SOA AND WEB SERVICES

1.0 SOA

1.1 Was ist SOA ?

SOA steht für serviceorientierte / dienstorientierte Architektur (service-oriented architecture).

Es gibt jedoch keine allgemein akzeptierte Definition. Je nachdem ob eine technische oder wirtschaftliche Aspekte im Vordergrund stehen existiert eine Vielzahl von Charakteristiken. Das viele Unternehmen im SOA-Hype allerlei Produkte mit „SOA“ gekennzeichnet haben, was eine klare Definition zusätzlich erschwert.

Im Folgenden eine oft zitierte Definition nach OASIS :

„SOA ist ein Paradigma (Denkmuster) für die Organisierung und Nutzung verteilter Kapazitäten, die von unterschiedlichen Besitzern gesteuert werden“. *(OASIS Reference Model SOA 1.0, S.8)*

Oder auch :

“Service Oriented Architecture (SOA) represents a collection of best practices principles and patterns related to service-aware, enterprise-level, distributed computing. SOA standardization efforts at OASIS focus on workflows, translation coordination, orchestration, collaboration, Lose Kopplung, business process modeling, and other concepts that support agile computing”. (OASIS, SOA).

Man erkennt an beiden Definitionen dass der Grundgedanke von SOA nicht überwiegend technischer Natur ist, sondern sich sehr stark an Geschäftsprozessen orientiert.

Z.B. kann man den Prozess „ Ein Buch kaufen“ in verschiedene Ebene abstrahieren und Services wie „Bonität prüfen“ , „Bezahlung vornehmen“ oder „Auslieferung durchführen“ bilden und dann für ähnliche Prozesse verwenden. Durch diese Zusammenstellung der Services können dann Geschäftsprozesse gebildet werden.

Im Mittelpunkt steht das vollständig automatisierte Suchen, Anbieten und Nutzen von Diensten über ein Netzwerk. Das Netzwerk kann unternehmensintern sein, aber auch extern.

1.1.1 Serviceorientierung vs. Objektorientierung

Serviceorientierung steht in offensichtlich im Gegensatz zu Objektorientierung.

-Trennung zwischen Daten und Verhalten

- keine Vererbung

- Die Services sind prozedural und zustandslos

- Geringe Abhängigkeiten zwischen den Komponenten, die auch in Redundanz münden kann.

- Interne Standards vs. Offene Standards

- statische vs. Dynamischer Konfiguration

Trotzdem schließen die Konzepte sich nicht aus, sondern können sich auch gut ergänzen indem die SOA Komponenten objektorientiert gestaltet werden.

1.2 Warum SOA

Der Bedarf nach einer SOA hat sich in den letzten Jahren zwangsläufig aus den Veränderungen der geschäftlichen sowie technologischen Seite ergeben.

Im geschäftlichen Bereich wurden immer mehr Dienste die nicht zur Kernkompetenz einer Unternehmung gehören outgesourct und die Wichtigkeit von anpassungsfähigen Geschäftsprozessen stieg stetig.

Auf der technologischen Seite wurde in den letzen 15 Jahren viel aus dem Erfolg und den Fehlern von verteilten Systemen und Nachrichten-orientierter Middleware gelernt. Da erkannte man die Notwendigkeit von Middleware-Standards und – Architekturen. (vgl. Leymann et al., 2006)

Um diese Veränderungen optimal zu unterstützen bedarf es einer neuen Architektur, die sich von der Objektorientierung weg hin zur Prozess- oder Serviceorientierung entwickelt. Dieses Architekturkonzept wird SOA genannt.

Einige Vorteile durch SOA :

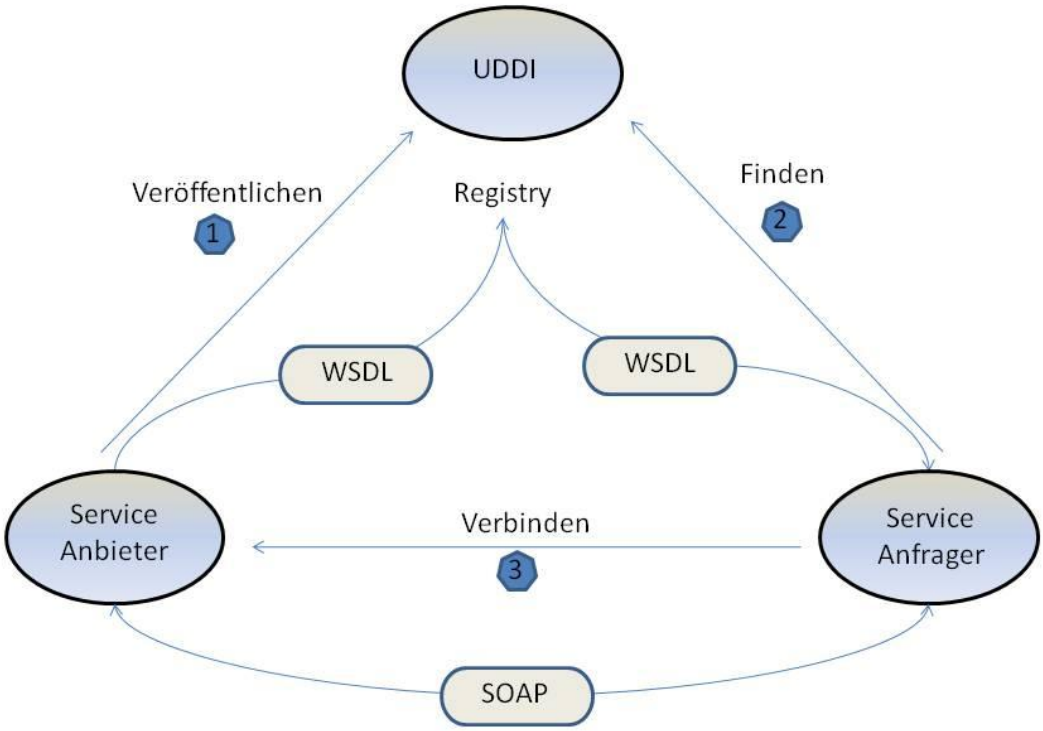
Geschäftsprozess kann auf viele Applikationen zugreifen

Informationsquellen sind durch mehrere Applikationen verfügbar (Redundanz)

Externe Datenquellen können leicht integriert werden

1.3 Konzept SOA :

Grundlegende Architektur



Prinzipiell gibt es in einer SOA drei Beteiligte : Den Anbieter eines Dienstes, den Nutzer und den Vermittler.

1.4 Was ist ein Service ?

Im alltäglichen Leben gibt es viele Services bzw. Dienste. Dazu gehören u.a die Dienste „ Versorgung mit Strom“, „Anbieten eines Internetzuganges“, „Anbieten einer Internet-Banking-Möglichkeit“, „Instant Messaging“, “Internet Radio“ und so weiter. Diese Dienste kann man als ein „veröffentlichtes Paket von Funktionalität“ betrachten. Die Dienste sind **kombinierbar**, denn man kann z.B. „Versorgung mit Strom“, „Nutzung eines Internetzugangs“ und „Instant Messaging“ gleichzeitig benutzen. Dienste sind aufgrund ihrer **Beschreibung**, Nutzungsbedingungen etc. durch Metadaten **auffindbar**. Allgemein kann

man für die Nutzung eines Services kein oder nur minimales Wissen wie ein Service implementiert ist und bereitgestellt wird. (vgl. Leymann, 2006, S. 13)

Die Eigenschaften eines Services in der IT sind :

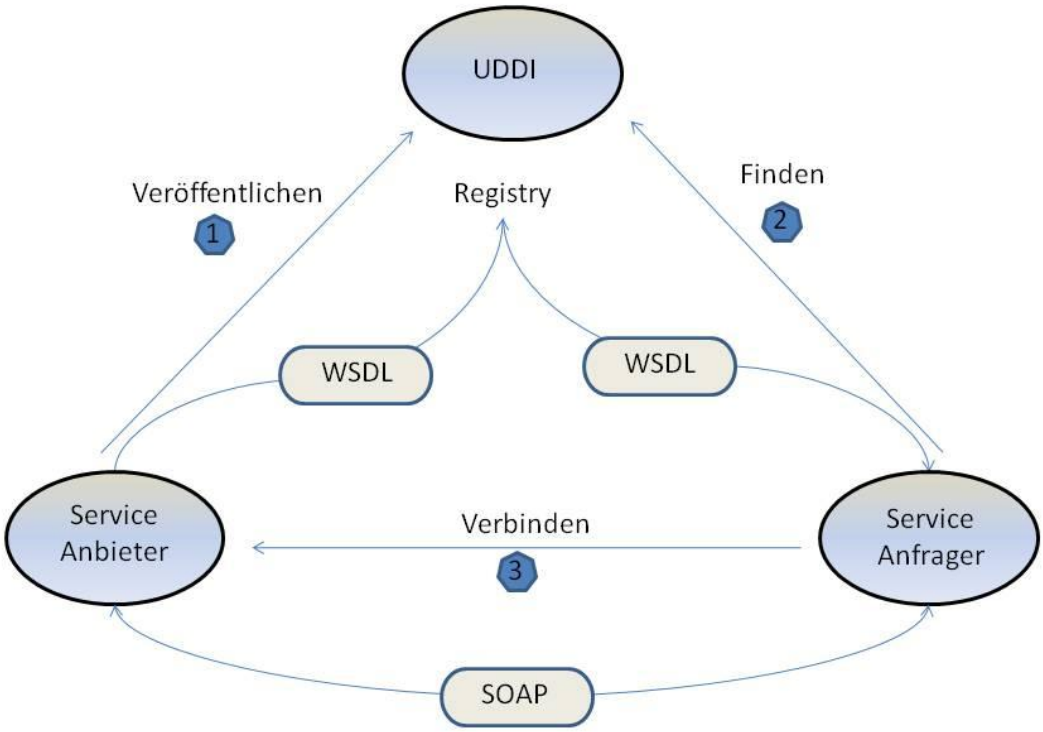
- Registrierung in einem Verzeichnis
- Dynamische Anbindung
- Abgeschlossenheit und eigenständige Nutzbarkeit
- Verfügbarkeit durch verschieden Transportmöglichkeiten, Formate und QoS
- Plattformunabhängigkeit zwischen Anbieter und Nutzer
- Dauerhafte Verfügbarkeit („always on“)
- Existenz einer einer veröffentlichten Schnittstelle

1.5 Lose Kopplung

In Systemen die mehrere Komponenten verbinden entsteht ein höheres Risiko von Fehlern und Ausfällen. Wenn ein Fehler das ganze System zum Erliegen bringt, dann handelt es sich um einen Systemfehler. Nach MASAK tauchen solche Fehler nicht als Folge der Komplexität auf, sondern meist als „Resultat auf den Versuch die Komplexität des Systems zu kontrollieren“. Eine enge Kopplung erhöht das Risiko dass aus einem Fehler multiple Fehler werden die zu einem Systemfehler auswachsen können.“Insofern ist die lose Kopplung der Services die ideal Voraussetzung für Emergenz als auch Überlebensfähigkeit eines aus Services bestehenden Gesamtsystemes“. (Masak 2007, S. 314).

Lose Kopplung ermöglicht erst die Kommunikation mit Diensten zwischen einem Dienstnutzer und dem zu nutzenden Dienst. Der Grundgedanke von Lose Kopplung schließt Wissen über die spezifische Plattform oder Formate und Protokolle des Dienstanbieters und –Nutzers aus. Ohne Lose Kopplung würde die Nützlichkeit oder die Wahl eines Services beschränkt. Lose Kopplung ist ein fundamentaler Bestandteil von SOA. (vgl. Leymann et al., 2006)

2.0 Web Services



Definition :

“A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-process able format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.” (W3C , Web Service Architecture , 2004)

Web Services sind nach dem SOA Paradigma aufgebaut wie man auch im strukturell gleichen Aufbau der Grafiken sieht. Dementsprechend finden sich dort auch die gleichen Merkmale wieder. Insbesondere gelten auch die Merkmale der Dienste für die Web Services.

Für den Aufbau des eines Web Services benötigt man die Spezifikationen WSDL (Web Service Description Language) sowie SOAP. UDDI (Universal Deskription Discovery and Integration) wird beim dynamischen Verbinden benötigt. Kernbestandteil ist die zentrale Registrierung (UDDI Registry).

SOAP dient dem Nachrichtenaustausch und WSDL ist die Beschreibung von Web Services. Die Beschreibung ermöglicht eine SOAP Nachricht für einen bestimmten Service zu erzeugen und verarbeiten.

Sind nun Web Services = SOA ?

2.1 Abgrenzung Web Service zu SOA

Web Service ist nicht gleich SOA. Web Service ist nur neben anderen Systemen die am häufigsten benutze Realisation einer SOA.

SOA ist ein Architekturparadigma, das keine bestimmte Implementierung vorschreibt. Web Services basieren u.a. konkret auf WSDL,SOAP und UDDI.

SOA ist nicht neu. Eine serviceorientierte Architektur wurde auch vor über eine Dekade durch COBRA versucht.

SOA ist keine Lösung für fachliche Probleme.

SOA ist so individuell wie Middleware in Unternehmen. SOA Muss immer individuell auf die Unternehmen zugeschnitten werden.

(vgl. Wikipedia Serviceorientierte Architektur, 2009)

2.2 SOAP

2.2.1 Was bedeutet SOAP

Terminologie :

SOAP : „Set of Conventions governing the format and processing rules of SOAP messages. Include interactions among SOAP nodes generating and accepting SOAP messages for purpose of exchanging information along a SOAP message path.” (Leymann, Skript, S.37)

Bis zur Version 1.1 war SOAP ein Akronym für „Simple Object Access Protocol“.
In der Zwischenzeit steht SOAP für sich selbst und ist kein Akronym mehr –
es steht für eine Nachrichten Architektur ! (vgl Leymann, Skript, S. 31)

Im Folgenden beziehe ich mich nur auf die aktuelle Version 1.2 von SOAP.

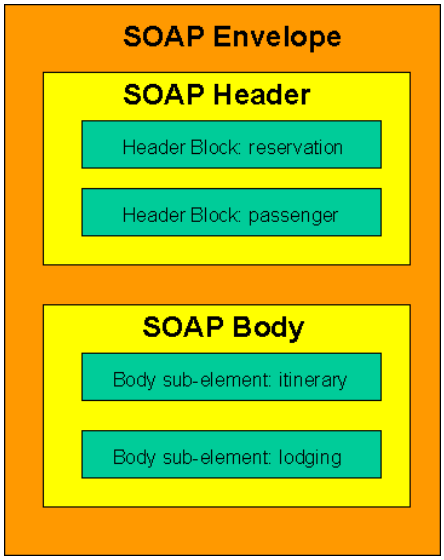
SOAP enthält Definitionen zu XML basierten Informationen, welche dem Austausch von strukturierten und typisierten Informationen zwischen Knoten (nodes) bzw. Dienstanbieter und Dienstnachfrager in einer dezentralen Umgebung dienen. (vgl. W3C SOAP Messaging Framework).

SOAP bestimmt also die grobe Struktur und die Vorschriften die zur Verarbeitung notwendig sind. Die Richtung des Nachrichtenaustausches erfolgt gewissermaßen nach einer Art Einbahnstraße („one way communication“). Durch die Kombination solcher einseitiger Austauschrichtungen können komplexe Interaktionsformen geschaffen werden, indem die Eigenschaften der darunter liegenden Protokollschicht genutzt werden.

2.2.2 SOAP Nachricht

Die SOAP Nachricht ist ein XML strukturiertes Paket welches einen Envelope („Umschlag“), einen Header („ Kopf“) und Body („Körper“) enthält.

Eine SOAP Nachricht ist als XML-Infoset definiert, welche eine abstrakte Zusammenstellung seines Inhaltes bereitstellt. Ein Infoset wird präzise definiert was man als Informationsmenge bezeichnet hat. Die Infoset-Empfehlung des W3C Konsortiums unterscheidet zwischen elf verschiedenen „information items“ (Informationseinheiten) mit genau definierte Eigenschaften .



(W3C, http://www.w3.org/2000/xp/Group/2/06/LC/primer-figure-1.png

2.2.3 SOAP Verfahrensmodel

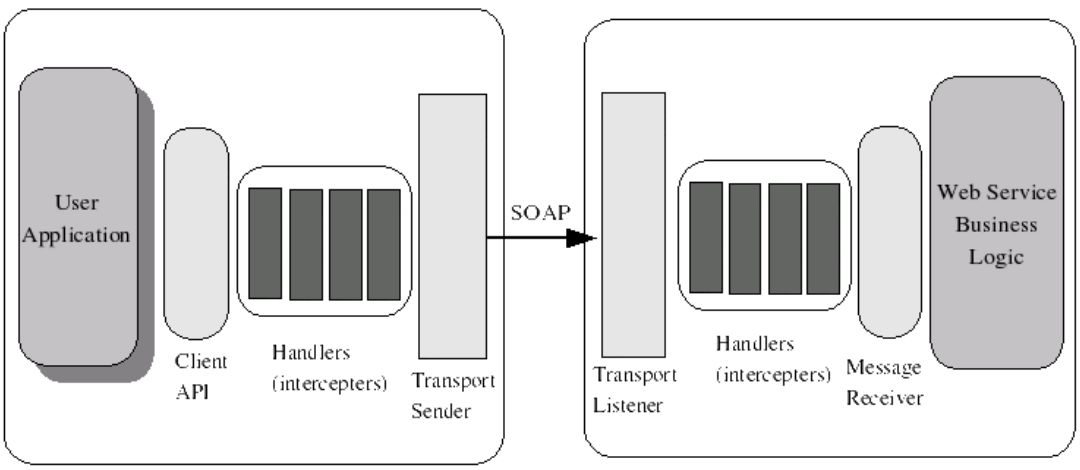
Die SOAP Nachrichten Spezifikation ist wie o.g. ein XML Infoset. Das bedeutet natürlich das der Sender ein Infoset sendet, dass der Empfänger wieder herstellen kann. Damit dies erfolgreich gelingt Muss der Sender das Infoset in einer bestimmten Art serialisieren, die der Empfänger auch für die Wiederherstellung nutzen kann. Üblicherweise wird dazu die XML 1.0 Syntax verwendet. (vgl. Leymann et al. 2006) Die Spezifikation erlaubt aber auch andere, womöglich besser geeignete, Vereinbarungen zu nutzen. Theoretisch kann man jeden beliebigen Inhalt verwenden. SOAP definiert nicht den Inhalt, sonder die Art wie ein SOA Node die Daten verarbeitet.

Der Header enthält nicht die Informationen die im eigentlichen Sinne übermittelt werden sollen, sondern Informationen über die Verarbeitung wie z.B. QoS oder Verschlüsselung.

Der Header kann auch einen großen Header-Block enthalten. Eine Entität kann dann z.B. verschiedenen Nodes auf dem Nachrichten-Pfad zugeordnet werden und jede Node kann dann unterschiedlich mit den Daten verfahren.

Die Erweiterungsmöglichkeit des Headers erlaubt es vollständig kompatible Protokolle auf SOA Basis zu erstellen. Es können also verschieden Service Spezifikationen durch unterschiedliche Header Inhalte entworfen werden. Sollte ein QoS benötigt werden, der eine Nachricht verschlüsselt, verlässlich und bestätigt versendet, so genügt es prinzipiell in den Header-Block drei Headers mit den jeweiligen Verfahrensanweisungen zu erstellen. Ein Block enthält definiert die Verlässlichkeit, einer die Verschlüsselung und der dritte die Bestätigung von den Nodes auf den Pfad.

Die erweiterbare SOAP Struktur mit ihren Verarbeitungsregeln erlaubt also die Service-Architektur in solch einer flexiblen Weise anzuordnen. (vgl Leymann et al. 2006).



2.2.4 SOAP Rollen

Wenn ein Node eine Nachricht bekommt, wird diese auf eine Rollendefinition untersucht und den Header entsprechend verarbeiten. Im Header können Standard- sowie benutzerspezifische Rollen stehen.

Standard Rollen :

“next” : Jeder Node auf dem Nachrichtenpfad sowie der letzte muss in dieser Rolle handeln.

(siehe http://www.w3.org/2003/05/soap-envelope/role/next).

„none“: Der Node muss in dieser Rolle nicht handeln. Der Header Block soll nicht direkt verarbeitet werden, da der Body nur ergänzende Informationen enthält.

(siehe http://www.w3.org/2003/05/soap-envelope/role/none)

„ultimateReciever“ : Der letzte Empfänger der Nachricht muss handeln. Wenn keine Rolle angegeben ist, wird diese Rolle als „default“ verwendet.

(siehe http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver)

Benutzerspezifische Rollen wie z.B. „Integritätscheck“ oder „LogEintragErstellen“ können auch definiert werden. Die SOA Spezifikation legt nicht fest wie man eine Rolle an eine Node vergibt. Dies hängt davon ab wie man SOA implementiert. SOA ist ja nur ein Konzept und keine Lösung.

2.2.5 Dokumente und RPC

SOAP unterstützt das Programmiermodel „Document literal“ und „remote procedure call“ (RPC).

Bei „Document literal“ enthält der Body ein Dokument wie z.B. eine Kreditkartennummer die vom Service dann bearbeitet wird. Der Service schickt nach dem Bearbeiten dann auch ein Dokument zurück das z.B. die letzten Transaktionen enthält.

Beim RPC enthält der Body einen Prozedurnamen der beim Empfänger aufgerufen wird. Als Ergebnis erhält der Dienstnachfrager ein Ergebnis oder eine Menge von Werten zurück.

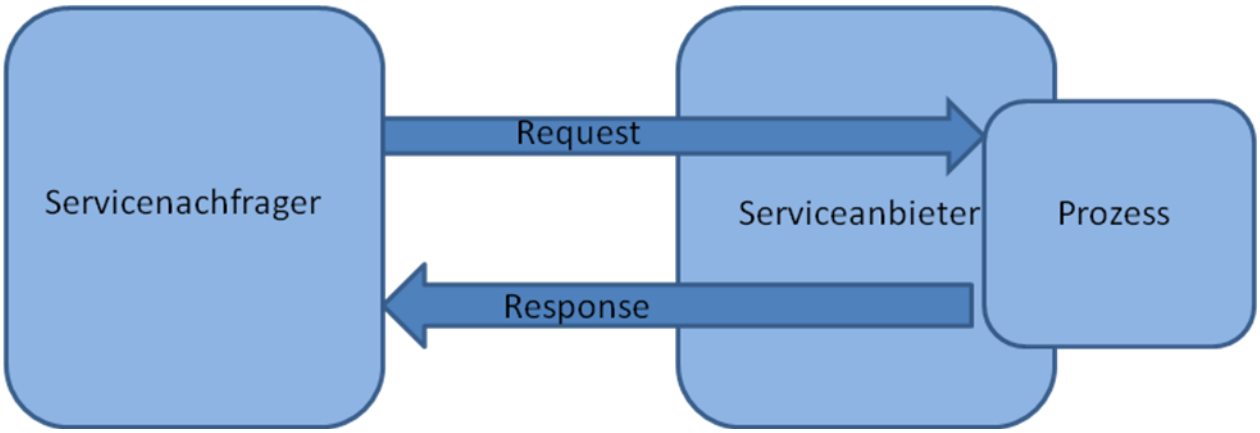
Da die Web Services aber auf XML basieren, wird hier natürlich nur „Document literal“ verwendet.

2.2.6 Message Exchange Patterns

SOAP beschreibt eine „one way“ Nachrichtenmodel zwischen einem SOAP Sender und einem SOAP Empfänger. Dies ist eine sehr kleine Einheit und alleine nicht sehr nützlich. Aber mit vielen dieser kleinen Einheit kann man jede beliebige Struktur aufbauen und sehr nützliche Interaktionen zwischen den Nodes erreichen. Diese Interaktionen werden Message Exchange Patterns genannt (MEPs) und stellen „Muster für die Übertragung“ dar.

Das einfachste Muster ist das

Request / Response MEP:



Folgender Ablauf tritt dann ein :

1. Der Servicenachfrager sendet eine Anfrage / Request – Mitteilung an einen Serviceanbieter.
2. Der Serviceanbieter bietet seinen Service an und bearbeitet die Mitteilung bzw. das Dokument.
3. Der Serviceanbieter sendet die bearbeitete Mitteilung als Response Mitteilung an den Absender der Anfrage zurück.

Da die Services wie schon erwähnt zustandslos sind, gibt es hier keine „Session“ wie bei http.

Long-Running Conversational MEP :

Gerade bei komplizierteren Geschäftsprozessen mit vielleicht mehreren beteiligten Nodes ist es mit einem einmaligen Request / Response MEP nicht getan. Die Kommunikation zwischen den Nodes benötigt mehr Zeit und damit auch Korrelationsinformationen um Störungen durch andere Dienstnutzer zu verhindern. Diese Informationen kann man

- im Body (env:Body) des Dokuments oder
- im Header hinterlegen.

Beispielhafter Ablauf eines Geschäftsprozesses :

1. Der Einkäufer startet eine Preisanfrage.
2. Der Lieferant gibt ein Angebot ab.
3. Der Einkäufer gibt eine Bestellung aufgrund des Angebotes ab.
4. Der Lieferant bestätigt die Bestellung.
5. Lieferant gibt (nach Absprache mit der Spedition) Liefertermin der Bestellung bekannt.
6. Einkäufer (prüft ob jemand im Wareneingang am Termin da ist und)bestätigt den Liefertermin .

.

2.2.7 SOAP Bindings

SOAP ermöglicht den Austausch von SOAP Nachrichten auf einer großen Basis von Protokollen. Die formalen Regeln wie man eine Nachricht auf oder in einem andern Protokoll transportiert, nennt man „binding“. Das SOAP Protocol Binding Framework bietet allgemeine Regeln für die Spezifizierung von Protokoll Bindings. Das Rahmenwerk beschreibt auch die Beziehung zwischen dem Bindung und SOAP Nodes die diese implementieren.

(vgl. W3C, http://www.w3.org/TR/2007/REC-soap12-part2-20070427/)

Eigenschaften eines Bindings nach W3C :

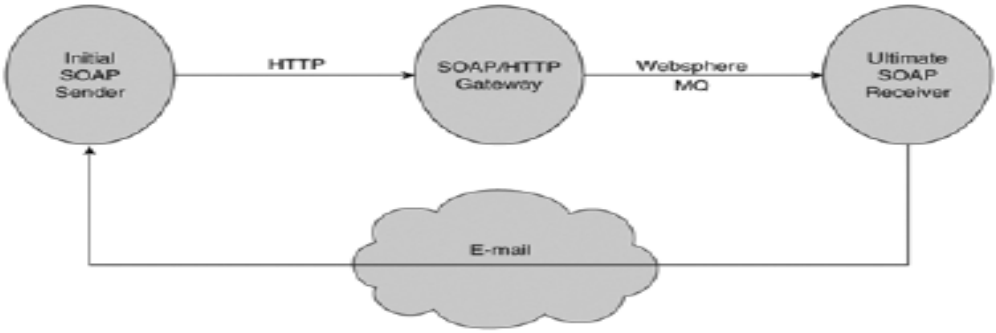
- Festlegung der bereitgestellten Eigenschaften
- Beschreibung wie die darunterliegenden Dienste benutzt werden um SOAP infosets zu transportieren.
- Beschreibung wie die Dienste des darunterliegenden Protokolls benutzt werden
- Festgelegte Fehlerbehandlung für alle möglichen Fehlerarten die beim Binding auftreten können.
- Definierung der Anforderungen um eine konsistente Implementierung zu spezifizieren.

Ziele des Binding Frameworks

1. Definierung der Anforderungen und Konzepte die zu den Binding Spezifizierungen passen.
2. Das Vereinfachen von ähnlichen Beschreibungen in Situationen wo Bindungen die gleichen Eigenschaften fördern um die Wiederverwendung (reuse) zu unterstützen.
3. Vereinfachung der Konsistenz für optionale Features.

(W3C, Messaging Framework, 2007)

SOAP Bindings



(Leymann et al. 2007, S. 79)

2.2.8 SOAP Attachements

SOA und damit auch SOAP beruhen auf dem Gedanken des Aufbaus nach Geschäftsprozessen auf der Grundlage von XML. In vielen Geschäftsprozessen ist jedoch die Übertragung von großen Datenmengen erforderlich und dabei hat XML einige Nachteile.

Mit der standardisierten Serialisierung für SOAP können nur Zeichen übertragen werden die von XML erlaubt sind. Durch Benutzung der Base64 Methoden kann man dies zwar realisieren, hat dann aber zwei gravierende Nachteile:

- Die En- und Decodierung benötigt viel Zeit.
- Die Nachrichtengröße wächst sehr stark.
- Nicht alle SOAP Implementierungen unterstützen dies.

Bei kleinen Netzwerkbandbreiten sowie geringer Node Rechenleistung ist dies ein Problem.

Folgende alternative Methoden existieren um das Problem zu mindern, aber nicht ganz zu beseitigen:

- Erzeugung eines MIME Dokuments und senden über http. (vgl IBM, Handling attachments in SOAP, 2002)
- Direct Internet Message Encapsulation (DIME), Microsoft
- MTOM, Message Transmission Optimization Mechanism
- XOP, XML-binary Optimized Packaging

Wobei MTOM und XOP überwiegend verwendet werden.

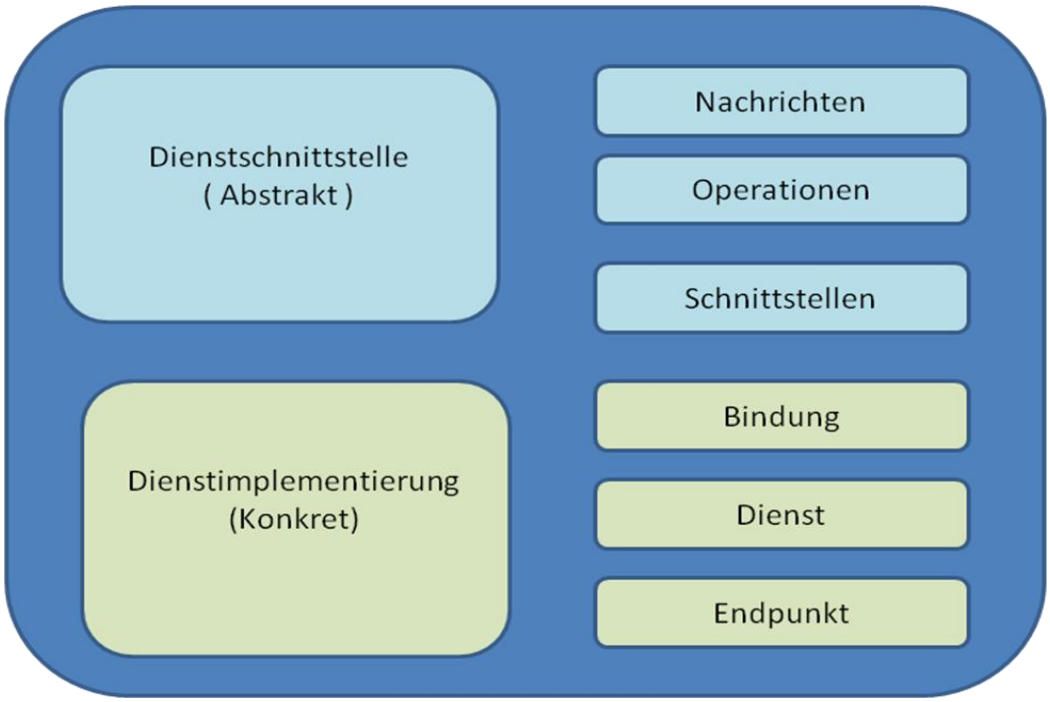
3.0 WSDL

3.1 Was ist WSDL ?

Web **S**ervice **D**escription **L**anguage (WSDL) ist eine XML basierende Beschreibungssprache um Web-Service Schnittstellen zu definieren.

Ein WSDL Dokument besteht aus zwei Teilen :

Einmal aus dem wiederverwendbaren abstrakten Teil und dem konkreten Teil. Der abstrakte Teil (Dienstschnittstelle) besteht aus dem Verhalten und der Funktionalität eines Dienstes. Der konkrete Teil (Dienstimplementierung) zeigt wie und wo man auf eine Dienstimplementierung zugreifen kann. (vgl. Leymann et. al. 2006)



Nach Leymann (2006) et. al. kann man sagen, „…das WSDL benutzt wird um anderen zu sagen was man kann."

3.2 Rolle von WSDL in SOA

WSDL ermöglicht erst viele Vorteile der SOA und Web Services. Ohne das Wissen das aus der WSDL Beschreibung hervorgeht kann man nicht flexibel die Serviceanbieter austauschen und die Zukunftsfähigkeit im Sinne der dynamischen Anpassung sowie die Lose Kopplung wurde stark beeinträchtigt.

WSDL übernimmt folgende beiden Rollen :

<u>Beschreibung eines Standard-Services für den Service Ersteller :</u>

Das WSDL Dokument übernimmt die Rolle einer Spezifikation für die Erstellung eines anderen Services. Zum Beispiel wird bei der Verkettung der Wertschöpfung von Automobilproduzenten und Zulieferern das WSDL Dokument herangezogen um die kompatible Entwicklung von anderen Schnittstellen zu gewährleisten. Besonders populär ist dafür EDIFACT. Werden mehr Autos gebaut, so kann das System selbstständig 4 Reifen bestellen und der Reifenproduzent kann selbst mehr Draht bei seinem Lieferanten bestellen.

Beschreibung eines Services für den Kunden :

Es wird einem Kunden der Leistungsumfang eines Service angezeigt. Die Beschreibung enthält Nachrichten- und Operationsdeklarationen, die Deklaration des Austausches und die Adresse des Services. Das Ziel ist es dass der Kunde den Service effizient nutzen kann.

3.3 Architektur von WSDL

Die Architektur beruht auf folgenden Eigenschaften (vgl. Leymann 2006) :

3.3.1 Erweiterbarkeit

Bei WSDL ist nur das beschrieben was unbedingt nötig ist um zukünftige Erweiterungen keine Grenzen zu setzen. Es wird nur das Nachrichtenformat, die Nachrichten-Interaktions-Muster , Repräsentationsart sowie die Nachrichtenquelle beschrieben. Es wurde so wenig wie möglich festgelegt, aber so viel wie nötig.

3.3.2 Unterstützung für verschiedene Definitionstypen

WSDL kann auch Services beschreiben dessen Daten nichts mit XML zu tun haben. WSDL kann z.B. verwendet werden um die MIME Struktur zu beschreiben. Jedoch ist XML als W3C Empfehlung bevorzugt zu verwenden.

3.3.3 Unifying Messaging und RPC

Unifying Messaging und RPC sind die beiden Lösungsansätze für die Systemintegration.

WSDL unterstützt beide Möglichkeiten. Um RPC zu verwenden benutzt WSDL 1.1 das <message> Konstrukt, in WSDL 2.0 es durch ein Nachrichtenmuster genutzt.

3.3.4 Trennung von „Was“ von „Wie“ und „Wo“

Durch die Trennung der Beschreibungen wird die Möglichkeit der Wiederverwendung stark erhöht. Folgende Teile sind getrennt :

- <portType> bzw. <interface> : Was macht der Dienst ?
- <port> bzw. <endpoint> : Wo wird der Dienst angeboten ?
- <binding> : Wie erfolgt die Interaktion ?

3.3.5 Unterstützung für verschiedene Protokolle und Transportmöglichkeiten

Durch das <binding> unterstützt WSDL viele verschieden Anbindungsmöglichkeiten an einen Dienst. Ein Dienst kann über SOAP mittels HTTP an einem Port angeboten werden, an einem anderen Port z.B. über RMI/IIOP durch TCP. Ein Bindung erklärt wie auf einen Service über ein bestimmtes Protokoll zugegriffen werden kann.

3.3.6 Keine Reihenfolge

Die Reihenfolge der Beschreibung der angebotenen Operationen eines Dienstes lassen keinen Rückschluss auf die Ausführungsreihenfolge zu. Die Reihenfolge ist in keine Weise festgelegt. Eine Ordnung kann mittels Business Process Execution Language for Web services, kurz BPEL, modelliert werden.

3.3.7 Keine Semantik

Als Beschreibungssprache hat sich WSDL darauf beschränkt nur auf einem elementaren Niveau das zu beschreiben was ein Service tut. Eine richtige Semantik im klassischen Sinne gibt es nicht. Der Dienst namens „ holeAnzahl“ sagt nicht aus ob es sich bei der Anzahl um Äpfel, Menschen, oder Maschinen handelt.

4.0 UDDI

Universal **D**escription, **D**iscovery and **I**ntegration (UDDI) handelt es sich um einen Verzeichnisdienst (Registry) der die Verbindung zwischen dem Serviceanbieter und –nachfrager initiieren kann. Bei einer statischen Anbindung an einen Web Service ist UDDI von geringer Bedeutung und wird nicht benötigt. Für die dynamische Anbindung jedoch ist UDDI unerlässlich wenn die Anbieter nicht schon vordefiniert sind.

UDDI besitzt eine SOAP-Schnittstelle und enthält die Informationen zu Unternehmen mit den jeweiligen Daten und Diensten.

Aufgebaut ist UDDI wie eine „Telefonbuchsammlung“. Es gibt die „White Pages“ (normale Telefonbuch), die „Yellow Pages“ (gelbe Seiten) sowie die „Green Pages“.

Folgende Funktionen erfüllen die „Telefonbücher“:

White Pages : Enthalten die Basisinformationen über den Anbieter. Darunter Geschäftsstammdaten, den Ansprechpartner und eine weltweit eindeutige ,sog. DUNS, Unternehmskennzahl.

Yellow Pages : Enthalten wie die Gelben Seiten eine Auflistung nach Unternehmenszweck, bzw. die Art des angebotenen Dienstes. Die Klassifikation in diesem „Branchenbuch“ erfolgt nach internationalen Standards wie UNSPSC.

Green Pages :

Enthält Schnittstellenbeschreibungen eines Web Services. In einem sog. T-Modell werden diese Informationen abgelegt. Wenn ein Web Service Anbieter mit einem Nachfrager eine Beziehung eingehen möchte, werden die T-Keys des T-Modells gegeneinander abgeglichen.

(vgl. Eric Newcomer 2002, S. 157)

5.0 Quellen :

IBM, Handling attachments in SOAP, **2002**, http://www.ibm.com/developerworks/webservices/library/ws-soapatt/

MASAK, DIETER : SOA?: Serviceorientierung in Business und Software, Springer Verlag **2007**

OASIS (Organization for the Advancement of Structured Information Standards), www.oasis-open.org

OASIS, REFERNECE MODEL, http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf

OASIS; SOA , http://www.oasis-open.org/committees/tc_cat.php?cat=soa

Leymann, F., et. al. Web Service Plattform Architecture, 4. Auflage **2006**

Newcomer, Eric : Understanding Web Services, Independent Technology Guides, **2002**

W3C , Web Service Architecture , **2004,** http://www.w3.org/TR/ws-arch/#whatis

W3C, SOAP, Messaging Framework, **2007** http://www.w3.org/TR/soap12-part1/

W3C, Messaging Framework**, 2007, http://www.w3.org/TR/soap12-part1/**

Wikipedia, Serviceorientierte Architektur, **2009,**
http://de.wikipedia.org/wiki/Serviceorientierte_Architektur.