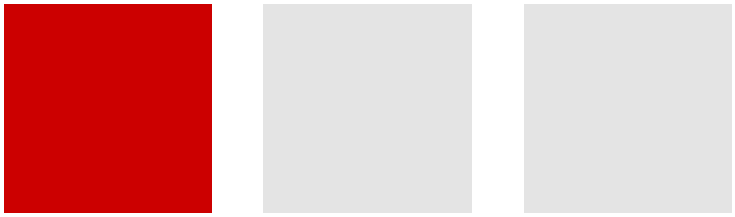


trivadis
makes IT easier.

# Service Contract Handling

# Best-Practices zum Umgang mit Service Contract Artefakten (WSDL) in Oracle SOA Suite 11g

Matthias Furrer
Principal Consultant
November 2013

**Dieses Dokument beschreibt allgemeine Hinweise und best-practices zum Umgang mit Service Contract Artefakten (WSDL Files) in Oracle SOA Suite 11g.**

- **Definition von konkreten oder abstrakten Service Contract Definitionen**
- **Sinnvoller Einsatz von Oracle Metadata Services (MDS)**
- **Umgebungsspezifische Definition von Endpoint Adresse des Service.**

**Die in diesem Dokument beschriebenen Szenarien wurden getestet mit Oracle SOA Suite 11.1.1.6 (11gR1 PS5).**

# 1. Einführung

Service Contracts – oder Service Schnittstellen – werden mit der Web Service Description Language (WSDL) als Definitionssprache beschrieben. Diese Schnittstelle oder Kontrakt sollte möglichst einfach zu lesen sein und verstanden werden können, Abhängigkeiten zwischen Service Consumern und Implementierung vermeiden sowie idealerweise ein einheitliches Datenaustauschformat in Form eines kanonischen Datenmodells unterstützen.

Um diese Ziele zu erreichen, wird generell der sogenannte *„Contract-First"* Ansatz empfohlen – dabei wird zuerst der Kontrakt in Form von WSDL- und XML-Schema-Dokumenten definiert, bevor mit der Service-Implementierung (Schreiben von Code) begonnen wird.

## 1.1 Generelle Richtlinien und Empfehlungen

Generelle Empfehlungen und Richtlinien für den Entwurf von Service Contract Dokumenten sind im Dokument „Service Kontrakt Design mit der Web Service Definition Language (WSDL)" **[TVD-WSDL-DES].**beschrieben. Darin werden die notwendigen Informationen zur Verfügung gestellt, damit Web Service Schnittstellen in konsistenter Art und Weise definiert werden.

## 1.2 Besonderheiten mit Oracle SOA Suite 11g

Über diese generellen Empfehlungen hinaus, gibt es beim Entwurf und der Verwaltung von Service Contract Dokumenten zur Verwendung in der Oracle Fusion Middleware einige spezifische Punkte zu beachten, um die Wiederverwendbarkeit – bspw. in Oracle Service Composites (SCA's) und Oracle Service Bus (OSB) - den problemlosen Einsatz der darauf basierenden Dienste zu gewährleisten. Auf diese besonderen Punkte wird in den nachfolgenden Kapiteln näher eingegangen und Richtlinien dazu definiert.

## 1.3 Notation im Dokument

Dieses Dokument soll eine Vorlage darstellen, um projektspezifische Richtlinien für den Entwurf und die Handhabung von technischen Service Contract Dokumenten zu erstellen. Darin wird eine Klassifizierung der Regeln wie im Kapitel 1.4 beschrieben vorgenommen, welche bei Bedarf übernommen werden kann. Die im Folgenden aufgeführten Hinweise können jedoch auch im Sinne von best-practices Hinweise formuliert und angewandt werden.

## 1.4 Formale Notationen im Dokument

### 1.4.1 Regel Gruppe

Die Regeln werden nach den Konventionen, die vom Request for Comment 2119 [REQ_COMM_2119] definiert wurden, in verschiedene, nachfolgend aufgelisteten Gruppen aufgeteilt.

| | |
|---|---|
| MUST | Bedeutet eine absolute Anforderung, die zwingend eingehalten werden muss. |
| MUST NOT | Bedeutet ein absolutes Verbot, d.h. es darf auf keinen Fall verwendet werden. |
| SHOULD | Bedeutet, dass es gute Gründe geben kann, die entsprechende Regel zu ignorieren. Die Auswirkungen müssen klar sein und gewichtet werden, bevor ein alternativer Weg beschritten wird. |
| SHOULD NOT | Bedeutet, dass es gute Gründe geben kann, dass das von der Regel beschrieben Verhalten akzeptabel oder sogar hilfreich ist. Die Auswirkungen müssen klar sein und gewichtet werden, bevor das entsprechende Verhalten implementiert wird. |
| MAY | Bedeutet, dass der durch die Regel beschriebene Begriff wirklich optional ist. Es ist Sache des Anwenders, ob er die Regel anwendet oder nicht. |

## 2. WSDL Typen – abstrakte und konkrete Definitionen

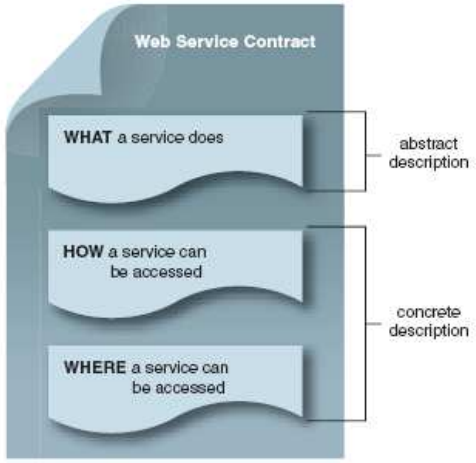
Die Struktur von Service Contract Dokumenten besteht aus 6 Hauptelementen oder Sections, welche wiederum in die Gruppe der abstrakten und konkreten Definitionen aufgeteilt werden kann:

<u>Abstrakte Definitionen</u>

- Types
- Messages
- PortTypes (oder Interfaces ab WSDL 2.0)

<u>Konkrete Definitionen</u>

- Bindings
- Ports (oder Endpoints ab WSDL 2.0)
- Services




Der abstrakte Teil beschreibt die Funktionalität des Service in implementations-neutraler Art während der konkrete Teil Nachrichtenformat, Übertragungsprotokoll und Netzwerkadresse definiert. Während des Service Designs und Implementierung werden in der Regel ausschliesslich die abstrakten Definitionen benötigt, während zur Laufzeit für die Ausführung auch die konkreten Definitionen notwendig sind.

### 2.1 Verwendung in Oracle Service Bus und SOA Suite

Normalerweise genügen während der Entwicklung eines Services in einer integrierten Entwicklungsumgebung (IDE) die abstrakten Definitionen in einem WSDL. So zum Beispiel können in *<u>JDeveloper rein abstrakte WSDL zur Entwicklung von SCA (Service Component Architecture) Komponenten</u>* verwendet werden. Auf der anderen Seite, wird *<u>in Eclipse für OSB Proxy Services mindestens die ‚Binding' Section</u>* aus den konkreten Definitionen benötigt um einen Service erstellen zu können.

| | | |
|---|---|---|
| GEN-0001 | Ein Service Contract Dokument **MUSS** zwingend auch die konkreten Definitionen gemäss Beispiel im Anhang A enthalten.<br><br>**Begründung**:<br>Die Definition von konkreten WSDL erlaubt die Verwendung des Dokumentes sowohl in Oracle Service Bus (OSB) als auch in SCA Komponenten der SOA-Suite (BPEL, Mediator, etc.) | MUST |

## 2.2 Definition der SOAP Location Address

Wie im Kapitel 4 beschrieben, kann die Endpoint Adresse der effektiv aufzurufenden Serviceoperation für SCA-Komponenten in den *Configuration Plan's*, die beim Deploy des Service verwendet werden, angegeben werden. Damit kann in BPEL- oder Mediator Komponenten die URI für den aufzurufenden Service umgebungsspezifisch festgelegt werden.

Falls die im Configuration Plan angegebene und beim Deployment verwendete URI über das `endpointURI` Property zum Zeitpunkt der Ausführung jedoch nicht verfügbar ist oder ein SOAP Fault retourniert wird, wird alternativ von der SOA Engine die ursprünglich im WSDL angegebene Adresse aufgerufen. Dies kann zu massivem Fehlverhalten führen, falls ursprünglich im WSDL eine gültige Aufrufadresse - beispielsweise auf der Entwicklungsumgebung - definiert wurde. Aus diesem Grund muss das `location` Attribut immer mit einem Wert versehen werden, welcher nicht über eine ansprechbare URI aufgelöst werden kann.

```
<wsdl:service name="SampleServiceService">
    <wsdl:port binding="tns:SampleServiceHttpSoap11Binding"
        name="SampleServiceSOAP">
        <soap:address location="to-be-replaced"/>
    </wsdl:port>
</wsdl:service>
```

Abbildung 1: Definition `location` Attribut in konkretem WSDL

Das oben beschriebene Fail-over Fehrhalten wird im SOA Developer Guide beschrieben [ORA_SOAD_EPR].

| | | |
|---|---|---|
| GEN-0002 | Das Attribut `location` im `soap:address` Element **MUSS** immer mit einem fiktivem Wert gefüllt sein, welcher nicht über eine tatsächlich existierende URI aufgelöst werden kann.<br><br>**Begründung**:<br>Vermeidung von unerwünschter Verwendung der im WSDL definierten Adresse als Fallback Endpoint in SCA Composites. Fehler bei der automatisierten Anpassung an jeweilige Laufzeitumgebungen werden sofort erkannt. | MUST |

# 3. Einsatz von Oracle Metadata Services (MDS)

Für die Entwicklung von Composite Services verwendete Artefakte werden typischerweise auch von mehreren anderen Objekten verwendet. Dazu gehören insbesondere XML Schema Files (XSD), Service Contracts (WSDL's) oder auch Fault-Policies.
In der Oracle SOA Suite existiert ein Datenbank-basiertes Repository um solche gemeinsam verwendete Objekte zu verwalten – Medatadata Services MDS.

## 3.1 Verwendung von MDS

Ein zentrales Repository von solchen Shared Objects bietet den Vorteil, dass diese Artefakte nicht in jedem einzelnen Entwicklungsprojekt gehalten und aktualisiert werden müssen und damit projekt-übergreifend verwendet werden können.

Zwar könnten beispielsweise XSD und WSDL Artefakte auch direkt über die auf dem Application Server verteilten Objekte referenziert werden, jedoch ist dieses Vorgehen im praktischen Betrieb keinesfalls zu empfehlen. Imports von Artefakten in den Entwicklungsobjekten über das `http` oder `https` scheme sollten auf keinen Fall verwendet werden.
Falls das referenzierte Objekte – z.B. WSDL Dokument eines konsumierten Services – nicht verfügbar ist, kann das referenzierende Objekt nicht deployed, undeployed oder gestartet werden. Da die Startreihenfolge der einzelnen Composites beim Service Start des SOA Servers nicht beeinflusst werden kann, führt dies zu erheblichen Problemen bei Composites die Abhängigkeiten zu anderen Deployments haben. Dadurch können «dead-lock» Situationen entstehen, die nur durch einen Service Restart und manuelles re-deploy gelöst werden können.

| | | |
|---|---|---|
| GEN-0003 | In SCA Komponenten **MÜSSEN** sämtliche WSDL Artefakte, die als Serviceschnittstellen für Consumer des jeweiligen Service zur Verfügung gestellt werden, im MDS verwaltet und darüber referenziert werden.<br><br>**Begründung**:<br>Zentrale Verwaltung aller Shared Artefacts und Vermeidung von Problemen beim Start der SOA Plattform durch gegenseitige Abhängigkeiten der Composites. | MUST |

## 3.2 Partnerlink WSDL's

In BPEL werden Endpoint Referenzen als sogenannte *Partner Link Roles* modelliert. Dabei repräsentiert eine Endpoint Referenz die notwendigen Informationen um den Service aufzurufen, wobei diese über den zugeordneten Porttype bezogen werden - welcher typischerweise wiederum in einem importierten WSDL Artefakt definiert ist.

```
<wsdl:definitions
    name="SampleService-v1-v1"
    targetNamespace=" http://www.trivadis.com/contract/core/SampleService/v1"
    xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:tns=" http://www.trivadis.com/contract/core/SampleService/v1"
    xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <plnk:partnerLinkType name="SampleReference">
        <plnk:role name="SampleServicePortType"
             portType="tns: SampleServicePortType "/>
        </plnk:partnerLinkType>
    <wsdl:import namespace="http://www.trivadis.com/contract/core/SampleService/v1"
      location="oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl"/>
</wsdl:definitions>
```

Abbildung 2: Definition Partnerlink WSDL für BPEL

Diese Partnerlink WSDL werden in JDeveloper automatisch mit der Dateiendung „Wrapper" generiert und müssen im Normalfall nicht modifiziert werden. Allerdings dürfen diese Dateien nicht im MDS gehalten werden, da damit unter bestimmten Release/Patch Konfigurationen schwer nachvollziehbare Laufzeitfehler auftreten können.

| | | |
|---|---|---|
| GEN-0004 | Partnerlink WSDL, die innerhalb eines SCA als interne Serviceschnittstelle für einen BPEL Prozess agieren, **DUERFEN NICHT** im MDS verwaltet werden. Diese werden ausschliesslich lokal im JDeveloper Projekt verwaltet.<br><br>**Begründung**:<br>Die Entwicklung wird durch dieses Vorgehen vereinfacht, darüber hinaus können in MDS keine allfällig spezifisch für ein Composite benötigte weitere Artefakte importiert werden.<br>Zusätzlich Vermeidung von Laufzeitfehlern durch Bugs in spezifischen Patches – bspw. p14406487 (PS5 Bundle Patch 4 (11.1.1.6.4)). Dieses Fehlverhalten ist bei Oracle dokumentiert unter Bug# 14104016 und kann mit dem Patch p14104016 für Version 11.1.1.6.4 behoben werden. | MUST NOT |

# 4. Umgebungsspezifische Definition

Werden aus einem bestimmten Service – bspw. BPEL Prozess oder OSB Integrationsflow – wiederum andere Services aufgerufen, so muss die Endpoint Adresse jeweils pro Umgebung angepasst werden. Diese Aufrufadressen der referenzierten Services sind in den einzelnen Umgebungen (bspw… Entwicklung, Test, Produktion) unterschiedlich. Das heisst zur Laufzeit sind diese Adressen absolut entscheidend, und zur Vermeidung von Fehlzugriffen sollte deren Konfiguration möglichst automatisiert erfolgen.

Innerhalb der Oracle SOA Suite stehen zur umgebungs-spezifischen Anpassung beim Deploy von Services zwei unterschiedliche Werkzeuge zur Verfügung:

- SCA-Komponenten: Configuration Plan
- Oracle Service Bus: Customization File

In beiden Fällen können die entsprechenden Artefakte über die IDE oder über WLST Scripting erzeugt werden und beinhalten die umgebungs-spezifischen Konfiguration – so wie typischerweise Service URI's. In beiden Fällen müssen diese Werte pro Umgebung angepasst werden und werden dann beim Deploy entsprechend appliziert.

| | | |
|---|---|---|
| GEN-0005 | `Import` Anweisungen in WSDL Dokumenten, dürfen nur über relative Pfadangaben erfolgen. Absolute Pfadangaben – z.B. über das `http` Scheme – **DÜRFEN NICHT** verwendet werden.<br><br>**Begründung**:<br>Automatisierte Anpassung der Import Location in WSDL Artefakten sind nur bedingt möglich. Durch die Verwendung von relativen Pfadangaben – ohne zusätzliche Angaben von Server/Port Informationen - wird sichergestellt, dass jeweils nur Artefakte der jeweiligen Laufzeitumgebung verwendet werden. | MUST NOT |

## 4.1 Definition von umgebungsspezifischen Endpoint URI's in SCA Komponenten

### 4.1.1 Referenzen auf Services über das SOAP Protokoll

Über das `location` Attribut kann der für den Aufruf des Webservice zu verwendende Endpoint definiert, und für das umgebungs-spezifische Deployment in den jeweiligen Configuration Plans angepasst werden. Dabei wird der Reference Service in JDeveloper über das im MDS verteilte Service Contract Dokument ausgewählt und danach im composite.xml das `location` Attribut manuell auf die entsprechende URI der effektiven Ausführungsadresse angegeben.

```
<reference name="SOAPProviderService"
    ui:wsdlLocation=" oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl ">
    <interface.wsdl interface=" …….>
    <binding.ws port ….."
        location="http://localhost:7001/SampleService?WSDL"
        soapVersion="1.1">
        <property name="weblogic.wsee.wsat.transaction.flowOption"
                  type="xs:string" many="false">WSDLDriven</property>
    </binding.ws>
</reference>
```

Abbildung 3: Definition `location` Attribut in SCA Composite

Wird danach der auf dieser Composite Definition erzeugte Configuration Plan erstellt, kann in den Binding Properties der umgebungsspezifische Endpoint über das `location` Attribut definiert werden, welcher beim Deploy in die jeweilige Umgebung verwendet werden soll.

```
<reference name="SOAPProviderService">
    <binding type="ws">
        <attribute name="port">
          <replace>…..</replace>
        </attribute>
        <attribute name="location">
            <replace>http://production-host:7001/SampleService?WSDL</replace>
        </attribute>
        <property name="weblogic.wsee.wsat.transaction.flowOption">
            <replace>WSDLDriven</replace>
        </property>
    </binding>
</reference>
```

Abbildung 4: Definition `location` Attribut in SCA Configuration Plan

| | | |
|---|---|---|
| GEN-0006 | In SCA Komponenten **SOLLTE** die Konfiguration von Endpoint URI's von Referenzen die mit dem SOAP Protokoll angesprochen werden, immer über das `location` Attribut erfolgen.<br><br>**Begründung**:<br>Sicherstellung einer einheitlichen und möglichst sicheren Behandlung von umgebungsspezifischen Ausführungsadressen für konsumierte Services. | SHOULD |

### *4..1.1.a.1 Alternativ: Referenz im `location` Attribut auf MDS Artefakt*

Grundsätzlich kann die Definition des WSDL Artefacts im `location` Attribut auf die Referenz im MDS belassen werden, so wie dies von JDeveloper beim Anlegen der Referenz im composite.xml generiert wird.

*In diesem Fall muss das Deployment – auch lokal im JDeveloper – immer zwingend über den zugehörigen Configuration Plan durchgeführt werden.* Andernfalls wird zur Laufzeit der Endpoint aus dem im MDS Artefakt definierten `location` Attribut der Service Section angesprochen.

```
<reference name="SOAPProviderService"
  ui:wsdlLocation="oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl">
    <interface.wsdl interface=" …….>
    <binding.ws port=”..”
      location="oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl"
      soapVersion="1.1">
      <property name="weblogic.wsee.wsat.transaction.flowOption"
                type="xs:string" many="false">WSDLDriven</property>
    </binding.ws>
</reference>
```

Abbildung 5: Definition `location` Attribut in SCA Composite mit Referenzierung auf MDS

Im für das Deployment verwendeten Configuration Plan muss danach zwinged das `location` Attribut wie in Abbildung 4 dargestellt ersetzt werden und auf die URI der entsprechenden Umgebung für diesen Webservice gesetzt werden.

*Um Probleme in der Entwicklungsumgebung im Falle von nicht verfügbaren Endpoint's zu vermeiden, sollten auch sämtliche Imports in anderen SCA Artefakten wie `ComponentType` Definition und Wrapper WSDL immer auf MDS Objekte referenzieren.*

### 4.1.2 Referenzen auf Services über das SOA-Direct Protokoll

Wie beim Aufruf von Services über das SOAP Protokoll, wird auch bei Services die via SOA-Direct Protokoll angesprochen werden, das entsprechende WSDL Dokument über das MDS ausgewählt. Es kann das gleiche WSDL Dokument wie für die SOAP Services verwendet werden. Da dieses jedoch in diesem Fall kein Binding für SOADirect enthält, muss das `address` Attribut manuell ergänzt werden.

Das Provider URL Property muss nur definiert werden, falls sich der aufgerufene Service nicht auf der physisch gleichen Maschine wie der SCA Service befindet. oder in einer Weblogic Domäne unter einem anderen Port ausgeführt wird. Andernfalls kann der lokale JNDI Provider verwendet werden.

```
<reference name="RMIProviderService"
    ui:wsdlLocation=" oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl ">
    <interface.wsdl interface="…."/>
    <binding.direct address="soadirect:/myPartition/myComposite/myDirectService"
      connection-factory="oracle.soa.api.JNDIDirectConnectionFactory"
                          useLocalJNDIProvider="false">
      <property name="java.naming.provider.url">t3://localhost:7301</property>
    </binding.direct>
</reference>
```

Abbildung 6: Definition `direct address` Binding Element in SCA Composite

Bei Bedarf kann danach im Configuration Plan über das `java.naming.provider.url Property` der physische Endpoint überschrieben werden, welcher für die jeweilige Umgebung verwendet werden soll.

```
reference name="RMIProviderService">
    <binding type="direct">
      <property name="java.naming.provider.url">
         <replace>t3://localhost:7301</replace>
      </property>
    </binding>
</reference>
```

Abbildung 7: Definition `java naming provider url` Property in SCA Configuration Plan

| | | |
|---|---|---|
| GEN-0007 | Sowohl für Aufrufe über das SOAP als auch das SOAP-Direct Protokoll, **SOLLTE** das gleiche WSDL Artefakt verwendet werden. Das `address` Attribut kann dabei in der SCA-Konfiguration manuell ergänzt werden. Für umgebungsspezifische Anpassungen kann das `java.naming.provider.url` Property im Configuration Plan verwendet werden.<br><br>**Begründung**:<br>Sicherstellung einer einheitlichen und möglichst sicheren Behandlung von umgebungsspezifischen Ausführungsadressen für konsumierte Services und Vermeidung von redundanten Service Contract Dokumenten. | SHOULD |

### 4e1.3 Verwendung des endpointURI SOA Composite Application Property

Das `endpointURI` SOA Application Property erlaubt die dynamische Übersteuerung des Ausführungsortes zur Laufzeit oder über den Enterprise Manager sowie die Definition von mehreren Endpoints zu Fail-Over Zwecken. Allerdings kann dieses Fail-Over Verhalten teilweise zu unbeabsichtigten Resultaten führen und sollte daher nur für spezifische Anwendungsfälle angewandt werden.

```
<reference name="TestReference"
    ui:wsdlLocation="oramds:/apps/wsdl/SampleService.wsdl">
    <binding.ws port=”.”>
      location="oramds:/apps/wsdl/samples/v1/SampleService-v1.0.wsdl"
        <property name="weblogic.wsee.wsat.transaction.flowOption"
                type="xs:string" many="false">WSDLDriven</property>
        <property name="endpointURI" type="xs:string" many="false"
        >http://to-be-replaced.com</property>
    </binding.ws>
</reference>
```

Abbildung 8: Definition `endpointURI` Property in SCA Composite

Danach kann der Configuration Plan erzeugt werden, welcher ein entsprechendes Property für den Endpoint URI enthält und jeweils in den umgebungs-spezifischen Config Plans angepasst werden kann:

```
<reference name="TestReference">
     <!--Add search and replace rules for the binding properties-->
     <binding type="ws">
        <property name="weblogic.wsee.wsat.transaction.flowOption">
          <replace>WSDLDriven</replace>
        </property>
        <property name="endpointURI">
          <replace>http://production-host:7001/SampleService</replace>
        </property>
     </binding>
</reference>
```

Abbildung 9: Definition `endpointURI` Property in SCA Configuration Plan

| | | |
|---|---|---|
| GEN-0008 | In SCA Komponenten **SOLLTE** die Konfiguration von Endpoint URI's von Referenzen die über das `endpointURI` Property nur erfolgen, wenn spezifisch ein Fail-Over Verhalten auf mehrere Endpunkte erwünscht wird.<br><br>**Begründung**:<br>Verhindern von unerwünschten Fail-Over Verhalten zu Endpoints welche nicht in der eigentlichen Composite Konfiguration definiert sind. | SHOULD NOT |
| GEN-0009 | Falls aus anderen Gründen das `endpointURI` Property verwendet wird, jedoch kein Fail-Over gewünscht wird, **MUSS** das `soap:address` Element mit dem `location` Attribut aus dem im MDS verteilten WSDL Dokument entfernt werden. Das zur Entwicklungszeit verwendete WSDL Dokument (in diesem Fall typischerweise dem lokalen, file-basierten MDS) muss das Element jedoch enthalten.<br><br>**Begründung**:<br>Vermeidung von unerwünschten Fail-Over Verhalten zu alternativen Endpoints und Fehlverhalten bei aufgerufenen Services .welche SOAP Faults retournieren. | MUST |

## 4.2 Definition von umgebungsspezifischen Endpoint URI's in OSB

In OSB erfolgt die Konfiguration der Endpoint Adressen von konsumierten Services über die jeweiligen Customization Files und/oder werden durch die Systemadministratoren über die Service Bus Console in der Business Service Konfiguration dynamisch verwaltet.

Dabei sind – mit Ausnahme der bereits Beschriebenen allgemeinen Guidelines für den Entwurf von Service-Kontrakt Dokumenten, keine spezifischen Besonderheiten in deren Definition zu berücksichtigen.

# 5. Referenzen


| | |
|---|---|
| [ORA_SOAD_EPR]. | *Developer's Guide for Oracle SOA Suite, Chapter A4.3.3 Resolving Endpoints*<br>Oracle Corp, March 2012<br>http://docs.oracle.com/cd/E28271_01/dev.1111/e10224/bp_appx_ref.htm#BABJIGIH |
| [REQ-COMM-2119] | *Key words for use in RFCs to Indicate Requirement Levels*<br>The Internet Engineering Task Force, March 1997<br>http://tools.ietf.org/html/rfc2119 |
| [TVD-WSDL-DES]. | *Service Kontrakt Design mit der Web Service Definition Language (WSDL)*<br>Trivadis AG, Guido Schmutz, Januar 2012 |

# 6. Appendix

## A Service Contract Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
name="SampleService-v1"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.trivadis.com/contract/core/SampleService/v1"
xmlns:err="http://www.trivadis.com/fault/common/v1"
xmlns:ext="http://www.trivadis.com/schema/SampleService/v1"
targetNamespace="http://www.trivadis.com/contract/core/SampleService/v1">

    <wsdl:documentation>Version 1.0</wsdl:documentation>
    <wsdl:import location="../public/wsdl/fault/v1/CommonFaults-v1.0.wsdl"
        namespace="http://www.trivadis.com/fault/common/v1"/>

    <wsdl:types>
        <xsd:schema >
            <xsd:import
               namespace=http://www.trivadis.com/schema/SampleService/v1
               schemaLocation="../xsd/messages/v1/SampleService-v1.0.xsd"/>
        </xsd:schema>
    </wsdl:types>

    <wsdl:message name="SampleOperationRequestMsg">
        <wsdl:part name="SampleOperationRequest"
           element="ext:SampleOperationRequest"/>
    </wsdl:message>
    <wsdl:message name="SampleOperationResponseMsg">
        <wsdl:part name="SampleOperationResponse"
            element="ext:SampleOperationResponse"/>
    </wsdl:message>

    <wsdl:portType name="SampleServicePortType">
        <wsdl:operation name="SampleOperation">
            <wsdl:documentation> Sample Service </wsdl:documentation>
            <wsdl:input name="SampleOperationRequest"
                message="tns:SampleOperationRequestMsg"/>
            <wsdl:output name="SampleOperationResponse"
                message="tns:SampleOperationResponseMsg"/>
            <wsdl:fault name="ValidationFault"
                message="err:ValidationFaultMsg"/>
        </wsdl:operation>
    </wsdl:portType>
```

```
    <wsdl:binding name="SampleServiceHttpSoap11Binding"
       type="tns:SampleServicePortType">
        <soap:binding style="document"
          transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="SampleOperation">
            <soap:operation
            soapAction="http://www.trivadis.com/sample/v1/SampleOperation"/>
            <wsdl:input>
                <soap:body parts="SampleOperationRequest" use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body parts="SampleOperationResponse" use="literal"/>
            </wsdl:output>
            <wsdl:fault name="ValidationFault">
                <soap:fault name="ValidationFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
    </wsdl:binding>

  <wsdl:service name="SampleServiceService">
    <wsdl:port binding="tns:SampleServiceHttpSoap11Binding"
      name="SampleServiceSOAP">
      <soap:address location="http://www.example.org/"/>
    </wsdl:port>
  </wsdl:service>

</wsdl:definitions>
```