ERLEBEN, WAS VERBINDET.

T-Systems Multimedia Solutions GmbH

WSO2-PRAKTIKUM

AUFGABEN IM RAHMEN DER SOA-ÜBUNG IM WS 2013/2014

IMPRESSUM

Dr. Stefan Pietschmann
T-Systems Multimedia Solutions GmbH
Portal-Technologies, Applications and Appliances
Service-Oriented Enterprise Applications

Riesaer Str. 5
01129 Dresden

Ansprechpartner	Telefon / Fax	E-Mail
Stefan Pietschmann	0351 2820-5436	Stefan.Pietschmann@t-systems.com
Robin Lutter	0351 2820-2658	Robin.Lutter@t-systems.com

Kurzinfo
Dieses Dokument gibt einen Überblick über die Aufgaben im Rahmen der Übung zur Vorlesung „Entwicklung verteilter Systeme auf Basis Serviceorientierter Architekturen“ im WS 2013/14.

INHALTSVERZEICHNIS

1. EINLEITUNG

Ziel der Übung ist es, mit Hilfe der WSO2 Middleware Plattform[1] ein Verständnis für die Aufgaben, Funktionsweisen und für die praktische Integration der verschiedenen Teile einer SOA-Architektur zu erhalten.
Neben einigen grundlegenden Aufgaben – im Wesentlichen die Erstellung eines Dienstes, eines zugehörigen Clients und der Realisierung des Dienstaufrufes – sollen sich vier verschiedene Teams im Rahmen der Übung jeweils mit spezifischen Aufgaben einer SOA beschäftigen. Diese werden in Kapitel 4 genauer beschrieben. Alle entstehenden Beispielanwendungen sollen sich thematisch an einem gemeinsamen Use-Case ausrichten, welcher im Folgenden vorgestellt wird.

2. USE CASE UND BESTANDTEILE

Die zu entwickelnde SOA-Architektur soll die Vision des vernetzten Autos[2] prototypisch unterstützen. Im konkreten Fall sollen aktuell vom Fahrzeug gelieferte Telemetriedaten, z.B. Benzinverbrauch, Ölstand und Fehlermeldungen, gesammelt, geprüft und schließlich sowohl den Endnutzern bzw. Fahrern als auch Werkstätten zur Verfügung gestellt werden.
Das dafür nötige Zusammenspiel bzw. die Integration aller Partner – u.a. Fahrzeugsensoren, Diagnose-Datenbank, Fahrzeugdiagnose-App des Endnutzers, Fernanalyse-Anwendung der Vertragswerkstatt – stellt ein typisches Anwendungsfeld für eine SOA dar.

Das Szenario, welches im Rahmen der Übung Schritt für Schritt (und mit verschiedenen Fokusgruppen) mit den „Bordmitteln“ von WSO2 realisiert werden soll, umfasst die folgenden Aspekte:

- Aktuelle Fahrzeug- bzw. Telemetriedaten werden kontinuierlich gesammelt.
- Die Daten werden geprüft, konsolidiert und in einer Diagnose-DB gespeichert.
- Über verschiedene Clients, z.B. eine Smartphone-App des Fahrers oder eine Browser-App der Vertragswerkstatt, kann auf die Daten zugegriffen werden.
- Im Rahmen eines (BPEL-) Geschäftsprozesses werden die Diagnosedaten analysiert und angereichert, um eine umfassende Auswertung bzw. Diagnose zu ermöglichen.
- Der Zugriff auf die Dienste und die Leistungsfähigkeit der SOA wird fortwährend überwacht.

Einblick in ein aktuelles Praxisbeispiel, welches bei der T-Systems MMS entwickelt wurde, gibt es unter http://www.my-car-is-connected.de/

[1] http://wso2.com
[2] http://youtu.be/f9rNJt7cYb8

VORGABEN

Die folgenden „Assets“ werden für die Bearbeitung der Aufgaben zur Verfügung gestellt. Sie können im Rahmen der Übung gern erweitert werden.

- **DiagDB - Car Diagnostics DB**
  - Repräsentiert die gesammelten, konsolidierten Fahrzeugdaten
  - Fahrzeuganalyse-Datenset in Form einer initialen Datenbank
    - Letzter Service (Datum und bei KM-Stand)
    - Durchschn. Geschwindigkeit
    - KM-Stand
    - Ölstand
    - Öldruck
    - GPS-Koordinaten aktuell
    - Temperatur (Öl, Motor, Bremsen, Kühlflüssigkeit)
    - Drehzahl (Fahrzeug, Kühler-Ventilator)
    - Fehlercodes
    - Reifendruck
    - *Kann ggf. mit eigenen Kenngrößen erweitert werden*

ZIELZUSTAND/-ARTEFAKTE

- **DiagService - Car Diagnostics Data Service(s)**
  - Stellt die Daten des DiagDB in Form eines Web Services (SOAP) bereit
- **DiagClient - Car Diagnostics Client**
  - Erlaubt Kunden und/oder Werkstatt (über DiagService) auf Fahrzeug-Analysedaten zuzugreifen, z.B. per Browser oder Mobile App
  - Potentiell mehrere mit versch. Rechten/Sichten
- **DiagEP - Car Diagnostics Enrichment Process**
  - Geschäftsprozess, der (auf Abfrage) die bestehenden Daten des DiagDB anreichert
  - Bindet externe Dienste zur Berechnung neuer bzw. Erweiterung bestehender Werte an

3. WSO2 ALS GRUNDLAGE

Bei WSO2[3] handelt es sich um eine umfassende Open-Source SOA Plattform, die maßgeblich von der gleichnamigen Firma (WSO2 Inc.) entwickelt wird. Die Plattform unterstützt durch verschiedene Produkte[4] alle wesentlichen Verantwortlichkeiten bzw. Schichten einer SOA-Middleware, z.B. die Dienst-Bereitstellung, die Verknüpfung von Diensten durch Geschäftsprozesse, Routing und Mediation von Dienstaufrufen, Identitätsmanagement, die Überwachung von Dienstaufrufe, u.v.m.
Die Mehrzahl der Produkte basiert auf Apache Projekten. Nicht zuletzt deshalb ist die gesamte WSO2-Produktpalette quelloffen und steht unter der Apache 2.0 Lizenz frei zur Verfügung. Alle Produkte setzen zudem auf dem gemeinsamen Basis-Framework Carbon auf, welches diverse „Features" in Form von OSGi-Bundles bereitstellt. „Features" aus Carbon können somit sehr einfach zu „Produkten" kombiniert werden, und deren gute Interoperabilität ist sichergestellt.
Abbildung 1 zeigt die gesamte Plattform bzw. deren Produkte im Überblick. Die orange hinterlegten Produkte sollen bei dieser Übung im Mittelpunkt stehen:

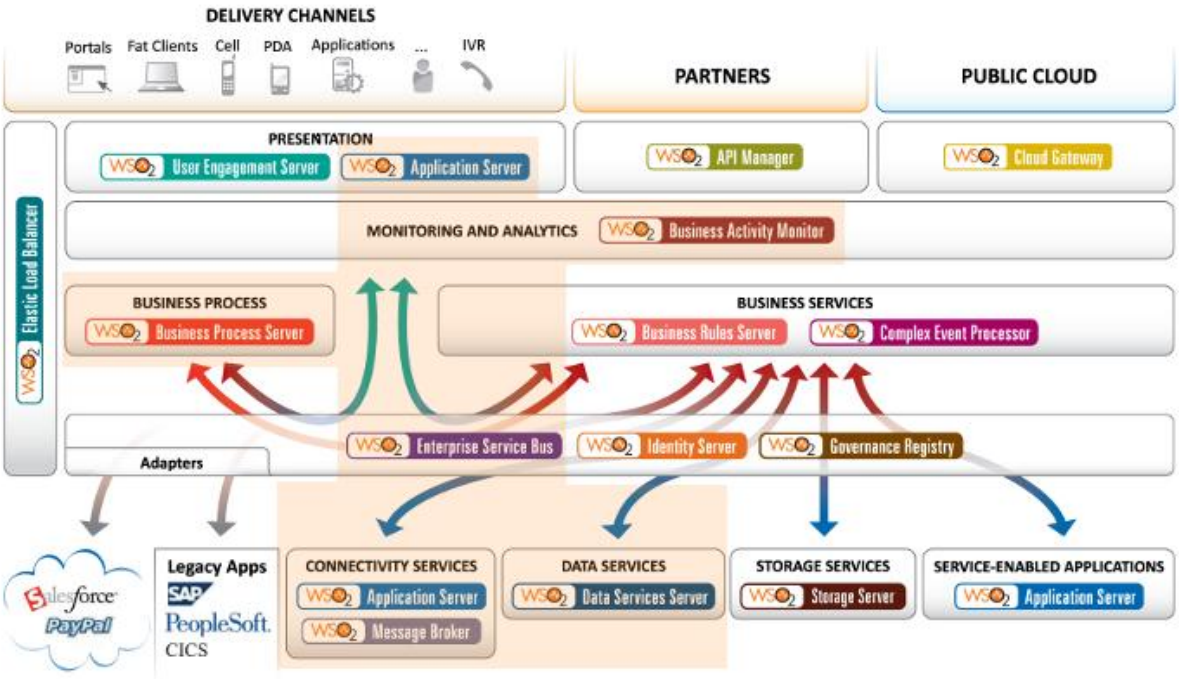


Abbildung 1. WSO2 Plattform Überblick

WSO2 Data Services Server (DSS) dient der Kapselung bzw. einfachen Bereitstellung von Datenquellen (DB, Excel, CVS, ...) in Form von Diensten.

[3] http://wso2.com
[4] http://wso2.com/products/

WSO2 Enterprise Service Bus (ESB) dient als zentrales „Verbindungsstück“ der SOA-Architektur. Für den gegenseitigen Zugriff bzw. die Kommunikation der SOA-Bestandteile stellt der ESB u.a. Mittel zur Mediation und Routing bereit.
WSO2 Application Server (AS) erlaubt das Hosting sowohl von Services als auch Service-Clients bzw. (ggf. interaktiven) dienstbasierten Anwendungen.
WSO2 Business Process Server (BPS) realisiert die Orchestrierung von Diensten, d.h. den Ablauf und die Steuerung von Geschäftsprozessen.
WSO2 Business Activity Monitor (BAM) dient der Überwachung der SOA bzw. der verschiedenen Produkte. So können Key Performance Indicators (KPI), wie z.B. Durchsatz, Last, Fehlerraten, etc. kontinuierlich überprüft werden.
WSO2 Message Broker (MB) dient der Sicherung von Nachrichten innerhalb der SOA. Eingehende Nachrichten werden in sog. Queues bereitgestellt, um dann asynchron von interessierten Clients verarbeitet werden zu können.

Einen guten Überblick zur gesamten Plattform bietet u.a. die Präsentation zur diesjährigen WSO2Con, http://wso2.com/library/wso2con2013/introducing-the-wso2-enterprise-integration-platform/. In Abschnitt 5.3 finden sich zudem eine Reihe weiterer Links zu Artikeln, Foliensätzen, Webinars, Dokumentationen u.v.m.

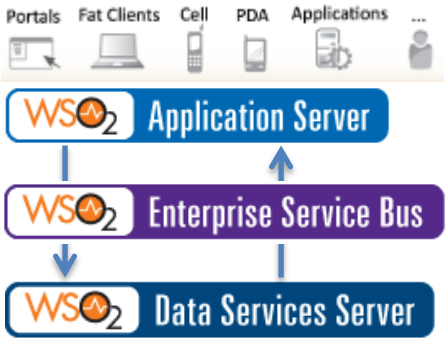
4. AUFGABEN

Im Rahmen der Übung sind zunächst drei allgemeine Aufgaben (Abschnitt 4.1) zu lösen, die die Grundlage für die spätere Vertiefung bilden. Sie sind für alle Vertiefungsgruppen gleich und dienen dazu, die SOA-Konzepte (Dienstnutzer, Dienst-Broker, Dienst-Anbieter) und die WSO2-Plattform kennenzulernen. Aufbauend auf den Ergebnissen erfolgt die Spezialisierung vier verschiedenen Gruppen:

(1) Mediation/Routing, Abschnitt 4.2
(2) Orchestration , Abschnitt 4.3

4.1 Grundlagen (Alle)

4.1.1 Erstellung eines SOAP Datendienstes



DIAGSERVICE

- Bereitstellung der vorgegebenen Daten in Form eines SOAP-Dienstes
- Unterstützung vorgegebener fachlicher Operationen
  - Diagnosedatensatz lesen
  - Diagnosedatensatz schreiben
- Erstellung einer WSDL mit den fachlichen Operationen

Technische Grundlagen

- DiagDB
- WSO2 Data Services Server

4.1.2 Erstellung eines Service-Clients

DIAGCLIENT

- Abruf der Daten vom DiagService (s.o.) über Client, entweder mobile App (Android) oder Web
- In Java, JSP, JSF, Jaggery.js, JavaScript, Portlet, Gadget, ..., whatever makes you happy
- Eingabeparameter: Auswahl eines Zeitraumes
- Ausgabe: Diagnosedaten, grafisch oder tabellarisch
- Verwendung der DiagService-WSDL

Technische Grundlagen

- WSO2 Application Server
- DiagService

4.1.3 Konfiguration des ESB für den Dienstzugriff

ESB

- Entkopplung von DiagClient und DiagServer durch WSO2 Enterprise Service Bus
- Realisierung des Routings über den ESB
  - d.h. DiagClient → ESB → DiagService → ESB → DiagClient
  - „Pass-Through-Proxy"

Technische Grundlagen

- WSO2 Enterprise Service Bus

4.1.4 Zielzustand/ -artefakte

- DiagClient (als mobile App oder WebApp im Application Server)
- Carbon Application aRchive (CAR)
    - Application Server Konfiguration
    - ESB-Konfiguration
    - DiagService (+ WSDL) = ESB Proxy
- DiagDB
- Dokumentation
- SOAPUI-Testfälle

4.2 Routing und Mediation beim Dienstzugriff (Gruppe A)

FOKUS: Enterprise Service Bus

INPUT

- DiagDB
- DiagService (aus 4.1)
- DiagClient (aus 4.1)

AUFGABEN

1. **Message-Routing und -Mediation**
   Alternative Anfragen des Werkstatt-Clients (SOAP) sollen unterstützt werden
   - Erstellung der WSDL (für ESB-Proxy) für Anfragen vom Werkstatt-Client
   - Transformation der ein-/ausgehenden Nachrichten, z.B. mittels XSLT und Filter Mediator
   - Content-based Routing: Je nach Anfrage Weiterleitung zu verschiedenen Datenquellen, z.B. DB und Excel Sheet
2. **Service-Mediation**
   - Unterstützung von Clientanfragen über REST/JSON
   - Umwandlung in SOAP/XML und zurück soll im ESB realisiert werden
   - optional: weitere Protokoll-Mediation
3. **Exception Handling**
   - Behandlung von Message/Mediation-Fehler
     - (Syntaktische) Schema-Validierung eingehender Nachrichten (z.B. Validate Mediator)
     - (Semantischer) Test auf invalide Parameter (z.B. Switch Mediator)
   - Behandlung von Endpoint-Fehlern
     - Reaktion auf DiagService Fehlercodes
     - Reaktion bei Nicht-Verfügbarkeit des DiagService, z.B. Alarmierung bzw. Notifikation einer dritten Instanz im „Fehlerkanal“

TECHNISCHE GRUNDLAGEN

- WSO2 Enterprise Service Bus

ZIELZUSTAND/ -ARTEFAKTE

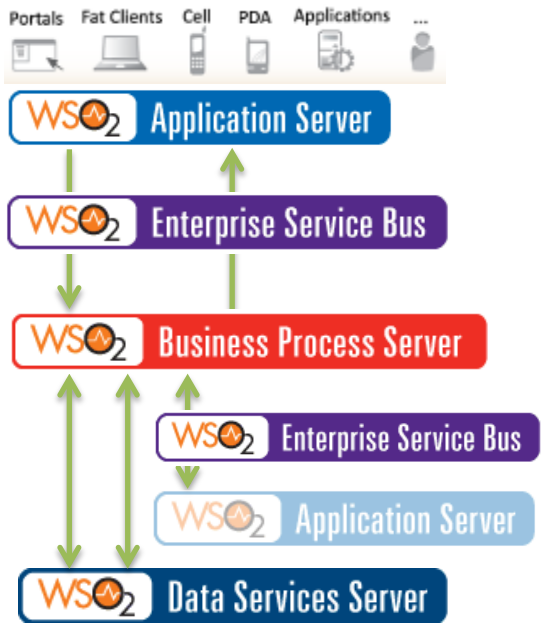
- Werkstatt-Client mit alternativer Anfrage (REST/JSON)
- WSDL (ESB-Proxy) für Anfragen vom Werkstatt-Client
- ESB-Konfiguration, d.h. Routen, (ggf. eigene) Mediatoren, Fehlerbehandlung
- Ausführbare Testfälle bzw. Beispielanfragen (z.B. per SOAPUI TestSuites)
- Dokumentation

4.3 Orchestrierung: Verarbeitung von Geschäftsprozessen (Gruppe B)

FOKUS: Business Process Server

INPUT

- DiagDB
- DiagService (aus 4.1)
- DiagClient (aus 4.1)
- Anforderungen bzgl. Analyse-Anfrage vom Werkstatt-Client
- Anforderungen bzgl. BPEL-Prozess



AUFGABEN

Realisierung der Abfrage komplexerer bzw. abgeleiteter Telemetriedaten, d.h.

1. DiagClient: Erweiterung um entsprechende Analyse-Anfrage und UI (grafisch/tabellarisch)
2. ESB: Routing zum Business Process Server
    a. Erweiterung des Proxys um entsprechende Operationen
    b. Erzeugung der Proxy WSDL entsprechend zusätzlichen fachlichen Operationen
3. Modellierung eines BPEL-Prozesses (DiagEP)
    a. Laden der benötigten Daten über DiagService
    b. Laden weiterer Daten aus Dummy- oder öffentlichen Diensten
    c. Eigener Dienst zur Be- und Verrechnung
    - z.B. Abgleich benötigte Reparaturzeit mit Werkstatt-Kalender
    d. Rückgabe an den DiagClient

TECHNISCHE GRUNDLAGEN

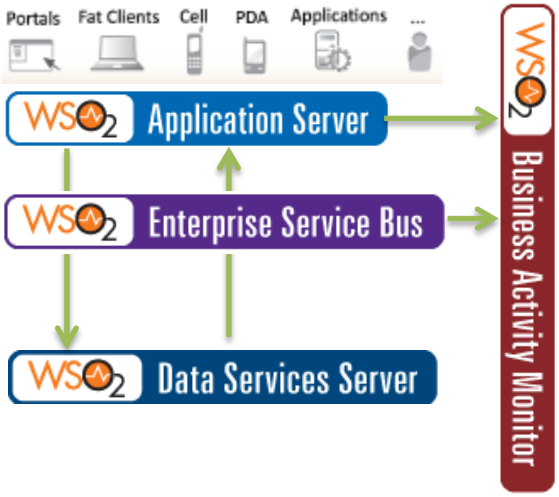
- WSO2 Business Process Server

ZIELZUSTAND/-ARTEFAKTE

- Erweiterter DiagClient mit Analyse-Anfrage
- WSDL (EB Proxy) für Analyse-Anfragen vom Client
- ESB-Konfiguration – Weiterleitung zu BPS
- BPEL-Prozess zur Verarbeitung und Anreicherung der DiagDB-Daten
- BPS-Konfiguration
- Dokumentation
- Ausführbare Testfälle bzw. Beispielanfragen (z.B. per SOAP/LoadUI TestSuites)

4.4 Monitoring: Auswertung von KPIs (Gruppe C)

FOKUS: Business Activity Monitor



INPUT

- DiagDB
- DiagService (aus 4.1)
- DiagClient (aus 4.1) – webbasiert
- TeleProducer
- Beispiel-KPIs

AUFGABEN

Überwachung und Visualisierung von KPIs bzgl. Application Server (eingehende Requests von TeleProcessor und Clients) sowie ESB (Routing zum DiagService), d.h.

1. Simulation von Anfragen des DiagClient mit unterschiedlicher Client-ID, z.B. mittels Selenium[5]
2. Monitoring der WebApp-Statistiken für DiagClient, z.B. Requests, Fehler, Anwortzeit
3. Monitoring *statistischer* KPIs im ESB für Anfragen des DiagClient und TeleProducer, z.B.
4. Monitoring *fachlicher* KPIs im ESB für Anfragen des TeleProducers, z.B. durchschn.
5. Visualisierung (ausgewählter) KPIs, z.B. durch mitgelieferte Gadgets im BAM Dashboard

TECHNISCHE GRUNDLAGEN

- WSO2 Business Activity Monitor

ZIELZUSTAND/-ARTEFAKTE

- Angepasster DiagClient
- ESB-Konfiguration, d.h. Kopplung mit BAM mittels BAM Mediator
- AS-Konfiguration, d.h. Kopplung mit BAM
- BAM-Konfiguration inkl. HiveQL-Queries
- Visualisierung ausgewählter KPIs
- Ausführbare Testfälle unter Nutzung von TeleProducer und Selenium

[5] http://docs.seleniumhq.org/

4.5 Überblick

Abbildung 2. Aufgaben im Überblick zeigt die Aufgaben im Überblick und in Relation zur WSO2-Plattform.

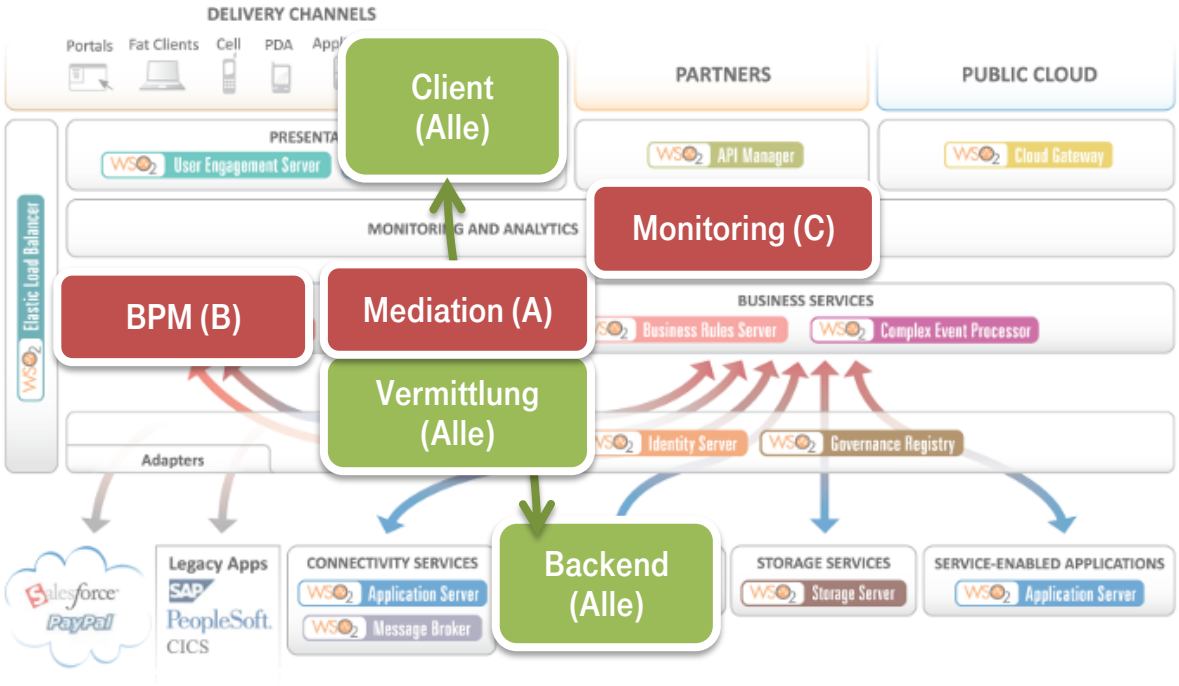


Abbildung 2. Aufgaben im Überblick

4.6 Abgrenzungskriterien

Bei der Umsetzung zu vernachlässigen sind zunächst:

- Identity Management bzw. Single-Sign-On
- Security-Aspekte
- Mehrbenutzerunterstützung bzw. Concurrency
- Performance

5. ORGANISATORISCHES

5.1 Ablauf

Für die Übung gelten die folgenden Rahmendaten:

- 22.10.2013: Vorstellung der Aufgaben
- 04.11.2013: Einführung / WSO2-Vorstellung
- 19.11.2013: Kurzvorstellung der Grundidee (5 min)
- 17.12.2013: Präsentation zu Grundlagen-Aufgaben (4.1)
- 28.01.2014: Abschlusspräsentation

5.2 Technologie und Werkzeuge

5.2.1 Allgemeines

Alle Dienste bzw. Client sind in OpenSource Java umzusetzen. Im Falle mobiler Clients sollte für Android (API Level 16, v 4.1.1) entwickelt werden. Zur Codeverwaltung kann Assembla oder Git verwendet werden. Es wird pro Gruppe eine virtuelle Maschine zur Verfügung gestellt.

5.2.2 Produkte

Zur Umsetzung sollen die WSO2-Produkte zum Einsatz kommen, d.h. insbesondere

- Für Services: **WSO2 Data Services Serv**er oder **Application Server**
- Für Clients: **WSO2 Application Server**, (User Engagement Server auf Anfrage)
- Zum Monitoring: **WSO2 Business Activity Monitor**

5.2.3 Entwicklungsumgebung und -werkzeuge

Bei Bedarf kann auf das **WSO2 Developer Studio** zurückgegriffen werden. Diese Eclipse-basierte Entwicklungsumgebung erlaubt u.a. die Modellierung von ESB-Routen und Geschäftsprozessen sowie die Entwicklung von Datendiensten und Webanwendungen mit Hilfe teils grafischer Werkzeuge. Die Ergebnisse können als Carbon Application aRchives (CAR) exportiert und in die entsprechenden Produkte importiert werden.

Für die Erstellung von Testfällen wird die Nutzung von SOAPUI [6]und LoadUI[7] empfohlen. Um Anfragen auf webbasierten Anwendungen, wie dem DiagClient, zu simulieren, kann Selenium[8] genutzt werden.

Die Entwicklung erfolgt voraussichtlich innerhalb dedizierter virtueller Maschinen.

[6] http://www.soapui.org/
[7] http://www.loadui.org/
[8] http://docs.seleniumhq.org/

5.3 Links

5.3.1 Allgemein

- WSO2 Middleware Platform
  http://wso2.com/
- Introducing the WSO2 Platform (Folien)
  http://de.slideshare.net/wso2.org/introducing-the-wso2-platform
- WSO2 Library (Artikel, Präsentationen, Whitepaper, Case Studies)
  http://wso2.com/library/
- WSO2 Architecture Blog
  http://wso2.com/blogs/architecture/
- WSO2 Product Documentation
  http://docs.wso2.org/dashboard.action
- WSO2@Slideshare (Foliensätze)
  http://de.slideshare.net/wso2.org
- WSO2 Tech Flicks @ YouTube
  http://www.youtube.com/user/WSO2TechFlicks
- WSO2 Q&A @ StackOverflow
  http://stackoverflow.com/questions/tagged/wso2

5.3.2 Grundlagen

- Data Services Definition (DSSL)
  http://docs.wso2.org/display/DSS310/Configuring+Data+Services+and+Resources
- Pass-Through-Proxies
  http://docs.wso2.org/display/ESB470/Pass+Through+Proxy+Template
- Create First Proxy service from WSO2 ESB
  http://madhukaudantha.blogspot.de/2012/05/create-first-proxy-service-from-wso2.html

5.3.3 Gruppe A - ESB

- Using REST with a Proxy Service
  http://docs.wso2.org/display/ESB470/Using+REST+with+a+Proxy+Service

5.3.4 Gruppe B – BPS

- Create First Proxy service from WSO2 ESB
  http://madhukaudantha.blogspot.de/2012/05/create-first-proxy-service-from-wso2.html

5.3.5 Gruppe C - BAM

- WSO2 ESB Data Load Monitoring with Business Activity Monitor
  http://wso2.com/library/articles/2013/07/esb-data-load-monitoring-with-bam/
- WSO2 ESB mediator to publish data to WSO2 BAM
  http://subashsdm.blogspot.de/2013/01/esb-mediator-to-publish-data-to-bam.html
- Setting up BAM Mediator
  http://docs.wso2.org/display/BAM230/Setting+up+BAM+Mediator

- Setting up Mediation Data Agent
  http://docs.wso2.org/display/BAM230/Setting+up+Mediation+Data+Agent
- Setting up Service Statistics Data Agent
  http://docs.wso2.org/display/BAM230/Setting+up+Service+Statistics+Data+Agent
- DBeaver - Cassandra GUI
  http://dbeaver.jkiss.org/