der SimTech Projektgruppe „Integriertes Datenmanagement, Workflow und interaktive Visualisierung für eine integrierte Systemwissenschaft“ beteiligt. Dort soll ein *Workflow Management System (WfMS)* zum Modellieren und Ausführen von Simulations-Workflows entwickelt werden[4]. Im Rahmen des Projektes „Modellierung von Simulations-Workflows“ wird ein Prototyp eines Tools zur Modellierung von Simulations-Workflows entwickelt.
Der Prototyp des Modellierungstools basiert auf dem *Eclipse BPEL Designer*[5]. Es sollen Simulationen erstellt werden können, in denen die *Distributed and Unified Numerics Environment (DUNE*[6]*)* genutzt wird. Die so erstellten Prozesse sollen auf der *Apache Orchestration Director Engine (ODE*[7]*)* ausgeführt werden.

2.2. Verwandte Arbeiten

In der Arbeit von Karolina Vukojević [Vuk09] wird der Eclipse BPEL Designer um sogenannte *SimTech Aktivitäten* erweitert. Diese Aktivitäten repräsentieren Fragmente von BPEL Prozessen. Eine SimTech Aktivität kann also mehrere BPEL Aktivitäten enthalten. SimTech Aktivitäten können als virtuelle Aktivitäten verstanden werden. Sie stellen dem Benutzer komplexe Prozessfragmente als eine einzelne Aktivität dar. Der derart modifizierte Eclipse BPEL Designer erzeugt jedoch auch weiterhin reinen BPEL Code, welcher dann auf der ODE ausgeführt wird. Der um SimTech Aktivitäten erweiterte Eclipse BPEL Designer wird im Folgenden als *SimTech BPEL Designer* bezeichnet.
In der Arbeit von Thomas Steinmetz [Ste08] wurde ein Eventmodell für BPEL 2.0 entwickelt. Es beschreibt, welche Ereignisse (*Events*) im Lebenszyklus von BPEL Prozessinstanzen sowie BPEL Aktivitäten auftreten können. Apache ODE wurde in dieser Arbeit so erweitert, dass bei der Ausführung von BPEL Prozessen entsprechende Events erzeugt und über *JMS* nach außen kommuniziert werden. Zusätzlich wurde ODE um die Fähigkeit erweitert, BPEL Prozesse unterbrechen zu können. Diese Funktionalität spielt in der vorliegenden Arbeit keine Rolle.

[3] http://www.iaas.uni-stuttgart.de/
[4] http://www.iaas.uni-stuttgart.de/forschung/projects/simtech/
[5] http://www.eclipse.org/bpel/
[6] http://www.dune-project.org/
[7] http://ode.apache.org/

2.3. Aufgabenstellung

Die vorliegende Arbeit behandelt das Überwachen (*Monitoring*) der Ausführung von DUNE Simulationen, welche mit dem SimTech BPEL Designer erstellt und auf der ODE ausgeführt werden.
Die Ausführung einer einzelnen SimTech Aktivität entspricht der Ausführung einer oder mehrerer BPEL Aktivitäten. Trotzdem soll dem Benutzer lediglich die Ausführung der virtuellen SimTech Aktivität dargestellt werden. Die Abstraktion, welche durch die Einführung der SimTech Aktivitäten bei der Modellierung geschaffen wurde, soll auch während des Monitorings aufrecht erhalten werden.
In dieser Arbeit soll zunächst überprüft werden, ob das von Thomas Steinmetz entwickelte BPEL Eventmodell den Anforderungen des Monitorings von SimTech Simulationen genügt. Ist dies nicht der Fall, so soll ein eigenes Eventmodell für SimTech Aktivitäten entwickelt und implementiert werden. Anschließend soll eine Anwendung zum einfachen Monitoring von SimTech Simulationen entwickelt und in den SimTech BPEL Designer integriert werden.

3 Verwendete Technologien

3.1. Web Services

Als Dienst (*Service*) bezeichnet man Funktionen (Programme), welche über eine eindeutige Netzwerkadresse zur Verfügung gestellt werden. Dienste haben eine „always on“ Semantik, d.h. sie sind immer verfügbar.
Dienste, die auf Web Technologien wie beispielsweise *HTTP* oder *XML* basieren, bezeichnet man als *Web Services*. Die Menge aller Standards und Technologien im Zusammenhang mit Web Services bezeichnet man gesammelt auch als *Web Service Technologie* [WCL+05].
Die *Web Service Description Language (WSDL)* ist ein vom *World Wide Web Consortium (W3C)* veröffentlichter Standard zur Beschreibung von Web Services. Er definiert ein Modell sowie eine XML Syntax, um die abstrakte Funktionalität sowie den Zugriff auf Web Services einheitlich zu beschreiben [WSDL].

3.2. Workflow Technologie

Workflow Technologie behandelt die Abbildung von realen Geschäftsprozessen auf IT Umgebungen. Geschäftsprozesse können mit Hilfe von Prozessmodellen beschrieben werden. Sie spezifizieren den genauen Ablauf eines Geschäftsprozesses als Folge von Geschäftsfunktionen.
Wird der Ablauf eines Geschäftsprozesses von einer IT Umgebung kontrolliert und bestehen die Geschäftsfunktionen aus Programmen, dann spricht man anstatt von Prozessmodellen und Prozessen von *Workflowmodellen* und *Workflows* [LR00].

3.2.1. BPEL

Die *Web Service Business Process Execution Language* (*WS-BPEL*, im Folgenden nur *BPEL*) ist eine XML basierte Sprache zur Beschreibung von Geschäftsprozessen. Geschäftsfunktionen sind in BPEL Prozessen als Web Services implementiert. Ein BPEL Prozess kann selbst wieder als Web Service zur Verfügung gestellt werden, weswegen BPEL auch als rekursive Aggregationssprache für Web Services bezeichnet wird. BPEL ist ein von *OASIS*[8] veröffentlichter Standard, die aktuelle Version ist *WS-BPEL 2.0*.

3.2.2. Eclipse BPEL Designer

Der Eclipse BPEL Designer ist ein graphisches Modellierungstool für BPEL Prozessmodelle. Die Software wird als Eclipse Projekt unter Open Source Lizenz entwickelt und als Eclipse Plugin zur Verfügung gestellt. Der Eclipse BPEL Designer unterstützt das Deployment von BPEL Prozessen auf der Apache ODE Workflowengine (siehe Abschnitt 3.2.3).
Eclipse ist ein modular aufgebautes Tool, welches ursprünglich als Java IDE entwickelt wurde. Der modulare Aufbau erlaubt eine nahezu beliebige Erweiterung von Eclipse durch sogenannte *Plugins*.

[8] http://www.oasis-open.org/

3.2.3. Apache ODE

Die *Apache Orchestration Director Engine (ODE)* ist eine unter Open Source Lizenz veröffentlichte Software, die in der Lage ist, BPEL Prozesse auszuführen. Man bezeichnet sie deswegen auch als *BPEL Engine* oder *Workflow Engine*.

3.3. Messaging

Als *Messaging* bezeichnet man Kommunikation, die auf dem Versand von Nachrichten (*Messages*) basiert. Neben der direkten Kommunikation zwischen zwei Kommunikationspartnern, dem in Abbildung 1 dargestellten sogenannten *Message Passing*, gibt es weitere Konzepte zur indirekten Kommunikation [HW03].

Abbildung 1: Message Passing

Beim in Abbildung 2 gezeigten *Message Queueing* werden Nachrichten nicht direkt an den Kommunikationspartner, sondern an Warteschlangen, sogenannte *Queues*, geschickt. Der Kommunikationspartner holt sich die gesendeten Nachrichten aus der Queue ab. Beim Message Queueing handelt es sich um eine 1:1 Kommunikation zwischen zwei Partnern. Das Konzept der Queue erlaubt dabei eine räumliche und zeitliche Entkopplung der Kommunikation.

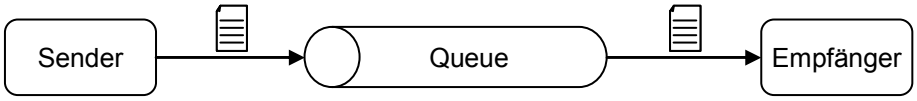


Abbildung 2: Queueing

Eine weitergehende Entkopplung von Sender und Empfänger kann durch *Publish & Subscribe (PubSub)* erreicht werden. Bei einer PubSub Kommunikation veröffentlicht der Sender seine Nachrichten auf einem sogenannten *Topic*. Empfänger von Nachrichten können sich bei Topics für den Empfang von Nachrichten registrieren. Wird eine Nachricht auf einem Topic veröffentlicht, so wird diese an alle Empfänger weitergeleitet, die sich bei diesem Topic registriert haben. Über PubSub kann eine n:m Kommunikation realisiert werden. Beliebige Sender können auf einem Topic publizieren, beliebige Empfänger können sich bei diesem Topic registrieren. In Abbildung 3 ist diese Art der Kommunikation schematisch dargestellt.

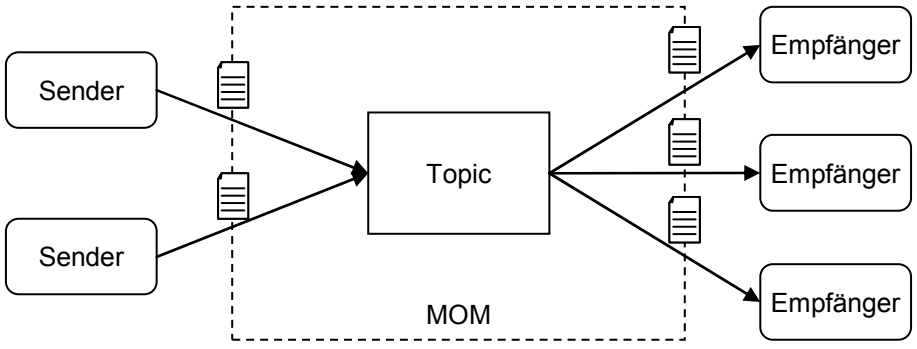


Abbildung 3: Publish & Subscribe

Softwaresysteme, welche die beschriebenen Kommunikationsarten zur Verfügung stellen, bezeichnet man als *Message Oriented Middleware (MOM)*. *Java Messaging Service (JMS*[9]*)* ist eine in Java implementierte Schnittstelle für den einheitlichen Zugriff auf MOM Systeme. *Apache Active MQ*[10] ist eine unter Open Source Lizenz veröffentlichte MOM Implementierung, welche über JMS verwendet werden kann.

[9] http://www.jcp.org/en/jsr/detail?id=914
[10] http://activemq.apache.org/

4 BPEL 2.0 Eventmodell

Dieses Kapitel gibt einen Überblick über die Arbeit von Thomas Steinmetz [Ste08]. Es wird das im Rahmen dieser Arbeit entwickelte Eventmodell für BPEL 2.0 vorgestellt, soweit es für die vorliegende Arbeit relevant ist.
Das Eventmodell für BPEL 2.0 wird durch mehrere einzelne Zustandsübergangsdiagramme beschrieben. In ihnen wird der Lebenszyklus von BPEL Prozessen und BPEL Aktivitäten beschrieben. BPEL Prozesse und Aktivitäten können verschiedene Zustände einnehmen. Für alle erlaubten Übergänge zwischen diesen Zuständen wird klar definiert, unter welchen Bedingungen sie aktiv werden, und welche Events bei einem solchen Übergang ausgelöst werden.

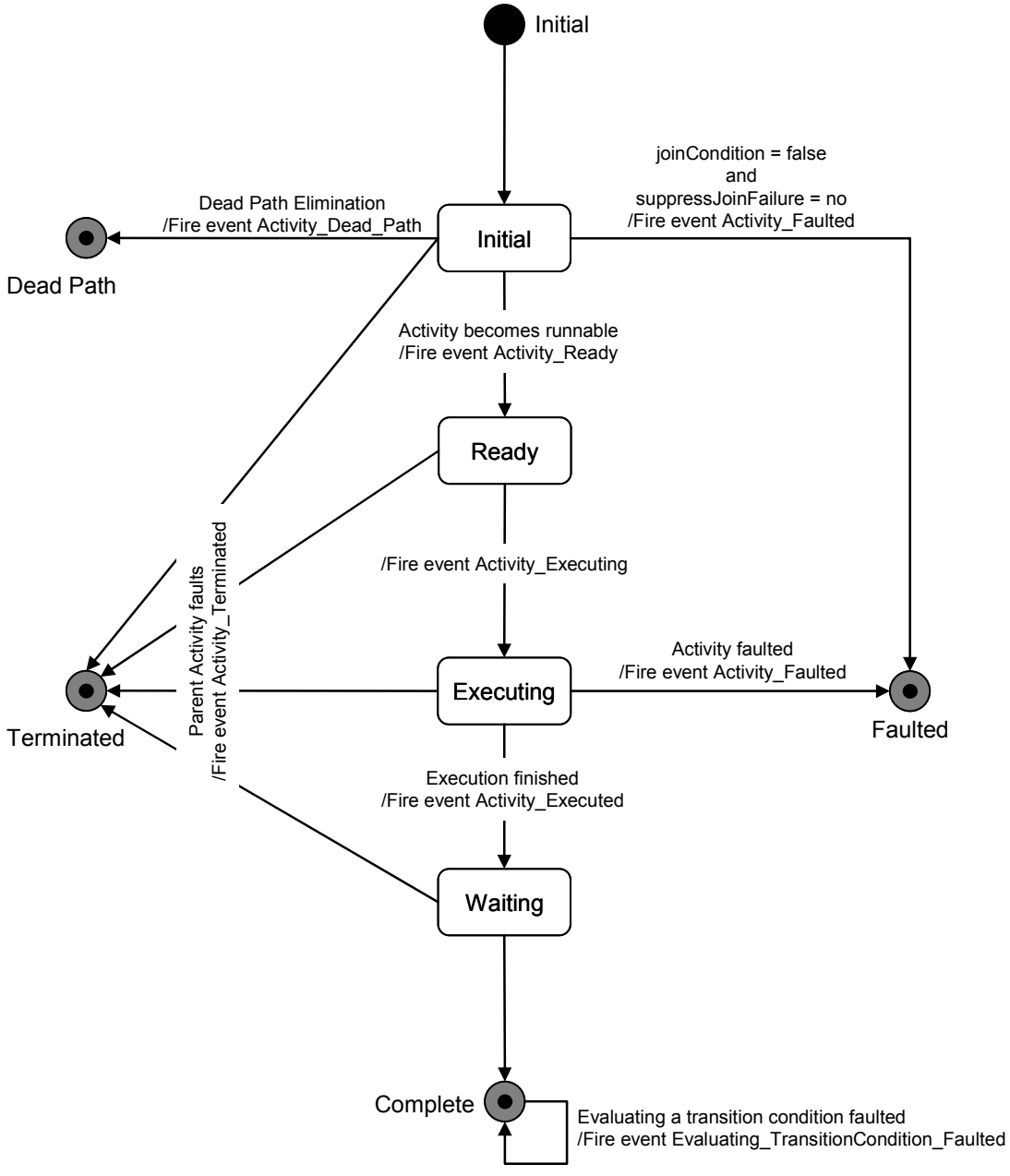


Abbildung 4: Allgemeines Eventmodell für BPEL 2.0 Aktivitäten nach [Ste08]

In Abbildung 4 ist das Zustandsübergangsdiagramm für allgemeine BPEL Aktivitäten dargestellt. Es enthält einen *Startzustand*, mehrere *Endzustände* sowie *reguläre Zustände*. Start- und Endzustände sind als einfarbig bzw. zweifarbig gefüllte Kreise dargestellt, reguläre Zustände als gerundete Rechtecke. Die Pfeile zwischen den Zuständen repräsentieren *Zustandsübergänge*. Jeder Zustandsübergang enthält Angaben über *Bedingungen* und *Aktionen*. Die Bedingungen beschreiben, wann der Zustandsübergang aktiviert wird. Die angegebenen Aktionen werden bei der Aktivierung des Zustandsübergangs ausgeführt. Aktionen werden in Abbildung 4 durch ein vorgestelltes „/“ gekennzeichnet. Die Zustandsübergangsdiagramme, durch welche das BPEL 2.0 Eventmodell beschrieben wird, enthalten als einzigen Aktionstyp das Auslösen von Events.
Das Eventmodell für BPEL 2.0 wurde in [Ste08] mit der zusätzlichen Zielsetzung entworfen, den Ablauf eines Prozesses von außen steuern zu können. Um dies zu ermöglichen, können einige Events blockierend sein. Ist ein Event blockierend, so wird die Ausführung der Prozessinstanz bei dessen Auslösung angehalten. Erst durch ein weiteres, von außen ausgelöstes Event, wird die Ausführung der Prozessinstanz fortgesetzt. Der Zustand *Waiting* hat den Zweck, genau dieses Verhalten zu unterstützen. Er wurde eingeführt, um die Unterbrechung des Prozesses zwischen erfolgreicher Ausführung und dem Übergang in den Endzustand *Complete* zu ermöglichen. Da blockierende Events in dieser Arbeit nicht betrachtet werden und in Abbildung 4 deswegen auch nicht dargestellt sind, erscheinen der Zustand *Waiting* und der bedingungslose Übergang in den Zustand *Complete* zunächst überflüssig. Aus den genannten Gründen wird er im Eventmodell für BPEL 2.0 jedoch benötigt.
Neben dem dargestellten allgemeinen Eventmodell für Aktivitäten gibt es noch jeweils ein separates Eventmodell für *Scopes*[11] und eines für *Schleifen*. Im Fall von Scopes wird das allgemeine Eventmodell für Aktivitäten um Zustände für *Event-*, *Fault- Compensation-* und *Termination Handling* erweitert. Das Eventmodell für Schleifen enthält zusätzlich zum allgemeinen Eventmodell für Aktivitäten zusätzliche Zustände für die Evaluierung der Schleifenbedingung und das Beenden einer Iteration.
Neben dem Eventmodell für BPEL Aktivitäten wurden in der Arbeit von Thomas Steinmetz noch zwei weitere Eventmodelle definiert. Das Eventmodell für BPEL Prozesse beschreibt den Lebenszyklus eines Prozesses. Im Normalfall durchläuft ein Prozess die Zustände *Deployed*, *Active*, *Instantiated*, *Running* und *Completed*. Das Eventmodell für *Links* beschreibt den Lebenszyklus von Links. Der Wert des Status eines Links ist zunächst unbestimmt, wird dann einmalig ausgewertet und bleibt ab diesem Zeitpunkt unverändert bestehen.

[11] Beachte: Ein Invoke verhält sich ebenfalls wie ein Scope.
http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html#_Toc164738505

5 SimTech Aktivitäten

Eine SimTech Aktivität repräsentiert eine oder mehrere BPEL Aktivitäten. Wie in Kapitel 4 vorgestellt, gibt es für eine beliebige BPEL Aktivität drei mögliche Eventmodelle, die zutreffen können.
Im Folgenden wird betrachtet, wie BPEL Aktivitäten und BPEL Events auf SimTech Aktivitäten und SimTech Events abgebildet werden können. Dazu wird zunächst eine Fallunterscheidung vorgenommen, um basierend auf den einzelnen Fallbetrachtungen anschließend eine allgemeine Lösung zu entwickeln.

5.1. Fallbetrachtung 1
Eine SimTech Aktivität repräsentiert genau eine BPEL Aktivität

Betrachtet man den Fall, dass eine SimTech Aktivität genau eine BPEL Aktivität repräsentiert, bieten sich zwei gegensätzliche Vorgehensweisen zum Aufbau eines Eventmodells für solche SimTech Aktivitäten an.
Will man im SimTech Eventmodell den gesamten Informationsgehalt der BPEL Eventmodelle erhalten, so kann das SimTech Eventmodell als die Vereinigung aller drei BPEL Eventmodelle für Aktivitäten entworfen werden. Dies ist möglich, da sich die drei Eventmodelle strukturell nicht wesentlich unterscheiden. Eine SimTech Aktivität hätte also ein Eventmodell, nach dem sie sich wie eine allgemeine Aktivität, wie ein Scope oder wie eine Schleife verhalten kann.
Neben dem Vorteil der Erhaltung aller Informationen hat dieser Ansatz aber auch Nachteile. Das so entworfene Eventmodell ist, da es mehrere Eventmodelle in einem zusammen fasst, relativ komplex. Dem Benutzer gegenüber besteht die Gefahr, dass das Monitoring, basierend auf diesem Eventmodell, inkonsistent und verwirrend erscheint. Es ist nicht klar, wann eine SimTech welche Zustände einnehmen kann. Das mögliche Verhalten der SimTech Aktivität hängt von der konkreten BPEL Aktivität ab, welche sie repräsentiert. Aber genau diese Informationen wird durch die SimTech Aktivität vor dem Benutzer versteckt. Es ist nicht ersichtlich oder verständlich, warum sich manche SimTech Aktivitäten wie ein Scope verhalten, andere dagegen wie eine Schleife.
Alternativ kann man das Eventmodell für SimTech Aktivitäten auch als den kleinsten gemeinsamen Nenner aller drei Eventmodelle für Aktivitäten aufbauen. In diesem Fall würde das Eventmodell für SimTech Aktivitäten dem allgemeinen Eventmodel für BPEL Aktivitäten entsprechen.
Ein solches Eventmodell wäre einfach, übersichtlich und im Monitoring klar nachvollziehbar. Alle SimTech Aktivitäten würden ein ähnliches Verhalten zeigen, unabhängig davon, welche konkrete BPEL Aktivität sie repräsentieren.
Der Nachteil dieses Ansatzes ist der Informationsverlust durch die Reduktion der Eventmodelle auf den kleinsten gemeinsamen Nenner. Repräsentiert die SimTech Aktivität beispielsweise einen *Scope*, so würden bei dieser Lösung Informationen über *Fault*-, *Termination*-, *Compensation*- und *Event Handling* verloren gehen, da das allgemeine Eventmodell für Aktivitäten keinerlei Zustände oder Events dafür definiert.

5.2. Fallbetrachtung 2
Eine SimTech Aktivität repräsentiert mehrere BPEL Aktivitäten

Im Allgemeinen Fall wird eine SimTech Aktivität nicht nur eine, sondern mehrere BPEL Aktivitäten repräsentieren und nach außen hin als eine einzelne Aktivität darstellen. Die in Abbildung 5 dargestellte SimTech Aktivität enthält als einfaches Beispiel die in einer Sequenz angeordneten BPEL

Aktivitäten A1, A2 und A3. Im Folgenden wird angenommen, dass die SimTech Aktivität dem Eventmodell für allgemeine BPEL Aktivitäten folgen soll.
Das Verhalten der SimTech Aktivität wird in verschiedenen Phasen ihres Lebenszyklus von verschiedenen der in ihr enthaltenen BPEL Aktivitäten bestimmt. Der Start der SimTech Aktivität wird durch die Aktivität A1 bestimmt. Sobald die Aktivität A1 die Zustände *Ready* oder *Executing* annimmt, wird auch die SimTech Aktivität in diese Zustände wechseln. Verlässt die Aktivität A1 den Zustand *Executing*, so hat dies keinen Einfluss auf den Zustand der SimTech Aktivität, sie befindet sich weiterhin im Zustand *Executing*. Erst das, erfolgreiche oder nicht erfolgreiche, Beenden der Aktivität A3 bedeutet gleichzeitig auch das Beenden der SimTech Aktivität. Die SimTech Aktivität geht in den Zustand *Completed*, wenn die Aktivität A3 in den Zustand *Completed* wechselt.

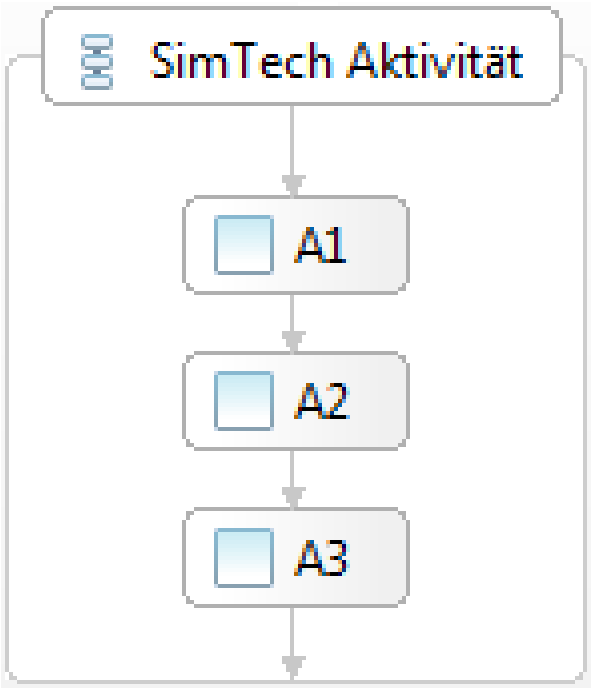


Abbildung 5: SimTech Aktivität mit enthaltenen BPEL Aktivitäten

Alle weiteren Ereignisse, die zwischen dem Start der Aktivität A1 und dem Beenden der Aktivität A3 auftreten, ändern nichts am Zustand der SimTech Aktivität. Sie befindet sich den gesamten Zeitraum über im Zustand *Executing*.
Diese Modellierung des SimTech Eventmodells beachtet keine Events, die während der Ausführung der SimTech Aktivität (also nicht beim Start oder Ende) auftreten. Dies folgt zunächst einmal dem eigentlichen Zweck der SimTech Aktivität. Sie soll die Komplexität der Aktivitäten, welche durch eine SimTech Aktivität verborgen werden, vor dem Benutzer verstecken. Es tritt ein Informationsverlust auf, der aber beabsichtig und gewollt ist.

6 Das SimTech Eventmodell

Im vorherigen Kapitel wurde aufgezeigt, wie BPEL und SimTech Aktivitäten zusammenhängen. Basierend auf diesen Betrachtungen wurde dargelegt, wie Eventmodelle für SimTech Aktivitäten aufgebaut werden können. Ebenso wurden die Vor- und Nachteile der unterschiedlichen Vorgehensweisen genannt.
Im Folgenden wird das neu entwickelte Eventmodell für SimTech Aktivitäten vorgestellt. Der Entwurf orientiert sich dabei an der Grundidee, die dem Konzept der SimTech Aktivitäten zurunde liegt. Die Komplexität, die hinter einer SimTech steht, soll vor dem Benutzer verborgen werden.
Basis des Eventmodells ist das Eventmodell für allgemeine BPEL Aktivitäten. Aufbauend auf diesem Modell wurden zusätzliche Endzustände eingeführt. Diese zusätzlichen Zustände sollen dem Benutzer ein genaueres Feedback über den Zustand der SimTech Aktivität geben.
Befindet sich eine Aktivität nach dem BPEL 2.0 Eventmodell im Zustand *Terminated*, so ist der Informationsgehalt dieser Meldung relativ niedrig. In Tabelle 1 sind die möglichen Übergänge in den Zustand *Terminated* dargestellt. Zusätzlich wird die Semantik des Zustands, abhängig vom erfolgten Übergang in den Zustand dargestellt.

Zustand	Event	Folgezustand	Semantik
Ready	ActivityTerminated	Terminated	Die Aktivität wurde vor Beginn ihrer Ausführung abgebrochen. Die Ausführung kann ohne Nebeneffekte wiederholt werden.
Executing	ActivityTerminated	Terminated	Die Aktivität wurde während der Ausführung abgebrochen. Der Zustand des Prozesses ist unklar.
Waiting	ActivityTerminated	Terminated	Die Aktivität wurde nach erfolgreicher Ausführung abgebrochen. Der Prozess kann ohne Nebeneffekte fortgesetzt werden.

Tabelle 1: Mögliche Zustandsübergänge in den Zustand *Terminated*

Die Semantik des Zustands *Terminated* erschließt sich also nicht nur aus dem Zustand selbst, sie hängt zusätzlich vom vorherigen Zustand ab. Im zu entwickelnden Monitoring Tool soll dem Benutzer immer der aktuelle Zustand aller Aktivitäten dargestellt werden, nicht jedoch deren gesamte Ausführungshistorie. Die neu eingeführten Endzustände erhöhen damit die Aussagekraft des Monitorings.

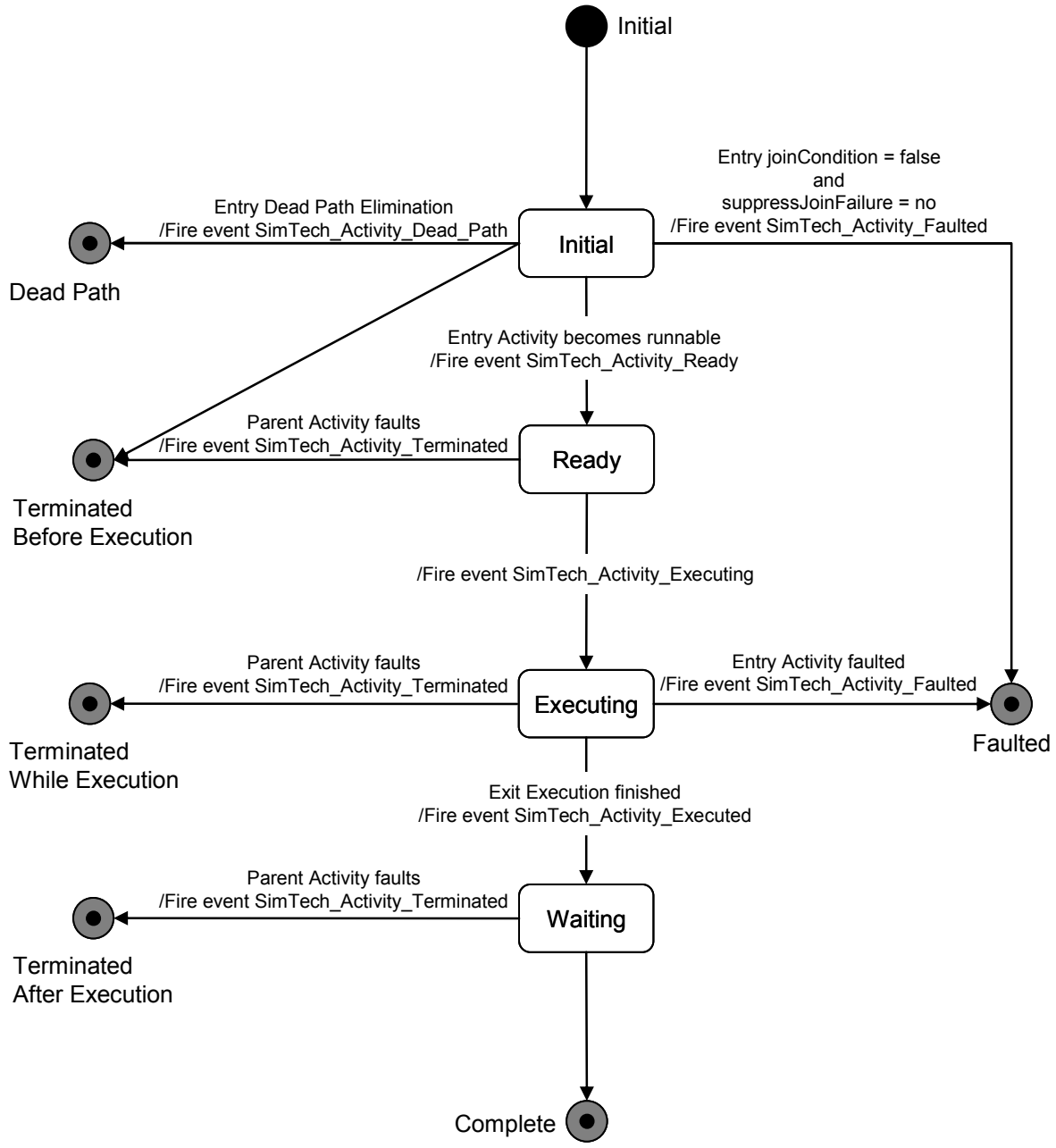


Abbildung 6: Eventmodell für SimTech Aktivitäten

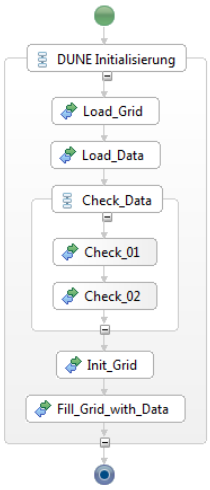
In Abbildung 6 ist das neu entwickelte Eventmodell für SimTech Aktivitäten in Form eines Zustandsübergangsdiagramms dargestellt. Abgesehen von den zusätzlichen Endzuständen entspricht es dem Eventmodell für allgemeine BPEL Aktivitäten. Die bei Aktivierung der Zustandsübergänge ausgelösten Events bilden, abgesehen von Umbenennungen, eine Untermenge der BPEL 2.0 Events.

7 Implementierung des Eventmodells für SimTech Aktivitäten

Der Ausgangspunkt für die Implementierung des Eventmodells ist die gemäß [Ste08] erweiterte ODE. Das Ziel der im Folgenden beschrieben Implementierung ist es, dass bei der Ausführung von BPEL Prozessen auf der ODE nicht nur BPEL Events, sondern zusätzlich auch SimTech Events produziert werden.
Zunächst wird gezeigt, in welcher Form SimTech Aktivitäten definiert werden. Diese in Abschnitt 7.1 vorgestellte Definition ist die Basis für die Abbildung von BPEL Events auf SimTech Events. Um das Eventmodell implementieren zu können, muss die vorhandene Version der ODE erweitert werden. Die Anpassung der ODE wird in Abschnitt 7.2 beschrieben. Der Aufbau und die Funktionsweise der Komponente, welche die eigentliche Abbildung von BPEL Events auf SimTech Events realisiert, werden dann in Abschnitt 7.3 gezeigt.

7.1. Definition der SimTech Aktivitäten

Eine SimTech Aktivität ist, wie bereits erwähnt, im wesentlichen eine Abbildung mehrerer BPEL Aktivitäten auf eine virtuelle SimTech Aktivität. Diese Abbildung muss definiert sein, um BPEL Events auf SimTech Events abbilden zu können.
Als Voraussetzung ist gegeben, dass die Menge aller SimTech Aktivitäten dynamisch ist. Es soll jederzeit möglich sein, weitere SimTech Aktivitäten und ihre Abbildung auf BPEL Aktivitäten zu definieren. Zusätzlich gilt, dass die Definition von SimTech Aktivitäten unabhängig von konkreten Prozessmodellen ist. Die SimTech Aktivitäten werden dem Benutzer als allgemeine Modellierungskomponenten zur Verfügung gestellt. Er kann sie in beliebigen Prozessmodellen frei einsetzen.
In Kapitel 5.2 wurde gezeigt, dass die in einer SimTech Aktivität enthaltenen BPEL Aktivitäten unterschiedlichen Einfluss auf den Lebenszyklus der SimTech Aktivität haben. BPEL Aktivitäten können den Beginn oder das Ende einer SimTech Aktivität definieren, oder sie befinden sich innerhalb einer SimTech Aktivität. Die erstgenannten Aktivitäten beeinflussen den Zustand der SimTech Aktivität, alle weiteren Aktivitäten haben darauf keinen Einfluss.
In Abbildung 7 ist dargestellt, wie die Definition einer SimTech Aktivität in XML Syntax dargestellt wird. Innerhalb eines Mappings können mehrere SimTech Aktivitäten definiert werden. Zu jeder SimTech Aktivität wird zunächst angegeben, welche BPEL Aktivitäten in ihr enthalten sind. Anschließend werden diejenigen BPEL Aktivitäten spezifiziert, die den Beginn oder das Ende der SimTech Aktivität definieren.
Das in Abbildung 7 dargestellte Beispiel zeigt zudem, dass nicht alle in der SimTech Aktivität enthaltenen BPEL Aktivitäten in der Definition der SimTech Aktivität angegeben werden müssen. Es ist ausreichend, nur die BPEL Aktivitäten anzugeben, die direkte Kinder der SimTech Aktivität sind. Im vorgestellten Beispiel ist die BPEL Aktivität *Check_Data* als Element der SimTech Aktivität deklariert. Die BPEL Aktivitäten *Check_01* und *Check_02* sind in der Definition nicht aufgeführt, obwohl sie ebenfalls in der SimTech Aktivität enthalten sind. Da sie aber Kinder der Aktivität *Check_Data* sind, und diese Aktivität als Teil der SimTech Aktivität definiert ist, werden die Aktivitäten *Check_01* und *Check_02* ebenfalls als Teil der SimTech Aktivität erkannt.
In der vorgestellten Definition werden die BPEL Aktivitäten über ihren Namen referenziert. Der BPEL 2.0 Standard [BPEL] schreibt eindeutige Benennungen aber nur in Einzelfällen vor. Dazu zählt beispielsweise die eindeutige Benennung aller *Scopes*, welche direkte Kinder des gleichen *Scopes* sind. Bei der Benennung aller weiteren Aktivitäten wird keine Eindeutigkeit gefordert.



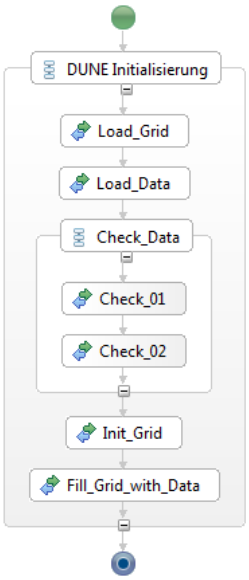

```
<bpel2simtech-mapping>
    <simtech-activity name="DUNE Initialisierung">
        <bpel-activities>
            <bpel-activity name="Load_Grid"/>
            <bpel-activity name="Load_Data"/>
            <bpel-activity name="Check_Data"/>
            <bpel-activity name="Init_Grid"/>
            <bpel-activity name="Fill_Grid_with_Data"/>
        </bpel-activities>
        <entry>
            <bpel-activity name="Load_Grid"/>
        </entry>
        <exit>
            <bpel-activity name="Fill_Grid_with_Data"/>
        </exit>
    </simtech-activity>
</bpel2simtech-mapping>
```

Abbildung 7: Definition einer SimTech Aktivität

Eine eindeutige Referenzierung von BPEL Aktivitäten ist damit nur möglich, wenn das Prozessmodell bekannt ist. In diesem Fall kann eine Aktivität beispielsweise durch eine Kombination aus *XPath* Ausdruck und ID des umgebenden *Scopes* eindeutig benannt werden (siehe [Ste08], Seite 63/64). Die Definition einer SimTech Aktivitäten soll aber, wie oben bereits erwähnt, unabhängig von konkreten Prozessmodellen sein.
Unter diesen Voraussetzungen kann eine eindeutige Referenzierung der BPEL Aktivitäten also nicht erreicht werden. Die Referenzierung über Aktivitätsnamen wurde gewählt, da der BPEL Standard den Namen als allgemeines Attribut für jede Art von Aktivität definiert. Zusätzlich ist dieser Ansatz intuitiv verständlich und nachvollziehbar, die Definition einer SimTech Aktivität ist „gut lesbar“. Damit die Zuordnung von BPEL Aktivitäten zu SimTech Aktivitäten auf diese Art funktioniert, muss von Seiten des Modellierungstools sichergestellt werden, dass die in der Definition der SimTech Aktivitäten verwendeten Benennungen im BPEL Prozessmodell eingehalten werden.

7.1.1. Points of Interest

Die Abbildung von BPEL Aktivitäten auf SimTech Aktivitäten verfolgt das Ziel, Komplexität vor dem Benutzer zu verbergen. Trotzdem kann es in manchen Fällen zu viel Abstraktion sein, wenn alle Ereignisse innerhalb einer SimTech Aktivität verborgen werden. Aus diesem Grund erlaubt die Definition einer SimTech Aktivität zusätzlich noch die Angabe von sogenannten *Points of Interest (POI)*.




```
<bpel2simtech-mapping>
    <simtech-activity name="DUNE Initialisierung">
        <bpel-activities>
            <bpel-activity name="Load_Grid"/>
            <bpel-activity name="Load_Data"/>
            <bpel-activity name="Check_Data"/>
            <bpel-activity name="Init_Grid"/>
            <bpel-activity name="Fill_Grid_with_Data"/>
        </bpel-activities>
        <entry>
            <bpel-activity name="Load_Grid"/>
        </entry>
        <exit>
            <bpel-activity name="Fill_Grid_with_Data"/>
        </exit>
        <points-of-interest>
            <bpel-activity name="Check_02"/>
        </points-of-interest>
    </simtech-activity>
</bpel2simtech-mapping>
```

Abbildung 8: Definition der Points of Interest

Events, welche als POI definierte Aktivitäten betreffen, werden nicht vor dem Benutzer versteckt. Auf diese Art und Weise können selektiv Informationen aus dem Inneren einer SimTech Aktivität nach außen bekannt gemacht werden.
In Abbildung 8 ist ein Beispiel für die Definition eines POI angegeben. Events, welche die Aktivität *Check_01* betreffen, werden nicht nach außen gegeben. Die Aktivität *Check_01* befindet sich innerhalb der SimTech Aktivität *DUNE_Initialisierung*, sie beeinflusst weder den Anfang noch das Ende der SimTech Aktivität. Gleiches würde für die Aktivität *Check_02* zutreffen. Da sie aber als POI angegeben ist, werden alle Events, welche diese BPEL Aktivität betreffen, nach außen publiziert.

7.2. Anpassung der ODE

Die im vorherigen Abschnitt vorgestellte Definition von SimTech Aktivitäten referenziert BPEL Aktivitäten über ihren Namen. Die von ODE erzeugten Events identifizieren die dazugehörige BPEL Aktivität aber nicht über den Namen, sondern über eine Kombination aus XPath Ausdruck und ID des direkt umgebenden *Scopes*. Wenn das zugrunde liegende Prozessmodell bekannt ist, dann kann mit diesen Informationen die betreffende BPEL Aktivität eindeutig identifiziert werden.
In Abschnitt 7.1 wurde argumentiert, dass diese Informationen im gegebenen Anwendungskontext nicht sinnvoll benutzbar sind. Da die von ODE erzeugten Events die benötigten Informationen nicht enthalten, wurde ODE im Rahmen dieser Arbeit an die gestellten Anforderungen angepasst.
Die in [Ste08] implementierten Erweiterungen ergänzen ODE unter anderem um die Fähigkeit, BPEL Events über JMS nach außen zu publizieren. Als Basis für diese Erweiterung dient das interne Eventmodell von ODE, welches über eine API zugreifbar ist. Auch die internen ODE Events enthaltenen keine Informationen über den Namen einer Aktivität. In Abbildung 9 ist der Aufbau des ODE Event Frameworks und des BPEL Eventmodells noch einmal schematisch dargestellt.
Um die von ODE publizierten BPEL Events um ein zusätzliches Attribut für den Aktivitätsnamen zu erweitern, mussten im ersten Schritt die internen ODE Events um ein solches Attribut ergänzt werden. Anschließend konnte dann die Implementierung des BPEL Eventmodells ebenfalls um dieses Attribut ergänzt werden.

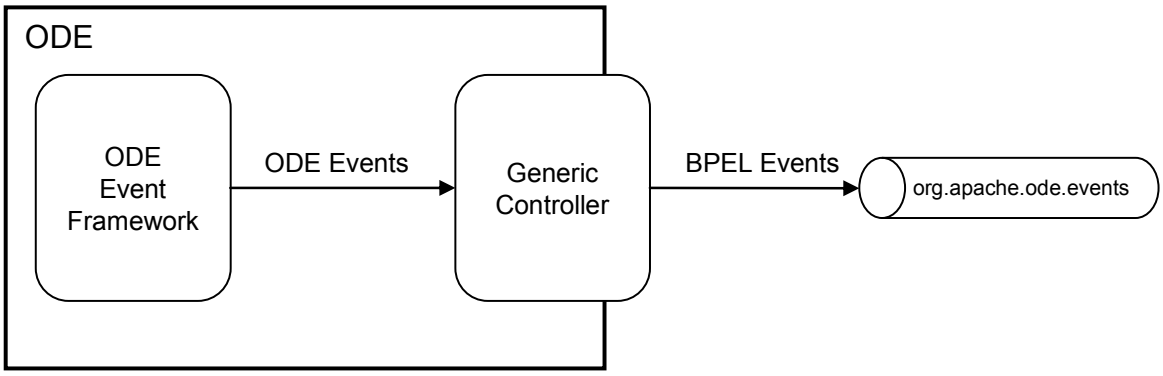


Abbildung 9: Erzeugung der BPEL Events

7.3. Abbildung von BPEL auf SimTech Events

Das Eventmodell für SimTech Aktivitäten wird in Form einer eigenständigen Mapping Komponente umgesetzt, welche im Folgenden als *BPEL2SimTech Mapper* bezeichnet wird. Diese Komponente empfängt die von der modifizierten ODE erzeugten BPEL Events. Abhängig von der gegebenen Definition der SimTech Aktivitäten publiziert der BPEL2SimTech Mapper selbst wiederum BPEL Events oder SimTech Events. Im Folgenden werden der Aufbau und die Funktionsweise des BPEL2SimTech Mappers näher erläutert.

7.3.1. Aufbau der Mapping Komponente

In Abbildung 10 ist der Aufbau des BPEL2SimTech Mappers schematisch dargestellt. Der Mapper besteht aus den Modulen *JMSCommunication*, *MappingDescriptor* und *BPEL2SimTechMapper*.

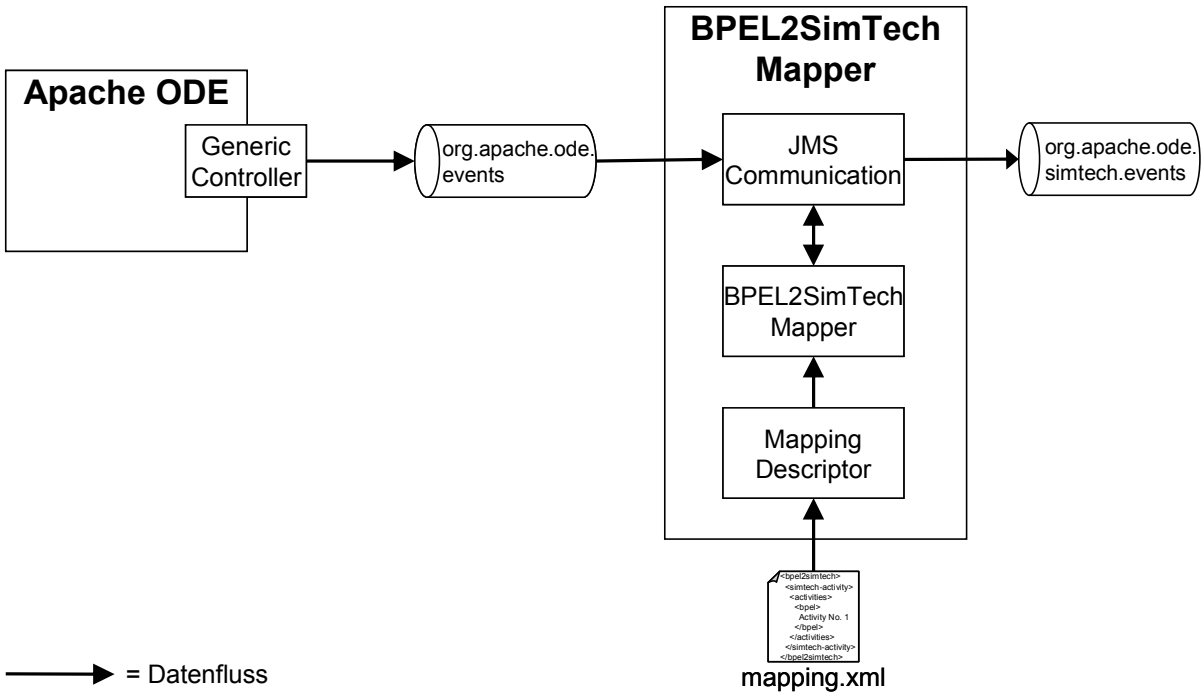


Abbildung 10: Aufbau der Komponente BPEL2SimTech Mapper

Beim Start liest das Modul MappingDescriptor zunächst das in Abschnitt 7.1 vorgestellte Mapping der SimTech Aktivitäten ein. Nach dem Einlesen der Definitionsdatei registriert sich das Modul JMSCommunication beim JMS Topic *org.apache.ode.events*. Werden BPEL Prozesse auf der ODE ausgeführt, so publiziert der in [Ste08] entwickelte *Generic Controller* entsprechende BPEL Events auf diesem Topic. Das Modul JMSCommunication empfängt diese Events und leitet sie an das Modul BPEL2SimTechMapper weiter. Basierend auf den im Modul MappingDescriptor hinterlegten Informationen über die Definition der SimTech Aktivitäten, verarbeitet der Mapper die übergebenen Events. Erzeugt das Modul BPEL2SimTechMapper BPEL Events oder SimTech Events, so werden diese wiederum über das Modul JMSCommunication auf dem JMS Topic *org.apache.ode.simtech.events* publiziert.

7.3.2. Funktionsweise/Verhalten der Mapping Komponente

Der BPEL2SimTech Mapper empfängt BPEL Events von der ODE und kann damit drei unterschiedliche Dinge tun. Die Events können unverändert weiter geschickt werden, sie können durch SimTech Events ersetzt werden, oder sie werden nicht weiter gesendet. Was getan wird, hängt davon ab, wie die SimTech Aktivitäten definiert sind und auf welche BPEL Aktivität sich das BPEL Event bezieht.

Betrifft das empfangene Event eine BPEL Aktivität, welche nicht Teil einer SimTech Aktivität ist, so wird das Event unverändert weiter gesendet. Betrifft das empfangene Event eine BPEL Aktivität, welche Teil einer SimTech Aktivität ist, dann gibt es mehrere Alternativen, wie dieses Event verarbeitet werden kann. Ist die betreffende BPEL Aktivität als POI definiert, so wird das BPEL Event unverändert weiter gesendet. Ist die BPEL Aktivität weder als POI, noch als Start oder Ende einer SimTech Aktivität definiert, so wird das Event nicht weiter gesendet. In Abbildung 11 wird dieses Verhalten an einem Beispiel gezeigt. Die BPEL Aktivität *Check_02* ist als POI deklariert. Events, die diese Aktivität betreffen, werden weiter geleitet. Die Aktivität *Load_Data* befindet sich innerhalb der SimTech Aktivität und ist nicht als POI definiert. Events, welche diese Aktivität betreffen, werden nicht weiter geleitet.

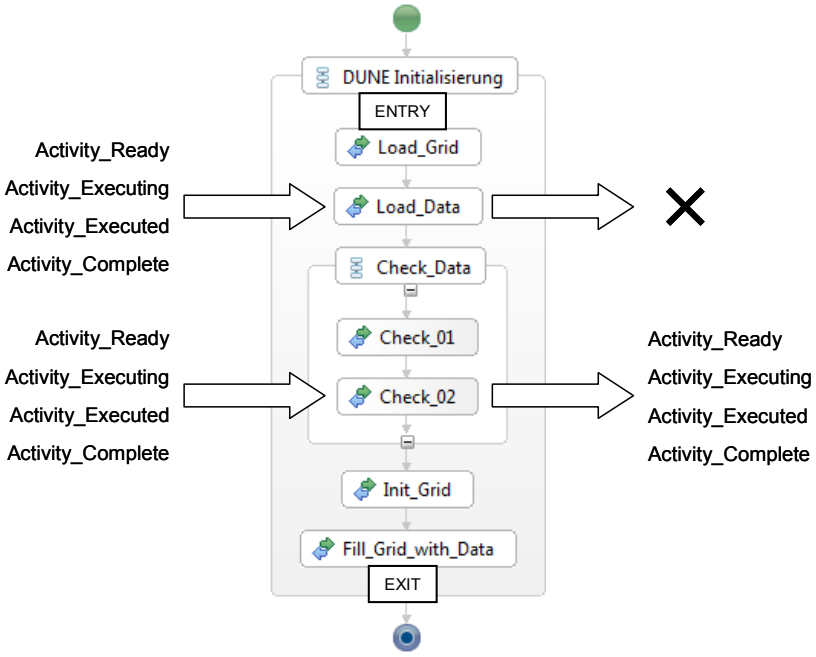


Abbildung 11: Events innerhalb einer SimTech Aktivität

Das Verhalten für Events, welche den Start oder das Ende einer SimTech Aktivität betreffen, ist in Abbildung 12 dargestellt. Auf der linken Seite sind Events für die BPEL Aktivitäten dargestellt, die den Beginn und das Ende der SimTech Aktivität darstellen. Auf der rechten Seite befinden sich die vom BPEL2SimTech Mapper weiter gesendeten SimTech Events, welche das Verhalten der SimTech Aktivität beschreiben.

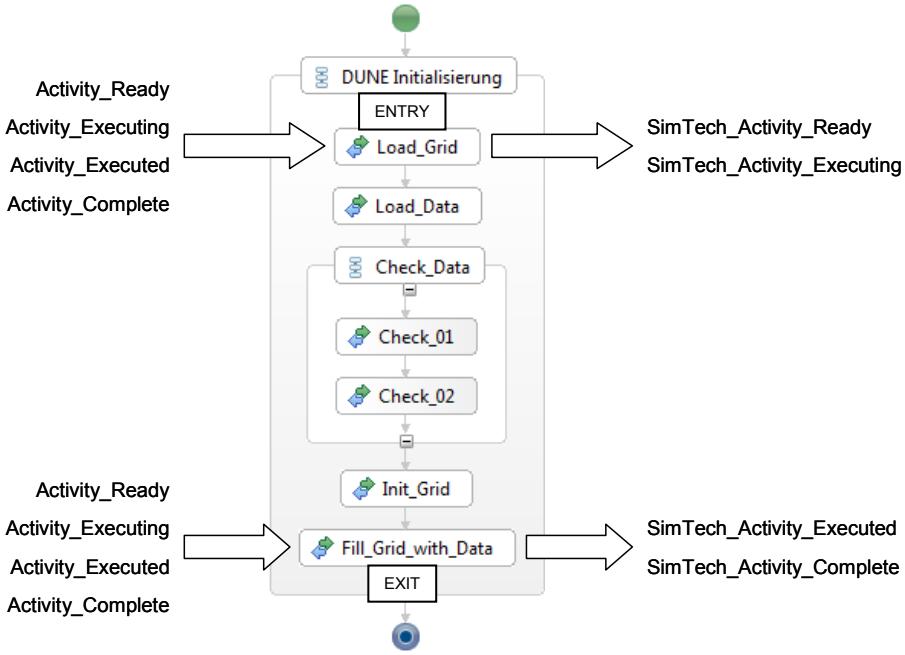


Abbildung 12: Events bei Beginn und Ende einer SimTech Aktivität

8 Implementierung des Monitorings für DUNE Simulationen

Die im Folgenden vorgestellte Anwendung zum Monitoring von DUNE Simulationen stellt dem Benutzer den aktuellen Zustand der Ausführung einer DUNE Simulation dar. Dieser Zustand ergibt sich aus den Zuständen der einzelnen Aktivitäten, aus denen die DUNE Simulation besteht.
Ausgangspunkt für den Entwurf der Monitoring Anwendung ist die in Kapitel 7 vorgestellte Mapping Komponente, welche das SimTech Eventmodell implementiert. Diese Komponente veröffentlicht während der Ausführung eines BPEL Prozesse sowohl BPEL Events als auch SimTech Events. Ein während der Prozessausführung ausgelöstes Event beschreibt eine potentielle Änderung des Zustands einer Aktivität, und damit auch eine Änderung des Zustands der ausgeführten Prozessinstanz. Wird ein Event ausgelöst, so bestimmt das Event in Kombination mit dem bisherigen Zustand den Folgezustand.
Um die graphische Anzeigekomponente des Monitorings leichtgewichtig zu halten, wird das Monitoring als System von zwei lose gekoppelten Komponenten entworfen. Die Komponente *Event2StateMapper* bildet die empfangenen Events auf Zustandsmeldungen ab. Die Komponente kennt den aktuellen Zustand aller ausgeführten Prozessinstanzen sowie der daran beteiligten Aktivitäten. Bewirkt ein eingehendes Event eine Änderung des Zustands, so erzeugt der *Event2StateMapper* eine entsprechende Zustandsmeldung, welche Informationen über den neuen Zustand enthält. Die Anzeigekomponente *Monitoring View* empfängt die vom Event2StateMapper erzeugten Zustandsmeldungen und stellt diese graphisch dar.
Im Folgenden werden die beiden Komponenten des Monitoring Systems und ihre Umsetzung näher vorgestellt. In Abbildung 13 sind die Komponenten und ihr Zusammenspiel schematisch dargestellt.

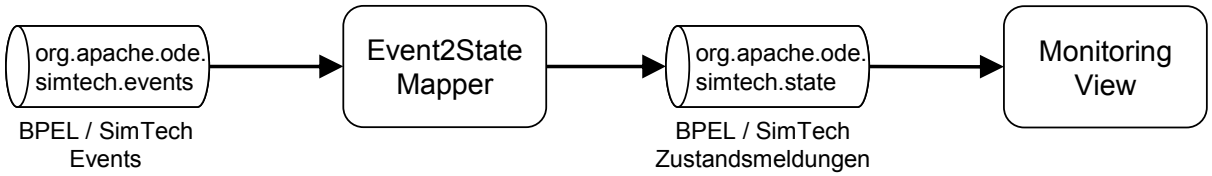


Abbildung 13: Aufbau der Monitoring Komponente

8.1. Abbildung von Events auf Zustände

Die Komponente Event2State Mapper empfängt BPEL und SimTech Events, bestimmt den aktuellen Zustand der betreffenden Aktivität, und veröffentlicht entsprechende Zustandsmeldungen. Die Komponente besteht aus mehreren Modulen. Die einzelnen Module sowie ihr Zusammenspiel werden im Folgenden vorgestellt.
Das Modul *JMSCommunication* ist für das Empfangen und Senden von Nachrichten zuständig. Es empfängt Events, die auf dem JMS Topic *org.apache.ode.simtech.events* publizierte werden, und reicht diese an das Modul *Event2StateMapper* weiter.
Im *Event2StateMapper* werden die aktuellen Zustände aller ausgeführten DUNE Simulationen gespeichert. Empfängt der Event2StateMapper ein Event, so werden zunächst die betreffende Aktivität und ihr aktueller Zustand bestimmt. Mit diesen Informationen kann der neue Zustand der Aktivität bestimmt werden. Abhängig davon, ob das aufgetretene Event eine BPEL oder eine SimTech Aktivität betrifft, wird diese Aufgabe vom Modul *BPELStateMachine* oder vom Modul *SimTechStateMachine* übernommen.

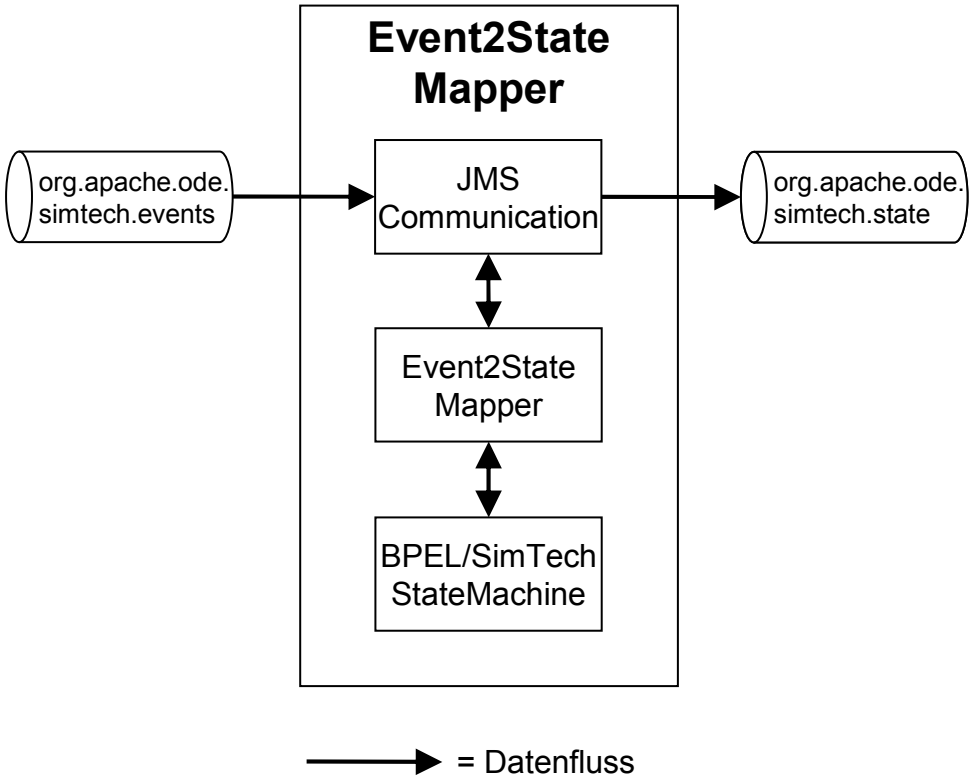


Abbildung 14: Aufbau Event2State Mapper

Die Implementierung der StateMachine Module folgt den Zustandsübergangsdiagrammen, durch welche die BPEL und SimTech Eventmodelle beschrieben werden. Den Modulen werden der aktuelle Zustand einer Aktivität sowie das aufgetretene Event übergeben. Sie bestimmen daraufhin den Folgezustand der Aktivität und geben diesen zurück.
Das Modul *Event2StateMapper* speichert den Folgezustand und verwirft dafür den bisherigen Zustand. Anschließend erzeugt es eine entsprechende Zustandsmeldung und übergibt sie an das Modul *JMSCommunication*. Die Zustandsmeldung wird dann auf dem JMS Topic *org.apache.ode.simtech.state* veröffentlicht. Das Zusammenspiel der vorgestellten Module ist in Abbildung 14 dargestellt.

8.2. Monitoring als View in Eclipse

Die graphische Darstellungskomponente des Monitorings wird als *Eclipse View* implementiert. Eine *View* ist ein Grundelement der graphischen Oberfläche von Eclipse (Eclipse GUI). Abbildung 15 zeigt eine exemplarische Eclipse GUI. Die mit (1) bezeichneten Elemente sind Views. Als weiteres besteht eine Eclipse GUI üblicherweise aus Menüs (2), Toolbars (3) und der Editor Area (4).

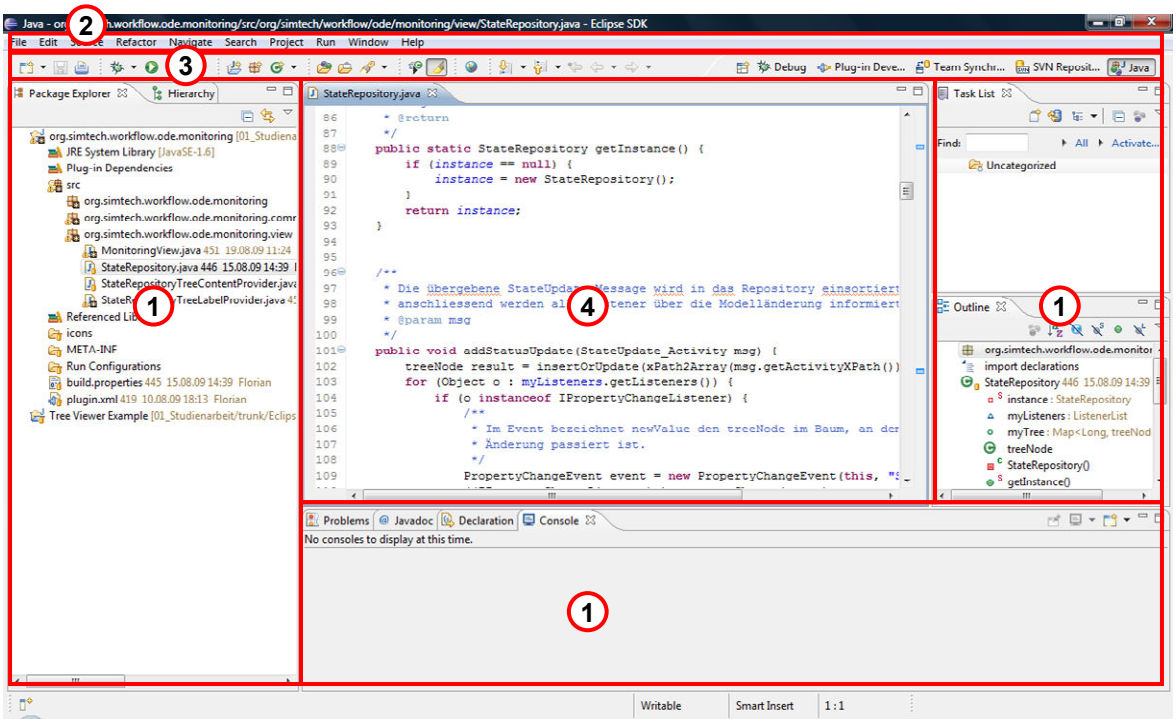


Abbildung 15: Eclipse GUI Komponenten

Der aktuelle Zustand der Ausführung einer DUNE Simulation wird in der entwickelten *Monitoring View* als Baum dargestellt. Elemente der ersten Ebene sind die Prozessinstanzen. Die einzelnen Aktivitäten der Prozessinstanzen werden im Baum als Kindelemente dargestellt. Da die Struktur eines BPEL Prozesses immer eine Baumstruktur ist, lässt sich auf diese Art und Weise jede DUNE Simulation eindeutig darstellen.

Zur graphischen Ausgestaltung von Views bietet Eclipse das *Standard Widget Toolkit (SWT*[12]*)* an. Die von diesem Toolkit angebotenen GUI Komponenten werden als *Widgets* bezeichnet. Widgets sind beispielsweise Textfelder, Buttons, Listen, Menüs oder Bäume[13]. Ein weiteres von Eclipse angebotenes GUI Toolkit ist *JFace*. JFace reichert SWT mit zusätzlicher Funktionalität an und erweitert es unter anderem um sogenannte *Viewer*. Viewer bilden eine zusätzliche Abstraktionsschicht für den Zugriff auf Widgets.

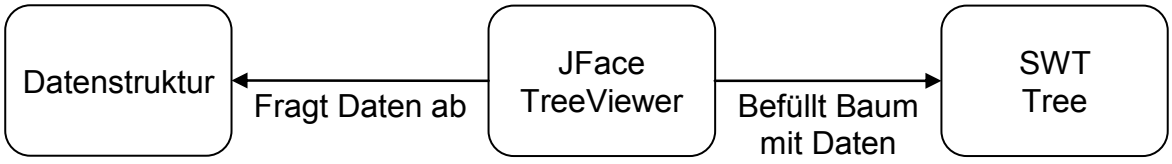


Abbildung 16: JFace TreeViewer

In Abbildung 16 ist schematisch dargestellt, wie ein Viewer verwendet werden kann, um ein Widget (in diesem Beispiel ein Baum) mit einer beliebigen Datenstruktur zu verknüpfen. JFace definiert für diesen Anwendungszweck Schnittstellen, die von der Datenstruktur implementiert werden müssen. JFace nutzt diese Schnittstellen dann, um auf die in der Datenstruktur enthaltenen Daten zuzugreifen und diese im Baum darzustellen.

[12] http://www.eclipse.org/swt/

[13] Eine komplette Übersicht aller Widgets findet sich unter http://www.eclipse.org/swt/widgets/

Die Monitoring View wurde nach diesem Konzept unter Nutzung eines SWT Tree Widget und eines JFace TreeViewer realisiert. In Abbildung 13 ist auf der rechten Seite dargestellt, dass die graphische Komponente des Monitorings aus mehreren Modulen besteht. Das Modul *JMSCommunication* empfängt die vom in Abschnitt 8.1 beschriebenen *Event2State Mapper* erzeugten Zustandsmeldungen und gibt sie an das Modul *StateRepository* weiter. Das StateRepository speichert diese Zustandsmeldungen und ersetzt dabei ggf. alte Zustandsmeldungen. Es repräsentiert die aktuellen Zustände aller ausgeführten DUNE Simulationen, also genau die Daten, die im Monitoring angezeigt werden sollen. Der *TreeViewer* greift auf das StateRepository zu, um die aktuellen Zustände aller ausgeführten DUNE Simulationen als Baum darzustellen. Die Komponenten Monitoring View ist in Abbildung 17 schematisch dargestellt.

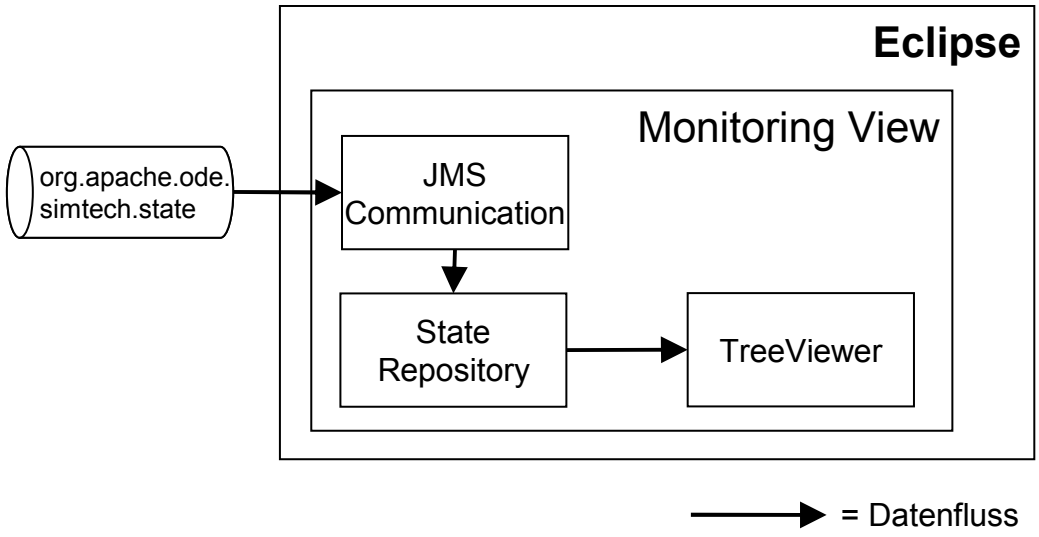


Abbildung 17: Aufbau Monitoring View

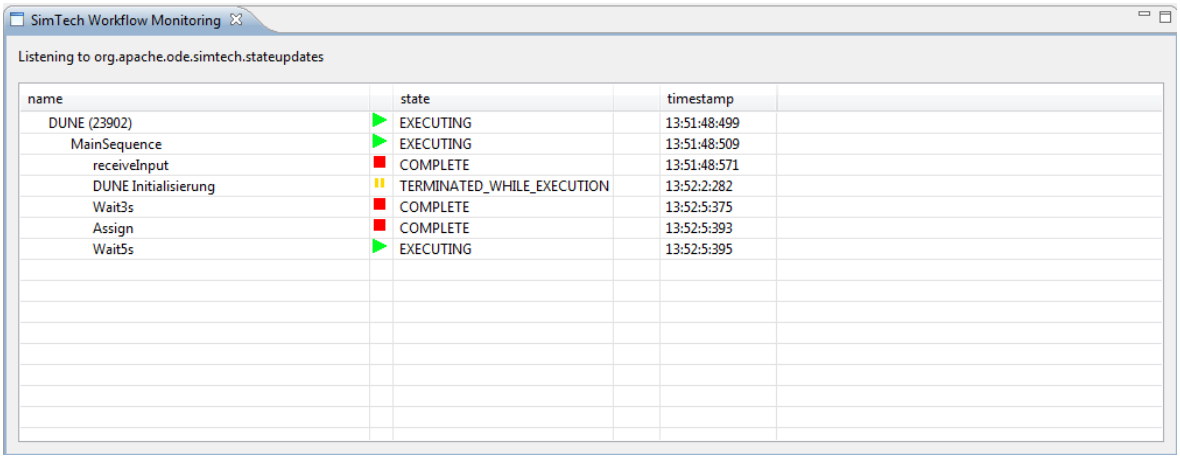


Abbildung 18: SimTech Monitoring View

In Abbildung 18 ist die entwickelte Monitoring View während der Ausführung einer DUNE Simulation dargestellt. Dem Benutzer werden Informationen über den Namen, den Zustand und den Zeitpunkt der letzten Zustandsänderung jeder Aktivität dargestellt.

9 Zusammenfassung und Ausblick

In Abbildung 19 sind alle in dieser Arbeit beschriebenen und implementierten Softwarekomponenten dargestellt. Ausgangspunkt dieser Arbeit war das in [Ste08] entwickelte Eventmodell für BPEL 2.0 und dessen Implementierung in ODE. Ausgehend davon wurde ein Eventmodell für SimTech Aktivitäten entworfen und mit der Komponente *BPEL2SimTech Mapper* umgesetzt. Um den aktuellen Zustand der Ausführung einer DUNE Simulation im Monitoring darstellen zu können, wurde zunächst die Komponente *Event2State Mapper* entwickelt. Sie wandelt die vom *BPEL2SimTech Mapper* erzeugten Events in Zustandsmeldungen um. Diese Zustandsmeldungen werden dann von der *Monitoring View* empfangen und in einer Baumstruktur dargestellt.

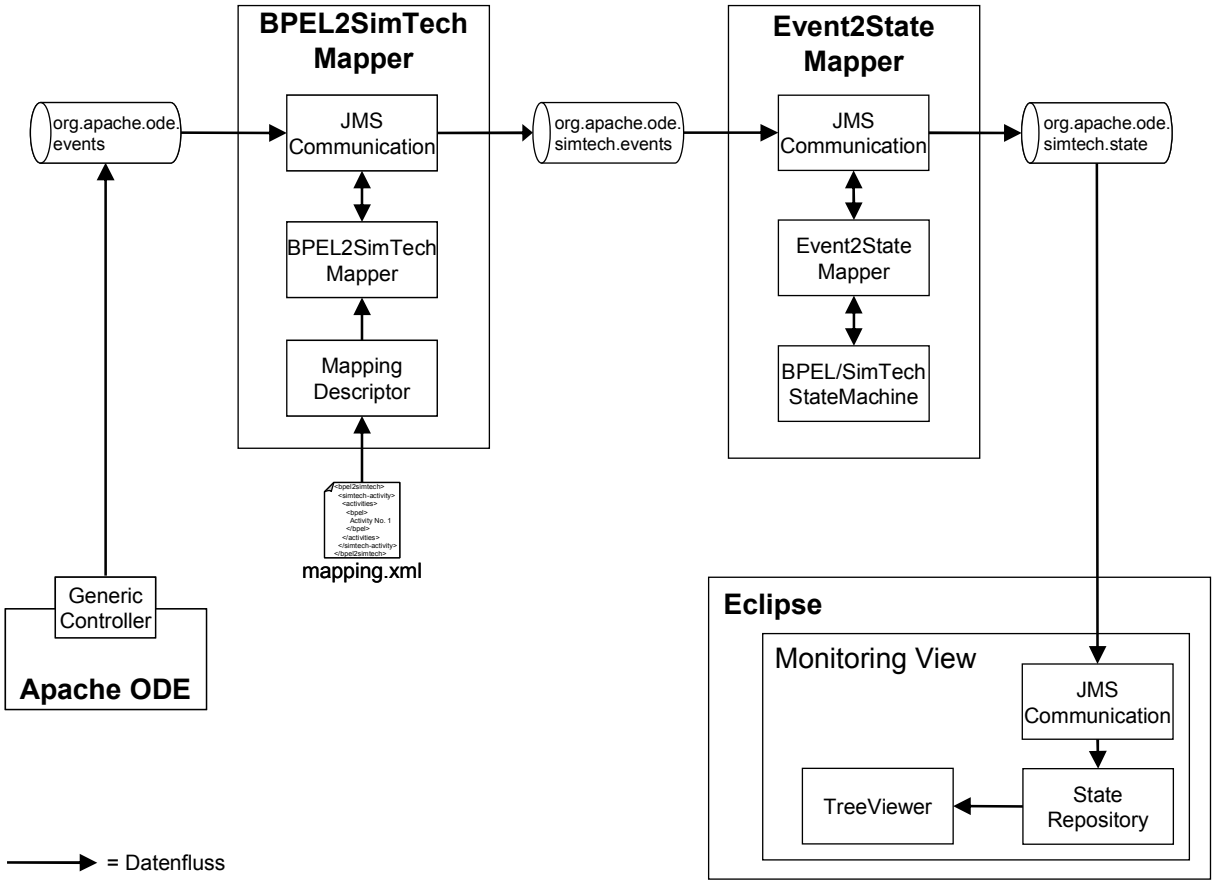


Abbildung 19: Überblick aller in dieser Arbeit entwickelten Komponenten

Die Komponente BPEL2SimTech Mapper wurde als generische Mapping Komponente entworfen und umgesetzt. Die Funktionalität der Komponente wird durch ein separat angegebenes Mapping konfiguriert. Werden in Zukunft neue SimTech Aktivitäten entwickelt und definiert, so muss lediglich die Mapping Datei (mapping.xml) angepasst werden.

Die Monitoring Komponente wurde in dieser Arbeit prototypisch implementiert. Sie realisiert eine einfache Anzeige des Ausführungszustandes von DUNE Simulationen. Bei einer Weiterentwicklung der Komponente kann sie um weitere Funktionalitäten erweitert werden. So kann es beispielsweise sinnvoll sein, dem Benutzer zu erlauben, die angezeigten Zustände zu filtern oder ganze Prozessinstanzen auszublenden. Mit der in [Ste08] entwickelten Funktionalität der blockierenden

Events ist es sogar möglich, die Monitoring Komponente so zu erweitern, dass Prozesse während der Ausführung nicht nur passiv beobachtet sondern auch aktiv beeinflusst werden können.

10 Referenzen

BPEL Organization for the Advancement of Structured Information Standards (OASIS): Web Services Business Process Execution Language Version 2.0, OASIS Standard, 11 April 2007. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

HW03 Hohpe, Gregor; Woolf, Bobby: Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley Professional, 2003

LR00 Leymann, F.; Roller, D.: Production Workflow. Concepts and Techniques. Prentice Hall Inc., 2000

Ste08 Steinmetz, Thomas: Ein Event-Modell für WS-BPEL 2.0 und dessen Realisierung in Apache ODE. Institut für Architektur von Anwendungssystemen der Universität Stuttgart, Diplomarbeit, August 2008

TDG+07 Taylor, I.J.; Deelman, E.; Gannon, D.B.; Shields, M. (Eds.): Workflows for e-Science, Scientific Workflows for Grids. Springer 2007

Vuk09 Vukojević, Karolina: Architektur eines Workflow-Frameworks zur graphischen Erstellung und Ausführung von Simulationsexperimenten. Institut für Architektur von Anwendungssystemen der Universität Stuttgart, Studienarbeit, September 2009

WCL+05 Weerawarana, S.; Curbera, F.; Leymann, F.; Ferguson, D.F.; Storey, T.: Web Services Platform Architecture. Prentice Hall International, 2005

WSDL World Wide Web Consortium (W3C): Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language. http://www.w3.org/TR/wsdl20/

Alle in diesem Dokument 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