

Grobentwurf

Version 1.0

29. Oktober 2009



Dok-Status: neu

QS-Status: nicht QS-geprüft

Prüf-Status: nicht geprüft

Review-Status: kein Review durchgeführt

End-Status: -

Inhaltsverzeichnis

1	Einleitung	6
1.1	Zweck des Dokuments	6
1.2	Das SIMPL Rahmenwerk	6
1.3	Anforderungen	6
1.3.1	Einfache Modellierung	6
1.3.2	Große Datenmengen	6
1.3.3	Auditing von Prozessen	6
1.3.4	Late Binding von Datenquellen	7
1.3.5	Authentifizierung und Autorisierung	7
1.3.6	Registry	7
1.3.7	Admin-Konsole	7
1.3.8	Erweiterbarkeit	7
1.3.9	Verwendbarkeit	7
1.4	Entwurfsprinzipien	8
1.4.1	Offenes Rahmenwerk	8
1.4.2	Modularisierung	8
1.4.3	Kopplung und Zusammenhalt	8
1.4.4	Entwicklungsrichtung	8
1.4.5	Plug-In	8
1.4.6	Adapter	8
1.5	Überblick über den Grobentwurf	8
2	Architektur	10
2.1	Apache Tomcat	10
2.2	Eclipse	10
3	Komponenten	11
3.1	Eclipse	11
3.1.1	Eclipse BPEL Designer Plugin	11
3.1.2	SIMPL Eclipse Plugin	12
3.2	Apache ODE	13
3.2.1	Extension Activities	14
3.2.2	Event System	14
3.2.3	Management API	14
3.2.4	Auditing Datenbank	15
3.2.5	Deployment Service	15
3.2.6	BPEL Runtime	15
3.3	SIMPL Core	15
3.3.1	Strategy Service	16
3.3.2	Datasource Service	16
3.3.3	Reference Service	16
3.3.4	Security Service	16
3.3.5	Administration Service	16
3.3.6	Transformation Service	16
3.3.7	Storage Service	16
3.3.8	Registry Service	16
3.4	Apache jUDDI	17

4 Schnittstellen	18
4.1 Apache ODE	18
4.1.1 ManagementAPI	18
4.1.2 DeploymentService	19
4.2 Apache jUDDI	19
4.3 SIMPL Core	19
4.3.1 DatasourceService	19
4.3.2 StrategyService	19
4.3.3 ReferenceService	19
4.3.4 SecurityService	19
4.3.5 AdministrationService	19
4.3.6 TransformationService	19
4.3.7 StorageService	20
4.3.8 RegistryService	20
Literaturverzeichnis	21
Abkürzungsverzeichnis	22
Abbildungsverzeichnis	23

Änderungsgeschichte

Version	Datum	Autor	Änderungen
0.1	29.10.2009	schneimi	Erstellung des Dokuments
0.2	30.10.2009	schneimi	Kapitel 1
0.3	04.11.2009	schneimi	Kapitel 2, Kapitel 3.3
0.4	07.11.2009	schneimi	Kapitel 4
0.5	09.11.2009	rehnre	Kapitel 3.2
0.6	9.11.2009	bruededl	Kap. 3.1

1 Einleitung

Dieses Kapitel soll dem Leser einen Überblick über das SIMPL Rahmenwerk geben und Zweck, Anforderungen und die eingesetzten Entwurfsprinzipien erläutern. Die Struktur und der Aufbau des Dokuments orientieren sich dabei an der Entwurfsvorlage [1] von Markus Knauß.

1.1 Zweck des Dokuments

Der Grobentwurf beschreibt das Rahmenwerk auf Komponentenebene und bildet die Grundlage für den späteren Feinentwurf. Es werden alle wichtigen Komponenten sowie Schnittstellen identifiziert und ihr Zusammenspiel beschrieben. Damit soll ~~herausgestellt~~ werden, dass das resultierende System funktionieren kann und den Anforderungen gerecht wird.

1.2 Das SIMPL Rahmenwerk

Das SIMPL Rahmenwerk soll dem Benutzer eine generische und erweiterbare Umgebung bieten, die eine einfache Modellierung von BPEL-Geschäftsprozessen mit Zugriff auf beliebige Datenquellen ermöglicht. Bei den Datenquellen kann es sich beispielsweise um Datenbanken, Sensornetze oder Dateisysteme handeln. Die Modellierung der Prozesse findet in Eclipse mit dem Eclipse BPEL Designer Plug-In statt, das für diesen Zweck um zusätzliche Aktivitäten, für den Zugriff auf Datenquellen, erweitert wird. Die Ausführung der Prozesse erfolgt durch die Apache ODE Workflow Engine, bei der das bestehende Event-Modell und das Auditing der Prozessdaten für diese Aktivitäten angepasst werden. Dienste, die für die Ausführung der Aktivitäten von der Workflow Engine benötigt werden, werden in Form von Web Services bereitgestellt. Diese Aktivitäten werden im folgenden Verlauf des Dokuments als Data-Management-Aktivitäten bzw. DM-Aktivitäten bezeichnet.

1.3 Anforderungen

In diesem Abschnitt werden die Anforderungen des SIMPL Rahmenwerks beschrieben.

1.3.1 Einfache Modellierung

Bei der Modellierung von Prozessen wiederholen sich in der Regeln häufig längere Statements in Anfragesprachen wie z.B SQL und XQuery oder auch längere Namen von Datenquellen oder Datencontainern (Tabellen, Dateien, XML-Dokumente, etc). Damit diese vom Prozess-Modellierer nicht jedes mal vollständig angegeben werden müssen, soll es die Möglichkeit geben diese in BPEL-Variablen zu hinterlegen, die anschließend als Referenzen in anderen Statements verwendet werden können.

1.3.2 Große Datenmengen

Der Schwerpunkt des Rahmenwerks liegt bei der Modellierung von wissenschaftlichen Prozessen, bei denen überwiegend mit großen Datenmengen gearbeitet wird. Damit diese Datenmengen nicht innerhalb des Prozess gehalten werden müssen, wird ein Reference Resolution System (RRS) realisiert, das es ermöglicht Daten zu referenzieren, die nur bei Bedarf aufgelöst werden und somit sehr schnell weitergegeben werden können.

21.55 cm

1.3.3 Auditing von Prozessen

Bei dem Auditing von Prozessen handelt es sich um das Protokollieren von Daten, die in Prozessen anfallen und ein Monitoring der Prozesse ermöglichen. Die Erfassung der Daten muss auf die zusätzlichen DM-Aktivitäten angepasst werden und dem Benutzer die Möglichkeit gegeben werden, die Granularität der Daten zu bestimmen. Zusätzlich soll das interne Auditing von Apache ODE, auf eine frei wählbare Datenbank umgeleitet werden können.

1.3.4 Late Binding von Datenquellen

Da bei der Modellierung nicht immer unmittelbar feststeht, auf welche Datenquelle zugegriffen werden soll, beispielsweise beim Ablegen von Daten, soll ein Late Binding von Datenquellen unterstützt werden. Damit kann durch die Formulierung von Anforderungen an die Datenquelle und das Wählen einer Auswahlstrategie durch den Prozess-Modellierer, eine passende Datenquelle zur Laufzeit bestimmt werden.

1.3.5 Authentifizierung und Autorisierung

Datenquellen erfordern in der Regel eine Authentifizierung und Autorisierung des Benutzers bei einem Zugriff. Das Rahmenwerk soll dem Prozess-Modellierer diesen Vorgang vereinfachen, so dass die dafür benötigten Informationen wie z.B. Name und Passwort nicht bei jedem wiederholten Zugriff erneut angegeben werden müssen. Dazu soll das Konzept des Single Sign On (SSO) angeboten bzw. unterstützt werden. (siehe [3], Kapitel 7.2.1)

1.3.6 Registry

Für die Verwaltung der für den Prozess-Modellierer zur Verfügung stehenden Datenquellen, soll eine Registry bereitgestellt werden, in der die Datenquellen vom Workflow-Administrator zentral definiert werden können. Dort werden auch die entsprechenden Eigenschaften der Datenquellen hinterlegt, die für das Late Binding benötigt werden.

1.3.7 Admin-Konsole

Für alle wichtigen Einstellungen des Rahmenwerks soll eine Admin-Konsole bereitgestellt werden, über die der Workflow-Administrator Einstellungen treffen kann. Dies betrifft vor allem alle Einstellungen die auch zur Laufzeit getätigt werden können, wie z.B. das Einbinden einer zusätzlicher Datenquelle.

1.3.8 Erweiterbarkeit

Das Rahmenwerk stellt folgende Anforderungen an die Erweiterbarkeit und soll dafür entsprechende "Hot Spots" bereitstellen:

- Weitere Typen von Datenquellen
- Unterstützung weiterer Anfragesprachen
- Erweiterung um neue DM-Aktivitäten
- Erweiterung um neue Events für das Auditing
- Unterstützung weiterer Authentifizierungs- und Autorisierungsverfahren
- Unterstützung weiterer Registries
- Austauschbare GUI der Admin-Konsole

1.3.9 Verwendbarkeit

Die Aktivitäten, die für SIMPL entwickelt werden, sollen auf beliebigen anderen BPEL Workflow-Engines ~~ausgeführt werden können~~. Dazu muss der von den Aktivitäten bei der Modellierung erzeugte, erweiterte Code, in Standard-BPEL transformiert werden. Eine zusätzliche Anforderung ist, dass die von SIMPL bereitgestellten Web Services, auf verschiedenen Web Containern lauffähig sein müssen.

1.4 Entwurfsprinzipien

In diesem Abschnitt werden die Prinzipien beschrieben, die bei dem Entwurf angewendet werden. (vgl. [2], Kapitel 17)

1.4.1 Offenes Rahmenwerk

Bei dem SIMPL Rahmenwerk handelt es sich größtenteils um ein offenes Rahmenwerk. Daher können viele Erweiterungen nicht ohne technisches Verständnis und Wissen über die Mechanismen und Abläufe des Rahmenwerks realisiert werden. Die Erweiterungsmöglichkeiten werden daher ausführlich dokumentiert und Beispiele erstellt, anhand denen eigene Erweiterungen umgesetzt werden können. Bereiche des Rahmenwerks bei denen im Laufe des Projekts ausreichend Erfahrung gesammelt wurde, werden wenn möglich in geschlossener Form für die Entwicklung bereitgestellt.

1.4.2 Modularisierung

Durch die Modularisierung werden die Komponenten in einfache und leicht verständliche Teile gegliedert. Die Realisierungsdetails eines Moduls werden nach dem Prinzip des Information Hiding versteckt und die Dienste nur über eine Schnittstelle angeboten. Ziel ist es, später Module ändern oder austauschen zu können, möglichst ohne dabei die Schnittstellen ändern zu müssen und damit Auswirkungen auf andere Module zu verursachen.

1.4.3 Kopplung und Zusammenhalt

Bei dem Entwurf der Module wird darauf geachtet, dass die Kopplung zu anderen Modulen **möglichst** gering bleibt und der Zusammenhalt innerhalb des Moduls **möglichst** hoch wird. Durch dieses Vorgehen wird eine hohe Lokalität und damit gute Wartbarkeit erreicht, da sich Fehler die bei Änderungen entstehen, nicht im System fortpflanzen können.

1.4.4 Entwicklungsrichtung

Bei der Entwicklung wird Top-down vorgegangen. Dabei wird die Aufgabe des Rahmenwerks rekursiv bis zur elementaren Ebene (der Programmiersprache) in Teilaufgaben zerlegt und damit schrittweise verfeinert.

1.4.5 Plug-In

Plug-Ins sind externe Software-Einheiten, durch die das Rahmenwerk um zusätzliche Funktionalität erweitert werden kann. Das Rahmenwerk bietet dafür entsprechende **“Hot Spots”**, an denen die Plug-Ins angeschlossen werden können.

1.4.6 Adapter

Adapter bzw. Konnektoren sind interne Verbindungsstücke, die dort entwickelt werden, wo Komponenten angebunden werden sollen, deren Schnittstellen nicht zu vorhandenen Schnittstellen des Rahmenwerks passen.

1.5 Überblick über den Grobentwurf

- Kapitel 2 “Architektur” beschreibt die Architektur des Rahmenwerks. Das Rahmenwerk wird in überschaubare Komponenten gegliedert, die jeweils genau definierte Funktionen erfüllen.
- Kapitel 3 “Komponenten” beschreibt die im Kapitel “Architektur” genannten Komponenten im Detail. Dabei werden Schnittstellen, Protokolle und Verhalten definiert.

- Kapitel 4 “Schnittstellen” beschreibt die in Kapitel “Komponenten” definierten Schnittstellen und ihre Verwendung.

2 Architektur

Abbildung 1 zeigt eine Übersicht des SIMPL Rahmenwerks. Die übergeordneten Komponenten und ihr Zusammenspiel werden im Folgenden kurz beschrieben. Auf die elementaren Komponenten des Rahmenwerks wird dann in Kapitel 3 eingegangen.

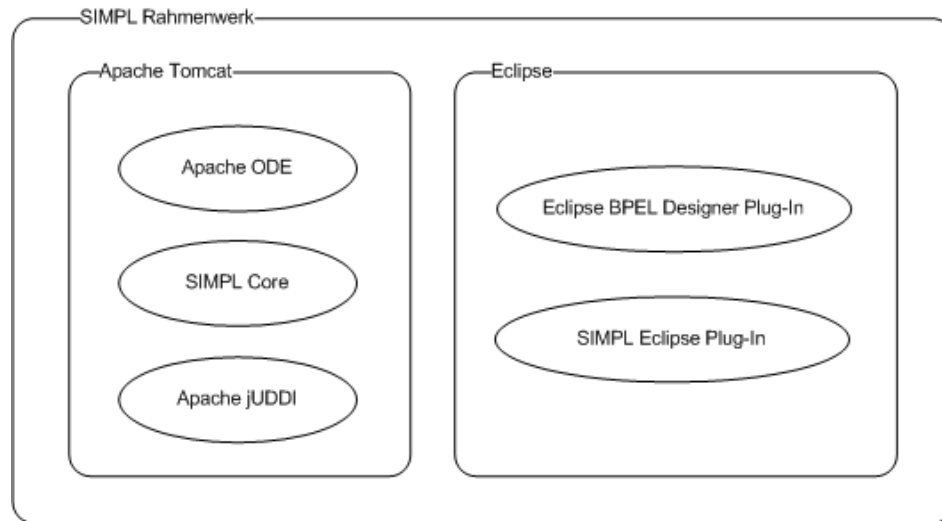


Abbildung 1: Architektur des SIMPL Rahmenwerks

2.1 Apache Tomcat

Apache Tomcat ist die Laufzeitumgebung für Apache ODE, sowie den SIMPL Core und die Registry Apache jUDDI. Apache ODE ist für die Ausführung der Prozesse nach der Modellierung zuständig und benötigt, wie bereits in Kapitel 1.2 erwähnt, bestimmte Dienste für die Ausführung der zusätzlichen DM-Aktivitäten in Prozessen. Diese Dienste werden von dem SIMPL Core bereitgestellt, der sich aus verschiedenen, nach Aufgaben eingeteilten, Web Services zusammensetzt. Mit der Registry Apache jUDDI werden die im Rahmenwerk zur Verfügung stehenden Datenquellen vom Workflow-Administrator zentral bereitgestellt.

2.2 Eclipse

Die Entwicklungsumgebung Eclipse bildet, mit den entsprechenden Plug-Ins, die GUI für den Prozess-Modellierer, als auch den Workflow-Administrator. Mit dem Eclipse BPEL Designer Plug-In kann der Prozess-Modellierer bereits BPEL Prozesse erstellen und auf dem Apache ODE zum Einsatz bringen (deployen). Das SIMPL Eclipse Plug-In erweitert nun zum Einen die bestehenden Aktivitäten des Eclipse BPEL Designer Plug-Ins um die DM-Aktivitäten und bietet zum Anderen die GUI mit den Einstellungen für den Workflow-Administrator, wie z.B die Admin-Konsole und die Globalen Einstellungen (siehe [3], Kapitel 4).

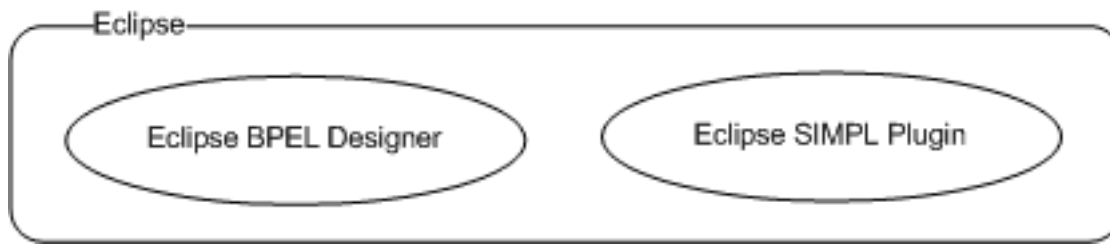


Abbildung 3: Eclipse Komponenten-Diagramm

3 Komponenten

In Abbildung 2 werden die für das SIMPL Rahmenwerk wichtigen Komponenten und Abhängigkeiten gezeigt, die in den folgenden Abschnitten näher beschrieben werden.

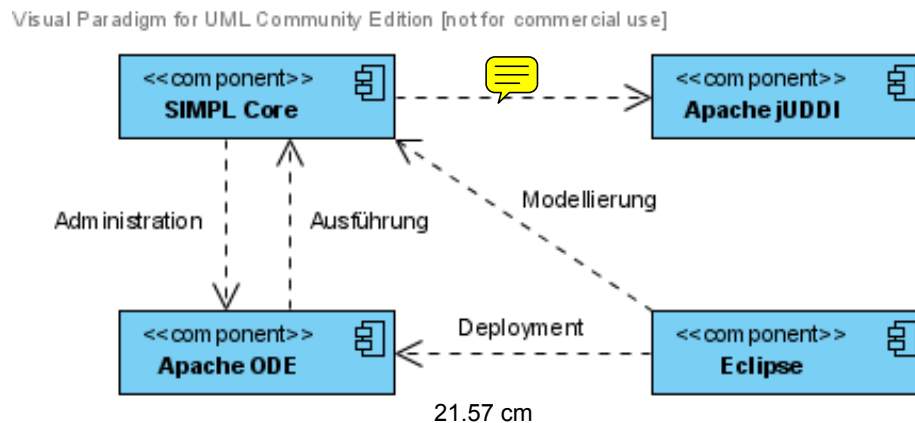


Abbildung 2: Komponenten des SIMPL Rahmenwerks

3.1 Eclipse

Die Entwicklungsumgebung Eclipse besteht im Rahmen des SIMPL Projektes aus der Eclipse Basis IDE sowie dem SIMPL Eclipse Plugin.

3.1.1 Eclipse BPEL Designer Plugin

Das BPEL Designer Plugin besteht aus folgenden Komponenten :

BPEL Modell

Ein Eclipse Modelling Framework (EMF) das die WS-BPEL 2.0 repräsentiert. Hier wird gemäß den BPEL Konventionen Java-Code erzeugt. Die Erweiterung der BPEL Funktionalitäten erfolgt durch entsprechend interpretierte Inhalte in den BPEL definierten **extension points**. Bei der Modellierung erfolgt bereits ein Zugriff auf den SIMPL Registry Service um die verfügbaren Datenquellen in das Modell einbinden zu können.

BPEL Designer GUI

Ein GEF-basierender (Graphical Editing Framework)Editor als grafisches Hilfsmittel um BPEL-Prozesse zu erstellen.

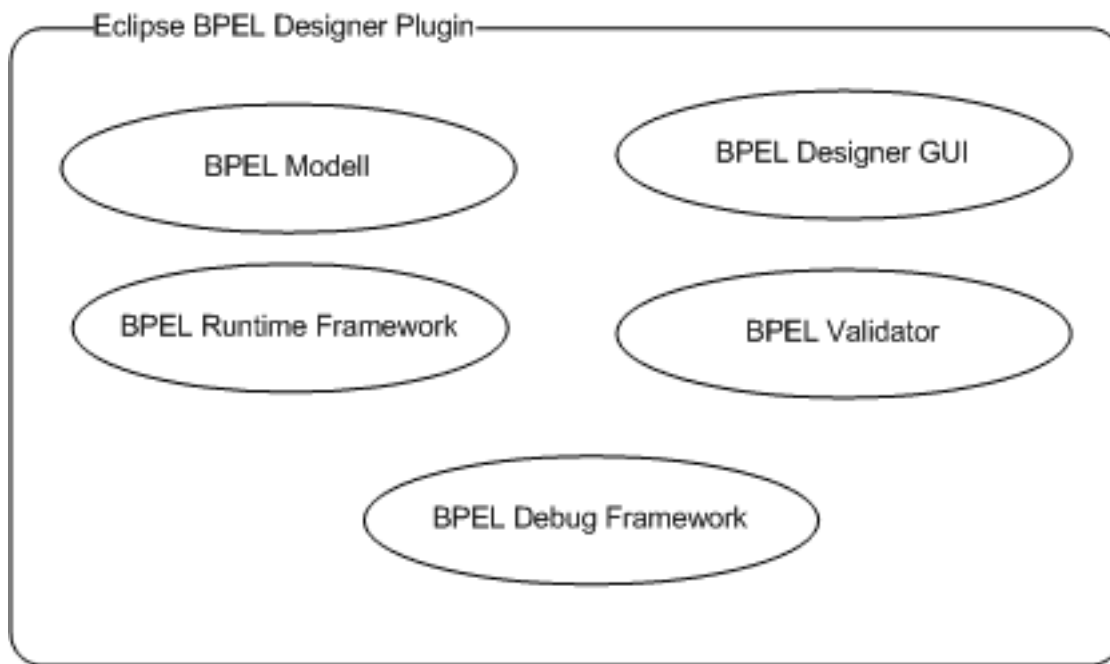


Abbildung 4: Eclipse BPEL Designer Plugin

BPEL Runtime Framework

Ein erweiterbares Rahmenwerk, welches das Deployment und die Ausführung von BPEL-Prozessen in eine BPEL Engine erlaubt. Für das SIMPL Rahmenwerk wird das Apache ODE Deployment- und Runtime Modell verwendet. Vor dem Deployment wird das erstellte Prozessmodell dem SIMPL Transformation Service gesendet. Dieser erstellt einen Standard-BPEL-Code, der direkt an die Workflow Engine gesendet wird. Zum Verarbeiten wird SIMPL Code erstellt, der vom SIMPL RRS aufgelöst und an die BPEL Engine gesendet wird. Durch diesen Ansatz wird die Ausführung des SIMPL Codes auf anderen Engines ermöglicht.

BPEL Validator

Der BPEL Validator arbeitet auf Grundlage des EMF-Modells und erzeugt bei Verletzung der WS-BPEL 2.0 Spezifikation Warnungen bzw. Fehler. Die Erweiterung der BPEL-Möglichkeiten im Rahmen von SIMPL erfolgt in den Extensions-Points und somit wird die Erweiterung WS-BPEL 2.0 konform sein und vom Validator nicht beanstandet.

BPEL Debug Framework

Das Debug Framework erlaubt dem Benutzer das schrittweise Ausführen eines Prozesses. Dabei wird auch das Setzen von Breakpoints erlaubt.

3.1.2 SIMPL Eclipse Plugin

Das SIMPL Eclipse Plug-In besteht aus der SIMPL Eclipse GUI sowie den SIMPL-DM-Aktivitäten.

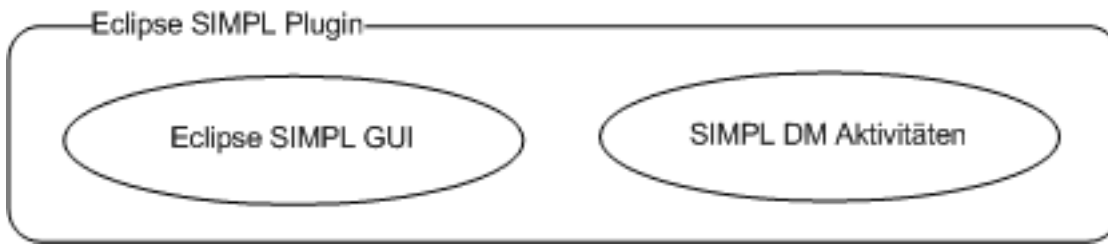


Abbildung 5: SIMPL Eclipse Plugin

SIMPLE Eclipse GUI

Die **SIMPLE** Eclipse GUI ermöglicht das einfache Ansprechen aller relevanten Funktionen des Rahmenwerkes über eine grafische Benutzeroberfläche. Diese Oberfläche wird durch Standard-Widget-Toolkit (SWT) realisiert und erfolgt nach dem Model-Viewer-Konzept (MVC). Durch die MVC Architektur wird die Präsentation der Funktionen vom Modell getrennt und ermöglicht so in späteren Entwicklungen das einfache Austauschen der Oberfläche (View).

Darzustellende Funktionen / Informationen in der GUI :

Admin Konsole Die Admin Konsole ermöglicht den Zugriff auf **Folgende** Funktionen / Optionen :

- Monitoring (Aktivieren / Deaktivieren)
- Globale Eigenschaften (Authentifizierung, Anmeldenname und Passwort)
- Referenzen (Eingabe **des** **Werts** Adresse sowie Auflistung der Referenzen mit der Möglichkeit zum Öffnen, Löschen oder Erstellen derselben)
- Datenquellen (Eingabe der Datenquellen-Registry, Anzeigen der Datenquellen sowie Festlegung der Attribute)
- Strategien (Auswahl der Strategie. **st** Find / Best Find / Matching Find)
- Auditing (Festlegung der Auditing **Datenquelle** wie Aktivierung und Deaktivierung)

Monitoring Festlegung der Granularität.

Settings Hier können Einstellungen getroffen werden **die** nicht systemkritisch sind und somit von jedem Benutzer verändert werden können.

Help Hilfestellungen zur Bedienbarkeit und Funktionen.

SIMPL Data-Management-Aktivitäten

Die Simpl-DM-Aktivitäten werden durch das SIMPL-DM-Plugin bereitgestellt und bei der Modellierung im BPEL Designer verwendet werden.

3.2 Apache ODE

Hier wird der Aufbau und die einzelnen Bestandteile von ODE **die** für das SIMPL Rahmenwerk von Bedeutung sind und ggf. erweitert werden müssen, erläutert. **im** folgenden Schaubild werden diese Bestandteile dargestellt.

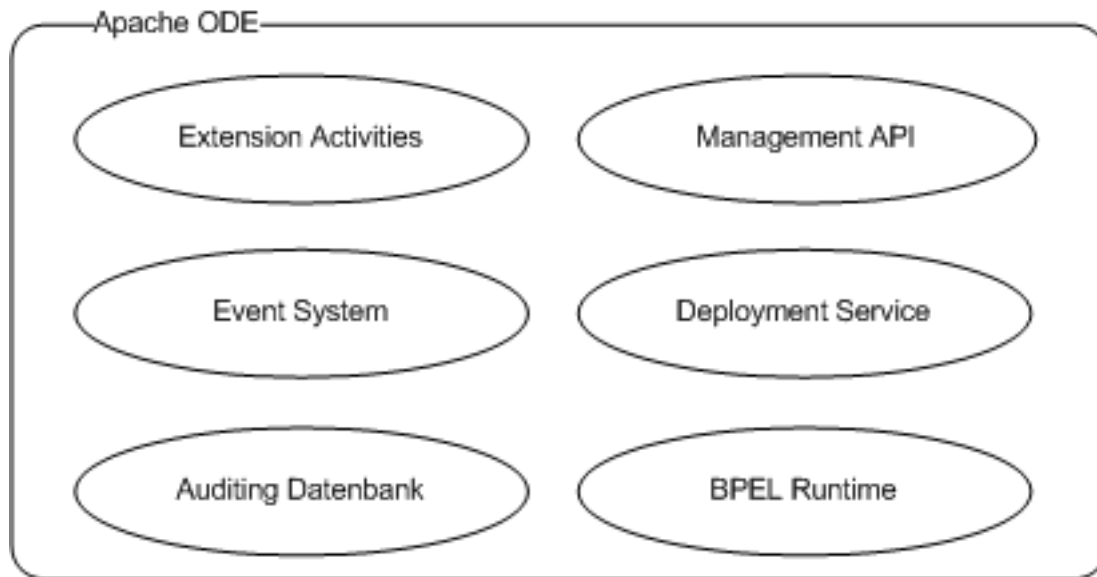


Abbildung 6: Aufbau von Apache ODE

3.2.1 Extension Activities

Extension Activities für die Apache ODE Workflow Engine werden als Java Klassen erstellt. Diese Klassen enthalten die Implementierung der neuen Aktivitäten. Um diese Aktivitäten in die ODE Engine hinzuzufügen, ist es notwendig diese als jar-Dateien in den Classpath zu kopieren. Anschließend ist es nur noch notwendig die Extension in der ODE Engine zu registrieren.

3.2.2 Event System

Es existieren bereits fünf Event-Typen innerhalb von Apache ODE. Dies sind:

- Instance life cycle events
- Activity life cycle events
- Scope handling
- Data handling
- Correlation events

Für das Auditing von SIMPL ist es notwendig dass eine Reihe von eigenen Events erstellt wird. Dies geschieht durch Erstellung neuer Event-Klassen, die von den bestehenden Event Klassen abgeleitet werden und anschließend innerhalb der Engine eingebunden werden.

3.2.3 Management API

Die Management API von Apache ODE bietet Funktionen auf Prozess- und auf Instanzebene an, dadurch können z.B. Prozesse gestartet oder beendet werden und Informationen zu den einzelnen Instanzen abgerufen werden.

Die Management API unterteilt sich in Instanz Management und Prozess Management. Beide bieten verschiedene Funktionen an um mit Instanzen oder Prozessen zu arbeiten. Es besteht auf der einen Seite die Möglichkeit verschiedene Informationen über die Prozesse und Instanzen zu erhalten,

zum Beispiel eine Liste der verschiedenen Prozesse oder Informationen über einen einzelnen Prozess, auf der anderen Seite ist es aber auch möglich direkt mit den Prozessen und Instanzen zu interagieren, so es beispielsweise möglich eine Prozessinstanz anzuhalten oder direkt zu beenden.

3.2.4 Auditing Datenbank

Apache ODE verfügt bereits über eine interne Derby-Datenbank. Es ist allerdings möglich die Auditing Daten mit Hilfe eines DAOs auf eine andere Datenbank umzubie Dies soll über den SIMPL Core realisiert werden, um die Anbindung einer beliebigen Datenbank zu ermöglichen.

3.2.5 Deployment Service

Der Deployment Service bietet verschiedene Funktionen für das Deployment von Prozessen an.

3.2.6 BPEL Runtime

Die BPEL Runtime ist für die eigentliche Ausführung der Prozesse zuständig. Bei Ausführung der DM-Aktivitäten wird auf den SIMPL Core zugegriffen um die entsprechenden DM-Operationen durchzuführen.

3.3 SIMPL Core

Der SIMPL Core stellt alle Funktionalität zur Verfügung, die während der Modellierung und der Ausführung von Prozessen benötigt werden. Er besteht aus mehreren Web Services (siehe 7), die jeweils fest definierte Aufgaben innerhalb des SIMPL Rahmenwerks haben. Die Web Services werden dabei innerhalb einer Apache Axis2 Installation bereitgestellt und mit der Java API JAX-WS erstellt, damit ein Einsatz auch mit anderen Web Containern, außer Apache Tomcat, gewährleistet ist (siehe Anforderung 1.3.9). Jeder dieser Web Services bietet nach Außen eine Schnittstelle. In den folgenden Abschnitten werden zunächst die Web Services und ihre Aufgaben näher beschrieben, Schnittstellen und ihre Verwendung sind dann Thema von Kapitel 4.

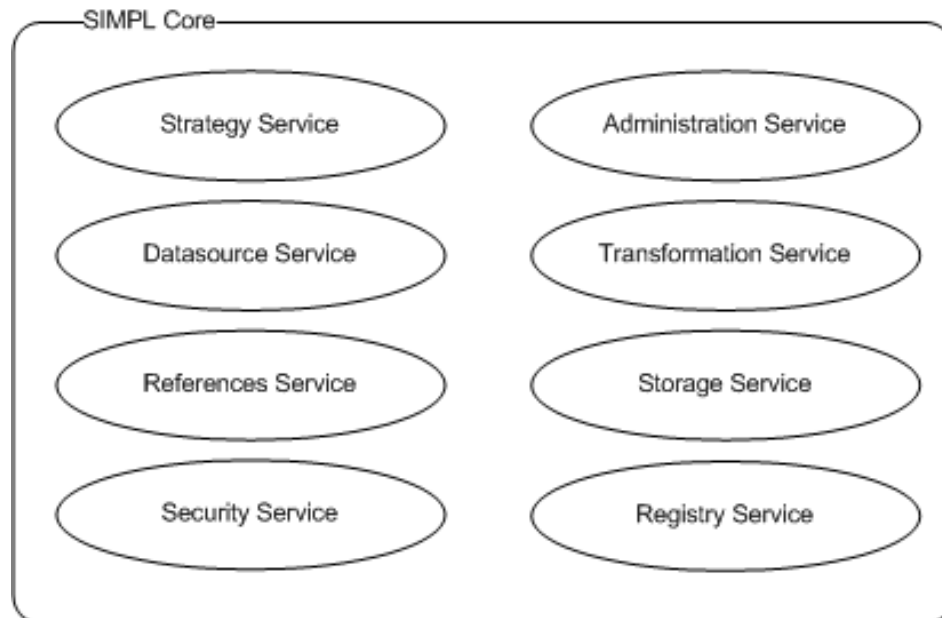



Abbildung 7: Aufbau der SIMPL Core

3.3.1 Strategy Service

Mit dem Strategy Service wird das Late Binding (siehe Anforderung 1.3.4) ermöglicht,  dort stehen verschiedene Auswahlstrategien bzw. -algorithmen zur Verfügung, um mit den im Prozess formulierten Anforderungen eine passende Datenquelle ausfindig zu machen.

3.3.2 Datasource Service

Der Datasource Service ist für alle Aufgaben zuständig, die den Zugriff auf die Datenquellen betreffen. Dort werden entsprechende Adapter (siehe 1.4.6) in Form von Direct Access Services (DAS) implementiert, die den Zugriff auf verschiedene Typen von Datenquellen ermöglichen und die Erweiterbarkeit für weitere Typen und Anfragesprachen (siehe Anforderung 1.3.8) garantieren. Des weiteren werden Plug-In-Schnittstellen (siehe 1.4.5) für zusätzliche funktionale Erweiterungen geschaffen, wie z.B. die Unterstützung verschiedener Dateitypen bei Dateisystemen.

3.3.3 Reference Service

Der Reference Service erfüllt die Anforderung, große Datenmengen in BPEL referenzieren zu können (siehe Anforderung 1.3.2). Über diesen Service lassen sich die Referenzen verwalten und auflösen, um die referenzierten Daten bei Bedarf in einen Prozess zu holen.

3.3.4 Security Service

Der Security Service ist zuständig für Aufgaben, die die Authentifizierung und Autorisierung gegenüber Datenquellen betreffen. Mit diesem Service können Authentifizierungs- und Autorisierungsinformationen verarbeitet und ggf. transformiert werden. Die Informationen können außerdem zwischengespeichert werden um das Konzept des SSO (siehe Anforderung 1.3.5) zu realisieren.


3.3.5 Administration Service

Über den Administration Service werden alle Einstellungen des Rahmenwerks verwaltet und die Funktionalität für die Admin-Konsole (siehe Anforderung 1.3.7) bereitgestellt. Für Einstellungen, mit denen zur Laufzeit Einfluss auf Apache ODE genommen wird, wie z.B. die Granularität des Auditing, werden die vorhandenen Schnittstellen von Apache ODE verwendet und, falls nötig, neue Schnittstellen geschaffen. Die geforderte austauschbare GUI (siehe Anforderung 1.3.8), wird durch die unabhängige WSDL-Schnittstelle erreicht.

3.3.6 Transformation Service

Damit modellierte Prozesse, die DM-Aktivitäten mit Referenzen enthalten, auch auf anderen Workflow Engines ausgeführt werden können (siehe Anforderung 1.3.9), wird der Transformation Service bereitgestellt, mit dem der erweiterte BPEL Code in Standard-BPEL-Code transformiert werden kann.

3.3.7 Storage Service

Der Storage Service bildet den zentralen Speicherort von Daten des SIMPL Rahmenwerks,  dort laufen alle Einstellungen, Daten und Informationen zusammen und werden persistent in einer Datenquelle gespeichert. Beispielsweise werden dort die Referenzen gespeichert, die über den Reference Service hinzugefügt werden.

3.3.8 Registry Service

Über den Registry Service kann auf die Registry des Rahmenwerks zugegriffen werden, in der die vorhandenen Datenquellen mit ihren Eigenschaften zentral verwaltet werden. (siehe Anforderung 1.3.6)

3.4 Apache jUDDI

Das SIMPL Rahmenwerk stellt mit Apache jUDDI bereits eine UDDI-Registry zur Verfügung, die mit den Registry Service genutzt werden kann. Über die WSDL Schnittstelle des Registry Service kann aber auch jede andere Implementierung einer Registry realisiert werden (siehe Anforderung 1.3.8).

4 Schnittstellen

In diesem Kapitel wird die Kommunikation der Komponenten über die Schnittstellen beschrieben und interne und externe Abhängigkeiten, sowie eingesetzte Protokolle und übertragene Objekte erläutert. Abbildung 8 zeigt dazu die Schnittstellen in einer Übersicht, die in den folgenden Abschnitten näher beschrieben werden. Bei allen Schnittstellen handelt es sich um WSDL-Schnittstellen, die an das Übertragungsprotokoll SOAP gebunden sind.

Visual Paradigm for UML Community Edition [not for commercial use]

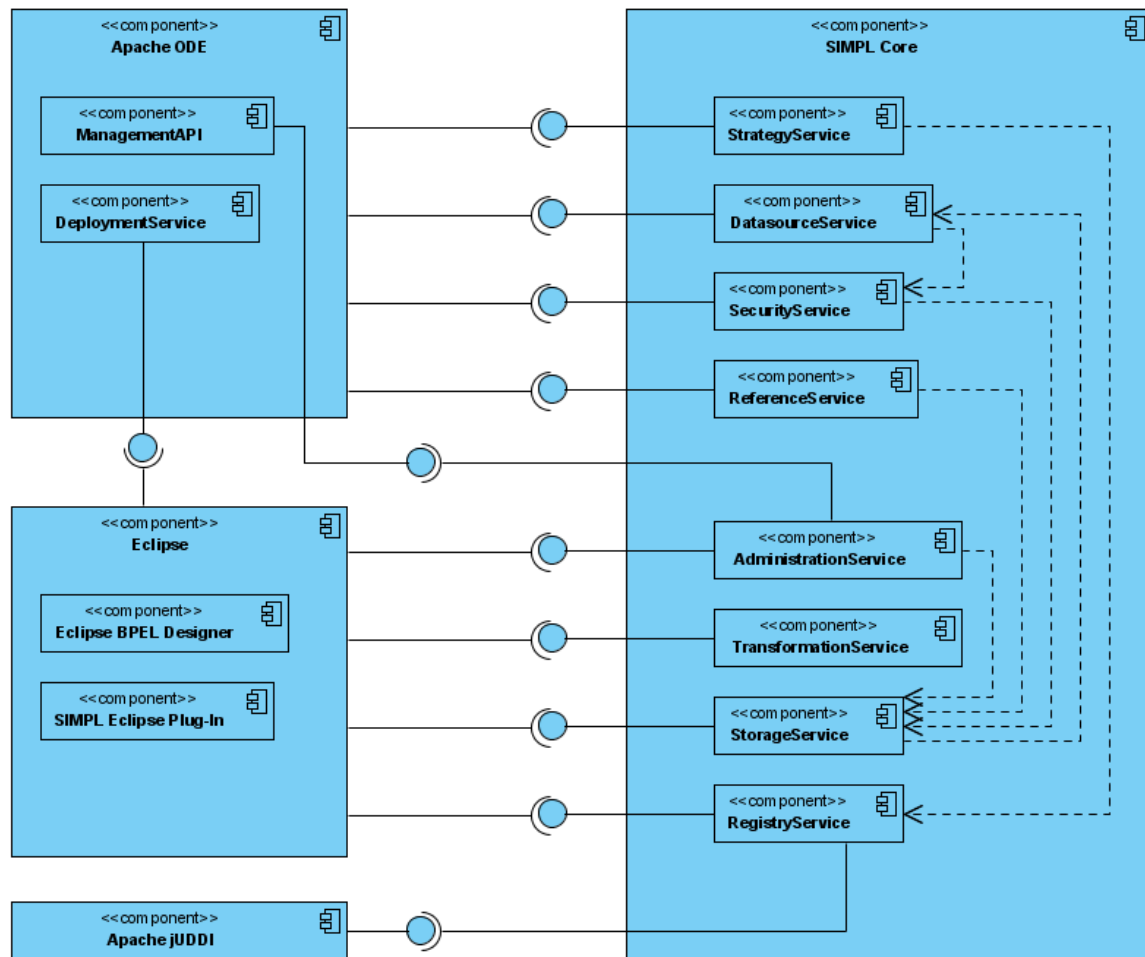


Abbildung 8: Schnittstellen des SIMPL Rahmenwerks

4.1 Apache ODE

Hier werden die bestehenden Schnittstellen von Apache ODE und ihre Verwendung von den Komponenten beschrieben.

4.1.1 ManagementAPI

Der Administration Service verwendet die ManagementAPI-Schnittstelle für die Einstellungen des Rahmenwerks, die zur Laufzeit gemacht werden.

4.1.2 DeploymentService

Die DeploymentService-Schnittstelle wird von Eclipse für das Deployment von Prozessen verwendet.

4.2 Apache jUDDI

Auf Apache jUDDI wird vom Rahmenwerk nur über den Registry Service benutzt.

4.3 SIMPL Core

Hier werden die Schnittstellen vom SIMPL Core und ihre Verwendung von den Komponenten beschrieben. Alle internen Abhängigkeiten des SIMPL Core werden ebenfalls über diese Schnittstellen realisiert.

4.3.1 DatasourceService

Die DatasourceService-Schnittstelle wird von Apache ODE bei dem Zugriff auf Datenquellen und für das Speichern der Auditing-Daten verwendet. Wenn die Schnittstelle vom Storage Service für die Datenhaltung genutzt wird, wird die Schnittstelle vom Storage Service für die Datenhaltung genutzt. Sie bietet Funktionen zum Abrufen von Daten von Datenquellen und Senden von Befehlen in unterstützten Anfragesprachen. Die Daten werden dabei als Service Data Object (SDO) übertragen.

4.3.2 StrategyService

Die StrategyService-Schnittstelle wird von Apache ODE für das Late Binding genutzt und bietet Funktionen für die verschiedenen Auswahlstrategien. Die Anforderungen an die Datenquellen werden dabei als WS-Policy-Objekte übertragen.

4.3.3 ReferenceService

Die ReferenceService-Schnittstelle wird bei der Modellierung in Eclipse zur Verwaltung von Referenzen genutzt und von Apache ODE zur Auflösung der Referenzen zur Laufzeit verwendet. Sie bietet Funktionen zur Verwaltung und Auflösung von Referenzen, wobei die Daten der aufgelösten Referenzen als SDO zurückgeliefert werden.

4.3.4 SecurityService

Die SecurityService-Schnittstelle bietet Funktionen für die Verwaltung und Transformation von Authentifizierungs- und Autorisierungsinformationen. Sie wird vom Datasource Service benötigt um entsprechende Informationen für einen Zugriff auf eine Datenquelle abzurufen. Die Informationen werden dabei unter anderem in der Security Assertion Markup Language (SAML) und der eXtensible Access Control Markup Language (XACML) übertragen.

4.3.5 AdministrationService

Die AdministrationService-Schnittstelle bietet Funktionen mit denen Einstellungen zur Prozess-Laufzeit gemacht werden können und bietet außerdem Funktionen für die Verwaltung der Einstellungen der Admin-Konsole. Die Einstellungen der Admin-Konsole werden dabei als SDO übertragen.

4.3.6 TransformationService

Die TransformationService-Schnittstelle bietet eine Funktion zur Umwandlung von, mit DM-Aktivitäten, erweitertem BPEL-Code in Standard-BPEL-Code. Dazu werden die entsprechenden BPEL-Dateien übergeben und transformiert zurückgeliefert.

4.3.7 StorageService

Die StorageService-Schnittstelle bietet Funktionen zur Verwaltung von Daten innerhalb des Rahmens und wird vom Reference Service, Administration Service und SecurityService verwendet. Zusätzlich bietet sie dem Administration Service die Möglichkeit eine Datenquelle für die Speicherung festzulegen. Alle Daten werden als SDO übertragen.

4.3.8 RegistryService

Die RegistryService-Schnittstelle bietet Funktionen für die Verwaltung von Datenquellen und ihrer Eigenschaften. Der StrategyService bekommt dadurch Zugriff auf die vorhandenen Datenquellen, die von Eclipse, ebenfalls über die Schnittstelle verwaltet werden. Die Datenquellen werden dabei als WSDL-Objekte und ihre Eigenschaften als WS-Policy-Objekte übertragen.

Literatur

- [1] Knauß, Markus (März 2008): *Entwurfsvorlage*, <http://www.iste.uni-stuttgart.de/se/>.
- [2] Ludewig, Jochen; Lichter, Horst (2007): *Software Engineering: Grundlagen, Menschen, Prozesse und Techniken*, punkt.verlag GmbH.
- [3] Stupro-A SIMPL: *Spezifikation* (2009), <http://code.google.com/p/simpl09/>.

Abkürzungsverzeichnis

API	Application Programming Interface
BPEL	Business Process Execution Language
DAS	Data Access Service
DM	Data-Management
GUI	Graphical User Interface
JAX-WS	Java API for XML - Web Services
ODE	Orchestration Director Engine
RRS	Reference Resolution System
SAML	Security Assertion Markup Language
SDA	Service Data Object
SIMPL	SimTech: Information Management, Processes and Languages
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSO	Single Sign On
UDDI	Universal Description, Discovery and Integration
WS	Web Service
WSDL	Web Service Description Language
XACML	eXtensible Access Control Markup Language
XQuery	XML Query Language

Abbildungsverzeichnis

1	Architektur des SIMPL Rahmenwerks	10
3	Eclipse Komponenten-Diagramm	11
2	Komponenten des SIMPL Rahmenwerks	11
4	Eclipse BPEL Designer Plugin	12
5	SIMPL Eclipse Plugin	13
6	Aufbau von Apache ODE	14
7	Aufbau der SIMPL Core	15
8	Schnittstellen des SIMPL Rahmenwerks	18