

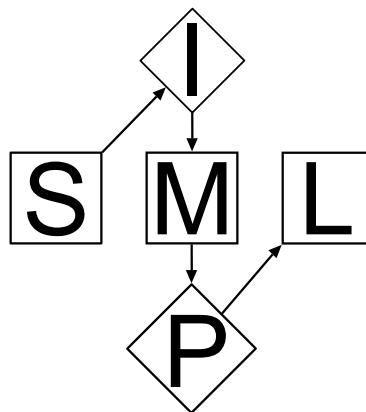


Studienprojekt SIMPL

Grobentwurf

Version 1.0

11. November 2009



Inhaltsverzeichnis

1	Einleitung	5
1.1	Zweck des Dokuments	5
1.2	Das SIMPL Rahmenwerk	5
1.3	Anforderungen	5
1.3.1	Einfache Modellierung	5
1.3.2	Große Datenmengen	5
1.3.3	Auditing von Prozessen	5
1.3.4	Late Binding von Datenquellen	6
1.3.5	Authentifizierung und Autorisierung	6
1.3.6	Registry	6
1.3.7	Admin-Konsole	6
1.3.8	Erweiterbarkeit	6
1.3.9	Verwendbarkeit	6
1.4	Entwurfsprinzipien	7
1.4.1	Offenes Rahmenwerk	7
1.4.2	Modularisierung	7
1.4.3	Kopplung und Zusammenhalt	7
1.4.4	Entwicklungsrichtung	7
1.4.5	Plug-In	7
1.4.6	Adapter	7
1.5	Überblick über den Grobentwurf	7
2	Architektur	9
2.1	Apache Tomcat	9
2.2	Eclipse	9
3	Komponenten	10
3.1	Eclipse	10
3.1.1	Eclipse BPEL Designer Plug-In	11
3.1.2	SIMPL Eclipse Plug-In	12
3.2	Apache ODE	13
3.2.1	Extension Activities	13
3.2.2	Event System	13
3.2.3	Management API	14
3.2.4	Auditing Datenbank	14
3.2.5	Deployment Service	14
3.2.6	BPEL Runtime	14
3.3	SIMPL Core	14
3.3.1	Strategy Service	15
3.3.2	Datasource Service	15
3.3.3	Reference Service	15
3.3.4	Security Service	15
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	16

4 Schnittstellen	17
4.1 Apache ODE	17
4.1.1 ManagementAPI	17
4.1.2 DeploymentService	18
4.2 Apache jUDDI	18
4.3 SIMPL Core	18
4.3.1 DatasourceService	18
4.3.2 StrategyService	18
4.3.3 ReferenceService	18
4.3.4 SecurityService	18
4.3.5 AdministrationService	18
4.3.6 TransformationService	19
4.3.7 StorageService	19
4.3.8 RegistryService	19
Literaturverzeichnis	20
Abkürzungsverzeichnis	21
Abbildungsverzeichnis	22

Änderungsgeschichte

Version	Datum	Autor	Änderungen
0.1	29.10.2009	schneimi	Erstellung des Dokuments
0.2	30.10.2009	schneimi	Kapitel 1 erstellt
0.3	04.11.2009	schneimi	Kapitel 2, Kapitel 3.3 erstellt
0.4	07.11.2009	schneimi	Kapitel 4 erstellt
0.5	09.11.2009	rehnre	Kapitel 3.2 erstellt
0.6	09.11.2009	bruededl	Kapitel 3.1 erstellt
1.0	11.11.2009	schneimi	Korrektur nach internem Review

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 und ihre Schnittstellen identifiziert, sowie ihr Zusammenspiel beschrieben. Damit soll gezeigt werden, dass das resultierende System funktionieren kann und den Anforderungen gerecht wird.

1.2 Das SIMPL Rahmenwerk

Das SIMPL Rahmenwerk soll dem Benutzer eine erweiterbare Umgebung bieten, die eine einfache Modellierung von **BPEL-Geschäftsprozessen** mit generischem 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. Diese Aktivitäten werden im folgenden Verlauf des Dokuments als Data-Management-Aktivitäten bzw. DM-Aktivitäten bezeichnet. Die Ausführung der Prozesse erfolgt durch die Apache ODE Workflow Engine, bei der das bestehende Event-Modell und das Auditing der Prozessdaten für die DM-Aktivitäten angepasst werden. Dienste, die für die Ausführung der DM-Aktivitäten von der Workflow Engine benötigt werden, werden in Form von Web Services bereitgestellt.

1.3 Anforderungen

In diesem Abschnitt werden die Anforderungen des SIMPL Rahmenwerks beschrieben.

1.3.1 Einfache Modellierung

Neben der vereinfachten Modellierung durch die DM-Aktivitäten, gibt es eine weitere Anforderung. Während der Modellierung von Prozessen wiederholen sich in der Regel häufig längere Befehle (Statements) in Anfragesprachen wie SQL oder XQuery und 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. Diese BPEL-Variablen können anschließend als Referenzen in anderen Befehlen verwendet werden.

1.3.2 Große Datenmengen

Der Schwerpunkt des Rahmenwerks liegt in der Modellierung von wissenschaftlichen Prozessen, bei denen überwiegend mit großen Datenmengen gearbeitet wird. Damit diese Datenmengen nicht innerhalb des Prozesses gehalten werden müssen, wird ein Reference Resolution System (RRS) realisiert. Damit wird es möglich Daten zu referenzieren, sie per Referenz sehr schnell weiterzugeben und nur bei Bedarf aufzulösen und in den Prozess zu holen.

1.3.3 Auditing von Prozessen

Bei dem Auditing von Prozessen handelt es sich um das Protokollieren von Prozessdaten, wie z.B. Variablenstatus oder aufgetretene Events, die unter Anderem ein Monitoring der Prozesse ermöglichen. Apache ODE besitzt bereits ein internes Auditing bei dem Prozessdaten persistent gespeichert werden.

Das interne Auditing muss für die zusätzlichen DM-Aktivitäten angepasst werden und der Benutzer soll außerdem die Möglichkeit bekommen, die Granularität der Protokollierung zu bestimmen. Zusätzlich soll die interne Datenspeicherung auf eine beliebige externe Datenbank umgeleitet werden können.

1.3.4 Late Binding von Datenquellen

Da bei der Modellierung nicht immer feststeht auf welche Datenquelle zugegriffen wird, beispielsweise beim Ablegen von Daten, soll das 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 automatisch 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. Benutzername und Passwort nicht bei jedem 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, ist eine zentrale Registry vorgesehen, in der die Datenquellen von Datenquellen-Administratoren 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 Aktualisieren von Referenzen.

1.3.8 Erweiterbarkeit

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

- Anbindung weiterer Datenquellentypen
- Unterstützung weiterer Anfragesprachen
- Erweiterung um neue DM-Aktivitäten
- Erweiterung um neue Events für das Auditing
- Unterstützung weiterer Authentifizierungs- und Autorisierungsverfahren
- Unterstützung weiterer Registries
- Anbindung zusätzlicher Graphical User Interfaces (GUI) für die Admin-Konsole

1.3.9 Verwendbarkeit

Die Aktivitäten, die für SIMPL entwickelt werden, sollen auf beliebigen anderen BPEL Workflow-Engines ausführbar sein. Dazu muss der, von den Aktivitäten bei der Modellierung erzeugte, erweiterte BPEL-Code, in standard-konformen BPEL-Code 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 für den Entwurf angewendet werden. (vgl. [2], Kapitel 17)

1.4.1 Offenes Rahmenwerk

Das SIMPL Rahmenwerk ist größtenteils ein offenes Rahmenwerk. Offen bedeutet, es gibt keine einheitliche Schnittstelle für alle Erweiterungen. 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, mit denen die Umsetzung eigener Erweiterungen erleichtert wird. 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

Beim Entwurf der Module wird darauf geachtet, dass die Kopplung zu anderen Modulen möglichst gering ist und der Zusammenhalt innerhalb eines Moduls möglichst hoch ist. 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

Die Entwicklung wird Top-down durchgeführt. 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, 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 die Architektur des SIMPL Rahmenwerks. Bei den farblich gekennzeichneten Komponenten, handelt es sich um Komponenten, die für das Rahmenwerk neu erstellt werden. Die übergeordneten Komponenten, Apache Tomcat und Eclipse, sowie ihr Zusammenspiel, werden in den folgenden Abschnitten kurz beschrieben. Auf die einzelnen elementaren Komponenten wird dann in Kapitel 3 eingegangen.

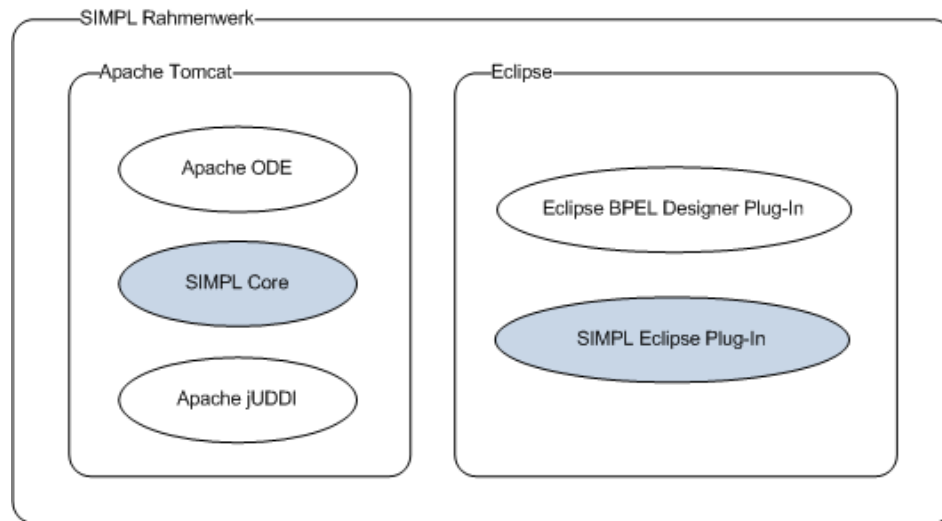


Abbildung 1: Architektur des SIMPL Rahmenwerks

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, dem Rahmenwerk zur Verfügung stehenden, Datenquellen von Datenquellen-Administratoren 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 zum Einen die bestehenden Aktivitäten des Eclipse BPEL Designer Plug-Ins um die DM-Aktivitäten. Zum Anderen bietet es die GUI mit den Einstellungen für den Workflow-Administrator, wie z.B. die Admin-Konsole und die Globalen Einstellungen (siehe [5], Kapitel 4).

3 Komponenten

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

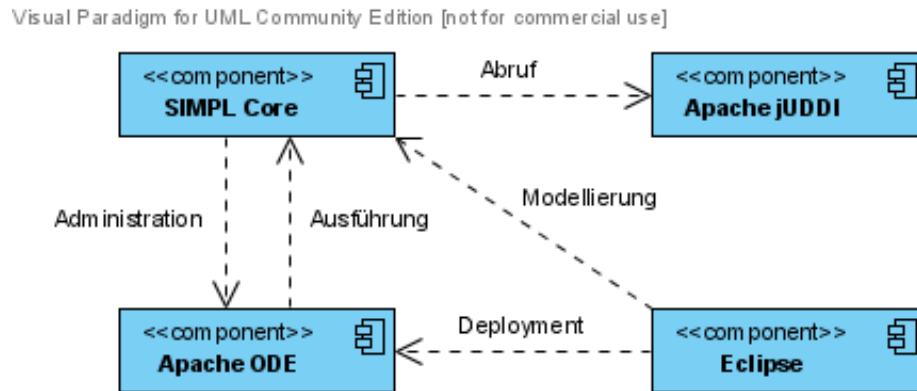


Abbildung 2: Komponenten des SIMPL Rahmenwerks

3.1 Eclipse

Die Entwicklungsumgebung Eclipse teilt im Rahmen des SIMPL Projektes aus der Eclipse Basis IDE sowie dem SIMPL Eclipse Plug-In.

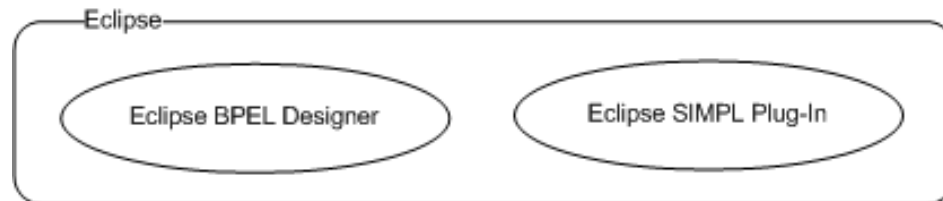


Abbildung 3: Eclipse Komponenten-Diagramm

3.1.1 Eclipse BPEL Designer Plug-In

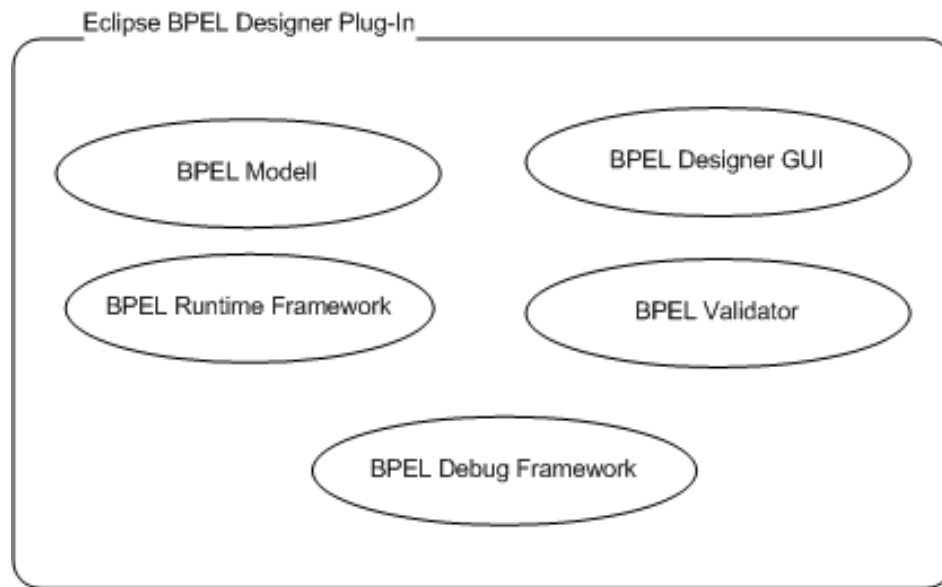


Abbildung 4: Eclipse BPEL Designer Plug-In

Das BPEL Designer Plug-In besteht aus folgenden Komponenten:

BPEL Modell

Die WS-BPEL 2.0 Spezifikation [3] wird durch ein entsprechendes Eclipse Modelling Framework (EMF) Modell repräsentiert. Aus diesem Modell kann gemäß der BPEL Konventionen Java-Code erzeugt werden. Das BPEL Modell kann über entsprechende Erweiterungspunkte (Extension Points) erweitert werden.

BPEL Designer GUI

Die BPEL Designer Oberfläche besteht aus einem, über das Graphical Editing Framework (GEF) realisierten, Editor für die grafische und textuelle Modellierung von BPEL-Prozessen.

BPEL Runtime Framework

Das BPEL Runtime Framework ist ein erweiterbares Rahmenwerk, welches das Deployment und die Ausführung von BPEL-Prozessen in Apache ODE ermöglicht. Für das SIMPL Rahmenwerk wird das Apache ODE Deployment- und Runtime-Modell verwendet. Vor dem Deployment wird das erstellte Prozessmodell über den SIMPL Core in ein BPEL standard-konformes Prozessmodell transformiert und steht anschließend für das Deployment zur Verfügung. Durch diesen Ansatz wird die Ausführung von erweiterten Prozessmodellen auf anderen BPEL Workflow-Engines ermöglicht. (siehe Anforderung 1.3.9)

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 über **Extensions-Points**, somit ist die Erweiterung WS-BPEL 2.0 konform und wird 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 Plug-In**

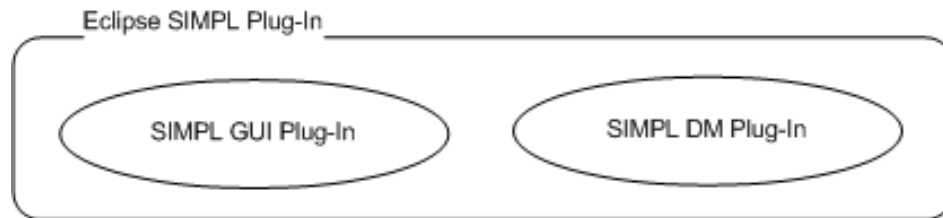


Abbildung 5: **SIMPL Eclipse Plug-In**

Das SIMPL Eclipse Plug-In besteht aus dem SIMPL GUI Plug-In sowie dem SIMPL DM Plug-In.

SIMPL GUI Plug-In

Die SIMPL Eclipse GUI ermöglicht das einfache Ansprechen aller relevanten Funktionen des Rahmenwerkes über eine grafische Benutzeroberfläche. Diese Oberfläche wird mit dem Standard-Widget-Toolkit (SWT) realisiert und erfolgt nach dem Model-View-Controller (MVC) Konzept. Durch die MVC-Architektur wird die Präsentation der Funktionen vom Modell getrennt und ermöglicht so, in späteren Entwicklungen, das einfache Austauschen der Oberfläche (View). Für Informationen und administrative Einstellungen, muss das Plug-In bei der Modellierung auf den SIMPL Core zurückgreifen.

Darzustellende Informationen in der GUI :

Admin Konsole Die Admin Konsole ermöglicht den Zugriff auf folgende Einstellungen:

- Globale Eigenschaften (Authentifizierung, Anmeldename und Passwort)
- Referenzen (Eingabe der RRS-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 wie z.B. First Find, Best Find oder Matching Find)
- Auditing (Festlegung der Auditing Datenbank sowie Aktivierung und Deaktivierung)

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

Help Hilfestellungen zur Bedienbarkeit und Funktionen.



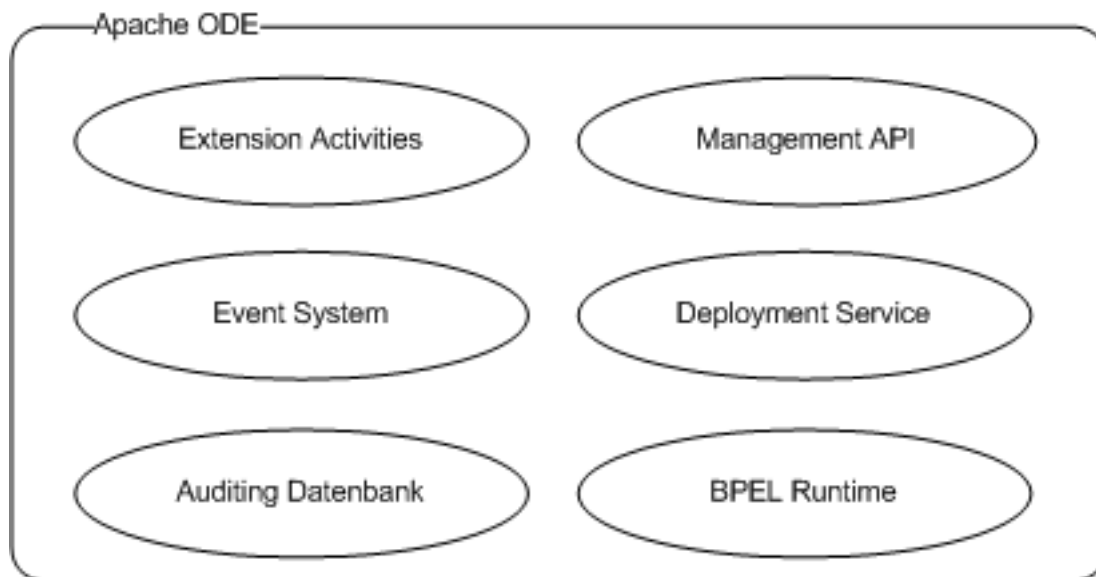


Abbildung 6: Aufbau von Apache ODE

SIMPL DM Plug-In

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

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.


3.2.1 Extension Activities

Apache ODE unterstützt Extension Activities, um die elementaren Aktivitäten von BPEL zu erweitern. Die Extension Activities für die Apache ODE Workflow Engine werden als Java Klassen erstellt. Diese Klassen enthalten die Implementierung der neuen Aktivitäten. Um diese Aktivitäten der 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  bestehenden Event Klassen abgeleitet 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 für Instanzen bzw. Prozessen an. Es besteht auf der einen Seite die Möglichkeit, verschiedene Informationen über die Prozesse und Instanzen zu erhalten, zum Beispiel eine Liste der verschiedenen  zesse oder Informationen über einen einzelnen  zess, auf der anderen Seite ist es aber auch möglich direkt mit ~~den~~ Prozessen und Instanzen zu interagieren, dadurch ist es beispielsweise möglich eine Prozessinstanz anzuhalten oder direkt zu beenden.

3.2.4 Auditing Datenbank

Apache ODE verfügt bereits über eine interne Apache Derby Datenbank. Es ist allerdings möglich die Auditing Daten mit Hilfe eines Data Access Objects (DAO) auf eine andere Datenbank umzuleiten. Dies soll über den SIMPL Core realisiert werden, um die Anbindung einer beliebigen Datenbank zu ermöglichen. (siehe Anforderung 1.3.3)

3.2.5 Deployment Service

Der **Deployment Service** bietet verschiedene Funktionen für das Deployment von Prozessen an. Dazu gehören neben einer Deploy- und Undeploy-Funktion, unter Anderem auch das **Auflisten von deployten Prozessen**. 

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 Abbildung 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. Die Schnittstellen und ihre Verwendung ~~sind dann Thema von~~ Kapitel 4.

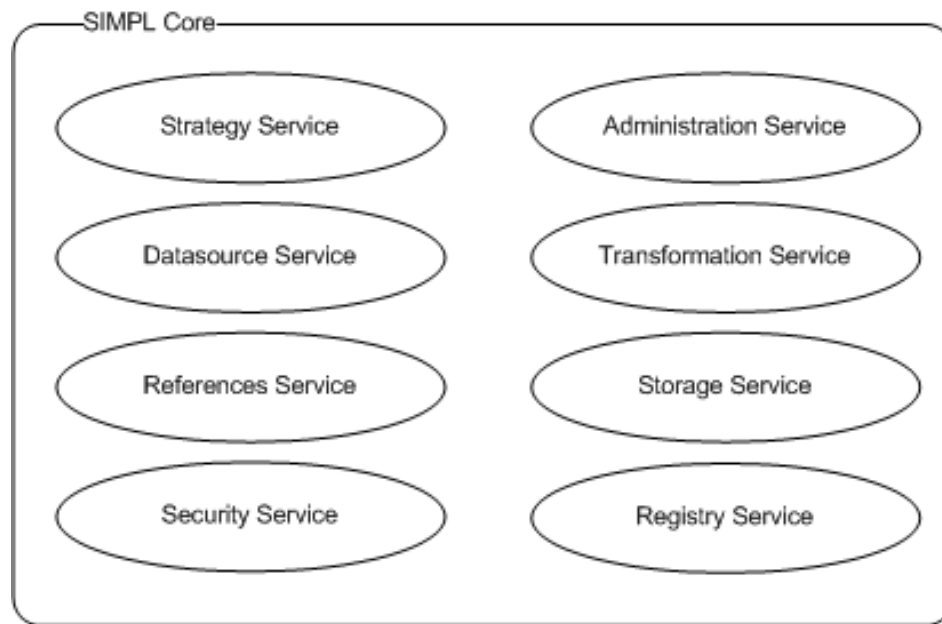



Abbildung 7: Aufbau des SIMPL Core

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. in benötigte Formate 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 enthalten, auch auf anderen Workflow Engines ausgeführt werden können (siehe Anforderung 1.3.9), der Transformation Service bereitgestellt, mit dem der erweiterte BPEL-Code in standard-konformen BPEL-Code transformiert werden kann.

3.3.7 Storage Service

Der Storage Service bildet den zentralen Speicherdienst für Daten des SIMPL Rahmenwerks. Dort laufen alle Einstellungen, Daten und Informationen zusammen und werden persistent in einer Datenquelle gespeichert. Durch diesen Service werden beispielsweise die Einstellungen des SIMPL Rahmenwerks aus der Admin-Konsole gespeichert.

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 dem Registry Service genutzt werden kann. Über die WSDL-Schnittstelle des Registry Service kann aber auch jede andere Implementierung, oder Anbindung einer anderen 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]

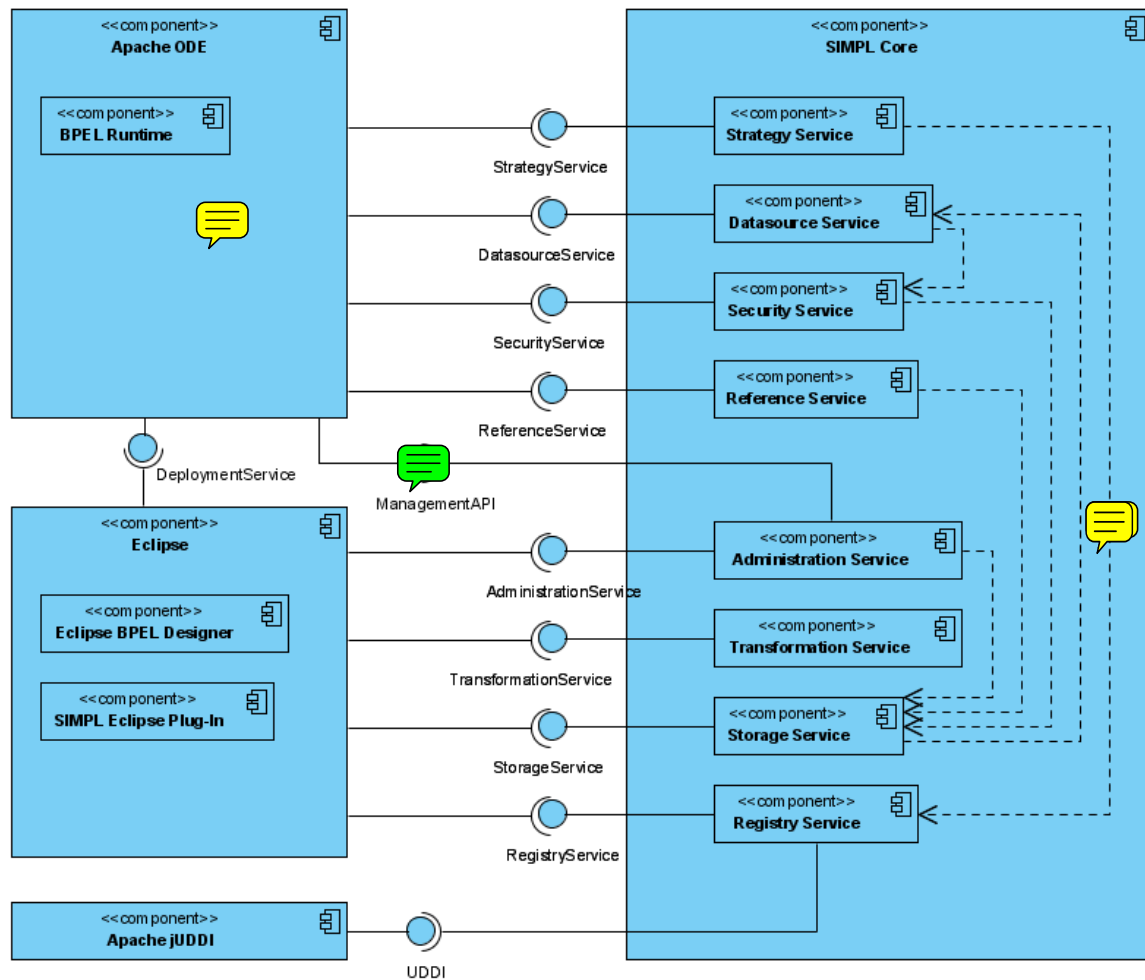


Abbildung 8: Schnittstellen des SIMPL Rahmenwerks

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 die Apache jUDDI Registry wird nur durch den Registry Service über die UDDI-Schnittstelle zugegriffen. Für Eclipse stehen Funktionen für das Auslesen zur Verfügung, während die Datenquellen-Administratoren darüber auch die Möglichkeit haben, Datenquellen und ihre Eigenschaften hinzuzufügen, zu ändern und zu löschen.

4.3 SIMPL Core

Hier werden die Schnittstellen des SIMPL Cores 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. Intern 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) [4] 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 des durch die DM-Aktivitäten erweiterten BPEL-Codes in standard-konformen BPEL-Code. Dazu werden die entsprechenden BPEL-Dateien übergeben und entsprechend transformierte Kopien der Prozessmodelle 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 ihren Eigenschaften in der angebundenen Registry. Der StrategyService, sowie Eclipse, bekommen dadurch Zugriff auf die vorhandenen Datenquelle, die über diese Schnittstelle abgerufen werden.

Literatur

- [1] Knauß, Markus: *Entwurfsvorlage*. Institut für Softwaretechnologie, März 2008.
- [2] Ludewig, Jochen; Lichter, Horst: *Software Engineering*. Grundlagen, Menschen, Prozesse und Techniken. dpunkt.verlag GmbH, 2007.
- [3] Alves, Alexandre; Arkin, Assaf; Askary, Sid; Barreto, Charlton; Bloch, Ben; Curbera, Francisco; Ford, Mark; Goand, Yaron; Gort, Alejandro; Kartha, Neelakantan; Liu, Canyang Kevin; Khalaf, Rania; König, Dieter; Marin, Mike; Mehta, Vinkesh; Thatte, Satish; van der Rijn, Dapeng; Yendluri, Prasad; Yiu, Alex: *Web Services Business Process Execution Language Version 2.0*. Version: 11 April 2007. <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf>, Abruf: 11.11.2009
- [4] Adams, Matthew; Andrei, Cezar; Barack, Ron; Blohm, Henning; Boutard, Christophe; Brodsky, Stephen; Budinsky, Frank; Bünnig, Stefan; Carey, Michael; Doughan, Blaise; Grove, Andy; Halaseh, Omar; Harris, Larry; von Mersewsky, Ulf; Moe, Shawn; Nally, Martin; Preotiuc-Pietro, Radu; Rowley, Mike; Samson, Eric; Taylor, James; Thiefaine, Arnaud: *Service Data Objects For Java*. Version: 2.1.0, November 2006. <http://www.osoa.org/download/attachments/36/Java-SDO-Spec-v2.1.0-FINAL.pdf>, Abruf: 11.11.2009
- [5] *Spezifikation v1.1*. S-A (2009).

Abkürzungsverzeichnis

API	Application Programming Interface
BPEL	Business Process Execution Language
DAO	Data Access Object
DAS	Data Access Service
DM	Data-Management
GEF	Graphical Editing Framework
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
SDO	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	9
2	Komponenten des SIMPL Rahmenwerks	10
3	Eclipse Komponenten-Diagramm	10
4	Eclipse BPEL Designer Plug-In	11
5	SIMPL Eclipse Plug-In	12
6	Aufbau von Apache ODE	13
7	Aufbau des SIMPL Core	15
8	Schnittstellen des SIMPL Rahmenwerks	17