POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

November, 4th, 2009

Lecture on

Web services retrieval

Pierluigi PLEBANI

Dipartimento di Elettronica ed Informazione - Politecnico di Milano
plebani@elet.polimi.it

Before starting ...

- Web service retrieval is only the last problem
- We have had:
  - plumber retrieval
  - data retrieval
  - document retrieval
  - software component retrieval
  - ... and now Web service retrieval
- We consider discovery and retrieval as synonyms

... a look at the real world

- Once upon a time...
  - Friends of mine
  - Friends of friends of mine
  - ... (Friends of)^n mine with $1 \leq n \leq 6$
- Advertising rules!
  - White pages
  - Yellow pages
- e-Advertising rules!
  - http://www.whitepages.com
  - http://www.yellowpages.com

POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Who, What, Where, When, Why, and How

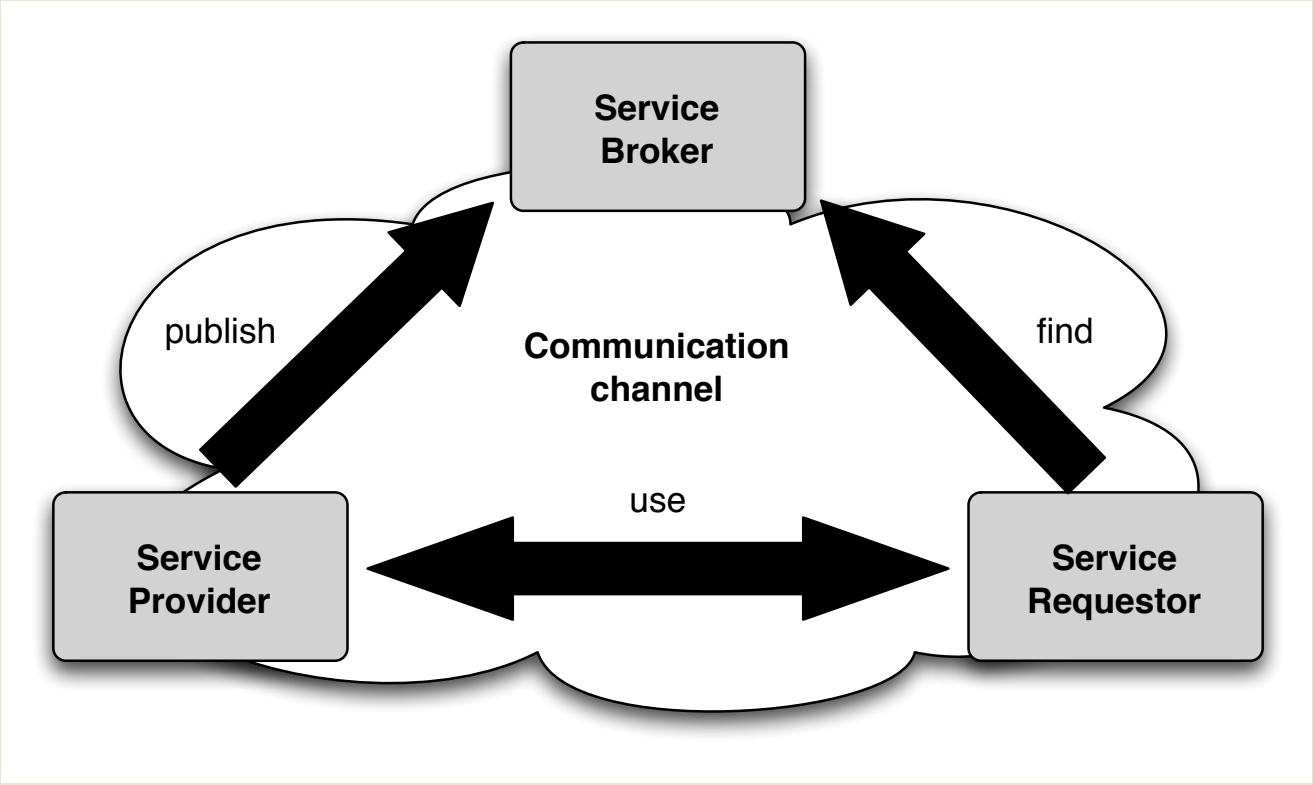
Lecture on "Web services retrieval"

Pierluigi Plebani

November, 4th, 2009

Who does retrieve Web services?

- Web services retrieval is one of the fundamental steps in SOA
- Final users need to retrieve Web services
- We need to consider Web service providers as well



What do we retrieve? 1/2

- We need to find a Web service (obviously)
- But, which one? The one:
  - able to perform what we need
  - accessible in a way we need
  - working in a way we need

What do we retrieve? 2/2

- A shared model for both Web service providers and Web service users is required
- This model must consider:
  - functionalities
  - conversation
  - quality
- Lot of specifications are available today:
  - WSDL
  - WS-CDL
  - WS-BPEL
  - WS-Policy
  - ... and many others

Where do we retrieve Web services?

- All the information should be collected and stored in well known places:
  - centralized solution
  - distributed or peer-to-peer solution
- Who has the ownership on this information?
  - registry
  - repository

When do we retrieve Web services?

- At design-time
  - we can code the client-side
- At deployment-time
  - we need a declarative model
- At run-time
  - we need... something

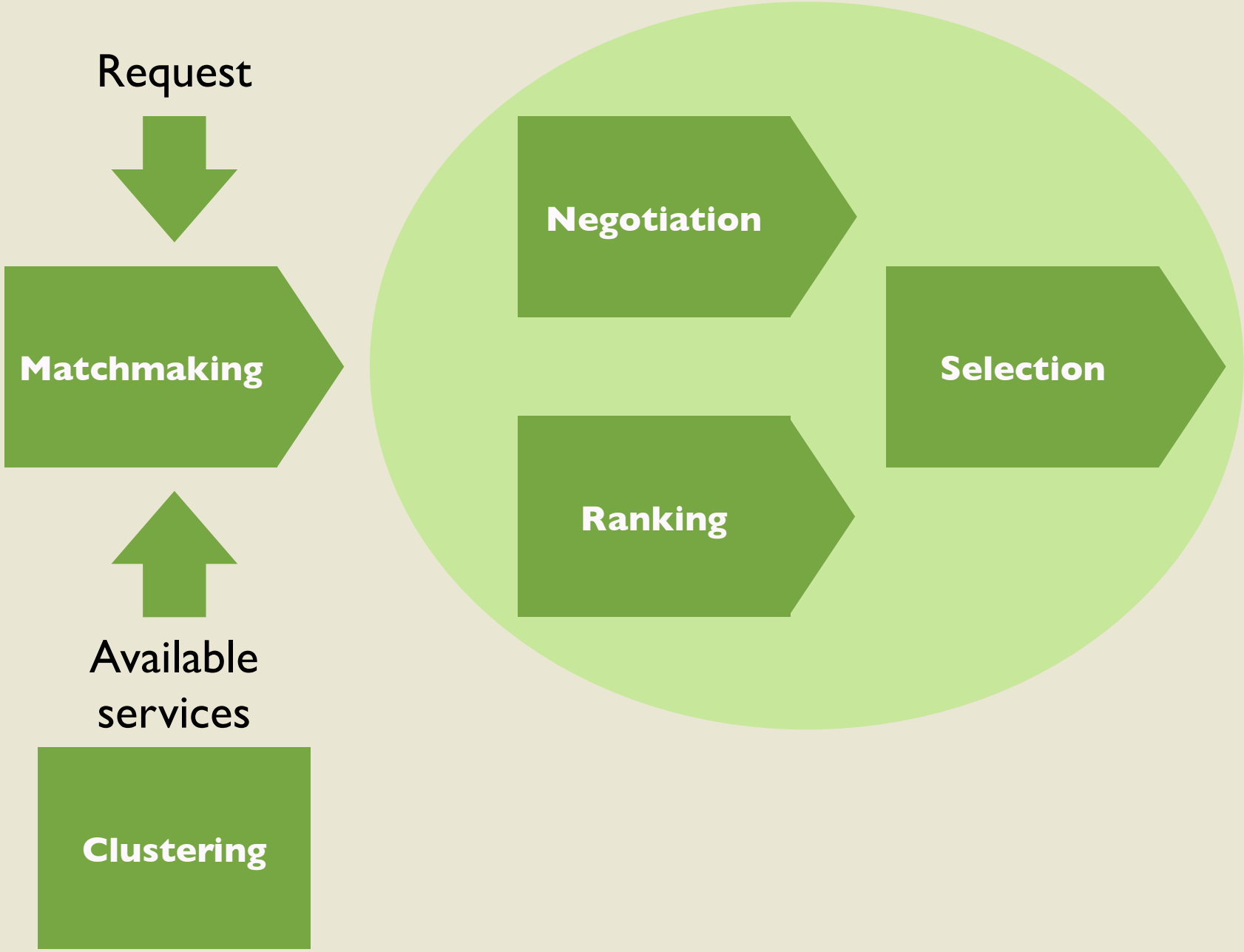
Why do we retrieve Web services?

- Only for a single invocation
- For building a partnership
- As a part of my application
- As the whole application

How do we retrieve Web services? 1/2

- (Friends of)^n mine with $1 \leq n \leq 6$
- Browsing the Web (XMethods, SALCentral (?))
- Googling
- Seekda!
- White pages
- Yellow pages
- Are we re-inventing the wheel?
  - if you think so, try to describe your plumber with WSDL
  - or to call him by SOAP

How do we retrieve Web services? 2/2



POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

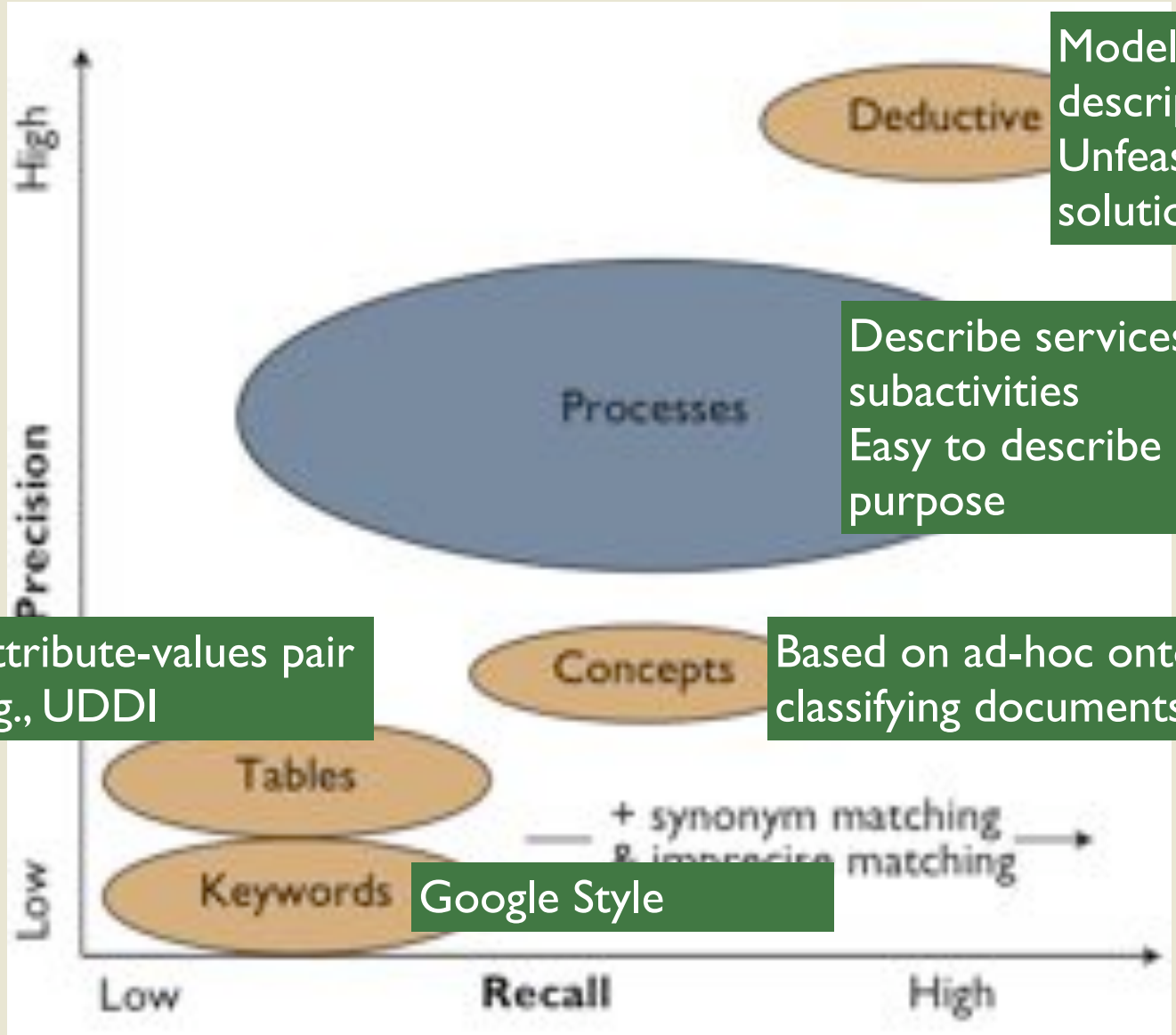
State of the Art

Lecture on "Web services retrieval"

Pierluigi Plebani

November, 4th, 2009

A possible roadmap



Modeling query and description with logics.
Unfeasible as a general purpose solution

Describe services as interlinked subactivities
Easy to describe and general purpose

Attribute-values pair
e.g., UDDI

Based on ad-hoc ontology classifying documents

Google Style

from E. Klein, A. Bernstein, Toward High-Precision Service Retrieval, IEEE Internet Computing, Jan-Feb 2004

Web service description

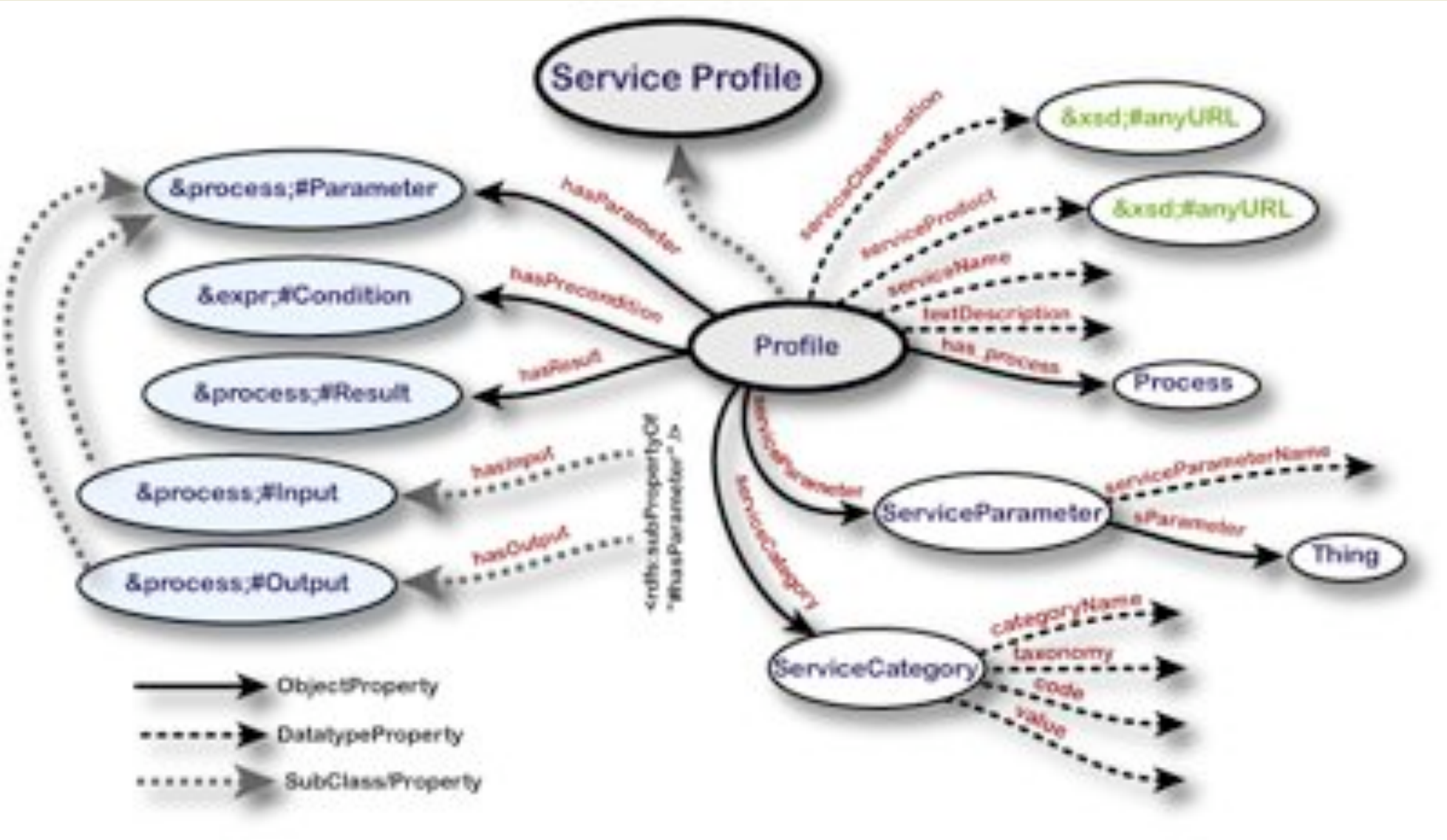
- Some current useful specifications
  - WSDL
  - SAWSDL (formerly WSDL-S)
  - OWL-S (formerly DAML-S)
  - WSMO
  - WS-Policy
  - WSOL
  - WS-CDL
  - WS-BPEL
  - ...
- How much Web service description costs?

OWL-S: the upper ontology

- Languages for specifying Web service ontology
- Based on OWL (formerly DAML)

All the images about OWL-S are from OWL-S Web site (http://www.ai.sri.com)

OWL-S: Service Profile (v.1.1)



WSMO

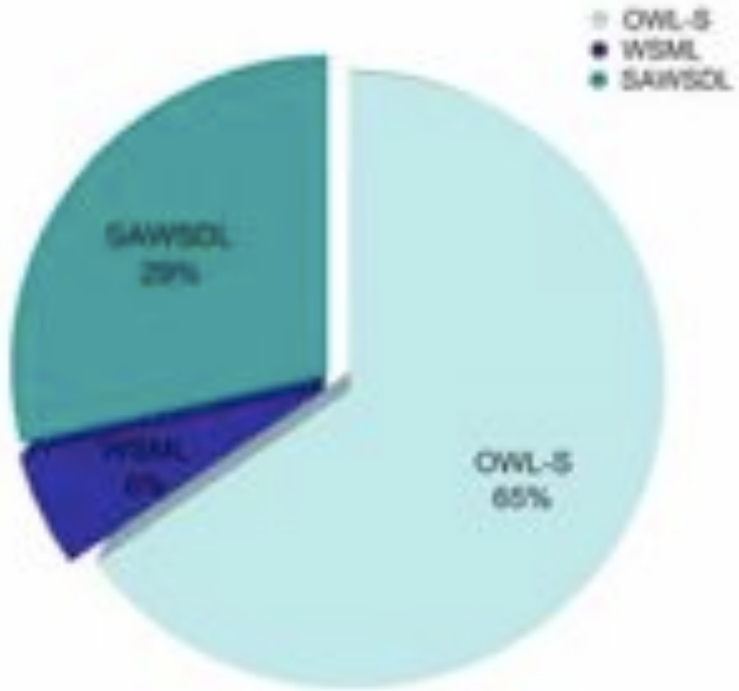
- WSMO provides ontological specifications for the core elements of Semantic Web services:
  - Web services, Goal, Mediator, Ontology
- We are mainly focused on the Web service definition

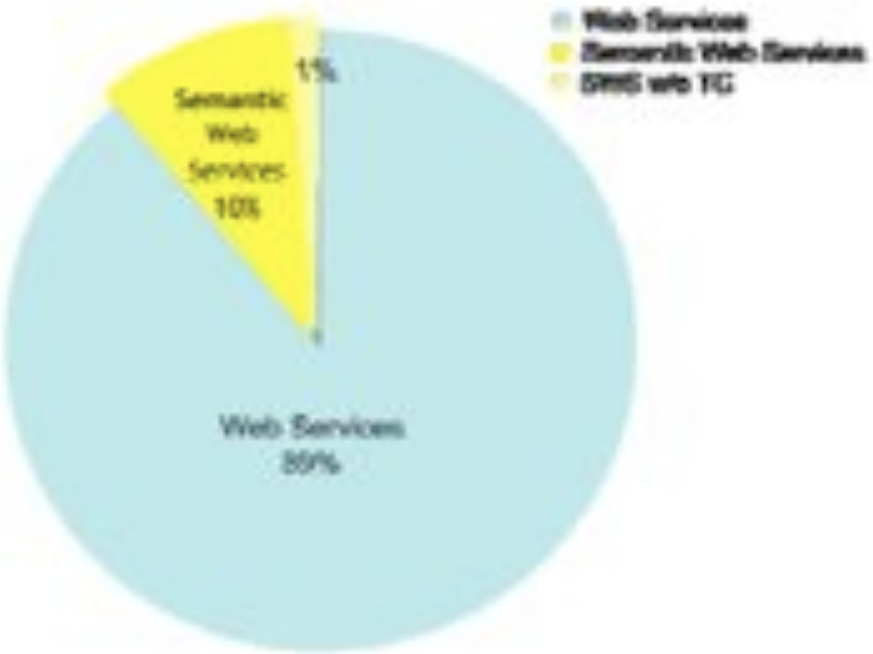
Class webService **sub-Class** wsmoElement
importsOntology **type** ontology
usesMediator **type** {ooMediator, wwMediator}
hasNonFunctionalProperties **type**?nonFunctionalProperty
hasCapability **type** capability *multiplicity = single-valued*
hasInterface **type** interface

Class capability **sub-Class** wsmoElement
importsOntology **type** ontology
usesMediator **type** {ooMediator, wgMediator}
hasNonFunctionalProperties **type**?nonFunctionalProperty
hasSharedVariables **type** sharedVariables
hasPrecondition **type** axiom
hasAssumption **type** axiom
hasPostcondition **type** axiom
hasEffect **type** axiom

SAWSDL

How many Web services are around?





Sousuo 24-09-09: **410** w/o test collections

3508 w/ test collections

Public semantic service retrieval test collections:

- OWL-S: OWLS-TC2 (semwebcentral.org), TC (ce.sharif.edu)
- SAWSDL: SAWSDL-TC1 (semwebcentral.org)
- None for WSML yet.

Seekda 24-09-09

Source: M. Klusch, 3rd International Semantic Service Selection Contest - International Semantic Service Selection Contest, October 2009

Some of the current approaches

- Interface matching
  - Stroulia and Yang, Woogle (WSDL)
- Semantic matching
  - OWL-S MM, WSMO MM
- Hybrid matching
  - Lumina (SAWSDL)
- Quality driven matching
  - WSOI (WSOL), UDDIe (Proprietary Language)
- Hybrid+Quality matching
  - URBE (WSDL, SAWSDL, WS-Policy)
- What about behavior?

Interface matching: Zaremski & Wing 1/2

- Proposed for reusable components
- Introduces concepts useful and used in Web service retrieval
- Two kinds of similarity evaluations:
  - signature matching
  - specification matching
- Various degrees of similarity
  - exact-match
  - several relaxed matches

Interface matching: Zaremski & Wing 2/2

- Signature matching: based on data type analysis
  - Exact match
  - Partial match (generalized match, specialized match)
  - Relax match (generalized relax match, specialized relax match)
- Specification matching: based on pre- and post-condition analysis
  - Exact match
  - Plug-in match
  - Plug-in post match
  - Guarded post match

Interface matching: Stroulia and Yang

- Two main aspects
  - structural similarity based only on data type analysis (casting)
  - semantic similarity based on operations and parameters names
- This approach also considers the documentation field
  - relies on IR approach (tf/idf)
- Term similarity evaluation is based on Wordnet

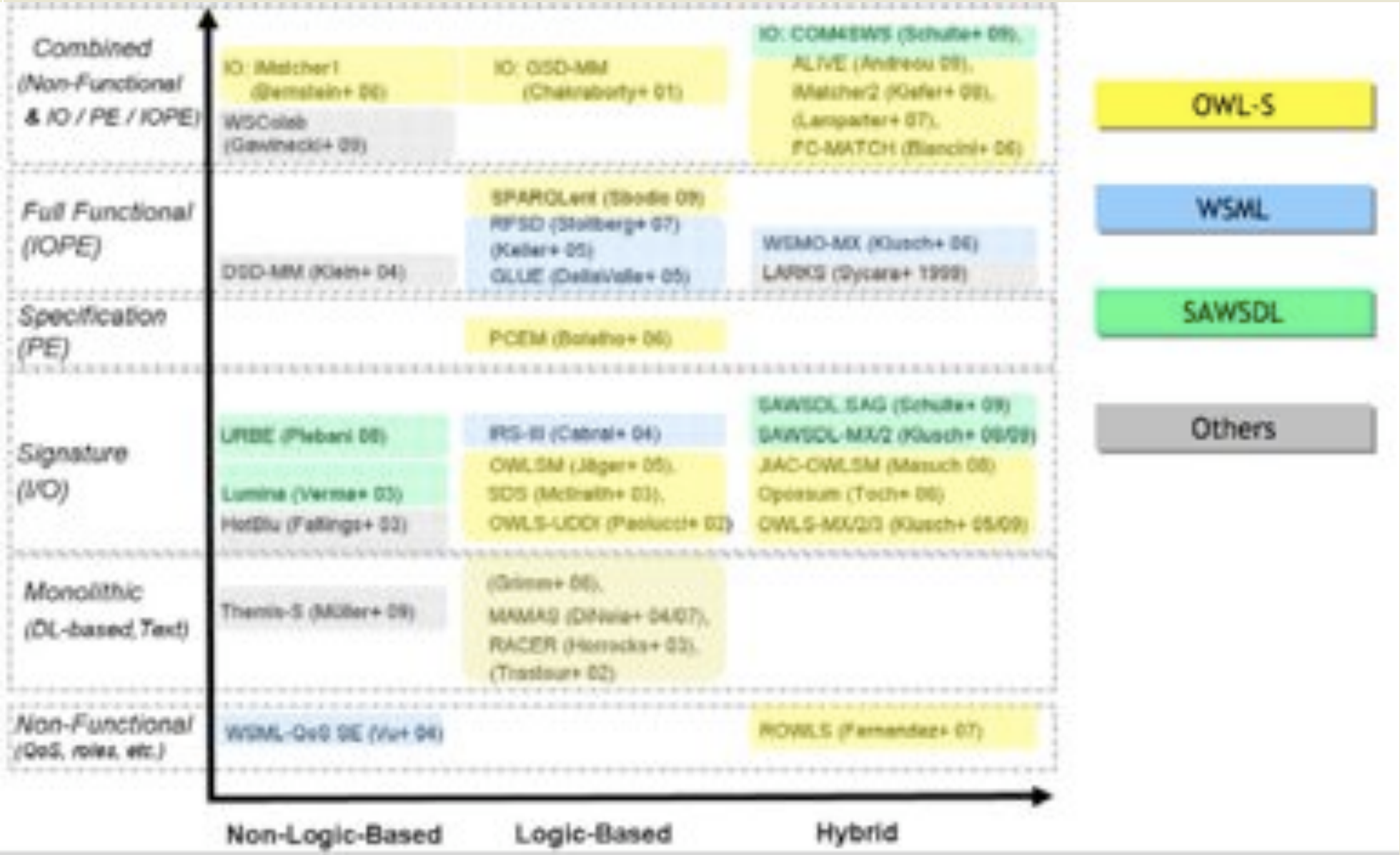
Interface matching: Woogle

- Proposed by Dong et al. at VLDB 2004
  - now it seems to be abandoned
- Operation-based query
- Based on parameter names clustering
  - parameters tend to express the same concept if they occur together often
- Operation matching is based on the defined clusters
- Tool available on line http://data.cs.washington.edu/webService/

OWL-S and WSMO Matchmaking

- Web services are semantically described using OWL-S or WSMO
- Matchmakers take advantage of these semantic descriptions
- Matchmaking relates to reasoning on ontology
  - concepts composing web services are related
  - the more strict is the relationship the more similar are the service
- Classes of similarities:
  - exact
  - plug-in
  - subsumes
  - fail

Existing semantic-based approaches



Source: M. Klusch, 3rd International Semantic Service Selection Contest - International Semantic Service Selection Contest, October 2009

Hybrid approaches

- Considers both interfaces and semantics
- Usually based on annotations
- SAWSDL extends WSDL with annotations offering semantic description about operations, messages, parameters
- Good balance between:
  - expressiveness
  - feasibility
- What about the matchmaker?

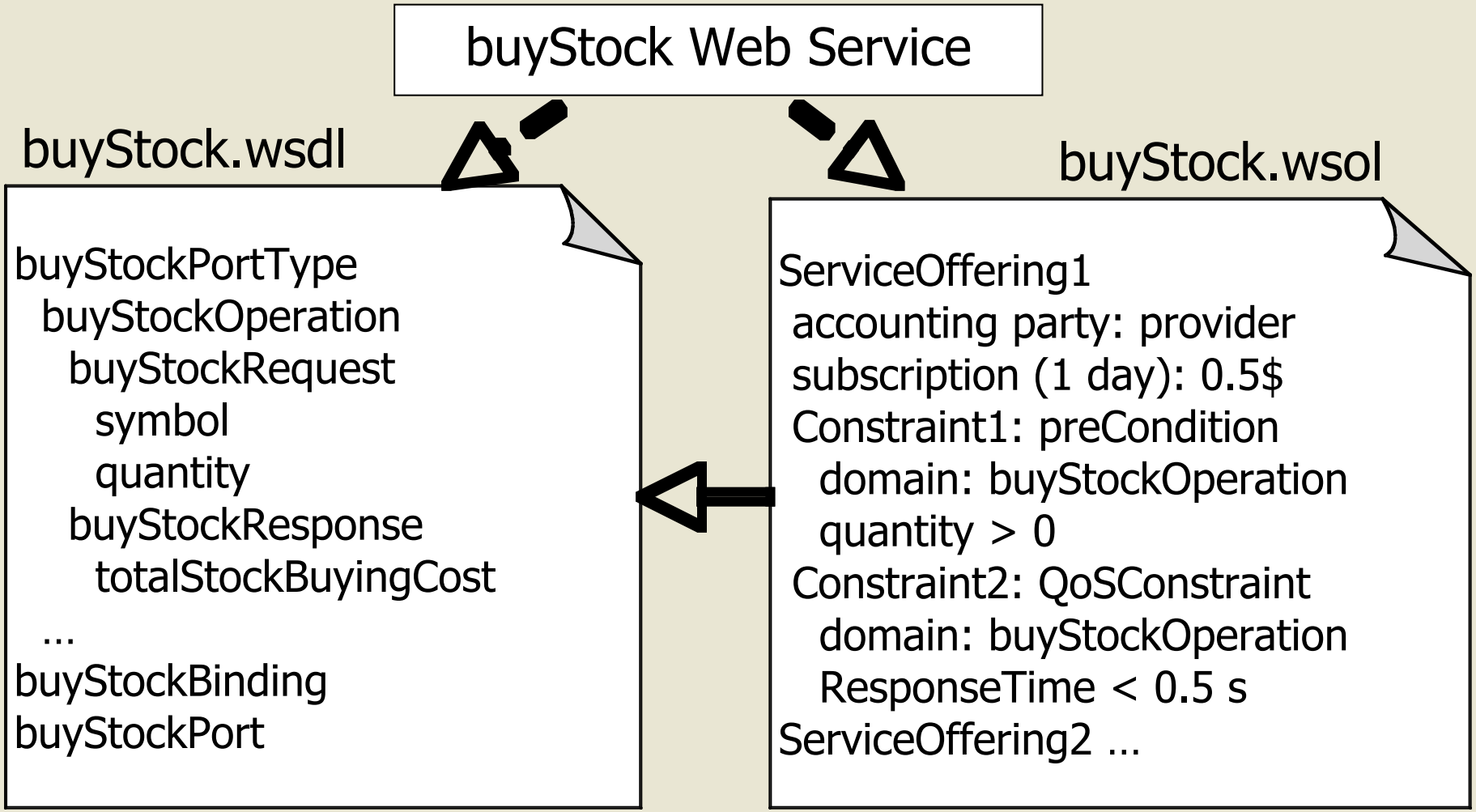
Quality Driven matching

- Service Oriented Computing is based on the loosely-coupling concept
- A common way to describe Web service quality is required
  - No quality language currently has the same consensus as WSDL for functionalities
- Web service retrieval will exploit such a description matching quality required and quality offered
- At this stage the quality matching approaches can be compared to constraints satisfaction problems

WSOL

- Proposed by V. Tosic, K. Patel, and B. Pagurek, Carleton University, Ottawa, CA
- Describes classes of services and their relationships
- Multiple categories of constraints:
  - Functional
  - QoS
  - Access rights
  - Context
- QoS metrics are defined in external ontologies

WSOL: example

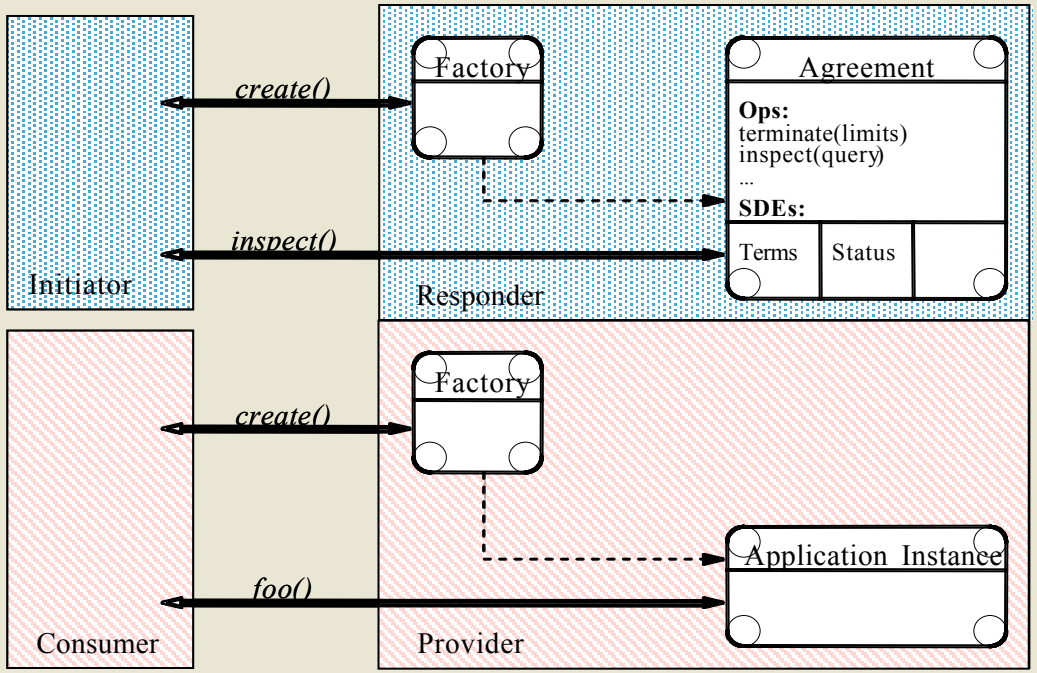


from Tosic, V. "Service Offerings for XML Web Services and Their Management Applications", Ph.D. dissertation, Carleton University, Ottawa, Canada, August 2004

WS-Agreement

- WS-Agreement itself is sort of a frame-work for getting agreements on several different domains. The details of things to be agreed are of course domain-specific and is out of the scope of the WS-Agreement Spec.
  - e.g., Job-submission using JSDL is a candidate to be used within the frame-work.
- Relation to other specification
  - Relies on WS-Addressing, WS-ResourceProperties, WS-ResourceLifetime and WS-Base Faults

WS-Agreement: the model



Agreement Layer

Agreement Layer: Provides a Web service-based interface that can be used to create, represent and monitor agreements with respect to provisioning of services implemented in the service

Service Layer

The service layer represents the application-specific layer of the service being provided.
The class of provided service MAY or MAY NOT be a Web service interface.

Whether Agreement Initiator becomes a Service Consumer or Service Provider (i.e. Agreement Responder becomes a Service Provider or service Consumer) is completely domain dependent

UDDIe

- Proposed by A. Shaikhali and O. Rana (University of Cardiff)
- Extends UDDI Registry with quality aspects introducing:
  - blue pages: discovery based upon information that describes the service known as the service properties
  - service leasing: limited and unlimited
  - new find_service method with queries including logical operators AND/OR

POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

URBE (Uddi Registry By Example)

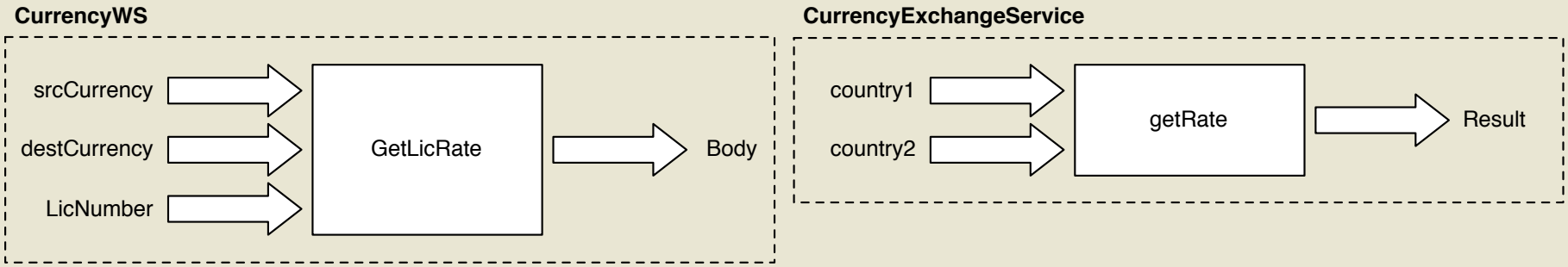
Lecture on "Web services retrieval"

Pierluigi Plebani

November, 4th, 2009

Main features of URBE

- Interface matching
- Semantic matching
- Quality driven matching
- So far:
  - we have studied the first and second points
  - we are going to validate the third point
- The main goal is: *retrieval for substitutability*



- Details in: P. Plebani and B. Pernici, URBE: Web service Retrieval based on Similarity Evaluation. IEEE Transaction on Knowledge and Data Engineering, ISSN: 1041-4347, to appear (now available on line)

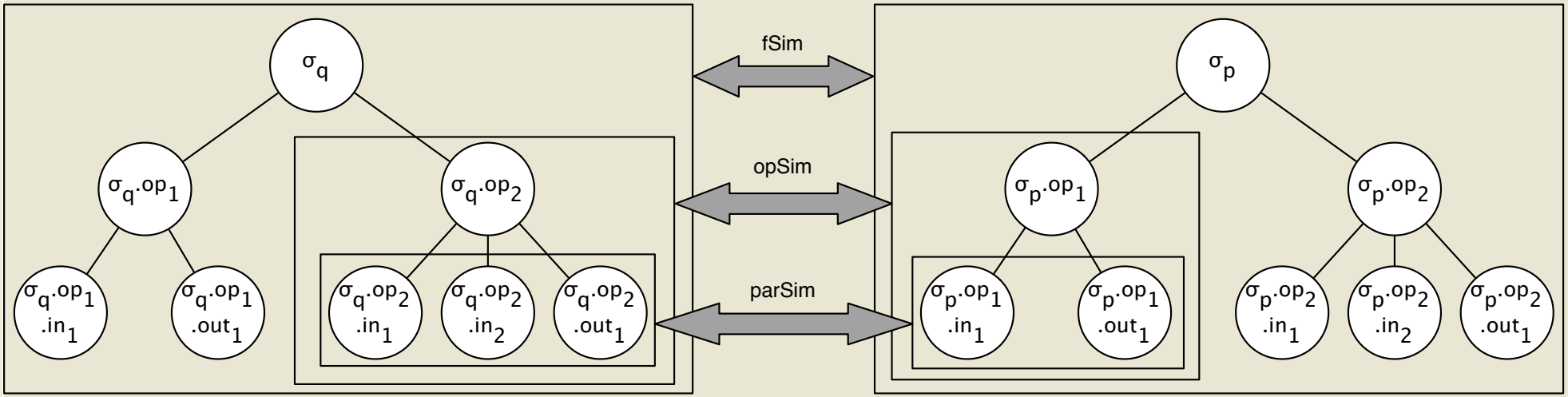
URBE in details

- Uddi Registry By Example
  - is compliant with UDDI (publishing, searching, data models)
  - performs content based query based
    - user submits a WSDL expressing the requirements
    - URBE returns a list of Web services close to the request
- Similarity function fSim is the core of URBE
  - semantic analysis
  - structural analysis

fSim properties

- Given:
  - σ_q as a query
  - σ_p as an available service
- fSim: $(\sigma_q, \sigma_p) \rightarrow [0..1]$
  - $fSim(\sigma_q, \sigma_q) = 1$
  - fSim is not symmetric
  - fSim relies on a linear programming model
- The value returned by fSim is used to rank services

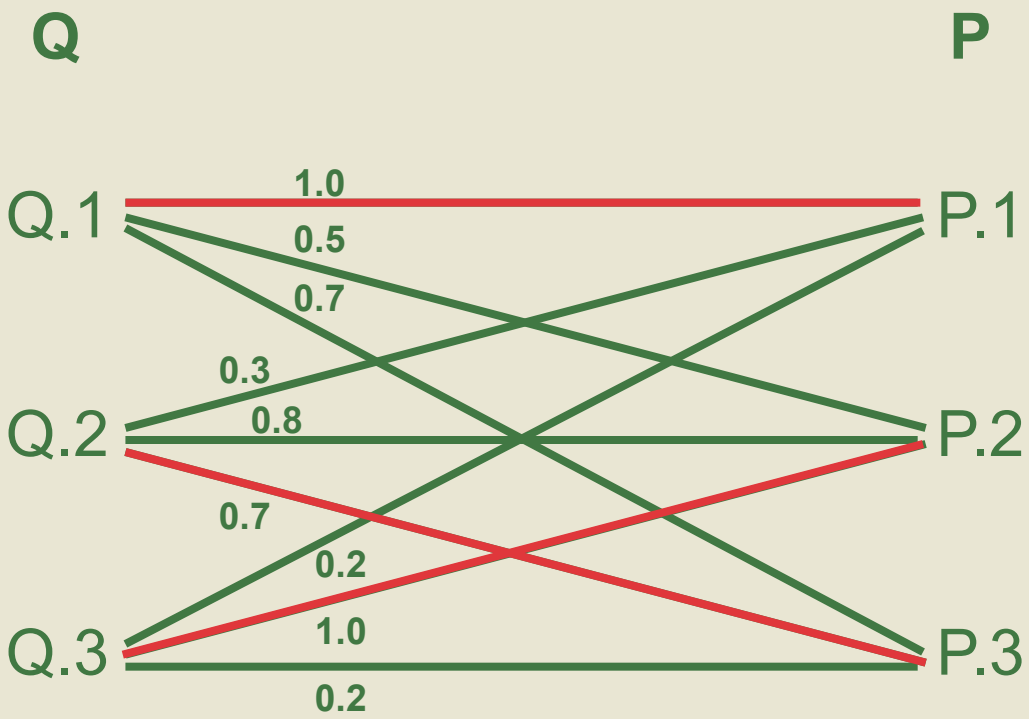
fSim components



- fSim compares the overall services description by using:
  - opSim that compares the operations description by using:
    - parSim that compares the parameters description

Adopted linear problem

- Assignment in bipartite graphs which compares:
  - terms, operation, services



$$maxSim(f, Q, P) =$$

$$\frac{1}{|Q|} \cdot max \sum_{i \in I}^{j \in J} f(q_i, p_j) \cdot x_{i,j}$$

$$\sum_{j \in J} x_{i,j} \leq 1 \quad \forall i \in I$$

$$\sum_{i \in I} x_{i,j} \leq 1 \quad \forall j \in J$$

$$I = [1..|Q|], \quad J = [1..|P|]$$

opt(sim(Q, P)) = (1.0 + 0.7+ 1.0) / 3 = 2.7 / 3 = 0.9

Names similarity

- We assume that the WSDL is automatically generated
- Names reflect coding conventions
- Stemming and tokenization are required before comparing names
  - getData, currencyExchange

Rule	Original term	Tokenized version
Case change	currencyExchange	currency, exchange
Case change	SendSMSTo	send, sms, to
Suffix numbers elimination	currency1	currency
Underscore separator	currency_exchange	currency, exchange

- Some terms have less meaning
  - body, result, parameters

DataType similarity 1/2

- Data types are grouped into categories [Stroulia and Yang]

Group	Simple and derived XSD Data Types
Integer group	integer, byte, short, long
Real group	float, double, decimal
String group	string, normalizedString
Date group	date, dateTime, duration, gDay, gMonth, gMonthDay, gYear, gYearMonth, time
Boolean group	boolean

- dataTypeSim is inversely proportional to the information loss if we cast from dt_q to dt_p

		dt_q				
	dataTypeSim	**Integer**	**Real**	**String**	**Date**	**Boolean**
dt_p	**Integer**	1.0	0.5	0.3	0.1	0.1
	Real	1.0	1.0	0.1	0.0	0.1
	String	0.7	0.7	1.0	0.8	0.3
	Date	0.1	0.0	0.1	1.0	0.0
	Boolean	0.1	0.0	0.1	0.0	1.0

DataType similarity 2/2

- Information loss is quantified according to the following reference scale

Information loss	Value
data types are totally incompatible	1.0
in some rare case casting does not produce information loss	0.7
information loss happens by casting	0.5
often casting does not produce information loss	0.3
data types are the same	0.0

Semantic extension

- Recall can be improved if SAWSDL description is available
- In this case name similarity is based on the annotations
  - Annotation refers to concept in the domain-specific ontology
  - Similarity evaluation depends on the nature of the annotation (it could be either class or property)
- Annotations similarity results
  - more precise than names similarity
  - faster to calculate

pathSim

- both a_q and a_p are classes, or
- both a_q and a_p are properties

$$pathSim(a_q, a_p) = \begin{cases} 0 & \text{if no subsumption path exists} \\ \dfrac{1}{(pathlength(a_q, a_p) + 1)} & \text{otherwise} \end{cases}$$

classPropSim and propClassSim

- a_q is a class
- a_p is a property

$$classPropSim(a_q, a_p) = \begin{cases} \dfrac{1}{\#\text{properties of } a_q} & \text{if } a_q \equiv \text{domain}(a_p) \\ 0 & \text{otherwise} \end{cases}$$

- a_q is a property
- a_p is a class

$$propClassSim(a_q, a_p) = \begin{cases} 1 & \text{if } a_q \equiv \text{domain}(a_p) \\ 0 & \text{otherwise} \end{cases}$$

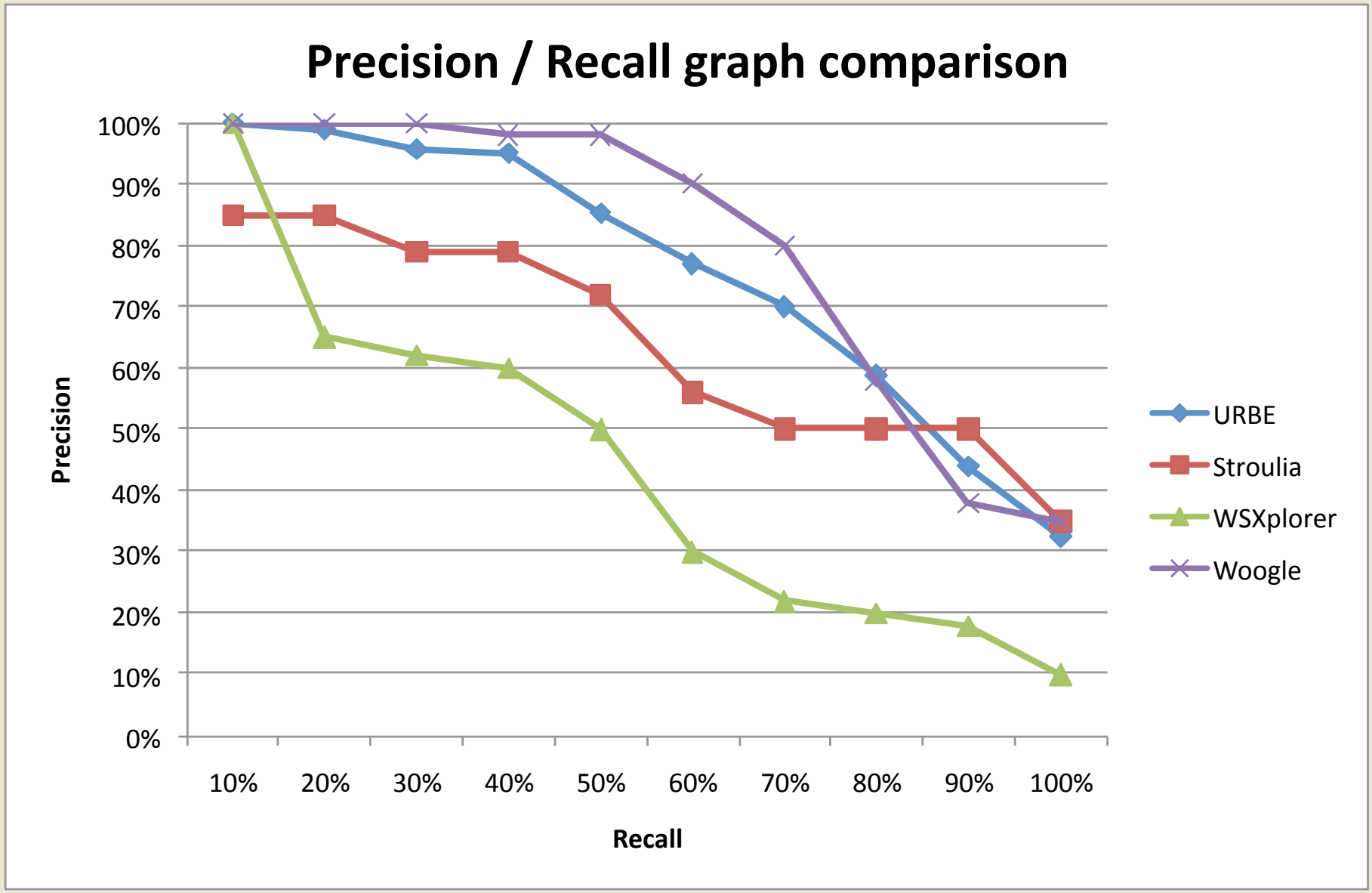
Benchmark

- Benchmark has been obtained from the OWL-S service retrieval test collection (OWL-S TC)
  - 570 Web services
  - 32 test queries
  - http://projects.semwebcentral.org/projects/owls-tc/
- Machine
  - IBM xSeries, 2 CPU Intel XEON 3GHz, 2 GByte RAM

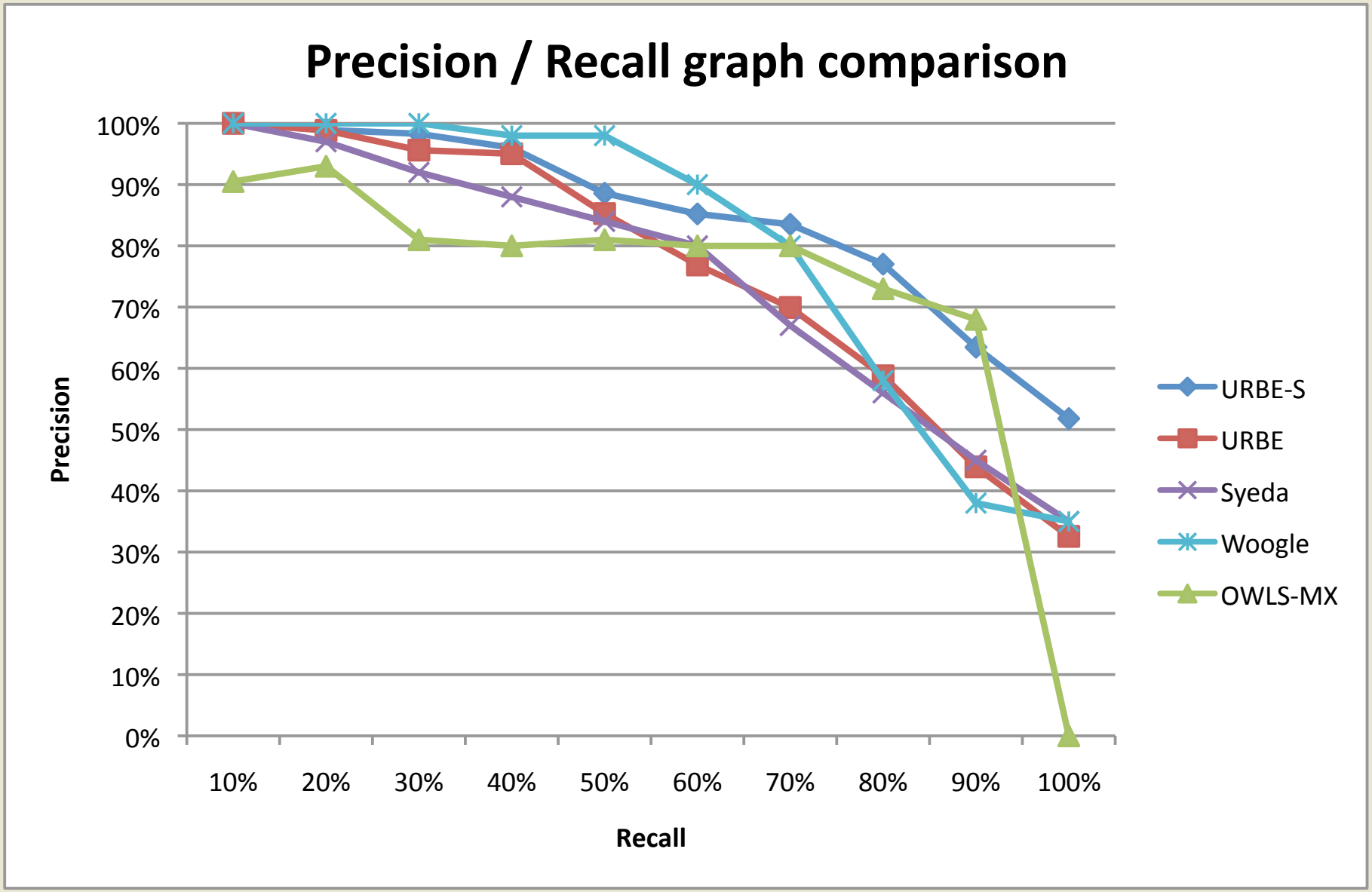
Evaluation parameters

- Precision = #relevant returned / #returned
- Recall = #relevant returned / #relevant in the corpus
- Top-5 = precision when 5 items are returned
- Top-10 = precision when 10 items are returned
- R-Precision = precision when the number of items returned corresponds to the number of relevant items
- Average Precision (AP) = precision calculated after a relevant item is returned

Results: precision/recall graph



Results: precision/recall graph



Semantic Service Selection Contest

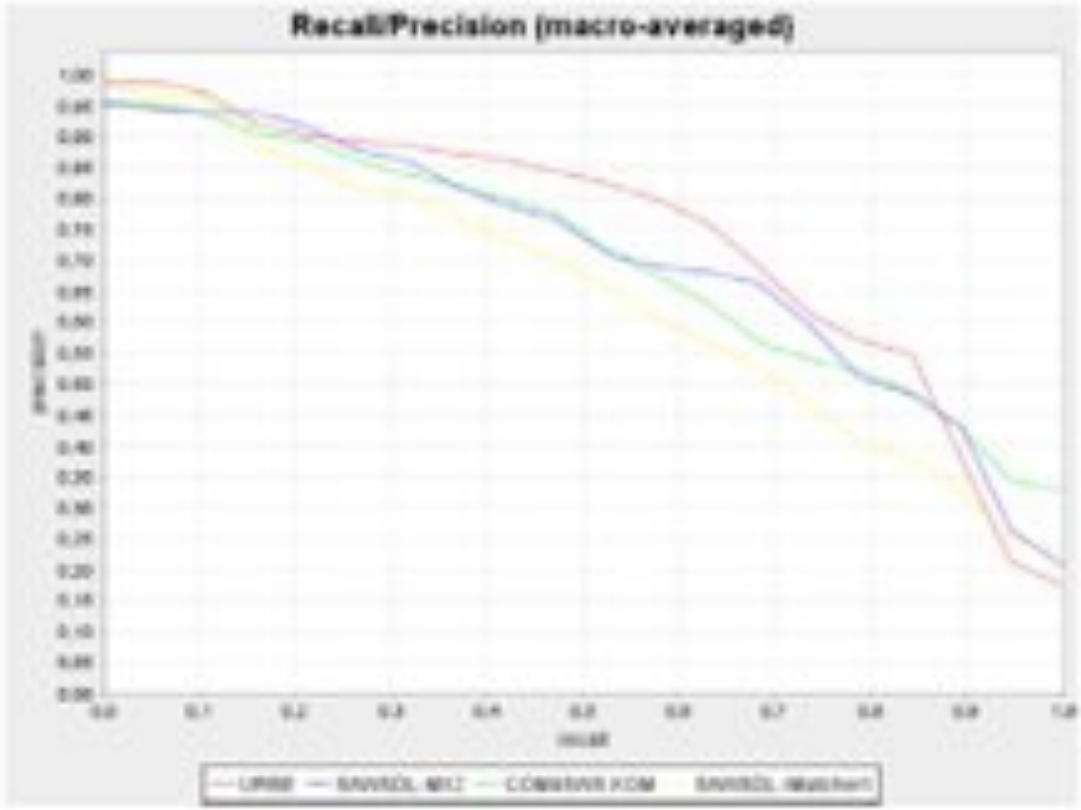
Average Precision:

1.	URBE	.727
2.	COM4SWS	.681*
3.	SAWSDL-MX2	.679
4.	SAWSDL-iMatcher3/1	.635

Avg Query Response Time (sec):

1.	SAWSDL-iMatcher3/1	.75
2.	COM4SWS	6.14**
3.	SAWSDL-MX2	7.9
4.	URBE	19.96

Macro-averaged Recall/Precision:



** W/o logic-based classification of service ontologies (building of matchmaker ontology) performed belatedly by COM4SWS at first query: else 62.29s ! * COM4SWS precision: Vairant w/o clustering (worse).

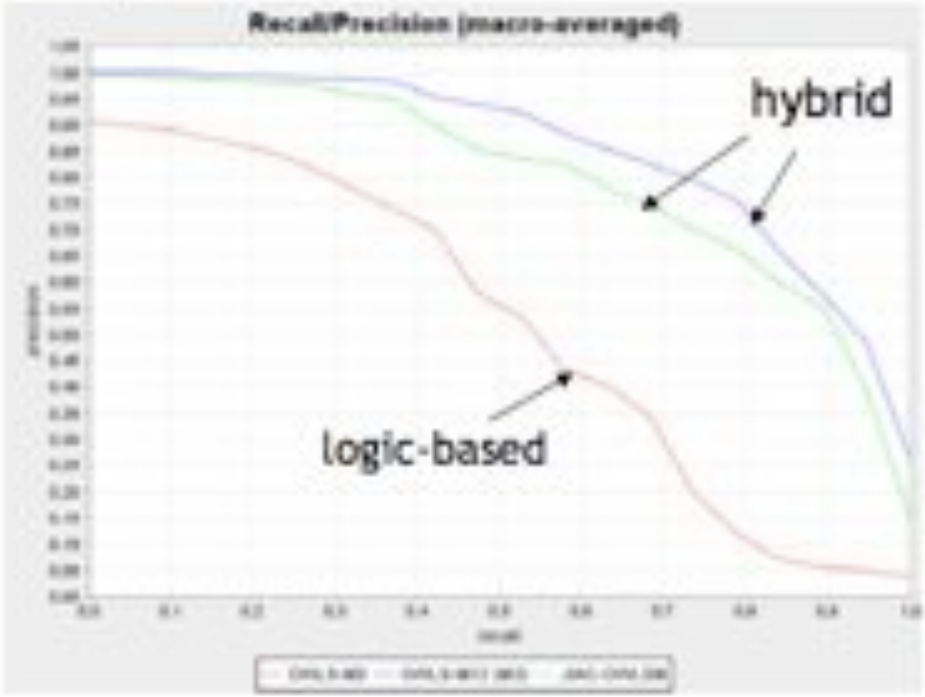
Source: M. Klusch, 3rd International Semantic Service Selection Contest - International Semantic Service Selection Contest, October 2009

Lesson learned in semantic selection

Logic-based vs. Hybrid semantic selection

1. *Integration of logic-based reasoning with text similarity may significantly improve precision at the cost of higher avg query response time.*

Example: Track 1 entries



		AP	AQRT
Logic-based	OWLS-M0	.74	2.66s
Hybrid	OWLS-MX2	.878	3.69s
	JIAC-OWLSM	.814	4.44s
Logic-based	SAWSDL-M0	.419	2.24s
Hybrid	SAWSDL-MX1	.556	2.83s

ALIVE variants (2 logic-based, 1 hybrid) with insignificant differences in precision.

Source: M. Klusch, 3rd International Semantic Service Selection Contest - International Semantic Service Selection Contest, October 2009

POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Concluding remarks

Lecture on "Web services retrieval"

Pierluigi Plebani

November, 4th, 2009

My standpoint

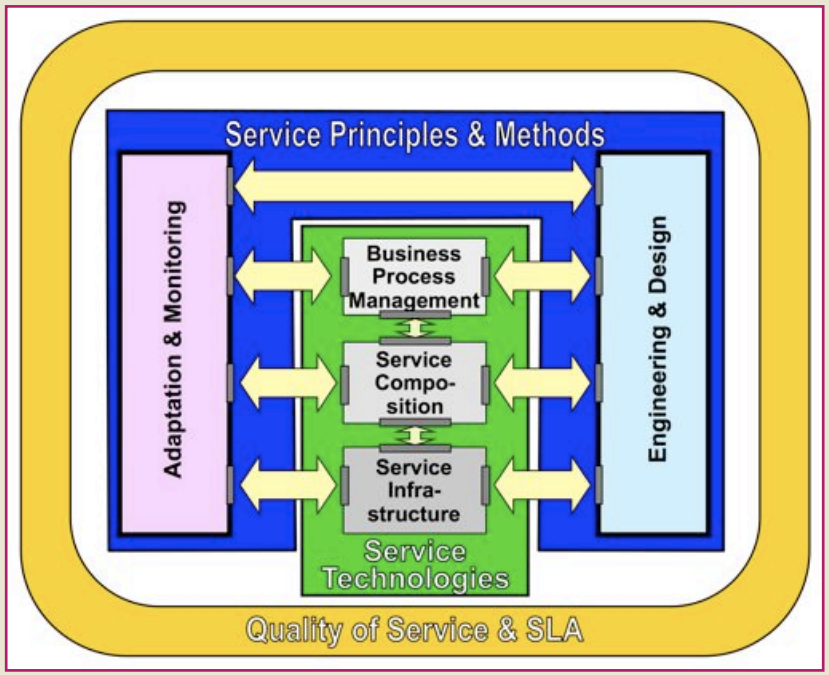
- Quality and negotiation matchmaking represent, at this stage, the biggest open issues
- About the functional matchmaking is matter of computation time
- Web service registry managing should be deeply investigated as well
- Semantic based approaches suffer of the need of services semantically described
- Web services retrieval must be, first of all, *usable*!
  - Holy grail: “I would like a service able to...”

Acknoledgments

S-CUBE

The Software Services and Systems Network

Information Society Technologies



S-CUBE service vision

S-CUBE focuses on the new-generation of services mainly on:

- Adaptive and Flexible execution
- Human Computing Interaction
- Context awareness
- Quality and SLA definition

These aspects will be investigated from BPM down to the infrastructural perspective.

Service engineering and governance hold a key role in the S-CUBE vision.

http://www.s-cube-network.eu/

SEVENTH FRAMEWORK PROGRAMME

S-Cube is funded by the European Community's Seventh Framework Programme FP7/2007-2013 under Objective 1.2 'Services and Software Architectures, Infrastructures and Engineering'

Contacts

Pierluigi PLEBANI

Dipartimento di Elettronica ed Informazione

Politecnico di Milano

Via Ponzio 34/5

20133 Milano - Italy

plebani@elet.polimi.it

http://home.dei.polimi.it/plebani