



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Seminarausarbeitung

Ralf Kruse

SOA: Service Komposition

Ralf Kruse
SOA: Service Kompositon

Seminararbeit eingereicht im Rahmen der Seminararbeit-
prüfung
im Studiengang Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Professor : Prof. Kai von Luck

Abgegeben am 31. Juli 2007

Inhaltsverzeichnis

Abbildungsverzeichnis	4
1 Einführung	5
1.1 Aufbau der Arbeit	5
2 SOA und Workflows	6
2.1 Service Orientierte Architekturen	6
2.1.1 SOA in Webservices	7
2.2 Workflow	8
3 BPEL4WS	9
3.1 BPEL4People	10
3.2 Bewertung	11
4 Service Komposition	12
4.1 Dynamische Service Komposition	13
5 Fazit mit Ausblick	16
5.1 Ausblick auf Folgeveranstaltungen	16
Literaturverzeichnis	17

Abbildungsverzeichnis

2.1	SOA: Servicezugriff	6
2.2	SOA: Referenzarchitektur	7
2.3	WFMC: Workflow Referenz Architektur	8
3.1	BPEL4WS Beispiel	9
3.2	BPEL: Hello World	10
4.1	Service Komposition	12
4.2	Beispiel einer semantischen Spezifikation von Diensten	13
4.3	Ablauf der dynamischen Service Komposition	14
4.4	Service Komposition mit Constraints und Templates	15

1 Einführung

Von jeher wird durch IT-Unterstützung versucht Prozesse effizienter zu gestalten. In diesem Zusammenhang wurden im Laufe der Zeit immer größere und komplexere Abläufe informationstechnisch umgesetzt. Technologien zur Umsetzung der Prozesse wechselten. Daraus entstand in vielen Unternehmen eine heterogene Anwendungslandschaft, in der die unterschiedlichen Systeme schwer miteinander interagieren können. Die Verkürzung von Entwicklungszyklen, kontinuierliche Verbesserungsprozesse und Business-Reengineering machen verlangen nach einem hohen Maß an Flexibilität in der Prozessgestaltung. In serviceorientierten Architekturen (SOA) bilden Funktionalität auf Dienste ab, bieten einen einheitlichen Zugriff auf diese Services an und schaffen so eine Integrationsmöglichkeit von Alt-systemen. Diese Dienste können zu ganzen Workflow komponiert werden. Aktuell geschieht dies statisch mit Sprachen wie BPEL4WS, die auf einer abstrakteren Ebene programmatisch die komplexeren Services zusammensetzen und so in der Prozessgestaltung ein großes Maß an Flexibilität schaffen.

Eine dynamischere Komposition von Diensten kann in Umgebungen mit wechselndem Dienstangebot unterstützen. Würde es beispielsweise in einem Unternehmen für eine Aufgabe diverse Dienste zu Lösung einer Aufgabe geben, könnte zur Laufzeit der kostengünstigste ausgewählt werden. Durch ein höheres Maß an Dynamik könnten aber auch neue Anwendungsgebiete sinnvoller erschlossen werden, wie mobile und ubiquitäre Anwendung.

1.1 Aufbau der Arbeit

In dem **zweiten Kapitel** werden die grundlegenden Konzepte einer **SOA** thematisiert, die Umsetzung dieser in **Webservices** und die Thematik der Workflow angesprochen. Das **dritte Kapitel** widmet sich der aktuell weit verbreiteten Sprache zu Service Komposition **BPEL4WS**. Die **Service Komposition** mit ihrer bisherigen statischen Ausprägung und den Aspekten einer höheren Dynamisierung werden im **vierten Kapitel** vorgestellt. Abschließend wird im **fünften Kapitel** ein **Resümee** gezogen und ein kurzer **Ausblick auf die Folgeveranstaltungen** gegeben.

2 SOA und Workflows

„Eine einheitliche und allgemein anerkannte Definition von SOA existiert bis heute noch nicht. Die Analysten, wie beispielsweise Gartner Research, Forrester Research und Experton-Group (vormals Met-Group), sowie die großen Hersteller IBM, Microsoft, Oracle und SAP gehen jedoch von ähnlichen Vorstellung des Aufbaus und des Nutzens von SOA aus.“ [Liebhart \(2007\)](#)

Das komponieren von Diensten findet in dienstorientierten Architekturen (**S**ervice-**o**rientierte Architekturen) statt. Dienste werden zusammengefügt bis hin zu ganzen Workflows. Auch wenn es keine einheitliche Definition einer SOA gibt, so gibt es zwischen den marktführenden Herstellern einen breiten Konsens über die Grundsätze. Nachfolgend wird die Terminologie der SOAs näher beleuchtet. Vor dienstorientierten Architekturen gab es in den 1990ern einen Focus auf Workflows. Die Prinzipien der Workflow-Management-Architekturen sind SOA eingeflossen und werden nachfolgend in einem Abschnitt dargestellt.

2.1 Service Orientierte Architekturen

„Das wichtigste Element einer SOA ist der Service als standardisierte Darstellung von Funktionalität. Service Oriented Computing (SOC) ist ein Paradigma, das Services (Dienste) als fundamentales Element für die Erstellung von Applikationen verwendet. Dieses fundamentale Element besteht aus einem Basisdienst, seiner Beschreibung sowie einer Basisoperationen (Publication, Discovery, Selection und Binding).“ [Liebhart \(2007\)](#)

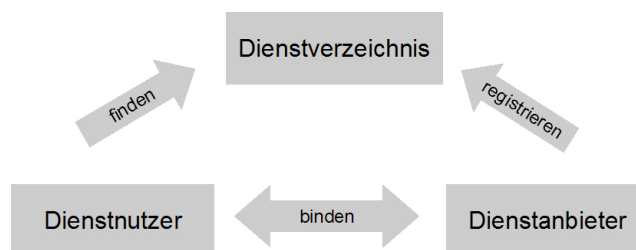


Abbildung 2.1: SOA: Servicezugriff

Die vorhergehende Aussage zeigt die zwei grundsätzlichen Aspekte einer SOA auf. Der Zugriff auf einen Dienst wird in Abbildung 2.1 verdeutlicht. Die implementierte Funktionalität wird über die Integrationsebene als Service standardisiert angeboten. Diese atomaren Dienste können zur die Orchestrationsebene zu neuen komplexeren Diensten, bis hin zu Workflows zusammengesetzt werden. Die atomaren und zusammengesetzten Dienste werden gemäß Abbildung 2.1 angeboten, die SOA-Schichten werden in Darstellung 2.2 verdeutlicht.

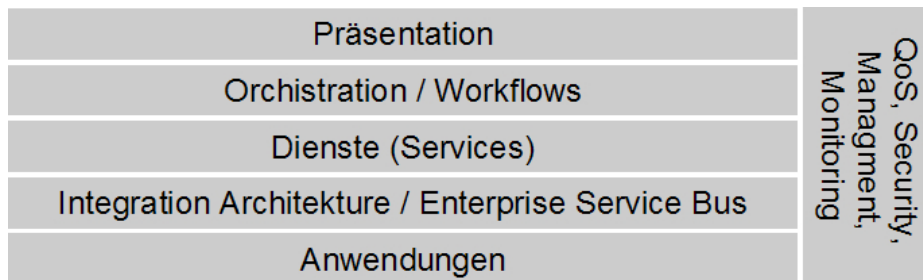


Abbildung 2.2: SOA: Referenzarchitektur

Für eine weiterführende Betrachtung sei auf [Dostal u. a. \(2005\)](#) verwiesen.

2.1.1 SOA in Webservices

In [Hoa03 \(2003\)](#) wird mit „SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents.“ verdeutlicht, dass SOA ein technologieunabhängiger Archtekturstil sind. Eine Technologie zur Umsetzung sind Webservices.

Web-Services sind eine mögliche Implementation des Konzeptes der service-orientierten Architekturen. Wie in dem vorhergehenden Kapitel, sind Web-Services nicht mit einer SOA gleichzusetzen, da Web-Services nur eine mögliche Implementierung von einer SOA sind.

In der Webservice Architektur sind die Dienste Web-Services, die Orchestration von Workflows findet mit BPEL statt. Die Dienste werden mit UDDI veröffentlicht.

In [Liebhart \(2007\)](#) wird die Umsetzung einer SOA in Webserivces weiter ausgeführt

2.2 Workflow

In der service-orientierten Architektur werden Dienste komponiert, bis hin zu ganzen Workflow. Ein Workflow ist die informationstechnische Umsetzung eines Geschäftsprozesses. Aktivitäten, die ein (Teil-)Ziel eines Unternehmens darstellen werden als Geschäftsprozess.

Die in Kapitel 1 dargestellten Anforderung an eine flexible Gestaltung von Geschäftsprozessen wurde mit **Workflow-Management-Systemen** Rechnung getragen. In den 1990er Jahren entstand die in [WFMCReferenz \(1995\)](#) beschriebene Referenzarchitektur. Die Grundzüge dieser Referenzarchitektur sind durch den Focus auf Prozessorientierung auch in die SOA eingeflossen. Nach der Terminologie Referenzarchitektur ist eine Prozessdefinition eine maschinelle Darstellung eines Workflows. Die Workflow-Engine instanziiert und steuert die Ausführung einer Prozessdefinition.

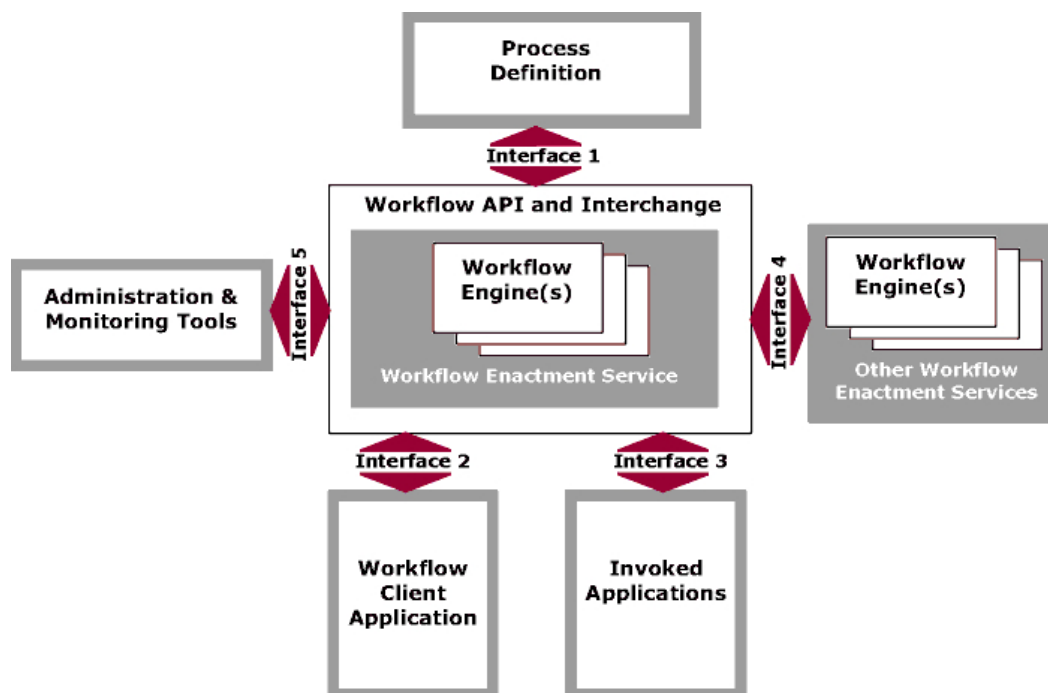


Abbildung 2.3: WFM-Referenzarchitektur
Abbildung aus [WFMCReferenz \(1995\)](#)

3 BPEL4WS

„BPEL4WS ist - genau genommen - auch eine imperative Programmiersprache. Sie wurde mit dem Ziel entwickelt, Geschäftsprozesse mit Hilfe von standardisierten und semantisch festgelegten Sprachelementen zu modellieren. Der Programmcode wird als XML-Dokument dargestellt, der demjenigen XML-Schema folgt, das von der BPEL4WS-Arbeitsgruppe (insbesondere von IBM und Microsoft, aber auch Siebel, BEA und SAP) erarbeitet wurde.“ [Dostal u. a. \(2005\)](#)

Dieses Zitat zeigt, dass mit Business Process Execution Language for Web-Services(BPEL4WS) bestehende Webservices programmatisch zu neuen Diensten zusammengesetzt werden. Das XML-File ist die Prozessdefinition, welche durch eine BPEL-Engine (Workflow-Engine) als neuer Dienst angeboten wird. Abbildung 3.1 zeigt einen komponierten Dienst, auf den der Client zugreift, welcher selbst wieder auf Webservices zurückgreift. Die

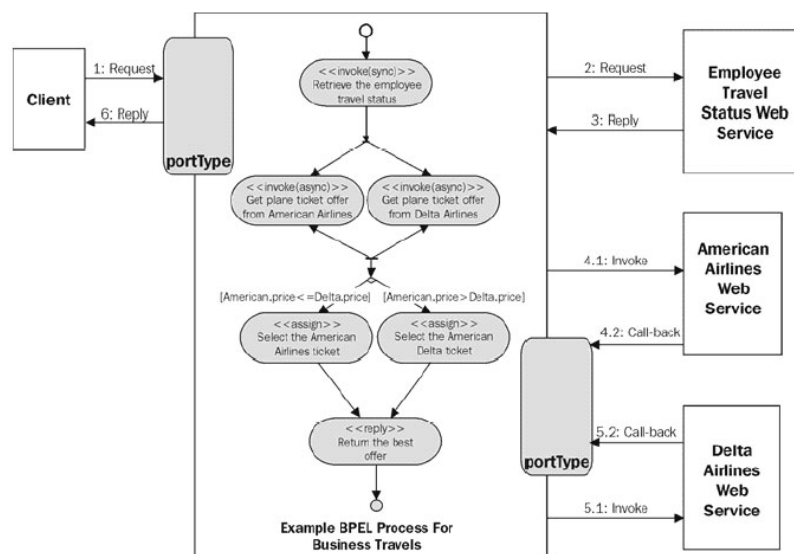


Abbildung 3.1: BPEL4WS Beispiel
Beispiel aus [Juric](#)

aktuelle Spezifikation ([bpelwsspec \(2006\)](#)) zeigt die Sprachelemente von BPEL4WS. Hervorzuheben ist, dass es konkrete Prozesse mit bereit gebundenen Diensten gibt und dass

es abstrakt Dienste gibt, an die beim Start der BPEL-Engine Services gebunden werden. Die XML-Darstellung von BPEL4WS kann anhand von Beispiel 3.2 verdeutlicht werden.

```

<process name="HelloWorld" >
  <!-- <partnerLinks> <variables> -->
  <sequence>
    <!-- receive the name of a person -->
    <receive operation="sayHello" partnerLink="caller"
      portType="tns:Greeter" variable="request"
      createInstance="yes" />
    <!-- compose a greeting phrase -->
    <assign>
      <copy>
        <from expression="concat('Hello, ',
          bpel:getVariableData('request', 'name'), '!')" />
        <to variable="response" part="greeting" />
      </copy>
    </assign>
    <!-- reply with the greeting -->
    <reply operation="sayHello" partnerLink="caller"
      portType="tns:Greeter"
      variable="response" />
  </sequence>
</process>

```

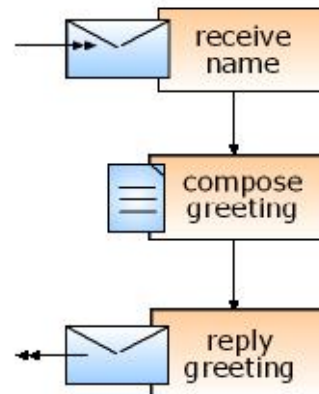


Abbildung 3.2: BPEL: Hello World
Beispiel aus [JBossHelloWorld](#)

3.1 BPEL4People

Da mit BPEL4WS nur Webservices eingebunden werden können, gibt es bisher keine Mittel zur Einbindung von menschlicher Interaktion. In der Regel gibt es aber einen hohen Anteil an unternehmerischen Abläufen, die genau dieses Element nachfragen. Aus diesem Grund resultieren Bestrebungen auch die Benutzerinteraktionen in BPEL einzubinden. "WS-BPEL Extension for People"(BPEL4People) ist eine solche Erweiterung, die unter anderem von SAP und IBM erstellt wurde. Für weitere Ausführungen sei auf [BPEL4PeopleSpec \(2007\)](#) verwiesen.

3.2 Bewertung

In BPEL4WS werden die komponierten Web-Services programmatisch beschrieben. Mit der grafischen Darstellung von BPEL-Abläufen soll die Übersichtlichkeit verbessert und die Prozessgestaltung auch Nicht-Entwicklern ermöglicht werden. Die Prozessbeschreibung wird mit steigender Komplexität unübersichtlich und damit schwerer wartbar. Dass Dienste nicht zur Laufzeit, sondern spätestens zum Start der Workflow-Engine gebunden werden, nimmt für einige Anwendungsfelder das benötigte Maß an Flexibilität.

In BPEL4WS werden mit Verzweigungen, Variablen, Schleifen und vielen weiteren Sprach-elementen Prozesse auf einem abstrakteren Level beschrieben. Sie verbessern damit den Status quo.

4 Service Komposition

„Eine Software-Komponente ist ein Software-Element, das konform zu einem Komponentenmodell ist und gemäß einem Composition Standard ohne Änderungen mit anderen Komponenten verknüpft und ausgeführt werden kann.“

Szyperski (1997)

Die Nutzung von Komponenten mindert die Komplexität der Entwicklung einer Anwendung und schafft ein höheres Mass an Wiederverwendbarkeit.

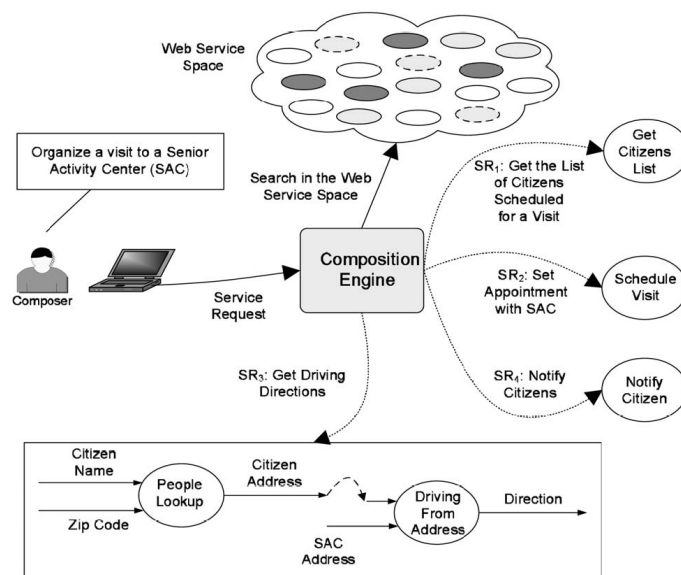


Abbildung 4.1: Service Komposition

Darstellung der Service Komposition aus Medjahed und Bouguettaya (2005)

In Liebhart (2007) heißt es: „Ein Service ist eine sich selbst beschreibende, offene Komponente, die eine schnelle und kostengünstige Zusammenstellung von verteilten Applikationen ermöglicht“. Unter Service Komposition wird das Zusammensetzen von bestehenden Diensten zu komplexeren Diensten, beispielsweise durch BPEL4WS, verstanden. In BPEL4WS werden die konkret genutzten Webservices in der Regel bei der Implementierung, spätestens beim Starten des BPEL-Servers gebunden. BPEL ist eine imperative Programmiersprache, die eine programmatisches und abstraktes Komponieren von Diensten ermöglicht. Bei

der statischen Komposition von Diensten wird i. d. R. beim Design, spätestens aber beim Start einer Anwendung ein konkreter Dienst gebunden.

4.1 Dynamische Service Komposition

Unter dynamischer Service Komposition versteht man, dass Dienste zu Laufzeit an einen komponierten Dienst gebunden werden können.

Diese Tatsache ermöglicht erst die reaktive Service Komposition, die mit ihren drei Phasen in der Abbildung 4.3 aus Kuroпка u. a. (2006) dargestellt wird. Bei der Erstellung eines Ablaufplans wird auf die semantische Spezifikation der Dienste zurückgegriffen. Für eine korrekte Interpretation eine speziell aufgestellte Ontologie genutzt. Die Abbildung 4.2 zeigt ein Beispiel für die semantische Spezifikation von Diensten.

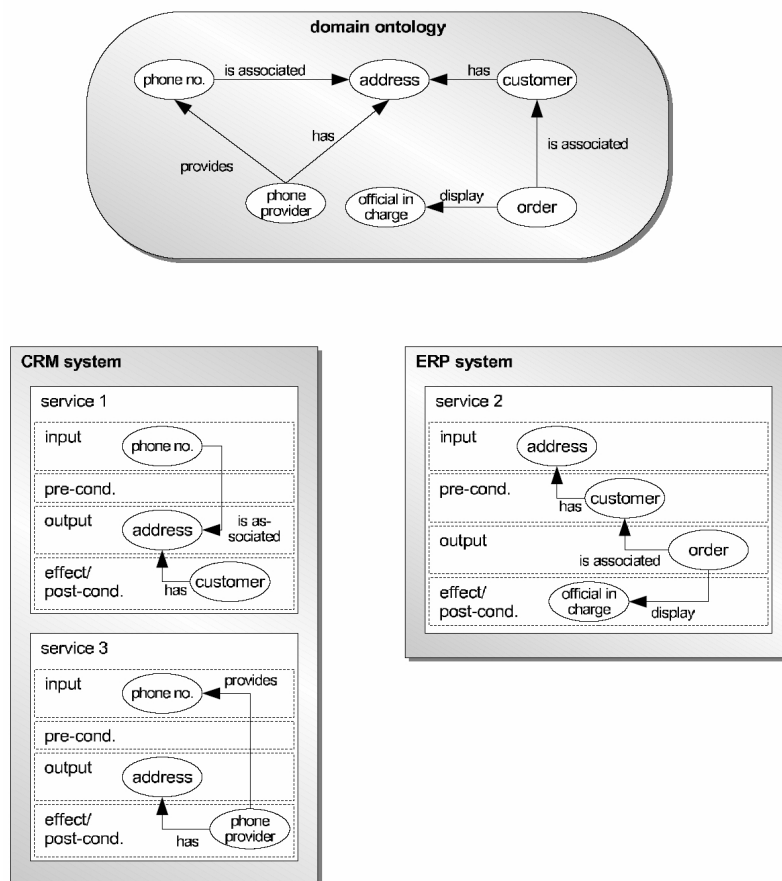


Abbildung 4.2: Beispiel einer semantischen Spezifikation von Diensten
Abbildung aus Kuroпка u. a. (2006)

Aus einer Anfrage wird zuerst ein Plan erstellt, an diese Plan werden konkrete Dienste gebunden und abschließend wird dieser konkrete Prozess ausgeführt. In der statischen Komposition wird ein konkreter Plan bereits beim Design festgeschrieben, bzw. beim Start der Laufzeitumgebung (z.B. BPEL-Engine) werden konkrete Dienste gebunden.

[McIlraith und Fadel \(2002\)](#) diskutieren wie sehr komplex die Erzeugung eines Plans ist. Gerade bei der Erstellung von dynamischen Komposition von Diensten auf Abruf kann dies hinderlich sein, bzw. bei einer grösseren Anzahl von Diensten nicht praktikabel sein. Aus diesem Grund gibt es verschiedene Ansätze die Komplexität zu minimieren. Ein Ansatz, dass die Bereitstellung von abstrakten Plänen oder Templates, welche anstelle der zu komplexen Planerzeugung genutzt werden.

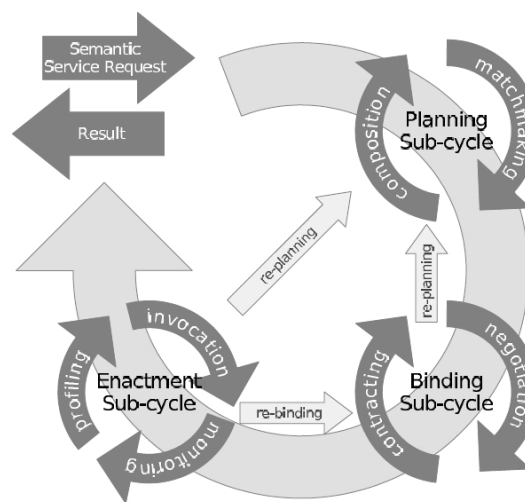


Abbildung 4.3: Ablauf der dynamischen Service Komposition
Abbildung aus [Kuropka u. a. \(2006\)](#)

Ein Überblick über verschiedene dynamische Service Kompositionsverfahren geben [Alamri und Saddik \(2006\)](#) und [Dustdar und Schreiner \(2005\)](#). [Kuropka u. a. \(2006\)](#) beschreibt den Lösungsansatz durch Zuhilfenahme von Semantic Web zur semantischen Beschreibung der Dienste und Planning wird. [Srivastava und Koehler \(2003\)](#) stellt die eine BPEL4WS Lösung einer Lösung mit Semantic Web gegenüber. Ein semi-automatischer Ansatz (Abbildung 4.4) wird in [Su u. a. \(2003\)](#) beleuchtet.

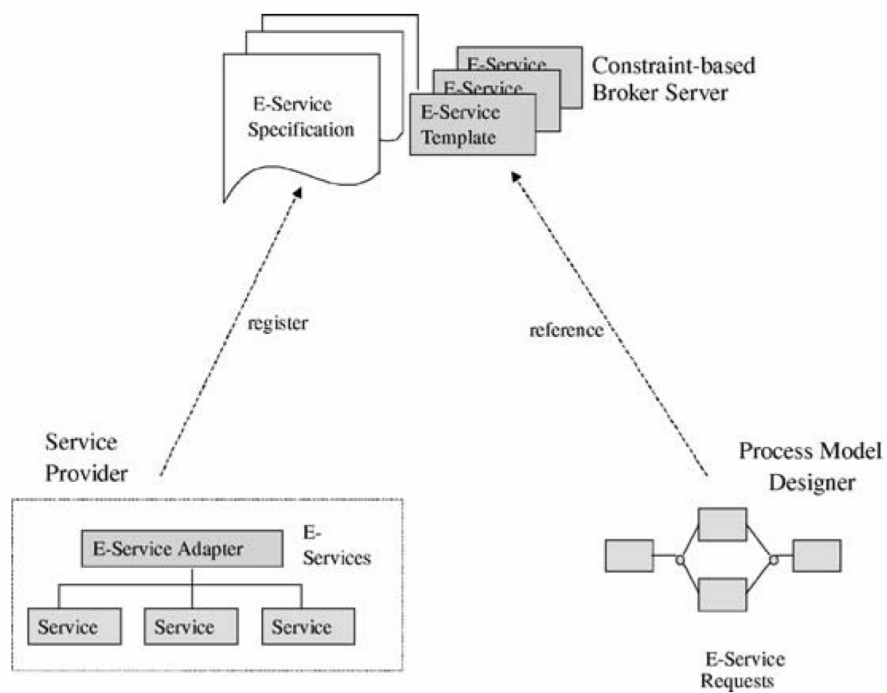


Abbildung 4.4: Service Komposition mit Constraints und Templates
aus [Su u. a. \(2003\)](#)

5 Fazit mit Ausblick

Das Konzept der SOA wird von allen marktführenden Softwareherstellern unterstützt bzw. vorangetrieben. Auch wenn es bis heute keine absolut einheitliche Definition einer SOA gibt, so findet das Konzept immer stärkere Anwendung in der Praxis.

Eine dynamische Komposition schafft nicht nur die Möglichkeit flexibel Services austauschbar in Unternehmen einzubinden, sondern eröffnet eine sinnvolle Nutzung in neuen Anwendungsfeldern mit veränderndem Dienstangebot. Das dynamische Zusammenfügen von Diensten ist anspruchsvoll, bisher kaum nutzungsfähig. Es wurde auch auf ältere Lösungskonzepte aus den Bereichen der Logik und der künstlichen Intelligenz zurückgegriffen, wie Verifikation, Constraints und Planning. Insgesamt handelt es sich um einen spannenden Themenbereich, der viele Themen zur näheren Betrachtung anbietet.

5.1 Ausblick auf Folgeveranstaltungen

Der Aufbau einer SOA mit der Betrachtung der aktuellen Umsetzung interessiert mich besonders. Inzwischen sind auch Open-Source Umsetzungen der Architektur vorhanden, beispielsweise mit Apache ServiceMix. Es gilt aber zu klären, ob der Aufbau einer solchen Infrastruktur nicht zu aufwendig ist im Verhältnis zum Erkenntnisgewinn.

Die Fokussierung auf die dynamische Komposition von Diensten könnte mir eine tiefer gehende Betrachtung dieses anspruchsvollen Themenkomplexes ermöglichen. Die semi-automatischen Lösungsverfahren mit abstrakten Plänen für eine prototypische Betrachtung bieten sich an.

Literaturverzeichnis

- [JBossHelloWorld] : *Hello World Example*. – URL <http://docs.jboss.com/jbpm/bpel/tutorial.hello.html>
- [WfmcReferenz 1995] : *The Workflow Reference Model*. Workflow Management Coalition, 1995. – URL <http://www.wfmc.org/standards/docs/tc003v11.pdf>
- [Hoa03 2003] : *What Is Service-Oriented Architecture*. 2003. – URL <http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html>
- [bpelwsspec 2006] : *Web Services Business Process Execution Language Version 2.0*. 2006. – URL http://www.oasis-open.org/apps/group_public/download.php/18714/wsbpel-specification-draft-May17.htm
- [BPEL4PeopleSpec 2007] : *WS-BPEL Extension for People (BPEL4People), Version 1.0*. 2007. – URL http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-bpel4people/BPEL4People_v1.pdf
- [Alamri und Saddik 2006] ALAMRI, Atif ; SADDIK, Mohamad Eid Abdulmotaleb E.: Classification of the state-of-the-art dynamic web services composition techniques. In: *International Journal of Web and Grid Services* 2 (2006), S. 148–166
- [Casati u. a. 2000] CASATI, F. ; ILNICKI, S. ; LI-JIE, Jin ; KRISHNAMOORTHY, V. ; SHAN, Ming-Chien: eFlow: a platform for developing and managing composite e-services. In: *Proc. Academia/ Industry Working Conference on Research Challenges*, apr 2000, S. 341–348
- [Channa u. a. 2005] CHANNA, Nizamuddin ; LI, Shanping ; SHAIKH, Abdul W. ; FU, Xiangjun: Constraint Satisfaction in Dynamic Web Service Composition. In: *DEXA Workshops*, IEEE Computer Society, 2005, S. 658–664
- [Dostal u. a. 2005] DOSTAL, Wolfgang ; JECKLE, Mario ; MELZER, Ingo: Service-orientierte Architekturen mit Web Services. *Konzepte - Standards - Praxis*. 1 (2005). ISBN 3827414571

- [Dustdar und Schreiner 2005] DUSTDAR, Schahram ; SCHREINER, Wolfgang: A Survey on Web services Composition. (2005). – URL http://scholar.google.com/url?sa=U&q=http://www.infosys.tuwien.ac.at/Staff/sd/papers/A%2520survey%2520on%2520web%2520services%2520composition_Dustdar_Schreiner_inPress.pdf
- [Gamm 2007] GAMM, Stefanie: *Mobile Prozesse: Kontextsensitive Service-Komposition*, Hochschule für Angewandte Wissenschaften Hamburg, Seminararbeit, 2007. – URL <http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master06-07/gamm/report.pdf>
- [Juric] JURIC, Matjaz B.: *A Hands-on Introduction to BPEL*. – URL http://www.oracle.com/technology/pub/articles/matjaz_bpell.html
- [Kuroпка u. a. 2006] KUROPKA, Dominik ; BOG, Anja ; WESKE, Mathias: Semantic Enterprise Services Platform: Motivation, Potential, Functionality and Application Scenarios. In: *EDOC '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06)*. Washington, DC, USA : IEEE Computer Society, 2006, S. 253–264. – ISBN 0-7695-2558-X
- [Liebhart 2007] LIEBHART, Daniel: *SOA goes Real*. Carl Hanser Verlag, 2007. – ISBN 9783446410886
- [McIlraith und Son 2002] MCILRAITH, S. ; SON, T.: Adapting Golog for composition of semantic Web services. In: *Proc. 8 International Conference on Principles of Knowledge Representation and Reasoning, 2002*, S. 482–496
- [McIlraith und Fadel 2002] MCILRAITH, Sheila A. ; FADEL, Ronald: Planning with complex actions. In: BENFERHAT, Salem (Hrsg.) ; GIUNCHIGLIA, Enrico (Hrsg.): *NMR*, 2002, S. 356–364
- [Medjahed und Bouguettaya 2005] MEDJAHED, Member-Brahim ; BOUGUETTAYA, Senior Member-Athman: A Multilevel Composability Model for Semantic Web Services. In: *IEEE Transactions on Knowledge and Data Engineering* 17 (2005), Nr. 7, S. 954–968. – ISSN 1041-4347
- [Müller 2004] MÜLLER, Joachim: *Workflow-based Integration*. Springer, September 2004. – ISBN 3540204393
- [Sirin u. a. 2004] SIRIN, E. ; PARSIA, B. ; D.WU ; HENDLER, J. ; NAU, D.: HTN Planning for Web Service Composition using SHOP2. In: *Web Semantics* 1 (2004), Nr. 4, S. 377–396
- [Srivastava und Koehler 2003] SRIVASTAVA, Biplav ; KOEHLER, Jana: Web Service Composition - Current Solutions and Open Problems. In: *ICAPS '03: Workshop on Planning for Web Services*, 2003

[Su u. a. 2003] SU, Stanley Y. W. ; MENG, Jie ; KRITHIVASAN, Raja ; DEGWEKAR, Seema ; HELAL, Sumi: Dynamic Inter-Enterprise Workflow Management in a Constraint-Based E-Service Infrastructure. In: *Electronic Commerce Research* 3 (2003), Nr. 1-2, S. 9–24

[Szyperski 1997] SZYPERSKI, Clemens: *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley Professional, December 1997. – URL <http://www.amazon.ca/exec/obidos/redirect?tag=citeulike04-20{\&}path=ASIN/0201178885>. – ISBN 0201178885

Alle Verweise auf Quellen im Internet wurden am 31.07.2007 auf ihre Aktualität überprüft.