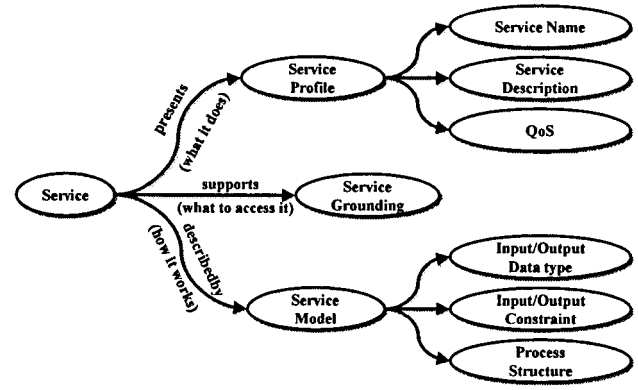
Algorithms for Intelligent Web Service Discovery considering Process Information and QoS

Jeong-Youn Yu

Dept. Computer Engineering
Chungnam National University

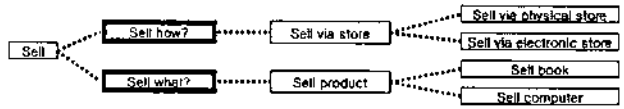
OWL-S Ontology

- Build on the OWL (Ontology Web Language)

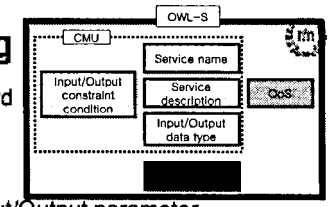


TM-S

- Topic Maps for Services
- Define Knowledge Model for Service using XTM(XML Topic Maps)
- Supports the strong points of OWL-S and MIT Process Handbook, but also complements their weak points
  - Service Profile Ontology : OWL-S
  - Service Model Ontology : OWL-S
  - Service/Process View Ontology : MIT Process Handbook

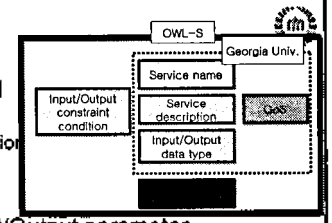


CMU's MatchMaking



- Matching based on Keyword
  - Context Matching
  - Profile comparison(TF-IDF)
- Data Type Matching of Input/Output parameter
  - name : using word distance DB
    - (computer,notebook), (computer,book)
  - Data type : using subtyping inferece rule
- Constraint Matching of Input/Output parameter
  - $(Pre_S \Rightarrow Pre_T) \wedge (Post_T \Rightarrow Post_S)$
  - Θ-subsumption relationship
- No Process Stucture Matching and QoS Matching

Georgia University's MatchMaking

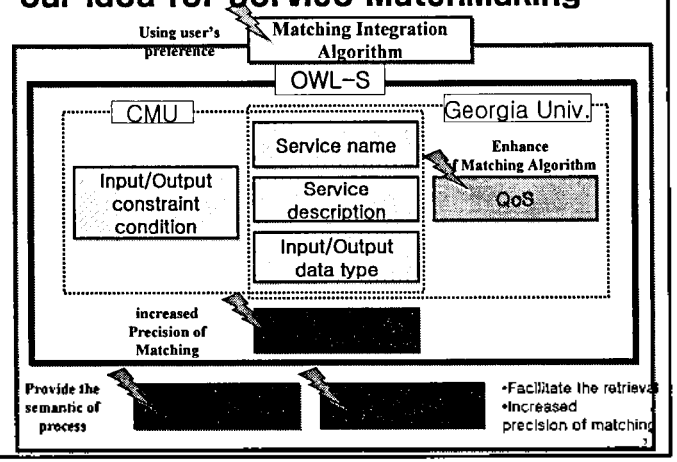


- Matching based on Keyword
  - Matching of Service name
  - Matching of Service Description
  - TF-IDF
- Data Type Matching of Input/Output parameter
  - Evaluate the similarity of data type
- QoS Matching
  - time, cost, reliability
  - Evaluate the similarity between Query QoS and Service QoS
- No Process Structure Matching and constraint Matching of Input/Output parameter

Problem of Semantic Web Service MatchMaking

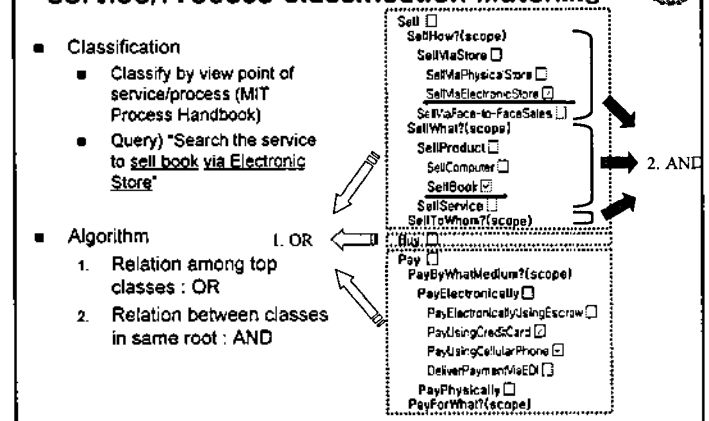
- Matching based on keyword
- Not support the constraint Matching of Input/Output parameter
  - "Search the service which the user can order the book, if and when it have the valid cadit card(precondition)
- Not support the Matching for Process Structure
  - "Search the book purchase service, where the user expects to login and pay(sequence)
- Not support the QoS Matching
  - Progressing the research in the QoS

Our idea for Service MatchMaking

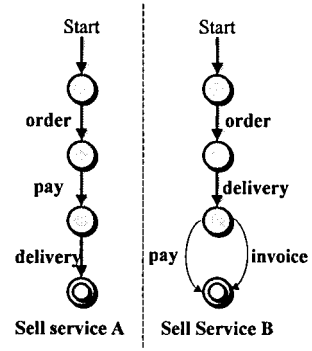


Service/Process Classification Matching

- Classification
  - Classify by view point of service/process (MIT Process Handbook)
  - Query) "Search the service to sell book via Electronic Store"
- Algorithm
  1. Relation among top classes : OR
  2. Relation between classes in same root : AND



Process Structure Matching



- Two sell service
- However, there are different services
- Require the Matching for Process Structure

9

Process Matching Algorithm

- Sequence Structure

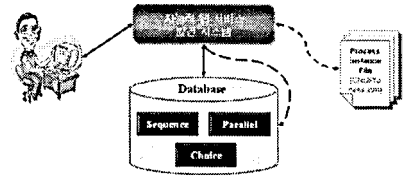
  - Sequence
  - Query) "Search the book purchase service, where the user expects to login and pay(sequence)"
- Parallel Structure

  - Split, split+Join
  - Query) "Search the book purchase service processing with delivery and stock update"
- Choice Structure

  - If-then-else
  - Query) "Search the book purchase service, where the user expects to credit card payment or account payment"

10

Process Matching Algorithm



- Sequence Structure

RootProcess	Process	NextProcess
CNUBuyBookProcess	Login	Payment

- Parallel Structure

RootProcess	ParallelProcess
CNUBuyBookProcess	Delivery
CNUBuyBookProcess	UpdateStock

- Choice Structure

RootProcess	ThenProcess	ElseProcess
CNUBuyBookProcess	PaymentCreditCard	PaymentOnlineBanking

11

Process Pattern Matching

- In ebXML
  - Business Transaction
  - Request/Confirm
  - Request/Response
  - Query/Response
  - Notification
  - Information Distribution
- When the user search the service aspect of business modeling
  - To provide an established semantics of process interactions
  - Query) "Find the service designed Business Transaction pattern ."

12

QoS Matching in Georgia University

- QoS components
  - time, cost, reliability : minimum, average, maximum
- Evaluate the similarity
  - Dcd : distance of two components (query QoS and service QoS)

$$dcd_{min}(ST,SO,dim) = 1 - \frac{|min(SO.qos(dim))-min(ST.qos(dim))|}{min(ST.qos(dim))}$$

  - QoSdimD : square root of min, aver, max dcd

$$QoSdimD(ST,SO,dim) = \sqrt[3]{dcd_{min}(ST,SO,dim)*dcd_{avg}(ST,SO,dim)*dcd_{max}(ST,SO,dim)}$$

  - final result : Square root of QoSdimD in All QoS

$$OpSimilarity(ST,SO) = \sqrt[3]{QoSdimD(ST,SO,time)*QoSdimD(ST,SO,cost)*QoSdimD(ST,SO,reliability)}$$

Problem 1 of QoS

1. It cannot a good service, even so the service have high QoS similarity

Reliability	Min	Avg	Max
Query	90	94	99
S1	90	93	99
S2	90	98	99
Distance between Query and S1	0	1	0
Distance between Query and S2	0	4	0
The similarity between S1 and QoS	0.995		
The similarity between S2 and QoS	0.978		

Problem 2 of QoS

2. A specific component can lead the whole similarity

Cost	Min	Avg	Max
Query	200	250	300
S1	200	240	300
S2	200	290	300
Distance between Query and S1	0	10	0
Distance between Query and S2	0	50	0

reliability	Min	Avg	Max
Query	90	94	99
S1	87	96	98
S2	90	94	99
Distance between Query and S1	3	2	1
Distance between Query and S2	0	0	0

Whole similarity

S1 similarity	0.973
S1 similarity	0.957

QoS Matching Algorithm

- Components
  - Cost, Time, Reliability, Accuracy, Availability, Last Update Time
- suppose
  - Select the average of QoS components
  - The more a whole QoS is low, the more preference and quality is high
- Algorithm
  - Select the service in the scope of query
  - Mediate the QoS value
  - Evaluate the standard value
  - Evaluate the whole QoS value
  - Sorting and decision

Mediate and evaluate of QoS value

- Mediate the QoS value
  - Complement the problem 1
  - The more a whole QoS is low, the more preference and quality is high : cost, time
  - The more a whole QoS is high, the more preference and quality is high
    - Reliability, Accuracy, Availability(max 100% - QoS value)
    - Last update time (current time - QoS value)
- Evaluate of the Standard value
  - Complement the problem 2
  - know the position where the service's QoS is in whole QoS

$$T(S1.cost) = \left[\frac{S1.cost - m}{s} \right] \times 10 + 50$$

$$s = \sqrt{\frac{(S1.cost - m)^2 + (S2.cost - m)^2 + \cdots + (Sn.cost - m)^2}{n}}$$

m = average, n = number of service

TotalScore(S) =
T(S,cost) + T(S,time) + T(S,reliability) + T(S,accuracy) + T(S,availability) + T(S,lastUpdateTime)

17

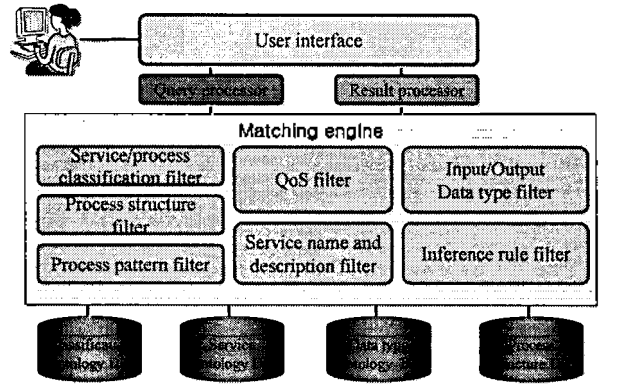
Example of QoS Matching

- Query : find the services what have the cost of $600 or under, the reliability of 85% and over

	S1	S2	S3	S4	S5	S6	S7
Cost	250	250	300	350	550	570	650
Reliability	95	97	90	90	85	99	80
Standard value of cost	40.35	40.35	44.11	47.87	62.91	64.41	
Standard value of reliability	39.74	39.51	40.33	40.33	40.91	39.27	
whole QoS value	80.09	79.86	84.44	88.20	103.82	103.68	
QoS rank	2	1	3	4	6	5	
QoS rank in Georgia university	5	6	4	3	1	2	

18

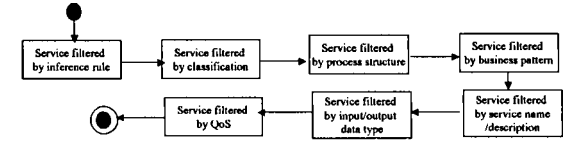
System Architecture



19

Query Process Algorithm

- Efficient query process
  - Select Matching filters which Matching speed is fast and efficient filter is high
  - The next, the Input/output data type filter and QoS filter that process slow



- Integration and sorting
- Considering the user's preference
- No the similarity
  - Service/process classification filter, inference rule filter, process structure filter, transaction filter
- Have the similarity value
  - Service name/description filter, Input/Output datatype filter, QoS filter

20

Integration algorithm

- The scope of similarity
  - Name and description filter
    - S_{name}, S_{desc} : 0~100
  - Input/Output datatype filter
    - S_{input}, S_{output} : 0~1
  - QoS filter
    - S_{qos} : 0~600
- Control the similarity value to setting the value 0 to 100
  - The more QoS value is high, the ranking is high
  - N_{name}, N_{desc} : (S_{name}, S_{desc}) × 1
  - N_{input}, N_{output} : (S_{input}, S_{output}) × 100
  - N_{qos} : (600 – S_{qos}) × 1/5
- Multiply the preference value
  - W_{qos} : N_{qos} × 3
- Add similarity values of whole filter
  - Total = W_{name} + W_{desc} + W_{input} + W_{output} + W_{qos}

21

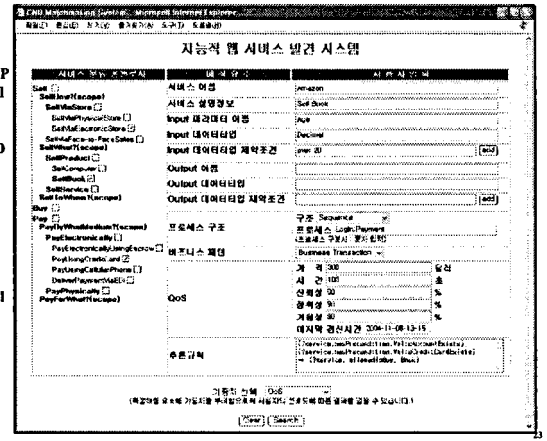
Example of Integration algorithm

		S1	S2	S3	S4	S5
S_{name} × 1 Service name ×1	S_{name}	99	90	85	70	50
	N_{name}	99	90	85	70	50
	W_{name}	99	90	85	70	50
S_{input} × 100 Input ×1	S_{input}	0.75	0.3	0.5	0.7	0.8
	N_{input}	75	30	50	70	80
	W_{input}	75	30	50	70	80
(600-S_{qos}) × 1/5 QoS ×3	S_{qos}	400	150	300	250	500
	N_{qos}	33.33	75	50	58.33	16.67
	W_{qos}	99.99	225	150	174.99	50.01
Total	Total	273.99	345	285	314.99	180.01
rank		4	1	3	2	5

22

User Interface

- OS : Windows XP Professional
- Language : JAVA(J2SDK 1.4.1_02)
- Environment
  - Jakarta Tomcat 4.1.29
  - OKS Professional 2.0.2
  - Jena 2.1



23

References

[1] Andreas Wombacher, Peter Fankhauser, Bendick Mahleko, Erich Neuhold, "Matchmaking for Business Processes," Proceedings of the IEEE International Conference on E-Commerce (CEC'03), 7-11, 2003

[2] Cardoso, J. and A. Sheth, "Semantic e-Workflow Composition," Journal of Intelligent Information Systems (JIIS), Vol. 21(3), pp.191 -225, 2003

[3] HP, "Jena," http //jena.sourceforge net/ index html, 2004

[4] Katia Sycara, Seth Widoff, Matthias Klusch and Jianguo Lu, "LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace," Autonomous Agents and Multi-Agent Systems, Vol. 5, pp.173–203, 2002

[5] OASIS, UN/CEFACT, "Patterns for Business to Business Transactions," http //193.113.58.84/twiki/pub/Papers/WebHome/Business Patterns pdf, 2002

[6] OASIS, "Technical Note, Representing Web Services Quality of Service Information in UDDI," 2004

[7] Ontopia, "The Ontopia Knowledge Suite", http://www.ontopia. net/ontopia/texts/product-wp.html, April, 2003

[8] Salton, G, "Automatic Text Processing: The Transformation," Analysis and Retrieval of Information by Computer, Addison-Wesley, 1988

[9] Steve Pepper, "The TAO of Topic Maps", XML EUROPE 2000

[10] Thomas W. Malone, "Tools for inventing organizations. Toward a handbook of organizational processes", ManagementScience Vol 45. No.3, March, 1999

[11] W3C, "OWL-S: Semantic Markup for Web Services,"http://www daml. org/ services/owl-s/1 1B/owl-s pdf, 2004

24