

Entwicklung einer SOA-basierten Webanwendung zur Buchung und Verwaltung von Segeltouren: Proprietäre Software vs. Open Source

Service-orientierte Architekturen (SOA) stellen einen Paradigmenwechsel bei der Konzeption und Realisierung von Geschäftsprozessen dar. Doch wie einfach ist die Umsetzung des SOA-Paradigmas in der Praxis? Dieser Frage ging ein Praktikum am Fachgebiet Multimedia Kommunikation (KOM) der Technischen Universität Darmstadt im Sommersemester 2007 nach. Das Praktikum wurde auf Initiative des Fachgebiets KOM ins Leben gerufen. Im Rahmen ihres University Relations Programms unterstützte die Darmstädter Software AG das Projekt mit Software, Produktschulungen sowie dem zugehörigen Produktsupport während des Projekts. Vier Teilnehmer des Praktikums, Melanie Siebenhaar, Tim Lehrig, Johannes Braun und Thomas Görgе berichten über ihre Erfahrungen.

DOI 10.1365/s11576-008-0069-2

Die Autoren

Melanie Siebenhaar
Tim Lehrig
Johannes Braun
Thomas Görgе

Technische Universität Darmstadt
 Fachgebiet Multimedia Kommunikation (KOM)
 Darmstadt
 Deutschland

Zuschriften zum Studierendenforum bitte an:
 Universität Bayreuth
 Lehrstuhl BWL VII (Wirtschaftsinformatik)
 Prof. Dr. Torsten Eymann
 Redaktion Studierendenforum
 z. Hd. Dipl.-Wirtschaftsinformatiker
 Stefan König
 Universitätsstraße 30
 95447 Bayreuth
 stefan.koenig@uni-bayreuth.de

1 Projektziel

Ausgangspunkt der Entwicklung einer SOA-basierten Anwendung ist in der Praxis selten eine „grüne Wiese“. In den meisten Fällen ist es nötig, bereits bestehende Anwendungen in die Entwicklung mit einzubeziehen, da diese als Komponenten der neuen Anwendung entweder weiterhin genutzt oder erweitert werden sollen. Auch in unserem Projekt existierte eine solche

Legacy-Anwendung, die dem Angebot und der Verwaltung von Segeltouren dient. Diese Anwendung des fiktiven Anbieters GlobeSail wird als Trainingsszenario im Rahmen von Schulungen seitens der Software AG eingesetzt.

Das Legacy-System bietet lediglich Grundfunktionalitäten zur Buchung und Verwaltung von Segeltouren an. So können die Reisen, die in einer zugehörigen Datenbank gespeichert sind, von Kunden gebucht und die zugehörigen Reise- und Personendaten durch Angestellte verwaltet werden. Die Legacy-Anwendung sollte gekapselt und um mehrere für den Kunden sinnvolle Zusatzleistungen in Form von Webservices von Drittanbietern erweitert werden. Des Weiteren war die Erstellung einer neuen benutzerfreundlichen Benutzeroberfläche angedacht, um Kunden eine Buchung über das Internet zu ermöglichen.

2 Projektorganisation

Teilnehmer an dem Praktikum waren vorwiegend Studierende der Wirtschaftsinformatik, deren Studienordnung als Pflichtbestandteil im Hauptstudium ein Wirtschaftsinformatik-Praktikum mit Schwerpunkt Anwendungssystementwicklung vorsieht. Aufgrund der großen Resonanz wurden die Teilnehmer in zwei Gruppen geteilt, die innerhalb des Projekts unterschiedliche Softwareprodukte – einerseits Open Source (erste Projektgruppe), andererseits Software AG-Pro-

dukte (zweite Projektgruppe) – verwenden sollten. Dadurch wurde die ursprüngliche Intention des Praktikums, die in der Umsetzung des SOA-Paradigmas anhand eines realen Szenarios bestand, um einen direkten Vergleich zwischen Open Source und proprietärer Software erweitert.

Neben dem Thema, das bereits durch einen leicht abgewandelten Schulungsfall der Software AG vorgegeben war, wurde auch das Vorgehen festgelegt. Zunächst sollte eine Modellierung der neuen Prozesse und der genutzten Services in BPMN (Business Process Modeling Notation) erfolgen und daraufhin die Orchestrierung mit BPEL (Business Process Execution Language) realisiert werden. Um das Vorgehen der beiden Gruppen bereits während des Projektes einschätzen zu können, wurde auch eine Zwischenpräsentation vorgesehen. Außer den genannten Punkten erhielten die Gruppen hinsichtlich der Organisation und der Umsetzung keine weiteren Vorgaben. Das Projekt dauerte drei Monate und endete mit einer Abschlusspräsentation, in der die Projektergebnisse vorgestellt wurden (**Bild 1** und **Bild 2**). Während des Praktikums erfolgte eine Betreuung durch das Fachgebiet KOM; weiterhin bestand die Möglichkeit, auf den Support der Software AG zurückzugreifen.

Die erste Aufgabe der einzelnen Gruppen bestand in der internen Organisation der Aufgaben. Aus der Analyse der Problemstellung resultierte eine Aufteilung in Untergruppen von zwei bis drei Personen, da bei dem Projekt deutlich getrennte Teilaufgaben festzustellen waren. Die Untergruppen arbeiteten sich

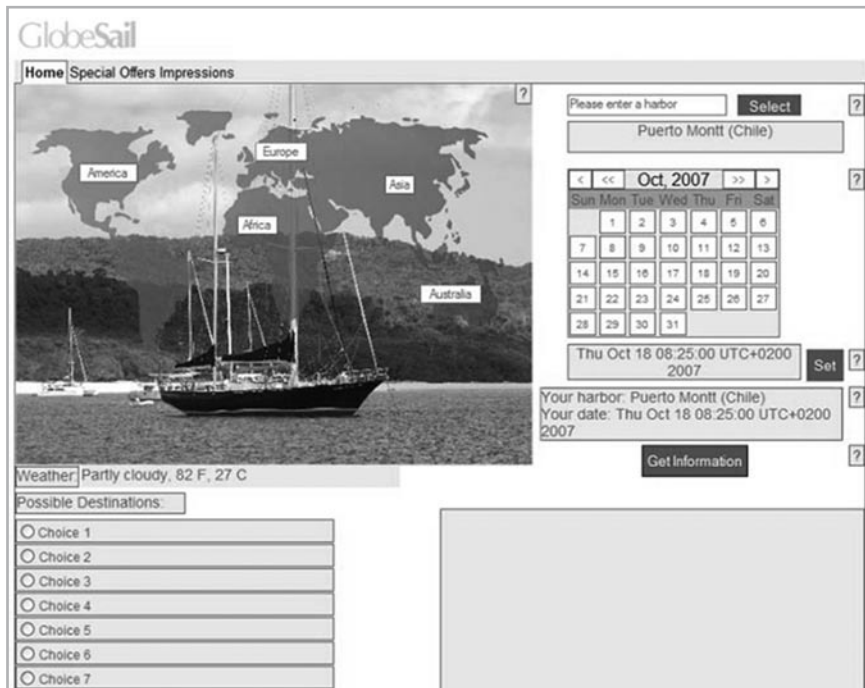


Bild 1 Oberfläche der ersten Projektgruppe



Bild 2 Oberfläche der zweiten Projektgruppe

jeweils intensiv in ein Fachgebiet ein. Der Austausch und die Koordination zwischen den Untergruppen erfolgten bei beiden Projektgruppen mittels wöchentlicher Meetings sowie der Projektplattform Assembla.

3 Projektverlauf

Die nachfolgenden Unterabschnitte des Projektverlaufs beschreiben die Tätigkeitsfelder der jeweiligen Untergruppen.

3.1 Kapselung der Legacy-Anwendung

Um die Integration einer Legacy-Anwendung in eine SOA-basierte Anwendung zu ermöglichen, ist die Kapselung der Legacy-Anwendung als Service-System hinsichtlich der relevanten Funktionalitäten analysiert werden, die dann mittels des EntireX Communicators gekapselt wurden (**Bild 3**).

Die nötigen Schritte für die Kapselung wurden anhand einer Beispiel-Anwendung im Rahmen einer Schulung der Software AG vermittelt, an der ein Student der zweiten Projektgruppe und ein wissenschaftlicher Betreuer seitens KOM teilnahmen. Für die erste Projektgruppe erwies sich die Einarbeitung zunächst nur mit Hilfe von Schulungsunterlagen als schwierig, jedoch konnte mit Hilfe der ersten Projektgruppe und mit Unterstützung des Supports letztlich die Kapselung erfolgreich durchgeführt werden. Auf die einzelnen Schritte soll hier nicht näher eingegangen werden.

3.2 Modellierung der Geschäftsprozesse

Beide Projektgruppen hatten unterschiedliche Anforderungen an die für die BPMN-Modellierung genutzten Programme. Die zweite Gruppe benötigte nur ein zusätzliches Werkzeug, um die BPMN-Prozesse grafisch abbilden zu können, da die eigentliche Implementierung mittels Software AG-Produkten erfolgen sollte. Die erste Gruppe versuchte, einen integrierten Ansatz umzusetzen, und begann mit der Suche nach Programmen, die die automatische Konvertierung von BPMN in BPEL ermöglichen.

Während sich die zweite Gruppe für den Visual Architect von Visual Paradigm entschied, testete die erste Gruppe zunächst verschiedene andere Programme. Nach ihren Recherchen im Internet fiel die Wahl auf ein Tool, welches einen integrierten Konverter enthält, der die Umwandlung von BPMN in BPEL ermöglichen sollte. Aufgrund der Instabilität und der umfangreichen Konfiguration des Programms beschloss die erste Gruppe letztlich ebenfalls, Visual Architect für die BPMN-Modellierung zu verwenden. Derartige unvorhergesehene Schwierigkeiten führten zu Zeitverlusten bei der ersten Gruppe.

Beide Projektgruppen modellierten die Geschäftsprozesse in zwei getrennten

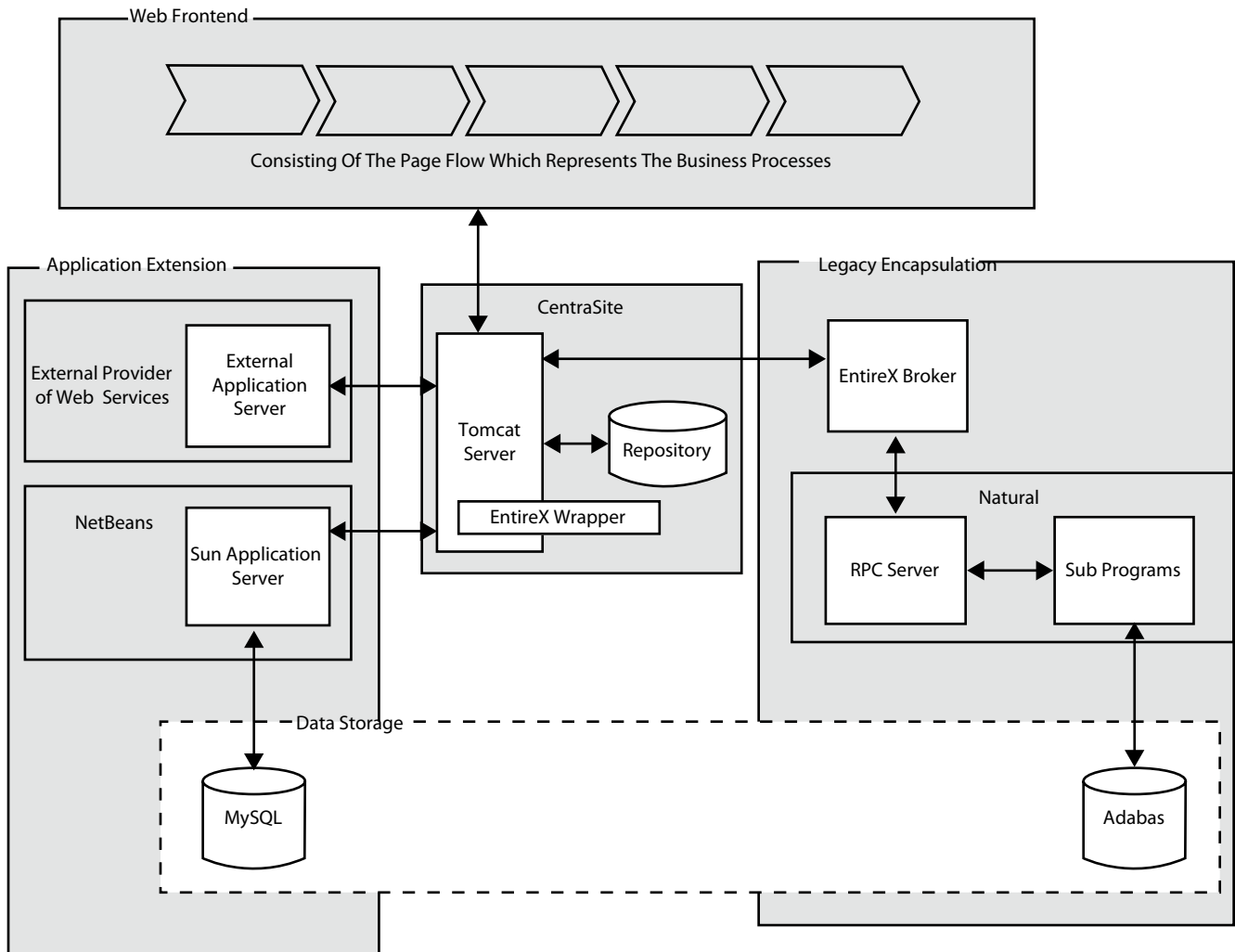


Bild 3 Architektur der zweiten Projektgruppe

Hauptprozessen, wobei der erste Prozess den internen Geschäftsprozess (Betriebsprozess) abbildet, also alle Transaktionen enthält, die von Mitarbeitern der Firma GlobeSail durchgeführt werden können, und der zweite Prozess den externen Geschäftsprozess, nämlich die Buchung einer Reise durch den Kunden, beschreibt (**Bild 4**).

Für beide Prozesse wurden zunächst seitens der zweiten Gruppe Use Cases erstellt und in einem Standardschema nach Balzert (2001) festgehalten. Aus diesen wurden die Geschäftsprozesse entwickelt und mit Hilfe von BPMN grafisch umgesetzt. Die erste Gruppe entschied sich gegen eine vorherige Modellierung von Use Cases und begann direkt mit der grafischen Ausarbeitung der BPMN-Diagramme.

3.3 Auswahl externer Webservices

Eine wichtige Voraussetzung, damit eine SOA auch erfolgreich eingesetzt werden

kann, ist, dass Services zur Verfügung stehen, die integriert werden können. Im vorliegenden Projekt wurde von beiden Gruppen SOAP als Protokoll zur Kommunikation zwischen der Oberfläche und den Webservices genutzt. Diese Restriktion ergab sich daraus, dass eine Integration anderer Protokolle mit der verwendeten Software nicht möglich war. Außerdem sollten die Services auch kostenlos zur Verfügung stehen.

In den Projektgruppen wurde zunächst überlegt, welche Zusatzleistungen dem Kunden einen höheren Nutzen bei der Buchung einer Reise bieten könnten. Die Ergebnisse dieser Überlegungen bildeten letztlich zusammen mit den Vorgaben in der Aufgabenstellung die Menge der zu integrierenden Zusatzfunktionen. So sollten zum Beispiel Flug-, Hotel- und Mietwagenbuchung mit Hilfe externer Webservices realisiert werden.

Die Recherche ergab jedoch, dass viele der Webservices kostenpflichtig sind oder in ihren Funktionen nicht den Anforderungen entsprechen, da sie nicht über eine geforderte SOAP-Schnittstelle verfügen. Anfragen an Unternehmen zur Nutzung kostenpflichtiger Webservices in einem akademischen Rahmen blieben leider unbeantwortet. Diese Probleme bestanden für beide Gruppen. Zudem bereiteten einige WSDL-Dateien, welche zur Nutzung von SOAP nötig sind, zusätzliche Probleme für die zweite Gruppe, da die Dateien entweder defekt oder nicht konform mit dem vom Application Composer geforderten Format waren.

Trotz des geringen Angebots an konformen externen Webservices ließen sich die beiden Projektgruppen nicht entmutigen und simulierten schließlich fehlende Services mittels selbst erstellter Implementierungen.

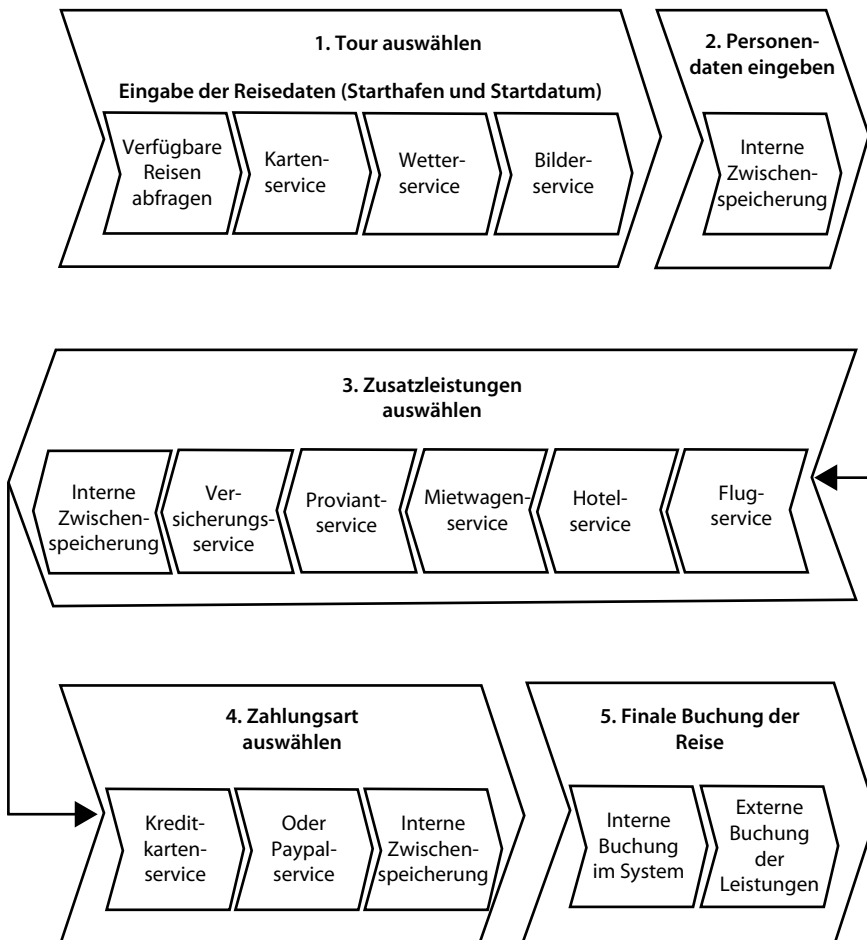


Bild 4 Übersicht „Geschäftsprozess“

3.4 Integration zusätzlicher Daten

Im Rahmen des Projektes und der Erweiterung der Anwendung war es nötig, zusätzliche Daten zu speichern. Da die existierende Datenbank im Legacy-System weiterhin genutzt werden sollte, boten sich hier zwei mögliche Vorgehensweisen an:

Eine bestand darin, alle zusätzlich benötigten Daten in die bestehende Datenbank einzufügen. Die positiven Aspekte dieses Ansatzes hätten in einer hohen Datenkonsistenz und einer verminderten Redundanz bestanden. Allerdings wäre die Einarbeitung in die Natural-Technologie nicht innerhalb des festgelegten Zeitrahmens möglich gewesen. Die andere, letztlich auch realisierte Vorgehensweise stützte sich auf eine neue, zweite Datenbank, in der alle zusätzlichen Daten hinterlegt wurden. Dies hatte zwar den Nachteil, dass einige Redundanzen nicht verhindert werden konnten, dennoch trafen beide Gruppen die Entscheidung, zusätzlich eine MySQL-Datenbank zu verwenden.

3.5 Implementierung

Während in den ersten Phasen des Projektes das Vorgehen der beiden Gruppen noch sehr ähnlich war, ergaben sich bei der Implementierung der Anwendung, bedingt durch die genutzte Software, große Unterschiede zwischen den Gruppen.

Für die erste Gruppe bestand erneut das primäre Problem in der Auswahl der Software zur Implementierung der Applikation. Die Suche und Einarbeitung in diverse Programme kostete sehr viel Zeit. Nach einigen Recherchen traf die erste Gruppe schließlich die Entscheidung, zur BPEL-Implementierung ActiveBPEL zu nutzen (**Bild 5**).

Dies bedeutete zwar einen gewissen Mehraufwand, da nun die geforderten BPMN-Modelle mit einer weiteren Modellierungsssoftware getrennt erstellt werden mussten. Doch wurde der Einstieg durch eine beinahe lückenlose Dokumentation und eine aktive Community sehr erleichtert. Die Einarbeitungszeit betrug trotzdem mehrere Wochen, da viele, meist komplexe Einstellungen zu tätigen waren.

Die zweite Gruppe verwendete zur Implementierung den Application Composer aus der Crossvision-Suite der Software AG. Dabei handelt es sich um ein Programm, welches zur professionellen

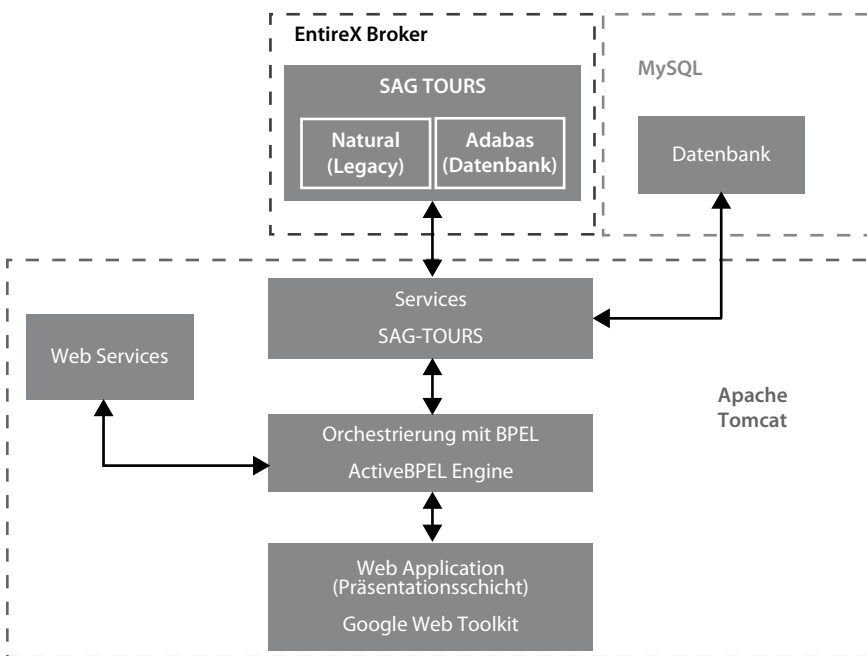


Bild 5 Architektur der ersten Projektgruppe

Erstellung von SOA-basierten Interfaces genutzt wird. Zunächst war eine gewisse Einarbeitungszeit notwendig. Hierbei diente die zur Verfügung gestellte Dokumentation als Grundlage.

Generell ist es mit dem Application Composer möglich, die Seitenlogik in BPEL zu erzeugen und auch die genutzten Services einfach per Drag-and-Drop zu integrieren. Allerdings traten schon nach kurzer Zeit Probleme auf. So konnten z. B. Rückgabewerte aus Services zur späteren Verwendung nicht abgespeichert werden. Deshalb wurde auf Empfehlung des Supports zur Umsetzung der Prozess-Logik auf Java umgestiegen. Dadurch erhöhte sich zwar der Implementierungsaufwand, allerdings konnte somit der volle Umfang von Java genutzt werden. Leider war der Umstieg mit einem gewissen Zeitverlust verbunden und auch die Integration mancher Services stellte sich schwieriger dar als erwartet. Dadurch konnte der ursprüngliche Zeitplan nicht eingehalten werden, was eine Verkürzung der Testphase sowie die rudimentäre Implementierung diverser Funktionen zur Folge hatte.

3.6 GUI-Design

Während die zweite Gruppe mit dem Application Designer eine adäquate integrierte Lösung verwenden konnte, musste die erste Gruppe für die Erstellung der HTML-Seiten erneut ein geeignetes Tool finden. Dabei fiel die Wahl auf das Google-Web-Toolkit. Allerdings zeigte sich im Laufe der Programmierung, dass das Einbinden von BPEL-Prozessen hiermit nicht ohne weiteres möglich ist, da sich in das Framework keine zusätzlichen Bibliotheken importieren lassen. Dies konnte nur über RPCs und Servlets gelöst werden, was wiederum zu einem sehr hohen Programmieraufwand führte.

4 Fazit / Persönliche Erfahrungen

An dieser Stelle möchten wir uns noch einmal recht herzlich sowohl beim Fachgebiet KOM der Technischen Universität Darmstadt als Anbieter des Praktikums als auch bei der Software AG für ihre Unterstützung bedanken.

Das Projekt war sowohl aus fachlicher als auch aus organisatorischer Sicht sehr lehrreich.

Vor allem bei der zeitlichen Planung wurde deutlich, dass bei solchen Projekten

die Einplanung von größeren Reserven zwingend erforderlich ist. Außerdem ist die Koordination einer großen Gruppe nicht zu unterschätzen und birgt zusätzlichen Aufwand.

Aus den Projekterfahrungen resultierend würden wir beim nächsten Mal mehr Zeit für die Konzeption und Recherche einplanen, aber auch bereits früher mit der Implementierungsphase beginnen; Teile der Implementierung würden somit schon parallel zur Recherche ausgeführt.

Die Verwendung einer für das SOA-Konzept ausgelegten Entwicklungsumgebung und somit die Existenz definierter Schnittstellen ermöglichte den Teilgruppen der zweiten Projektgruppe nahezu autonomes Arbeiten. Es ist zu bemerken, dass bislang im Open-Source-Bereich keine solche umfassende Entwicklungsumgebung für SOA-Anwendungen existiert. Daher musste die erste Projektgruppe verschiedene Programme für die Implementierung kombiniert verwenden und war deshalb zur Adaption der unterschiedlichen Schnittstellen auf eine enge Zusammenarbeit der Teilgruppen während der finalen Umsetzung angewiesen.

Zusammenfassend und rückblickend auf das Projekt kann festgehalten werden, dass die Potenziale der Open-Source-Software zur Umsetzung einer SOA mittels Webservices noch nicht ausreichend ausgeschöpft sind, um den theoretischen Anforderungen an eine SOA zu genügen – die Entwicklungsphase hat hier noch kein Ende gefunden.

Aus Sicht der zweiten Gruppe kann gesagt werden, dass die zur Verfügung gestellten Produkte ineinander greifen und ein effizientes Arbeiten mit SOA ermöglichen. Allerdings ist noch keine hundertprozentige Unterstützung von Webservices gegeben, wie sie zurzeit im Internet gefunden werden können. Des Weiteren ist die Nutzung von BPEL als Prozesssprache noch nicht ausgereift und konnte unseren Ansprüchen nicht genügen. Der Einsatz von BPEL wurde auch vom Support der Software AG nicht zur Verwendung empfohlen.

Die Erfahrungen, die wir während des Praktikums sammeln konnten, sind für uns und auch hinsichtlich unseres Studiums sehr wertvoll und wichtig. Wir freuen uns deshalb, dass dieses Projektszenario auch in Zukunft an anderen Universitäten eingesetzt wird.

Literatur

Balzert, Helmut (2001): Lehrbuch der Software-Technik: Software-Entwicklung. 2. Aufl., Spektrum Akademischer Verlag, Heidelberg.