FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Enterprise Mashups

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Mag.rer.soc.oec.

im Rahmen des Studiums

Informatikmanagement

eingereicht von

Dipl.-Ing. Georg Ungerböck
Matrikelnummer 0728449

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer/in: Ao.Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Christian Huemer
Mitwirkung: Projektass. Dipl.-Ing. Mag. Dieter Mayrhofer
Mitwirkung: Univ.Ass. Mag.rer.soc.oec. Petra Brosch

Wien, 16.03.2012

(Unterschrift Verfasser) (Unterschrift Betreuer)

Erklärung zur Verfassung der Arbeit

Georg Ungerböck, Sporngasse 11, 3380 Pöchlarn

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 16.3.2012

_____

Vorname Nachname

Kurzfassung

Der Einsatz von Enterprise Mashups erlaubt Anwendern ohne technische Kenntnisse aus vorhandenen Daten in unterschiedlichen Systemen ad hoc eine neue Anwendung zu erstellen. Die Integration heterogener Daten in eine neue Anwendung durch den Endanwender stellt gerade im Einsatz von Unternehmen ein großes noch ungenutztes Potential dar.

So arbeiten viele Unternehmen mit hoch integrierten ERP Systemen, die einen Großteil der Daten zur Steuerung der Prozesse enthalten. Allerdings sind nicht alle Informationen vollständig im ERP System enthalten bzw. können nicht alle Operationen über das ERP System ausgeführt werden. Es entsteht die Situation bei der Daten aus unterschiedlichen Systemen zusammengeführt und daraus Entscheidungen getroffen werden müssen. Der Einsatz einer Mashup Applikation, welche bei Bedarf entsprechend der Anforderung erstellt wird, bringt einen entscheidenden Mehrwert für die Fachabteilungen.

Im Rahmen der vorliegenden Arbeit werden die speziellen Anforderungen wie Datenqualität, die erhöhte Sicherheit bei unternehmerisch sensiblen Daten und die Einbindung von integrierten ERP Systemen als auch von Legacy Applikationen in Enterprise Mashups untersucht. Unter Legacy Applikationen werden alle Arten von unternehmerischen Anwendungen verstanden, die unter Umständen technisch nicht mehr auf dem neuesten Stand sind, allerdings aus verschiedenen Gründen weiterhin im Unternehmen eingesetzt werden. Des Weiteren wird ein Prototyp für ein Enterprise Mashup entwickelt, welcher die Verwendung von Informationen aus dem ERP System Microsoft Dynamics AX in Verbindung mit anderen Mashup Daten erlaubt. Für die Realisierung des Prototyps wird innerhalb Dynamics AX eine Möglichkeit zur Anbindung von Mashups entwickelt. Für das Mashup selbst werden vorhandene Frameworks evaluiert und ein Framework für die Umsetzung des Prototyps gewählt, welches die entsprechenden Dienste zum Erstellen des Mashups und Bereitstellen der Daten für die Clients bietet.

Mit dem exemplarisch umgesetzten Mashup wird die schnelle und unkomplizierte Kombination der Daten unterschiedlicher Anwendungen in den Vordergrund gestellt. Der Prototyp stellt einen Leitfaden für Anwender und Unternehmen dar, um die oft komplizierte technische und organisatorische Umsetzung neuer Applikationen zu vereinfachen. Mit Mashups wird Mitarbeiten ein Werkzeug bereit gestellt, um bei Bedarf selbst ohne Programmierkenntnisse und ohne die Unterstützung der IT Abteilung die notwendige Anwendung oder Auswertung erstellen zu können.

Abstract

The usage of Enterprise Mashups enables users to create ad hoc applications without the need of technical knowledge. These applications can use and combine data sources of different systems. The integration of heterogeneous data in a new application is still a rather unused potential within a company.

Many companies work with a highly integrated ERP system, which contains the bigger part of the data to control the business processes. However not all necessary data is covered and accordingly not all processes can be controlled by the ERP system. Therefore the situation arises that information from various systems must be consolidated to make a decision. The usage of an ad hoc Mashup application, which is created according to necessary claims, leads to an important added value for the departments.

In line with this diploma thesis special requirements like data quality, the raised security of corporate sensitive information and the integration of legacy applications into Enterprise Mashups will be proved. Moreover a prototype for an Enterprise Mashup is implemented. The prototype shows the combination of data from the ERP system Microsoft Dynamics AX in conjunction with other data sources. To put it into practice a solution to use the ERP system as a Mashup data source is developed. Available Mashup frameworks have been evaluated and the one with the best correlation according to the requirements has been chosen for the prototype.

The implemented Mashup shows the fast and straightforward combination of data from various applications. The prototype illustrates a teaching case for companies which want to supply their employees with the possibility to use existing data sources in the context of an Enterprise Mashup. User obtain the ability to create themselves instruments on demand for the daily business without programming skills.

Inhaltsverzeichnis

Teil I.

Einführung

1. Einleitung

Der Einsatz des World Wide Web als Medium für das reine Publizieren von Informationen hat sich seit den ersten Web Anwendungen nachhaltig verändert. Waren die ersten textbasierenden Systeme auf das zur Verfügung stellen von Daten ausgelegt, sind aktuell multimediale Web Applikationen in unterschiedlichen Anwendungsgebieten in Betrieb. In Zeiten des Web 2.0 wo Benutzer den Content zu großen Teilen selbst erstellen ist der nächste evolutionäre Schritt auch die Anwendungen und Auswertungen durch den Endanwender selbst erzeugen zu lassen. Hierbei sind keine Applikationen im herkömmlichen Sinn gemeint, sondern Mashups d.h. die Kombination von verschiedenen Datenquellen in einer ad hoc Manier zu eine neuen Aussage. Neben der klassischen Verwendung als Informationsbereitstellung für Anwender gibt es auch Techniken um die bereitgestellten Informationen für die automatische Verarbeitung einsetzen zu können. Die im Web publizierten Daten sollen von Maschinen interpretiert und automatisch weiterverarbeitet werden können. Informationen über Unternehmen, Personen, Orte, usw. sollen von maschinellen Systemen in Verbindung gebracht werden können. Die angesprochenen Möglichkeiten können rein durch die Funktionen des WWW nicht ausreichend abgedeckt werden. Die Erweiterungen zur Erfüllung dieser Anforderungen werden in einem Konzept des WWW Begründers Tim Berners mit dem Begriff *Semantisches Web* beschrieben. Neben der 'Verarbeitung' der Daten durch Menschen soll es möglich gemacht werden, die Daten automatisiert mit Rechnersystemen verarbeiten zu können [60]

Auf diesen Grundüberlegungen und Voraussetzungen bauen Mashups auf - Daten können miteinander in Beziehung gestellt werden. Durch die Kombination von bisher alleinstehenden Daten generieren Mashups aus Informationen von zwei oder mehreren Quellen Inhalte in einem neuen Zusammenhang. Der generierte Mehrwert stellt in Form von Mashups eine neue interessante Art und Weise dar, um Daten aus dem Web zu konsumieren. Es ist allerdings nicht nur das Konsumieren der neu erzeugten Daten relevant. Auch die einfache Erstellung von Mashups durch den Benutzer selbst ist eine Zielsetzung. Das Erzeugen von Mashups wird in Form von Mashup Editoren in Verbindung mit Mashup Frameworks durchgeführt.
Auf diesen Konzepten bauen Enterprise Mashups auf, mit dem Unterschied, dass hier nicht rein Daten aus dem World Wide Web kombiniert werden sollen, sondern unter anderem auch Daten aus den in Unternehmen eingesetzten Applikationen. Die Erzeugung von Mashups soll dabei so weit als möglich dem Benutzer in den Fachabteilung überlassen werden, indem ihm die entsprechenden Werkzeuge und Berechtigungen zur Verfügung gestellt werden. Bei Enterprise Mashups muss den gesteigerten Anforderungen unter anderem in Bezug auf Sicherheit, Sensibilität und Integrität der bereitgestellten Daten Rechnung getragen werden.
Die Datenbereitstellung und Interpretation der Bedeutung der Informationen muss im Rahmen von Enterprise Mashups ebenfalls unter anderen Voraussetzungen erfolgen. Als Datenquelle kann jede beliebige Datenbank oder Applikation dienen, welche in der IT Landschaft eines Unternehmens Einsatz findet. Die gelieferten Daten müssen durch geeignete Maßnahmen korrekt interpretiert und mit Daten aus anderen Unternehmens-

applikationen oder aus dem WWW kombiniert werden.
Ziel dieser Arbeit ist für die ausgeführten Herausforderungen bei der Verwendung von Enterprise Mashups geeignete Wege und Lösungsmöglichkeiten anzubieten. Im Rahmen der Arbeit wird als Referenzlösung ein einsetzbarer Prototyp eines Enterprise Mashups entwickelt und die umgesetzten Konzepte und Ideen dokumentiert, um als Ausgangsbasis für die Implementierung weiterer Enterprise Mashups als Teaching Case verwendet werden zu können.

1.1. Motivation

Hinter dem Konzept von Enterprise Mashups als Weiterführung der Mashup Idee steckt ein enormes Potential an Möglichkeiten und Chancen. Um als Unternehmen gegenüber den Mitbewerbern immer einen Schritt voraus zu sein, müssen die eigenen Prozesse und Abläufe möglichst effizient gestaltet sein.
Angesprochen sind hierbei Änderungen, Ergänzung oder Neuanforderungen einer Fachabteilung an die Funktionen der Informationssysteme im Unternehmen, um benötigte Daten auszulesen, zu exportieren oder zur Entscheidungsfindung analysieren zu können. Waren dazu bisher umfangreiche Ausarbeitungen und Definitionen der geforderten Funktionen als Vorbereitung zur Umsetzung notwendig, stellt die Idee der Enterprise Mashups eine neue und kürzere Verfahrensweise für Implementierungen dieser Art dar.
Nicht mehr die IT-Abteilung (oder ein externer Partner) ist für die Umsetzung verantwortlich, sondern die Fachabteilung selbst. Diese in einem gewissen Bereich autonome Vorgehensweise einer Abteilung bzw. eines Mitarbeiters kann die Auslastung der Unternehmensressourcen deutlich reduzieren und Kommunikationswege abkürzen. Voraussetzung für das Funktionieren ist die Verfügbarkeit eines Frameworks zum Aufbau von Enterprise Mashups (siehe 5.4), ein einfach und intuitiv zu bedienender Mashup Editor (siehe 4.5.2) und gut geschulte Mitarbeiter in den Fachabteilungen, welche über die Begriffe, Mechanismen und Kenntnisse zum Erstellen von Mashups verfügen (siehe 6).

Der Bearbeitung dieses Themas geht eine intensive Auseinandersetzung mit den internen Abläufen einer IT-Abteilung und der unternehmensinternen Zusammenarbeit zwischen Abteilungen voraus. Möglichkeiten zur Vereinfachung von Vorgängen und Verkürzung von Kommunikationsabläufen waren die Motivation zur Wahl des Themas. Die Arbeit soll Unternehmen Wege in Form eines exemplarisch umgesetzten Enterprise Mashups zeigen, die eigenen Strukturen durch Ergänzung mit neuen Technologien ökonomischer und wirkungsvoller gestalten zu können (Leitfaden zur Erstellung eines Enterprise Mashups siehe 7).

1.2. Zielsetzung

Die Zielsetzung der Arbeit kann in zwei Teilbereiche untergliedert werden. Zum einen sollen die Funktion sowie die eingesetzten Technologien von Mashups beschrieben und analysiert werden. Die Tiefe der Analyse und Beschreibung ist ausgelegt, um die Entwicklung eigener Mashups und Frameworks bewerkstelligen zu können. Der analytische Teil beschäftigt sich neben den Technologien mit den speziellen Anforderungen an Enterprise Mashups. Der Einsatz von Mashups in Unternehmen und Unternehmenslösungen stellt in verschiedenen Punkten anders gelagerte Voraussetzungen dar (im Vergleich

zu 'normalen' Mashups). Auf Punkte wie Datenqualität, die erhöhte Sicherheit bei unternehmerisch sensiblen Daten und ganz speziell auf die Einbindung von integrierten ERP Systemen müssen unter einem besonderen Augenmerk betrachtet werden. Einen hohen Stellenwert im Rahmen der Arbeit hat deshalb der Punkt Sicherheit bei Enterprise Mashups. Um Mashups in die IT-Landschaft eines Unternehmens integrieren zu können, muss sichergestellt werden, dass neben modernen ERP Systemen auch Daten von Legacy Applikationen eingebunden und verwendet werden können. Erst dadurch kann der Mehrwert von Enterprise Mashups in einem Unternehmen vollständig genutzt werden.

Zum anderen wird als Weiterführung des ersten Teils, im zweiten Teil der Arbeit ein Prototyp für ein Enterprise Mashup entwickelt, das produktiv in Unternehmen eingesetzt werden kann. Der Prototyp baut auf den aus der Analyse gewonnen Erkenntnissen und beschriebenen Technologien auf, und kombiniert Daten aus dem ERP System (Microsoft Dynamics AX) mit einem internetbasierenden Datendienst.
Der Prototyp setzt ein Mashup Framework ein, das Daten aus der ERP Applikation mit einem im Internet verfügbaren Web Service kombiniert. Das Hauptziel des Prototyps ist eine Referenzimplementierung bereitzustellen, um Unternehmen einen Ansatzpunkt für den erstmaligen Einsatz von Enterprise Mashups zu geben. Dabei wird besonders auf folgende Punkte eingegangen:

- Implementierung eines Mashups: Es wird gezeigt wie ein Mashup anhand eines Vorgehensmodell entwickelt werden kann. Dabei wird auf vorbereitende Schritte, die durch die IT-Abteilung eines Unternehmens durchgeführt werden müssen, eingegangen. Sind diese Schritte erfolgreich abgeschlossen, kann die Erstellung von Mashups durch die Endanwender beginnen. Dies wird im entwickelten Leitfaden dargestellt. Darin wird gezeigt, welche Schritte von Benutzern durchgeführt werden müssen, um ad hoc ein Mashup zu entwickeln.

- Sicherheitsaspekte: Werden Enterprise Mashups in einem Unternehmen eingesetzt, muss der Sicherheit die entsprechende Aufmerksamkeit gewidmet werden. Die im Mashup verwendeten Daten kommen aus Anwendungen mit unternehmerisch sensiblen Daten, z.B. aus einem ERP System. Diese Daten dürfen das Unternehmen im Regelfall nicht verlassen. Damit ist eine Anforderung an das Mashup, dass die Ausführung auf unternehmensinternen Rechnern erfolgt. Des weiteren dürfen nicht alle Mitarbeiter auf alle Daten aus den unterschiedlichen Quellsystemen zugreifen. Z.B. gilt das unter anderem für Daten aus der Lohnbuchhaltung. Es muss eine Möglichkeit geschaffen werden, um den Zugriff auf die Daten als auch auf die erstellten Mashups für Benutzer bzw. Benutzergruppen individuell verwalten zu können.

- Einbindung von Unternehmensanwendungen (Legacy Applications): Unternehmen verwenden üblicherweise mehrere unterschiedliche Systeme für die Erfüllung diverser Aufgaben, z.B. ein ERP System, eine Buchhaltungssoftware, diverse Produkte für die Produktionssteuerung. Diese heterogene Umgebung muss in ein Mashup integriert werden können. Dazu ist es notwendig, dass das eingesetzte Framework unterschiedliche Einbindungsmöglichkeiten für Datenquellen bereitstellt.

Die Erreichung der beschriebenen Zielsetzungen werden unter 7.5 evaluiert.

2. Mashups

Im folgenden Kapitel werden die Grundbegriffe des Mashups selbst sowie Begriffe rund um das Thema beschrieben. Außerdem wird auf die Einsatzgebiete und der Nutzen von Mashups eingegangen.

Mashup: Der Begriff *Mashup* kommt ursprünglich aus der Musik und bezeichnet einen Remix, der aus zwei oder mehr Titeln zusammengemischt wurden. Im Web 2.0 wurde der Begriff sinngemäß übernommen und bezeichnet ebenfalls die Vermischung von mehreren Inhalten [3], [71]. Für den Begriff im Zusammenhang mit Internet ist noch keine akzeptierte Definition in der Literatur zu finden. An dieser Stelle wird zur Klärung des Begriffes deshalb auf die Definition in [72] verwiesen: "Mashup (von engl.: 'to mash' für vermischen) bezeichnet die Erstellung neuer Medieninhalte durch die nahtlose (Re-)Kombination bereits bestehender Inhalte." In den meisten Fällen sind dabei (Web-) Anwendungen gemeint, die Daten von bestehenden Anwendungen kombinieren und als Ergebnis eine neue Anwendung darstellen. Die Ausgangsdaten werden über Schnittstellen der Quellsysteme bezogen, das Mashup selbst wird durch den Einsatz von entsprechenden Frameworks und Editoren erstellt und ausgeführt.

Ziel eines Mashups ist die Kombination vorhandener Einzeldaten unter einem neuen Gesichtspunkt zur Schaffung eines Mehrwertes für den Konsumenten, der ohne Mashup nicht gegeben wäre oder erst unter Zuhilfenahme manueller Werkzeuge erzeugt werden könnte. Ein typisches Exempel für ein Mashup wäre z.B. die Kombination von Google Maps mit darauf eingeblendeten Standorten der Top 10 Kunden eines Unternehmens.

2.1. Einsatz

Mashups werden in der Regel als Web Anwendungen verstanden, die Ausführung und die Vernetzung der Informationen aus den verschiedenen Datenquellen erfolgt serverseitig, die Präsentation des generierten Mashups erfolgt clientseitig in einem Browser. Eine detaillierte Beschreibung der eingesetzten Architekturen folgt im nächsten Abschnitt (siehe 2.2), eine detaillierte Beschreibung der Technologien folgt im nächsten Kapitel (siehe 4). Um einen grobe Vorstellung über die Anzahl der bereits verfügbaren Mashups im Web zu erhalten, wird auf [70] verwiesen. Die Seite enthält eine Liste mit mehr als 4300 Mashups die täglich erweitert wird. Mashups werden verwendet um z.B. die freien Parkplätze eines Flughafens grafisch auf einer Karte darzustellen oder um die besten Golfplätze eines Landes auf einer Landkarte einzublenden. Die Liste könnte unendlich weitergeführt werden. Bei allen Mashups liegt jedoch die gleiche Idee dahinter: Die Rekombination von zwei oder mehr Datenquellen zu einer neuen Anwendung mit dem Nutzen, dem Anwender Informationen in einem neuen Kontext zur Verfügung stellen zu können.

2.2. Architektur

Zur Umsetzung von Mashups gibt es eine Vielzahl von Möglichkeiten. Die beiden wichtigsten Architekturen werden in den folgenden Abschnitten näher beschrieben: *client-side Mashups* und *server-side Mashups*. Neben reinen client-side und server-side Mashups sind auch Mischformen zwischen den beiden Varianten oder Mischformen mit einer der beiden und einer anderen Variante möglich. Die beschriebenen Architekturen beruhen überwiegend auf den Inhalten von [53], [54] und [31].

Die Verbindung unterschiedlicher Datenquellen zu einem Mashup setzt eine Schnittstelle zum Zugriff auf die Datenquelle voraus. Diese Notwendigkeit führt zu einer Reihe an entwickelten APIs für Mashup Datenquellen. Ein ungefährer Überblick über die Anzahl der verwendeten APIs ist unter [70] zu sehen (Die angegeben Daten beziehen sich nur auf jene Informationen, die von Erstellern von Mashups auf freiwilliger Basis hinterlegt werden, Stand 2010). Grundsätzliche Trends können davon abgelesen werden: Die meistgenutzten APIs für Mashups sind Google Maps (50%), Flickr (11%) und Amazon (8%). Die angebotenen APIs erlauben den Zugriff über ein oder mehrere Protokolle, wobei Web Service Protokolle für die Anbindung der APIs an erster Stelle stehen.

Das *Facade Pattern* (siehe Abbildung 2.1) aus dem Bereich der objektorientierten Softwareentwicklung kann als Basisstruktur für Mashups betrachtet werden [25], [55]. Das Facade Pattern stellt eine einfache Schnittstelle zur Verfügung, um eine Menge an Funktionen und Schnittstellen von Subsystemen zu verwenden. Die Funktionen des Subsystems sind meist technisch orientiert und erfordern Informationen über den korrekten Einsatz. Die Aufgabe der Facade ist es den Einsatz des Subsystems für den Aufrufer zu vereinfachen und technische Details zu verdecken. Änderungen an Teilen des Subsystems sind für den Aufrufer nicht transparent indem sie durch eine Adaptierung der Facade abgefangen werden. Das Facade Pattern wird häufig im Bereich Enterprise Computing zur Vereinfachung komplexer Vorgänge eingesetzt. Bei Java Enterprise Umgebungen werden *Session Facades* für die Zusammenfassung von Abläufen in der Business Logic in einer einheitlichen Schnittstelle verwendet. Bei Dynamics AX Systemen stellen Facades die Zugangspunkte für umfangreiche Abläufe wie das Durchführen einer Buchung dar.

Die Struktur des Facade Pattern ermöglicht eine lose Kopplung [59] zwischen dem Subsystem und dem Anwendungssystem, da es die im Subsystem verfügbaren Funkionen verdeckt. Änderungen an Teilen des Subsystems haben eine geringe bis keine Auswirkung auf den Aufrufer. Das Prinzip der losen Kopplung des Facade Patterns wird implizit bei Mashups angewendet. Mashups stellen eine Verknüpfung von einzelnen Services mit Hilfe eines Mashup Frameworks dar. Für den Anwender werden bei der Ausführung des Mashups die darunterliegenden Strukturen der Datenquellen verborgen. Das Mashup Framework entspricht der Facade. Änderungen an den Datenquellen (durch den Anbieter) müssen bei Mashups je nach Art der Änderung zwar nachgezogen werden, trotzdem bleibt die grundsätzliche Struktur der losen Kopplung aufrecht.

Mashups werden im Allgemeinen aus Teilen, Daten und Diensten von bereits bestehenden Artefakten aus dem Web oder dem Intranet von Unternehmen gebildet. Zusätzlicher Sourcecode ist notwendig, um aus den Einzelteilen das Mashup zu bilden. Dieser Sourcecode kann entweder in Form eigens dafür entwickelter Software oder dem Einsatz eines Frameworks bestehen.

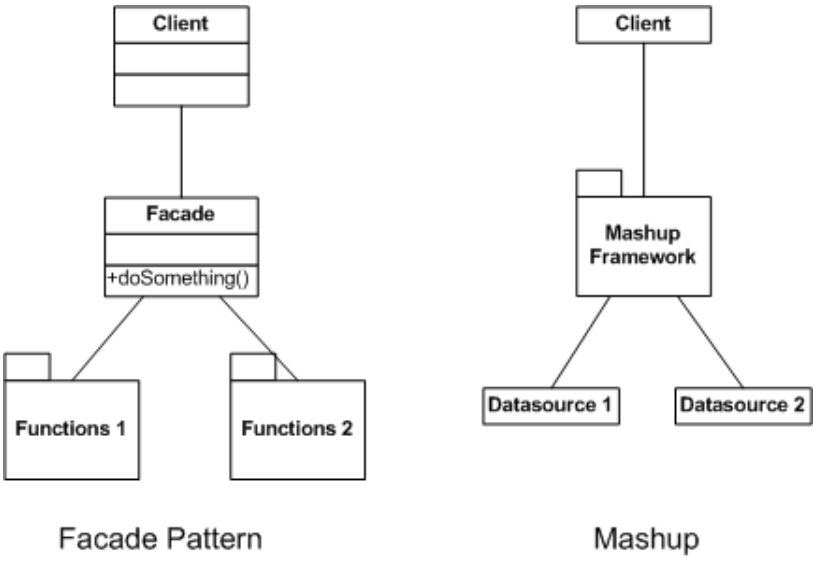

Abbildung 2.1.: Vergleich Facade - Mashup

2.2.1. Client-Side Mashups

Bei client-side Mashups findet die Verarbeitung aller relevanten Daten zum Mashup vollständig am Client, typischerweise einem Browser, statt. Die Logik zum Generieren des Mashups ist meist in Form von Java Script Funktionen innerhalb einer Webseite umgesetzt. Problematisch bei dieser Architektur kann die Tatsache sein, dass die Interpretation von Java Script in manchen Details nicht bei allen Browsern gleichermaßen umgesetzt ist. Auf Browserspezifika muss bei der Planung des Mashups Rücksicht genommen oder Bibliotheken eingesetzt werden, die diesen Gegebenheiten Rechnung tragen. Die Einbindung von aufgerufenen Services ist aufgrund der Browser Security Sandbox nicht ohne Einschränkungen möglich. Die Sandbox des Browsers verhindert, dass Verbindungen zu anderen Seiten als der ursprünglichen Ausgangsseite aufgebaut werden.

Trotzdem sprechen einige Gründe für den Einsatz eines client-side Mashups: Mashups dieser Art können aus **wenigen Komponenten** in Verbindung mit Java Sript erstellt werden. Für die Ausführung ist lediglich die Erstellung eines HTML Dokuments mit den Mashup Funktionen bzw. die Einbindung und Bereitstellung einer Library am Server notwendig.
Es braucht **keine eigene Web Applikation** entwickelt und bereit gestellt werden. Die Web Applikation benötigt neben dem Web Server zusätzlich eine Umgebung zur Ausführung, die ebenfalls entfällt.
Die **Last am Server** wird reduziert, die Ausführung der Mashup Funktionen erfolgt rein am Client.

Ablauf (zu Abbildung 2.2)

1. Der Browser fordert über einen http Request die Startseite des Mashups an. Über diese wird die weitere Logik des Mashups gestartet. Die Seite enthält üblicherweise Java Script Funktionen oder referenziert auf eine Java Script Bibliothek am Server.

2. Der Server transferiert die Mashup Seite zum Client.

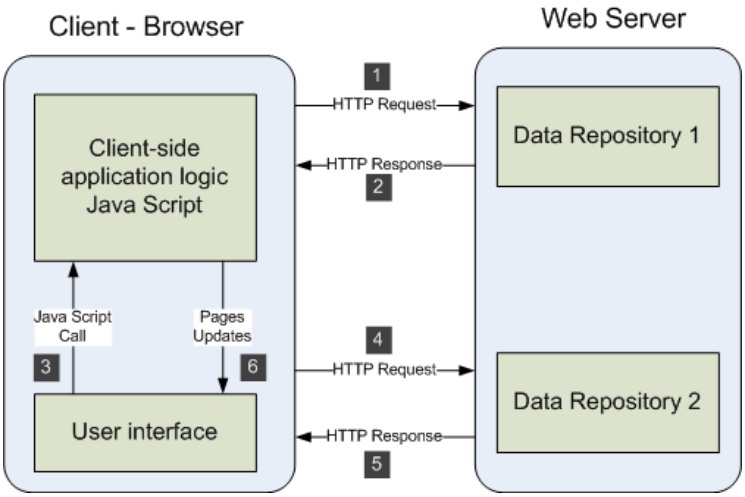

Abbildung 2.2.: Client-Side Mashup, basierend auf [53]

3. Ein vom Benutzer oder automatisch beim Laden der Seite ausgelöstes Ereignis triggert den Aufruf der spezifischen Funktionen zum Aufruf des Mashups.

4. Die Mashup Funktionen rufen alle notwendigen Services zur Erstellung des Mashups auf. Je nach Umfang der Funktionen können aufgerufene Seiten mit unterschiedlichen Formaten unterstützt werden. Zumindest das für Java Script übliche Datenübertragungsformat JSON (siehe 2.4.2) sollte einsetzbar sein.

5. Das Service stellt die angeforderten Daten im entsprechenden Format für den Client bereit.

6. Die Mashup Funktionen und Bibliotheken erstellen aus den erhaltenen Informationen der Datenquellen das Ergebnis, und stellen es in der HTML Seite dar.

2.2.2. Server-Side Mashups

Bei diesem Architekturmodell werden die Services, welche die Eingangsdaten für das Mashup liefern, am Server aufgerufen und zum Gesamtergebnis integriert. Der Server fungiert als Proxy für den Client, der die notwendigen Aufrufe zur Erstellung des Mashups ausführt. An den Client werden nur die generierten Resultate zurück geschickt. Die Eingangsdaten für die serverseitige Zusammenstellung des Mashups können von der Server Applikation per API Calls von mehreren entfernten Diensten abgeholt werden oder auch mit lokalen Daten kombiniert werden. Der Client kommuniziert nur mit einem Server.
Das erzeugte Ergebnis kann bereits am Server in das geforderte Ausgabeformat (siehe 2.4) des Clients konvertiert werden. Daten können beim Zusammenstellen des Ergebnisses am Server nach hinterlegten Kriterien des Anwenders modifiziert werden, indem die Informationen gefiltert werden, und nur relevante Daten an den Client weitergegeben werden.

Ablauf (zu Abbildung 2.3)

1. Der Benutzer ruft mit einem HTTP Request eine Webseite auf. Der gelieferte Response enthält neben HTML Code auch Java Script Funktionen. Durch ein vom Benutzer ausgelöstes Ereignis (z.B. Klick auf einen Button) wird ein Ereignis ausgelöst, das einen Aufruf einer Java Script Funktion zur Folge hat.

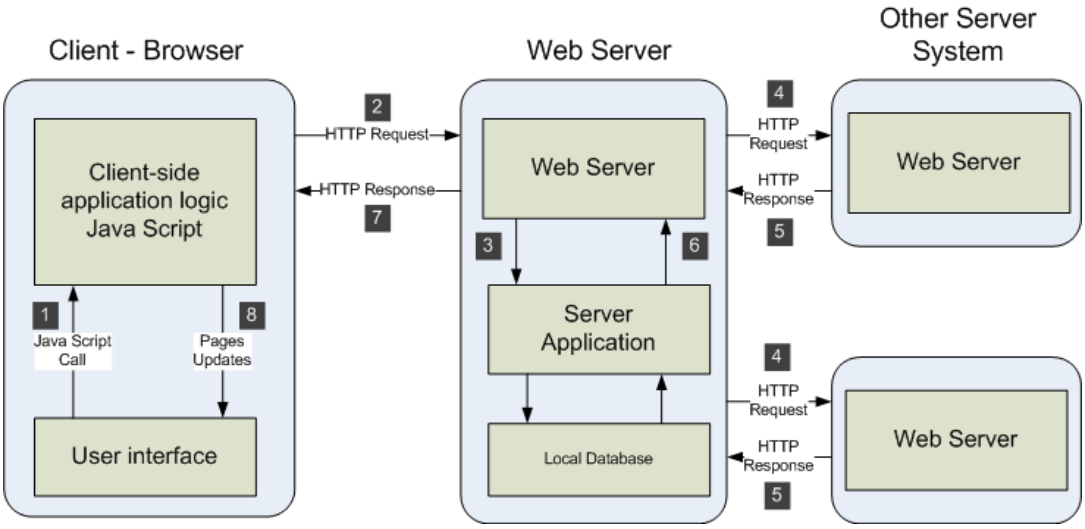

Abbildung 2.3.: Server-Side Mashup, basierenda auf [53]

2. Die angestoßene Java Script Funktion schickt einen Request, typischerweise mit Hilfe eines Aufrufes von `XmlHttpRequest`, an eine Web Komponente am Server.

3. Die aufgerufene Komponente (z.B. ein Servlet) agiert als Proxy für den Client und bereitet die Zusammenstellung des Mashups vor.

4. Die Web Applikation bereitet alle notwendigen Services auf, die für die Generierung des Mashups notwendig sind. In welcher Form und mit welchem Protokoll der Aufruf erfolgt spielt keine Rolle, sofern die aufrufende Web Applikation und das gerufene Service darauf abgestimmt sind.

5. Das aufgerufene Service generiert aus der Anfrage die notwendigen Daten und retourniert diese an die Web Applikation.

6. Die erhaltenen Daten werden auf das für den Client passende Format konvertiert. Die Daten können für weitere Aufrufe des Mashups zwischengespeichert werden.

7. Die Web Applikation liefert die Daten zurück an den Client.

8. Eine Java Script Callback Funktion am Client baut die notwendigen Teile der Webseite mit den erhaltenen Inhalten neu auf.

2.2.3. Mischformen

Nicht immer kann ein Mashup klar einer der beiden genannten Architekturen zugeordnet werden. Oft besteht es aus einer Kombination der beiden Varianten. Die Aufgaben können je nach Komplexität und Funktion auf dem Server oder dem Client ausgeführt werden. Das Aggregieren von Daten aus den einzelnen Services und die Konvertierung in passende Datenformate könnten am Server ausgeführt werden. Die Kombination der vom Server gelieferten Teilergebnisse und grafischen Elementen von weiteren Diensten (z.B. eine Landkarte) könnte am Client erfolgen. Tendenziell werden Aufgaben die eine erhöhte Performance für die Ausführung benötigen oder komplexe Algorithmen und Berechnungen beinhalten am Server implementiert werden, Anforderungen im grafischen Bereich am Client.

2.3. Gegenüberstellung Client-Side und Server-Side Mashup

Die direkte Gegenüberstellung der beiden Architekturen lässt jeweils bevorzugte Einsatzszenarien erkennen.

Die Implementierung eines server-side Mashups hat aufgrund der großen Anzahl von verfügbaren server-seitigen Sprachen und dem sehr großen Umfang an verfügbaren Bibliotheken klare Vorteile was die Erstellungszeit betrifft. Die Entwicklung eines client-seitigen Mashups birgt aufgrund der nicht vollständigen Kompatibilität der Java Script Interpretation am Client Risiken, die durch die Verwendung von Libraries mit bestimmten Basisfunktionen zwar minimiert aber nicht vollständig beseitigt werden können. Ständige Versionswechsel bei den Browsern und damit die verbundenen Änderungen in der Ausführung der Mashup relevanten Funktionen erhöhen das Risiko für client-side Mashups ebenfalls.

Auftretende Fehler bei der Erstellung des Mashups können bei einer server-seitigen Programmiersprache einfacher behandelt werden. Zwar bietet Java Script genauso wie Java oder C# Exception Handling an, das Testen und Debuggen von Java Script im Vergleich zu server-seitigen Programmiersparchen ist deutlich zeitintensiver. Fehler entstehen in erster Linie durch nicht erreichbare Services und durch nicht passenden Datenformate, wobei bei falschen Datenformaten durch eine entsprechende Konvertierung programmatisch eingegriffen werden kann (siehe 2.4).

Für das client-side Mashup spricht die fehlende Notwendigkeit der Bereitstellung einer Umgebung für die Ausführung der Web Applikation. Eine statische HTML Seite mit eingebetteten Java Script Funktionen und der Verweis auf Bibliotheken ist ausreichend.

Der Begriff Mashup wird gewöhnlich mit der Anzeige der Ergebnisse in einem Browser in Verbindung gebracht. Wie unter [37] angesprochen sind auch andere Client Systeme denkbar und in Verwendung. Beispielsweise verwendet *Google Earth* für die Darstellung der digitalen Karten einen eigenen Client, der das Kartenmaterial vom Google Earth Server erhält aber auch Material von anderen Quellen einbinden kann. Daneben können eigene Daten im Client auf den Karten eingeblendet werden, wie Geo-Koordinaten von Orten.

2.4. Ausgabe- und Transferformate

Die von den aufgerufenen Services gelieferten Daten werden in einem bestimmten Ausgabeformat zurück gegeben. Das Format ist entweder je nach Service vorgegeben oder kann gewählt werden. Grundsätzlich ist eine Konvertierung zwischen Datenformaten, unter der Voraussetzung der Verfügbarkeit der notwendigen Funktionen, möglich. Je nach Größe der Datenpakete und der Komplexität der Quell- und Zielformate kann dies sehr zeitaufwendig werden. Die folgenden Kapitel die wichtigsten Datenformate im Detail.

2.4.1. XML

Die XML (Extensible Markup Language) ist eine Auszeichnungssprache zur Darstellung semi-strukturierter Daten in Textform und wird sehr häufig für den Datenaustausch

zwischen Computersystemen oder der hierarchischen Beschreibung bestimmter Komponenten verwendet (besonders von UI Komponenten: JSF, ASP.NET, XAML, XUL, ...). XML kann auch als Metasprache und daher selbst für die Definition von Sprachen (und deren Keywords) verwendet werden. XML dient als Basis für Ausgabeformate wie RSS / ATOM (siehe 2.4.3) und SOAP (siehe 2.4.4). Die XML Spezifikation wird vom W3C weiterentwickelt [68].
Listing 2.1 zeigt eine einfaches Beispiel für Daten im XML Format zur Darstellung von Personendaten.

Listing 2.1: Beispiel XML

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <daten>
3       <titel>Teilnehmer</titel>
4       <eintrag>
5            <name>Mike Rhosoft</name>
6            <plz>3300</plz>
7       </eintrag>
8       <eintrag>
9            <name>Class Scheibe</name>
10           <plz>3100</plz>
11      </eintrag>
12 </daten>
```

Der Aufbau von XML Dokumenten folgt einer festgelegten Struktur, die vorkommenden Elemente müssen bestimmten Regeln folgen, die Bezeichnung der Elemente selbst ist jedoch frei. Um XML Dokumente validieren zu können, werden DTDs (Document Type Definition) oder XML Schemas eingesetzt.

Nach dem Erhalt von XML Daten von einem durch das Mashup aufgerufene Service müssen die Daten verarbeitet werden. Die Verarbeitung erfolgt in der Regel mit einer der folgenden Möglichkeiten:

- **DOM (Document Object Model):** Das XML Dokument wird auch softwaretechnisch als Baumstruktur repräsentiert, der Zugriff auf die Elemente kann wahlfrei erfolgen. Elemente können dabei sowohl gelesen als auch geändert werden, geänderte Dokumente im Speicher können in ein File zurück geschrieben werden. Da der Zugriff wahlweise erfolgen kann, muss das gesamte Dokument im Speicher gehalten werden. Ein DOM Parser ist im Vergleich zu einem SAX Parser langsamer und speicherintensiver, sein Vorteil liegt jedoch im wahlfreien Zugriff auf Dokumente (dies ist besonders bei interaktiven Anwendungen von Bedeutung).

- **SAX (Simple API for XML):** Das Dokument wird durch einen Datenstrom repräsentiert, auf den nur sequentiell (vorwärts oder rückwärts) zugegriffen werden kann. Ein SAX Parser durchläuft die einzelnen Elemente des Dokuments in der Reihenfolge ihres Auftretens, wobei für jeden auftretenden Elementtyp eigene Callback Funktionen definiert werden können (sind keine eigenen Callback Funktionen festgelegt, werden Standardfunktionen vom Parser aufgerufen). SAX wurde ursprüngliche in JAVA entwickelt, ist aber mittlerweile in praktisch allen gängigen Sprachen verfügbar. Ein SAX Parser ist vergleichsweise einfach und performant. Sollten bei der Validierung mittels SAX Fehler gefunden werden, kann dies im schlechtesten Fall auch erst gegen Ende der Verarbeitung auftreten (eventuell bereits ausgeführte Aktionen müssen dann wieder rückgängig gemacht werden).

- **XPath / XQuery:** XPath und XQuery sind die vom W3C spezifizierte Abfragesprache für XML Dokumente. XPath kann als Vorgänger von XQuery be-

trachtet werden. XPath ist eine domänenspezifische Sprache, XQuery ist Turingvollständig. XPath adresiert Teile des XML Dokuments, welches als Baum betrachtet wird.

- **Regular Expressions:** Finden besonders im Bereich der Softwareentwicklung Anwendung. Regular Expressions beschreiben syntaktische Regeln in Form von Zeichenketten zur Filterung einer Untermenge an Zeichen aus einer Menge an gegebenen Zeichen. Der Einsatz von Regular Expressions ist so gut wie in jeder Sprache durch die Einbindung von Bibliotheken möglich.

2.4.2. JSON

JSON (Java Script Object Notation) ist wie die meisten verwendeten Formate ein reines Textformat. Wenngleich der Name Java Script enthält, ist das Format nicht auf die Programmiersprachen Java Script beschränkt. Der Grund für die Bezeichnung ist mit dem Aufbau des Datenformates zu erklären: Für das Speichern der Daten werden anonyme Java Script Objekte verwendet, die sich wiederum aus Objekten oder primitiven Datentypen wie Strings, Zahlenwerten, usw. zusammensetzen können. Listing 2.2 zeigt die Darstellung von Personendaten mit JSON.

Listing 2.2: JSON Beispiel

```
{
   "ResultSet":{
      "totalResults": 2,
      "daten": [
         {
            "Vorname" : Mike,
            "Nachname": Rhosoft,
            "plz"     : 3300
         },
         {
            "Vorname" : Class,
            "Nachname": Scheibe,
            "plz"     : 3100
          } ]
    }
}
```

JSON Objekte werden als Text umgeben von geschweiften Klammern dargestellt. Objekte bestehen aus einer Kombination von Eigenschaften - Werte Paaren, die jeweils durch einen Beistrich getrennt werden. Ein solches Paar setzt sich aus dem Eigenschaftsnamen (String) und dem zugewiesenen Wert getrennt durch einen Doppelpunkt zusammen. Im direkten Vergleich zur XML ergibt sich ein deutlich geringerer Overhead durch die kompaktere Schreibweise. JSON Daten sind valider Java Script Source Code, der mit der `eval()` Funktion in ein Java Script Objekt umgewandelt werden kann.

Arrays werden innerhalb von eckigen Klammern eingeschlossen und können eine Liste von Werten (oder Objekten) getrennt durch einen Beistrich enthalten. Eine ähnliche Technik ist YAML (siehe 2.4.5), wobei es sich dabei nicht um gültigen Code in einer Sprache handelt, sondern um eine Auszeichnungssprache.

2.4.3. RSS / ATOM

Bei den Begriffen handelt es sich um zwei sehr ähnliche Technologien, die im eigentlichen Sinne miteinander konkurrieren, wobei ATOM als die Weiterentwicklung von RSS gesehen wird. RSS (Rich Site Summaray) wurde ursprünglich von der Firma Netscape entwickelt und konnte sich erst in den letzten Jahren im Bereich von Blogs und News Feeds durchsetzen. Bei der aktuellen Version RSS 2.0 wird RSS als *Really Simple Syndication* verstanden [24]. Listing 2.3 zeigt ein Beispiel für RSS Daten.

Listing 2.3: RSS 2.0 Beispiel

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <title>Data Feed</title>
    <description>A sample data feed</description>
    <link>http://data.org/</link>
    <lastBuildDate>
        Wed, 23 Sep 2009 10:02:30 GMT
    </lastBuildDate>
    <managingEditor>
        sue.permarkt@data.org (Sue Permarkt)
    </managingEditor>
    <item>
      <title>RSS Sample Feed</title>
      <link>http://data.org/2009/</link>
      <guid isPermaLink="false">
         urn:uuid:8575c695-ae6c-69ea-fbf0-45de342eec3f
      </guid>
      <pubDate>Wed, 23 Sep 2009 10:02:30 GMT</pubDate>
      <description>New data</description>
    </item>
  </channel>
</rss>
```

RSS Daten enthalten neben einem Titel und einem Autor nur einen kurzen Textauszug und einen Link zum Originaldokument. Ein RSS Feed liefert nach der erstmaligen Abbonierung in regelmäßigen Abständen neue Einträge, die mittels eines Clients (Feedreaders) heruntergeladen werden. Ein Feedreader kann eine eigene Software sein oder ist bereits im Browser inkludiert. Der Feedreader liest in periodischen Abständen von unterschiedlichen Quellen neue Einträge der RSS Feeds und kann der Definition zur Folge bereits selbst als eine Art *Mashup* betrachtet werden.

Die Notwendigkeit für die Entwicklung einer Alternative zu RSS folgt aus den Unzulänglichkeiten im Format [2], sowie aus der Tatsache, dass RSS von der Harvard Universität geschützt ist, und keine laufende Weiterentwicklung in Form eines Gremiums stattfindet [24]. Die RSS Spezifikationen sind urheberrechtlich geschützt und vorerst eingefroren, Weiterentwicklungen sollen unter einem anderen Namen stattfinden, z.B. ATOM. Listing 2.4 zeigt ein Beispiel.
Der ATOM Standard ist im RFC 4287 [40] niedergeschrieben und behebt Unzulänglichkeiten, welche sich in der RSS Spezifikation finden:

- RSS kann als Inhaltstypen nur plain text oder escaped HTML haben. Bei ATOM kann eine breite Auswahl an Inhalten verwendet werden. Der verwendete Typ

kann mit einem Attribut beschrieben werden.

- Im RSS Inhalt ist kein valides XML möglich. Der Inhalt von ATOM kann aus gültigem XML bestehen.
- Bei RSS Daten kann nur für den kompletten Dateninhalt die Sprache festgelegt werden. Einzelne Teilinhalte können mit der entsprechenden Sprache markiert werden.
- RSS besitzt nur drei obligatorische Elemente (Titel, Link, Beschreibung) und eine Reihe von optionalen Elementen. Vermisst werden allerdings Elemente wie z.B. ein Wert für das letzte Änderungsdatum.

Listing 2.4: ATOM 1.0 Beispiel

```
<feed xmlns="http://www.w3.org/2005/Atom">

 <title>Data Feed</title>
 <subtitle>A sample data feed</subtitle>
 <link href="http://data.org/feed/" rel="self" />
 <link href="http://data.org/" />
 <id>urn:uuid:8575c695-ae6c-69ea-fbf0-45de342eec4f</id>
 <updated>2009-09-23T10:02:30Z</updated>
 <author>
   <name>Class Scheibe</name>
   <email>scheibe@data.org</email>
 </author>

 <entry>
   <title>ATOM Sample Feed</title>
   <link href="http://data.org/2009/" />
   <id>urn:uuid:8575c695-ae6c-69ea-fbf0-45de342eec5f</id>
   <updated>2009-09-23T10:02:30Z</updated>
   <summary>New data</summary>
 </entry>

</feed>
```

ATOM stellt eine sehr robuste Spezifikation mit einem umfangreichen Feature Set dar, das von der IETF weiterentwickelt wird. Teil der ATOM Deklaration ist das **ATOM Publishing Protocol** (AtomPub oder APP), das im RFC 5023 [23] beschrieben ist.

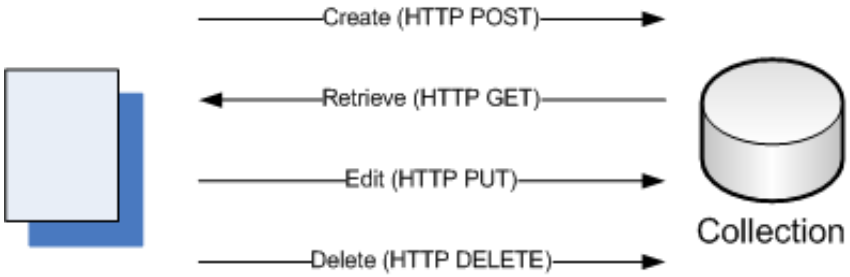

Abbildung 2.4.: ATOM Publishing Protocol

ATOM Publishing Protocol. AtomPub ist ein Protokoll auf der Applikationsebene, welches auf HTTP basiert und XML als Datenformat einsetzt. Der Client sendet XML Nachrichten mittels HTTP Kommandos (z.B. GET, POST, ...) an den Server. Die ausgeführten Operation beim AtomPub basieren auf *Collections* um die Ressourcen bearbeiten zu können. Um ein ATOM Dokument zu erhalten, wird ein HTTP GET Request an die URI des Feeds ausgeführt. Änderungen werden durch einen HTTP POST Aufruf an die URI des Feeds durchgeführt. In der Collection ergänzte Einträge erhalten jeweils einen eigene URI. Einträge können durch Aufruf über die spezifische URI auf den Client geladen, bearbeitet und wieder zurück geschrieben werden. Das Löschen von Einträgen erfolgt mit einem HTTP DELETE Aufruf. Alle Operationen haben gemeinsam, dass sie mittels Standard HTTP Aufrufen erfolgen, und können prinzipiell mittels Texteditor und Befehlszeile ausgeführt werden.

2.4.4. SOAP

SOAP ist ein Applikationsprotokoll basierend auf XML mit dem Daten zwischen Systemen ausgetauscht und Remote Procedure Calls [45] ausgeführt werden können. Das Protokoll wurde ursprünglich von einem Gremium bestehend aus IT Firmen (u.a. Microsoft, IBM) entwickelt und beim W3C eingereicht, wo es weiterentwickelt wird [47]. SOAP wird als Weiterentwicklung von XML-RPC gesehen [10].

SOAP stellt ein Framework mit Spezifikationen über den Aufbau und die Struktur der zu verwendenden Nachrichten dar. Es gibt keine Vorschriften über die Bedeutung der zwischen Systemen ausgetauschten Daten. Das Protokoll selbst ist zustandslos, im Rahmen der Applikation können aber komplexerer Interaktionsmuster durch die Kombination von Nachrichten implementiert werden.

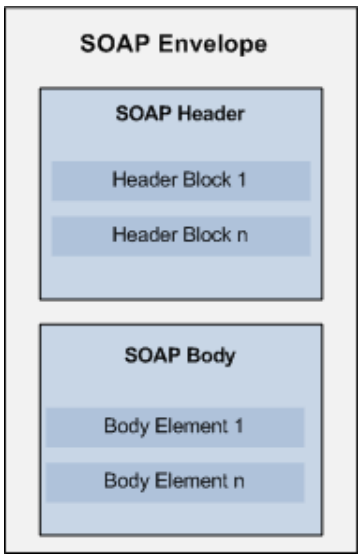

Abbildung 2.5.: SOAP Message

Als Transportprotokoll wird in vielen Fällen HTTP verwendet, SOAP arbeitet dennoch mit einer Reihe von Protokollen zusammen, beispielsweise FTP, SMTP, JMS. Der Einsatz von HTTP wird oftmals aufgrund der üblichen Netzwerkarchitekturen bevorzugt, da es in Verbindung mit Firewalls problemlos einsetzbar ist.

Die Verbindung zwischen Systemen, die mittels SOAP Daten austauschen, wird als lose Kopplung beschrieben [58]. Darunter wird der geringe Grad der Abhängigkeit zwischen den Systemen verstanden. Die Einzelsysteme agieren als autonome Komponenten, Änderungen der Hardware oder des Betriebssystems wirken sich relativ wenig bis gar nicht auf die anderen Einzelsysteme bzw. auf das Komplettsystem aus.

Eine SOAP Message besteht aus dem Element **Envelope** der die beiden Subelemente **Header** und **Body** enthält, siehe Abbildung 2.5. Der Header ist optional und kann zusätzliche Informationen übertragen, die nicht Teil der Nutzlast sind. Der Body ist obligatorisch und enthält die eigentlichen Informationen, die zwischen den Endsystemen ausgetauscht werden sollen. In Listing 2.5 ist ein Beispiel für eine SOAP Nachricht zu sehen.

Listing 2.5: SOAP Struktur

```
<?xml version="1.0"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
    <env:Header>
    </env:Header>
    <env:Body>
    </env:Body>
</env:Envelope>
```

Das SOAP Protokoll unterstützt 2 Arten von Nachrichtenformaten:

- Conversational Message: für den zustandslosen Austausch von Daten zwischen Systemen, bei dem der Hauptinhalt im *env:Body* Element enthalten ist.
- Remote Procedure Call Messages: kapseln die RPC Funktion im SOAP Protokoll, wobei das *env:Body* Element eine Struktur mit den aufzurufenden Funktionen enthält.

Die SOAP Spezifikation beschreibt noch eine Reihe weiterer Features wie z.B. Fehlerbehandlung und Verschlüsselung. An dieser Stelle wird für Detailinformationen auf die SOAP Dokumente vom W3C verwiesen [47].

2.4.5. Weitere Formate

Neben den in den vorigen Abschnitten beschrieben Formaten, sind für die Implementierung von Mashups eine Reihe weiterer Formate verbreitet.

HTML. Manche Mashup APIs liefern als Ausgabeformat HTML [33]. Datenstrukturen wie bei XML können hier nicht in beliebiger Form aufgebaut werden. Falls eine struktierte Darstellung notwendig ist, muss mit den in HTML vorhanden Möglichkeiten gearbeitet werden.

Plain Text und CSV. Im Unterschied zu den bisher beschriebenen Formaten haben reine Text- bzw. CSV Daten keine Struktur. Die Möglichkeiten zur Abbildung komplexer Dateninhalte ist mit diesem Ausgabeformat begrenzt. Mit CSV können nur relativ einfache Strukturen verarbeitet werden.

YAML. Ist ein Daten Serialisierungs Standard in einem einfach (auch für Menschen) zu lesenden Format. YAML (YAML Ain't Markup Language) Implementierung gibt es

für alle verbreiteten Programmiersprachen, die Spezifikationen sind unter [6] beschrieben. Ein Beispiel für Daten im YAML Format ist in Listing 2.6 zu sehen. Im direkten Vergleich zu XML werden die Daten nicht in Form von strukturierten Elementen sondern in Form von Listen oder assoziativen Listen dargestellt.

Listing 2.6: YAML Beispiel

```
--- # Block
name: Mike Rhosoft
plz: 3300
--- # Inline
{name: Class Scheibe, plz: 3100}
```

2.5. Klassifizierung von Mashups

Mashups werden je nach ihrer Anwendung und Inhalten in mehrere Kategorien eingeteilt [9], [63].

Mapping Mashups. Mashups dieser Art stellen die am weitesten verbreitete Art mit etwa 40% dar, und werden in [26] auf ihre strukturelle und sozio-technischen Eigenschaften untersucht. Mapping Mashups kombinieren typischer Weise unterschiedliche kartografischen Inhalte mit weiteren Informationen. Diese Informationen werden in grafischer Form als Overlay auf den Karten dargestellt. In sehr kurzer Zeit haben sich Mashups, die z.B. Google Maps, Microsoft Virtual Earth oder Yahoo! Maps als Basiselemente haben, etabliert. Die damit kombinierten Daten werden entweder manuell bereitgestellt oder aus einem Informationssystem automatisch mit den Kartendaten verknüpft.
Zum Beispiel können auf einer Karte öffentliche WLAN Hot Spots eingezeichnet werden.

Video und Foto Mashups. Diese Art von Mashups hat besonders in den letzten Jahren dank Plattformen wie `facebook.com` oder `myspace.com` einen rasanten Aufstieg erlebt. Die Fotos und Videos in sozialen Netzwerken besitzen meistens Metainformationen (was ist am Foto dargestellt, wann wurde es aufgenommen, wo wurde es aufgenommen), die in den Masuphs mit anderen Informationen kombiniert werden können.

News Mashups. News Mashups kombinieren Inhalte aus News Feeds, basierend auf RSS oder ATOM, zu personalisierten Online Newspapers, die auf Interessen des Anwenders abgestimmt sind. Beispielsweise könnten Inhalte bestimmter Kategorieren von `slashdot.org` und `digg.com` auf einer Webseite kombiniert werden.

Enterprise Mashups. Enterprise Mashups stellen ebenfalls eine eigene Kategorie dar und sind der zentrale Punkt dieser Arbeit. Für Enterprise Mashups sind besonders Punkte relevant, die sich aus dem Einsatz innerhalb einer Unternehmens-Infrastruktur ergeben. Dazu zählen die erhöhten Sicherheitsanforderungen, die Verfügbarkeit sowie die Datenqualität und Datenintegrität. Eine detaillierte Beschreibung und Abrenzungn zu den anderen Mashup Kategorien ist im Kapitel 3 zu finden.
Als Subformen eines Enterprise Mashups können Data Mashup und Process Mashup unterschieden werden:

Data Mashups. Diese werden zur aggregierten Darstellung diverser Datenquellen innerhalb eines Unternehmens verwendet. Datenquellen, die in Mashups kombiniert werden können, müssen innerhalb des Unternehmens verfügbar sein bzw. gemacht werden.

Process Mashups. Wird der Einsatz eines Mashups nicht nur als ad hoc Applikation betrachtet, sondern als wiederverwendbare Anwendung, spricht man von einem Process Mashup.

2.6. Wirtschaftlicher Nutzen von Mashups

Durch den Einsatz von Mashups und Enterprise Mashups (siehe 3) können Unternehmen eine Kosteneinsparung bei den IT Systemen erzielen sowie gleichzeitig die Agilität in den Geschäftsabläufen erhöhen [37]. Mitarbeiter erhalten mit einem Enterprise Mashup ein Werkzeug, das sich durch die Möglichkeit einer Verwendung je nach aktuellem Bedarf auszeichnet [62], [41]. Ist es mit Software und IT Systemen im klassischem Sinn so, dass die Umsetzung nach einem definierten Schema abläuft, fallen diese vorgegebenen Strukturen bei Mashups weg. Entsteht die Notwendigkeit im Rahmen des täglichen Geschäftsablauf nach einer speziellen Auswertung oder Anwendung, kann direkt darauf reagiert werden. Die Fachabteilungen können sich aus den bereitgestellten Datendiensten oder Diensten zur Prozesssteuerung die geforderte Anwendung erstellen. Durch die kurze Reaktionszeit von der Anforderung bis zum Einsatz und der begrenzte Bedarf an Ressourcen (im Sinne von Mitarbeitern) zur Erstellung, ist die Umsetzung eines Mashups relativ kostengünstig. Darüber hinaus erhöht sich die Flexibilität für die Fachabteilung um auf wirtschaftliche Gegebenheiten schnell und eigenständig reagieren zu können.
Ist in einem Unternehmen bereits ein SOA Projekt in der Umsetzung oder bereits umgesetzt, ist der Schritt zu Enterprise Mashups praktisch nur mehr darauf begrenzt, ein entsprechenden Mashup Framework bereit zu stellen. Die vorhandenen Datenquelle für die SOA Implementierung können auch als Input für Mashups verwendet werden. Enterprise Mashups können auch als eine alternative Visualisierungsschicht für eine SOA verstanden werden. Wird dieser Weg gewählt, muss allerdings davon ausgegangen werden, dass die Umsetzung der Mashups nicht vollständig in den Fachabteilung erfolgen wird bzw. kann. Bestimmte Basisfunktionen müssen anlässlich der Implementierung der SOA Grundstruktur bereitgestellt werden.

Je mehr spezialisierte Anwendungen im Unternehmen zum Einsatz kommen, desto größer ist die Gefahr, dass Daten redundant gespeichert werden. So sind z.B. Adressdaten im CRM System und im ERP System abgelegt. Sinnvoll ist der Zugriff von einem der beiden Systeme auf das andere System, das die Adressdaten als Datenquelle bereit stellt. Mit einer steigenden Anzahl von Systemen steigt auch die Zahl der Schnittstellen und der damit verbundene Wartungsaufwand mit der korrekten Funktion. Diese Bedürfnisse werden schon längere Zeit in *Enterprise Application Integration (EAI)* Projekten versucht abzudecken [39]. Die Verbindung und Synchronisierung von Systemen auf Datenebene stellt allerdings eine sehr kosten- und zeitintensive Aufgabe dar. Auch wenn hier eine SOA eine flexiblere Form der Integration von Systemen nicht nur auf der Datenebene sondern auch auf der Ebene der Geschäftslogik bieten kann [46], ist eine Umsetzung in vielen Unternehmen noch nicht angedacht. Eine relativ einfache und kostengünstige Form könnten als Zwischenlösung Enterprise Mashups darstellen. Das Potential von Mashups liegt darin, aus bereitgestellten Datenquellen ad

hoc durch Kombinationen von Datenquellen eine neue Anwendung zu erhalten, um die Aufgabenstellung zu lösen. Als Basis können definierten Funktionen und Daten über Dienste verfügbar gemacht werden, ohne der Notwendigkeit einer klar vorgegebenen strukturierten Umsetzung wie bei einer SOA folgen zu müssen.
Die Verfügbarkeit von Datenquellen aus verschiedenen Systemen bietet die Option Datenbestände kombinieren zu können, die ansonsten nicht direkt miteinander in Verbindung gebracht werden könnten. Hier werden Anwender aus rein technischer Sicht keine Schranken auferlegt. Einzig die Sinnhaftigkeit der erstellten Kombination ist eine Einschränkung.

Ein wichtiger Erfolgsfaktor eines Unternehmens ist das zur Verfügung stehende Wissen. Ziel jeder Organisation muss die kontinuierliche Erweiterung und Ergänzung des Wissesns sowie der konsequente Einsatz des gesammelten Wissens sein. Um *Wissensmanagement* innerhalb einer Organisation effizient umsetzen zu können, spielt neben einer Reihe organisatorsicher Faktoren der Einsatz eines entsprechenden IT-Systems eine große Rolle (In der voliegenden Arbeit wird ausschließlich auf die technologischen Faktoren eingegangen. Die angesprochenen Organisatorischen Aspekte sind unter [56] zu finden). Da das Wissen eines Unternehmens (oft aus historischen) Gründen nicht rein innerhalb eines Systems gespeichert ist, sondern unter Umständen auf eine Vielzahl von Systemen verteilt ist, muss der Prozess zum Suchen und Aufbereiten der Daten mehrere Quellsysteme unterstützen. Das Verbinden der Daten ausgehend von den Quellsystemen kann mittels einer Mashup-Applikation umgesetzt werden. Das Wissensmanagementsystem stellt einen Zugang zu einer Anzahl von Systemen dar, in denen das Wissen gespeichert ist. Der Such- und Verknüpfungsvorgang und die Präsentation der Ergebnisse sind die wichtigsten Funktionen dieser Software. Sind die Ursprungsdaten mit semantischen Anreicherungen gespeichert, kann im Rahmen des Suchprozesses eine Verknüpfung der Metadaten erfolgen. Durch die semantischen Metadaten können die gefundenen Informationen in Form von Ableitungen (z.B. semantische Netze) oder Strukturen dargestellt werden. Moderne Wissensmanagementanwendungen benutzen zum Speichern der Daten z.B. semantische Netze [12] oder Topic Maps [36]. Zur Darstellung von Wissen mit semantischen Ergänzungen eignen sich Metadaten in Form von Ontologien (siehe 4.2). Da das Wissen mit semantischen Informationen in navigierbarer Form aufbereitet werden kann, kann der Einsatz eines Wissenmanagementsystem auch aus einer anderen Perspektive gesehen werden, z.B. um es gleichzeitig als E-Learning System zu verwenden.

Auch wenn der wirtschaftliche Nutzen von Mashups bzw. Enterprise Mashups klar ersichtlich ist, gibt es in einem Unternehmen einige Hindernisse zu überwinden. So sind zwar bereits entsprechende Tools und Frameworks für die Erstellung von Mashups vorhanden, es fehlt allerdings noch an Referenzimplementierungen, um Unternehmen von der Sicherheit einer solchen Investition überzeugen zu können. Die wenigsten Unternehmen wollen sich bei Anschaffungen im IT Bereich auf Experimente einlassen [65].
Die Basiskomponenten für Mashups sind in der Regel Web Services. Die wenigsten Daten und Funktionen sind in einer Unternehmens IT über Web Services verfügbar, sofern keine SOA Infrastruktur vorhanden ist. Diese müssten speziell für Mashups entwickelt werden.
Auch wenn es bis dato nur wenig Erfahrung mit Enterprise Mashups gibt, ist das Potential vorhanden. Diese Arbeit soll eine Referenz für den erstmaligen Einsatz eines Mashups im unternehmerischen Umfeld darstellen, und damit verbunden Chancen und Möglichkeiten aufzeigen.

2.7. Abgrenzung zu anderen Technologien

Mashups stellen keine neue Technologie im eigentlichen Sinne dar. Sie sind die Anwendung einer Vielzahl von bereits bekannten und verwendeten Komponenten in einem neuen Zusammenhang. Aus diesem Aspekt besitzen Mashups Ähnlichkeiten zu anderen Mechanismen. Eine der Basiskomponenten von Mashups sind Datenquellen, die meist mit Hilfe von Web Services dem Ergebnis zugeführt werden. Web Services stellen den Kernbaustein einer SOA dar.

Definition SOA laut [8]:

> A Service-Oriented Architecture (SOA) is a software architecture that is based on the key concepts of an application frontend, service, service repository, and service bus. A service consists of a contract, one or more interfaces, and an implementation.

Freie Übersetzung:

> Eine serviceorientierte Architektur ist eine Softwarearchitektur, die auf den Schlüsselkonzepten eines Anwendungsfrontend, von Diensten, einem Repository und einem Service Bus aufbaut. Ein Dienst besteht aus einer Definition, einer oder mehr Schnittstellen und einer Implementierung.

Eine Hauptaufgabe einer SOA ist die Entkopplung der Business Logic von den eingesetzten Technologien. Services bieten Daten und Prozesse an, die konkret dahinter liegenden Techniken sind irrelevant. Im Regelfall wird das Frontend laufend mit den Anforderungen der Organisation verändert und erweitert, die Services selbst haben meist einen längeren Lebenszyklus.

Eine SOA besteht aus folgenden Schlüsselkomponenten [8]:

Application Frontend. Das Application Frontend sind die aktiven Komponenten einer SOA, sie initieren und kontrollieren die auszuführenden Aktivitäten. Das Frontend entspricht der oberen Schicht einer Multilayer Architektur.
Es werden unterschiedliche Arten von Frontends unterschieden. Das klassische Frontend ist eine Applikation mit einem grafischen User Interface, das dem Benutzer die möglichen Funktionen bereitstellt. Alternative Frontends sind Web Applikation oder Batch-Clients.

Services. Die Services führen die eigentlichen Aufgaben durch. Ein Service kann als Software Komponente verstanden werden, die einen bestimmten Prozess kapselt. Services bestehen nicht nur aus der eigentlichen Business Logic in Form von Source Code. Für den Aufrufer wichtig sind die Konventionen für den korrekten Aufruf. Diese Informationen werden mit Sprachen wie IDL (Interface Definition Language) oder WSDL (Web Service Description Language) bereit gestellt. Sie beschreiben das Interface des Services.

Service Repository. Für das Auffinden von Services wird ein Service Repository eingesetzt. Im Repository können weitere Information, die das Service beschreiben, hinterlegt sein, z.B. physikalische Position des Services, Daten über den Anbieter, Kontaktpersonen, Gebühren und Service Level. Im Rahmen einer SOA ist das Service Repository als Dienst innerhalb eines Unternehmens zu verstehen. Ein Repository ist ein

optionaler Bestandteil, und wird oft in Abhängigkeit der Serviceanzahl eingesetzt. Die von einem Repository angebotenen Services können in Form von *Development-Time Binding* (Service Signaturen werden bereits während der Entwicklung fest eingebunden) und *Runtime Binding* (Service Signaturen werden während der Ausführung gelesen und eingebunden) ausgeführt werden.

Service Bus. Die Anbindung aller SOA Teilnehmer innerhalb einer SOA erfolgt über den Service Bus. Der Service Bus muss eine Reihe unterschiedlicher Technologien unterstützen. IT Umgebungen in Unternehmen sind *heterogen* aufgebaut. Demzufolge muss es mit dem Service Bus möglich sein, Teilnehmer basierend auf verschiedenen Betriebssystemen, Programmiersprachen oder Laufzeitumgebungen zu verbinden, siehe Abbildung 2.6. Typische Techniken sind z.B. CORBA, RMI, .NET Remoting, diverse Messaging Protokolle, FTP, und weitere. Eine Charakteristik eines Service Busses ist die Möglichkeit die Nachrichten zwischen den Protokollen zu konvertieren. Damit kann der Input auf den Bus z.B. per FTP erfolgen, der Output aus dem Bus erfolgt mittels Web Service.
Eine Hauptaufgabe des Busses ist die *Verbindung* der teilnehmenden Komponenten. Er stellt Mechanismen zur Kommunikation von Frontend und Diensten bereit.
Ähnlich den heterogenen Umgebungen der Teilnehmer ist die *Heterogenität der Kommunikation.* Zumindest eine Möglichkeit zum Einsatz von synchroner und asynchroner Nachrichtenübertragung sollte vorhanden sein.

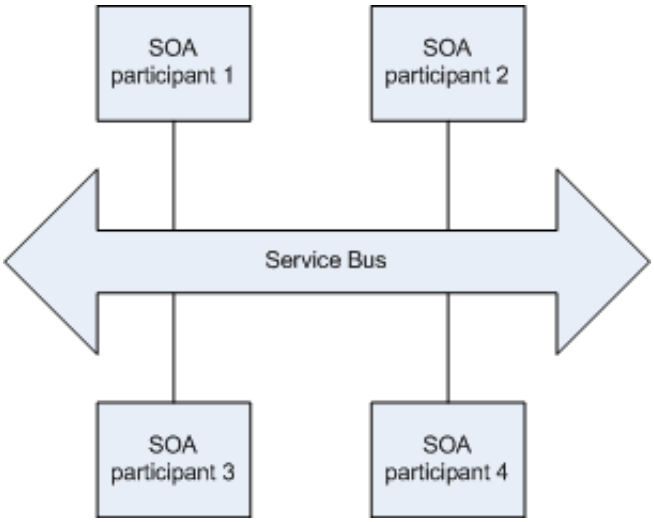

Abbildung 2.6.: SOA Service Bus

Ein Grundgedanke, neben Faktoren wie Kostenreduktion oder höherer Wiederverwendbarkeit von erstellten Diensten, ist es dem Anwender flexiblere und mächtigere Werkzeuge (im Sinne von Software) anzubieten. Eine SOA ist zwar als Architekturansatz für eine Unternehmens-IT sinnvoll, stellt selbst aber die angesprochenen flexibleren Werkzeuge für den Benutzer nicht zur Verfügung. Um dem Benutzer die Möglichkeit zur ad hoc Erstellung von neuen Anwendungen zu geben ist ein besserer Ansatzpunkt als eine SOA die Verwendung von Mashups. Diese setzen allerdings Dienste und Funktionen voraus, welche von Endanwendern nach Bedarf zu neuen Applikationen kombiniert werden können. Für die Bereitstellung dieser Dienste eignen sich als Ausgangsbasis Services, die bereits im Rahmen eines SOA Projekts erstellt wurden. Benutzer können die lose gekoppelten Dienste über ein Mashup-Framework in einem anderen Kontext

kombinieren. Aus diesem Aspekt sind Enterprise Mashups Schlüsselkomponenten für den initialen Start eines SOA Projektes.

Web Service Compositions Im Rahmen einer SOA werden die Dienste häufig in Form sogenannter Web Service Compositions (Dienstkompositionen) verknüpft. Es können zwei Arten unterschieden werden:

- Orchestrierung: es werden Services flexibel zu einer Komposition zusammengefügt, die einen Geschäftsprozess beschreibt. Die Ablauf des Prozesses wird dabei von einem Teilnehmer kontrolliert. Jeder der beteiligten Dienste hat einen beschränkten Bereich (Scope). Ein Beispiel für eine Sprache zur Orchestrierung ist WS-BPEL (Web Service - Business Process Execution Language).
- Choreographie: Hier liegt das Augenmerk auf dem Austausch von Nachrichten zwischen den Diensten. Es gibt keine zentrale Instanz, welche den korrekten Ablauf kontrolliert. Beispiel für eine Sprache zur Choreographie ist WS-CDL (Web Service - Choreography Description Language).

3. Enterprise Mashups

Wie bereits unter 2.5 angesprochen, stellen Enterprise Mashups eine eigene Kategorie von Mashups dar. Im Unterschied zu den anderen Mashup Kategorien, gelten für Enterprise Mashups besondere Voraussetzungen, die im folgenden Kapitel herausgearbeitet werden. Von der Basisfunktion als auch von den eingesetzten Technologien, unterscheiden sie sich nicht von herkömmlichen Mashups, allerdings gelten für Enterprise Mashups in bestimmten Punkten strengere oder andere Voraussetzungen. Diese Punkte betreffen unter anderem Verfügbarkeit, Datenqualität / Datenintegrität und Sicherheit.

3.1. Klassifizierung von Enterprise Mashups

Wie unter 2.5 beschrieben, stellen Enterprise Mashups eine Kategorie von Mashups mit besonderen Anforderungen dar. In gleicher Weise können Enterprise Mashups je nach deren konkreten Anwendungsbereich in zwei Arten unterteilt werden.

Data Mashups sind situationsbedingte ad hoc Anwendungen, welche kurzfristig die Bedürfnisse von Arbeitsgruppen oder Mitarbeitern erfüllen und keinen Anspruch auf längerfristige Verwendung erheben. Der Zugriff auf Datenquellen erfolgt durch exponieren in Form eines Dienstes. Bei einer eingeschränkten Anzahl an Datenquellen können diese in Form einer manuell gewarteten Liste bekannt gegeben werden. Steigt die Zahl der potentiellen Datenquellen im Rahmen eines SOA Projektes auf eine erhebliches Maß an, bietet sich der Einsatz eines Data Repository in Form einer Service Registry an. Benutzer können mit Hilfe des Repository die gewünschten Inputs für das Erstellen des Mashups recherchieren. Voraussetzung für den Einsatz einer Datenquelle sind Zugriffsrechte des Benutzers.
Data Mashups müssen nicht notwendigerweise ein grafisches oder ein web-basiertes User Interface besitzen. Unter Umständen besitzt diese Art von Mashups überhaupt kein User Interface. Das Ziel eines Data Mashups ist das Erzeugen einer neuen Aussage aus den kombinierten Daten. Die Darstellung selbst spielt eine untergeordnete Rolle. Umfasst das Data Mashup selbst kein User Interface kann das Rendering der gewonnen Daten in Form eines Service innerhalb der SOA angeboten werden. Dieses Service erhält als Input die erzeugten Daten des Data Mashups und liefert als Output eine grafische Aufbereitung der bezogenen Daten.

Im Unterschied zu Data Mashups, wo eine Verwendung meist auch die Erstellung des Mashups bedeutet, stellt ein **Process Mashup** eine regelmäßig eingesetzte Applikation dar. Neben der Wiederverwendbarkeit haben Process Mashups die spezielle Eigenschaft, dass neben dem Einsatz von reinen Datenquellen, auch Geschäftsprozesse eingesetzt werden können. Das Ergebnis eines Process Mashup stellt daher nicht nur eine Kumulierung mehrere Datenquellen dar, sondern lässt auch die Auslösung von Geschäftsprozessen zu. Das erstellte Mashup wird dadurch zu einem Werkzeug zur Steuerung von Unternehmensprozessen. In den Hintergrund rückt die reine situationsbezogene Erzeugung und der Einsatz eines Mashups. Primär geht es um die Automa-

tisierung von Abläufen sowie um die Kombination zu vollständig neuen Prozessen. Die Erweiterung um Geschäftsprozesse erklärt auch die erhöhte Wiederverwendbarkeit, da eine Erstellung eines Mashups mit Prozessanbindung für einen einmaligen Einsatz nicht kosteneffizient ist.

Gemeinsam haben beide Mashup-Arten die Positionierung im Rahmen der Unternehmens IT. Die Implementierung von Enterprise Mashups erfolgt entweder im Rahmen einer bereits bestehenden SOA oder verwendet Einzelkomponenten einer SOA. Die Datenquellen oder Geschäftsprozesse werden über SOA Technologien wie z.B. Web Services bereitgestellt und angesprochen. Ist eine SOA Infrastruktur unternehmensintern bereits vorhanden, kann das Enterprise Mashup die verfügbaren Dienste einsetzen. Für den Fall, das keine SOA Infrastruktur zur Verfügung steht, müssen für den erstmaligen Einsatz von Mashups, die notwendigen Daten- oder Prozessquellen in Form von Web Servcies zugänglich gemacht werden. In beiden Fällen ist die Platzierung des Mashups innerhalb der SOA Einzelkomponenten zu sehen, eine SOA muss jedoch nicht zwangsweise den Einsatz von Enterprise Mashups inkludieren. Vorhandene SOA Komponenten reduzieren die Erstellung von Mashups um die Aufbereitung der notwendigen Datenquellen, sofern diese bereits im Gefüge der SOA Infrastruktur aufbereitet sind. Ein Enterprise Mashup innerhalb einer SOA kann als Ergänzung in der Präsentationsschicht verstanden werden. Es wird in die Architektur eine Visualisierung eingezogen, um unterschiedliche Datenquellen zu kombinieren.

3.2. Anforderungen an Enterprise Mashups

An Enterprise Mashups werden spezielle Anforderungen, die aus dem Einsatz in einem Unternehmen resultieren, gestellt. Wird ein Mashup innerhalb eines Unternehmens eingesetzt, im Sinne eines Data Mashups oder eines Process Mashups, muss in bestimmten Punkten ein höheres Level sichergestellt sein. Besonders bei Process Mashups, wo eine Steuerung von Unternehmensprozessen stattfindet, muss großen Wert auf die Sicherstellung der erhöhten Anforderungen gelegt werden. Die zu unterstützenden Voraussetzungen werden in den folgenden Kapiteln erläutert.

3.2.1. Verfügbarkeit

Für eine Enterprise Architektur ist die Verfügbarkeit ein kritischer Parameter. Ein System muss mit den garantierten Verfügbarkeitswerten tatsächlich einsatzbereit sein. Das gleiche gilt für Enterprise Mashups, wo zur Sicherstellung der garantierten Verfügbarkeit einerseits das Mashup Framework als auch die einzelnen Dienste verfügbar sein müssen. Die quantifizierten Werte der Verfügbarkeit werden in den SLAs (Service Level Agreements) beschrieben. Ein wichtiger Wert innerhalb der SLAs ist die Uptime eines Systems. Die SLAs für Mashup Systeme hängen aufgrund der heterogenen und verteilten Basisdienste weitaus weniger vom Mashup Framework ab als von den Services. Die Uptime eines Mashups setzt sich sich aus der Uptimte des Frameworks sowie aus der Uptime der beiden Services zusammen:

$$Uptime = UptimeFramework * UptimeService1 * UptimeService2$$

Z.B.
$Uptime = 99\% * 98\% * 98\% = 95.1\%$

Generell gilt, jene Komponenten mit der geringsten Uptime hat den größten Einfluss auf die Uptime des Gesamtsystems. Zur Verbesserung der gesamten Uptime muss der Fokus auf die Verbesserung der SLAs der schwächsten Komponenten gelegt werden.

3.2.2. Datenintegrität

Der Begriff Datenintegrität fasst die *Konsistenz*, *Genauigkeit* und *Korrektheit* der Daten zusammen [8]. Die Mechanismen zur Sicherstellung der Datenintegrität sind konzeptionell die gleichen Mechanismen wie sie in relationalen Datenbanken enthalten sind. Die eigentliche Aufgaben dieser Verfahren ist die Sicherstellung, dass mit keinen 'falschen' Daten gearbeitet wird. Darunter ist zu verstehen, dass die Korrektheit der Daten innerhalb bestimmter Grenzen mit Hilfe von Regeln formuliert und gegen diese Regeln geprüft wird. Im Allgemeinen werden drei Mechanismen im Rahmen der Datenintegrität unterschieden:

Entitätenintegrität ist eines der Schlüsselkonzepte einer relationalen Datenbank. Mit ihr wird sichergestellt, dass jedes Tupel in einer Datenbank eindeutig identifizierbar ist. Die Umsetzung erfolgt mit Hilfe eines Primärschlüssels (Primary Key Contraint), der für jede Relation in der Datenbank angegeben wird. Mit Hilfe eines Primärschlüssels kann ein Tupel aus einer Relation eindeutig angesprochen werden. Der Schlüssel selbst kann aus einem einzigen Attribut oder aus mehreren Attributen bestehen.

Domänenintegrität oder Attributintegrität beschreibt die möglichen Werte, welche ein Attribut einer Relation annehmen kann. Mit der Domänenintegrität können Benutzereingaben oder Prozessergebnisse direkt auf ihre Gültigkeit den Wert betreffend validiert werden. Die Validierung kann auf Basis der Relation oder datenbankweit durchgeführt werden. Auf Attributebene ist unter anderem die Definition des Datentyps bereits ein Teil der Domänenintegrität.

Referentielle Integrität stellt die Synchronisierung zwischen Relationen sicher. Die Umsetzung erfolgt mit Hilfe von Kombinationen aus Primärschlüsseln und Fremdschlüsseln. Der Fremschlüssel wird auf einem oder mehreren Attributen einer Child-Relation spezifiziert. Die Attribute können nur jene Werte annehmen, für die es in der Parent-Relation entsprechende Einträge gibt. Mit der referentiellen Integrität sollen isolierte Daten verhindert werden, d.h. es kann keine Tupel in einer Child-Relation ohne Bezug zur Parent-Relation existieren. Referentielle Integrität muss bei allen Datenbankoperationen, die Daten verändern, geprüft werden.

Die beschriebenen Integritätsprüfungen können entweder auf der Datenbankebene oder in der Applikationsebene implementiert werden. Die Implementierung auf Datenbankebene besitzt den Vorteil der Unabhängigkeit von der eingesetzten Applikation. Bei der Umsetzung in der Anwendungsebene müssen die notwendigen Regeln in jeder entwickelten Applikation umgesetzt werden. Beim Enterprise Mashups spielt gerade diese Methode eine große Rolle. Bei einem Mashup wird in der Regel keine eigene Datenbank oder Datenbankebene entwickelt. Es basiert auf angebotenen Diensten, die ihre Daten aus Datenbanken beziehen bzw. abspeichern. Um Datenintegrität sicherstellen zu können, müssen Prüfungen innerhalb der Logik des Mashups erfolgen. Bei reinen Daten

Mashups , wo aus mehreren Datenquellen eine neue Gesamtaussage getroffen wird, spielten die Integritätsprüfungen keine Rolle. Es werden keine Daten verändert oder neue Daten erstellt. Hingegen Process Mashups können durch die Ansteuerung von Unternehmensprozessen auch Daten verändern. In diesem Fall müssen Integritätsprüfungen in der Business Logic des Mashups implementiert werden. Die Implementierung bedeutet einen vermehrten Programmieraufwand. Beim Weglassen dieser Prüfungen kann dies inkorrekte Daten bedeuten.

3.2.3. Datenqualität

Die Datenqualität oder Informationsqualität beschreibt die Qualität von Informationen. Unter der Qualität von Daten versteht man wie gut ein Datensatz brauchbar ist, um die Realität zu beschreiben. *Eurostat* definiert die Datenqualität anhand folgender Kriterien [18]:

- Relevanz
- Genauigkeit und Zuverlässigkeit
- Aktualität und Pünktlichkeit
- Kohärenz und Vergleichbarkeit
- Zugänglichkeit und Klarheit

Die Datenqualität bei einem Enterprise Mashup hat insofern direkte Auswirkungen, da die gelieferten Daten als Entscheidungsgrundlage für Mitarbeiter dienen. Weisen die Daten eine mangelhafte Qualität auf, werden unter Umständen strategisch schlechte Entscheidungen getroffen. Die Konsequenz sind nicht realisierte Gewinne bzw. im schlimmsten Fall Verluste für das Unternehmen.
Nicht alle Faktoren können bei einem Enterprise Mashup direkt beeinflusst werden. Das Mashup kombiniert bestehende Daten in einer ad hoc Vorgehensweise zu einer neuen Aussage. Für die Ausführung ist es auf bereits vorhandene Datenbestände angewiesen. Um eine hohe Datenqualität des Ergebnisses zu ermöglichen, müssen die Parameter *Genauigkeit und Zuverlässigkeit*, *Aktualität und Pünktlichkeit* sowie *Zugänglichkeit und Klarheit* bei den verwendeten Datenquellen gegeben sein. Die Parameter *Relevanz* und *Kohärenz und Vergleichbarkeit* liegen im Einflussbereich des Mashuperstellers selbst. D.h. die ersten drei Parameter werden durch Daten die Datenquellen selbst bzw. durch die technische Umsetzung der Datenquelle bestimmt.
Um optimale Voraussetzungen für aussagekräftige Mashups bieten zu können, muss die Datenqualität auf möglichst hohem Niveau angesiedelt sein. Die Verbesserung der Datenqualität ist ein Prozess, der durch ein Datenqualitätsmanagement begleitet wird. Weiterführende Informationen zu diesem Thema sind unter [22] zu finden.

3.2.4. Sicherheit

Beim Einsatz von Mashups im Enterprise Bereich spielt die Sicherheit eine große Rolle. Der Sicherheitsaspekt muss in den Bereich Übertragungssicherheit und Authentifizierung bzw. Autorisierung geteilt werden [14].
Auf Ebene der **Datenübertragung** kann die Sicherheit auf Message-Ebene oder auf Transportebene sichergestellt werden. Auf Transportebene kann die Verschlüsselung der Daten durch das SSL (Secure Socket Layer) Protokoll erfolgen. Der Vorteil dieser

Methode ist, dass der komplette Datentransfer verschlüsselt ist (Payload als auch Overhead). Es können keine Rückschlüsse aufgrund des analysieren Traffics auf den Inhalt oder die Bedeutung der übertragenen Nachrichten gemacht werden. SSL Implementierungen sind praktisch auf allen Geräten und Systemen verfügbar. Eine Manko dieser Variante ist das potentielle Risiko einer man-in-the-middle Attacke. Die gesicherte Verbindung kann entlang des Übertragungsweges geöffnet werden. Mit dem Endanwender wird zwar wieder eine gesicherte Verbindung aufgebaut, dieser hat jedoch keine Kenntnis davon, dass die Verbindung abgehört wurde. Der Grund für das Funktionieren dieses Angriffes, ist eine fehlende Authentifizierung bei SSL Verbindungen. Abhilfe kann durch den Einsatz von Zertikfikaten in Verbindung mit SSL erfolgen.
Als Alternative kann bei Web Service Protokollen wie z.B. SOAP nur der Payload der Nachrichten verschlüsselt werden. Die Grundstruktur der SOAP Message (siehe 2.4.4) ist unverschlüsselt in der Übertragung enthalten. Der Vorteil dieser Variante ist die gleiche Behandlung der gesendeten Messages entlang des Übertragungsweges wie die Behandlung unverschlüsselter Messages. z.B. im Bereich eines Firewall Ruleset, beide Varianten von Nachrichten können mit den gleichen Regeln abgedeckt werden.

In Verbindung mit Enterprise Mashups ist die Übertragungssicherheit zwischen dem Mashup Framework und dem Client abhängig von den Mechanismen des eingesetzten Frameworks. In der Regel setzen aktuelle Frameworks einen Web Browser als Client voraus. Demzufolge kann die Sicherung der Übertragung durch eine https Verbindung erfolgen. Detaillierte Informationen zu den Möglichkeiten sind im Kapitel 4.5 in Zusammenhang mit der Beschreibung der am Markt befindlichen Frameworks zu finden.
Die Sicherung der Datenübertragung von den Services zum Mashup Framework hängt von den Protokollen der Services ab. Werden die Dienste in Form von Web Services bereitgestellt, können je nach Protokoll die oben beschriebenen Varianten eingesetzt werden. D.h. entweder unterstützt das Protokoll selbst eine Verschlüsselung der Daten oder die Datenübertragung kann auf Ebene des Transportlayers gesichert werden. Bei Daten die aus z.B. Legacy Applikationen bereit gestellt werden, muss abhängig von der Applikation eine Lösung gefunden werden. Eine ausführliche Beschreibung zu diesem Thema ist für den implementierten Prototyp im Kapitel 6 zu finden.

Die **Authentifizierung** im Bereich eines Enterprise Mashups kann entweder durch Authentifizierung gegen die einzelnen Dienste erfolgen oder als einmaliges Single-Sign-On gegen das Mashup Framework. Die Authentifizierung gegen das Framework ist aus mehreren Gründen zu bevorzugen. Mit gemeinsamen Benutzerdaten für alle Services können Aktionen eines Benutzers protokolliert werden. Dies kann einerseits bei unternehmerisch sensiblen Daten relevant sein oder im Falle einer unternehmensinternen Kostenabrechnung über die Nutzung bestimmter Dienste relevant sein.

Die **Autorisierung** eines Benutzers ist im einfachsten Fall das Recht das Service zu verwenden. Im Bereich Enterprise Computing ist die Autorisierung über Dienste nicht immer so einfach zu treffen. Enterprise Dienste betreffen nicht nur die Bereitstellung von Daten. Es werden über Dienste vollständige Prozessaufrufe oder Teile von Prozessaufrufen angeboten. Nicht alle Benutzer dürfen alle Prozessabläufe auslösen. Für die korrekte Autorisierung müssen die Prozesse auf Dienstebene gesichert werden. Die Entscheidung über den Einsatz eines Services wird vom Service über die bereitgestellten Authentifizierungsdaten beim Aufruf getroffen. Eine ausführliche Beschreibung der Möglichkeiten zur Authentifizierung und Autorisierung im Enterprise Bereich ist unter [8] zu finden.

Unternehmerisch sensible Daten müssen mit den beschriebenen Mechanismen geschützt werden. Unter sensiblen Daten werden nicht nur jene Daten wie im Datenschutzgesetz 2000 [1] beschrieben verstanden. Im Datenschutzgesetz werden nur personenbezogene Daten als sensibel eingestuft. In Unternehmen sind allerdings weitaus mehr Daten als sensible anzusehen und mit dementsprechenden Maßnahmen handzuhaben. Als untermehmerisch sensibel gelten alle Daten im Unternehmen, die im Falle eines Diebstahls, für das Unternehmen schädigend verwendet werden können. Speziell wenn Daten über Dienste zur Verfügung gestellt werden, muss darauf geachtet werden, dass keine sensiblen Daten an nicht autorisierte Anwender übermittelt werden. Die Authentifizierung und Autorisierung der Services hat höchste Priorität.

3.3. Abgrenzung zu anderen Technologien

Enterprise Mashups überschneiden sich in Teilbereichen mit anderen Technologien. Ein Enterprise Mashup kann aber vom Gesamtpaket nicht diesen Technologien zugeordnet werden.

Composite Application. Viele Überschneidungspunkte vom prinzipiellen Ansatz gibt es zwischen einer Composite Application und einem Enterprise Mashup [15]. Composite Applications sind ein Teilaspekt des Software Engineerings, bei dem Anwendungen aufgrund bereits bestehender Teilfragmente kombiniert werden. Die Gesamtfunktion der Composite Application basiert teilweise auf bereits bestehenden Funktion und Datenquellen sowie aus ergänzenden Logiken. Die Techniken für Composite Applications sind nicht festgelegt. Die Basisarchitektur und Technologie kann dem einer SOA ähnlich sein, eine Composite Application ist jedoch nicht per Definition Teil einer SOA.

Ein Mashup (Anm. nicht Enterprise Mashup) kann als Gegenstück zu einer Composite Application betrachtet werden. Die primäre Anwendungsgebiete von Mashups sind im Web Bereich zu finden und bestehen in der Regel aus web-basierten (oftmals frei zugänglichen) Datenquellen. Demgegenüber steht eine Composite Application, die unternehmensintern entwickelt und ausgeführt wird. Als Datenquellen dient größtenteils das im Unternehmen vorhandene Datenmaterial, wobei der genaue Ursprung keine Rolle spielt. Quelle für diese Informationen sind üblicherweise Informationssysteme (z.B. ERP, CRM, usw.) mit entsprechenden Schnittstellen, Legacy Applications mit angepassten Interfaces oder auch ein direkter Zugriff auf Datenbanken.

Auch wenn Enterprise Mashups und Composite Applications viele Gemeinsamkeiten aufweisen, unterscheiden sich die beiden Mechanismen in wesentlichen Punkten.
Der Entwurf und die Entwicklung einer Composite Application wird von Personen mit entsprechender technischer Ausbildung durchgeführt. Der Rahmen für die Umsetzung der Applikation kann als unternehmensinternes Projekt durch die IT Abteilung oder als Projekt mit einem externen Dienstleister gesetzt werden. In keinem Fall wird eine Composite Application von Mitarbeiten aus den Fachabteilungen eines Unternehmens entworfen, ganz im Gegenteil zu einem Enterprise Mashup. Hier wird der komplette Prozess von der Planung über die Umsetzung und den Einsatz der Applikation vom Endanwender selbst ausgeführt. Vorausgesetzt die notwendigen Datenquellen und das Mashup Framework sind bereits verfügbar.

Der Weg zum fertigen Produkt unterscheidet sich grundlegend zwischen einer Composite Application und einem Enterprise Mashup. Im ersteren Fall wird die Umsetzung

als IT Projekt im klassischen Sinn verstanden, bei dem ausgehend von einer Anforderung eine Architektur für die Umsetzung erstellt wird. Die fehlende Architektur eines Enterprise Mashups ist ein konkreter Unterschied zur Composite Application. Ein Enterprise Mashup entsteht in Form einer Anforderung, das Mashup wird in einer ad hoc Vorgehensweise 'zusammengestellt'. Die Art der Umsetzung kann nicht mit den Abläufen aus dem Software Engineering verglichen werden. Prozessschritte wie das dokumentierte Testen oder eine Systemintegration fehlen gänzlich, der Vorgang kann sinngemäß als Trial and Error beschrieben werden.

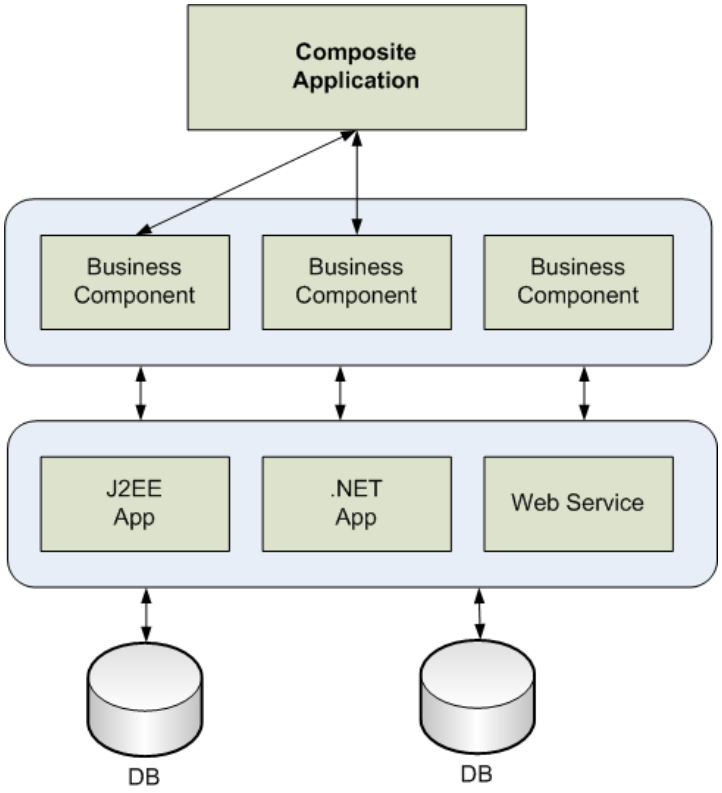

Abbildung 3.1.: Composite Application, angelehnt an [15]

Eine Composite Application basiert auf einem Composite Application Framework, in dem eine Schicht die Composite Business Logic ist, siehe Abbildung 3.1. Diese Schicht kombiniert die Basisdienste der Application zu den Bausteinen für die Entwicklung der Composite Application. Bausteine können neben dem Einsatz unterschiedlicher Datenquellen auch eine Business Logic beinhalten. Die Business Logic kann aus der erstellten Applikation aufgerufen werden, um Geschäftsprozesse anzustoßen bzw. zu steuern. Ein Enterprise Mashup besitzt ebenfalls eine Schicht, in der Logik enthalten sein kann. Diese dient in der Regel allerdings nur zur Kombination der Datenquellen. Eine Business Logic im eigentlichen Sinn, ist nicht Teil eines Enterprise Mashups.

BPM Business Process Management. Ein Business Process Management System unterstützt Unternehmen in der Formulierung und Umsetzung der Unternehmensprozesse [48]. Das BPM-System stellt dabei ein neutrales Rahmenwerk für die Umsetzung der Prozesse dar. Das BPMS aggregiert eine Reihe von Funktionen, die Geschäftsprozesse steuern. Die gesteuerten Prozesse als auch die ausführenden Personen können sich unternehmensintern oder extern befinden. BPM-Systeme sind in der Re-

gel auch mit Anforderungen konfrontiert, Geschäftsprozesse über Unternehmensgrenzen hinweg anzustoßen. Ein Unternehmen integriert Teile der Prozesskette eines Lieferanten oder eines Partnerunternehmens mit Hilfe des BPMS in seine eigenen Geschäftsprozesse.

Eine wichtige Eigenschaft eines BPMS ist die Integration in ein bestehendes IT Umfeld eines Unternehmens. Ziel des BPMS ist es, mit bestehenden Systemen zusammenzuarbeiten können, in der Form, dass Prozesse oder Teilprozesse aus den vorhandenen Systemen im Rahmen des BPMS zu einem gesamten Prozess ergänzt werden. Als typische Vertreter von BPM-Systemen sind ERP (Enterprise Resource Planning), CRM (Customer Relationship Management) oder WFM (Worlkflow Management) zu nennen. Die BPM-Systeme und Enterprise Mashups besitzen die Gemeinsamkeit, dass sie als Bindeglied in der Prozesskette fungieren. Es werden bereits bestehende Prozesse zu einem neuen Prozessablauf kombiniert. Ein Process Mashup umfasst ebenfalls diese Funktion. Der große Unterschied zwischen BPMS und Mashup besteht in der Art der Entwicklung der Applikationen. Mashups werden in einer ad hoc Manier entwickelt, BPMS werden wie unter 3.3 beschrieben in einem klassichen Software-Entwicklungs-Prozess erstellt. BPMS sind gewöhnlich hoch intergrierte Systeme, die aus einer Vielzahl von Programmteilen und dahinterliegenden Datenmodellen bestehen. Ein Eingriff in einem Teil eines solchen Systemems hat durch die enge Kooplung Auswirkungen auf andere Teilbereiche. Änderungen müssen deshalb innerhalb eines definierten Prozesses durchgeführt und in das System integriert und getestet werden. Änderungen oder Ergänzungen in Form von spontanen Elementen führen in diesen Systemen zu Komplikationen und müssen vermieden werden. Im Gegensatz dazu stehen Mashups, wo eine Neuentwicklung, Änderung oder Ergänzung praktisch immer spontan nach Bedarf erfolgt.

Data-Warehouse. Ein Data-Warehouse ist das Kernstück eines Data-Warehouse-Systems. Der Grundgedanke dieses Systems ist die Beschaffung und die Zusammenführung von Daten aus unterschiedlichen Basissystemen [28]. Die Daten werden bei Bedarf bereinigt und transformiert und im zentralen Datenlager, dem eigentlichen Data-Warehouse abgelegt. Die Befüllung der Datenbank erfolgt über einen ETL Prozess (Extract - Transform - Load) und wird auch mit dem Begriff der Informationsintegration beschrieben. Ziel eines Data-Warehouses ist die Unterstützung im Rahmen einer betriebswirtschaftlichen Entscheidungsfindung. Das gefüllte Data-Warehouse liefert eine zentrale Sicht auf heterogene Daten. Es ist der Ausgangspunkt für weiterführende Auswertungen wie z.B. OLAP (Online Analytical Processing).
Data-Warehouse und Enterprise Mashup (speziell ein Data Mashup) haben Gemeinsamkeiten bei den gelieferten Daten: beide erstellen aus mehreren Datenbeständen eine neue Aussage. Zwischen Date-Warehouse und Enterprise Mashup gibt es aber eine deutliche Abgrenzung bei der Erstellung und beim Einsatz der Systeme. Ein Data-Warehouse ist durch eine strukturierte Umsetzung und eine regelemäßige Pflege während des Betriebs charakterisiert. Dem gegenüber steht die ad hoc Vorgehensweise bei Mashups. Ein Mashup wird als Ergebnis auf einen kurzfristigen Bedarf erstellt und angewandt. Eine regelmäßige Pflege der Mashup Applikation oder der Datenbestände wird nicht durchgeführt.

Global. Generell können Mashups und Enterprise Mashups von den im vorigen Abschnitt beschriebenen Systemen durch die Art der Erstellung unterscheiden werden. Hinter einer BPM Applikation steht ein ingenieurmäßiger Prozess zur Entwicklung die-

ses Systems. Die Software wird im Rahmen eines Vorgehensmodells, das in zeitlich und inhaltlich abgegrenzten Phasen geteilt ist, erstellt. Der Ablauf wird üblicherweise von einem Projektmanagement sowie einer Qualitätssicherung begleitet. Bei den Vorgehensmodellen [64] werden unterschiedlich Ansätze unterschieden, z.B. sequentiell (Wasserfallmodell), iterativ (spiralförmig), usw. In jedem Fall nähert sich das erstellte Produkt mit jedem weiteren Schritt im Modell näher an das Endprodukt an. Das Ziel aller Vorgehensmodelle ist die kontrollierbare, vorhersagbare und flexible Erstellung der Software.
Der Basisvorgang zur Erstellung eines Mashups ist primär eine **Orchestrierung** [46] von bereits bestehenden Services und Funktionen zu einer gesamtheitlichen neuen Anwendung. Ein unterstützender Prozess ist nicht vorhanden, ein Mashup wird im Rahmen einer relativ kurzfristigen Anforderung umgesetzt. Vorgehensmodelle oder Qualitätssicherungsmaßnahmen werden nicht angewandt.

3.4. Mashups und Legacy Applikationen

In vielen Unternehmen spielen Legacy Applikationen eine wichtige Rolle. Eine Legacy Application stellt ein System dar, welches in der Regel nicht mehr am neuesten Stand der Technik ist, trotzdem aber weiterhin eingesetzt wird. Gründe für den weiteren Einsatz sind die hohen Kosten für die Ablöse durch ein neues System, der geringe Zugewinn an Effizienz durch ein neues System oder die Zufriedenheit mit dem bestehenden System.

Durch diese Gegebenheit ist es für Enterprise Mashups wichtig Legacy Applikation zu unterstützen. Oftmals decken diese Applikationen unternehmenskritische Bereiche ab. Ein fehlender Support für Legacy Applikationen würde ein Kriterium gegen den Einsatz von Enterprise Mashups darstellen. Um die Anbindung einer Legacy Applikation zu ermöglichen, muss das Enterprise Mashup (bzw. das verwendete Framework) entsprechende Möglichkeiten bieten. Konkret kann die Anbindung in Form eines direkten Zugriffs auf die Datenbank der Legacy Applikation erfolgen. Der direkte Datenbankzugriff stellt insofern einen Sicherheitsaspekt dar, dass unter Umständen Datenbereiche offengelegt werden, die als sensibel einzustufen sind (siehe 3.2.4). Ein alternative Anbindung ist die Entwicklung eines Wrappers für die Legacy Applikation. Die Wrapper Komponente greift auf die Daten, mit der von der Legacy Applikation bereitgestellten Schnittstelle, zu. Die Daten werden in die notwendige Darstellung transformiert, um mittels des Mashup Frameworks die Daten weiterverarbeiten zu können. Die Umsetzung des Wrappers bietet sich im Form eines Web Service an, das in das Mashup eingebunden wird. Eine detaillierte Beschreibung der Möglichkeiten ist im Kapitel 5.7 zu finden.

In den seltensten Fällen werden Legacy Applikationen bereits eine Schnittstelle bieten, mit der Daten als Web Service oder als XML Nachrichten abgeholt werden können. Eine Wrapper als Web Service ist eine universelle Vorgehensweise zur Anbindung von Applikationen ohne dieser bereitgestellten Unterstützung.

3.5. Semantic ERP

Die Anreicherung der Daten aus einem ERP System mit semantischen Informationen wird mit dem Begriff *Semantic ERP* beschrieben. Die Daten werden in einen neu-

en Kontext gesetzt. Die Motiviation ist die maschinelle Verarbeitung der Daten. Die Erzeugung der kombinierten Informationen erfolgt über Web Services, die über ein zentrales Enterprise Service Repository gesteuert werden. Der erzeugte Datenstrom wird mit semantischen Erweiterungen wie *RDF* (Resource Description Framework, siehe 4.2.1) bzw. *OWL* (Web Ontology Language, siehe 4.2.2) angereichtert. Die daraus entstehenden Ontologien zu den ERP Daten können durch automatische Prozesse weiterverarbeitet werden. Erste Ansätze in diese Richtung sind bei SAP oder beim Theseus Projekt zu finden [5], [49]. Das Theseus Projekt ist ein vom deutschen BMWi (Bundesministerium für Wirtschaft und Technolgie) initiertes Forschungsprogramm. Es hat als Ziel Zugang zu Informationen zu vereinfachen und Daten zu neuem Wissen zu vernetzen. Ein Teilbereich ist die angesprochene Erweiterung von ERP Systemen mit semantischen Daten.
Die Anreicherung von ERP Systemen beruht auf dem REA Modell (Resource-Event-Agent) [51]. Das Modell beschreibt, ähnlich wie im Semantic Web, die Verlinkung von ökonomischen Ereginissen zwischen Handelspartnern. Ziel ist die Verbindung von ERP Systemen über Unternehmensgrenzen hinweg zu einem eignen 'Semantic Web'. Nach wie vor existiert das REA Modell nur als theoretisches Konzept. Erste Ansätze zur Umsetzung sind wie oben beschrieben in aktuellen ERP Versionen zu finden.

Teil II.

Prototyp

4. Technologien

Nachdem in den Kapiteln 2 und 3 die Basiskomponenten für Mashups erläutert wurden, wird in diesem Kapitel auf die speziellen Technologien für den Prototyp näher eingegangen. Die beschriebenen Themen beziehen sich einerseits auf den Einsatz von Mashups selbst (Web Service, Dynamics AX) als auch auf notwendige technologische Ergänzungen (Ontologien, Semantic Web), um die Anforderungen des umgesetzten Prototyps erfüllen zu können (siehe Kapitel 5).

4.1. Web Service

Als typische Datenquelle für Mashups dienen üblicherweise Web Services. Ein Web Service stellt Daten in einem bestimmten Format bereit, wobei die Unabhängigkeit von der Plattform, der Programmiersprache und dem Betriebssystem ein vorrangiges Ziel eines Web Services ist. Basis für diese Unabhängigkeit sind standardisierte Protokolle, mit denen die Schnittstellen von Web Services beschrieben, Web Services veröffentlicht und gesucht bzw. gefunden sowie Web Services aufgerufen werden können. Web Services auf der Basis von SOAP (siehe 2.4.4) und REST bilden die größte Gruppe. Die eigentliche W3C Spezifikation beschreibt unter einem Web Service die Implementierung mit SOAP [73].

Für die Interoperabilität zwischen einem Web Service und einem Client gibt es abgesehen von den Protokollen keine Voraussetzungen. Um die Zusammenarbeit zwischen Web Service und dem Consumer eines Web Services zu gewährleisten, ist es im Gegensatz zu anderen Mechanismen zum Aufbau von verteilten Systemen, nicht relevant welches Betriebssystem oder Programmiersprache für die Erstellung und den Betrieb des Web Services verwendet wird. Dieser geringe Grad der Abhängigkeit zwischen den Einzelkomponenten des Gesamtsystems wird als *lose Kopplung* bezeichnet [58]. Änderungen an den einzelnen Komponenten (bezogen auf z.B. Hardware oder Betriebssystem) wirken sich wenig bis gar nicht auf das Gesamtsystem aus.
War am Beginn der typische Einsatzbereich eines Web Services als ergänzende Komponente im Rahmen von Enterprise Application Integration zu suchen, hat sich das Einsatzgebiet von Web Services als eines der Hauptelemente für die Umsetzung einer SOA Architektur entwickelt. Zwar sind Web Services nicht die einzige Datenquelle im Kontext einer SOA Umgebung, sie stellen aber einen wichtigen Bestandteil dar, siehe Abbildung 4.1. Ein weiteres Einsatzgebiet für ein Web Service ist die Kopplung von Anwendungen innerhalb einer Unternehmens-IT, siehe auch Kapitel 2.6 und 3.4. Um die Daten zwischen Anwendungen zu synchronisieren oder direkt die Daten aus einer anderen Anwendung einzubinden, können Web Services eingesetzt werden. Die Web Services können in diesem Zusammenhang als eine Art *Middleware* verstanden werden, über die Unternehmensanwendungen kommunizieren (ähnlich einem Enterprise Service Bus, siehe Kapitel 2.7). Im Web haben sich Web Services als Schnittstelle für das Bereitstellen von Funktionen etabliert. Die Services sind oftmals frei zugänglich, bzw. sind per Authentifizierung und damit einhergehendem Accounting erreichbar. Typische

Dienste sind z.B. die Einbindung von Suchfunktionalitäten einer Suchmaschine in einer Applikation oder die Validierung von Postanschriften.

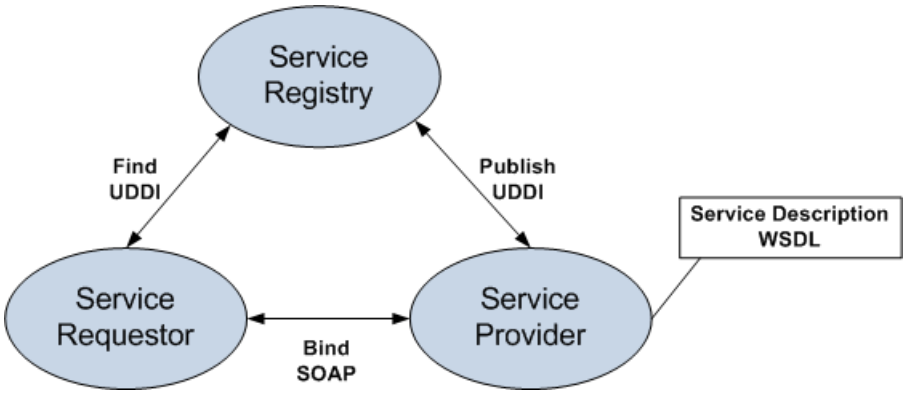

Abbildung 4.1.: SOA - Web Service

Für den Einsatz und Betrieb von Web Services im Unternehmensumfeld sind besonders Plattformen und Programmiersprachen im Bereich Microsoft .NET und Java Enterprise zu finden. Grundsätzlich können Web Services mit praktisch allen verbreiteten Sprachen erstellt werden. Voraussetzung ist, dass die entsprechenden Bibliotheken bereits im Umfang der Sprache vorhanden sind, oder zusätzlich installiert werden. Für den Betrieb im Unternehmens- oder Webumfeld sind außer der erstellten Software prinzipiell nur ein Web Server notwendig. Vorteilhaft in Verbindung mit Firewalls ist der Einsatz von http als Übertragungsprotokoll. Dieser vermeintliche Vorteil, dass nur wenig Schritte für die Inbetriebnahme des Web Service notwendig sind, kann sich im Bezug auf die Sicherheit als Nachteil erweisen. Durch Unachtsamkeit oder Unüberlegtheit könnten sensible Daten über eine Service nach außen hin bereit gestellt werden.

4.2. Ontologien

Für die maschinelle Verarbeitung von digitalem Wissen werden in der Informatik Ontologien verwendet. Ontologien beschreiben formal mit einer Menge von Begrifflichkeiten die Beziehungen zwischen bestimmten Bereichen. Für die Interpretation von Daten ist es essentiell die verwendeten Benennungen nach einer gemeinsamen Vorschrift zu verarbeiten. Diese Gesetzmäßigkeit ist bei Ontologien in Form von Inferenz- und Integritätsregeln vorhanden [67], [57]. In Verbindung mit dem Semantic Web (siehe Kapitel 4.3) werden Ontologien für die Repräsentation des Wissens eingesetzt. Wie im Prototyp Ontologien verwendet werden können, ist im Kapitel 5.2 beschrieben.
Ontologien stellen Metainformationen dar mit denen das gespeicherte Wissen in zusammenhängender Form verarbeitet werden kann. Die Verarbeitung der Daten erfolgt durch Modellierung der Ontologie mit Klassen, Relationen, Funktionen und Axiomen.

Die in der Informatik am weitesten verbreitete Definition einer Ontologie ist unter [50] zu finden:

> An ontology is an explicit specification of a conceptualization.

Die Konzeptualisierung beschreibt ein gemeinsames Verständnis der Begriffe eines Anwendungsbereiches. Eine Ontologie wird demnach immer speziell für ein bestimmtes Gebiet entwickelt. Die entwickelte Ontologie ist eine formale Beschreibung der Gegebenheiten der realen Welt, jeweils bezogen auf den relevanten Kern. Die Bestandteile

von Ontologien sind Begriffe (bzw. Klassen) beschrieben durch Typen und Relationen sowie von den Klassen erzeugte Instanzen. Die Bestandteile einer Ontologie existieren in einem bestimmten Kontext und stehen miteinander in Beziehung. Die Repräsentation des Wissens erfolgt über Axiome. Diese Aussagen werden verwendet um die Zusammenhänge zwischen Objekten innerhalb und zwischen Ontologien zu beschreiben. Aufgrund dieser Beziehungen können Aussagen und Schlussfolgerungen aus den vorhandenen Daten abgeleitet werden.

In der Informatik werden Ontologien in drei Anwendungsbereiche unterteilt [38]: *Kommunikation*, *Automatisches Schließen und Repräsentation* sowie *Wiederverwendung*. Die Kommunikation zwischen Anwendungen erfordert eine gemeinsame Vorschrift zur Interpretation der übermittelten Daten. Diese Basis muss entweder allen Teilnehmern bekannt sein oder sie wird in Form einer Ontologie bekannt gegeben. Das automatische Schließen ermöglicht es Anwendungen logische Folgerungen aufgrund der Ontologien aus den Daten zu ziehen. Gleiches gilt für die Wissensrepräsentation und Wissenswiederverwendung [56].

Konkret reichen Ontologien in der Informatik von einer einfachen Taxonomie bis hin zu einer umfangreich ausgeführten Ontologie [30]. Neben dem bereits im Kapitel 2.6 angesprochenen Einsatz bei Wissensmanagementsystemen sind folgende Anwendungsfelder in der Praxis anzutreffen:

E-Business. Zur Beschreibung und Klassifizierung von Produkten sowie Dienstleistungen wird ein standardisiertes Vokabular verwendet. Die formale Beschreibung von Artikeln ermöglicht Kunden und Verkäufern die Überwindung der Heterogenität im Verständnis von Waren.

EAI (Enterprise Application Integration) und SOA. Die Daten in unabhängigen Einzelsystemen einer Unternehmens-IT können ohne weiteres Eingreifen nicht mit Hilfe einer maschinellen Verknüpfung aufbereitet werden. Die meist historisch gewachsenen Systeme einer IT-Landschaft sind Einzelsysteme, die jeweils über ihre eigene Domäne an Begriffen verfügen. Die Herausforderung besteht in der Entwicklung einer neuen bzw. dem Einsatz einer bereits verfügbaren Ontologie, um mit Einzelsystemen innerhalb einer EAI Umgebung die automatische Interpretation der Daten sicherzustellen [7]. Das Ermöglichen der maschinellen Datenverarbeitung im Rahmen von EAI kann auch über die Unternehmensgrenzen hinaus gehen. Der Aufbau einer Kommunikation über Organisationsgrenzen hinweg stellt ebenfalls eine Anwendung dar.

Der Anwendungsbereich EAI stellt das eigentliche Thema der vorliegenden Arbeit dar. Die exemplarische Implementierung eines Prototyps für den EAI Einsatz wird ab Kapitel 5 behandelt.

Im Bereich SOA Integration kann durch Weiterentwicklung von Web Services zu Semantic Web Services erreicht werden, dass Web Services anhand semantischer Ergänzungen auf der Basis von Ontologien gefunden und ausgeführt werden können [13].

Semantic Web. Sollen bestimmte Informationen durch Mitarbeiter eines Unternehmens gefunden werden, ist momentan die manuelle Suche nach bestimmten Schlüsselwörten in Datenbeständen der einzige Weg. Diese Art der Suche führt bei immer größer werdenden Datenmengen und steigender Anzahl von Quellsystemen zu Ergebnissen, die von der ‘Geschicklichkeit’ und dem Einsatz des Benutzer abhängen. Die Qualität und

die Konstanz der Ergebnisse unabhängig vom Benutzer kann durch den Einsatz semantischer Informationen verbessert werden. Mit semantischen Ergänzungen bleibt die Entscheidung über die Relevanz von Information nicht allein dem Anwender über, die Entscheidung kann maschinell unterstützt werden, siehe auch Kapitel 4.3.

Für die digitale Darstellung von Ontologien in der Informatik haben sich das *Resource Description Framework* und die *Web Ontology Language* etabliert:

4.2.1. Resource Description Framework - RDF

Das RDF stellt eine Sprache zur Repräsentation von Ressourcen im World Wide Web dar [52]. Die ursprüngliche Entwicklung diente zur ergänzenden Beschreibung von Webseiten mit Informationen wie Titel, Autor, Änderungsdatum usw. Diese ergänzten Objekte oder Ressourcen werden durch URIs identifiziert. Durch die Erweiterung des Begriffes Ressource von den ursprünglich durch RDF unterstützten auf alle durch URIs beschreibbare Ressourcen kann RDF für eine Vielzahl von Anwendungen eingesetzt werden. Ziel von RDF ist die Verwendung in Situationen wo Daten automatisiert durch Software verarbeitet werden sollen. RDF stellt eine Möglichkeit zum Datenaustausch zwischen Sytemen dar. Für den Einsatz in einem Programm kann auf eine Reihe von RDF Werkzeugen und Parser zurückgegriffen werden. Neben der Kommunikation stellt die Suche in Datenbeständen (mit z.B. SPARQL) einen eigenen Anwendungsbereich dar.

Die Modellierung von RDF Daten erfolgt über gerichtete Graphen. Das RDF Modell enthält eine Reihe von Aussagen über Ressourcen, die in Form von Tripel dargestellt werden. Jedes Tripel besteht aus einem Subjekt einem Prädikat und dem Objekt. Das Subjekt entspricht der Ressource, die beschrieben wird. Das Prädikat entspricht einer Eigenschaft des Subjekts und des Objekt als Argument zum Prädikat. Alle vorhandenen Tripel bilden den Graph. Die Aussagen sind jeweils in gerichteter Form, vom Subjekt zum Objekt und mit dem Prädikat gekennzeichnet zu verstehen. Subjekt und Prädikat müssen immer Ressourcen sein, das Objekt kann auch ein Literal sein.

RDF wurde entwickelt, um eine einfache Möglichkeit zu schaffen über alle Arten von Ressourcen aussagen machen zu können. Die Identifizierung von Ressourcen über URIs lässt eine globale Verwendung von Ressourcen zu, also alle jene Informationen, die in ihrere Domäne eindeutig beschrieben werden können. Als Ressourcen können Webseiten mit URL (z.B. http://big.tuwien.ac.at), E-Mail Accounts (z.B. officebig.tuwien.ac.at) oder Bankkonten durch BIC und IBAN eindeutig beschrieben werden. Eine Ressource mit ihrer URI muss nicht zwangsläufig über Web erreichbar sein, die Aussagen mit Hilfe der RDF beziehen sich nicht auf die technische Erreichbarkeit der Ressource.

Folgendes Beispiel soll eine gültige Aussage verdeutlichen:
Die Person mit der Matrikelnummer 0555 hat den Namen Class Scheibe und besitzt die E-Mail Adresse class.scheibe@tuwien.ac.at
Das Beispiel ist in Listing 4.2 in RDF Schreibweise und in Listing 4.2 in N3 Schreibweise dargestellt. Diese Aussage würde in RDF modelliert wie in Abbildung 4.2 dargestellt werden.

Als Datenformat kann das vom W3C spezifizierte XML- Format RDF/XML verwendet werden. Alternativ gibt es eine für den Menschen besser lesbare Form, die Notation 3 (N3) [61].

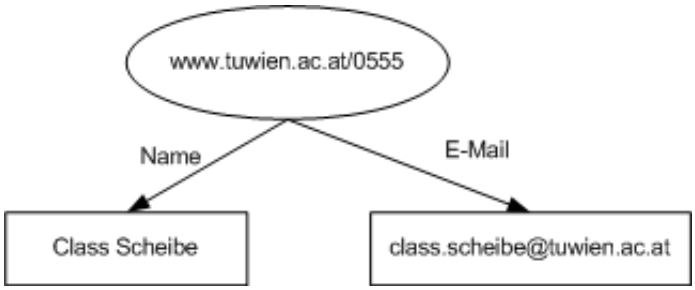

Abbildung 4.2.: RDF

Listing 4.1: RDF/XML

```
1  <rdf:RDF>
2     <rdf:Description about = "www.tuwien.ac.at/0555">
3        <v:Name>Class Scheibe</Name>
4        <v:E-Mail>class.scheibe@tuwien.ac.at</v:Email>
5     </rdf:Desription>
6  </rdf:RDF>
```

Listing 4.2: N3

```
1  <Class Scheibe> has <http://purl.org/dc/elements/1.1/email> "class.scheibe@tuwien.ac.at"
```

4.2.2. Web Ontology Language - OWL

Zur Darstellung von Ontologien in Form einer formalen Beschreibung kann OWL eingesetzt werden [69]. Wie bei der RDF geht es darum, Ausdrücke und deren Bedeutung so zu beschreiben, dass sie automatisiert zu verarbeiten sind. Die OWL kann als Erweiterung von bereits bestehenden Standards wie XML, RDF und RDF Schema verstanden werden. Die OWL ist in drei Versionen unterteilt. Zwischen den Versionen ist ein Unterschied in der Ausdrucksstärke gegeben, wobei in Abhängigkeit der notwendigen Ausdrucksstärke ein deutlicher Unterschied in der notwendigen Rechenleistung zur Verarbeitung zu vermerken ist. Die Versionsunterteilung der OWL ist folgendermaßen gegliedert:

OWL Lite In der OWL Lite liegt das Hauptaugenmerk auf der einfachen Verwendung der Sprache. Es fehlen Ausdrucksmöglichkeiten für Mengen (Konjunktion, Disjunktion, usw.), Kardinalitäten können nur mit 0 oder 1 festgelegt werden und äquivalente Klassen können nur eingeschränkt festgelegt werden. Einsatzbereich der OWL Lite ist z.B. die Defintion von Taxonomien.

OWL DL OWL DL ist jene Subsprache von OWL, die ein Maximum an Ausdrucksstärke mit der Berechnungsvollständigkeit sowie der Entscheidbarkeit kombiniert. OWL DL enthält alle Sprachkonstrukte der OWL mit einigen Einschränkungen im Bereich der Typen, Klassen und Kardinalitäten. Die Kompatibilität mit RDF Dokumenten ist nicht gegeben. Die Darstellung in OWL DL setzt eine Umformung voraus.

OWL Full OWL Full ist vollständig zu RDF kompatibel. Jeder Schluss in RDF ist auch in OWL Full gültig bzw. umgekehrt. Durch die Aufhebung der Einschränkungen aus der OWL DL ist die OWL Full zwar mächtiger in ihrer Ausdrucksstärke, besitzt jedoch keine Entscheidbarkeit und Berechnungsvollständigkeit. Das in OWL Full gespeicherte Wissen kann exakt wiedergeben werden, und Datenbestände können auf ihre Richtigkeit überprüft werden.

Listing 4.3 zeigt ein OWL Beispiel bei dem eine OWL Klasse `Person` angelegt wird. Von der Klasse wird eine konkrete Instanz erstellt und mit Werten belegt.

Listing 4.3: OWL

```
<rdf:RDF
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntaxns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#">

 <owl:Class rdf:ID="Person"/>

 <owl:DatatypeProperty rdf:ID="name"
  rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Person"/>
 </owl:DatatypeProperty>
 <owl:DatatypeProperty rdf:ID="email"
  rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Person"/>
 </owl:DatatypeProperty>

 <Person rdf:ID="Class Scheibe" name="Class Scheibe">
  <email rdf:resource="class.scheibe@tuwien.ac.at"/>
 </Person>
</rdf:RDF>
```

4.3. Semantisches Web

Die Basis für das semantische Web wurde im vorigen Kapitel beschrieben. Auf dieser Basis aufbauen wird im aktuellen Kapitel kurz das semantische Web erläutert. Unter 5.2 ist ein Weg angedacht, wie die semantische Anreicherung der Daten im Prototyp eingesetzt werden kann.

Eine ausführliche Definition des Begriffs findet sich beim W3C Konsortium [32]:

> The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners.

Bei dieser Definition steht die Interoperabilität zwischen unterschiedlichen Systemen im Vordergrund und die damit verbundene Aufhebung von Grenzen zur automatischen Verarbeitung von Informationen. Im Semantic Web steckt die Idee von Tim Berners-Lee durch Anreicherung der Daten mit semantischen Metadaten eine maschinelle Verarbeitung zu ermöglichen. Die Voraussetzung dafür ist die Schaffung von Ontologien, die als Metadaten die Bedeutung und Verknüpfung von Daten festlegen, siehe Kapitel 4.2. Als Resultat können Suche und Kommunikation durch Systeme unterstützt bzw. selbstständig durchgeführt werden. Zur Umsetzung des Semantic Web müssen zu den Informationsquellen die Bedeutung der Daten ergänzt werden. Die Voraussetzung der maschinellen Verarbeitung beruht auf dem Vorhandensein dieser Ergänzungen. Um das maschinelle Verständnis zu ermöglichen, muss möglichst ein einheitliches Format verwendet werden. Für die Verknüpfung von Daten über Domänengrenzen hinweg müssen die Daten anhand ihrer Bedeutung korrekt kombiniert werden können. Dafür werden Ontologie-Sprachen wie RDF und OWL, wie in den vorhergehenden Abschnitten beschrieben, eingesetzt.

Als grundsätzliche Alternative zum Ansatz des Semantic Web könnte künstliche Intelligenz mit ähnlichen kognitiven Fähigkeiten wie die des Menschen eingesetzt werden.

Mit neuronalen Netzen könnten die Informationen im Web in maschinell interpretierbare Daten umgewandelt werden. Dieses Forschungsfeld ist zum aktuellen Stand noch nicht genügend weit fortgeschritten, um es in der Praxis einsetzen zu können.

Abfragesprache SPARQL SPARQL (SPARQL Protocol and RDF Query Language) ist eine graph-basierte Abfragesprache zum Suchen und Extrahieren von RDF Daten [11]. Neben SPARQL existieren noch weitere Abfragesprachen wie Vera, SeRQL oder RDQL. SQARQL ist von der Syntax ähnlich zu SQL. Mit SPARQL werden deklarative Abfragen formuliert, Subjekte, Prädikate und Objekte können durch Variablen ersetzt werden. Das beschriebene Graphmuster stimmt mit einem Teilgraphen überein, wenn die Variablen durch die Werte aus dem Teilgraphen substituiert werden können [11].

4.4. Microsoft Dynamics AX

Microsoft Dynamics AX ist ein ERP System, das speziell für den Einsatz in mittelständischen und größeren Unternehmen ausgelegt ist. Dynamics AX ist eng mit den restlichen Microsoft Produkten integriert, bietet jedoch genug flexible und offene Schnittstellen um mit anderen Produkten in einer Unternehmenslandschaft eingebunden werden zu können [42], [43]. Dieses Produkt wird für den Prototyp als ERP System eingesetzt, da der Autor der Diplomarbeit mit diesem Produkt langjährige Praxiserfahrung hat.

Das System ist in Module geteilt, die je nach Kundenanforderung lizenziert werden können. In Abbildung 4.3 ist ein Formular des CRM Modules zu sehen. Alle Objekte der ERP Applikation selbst (nicht das Basissystem) sind in der Entwicklungsumgebung mit Source Code verfügbar (unter der Voraussetzung, dass die entsprechende Lizenz erworben wurde). Im Rahmen des kundenspezifischen Customizing können Objekte (wie Tabellen, Formulare, Klassen, Reports) angepasst werden.

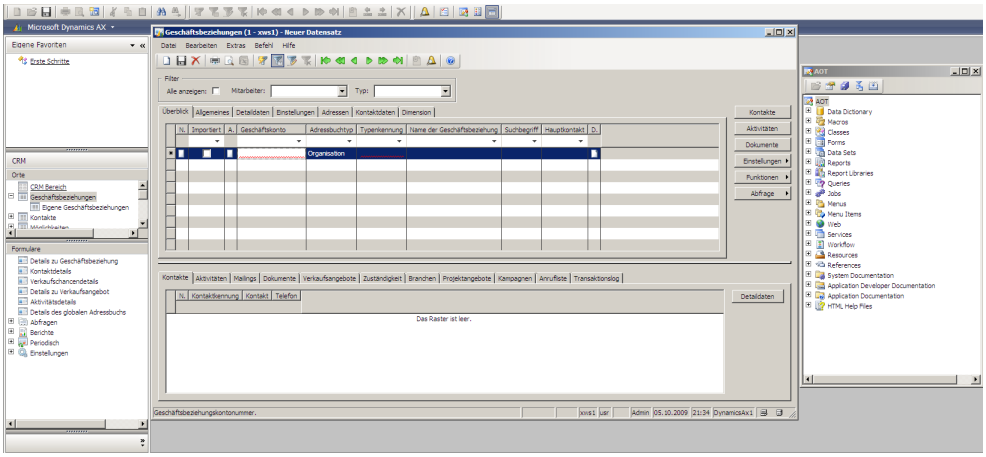

Abbildung 4.3.: Dynamics AX 2009

4.4.1. Technik

Die Entwicklungsumgebung ist fix in Dynamics AX integriert, eine Trennung zwischen Entwicklungs- und Ausführungsplattform ist nicht gegeben. Die notwendigen Funktionen können über das Berechtigungssystem für Benutzer freigeschaltet werden. Die

Programmierung erfolgt in der integrierten Sprache X++, die an C# und Java angelehnt ist. Es werden die wesentlichen objektorientierten Konzepte unterstützt. Der Zugriff auf die Datenbank erfolgt in der eingebetteten SQL Syntax. Die Struktur der Tabellen kann vollständig in Dynamics AX verwaltet und geändert werden. Aufgrund einer Layerstruktur bestehend aus 16 Layern können die Objekte je nach Anforderungen angepasst werden, dabei überdecken Objekte auf höheren Layern jene auf den unteren Ebenen. Mit dieser Funktionalität können z.B. länderspezifische Funktionen durch Importieren am entsprechenden Layer ergänzt werden.

Das System ist als 3-Schichten Architektur ausgelegt, bestehend aus Datenbank, Applikationsserver und Client. Als Ausführungsplattform für den Server ist ein Windows 2003 oder 2008 Server notwendig. Grund dafür ist unter anderem, dass die Authentifizierung über Active Directory erfolgt. Als Clientplattform werden alle Windows Betriebssysteme unterstützt, die als Domänenmitglied einer Domäne fungieren (aufgrund der Anmeldung über Active Directory notwendig). Als Ergänzung ist ein Zugriff per Web Portal möglich, das in der letzten Dynamics AX Version in den Sharepoint Server integriert wurde. Die Daten können in einer Microsoft SQL Server oder Oracle Datenbank abgelegt werden.

4.4.2. Schnittstellen

Im folgenden Abschnitt werden die Schnittstellen von Dynamics AX beschrieben. Für die Entwicklung des Prototyps, der auch Daten von Dynamcis AX einbeziehen soll, stellen diese vorhandenen Schnittstellen die grundlegenden Anbindungsmöglichkeiten dar. Dynamics AX bietet einerseits eine enge Integration für andere Microsoft Produkte, ermöglicht jedoch auch die Verwendung von herstellerunabhängigen Schnittstellen:

Einbindung von .NET. In Dynamics AX selbst können .NET DLLs eingebunden und aufgerufen werden. Die Verwendung der DLL Funktionen und Klassen erfolgt wie der Einsatz interner Dynamics AX Funktionen. Auch der umgekehrte Weg, der Einsatz von Dynamics AX Funktionen in .NET ist über den .NET Business Connector möglich. In .NET Projekten (z.B. C#) muss eine Referenz auf die DLL des Connectors ergänzt werden. Nach dem Aufruf der Login Funktion wird über den bereits am Betriebssystem angemeldeten Benutzer eine Session zu Dynamics AX aufgebaut. Bei bestehender Verbindung können über den .NET Business Connector direkt Daten in Tabellen gelesen oder modifiziert werden und alle verfügbaren Klassen instantiiert und aufgerufen werden.

Listing 4.4: .NET Business Connector

```
1  Microsoft.Dynamics.BusinessConnectorNet.Axapta dynAx =
2  new Microsoft.Dynamics.BusinessConnectorNet.Axapta();
3  Microsoft.Dynamics.BusinessConnectorNet.AxaptaRecord record;
4
5  // Authenticate and open session
6  dynAx.Logon(null, null, null, null);
7
8  // Define a empty ProjTable record
9  record = dynAx.CreateAxaptaRecord("ProjTable");
10
11 // Execute select statement,
12 // cursor is positioned on first record
13 DynRec.ExecuteStmt("select * from %1 where %1.CustAccount > 1000");
14
15 // Loop through result.
16 while(record.Found)
17 {
18     // read Name and ProjId field from ProjTable cursor
19     String fieldName   = acRecord.get_Field("Name");
20     String fieldProjId = acRecord.get_Field("ProjId");
21
22     Console.WriteLine(String.Format("{0} {1}",
```

```
23                          fieldName, fieldProjId));
24  }
25
26  int sID = 10;
27  Object o;
28  o = ax.CallStaticClassMethod("SysLabel",
29                               "labelId2String", sID);
30
31  dynAx.Logoff();
```

Im Listing 4.4 wird gezeigt, wie eine Verbindung aufgebaut werden kann und Datensätze aus der `ProjTable` gelesen und verarbeitet werden. Im Anschluss wird die statische Methode `labelId2String` der Klasse `SysLabel` mit dem Parameter `sID` aufgerufen.

Web Service. Die Einbindung von Web Services innerhalb Dynamics AX ist weitgehendst toolunterstützt möglich. So können durch Angabe der URL des Web Services, die clientseitigen Proxy Klassen automatisch erzeugt werden. Voraussetzung ist, dass es sich um ein SOAP basierendes Web Services handelt. Web Services mit anderen Kommunikationsprotokollen, können ebenfalls eingesetzt werden, vorausgesetzt die entsprechenden Funktionen zur Kommunikation zwischen Service Provicer und Service Requestor werden in Dynamics AX erstellt bzw. importiert.

Sharepoint. Als Alternative zur Verwendung des proprietären Dynamics AX Clients kann in Verbindung mit den Share Point Services ein Zugriff per Browser erfolgen. Bei dieser Variante sind nur ausgewählte Funktionen von Dynamics AX verfügbar.

XML Verarbeitung. Alle notwendigen Funktionen zur Bearbeitung und Verarbeitung von XML Daten sind in Dynamics AX vorhanden. Die existierenden Funktionen werden in Form von Wrapper Klassen für die Funktionen Microsoft XML Core Services (MSXML) bereitgestellt. Eine detaillierte Beschreibung zu den Funktionen von MSXML ist unter [44] zu finden.

Direkte DB Anbindung. Neben den beschriebenen Möglichkeiten zur Anbindung über Schnittstellen der Applikation, kann auch der Import oder Export von Daten über einen direkten Zugriff auf eine relationale Datenbank erfolgen. Voraussetzung ist die Verfügbarkeit eines ODBC Treibers für die Datenbank.

4.5. Mashup Editoren und Frameworks

Das Ziel eines Mashup Editors und eines Mashup Frameworks ist die Erstellung von Mashups durch nicht IT-Mitarbeiter. Die Anforderungen hinsichtlich der Bedienung und des technischen Hintergrundwissens müssen dementsprechend gehalten werden.

Ein Mashup Framework ist ein Softwareprodukt, das die Erstellung und Ausführung von Mashups ermöglicht [35]. Es stellt dem Anwender eine Umgebung bereit in der ohne technisches Hintergrundwissen durch Kombination beliebieger Datenquellen ad hoc eine neue Anwendung bzw. Auswertung erstellt werden kann. Die am Markt befindlichen Frameworks verwenden in der Regel Standard-Web-Technologien. Die Ausführung der erstellten Mashups erfolgt üblicherweise in einem Browser.

4.5.1. Arten von Editoren und Frameworks

Für die Analyse von Editoren und kompletten Frameworks werden Literatur und Unterlagen aus dem kommerziellen IT Bereich herangezogen bzw. werden die Informationen aus Tests der Produkte selbst ermittelt.

Arten von Mashup Editoren Grundsätzlich können drei Arten von Editoren unterschieden werden.

- Browserplugin: Installation eines spezifischen Plugins für das Mashup im Browser.
- Web Service / Web Applikation: Der Editor wird im Browser in Form einer Web Applikation eingesetzt. Das Service bzw. die Applikation kann im Internet oder unternehmensintern zur Verfügung gestellt werden. Diese Form ist speziell im Bereich der Privatanwender am häufigsten verbreitet.
- Applikation: Der Editor bzw. das Framework wird als eigene Applikation am Rechner installiert. Die Bedienung erfolgt über ein eigenes Benutzerinterface.

4.5.2. Mashup Frameworks

Im folgenden Kapitel werden die aktuell verfügbaren Mashup Frameworks und die dazugehörigen Editoren beschrieben. Die Produkte stammen aus dem kommerziellen Bereich als auch aus dem Open Source Bereich. Die Produkte werden dabei auf Anforderungen untersucht, die für die Umsetzung der Referenzimplementierung wichtig sind. Als Kriterien werden folgende Punkte herangezogen [66]:

- Art und Bedienung des Editors: Bedienung des Frameworks über einen Browser oder über eine lokal installierte Clientapplikation. Benötigt das Framework oder der Client ein spezielles Betriebssystem oder eine spezielle Version.
- Funktionen, Datenquellen: Bietet das Tool alle Funktionen, die für ein Enterprise Mashup notwendig sind. Ist die direkte Einbindung von Legacy Applikationen möglich.
- User (notwendiges Vorwissen, Lernkurve): Aus Sicht der IT Mitarbeiter als auch aus Sicht des Endanwenders. Damit in Verbindung die notwendige Einarbeitungszeit und die Zeit für die Erstellung eines Mashups.
- Support (Beispiele, E-Mail / Forum / Wiki, Telefon): Ist für das Produkt ein Support verfügbar.
- Unternehmen (Komplexität, Kosten, Sicherheit): Ein wichtiges Kriterium für den unternehmerischen Einsatz sind die Anschaffungskosten des Produktes sowie die Wartungskosten für den laufenden Betrieb.

Außerdem werden in der Beschreibung der Produkte auf nicht kategorisierbare Eigenschaften eingegangen.

IBM Mashup Center

Das Mashup Center von IBM vereint die Produkte IBM Lotus Mashup und IBM InfoSpere MashupHub zu einem integrierten Mashup Produkt [17]. Mit dem Produkt können unternehmensweit Informationsquellen zusammengefasst werden. Als Datenquellen können Datenbanken, Anwendungen und Internet-Quellen verwendet werden. Die Daten werden in Feeds konvertiert und dann miteinander kombiniert, siehe Abbildung 4.4. Die Daten der Feeds können für eine Optimierung der Leistung in einem Cache gespeichert werden. Als Anwendungs-Datenquellen können verbreitete Systeme wie Excel, SAP dienen sowie Applikationen, die über eigens erstellte Verbindungen durch das integrierte Plug-in-Framework eingebunden werden. Das Produkt erlaubt die Vergabe von Rollen und Zugriffsrechte auf Datenquellen. *Legacy Systeme* können beim IBM Mashup Center direkt per Zugriff auf deren SQL Datenbank eingebunden werden. In der Datenbank muss lediglich ein Account für den Zugriff des Mashup Frameworks angelegt werden, der Leserechte hat. Sollen nicht alle Tabellen für Mashups verfügbar sein, kann auf der Datenbank eine Einschränkung über die verfügbaren Tabellen vorgenommen werden. Als Alternative für die Bereitstellung von Legacy Systemen in Mashups kann die Bereitstellung über Web Services genutzt werden. Details zu den Möglichkeiten siehe 5.7.

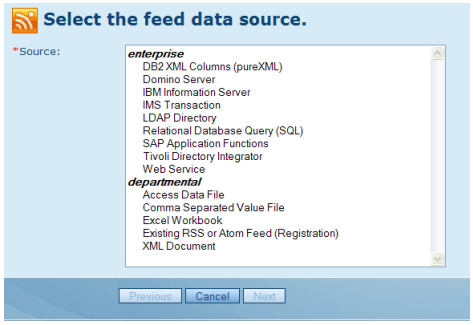

Abbildung 4.4.: IBM Mashup Center Datenquellen [17].

Neben dem Mashup Center bietet das Produkt IBM WebSphere sMash eine Umgebung zur Entwicklung dynamischer Scripts. Das Werkzeug erlaubt eine grafisch unterstützte Erstellung von Scripts in PHP oder Groovy bzw. REST-ähnlicher Elemente. Die einzelnen Artefakte können mit dem WebSphere sMash zu Widgets zusammengefügt werden. Die Widgets wiederum können mit dem Mashup Center zu neuen Webanwendungen verbunden werden.
Bei der Erstellung von Widgets sind keine Programmierkenntnisse erforderlich. Über die Widget Factory können Widgets mit Assistentenunterstützung erzeugt werden, siehe Abbildung 4.5. Insgesamt ist das notwendige Vorwissen beim Mashup Center im Vergleich zu den anderen Produkten aufgrund der umfangreichen Funktionalität relativ hoch. Beim Produkt sind allerdings diverse Beispiele enthalten, die als Vorlage verwendet werden können.

Die erstellten Feeds können durch transformieren, gruppieren, sortieren und filtern zu neuen Feeds zusammengefasst und als neue Informationsquelle dargestellt werden. Als Verarbeitungsmöglichkeiten stehen neben vordefinierten Funktionen und Operationen auch reguläre Ausdrücke zur Verfügung. Die erstellten Feeds können in Form von XML, ATOM oder RSS bereitgestellt werden. Die erzeugten Feeds können über einen Katalog veröffentlicht und anderen Anwendern zur Verfügung gestellt werden.

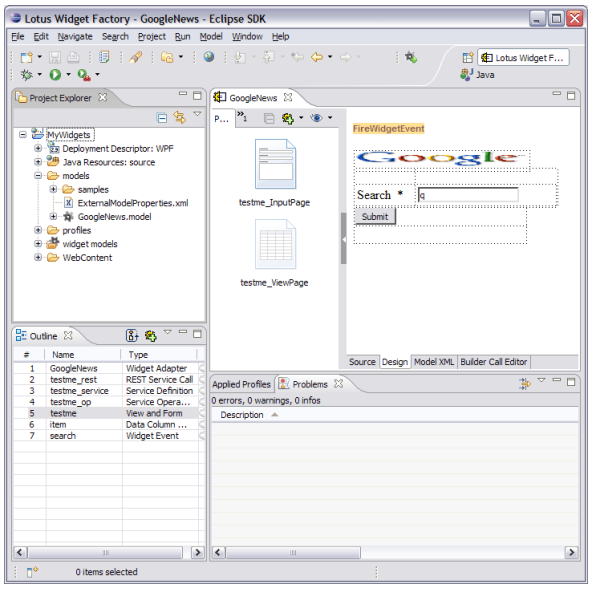

Abbildung 4.5.: IBM Mashup Center Widgets [17].

Systemanforderungen Das IBM Mashup Center hat folgende Anforderungen:

- Betriebssystem: Windows Server 2003 / 2008, AIX, Red Hat, SUSE
- Browser: IE 6,7 und 8, Firefox 3.6, Safari 5.0
- Datenbanken: DB2, Informix, Microsoft SQL Server, Oracle

Für das Produkt wird vom Hersteller Support für registrierte Benutzer angeboten. Außerdem gibt es Tutorials und Dokumente auf der IBM Homepage. Die Installation und der Betrieb aus Sicht der IT Abteilung wird mit mittel bewertet. Da es sich um ein Produkt handelt, das lokal im Unternehmen betrieben wird, muss eine Infrastruktur (Server, Betriebssystem) bereit gestellt und gewartet werden. Das Mashup Center besitzt eine integrierte Benutzerverwaltung mit der das System selbst sowie Datenquellen mit Rechten versehen werden können. Das Produkt ist ab 2.500 EUR erhältlich (Stand 11.2011, genauer Preis abhängig von den gewählten Modulen).

Intel Mash Maker

Der Intel Mash Maker ist eine Browsererweiterung ab Firefox 2.0. Die Grundidee dabei ist, dass die aktuelle Seite mit zusätzlichen Informationen angereichert werden kann [20]. Der Browser wird dabei mit Menüs und Funktionen erweitert. Die Zusatzfunktionen sind erst nach einer Registrierung und einem Login auf der entsprechenden Intel Webseite verfügbar. Die ergänzenden Funktionen ermöglichen die Erstellung und Verteilung von Mashups. Die Mashups werden dabei für eine bestimmte Webseite erstellt. Die generierten Mashups können in weiterer Folge verteilt und anderen Benutzern zur Verfügung gestellt werden. Des weiteren können bereits für die aktuelle Webseite erstellte und von Anwendern freigegebene Mashups eingebunden werden. Die Mash Maker Funktionen schlagen in Abhängigkeit des aktuellen Seitenkontextes verfügbare Mashups vor, die vom Anwender in die Seite integriert werden können.
Als Datenquellen sind nur Webseiten möglich. Eine Einbindung von unternehmensinternen Datenbanken ist direkt nicht möglich. Es können weder RSS Feeds noch Web Services eingebunden werden. Damit einhergehend ist auch die Anbindung von *Legacy*

Systemen nicht möglich.
Außerdem ergänzt das Mash Maker Plugin den Browser um eine semantische Interpretation von Webseiten. Im Idealfall müssen dafür alle Webseiten mit den notwendigen semantischen Informationen versehen sein, um die maschinelle Interpretation der Daten durchführen zu können. Im Mash Maker ist ein Extraction Tool enthalten, das die automatische Extraktion der semantischen Daten aus der HTML Seite durchführt. Diese Technik wird als Scrapping bzw. Web Scrapping bezeichnet [29]. Die Extraction Engine greift auf ein Data Repository zu, das von Benutzern aus der Community gefüllt und editiert wird. Über das Data Repository erhält die Engine die essenziellen Informationen für das extrahieren der Daten aus der Web Seite. Der Einsatz der Scrapping Funktion erfordert, dass der Benutzer mit Techniken wie XML und XPath (siehe Kapitel 2.4.1 und 2.4.1) vertraut ist. Grundsätzlich ist das Produkt sehr einfach und intuitiv bedienbar, es ist nur eine relativ kurze Einarbeitungszeit notwendig. Support vom Hersteller wird nicht angeboten.

Das Erzeugen von Mashups erfolgt mit einem einfach zu bedienenden Copy and Paste Interface. Um den Inhalt von zwei Webseiten zu kombinieren wird die erste Seite aufgerufen und über eine Funktion aus der Mash Maker Erweiterung durch Aufrufen der zweiten Seite kombiniert. Die korrekte Kombination der Daten wird über einen Assistenten konfiguriert.

Die Architektur des Tools besteht aus Mash Maker Database mit der Clients (Browser mit Mash Maker Plugin) kommunizieren, siehe Abbildung 4.6. Über die Datenbank werden erforderliche Metadaten (vorhandene Mashups, Userdaten, usw.) ausgelesen. Der Client sammelt und kombiniert die Daten von unterschiedlichen Web Seiten zu einem Mashup. Die Darstellung erfolgt in Widgets auf der aktuellen Seite.

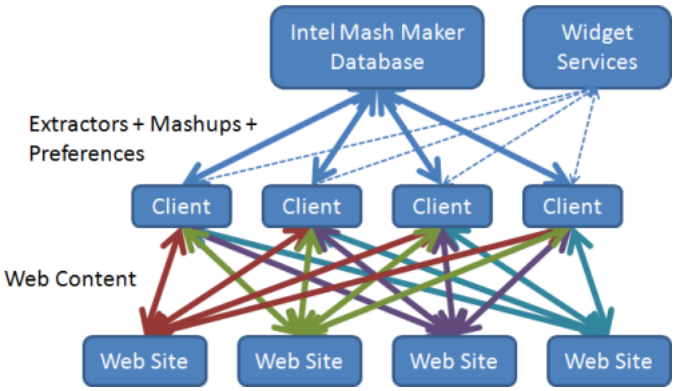

Abbildung 4.6.: Intel Mash Maker Architektur [20].

Die ursprüngliche Ausgangsbasis des Mash Makers war eine interaktive Programmiersprache, die in weiterer Folge zu dem aktuelle verfügbaren Tool weiterentwickelt wurde. Die *Mash Maker language* ist eine turing vollständige Sprache in der alle erstellten Mashups hinterlegt werden.

Das User Interface des Mash Makers (siehe Abbildung 4.7) sind der Browser selbst und die ergänzenden Erweiterungen. Damit die Bedienung des Tools einfach bleibt werden möglichst wenig Eingriffe in der Standard UI des Browsers durchgeführt. Die Anzeige zusätzlicher Informationen erfolgt über das Einblenden von z.B. Widgets, Daten-Panels und Filterdialogen. Bereits vorhandene Mashups (auch von anderen Benutzern) werden in der Liste der *Suggested Mashups* oder über die Suchfunktion ausgewählt.

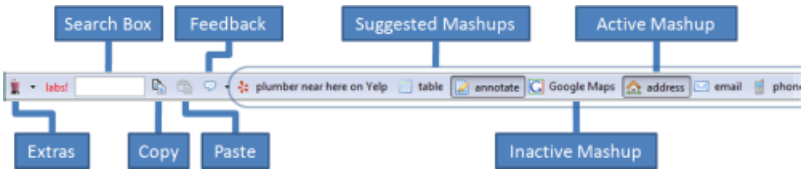

Abbildung 4.7.: Intel Mash Maker User Interface [20].

Die technische Komplexität des Produkts ist relativ niedrig, einzig die Installation des Browserplugins ist notwendig. Es ist keine Infrastruktur im Unternehmen notwendig. Es ist keine Benutzerverwaltung oder Vergabe von Rechten möglich, womit die geforderten Sicherheitspunkte nicht erfüllt sind. Für den Einsatz dieses Produktes fallen keine Kosten an. Die Software kann kostenlos von der Homepage des Herstellers heruntergeladen werden.

Open MashUp Studio

Ein frei erhältliches Mashup Tool ist das Open MashUp Studio. Das Produkt wird als Firefox Plugin (ab Firefox 3.0) eingebunden und kann über ein erweitertes Browser Menü bedient werden. Die Anwendung ist in die Bereiche Project, Interface und Behaivor getrennt, siehe Abbildung 4.8.

Im Projektbereich werden Mashup Projekte angelegt und verwaltet. Das Userinterface im Interfacebereich kann per Drag and Drop mit Hilfe vorhandener Display-Elemente erstellt werden. Die Logik des Mashups wird im Behaviorbereich erstellt. Aus einer vorgefertigten Auswahl können Logikkomponenten in das Mashup eingefügt werden. Für weitergehende Programmlogik ist ein tieferes Verständnis der zugrundeliegende Programmiersprache notwendig.
Es können RSS Feeds und Web Services als Datenquellen eingebunden werden. Die Verwendung von SQL Datenbanken ist direkt nicht möglich, sondern kann nur indirekt über Web Services erfolgen. Das Einfügen der Datenquellen erfolgt mit Drag and Drop Unterstützung, allerdings muss die weitere Konfiguration der Datenquellen manuell vorgenommen werden (deswegen nur mittlere Einstufung bei diesen beiden Datenquellen in der Übersicht unter Abbildung 4.13). Eine Einbindung von *Legacy Systemen* kann nur durch Bereitstellung der Daten über Web Services erfolgen.
Für das Erstellen einfacher Mashups ist praktisch kein Vorwissen notwendig, soll allerdings Logik ergänzt werden, die nicht per Drag and Drop eingefügt werden kann, ist technisches Hintergrundwissen notwendig.

Aufgrund der sehr eingeschränkten Funktionalität, der fehlenden ausführlichen Dokumentation, des fehlenden Supports und der hohen Anforderungen an die Programmierkenntnisse ist Open MashUp Studio nicht für den Einsatz beim Endanwender im Enterprise Umfeld geeignet.

Yahoo Pipes

Yahoo Pipes ist eine freie Web Applikation zur Erzeugung von Mashups [19]. Die Applikation besteht aus einem grafischen User Interface, das online im Browser ausgeführt wird. Die Mashups verwenden Feeds, Webseiten und Web Services als Datenquellen. Die Grundfunktion ist das Zusammenfügen von Informationen von unterschiedlichen Quellen und das Definieren von Regeln um den gelesenen Inhalt zu modifizieren (z.B. filtern, sortieren). Der Name *Pipe* kommt von der *Unix Pipe*, die es ermöglicht Ausgaben

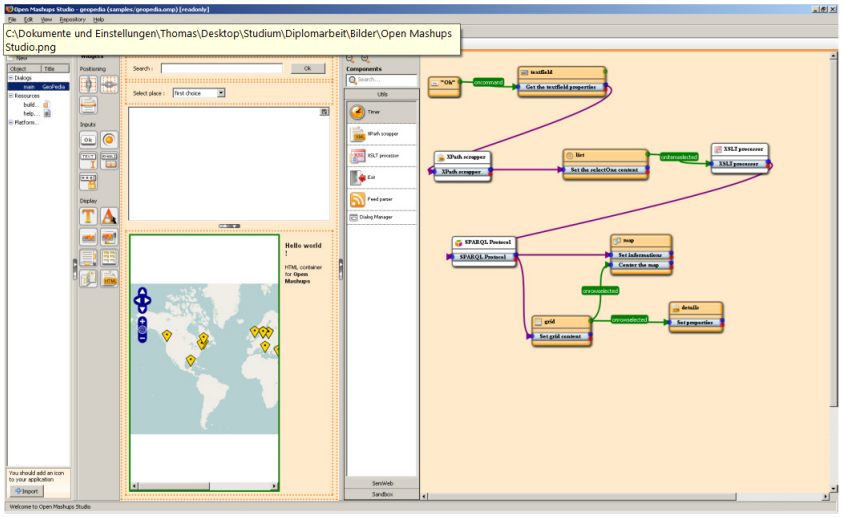

Abbildung 4.8.: Open MashUp Studio

von diversen Tools aneinanderzureihen. Das Ergebnis einer Yahoo Pipes Anwendung ist eine Web Seite, die Feeds von unterschiedlichen Quellen aggregiert und darstellt.

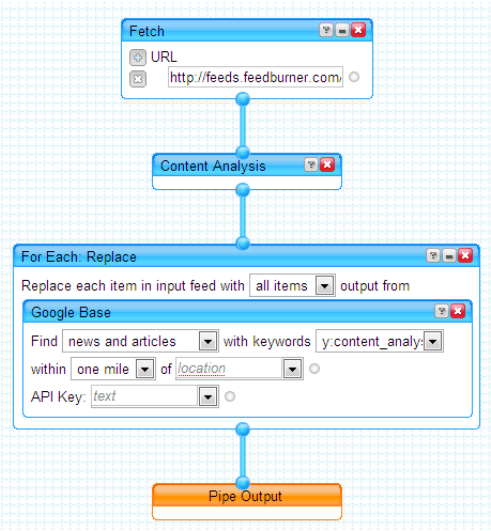

Abbildung 4.9.: Yahoo Pipes

Die Erzeugung einer Pipe erfolgt mit Drag and Drop Unterstützung durch hinzufügen und verbinden von Modulen aus der Bibliothek, siehe Abbildung 4.9. Jede Pipe besteht aus zwei oder mehr Modulen, wobei jedes Modul eine spezifizierte Aufgabe erledigt (Z.B. gibt es ein *Fetch* Modul, das eine URL ausliest. Oder ein *Sort* Modul, das gelesene Daten sortiert). Die Inputs und Outputs der Module werden zu einer Kette verbunden. Neben Dateninputs erlaubt Yahoo Pipes die Eingabe von Userinputs. Diese werden zur Laufzeit der Pipe als Eingabefeld im Browser dargestellt. Erstellte Pipes können am Yahoo Server gespeichert und veröffentlicht werden bzw. können von anderen Benutzern bereitgestellte Pipes eingebunden werden.

Als Datenquellen können SQL Datenbanken, RSS Feeds, Web Services, JSON und XML Daten verwendet werden. Bei der Einbindung von Datenbanken und Web Services

ist allerdings ein manueller Eingriff notwendig. Die Einbindung von *Legacy Systemen* kann mit allen unter 5.7 beschriebenen Techniken erfolgen. Ansonsten ist praktisch kein Vorwissen notwendig und die Einarbeitung in das Produkt ist relativ einfach. Support wird nur in Form eines Forums und Wikis angeboten. Da es sich bei diesem Produkt um eine Web Anwendung handelt, ist praktisch keine zusätzliche Infrastruktur im Unternehmen notwendig. Die geforderten Sicherheitspunkte für Enterprise Mashups sind damit allerdings nicht erfüllt.

Mozilla Ubiquity

Ein Plugin für den Browser Firefox zur Erstellung von Mashups ist Mozilla Ubiquity [34]. Das Tool bietet innerhalb von Firefox eine Commandline-Umgebung in der mit den verfügbaren Befehlen Mashups erzeugt werden können. Dazu stehen eine Reihe von Befehlen zur Verfügung, die es ermöglichen Informationen oder Karten von URLs zu laden, Daten zu filtern oder sortieren oder E-Mails mit z.B. übersetzten Inhalten zu versenden, siehe Abbildung 4.10.

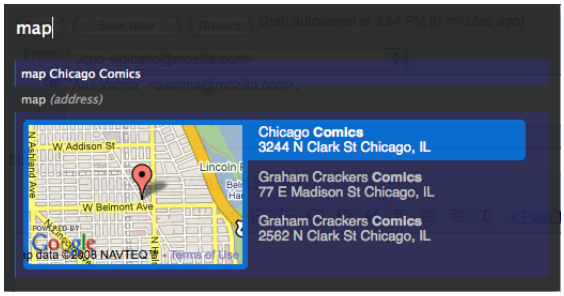

Abbildung 4.10.: Mozilla Ubiquity

Als mögliche Datenquellen für Mozilla Ubiquity sind nur Informationen aus dem Web möglich. Eine Anbindung von Datenbanken oder *Legacy Systemen* ist nicht direkt möglich. Diese Art von Datenquellen sind nur über Web Services einzubinden.

Um die Software bedienen zu können ist einiges an Vorwissen notwendig. Support vom Hersteller gibt es in Form eines Wikis. Die Inbetriebnahme im Unternehmen setzt nur die Installation des kostenlosen Browser Plugins voraus.
Die Entwicklung von Mozilla Ubiquity wurde 2008 und 2009 aktiv durchgeführt. Aktuell wurde die Entwicklung auf unbestimmte Zeit stillgelegt. Die vorhandenen Funktionen und Möglichkeiten sind grundsätzlich für die Erstellung eines Enterprise Mashups ausreichend. Spezielle Anforderungen im Unternehmensbereich wie festlegen von Zugriffsrechten oder die direkte Einbindung von Datenbanken oder Legacy Applikationen schränken die Einsetzbarkeit deutlich ein.

JackBe Presto

Mit Presto bietet JackBe eine Enterprise Mashup Plattform an [27]. Als echtes Enterprise Produkt ist eine umfangreiche Auswahl an möglichen Techniken für Datenquellen verfügbar: RSS Feeds, Web Services, CSV und XML Dateien, SQL Datenbanken, Sharepoint, Excel. Datenquellen ohne direkten Zugriff werden via Web Service angebunden. Das Framework stellt fertig verwendbare Connectoren für eine Reihe von Datenquellen bereit. Für nicht unterstützte Datenquellen können eigene Connectoren entwickelt werden. Die Einbindung von *Legacy Systemen* kann wie unter 5.7 beschrieben erfolgen

(d.h. über direkten Zugriff auf die SQL Datenbank oder durch Bereitstellung über Web Services). Als Ergänzung kann mit JackBe Presto ein eigener Connector für ein Legacy Sytem entwickelt werden.Des weiteren können erstellte Mashups in einem App Store für andere Entwickler zum Kauf angeboten werden.

Abbildung 4.11.: Jackbe Presto

Abbildung 4.12.: Jackbe Presto Studio [27]

Ein weiterer wichtiger Punkt für ein Enterprise Produkt ist die Sicherheit. Die erzeugten Mashups können mit einer Zugriffskontrolle belegt werden.
Die Erstellung eines Mashups wird bei diesem Produkt grundsätzlich vom Endanwender durchgeführt. Für die Lösung komplexerer Aufgabenstellungen gibt es eine eigene Entwicklungsumgebung, die für die Verwendung durch Entwickler gedacht ist. Der Zugriff auf die Umgebung zur Erstellung von Mashups erfolgt über den Browser, siehe Abbildung 4.11.

Presto stellt eine eigene Plattform für das Erzeugen und Ausführen von Mashups dar. Das Produkt besteht aus folgenden Komponenten:

- Einem Server mit einer Ausführungsumgebung für die erstellten Mashups.
- Einen grafischen Editor und eine IDE zur Erstellung für Endanwender und Entwickler.
- Ein eigenes SDK zur Anbindung von Clients.
- Eine DSL (EMML, Enterprise Mashup Markup Language [4]).

Mit der Presto Umgebung können drei Arten von Artefakten erstellt werden:

- **Services**: Sind Applikationen, Dokumente oder andere Arten von Informationsquellen, die firmenintern oder extern vorhanden sein können. Der Zugriff auf ein Service kann mit einem beliebigen Protokoll erfolgen. Jedes Service gibt in einem bestimmten Format eine Menge an Daten zurück.
- **Mashups**: In Presto wird unter Mashups eine Kombination oder Transformation von Inhalten je nach Anforderung verstanden. Auch ein Wrapping von Applikationen, um Daten in Presto verfügbar zu machen, wird als Mashup verstanden.
- **Mashlets**: Für den einfachen Zugriff auf Mashups oder Services wird ein Mashlet erstellt. Ein Mashlet entspricht einem Widget, das mit Logik angereichter werden kann, und Ergebnisse von Mashups oder Services darstellt.

Für die Erstellung ist der *Presto Composer* vorhanden. Damit können Services, Mashups und Mashlets erstellt und veröffentlicht werden. Die Entwicklungsumgebung (als Eclipse Plugin) ist *Mashup Studio*, siehe Abbildung 4.12. Das Tool ist eine vollständige Entwicklungsumgebung mit EMML Unterstützung. Die erstellten Mashups können in weiterer Folge als Mashlet dargestellt werden.

Die Ausführung eines Mashups erfolgt am Mashup Server. Aller erzeugten Artefakte werden am Server veröffentlicht und im Repository abgelegt. Clients können angebotene Services und Mashups aufrufen. Beim Aufruf sorgt eine regelbasierte Zugriffskontrolle für Sicherheit.
Für das Produkt ist ein entsprechendes Hintergrundwissen und eine entsprechende Einarbeitungszeit notwendig. Bei der Installation werden Beispiele von fertigen Mashups mitgeliefert. Support wird vom Hersteller über E-Mail, Forum, Wiki und telefonisch angeboten.
Aufgrund der notwendigen unternehmensinternen Infrastruktur erhält das Produkt im Bereich Komplexität die Einstufung hoch. Auch die Kosten mit etwa 40.000 USD pro Jahr pro CPU sind relativ hoch (Preise werden nur auf direkte Anfrage bekannt gegeben).

4.5.3. Vergleich

Der Vergleich in Form eine tabellarischen Gegenüberstellung aller Produkte soll die wesentlichen Unterschiede hervorheben. Es werden hierfür die am Beginn des Kapitels beschriebenen Kriterien herangezogen.

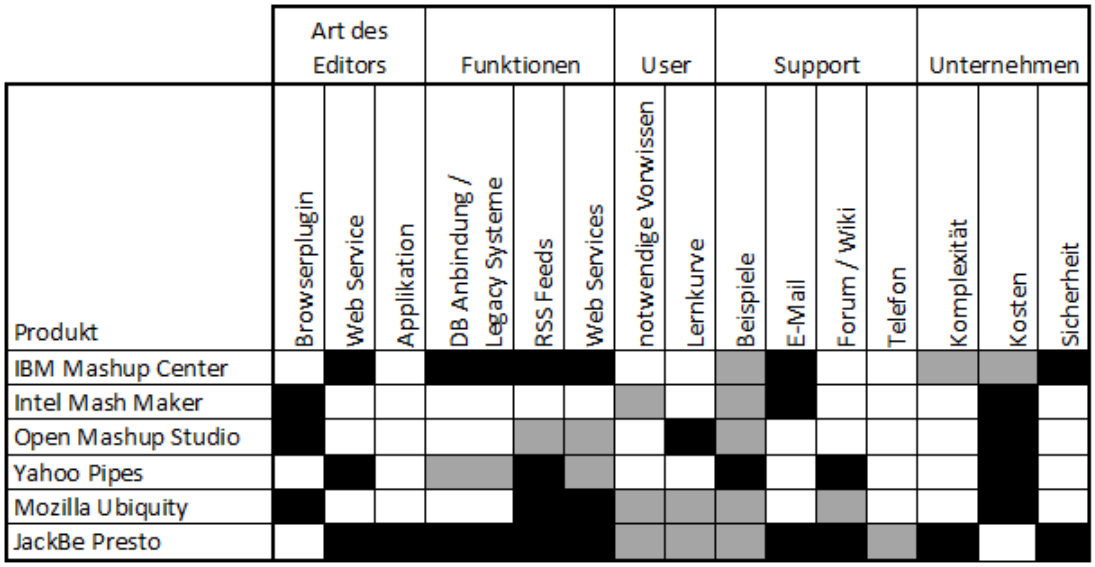

Abbildung 4.13.: Produktvergleich

- weiß: Wird nicht unterstützt oder ist nicht verfügbar. Für *Vorwissen* und *Lernkurve*: viel notwendig bzw. schlecht.
- grau: Wird teilweise unterstützt bzw. liegt im mittleren Bereich.
- schwarz: Wird vollständig unterstützt bzw. liegt im oberen Bereich. Für *Vorwissen* und *Lernkurve*: wenig notwendig bzw. gut.

5. Zielsetzung für den Prototyp

Im Rahmen der Arbeit wird ein praxisnahes Enterprise Mashup entwickelt. Die Implementierung des Prototyps wird anhand des *SOA Aproach* als Vorgehensmodells umgesetzt [21]. Der Prototyp zeigt eine mögliche Aufgabenstellung innerhalb eines Unternehmens, die als Enterprise Mashup umgesetzt werden kann. Der wichtigste Aspekt neben der technischen Realisierung ist der Einsatz eines Vorgehensmodells. Damit soll für ein Unternehmen eine Refernzvorgehensweise zur Umsetzung von Anforderungen mittels Mashups geschaffen werden.

5.1. Allgemeine Beschreibung

Die Grundidee besteht darin, ein Enterprise Mashup zu entwickeln, das dem Endanwender kombinierte Informationen aus unterschiedlichen Datenquellen bereitstellt. Als primäre Datenquelle dient das ERP System aus dem Daten über zu liefernde Artikel geladen werden. Als konkretes ERP System wird für den Prototyp Microsoft Dynamics AX in der Version 3 mit einem Branchenmodul für Anlagenbauer eingesetzt. Die Planung von Aufträgen erfolgt hier in einer hierarchischen Projekt- bzw. Artikeldarstellung. Externe Datenquelle ist ein Service zur Konvertierung der Gewichtseinheiten der Auftragspositionen. Das Mashup stellt die Kombination dieser beiden Daten als neue Anwendung dar. Es wandelt für jede Auftragsposition das Gewicht in der hinterlegten Einheit in eine beim Mashupaufruf vom Benutzer eingegebe Einheit um. Das Ergebnis sind alle Artikeldatensätze des gewählten Projektes mit den konvertierten Gewichtswerten.
Im Prototyp wird dafür ein frei verfügbares Web Service aus dem Internet eingesetzt. Für die Analyse und die Umsetzung wird dieses konkrete Web Service herangezogen. Prinzipiell kann jedes beliebige Web Service im Mashup verwendet werden. Im Kapitel 7 werden anhand eines Leitfadens die einzelnen Schritte gezeigt, um ein Mashup mit einem anderen Web Service zu entwickeln. Der Leitfaden stellt eine Schritt für Schritt Vorgehensweise zum Erstellen eines unternehmerisch genutzten Mashups dar.

ERP System Der zentrale Ausgangspunkt für die Daten aus dem ERP System sind die Kundenaufträge. Jeder Auftrag eines Kunden wird in einer Baumstruktur in Form eines Projektes verwaltet. Die Subknoten eines Projektes entsprechen den einzelnen Abläufen innerhalb eines Verkaufsprozesses. In Abbildung 5.1 ist ein einfaches Projekt dargestellt. Es enthält den *Hauptknoten A00114*, den *Unterknoten .CN - Credit Note - Gutschrift*, den *Unterknoten .OR - Order - Auftrag*, den *Unterknoten .OS - Small Order - Kleinauftrag* und den *Unterknoten .PO - Purchase Order - Vorabdispo*. Die Projektstruktur wird aufgrund von im ERP System definierten Vorlagen zum jeweiligen Kunden nach Bedarf aufgebaut. Dabei können aus einer Reihe von Vorlagesubknoten die gewünschten Knoten zum Kundenprojekt ergänzt werden.

In den jeweiligen Subknoten für die Kundenabläufe befinden sich die Artikelstrukturen, siehe Abbildung 5.2. Die Artikelstruktur ist wie die Projektstruktur in Form

Abbildung 5.1.: Projektaufbau ERP

eines Baumes aufgebaut. Der Baum enthält ausgehend vom Projektsubknoten die einzelnen Artikel, die zum jeweiligen Projekt gehören, und an den Kunden bzw. an die entsprechende Lieferadresse geliefert werden. Im Beispiel ist der Subprojektnoten *A00114.OS.001* in der Artikeldarstellung zu sehen. Der Auftrag besteht aus 2 Titel (*1 [] KommTeil mit Sekbed* und *2 [] KommTeil mit Sekbed*) und den darin enthaltenen Artikel (*1.1 [] Pult* und *2.1 [] Pult*), wobei die beiden Artikel eine Stückliste darstellen. Die gezeigte Stückliste in diesem Beispiel ist sehr einfach gehalten und enthält jeweils nur einen weiteren Subartikel.

Abbildung 5.2.: Aufbau Artikelstruktur ERP

Externe Daten Als externe Datenquelle aus dem Web wird ein Dienst zur Konvertierung von Gewichtsmaßen eingesetzt. Das Service erhält als Inputparameter drei Parameter: das Gewichtsmaß, die Ausgangseinheit und die Zieleinheit. Der Output ist das konvertierte Gewichtsmaß. Die technischen Details zum Web Service sind in der Component Specification unter 6.5 beschrieben.
Grundsätzlich kann jedes Web Service in einem Mashup eingebunden werden. Im Leitfaden unter 7 wird ein Dienst zur Konvertierung von Währungen verwendet.

Enterprise Mashup Die Kombination der ERP Daten und der externen Datenquelle erfolgt über das Mashup . Aus den zu einem Subprojektknoten gehörenden Artikeln und dem beim Artikel hinterlegten Gewicht bzw. Einheit sowie der vom Benutzer eingegebenen Umrechnungseinheit ergibt sich der Input für das Mashup. Mit Hilfe dessen wird das auf die gewünschte Einheit umgerechnete Gewicht ermittelt. Zur Umrechnung der Daten jedes Artikels muss das Web Service für jeden Artikeldatensatz aufgerufen werden. Die so erhaltenen Ergebnisse werden in Folge wieder in ein Gesamtergebnis (in Form eines XML Dokuments) integriert. Bei der Implementierung, siehe 6.7, muss dabei speziell auf eine saubere Trennung der Funktionen in Einzelkomponenten Rücksicht genommen. Diese Trennung in einzelne Blöcke ermöglicht einen wiederholten Aufruf der Konvertierungskomponente für jeden gelesenen Datensatz. Positiver Nebeneffekt der

Aufteilung ist die erhöhte Wiederverwendbarkeit der Komponenten (die Wiederverwendbarkeit war Teil der Anforderung).

5.2. Ontologien

Wie in den Kapiteln 4.2 und 4.3 beschrieben, werden mit Hilfe von Ontologien im Semantic Web die Daten maschinell verarbeitbar gemacht. Die semantische Markierung von Daten kann auch in Verbindung mit Enterprise Mashups eingesetzt werden. Speziell wenn die Daten aus ERP Systemen oder anderen Unternehmenssystemen kommen, welche üblicherweise über eine Datenbank mit einem umfangreichem Tabellenschema ausgestattet sind. Um diese Daten automatisiert verarbeiten zu können, ist eine semantische Markierung notwendig. Die semantische Anreicherung kann erfolgen, wenn die Daten über Web Services den Mashups zur Verfügung gestellt werden. Die Ergänzung der semantischen Komponente ist nicht möglich, wenn die Daten direkt aus der SQL Datenbank geladen werden (eine detaillierte Erörterung der Möglichkeiten zur Bereitstellung von Daten ist unter 5.7 zu finden).
Für den entwickelten Prototyp könnte dies bedeuten, dass die vom ERP System gelesenen Daten semantisch markiert werden. Im Prototyp ist geplant Projekt- und Artikeldaten im Mashup zu verarbeiten, die vom ERP System bereit gestellt werden. Damit diese Daten automatisiert mit anderen Daten innerhalb eines Mashups in Relation gesetzt werden können, könnten sie mit der entsprechenden semantischen Markierung versehen werden. Der Prototyp selbst ist ohne semantische Anreicherung umgesetzt. Die hier beschriebenen Ideen sind mögliche Erweiterungen, um den Prototyp auch mit semantischen Daten einsetzen zu können.

5.3. Architektur

Die für die Umsetzung gewählte Architektur ist in Abbildung 5.3 dargestellt. Kern des Prototyps ist das Mashup Framework. Das gewählte Framework und die Begründung für dessen Auswahl wird unter 5.4 beschrieben. Die Anbindung des Clients an das Framework erfolgt per http, d.h. der Endanwender erstellt und bedient das Mashup über einen Browser. Die Anbindung des Clients per https stellt aufgrund des firmeninternen Einsatz des Mashups keine unbedingte Anforderung dar.
Die externe Datenquelle (Web Service) wird per SOAP in das Framework eingebunden. Die unternehmensinterne Datenquelle (ERP System) wird mittels einer unter 5.7 beschriebenen Möglichkeit integriert.

Logging Teil der Architektur ist eine Logging-Komponente mit der die im Mashup ausgeführten Aktionen protokolliert werden. Die Motivation für das Logging entsteht zum einen aus der Sensibilität der im Mashup verarbeiteten Daten. Die von den Unternehmenssystemen kommenden Daten haben eine wirtschaftliche Bedeutung für das Unternehmen und dürfen nicht ohne weiteres an der Außenwelt gelangen. Daher ergibt sich die Notwendigkeit die Art des Zugriffes und wer darauf zugreift zu protokollieren. Durch die relativ offene Bereitstellung der Daten für den Einsatz in Mashups wäre ohne entsprechende Protokollierungsmaßnahmen nicht mehr nachvollziehbar wann wer welche Daten in Verwendung hat. Eine weitere Motivation für den Einsatz einer Loggingumgebung ensteht aus der Tatsache, dass nicht alle in Internet verfügbaren Web Services kostenlos sind. Diese Dienste werden meist nach einer Registrierung mit Benutzername und Passwort zur Verfügung gestellt und werden entweder pauschal oder

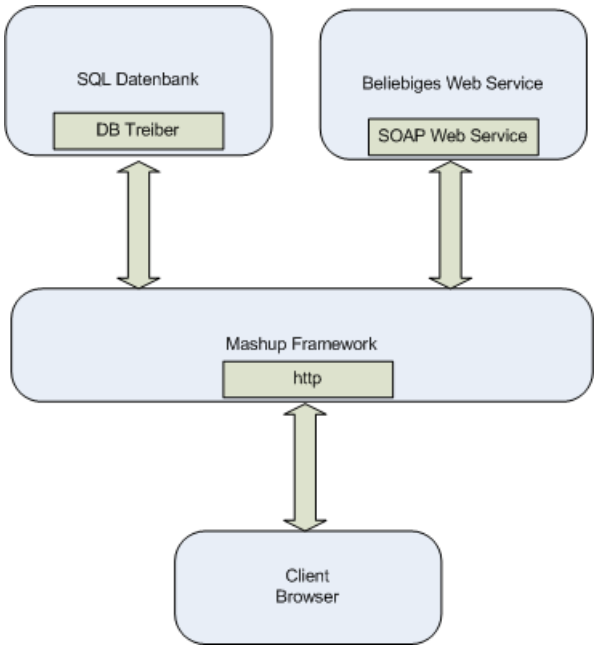

Abbildung 5.3.: Prototyp Architektur

nach Bedarf abgerechnet. Handelt es sich um ein Web Service, das nach Bedarf abgerechnet wird, müssen die für das Unternehmen anfallenden Kosten im Sinne der Kostenrechnung auf die verursachenden Abteilungen aufgeteilt werden können. Um das Accounting durchführen zu können, müssen die Aufrufe des kostenpflichtigen Dienstes über das Logging nachvollzogen werden können.
Als weiterer Punkt für das Logging spricht die Nachvollziehbarkeit von auftretenden Fehlern während der Ausführung eines Mashups. Um Probleme debuggen zu können müssen die Vorgänge reproduzierbar werden. Dazu ist es notwendig den Mashup Prozess zu protokollieren.

5.4. Auswahl eines Mashup Frameworks

Die Umsetzung des Prototyps basiert auf einem im Kapitel 4.5 beschriebenen Mashup Frameworks. Als Kriterien werden die Anforderungen unter 4.5.2 an Enterprise Mashups herangezogen. Aus den genannten Kriterien in Verbindung mit der Aufgabenstellung unter 5.1 wird im folgenden die Entscheidung für die Wahl des Frameworks getroffen und begründet.

Aufgrund der oben beschriebenen Ausgangssituation wird für die Implementierung des Prototyps das Produkt **JackBe Presto** verwendet. Als wichtige Entscheidungsgrundlage dienen die Punkte *Verfügbarkeit* und *Sicherheit*. Die Verfügbarkeit stellt im unternehmerischen Einsatz ein bedeutsames Kriterium dar. Diese Anforderung ist mit allen Produkten bei denen ein lokaler Server im Unternehmensnetzwerk für die Ausführung des Frameworks eingesetzt wird bestmöglich erfüllt. Bei Produkten, die rein als Browser Applikation allerdings ohne lokalen Server (der Server wird im Internet vom Anbieter des Frameworks bereit gestellt) ausgeführt werden, hängt die Verfügbarkeit maßgeblich von der Verfügbarkeit des Server beim Anbieter ab. Der zweite wichtige Punkt ist die Sicherheit, dabei speziell die Datensicherheit. Unternehmensdaten in Verbindung mit dem ERP System stellen eine der sensibelsten wenn nicht sogar

die sensibelsten Daten eines Unternehmens selbst dar. Sollen diese Daten in Form eines Enterprise Mashups dessen Framework bei einem externen Anbieter ausgeführt wird, bearbeitet werden, so sollten mit diesem Anbieter entsprechende Verträge unterzeichnet werden. Auch wenn gesamte ERP Systeme für ein Unternehmen out-house gehostet werden können, ist der Einsatz eines out-house Mashup Frameworks anders zu behandeln. Anbieter von out-house ERP Systemen verfügen einerseits über die technologischen Sicherungsmaßnahmen für die Systeme als auch über entsprechende Verträge, die mit den Kunden geschlossen werden. Diese Art von Anbieter sind auf den sensiblen Umgang ihrer Kundendaten angewiesen. Ganz anders ist dies mit Anbietern von Mashup Frameworks zu handhaben. Die Kunden von Standard Anbietern für Frameworks sind im Consumer Bereich angesiedelt und haben in der Regel keine oder wenig Erfahrung mit Business Kunden. Aufgrund dieser beiden Punkte sind alle Produkte, welche keinen lokalen Server für das Ausführen des Frameworks verwendet, nicht vernünftig einsetzbar.
Damit bleiben bei den möglichen Produkten das *IBM Mashup Center* und *JackBe Presto* über. Die Entscheidung pro JackBe Presto fällt aufgrund folgender Kriterien: JackBe Presto ist von der Komplexität zwar höher als das IBM Mashup Center, die Einarbeitung in das Produkt und die Lernkurve ist jedoch deutlich besser, das notwendige Vorwissen kann geringer sein. Des Weiteren bietet JackBe Presto einen Support in Form von E-Mail, Telefon, Forum und Wiki an. Die Anforderung an die Sicherheit ist bei beiden Produkten abgedeckt, die Kosten befinden sich im gleichen Bereich.

5.5. Vorgehensmodell für die Implementierung

Als Vorgehensmodell für die Referenzimplementierung des Prototyps wird der *Service-oriented Architecture Approach* eingesetzt [16]. Der SOA Approach ist ein von IBM entwickeltes und offen gelegtes Vorgehensmodell für die Umsetzung von SOA Projekten und Komponenten. Das Hauptaugenmerk beim SOA Approach liegt auf der Identifizierung der notwendigen Services zur Implementierung der vollständigen SOA. Der SOA Approach verwendet festgelegte Phasen, siehe Abbildung 5.4. Die Abfolge der Phasen ist nicht notwendigerweise linear und sequentiell, es können Phasen parallel abgearbeitet werden. In den folgenden Abschnitten wird der SOA Approach allgemein beschrieben. In weiterer Folge im Kapitel 6 wird der SOA Approach anhand des Prototyps angewandt und umgesetzt.
In der IBM Dokumentation zum SAO Approach gibt es ein **dokumentiertes Set an grafischen Modellierungsansätzen** [16]. Diese sind zum einen Teil UML-ähnliche Diagramme zum anderen Teil proprietäre Diagramme, die speziell für den SOA Approach entwickelt wurden. In der IBM Dokumentation werden die Diagramme anhand eines konkreten Beispiels demonstriert und beschrieben. Die Vorbereitung der Implementierung des Prototyps im Kapitel 6 erfolgt ebenfalls anhand dieser Modellierungsansätze.
Speziell für UML-ähnliche Diagramme wird darauf hingewiesen, dass diese in der gleichen Form wie im SOA Approach eingesetzt werden. Die Struktur der Diagramme enthält verwandte Notationen zur UML. Alle Diagramme haben gemeinsam, dass sie um proprietäre nicht standardisierte Elemente ergänzt wurden.

5.5.1. Domain Decomposition

Im ersten Schritt wird der gesamte Vorgang in einzelne Teilkomponenten zerlegt. Die Teilkomponenten stellen die technologischen Subsysteme dar. Nach dem Zerlegen wer-

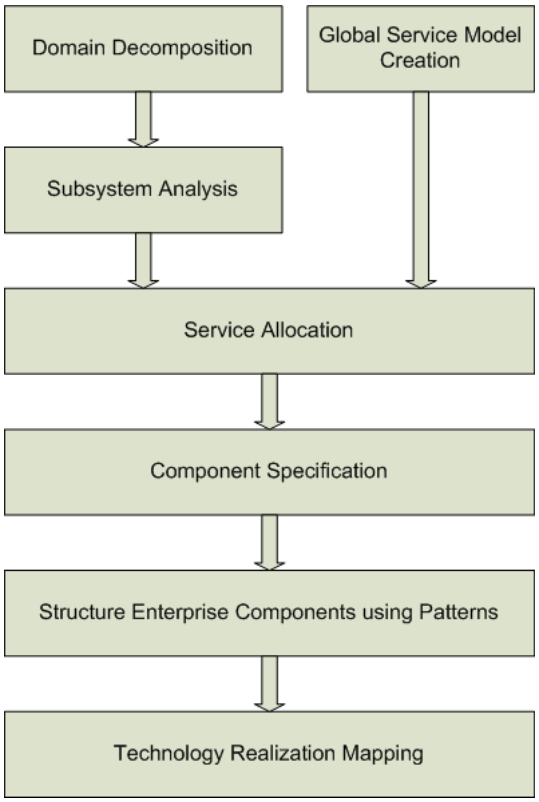

Abbildung 5.4.: SOA Approach

den die einzelnen Teilbereich in Prozesse, Subprozesse und Anwendungsfälle geteilt. Die gefundenen Anwendungsfälle stellen die Ausgangsbasis für Dienste dar, die als Web Service angeboten werden. Für die Implementierung der Anwendungsfälle und damit der Services ist es wichtig die Inputs und Outputs jedes Services zu definieren.

5.5.2. Goal-Service Model Creation

In diesem Schritt wird ein Service Modell entwickelt, um die Vollständigkeit der im vorigen Schritt gefundenen Kandidaten für Services zu validieren. Die Erstellung des Goal-Service Model erfolgt in einem top-down Prozess. Ausgehend von den Zielen, die im Rahmen der Anforderung definiert wurden, wird ein Baum aus Sub-Zielen aufgebaut. Jedes Ziel wird dabei in eine Reihe von untergeordneten Zielen zerlegt. Der Prozess wird ausgeführt bis die zerlegten Sub-Ziele von Diensten abgedeckt werden können. Mit Hilfe des erstellten Baumes ist die Zuordnung der Dienste zu den Anforderungen nachvollziehbar. Damit wird die Lücke zwischen den Anforderungen und den implementierten Diensten auf ein Minimum reduziert.

5.5.3. Subsystem Analysis

Die Subsystem Analyse ist im Bereich des Designs bzw. der Architektur angesiedelt. Es werden die geschäftlichen Anwendungsfälle in System-Anwendungsfälle verfeinert. Subsysteme bestehen aus geschäftlichen Komponenten (z.B. Kunde, Lieferant, Artikel) und technischen Komponenten (z.B. Sicherheit, Logging, Nachrichten). Während der Subsystem-Analyse werden die geschäftlichen und technischen Komponenten folgendermaßen identifiziert:

- Analysieren des Prozessablaufes innerhalb eines Subsystems (besteht häufig aus einer Abfolge von Anwendungsfällen), um die Kandidaten für die geschäftlichen Komponenten zu identifizieren.
- Die technischen Komponenten werden über alle nicht-funktionalen Anforderungen gefunden.
- Identifizieren der geforderten Funktionalität für die geschäftlichen Komponenten.

Durch die so ermittelten Teilkomponenten wird sichergestellt, dass jeder geschäftliche Anwendungsfall auf einem Set von technischen Komponenten beruht. Diese sind im Subsystem zu einer Einheit zusammengefasst.

5.5.4. Service Allocation

Nachdem in den vorigen Schritten die notwendigen Dienste festgelegt wurden, werden in diesem Schritt die Dienste den technischen Komponenten zugeordnet. Außerdem wird sichergestellt, dass alle Dienste erkannt wurden und für jeden Dienst festgelegt wird, in welcher Komponente die Implementierung und Verwaltung angesiedelt sein wird. Je nachdem ob ein Service aus Sicht des Consumers / Providers gesehen wird, werden andere Gesichtspunkte wichtig sein. Ein Consumer wird auf Kriterien wie die Flexibilität bei der Implementierung neuer Funktionen, Charakteristiken wie Down-Time oder Response Zeit und ökonomische Kriterien (z.B. die Kosten für die Nutzung des Dienstes) Wert legen. Der Provider wird die Implementierung mit möglichst bereits vorhandenen Komponenten, Systemen oder Programmiersprachen durchführen wollen.

5.5.5. Component Specification

Nachdem die Subsysteme, die System-Anwendungsfälle, die geschäftlichen und technischen Komponenten definiert und zugeordnet wurden, ist der nächste Schritt die Ausarbeitung der detaillierten Spezifikation jeder Komponente. Für jede Komponente werden die Eigenschaften festgelegt:

- Identifzieren der Attribute der Komponente.
- Identifizieren der Ereignisse und Nachrichten der Komponente.
- Identifizieren des internen Ablaufes.

5.5.6. Technology Realization Mapping

Nachdem die Funktionalität der Dienste und Komponenten im Detail spezifiziert wurden, können alle Teile implementiert werden. Zentrale Frage innerhalb einer IT Abteilung eines Unternehmens ist *build or buy*. Alle notwendigen Dienste können durch die Mitarbeiter der IT Abteilung selbst implementiert werden oder zugekauft werden. Die Realität liegt zwischen diesen beiden Extremsituationen. Wichtiger Aspekt in diesem Schritt das Wrapping von Legacy Applikationen mit entsprechenden Services. In den meisten Fällen wird der umhüllende Dienst ein Web Service sein. Alternative Technologien sind Messaging Dienste wie z.B. JMS.

5.6. Mashup Integration

Das unter 5.5 vorgestellte und bei der Umsetzung gewählte Modell stellt eine Vorgehensweise für die Implementierung von SOA Projekten dar. Dieses Standardmodell ist nicht speziell für den Einsatz in Projekten mit Mashups gedacht. Aufgrund der Tatsache, dass Mashups architekturell Ähnlichkeiten zu Teilaspekten einer SOA aufweisen, wird für die Implementierung der SOA Approach verwendet. Die im SOA Approach fehlende gesonderte Behandlung von Mashups wird im Zuge der Umsetzung im Kapitel 6 in den entsprechenden Phasen des Modells ergänzt.
Prinzipiell wird die Tatsache, dass Teile eines Projekts als Mashup umgesetzt werden bereits in der ersten Phase in der *Domain Decomposition* (siehe 5.5.1) erkannt bzw. festgelegt. Für die weiteren Schritte ergibt sich aufgrund dieser frühen Festlegung eine relativ klare Vorgehensweise. Das Mashup kann im Rahmen des SOA Approaches Schritt für Schritt analysiert und umgesetzt werden. In der *Goal-Service Model Creation* werden wie für die anderen Domänene die Ziele iterativ auf Sub-Ziele heruntergebrochen, bis die Services selbst übrig bleiben. In der *Sub System Analysis* kann das Mashup Framework als eigenes Teilsystem betrachtet werden. In den weiteren Prozessschritten *Service Allocation*, *Component Specification*, *Structure Enterprise Components using Patterns* und *Technology Realization Mapping* kann das Mashup wie alle anderen Komponenten betrachtet werden.

5.7. Dynamische Auswahl der Datenquellen aus Legacy bzw. ERP Systemen

Damit Endanwender mit Hilfe des Frameworks ad hoc Mashups erstellen können, muss im Prototyp eine Möglichkeit geschaffen werden, Daten aus dem ERP System beliebig wählen zu können. Ein ERP System besitzt in der Regel eine sehr umfangreiche Anzahl an Tabellen. Jede Tabelle im ERP ist eine potentielle Datenquelle für das Mashup. Eine Anforderung an die vorliegende Arbeit ist die einfache und selbstständige Erstellung eines Mashups durch Mitarbeiter aus den Fachabteilungen. Der Erstellungsprozess soll vollständig ohne Unterstützung der IT Abteilung erfolgen können. Um diese Anforderung optimal zu erfüllen, müssen Endanwender Datenquellen aus dem ERP beliebig wählen können. Diese ERP Datenquelle soll mit einer beliebiegen weiteren, die ebenfalls aus dem ERP- oder einem anderen System kommen kann, kombiniert werden können. Dabei sind auch Datenquellen aus dem Internet (z.B. ein Web Service möglich).
Bei der Auswahl der möglichen Daten aus dem ERP System muss das Thema Sicherheit betrachtet werden. Je nachdem welche Module eines ERP Systems im Unternehmen eingesetzt werden, befinden sich darin sensible Daten. Dies betrifft einerseits Daten, die zwar von Mitarbeitern gelesen und verwendet werden dürfen, allerdings nicht nach außen gelangen sollten. Dabei geht es um Daten wie Angebote, Aufträge, Kundendaten, usw. Dem gegenüber stehen Daten, die auch von den Mitarbeitern bzw. nicht allen Mitarbeitern verwendet werden sollen. Speziell angesprochen sind dabei Buchhaltungsdaten oder Daten aus der Lohnbuchhaltung. Diese Daten sind in der Regel auch unternehmensintern nur einer speziellen Gruppe von Mitarbeitern zugänglich. Diese Anforderung muss auch im Rahmen des Prototyps abgedeckt werden können. Die folgenden Absätze beschreiben mögliche Lösungsansätze.

SQL Anbindung. Das Mashup Framework hat einen direkten Zugriff auf die SQL Datenbank des ERP Systems. Im Editor des Frameworks können alle verfügbaren Da-

tenquellen in Form von Tabellen in das Mashup eingefügt werden. Für die Implementierung muss nur ein Account für den lesenden Zugriff auf die Datenbank eingerichtet werden. Voraussetzung ist, dass das Framework die Einbindung von SQL Datenbanken erlaubt. Der Punkt der Datensicherheit (welche Mitarbeiter dürfen welche Tabellen verwenden) kann hier so gelöst werden, dass der Datenbank-Account nur den Zugriff auf die entsprechenden Tabellen erlaubt. Sensible Tabellen sind damit nicht für das Mashup verfügbar. Sollen auch Mashups mit diesen sensiblen Daten (nur von ausgewählten Mitarbeitern) erstellt werden können, kann dafür eine eigener Account mit den notwendigen Berechtigungen auf der Datenbank angelegt werden.

Web Service pro Datenquelle. Eine weitere Möglichkeit ist die Bereitstellung der Daten mittels Web Service, wobei jede Tabelle in ein eigenes Web Service verpackt wird. Der Endanwender hat damit die Möglichkeit aus einer Reihe von Web Services zu wählen und für das zu erstellende Mashup die entsprechenden Dienste auszuwählen. Jedes verfügbare Service ist mit einer eigenen URL unterscheidbar und ansprechbar. Die Dienstbeschreibung wird pro Web Service mittels eines WSDL Dokuments bereitgestellt. Die Einbindung in das Framework erfolgt entweder über vorgenerierte WSDL Dokumente oder kann dynamisch von der URL des Services geladen werden (je nachdem welche Möglichkeit vom Mashup Framework unterstützt wird). Nachteilig bei dieser Lösung ist, dass für jede neue Datenquelle, die vom Anwender für die Mashuperstellung benötigt wird, ein Web Service entwickelt werden muss. Es kann davon ausgegangen werden, dass die Entwicklung nicht direkt vom Endanwender erledigt werden kann, sondern durch einen IT- oder externen Mitarbeiter durchgeführt werden muss. Dies widerspricht allerdings der Anforderung, Mashups ad hoc von Benutzern erstellen lassen zu können. Der Vorteil dieses Lösungsansatzes ist, dass die Datenquellen auch von mehreren Basissystemen kommen können und diese einfach miteinander kombiniert werden können.

Generisches Web Service. Als weiterführende Möglichkeit zum Web Service wie im vorigen Abschnitt beschrieben, ist der Einsatz eines generischen Web Services (pro System aus dem Daten verwendet werden sollen). Dieses Web Service bietet die Möglichkeit über Parameter die gewünschten Daten vom System laden zu können. Mit dem Parameter kann festgelegt werden von welcher Tabelle das Web Service die Daten bereitstellt. Mit weiteren Parametern können Einschränkungen für die Datenselektion getroffen werden. Prinzipiell können mit diesem Web Service alle verfügbaren Daten eines Systems in das Mashup eingefügt werden.

Fazit. Die Vorteile der Bereitstellung der Daten in Form einer direkten SQL Anbindung liegen in der relativ einfach Realisierung, vorausgesetzt das Framework bietet die Möglichkeit zur Einbindung von SQL Datenbanken, und in der Tatsache, dass der Benutzer ad hoc beliebige Datenquellen wählen kann. Bei der Implementierung sollte Rücksicht genommen werden, dass nicht alle Tabellen von allen Benutzern verwendet werden dürfen.
Die Bereitstellung jeder Datenquelle als eigenes Web Service bietet den Vorteil, dass nur Daten, die tatsächlich von Benutzern für Mashups verwendet werden sollen, bereit gestellt werden können. Ansonsten hat diese Möglichkeit nur Nachteile im Vergleich zu den anderen beiden beschriebenen.
Mit dem generischen Web Service können Daten ad hoc vom Benutzer gewählt werden. Es kann davon ausgegangen werden, dass diese Art der Datenbereitstellung von den meisten Frameworks unterstützt wird (im Vergleich zur Anbindung über SQL).

Nachteilig ist hier der relativ hohe Aufwand der Implementierung des generischen Web Services. Weiters muss sichergestellt werden, dass das Web Service nicht Daten von sensiblen Tabellen bereitstellt (z.B. Daten aus der Lohnbuchhaltung). Entweder werden bestimmte Tabellen generell ausgeschlossen oder es kann am Web Service eine Authentifizierung erfolgen, mit der festgelegt wird, welche Tabellen gelesen werden können.

6. Implementierung des Prototyps anhand des Vorgehensmodells

In diesem Kapitel wird die technische Umsetzung des Prototyps beschrieben. Für die Implementierung werden wie im Kapitel 5 begründet der *SOA Approach* als Vorgehensmodell und *JackBe Presto* als Mashup Framework eingesetzt.
Das Kapitel 6 zeigt die konkrete Vorgehensweise anhand des SOA Approaches, um ausgehend von den Anforderungen eine lauffähige Software zu erhalten. Die Kapitel 6.1 bis 6.6 stellen die diskreten Schritte im SOA Approach dar. Das vorliegende Kapitel soll als Referenz für ein Unternehmen dienen, das den Mitarbeitern Mashups als Instrument bereitstellen möchte. Es zeigt welche vorbereitenden technischen Schritte von den Planung bis hin zum Ausrollen des fertigen Produktes notwendig sind. Die durchgeführten Abschnitte werden anhand der Phasen des SOA Approach durchgeführt und erklärt.
Der Prototyp enthält wie im Kapitel 5 beschrieben als Teilkomponente ein Web Service. Im Rahmen der Implementierung wird ein Web Service zur Einheitenumrechnung von Gewichtsmaßen eingesetzt (`http://www.webservicex.net/ConvertWeight.asmx`).
Ergänzend wird an den entsprechenden Stellen während der Implementierung darauf eingegangen, was zu tun ist, wenn das hier eingesetzte Web Service gegen ein anderes ausgetauscht werden soll.
Wie am Beginn der Arbeit in der Zielsetzung definiert, sollen durch den Einsatz von Enterprise Mashups unter anderem Kosten für das Unternehmen gespart werden. Dem Endanwender soll mit dem Mashup ein Tool zur Verfügung gestellt werden, das ihm eine ad hoc Erstellung einer Auswertung oder einfachen Applikation erlaubt ohne die IT Abteilung dafür konsultieren zu müssen. Diese Zielsetzung wird im aktuellen Kapitel umgesetzt.
Das Kapitel 7 stellt einen Leitfaden für Anwender dar, die vom Unternehmen ein Mashup Framework bereitgestellt bekommen und autonom Mashups entwickeln. Es zeigt wie ausgehend vom entwickelten Prototyp ein Mashup mit einer anderen Aufgabenstellung implementiert werden kann, das wiederrum Daten aus dem ERP System mit einem Web Service verknüpft.

Die in den folgenden Abschnitten verwendeten Diagramme sind in Anlehnung an die des IBM SOA Aproach erstellt [16]. Die Diagramme weisen Ähnlichekeiten zur UML Notation auf, sind jedoch um proprietäre Merkmale ergänzt.

6.1. Domain Decomposition

Das Ergebnis der Domain Decomposition ist die Zerlegung in funktionelle Bereiche der Gesamtapplikation. In diesem Schritt wurden zwei Bereich ermittelt, siehe Abbildung 6.1:

- *Web Browser*: Das zentrale Bedienelement für den Benutzer ist der Web Browser. Der Browser hat für den weiteren Analyseprozess keine Relevanz und wird in den folgenden Schritten nicht betrachtet.

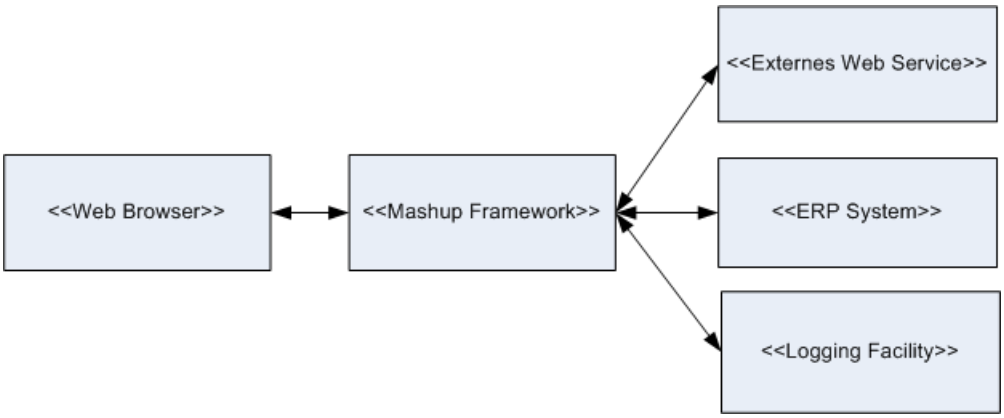

Abbildung 6.1.: Domain Decomposition, basierend auf [16]

- *Mashup Framework*: Das Mashup Framework, welches vom Endanwender bedient wird.
- *ERP System*: Prozesse und Daten, die aus dem ERP System stammen.
- *Externes Web Service*: Dienste, die aus dem Internet in Anspruch genommen werden.
- *Logging Facility*: Tool zur zentralen Sammlung aller Logging Daten.

Use Case Diagramm Abbildung 6.2 zeigt die im Prototyp umgesetzten Anwendungsfälle.

- UC1 Mashup Einheitenkonvertierung: Der Aufruf des Mashups wird am Framework durch den Anwender angestoßen. Dieser Vorgang inkludiert eine Reihe weiter Anwendungsfälle.
- UC2 Projekt laden: Die Ausführung dieser Funktion wird vom Mashup Framework angestoßen. Nach der Auswahl des Projektknotens werden die Basisinformationen des Projektes aus dem ERP System geladen. In weiterer Folge werden der zum Projekt gehörende Auftrag sowie die Auftragspositionen geladen. Die bei den Auftragspositionen hinterlegten Gewichtsmaße sind der Input für das Web Service zur Einheitskonvertierung.
- UC3 Einheit konvertieren: Dieser Anwendungsfall wird vom Mashup Framework für jeden zum Projekt gehörenden Artikel ausgeführt. Die eigentliche Berechnung des Faktors zur Konvertierung der Einheit wird vom verwendeten Web Service geliefert.
- UC4 View Events: Die gespeicherten Logdaten können über eine Funktion des Logging Facility angezeigt und gefiltert werden.
- UC5 Log Events: Die Vorgänge im ERP System werden zu Zwecken der Nachverfolgung (wer welche Daten angefordert hat) protokolliert. Das Logging Facility bietet dazu eine Funktion mit der Logdaten geschrieben werden können. Detaillierte Beschreibung des Loggings siehe Kapitel 5.3.

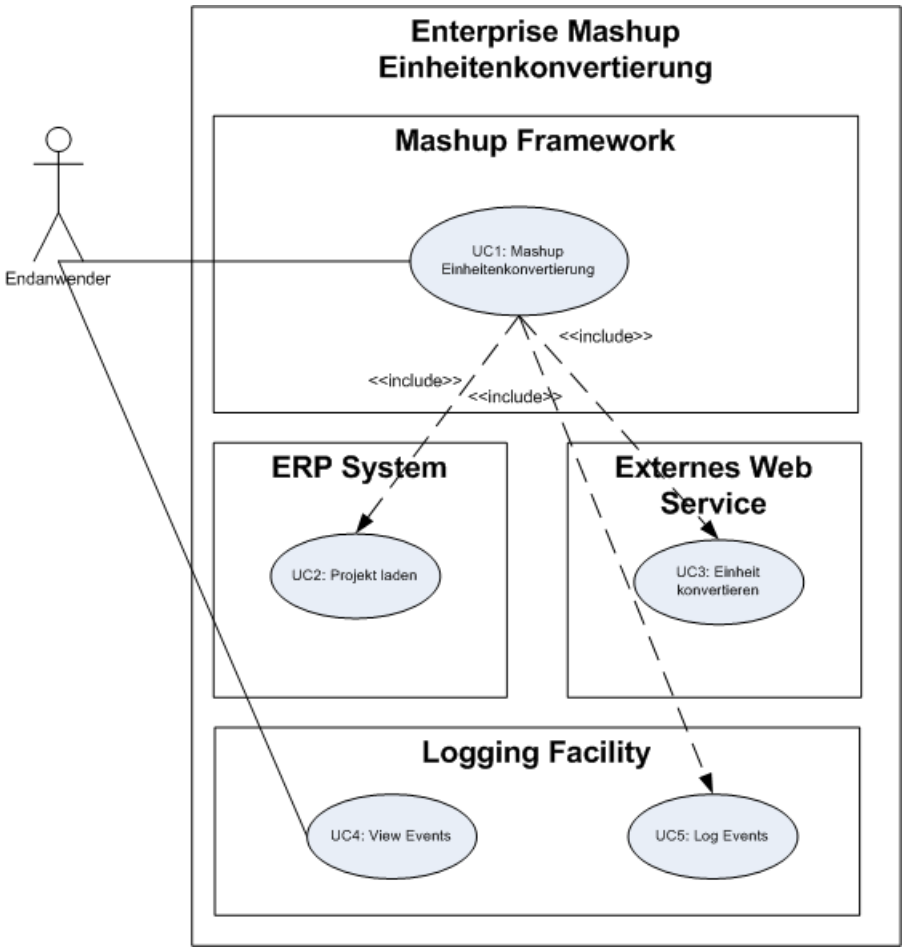

Abbildung 6.2.: Prototyp Use Case

Die Anwendungsfälle für das umgesetzte Mashup sind sehr begrenzt gewählt. Der Prototyp soll primär die Vorgehensweise für die Implementierung eines Enterprise Mashups demonstrieren. Weitere Anwendungsfälle können nach dem gezeigten Vorgehensmodell beliebig ergänzt werden.

In weiterer Folge werden die funktionellen Bereiche auf die Subsystem gemappt. In den meisten Fällen ist zwischen den funktionellen Bereichen und dem Subsystem ein direkter Zusammenhang - alle Use Cases eines funktionellen Bereiches werden im gleichen Subsystem implementiert.

Business und Application Integration Patterns. Als letzter Punkt der Domain Decomposition werden Business und Application Integration Patterns angewendet [16]. Tabelle 6.1 zeigt die identifizierten Use Cases sowie deren Pattern für die Umsetzung. Die eingesetzten Business Pattern sind speziell für den Einsatz im Enterprise- und E-Business-Bereich von IBM entwickelt. Alle Muster haben die Gemeinsamkeit, dass sie ein Layeraufbau besitzen, bei dem der nächst höhere Layer auf dem darunterliegenden Layer aufbaut. Im Rahmen des Vorgehensmodells werden die entsprechenden Pattern ausgewählt und auf die Komponenten angewendet. Die IBM Business Pattern sind in folgende Bereiche gegliedert (Details zu den Pattern siehe [16]):

- *Business* - beschreiben den Zusammenhang zwischen dem Anwender, dem Geschäfts-

ablauf und der in der Applikation verwendeten Daten: Self-Service (User-to-Business), Information Aggregation (User-to-Data), Collaboration (User-to-User), Extended Enterprise (Business-to-Business).

- *Integration* - werden für die Verbindung der Business Pattern verwendet: Access Integration, Application Integration.
- *Composite* - typische Verwendungen von Business- und Integration-Patterns sind in Form von Composite-Patterns zusammengefasst. Es werden Standard-Composite-Pattern für folgende Bereiche angeboten: Electronic Commerce, Portal, Account Access, Trading Exchange, Sell-Side-Hub (Supplier), Buy-Side-Hub (Purchaser).
- *Custom design* - sind keine der vorgeschlagenen Patterns anwendbar, kann je nach Anforderung ein eigenes Design implementiert werden.

Für die Umsetzung des Prototyps wurden wie in Tabelle 6.1 aufgelistet für die Komponenten die entsprechenden Pattern gewählt. Die gewählten Pattern werden im folgenden detaillierter beschrieben:

Self-Service. Das Self-Service Pattern beschreibt eine Applikation, wo der Anwender einen Geschäftsprozess über Internet oder Intranet steuert. Im einfachsten Fall handelt es sich dabei um eine relativ simple Webanwendung.

Extended Enterprise. Hier wird eine Verbindung zwischen zwei oder mehr Geschäftsprozessen etabliert. Die verbundenen Geschäftsprozesse existieren dabei nicht im gleichen Unternehmen, sondern sind über unterschiedliche Unternehmen verteilt. Ein Beipiel dafür ist der Datenaustausch zwischen Business-Systemen über EDI (electronic data interchange).

Information Aggregation. Das Information Aggregtation Pattern wird verwendet, wenn von einer großen Datenmenge wie sie z.B. in Business Intelligence Systemen vorkommen, durch den Anwender spezielle Daten extrahiert werden sollen.

6.2. Goal-Service Model Creation

Die im vorigen Schritt identifizierten Kandidaten für Services werden in diesem Schritt auf Vollständigkeit überprüft. Der erzeugte Baum an Zielen wird solange in Sub-Ziele unterteilt, bis die Services eindeutig sind. Für den Prototyp werden die Bereiche aus der Domain Decomposition getrennt auf ihre Ziele untersucht. Abbildung 6.3 zeigt für jeden Bereich die ermittelten Services (Services sind in kursiver Schrift dargestellt).

Für den Bereich des *Mashup Frameworks* werden über die von der IT bzw. Geschäftsleitung definierten Anforderung zu konkreten Services gefunden: **Kostenreduktion** das Sub-Ziel **autonome Erstellung einfacher Anwendungen und Auswertungen durch Mitarbeiter**.
Im Bereich des *externen Web Services* ist das Goal-Service Model mit dem vom Anbieter zur Verfügung gestellten Dienst **Einheit konvertieren** abgeschlossen.
Beim *ERP System* ist als Hauptziel die **Verbesserung der Projekteffizienz** definiert. Zum Erreichen der Anforderung werden zwei Subziele **Analyse der Projektdaten durch externe Tools** und **Bereitstellen der Auftragsdaten über Schnittstellen** ermittelt. Über diese Ziele kann zu den Services gefunden werden.

Use Case	Description	Invoker	Implemented by	Pattern
UC1: Mashup Einheitenkonvertierung	Anwender kann das Mashup über den Browser bedienen	Endanwender (Web Browser)	Mashup Framework	Self-Service
UC2 : Projekt laden, Auftrag laden, Auftragspositionen laden	laden aller Projektdaten aus dem ERP System	Mashup Framework	ERP System	Extended Enterprise
UC3: Einheit konvertieren	Konvertierung der Ausgangseinheit auf die Zieleinheit	Mashup Framework	externes Web Service	nicht relevant
UC4: View Events	anzeigen und filtern der Logdaten	Administrator	Logging Facility	Information Aggregation
UC5: Log Events	Event in Log schreiben	ERP System	Logging Facility	Extended Enterprise

Tabelle 6.1.: Business and Application Integration patterns, basierend auf [16], detaillierte Beschreibung der Use Cases siehe 6.1

Beim *Logging Facility* können über das Ziel **Verwalten von Logdaten** die beiden notwendige Dienste ermittelt werden.

Ergebnis der Goal-Service Model Creation ist eine eindeutige Zuordenbarkeit der Services zu den Bereichen sowie für welches geschäftliche Ziel ein bestimmtes Service notwendig ist.

6.3. Subsystem Analysis

In der Subsystem Analysis werden die geschäftlichen Use Cases auf Systeme aufgeteilt und verfeinert. Jeder System Use Case unterstützt einen bestimmten geschäftlichen Use Case. Jedes Subsystem ist aus geschäftlichen und technischen Komponenten aufgebaut. Die Analyse der Aufgabenstellung für den Prototyp bringt folgende Subsysteme als Ergebnis:

- **ERP System**, siehe Abbildung 6.4 (Die Subsysteme verwenden Business- und Technical Components, siehe Legende): Das ERP System bietet Services zum Laden von Projekten und Aufträgen an. Die Dienste werden mit Hilfe der geschäftlichen Komponenten *Projekt* und *Artikel* umgesetzt. Diese beiden Komponenten sind im ERP System in Form von Tabellen vorhanden. Auf diese Tabellen wird im Mashup Framework per SQL zugegriffen. Das Hosting der Datenbank mit den Tabellen erfolgt auf einem Mircosoft SQL Server. Die technische Realisierung erfolgt mittels eines JDBC Treibers für den SQL Server (dieser wird vom Hersteller angeboten), da das Framework als Java Applikation ausgeführt ist.
  Die in der Subsystem Analysis verwendeten Diagramme zeigen jene Vorgänge, die systemübergreifend ausgeführt werden, als aus dem Subsystem herausragende Ellipse mit einem darauf gerichteten Pfeil. Dies soll einen Aufruf des Vorganges aus

Mashup Framework
1 Kostenreduktion
1.1 autonome Erstellung einfacher Anwendungen und Auswertungen durch Mitarbeiter
1.1.1 Projektauswertungen
1.1.1.1 Mashup Einheitenkonvertierung

Externes Web Service
2 Einheit konvertieren

ERP System
3 Verbessern der Projekteffizienz
3.1. Analyse von Projekten durch externe Tools
3.1.1 Bereitstellen der Projektdaten über Schnittstellen
3.1.1.1 Projekt laden
3.2 Bereitstellen der Auftragsdaten über Schnittstellen
3.2.1 Auftrag laden
3.2.2 Auftragspositionen laden

Logging Facility
4 Verwalten von Logdaten
4.1 Log Events
4.2 View Events

Abbildung 6.3.: Goal Service Model, basierend auf [16]

dem vorgelaterten Subsystem verdeutlichen. Die eigentliche Ausführung erfolgt jeweils im dargestellten System.

- **Logging Facility**, siehe Abbildung 6.5: Das Logging Tool stellt Möglichkeiten zum Schreiben und Lesen bzw. Filtern von Loginformationen bereit. Geschäftliche Komponenten sind hier die einzelnen *Log Einträge*, technische Komponente ist der *Filter*.
- **Mashup Framework**, siehe Abbildung 6.6: Das Mashup Framework kombiniert die angebotenen Dienste aus dem ERP System, dem externen Web Service und dem Logging Facility und bietet diese in Form einer neuen Applikation an. Das Framework arbeitet mit den geschäftlichen Komponenten *Artikel* und *Projekt* aus dem ERP System. Weiters ist stellt das Mashup die Schnittstelle zum *Anwender* dar. Als technische Komponente im Framework sind die *Einbindung von SOAP Web Services* und der *Zugriff auf das ERP System* notwendig.
- **Externes Web Service**: Das externe Web Service stellt ein eigenes Subsystem dar. Auf den technischen und funktionellen Aufbau kann beim Einsatz eines externen Services keinen Einfluss genommen werden. Für den Prototyp wird wie

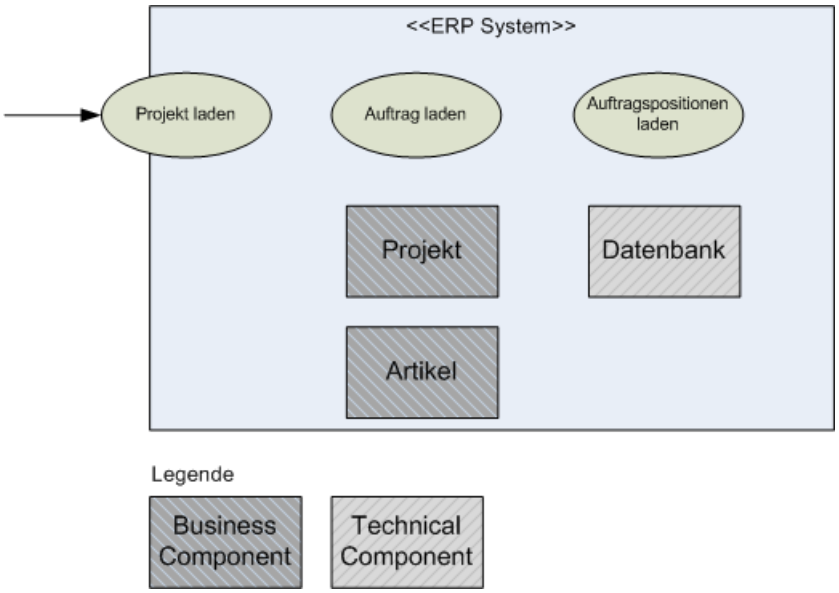

Abbildung 6.4.: Subsystem ERP, basierend auf [16]

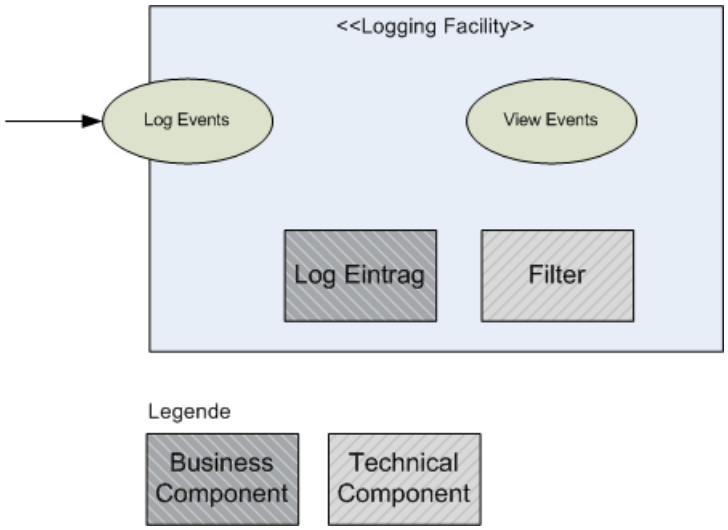

Abbildung 6.5.: Subsystem Logging Facility, basierend auf [16]

einleitend angemerkt ein Web Service zur Umrechnung von Gewichtseinheiten eingebunden. Mit diesem wird das bei den Auftragspositionen gespeicherte Gewicht im ERP System in eine andere Einheit umgerechnet.
Im Leitfaden, siehe Kapitel 7, wird ein weiteres Mashup entwickelt, bei dem eine anderes Web Service eingesetzt wird. Welches Web Service vom Benutzer im Mashup integriert wird spielt grundsätzlich keine Rolle. Mit Hilfe eines Mashups wäre es auch möglich, mehrere Web Services zu einer neuen Anwendung zu orchestrieren. Ob diese Dienste extern im Web oder intern im Unternehmen angeboten werden spielt ebenfalls keine Rolle.

Mit der Durchführung der Subsystem Analyse wurden alle notwendigen Services identifiziert und können in den weiteren Schritten implementiert werden.

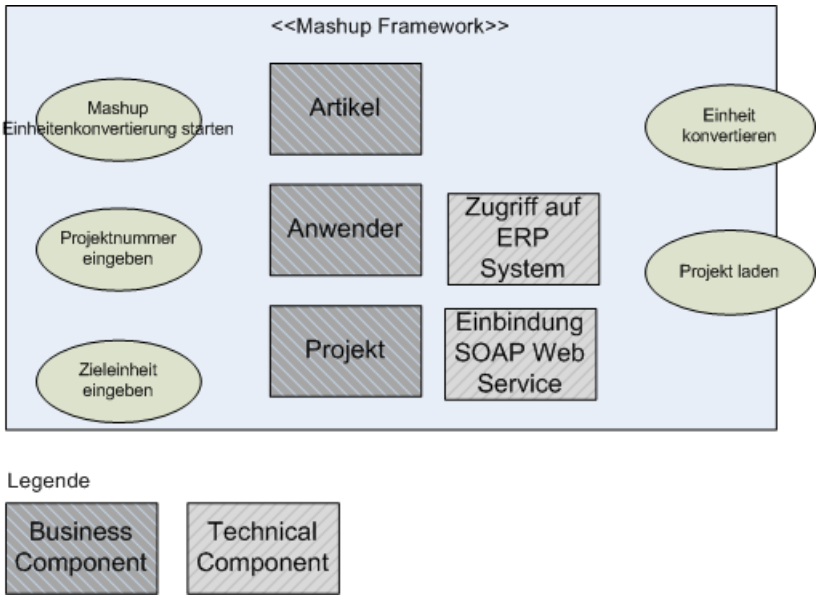

Abbildung 6.6.: Subsystem Mashup Framework, basierend auf [16]

6.4. Service Allocation

In der Service Allocation werden die identifizierten Services aus der Goal-Service Model Creation mit den in der Subsystem Analyse ermittelten Komponenten zusammengeführt, siehe Abbildung 6.7.

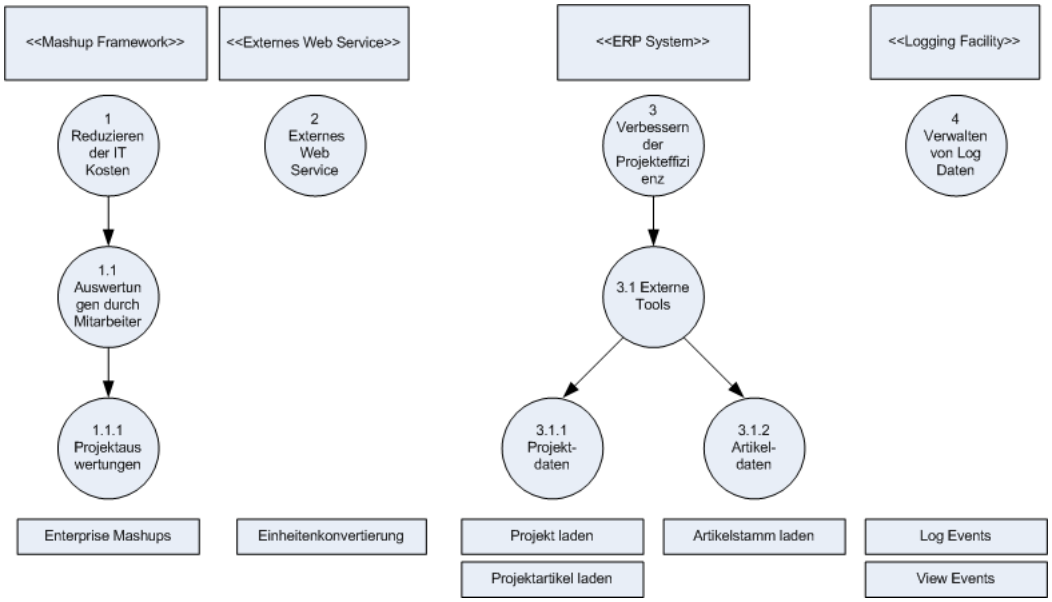

Abbildung 6.7.: Service Allocation, basierend auf [16]

Im Zuge der Goal-Service Model Creation wurden keine neuen Services mehr identifiziert. Damit ist die Zuordnung der bereits gefundenen Services zu den Komponenten direkt möglich. Das Logging wird innerhalb des Mashup Frameworks durchgeführt. Alle durchgeführten Aktionen werden in einer Tabelle gespeichert. Die Logdaten können über ein User Interface gefilter und soritert abgerufen werden.

6.5. Component Specification

In dieser Phase werden die Schnittstellen der einzelnen Komponenten festgelegt. Die Schnittstellen wurden bereits in der Subsystem Analysis identifiziert, die Zuordnung zu den Komponenten erfolgte in der Service Allocation. Der aktuelle Schritt ist die konkrete technische Spezifizierung aller Schnittstellen in der Form, dass Name, Anzahl und Typ der Parameter sowie der Rückgabeparameter festgelegt werden.

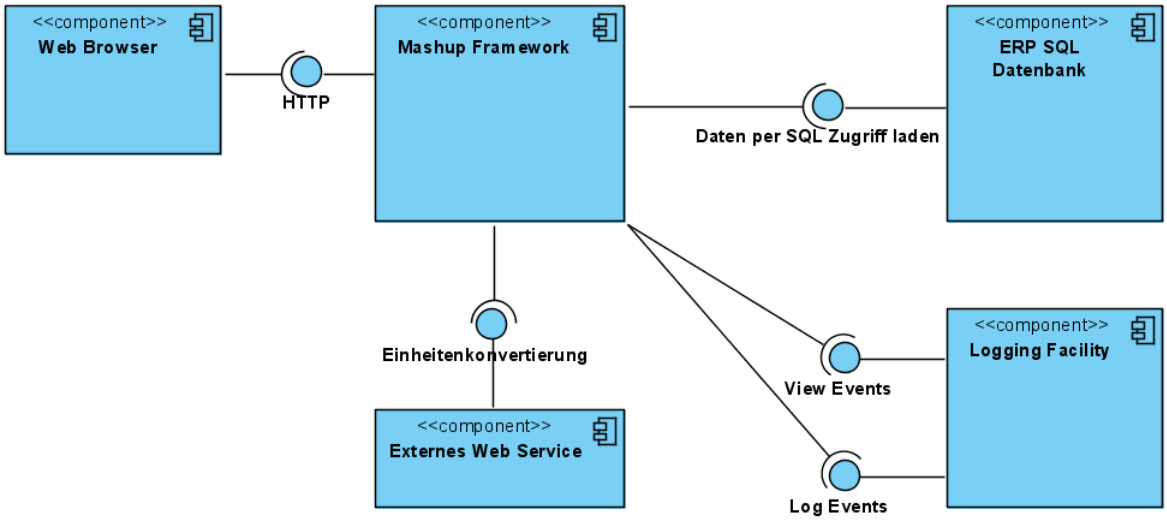

Abbildung 6.8.: Component Specification

Web Browser Der Browser ist das Instrument für den Benutzer um den Zugriff auf das Mashup herzustellen. Die verwendete Schnittstelle ist der im Framework integrierte Web Server auf den per HTTP Protokoll zugegriffen wird.

Mashup Framework Das Framework bietet selbst abgesehen vom Zugriff per HTTP keine Schnittstellen an. Es verwendet die Schnittstellen der anderen Systeme (SQL Datenbank, Web Service und Logging Facility).

Web Service Die Schnittstelle des Web Services zur Einheitenumrechnung ist vom Anbieter vorgegeben und ist wie in Listing 6.1 gezeigt aufgebaut. Beim Aufruf werden neben dem Gewicht als `Double` die beiden `Strings` mit der Ausgangseinheit und der Zieleinheit angegeben. Der Rückgabewert ist ein `Double` mit dem in die Zieleinheit konvertierten Gewicht.

Listing 6.1: Prototyp Web Service Einheitenkonvertierung

```
Double conversionRate(Double weight, String formUnit, String toUnit);
```

ERP SQL Datenbank Auf die SQL Datenbank wird mit Hilfe des entsprechenden JDBC Treiber zugegriffen (das gewählte Framework JackBe Presto ist in Java codiert). Darüber können vom Framework alle notwendigen SQL Befehle abgesetzt werden. Je nach eingesetzter Datenbank und Treiberversion ändert sich der Umfang der konkret ausführbaren Funktionen. Details dazu sind in den Unterlagen des Datenbankherstellers bzw. des Anbieters des JDBC Treibers nachzulesen.

In Abbildung 6.9 ist das im Prototyp eingesetzte Datenbankmodell zu sehen. Es handelt sich dabei um einen Auszug der verwendeten Tabellen als auch der verwendeten Spalten aus den dargestellten Tabellen (das komplette Datenbankmodell des eingesetzten ERP Systems umfasst ca. 2800 Tabellen).
Ausgangspunkt ist die Tabelle mit Projekten *ProjTable* mit dem Primärschlüssel *ProjId*. Dieser wird beim Aufruf des Mashups vom Benutzer eingegeben. Zu jedem Projekt können mehrere Aufträge in der *SalesTable* vorhanden sein, mit dem Fremdschlüssel *ProjId*. Jeder Auftrag kann mehrere Auftragspositionen in der *SalesLine* enthalten, verbunden über den Fremdschlüssel *SalesId*. Aus der *SalesLine* werden die Spalten *CurrencyCode, SalesUnit und Weight* im Mashup herangezogen und als Input für die aufgerufenen Web Services verwendet.

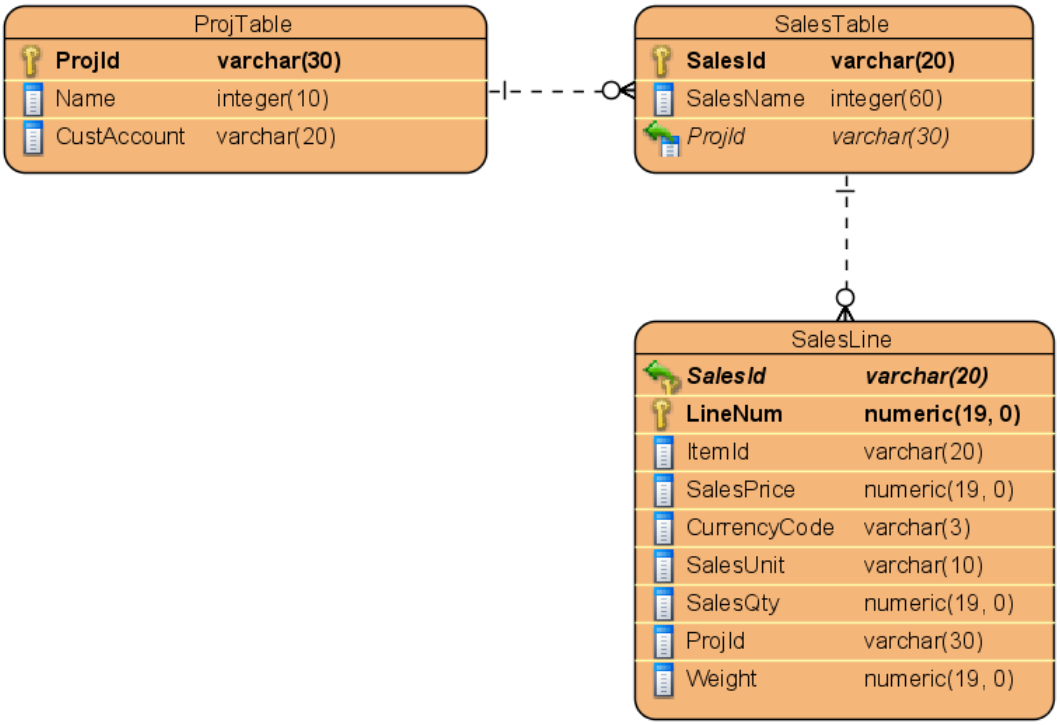

Abbildung 6.9.: Datenbank Modell (Auszug)

Logging Facility Das Logging Facility stellt jeweils eine Schnittstelle für das Erstellen von Log Events und das Anzeigen von Events zur Verfügung. Im Prototyp wird das integrierte Logging des Mashup Frameworks verwendet. Die detaillierte Schnittstellenspezifikation ist deswegen nicht relevant.

6.6. Technology Realization Mapping

Nachdem alle Details für die Implementierung des Prototyps festgelegt wurden, muss für die einzelnen Komponenten die Make or Buy Entscheidung getroffen werden.

Abbildung 6.10 zeigt die Infrastruktur des Prototyps. Die Trennung zwischen Internet und Unternehmensnetzwerk (Enterprise Domain) erfolgt über einen Router. In der DMZ (Demilitarized Zone oder Perimeter Network) befinden sich die Server, darunter auch der Rechner auf dem das Mashup Framework ausgeführt wird. Die Trennung der

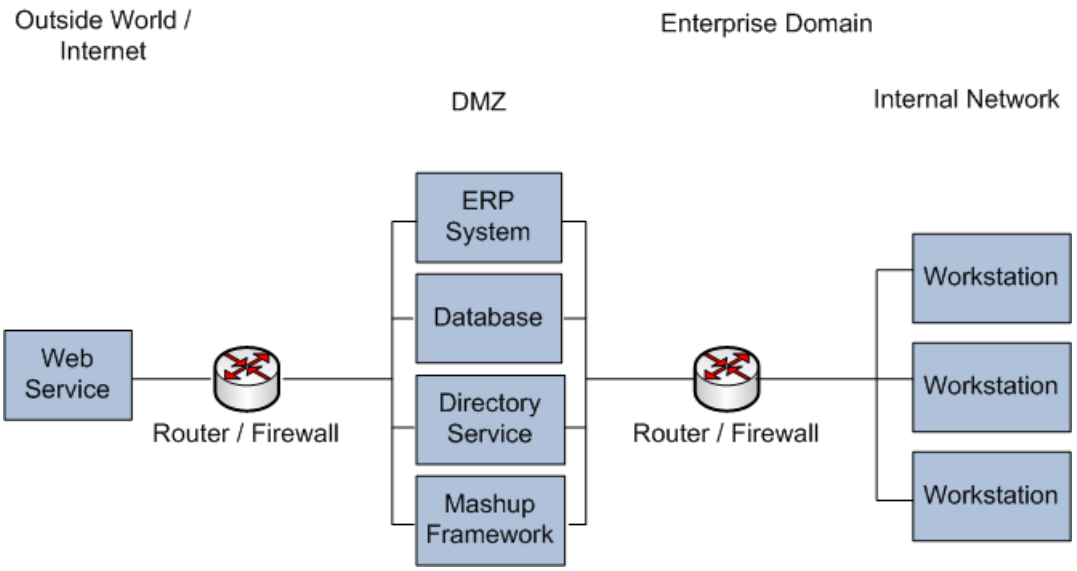

Abbildung 6.10.: Technology Realization Mapping, basierend auf [16]

DMZ vom unternehmensinternen Netzwerk mit den Workstations der Benutzer erfolgt über einen weiteren Router. In der Phase *Technology Realization Mapping* wird auch die konkrete Spezifikation der Hardware und des Betriebssystems der Systeme festgelegt:

- *ERP System*: Hardware: Virtuelle Maschine als eine Instanz einer VM Ware Workstation 6.1. Betriebssystem: Windows Server 2008 x64.
- *Database*: Hardware: Virtuelle Maschine als eine Instanz einer VM Ware Workstation 6.1. Die Datenbank befindet sich auf der gleichen virtuellen Instanz wie das ERP System. Betriebssystem: Windows Server 2008 x64. Datenbank: Microsoft SQL Server 2008.
- *Directory Service*: Hardware: Virtuelle Maschine als eine Instanz einer VM Ware Workstation 6.1. Die Datenbank befindet sich auf der gleichen virtuellen Instanz wie das ERP System. Betriebssystem: Windows Server 2008 x64. Directory Service: Microsoft Active Directory.
- *Mashup Framework*: Hardware: Physischer Rechner mit Intel i7 Core Prozessor. Betriebssystem: Windows 7 x64 SP1.
- *Workstation*: Hardware: Physischer Rechner mit Intel i7 Core Prozessor. Betriebssystem: Windows 7 x64 SP1.

6.7. Implementierung

Nachdem das Mashup nach dem beschriebenen Vorgehensmodell geplant wurde, kann die Implementierung erfolgen. An dieser Stelle werden die erstellten Artefakte vorgestellt. Ein ausführliche Beschreibung der durchgeführten Schritte ist im Kapitel 7 zu finden.

Bei Enterprise Mashups ist wie am Beginn der Arbeit unter 1.2 beschrieben die **Datensicherheit** ein zentrales Thema. Bei der Umsetzung des Prototyps wird aus

den unter 5.7 beschriebenen Möglichkeiten zur Anbindung von Legacy Systemen, jene Möglichkeit mit einem direkten Zugriff auf die SQL Datenbank gewählt. Um die Datensicherheit gewährleisten zu können, muss beachtet werden, dass der Zugriff nur auf ausgewählte Tabellen erlaubt wird. Unternehmensintern sollten nicht alle Benutzer Zugriff z.B. auf die Tabellen einer Lohnbuchhaltung besitzen. Beim Erstellen der Accounts auf der Datenbank muss diesem Umstand Rechnung getragen werden. Der Schreibzugriff auf die SQL Datenbank ist grundsätzlich nicht notwendig und sollte deshalb unterbunden werden. Die Einrichtung der Benutzer auf einem Microsoft SQL Server in Hinblick auf Sicherheitsanforderungen ist unter 7.1.1 im Leitfaden beschrieben.

Im Prototyp wird für die Konvertierung der Einheiten ein freies Web Service eingesetzt, `http://www.webservicex.net/ConvertWeight.asmx`. Neben einer Reihe von freien im Internet verfügbaren Diensten gibt es kommerzielle Anbieter für Web Services. Diese können üblicherweise mit Hilfe von Suchmaschinen gefunden werden, bzw. stellen Unternehmen wie Google oder Yahoo frei verwendbare Dienste bereit. Eine Sammlung von kostenlosen Services mit unterschiedlichen Funktionen kann unter `http://www.webservicex.net` gefunden werden. Darin ist auch das Web Service für den Prototyp enthalten. Kommerzielle Anbieter wie z.B. Navteq stellen Dienste bereit, die eine garantierte Verfügbarkeit sowie eine entsprechende Dokumentatione für die Einbindung umfassen. Die Web Services von kommerziellen Anbietern können oft in einem Testzeitraum kostenlos getestet werden. Danach fallen die entsprechenden Gebrühren auf Basis einer pauschalen Abrechnung oder einer Abrechnung nach Dienstnutzung an.

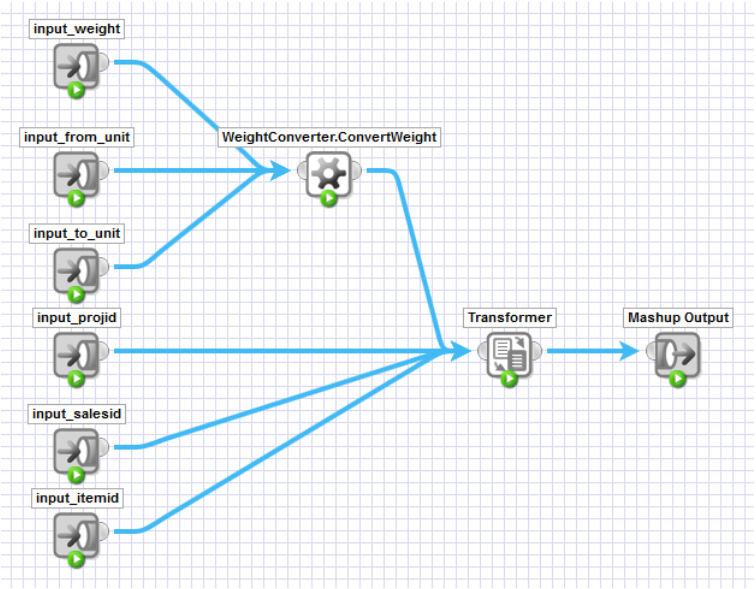

Abbildung 6.11.: Teil 1 Gewichtskonvertierung

Vorgehensweise Das Mashup besteht aus zwei Teilen. Der Teil 1 dient zur Konvertierung der Währungen, siehe Abbildung 6.11. Dazu wird ein Web Service mit dem Gewichtsmaß, der Ausgangseinheit und der Zieleinheit aufgerufen. Das ermittelte Ergebnis wird zusammen mit weiteren Parametern in einen *Transformer* geschickt, der das Ergebnis in einem XML Dokument aufbaut. Der Teil 2, siehe Abbildung 6.14, lädt die Daten aus der Datenbank zum eingegeben Projekt und führt im *Loop* Block für jeden

Datensatz die Funktion aus Teil 1 durch. Endergebnis ist eine Reihe von Datensätzen mit den konvertierten Gewichten der Auftragspositionen.

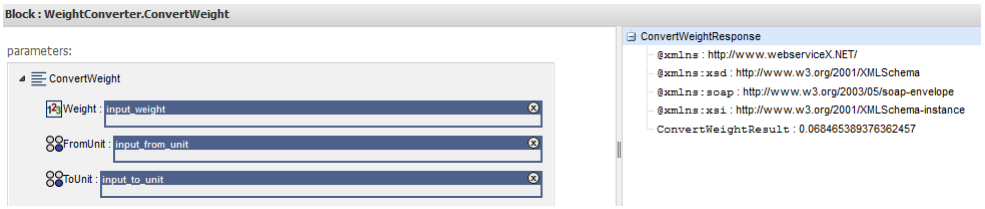

Abbildung 6.12.: Ergebnis Block *Web Service Einheitenkonvertierung*

Teil 1 Einheitenkonvertierung, siehe Abbildung 6.11: Im Teil 1 des Mashups wird die Umrechnung der Gewichtseinheiten mit Hilfe des Web Services vorgenommen. Wie ein Web Service im Framework registriert wird, ist detailliert im Kapitel 7.1 beschrieben. An dieser Stelle wird von einem korrekt registrierten und einsatzbereiten Web Service ausgegangen. Es erhält einen Inputparameter *input_from_unit* der die Einheit des Artikels darstellt wie er im ERP System gespeichert ist. Der Inputparameter *input_to_unit* stellt die Einheit dar, in welche das Gewicht konvertiert wird. Der Parameter *input_weight* ist das Gewichtsmaß des Artikels im ERP System. Das Web Service berechnet ausgehend von den drei Parametern das Gewichtsmaß des Artikels in der gewünschten Zieleinheit. Mit dem *Transform* Block werden die aus dem Web Service ermittelten Daten und den weiteren Inputparametern als Output aufbereitet. Der Transformer ist so konfiguriert, siehe Abbildung 6.13, dass er aus den erhalten Werten ein neues XML Dokument erstellt. Dieses ist gleichzeitig der Output des Mashups aus Teil 1.

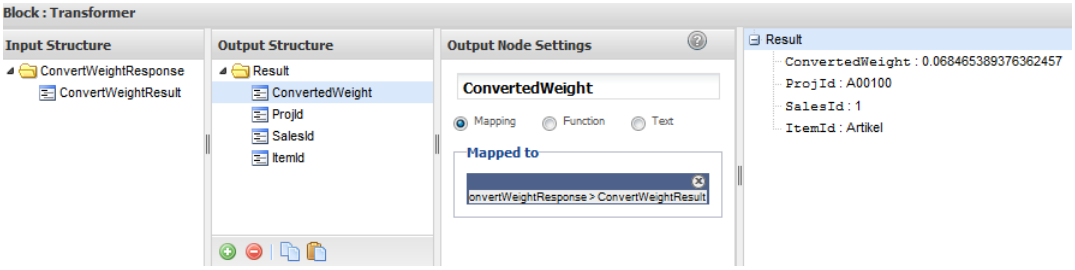

Abbildung 6.13.: Ergebnis Block *Transformer*

Ergebnisse Teil 1: Aus dem Teil 1 können zwei interessante Zwischenergebnise gewonnen werden, welche bei der schrittweisen Entwicklung des Mashups als permanente Kontrolle dienen. Das Ergebnis aus dem *Web Service zur Einheitenkonvertierung* ist in Abbildung 6.12 zu sehen. Es umfasst ein XML Dokument mit dem in die Zieleinzeit konvertiertem Gewichtsmaß des Artikels. Dieser Block ist so konfiguriert, dass die entsprechenden Inputparameter des Mashups mit den Inputs des Web Services verbunden werden. Das Ergebnis aus dem Web Service wird im *Transformer* Block zumsammen mit den restlichen Parametern zu einem neuen XML Dokument aufbereitet, siehe Abbildung 6.13. In der Konfiguration dieses Blockes kann ein XML Dokumentenstruktur aufgebaut werden und die einzelnen Elemente mit den Inputs des Blockes befüllt werden. Der Block liefert als Ergebnis ein XML Dokument, welches gleichzeitig der Output des Mashups ist.

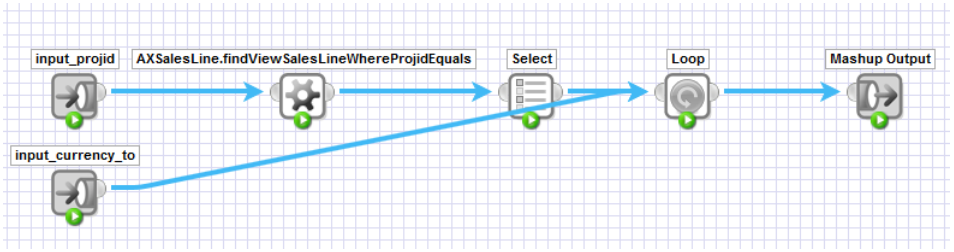

Abbildung 6.14.: Daten laden und Output erzeugen

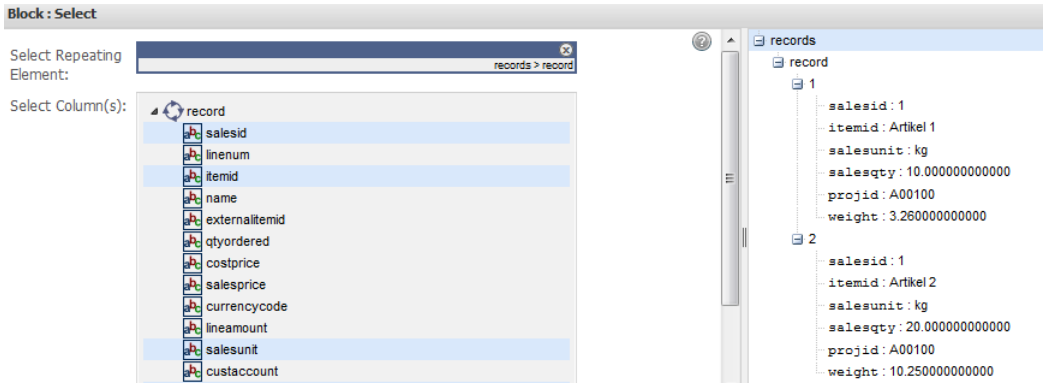

Abbildung 6.15.: Ergebnis Block *Select*

Teil 2 Daten laden und Output erzeugen, siehe Abbildung 6.14: Der Teil 2 lädt die entsprechenden Daten aus der Datenbank und ermittelt daraus durch wiederholten Aufruf des Teiles 1 für jeden Datensatz die Endergebnisse. Die Daten werden über den Block *AXSalesLine.findViewSalesLineWhereProjidEquals* aus der Datenbank mit einer Selektion auf eine bestimmte Projektnummer ermittelt. Die Projektnummer wird über den Inputparamter *input_projid* festgelegt. Dieser wird später vom Benutzer beim Aufruf des Mashups mit einem Wert belegt. Der *Select* Block dient rein zur Einschränkung der selektieren Spalten, könnte prinzipiell auch weggelassen werden. Der folgende *Loop* Block iteriert über alle vom *Select* Block gelieferten Datensätze und ruft für jeden den Teil 1 zusammen mit den Inputparametern *input_unit_from* und *input_unit_to* auf. Die beiden Parameter werden vom Benutzer beim Aufruf des Mashups eingegeben und legen die Ausgangs- und Zieleinheit fest.

Ergebnisse Teil 2: Das Zwischenergebnis aus dem *Select* Block stellt eine Liste von Datensätzen mit den selektierten Feldern dar, siehe Abbildung 6.15. Im Block können alle von der Datenbank gelieferten Spalten der Tabelle selektiert werden. Das Ergebnis aus dem folgenden *Loop* Block ist ein XML Dokument, wie bereits als Ergebnis im Teil 1 gezeigt, mit dem Unterschied, dass das Dokument aus mehreren *Result* Knoten besteht, siehe Abbildung 6.16. In Listing 6.2 ist das Ergebnis als XML Dokument dargestellt. Dieses enthält einen `<result>` Knoten der eine beliebige Anzahl (einen Pro Datensatz) von `<Result>` Knoten enthalten kann. Pro `<Result>` Knoten werden die Einzelinformationen in den entsprechenden Subknoten dargestellt.

6.8. Zusammenfassung

Im vorliegenden Kapitel wurde gezeigt wie anhand des im vorigen Kapitel vorgestellten *SOA Approach* als Vorgehensmodell ein Mashup analysiert und umgesetzt werden kann.

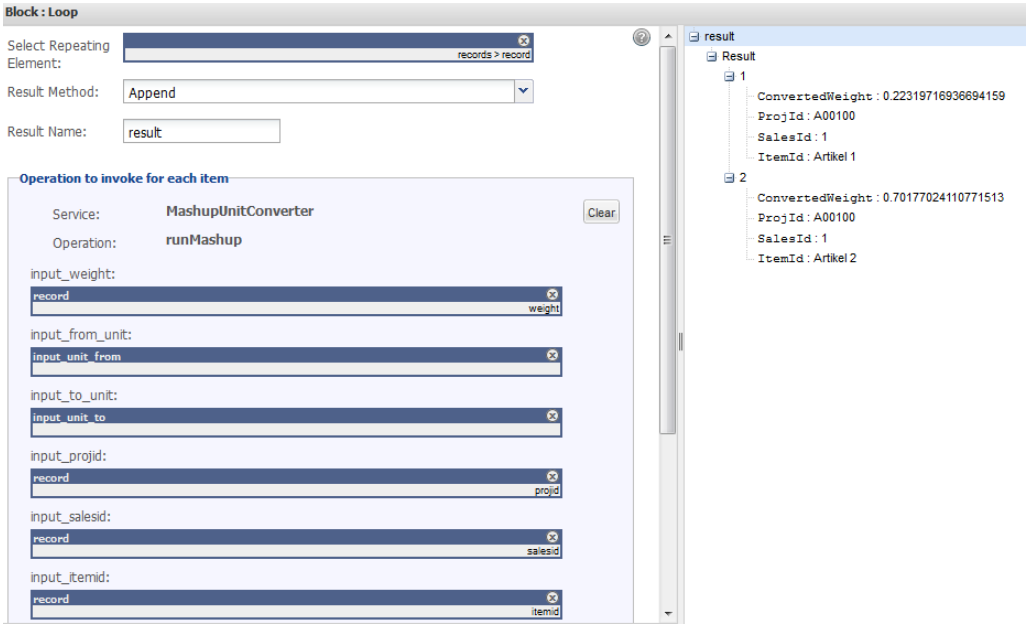

Abbildung 6.16.: Ergebnis Block *Loop*

Die Anlyse wird Schritt für Schritt anhand des Modells durchgeführt. Mit dieser Vorgehensweise kann sehr rasch und risikofrei die Implementierung gestartet werden. Diese gestaltet sich nach Auswahl eines geeigneten Frameworks nach einer Einarbeitungszeit unkompliziert. Das Mashup kann in Einzelkomponenten zerlegt, aufgebaut und getestet werden. Besonders die Ausführung einzelner Blöcke und Stränge ermöglichen eine schnelle und einfache ad hoc Handhabung. Mit verschiedenen Blöcken zur Transformation von Daten in andere Formate können Ergebnisse für eine optimale Darstellung aufbereitet werden. Für die meisten Aufgabenstellung bietet sich die Verwendung von XML zur Darstellung des Endergebnisses an. Das Ergebnis kann entweder mit Hilfe des Frameworks für die Visualisierung im Web vorbereitet oder maschinell weiterverarbeitet werden. Die Weiterverarbeitung wird ermöglicht indem das Framework für jedes erstellte Mashup eine Aufrufmöglichkeit via Web Service bereitstellt.

Listing 6.2: Ergebnis Block *Loop* als XML

```
<?xml version="1.0">
<result>
  <Result>
    <ConvertedWeight>0.22319716936694159</ConvertedWeight>
    <ProjId>A00100</ProjId>
    <SalesId>1</SalesId>
    <ItemId>Artikel 1</ItemId>
  </Result>
  <Result>
    <ConvertedWeight>0.70177024110771513</ConvertedWeight>
    <ProjId>A00100</ProjId>
    <SalesId>1</SalesId>
    <ItemId>Artikel 2</ItemId>
  </Result>
</result>
```

7. Leitfaden zur Erstellung eines Enterprise Mashups durch den Endanwender

Dieses Kapitel stellt einen Leitfaden für die konkrete Implementierung eines Mashups unter Einbindung eines Web Services dar. Dabei wird auf die beiden vorigen Kapitel, in denen die Vorgehensweise anhand des gewählten Modelles erörtert wird, aufgebaut. Die gezeigte Umsetzung im vorliegenden Kapitel bindet ein Web Service zur *Währungskonvertierung* ein. Im vorliegenden Leitfaden wird ein anderes Web Service als für die Umsetzung des Prototyps verwendet. Die soll zeigen, dass die Erstellung eines neuen Mashups mit einem Web Service durch den Endanwender in einfachen Schritten möglich ist.

Aufgabenstellung Es wird ein Enterprise Mashup implementiert, das Daten aus dem ERP System Microsoft Dynamics AX extrahiert. Das ERP System verwendet als Datenbank den Microsoft SQL Server. Bei den extrahierten Daten handelt es sich um Auftragspositionen (Artikel) eines Kundenprojektes. Die Selektion des Projektes und damit der zugehörigen Positionen erfolgt über die Projektnummer. Die Projektnummer kann vom Anwender vor der Ausführung des Mashups eingegeben werden. Für die gefundenen Auftragspositionen wird eine Umrechnung von der erfassten Währung des Positionspreises in eine Zielwährung durchgeführt. Die Zielwährung kann als zweiter Parameter vor der Ausführung des Mashups eingegeben werden. Die Umrechnung der Währung erfolgt mit einem frei verfügbaren Web Service.

Ablauf zur Umsetzung Die Implementierung eines Mashups gliedert sich in die folgenden Bereiche:

- *Vorbereitungen*: Bevor ein Mashup umgesetzt werden kann, müssen die unter 7.1 beschriebenen Punkte durchgeführt werden.
- *Implementierung*: Wie unter 7.2 beschrieben kann durch den Endanwender ein Mashup erstellt werden.
- *Ausführung*: Für die reine Ausführung eines Mashups sind die unter 7.3 beschriebenen Schritte notwendig.

7.1. Vorbereitungen

Bevor die Implementierung des Mashups begonnen werden kann, werden auf den eingesetzten Systemen vorbereitende Schritte durchgeführt. Diese Punkte müssen nur einmalig vor der Inbetriebnahme des ersten Mashups durchgeführt werden.

7.1.1. Datenbank ERP System

Die Einbindung der Daten aus dem ERP System im Mashup erfolgt durch direkten Zugriff auf die SQL Datenbank, Details siehe 5.7. Dazu wird am SQL Server ein neuer Benutzer mit reinem Lesezugriff vom Typ *SQL Server-Authentifizierung* angelegt. Alternativ können auch die Benutzer aus dem *Active Directory* für den Zugriff auf den SQL Server konfiguriert werden (Damit am Microsoft SQL Server Konten mit SQL Server-Authentifizierung angelegt werden können, muss die entsprechende Eigenschaft in den Servereinstellungen aktiviert werden). Dem Benutzer muss Zugriff auf die Datenbank des ERP Systems (im Beispiel Datenbank `DynamicsAx1` und Benutzerkonto `mashup`) sowie die Rollen zugewiesen werden.

Nach dem Erstellen des Benutzerkontos hat der Benutzer uneingeschränkten Lesezugriff auf alle Tabellen in der Datenbank. Um eine Einschränkung auf ausgewählte Tabellen zu treffen, können dem Benutzer die Rechte für die betroffenen Tabellen entzogen werden.

Datensicherheit. Im ERP System eines Unternehmens sind je nach eingesetzten Modulen mehr oder weniger unternehmerisch sensible Daten enthalten, z.B. Buchhaltung oder Auftragsdaten. Je nach Wichtigkeit dürfen Daten nicht oder nur von bestimmten Benutzern eingesetzt werden. Zur Einhaltung dieser Anforderung können auf der Datenbank mehrere Benutzer für lesenden Zugriff angelegt werden. Jedem Benutzer werden die erlaubten Tabellen zugewiesen und die Benutzerkonten mit einem Passwort versehen.
Technisch besteht auch die Möglichkeit mit einem Mashup Framework einen schreibenden Zugriff auf die ERP Datenbank durchzuführen. Z.B. könnten Berechnungsergebnisse wieder in die Datenbank zurück geschrieben werden. Ist ein schreibender Zugriff notwendig, muss dafür ein eigener Benutzer angelegt werden, der auf ausgewählte Tabellen oder nur auf ausgewählte Spalten einen schreibenden Zugriff erhält. Grundsätzlich sollte diese Möglichkeit sehr überlegt eingesetzt werden. Durch die Ausführung eines fehlerhaften Mashups könnten Daten unwiderruflich verändert werden.

7.1.2. Mashup Framework

Zum Zeitpunkt der Implementierung (01.2012) ist die Version *JackBe Presto 3.2* des Mashup Frameworks aktuell. Für die Inbetriebnahme des Frameworks ist eine *Java Runtime 1.6* oder höher notwendig. Für die *Starter Edition* von JackBe Presto kann eine kostenlose Lizenz online beantragt werden. Damit verbundene Einschränkungen ist, dass es sich um einen nichtkommerziellen Einsatz handelt.
Die Installation des Frameworks ist im mitgelieferten Tutorial beschrieben. Für einen Einsatz in einer Testumgebung ist nur die Ausführung eines Setup-Skriptes notwendig. Als interne Datenbank im Framework wird eine *HSQL* Datenbank eingesetzt. Im Rahmen eines Produktiveinsatzes sollte die interne HSQL Datenbank gegen einen Datenbankdienst ausgetauscht werden. In dieser Datenbank werden die Konfiguration sowie die Source Codes aller erstellten Artefakte gespeichert. Eine regelmäßige **Sicherung** dieser Datenbank ist daher notwendig. Die Bedienung des Produktes erfolgt entweder über das Web Interface oder über ein Eclipse Plugin. Für die Umsetzung des Prototyps wurde das Web Interface gewählt.

Nach erfolgreicher Installation kann das Framework in zwei Schritten gestartet werden:

- Starten der internen Datenbank (entfällt falls eine externe Datenbank eingesetzt wird).
- Starten des Web Servers, der die Basis des Frameworks darstellt (*Tomcat*).

Benutzerveraltung. In JackBe Presto ist eine Benutzer- und Gruppenverwaltung inkludiert, die standardmäßig in der internen HSQL Datenbank gespeichert wird. Jedem Benutzer können eine oder mehrere Gruppen zugewiesen werden über die der Benutzer die entsprechenden Rechte erhält. Für jedes erstellte Artefakt kann festgelegt werden, welche Benutzergruppe oder welcher Benutzer es verwenden darf.
Als Alternative Möglichkeit zur internen Benutzerverwaltung kann eine Anbindung zu einem LDAP oder Active Directory konfiguriert werden. Ist diees Einstellung aktiv erfolgt die Authentifizierung der Benutzer über den hinterlegten Server.

Datenbanktreiber und Datenquelle. Für alle Datenbanken, die in Mashups eingebunden werden, müssen die Treiber am Framework verfügbar sein. Bei den Treibern handelt es sich um *JDBC* Treiber. Diese werden üblicherweise vom Datenbankherstellt bereitgestellt. Die `JAR` Files der Treiber können über die Adminkonsole eingespielt werden und eine Anbindung zur ERP Datenbank eingerichtet werden, siehe Abbildung 7.1. Notwendige Angaben sind die Treiberklasse, die URL des Datenbankdienstes in JDBC Schreibweise, der Username und das Passwort des Kontos am SQL Server. Ist die Datenquelle eingerichtet, kann in den erstellten Mashups auf die Tabellen der Datenbank zugegriffen werden.

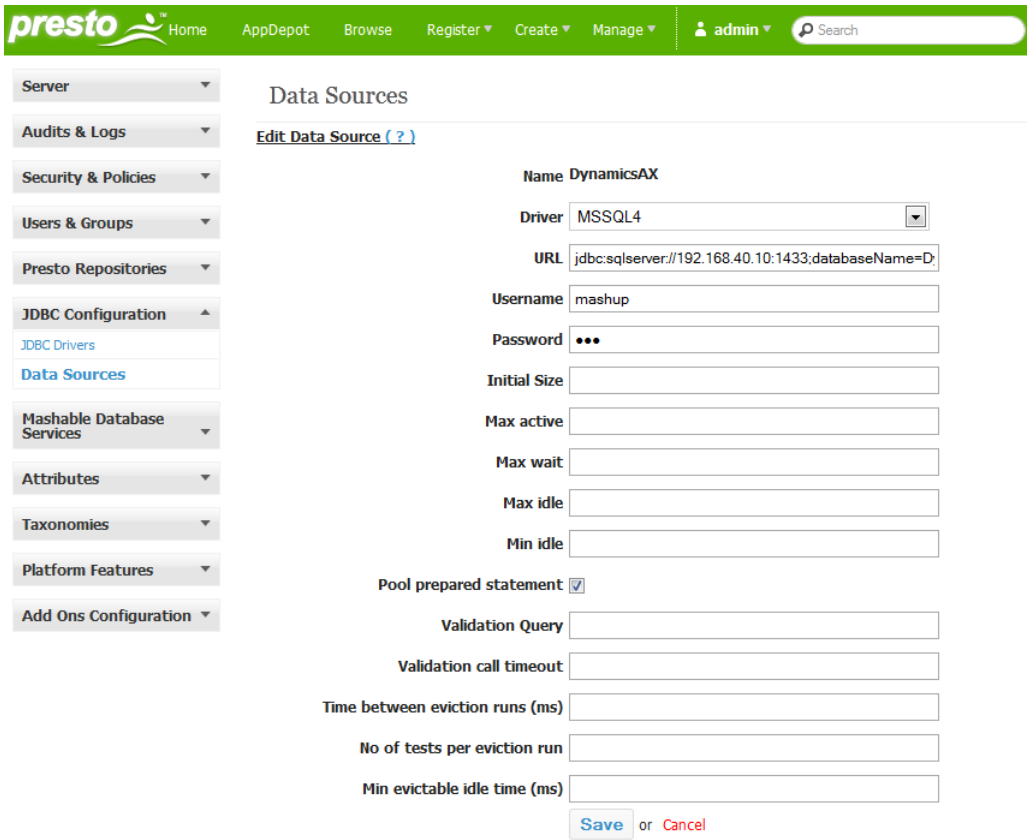

Abbildung 7.1.: Datenquelle einrichten

Einbindung von weiterern Systemen bzw. Legacy Systemen. Der Zugriff auf weitere Systeme kann durch Einbindung der entsrpechenden Datenbanktreiber sowie die Bereitstellung eines Zugriffs auf deren SQL Datenbank bewerkstelligt werden. Im Mashup Framework muss durch einen Administrator in der Management-Console der notwendige Treiber als `JAR` Archiv ins System integriert werden und eine Datenquelle konfiguriert werden. Im Mashup können Daten aus beliebigen Datenbanken miteinander kombiniert werden.
Damit ist die zu Beginn der Arbeit gestellte Anforderung der beliebigen Einbindung von Legacy Systemen erfüllt. Sofern JDBC Treiber für die Datenbank des Systems verfügbar sind und es sich um eine relationale Datenbank handelt, kann sie in Mashups integriert werden.
Alternativ zur Anbindung über direkten SQL Zugriff auf die Datenbank können Leagacy Systeme mit Hilfe von Web Services eingebunden werden. Eine Beschreibung aller Möglichkeiten ist unter 5.7 zu finden.

Logging. Das Logging wird über das im Framework integrierte Logging Tool durchgeführt. Alle Vorgänge im Framework werden in der internen Datenbank protokolliert. Dabei werden allgemeine Informationen wie eine Authentifizierung über das Web Interface sowie spezifische Informationen zur Ausführung einzelner Mashups gespeichert. Die Protokolle können über die Management-Console gefiltert angezeigt werden. Die Konfiguration erlaubt eine ähnliche Einstufung der Loglevel vergleichbar mit *Syslog*, es sind die Level *DEBUG, INFO, WARN, ERROR und FATAL* verfügbar.

7.2. Implementierung

Nachdem die beschriebenen einmaligen Vorbereitungen erledigt sind, können durch die Endanwender Mashups erstellt und ausgeführt werden. Für die Erstellung ist ein aktives Benutzerkonto am Framework wie im vorigen Abschnitt beschrieben erforderlich. Der Zugriff auf das Framework erfolgt im Leitfaden über das Web Interface. Alternativ kann für erfahrene Anwender auch ein Zugriff über ein Eclipse Plugin angeboten werden. Ausgangspunkt für alle Aktivitäten ist die Überblicksseite oder *Presto Hub*:

- Erstellen neuer Mashups mit *Open Wires*.
- Erstellen neuer Applikationen aus Mashups mit dem *Open App Editor*.
- Anzeigen aller verfügbaren Artefakte über die *Browse Funktion*.
- Verzweigen in die *Management-Console*.

Mashup Architektur. In den folgenden Abschnitten werden die Einzelschritte zur Erstellung der am Kapitelbeginn beschriebenen Aufgabenstellung erklärt. Es wird ein in der Struktur ähnliches Mashup wie im Kapitel 6 verwendet. Geändert wird die Datenbasis und das Web Service.

Web Service. Im Mashup wird ein freies Web Service zur Konvertierung zwischen Währungen eingesetzt: `http://www.webservicex.net/CurrencyConvertor.asmx`. Das Web Service erhält beim Aufruf zwei String Parameter mit dem die Ausgangswährung und die Zielwährung festgelegt werden. Der Rückgabewert ist eine Fließkommazahl mit dem aktuellen Umrechnungsfaktor zwischen den beiden Währungen. Listing 7.1 zeigt den Prototyp des Web Services.

Listing 7.1: Prototyp Web Service Währungskonvertierung

```
Float conversionRate(String fromCurrency, String toCurrency);
```

Für die Einbindung des Web Services in JackBe Presto muss ein neues WSDL Web Service durch Angabe der URL zum WSDL Dokument registriert werden, siehe Abbildung 7.2. Das Framework ermittelt daraus den Aufruf mit den erforderlichen Parametern. Wird das Web Service in späterer Folge in ein Mashup integriert, können die Inputparameter entweder vom Benutzer direkt eingegeben werden, oder von vorgelagerten Komponenten im Mashup befüllt werden (z.B. Werte aus eine Tabelle).

Im Leitfaden wird für die Konvertierung der Währungen ein frei verfügbares Web Service eingesetzt. Werden Dienste mit anderen Funktionen benötigt, können Quellen wie unter 6.7 beschrieben zum Suchen des passenden Dienstes herangezogen werden. Die im Mashup verwendeten Web Services können auch unternehmensintern implementiert sein. Wurden im Rahmen eines SOA Projektes im Unternehmen bereits verschiedene Anforderungen als Dienst umgesetzt, können diese in das Mashup integriert werden. Voraussetzung in beiden Fällen ist die Verfügbarkeit eines aktuellen WSDL Dokuments.

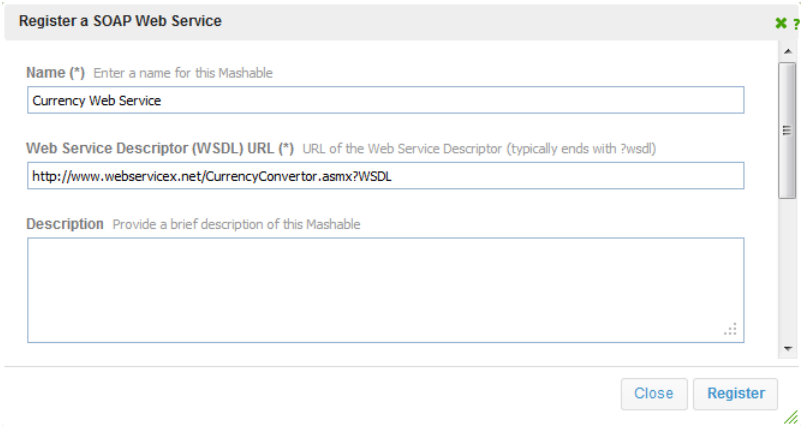

Abbildung 7.2.: Registrieren eines Web Services

Datenquellen Der Zugriff auf die SQL-Datenbank des ERP Systems (sowie auf weitere Datenbanken von anderen Systemen) kann über folgende Möglichkeiten erfolgen:

- Im JackBe Presto Hub kann eine neue Datenquelle erzeugt werden, die als Datenbasis eine der zuvor registrierten Datenbanken verwendet, siehe Abbildung 7.3. Beim Erzeugen der Datenquelle kann von der gewählten Datenbank jede verfügbare Tabelle, View oder Stored Procedure eingebunden werden. Die so erzeugten Datenquellen scheinen in der Toolbox in *Open Wires* beim Erstellen von Mashups als eigene Elemente auf. Diese Datenquellen können nur in der hinterlegten Konfiguration in Mashups eingesetzt werden. Eine Änderung der Selektion oder Projektion der SQL Daten ist nicht möglich. Wird eine von der konfigurierten Datenquelle abweichende Datenauswahl benötigt, muss dafür eine weitere eigene Datenquelle erstellt werden.

- Im Mashup Editor können direkt aus den im Framework hinterlegten Datenbanken Tabellen integriert werden. Die wie im vorigen Punkt beschriebene Hinterlegung der Datenquellen in der Management-Console ist dafür nicht notwendig. Mit Hilfe von SQL Statements können Tabellen und Views sowie Joins als Datenquelle für Mashups dienen, siehe Abbildung 7.4. Die Selektionen und Projektionen

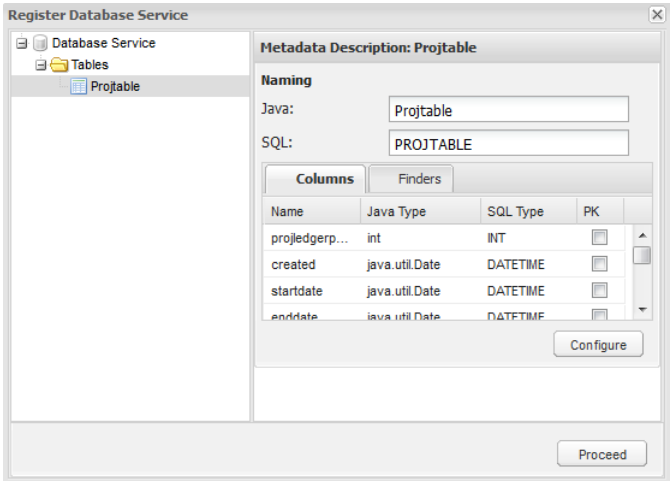

Abbildung 7.3.: Registrieren einer Datenquelle

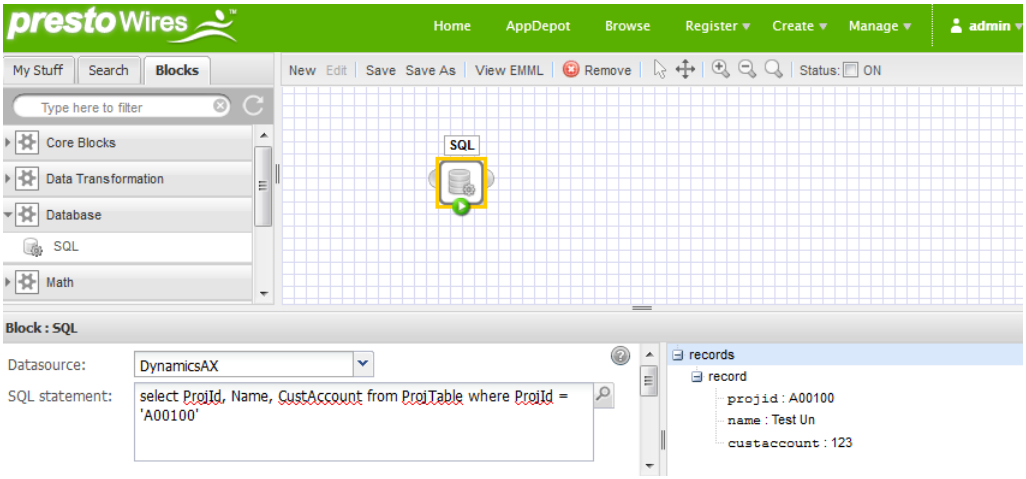

Abbildung 7.4.: Benutzerdefinierte Datenquelle

werden in einem Eingabefeld im Editor hinterlegt und können im Rahmen der Mashup Erstellung getestet und angepasst werden.

Mashuperstellung Die Erstellung des Mashups wird über das Tool *Open Wires* durchgeführt. Um die Implementierung und das Testen des Mashups zu vereinfachen wird es in zwei Teilbereiche gegliedert. Zudem erhöht die Aufteilung in mehrere Einzelblöcke mit spezifischen Funktionen die Wiederverwendbarkeit der Einzelteile:

Teil 1 Währungskonvertierung, siehe Abbildung 7.5: Für die Ermittlung des aktuellen Umrechnungskurses wird wie in diesem Kapitel beschreiben ein frei verfügbares Web Service aus dem Internet eingesetzt. Es erhält als Parameter die beiden Währungen: die Ausgangswährung *input_currency_from* und die Zielwährung *input_currency_to*. Der Aufruf des Web Services ermittelt den Umrechnungskurs und stellt gleichzeitig den Output des Mashups dar. Der Teil 1 wird später im Teil 2 als Subkomponente integriert. Die Inputs vom Teil 1 werden mit Daten aus dem Teil 2 belegt. Der Teil 1 wird vom Teil 2 für jeden Datensatz aufgerufen und berechnet jeweils aus der im Datensatz hinterlegten Währung den Umrechnungskurs auf die vom Benutzer eingegebene Zielwährung.

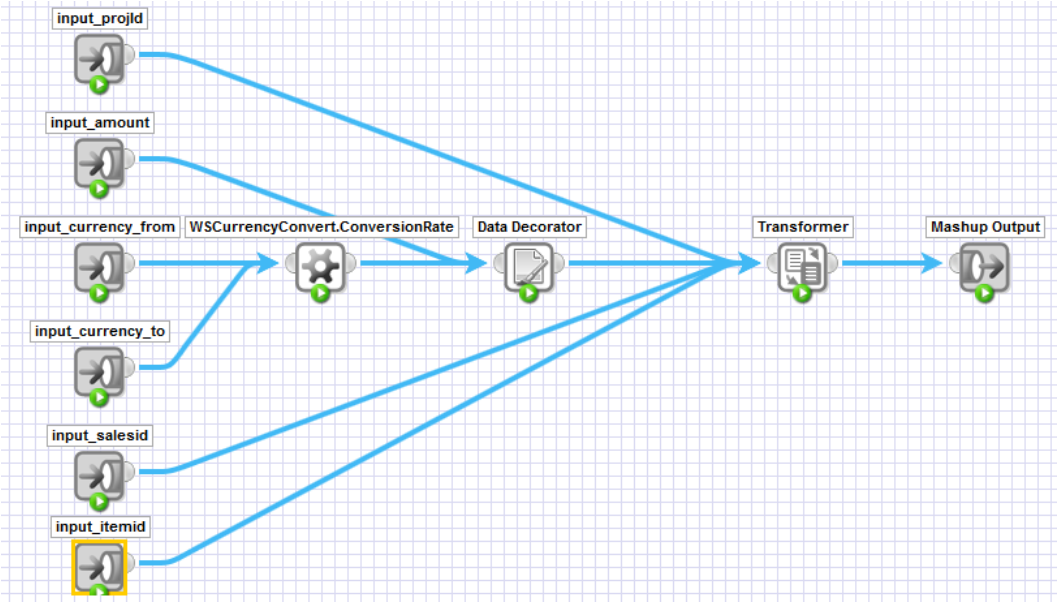

Abbildung 7.5.: Teil 1 Währungskonvertierung

Abbildung 7.6.: Ergebnis Block *Web Service Währungskonvertierung*

Das Ergebnis aus dem Web Service (der Umrechnungskurs der Währungen) zusammen mit dem Inputparameter *input_amount* (der Betrag der Position aus dem ERP System) wird mit einem *Data Decorator* Block multipliziert. Das Ergebnis der Multiplikation zusammen mit den weiteren Inputparametern *input_projid*, *input_salesid* und *input_itemid* werden durch einen *Transformer* Block geschickt. Dieser führt die Konvertierung der vier Inputs in ein neues XML Dokument durch. Das Ergebnis aus dem *Tranformer* stellt den Output dieses Mashups dar.

Für die Erstellung werden die notwendigen Elemente im *Open Wires* aus der Toolbox auf die Arbeitsfläche gezogen und miteinander verbunden. Je nach Blocktyp muss am Block eine entsprechende Konfiguration der Funktion vorgenommen werden. Im konkreten Beispiel muss auf den beiden Inputs ein Name festgelegt und am Web Service Block die beiden Inputs als Aufrufparameter eingestellt werden. Jeder Block kann durch Klick auf den grünen Pfeil für sich getestet werden. Das Mashup wird unter dem Namen *MashupMultiply* gespeichert.

Ergebnisse Teil 1: Das erste Zwischenergebnis ist die Rückgabe des XML Dokuments aus dem Aufruf des Web Service für die Ermittlung des Umrechnungskurses zwischen den Währungen, siehe Abbildung 7.6. Der ermittelte Wert wird zusammen mit dem Positionsbetrag des Artikels in den *Data Decorater* geschickt. Dieser ermittelt die Multiplikation der beiden Werte, siehe Abbildung 7.7. In einem weiteren Schritt werden die restlichen Inputparameter zusammen mit dem Multiplikationsergebnis über einen *Transformer* in ein neues XML Dokument konvertiert, siehe Abbildung 7.8. Die-

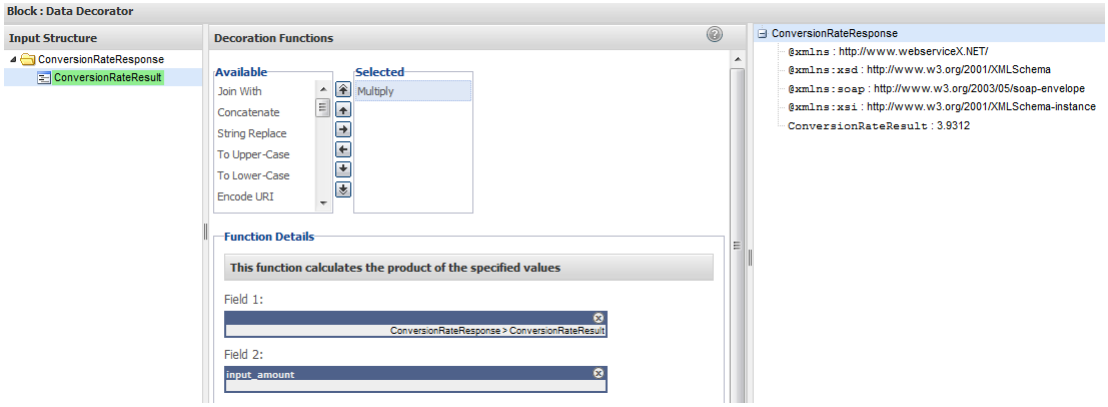

Abbildung 7.7.: Ergebnis Block *Data Decorator*

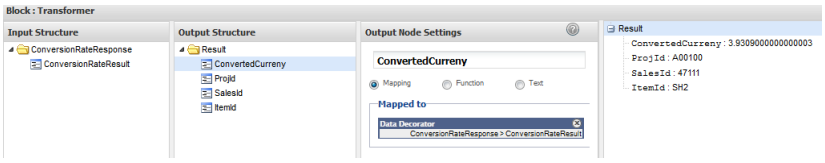

Abbildung 7.8.: Ergebnis Block *Transformer*

ses stellt zugleich auch das Endergebnis des Teiles 1 dar.

Teil 2 Daten laden und Output erzeugen, siehe Abbildung 7.9: Der Teil 2 stellt die Hauptkomponente dar, der alle Teile zusammenfügt, die Daten von der Datenbank lädt, mit den Benutzereingaben kombiniert und das Gesamtergebnis bereitstellt. Der Ladevorgang aus der SQL Datenbank wird mit Hilfe einer zuvor registrierten Datenquelle für die Tabelle *SalesLine* unter Angabe einer Projektnummer durchgeführt. Das Ergebnis sind alle zum Projekt gehörenden Auftragspositionen. Aus den geladenen Daten werden zur Vereinfachung der weiteren Schritte mit einem *Select* Block nur die benötigten Spalten selektiert und als Input für den folgenden *Loop* Block zur Verfügung gestellt (Der Select Block kann weggelassen werden). Im Select Block werden folgende Spalten ausgewählt:

- ProjId: Projektnummer
- SalesId: Auftragsnummer
- ItemId: Artikelnummer
- LineAmount: Positionspreis in der Ausgangswährung
- CurrencyCode: Ausgangswährung der Position

Die selektierten Daten aus der Datenbank in Verbindung mit dem Parameter für die Zielwährung, der vom Benutzer eingegeben wird, stellen die Inputs für den Loop Block dar. Dieser führt eine Iteration über jeden Datensatz durch, wobei bei jedem Iterationsschritt die im Block festgelegte Aktion aufgerufen wird. Als Vorgang im Block kann jedes am Framework verfügbare Mashup eingefügt werden, im konkreten Fall wird hier der Teil 1 ausgeführt. Die Inputparamter des Loop Blockes werden als Input für den Teil 1 verwendet (die Verbindung wird in der Blockkonfiguration durchgeführt). Ergebnis des Teiles 2 ist eine Liste mit den konvertierten Preisen der Auftragspositionen und stellt gleichzeitig das Endergebnis des Mashups dar. Das Mashup wird unter dem Namen *MashupCurrencySalesLine* gespeichert.

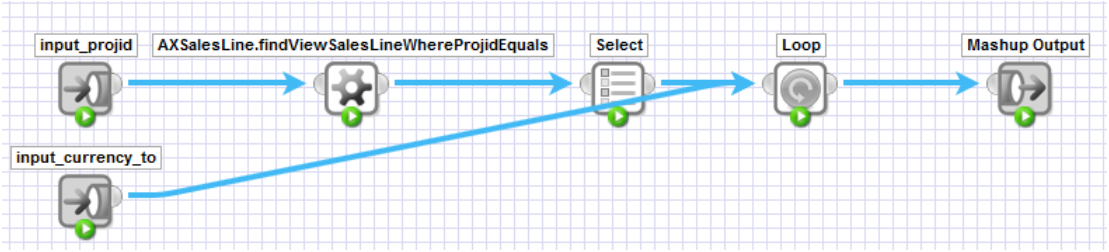

Abbildung 7.9.: Teil 2 Daten laden und Output erzeugen

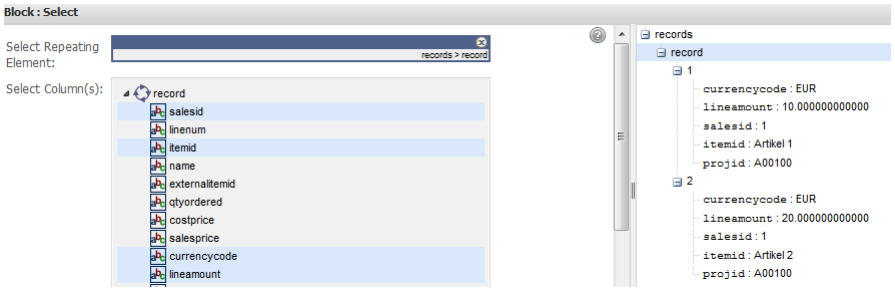

Abbildung 7.10.: Ergebnis Block *Select*

Ergebnisse Teil 2: Wichtige Zwischenergebnisse aus dem Teil 2 sind die Werte aus dem *Select*- und *Loop* Block. Aus dem ersten werden Spalten aus den von der Datenbank geladenen Daten eingschränkt, siehe Abbildung 7.10. Das Ergebnis des folgenden *Loop* Blockes ist zugleich auch das Endergebnis des Teiles 2. Es handelt sich dabei um ein XML Dokument, welches pro Datensatz einen `Result`-Knoten mit den berechneten Werten enthält, siehe Abbildung 7.11. Dieses XML Dokument wird in den folgenden Schritten mit einer Applikation am Framework in einen formatierten Output umgewandelt.

7.3. Ausführung

Das im vorigen Kapitel 7.2 implementierte Mashup kann von jedem Benutzer mit einem Account ausgeführt werden. Für die Ausführung stehen mehrere Möglichkeiten bereit, die im folgenden beschrieben werden.

Ausführung im Webbrowser. Im Regelfall wird das Mashup interaktiv mit Hilfe eines Webbrowsers am Framework ausgeführt. Die notwendigen Parameter werden dabei vom Benutzer in Eingabefeldern hinterlegt, das Ergebnis wird als HTML gerendert im Browser dargestellt, siehe Abbildung 7.12. Vor der Ausführung kann das Mashup in ein *App* integriert werden. Apps sind Browseranwendungen, die ein oder mehrere Mashups ausführen und das Ergebnis als HTML darstellen. Die Ausführung ist grundsätzlich auch ohne App möglich. Eine App stellt eine Ausführungsumgebung mit der Abfrage der Eingabeparamter als Dialog und formatierte Ausgabe der Daten als HTML dar.

Der Vorteil einer App besteht in der visuellen übersichtlichen Aufbereitung der Daten als auch der Eingabe der Parameter. Diese werden über einen Dialog wie in der Abbildung zu sehen vom Benutzer ermittelt. Das Ergebnis kann durch Auswahl unterschiedlicher Views (z.B. Grid oder Tree) je nach Bedarf gestaltet werden.

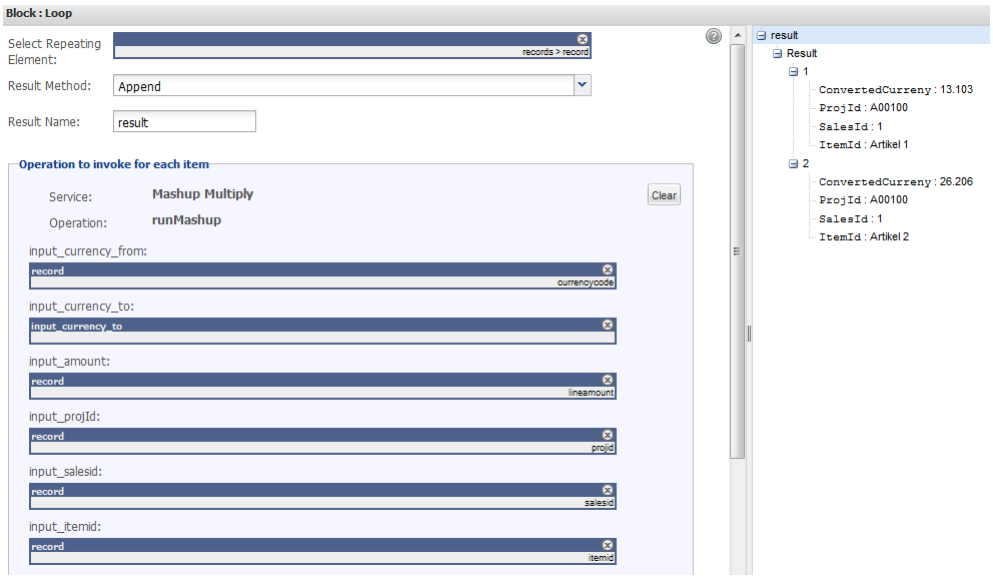

Abbildung 7.11.: Ergebnis Block *Loop*

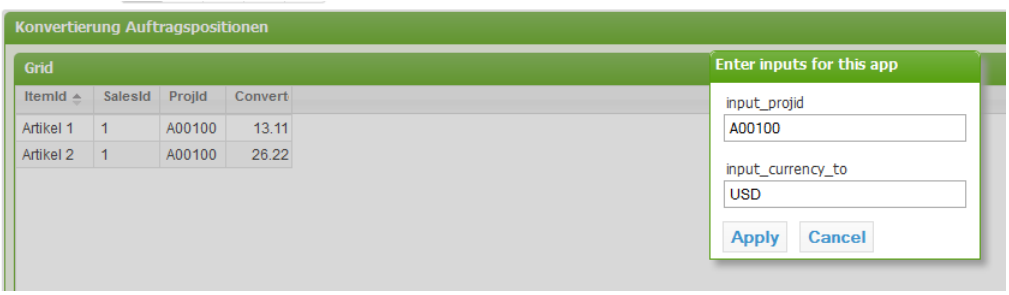

Abbildung 7.12.: Mashupausführung im Webbrowser

Ausführung als Web Service. Um das Mashups in andere Mashups oder Anwendungen integrieren zu können, kann es als Web Service aufgerufen werden. Das zugehörige WSDL Dokument kann ebenfalls erzeugt werden. Dazu stellt das Framework für jedes erstellte Mashup eine URL zum Aufruf als Web Service bereit. Auch die Erzeugung des WSDL Dokuments wird unterstützt. Eine maschinelle Verarbeitung der Daten in anderen Anwendungen oder Mashups kann somit relativ einfach bewerkstelligt werden.

7.4. Erstellung eines Enterprise Mashups - Evaluierung

Im Rahmen der Arbeit wurde mit einer Testgruppe die praktische Erstellung des im Leitfaden beschriebenen Mashups durchgeführt. Die erlangten Erfahrungen und Eindrücke der Teilnehmer wurden mit einem Fragebogen ausgewertet (Fragebogen siehe Anhang A). Die Evaluierung wurde mit einer Laborgruppe des 5. Jahrganges der Fachrichtung Netzwerktechnik der IT-HTL Ybbs an der Donau durchgeführt.

Rahmenbedingungen und Ablauf. Ziel der praktischen Evaluierung war die Ermittlung der Verständlichkeit des erstellten Leitfadens und ob ein minimales Vorwissen für die ad hoc Entwicklung eines Enterprise Mashups ausreichend ist. Diese Auswer-

tung soll als Hilfestellung für Unternehmen dienen, die in Betracht ziehen Mashups einzusetzen.
Die Testgruppe bestand aus zehn Schülern, die eine kurze Einführung und eine Demonstration des fertigen Mashups erhielten. Aufgabenstellung war mit dem Leitfaden das beschriebene Mashups umzusetzen. Der vorgegebene Zeitrahmen wurde mit 80 Minuten reiner Arbeitszeit, 10 Minuten Einführung und Erklärung der Aufgabenstellung und 10 Minuten für das abschließende Ausfüllen des Fragebogens bemessen. Der Fragebogen besteht aus 16 Fragen, die jeweils mit einer Skala von 1 bis 5 beantwortet werden können (nur die diskreten Werte 1, 2, 3, 4 und 5 sind durch ankreuzen auswählbar). Die Werte 1 bis 5 haben folgende Bedeutung:

- 1: Sehr Gut
- 2: Gut
- 3: Ausreichend
- 4: Wenig
- 5: Zu Wenig

Erwähnt werden muss, dass die Testgruppe über fundiertes technisches Vorwissen im IT Bereich sowie einschlägige Programmierkenntnisse in mehreren Sprachen verfügt. Die Gruppe hat relativ wenig Wissen über organisatorische Prozesse und Abläufe in Unternehmen und verfügt abgesehen von Ferialpraktika über keine Berufserfahrung.

Auswertung. Die Auswertung des Fragebogens mit der Testgruppe ergibt das Ergebnis wie in Abbildung 7.13 dargestellt. Die Tabelle 7.1 zeigt die Ergebnisse im Detail, bei jeder Frage haben alle Teilnehmer eine Antwort ausgewählt (n = 10 für alle Fragen).

Interpretation. Aus den erhaltenen Antworten geht hervor, dass der Leitfaden zur Lösung der Aufgabenstellung grundsätzlich ausreichend ist (Frage 1 und 2). Im Rahmen der Evaluierung mit den Teilnehmern hat sich trotzdem noch Verbesserungspotential gezeigt. Der Leitfaden könnte für eine bessere Nachvollziehbarkeit in eine Liste von Arbeitsschritten umgewandelt werden. Damit würde die Verständlichkeit noch besser werden.

Die Teilnehmer würden nach der Durchführung des Leitfadens andere Aufgabenstellungen wieder in Form eines Mashups ausführen (Frage 3). Einem Unternehmen würden sie den Einsatz eines Mashups für eine konkrete Anforderung prinzipiell empfehlen (Frage 4). Die Teilnehmer würden selbst als Mitarbeiter in einem Unternehmen ein Mashup einsetzen (Frage 5). Die Wiederverwendbarkeit des gesamten Mashups (Frage 6) sowie die Wiederverwendbarkeit dessen Einzelkomponenten (Frage 7) wurde als hoch eingestuft.

Die Bedienung des Frameworks (Frage 8) wurde relativ schlecht bewertet. Der Grund dafür könnte an den Fehlern des UIs liegen. Z.B. wird die Konfiguration der Blöcke nicht immer korrekt gespeichert und muss wiederholt werden. Die Performance des fertigen Mashups wird als gut bewertet (Frage 9). Das Einbinden eines Web Services (Frage 10) und einer Datenbank (Frage 11) ins Framework wird als sehr einfach eingestuft. Der Aufbau des kompletten Mashups wird als ausreichend gesehen (Frage 12). Hier muss

angemerkt werden, dass dies auch mit dem noch verbesserungwswürdigem Leitfaden zusammenhängen kann.

Die Kandidaten bewerten ihre eigenen Programmierkenntnisse mit gut (Frage 13) ebenfalls die Kenntnisse mit Bereich verteilte Systeme (Frage 14). Die Kenntnisse von organisatorischen Abläufen werden mit wenig eingestuft (Frage 15). Die Verständlichkeit des Fragebogens wird mit sehr gut eingestuft (Frage 16).

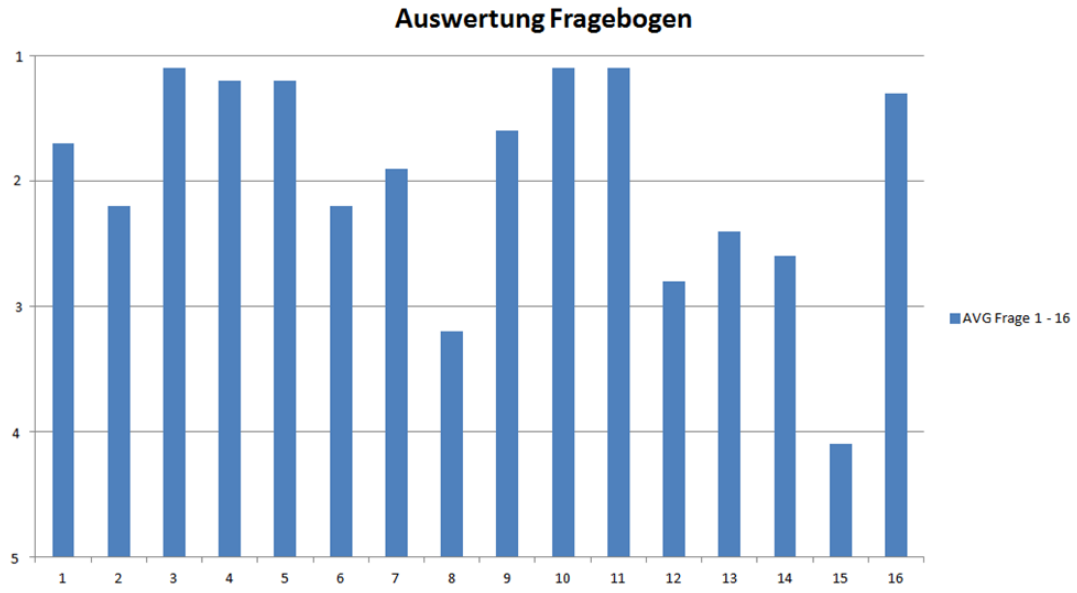

Abbildung 7.13.: Auswertung Fragebogen

7.5. Kontrolle der Zielsetzungen

Die am Beginn unter 1.2 beschriebenen Ziele sollen auf ihre Erreichung überprüft werden:

- Mashuperstellung: Die einfache und rasche Erstellung eines Mashups wird in den vorangehenden Abschnitten dieses Kapitels anhand des Leitfadens dargestellt. Ergänzend wurde mit einer Gruppe von Schülern der Leitfaden parktisch erprobt. Die erfolgreiche Erstellung ist in der Auswertung der durchgeführten Übung unter 7.4 erörtert.

- Sicherheit: Zur Sicherstellung, dass nicht alle Mitarbeiter alle Daten in Mashups verwenden können, werden wie unter 7.1 beschrieben, verschiedene Accounts für den Zugriff auf die zugrundeliegende Datenbank erstellt. Werden für den Zugriff nicht wie im Prototyp eigene Accounts auf der Datenbank angelegt, können auch die Accounts aus dem Verzeichnisdienst mit den entsprechenden Rechten auf der Datenbank versehen werden.
  Die Einschränkungen für den Aufruf der erstellen Mashups können mit Hilfe des Frameworks festgelegt werden. Dazu kann vom Ersteller eines Mashups festgelegt werden, welche Benutzer Rechte für das Ausführen bzw. Ändern erhalten sollen.
  Die Anforderung, dass die Daten das Unternehmen beim Ausführen eines Mashups nicht verlassen, wird umgesetzt, indem das Framework auf einem unternehmensinternen Server ausgeführt wird.

Frage	**AVG (n = 10)**
1. Konnten Sie mit Hilfe des Leitfadens die Aufgabenstellung lösen?	1.7
2. Ist die Struktur und Abfolge der Punkte im Leitfaden angemessen?	2.2
3. Würden Sie für andere Aufgabenstellung wieder ein Mashup einsetzen?	1.1
4. Würden Sie einem Unternehmen den Einsatz von Mashups empfehlen?	1.2
5. Würden Sie selbst in einem Unternehmen als Endanwender Mashups einsetzen?	1.2
6. Wie hoch bewerten Sie die Wiederverwendbarkeit des erstellten Mashups?	2.2
7. Wie hoch bewerten Sie die Wiederverwendbarkeit der Einzelteile des erstellten Mashups.	1.9
8. Wie gut haben Sie sich mit der Bedienung des Mashup Frameworks zurecht gefunden?	3.2
9. Wie bewerten Sie die Performance des ausgeführten Mashups?	1.6
10. Wie einfach (1) / schwierig (5) war die Einbindung eines externen Web Services im Framework?	1.1
11. Wie einfach (1) / schwierig (5) war die Einbindung von SQL Datenquellen im Mashups?	1.1
12. Wie einfach (1) / schwierig (5) war der Aufbau der Mashup Struktur mit den vorhandenen / erstellten Komponenten?	2.8
13. Wie bewerten Sie Ihre Programmierkenntnisse?	2.4
14. Wie bewerten Sie Ihre Kenntnisse im Bereich Web Services und verteilte Systeme?	2.6
15. Wie schätzen Sie Ihre Kenntnisse mit organisatorischen Abläufen in Unternehmen ein? (z.B. Einkaufsprozess, Verkaufsprozess, ...)	4.1
16. Ist der Fragebogen verständlich formuliert?	1.3

Tabelle 7.1.: Auswertung Fragebogen

- Einbindung von Unternehmensanwendungen: Die Erfüllung dieser Anforderung kann durch das Auswählen eines passenden Frameworks sichergestellt werden. Im Kapitel 4.5 wurden die wichtigsten am Markt befindlichen Produkte getestet und miteinander verglichen. Durch die gewonnenen Informationen wurde ein geeignetes Framework ausgewählt und unter 5.4 dessen Einsatz begründet.
  Die High End Produkt bieten jeweils mehrere Möglichkeiten für die Anbindung von Daten aus Unternehmenssystemen. Die Techniken sind unter 5.7 beschrieben. Für den Prototyp wurde die Anbindung mittels SQL Zugriff auf die Datenbank gewählt.

7.6. Kritische Würdigung

Das vorgestellte Produkt JackBe Presto ist ein vollwertiges für den Enterprise-Einsatz geeignetes Produkt. Der Produktumfang ist hinlänglich und deckt nahezu alle Aufgabenstellungen im Bereich Mashups ab. Die technische Basis ist auf dem Stand der Technik. Als besondere Ergänzung muss die Aufbereitung der erstellten Mashups für den Einsatz auf mobilen Geräten hervorgehoben werden (wurde nicht im Rahmen der Arbeit gezeigt). Sollten dennoch Anforderungen entstehen, die nicht mit dem Standardumfang des Produktes umgesetzt werden können, besteht die Möglichkeit eigene Plugins zu codieren und einzubinden.
Als Kritikpunkt muss allerdings die suboptimale Unterstützung für tabellarische Daten festgehalten werden. Das Produkt ist optimiert für die Verwendung von XML formatierten Daten. Bei Daten, die aus SQL Quellen kommen, könnten die bereitgestellten Funktionen umfangreicher sein. Um diese optimal verarbeiten zu können muss für die meisten Aufgabenstellungen eine Konvertierung auf XML vorgenommen werden. Im Rahmen von Mashups, die ohnehin im Web ausgeführt werden, stellt dies aber keine wichtige Einschränkung dar. Je nach Gesichtspunkt kann es auch als Vorteil betrachtet werden.

8. Zusammenfassung und Ausblick

Enterprise Mashups stellen in vielen Unternehmen ein noch großteils ungenutztes Potential dar. Mit Mashups können Mitarbeiter ad hoch selbst Daten kombinieren und neue Anwendungen erstellen. Je nach eingesetztem Produkt sind dabei mehr oder weniger technische Kenntnisse notwendig. Als zentraler Diskussionspunkt beim Einsatz von Mashups in Unternehmen muss die Frage der Datensicherheit im Mittelpunkt stehen. Wie in der vorliegenden Arbeit erörtert, dürfen sensible Unternehmensdaten keinesfalls das Unternehmen verlassen bzw. nicht von allen Mitarbeitern eingesehen werden. Im Rahmen der Arbeit und der Implementierung wurden Möglichkeiten und Techniken beschrieben diese essentielle Forderung erfüllen zu können.

Bei der Auswahl des Mashup Frameworks sollte wohl überlegt und strategisch vorgegangen werden. Nicht alle am Markt befindlichen Produkte sind für den Enterprise Bereich geeignet. Das kann wie in den entsprechenden Kapitel beschrieben vom Funktionsumfang der Software selbst abhängen als auch mit den bereits angesprochenen Sicherheitsanfoderungen zusammen hängen. Beim Funktionsumfang stehen besonders die Anbindung von typischen Businessprodukten wie ERP, PPS oder anderen Legacy Systeme im Vordergrund. Für die Anbindung gibt es meist mehrere Möglichkeiten. Einfach und effizient kann die Einbindung über direkten Zugriff auf die SQL Datenbank erfolgen.

Weitere wichtige Kriterien betreffen die Kosten des Produktes. Einerseits die Anschaffungskosten selbst, die zwischen den Produkten deutlich variieren. Andererseits alle Kosten, die für den laufenden Betrieb anfallen. Beim Thema Kosten müssen auch die Aufwände für die notwendige Server-Hardware herangezogen werden. Die in der Arbeit evaluierten Produkte können allerdings allesamt auch auf virtueller Hardware eingesetzt werden.

Die Bedienung des ausgewählten Produktes sollte für den Anwender möglichst ohne spezielles Vorwissen und ohne technische Kenntnisse möglich sein. Gute Produkte stellen dafür ein intuitiv bedienbares User Interface bereit, welches die Basis für die Implementierung darstellt. Die UIs sind bei den renommierten Produkten entsprechend weit entwickelt und lassen ein rasches und einfaches Arbeiten zu. Trotzdem kommt der Anwender bisweilen in die Verlegenheit Aufgabenstellungen umzusetzen, die einen tieferen Einblick in die Materie voraussetzen. Dafür bieten die Produkte eine umfassende Online Hilfe an. In manchen Situationen mag allerdings dies einem technisch nicht versierten Anwender nicht reichen und eine Rückfrage in der unternehmensinternen IT-Abteilung ein möglicher Ausweg sein. Ganz ohne technische Details, so wie es die Hersteller der Mashup Frameworks versprechen, geht es in der Praxis nicht.

Ausblick. Der Einsatz von Mashup in Unternehmen stellt wie in der Arbeit gezeigt einen Mehrwert dar. Mashups stellen die logische Weiterentwicklung von diversen Reportingtools dar, die bisher als Applikation mit eigenem Client lokal am Rechner aus-

geführt wurden. Hier liegt der entscheidende Vorteil von Mashups, da sie auf einem Server bzw. Framework verwaltet und abgelegt werden. Die Wiederverwendbarkeit für andere Abteilungen und Anwender wird dadurch erhöht. Wie sich der Markt und die Technik weiterentwickeln wird, bleibt momentan spannend. Aktuell entstehen eine Reihe von Mashup Frameworks und bestehende Produkte werden mit neuen Features ausgestattet. Konkret wird an Ergänzungen für die semantische Aufbereitung der Daten gearbeitet. Die semantischen Ergänzungen lassen sich optimal mit Produkten aus dem Bereich Semantic ERP kombinieren. Die Zukunft wird spannend...

A. Fragebogen

Fragebogen zur Erstellung eines Enterprise Mashups

Bewertung der Fragen: 1 - Sehr Gut, 2 - Gut, 3 - Ausreichend, 4 - Wenig, 5 - Zu Wenig
Notwendige Zeit zur Beantwortung: ca. 5 Minuten
Zweck: Evaluierung, ob mit minimalem Vorwissen die Erstellung von Mashups anhand eines Leitfadens möglich ist.
Verwertung der Ergebnisse: Der Fragebogen ist anonym, die Auswertung erfolgt im Rahmen der Diplomarbeit "Enterprise Mashups".

Leitfaden	1	2	3	4	5
1. Konnten Sie mit Hilfe des Leitfadens die Aufgabenstellung lösen?	☐	☐	☐	☐	☐
2. Ist die Struktur und Abfolge der Punkte im Leitfaden angemessen?	☐	☐	☐	☐	☐

Mashup	1	2	3	4	5
3. Würden Sie für andere Aufgabenstellung wieder ein Mashup einsetzen?	☐	☐	☐	☐	☐
4. Würden Sie einem Unternehmen den Einsatz von Mashups empfehlen?	☐	☐	☐	☐	☐
5. Würden Sie selbst in einem Unternehmen als Endanwender Mashups einsetzen?	☐	☐	☐	☐	☐
6. Wie hoch bewerten Sie die Wiederverwendbarkeit des erstellten Mashups?	☐	☐	☐	☐	☐
7. Wie hoch bewerten Sie die Wiederverwendbarkeit der Einzelteile des erstellten Mashups.	☐	☐	☐	☐	☐

Mashup Framework / User Interface	1	2	3	4	5
8. Wie gut haben Sie sich mit der Bedienung des Mashup Frameworks zurecht gefunden?	☐	☐	☐	☐	☐
9. Wie bewerten Sie die Performance des ausgeführten Mashups?	☐	☐	☐	☐	☐
10. Wie einfach (1) / schwierig (5) war die Einbindung eines externen Web Services im Framework?	☐	☐	☐	☐	☐
11. Wie einfach (1) / schwierig (5) war die Einbindung von SQL Datenquellen im Mashups?	☐	☐	☐	☐	☐
12. Wie einfach (1) / schwierig (5) war der Aufbau der Mashup Struktur mit den vorhandenen / erstellten Komponenten?	☐	☐	☐	☐	☐

Angaben zur Person	1	2	3	4	5
13. Wie bewerten Sie Ihre Programmierkenntnisse?	☐	☐	☐	☐	☐
14. Wie bewerten Sie Ihre Kenntnisse im Bereich Web Services und verteilte Systeme?	☐	☐	☐	☐	☐
15. Wie schätzen Sie Ihre Kenntnisse mit organisatorischen Abläufen in Unternehmen ein? (z.B. Einkaufsprozess, Verkaufsprozess, …)	☐	☐	☐	☐	☐

Sonstiges	1	2	3	4	5
16. Ist der Fragebogen verständlich formuliert?	☐	☐	☐	☐	☐

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Literaturverzeichnis

[1] Datenschutzgesetz 2000 - DSG 2000, 2000. https://www.dsk.gv.at/site/6200/default.aspx [Stand 02.2012].

[2] Brazell A. RSS vs. Atom, 2006. http://www.problogger.net/archives/2006/03/30/rss-vs-atom-whats-the-big-deal [Stand 09.2009].

[3] Wun A. *Encyclopedia of Database Systems*. Springer US, 2009.

[4] Open Mashup Alliance. OMA EMML Documentation, 2011. http://www.openmashup.org/omadocs/v1.0/index.html [Stand 03.2011].

[5] B4Bmedia. SAP ERP6.0 Releasewechsel. Technical report, B4Bmedia, 2008.

[6] Ben-Kiki O. Evans C. and döt Net I. YAML Ain't Markup Language (YAML) Version 1.2, 2009. http://www.yaml.org/spec/1.2/spec.html [Stand 02.2012].

[7] Fensel D. *A Silver Bullet for Knowledge Management and Electronic Commerce*. Springer, 2004.

[8] Krafzig D. Banke K. Slama D. *Enterprise SOA: Service Oriented Architecture Best Practices*. Prentice Hall International, 2004.

[9] Merrill D. Mashups: The new breed of Web App, 2006. http://www.ibm.com/developerworks/library/x-mashups.html [Stand 09.2009].

[10] Winer D. XML-RPC Specification, 1999. http://www.xmlrpc.com/spec [Stand 09.2009].

[11] Prud'hommeaux E. and Seaborne A. SPARQL Query Language for RDF, 2008. http://www.w3.org/TR/rdf-sparql-query/ [Stand 02.2012].

[12] Sowa J. (ed). *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann, 1991.

[13] Martin D. Burstein M. et al. Bringing Semantics to Web Services with OWL-S. *World Wide Web 10(3), 243-277*, 2005.

[14] Rosenberg F. Khalaf R. Duftler M. Curbera F. and Austel P. End-to-End Security for Enterprise Mashups. In *ICSOC-ServiceWave 2009*, pages 389–403. Springer-Verlag Berlin Heidelberg, 2009.

[15] GLiNTECH. Composite Applications with webMethods Portal: Helping IT to strategically align with Business, 2005.

[16] IBM. *Patterns: Service-Oriented Architecture and Web Services*. Endrei M. Ang J. Arsanjani A. Chua S. Comte P. Krogdahl P. Luo M. and Newling T., 2004.

[17] IBM. IBM Mashup Center, 2011. http://www-01.ibm.com/software/info/mashup-center/ [Stand 07.2011].

[18] Working Group Quality in Statistics. Standard Quality Indicators, 2005. http://epp.eurostat.ec.europa.eu [Stand 05.2011].

[19] Yahoo Inc. Yahoo Pipes, 2011. http://pipes.yahoo.com/pipes/docs [Stand 02.2012].

[20] Intel. Intel Mash Maker Documentation, 2007. http://software.intel.com/en-us/articles/intel-mash-maker-documentation/?wapkw=(mash+maker) [Stand 09.2011].

[21] Bieberstein B. Laird R. G. Dr. Keith J. and Mitra T. *Executing SOA: A Practical Guide for the Service-Oriented Architect.* IBM Press, 2008.

[22] Eppler M. J. *Managing Information Quality.* Springer, 2006.

[23] Gregorio J. and de hOra B. The Atom Publishing Protocol, RFC 5023. Technical report, IETF, 2007.

[24] Meiert J. RSS 2.0 und Atom 1.0 im Vergleich, 2006. http://meiert.com/de/publications/translations/intertwingly.net/rss-2.0-and-atom-1.0/ [Stand 09.2009].

[25] Metsker S. J. and Wake W. *Design Patterns in Java (Software Patterns).* Addison-Wesley Longman, 2006.

[26] Novak J. and Voigt B. J. J. Mashups: Strukturelle Eigenschaften und Herausforderungen von End-User Development im Web 2.0. *i-com 6(1)*, pages 19–24, 2007.

[27] JackBe. Presto Developer Documentation, 2011. http://www.jackbe.com/prestodocs/v2.7.0/prestolibrary/wwhelp/wwhimpl/js/html/wwhelp.htm [Stand 12.2011].

[28] Winter R. Jung R. *Data Warehousing 2000.* Physica-Verlag Heidelberg, 2000.

[29] Hemenway K. and Calishain T. *Spidering Hacks.* O'Reilly Media, 2003.

[30] Kruczynski K. and Kozanecki F. Ontologien halten Einzug in die IT-Praxis. *IS Report, 12(3)*, pages 50–54, 2008.

[31] Breitfeld M. Engler L. and Kern M. Vergleich von Server-side Mashup Systemen. Technical report, Universität Stuttgart, 2007.

[32] Feigenbaum L. W3C Semantic Web Frequently Asked Questions, 2009. http://www.w3.org/RDF/FAQ [Stand 02.2012].

[33] Hors A. L. and Jacobs I. HTML 4.01 Specification, 1999. http://www.w3.org/TR/html401/ [Stand 09.2009].

[34] Mozilla Labs. Ubiquity, 2011. https://mozillalabs.com/ubiquity [Stand 09.2011].

[35] Albinola M. Baresi L. Carcano M. and Guinea S. Mashlight: a Lightweight Mashup Framework for Everyone. 18th International World Wide Web Conference (WWW 2009), Deepse Group Dipartimento di Elettronica e Informazione, 2009.

[36] Biezunski M. Newcomb S. R. Rivers-Moore D. Ahmed K. Altheim M. and Hunting S. XTM: XML Topic Maps (XTM) 1.0, 2010. http://www.topicmaps.org/xtm [Stand 09.2009].

[37] Carl D. Clausen J. Hassler M. and Zund A. *Mashups Programmieren - Grundlagen, Konzepte, Beispiele.* O'Reilly, 2008.

[38] Gruninger M. and Lee J. Ontology – applications and design. *Comm.ACM 45(2)*, page 39–41, 2002.

[39] Heinrich B. Klier M. and Bewernik M. A. Unternehmensweite Anwendungsintegration. *Wirtschaftsinformatik 48(3)*, pages 158–168, 2010.

[40] Nottingham M. and Sayre R. The Atom Syndication Format, RFC 4287. Technical report, IETF, 2005.

[41] Roy M. Towards End-user Enabled Web Service Consumption for Mashups. In *ICSE '10 Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 2*, pages 413–416. ACM New York, NY, USA, 2010.

[42] Microsoft. Microsoft Axapta Developer's Guide, 2003.

[43] Microsoft. Microsoft Dynamics, 2009. http://www.microsoft.com/dynamics [Stand 10.2009].

[44] Microsoft. MSXML, 2009. http://msdn.microsoft.com/en-us/library/ms763742(VS.85).aspx [Stand 10.2009].

[45] Sun Microsystems. RPC: Remote Procedure Call Protocol specification, RFC 1050. Technical report, Sun Microsystems, 1988.

[46] Josuttis N. *SOA in der Praxis.* dpunkt.verlag, 2008.

[47] Mitra N. and Lafon Y. SOAP Version 1.2 Part 0: Primer (Second Edition), 2009. http://www.w3.org/TR/2007/REC-soap12-part0-20070427/ [Stand 09.2009].

[48] Thonse S. Nagaraj N. S. and Balasubramanian S. Business Process Management an Emerging Trend, 2001.

[49] Theseus Programmbüro. Theseus, 2010. http://theseus-programm.de [Stand 01.2012].

[50] Gruber T. R. A Translation Approach to portable Ontology Specifications. *Knowledge Acquisition 5(2), 199-200*, 1993.

[51] Haugen R. and McCarthy W. E. REA a semantic model for Internet supply chain collaboration. *Internet Business*, pages 1–16, 2006.

[52] RDF Working Group. Resource Description Framework, 2004. http://www.w3.org/RDF [Online Stand 09.2010].

[53] Ort E. Brydon S. and Basler M. Sun Developer Network (SDN): Mashup Styles, Part 1: Server-Side Mashups. Technical report, Sun Developer Network, 2007. http://java.sun.com/developer/technicalArticles/J2EE/mashup_1 [Stand 09.2009].

[54] Ort E. Brydon S. and Basler M. Sun Developer Network (SDN): Mashup Styles, Part 2: Client-Side Mashups in the Java EE Platform. Technical report, Sun Developer Network, 2007. http://java.sun.com/developer/technicalArticles/J2EE/mashup_2/ [Stand 09.2009].

[55] Peenikal S. Mashups and the Enterprise. Technical report, Mphasis, 2009.

[56] Staab S. Wissensmanagement mit Ontologien und Metadaten. *Informatik Spektrum 25(3)*, page 194–209, 2002.

[57] Staab S. and Studer R. *Handbook on Ontologies*. Springer, 2009.

[58] Tanenbaum A. S. *Distributed Sytems, Internation Edition*. Prentice Hall, 2002.

[59] Tanenbaum A. S. *Moderne Betriebssysteme*. Pearson Studium, 2009.

[60] Berners-Lee T. Semantic Web Road Map, 1998. http://www.w3.org/DesignIssues/Semantic.html [Stand 02.2012].

[61] Berners-Lee T. Erste Schritte ins Semantische Netz - RDF mithilfe von N3, 2003. http://www.bitloeffel.de/DOC/2003/N3-Primer-20030415-de.html [Stand 08.2010].

[62] Hoyer V. Stanoesvka-Slabeva K. Janner T. and Schroth C. Enterprise Mashups: Design Principles towards the Long Tail of User Needs. In *Proceedings of the 2008 IEEE International Conference on Services Computing (SCC 2008)*, pages 1–2. IEEE Computer Society (Washington), 2008.

[63] Metschke T. Analyse und Bewertung von Mashup-Werkzeugen für den Einsatz in Industrieunternehmen. Master's thesis, TU Berlin, 2009.

[64] Wilczek T. *Wirtschaftlichkeitsanalysen in IT-Projekten - Methoden, Verfahren, Werkzeuge und Vorgehensmodelle*. GRIN Verlag, 2008.

[65] Hoyer V. and Stanoevska-Slabeva K. The Changing Role of IT Departments in Enterprise Mashup Environments. *Lecture Notes in Computer Science 5472*, pages 148–154, 2009.

[66] Hoyer V. and Fischer M. Market Overview of Enterprise Mashup Tools. In *ICSOC '08 Proceedings of the 6th International Conference on Service-Oriented Computing*, pages 708–721. Springer (Berlin), 2008.

[67] Hesse W. Ontologien. *Informatik Spektrum 25(6)*, pages 477–480, 2002.

[68] W3C. Extensible Markup Language (XML), 1996-2010. http://www.w3.org/XML [Stand 01.2010].

[69] W3C. OWL Web Ontology Language, 2009. http://www.w3.org/TR/owl-semantics [Stand 09.2010].

[70] Programmable Web. Programmable Web - Mashups, APIs, and the Web as Platform. http://www.programmableweb.com [Stand 09.2009].

[71] Wikipedia. Bastard Pop. http://de.wikipedia.org/wiki/Bastard_Pop [Stand 02.2011].

[72] Wikipedia. Wikipedia: Mashup (internet), 2009. http://de.wikipedia.org/wiki/Mashup_(Internet) [Stand 09.2009].

[73] Lafon Y. Web Service Activity. http://www.w3.org/2002/ws/ [Stand 02.2012].