Conservative re-use ensuring matches for service selection

M. Baldoni, C. Baroglio, V. Patti, and C. Schifanella

Dipartimento di Informatica — Università degli Studi di Torino
C.so Svizzera, 185 — I-10149 Torino (Italy)
{baldoni,baroglio,patti,schi}@di.unito.it

November 17, 2008
(Web) Services re-use

- Automatic retrieval, selection, composition of (Web) Services
- The need for a semantic layer, e.g. OWL-S [7] and WSMO [4]
- A richer annotation, aimed at representing the so-called IOPEs (inputs, outputs, preconditions and effects of the service)
- Design by contract [6]: preconditions and postconditions

M. Baldoni, C. Baroglio, V. Patti, and C. ScConservative re-use ensuring matches for serv

November 17, 2008 2 / 25
Semantic annotation allows the discovery of services, whose descriptions do not exactly match with the corresponding queries.

E.g., PIM (Plugin Match):

\[ Q_{\text{pre}} \Rightarrow S_{\text{pre}} \land S_{\text{post}} \Rightarrow Q_{\text{post}} \]
Semantic matchmaking focuses on the discovery of single services, a *single operation*.

In general, however, the use of a web service implies the execution of a *complex sequence of operations* in a particular *order*, which might even involve other services.

WS-CDL [10] is aimed at specifying (complex) patterns of interaction.
(Web) Services re-use: semantic matchmaking

- Semantic matchmaking focuses on the discovery of single services, a single operation.

- In general, however, the use of a web service implies the execution of a complex sequence of operations in a particular order, which might even involve other services.

- WS-CDL [10] is aimed at specifying (complex) patterns of interaction.
Selecting existing services that have to play the roles of a given choreography. This task implies verifying two things:

1. the *conformance* of the service to the specification of a role of interest
2. the use of that service allows the achievement of the *goal*, that caused its search
(Web) Services re-use: unbound roles

- The achievement of the goal depends on the operation sequence because each operation can influence the executability and the outcomes of the subsequent ones.
- Many of the operations, however, are offered by the partners in the interaction which, at the time when the service reasons about the choreography, they are still unknown.
- Unbound operations (selection process will link them to actual players)
The reasoning process must, therefore, use the specifications of the operations that will be supplied by the partners, specifications which are included in the choreography.

A selection process will link unbound operations with operations offered by existing services, and it does so by applying some kind of (possibly flexible) match.
In [2], however, we showed that performing a match operation by operation, by applying the definitions in [11], does not preserve the global goal. Therefore, the matchmaking process, that is applied to discover services, should not only focus on local properties of the single operations, e.g. IOPEs, but it should also consider constraints that derive from the global schema of execution, which is given by the choreography.
In this paper

- We exploit an *action-based representation* of the specifications of the operations of a service: each operation is described in terms of its preconditions and effects, input and output.
- We show how to enrich the class of *re-use ensuring matches* [3] proposed in the literature so to produce substitutions that preserve goals.

**Definition (Re-use ensuring match [3])**

A specification match $M$ is re-use ensuring iff for any $S$ and $Q$,

\[ M(I, Q) \land \{ S_{\text{pre}} \} S \{ S_{\text{post}} \} \Rightarrow \{ Q_{\text{pre}} \} S \{ Q_{\text{post}} \}. \]
In this paper

- We exploit an \textit{action-based representation} of the specifications of the operations of a service: each operation is described in terms of its preconditions and effects, input and output.
- We show how to enrich the class of \textit{re-use ensuring matches} [3] proposed in the literature so to produce substitutions that preserve goals.

**Definition (Re-use ensuring match [3])**

A specification match $M$ is re-use ensuring iff for any $S$ and $Q$, $M(I, Q) \land \{S_{pre}\} S \{S_{post}\} \Rightarrow \{Q_{pre}\} S \{Q_{post}\}$. 

M. Baldoni, C. Baroglio, V. Patti, and C. Sc

Conservative re-use ensuring matches for serv

November 17, 2008
Reasoning about services

- The notation is based on a logical theory for reasoning about actions and change in a modal logic programming setting.
- The problem of reasoning amounts either to build or to traverse a sequence of transitions between states.
- A state is a set of fluents, i.e., properties whose truth value can change over time, due to the application of actions.
- There are four basic kinds of operations [1] (or atomic processes, when using OWL-S terminology [7]):
  - one-way
  - notify
  - request-response
  - solicit-response
One-Way Operation

(a) operation\(\triangleright ow(m_{in})\) possible if \(B^{Invoker}m_{in} \land P_s\)
(b) operation\(\triangleright ow(m_{in})\) causes \(B^{Invoker}sent(m_{in})\)
(c) operation\(\triangleright ow(m_{in})\) causes \(E_s\)

One-Way, supplier point of view:
(a) operation\(\triangleleft ow(m_{in})\) possible if \(R_s\)
(b) operation\(\triangleleft ow(m_{in})\) causes \(B^{Offerer}m_{in}\)
(c) operation\(\triangleleft ow(m_{in})\) causes \(S_s\)
Notify Operation

Notify, invoker point of view:
(a) \( \text{operation}_n^{\rightarrow}(m_{out}) \) possible if \( P_s \)
(b) \( \text{operation}_n^{\rightarrow}(m_{out}) \) causes \( B^{\text{Invoker}} m_{out} \)
(c) \( \text{operation}_n^{\rightarrow}(m_{out}) \) causes \( E_s \)

Notify, supplier point of view:
(a) \( \text{operation}_n^{\leftarrow}(m_{out}) \) possible if \( B^{\text{Offerer}} m_{out} \land R_s \)
(b) \( \text{operation}_n^{\leftarrow}(m_{out}) \) causes \( B^{\text{Offerer}} \text{sent}(m_{out}) \)
(c) \( \text{operation}_n^{\leftarrow}(m_{out}) \) causes \( S_s \)
Request-response Operation

Request-response, invoker point of view:
(a) \( \text{operation}_{rr} (m_{in}, m_{out}) \) possible if \( B_{Invoker} m_{in} \land P_s \)
(b) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( B_{Invoker} \text{sent}(m_{in}) \)
(c) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( B_{Invoker} m_{out} \)
(d) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( E_s \)

Request-response, supplier point of view:
(a) \( \text{operation}_{rr} (m_{in}, m_{out}) \) possible if \( R_s \)
(b) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( B_{Offerer} m_{in} \)
(c) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( B_{Offerer} m_{out} \)
(d) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( B_{Offerer} \text{sent}(m_{out}) \)
(e) \( \text{operation}_{rr} (m_{in}, m_{out}) \) causes \( S_s \)
**Solicit-response Operation**

![Diagram of Solicit-response Operation]

**Solicit-response**, invoker point of view:

(a) operation $\Rightarrow_{sr} (m_{in}, m_{out})$ possible if $P_s$
(b) operation $\Rightarrow_{sr} (m_{in}, m_{out})$ causes $B_{invoker} m_{out}$
(c) operation $\Rightarrow_{sr} (m_{in}, m_{out})$ causes $B_{invoker} m_{in}$
(d) operation $\Rightarrow_{sr} (m_{in}, m_{out})$ causes $B_{invoker} sent(m_{in})$
(e) operation $\Rightarrow_{sr} (m_{in}, m_{out})$ causes $E_s$

**Solicit-response**, supplier point of view:

(a) operation $\Leftarrow_{sr} (m_{in}, m_{out})$ possible if $B_{Offerer} m_{out} \land R_s$
(b) operation $\Leftarrow_{sr} (m_{in}, m_{out})$ causes $B_{Offerer} sent(m_{out})$
(c) operation $\Leftarrow_{sr} (m_{in}, m_{out})$ causes $B_{Offerer} m_{in}$
(d) operation $\Leftarrow_{sr} (m_{in}, m_{out})$ causes $S_s$
Choreographies and roles

- $\mathcal{P}$ encodes the complex behavior of the service; it is a collection of clauses of the kind:

  $$p_0 \text{ is } p_1, \ldots, p_n$$

- $p_i, i = 1, \ldots, n$, is either an atomic action (operation), a test action (denoted by the symbol ?), or a procedure call.

- Procedures can be recursive and are executed in a goal-directed way, similarly to standard logic programs, and their definitions can be non-deterministic as in Prolog.

- A choreography is made of a set of interacting roles, a role being a subjective view of the interaction that is encoded.
Reasoning on goals

- We need a mechanism that verifies if a goal condition holds after the interaction with the service has taken place.
- $Fs$ is the set of facts that we would like to hold “after” $p$.
- This form of reasoning is known as \textit{temporal projection}.
- Queries of the form:

  \[ Fs \text{ after } p \]

- The execution of the above query returns as a side-effect an \textit{execution trace} $\sigma$ of $p$. 
Choreographies and roles

- When a service plays a role in a choreography, its policy will contain some operations which are not of the service itself but belong to some other role of the choreography, with which it interacts.

- Basic operations: a set of *bound operations* and a set of *unbound operations*, that must be supplied by some counterpart(s).

- Until the counterpart service(s) is (are) not defined, such operations will be those specified in the choreography.

- Such operations will be offered by the interlocutors as ≫ operations.
Conservative matching

Substitution

Let \( S_d = \langle S, P \rangle \) be a service description, and let \( S_u \) be a subset of \( S \), containing unbound operations that are to be supplied by a same counterpart \( S_i \). Let \( S_{S_i} \) be the set of operations in \( S_i \) that we want \( S_d \) to use, binding them to \( S_u \). We represent the binding by the substitution \( \theta = [S_{S_i}/S_u] \) applied to \( S_d \), i.e.: \( S_d\theta = \langle S\theta, P\theta \rangle \), where every element of \( S_u \) is substituted by/bound to an element of \( S_{S_i} \).

Definition (Conservative substitution)

Let us consider a service \( S_i = \langle S, P \rangle \) playing a role \( R_i \) in a given choreography, and a query \( G \) such that, given an initial state \( S_0 \), \((\langle S, P \rangle, S_0) \vdash G \) w.a. \( \sigma \). Consider a substitution \( \theta = [S_{S_j}/S_{u(R_j)}^\sigma] \), where \( S_{u(R_j)}^\sigma = \{ o_u \in S \mid o \text{ occurs in } \sigma \} \) is the set of all unbound operations that refer to another role \( R_j, j \neq i \), of the same choreography, that are used in the execution trace \( \sigma \). \( \theta \) is conservative when the following holds: \((\langle S\theta, P\theta \rangle, S_0) \vdash G \) w.a. \( \sigma\theta \).
Conservative matching

Substitution

Let $S_d = \langle S, P \rangle$ be a service description, and let $S_u$ be a subset of $S$, containing unbound operations that are to be supplied by a same counterpart $S_i$. Let $S_{S_i}$ be the set of operations in $S_i$ that we want $S_d$ to use, binding them to $S_u$. We represent the binding by the substitution $\theta = [S_{S_i}/S_u]$ applied to $S_d$, i.e.:

$$S_d\theta = \langle S\theta, P\theta \rangle,$$

where every element of $S_u$ is substituted by/bound to an element of $S_{S_i}$.

Definition (Conservative substitution)

Let us consider a service $S_i = \langle S, P \rangle$ playing a role $R_i$ in a given choreography, and a query $G$ such that, given an initial state $S_0$, $(\langle S, P \rangle, S_0) \vdash G$ w.a. $\sigma$. Consider a substitution $\theta = [S_{S_j}/S^\sigma_{u(R_j)}]$, where $S^\sigma_{u(R_j)} = \{ o_u \in S \mid o \text{ occurs in } \sigma \}$ is the set of all unbound operations that refer to another role $R_j$, $j \neq i$, of the same choreography, that are used in the execution trace $\sigma$. $\theta$ is conservative when the following holds:

$(\langle S\theta, P\theta \rangle, S_0) \vdash G$ w.a. $\sigma\theta$
**Conservative matching**

**Definition (Conservative substitution)**

Let us consider a service $S_i = \langle S, P \rangle$ playing a role $R_i$ in a given choreography, and a query $G$ such that, given an initial state $S_0$, $(\langle S, P \rangle, S_0) \vdash G$ w.a. $\sigma$

Consider a substitution $\theta = [S_j / S_{\sigma u(R_j)}]$, where $S_{\sigma u(R_j)} = \{ o_u \in S \mid o \text{ occurs in } \sigma \}$ is the set of all unbound operations that refer to another role $R_j$, $j \neq i$, of the same choreography, that are used in the execution trace $\sigma$. $\theta$ is conservative when the following holds: $(\langle S\theta, P\theta \rangle, S_0) \vdash G$ w.a. $\sigma\theta$

In [2] we have proved that, in general, flexible matches are not conservative substitution.

---

**S has proved that $R \vdash G$ w.a. $\sigma$**

$R$ contains **unbound operations** that will be offered by its interlocutor

**Candidate partner**

**Operations**

**Match**

**Substitution $\theta$**

**Question:**

$R \theta \vdash G$ w.a. $\sigma \theta$ ??
Conservative matching

**Definition (Conservative substitution)**

Let us consider a service \( S_i = \langle S, P \rangle \) playing a role \( R_i \) in a given choreography, and a query \( G \) such that, given an initial state \( S_0 \), \((\langle S, P \rangle, S_0) \vdash G \) w.a. \( \sigma \) Consider a substitution \( \theta = \left[ S_{S_j} / S_{u(R_j)} \right] \), where \( S_{u(R_j)} = \{ o_u \in S \mid o \text{ occurs in } \sigma \} \) is the set of all unbound operations that refer to another role \( R_j, j \neq i \), of the same choreography, that are used in the execution trace \( \sigma \). \( \theta \) is conservative when the following holds: \((\langle S\theta, P\theta \rangle, S_0) \vdash G \) w.a. \( \sigma\theta \)

In [2] we have proved that, in general, flexible matches are not conservative substitution.
In the literature it is possible to find many match algorithms, mostly based on the seminal work by Zaremski and Wing [11] on software components, and surveyed in

A lattice of matches

Given a software component $I$, with precondition $S_{pre}$ and postcondition $S_{post}$, and a specification (or query, in the match-making community) $Q$, with precondition $Q_{pre}$ and postcondition $Q_{post}$, the most important kinds of relaxed match between $Q$ and $S$ are:

- **EM (Exact Pre/Post Match):**
  \[ Q_{pre} \iff S_{pre} \land Q_{post} \iff S_{post} \]

- **EPREM (Exact Pre Match):**
  \[ Q_{pre} \iff S_{pre} \land S_{post} \implies Q_{post} \]

- **EPOM (Exact Post Match):**
  \[ Q_{pre} \implies S_{pre} \land Q_{post} \iff S_{post} \]

- **PIM (Plugin Match):**
  \[ Q_{pre} \implies S_{pre} \land S_{post} \implies Q_{post} \]

- **POM (Plugin Post Match):**
  \[ S_{post} \implies Q_{post} \]

- **GPIM (Guarded Plugin Match, a.k.a. Weak-Plugin [8]):**
  \[ Q_{pre} \implies S_{pre} \land \left( (S_{pre} \land S_{post}) \implies Q_{post} \right) \]

- **SM (Satisfies Match, a.k.a. relaxed plug-in in [3], plug-in compatibility [5]):**
  \[ Q_{pre} \implies S_{pre} \land \left( Q_{pre} \land S_{post} \implies Q_{post} \right) \]

- **GPOM (Guarded Post Match, a.k.a. Weak-Post [8]):**
  \[ \left( S_{pre} \land S_{post} \right) \implies Q_{post} \]

- **GGP (Guarded-Generalized Predicate):**
  \[ \left( Q_{pre} \implies S_{pre} \right) \land \left( (S_{pre} \implies S_{post}) \implies \left( Q_{pre} \implies Q_{post} \right) \right) \]
A lattice of matches

Given a software component $S$, with precondition $S_{\text{pre}}$ and postcondition $S_{\text{post}}$, and a specification (or query, in the match-making community) $Q$, with precondition $Q_{\text{pre}}$ and postcondition $Q_{\text{post}}$, the most important kinds of relaxed match between $Q$ and $S$ are:

- **EM (Exact Pre/Post Match):**
  \[ Q_{\text{pre}} \iff S_{\text{pre}} \land Q_{\text{post}} \iff S_{\text{post}} \]

- **EPREM (Exact Pre Match):**
  \[ Q_{\text{pre}} \iff S_{\text{pre}} \land S_{\text{post}} \Rightarrow Q_{\text{post}} \]

- **EPOM (Exact Post Match):**
  \[ Q_{\text{pre}} \Rightarrow S_{\text{pre}} \land Q_{\text{post}} \iff S_{\text{post}} \]

- **PIM (Plugin Match):**
  \[ Q_{\text{pre}} \Rightarrow S_{\text{pre}} \land S_{\text{post}} \Rightarrow Q_{\text{post}} \]

- **POM (Plugin Post Match):**
  \[ S_{\text{post}} \Rightarrow Q_{\text{post}} \]

- **GPIM (Guarded Plugin Match, a.k.a. Weak-Plugin [8]):**
  \[ Q_{\text{pre}} \Rightarrow S_{\text{pre}} \land ((S_{\text{pre}} \land S_{\text{post}}) \Rightarrow Q_{\text{post}}) \]

- **SM (Satisfies Match, a.k.a. relaxed plug-in in [3], plug-in compatibility [5]):**
  \[ Q_{\text{pre}} \Rightarrow S_{\text{pre}} \land (Q_{\text{pre}} \land S_{\text{post}} \Rightarrow Q_{\text{post}}) \]

- **GPOM (Guarded Post Match, a.k.a. Weak-Post [8]):**
  \[ ((S_{\text{pre}} \land S_{\text{post}}) \Rightarrow Q_{\text{post}}) \]

- **GGP (Guarded-Generalized Predicate):**
  \[ (Q_{\text{pre}} \Rightarrow S_{\text{pre}}) \land ((S_{\text{pre}} \Rightarrow S_{\text{post}}) \Rightarrow (Q_{\text{pre}} \Rightarrow Q_{\text{post}})) \]

---

**Definition (Re-use ensuring match [3])**

A specification match $M$ is re-use ensuring iff for any $S$ and $Q$,

\[ M(S, Q) \land \{ S_{\text{pre}} \} S \{ S_{\text{post}} \} \Rightarrow \{ Q_{\text{pre}} \} S \{ Q_{\text{post}} \}. \]
Verrifying conservative matches

- Intuitively, we take into account the dependencies between operations, which produce as effects fluents, that are used as preconditions by subsequent operations.

- The idea is to verify that the “causal chains” which allow the execution of the sequence of operations, are not broken after the substitution.

- The obvious hypothesis is that we have a choreography and that we know that it allows to achieve the goal of interest, i.e. that there is an execution $\sigma$, which allows the achievement of the goal.

- We will use this trace for defining a set of constraints that, whenever satisfied by a substitution obtained by a re-use ensuring match, guarantee that the substitution is also conservative.

- This is a “sufficient” condition because there might exist conservative substitutions that do not satisfy this set of constraints.
Verrifying conservative matches

Definition
A substitution $\theta$ is called uninfluential iff for any substitution $[s/o_u]$ in $\theta$, all beliefs in $E_s(s) - E_s(o_u)$ are uninfluential fluents w.r.t. $\sigma$.

Proposition
Let $M$ be a re-use ensuring match, any substitution $\theta_M$ that is uninfluential is also conservative.

Theorem
Let $M$ be a re-use ensuring match, $S_i = \langle S, \mathcal{P} \rangle$ be a service which plays a role $R_i$ in a given choreography, and $G$ a query such that, $(\langle S, \mathcal{P} \rangle, S_0) \vdash G$ w.a. $\sigma$, where $S_0$ is the initial state. Let $\theta_M$ be a substitution for all unbound operations of $S_i$ that refer to another role $R_j$ played by the service $S_j$, $j \neq i$. The problem of determining whether $\theta_M$ is conservative w.r.t. $G$ is decidable.
Verrifying conservative matches

Definition

A substitution $\theta$ is called *uninfluenzial* iff for any substitution $[s/o_u]$ in $\theta$, all beliefs in $E_s(s) - E_s(o_u)$ are uninfluential fluents w.r.t. $\sigma$.

Proposition

Let $M$ be a re-use ensuring match, any substitution $\theta_M$ that is uninfluenzial is also conservative.

Theorem

Let $M$ be a re-use ensuring match, $S_i = \langle S, \mathcal{P} \rangle$ be a service which plays a role $R_i$ in a given choreography, and $G$ a query such that, $(\langle S, \mathcal{P} \rangle, S_0) \vdash G$ w.a. $\sigma$, where $S_0$ is the initial state. Let $\theta_M$ be a substitution for all unbound operations of $S_i$ that refer to another role $R_j$ played by the service $S_j$, $j \neq i$. The problem of determining whether $\theta_M$ is conservative w.r.t. $G$ is decidable.
Verrifying conservative matches

**Definition**
A substitution $\theta$ is called *uninfluential* iff for any substitution $[s/o_u]$ in $\theta$, all beliefs in $E_s(s) - E_s(o_u)$ are uninfluential fluents w.r.t. $\sigma$.

**Proposition**
Let $M$ be a re-use ensuring match, any substitution $\theta_M$ that is uninfluential is also conservative.

**Theorem**
Let $M$ be a re-use ensuring match, $S_i = \langle S, \mathcal{P} \rangle$ be a service which plays a role $R_i$ in a given choreography, and $G$ a query such that, $(\langle S, \mathcal{P} \rangle, S_0) \vdash G$ w.a. $\sigma$, where $S_0$ is the initial state. Let $\theta_M$ be a substitution for all unbound operations of $S_i$ that refer to another role $R_j$ played by the service $S_j$, $j \neq i$. The problem of determining whether $\theta_M$ is conservative w.r.t. $G$ is decidable.
Conclusions

- Locality is a problem when a service has to supply a set of operations to play a role.
- Knowledge about the choreography is knowledge about the context, can be exploited to define a WS matching process that preserves goals after substitution.
- A WS can reason about:
  - whether playing a role
  - which partner is better for it
- Ontology matching
- More sophisticated reasoning mechanism (reasoning on coalitions?)
G. Alonso, F. Casati, H. Kuno, and V. Machiraju. 
*Web Services.*

Service selection by choreography-driven matching.

Y. Chen and B. H. C. Cheng.

Dieter Fensel, Holger Lausen, Jos de Bruijn, Michael Stollberg, Dumitru Roman, and Axel Polleres.
B. Fischer and G. Snelting.
Reuse by contract.

Bertrand Meyer.
Applying ”design by contract”.

OWL-S Coalition.
http://www.daml.org/services/owl-s/.

John Penix and Perry Alexander.
Efficient specification-based component retrieval.

Herbert Toth.
On theory and practice of assertion based software development.
WS-CDL.
http://www.w3.org/tr/ws-cdl-10/.

A. Moormann Zaremski and J. M. Wing.
Specification matching of software components.