Service-Oriented Architectures

A Web service is a component deployed on a Web accessible platform provided by a service provider to be discovered and invoked over the Web by a service requestor.

Not enough to allow dynamic discovery and binding!
Example: Car Rental Service

```plaintext
<<interface>>
RentalServiceRequired

reserveCar(c: Customer, car: Car, ri: RentalInfo)
```

```plaintext
<<interface>>
RentalServiceProvided

makeReserv(c: Customer, car: Car, ri: RentalInfo): EContract...
```

Matching **provider** and **requestor** specification within registry must ensure compatibility of:

**Data types**
- Does Customer have the same meaning for requestor and provider?

**Operation signatures**
- Can provider operation be supplied with suitable parameters from a call of requestor operation?

**Behavior**
- Does provided operation actually carry out what is expected by a requestor?

---

Data Types and Signatures

```plaintext
<<interface>>
RentalServiceRequired

reserveCar(c: Customer, car: Car, ri: RentalInfo)
```

```plaintext
<<interface>>
RentalServiceProvided

makeReserv(c: Customer, car: Car, ri: RentalInfo): EContract...
```

**Data types:** parties use common domain model (ontology)

**Operation signatures:**

- Reorder and rename pars
- Skip input of requestor
- Ignore output of provider
Behavior: Operation Contracts

**Pre-condition:**
Customer provides rental info and chooses car

**Effect:**
Car is reserved for customer

Required
- Formal specification (logic, graph transformation, …) for automatic matching
- Integration into mainstream SW development methods (UML) for wider applicability

Outline
- Contracts as graph transformation rules
- Semantics of rules
- Semantic / syntactic compatibility, soundness

Contracts as Graph Transformation Rules

**Signature:**
`reservCar(c:Customer, my_car:Car, ri:RentInfo)`

**Behavior:**
GT rule

**Pre-condition:**
- `c:Customer` provides `my_car:Car`
- `ri:RentInfo`

**Effect:**
- `c:Customer` reserves `my_car:Car`
- `ri:RentInfo`

**Typed DPO**
[Corradini et al 96]

**Data types:**
type graph
What is the right notion of compatibility? That depends on…

how services should interact:

1. Requestor guarantees \( \text{pre}_R \)
   \( \Rightarrow \) Provider assumes \( \text{pre}_P \)
2. Provider guarantees \( \text{effect}_P \)
   \( \Rightarrow \) Requestor assumes \( \text{effect}_R \)

… a contravariant relation.

what it should mean, that:

- an assumption is correct
- a guarantee is fulfilled

… a question about the semantics of contracts.

Operational Semantics: The DPO Approach

- \( L \) is embedded into graph \( G \).
- The elements of \( G \) matched by \( L - l(K) \) are removed.
- The elements matched by \( R - r(K) \) are added to \( D \).
Loose Semantics of Contracts

Requestor has only loose idea of behavior of the other service

\[ \text{Requestor} \]

\[ \text{pre}_R \rightarrow \text{effect}_R \]

1. call

\[ \text{pre}_P \rightarrow \text{effect}_P \]

2. return

Provider has complete info, but may prefer not to publish everything

\[ \text{Provider} \]

\[ \rightarrow \text{Contracts are incomplete specifications of service behavior} \]

\[ \text{G} \rightarrow \text{D} \]

Formally: Double-Pullback (DPB), allows unspecified

Deletion: at least elements of G matched by L - I(K) are removed

Creation: at least elements matched by R - r(K) are added to D

(faithful) transition vs. transformation

Contracts as Rules, revisited

\[ \rightarrow \text{Positive Application Conditions} \]

**Precondition:** what must be present before, no matter what happens later

- deleted
- preserved
- created

**Effect:** what must be

- provides
- reserves

unnamed

c1:Customer

name="upb"

r1:RentalInfo

pick-upDate=21.02.04
returnDate=25.02.04
location=Pisa

c1:Car

id="VWMultiVan01"

G

D

H
What is the right notion of compatibility?
That depends on …

how services should interact:
1. Requestor guarantees \( \text{pre}_R \)
   \( \Rightarrow \) Provider assumes \( \text{pre}_P \)
2. Provider guarantees \( \text{effect}_P \)
   \( \Rightarrow \) Requestor assumes \( \text{effect}_R \)

… a contravariant relation.

what it should mean, that:
- an assumption is correct
- a guarantee is fulfilled

… a question about the semantics of contracts.

Semantic Compatibility

1. \( \text{pre}_R \Rightarrow \text{pre}_P \) : applicability of requestor rule \textbf{implies} applicability of provider rule

2. \( \text{effect}_P \Rightarrow \text{effect}_R \) : transition via provider rule \textbf{implies} transition via requestor rule.
Semantic Compatibility

\begin{align*}
\text{R:} & \quad c_{\text{Customer}} \rightarrow r_{\text{RentInfo}} \\
& \quad \downarrow \quad \downarrow \\
& \quad \text{my_carCar} \\
& \quad \text{L}_r \\
\text{P:} & \quad c_{\text{Customer}} \rightarrow r_{\text{RentInfo}} \\
& \quad \uparrow \quad \uparrow \\
& \quad \text{carCar} \\
& \quad \text{L}_p \\
\end{align*}

Semantic Compatibility: formally

\begin{align*}
\hat{L}_1 \rightarrow d_{L_1} \rightarrow L_1 \\
\hat{L}_2 \rightarrow d_{L_2} \rightarrow L_2 \\
G \rightarrow d_G \rightarrow \hat{G} \\
\end{align*}

\begin{align*}
\text{R:} & \quad c_{\text{Customer}} \rightarrow r_{\text{RentInfo}} \\
& \quad \downarrow \quad \downarrow \\
& \quad \text{my_carCar} \\
& \quad \text{L}_r \\
\text{P:} & \quad c_{\text{Customer}} \rightarrow r_{\text{RentInfo}} \\
& \quad \uparrow \quad \uparrow \\
& \quad \text{carCar} \\
& \quad \text{L}_p \\
\end{align*}

\begin{itemize}
\item[(i)] For all graphs $G$, if there exists $d_{L_1} : \hat{L}_1 \rightarrow G$ s.t. $d_{L_1} \circ \hat{L}_1$ satisfies $IC$ of $p_1$, then there exists $d_{L_2} : \hat{L}_2 \rightarrow G$ s.t. $d_{L_2} \circ \hat{L}_2$ satisfies $IC$ of $p_2$, and
\item[(ii)] For all spans $t : \langle G, d_G, L_1, \hat{G}, L_2, \hat{H} \rangle$, if there exists a transition $G \xrightarrow{p_1/d_1} H$, then there exists a transition $G \xrightarrow{p_1/d_1} H$ using the same bottom span $t$.
\end{itemize}
What do we have?

Semantic compatibility relation $\models$ over rules
- quantified over sets of all graphs and transitions
- cannot be verified directly

**Goal:** syntactic matching relation $\vdash$ over rules such that
- Soundness: $p_2 \vdash p_1$ implies $p_2 \models p_1$
- Completeness: $p_2 \models p_1$ implies $p_2 \vdash p_1$

---

**Syntactic Matching Relation**

$pre_R \rightarrow pre_P$: requestor must provide all information necessary for the execution of the provider operation,

$effect_R \rightarrow effect_P$: effect of the provided operation must include those expected by the requestor.
Syntactic Matching: formally

\[(p_1, \hat{L}_1)\text{ syntactically matches } (p_2, \hat{L}_2), \text{ in symbols } (p_2 : s_2, \hat{L}_2) \vdash_{\text{match}} (p_1 : s_1, \hat{L}_1), \text{ iff:}\]

(i) there exists an injective graph homomorphism \(h_L : \hat{L}_2 \rightarrow \hat{L}_1\) s.t. \(h_L \circ \hat{L}_2\) satisfies IC’ of \(p_2\), and

(ii) there exist graph homomorphisms \(h_L : L_1 \rightarrow L_2\), \(h_K : K_1 \rightarrow K_2\), and \(h_R : R_1 \rightarrow R_2\) s.t. the diagrams (a), (b), and the diagram on the left commute, and the diagrams (a) and (b) represent a faithful transition.

What do we have?

Semantic compatibility relation \(\models\) over rules

- quantified over sets of all graphs and transitions
- cannot be verified directly

Goal: syntactic matching relation \(\vdash\) over rules such that

- Soundness: \(p_2 \vdash p_1\) implies \(p_2 \models p_1\)
- Completeness: \(p_2 \models p_1\) implies \(p_2 \vdash p_1\)
Summary & Future Work

- Formal approach to service specification matching.
- Operation contracts are GT-rules with loose semantics.
- Semantic and syntactic matching relations.
- Soundness of matching.

- Refinement of semantic compatibility (→ completeness of syntactic matching).
- Extension to typed graphs with attributes and subtyping.
- Logic / XML-representation of contracts: RDF in DAML-S
- Tool support for computing syntactic matching based on RDF graph matching with RDQL

Proof of Soundness

To prove: \( \mathcal{p}_2 \vDash_{\text{match}} \mathcal{p}_1 \) implies \( \mathcal{p}_2 \nvdash_{\text{match}} \mathcal{p}_1 \)

(i) for all graphs \( G \), if there exists \( d_{\mathcal{L}_1} : \mathcal{L}_1 \rightarrow G \) s.t. \( d_{\mathcal{L}_1} := d_{\mathcal{L}_2} \circ h \) satisfies IC of \( \mathcal{p}_1 \), then there exists \( d_{\mathcal{L}_2} : \mathcal{L}_2 \rightarrow G \) s.t. \( d_{\mathcal{L}_2} := d_{\mathcal{L}_3} \circ h \) satisfies IC of \( \mathcal{p}_2 \).

\( d_{\mathcal{L}_2} \circ h \) is diagram (3) commutes.

\( d_{\mathcal{L}_3} = d_{\mathcal{L}_2} \circ h \) satisfies IC of \( \mathcal{p}_2 \) because of this commutativity and the fact that \( h \circ d_{\mathcal{L}_2} \) satisfies IC of \( \mathcal{p}_2 \).
(ii) for all spans \( t : (G \xrightarrow{a} D \xrightarrow{b} H) \), if there exists a transition \( G \xrightarrow{p_2/d_2} H \),
then there exists a transition \( G \xrightarrow{p_1/d_1} H \) using the same bottom span \( t \).

Both transitions can be vertically composed using the composition of the underlying pullback squares.

Faithfulness of the composed transition follows from the fact that IC of \( d_{L_1} \) follows from that of \( h_L \) and \( d_{L_2} \) (analogously for the right-hand side).