Evaluating Contract Compatibility for Service Composition in the SeCO$_2$ Framework

Marco Comerio[1], Hong-Linh Truong[2],
Flavio De Paoli[1], and Schahram Dustdar[2]

[1] ITIS lab - Innovative Technologies for Information & Services Laboratory
Università di Milano-Bicocca
{comerio,depaoli}@disco.unimib.it

[2] Distributed Systems Group - Vienna University of Technology
{truong,dustdar}@infosys.tuwien.ac.at

ICSOC-ServiceWave 2009
Stockholm, 23-27 November 2009

Contents

- Motivation and Background
- The SeCO2 Framework
- Modeling and Mapping Service Contracts
- Evaluating Service Contract Compatibility
- Conclusions and Future Works

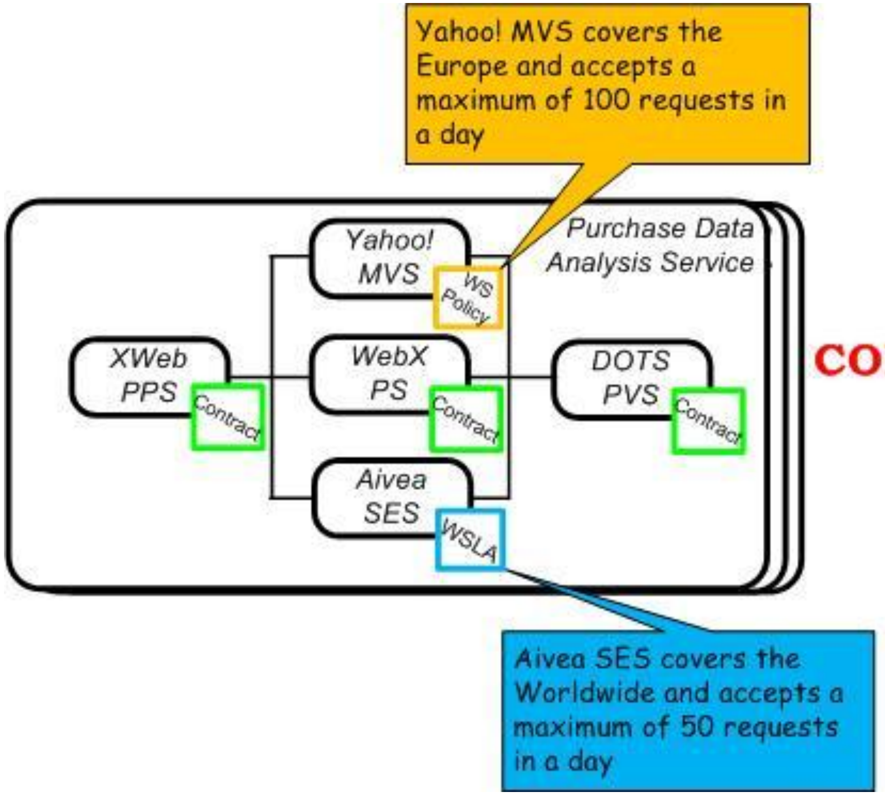
Motivation and Background

- Besides a WSDL document stating the offered functionalities, a Web Service can be characterized by a *service contract* .
- A service contract
  - ✓ establishes the understanding between a service consumer and a service provider;
  - ✓ specifies conditions on NFPs such as:
    - *Quality of Service* (e.g., response time);
    - *Business terms* (e.g., service price);
    - *Context terms* (e.g., service coverage);
    - *License terms* (e.g., limitation of liability).
- No/several standard languages for service contract descriptions
  - ✓ Several proposals (e.g., WSLA[Ludwig03], WSOL[Tosic05] , ODRL-S [Gangadharan07], WS-Policy[wspolicy06])

Motivation and Background (cont.)

- The SaaS model allows service providers to compose different services to provide converged services.
  - ✓ Services are potentially characterizing by different service contracts specified by different languages.
- The emerging DaaS (Data as a Service) offers different views on service contracts (service APIs versus data)
- The service compositions must not include conflicting service contracts.

Motivation and Background (cont.)



Yahoo! MVS covers the Europe and accepts a maximum of 100 requests in a day

Aivea SES covers the Worldwide and accepts a maximum of 50 requests in a day

COMPATIBLE ?

Purchase Processing Service (**PPS**)
Merchant Validation Service (**MVS**)
Payment Verification Service (**PS**)
Shipping Evaluation Service (**SES**)
Purchase Validation Service (**PVS**)

- The heterogeneity of languages specifying contracts
- The compatibility among services in a composition
- The compatibility between a (composite) service and a consumer’s specific-conditions

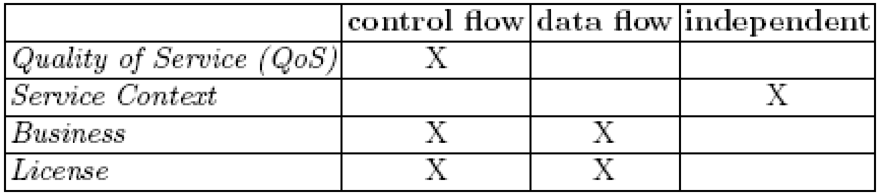
Motivation and Background (cont.)

Past research…

- has neglected contracts of composite services when performing service composition
  - ✓ by considering mainly functional parameters
  - ✓ by assuming that contracts are described by a single language.
- has not focused on tools and algorithms dealing with contract compatibility evaluation when combining different services from different providers.
  - ✓ mainly contract negotiation between consumer and service in a point-to-point manner.

Motivation and Background (cont.)

- Some works (e.g., [Zeng03]) address QoS-based compatibility for control flows of service compositions.
- Currently, no techniques to check contract compatibility for data (i.e., the input/output of services), whose contract terms are not always the same to that of the service operations.
  - ✓ An example is Google Maps: a *free-for-charge service* but the *copyrighted data* (i.e., the maps)
  - ✓ There is still a big debate on data licensing but you can sell your data, e.g., see http://infochimps.org/
- QoS, Business, License and Context terms differently influence data/control flows of the service composition.



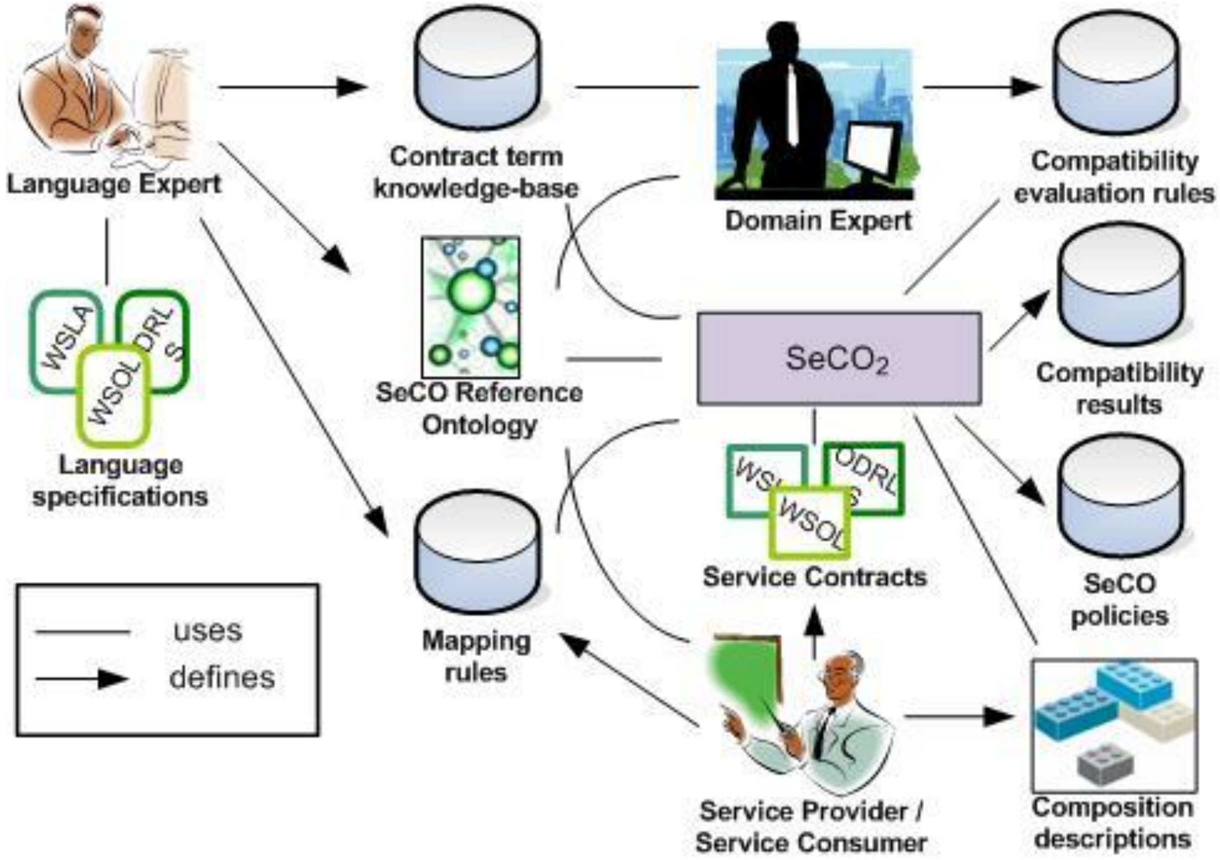
	control flow	data flow	independent
Quality of Service (QoS)	X		
Service Context			X
Business	X	X	
License	X	X	

Table 1. Data and control flows in contract compatibility evaluation

The SeCO$_2$ Framework

- SeCO$_2$ deals with service contract compatibility by considering
  - ✓ two aspects – *service APIs* and *provided data* concerns;
  - ✓ a rich set of contract properties (e.g., *QoS, Data quality, Business, License* and *Context terms*);
  - ✓ several service contract specification languages (e.g., WSLA, WSOL, ODRL-S) together.
- SeCO$_2$ supports
  - ✓ semantic service contract descriptions (namely, *SeCO policies*);
  - ✓ service contract compatibility evaluation and recommendation;
  - ✓ compatibility based on both data and control flows of the service composition;
  - ✓ an extensible reference ontology (namely, *SeCO reference ontology*) and a *Contract term knowledge-base;*
  - ✓ a rich set of mapping and compatibility evaluation rules.

The SeCO2 Framework



The main part of this paper deals with **modeling and mapping service contracts** and **contract compatibility evaluation among services in a composition**

Modeling and Mapping Service Contracts

- ***Problem***: Heterogeneity in service contract specifications.

- Three types of languages for the specification of service contract properties:
  - ✓ **Type A** (e.g., ODRL-S): includes *languages allowing the specification of predefined properties*.
  - ✓ **Type B** (e.g., WSLA): includes *languages allowing the specification of user-defined properties*.
  - ✓ **Type C** (e.g., WSOL): includes *languages allowing the specification of properties defined in user ontologies*.

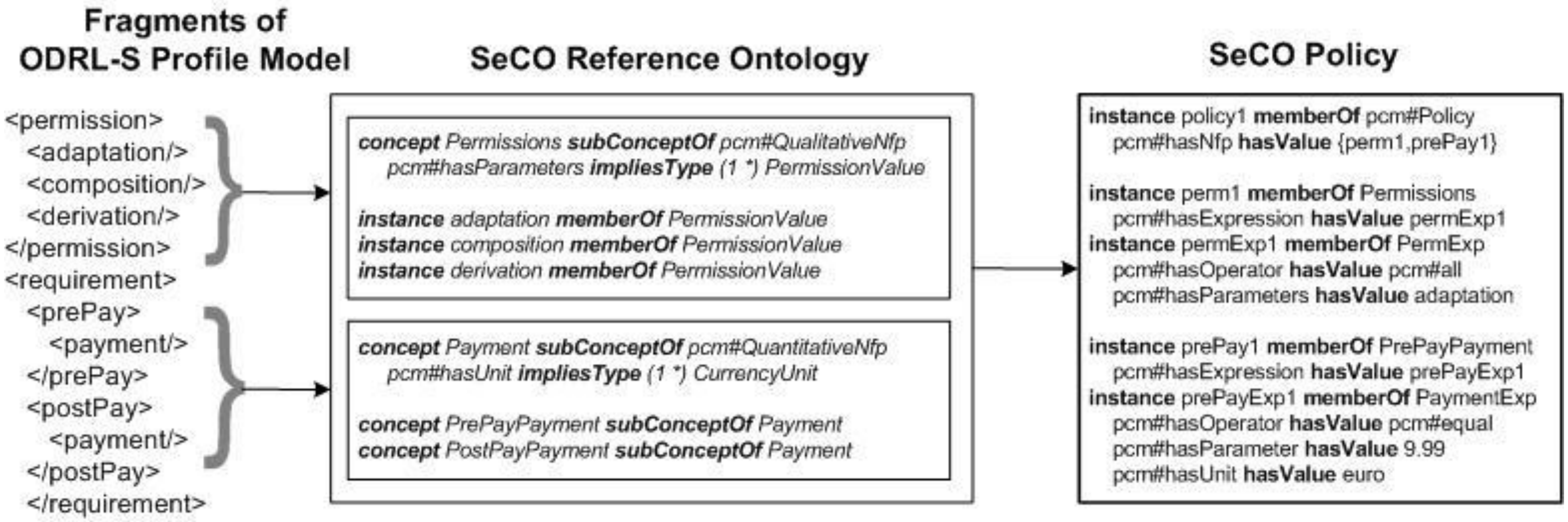
- Ontology alignment tools cannot be used to fully automate the mapping between different specifications.

Modeling and Mapping Service Contracts

- ***Solution:*** SeCO$_2$ makes service contracts comparable through the wrapping to specifications (i.e., *SeCO Policies)* built on a common meta-model
  - ✓ without loss of information;
  - ✓ by means of the *SeCO Reference Ontology* and predefined mapping rules;
  - ✓ supporting the use of lexical databases (e.g., *WordNet*) and ontology alignment tools (e.g., *H-match*).

SeCO Reference Ontology and SeCO Policies

- *SeCO Reference Ontology* and *SeCO Policies*
  - ✓ built on the Policy Centered Meta-model (PCM) [DePaoli08].
- *SeCO Reference Ontology*
  - ✓ built applying general modeling rules to profile models;
  - ✓ defines expressive descriptions of contract properties.
- *SeCO Policies*
  - ✓ represent service contracts defined as clusters of contract property istances.



Fragments of ODRL-S Profile Model

```
<permission>
  <adaptation/>
  <composition/>
  <derivation/>
</permission>
<requirement>
  <prePay>
    <payment/>
  </prePay>
  <postPay>
    <payment/>
  </postPay>
</requirement>
```

SeCO Reference Ontology

```
concept Permissions subConceptOf pcm#QualitativeNfp
    pcm#hasParameters impliesType (1 *) PermissionValue

instance adaptation memberOf PermissionValue
instance composition memberOf PermissionValue
instance derivation memberOf PermissionValue
```

```
concept Payment subConceptOf pcm#QuantitativeNfp
    pcm#hasUnit impliesType (1 *) CurrencyUnit

concept PrePayPayment subConceptOf Payment
concept PostPayPayment subConceptOf Payment
```

SeCO Policy

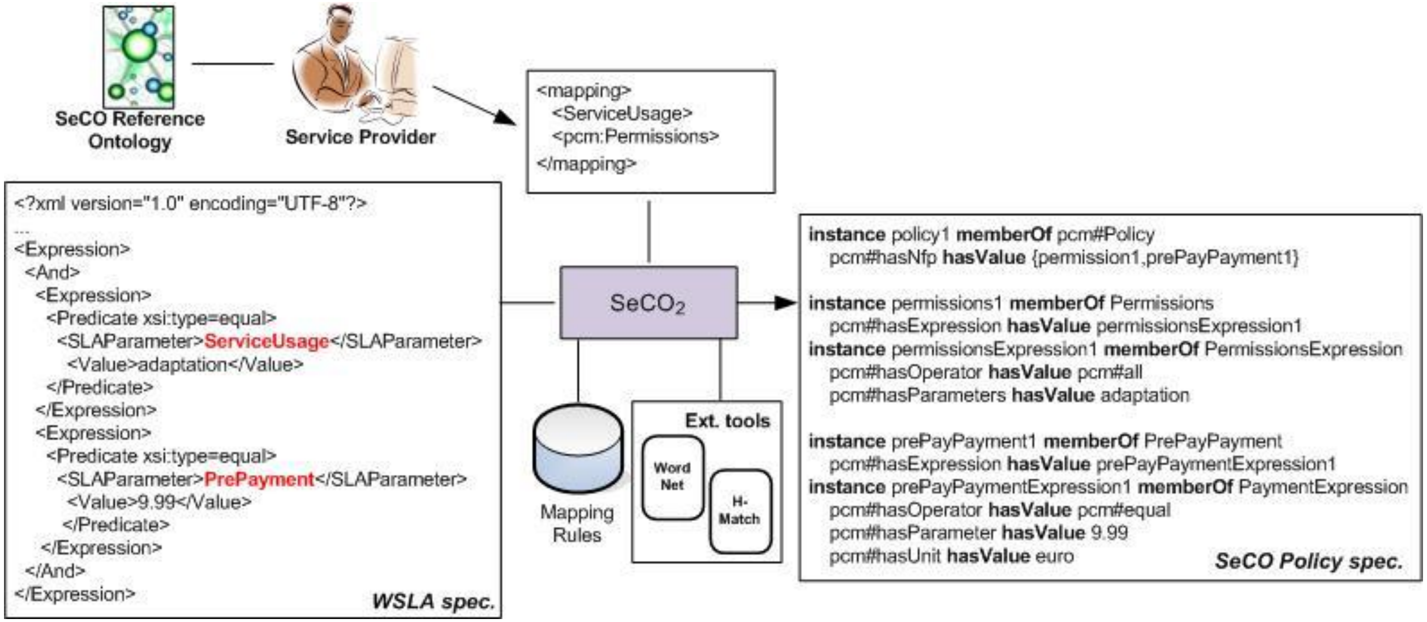
```
instance policy1 memberOf pcm#Policy
  pcm#hasNfp hasValue {perm1,prePay1}

instance perm1 memberOf Permissions
  pcm#hasExpression hasValue permExp1
instance permExp1 memberOf PermExp
  pcm#hasOperator hasValue pcm#all
  pcm#hasParameters hasValue adaptation

instance prePay1 memberOf PrePayPayment
  pcm#hasExpression hasValue prePayExp1
instance prePayExp1 memberOf PaymentExp
  pcm#hasOperator hasValue pcm#equal
  pcm#hasParameter hasValue 9.99
  pcm#hasUnit hasValue euro
```

Mapping Service Contracts

- A proper technique for each type of language
  - ✓ Specifications in **Type A** are wrapped applying fixed mapping rules.
  - ✓ Specifications in **Type B** and **Type C** can require interactions with service providers to handle the absence of knowledge (i.e., mapping rules).
    - The definition of new mapping rules is supported by lexical databases and ontology alignment tools.



```
<mapping>
  <ServiceUsage>
  <pcm:Permissions>
</mapping>
```

```
<?xml version="1.0" encoding="UTF-8"?>
...
<Expression>
 <And>
  <Expression>
   <Predicate xsi:type=equal>
    <SLAParameter>ServiceUsage</SLAParameter>
     <Value>adaptation</Value>
   </Predicate>
  </Expression>
  <Expression>
   <Predicate xsi:type=equal>
    <SLAParameter>PrePayment</SLAParameter>
     <Value>9.99</Value>
    </Predicate>
  </Expression>
 </And>
</Expression>
```

WSLA spec.

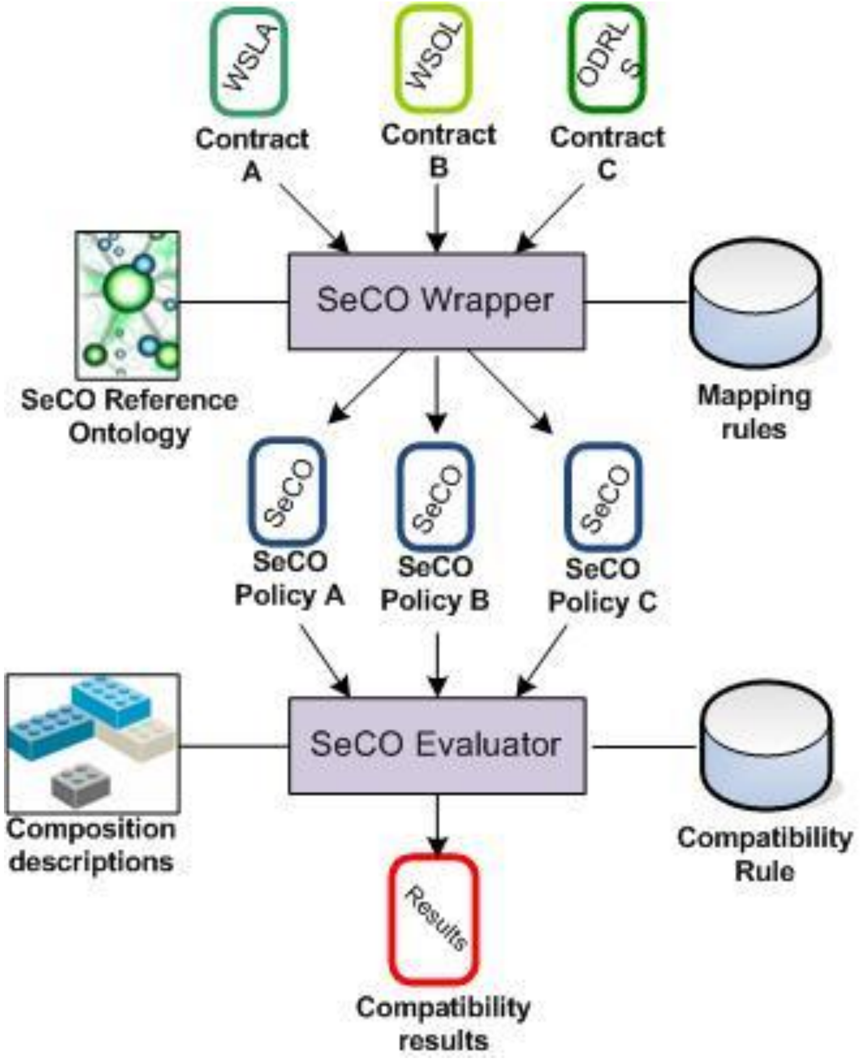
```
instance policy1 memberOf pcm#Policy
  pcm#hasNfp hasValue {permission1,prePayPayment1}

instance permissions1 memberOf Permissions
  pcm#hasExpression hasValue permissionsExpression1
instance permissionsExpression1 memberOf PermissionsExpression
  pcm#hasOperator hasValue pcm#all
  pcm#hasParameters hasValue adaptation

instance prePayPayment1 memberOf PrePayPayment
  pcm#hasExpression hasValue prePayPaymentExpression1
instance prePayPaymentExpression1 memberOf PaymentExpression
  pcm#hasOperator hasValue pcm#equal
  pcm#hasParameter hasValue 9.99
  pcm#hasUnit hasValue euro
```

SeCO Policy spec.

Evaluating Service Contract Compatibility: activities and flows



Service Contract Mapping

Service Contract Evaluation

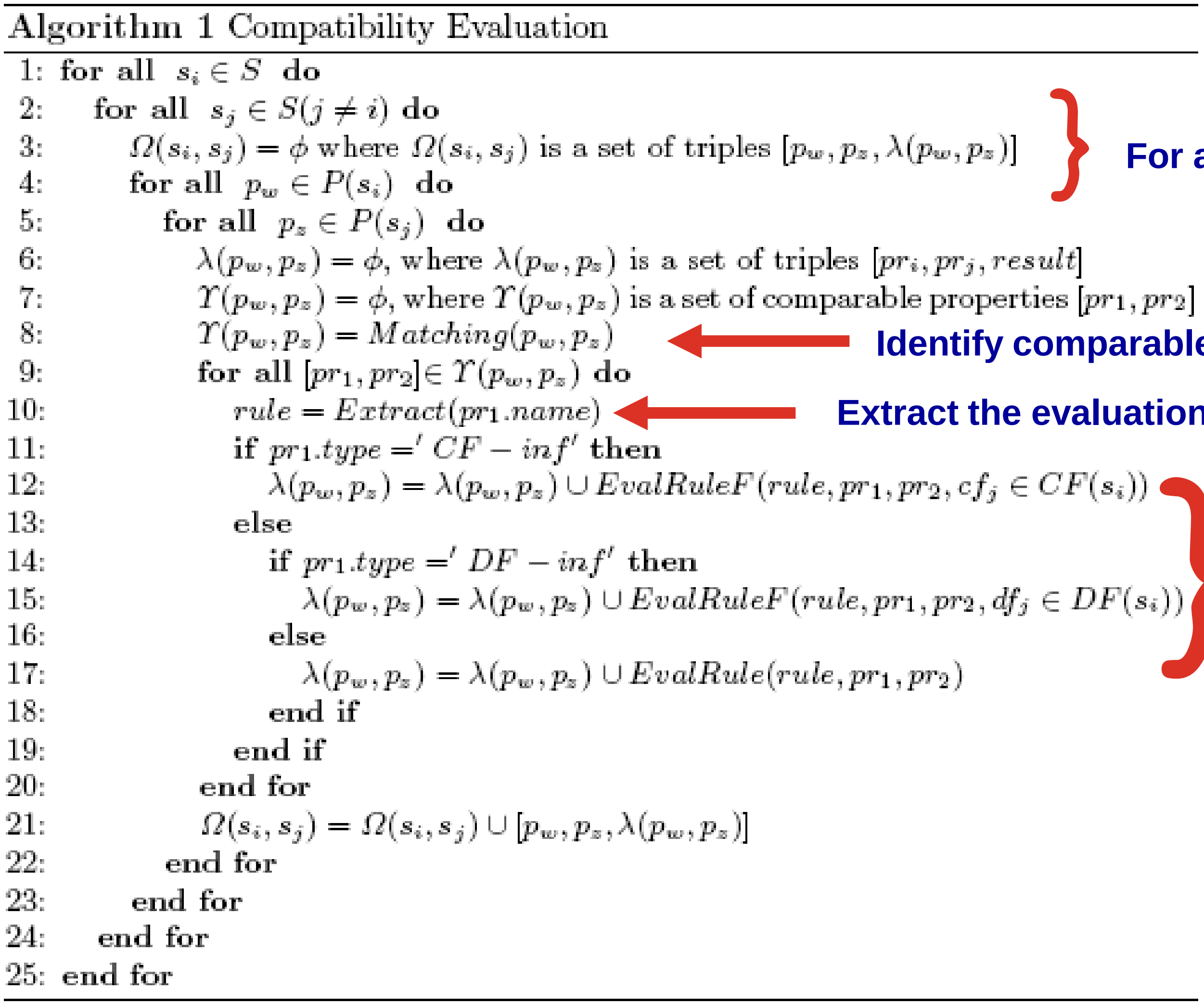
Evaluating Service Contract Compatibility

- ***Problem:*** evaluation of contract compatibility in a service composition.
- Input:
  - ✓ service composition description in terms of data and control flows;
  - ✓ contracts of the services involved in the composition.
- Output:
  - ✓ compatible/incompatible service contract properties.
- The compatibility is checked considering
  - ✓ semantic relations among values associated with qualitative contract properties;
  - ✓ constraint operators used to define quantitative contract properties;
  - ✓ data and control flows of the service composition.

Compatibility Evaluation Rules

Property	*Type*	*Data Flow*	*Control Flow*	*Rule*
Service Coverage	Service Context			Partnership
Pricing	Business	X		Compatible value list
Payment (for data usage)	Business	X		Binary, Ternary
Payment (for service usage)	Business		X	Binary, Ternary
Scalability	QoS		X	Binary, Ternary
Permissions	License		X	Subsumption
Data Ownership	License	X		Compatible value list

Evaluating Service Contract Compatibility



Algorithm 1 Compatibility Evaluation

```
for all $s_i \in S$ do
    for all $s_j \in S(j \neq i)$ do
        $\Omega(s_i, s_j) = \phi$ where $\Omega(s_i, s_j)$ is a set of triples $[p_w, p_z, \lambda(p_w, p_z)]$
        for all $p_w \in P(s_i)$ do
            for all $p_z \in P(s_j)$ do
                $\lambda(p_w, p_z) = \phi$, where $\lambda(p_w, p_z)$ is a set of triples $[pr_i, pr_j, result]$
                $\Upsilon(p_w, p_z) = \phi$, where $\Upsilon(p_w, p_z)$ is a set of comparable properties $[pr_1, pr_2]$
                $\Upsilon(p_w, p_z) = Matching(p_w, p_z)$
                for all $[pr_1, pr_2] \in \Upsilon(p_w, p_z)$ do
                    $rule = Extract(pr_1.name)$
                    if $pr_1.type =' CF - inf'$ then
                        $\lambda(p_w, p_z) = \lambda(p_w, p_z) \cup EvalRuleF(rule, pr_1, pr_2, cf_j \in CF(s_i))$
                    else
                        if $pr_1.type =' DF - inf'$ then
                            $\lambda(p_w, p_z) = \lambda(p_w, p_z) \cup EvalRuleF(rule, pr_1, pr_2, df_j \in DF(s_i))$
                        else
                            $\lambda(p_w, p_z) = \lambda(p_w, p_z) \cup EvalRule(rule, pr_1, pr_2)$
                        end if
                    end if
                end for
                $\Omega(s_i, s_j) = \Omega(s_i, s_j) \cup [p_w, p_z, \lambda(p_w, p_z)]$
            end for
        end for
    end for
end for
```

For all SeCO Policy couples

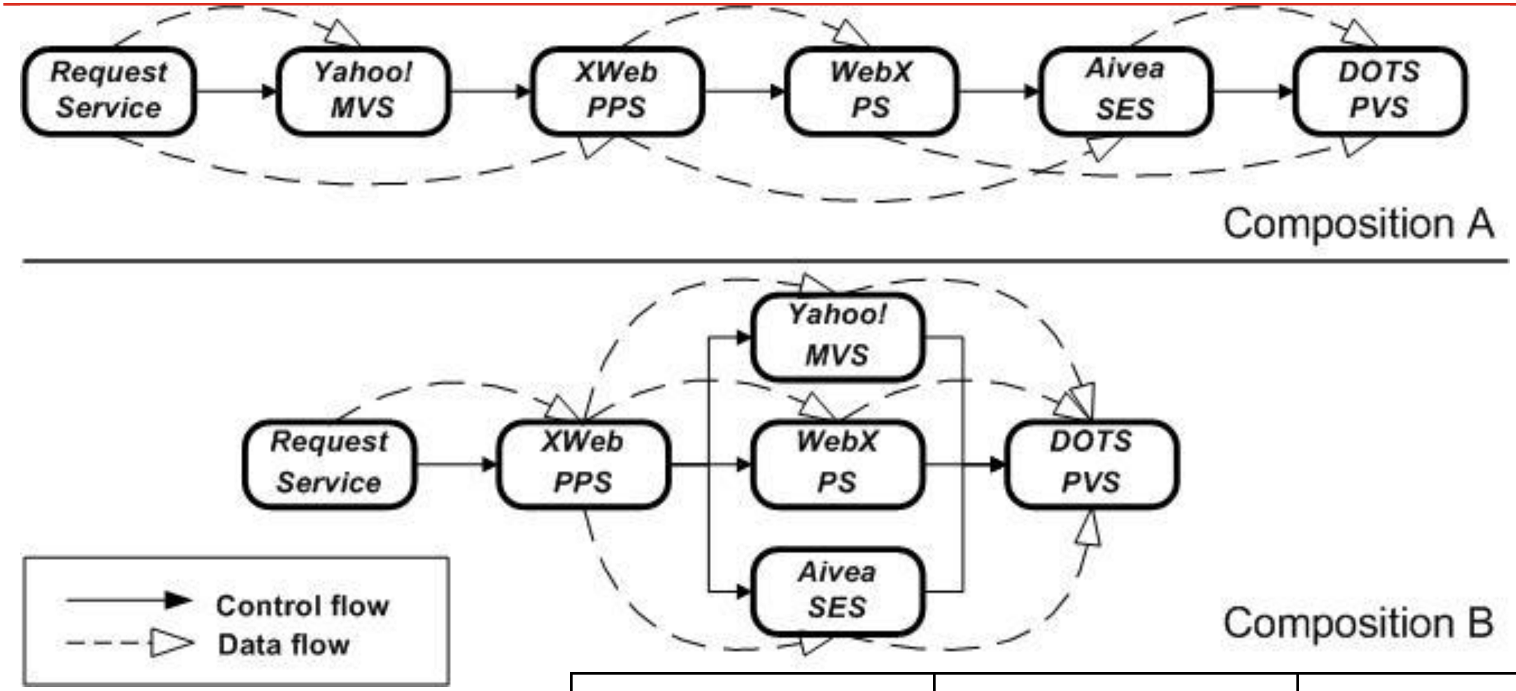
Identify comparable SeCO properties

Extract the evaluation rule

Evaluate according to flow influences

Illustrating Example

Purchase Data Analysis Service



Purchase Processing Service (**PPS**)
Merchant Validation Service (**MVS**)
Payment Verification Service (**PS**)
Shipping Evaluation Service (**SES**)
Purchase Validation Service (**PVS**)

	Data Ownership	**Scalability**
Request Service	Personal-use	100 tr/min
Yahoo! MVS	Copyrighted	100 tr/min
XWeb PPS	Free-distribution	100 tr/min
Aivea SES	Free-distribution	100 tr/min
WebX PS	Free-distribution	500 tr/min
DOTS PVS	Free-distribution	500 tr/min

Illustrating Example

- *Data Ownership* :
  - ✓ a *License* term stating how the data are protected;
  - ✓ influences the *data flow* of the service composition;
  - ✓ assumes values characterized by relations of compatibility/incompatibility
    - *copyrighted* is compatible with *personal-use*
    - *copyrighted* is incompatible with *free-distribution*
- *Scalability* :
  - ✓ a *QoS* term indicating the maximum number of transactions accepted per minute.
  - ✓ influences the *control flow* of the service composition;
  - ✓ assumes numeric values.

Illustrating Example

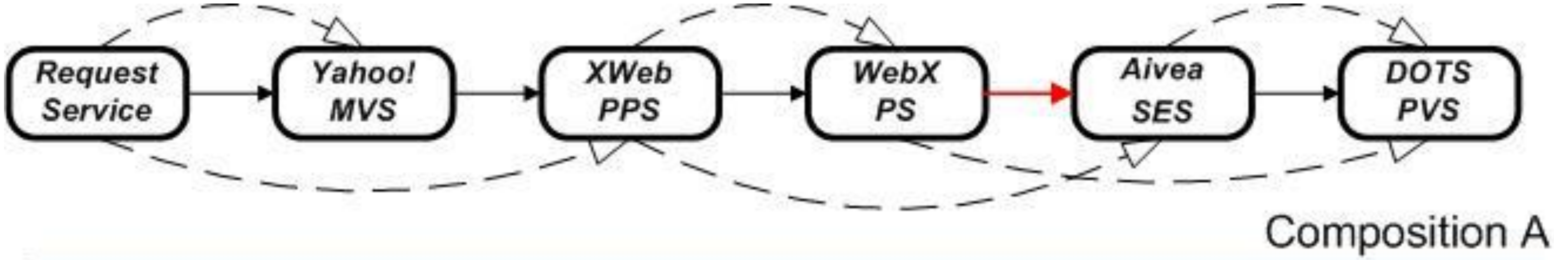
- *Data Ownership is evaluated exploiting the axiom:*

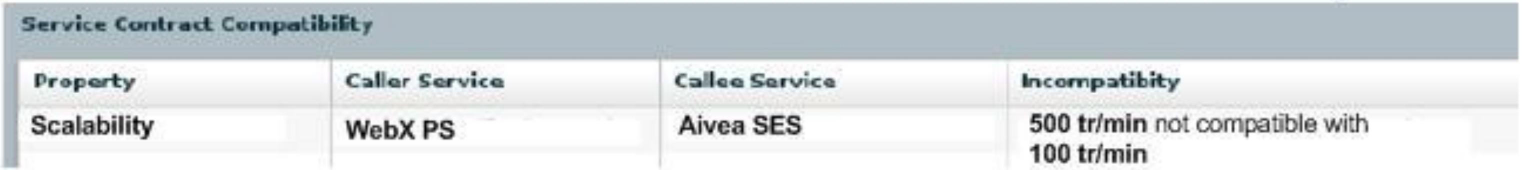
```
axiom dataOwnershipCompatibility
    definedBy
          compatible ( ?X , ?Y) :-
                    ( ?X memberOf seco#DataOwnValue) and
                    ( ?Y memberOf  seco#DataOwnValue) and
                    seco#compatible( ?X, ?Y)
```

- *Scalability is evaluated applying the algorithm*

```
Given pr1,pr2
if(([pr1,pr2].equals("seq"))||([pr1,pr2].equals("par"))){
          if(pr2.value<pr1.value)
                    result = "INCOMPATIBLE";
          else
                    result = "COMPATIBLE"; }
```

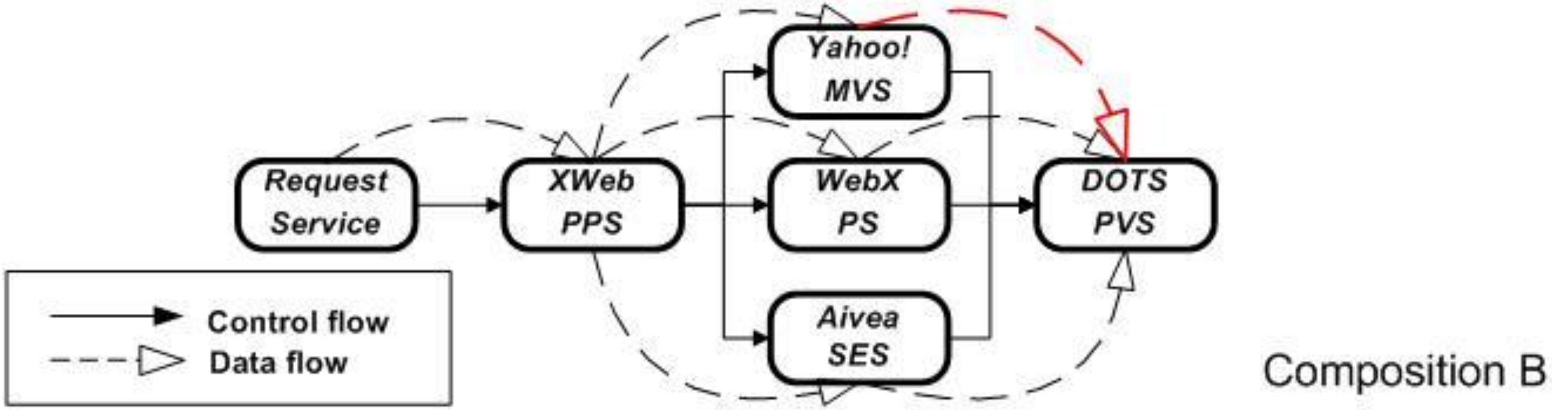
Illustrating Example

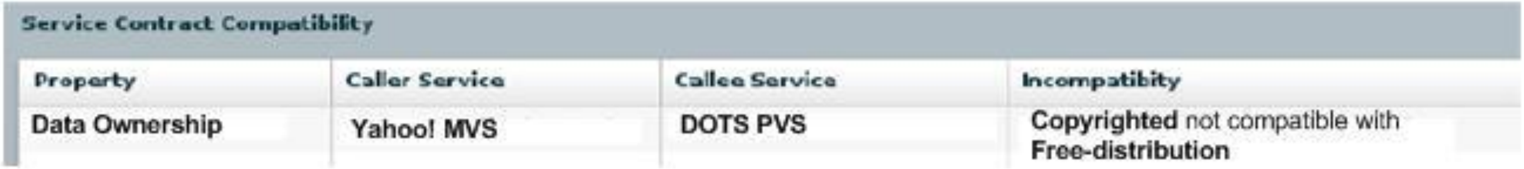




Service Contract Compatibility

Property	Caller Service	Callee Service	Incompatibity
Scalability	WebX PS	Aivea SES	500 tr/min not compatible with 100 tr/min





Service Contract Compatibility

Property	Caller Service	Callee Service	Incompatibity
Data Ownership	Yahoo! MVS	DOTS PVS	Copyrighted not compatible with Free-distribution

Some open issues

- Human activity/workflow dealing with modeling and mapping service contract specifications
  - ✓ define how to interact with service providers when automatic mapping cannot be done.
- The role of the community in the mapping activity
  - ✓ reuse of user-defined mapping rules.
- Compatibility Evaluation Rules
  - ✓ support the definition of general rules.
  - ✓ allow the customization of general rules.
  - ✓ manage conflicting rules and rule priority.
  - ✓ optimization of the compatibility algorithm.

Conclusions and Future Works

- SaaS and DaaS and cloud computing require a strong support on contract compatibility
  - ✓ Deal with multiple languages, focus multiple aspects in particular those related to data (quality, licensing, and governance)
- Our SeCO2 in this paper
  - ✓ proposes some solutions for dealing with multiple languages and service contract compatibility
- Future works
  - ✓ Incorporating human activities and community support into contract mapping and sharing
  - ✓ Recommending contracts for service composition

Thank you!
Questions?

Source codes will be available in sourceforget.net in Spring 2010

References

- [Gangadharan07] Gangadharan, G.R., D'Andrea, V., Iannella, R., Weiss, M.: "ODRL Service Licensing Profile (ODRL-S)". In: "Proceedings of the 5th International Workshop for Technical, Economic, and Legal Aspects of Business Models for Virtual Goods". (2007)
- [Ludwig03] Ludwig, H., Keller, A., Dan, A., King, R., Franck, R.: "Web Service Level Agreement (WSLA) Language Specification". IBM Coporation (2003)
- [owls03] OWL-S. Semantic Markup for Web Services. Available at: http://www.daml.org/services/owl-s/1.0/owl-s.html, 2003.
- [Tosic05] Tosic, V., Pagurek, B., Patel, K., Esfandiari, B., Ma, W.: "Management Applications of the Web Service Offerings Language (WSOL)". Information Systems 30(7) (2005) 564-586.
- [Akkiraiu05] Akkiraju, R., Farrell, J., Miller, J., Nagarajan, M., Schmidt, M.- T., Sheth, A., and Verma, K. (2005). Web Service Semantics - WSDL-S. W3C Member Submission 7 November 2005. http://www.w3.org/Submission/WSDL-S/.
- [wspolicy06] Ws-Policy. *Web Service Policy 1.2 - Framework*. Available at: http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/, 2006.
- [wsmo05] WSMO. The Web Service Modeling Ontology (WSMO). Final Draft. Available at: http://www.wsmo.org/TR/d2/v1.2/20050413/, 2005.
- [Zeng03] Zeng L., Benatallah B., Dumas M., Kalagnanam J. and Sheng Z.. Quality Driven Web Services Composition, WWW '03, pages 411–421, 2003.
- [Castano05] S. Castano, A. Ferrara and S. Montanelli. Matching Ontologies in Open Networked Systems: Techniques and Applications. Journal on Data Semantics (JoDS), 2005.
- [DePaoli08] De Paoli F., Palmonari M., Comerio M. and Maurino A. *A Meta-Model for Non-Functional Property Descriptions of Web Services.* In proc. of ICWS 2008.