

Service Orientierte Architektur Übersicht und Einordnung

Inhaltsverzeichnis

Einführung	3
SOA Versprechungen	3
Das SOA Siegel	4
Eine erste Näherung	4
Die Lego Metapher	4
Entwickler-Viewpoint auf SOA	5
Die Evolution zu SOA	5
Evolution der Transportschicht	5
Evolution der Prozess-Integration	6
Vertiefung SOA Design Ansatz	7
Zusammenhang Services, Komponenten, Objekte	7
Klassifizierung von SOA Services	8
Technologische Abgrenzungen	9
Abgrenzung SOA und Web Services	9
Abgrenzung SOA und CORBA	10
Abgrenzung SOA und J2EE	11
Business Process Management	11
SOA Blueprints	12
Fazit	13
Literaturverzeichnis	15

Einführung

Der Versuch sich einen Einblick zu verschaffen, was eine Service Orientierte Architektur (SOA) denn eigentlich ist, kann in einer großen Verwirrung enden, schnell stößt man bei der Recherche auf Widersprüche und Unschärfen. Wie bei IT „Hypes“ üblich, sparen Hersteller diverser Produkte nicht mit Versprechungen: eine SOA ändere alles, wäre eine Revolution, führe zu nie da gewesenen Einsparungen. Vielfältigsten Produkten wird dann auch gleich ein SOA-Stempel aufgedrückt. Dass eine SOA nicht auf einer Spezifikation beruht wie beispielsweise CORBA oder J2EE, führt neben den Marketingkampagnen zu zusätzlicher Intransparenz. Darüber hinaus erfolgt die Abgrenzung zu Technologien und Architekturen wie beispielsweise Web Services, CORBA und J2EE oftmals nicht oder willkürlich.

Diese Arbeit ist ein Versuch herauszukristallisieren, was unter einer SOA entsprechend der „herrschenden Meinung“ zu verstehen ist und zwar anhand expliziter Beschreibungen was SOA ist und was SOA eben nicht ist. Hierfür werden vielzählige Quellen ausgewertet die sich aus unterschiedlichen Richtungen dieser Fragestellung nähern. Der Leser wird auf Überlappungen und Wiederholungen stoßen; Dies ist beabsichtigt: die Überlappungen dürfen als Konsens für das aktuelle Verständnis einer SOA in der Fachwelt verstanden werden.

SOA Versprechungen

Beginnen wir zuerst mit den üblichen Versprechungen, die einem SOA Verständnis im Wege stehen.

Obwohl Gartner 1996 als erster den Begriff SOA prägte [Nati2003] und demnach als der Erfinder der SOA gilt, schreckt auch Gartner nicht davor zurück SOA als eine Revolution darzustellen, vergleichbar mit dem Wandel vom Mainframe zur Web-Anwendung: *„Service Orientation Catalyzes Latest Software Revolution“* [AbAn2004] und erweckt dazu noch den Eindruck man könne SOA ab dem Jahr 2005 kaufen: *„... you can purchase them (Anm.: SOA Business Applications) from business application vendors ...“*. In der deutschen Fachpresse ist die SOA das Allheilmittel für Rechenkapazität und Integrationsprobleme *„Rechenleistung kann nach Bedarf bereitgestellt werden, bei Änderungen der Geschäftsprozesse werden schnell die benötigten Anwendungen integriert“* [Comp2003]. Versteht sich von selbst, das mit einer SOA alles billiger wird: *„Zeit, Kosten und Komplexität von Integrationsprojekten lassen sich durch den Einsatz einer Service-orientierten Architektur deutlich reduzieren“* verspricht [CIO2004a]. Manche Autoren können den SOA Nutzen exakt quantifizieren: [PACo2004, S. 24] schildert, dass durch eine SOA die Entwicklungskosten um 40% sinken, [Seeb2004] weiß, dass sich mit einer SOA die Infrastrukturkosten gleich halbieren. [CIO2004b] berichtet, dass sich *„nur“* mit einer SOA das *„Echtzeitunternehmen“* realisieren lässt. [SAG2004] weiß auch genau wie, nämlich in drei Schritten, wobei der erste Schritt die Überführung der bestehenden Anwendungsarchitektur in eine SOA ist. Das ganze natürlich vollautomatisch: *„Die gewünschte Schnittstellenmethode wird von den leistungsstarken Wrapping-Werkzeugen von EntireX erzeugt. Sobald die Prozesse gekapselt sind, stehen sie als Service zur Verfügung“*. Es drängt sich der Eindruck auf, es gäbe so etwas wie eine *„1-click SOA“*.

Das SOA Siegel

Fakt ist, SOA ist in aller Munde. Dies erklärt auch warum sich allerlei Produkte mit einem SOA Siegel schmücken. Content Management Systeme werden als SOA basiert ausgewiesen [Abou2004], J2EE Applikationsserver stellen die Investitionssicherheit einer SOA sicher [Bea2004b], Enterprise Application Integration (EAI) Werkzeuge sind eine notwendige Voraussetzung für eine SOA [Atta2004, Seeb2004] und selbst „Oldies“ wie die RZ Automatisierungs- und Monitoringsoftware HP Openview werden als unabdingbar für den Aufbau einer SOA geschildert [HP2004].

Eine erste Näherung

Genauso wenig, wie die großzügigen Versprechungen einer kritischen Überprüfung standhalten, bestehen zwangsläufige Beziehungen zwischen einer SOA und der Vielzahl Produkte mit „SOA-Gütesiegel“. Von diesen Unschärfen in der Betrachtung befreit, soll nun eine erste Schilderung erfolgen, was eine SOA gemäss dem aktuellen Fachwelt Konsens ist.

Als erster erwähnte Gartner SOA 1996 in der Research Note SPA-401-068 „*Service-Oriented Architecture Scenario*“. Der einer SOA zugrundeliegende Gedanke ist dabei nicht technischer Natur. Es geht Gartner nicht darum, wie die Software Anwendung von morgen aussieht, vielmehr will Gartner durch eine SOA die Technik in den Hintergrund und Geschäftsprozesse in den Vordergrund stellen: „*Drawing a parallel with the airline industry, we began by focusing on what kinds of airplanes could be built. We now focus on what useful transportation services we can provide*“ [Dugg2002]. Zwangsläufig hat dies auch Auswirkungen auf die Technik, aber der technologische Wandel ist nur Mittel zum Zweck. [BuLa2003] stellt die These auf, dass eben diese Orientierung auf Geschäftsprozesse zu der Aufmerksamkeit geführt hat, die SOA derzeit faktisch zukommt.

SOA ist also ein Software Design Ansatz mit dem Grundgedanken, dass Geschäftsprozesse im Vordergrund stehen. Daher rührt auch der Begriff des „Echtzeitunternehmens“, denn eine SOA soll die flexible Anpassung von Geschäftsprozessen an die sich ständig wandelnden Umweltbedingungen der Unternehmens ermöglichen. Dies erfolgt durch eine Architektur grobkörniger Services die miteinander kommunizieren und flexibel zusammengestellt werden können [BuLa2003, TwPs2004]. Die flexiblen Möglichkeiten der Kommunikation und Zusammenstellung werden durch eine Standardisierung der Service Schnittstellen ermöglicht. Gartner redet in diesem Zusammenhang auch von einer „*Interface oriented Architecture*“ [Nati2003].

Die Lego Metapher

Zur Beschreibung des SOA Gedanken verwendet [Ullm2004] die Metapher eines Lego Baukastens. Der Autor schildert, wie man durch die Entwicklung lose gekoppelter Services einen Servicebaukasten erhält. Um die Bausteine zu etwas Ganzem zusammenfügen zu können, braucht es einen Ansatz wie dieses Zusammenfügen erfolgen kann, eine Art Lego-Brett, also die Lego Grundplatte auf der die Lego Bausteine einrasten. Die Lego-Bausteinen wiederum haben eine „standardisierte Schnittstelle“ damit sie miteinander verbunden werden können usw. SOA beschreibt der Autor als einen möglichst optimalen Entwurf wie Lego-Brett und Lego-Bausteinen aussehen sollen.

Entwickler-Viewpoint auf SOA

[Bien2004] beschreibt SOA von zwei Standpunkten aus, dem „*Manager-Viewpoint*“ und dem „*Entwickler-Viewpoint*“. Die bisherigen Schilderungen dürften den Manager-Viewpoint abdecken, wenden wir uns nun dem Entwickler-Viewpoint zu, also dem technischen Wandel der zwangsläufig aus dem Streben nach einer SOA resultiert.

[Bien2004] beschreibt SOA als eine Fortführung komponentenbasierter Architekturen. Komponenten unterscheiden sich von den SOA Services durch die Granularität, Services sind größer granuliert. Diese Abgrenzung wird uns bei der Abgrenzung SOA zu J2EE und CORBA Komponenten in den späteren Kapiteln hilfreich sein. Die gröbere Granularität von Services resultiert daraus, dass Services „*fachliche Transaktionen oder sogar Teile eines Geschäftsprozesses*“ realisieren. Gartner spricht von „*software Modules large enough to represent a complete business function*“ [Nati2003]. Komponentenbasierte Architekturen lassen sich nach [Bien2004] leicht in eine SOA überführen. In den nachfolgenden Kapiteln werden wir sehen, dass Komponenten auch Basis einer SOA sein können. Dieser mögliche, aber keinesfalls zwangsläufige Zusammenhang zwischen Komponenten und SOA führt zu solchen Unschärfen in der Diskussion, dass beispielsweise J2EE Applikationsserver notwendig seien für den Aufbau einer SOA. Abschließend sei bei dieser ersten Näherung an den Begriff SOA noch erwähnt, dass SOA aufgrund dessen, dass es sich um einen Design Ansatz handelt, vollständig technologie-unabhängig ist. Keinesfalls besteht z.B. ein zwangsläufiger Zusammenhang zwischen einer SOA und Webservices.

Die Evolution zu SOA

Mit einer Einordnung von SOA in die Entwicklung von IT Architekturen soll nun eine weitere Näherung an das Verständnis einer SOA erfolgen.

Evolution der Transportschicht

[TwPs2004] erläutert die Ursprünge von SOA anhand der Evolution der einer Integration zugrunde liegenden Transportschicht für Daten- und Nachrichtenströme, beginnend mit Punkt-zu-Punkt Verbindungen und einer damit verbundenen Vielfalt an Protokollen (die Anmerkung mit den Web-Services in der Abbildung sollte dabei überlesen werden, hier wird fälschlicherweise ein zwangsläufiger Zusammenhang zwischen einer SOA und Web-Services vorbereitet).

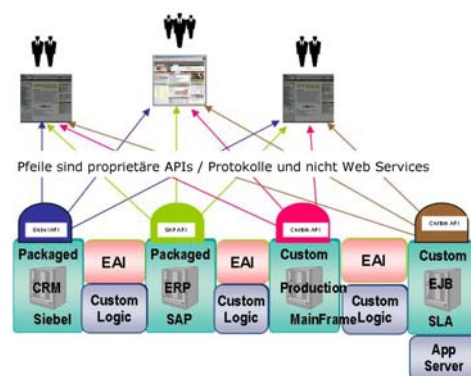


Abbildung 1 – Evolution Transportschicht Punkt-zu-Punkt [TwPs2004]

Als zweite Evolutionsstufe wird Middleware und der Message Bus dargestellt, resultierend in einer Reduktion von Schnittstellen durch Vermeidung von n:m Verbindungen.

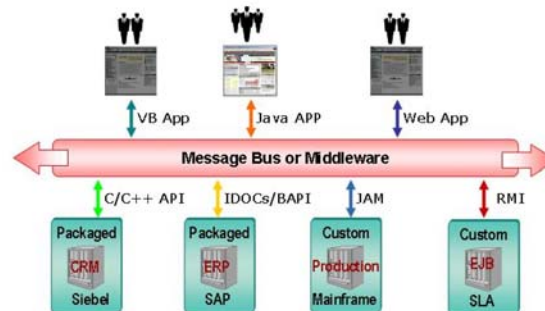


Abbildung 2 – Evolution Transportschicht Middleware [TwPs2004]

Anhand dieser Grafik kann man noch die Orientierung der Architektur am Datentransport erkennen die in der nächsten Evolutionsstufe durch eine Orientierung an Geschäftsprozessen abgelöst wird. Mit SOA wandelt sich das Bild von Applikationen die Nachrichten austauschen hin zu Geschäftsprozessen die durch eine Zusammenstellung von Services realisiert werden. Dies wird ermöglicht durch Standards die die in den Applikationen verborgenen Funktionalitäten als Services ansprechbar und auffindbar machen.

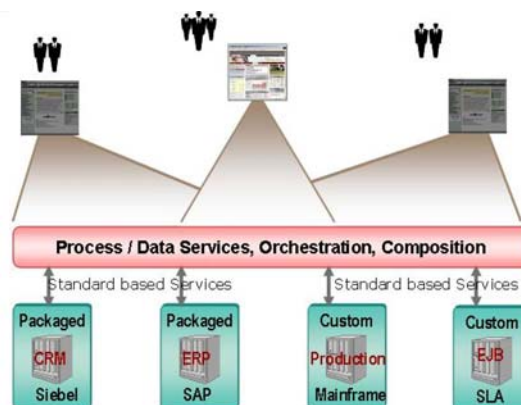


Abbildung 3 – Evolution Transportschicht SOA [TwPs2004]

Man kann der Abbildung auch gut entnehmen, dass dem Daten- und Nachrichtentransport bei einer SOA ein ganz wesentliche Rolle zukommt, die in der Regeln verteilten Services müssen ja schließlich miteinander kommunizieren können. Dies ist der Grund dafür, dass im Zusammenhang mit einer SOA häufig Message Orientierte Middleware (MOM) und Enterprise Application Integration (EAI) Werkzeuge erwähnt werden.

Evolution der Prozess-Integration

Die Veranschaulichung anhand der Transportschicht vermittelt eine Idee einer SOA, letztendlich hat aber auch diese Veranschaulichung Unschärfen. [Ecau2004] schildert ebenfalls eine Evolution und hebt von Beginn an auf die Integration von Prozessen ab:

Die Einführung von Client-/Server und Personal Computern in den 90er Jahren ermöglichte demnach einen „raschen Aufbau neuer komplexer Systeme, wobei die Integration

in unternehmensweite Prozess- und Datenmodelle zweitrangig behandelt wurde“. Ende der neunziger Jahre schließlich mussten durch die Umwälzungen des Internets neue Geschäftsprozesse, basierend auf bestehenden Funktionalitäten, geschaffen werden. Zur Problemlösung wurden EAI Systeme eingesetzt. Auch EAI führte jedoch nicht zu der gewünschten Flexibilität, erst Standards wie XML boten die Möglichkeit Geschäftsprozesse technologieunabhängig zu beschreiben, dadurch zu standardisieren und somit eine flexiblere Art der Integration zu schaffen. Dies führte dann zum Gedanken an eine SOA.

Vertiefung SOA Design Ansatz

Fügen wir die Beschreibungen der vorangehenden Kapitel zusammen, so festigt sich folgendes Bild:

- In einer SOA stehen Geschäftsprozesse im Vordergrund
- Geschäftsprozesse können in einer SOA durch die Zusammenstellung von Services abgebildet werden, diese Zusammenstellung erfolgt mit standardisierten Darstellungsformen.
- SOA Service Schnittstellen werden durch eine standardisierte Darstellung beschrieben.
- SOA Services sind grob granuliert, können durch Altanwendungen oder Web Services oder Komponenten implementiert sein oder einer Zusammenstellung dieser Bausteine.
- Eine SOA ist „geschwätzig“, Technologien für Daten- und Nachrichtenaustausch werden für den Aufbau einer SOA eingesetzt.

Zusammenhang Services, Komponenten, Objekte

Nach [Bien2004] beschäftigt sich SOA mit *„Entwurf, Implementierung, Deployment, Wiederverwendung, vereinfachter Verwendung, und Suche nach Services“*. Danach schildert [Bien2004] *„Services vereinfachen den Zugriff auf die Logik eines Objekts, einer Komponente oder einer Anwendung“*. Diese Zusammenhänge werden nochmals sehr gut in einer Grafik veranschaulicht:

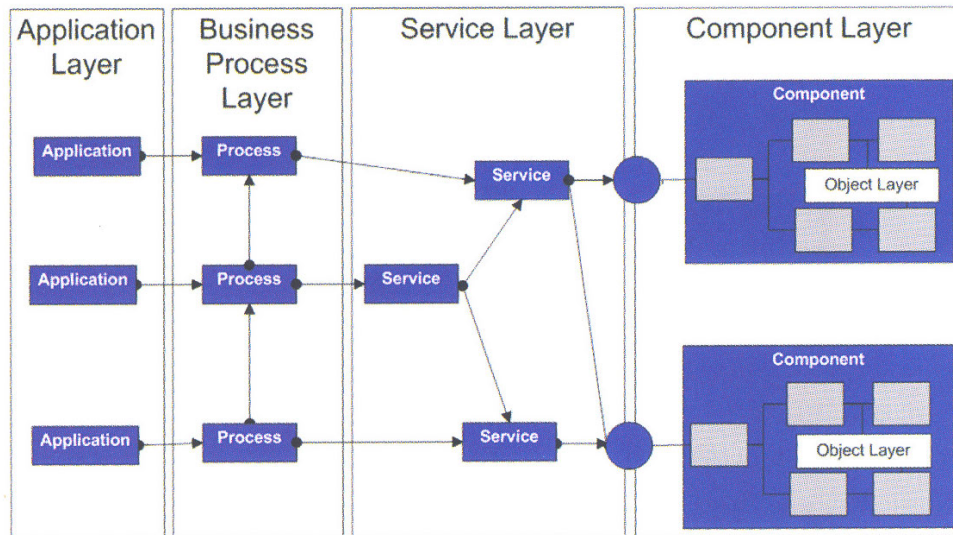


Abbildung 4 – Zusammenhang Services, Komponenten, Objekte [Bien2004]

Soweit sind dies keine Neuigkeiten, sondern eine Bestätigung des bisher Besprochenen. Betrachten wir nun als nächstes die SOA Services näher. [Bien2004] stellt hierfür Klassifizierungen von Services vor.

Klassifizierung von SOA Services

Zum Beginn eine Klassifizierung wie sie aus den SOA Blueprints [WiHa2004b] stammt (die SOA Blueprints werden später nochmals gesondert vorgestellt):

- Component Services: Atomare Services die von anderen Services unabhängig sind
- Composite Services: Koordinieren Component Services und stellen eine höheren Abstraktionslevel dar

Eine weitere Klassifizierung aus den SOA Blueprints [WiHa2004b]:

- Data Service: Service der die Persistenzschicht kapselt
- Service Broker: Service der Anfragen an die zuständigen Services weiterleitet
- Workflow Service: Service der feiner granulierte Services zu einem Workflow zusammenfasst.
- Synchronous/Asynchronous Service: Unterscheidung von Services nach Art ihrer Kommunikation.
- Serial, Parallel Orchestration: Unter „Orchestration“ versteht man die Zusammenstellung von Services zu einem Geschäftsprozess. Werden die Services seriell angeordnet um den Geschäftsprozess abzubilden, dann handelt es sich um eine Serial-Orchestration. Arbeiten die Services parallel, spricht man von einer Parallel-Orchestration.
- Compensating Services: Fehlertolerante Services, die z.B. Transaktionsfähig sind.

- Publish/Subscribe Services: Services die mit Publish-/Subscribe Verfahren kommunizieren.

Häufig findet man auch eine Unterscheidung von Services anhand eines Rollenkonzepts. Hier werden Services unterschieden in Service-Provider (implementiert einen SOA Service), Service-Consumer (benutzt einen SOA Service) und Service-Broker (Veröffentlicht einen SOA Service, steht für die Suche nach einem SOA Service zur Verfügung), siehe z.B. [TwPs2004].

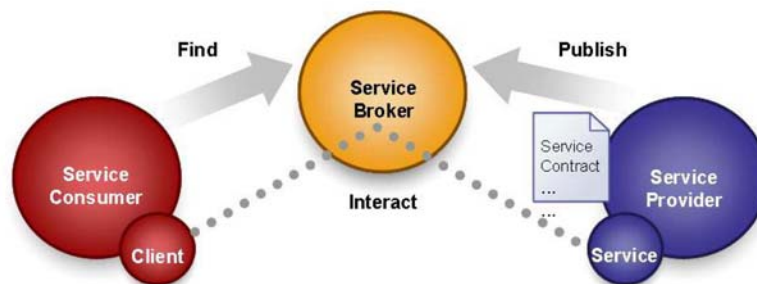


Abbildung 5 – Service-Consumer, -Provider, -Broker [TwPs2004]

Technologische Abgrenzungen

Der SOA Design Ansatz ist technologieunabhängig, kann also durch die Wahl unterschiedlicher Technologien umgesetzt werden. An dieser Stelle soll nun angerissen werden welche Technologien oft im Zusammenhang mit SOA genannt werden. Dabei wird erneut auf häufige Unschärfen der Diskussion hingewiesen.

Abgrenzung SOA und Web Services

Web Services sind lose gekoppelt und mit der Web Services Description Language (WSDL) existiert darüber hinaus eine standardisierte Schnittstellenbeschreibung von Webservices [Wiki2004]. Gartner führt dazu aus: „*Web services' WSDL is an SOA-suitable interface definition standard: that is where Web services and SOA fundamentally connect*“. Halten wir also fest, dass sich Web Services gut zum Aufbau einer SOA eignen.

Im Umkehrschluss jedoch führt die Nutzung von Web Services keinesfalls automatisch zu einer SOA. Erst eine entsprechend grobe Granularität, Wiederfindbarkeit, Ausrichtung an Geschäftsprozessen und die Möglichkeit die Web Services so zusammenstellen (orchestrieren) zu können, dass ein Geschäftsprozess abgebildet wird, führt zu einer SOA.

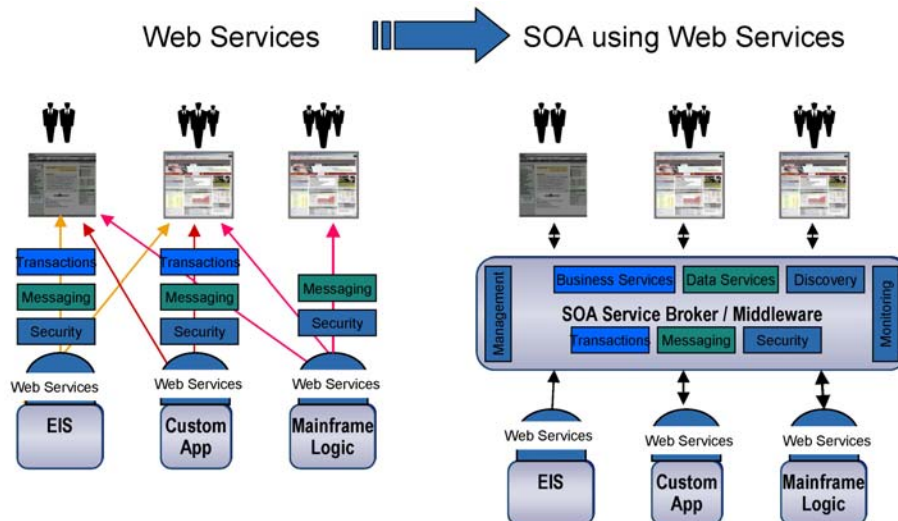


Abbildung 6 – Die Nutzung von Web-services führt nicht zwangsläufig zu einer SOA [TwPs2004]

Diese Unterscheidung wird oftmals nicht getroffen und führt zu Verwirrung. Der bereits erwähnte Gartner Artikel [AbAn2004] der SOA als Revolution bezeichnet, nennt SOA und Web Services in einem Zug. [KrJa2004] schreibt „*Web Services basieren auf einer Service Oriented Architecture*“ und weiter „*SOA = Modell bei dem drei verschiedene Rollen unterschiedliche Interaktionen auf einem Web Service durchführen*“. [CIO2004a] schreibt „*Die wichtigste Neuerung bei SOA sind bewährte Web-Service-Standards*“.

Die Verwendung von Web Services zum Aufbau einer SOA muss anhand des konkreten Einsatzgebiets abgewägt werden, denn Web Services sind in Bezug auf Performanz teuer. Web Services nutzen für den Datentransport das HTTP Protokoll, binäre Protokolle wie sie Java-RMI oder CORBA zu Grunde liegen sind schneller: das zu transportierende Datenvolumen ist geringer, die Nachrichten müssen nicht aufwendig geparkt werden [Ullm2004].

Abgrenzung SOA und CORBA

[Ullm2004] beschäftigt sich etwas detaillierter mit der Abgrenzung CORBA / SOA, speziell mit der Frage, warum CORBA nicht sehr häufig als geeignete Technik für den Aufbau einer SOA genannt wird, obwohl dies vom technischen Standpunkt aus möglich wäre. Der Autor geht davon aus, dass Service-Provider, -Requestor und -Broker ebenso mit CORBA implementiert werden können. Mit der Interface Definition Language (IDL) bietet CORBA beispielsweise eine standardisierte Schnittstellenbeschreibungs-Sprache, mit dem Interface Repository steht ein Schnittstellen Repository zur Verfügung und durch das IDL Konzept sind CORBA Objekte sogar sprachunabhängig.

Als Hauptgründe für die geringe Nutzung von CORBA für den Aufbau einer SOA nennt [Ullm2004] folgende Gründe:

- Die Verwendung des neuen CORBA-spezifischen Protokolls IIOP anstatt dem bereits etablierten HTTP-Protokoll.

- Die bis zu CORBA 3 nicht erfolgte Standardisierung des IIOP Ports, was bei einer hohen Anzahl CORBA Kommunikationspartnern zu einer hohen Anzahl zu öffnender Ports in der Firewall führt.
- Die Nutzung eines binären Protokolls welches in der Firewall nicht gesniffet werden kann.
- Die unzureichende Spezifikation des CORBA Basic Object Adapters in den ersten CORBA Spezifikationen, was zu herstellerspezifischen Implementierungen und schließlich zu einer eingeschränkten Interoperabilität zwischen den Object Request Brokern (ORB) unterschiedlicher Hersteller führte.

Diese Nachteile kommen jedoch hauptsächlich in unternehmensübergreifenden Kommunikation zum Tragen weshalb innerhalb des Unternehmens eine CORBA Implementierung durchaus in Betracht gezogen werden kann. Vor allem wenn hohe Performanz und Sprachunabhängigkeit gefordert ist, scheint CORBA Vorteile gegenüber einer Web Service (Performanz) oder J2EE (Sprachunabhängigkeit) Implementierung zu besitzen.

Abgrenzung SOA und J2EE

Eingangs wurde bereits erwähnt, wie Java Objekte, Java Komponenten und SOA Services zusammenhängen können. Java Objekte bilden fein granulare Funktionalität ab, Java Komponenten werden als Java Enterprise Beans (Entity-, Session- oder Message Driven Bean) implementiert und bilden etwas höher granulierte (Teil-) Geschäftsprozesse ab. SOA Services wiederum decken die höchste Granularität ab, realisieren umfassende Geschäftsprozesse und können z.B. durch Zusammenstellung von Java Komponenten implementiert werden.

Für Entwicklung von Geschäftsprozessen durch das Zusammenstellen von SOA Services, die wiederum auf Java Komponenten zugreifen, sowie für die dafür notwendigen Schnittstellenbeschreibungen gibt es in Java keinen Standard. BEA jedoch hat ein Framework für die Implementierung SOA konformer Services mit Java oder Web Services entwickelt und der Apache Software Foundation übergeben. Dieses Open Source Projekt namens Beehive erweitert z.B. die Java Annotations, so dass die Schnittstellen der Java Enterprise Beans und Java Objekte mit Meta-Daten angereichert werden wie es für SOA Services unabdingbar ist. Darüber hinaus wird beim Service Design noch nicht entschieden ob die Services durch Java Komponenten oder Web Services implementiert werden. Diese Entscheidung fällt zu einem späteren Zeitpunkt und beide Wege werden durch Beehive unterstützt. Siehe dazu [Ullm2004] und [Bien2004]. Die populäre Java IDE eclipse unterstützt Beehive massiv, es ist absehbar, dass sich dieses Konzept zu einem de-facto Standard entwickelt.

Das Wiederauffinden von Services schließlich lässt sich mit Java Technologien umsetzen, wie beispielsweise dem Java Naming and Directory Interface (JNDI).

Business Process Management

Es war nun schon oft die Rede davon, dass Geschäftsprozesse durch eine Zusammenstellung von Services abgebildet oder implementiert werden. Diese Tätigkeit im Rahmen des

sogenannten Business Process Management (BPM) durchgeführt. BPM ist ebenso wie SOA ein nicht standardisierter Begriff der unterschiedlichst belegt wird. An dieser Stelle sei kurz eine BPM Definition aufgeführt, die den Begriff recht gut umfasst: *“Business Process Management umfasst also Methoden und Prozesse zur Erarbeitung und Beurteilung von Geschäftsprozessen, zum Design und zur Simulation an PC-gestützten Systemen, zur Implementierung in IT-Systeme sowie zur ständigen Kontrolle und der Anpassung von elektronischen Geschäftsprozessen”* [Kram2004].

Die Business Process Execution Language (BPEL) ist eine XML basierte, von OASIS standardisierte XML Notation zur Beschreibung von Geschäftsprozessen. Zahlreiche Werkzeughersteller setzen BPEL ein. Meistens wird ein Geschäftsprozess auf einer grafischen Oberfläche entworfen und als BPEL Dokument gespeichert. Die Werkzeuge generieren dann aus dem BPEL Dokument einen Web Service der als Composit-Service einzelne feiner granuliert, als Web Service implementierte Component-Services aufruft, die die eigentliche Funktionalität liefern. BPEL ist sprachabhängig, generiert werden ausschließlich Web Services. Aus diesen Grund wird BPEL oft auch BPEL4WS (BPEL for Web Services) oder WSBPEL (Web Services BPEL) genannt.

BEA und IBM haben die Business Process Execution Language for Java (BPELJ) entworfen. BPELJ reichert BPEL Dokumente um Java Code, den sogenannten Java Snippets, an um im BPEL Dokument Ablauflogik, Verzweigungen, Variablen-Initialisierungen, Web Service Initialisierungen und sogar Business Logik hinterlegen zu können. Hauptsächlich dient BPELJ jedoch dazu, mit BPEL beschriebene Geschäftsprozesse nicht nur durch Web Services, sondern auch durch J2EE Komponenten implementieren zu können [IBM2004].

SOA Blueprints

21 Firmen haben sich zusammengeschlossen um mit den SOA Blueprints die beispielhafte Implementierungen einer SOA für die imaginäre Firma GeneriCo zu entwickeln. Mit dieser Initiative werden hauptsächlich zwei Ziele verfolgt:

- Es soll eine Sammlung von best practises entstehen aus der sich jeder Interessierte bedienen kann.
- Software Hersteller können durch die beispielhafte Implementierung diese best practises zeigen wie Ihre Produkte für den Aufbau einer SOA verwendet werden können.

Bestandteil der SOA Blueprints sind fein granulare Funktionen wie eine Authentifizierung bis hin zur kompletten Human Resources (HR) Anwendung. Sie dazu auch [Ullm2004] und [WiHa2004a]. Auf der Architekturskizze von GeneriCo findet sich die oben erwähnte Service Klassifizierung wieder:

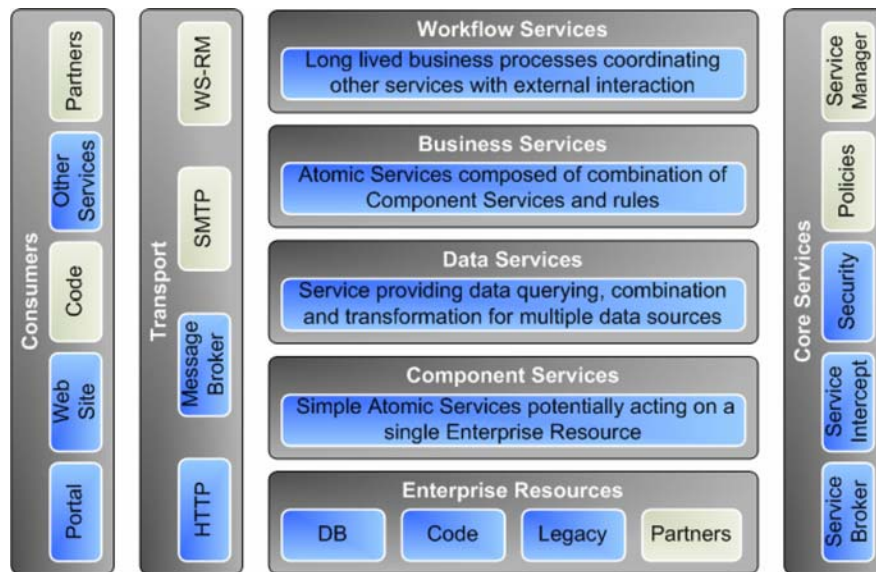


Abbildung 7 – GeneriCo Architektur [WiHa2004a]

Fazit

Abschließend soll nun zusammengefasst werden, was unter einer SOA gemäß den Quellen der Fachwelt derzeit verstanden wird. Hierfür wird der bereits geschilderte Zwischenstand mit den Erkenntnissen der letzten Kapitel angereichert:

- In einer SOA stehen Geschäftsprozesse im Vordergrund
- Geschäftsprozesse können in einer SOA durch die Zusammenstellung von Services abgebildet werden, diese Zusammenstellung erfolgt mit standardisierten Darstellungsformen, wie beispielsweise BPEL für eine Implementierung der SOA Services durch Web Services oder BPELJ für eine Implementierung durch Web Services und Java Komponenten.
- SOA Service Schnittstellen werden durch eine standardisierte Darstellung beschrieben. Dies erfolgt je nach gewählter Technologie für die Umsetzung, beispielsweise durch WSDL (Web Services), IDL (CORBA) oder durch spezielle Java Annotations kombiniert z.B. mit der Signatur von Enterprise Java Beans.
- SOA Services sind grob granuliert, können durch Altanwendungen oder Web Services oder Komponenten implementiert sein oder einer Zusammenstellung dieser Bausteine.
- Services können nach einer Vielzahl Kriterien klassifiziert werden.
- Eine SOA ist „geschwätzig“, Technologien für Daten- und Nachrichtenaustausch werden für den Aufbau einer SOA eingesetzt.
- Mit den SOA Blueprints existieren beispielhafte Implementierungen einzelner Funktionen bis hin zu kompletten Anwendungen gemäß einer SOA.

Wenn selbst das Herausarbeiten eines gemeinsamen SOA Verständnisses aufwendig ist, dann ist höchst fraglich, ob eine Umsetzung gelingen kann. Vor allem in der betrieblichen Praxis wird es diesbezüglich in den nächsten wenigen Jahren „zum Schwur“ kommen. Es wird sich erweisen ob SOA eine Umwälzung verursacht oder als Eintagsfliege in Vergessenheit gerät. Den Eingangs erwähnten Versprechungen kann bis dahin nur mit größter Skepsis entgegnet werden.

Literaturverzeichnis

- AbAn2004 Charles Abrams, Whit Andrews: Service Orientation Catalyzes Latest Software Revolution, Gartner Research Note G00124105, 2004.
- Abou2004 aboutIT: Interwoven stellt strategische Roadmap für integriertes Enterprise Content Management vor, Online-Ressource <http://www.aboutit.de/03/42/07.html>, (Abruf 29.10.2004).
- Atta2004 attachmate: Attachmate stärkt die Säulen der Unternehmens-IT, Online-Ressource http://ch.attachmate.com/press/press_release/0,1045,4213_12,00.html, (Abruf 29.10.2004).
- Bea2004a BEA: Schneller Einstieg in service-orientierte Architektur mit BEA WebLogic Server Process Edition, Online-Ressource <http://de.bea.com/presse/2004/040722.jsp>, (Abruf 29.10.2004).
- Bien2004 Adam Bien: Das XML der Komponenten, Java Magazin, 2004, Heft 11
- BuLa2003 Dr. Bärbl Burkhard, Guido Laures: SOA – Wertstiftendes Architektur-Paradigma, Objektspektrum, Heft 06/2003
- CIO2004a CIO Magazin: SOA als Mittel gegen Isolation, Online-Ressource <http://www.cio.de/index.cfm?Pageid=255&cat=det&maid=5738>, (Abruf 29.10.2004).
- CIO2004b CIO Magazin: Service-orientierter Software gehört die Zukunft, Online-Ressource <http://www.cio.de/index.cfm?PageID=255&cat=det&maid=5162>, (Abruf 29.10.2004).
- Comp2003 Computer Zeitung: Serviceorientierte Architektur bekommt Geschäftsregeln, 2003, Heft 22.
- Dugg2002 Jim Duggan: Simplify your Business Processes with an SOA Approach, Gartner Research Note AV-18-6077, 2004.
- Ebig2004 ebigo: Web Services – Wenn Software intelligent wird, Online-Ressource <http://www.ebigo.de/unternehmensbereiche/00095/>, (Abruf 29.10.2004).
- Ecau2004 ecaustria: Service-orientierte Architektur, Online-Ressource <http://ecaustria.at/?url=/?id=1666709>, (Abruf 29.10.2004).
- Econ2004 econet: Service-oriented Architecture, Online-Ressource <http://www.econet.de/magazin/default.htm>, (Abruf 29.10.2004).
- HP2004 Hewlett Packard: Service-orientierte Architektur sorgt für flexible Unternehmens-IT, Online-Ressource <http://h41131.www4.hp.com/de/de/pr/DEde01062004000343.html>, (Abruf 29.10.2004).
- IBM2004 IBM: BPELJ: BPEL for Java Technology, Online-Ressource <http://www-106.ibm.com/developerworks/webservices/library/ws-bpelj/>, (Abruf 6.11.2004).

- Kram2004 Christof Krameyer: Business Process Management, Online-Ressource http://www.avinci.biz/news/artikel/2004.02_BPM.html, (Abruf 6.11.2004).
- KrJa2004 Wolfram Kraushaar, Tanja Jander: Web Services – Die Technologie der Zukunft?, Online-Ressource http://www.informatik.fh-muenchen.de/~erik/labor/cim/Seminare/ws02/Thema06_Praesentation.pdf, (Abruf 29.10.2004).
- MaCi2004 Management Circle: Agenda Kongress Service-orientierte Architekturen, Online-Ressource http://www.managementcircle.de/pdf_upload/11-45367web.pdf, (Abruf 29.10.2004).
- Nati2003 Yefim Natis: Service-Oriented Architecture Scenario, Gartner Research Note AV-19-6751, 2003.
- PACo2004 PA Consulting Group: Service-orientierte Architektur – Radikale Innovation oder IT-Modeerscheinung?, Online-Ressource [http://www.competence-site.de/it-infrastructure.nsf/EB9D5DB4D84F5081C1256F120049DFC4/\\$File/soa_v9.pdf](http://www.competence-site.de/it-infrastructure.nsf/EB9D5DB4D84F5081C1256F120049DFC4/$File/soa_v9.pdf), (Abruf 29.10.2004).
- SAG2004 Software AG: EntireX: XML-powered Integration in nur drei Schritten, Online-Ressource http://www2.softwareag.com/de/products/entirex/prod_info/concepts/default.asp, (Abruf 29.10.2004).
- Seeb2004 Seebeyond: Business Integration 2004, Online-Ressource http://www.seebeyond.com/internationalsite/germany/press_release.asp?id=204, (Abruf 29.10.2004).
- TwPs2004 Thomas Walter, Peter Soth: Service Orientierte Architekturen mit BEA Weblogic, Online-Ressource http://www.java-forum-stuttgart.de/folien/A2_BEA.pdf, (Abruf 29.10.2004).
- Ullm2004 Andreas Ullmann: Was sich hinter SOA verbirgt – Lego für Fortgeschrittene, Java Magazin, 2004, Heft 10.
- WiHa2004a The Middleware Company: SOA Blueprints Initiative Definition, Online-Ressource http://www.middlewareresearch.com/endeavors/artifacts/soa-blueprints/SOA_Blueprints_Definition_v0_5.pdf, (Abruf 29.10.2004).
- WiHa2004b The Middleware Company: SOA Blueprints Concepts, Online-Ressource http://www.middlewareresearch.com/endeavors/artifacts/soa-blueprints/SOA_Blueprints_Concepts_v0_5.pdf, (Abruf 8.11.2004).
- Wiki2004 Wikipedia.de: WSDL, Online-Ressource <http://de.wikipedia.org/wiki/WSDL>, (Abruf 6.11.2004).