OWLS2WSDL: Semi-automatische Translation von OWL-S Diensten in WSDL und experimentelle Evaluierung

Vergleichende praktische Untersuchung zur Relation zwischen semantischen und konventionellen Webdiensten in OWL-S und WSDL (Matchmaking)

Oliver Fourman

Fachbereich Grundlagen Informatik Sensortechnik
Hochschule für Technik und Wirtschaft des Saarlandes

Deutsches Forschungszentrum für Künstliche Intelligenz Saarbrücken

Masterthesis KIM 2006.2007

Kolloquiumsvortrag 14. Juni 2007

Einführung

Hochschule für Technik und Wirtschaft des Saarlandes
University of Applied Sciences

HTW

DFKI Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

- Thema bearbeitet im Forschungsbereich DMAS am DFKI
- Brückenthema der Projekte ATHENA-IP und SCALLOPS

ATHENA
European Integrated Project
IST-507849
Information Society

SCALLOPS

Inhalt

- Teil I: Grundlagen
- Teil II: Translation OWL-S nach WSDL

Inhalte von Teil I

1. Service Oriented Architecture (SOA)
   - Technologie
   - Eigenschaften
   - Das Web Services Framework
   - Beispiele
   - Scenario: SOA im Bereich Health Care, Telemonitoring
2. Semantic Web Services
   - Semantic Web
   - Web Ontology Language for Services (OWL-S)
3. Interoperabilität von Semantischen Web Services
   - Technologien

Inhalte von Teil II

Teil I

Grundlagen

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Service Orientierte Architektur

engl.: service oriented architecture (SOA)

- Managementkonzept
- Systemarchitekturkonzept
- Infrastruktur zur Abbildung von Geschäftsprozessen
- Trennung der Geschäftsprozess-Logik von der Implementierung
- Implementierung von Service-Komponenten (Services)
- Komposition und Orchestrierung von Service-Komponenten *(composite applications)*

Anwendungen innerhalb einer SOA

Rollen in einer SOA, Anwendungskomponenten

- Service Provider
- Service Consumer
- Zusammenspiel mehrerer Services *(interoperability)*

Eigenschaften einer SOA, Anwendungsentwicklung

- Lose Kopplung der Komponenten (Services)
- Abbildung von Geschäftsprozessen durch flexible Service-Komposition und Orchestrierung (Ablaufspläne)
- Standardisierte Schnittstellen
- Realisation eines Services als *BlackBox*

Anwendungen innerhalb einer SOA

Rollen in einer SOA, Anwendungskomponenten

- Service Provider
- Service Consumer
- Zusammenspiel mehrerer Services *(interoperability)*

Eigenschaften einer SOA, Anwendungsentwicklung

- Lose Kopplung der Komponenten (Services)
- Abbildung von Geschäftsprozessen durch flexible Service-Komposition und Orchestrierung (Ablaufspläne)
- Standardisierte Schnittstellen
- Realisation eines Services als *BlackBox*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Vorteile der SOA

- Wiederverwendbarkeit von Services (Software-Komponenten)
- Flexible Service Komposition ermöglicht schnelle Reaktion auf Bedürfnisse *(time to market)*
- Servicebausteine unabhängig voneinander erweiterbar
- Plattformunabhängigkeit
- Migration wird durch SOA-Wrapper vereinfacht

Bedeutung von Schnittstellen:

- SOA unterstützt die Nutzung von Schnittstellenteilen
- Standardisierung von Schnittstellen (Flexibilität)
  Beispiel: flexible Anpassung der *Supply Chain*

Das Web Services Framework

- Grundbausteine sind Web Services
- Austausch von Nachrichten (Messaging Framework SOAP) zwischen *Service Provider* und dem *Service Requester*
- Herstellerunabhängig, plattformneutral
- Gemeinsames Kommunikationsagreement (WSDL)
- Registrierung und erweiterte Beschreibung des Service in einem Verzeichnisdienst (UDDI)

Tasks innerhalb einer SOA (Agentensysteme)

Tasks, typischerweise durch Agenten durchgeführt:

- Lokalisierung von Web Services *(engl. discovery)*
- Ausführung von Web Services *(engl. invocation)*
- Zusammenstellung von Diensten *(engl. composition)*
- Verifikation von Service-Beschreibungen
- Beobachtung der Ausführung *(engl. monitoring)*

Technologien zum Auffinden von Web Services:

- UDDI Registry
- Ähnlichkeitsbasierte Suche (Matchmaking)

Tasks innerhalb einer SOA (Agentensysteme)

Tasks, typischerweise durch Agenten durchgeführt:

- Lokalisierung von Web Services *(engl. discovery)*
- Ausführung von Web Services *(engl. invocation)*
- Zusammenstellung von Diensten *(engl. composition)*
- Verifikation von Service-Beschreibungen
- Beobachtung der Ausführung *(engl. monitoring)*

Technologien zum Auffinden von Web Services:

- UDDI Registry
- Ähnlichkeitsbasierte Suche (Matchmaking)

SOA Anwendungsbeispiele

- eCommerce: Preisauskunft, Shopsysteme
- HTW: Einführung eines **Mo**dul **P**rogramm**P**lanungs**S**ystems
- Bereich Health Care: Telemedizin, Telemonitoring

Ausrichtung des *Business IT Alignment* bzgl.:

- Anforderungsmanagement
- Requirementmanagement
- Workflow Management (Geschäftsmodell)

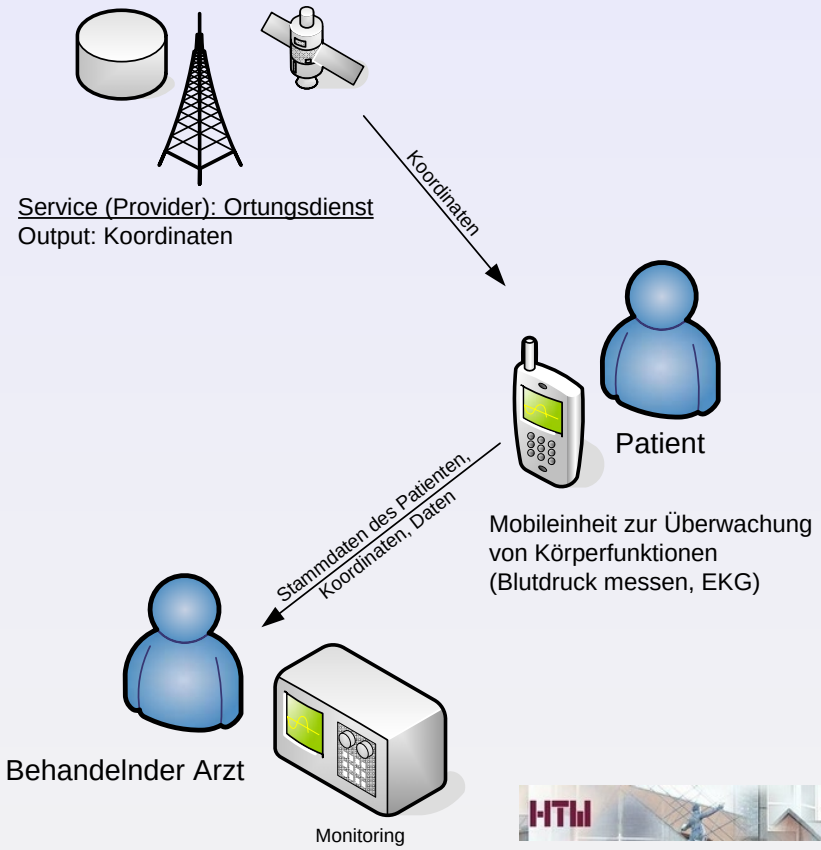
SOA Anwendungsbeispiele

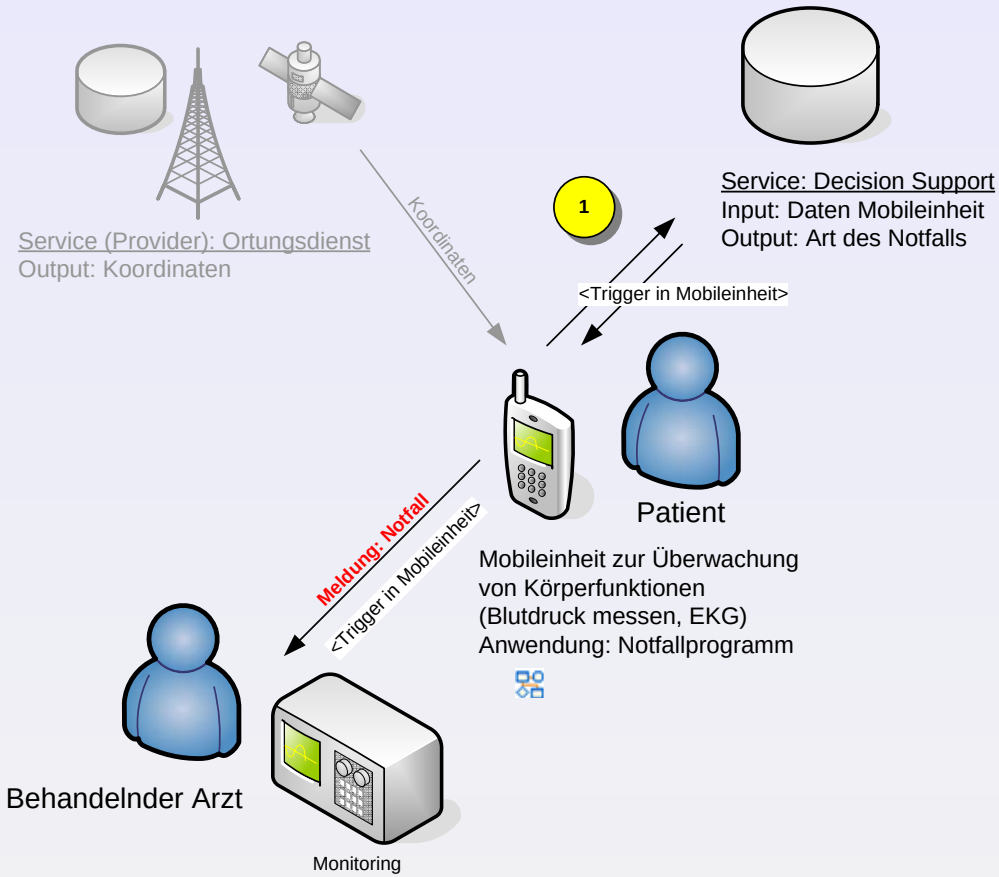
- eCommerce: Preisauskunft, Shopsysteme
- HTW: Einführung eines **Mo**dul **P**rogramm**P**lanungs**S**ystems
- Bereich Health Care: Telemedizin, Telemonitoring

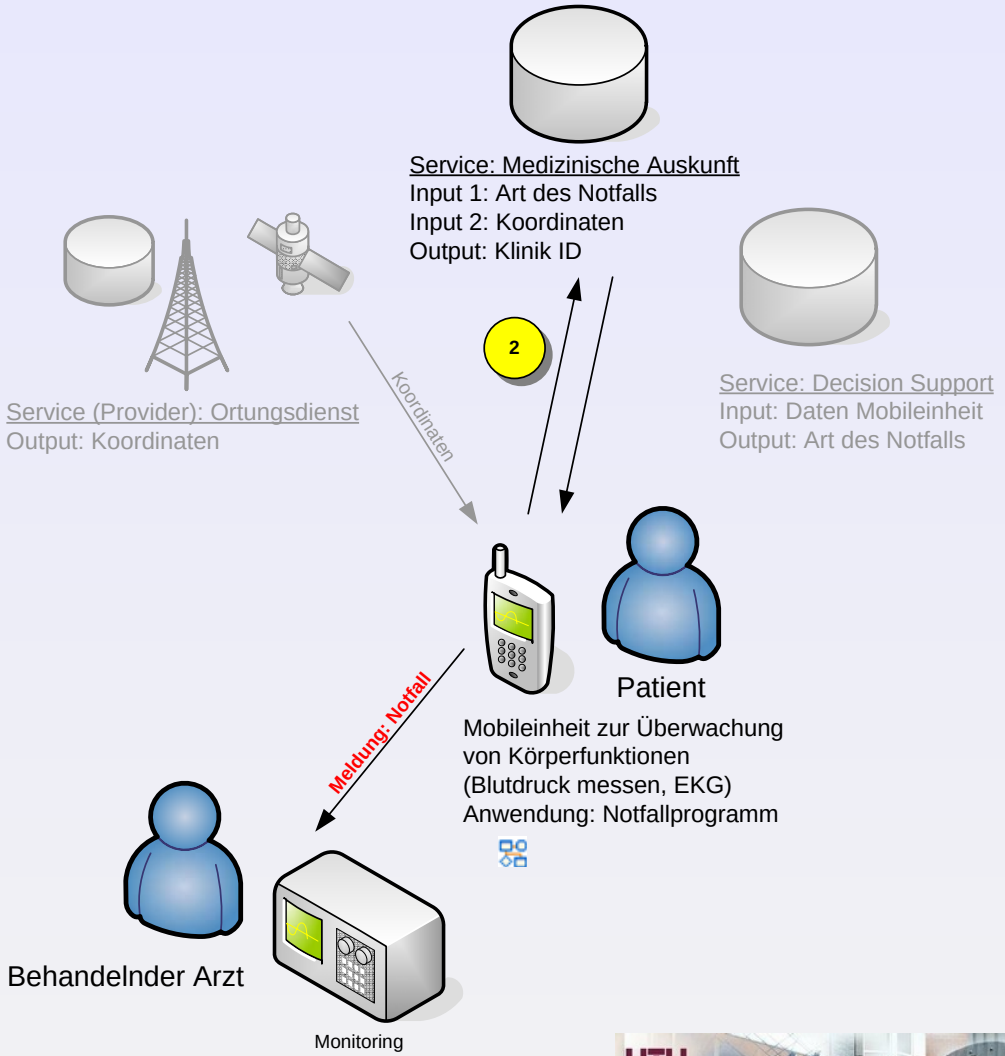
Ausrichtung des *Business IT Alignment* bzgl.:

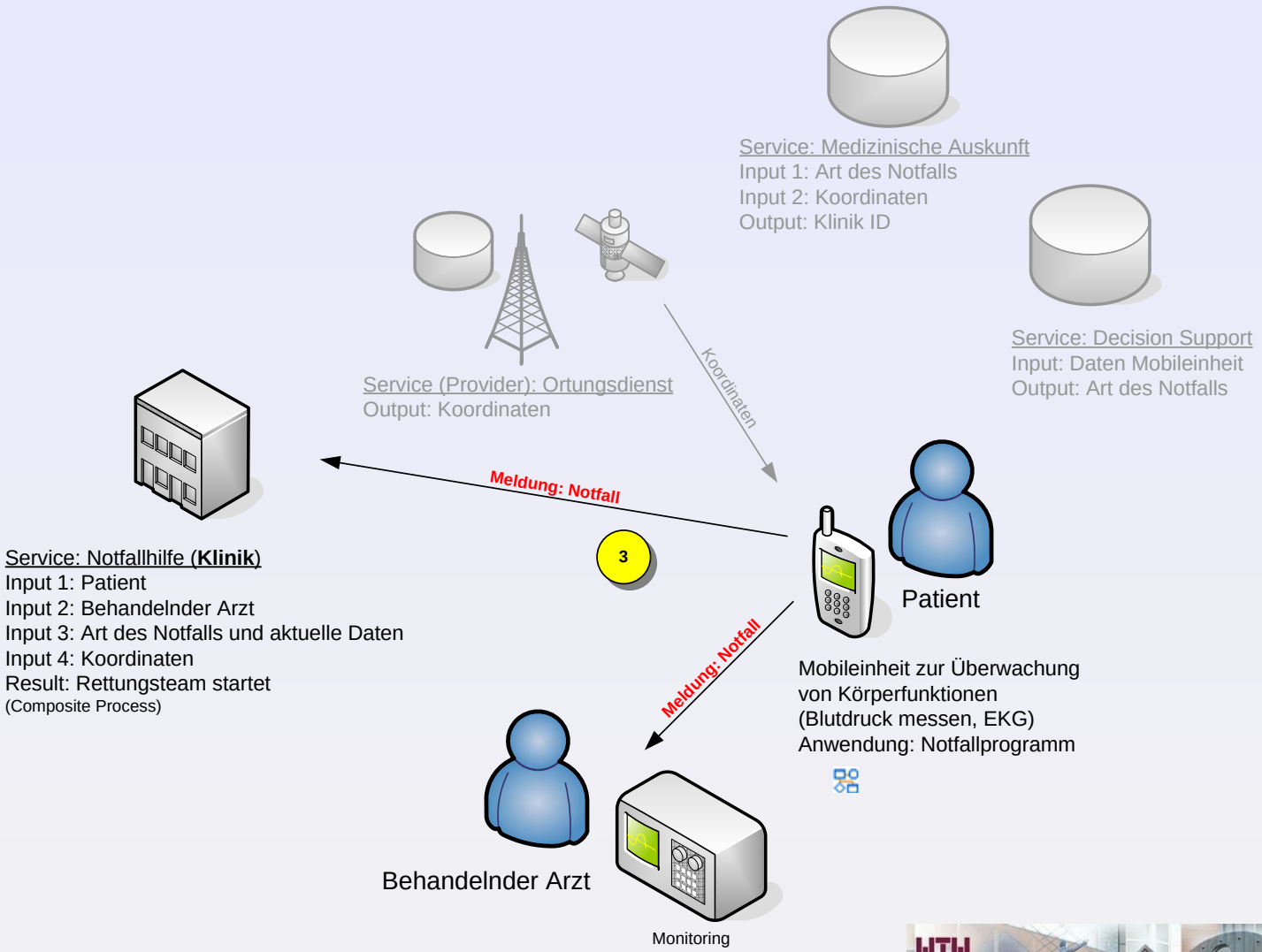
- Anforderungsmanagement
- Requirementmanagement
- Workflow Management (Geschäftsmodell)

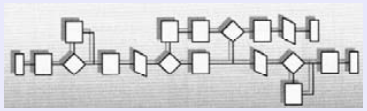
Telemonitoring mit einem Embbeded System




Service (Provider): Ortungsdienst
Output: Koordinaten
Koordinaten
1
Service: Decision Support
Input: Daten Mobileinheit
Output: Art des Notfalls
<Trigger in Mobileinheit>
Patient
Mobileinheit zur Überwachung
von Körperfunktionen
(Blutdruck messen, EKG)
Anwendung: Notfallprogramm
Meldung: Notfall
<Trigger in Mobileinheit>
Behandelnder Arzt
Monitoring


Service: Medizinische Auskunft
Input 1: Art des Notfalls
Input 2: Koordinaten
Output: Klinik ID
2
Service (Provider): Ortungsdienst
Output: Koordinaten
Koordinaten
Service: Decision Support
Input: Daten Mobileinheit
Output: Art des Notfalls
Patient
Mobileinheit zur Überwachung
von Körperfunktionen
(Blutdruck messen, EKG)
Anwendung: Notfallprogramm
Meldung: Notfall
Behandelnder Arzt
Monitoring


Service: Medizinische Auskunft
Input 1: Art des Notfalls
Input 2: Koordinaten
Output: Klinik ID
Service: Decision Support
Input: Daten Mobileinheit
Output: Art des Notfalls
Service (Provider): Ortungsdienst
Output: Koordinaten
Koordinaten
Meldung: Notfall
3
Service: Notfallhilfe (Klinik)
Input 1: Patient
Input 2: Behandelnder Arzt
Input 3: Art des Notfalls und aktuelle Daten
Input 4: Koordinaten
Result: Rettungsteam startet
(Composite Process)
Patient
Mobileinheit zur Überwachung
von Körperfunktionen
(Blutdruck messen, EKG)
Anwendung: Notfallprogramm
Meldung: Notfall
Behandelnder Arzt
Monitoring

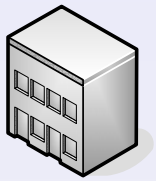


Services (Orchestrierung + BPEL):
- Auswahl des Transportmittels
- Buchung des Transportmittels
- Buchung von Sanitätern

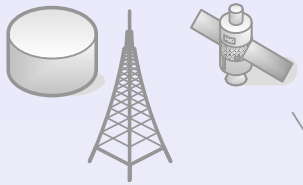
Je nach Fall:
- Buchung Ärzte-Team
- Buchung OP

usw.

Service: Arztinformation
Output 1: Patient
Output 2: Maßnahmen

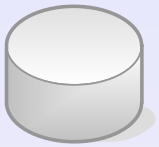


Service: Notfallhilfe (**Klinik**)
Input 1: Patient
Input 2: Behandelnder Arzt
Input 3: Art des Notfalls und Daten
Input 4: Koordinaten
Result: Rettungsteam startet
(Composite Process)

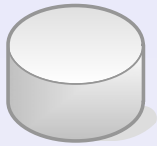


Service (Provider): Ortungsdienst
Output: Koordinaten

Koordinaten



Service: Medizinische Auskunft
Input 1: Art des Notfalls
Input 2: Koordinaten
Output: Klinik ID



Service: Decision Support
Input: Daten Mobileinheit
Output: Art des Notfalls

Meldung: Notfall

Meldung: Hilfe unterwegs!

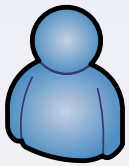
Meldung: Hilfe für Patient unterwegs

Meldung: Notfall

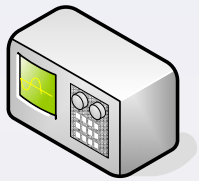
Patient

Mobileinheit zur Überwachung von Körperfunktionen (Blutdruck messen, EKG)
Anwendung: Notfallprogramm

4



Behandelnder Arzt



Monitoring

Semantic Web Services

- Web Services Framework (WSDL)
- Semantic Web (OWL)
- Semantische Dienstbeschreibung
  - Neue Definition (OWL-S) inkl. Grounding (WSDL)
  - Annotationen (WSDL-S)

Das Semantic Web

Wissensrepräsentation im Web über

- **Taxonomien**: Monohierarchische Klassenstruktur
- **Ontologien**: Netz von Klassenhierarchien und Beziehungen
- **Logik**: Schlussfolgerungen *(engl. reasoning)*

> *"An ontology is a specification of a conceptualization."*
> *Tom Gruber (1994)*

Definiton Semantic Web:

> *"Netz von Daten, die direkt und indirekt von Maschinen verarbeitet werden können." (Tim Berners-Lee, W3C)*

Das Semantic Web

Wissensrepräsentation im Web über

- **Taxonomien**: Monohierarchische Klassenstruktur
- **Ontologien**: Netz von Klassenhierarchien und Beziehungen
- **Logik**: Schlussfolgerungen *(engl. reasoning)*

> *"An ontology is a specification of a conceptualization."*
> *Tom Gruber (1994)*

Definiton Semantic Web:

> *"Netz von Daten, die direkt und indirekt von Maschinen verarbeitet werden können." (Tim Berners-Lee, W3C)*

Web Ontologogy Language (OWL)

OWL Sprachumfang:

- Klassen *(engl. concepts)* und Individuen
- Eigenschaften: Beziehungen in Tripleform (RDF)
  - Datatype properties
  - Object properties (Beziehungen zwischen Klassen)
- Restriktionen von Eigenschaften

OWL Untersprachen:

- OWL-Lite
- OWL-DL *(description logic)*
- OWL-Full

Web Ontologogy Language (OWL)

OWL Sprachumfang:

- Klassen *(engl. concepts)* und Individuen
- Eigenschaften: Beziehungen in Tripleform (RDF)
  - Datatype properties
  - Object properties (Beziehungen zwischen Klassen)
- Restriktionen von Eigenschaften

OWL Untersprachen:

- OWL-Lite
- OWL-DL *(description logic)*
- OWL-Full

Web Ontology Language for Services (OWL-S)

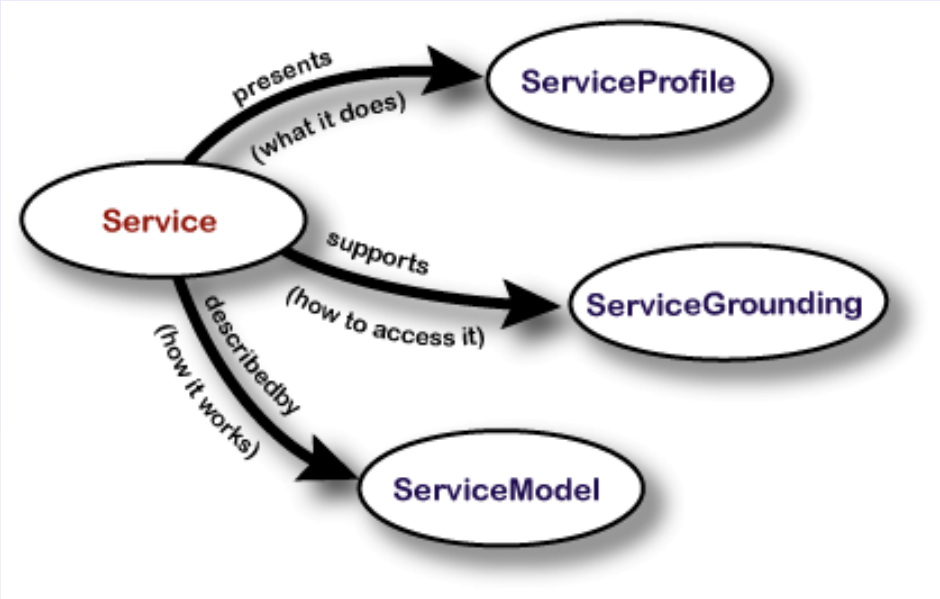


Abbildung: OWL-S: Upper ontology

OWL-S Ontologien (OWL)

- *ServiceProfile* enthält Service Informationen (vgl. UDDI)
- *ProcessModel* Ablaufssteuerung
- *Grounding* referenziert WSDL

Definition der Schnittstelle

Service Parameter *(AtomicProcess)* primitiv oder semantisch definiert.

Nutzung weiterer Ontologien wie z.B. ActorDefault.owl: “Actor represents a Requester or Provider who might request or offer a service.”

Web Ontology Language for Services (OWL-S)

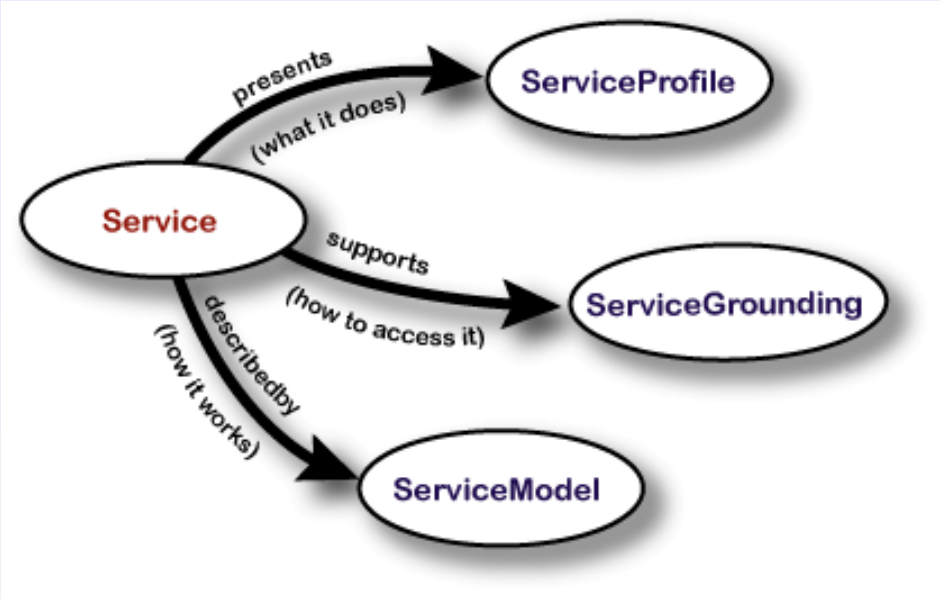


Abbildung: OWL-S: Upper ontology

OWL-S Ontologien (OWL)

- *ServiceProfile* enthält Service Informationen (vgl. UDDI)
- *ProcessModel* Ablaufssteuerung
- *Grounding* referenziert WSDL

Definition der Schnittstelle

Service Parameter *(AtomicProcess)* primitiv oder semantisch definiert.

Nutzung weiterer Ontologien wie z.B. ActorDefault.owl: “Actor represents a Requester or Provider who might request or offer a service.”

Web Ontology Language for Services (OWL-S)

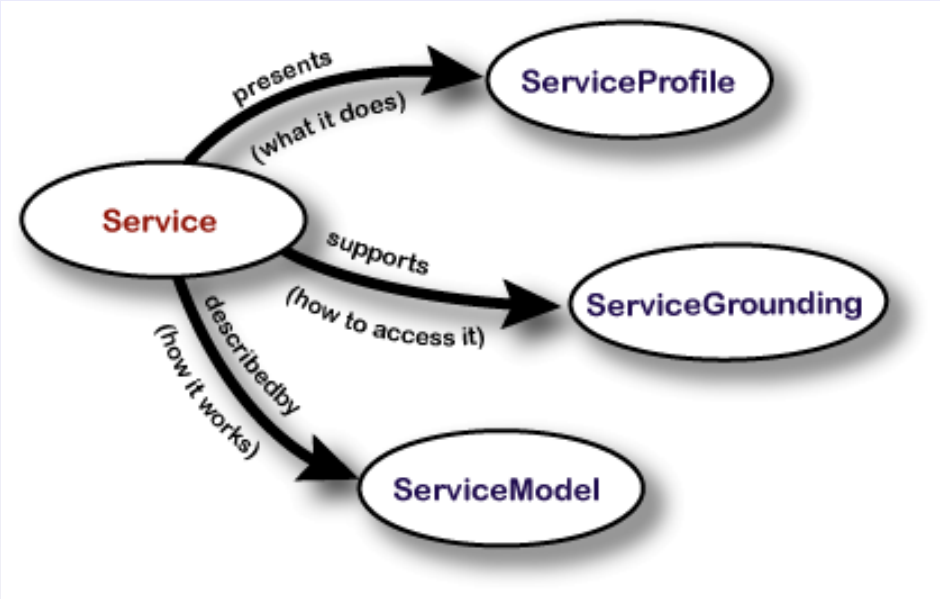


Abbildung: OWL-S: Upper ontology

OWL-S Ontologien (OWL)

- *ServiceProfile* enthält Service Informationen (vgl. UDDI)
- *ProcessModel* Ablaufssteuerung
- *Grounding* referenziert WSDL

Definition der Schnittstelle

Service Parameter *(AtomicProcess)* primitiv oder semantisch definiert.

Nutzung weiterer Ontologien wie z.B. ActorDefault.owl: “Actor represents a Requester or Provider who might request or offer a service.”

Web Ontology Language for Services (OWL-S)

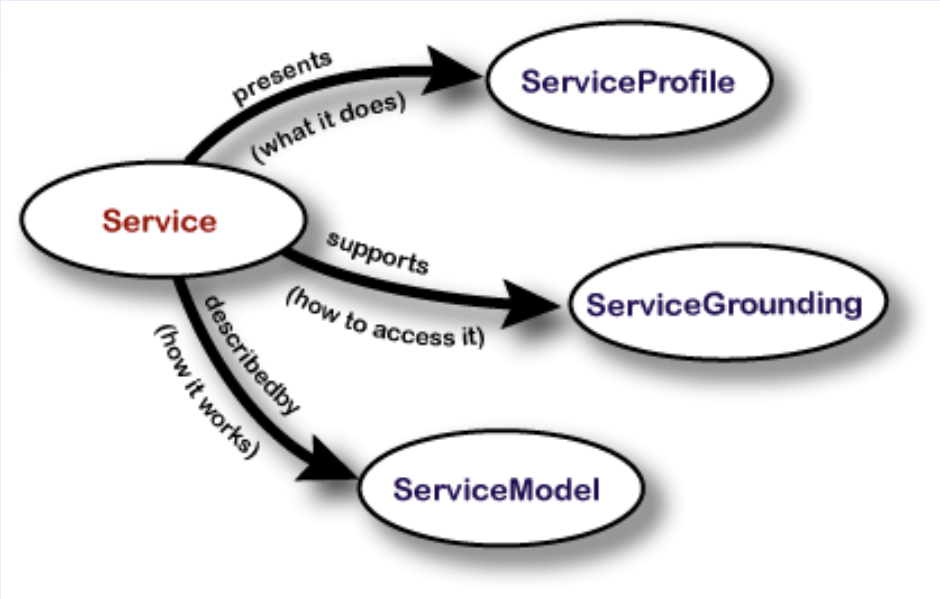


Abbildung: OWL-S: Upper ontology

OWL-S Ontologien (OWL)

- *ServiceProfile* enthält Service Informationen (vgl. UDDI)
- *ProcessModel* Ablaufssteuerung
- *Grounding* referenziert WSDL

Definition der Schnittstelle

Service Parameter *(AtomicProcess)* primitiv oder semantisch definiert.

Nutzung weiterer Ontologien wie z.B. ActorDefault.owl: "Actor represents a Requester or Provider who might request or offer a service."

Service Beschreibung

Prozesse *(ProcessModel)*

- AtomicProcess, SimpleProcess, CompositeProcess
- Parametertypen: Primitive Datentypen oder OWL-Klassen

AtomicProcess *(engl.capabilities)*

- Eingaben *(inputs)*
- Ausgaben *(outputs)*
- Vorbedingungen *(preconditions)*
- Nachbedingungen *(results)*

Ein- und Ausgaben eines Prozesses bilden dessen Signatur.

Service Beschreibung

Prozesse *(ProcessModel)*

- AtomicProcess, SimpleProcess, CompositeProcess
- Parametertypen: Primitive Datentypen oder OWL-Klassen

AtomicProcess *(engl.capabilities)*

- Eingaben *(inputs)*
- Ausgaben *(outputs)*
- Vorbedingungen *(preconditions)*
- Nachbedingungen *(results)*

Ein- und Ausgaben eines Prozesses bilden dessen Signatur.

Service Beschreibung

Prozesse *(ProcessModel)*

- AtomicProcess, SimpleProcess, CompositeProcess
- Parametertypen: Primitive Datentypen oder OWL-Klassen

AtomicProcess *(engl.capabilities)*

- Eingaben *(inputs)*
- Ausgaben *(outputs)*
- Vorbedingungen *(preconditions)*
- Nachbedingungen *(results)*

Ein- und Ausgaben eines Prozesses bilden dessen Signatur.

Grounding / Mapping der Service-Signatur

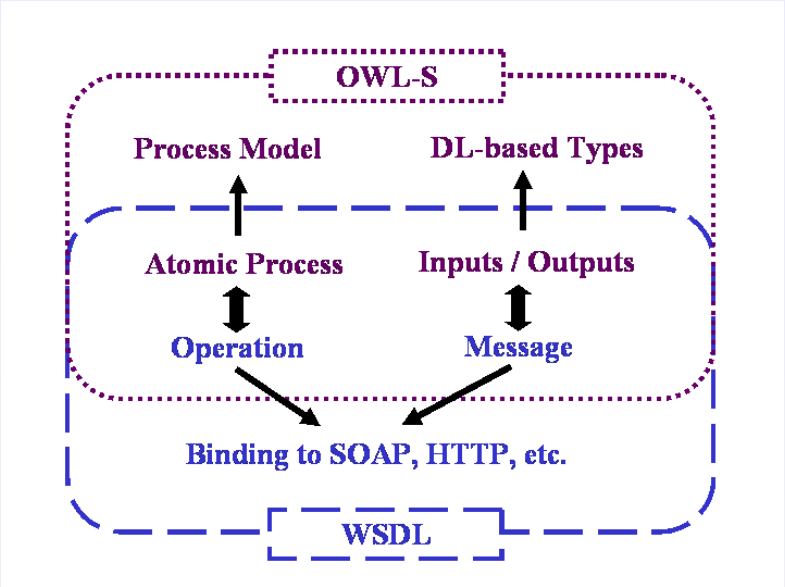


Abbildung: WSDL Grounding (W3C)

Grounding / Mapping der Service-Signatur

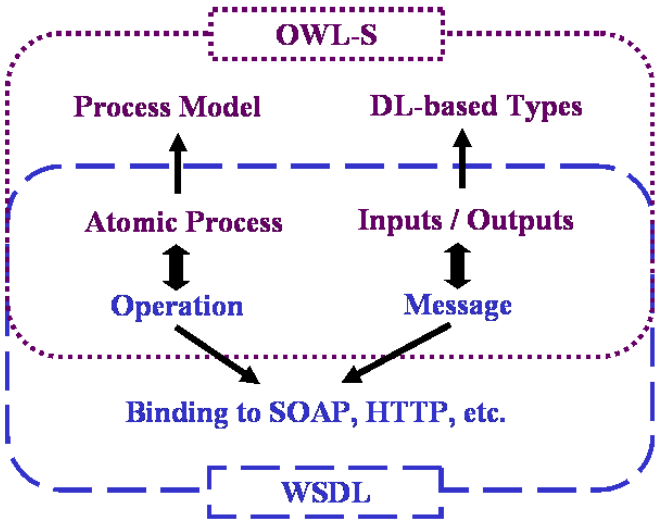


Abbildung: WSDL Grounding (W3C)

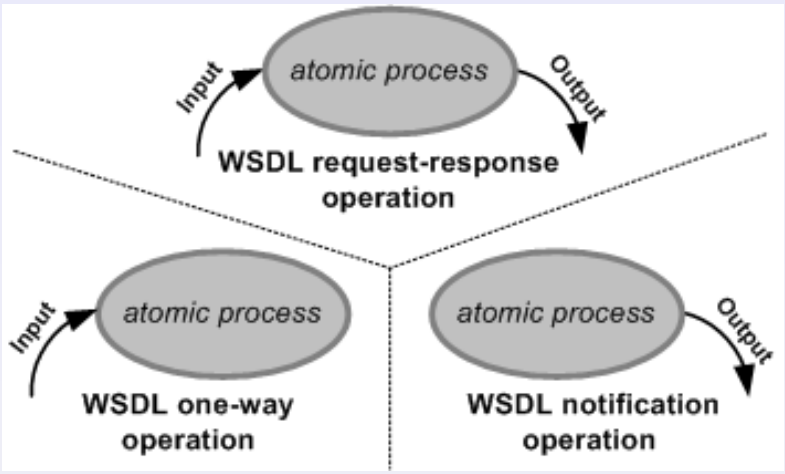


Abbildung: Mapping OWL-S, WSDL

Interoperabilität von Semantischen Web Services

Semantische Information unterstützt:

- Service Composition und Orchestrierung
  - Service Composition Planer
  - Business Process Execution Language (BPEL)
- Service Discovering
  - Verzeichnisdienst UDDI (konventionell)
  - Ähnlichkeitsbasierte Suche (Matchmaking)

Matchmaking

- Syntaktisches Matchmaking (WSDL Analyzer)
- Semantisches Matchmaking
- Hybrid Semantisches Matchmaking (OWLS-MX)

Interoperabilität von Semantischen Web Services

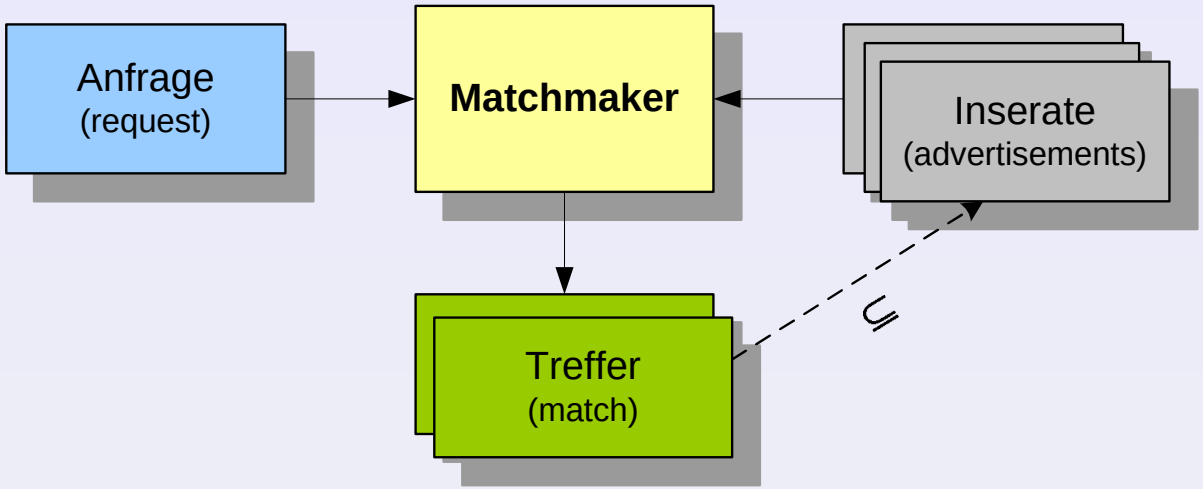
Semantische Information unterstützt:

- Service Composition und Orchestrierung
  - Service Composition Planer
  - Business Process Execution Language (BPEL)
- Service Discovering
  - Verzeichnisdienst UDDI (konventionell)
  - Ähnlichkeitsbasierte Suche (Matchmaking)

Matchmaking

- Syntaktisches Matchmaking (WSDL Analyzer)
- Semantisches Matchmaking
- Hybrid Semantisches Matchmaking (OWLS-MX)

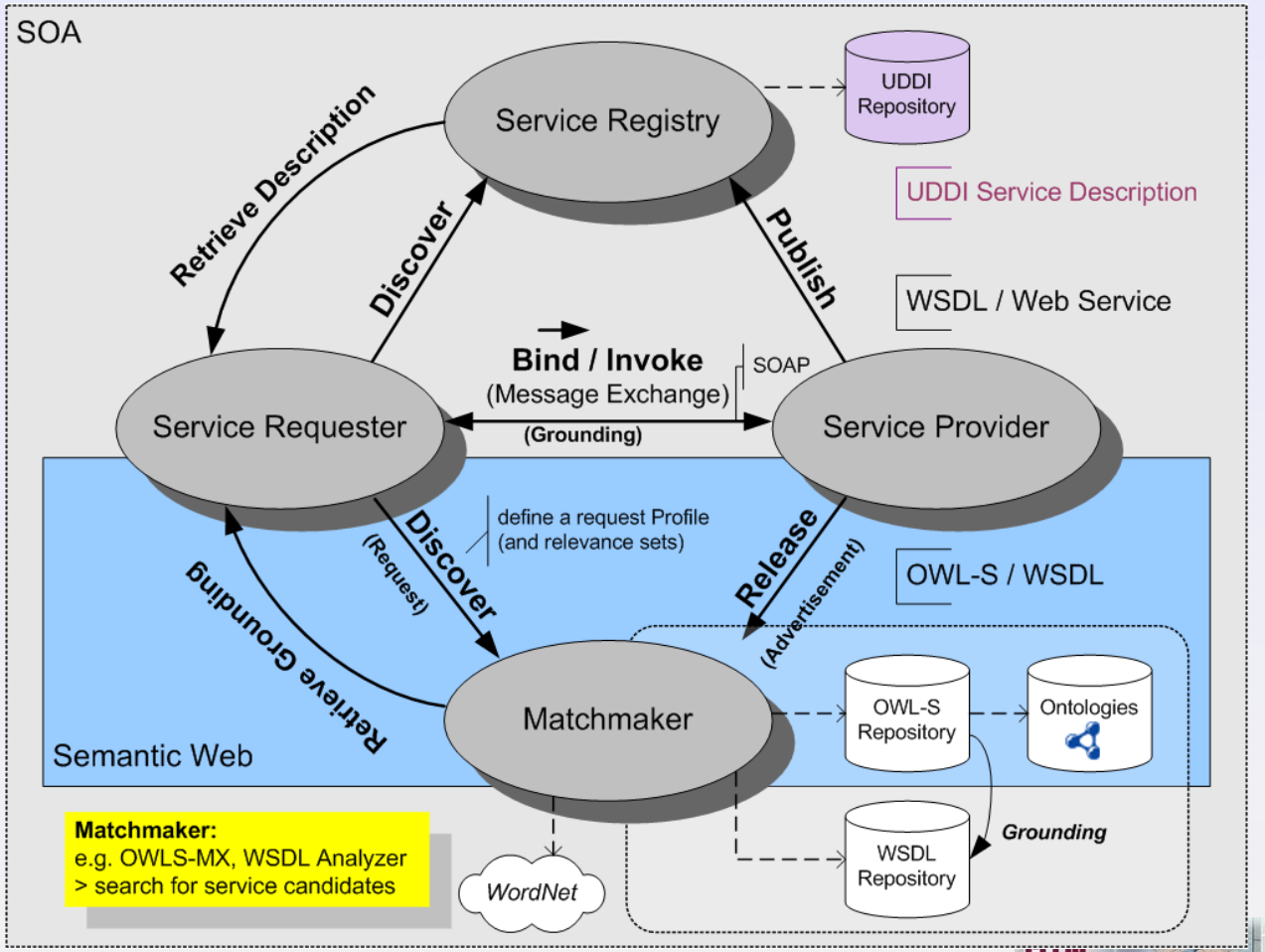
Semantische Matchmaker



Ähnlichkeitsbestimmung über

- Vergleich von Signaturen
- Auswertung von Vererbungshierarchien
- Schlussfolgerungen *(engl. reasoning)*

Technologien zum Auffinden von Web Services



Übersicht Technologien

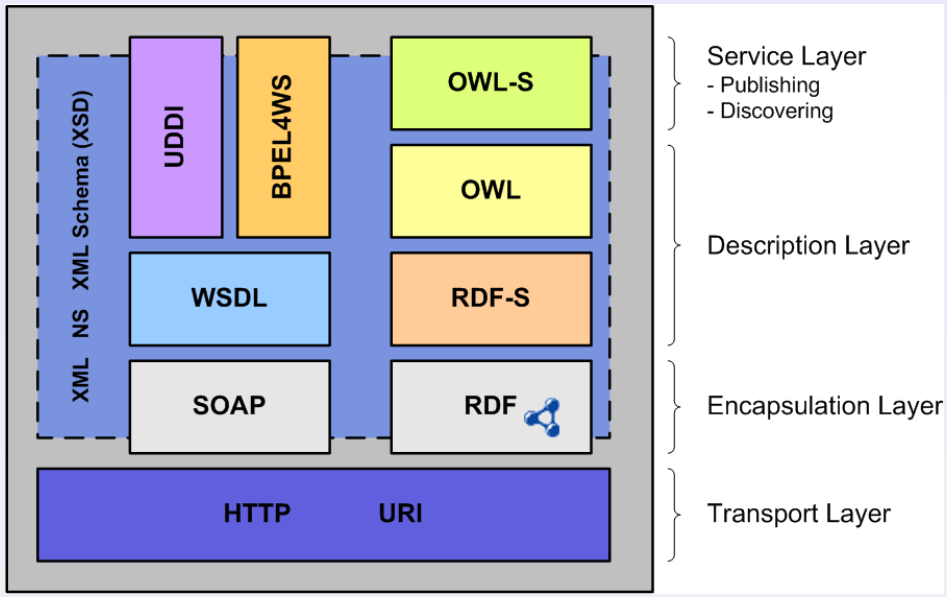


Abbildung: Vergleich “Protokollstack” von WSDL und OWL-S

Teil II

Besprechung des Themas

Inhalte von Teil II

Ziele dieser Arbeit

1. Untersuchung der Möglichkeiten der Generierung von WSDL aus OWL-S und Konkretisierung von WSDL Groundings
2. Semi-automatische Translation von OWL-S nach WSDL (Toolentwicklung)
3. Vergleichende Analyse der Performanz einer ähnlichkeitsbasierten Dienstsuche mit *WSDL Analyzer* und *OWLS-MX* über eine entsprechend aufgebaute Testkollektion.

Ziele dieser Arbeit

1. Untersuchung der Möglichkeiten der Generierung von WSDL aus OWL-S und Konkretisierung von WSDL Groundings
2. Semi-automatische Translation von OWL-S nach WSDL (Toolentwicklung)
3. Vergleichende Analyse der Performanz einer ähnlichkeitsbasierten Dienstsuche mit *WSDL Analyzer* und *OWLS-MX* über eine entsprechend aufgebaute Testkollektion.

Ziele dieser Arbeit

1. Untersuchung der Möglichkeiten der Generierung von WSDL aus OWL-S und Konkretisierung von WSDL Groundings
2. Semi-automatische Translation von OWL-S nach WSDL (Toolentwicklung)
3. Vergleichende Analyse der Performanz einer ähnlichkeitsbasierten Dienstsuche mit *WSDL Analyzer* und *OWLS-MX* über eine entsprechend aufgebaute Testkollektion.

Analyse

- Diskussion *top down* Ansatz
- Untersuchung des Mappings OWL-S auf WSDL
- Einordnung des Themas, Abgrenzung
- Bedeutung der Translation OWL nach XML Schema
- Abhängigkeiten (Validierung, Evaluierungsergebnisse)

Vorgehensmodell

- Anlehnung an das *Clean Room* Vorgehensmodell

Analyse

- Diskussion *top down* Ansatz
- Untersuchung des Mappings OWL-S auf WSDL
- Einordnung des Themas, Abgrenzung
- Bedeutung der Translation OWL nach XML Schema
- Abhängigkeiten (Validierung, Evaluierungsergebnisse)

Vorgehensmodell

- Anlehnung an das *Clean Room* Vorgehensmodell

Use Cases (SOA)

Rollen

- Ontology Engineer
- Web Services Engineer
- Application Developer
- SOA Expert
- Service Consumer

Basic Use Cases

- Provide (Semantic) Web Service
- Discover (Semantic) Web Service
- Invoke (Semantic) Web Service

Translations

- OWL2XSD
- **OWLS2WSDL**, WSDL2OWL-S
- OWLS2BPEL

Use Cases (SOA)

Rollen

- Ontology Engineer
- Web Services Engineer
- Application Developer
- SOA Expert
- Service Consumer

Basic Use Cases

- Provide (Semantic) Web Service
- Discover (Semantic) Web Service
- Invoke (Semantic) Web Service

Translations

- OWL2XSD
- **OWLS2WSDL**, WSDL2OWL-S
- OWLS2BPEL

Use Cases (SOA)

Rollen

- Ontology Engineer
- Web Services Engineer
- Application Developer
- SOA Expert
- Service Consumer

Basic Use Cases

- Provide (Semantic) Web Service
- Discover (Semantic) Web Service
- Invoke (Semantic) Web Service

Translations

- OWL2XSD
- **OWLS2WSDL**, WSDL2OWL-S
- OWLS2BPEL

Konzeptionierung und Design

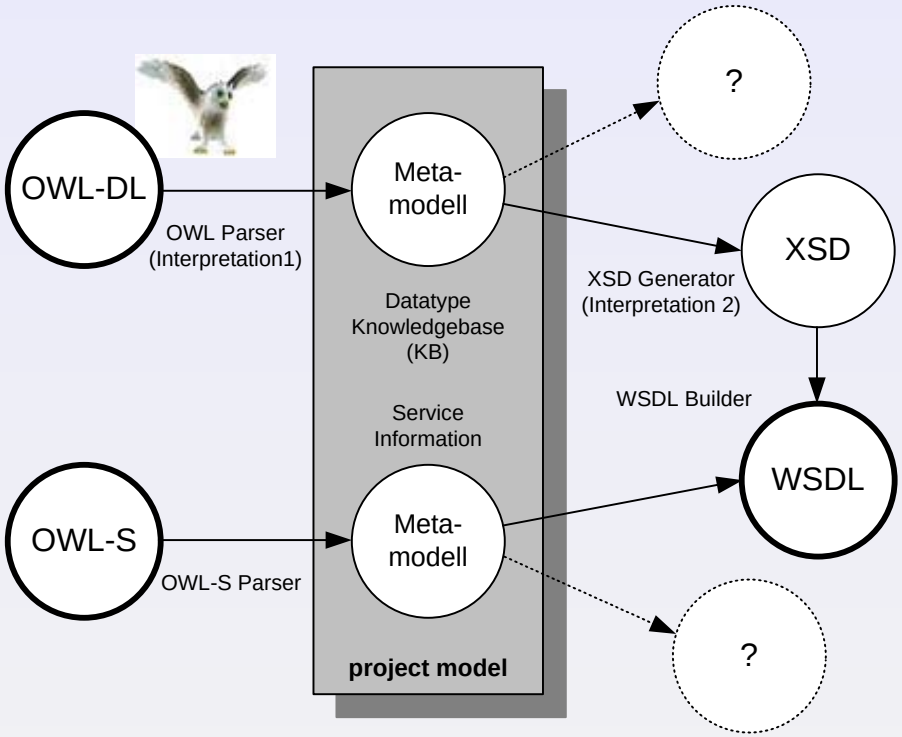
Design Direktiven

- Nutzung von Standards
- Trennung von Parsen und Code Generierung
- Wissensbasis für Datentypen
- Metamodell für Datentypen und Service-Beschreibungen

Konzeptionierung und Design

Design Direktiven

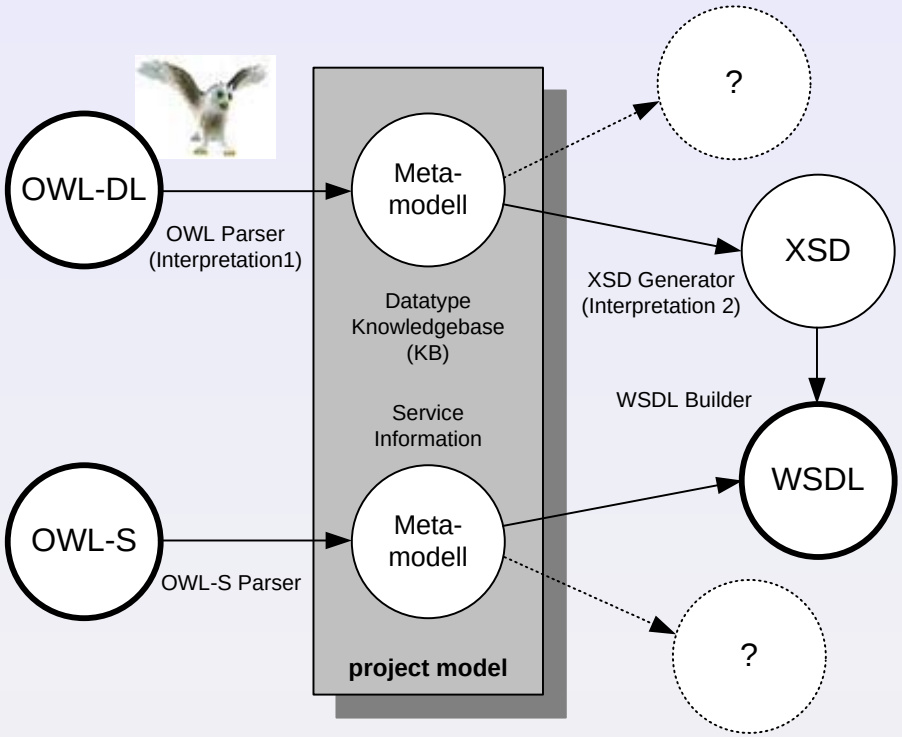
- Nutzung von Standards
- Trennung von Parsen und Code Generierung
- Wissensbasis für Datentypen
- Metamodell für Datentypen und Service-Beschreibungen



Konzeptionierung und Design

Design Direktiven

- Nutzung von Standards
- Trennung von Parsen und Code Generierung
- Wissensbasis für Datentypen
- Metamodell für Datentypen und Service-Beschreibungen



Herausforderung: Interpretation von OWL-DL

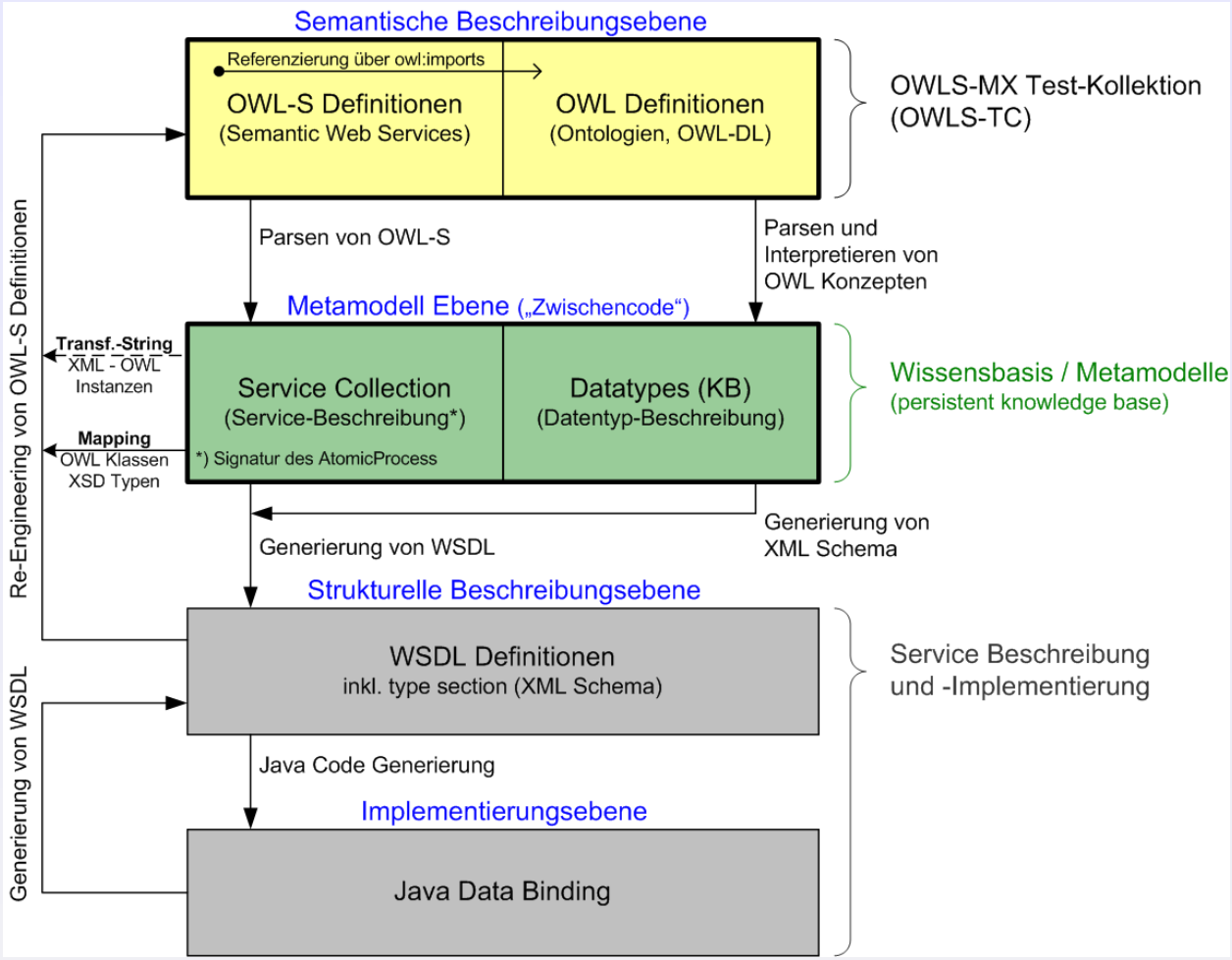


Abbildung: Translation-Stack

OWL Parser

Parsen von OWL (Definitionen)

- Reguläre Klassen *(concepts)* und anonyme Klassen (DL)
- Eigenschaften
- Kardinalitäten und Wertebereich (Individuals)
- Klassenhierarchien *(depth)*

Interpretieren von OWL Konzepten (Meta-Ebene)

- Bildung von Datentypen aus OWL-Klassen
- Mapping auf passende XML Schema Elemente
- Abbildung der OWL Grammatik mit XML Schema *(experimental)*

OWL Parser

Parsen von OWL (Definitionen)

- Reguläre Klassen *(concepts)* und anonyme Klassen (DL)
- Eigenschaften
- Kardinalitäten und Wertebereich (Individuals)
- Klassenhierarchien *(depth)*

Interpretieren von OWL Konzepten (Meta-Ebene)

- Bildung von Datentypen aus OWL-Klassen
- Mapping auf passende XML Schema Elemente
- Abbildung der OWL Grammatik mit XML Schema *(experimental)*

XSD Generator

Generierung von

- Elementen mit Typinformation
- Kardinalitäten *(particles)*
- SimpleType Typen
- ComplexType Typen
- Ableitungen neuer Typen *(restriction, extension)*

Entwurfsmuster

- Venetian Blind
- Hierarchy Pattern

Konfigurationsmöglichkeiten

- Default für primitiver Typ
- Hierarchy Pattern
- Anonyme Typen

XSD Generator

Generierung von

- Elementen mit Typinformation
- Kardinalitäten *(particles)*
- SimpleType Typen
- ComplexType Typen
- Ableitungen neuer Typen *(restriction, extension)*

Entwurfsmuster

- Venetian Blind
- Hierarchy Pattern

Konfigurationsmöglichkeiten

- Default für primitiver Typ
- Hierarchy Pattern
- Anonyme Typen

XSD Generator

Generierung von

- Elementen mit Typinformation
- Kardinalitäten *(particles)*
- SimpleType Typen
- ComplexType Typen
- Ableitungen neuer Typen *(restriction, extension)*

Entwurfsmuster

- Venetian Blind
- Hierarchy Pattern

Konfigurationsmöglichkeiten

- Default für primitiver Typ
- Hierarchy Pattern
- Anonyme Typen

Parsen von OWL-S

- Signatur des atomaren Prozesses
- Festlegung der Parameter-Reihenfolge
- Speicherung des Mappings in Metamodell (Grounding)

Generieren von WSDL (XSD)

- *Straight Forward* anhand der abstrakten Service-Beschreibung
- Abhängigkeiten zu Parametertypen
- Manuelle Anpassungen an Datentypen
- Generierung von XML Schema Typen (für Parameter)

Parsen von OWL-S

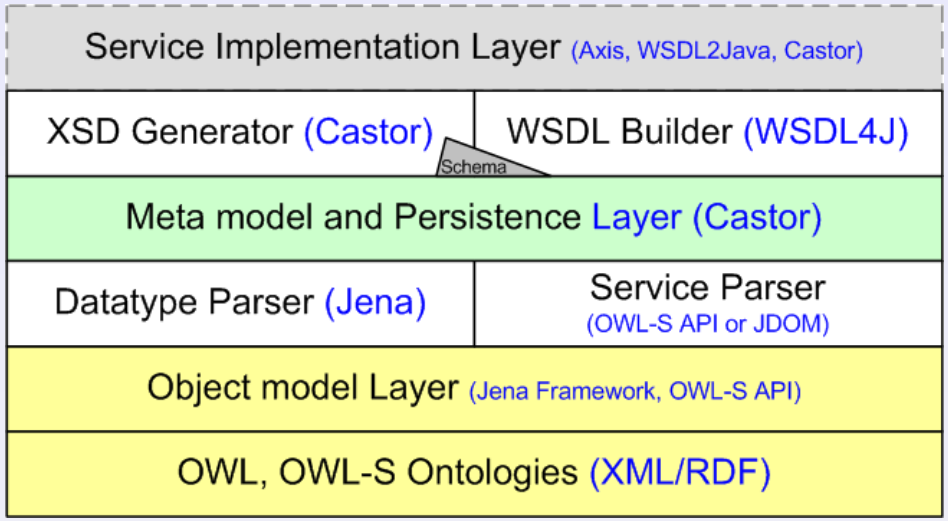
- Signatur des atomaren Prozesses
- Festlegung der Parameter-Reihenfolge
- Speicherung des Mappings in Metamodell (Grounding)

Generieren von WSDL (XSD)

- *Straight Forward* anhand der abstrakten Service-Beschreibung
- Abhängigkeiten zu Parametertypen
- Manuelle Anpassungen an Datentypen
- Generierung von XML Schema Typen (für Parameter)


OWL-S
Translation
to WSDL
OWLS2WSDL

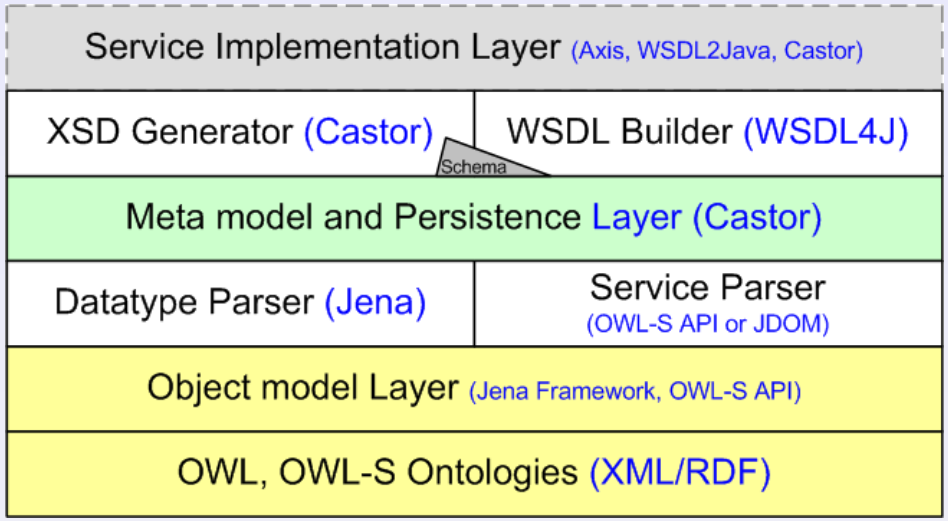
Programmteile



Weitere Funktionen

- Projekt (umfasst Service Collection und Knowledgebase)
- Command Line Interface
- Benutzeroberfläche (Konfiguration)

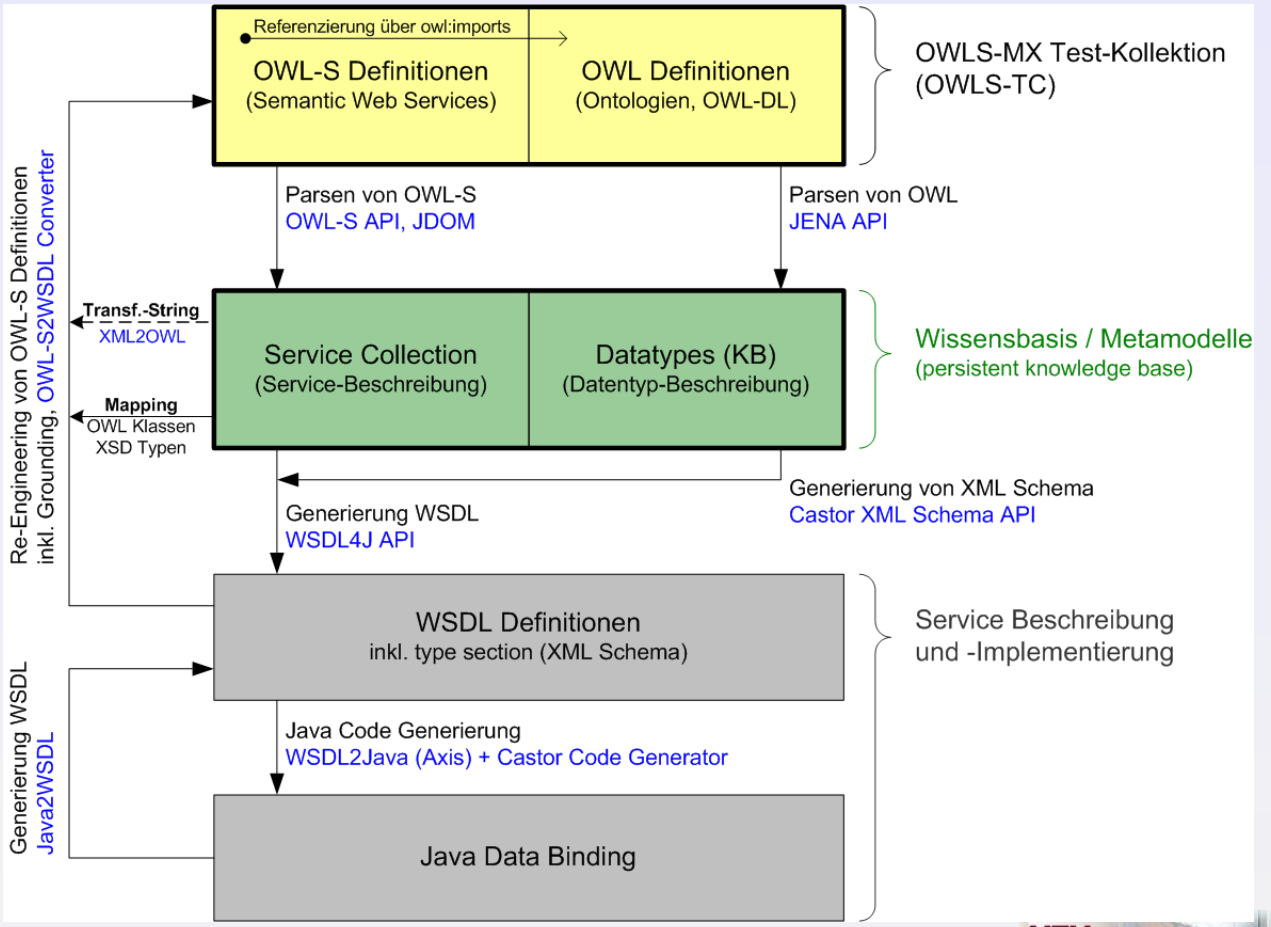
Programmteile



Weitere Funktionen

- Projekt (umfasst Service Collection und Knowledgebase)
- Command Line Interface
- Benutzeroberfläche (Konfiguration)

Technologien (APIs)



OWLS2WSDL Tool

Programmfunktionen

- Generierung von XML Schema aus OWL (OWL2XSD)
- Translation OWL-S nach WSDL (OWLS2WSDL)
- Re-Engineering: Konkretisierung eines WSDL Groundings

Arbeitsweise

1. Anlegen eines Projektes
2. Laden der OWL-S Definitionen (Parser)
3. Auflösen von Abhängigkeiten der Schnittstelle zu Datentypen
4. Konfiguration, optionale Bearbeitung der Typen
5. Generierung von WSDL-Beschreibungen (inkl. XML Schema)

Beispiele: ZipCode, Student (OWL2XSD); CarPriceService

OWLS2WSDL Tool

Programmfunktionen

- Generierung von XML Schema aus OWL (OWL2XSD)
- Translation OWL-S nach WSDL (OWLS2WSDL)
- Re-Engineering: Konkretisierung eines WSDL Groundings

Arbeitsweise

1. Anlegen eines Projektes
2. Laden der OWL-S Definitionen (Parser)
3. Auflösen von Abhängigkeiten der Schnittstelle zu Datentypen
4. Konfiguration, optionale Bearbeitung der Typen
5. Generierung von WSDL-Beschreibungen (inkl. XML Schema)

Beispiele: ZipCode, Student (OWL2XSD); CarPriceService

OWLS2WSDL Tool

Programmfunktionen

- Generierung von XML Schema aus OWL (OWL2XSD)
- Translation OWL-S nach WSDL (OWLS2WSDL)
- Re-Engineering: Konkretisierung eines WSDL Groundings

Arbeitsweise

1. Anlegen eines Projektes
2. Laden der OWL-S Definitionen (Parser)
3. Auflösen von Abhängigkeiten der Schnittstelle zu Datentypen
4. Konfiguration, optionale Bearbeitung der Typen
5. Generierung von WSDL-Beschreibungen (inkl. XML Schema)

Beispiele: ZipCode, Student (OWL2XSD); CarPriceService

Future Work

- Erweiterte Fehlerbehandlung (OWL Parser)
- Erweiterung OWL-S Parser (OWL-S API)
- WSDL Builder
  - Variation des Zielformats (WSDL-S)
  - Automatische Konfiguration (WA)
- XSL Transformation (Re-Engineering)
- Generierung von OWL-S 1.0
- Validierung von OWL-S Definitionen
- Integration von WSDL2Java

voluntarily contributions (OpenSource)

Future Work

- Erweiterte Fehlerbehandlung (OWL Parser)
- Erweiterung OWL-S Parser (OWL-S API)
- WSDL Builder
  - Variation des Zielformats (WSDL-S)
  - Automatische Konfiguration (WA)
- XSL Transformation (Re-Engineering)
- Generierung von OWL-S 1.0
- Validierung von OWL-S Definitionen
- Integration von WSDL2Java

voluntarily contributions (OpenSource)

Durchführung der Evaluierung

1. Gegeben: OWLS-TC mit Matchmakingwerten des OWLS-MX
   - 29 Queries (Referenzdienste)
   - *Relevance Sets*
2. Translation OWL-S nach WSDL
   - OWL-S *Query* wird zu WSDL *Requirement*
   - Dienstbeschreibungen des *Relevance Set* werden *Candidates*
3. Ermittlung von Ähnlichkeitswerten mit dem WA (Ranking)
4. Vergleich der Matchmakingwerte (WA und OWLS-MX)

Durchführung der Evaluierung

1. Gegeben: OWLS-TC mit Matchmakingwerten des OWLS-MX
   - 29 Queries (Referenzdienste)
   - *Relevance Sets*
2. Translation OWL-S nach WSDL
   - OWL-S *Query* wird zu WSDL *Requirement*
   - Dienstbeschreibungen des *Relevance Set* werden *Candidates*
3. Ermittlung von Ähnlichkeitswerten mit dem WA (Ranking)
4. Vergleich der Matchmakingwerte (WA und OWLS-MX)

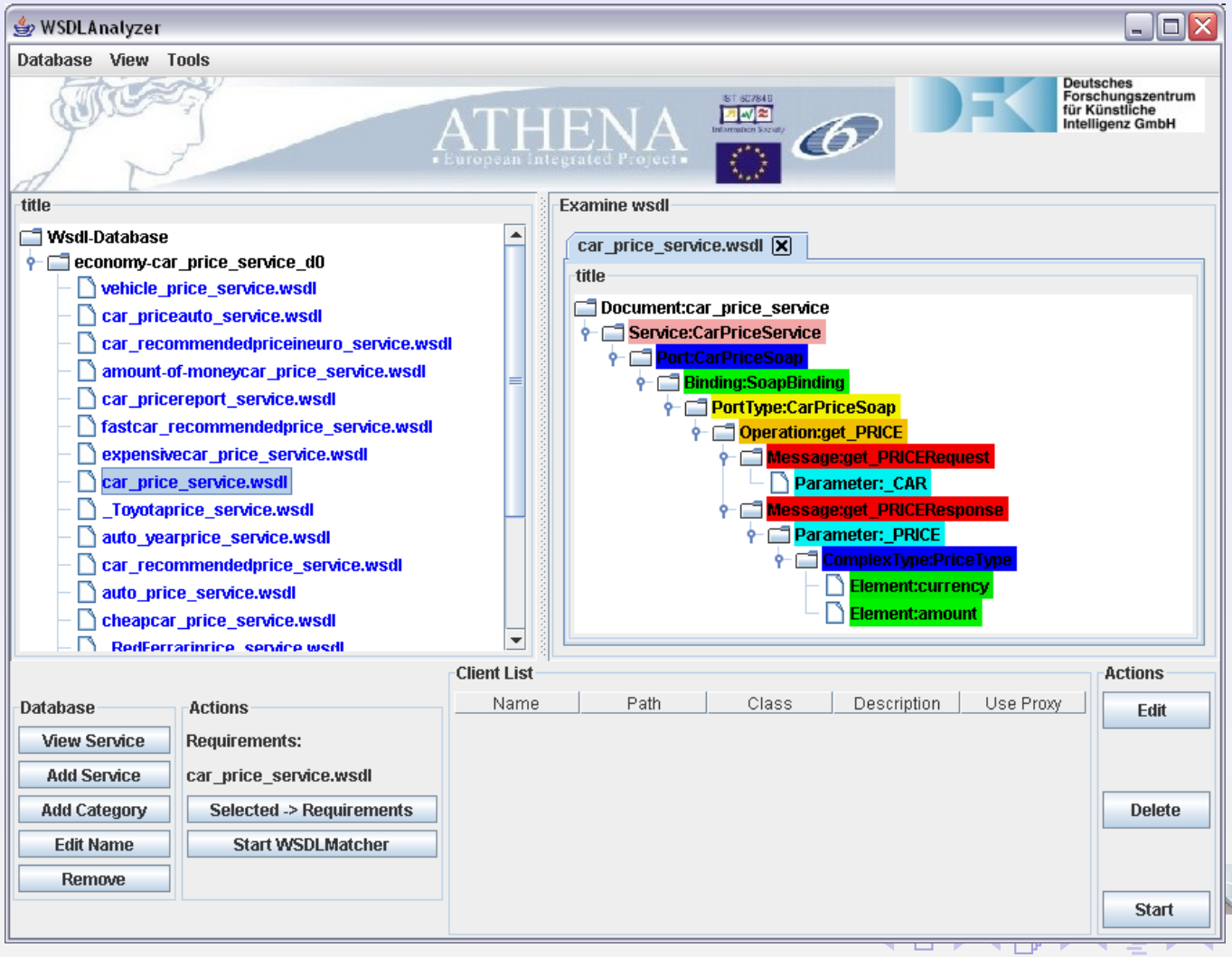
Durchführung der Evaluierung

1. Gegeben: OWLS-TC mit Matchmakingwerten des OWLS-MX
   - 29 Queries (Referenzdienste)
   - *Relevance Sets*
2. Translation OWL-S nach WSDL
   - OWL-S *Query* wird zu WSDL *Requirement*
   - Dienstbeschreibungen des *Relevance Set* werden *Candidates*
3. Ermittlung von Ähnlichkeitswerten mit dem WA (Ranking)
4. Vergleich der Matchmakingwerte (WA und OWLS-MX)

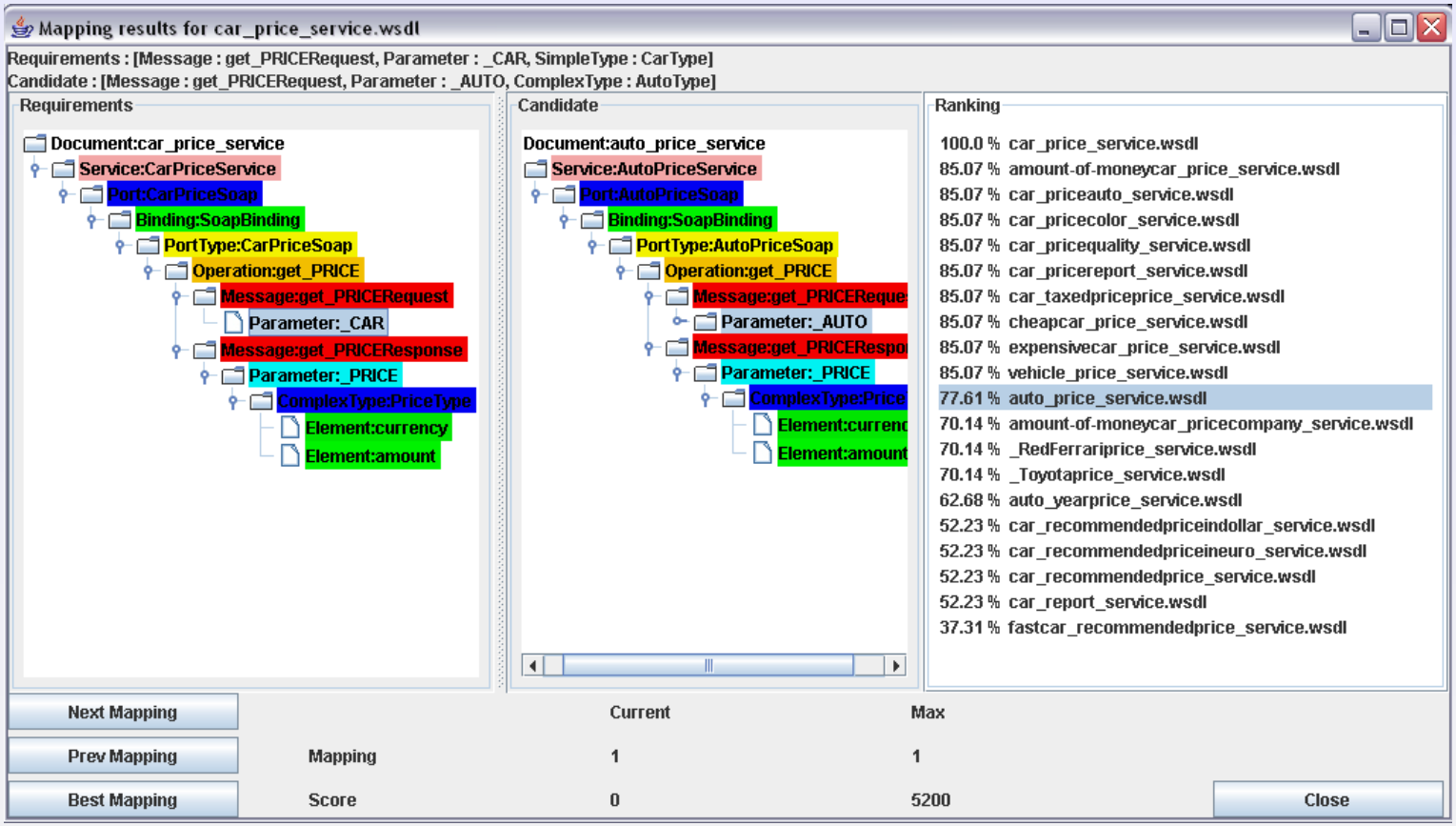
Durchführung der Evaluierung

1. Gegeben: OWLS-TC mit Matchmakingwerten des OWLS-MX
   - 29 Queries (Referenzdienste)
   - *Relevance Sets*
2. Translation OWL-S nach WSDL
   - OWL-S *Query* wird zu WSDL *Requirement*
   - Dienstbeschreibungen des *Relevance Set* werden *Candidates*
3. Ermittlung von Ähnlichkeitswerten mit dem WA (Ranking)
4. Vergleich der Matchmakingwerte (WA und OWLS-MX)

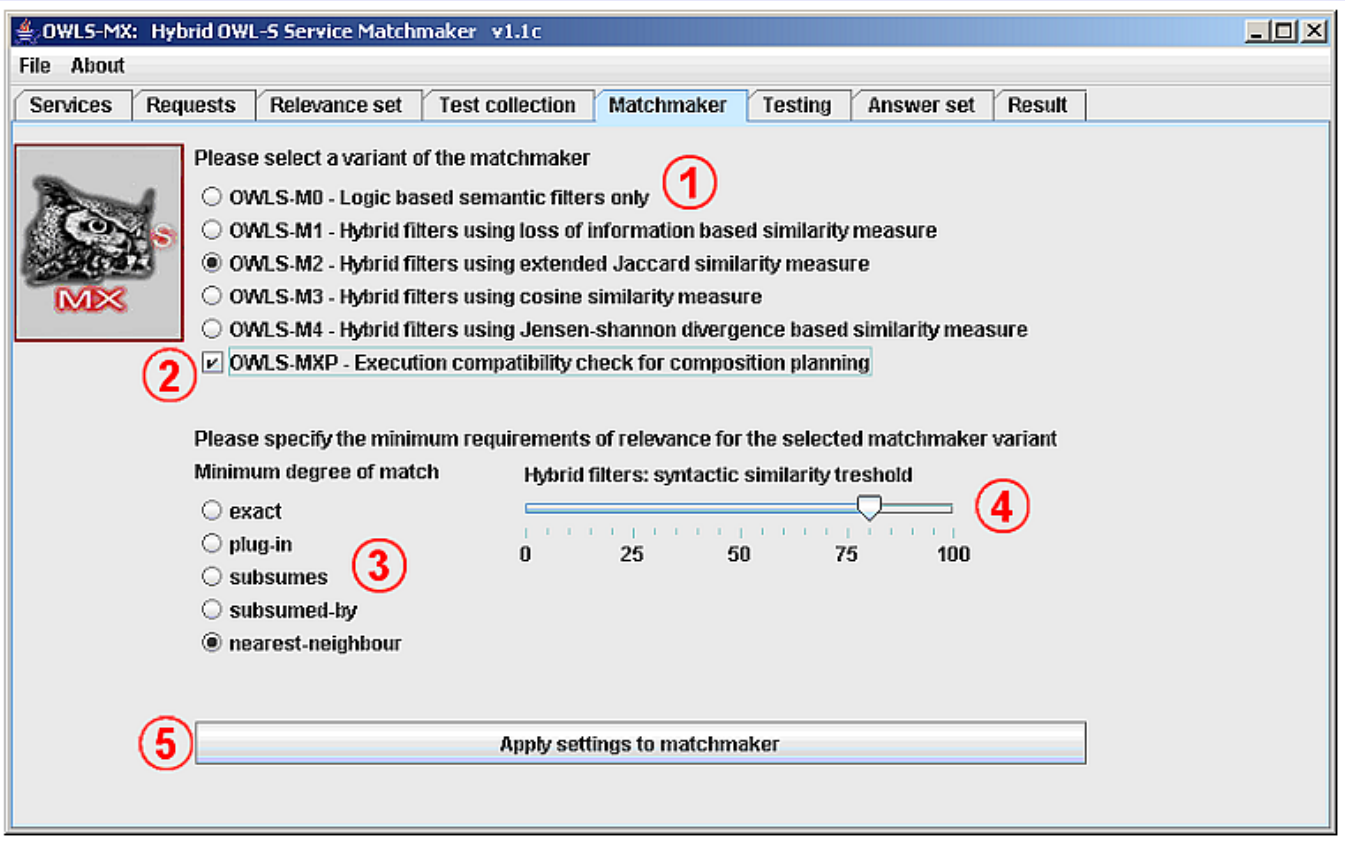
WSDL Analyzer, Service Verwaltung



WSDL Analyzer, Ranking

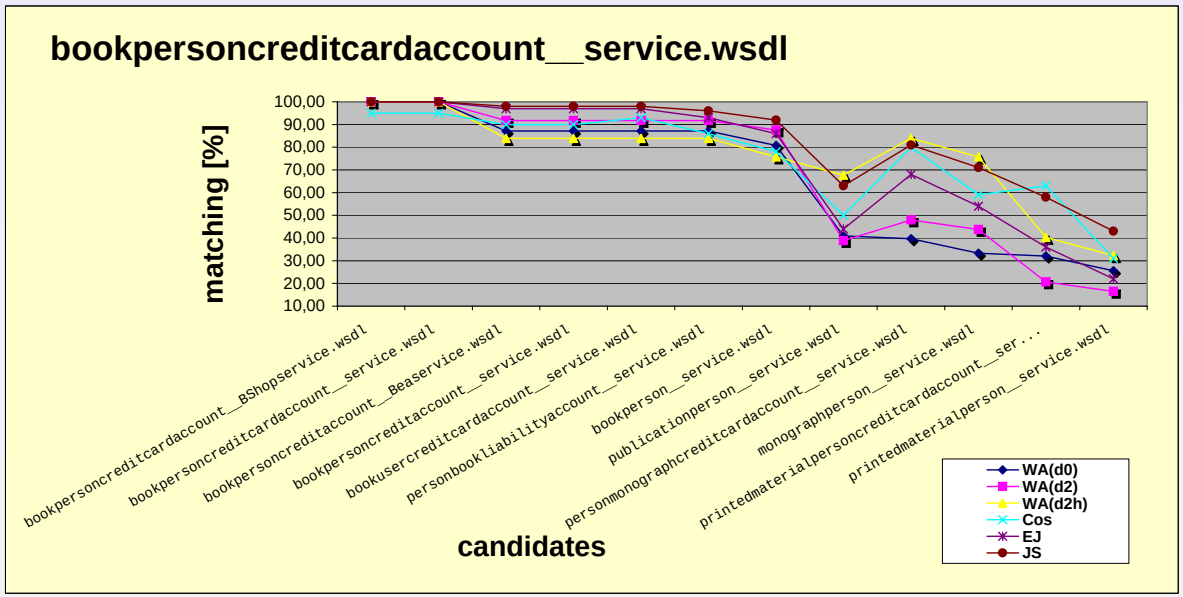


Der OWLS-MX, Konfiguration



Vergleich WA und OWLS-MX (Query 7)

Candidate service	WA(d0)	WA(d2)	WA(d2h)	Cos	EJ	JS	Semantic degree
bookpersoncreditcardaccount__BShopservice.wsdl	100,00	100,00	100,00	95,00	100,00	100,00	Exact
bookpersoncreditcardaccount__service.wsdl	100,00	100,00	100,00	95,00	100,00	100,00	Exact
bookpersoncreditaccount__Beaservice.wsdl	87,18	91,73	83,87	90,00	97,00	98,00	Plugin
bookpersoncreditaccount__service.wsdl	87,18	91,73	83,87	90,00	97,00	98,00	Plugin
bookusercreditcardaccount__service.wsdl	87,18	91,73	83,87	93,00	97,00	98,00	Failed
personbookliabilityaccount__service.wsdl	87,18	91,73	83,87	86,00	93,00	96,00	Plugin
bookperson__service.wsdl	80,78	87,60	75,80	78,00	86,00	92,00	Exact
publicationperson__service.wsdl	40,99	38,82	67,74	50,00	44,00	63,00	Plugin
personmonographcreditcardaccount__service.wsdl	39,71	47,91	83,87	80,00	68,00	81,00	Plugin
monographperson__service.wsdl	33,31	43,78	75,80	59,00	54,00	71,00	Plugin
printedmaterialpersoncreditcardaccount__service.wsdl	32,03	20,65	40,32	63,00	36,00	58,00	Plugin
printedmaterialperson__service.wsdl	25,62	16,52	32,25	31,00	22,00	43,00	Plugin



Diskussion der Ergebnisse

- Matchmaking-Ergebnisse durchaus vergleichbar
- Unterschiedlich gute Ergebnisse je nach Konfiguration
- Teilweise sehr große Schnittstellen
- Zyklen in XML Schema Definition

Arbeit mit dem WSDL Analyzer

- Nicht alle validierten WSDL Beschreibungen können verarbeitet werden. Zyklen machen Probleme.
- Verbesserung des WA aufgrund des Vorgehensmodells.

Diskussion der Ergebnisse

- Matchmaking-Ergebnisse durchaus vergleichbar
- Unterschiedlich gute Ergebnisse je nach Konfiguration
- Teilweise sehr große Schnittstellen
- Zyklen in XML Schema Definition

Arbeit mit dem WSDL Analyzer

- Nicht alle validierten WSDL Beschreibungen können verarbeitet werden. Zyklen machen Probleme.
- Verbesserung des WA aufgrund des Vorgehensmodells.

Diskussion der Ergebnisse

- Matchmaking-Ergebnisse durchaus vergleichbar
- Unterschiedlich gute Ergebnisse je nach Konfiguration
- Teilweise sehr große Schnittstellen
- Zyklen in XML Schema Definition

Arbeit mit dem WSDL Analyzer

- Nicht alle validierten WSDL Beschreibungen können verarbeitet werden. Zyklen machen Probleme.
- Verbesserung des WA aufgrund des Vorgehensmodells.

Diskussion der Ergebnisse

- Matchmaking-Ergebnisse durchaus vergleichbar
- Unterschiedlich gute Ergebnisse je nach Konfiguration
- Teilweise sehr große Schnittstellen
- Zyklen in XML Schema Definition

Arbeit mit dem WSDL Analyzer

- Nicht alle validierten WSDL Beschreibungen können verarbeitet werden. Zyklen machen Probleme.
- Verbesserung des WA aufgrund des Vorgehensmodells.

ENDE

Vielen Dank für Ihre Aufmerksamkeit.