

Toolunterstützung bei der vermarktungsorientierten Entwicklung von Web Services als Bausteine komplexer betrieblicher Anwendungssysteme

Nico Brehm, Jorge Marx Gómez, Andreas Ziesenitz

Abteilung Wirtschaftsinformatik

Carl von Ossietzky Universität Oldenburg, Department für Informatik
Ammerländer Heerstr. 114-118, 26129 Oldenburg
{nico.brehm, jorge.marx.gomez}@uni-oldenburg.de
andreas.ziesenitz@informatik.uni-oldenburg.de

Abstract. Der Beitrag beschreibt die Notwendigkeit zur erweiterten Untersuchung von Vorgehensmodellen und Entwicklungs-Werkzeugen für den Web Service-Kontext, die sich aus der mangelnden Integrationsfähigkeit von Web Services ergibt, die von unabhängigen Organisationen entwickelt werden. Bei der Betrachtung von Web Services als Softwarekomponenten, deren formale oder informale Spezifikation zur ausgelagerten Entwicklung durch Dritte herangezogen werden kann, ergeben sich Möglichkeiten zum konkurrierenden Angebot dieser, woraus wiederum verschiedene Vorteile resultieren. Im Beitrag wird ein Vorgehensmodell zur ausgelagerten Entwicklung von Web Services beschrieben, deren Nutzung im Kontext komplexer betrieblicher Anwendungssysteme vorgesehen ist. Darauf aufbauend wird die Schaffung von Werkzeugen motiviert, die neben der Umsetzung des Vorgehensmodells ebenfalls die Interpretation und Erstellung von Web Service-Spezifikationen unterstützen. Zur zusammenfassenden Spezifikation sind unterschiedliche Standardtypen erforderlich, die sowohl funktionale als auch nicht-funktionale Eigenschaften beschreiben. Der Vermarktungsaspekt ist dabei jedoch bislang nur unzureichend behandelt worden. Es wird abschließend ein Prototyp zur werkzeuggestützten Entwicklung und Vermarktung von Web Services vorgestellt.

1 Einleitung

Damit Softwarekomponenten in verschiedenen Anwendungssystemen eingesetzt werden können, ist eine Standardisierung der Beschreibung nötig, anhand derer die (Wieder)verwendbarkeit für individuelle Zwecke überprüft werden kann. In [AB⁺02] wurde eine einheitliche Spezifikation für Softwarekomponenten vorgeschlagen, die es einem Hersteller ermöglicht, eine Komponente möglichst eindeutig und vollständig zu beschreiben¹. Diese Beschreibung kann für die Bewertung der Verwendbarkeit einer Komponente in einem Anwendungssystem herangezogen werden, ohne dass Details über die komponenten-interne Umsetzung bekannt sein müssen.

¹ Overage beschreibt in seiner Arbeit [Ove06] ein erweitertes Modell eines Spezifikationsrahmens, das u. a. auf [AB⁺02] aufbaut.

Ein Web Service kann als Softwarekomponente aufgefasst werden (vgl. [HM⁺06], S. 70). Jedoch unterliegen Web Services erweiterten Anforderungen, da sie im Gegensatz zu konventionellen Softwarekomponenten nicht physisch in ein System integriert, sondern über das Internet oder Intranet bereitgestellt werden (vgl. [Krü06]). Ein Konsument (Nutzer) kann auf die angebotenen Leistungen zugreifen und auf diese Weise eine Integration externer Anwendungsfunktionen erreichen². Laut [SDR03] kann die Spezifikation für Softwarekomponenten des Arbeitskreises 5.10.3 der Gesellschaft für Informatik [AB⁺02] auch für die Spezifikation von Web Services herangezogen werden. In [OvT05] wird ein Modell zur Spezifikation von Web Services dargestellt, was aus einem Modell zur Spezifikation von Softwarekomponenten hergeleitet wurde. Dabei wird die Nutzung existierender Spezifikationsstandards auf unterschiedlichen Spezifikationsebenen vorgeschlagen.

Eine bedeutende Rolle spielen standardisiert beschriebene Web Services bei der Idee eines Föderierten Enterprise Resource Planning-Systems (FERP-System). Dabei werden Anwendungsfunktionen des ERP-Systems durch Web Services von unabhängigen Anbietern bereitgestellt (vgl. [BMR06]). Ziel dabei ist es, den Wert eines ERP-Systems durch die Verfügbarkeit einer Vielzahl zueinander kompatibler Komponenten (Web Services) zu erhöhen, wodurch wiederum ein Marktpotential für unabhängige Anbieter einzelner Web Services entstehen kann. Durch ein zentrales Konsortium sollen dabei die Schnittstellen, Aufgaben und weiteren Merkmale eines Web Service weitestgehend standardisiert werden (vgl. [BrM07]). Die dazu notwendigen Standards werden nachfolgend als gegeben betrachtet. Für eine detaillierte Beschreibung der Struktur solcher Standards soll auf [OvT05] und [BrM07] verwiesen sein.

Bei der Implementierung eines Web Services ist eine Toolunterstützung für den Entwickler von Vorteil [HöW02], da während des Entwicklungsprozesses verschiedene automatisierbare Aufgaben anfallen. Eine Toolunterstützung kann den Web Service-Entwickler ausgehend von der Spezifikation eines Web Services durch den Entwicklungsprozess leiten. Der Entwickler bzw. der Anbieter³ kann sich dadurch auf die Kernaufgaben, wie die Implementierung der (Fach-)Logik und die Vermarktung des Web Services, konzentrieren, wodurch sich die Entwicklungszeit von Web Services verkürzt. Als Folge ergibt sich ein vereinfachter Zugang zu einem Markt für Web Services aus der Sicht eines Anbieters.

Der vorliegende Beitrag beschreibt ein Modell für die toolgestützte Implementierung von Web Services, zur Konstruktion komplexer betrieblicher Anwendungssysteme.

² Ein Web Service ist aus der Sicht eines Nutzers eine Softwarekomponente, die ausschließlich Dienste anbietet (exportiert) und keine Dienste nachfragt (importiert). Die Schnittstellen der angebotenen Dienste werden durch Konstrukte standardisierter XML-Sprachen spezifiziert. Die Nutzung eines Dienstes ist abhängig von einem Austausch von Nachrichten, deren Struktur durch die Schnittstellenspezifikation festgelegt wird. Nachrichten werden grundsätzlich über ein bestehendes Netzwerk übertragen. Aus der Sicht eines Anbieters ist ein Web Service ein Anwendungssystem, das die benannten Eigenschaften aus Nutzersicht für jeden Nutzer erfüllt. Die Verfügbarkeit aller importierten Dienste wird durch den Anbieter in Eigenverantwortung (autonom) organisiert. Die ausgelagerte Entwicklung von Komponenten, die durch Web Services repräsentiert werden, wird durch die mögliche „Ausblendung“ von Abhängigkeitsbeziehungen vereinfacht.

³ Nachfolgend werden die Rollen des Entwicklers und des Anbieters eines Web Services zur Vereinfachung zusammengefasst, d.h., dass der Entwickler auch als Anbieter agiert.

In einem ersten Schritt werden dazu die Anforderungen an eine Toolunterstützung ermittelt, wobei ein Vorgehensmodell für die Implementierung von bereits im Vorfeld spezifizierten Web Services hergeleitet wird (Kapitel 2). Anschließend werden die Standardtypen benannt, die für die Beschreibung dieser Web Services verwendet werden (Kapitel 3). Aufbauend auf den Anforderungen des Vorgehensmodells wird schließlich die Architektur eines unterstützenden Tools beschrieben (Kapitel 4) und ein Prototyp dieses Tools vorgestellt (Kapitel 5).

2 Anforderungen an die Toolunterstützung

Zunächst sollen die Anforderungen an die Toolunterstützung benannt werden. Dazu werden zuerst allgemeine Anforderungen an eine Toolunterstützung aufgezeigt. Dann wird ein Vorgehensmodell für die Implementierung bereits spezifizierter Web Services im betrieblichen Kontext vorgeschlagen, welches durch das Tool unterstützt werden soll.

2.1 Allgemeine Anforderungen an eine Toolunterstützung

Generell sollte ein Tool eine Unterstützung bei der Ausführung einer bestimmten Aufgabe oder einer Menge von Aufgaben bieten und kann vor allem dazu genutzt werden, automatisierbare Aufgaben erledigen zu lassen. Dadurch wird der Benutzer (Web Service-Entwickler) entlastet, so dass sich vor allem eine Zeitersparnis ergibt. Darüber hinaus kann durch ein Tool auch die Komplexität einer Aufgabe verborgen werden. Ein Beispiel dafür ist die Erstellung von standardkonformen Dokumenten, wobei ein Tool alle benötigten inhaltlichen Angaben zur Dokumenterstellung assistentengestützt abfragt und die Generierung des eigentlichen Dokuments dann im Hintergrund vornimmt. Weiterhin sollte ein Tool dazu beitragen, die Entstehung von Fehlern zu vermeiden, die bei einer manuellen Durchführung von Aufgaben potentiell entstehen.

2.2 Einordnung der Web Service-Implementierung

Zunächst soll die Web Service-Implementierung in die übergeordneten Prozesse der Entwicklung komplexer Anwendungssysteme, wie beispielsweise von Very Large Business Applications (VLBA) ([Rau07], S. 3), eingeordnet werden. Die Entwicklung eines Web Service kann unter zwei Zielstellungen erfolgen. Zum einen kann die Implementierung eines Web Service im Rahmen der Entwicklung eines Anwendungssystems erfolgen, wenn dieses den Web Service benötigt. Zum anderen kann der Web Service losgelöst von einer darüberstehenden Anwendung entwickelt werden, zum Beispiel um diesen in verschiedene Anwendungssysteme zu integrieren.

Wird der Web Service zusammen mit einer Anwendung entwickelt, gliedert sich die Implementierung des Web Service in den Entwicklungsprozess dieser Anwendung ein. Im Software-Engineering ist vorgesehen, dass die Entwicklung von Software nach einem Vorgehensmodell, wie zum Beispiel dem Wasserfallmodell, erfolgt (vgl. [Mey97], S. 464).

Unabhängig davon, welches Vorgehensmodell verwendet wird, ist eine Implementierungs- oder Realisierungsphase vorgesehen. Wenn für die zu erstellende Anwendung ein externer Web Service benötigt wird, kann dessen Implementierung als Subprozess der Implementierungsphase gesehen werden, der wiederum einem eigenen speziellen Vorgehensmodell folgt. Dem Web Service-Entwickler ist dazu eine Spezifikation des zu erstellenden Web Services vom übergeordneten Entwicklungsprozess zur Verfügung zu stellen. Diese Spezifikation sollte den Web Service möglichst vollständig beschreiben. Hierzu sind beispielsweise eine Schnittstellenbeschreibung und eine Beschreibung der (fachlichen) Funktionalität des Web Services nötig. Darauf aufbauend kann die Implementierung des Web Services unabhängig von der Implementierung der übergeordneten Anwendung erfolgen. Nach der Implementierung steht der nutzbare Web Service zur Verfügung, der wieder dem übergeordneten Entwicklungsprozess des Anwendungssystems zugeführt wird, so dass dieser fortschreiten kann.

Wird der Web Service unabhängig von einer Anwendung entwickelt, muss (für den hier betrachteten Anwendungsfall) dem Entwickler ebenfalls eine Spezifikation zur Verfügung stehen. Beispielhaft soll als konkretisierter Anwendungsfall für diese Alternative die Erstellung eines Web Service für ein ERP-System genannt sein. Da die Erstellung von Web Services in diesem Umfeld auf standardisierten Web Service-Typen aufsetzt, ist das Vorhandensein einer geeigneten Spezifikation gewährleistet [BrM07].

2.3 Phasen des Vorgehensmodells

Zur Herleitung eines Vorgehensmodells für die Implementierung von Web Services, können Phasenmodelle des Software-Engineerings als Grundlage verwendet werden. Diese Modelle umfassen laut [Mey97] die Phasen Problemanalyse, Entwurf, Implementierung, Funktionsüberprüfung, Installation und eine sich wiederholende Phase der Wartung. Da sich die Entwicklung eines Web Service von der Entwicklung einer allein stehenden Software unterscheidet, ist eine Anpassung konventioneller Phasenmodelle erforderlich. Dabei soll auch berücksichtigt werden, dass das Vorgehensmodell auf die direkte Vermarktung von Web Services ausgerichtet sein soll, da dies der hier besonders hervorzuhebenden Vereinfachung des Marktzugangs für Anbieter führt.

Da die Spezifikation des Web Services als vorhanden betrachtet wird, sind eine Problemanalyse und ein Entwurf, wie sie im konventionellen Phasenmodell des Software-Engineerings beschrieben werden, nicht in direkt vergleichbarer Form nötig. Stattdessen wird eine neue Phase eingeführt, die sich auf die Analyse der Spezifikation bezieht. Der Web Service-Entwickler hat unter Umständen kein Vorwissen über den geforderten Web Service, weil er nicht an der Erstellung der Spezifikation beteiligt war. Daher ist es sinnvoll, dass der Entwickler die vorgegebene Spezifikation analysiert, um die geforderte Funktionalität des späteren Web Services zu überblicken. Außerdem kann dabei ein Entwurf des Web Service als optional betrachtet werden, der für den Fall sehr komplexer Funktionen relevant ist. Allerdings ist prinzipiell von einer verminderten Komplexität von Web Services im Vergleich zu betrieblichen Anwendungssystemen auszugehen.

Die Implementierungsphase ist essentiell für ein Vorgehensmodell zur Entwicklung einer Software und wird daher auch in das Vorgehensmodell übernommen.

Die auf die Implementierung folgende Phase der Funktionsüberprüfung (im folgenden Test genannt) wird ebenfalls übernommen. Zu beachten ist allerdings, dass an dieser Stelle nur Komponenten- bzw. Modultests durchgeführt werden können. Wird der Web Service zusammen mit einer Anwendung entwickelt, die den Web Service benötigt, bleiben die Tests der Integration des Web Service durch die übergeordnete Anwendung Aufgabe des übergeordneten Entwicklungsprozesses.

Da das Vorgehensmodell die Vermarktung von Web Services einschließen soll, muss der Entwickler die nötigen Vermarktungsparameter konfigurieren können. Die Bestimmung dieser Parameter erfolgt in einer separaten Phase für die Konfiguration der Vermarktung, die in das Vorgehensmodell eingefügt wird.

Der Begriff Vermarktung lässt sich im Allgemeinen als Menge von Bemühungen bzw. methodischer Maßnahmen eines Anbieters zusammenfassen, die sich auf die aktive Absatzförderung beziehen. Zur Ausrichtung eines Anbieters auf die Erfordernisse des Absatzmarktes ist dabei zunächst die Bereitstellung von Web Services notwendig, deren funktionale und nichtfunktionale Eigenschaften der Nachfrage des Marktes entsprechen. Zur Vermarktung von Web Services sind insbesondere die drei Parameter Funktionsumfang, Qualität und Preis zu berücksichtigen. Der Funktionsumfang ist bereits durch die Spezifikation festgelegt, so dass der Entwickler nur beschränkte Möglichkeiten hat, diese nur in Ausnahmefällen anzupassen, z.B. bei der Erweiterung der spezifizierten Funktionalität. Die Parameter Qualität, wozu beispielsweise die Zusicherung einer gewissen Verfügbarkeit oder Performance gehört, und Preis für die Nutzung können allerdings vom Entwickler beeinflusst werden. Dabei ist zu beachten, dass die Qualitätseigenschaften zum Teil von der Umgebung abhängen, in der der Web Service später bereitgestellt wird. Daher ist sicherzustellen, dass die zugesicherten Qualitätseigenschaften eingehalten werden können. Dies ist insbesondere dann nötig, wenn der Entwickler nicht selbst Anbieter sein sollte und dadurch keinen direkten Einfluss auf die Laufzeitumgebung hat.

Die Phase der Installation gestaltet sich bei Web Services anders, als bei der Entwicklung konventioneller Software-(komponenten). Ein Web Service wird nicht direkt beim Kunden bzw. Abnehmer, sondern auf eigenen Systemen installiert (Deployment). Der Kunde kann den Web Service anschließend nutzen. Das Deployment des Web Service wird in das Vorgehensmodell aufgenommen, so dass dieser nach dieser Phase genutzt werden kann. Die Wartung eines Web Service soll nicht in das Vorgehensmodell integriert werden, da die Erweiterung oder Verbesserung eines Web Services entweder als Erstellung eines neuen Web Service zu betrachten ist, oder bereits indirekt durch das Vorgehensmodell aufgrund der Rückkopplung zwischen den Phasen erlaubt wird.

Abbildung 1 fasst die Phasen der Entwicklung und Bereitstellung von Web Services, aus denen das Vorgehensmodell gebildet wird, noch einmal zusammen. Ausgehend von der *Analyse der Spezifikation* eines Web Service wird die *Implementierung* mit anschließendem *Test* durchgeführt. Darauf folgen die *Konfiguration der Parameter für die Vermarktung* und das *Deployment* des Web Service. Das Ergebnis des Prozesses ist der veröffentlichte Web Service. Zusätzlich zu den Phasen werden die Funktionen aufgelistet, die durch ein Tool unterstützt werden können.

Wie in der Abbildung ersichtlich, erfordert das Vorgehensmodell keinen strikten sequenziellen Ablauf. Es ist zwar nötig, dass alle Phasen mindestens einmal durchlaufen werden und kein Überspringen einer Phase möglich ist, Rückschritte müssen allerdings zulässig sein. Dabei ist es auch möglich, mehrere Phasen zurückzuspringen. Beispielsweise könnte von der Phase der Konfiguration der Vermarktungsparameter ein Rückschritt zur Implementierungsphase nötig sein. Dies wird zugelassen und wäre mit einem Zwischenschritt über die Testphase möglich.

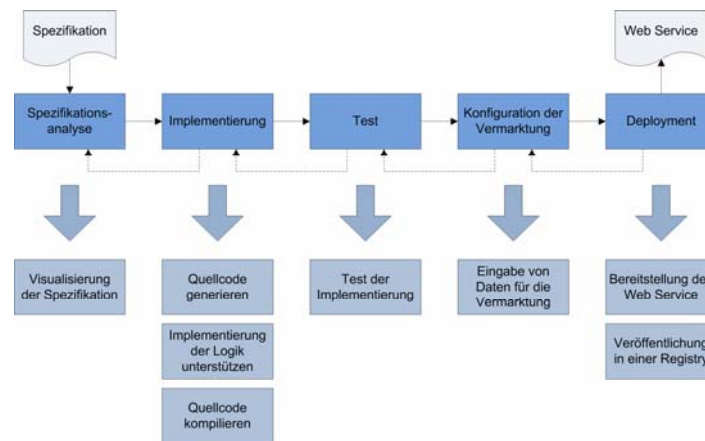


Abbildung 1: Phasen des Vorgehensmodells und Funktionen der Toolunterstützung

2.4 Voraussetzungen an die Laufzeitumgebung

Durch die festgelegte Ausrichtung auf die Web Service-Vermarktung müssen einige Voraussetzungen an die Laufzeitumgebung gestellt werden, die zur Bereitstellung des Web Services eingesetzt wird.

Ein Tool soll zwar die Eingabe von Daten zur Vermarktung unterstützen, jedoch reicht dies allein für eine erfolgreiche Vermarktung nicht aus. Denn dadurch ist es lediglich möglich ein Angebot für die Nutzung eines Web Service zu publizieren. Weiterhin ist es erforderlich, dass die Nutzung auch abgerechnet wird [BoS03], was wiederum eine Authentifizierung von Nutzern des Web Services sowie eine Zugriffskontrolle erfordert. Zum andern müssen Aufrufe (Nutzungen) protokolliert werden, damit später eine Abrechnung der genutzten Leistungen möglich ist. Beides ist allerdings nicht direkte Aufgabe der zu implementierenden (Fach-)Funktionalität des Web Service selbst, so dass auch die Toolunterstützung hier keine Hilfe bieten muss. Vielmehr muss die eingesetzte Laufzeitumgebung diese Anforderungen erfüllen. Die Überwachung der zugesicherten Qualitätskriterien muss ebenfalls zur Laufzeit, z.B. von einem anderen Tool oder der Laufzeitumgebung selbst, durchgeführt werden (vgl. [DD⁺04]).

Weitere Konzepte, die zum Beispiel die Sicherheit oder eine Transaktionsunterstützung vorsehen und zum Teil schon durch Standards, wie Web Service Security (WSS) oder WS-Reliability, beschrieben werden, sollen ebenfalls aus der Funktionalität der Toolunterstützung ausgenommen werden. Wenn dies benötigt wird, muss eine Realisierung über die Laufzeitumgebung erfolgen, da das Tool für eine möglichst einfache und schnelle Implementierung von Web Services verwendet werden soll, die auf die anschließende Vermarktung ausgerichtet sind. Die Entwicklung einer solchen Laufzeitumgebung ist daher als Motivation für aufbauende Arbeiten zu betrachten, deren Ergebnis eine Grundlage für die vereinfachte Vermarktung von Web Services für zukünftige Anbieter-(Netzwerke) sein kann.

3 Standards für die Toolunterstützung

Die Unterstützung von allgemeinen Standards ist an zwei Stellen des hergeleiteten Vorgehensmodells von Bedeutung. Sowohl die Spezifikation, als Input in den Entwicklungsprozess des Web Service, als auch die Eingabe der Daten für die Vermarktung, die im Laufe der Entwicklung festgelegt werden, müssen auf Standards basieren.

3.1 Standards für die Spezifikation

Grundsätzlich gibt es zwei verschiedene Ansätze für die Web Service-Entwicklung, die als Code-First- und Contract-First-Ansatz bezeichnet werden. Beim Code-First-Ansatz wird zuerst die Logik des Web Service implementiert. Beim Contract-First-Ansatz wird zuerst eine Spezifikation festgelegt und darauf aufbauend die Implementierung des spezifizierten Web Service durchgeführt. Im Allgemeinen wird der Contract-First-Ansatz empfohlen (vgl. [FTW07], S. 58). Auch das vorgeschlagene Vorgehensmodell setzt den Contract-First-Ansatz voraus. Dadurch kann sichergestellt werden, dass der resultierende Web Service die gestellten Anforderungen erfüllt und kompatibel zu anderen Komponenten ist, die sich auf die zugrunde liegende Spezifikation beziehen. Außerdem können mit einer vorhandenen Spezifikation der Web Service und die nutzende Software(-komponente) unabhängig voneinander entwickelt werden.

Für den Entwickler bietet sich mit der Einhaltung einer vorgegebenen Spezifikation zudem die Sicherheit, dass der Web Service benötigt und auch eingesetzt wird, da durch das Vorhandensein einer Spezifikation von einem funktionsbezogenen Bedarf auszugehen ist.

Als Grundlage der Beschreibung eines Web Service soll für einen späteren Test die Web Service Secure Marketing Language (WSSML) genannt sein, die im Rahmen der Projektgruppe "Föderierte ERP-Systeme auf Basis von Web Services und P2P-Systemen" im Department für Informatik an der Carl von Ossietzky Universität Oldenburg entwickelt wurde⁴.

⁴ <http://ferp.informatik.uni-oldenburg.de/>

Da die Standards zur Spezifikation von einem zentralen Konsortium vorgegeben werden, soll an dieser Stelle auf eine konkrete Beschreibung des Aufbaus der verwendeten Standards verzichtet werden. Ein Verweis auf [Ove05] soll hier ausreichen.

3.2 Standards für die Vermarktung

Damit ein Web Service vermarktet werden kann, ist eine Beschreibung der Vermarktungsparameter nötig. Dazu gehören insbesondere die Kosten, die für die Nutzung eines Web Service anfallen, und die Qualitätskriterien, die der Provider garantiert. Die Beschreibung für die Vermarktung sollte automatisch ausgewertet werden können, da dies eine Voraussetzung für die automatisierte Auswahl und Einbindung von Web Services zur Laufzeit ist. Durch die Beschreibung in einem standardisierten Format, kann Web Service-Nutzer die Angebote unterschiedlicher Anbieter (automatisiert) miteinander vergleichen.

Für die Beschreibung der Kosten für die Nutzung soll die FERP SPL (FERP Service Pricing Language) als Untersprache in die WSSML benannt werden. Die FERP SPL wurde im Zusammenhang mit WSSML vorgeschlagen (siehe Abschnitt 3.1). Mit dieser Sprache ist es zur Vereinfachung der Problematik derzeit zunächst nur möglich, für jeden Web Service einen Preis pro Operationsaufruf festzulegen. Ein erweitertes XML-Schema zur Festlegung von Preisen wird in [BoS03] vorgestellt. Zur Angabe der Qualitätskriterien soll beispielhaft auf die Vorschläge in [LSE03], [Ove05], [DD⁺04] und [Tia05] verwiesen sein.

4 Architektur eines erweiterten Tools zur Web Service-Entwicklung

Die Architektur des Tools (Abbildung 2) ergibt sich aus den Anforderungen an die Toolunterstützung. Da das hergeleitete Vorgehensmodell nicht an bestimmte Programmiersprachen oder Laufzeitumgebungen gebunden ist, soll auch das Tool verschiedene Sprachen unterstützen. Besonders die Umsetzung des kompletten Vorgehensmodells und die Möglichkeit beliebige Programmiersprachen abzudecken, stellen eine Neuerung gegenüber bisher existierenden Tools, wie beispielsweise *Apache Axis2*, *Altova Mission Kit 2007*, *Web Standard Tools für Eclipse* oder *Stylus Studio 2007* dar.

4.1 Runtime Engine

Die Runtime Engine stellt den Mittelpunkt der Architektur des Tools dar und enthält die Ablauflogik für die Steuerung des Entwicklungsprozesses eines Web Service. Die Runtime Engine muss überwachen, dass der vorgegebene Prozess eingehalten wird und dem Benutzer die jeweils korrekte Sicht für die aktuell anstehenden Aufgaben angezeigt wird. Die Runtime Engine verfügt über eine direkte Schnittstelle zum GUI (Graphical User Interface). Außerdem verfügt die Runtime Engine über drei Module, die für die Konfigurationsverwaltung, die Projektverwaltung und die Einbindung von Plugins benötigt werden.

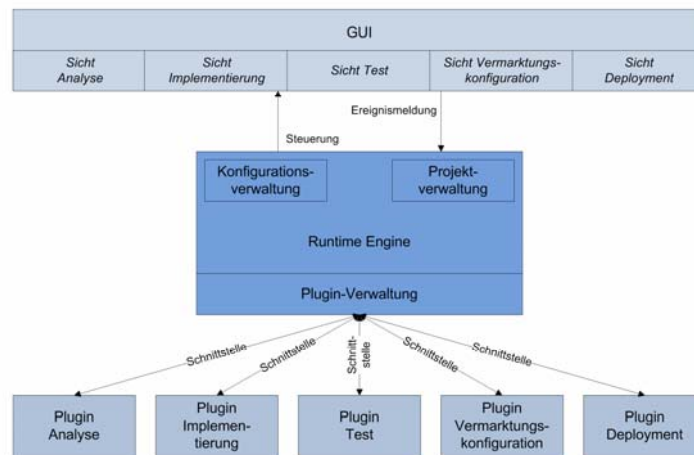


Abbildung 2: Architektur des Tools

4.2 GUI

Das GUI ist die Schnittstelle zwischen Runtime Engine und dem Benutzer des Tools, d.h. dem Entwickler. Durch das GUI kann die gesamte Entwicklung und die Vermarktung eines Web Service durchgeführt werden. Daher muss das GUI verschiedene Sichten auf den zugrunde liegenden Entwicklungs- und Vermarktungsprozess bieten. Für die Steuerung der Anzeige ist die Runtime Engine zuständig, an die die Eingaben des Benutzers gesendet werden. Die Oberfläche wird nicht nur statisch von der Runtime Engine vorgegeben, sondern auch dynamisch von den angebotenen Plugins beeinflusst. Dies darf allerdings nur in einem geordneten Rahmen erfolgen, der von der Plugin-Verwaltung überwacht wird und sich auf das Vorgehensmodell bezieht.

4.3 Plugins

Die Plugins realisieren die Funktionalität zur Unterstützung des Benutzers (Funktionslogik) in den einzelnen Phasen des Vorgehensmodells. Für jede Phase wird ein Plugin verwendet, so dass insgesamt fünf Plugins vorgesehen sind. Die Plugins kommunizieren direkt nur mit der Plugin-Verwaltung der Runtime Engine. Einzelne Plugins sollten voneinander weitestgehend unabhängig sein. Das bedeutet, dass ein Plugin die Funktionalität für eine gesamte Phase des Vorgehensmodells enthalten muss. Abhängigkeiten zwischen einzelnen Plugins sollten vermieden werden. Sollten dennoch Abhängigkeiten zu anderen Plugins bestehen, müssen diese beschrieben werden. Jedoch wird dieser Aspekt hier aus Platzgründen nicht betrachtet. Dadurch, dass die Funktionslogik durch austauschbare Plugins realisiert wird, ist eine beliebige Erweiterung des Tools durchführbar. Vorrangig ist dabei die Toolunterstützung für verschiedene Programmiersprachen zu nennen, die durch die Entwicklung verschiedener Plugins unterstützt werden können.

Darüber hinaus ist auch die Entwicklung unterschiedlicher Plugins für die gleiche Programmiersprache möglich, die sich zum Beispiel in ihrem Funktionsumfang oder der Qualität, z. B. durch eine besondere Übersichtlichkeit des Plugins zur Visualisierung der Spezifikation des Web Services, unterscheiden.

5 Prototyp

Der Prototyp wurde als Feature für das Framework Eclipse⁵ realisiert. Die Entscheidung für Eclipse und gegen eine komplett selbständige Anwendung wurde getroffen, da Eclipse bereits viele Konzepte der Tool-Architektur bietet, die wiederverwendet werden können. Im Rahmen der Entwicklung des Prototyps genügte es, Eclipse um die Plugins für die einzelnen Phasen des Vorgehensmodells zu erweitern. Da dem Eclipse-Framework bereits eine Plugin-Architektur zugrunde liegt, können diese Plugins die vorhandenen Schnittstellen verwenden und so integriert werden. Abbildung 3 zeigt beispielhaft, wie sich das Tool in das Eclipse-Framework eingliedert. Über mehrere Menüpunkte sind die verschiedenen Funktionen des Tools für den Benutzer zugänglich. Die Evaluation des Prototyps für einen beispielhaften Web Service eines FERP-Systems hat ergeben, dass sich das zugrunde gelegte Vorgehensmodell für die Entwicklung von Web Services, die für die Vermarktung vorgesehen sind, eignet. Aus Platzgründen kann dieser Web Service nicht näher beschrieben werden. Es bleibt jedoch festzuhalten, dass mit dem Entwicklungsprozess, der durch das Vorgehensmodell beschrieben wird, ein Web Service ausgehend von einer Spezifikation implementiert und veröffentlicht werden kann. Zudem ist die Angabe von Vermarktungsparametern möglich. Insbesondere die Spezifikationskonforme Implementierung der Funktionalität und die Veröffentlichung eines kostenpflichtigen Web Services, unter Angabe von Preisen, konnten durch das Tool besonders vereinfacht werden.

6 Zusammenfassung

Im vorliegenden Beitrag wurde beschrieben inwieweit Web Services als Softwarekomponenten betrachtet werden können, die von unabhängigen Entwicklern implementiert und zur Nutzung in komplexen Anwendungssystemen bereitgestellt werden können. Dazu wurde ein Vorgehensmodell vorgestellt, das diese Betrachtungsweise aufgreift und die ausgelagerte Entwicklung von Web Services unterstützen soll. Dabei wird im Speziellen der Aspekt der Vermarktung von Web Services einbezogen, wodurch sich Unterschiede zu konventionellen Vorgehensweisen der reinen Implementierung von Web Services ergeben. Dieser Unterschied wird insbesondere bei der Betrachtung derzeitiger Tools deutlich, die zur Entwicklung von Web Services zur Verfügung stehen. Weiterhin wurden notwendige Standardtypen benannt, die zur Verbesserung der Integrationsfähigkeit von Web Services auch bei der Bereitstellung von Tools zu berücksichtigen sind, so dass die ausgelagerte (und konkurrierende) Entwicklung von Web Services im Rahmen komplexer Projekte möglich wird.

⁵ <http://www.eclipse.org>

Unter der Motivation, sowohl das vorgeschlagene Vorgehensmodell als auch die Nutzung von Standards bei der Entwicklung und Vermarktung von Web Services durch Tools zu unterstützen, wurde eine Architektur vorgeschlagen, deren Umsetzbarkeit durch die prototypische Implementierung auf der Basis einer Sammlung von Eclipse-Plugins gezeigt werden konnte. Die grundlegende Annahme der Verfügbarkeit einer vorhandenen Spezifikation für einen zu entwickelnden Web Service, als Voraussetzung zur Anwendbarkeit des Vorgehensmodells, zeigt die Notwendigkeit zur weiterführenden Untersuchung dieser Thematik. Es ergibt sich folgende Frage: *Wie kann der Prozess zur Erstellung der vorausgesetzten Spezifikation eines Web Services unter Einbeziehung von Standards durch geeignete Werkzeuge unterstützt werden?*

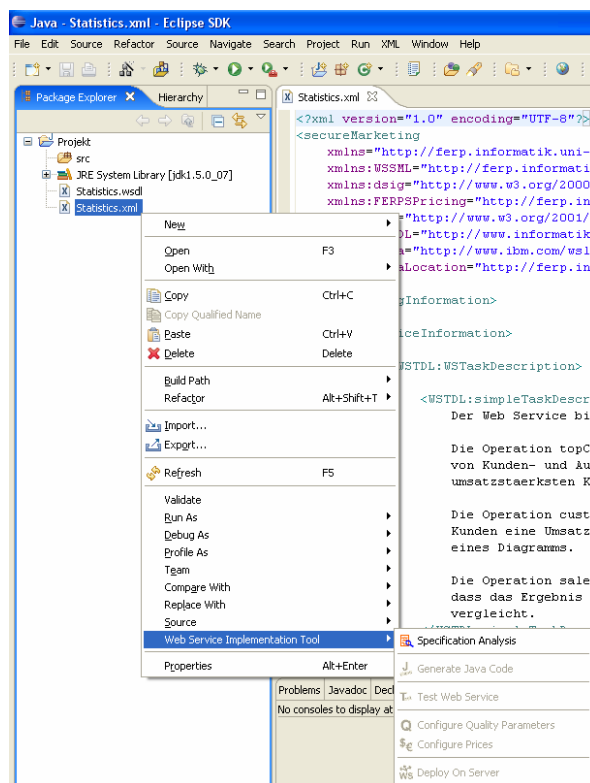


Abbildung 3: Screenshot des Tools

Literaturverzeichnis

- [AB+02] Ackermann, J., Brinkop, F., Conrad, S., Fettke, P., Frick, A., Glistau, E., Kotlar, O., Loos, P., Mrech, H., Ortner, E., Raape, U., Overage, S., Sahn, S., Schmietendorf, A., Teschke, T., Turowski, K.: Vereinheitlichte Spezifikation von Fachkomponenten. In: Turowski, K. (Hrsg.). Gesellschaft für Informatik (Arbeitskreis 5.10.3) (2002)

- [BrM07] Brehm, Nico; Marx Gómez, Jorge: Web Service-based specification and implementation of functional components in Federated ERP-Systems, Proceedings of 10th International Conference on Business Information Systems (BIS2007), Poznan (Poland), pp. 133-146.
- [BMR06] Brehm, Nico; Marx Gómez, Jorge; Rautenstrauch, Claus: An ERP-solution based on Web-Services and Peer-to-Peer-Networks for Small and Medium Enterprises, International Journal of Information Systems and Change Management (IJISCM), Volume 1 - Issue 1- (2006), pp. 99-111.
- [BoS03] Boles, D., Schmees, M.: Kostenpflichtige Web Services. In: Uhr, W., Esswein, W., Schoop, E. (Hrsg.): Wirtschaftsinformatik 2003: Märkte - Medien - Mobilität. Physica-Verlag, Dresden (2003), pp. 385-403
- [DD⁺04] Dan, A., Davis, D., Kearney, R., Keller, A., King, R., Kuebler, D., Ludwig, H., Polan, M., Speitzer, M., Youssef, A.: Web services on demand: WSLA-driven automated management. IBM SYSTEMS JOURNAL, 43, (2004), pp.136-158
- [FTW07] Fischer, Thilo; Teufel, Marc; Wang, Dapeng: Java Web Services mit Apache Axis2, entwickler.press, (2007).
- [HM⁺06] Herden, S., Marx Gómez, J., Rautenstrauch, C., Zwanziger, A.: Software-Architekturen für das E-Business: Enterprise-Application-Integration mit Verteilten Systemen Springer, Berlin et al. (2006)
- [HöW02] Höß, O., Weisbecker, A.: Konzeption eines Repositories zur Unterstützung der Wiederverwendung von Software-Komponenten. In: Turowski, K. (Hrsg.): 4. Workshop Komponentenorientierte betriebliche Anwendungssysteme (WKBA 4). Augsburg (2002), pp. 75-85
- [Krü06] Krüger, Lars: Von der Fachkomponente zum Web Service, Otto-von-Guericke-Universität Magdeburg, Diplomarbeit, (2006).
- [LSE03] Lamanna, D.D., Skene, J., Emmerich, W.: SLAng: A language for defining Service Level Agreements. 9th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS'03). IEEE-CS Press, San Juan, Puerto Rico (2003), pp. 100-106
- [Mar04] Martin, David; et al.: OWL-S – Semantic Markup for Web Services, World Wide Web Consortium, 2004, URL: <http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/>.
- [Mey97] Meyers Lexikonredaktion: Schülerduden Informatik, Bibliographisches Institut & F.A. Brockhaus AG, Mannheim, (1997).
- [OvT05] Overage, S., Thomas, P.: WS-Specification: Ein Spezifikationsrahmen zur Beschreibung von Web-Services auf Basis des UDDI-Standards. In: Ferstl, O.K., Sinz, E.J., Eckert, C., Isselhorst, T. (Hrsg.): Wirtschaftsinformatik 2005: eEconomy - eGovernment - eSociety. Physica-Verlag Bamberg (2005), pp. 1539-1558
- [Ove06] Overage, S.: Vereinheitlichte Spezifikation von Komponenten: Grundlagen, UnSCom Spezifikationsrahmen und Anwendung. Wirtschaftswissenschaftliche Fakultät, Dissertation. Universität Augsburg (2006)
- [Rau07] Rautenstrauch, C.: Architekten, Landschaftspfleger und Kulturingenieure - neue Aufgabenfelder für Wirtschaftsinformatiker. In: Rautenstrauch, C. (Hrsg.): Die Zukunft der Anwendungssoftware - die Anwendungssoftware der Zukunft. Shaker Verlag, Aachen (2007) pp. 1-10
- [SDR03] Schmietendorf, Andreas; Dumke, Reiner; Reitz, Daniel: Erfahrungen im Umgang mit der Spezifikation von Web Services. In: Turowski, Klaus (Hrsg.): Tagungsband 4 - Workshop Modellierung und Spezifikation von Fachkomponenten, (2003).
- [Tia05] Tian, M.: QoS integration in Web services with the WS-QoS framework. Fachbereich Mathematik und Informatik, Dissertation. Freie Universität, Berlin (2005)