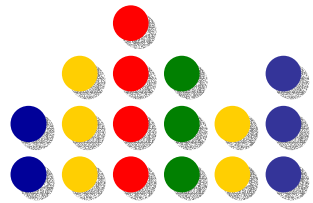
Semantic Web Services and Processes: Semantic Composition and Quality of Service

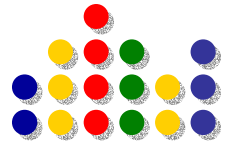


Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4] , Dieter Fensel[3]
[1]LSDIS Lab, Computer Science, University of Georgia
[2]Oracle Corporation
[3] Universität Innsbruck
[4] Semagix, Inc

Tutorial at Federated Conferences
On the Move to Meaningful Internet Computing and Ubiquitous Computer 2002, Irvine CA, October 2002.

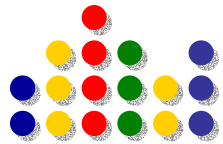
Web Resource for this tutorial:
http://lsdis.cs.uga.edu/lib/presentations/SWSP-tutorial/resource.htm

Big Challenges



- Semantics are critical to support the next generation of the Web.
- The important contribution of the “Semantic Web”, vis-à-vis the current Web, is the ability to represent and process descriptions of every resource on the Web.
- A resource description, informally called its “semantics”, includes that information about the resource that can be used by machines - not just for display purposes, but for using it in various applications.

The Vision



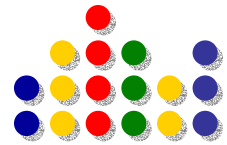
500 million user
more than 1 billion pages

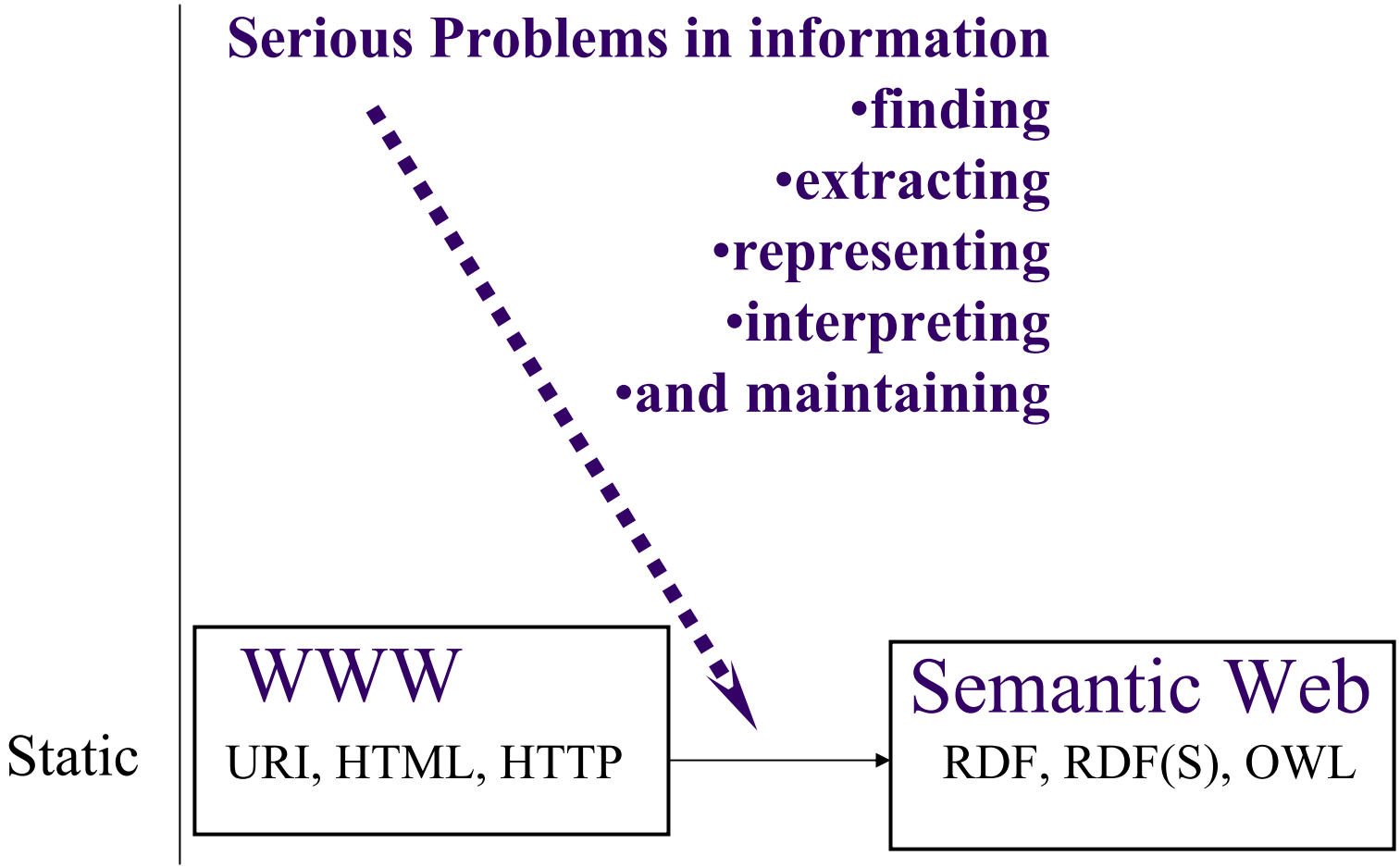
WWW

Static

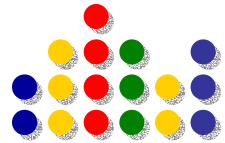
URI, HTML, HTTP

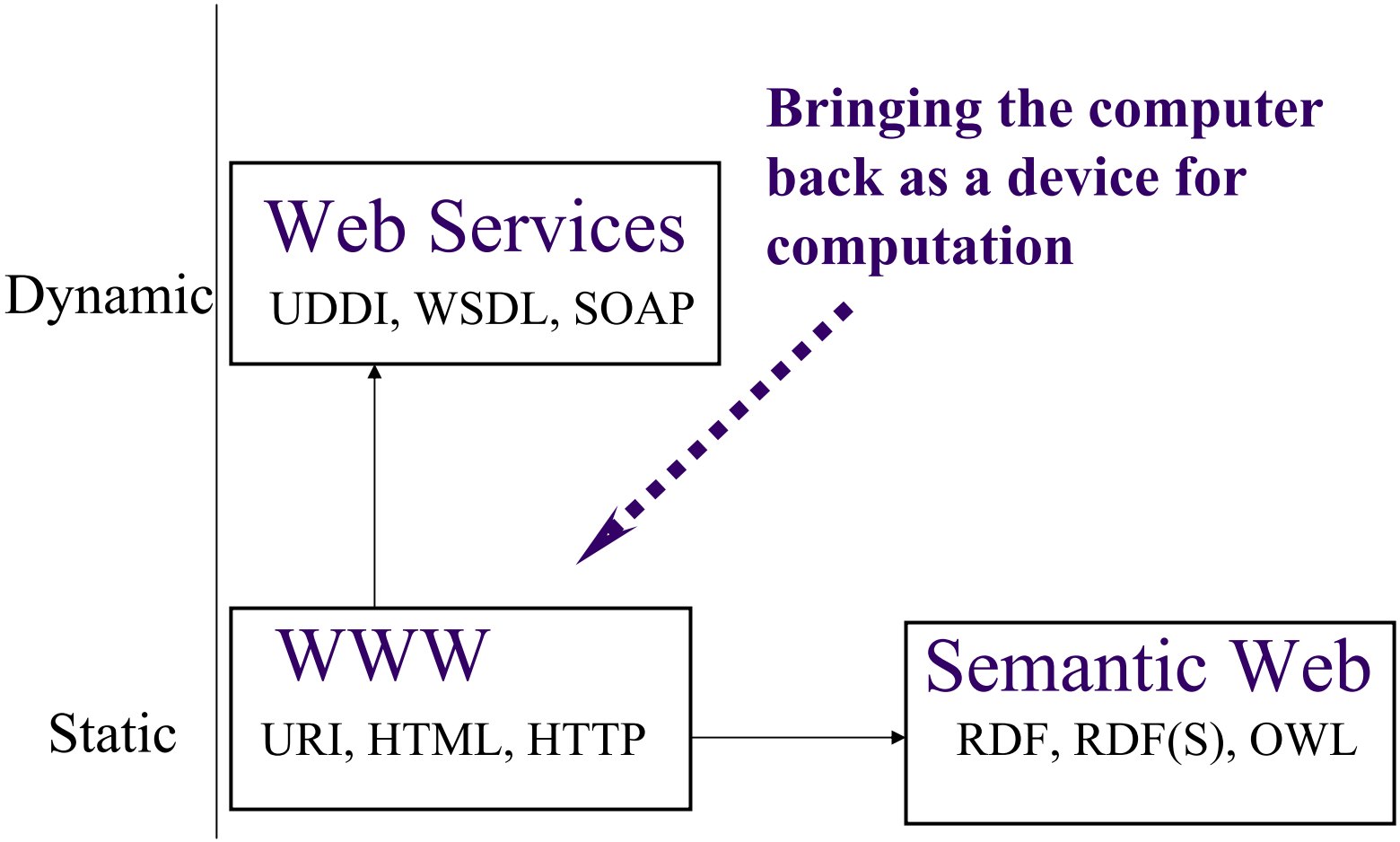
The Vision




Serious Problems in information
•finding
•extracting
•representing
•interpreting
•and maintaining
Static
WWW
URI, HTML, HTTP
Semantic Web
RDF, RDF(S), OWL

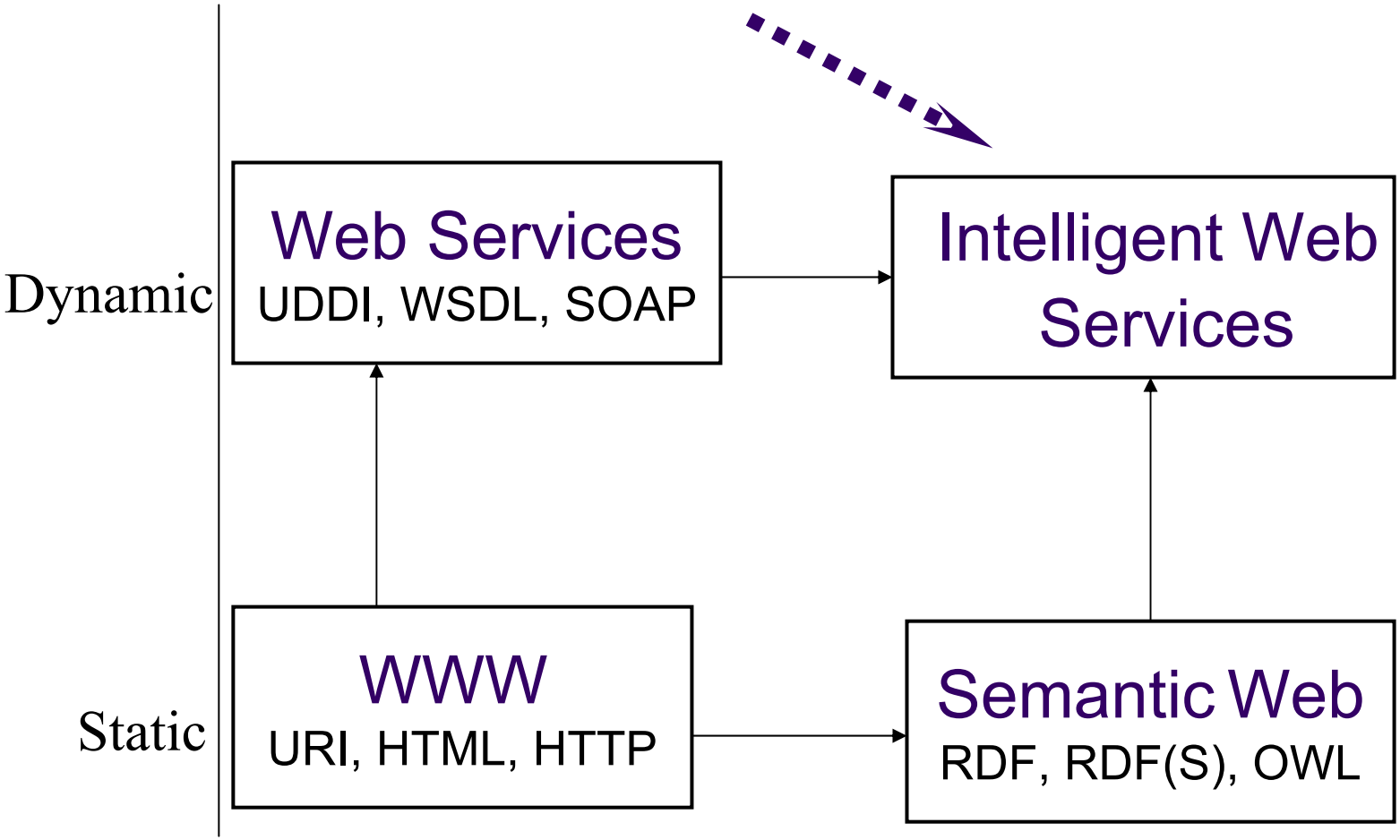
Current Affairs



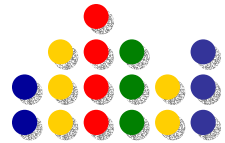


The Vision

Bringing the web to its full potential



Components of a Solution

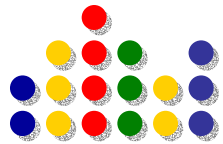


This tutorial focuses on two issues:

Semantic Web Services are Web Services with a formal description (semantics) that can enable a better discovery, selection, composition, monitoring, and interoperability.

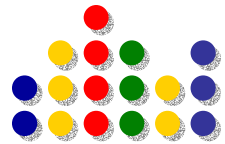
Processes are next steps to carrying out core business activities, such as e-commerce and e-services, and are created from the composition of Web Services or other components.

Our Focus



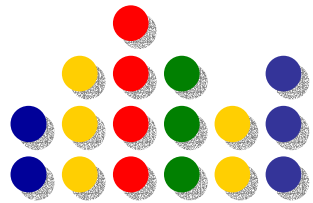
- In a nutshell, this tutorial is about associating semantics to Web Services, and exploiting it in process composition
  - Frameworks, Standards
- Functional perspective takes form of process composition involving Web Service Discovery, addressing semantic heterogeneity handling.
- Operational perspective takes form of the research on QoS specification for Web Services and Processes.

Outline



- Web Services
  - A Working Technology
  - Truth & Vision
- Web Service Composition
  - Introduction
  - Discovery and Integration
  - Quality of Service
- Conclusions

Web Services
A Working Technology



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

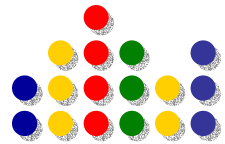
LSDIS Lab, Computer Science, University of Georgia

[2]Oracle Corporation

[3] Universität Innsbruck

[4] Semagix, Inc

Definition

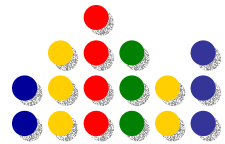


"Web services are a new breed of Web application. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes. …

Once a Web service is deployed, other applications (and other Web services) can discover and invoke the deployed service."

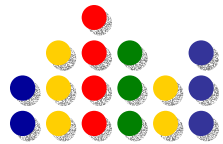
IBM web service tutorial

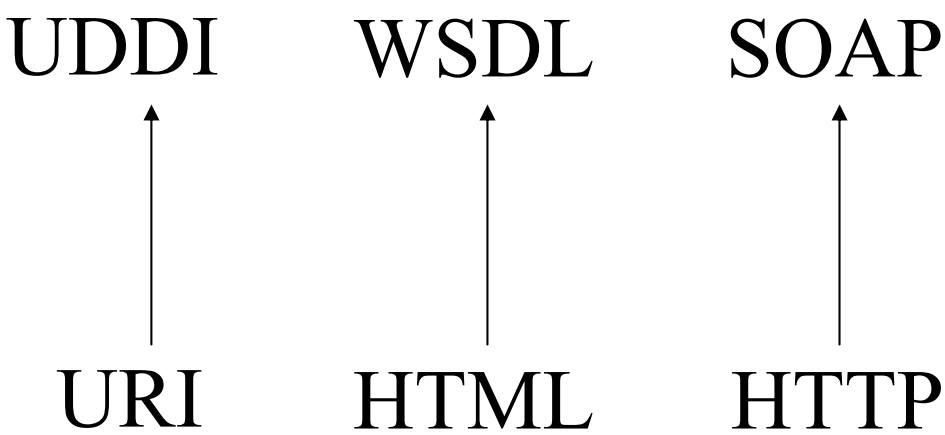
What are Web-Services ?



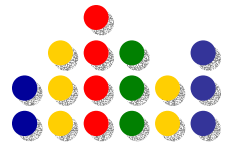
- Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways.
- The key to Web Services is on-the-fly software creation through the use of loosely coupled, reusable software components.
- Software can be delivered and paid for as streams of services as opposed to packaged products.
- Business services can be completely decentralized and distributed over the Internet.
- The dynamic enterprise and dynamic value chains become achievable and may be even mandatory.

State of the Art





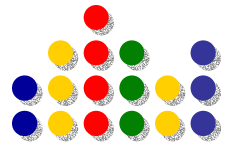
Attributes of Web-Services



- Web-based Protocols : Web-services based on HTTP are designed to work over the public internet. The use of HTTP for transport means these protocols can traverse firewalls, and can work in a heterogeneous environment.
- Interoperability : SOAP defines a common standard that allows differing systems to interoperate. E.g., the tooling allows Visual Basic clients to access Java server components and vice versa.
- XML-based : The Extensible Markup Language is a standard framework for creating machine-readable documents.

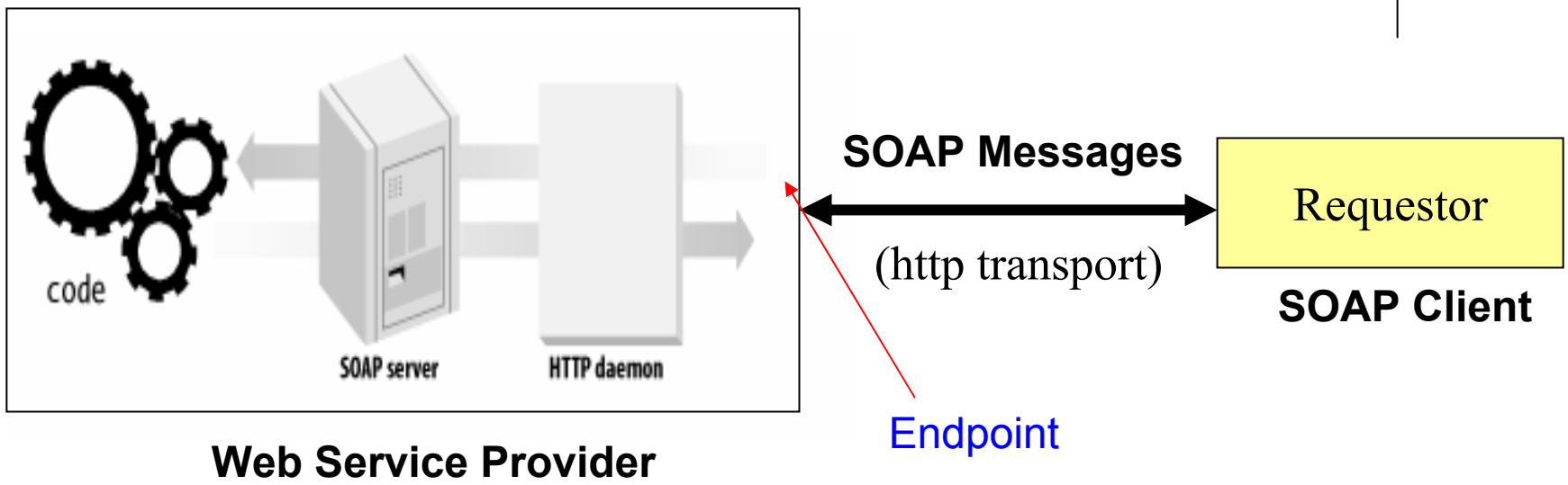
Fremantle et al. 2002, Enterprise Services, CACM. Oct

State of the Art



- **UDDI** provides a mechanism for clients to find web services. A UDDI registry is similar to a CORBA trader, or it can be thought of as a DNS for business applications.
- **WSDL** defines services as collections of network endpoints or *ports*. A port is defined by associating a network address with a binding; a collection of ports define a service.
- **SOAP** is a message layout specification that defines a uniform way of passing XML-encoded data. It also defines a way to bind to HTTP as the underlying communication protocol. SOAP is basically a technology to allow for “RPC *over the web*”.

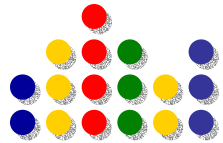
Web Service : How They Work?

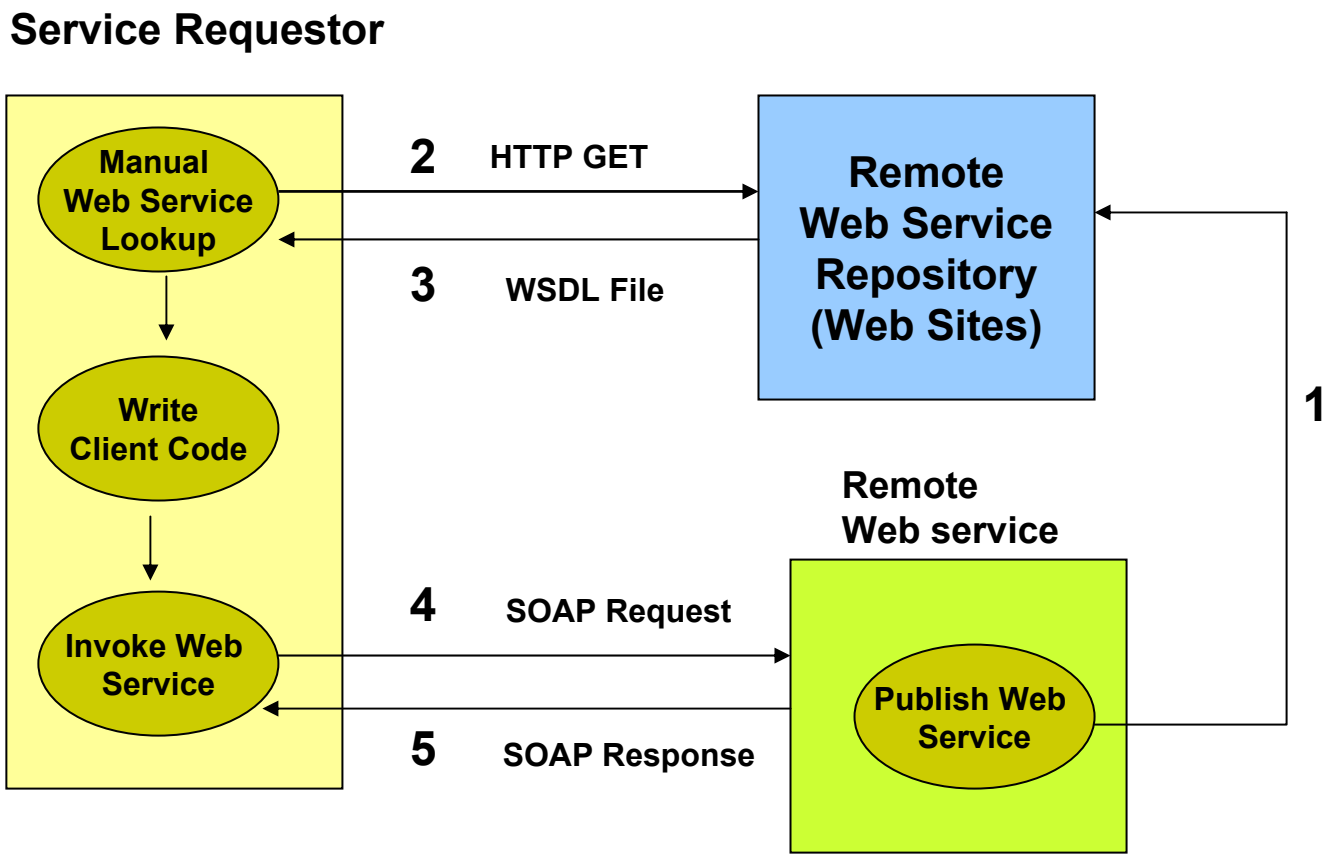


- Components required
  - Software which needs to be exposed as a Web service
  - A SOAP Server (Apache Axis, SOAP::Lite, etc.)
  - HTTP Server (if HTTP is used as the transport level protocol)
  - SOAP Client (Apache Axis, SOAP::Lite etc.)

From S. Chandrasekaran's Talk

Simple Web Service Invocation

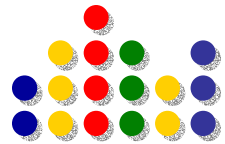




WSDL - Web Service Description
SOAP - Web Service Message Protocol

From S. Chandrasekaran's Talk

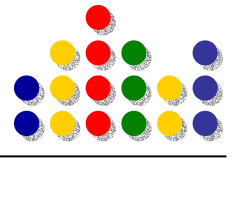
Web Service Description



- Why describe Web services?
  - A service requestor needs to analyze a service for his requirements
  - A Web service needs to provide the following information
    - the operations it supports
    - the transport and messaging protocols on which it supports those operations
    - the network endpoint of the Web service
- Languages such as WSDL, DAML-S, RDF can be used for describing Web services
  - WSDL – describes the syntactic information of a service
  - DAML-S and RDF – describe the syntactic as well as the semantic information

From S. Chandrasekaran's Talk

Web Service Description (WSDL)



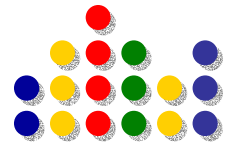
```
<definitions>
    <types>
          definition of types..
    </types>

    <message>
          definition of messages...
    </message>

    <portType>
          <operation> ..... </operation>
          <operation> ..... </operation>
    </portType>

    <binding>
          definition of binding....
    </binding>

    <service>
          <port>....</port>
          <port>....</port>
    </service>
</definitions>
```

Abstract Description

Concrete Description

From S. Chandrasekaran's Talk

Web Service Message Protocol - SOAP



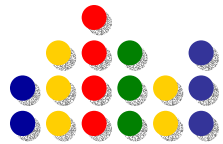
- SOAP is an XML Messaging Protocol
  - that allows software running on disparate operating systems, running in different environments to make Remote Procedure Calls (RPC).

```
<SOAP:Envelope
   xmlns:SOAP='http://schemas.xmlsoap.org/soap/envelope/'
   SOAP:encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'
   xmlns:v='http://www.topxml.com/soapworkshop/'>
      <SOAP:Header>
            <v:From SOAP:mustUnderstand='1'>
                  cdix@soapworkshop.com
            </v:From>
      </SOAP:Header>
      <SOAP:Body>
            <v:DoCreditCheck>
                  <ssn>123-456-7890</ssn>
            </v:DoCreditCheck>
      </SOAP:Body>
</SOAP:Envelope>
```

Header

Body

UDDI (Universal Description, Discovery and Integration)



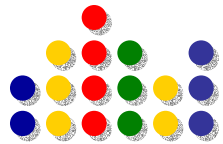
- UDDI serves as a "Business and services" registry and are essential for dynamic usage of Web services

- UDDI APIs
  - Publication API - Authenticated set of operations that allow organizations to publish businesses, services, service type specifications

  - Inquiry API - Non authenticated public set of operations that allows users to extract information out of the UDDI registry.

From S. Chandrasekaran's Talk

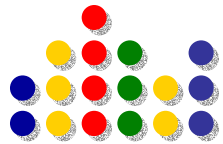
UDDI



- UDDI classifies businesses and services according to standard taxonomies
- Why Classification ?
  - Searches based on keywords alone, could return a large set of hits for a particular search
  - Classification of services and businesses allows to perform better searches
- Registry Data
  - White Pages
  - Yellow Pages
  - Green Pages
  - ServiceType Registrations

From S. Chandrasekaran's Talk

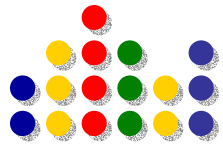
UDDI



- **White Pages**
  - contains business name, text description, contact info and other related info.
- **Yellow Pages**
  - contains classification information about the business entity and types of the services the entity offers.
    - e.g. a business entity could have itself classified as a sports equipment manufacturer and also as a skateboard manufacturer.
- **Green Pages**
  - contains information about how to invoke the offered services.
    - If a business entity were to offer its catalog online, its Green pages entry would have a reference to its catalog URL

From S. Chandrasekaran's Talk

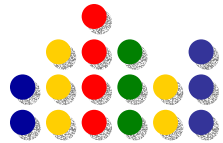
UDDI



- Service Types
  - Reusable, abstract definitions of services (~ abstract part of WSDL) that are defined by industry groups and standard bodies.
  - These reusable abstractions are referred to as “Technology Models”
  - The UDDI data structure corresponding to this is called “TModels”
- TModels
  - Any abstract concept can be registered within UDDI as a TModel.
    - e.g. If you define a new WSDL port type, you can define a TModel that represents the port type within the UDDI

From S. Chandrasekaran’s Talk

How UDDI Works ?

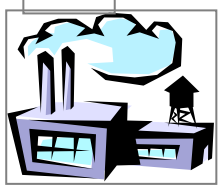


1. 

SW companies, standards bodies, and programmers populate the registry with descriptions of different types of services

4. 

Marketplaces, search engines, and business apps query the registry to discover services at other companies

2. 

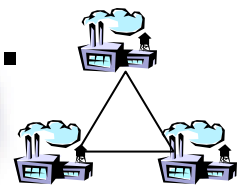
Businesses populate the registry with descriptions of the services they support

UDDI Business Registry

Business Registrations

Service Type Registrations

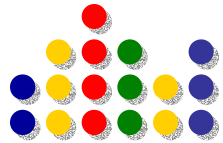
3. UBR assigns a programmatically unique identifier to each service and business registration

5. 

Business uses this data to facilitate easier integration with each other over the Web

Source : http://www.uddi.org/pubs/UDDI_Overview_Presentation.ppt

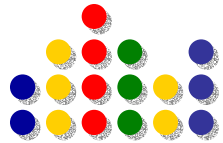
Services Aspect of Web-Services



- Modular : Service Components are useful in themselves, reusable, and it is possible to compose them into larger components.
- Available : Services are available to systems that wish to use them. Services must be exposed outside of the particular paradigm or system they are available in.
- Described : Services have a machine-readable description that can be used to identify the interface of the service, and its location and access information.
- Implementation-independent : The service interface must be available in a way that is independent of the ultimate implementation.
- Published : Service descriptions are made available in a repository where users can find the service and use the description to access the service.

Fremantle et al. 2002, Enterprise Services , CACM. Oct

Why Web services?

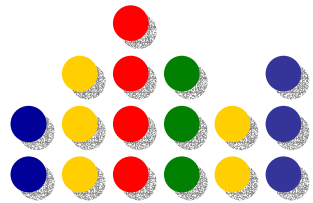


Feature	CORBA	Web Services
Data Model	Object Model	SOAP Message exchange model
Client Server Coupling	Tight Coupling	Loose Coupling
Parameter Passing	Pass by reference/value	Pass by value only
Type Checking	1.Static + Runtime type checking (Regular) 2. Runtime type checking only (DII)	RunTime type checking only
State	Stateful	1. Stateless, Uncorrelated (Web Services) 2. Stateful (Web Process)
Firewall Traversal	Work in Progress	Uses HTTP port 80
Service Discovery	CORBA naming/trading Service	UDDI
Communication Mode	1-way, 2-way sync 2-way async	2-way sync (Web Services) 1-way, 2-way sync, 2-way async (Web Process)

Gokhale et al, Reinventing the Wheel ? CORBA vs Web-services

Web Services

Truth & Vision



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

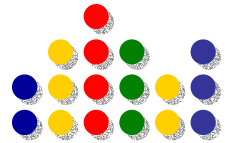
LSDIS Lab, Computer Science, University of Georgia

[2]Oracle Corporation

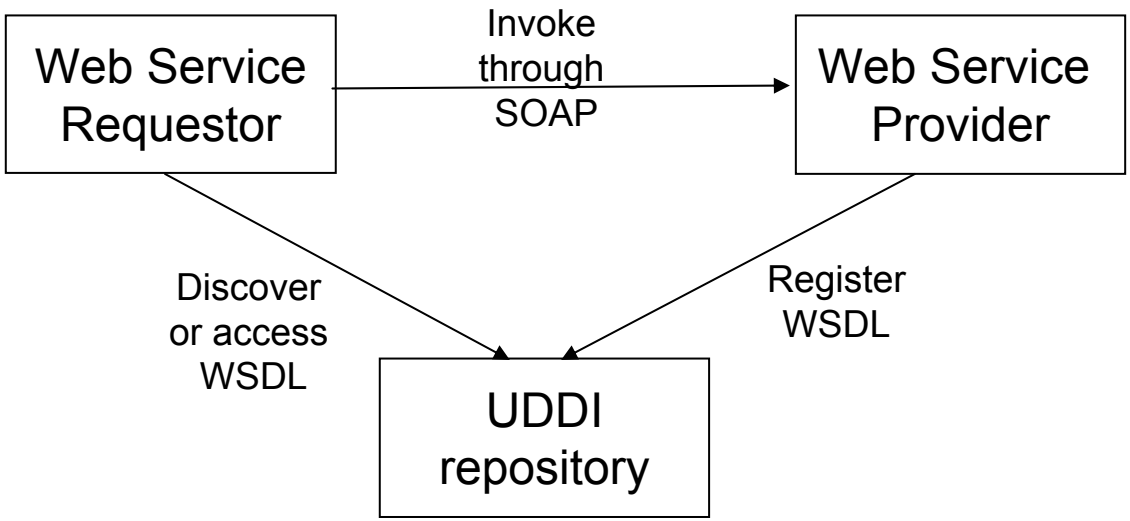
[3] Universität Innsbruck

[4] Semagix, Inc

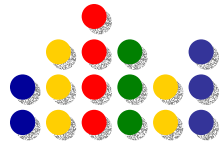
Truth & Vision



- Web Services (SOAP, UDDI, WSDL)
  - Data exchange between two programs in XML format
  - Operate on syntactic level : Web services infrastructure do not access data content.

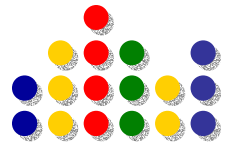


Truth & Vision



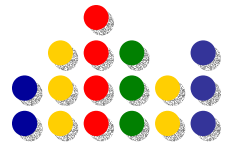
- Detour : Web Services infrastructure
  - Application Servers
    - Like Oracle's 9iAS, IBM's WebSphere or BEA's Weblogic
    - Provide infrastructure to create SOAP Messages, initiate SOAP invocation, and receive SOAP invocations
    - Provide WSDL generation and interpretation functionality
    - Provide UDDI Connectivity
  - Non-application server implementation
    - Example : CapeClear (http://www.capeclear.com)
  - Web service definition and implementation (i.e. web services logic) done by programmer in context of a web service infrastructure
- End detour

Truth & Vision



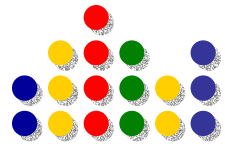
- Invocation Model
  - One way invocation
  - Request/Reply invocation
  - Solicit/Response invocation
- Invoked Entity (Service Provider)
  - Publishes WSDL operation with input and output message
- Invoker (Service Requester) : No concept
  - Especially not a “subroutine” call a la RPC with appropriate stack operations or stub generation

Truth & Vision



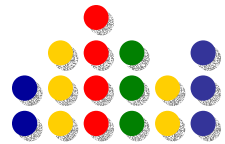
- Web Services Interoperability
  - Web Services Interoperability Organization
  - Define interoperable standards versions
  - Provide tools for interoperability testing
  - http://www.ws-i.org

Truth & Vision



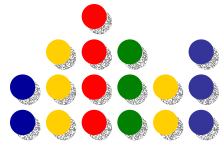
- Example of WSDL
  - Christmas Tree
    - h : height of the tree
    - r : radius of the tree
    - l : radius of the flare of the light
    - Returns number of lights in the tree
  - Example : 
xmethods_xmas_iii.pdf

Truth & Vision



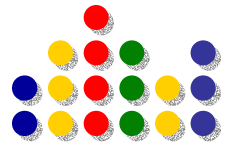
- Missing Concepts in Web services
  - Data definition
    - XML Schema is definition language for input and output message
    - No domain specific data definitions
  - Invocation behavior
    - No operation sequence definition
    - All operations are equal w.r.t. behavior. Any restriction to be known (by magic) by invoker
  - Mediation
    - No mediation of data
    - No mediation of behavior

Vision & Truth



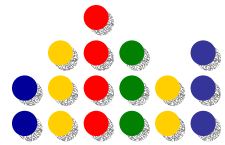
- Missing elements in Web services (continued)
  - Composition
    - No concepts for composition
  - Trading Partner Management
    - Web services recognize URIs as endpoints and do not incorporate trading partner management
  - Service level guarantees
    - Web services do not contain any service level agreements
    - Emerging Work
      - Web Services Security
        - http://www-106.ibm.com/developerworks/library/ws-secure
      - Business Transaction (OASIS)
        - http://www.oasis-open.org/committees/business-transactions

Vision & Truth



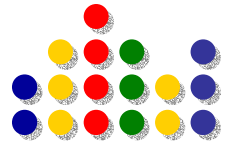
- WSMF (Web Services Modeling Framework)
  - Addresses all deficiencies of web services by providing a complete set of concepts
  - WSMF Paper will describe WSMF in detail
  - Rest of this session will discuss related work

Truth & Vision



- Related Work
  - Data definition
  - Invocation behavior
  - Mediation
  - Composition
  - Trading Partner Management
  - Service level guarantees

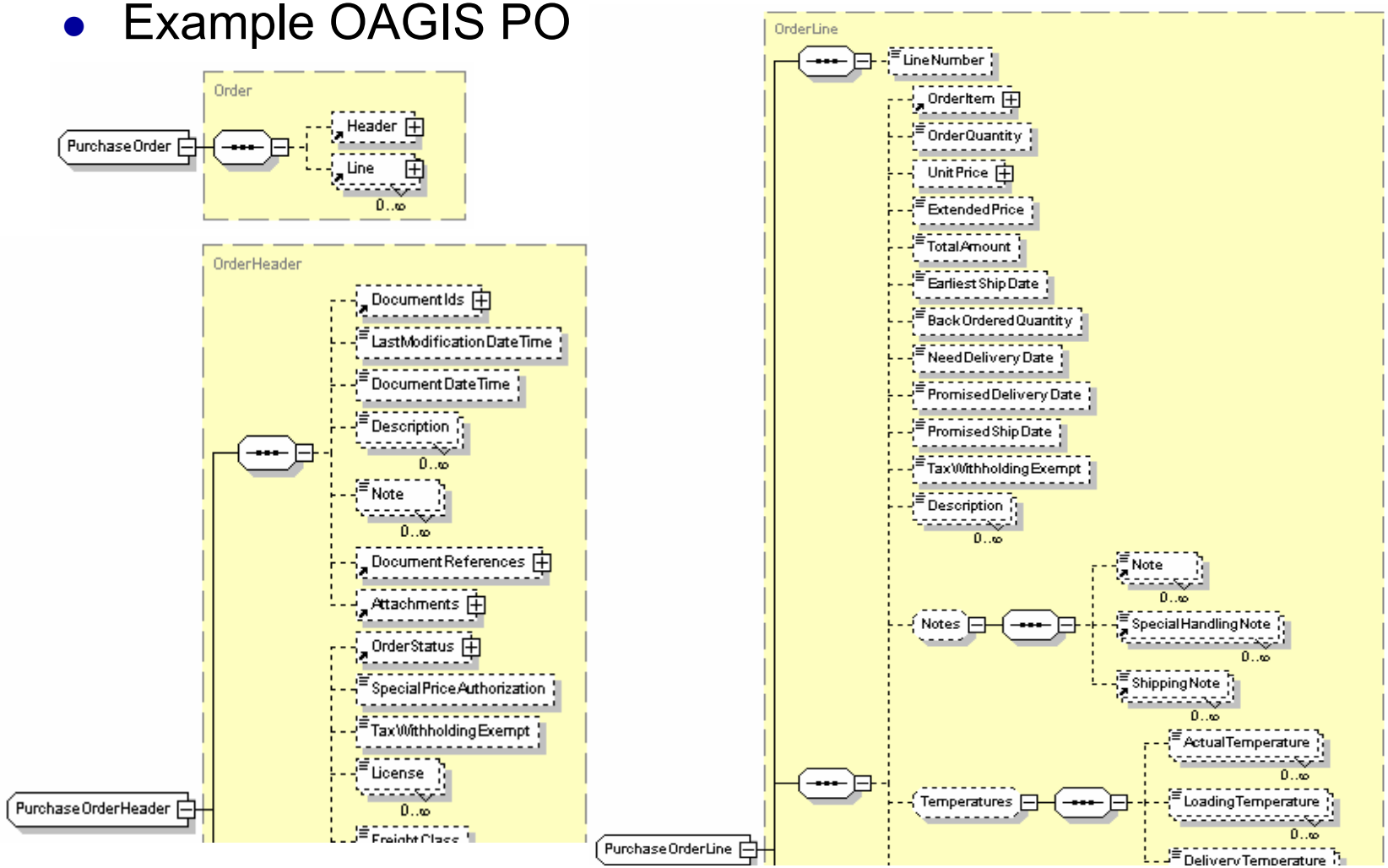
Truth & Vision



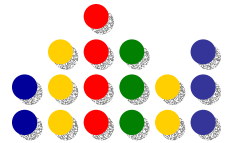
- Data Definitions
  - Open Application Group (http://www.openapplications.org)
  - RosettaNet (http://www.rosettanet.com)
  - EDI (http://www.x12.org)
  - SWIFT (http://www.swift.com)
  - UBL (http://www.oasis-open.org/committees/ubl)
  - Many, many more in all major application domains
    - See The XML Cover Pages :
      http://www.oasis-open.org/cover/sgml-xml.html
  - Not yet using ontology languages, but XML schema or equivalent syntax to define documents

Truth & Vision

- Example OAGIS PO



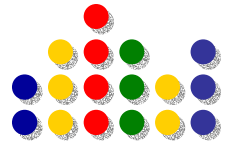
Truth & Vision



- Example EDI 850 (Purchase Order)

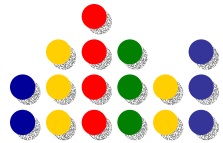
GS~PO~ERPTEST~IMPEXP~20000512~1352~142~X~004010
ST~850~0001
BEG~00~NE~616000000096~~20000420
PER~BD~JACKSON, DEBBIE
FOB~DF~ZZ~Road Freight
N1~SU~SUPPLIER NAME INC~92~999888
N2~SUPPLIER REP
PO1~0001~2~EA~~~PN~BOEING PART NUMBER 19
CTP~~~0~2~EA
PID~F~~~~Burns Part
TD5~~GA~RD
TXI~LS~~100~CD~46020106~~~~178 000 010
SCH~2~EA~~~002~20000420
N9~55~0000
N9~TX~R&D EX
N9~C7~0000

Vision & Truth

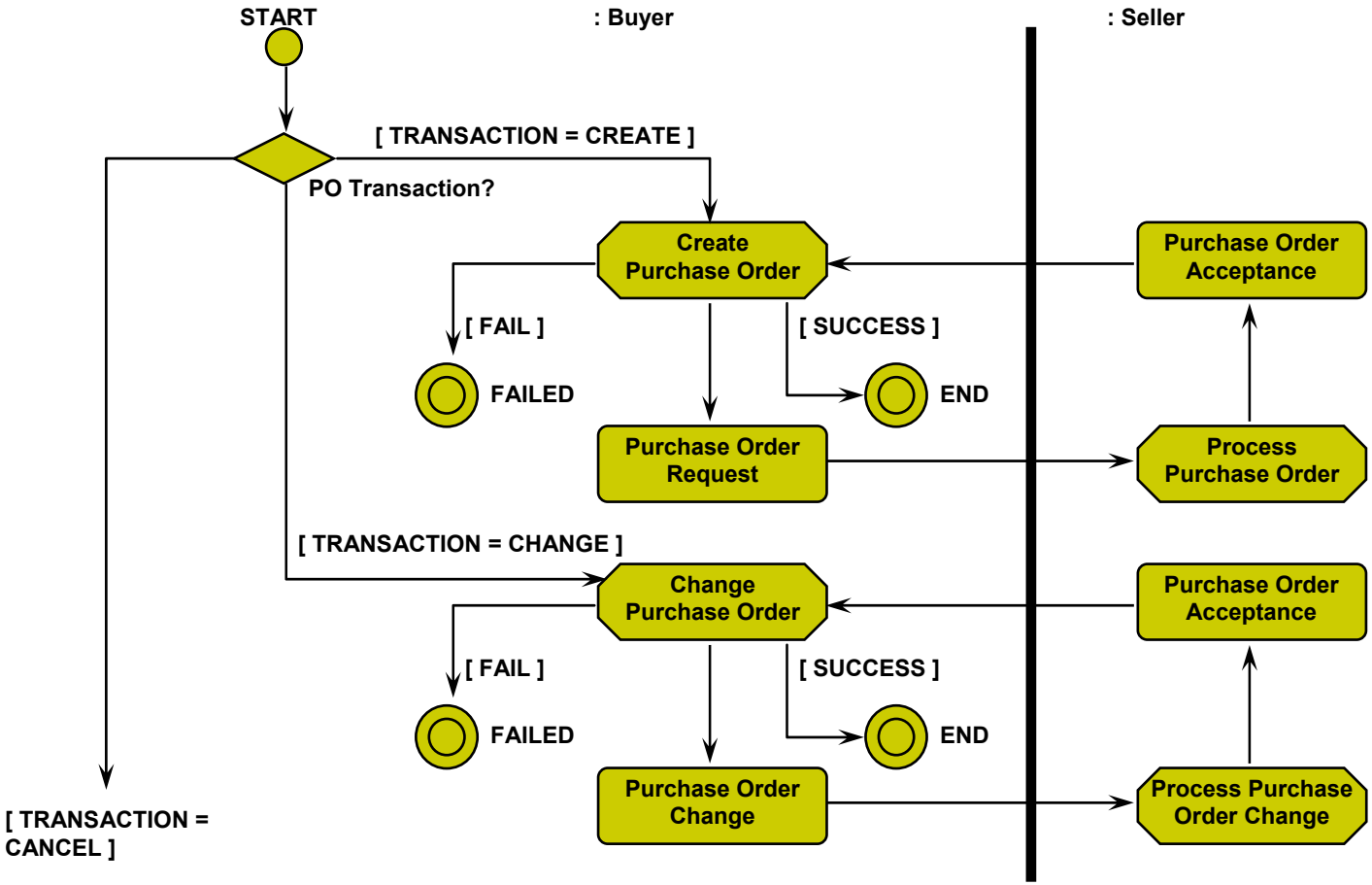


- Invocation Behavior
  - No related work for synchronous invocations
  - RPC Would be a stretch
  - P2P approach for asynchronous invocations
    - RosettaNet (http://www.rosettanet.org)
      - Partner Interface Process (PIPs) defining the behavior of both interaction trading partner
      - Domain specific behavior definition
    - Web-services conversation language (WSCL)
      - http://www.w3.org/TR/wscl10
      - Specification language for behavior

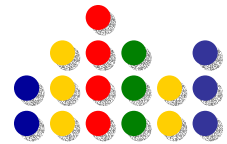
Vision & Truth



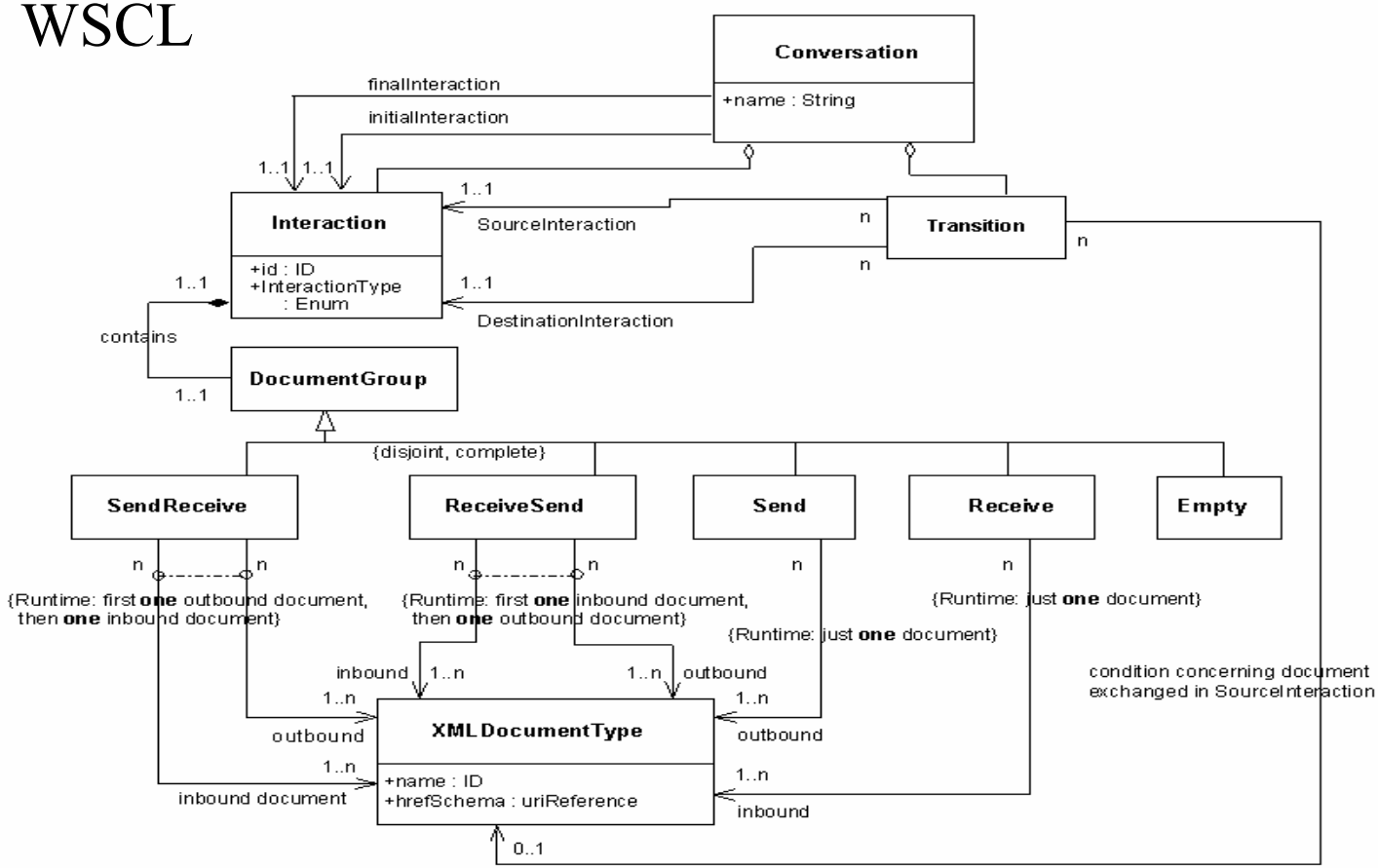
- Example RosettaNet PIP 3A4



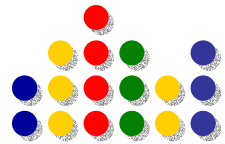
Vision & Truth



WSCL

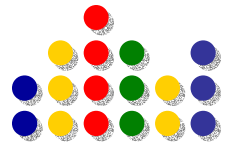


Vision & Truth



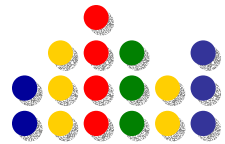
- Mediation
  - Problem definition
    - Matching internal and external
      - Data definition
      - Event exchange behavior
    - Data definition example
      - EDI purchase order to be matched with RosettaNet purchase order
    - Behavior Example
      - EDI behavior (No acknowledgements) to be matched with RosettaNet partner interface process (with acknowledgements)

Vision & Truth



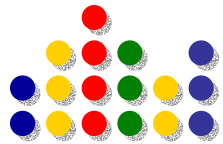
- Composition
  - So far unclear definition of “Composition” :
    - Composition in the part-of sense, i.e. larger part encapsulates web-services and exposes itself as a web-service
      - Analogy : method invocations as part of method definition
    - Composition in the sequencing sense, i.e. definition of the invocation order of web-services
      - Behavior as discussed earlier
  - Proposed language for “composition”
    - WSFL (Web Services Flow Language)
    - BPML (Business Process Modeling Language)
    - ebXML BPSS (Business Process Specification Schema)
    - BPEL4WS (Business Process Execution Language for Web Services)

Vision & Truth

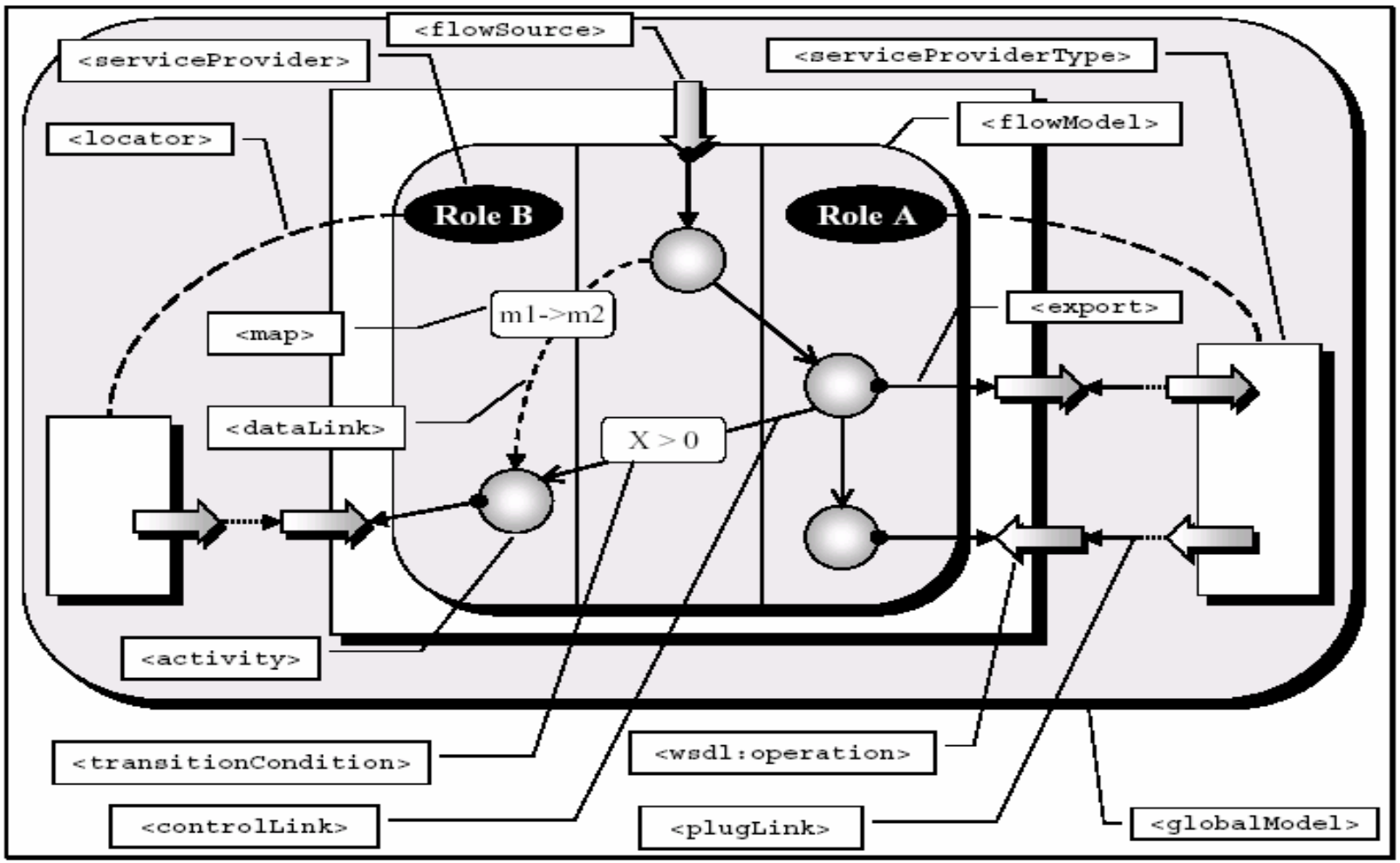


- Composition (Continued)
  - WSFL
    - http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf
    - Message Definitions
    - Port types
      - Set of operations with their input and output messages
    - Service Provider
      - Set of Port types
    - Flow Model
      - Flow model for each service provider. Defines invocation sequence of operations of port types
    - Global Model
      - Relates operations of all service providers.
  - Page 85, ticket order example gives good insight into the workings of WSFL

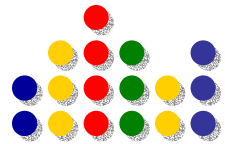
Vision & Truth



WSFL Example

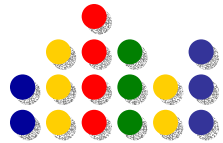


Vision & Truth



- Composition (Continued)
  - BPML (http://www.bpmi.org)
    - BPML is a workflow definition language with no references to web services or their composition
    - Data format is XML since process language contains XPATH expressions
    - Very elaborate process model that includes concepts for
      - Inter-workflow communication (message exchange between ongoing workflow instances)
      - Participants
      - Closed and open transactions
      - Compensation
      - recovery

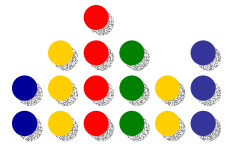
Vision & Truth



- **BPML example**

```
<process name = “TrackTrouble”>
<supports abstract = “Customer”/>
<message name = “troubleReportinput” type = “request”>
        <xsd:element name = “service” type = “Service”/>
        <xsd:element name = “trouble” type = “xsd:string”/>
</message>
<message name = “troubleReportoutput” type = “response”>
        <xsd:element name = “cookie” type = “TrackTrouble”/>
</message>
<message name = “getStatusinput” type = “request”/>
<message name = “getStatusOutput” type = “response”>…</message>
<sequence name = “reportAndTrack”>
        <operation name = “report Trouble”>
                <participant name = “reportTroubleForm”/>
                <input name = “trobleReportinput”/>
                <output name = “trobleReportOnput”>
                        <assigned target = “cookie” select = “TrackTrouble/text()”/>
                </output>
        </operation>
```

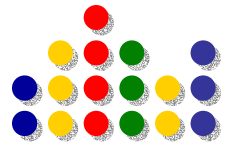
Vision & Truth



- **BPML example (Continued)**

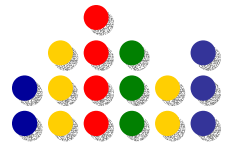
```
<operation name = “findProvider”>
        <participant select = “troublereportinput/service”/>
        <output message = “getproviderinput”/>
        <input message = “getproviderOutput”/>
</operation>
<operation name = “createticket”>
        <participant select = “getProviderOutput/provider”/>
        <output message = “openTicketinput”>
                <assign select = “troubleReportinput/trouble”/>
                <assign select = “trackTrouble/text()” target = “customer”/>
        </output>
        <input message = “openTicketOutput”/>
</operation>
<consume name = “notifyCustomer”>
        <input message = “ticketClosed”/>
</consume>
</sequence>
</process>
```

Vision & Truth



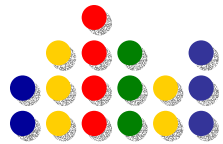
- Composition (Continued)
  - ebXML BPSS (http://www.ebxml.org)
    - Look under “Specifications”
  - ebXML BPSS is a Process Specification Language
  - Specific emphasis on document exchange
    - Business data messages
    - Acknowledgement messages
  - Message activities
    - Non-repudiation
    - Confidential
    - Encrypted

Vision & Truth



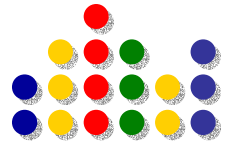
- ebXML BPSS example
  - Please refer to the specification; language (XML) is chatty ;-)

Vision & Truth



- Composition(continued)
  - XLANG
    - www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm
  - Extension of WSDL for behavior definition
  - Main constructs (block structured)
    - Activation operation, i.e WSDL operation that starts the behavior
    - Operation, delayFor, delayUntil, raise
    - Empty, Sequence, Switch, While, All, Pick
    - Correlation
    - Context

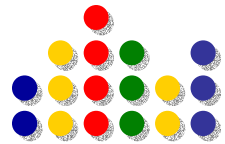
Vision & Truth



- **XLANG Example**

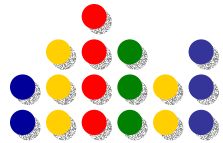
```
<?xml version = “1.0”?>
<definitions name=“StockQuoteProvider”..>
<service>
   <xlang:behaviour>
                <xlang:body>
                          <Xlang:sequence>
                          <xlang:action operation=“AskLastTradePrice”
                                    port=“pGetRequest” activation=“true”/>
                          <xlang:action operation=“SendLastTradePrice”
                                    port=“pSendRespnse”/>
                           </Xlang:sequence>
                </xlang:body>
   </xlang:behaviour>
</service>
</definitions>
```

Vision & Truth



- Trading partner management
  - ebXML
    - CPP:Collaboration partner profile: Properties of collaboration partners
    - CPA : Collaboration partner agreement.Agreement between collaboration partners about the rules of engagement
  - EDI
    - Document type 838 that allows the communication of trading partner attributes
  - ERPs
    - ERP internal management of trading partner information that is available and accessible

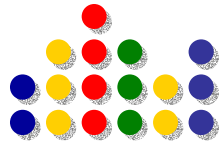
Vision & Truth



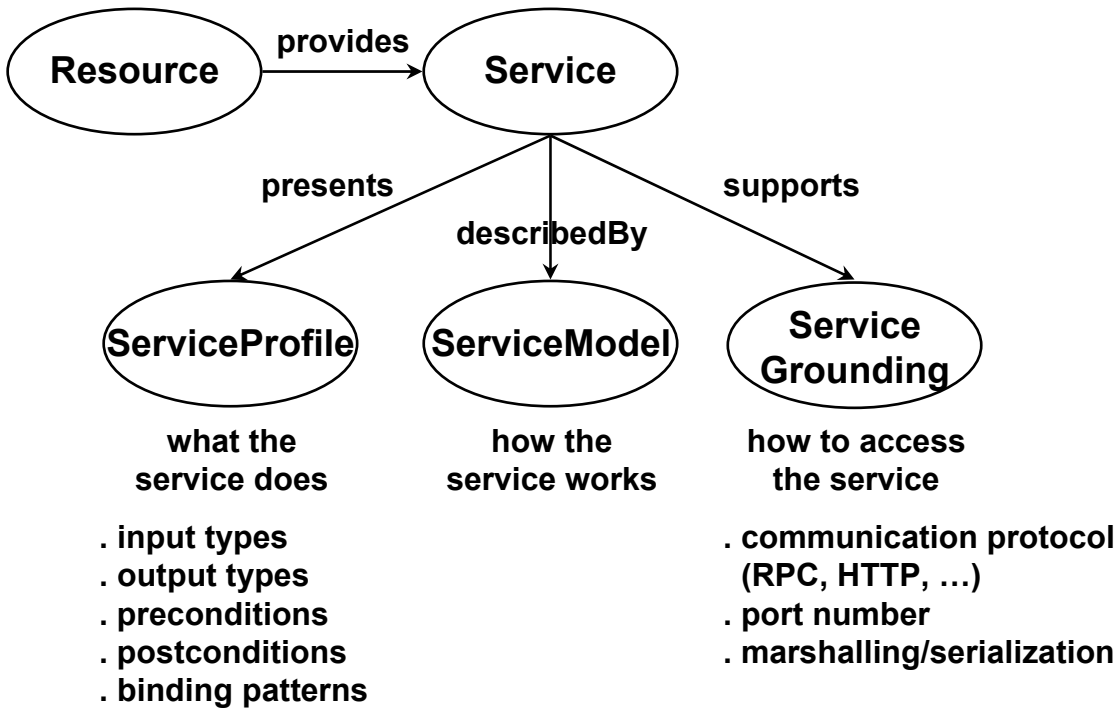
- Service level guarantees
  - Reliable message transmission over unreliable network
    - RosettaNet
      - Time-outs for expected delays in responses (“time to perform”)
      - Retry counter
      - Resending of messages
      - Agreement in which state interaction considered failure or success, no explicit message sent to indicate failed or succeeded behavior
    - Emerging : business transactions

  Security
    - Signatures, encryption, non-repudiation
    - Emerging: web services security (see earlier)

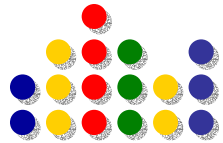
Vision & Truth



- DAML-S - An overview
  - DAML (DARPA Agent Markup Language)
  - DAML-S: Upper ontology of web services

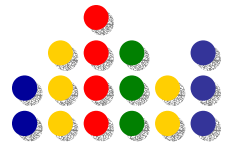


Truth & Vision



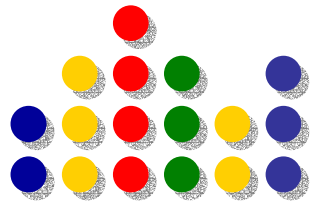
- Trading partner management
  - www.daml.org/services
  - Subclass Of Service Model : ProcessModel
    - Process (defined in Process Ontology)
    - Process control (defined in process Control Ontology)
  - Process Ontology
    - Process
      - Atomic, simple process and composite process
    - Control constructs
      - Sequence, Spit, Unordered, Split+Join, Choice, If-Then-Else, Iterate, Repeat_Until
    - Process Control Ontology, Time, Resources
  - Section 2 of the tutorial will introduce DAML-S in more detail

Vision & Truth



Questions ?

Web Services
Composition



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

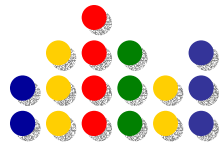
[1]LSDIS Lab, Computer Science, University of Georgia

[2]Oracle Corporation

[3] Universität Innsbruck

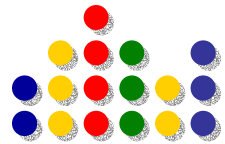
[4] Semagix, Inc

Objective



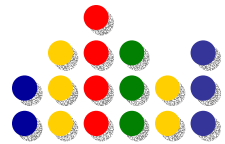
- The Internet provides a valuable infrastructure to support new business models such as
  - E-services, E-commerce, Business-to-Business (B2B), Business-to-Customer (B2C), Customer-to-customer (C2C), Virtual Organizations, etc.
- To support these models, research and new solutions need to be explored.
- One important aspect is the **composition of processes.**

Requirements for Making Web Services a Working Technology



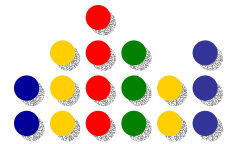
- UDDI, WSDL, and SOAP are important steps into the direction of a web populated by services.
- However, they only address part of the overall stack that needs to be available in order to achieve the above vision eventually.
- There are many layer required to achieve automatic web service discovery, selection, mediation and composition into complex services.

Requirements for Making Web Services a Working Technology



- Document Structure
- Semantics
- Process Definition
- Message layer (receiving, understanding)
- Packaging
- Transport binding
- Security ...

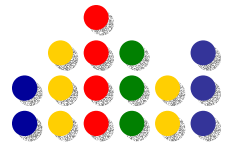
Requirements for Making Web Services a Working Technology



Document Structure

- Document types describe the content of business documents like purchase orders or invoices.
- The content is defined in terms of elements like an order number or a line item price.
- Document types are instantiated with actual business data when a service requester and a service provider exchange data.
- The payload of the messages sent back and forth are structured according to the document types defined.

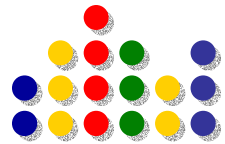
Requirements for Making Web Services a Working Technology



Semantics

- The elements of document types must be populated with correct values so that they are semantically correct and are interpreted correctly by the service requesters and providers.
- This requires that vocabulary is defined that enumerates or describes valid element values.
- For example, a list of product names or products that can be ordered from a manufacturer. Further examples are unit of measures as well as country codes.

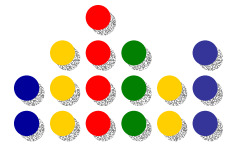
Requirements for Making Web Services a Working Technology



Process definition

- Based on the assumption that messages can be exchanged the business logic has to be defined in terms of the business message exchange sequence.
- For example, a purchase order might have to be confirmed with a purchase order acknowledgment. Or, a request for quotation can be responded to by one or more quotes.
- These processes define the required business message logic in order to derive to a consistent business state. For example, when good are ordered by a purchase order and confirmed by a purchase order acknowledgment they have to be shipped and paid, too.

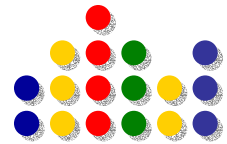
Requirements for Making Web Services a Working Technology



Exchange sequence definition

- Communication over networks are currently inherently unreliable.
- It is therefore required that service requester and service provider make sure themselves through protocols that messages are transmitted exactly once.
- The exchange sequence definition achieves this by defining a sequence of acknowledgment messages in addition to time-outs, retry logic and upper retry limits.

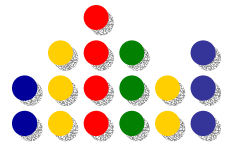
Requirements for Making Web Services a Working Technology



Transport binding

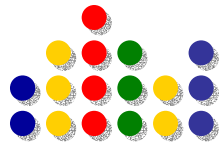
- Several transport mechanisms are available like HTTP/S, S/MIME, FTP or EDIINT.
- A service requester as well as service provider have to agree on the transport to be used when service requests are executed.
- For each available transport the layout of the message must be agreed upon and how the document sent is represented in the message sent.

Web Services
Composition



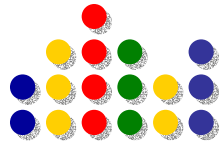
- Composition is the task of combining and linking existing Web Services and other components to create new processes.
- It adds value to the collection of services, by orchestrating them according to the requirement of the problem.
- Types of Composition
  - Static Composition - services to be composed are decided at design time
  - Dynamic Composition - services to be composed are decided at run-time

Web Service Composition
Issues

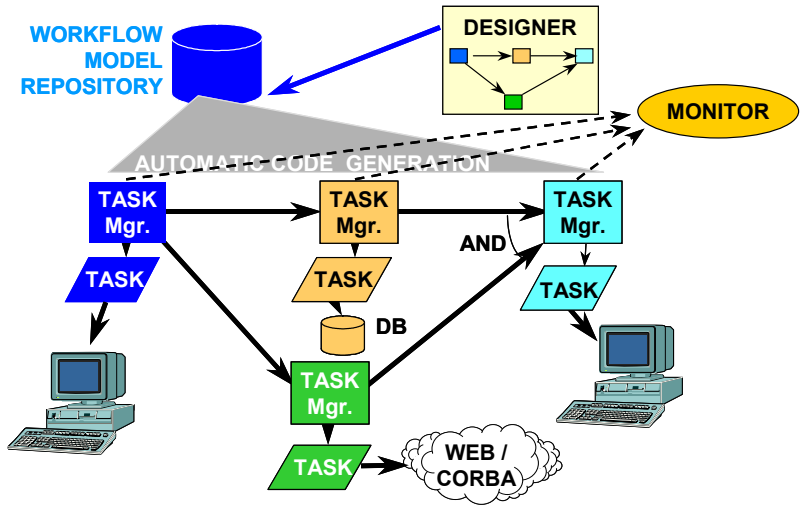


- ✓ Representation of an Abstract Web Process
  - Representing/specifying the abstract process in a proper form
- ✓ Discovery and Interoperability of Services
  - Need to manually or automatically search for appropriate services
  - The discovered services should interoperate
- ✓ Efficiency of a Composed Web Process
  - Need to compose processes which are efficient in terms of performance
- ❑ Process Execution
  - Adopting a suitable technique for executing the composed concrete process
- ❑ Process Monitoring
  - Using a monitoring technique for run time analysis of the Web process execution

Web Services and Workflow Systems



- Web Services can be orchestrated with hard-coded applications or by using workflows.

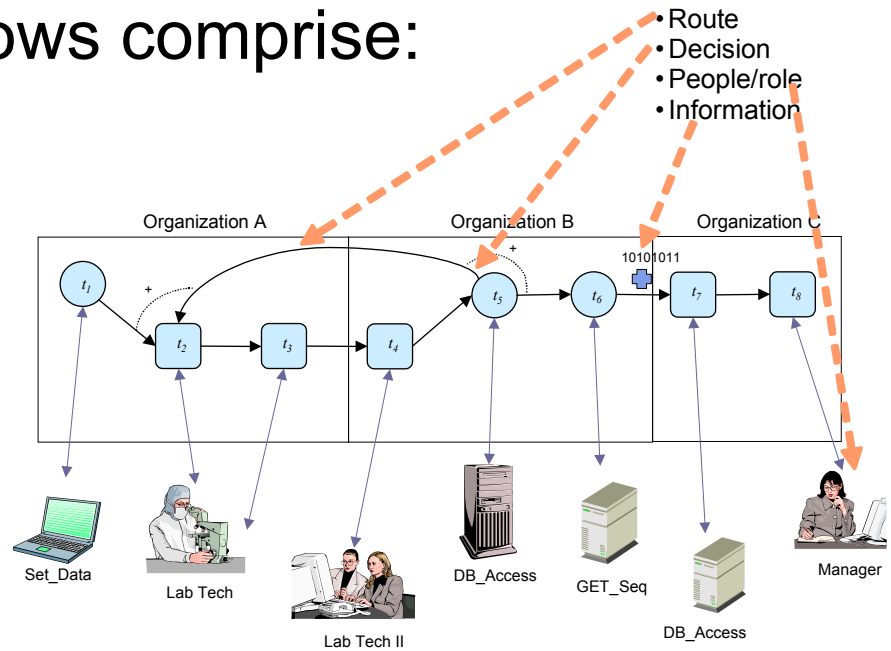


Workflow management systems are capable of integrating business objects for setting up e-services (Web Services) in an amazingly short time and with impressively little cost*.

*Shegalov, Gillmann et al. 2001

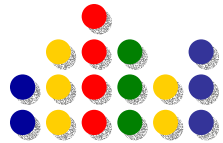
Web Processes and Workflows
Comparison

- Web Processes/Workflows comprise:
  - Web Services/Tasks,
  - Routing rules,
  - Decisions,
  - Participants and,
  - Information

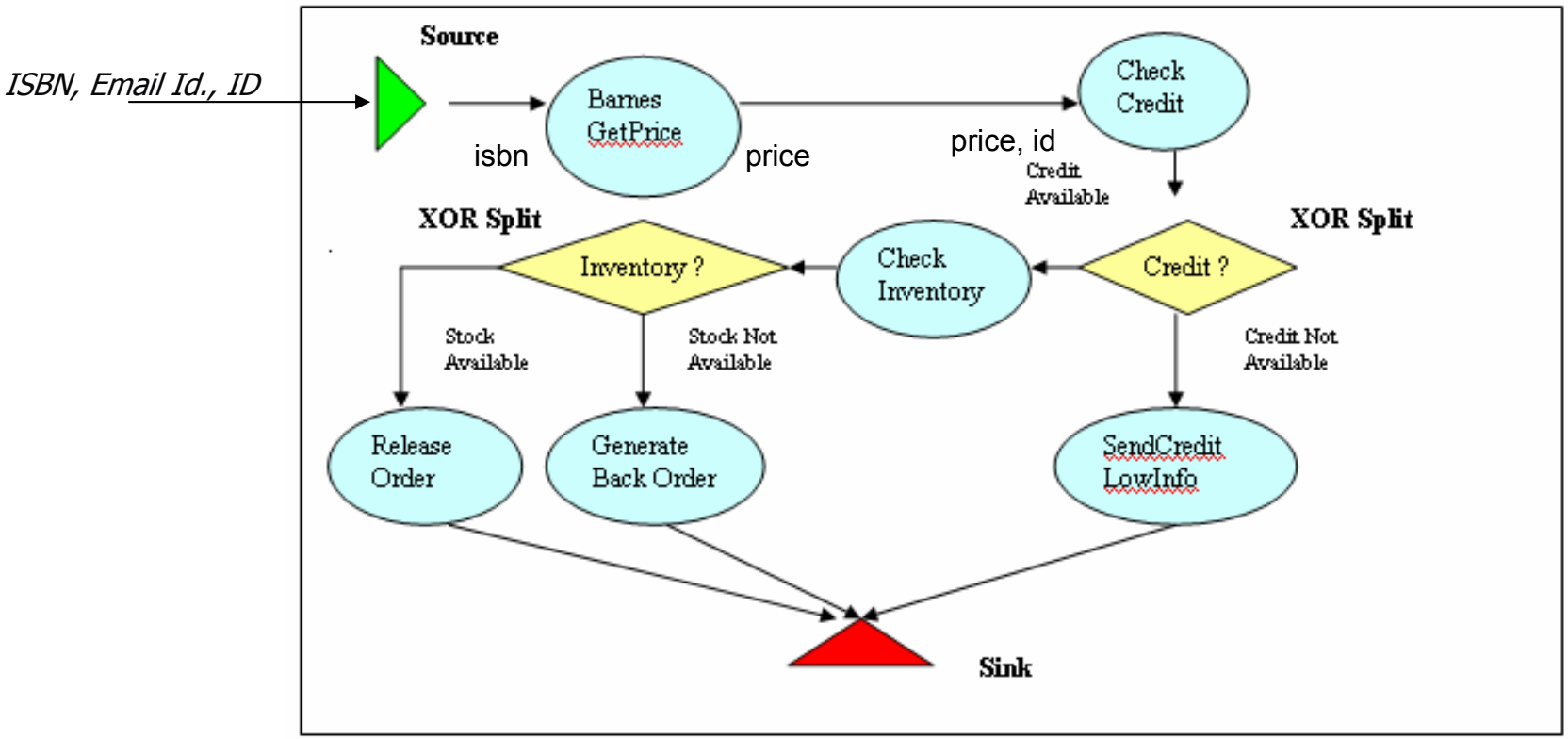


A Workflow Management System (WfMS) is a system or set of tools that completely defines, manages, and executes a workflow or Web Process.

Processes
A simple example



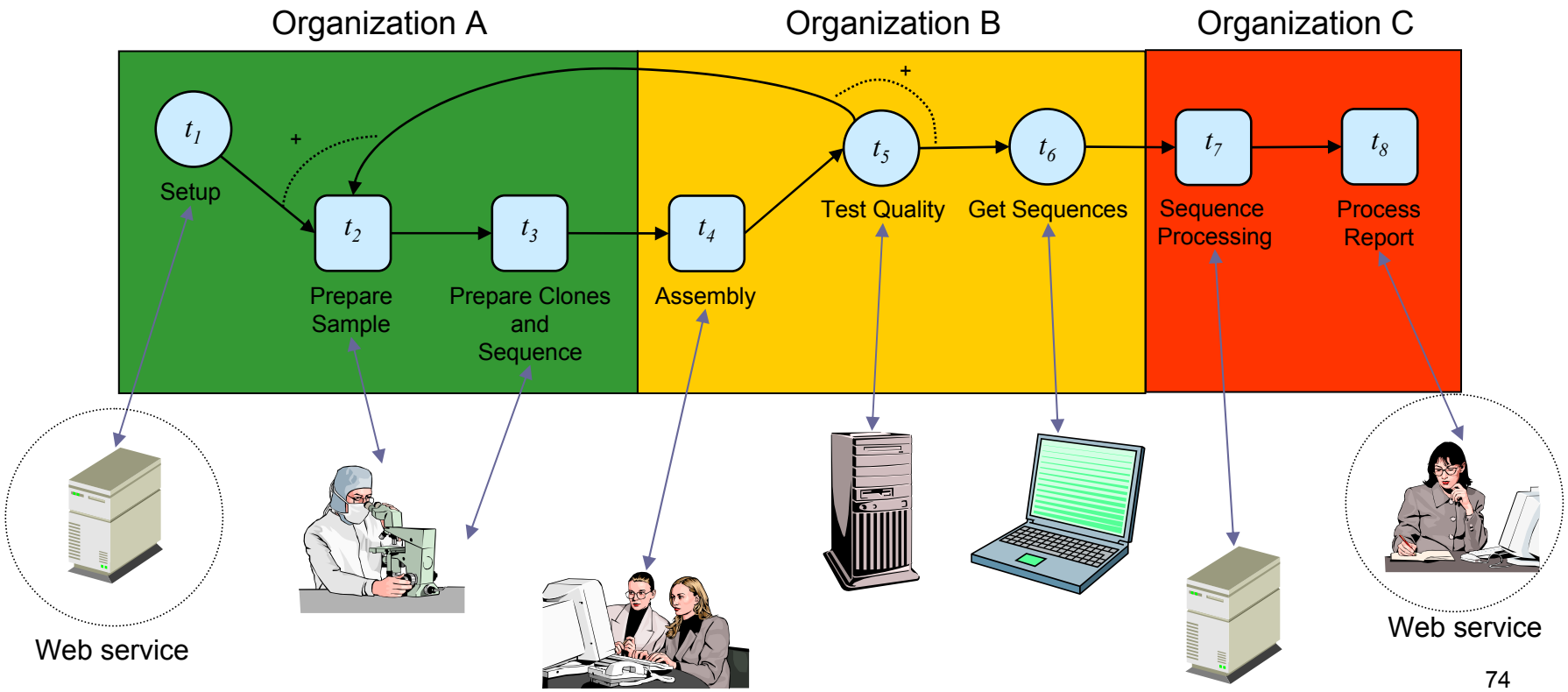
- A process is an abstract representation of a business process.



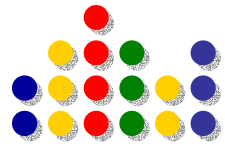
The BarnesBookPurchase process

Processes
A more complex example

A Web process can be viewed as a workflow for which the tasks are represented with Web services



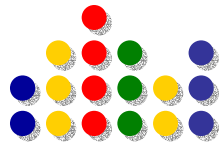
Processes
Execution



- Once the design of a process is completed, it can be executed.
- Processes can be executed with hard-coded applications or by using workflows.
- Workflows are enacted with Workflow Management System (WfMS) or other process orchestration technology.

WfMS: A system or set of tools that completely defines, manages, and executes a workflow.

Process Composition
Challenges



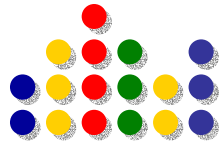
The composition of cross-organizational Internet-based processes requires new technological developments which include:

- **Discovery of Web Services**

- **Integration of Web Services**

- **End-to-End Process Analysis**
  - **Correctness/validation, performance**

References



Berners-Lee, T. (2001). Keynote presentation on web services and the future of the web. Software Development Expo 2001 Visionary Keynote, http://www.technetcast.com/tnc_play_stream.html?stream_id=616.

Cardoso, J., J. Miller, A. Sheth and J. Arnold (2002). "Modeling Quality of Service for Workflows and Web Service Processes." the Very Large Data Bases Journal submitted in May 2002.

Casati F., M-C. Shan and D. Georgakopoulos (2001). "E-Services - Guest editorial." The VLDB Journal 10(1): 1.

Chadrasekaran S., J. Miller, G. Silver, I. B. Arpinar and Amit Sheth (2002). Composition Performance Analysis and Simulation of Web Services, September. http://lsdis.cs.uga.edu/lib/download/CMSA+02.pdf

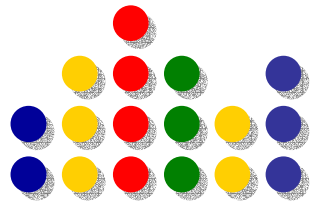
Fensel, D. and C. Bussler (2002). The Web Service Modeling Framework. Vrije Universiteit Amsterdam (VU) and Oracle Corporation, http://www.cs.vu.nl/~dieter/ftp/paper/wsmf.pdf.

Kochut, K. J., A. P. Sheth and J. A. Miller (1999). "ORBWork: A CORBA-Based Fully Distributed, Scalable and Dynamic Workflow Enactment Service for METEOR," Large Scale Distributed Information Systems Lab, Department of Computer Science, University of Georgia, Athens, GA. (also: http://lsdis.cs.uga.edu/lib/download/KSM99.pdf)

Paolucci, M., T. Kawamura, T. R. Payne and K. Sycara (2002). Semantic Matching of Web Services Capabilities. Proceedings of the 1st International Semantic Web Conference (ISWC2002), Sardinia, Italia.

Uschold, M. and M. Gruninger (1996). "Ontologies: Principles, methods and applications." Knowledge Engineering Review 11(2): 93-155.

Web Services
Discovery and Integration



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

[1]LSDIS Lab, Computer Science, University of Georgia

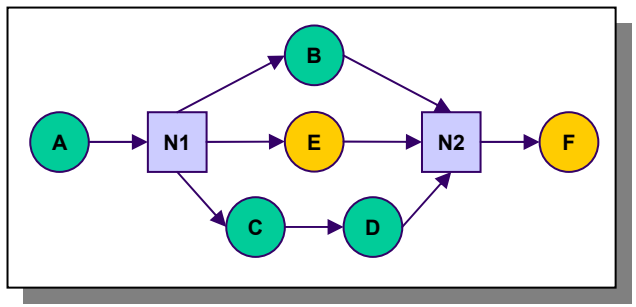
[2]Oracle Corporation

[3] Universität Innsbruck

[4] Semagix, Inc

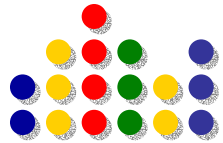
Introduction

- E-services (Web Services) have been are announced as the next wave of Internet-based business applications that will dramatically change the use of the Internet[1].
- While in some cases Web services may be utilized in an isolated form, it is natural to expect that Web services will be integrated as part of processes[2] (Web processes).



[1]Fabio Casati, Ming-Chien Shan et al. 2002, [2] Berners-Lee 2001; Fensel and Bussler 2002.

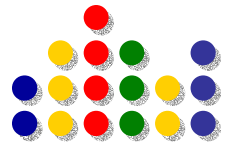
Web Services
Discovery and Integration



To compose a process it is necessary to discover and integrate a set of Web services.

- **Web Service Discovery**
- **Web Service Integration**

Web Services
Discovery and Integration

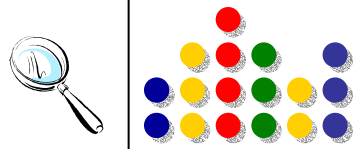


- Web Services must be located (Discovery) that might contain the desired functionality, operational metrics, and interfaces needed to carry out the realization of a given task.

- Once the desired Web Services have been found, mechanisms are needed to facilitate the resolution of structural and semantic differences (Integration).

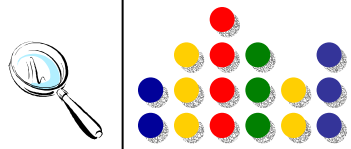
- This is because the heterogeneous Web services found in the first step need to interoperate with other components present in a workflow host.

Discovery
New Requirements



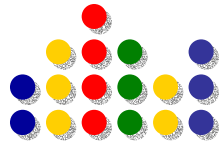
- In traditional workflow processes the selection of tasks is made from a repository.
  - Contains tens to a few hundreds of tasks.
  - The selection is humanly manageable.
- In Web processes.
  - Potentially thousands of Web services are available.
  - It is impossible for a designer to manually browse through all of the Web services available and select the most suitable ones.
  - Requires the analysis of Web services QoS.

Discovery
New Requirements



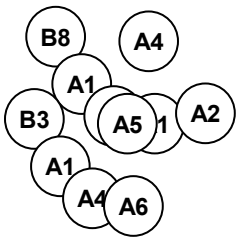
- The autonomy of Web services does not allow for designer to identify their operational metrics at design time.
- Nevertheless, when composing a process it is indispensable to inquire the Web services operational metrics.
- Operational metrics characterize the Quality of Service (QoS) that Web services exhibit when invoked.

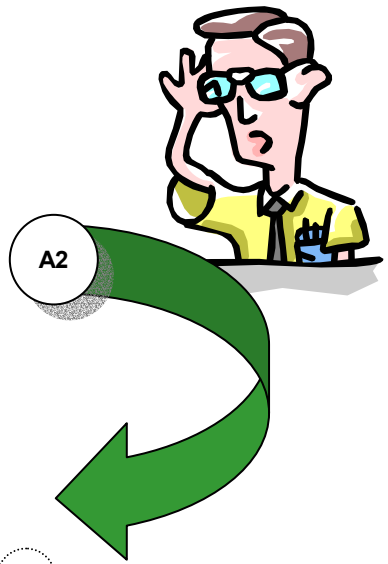
Discovery
New Requirements



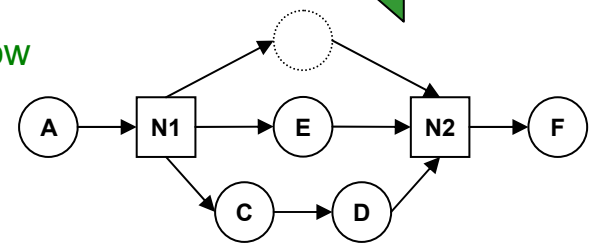
Before	Now

Tasks



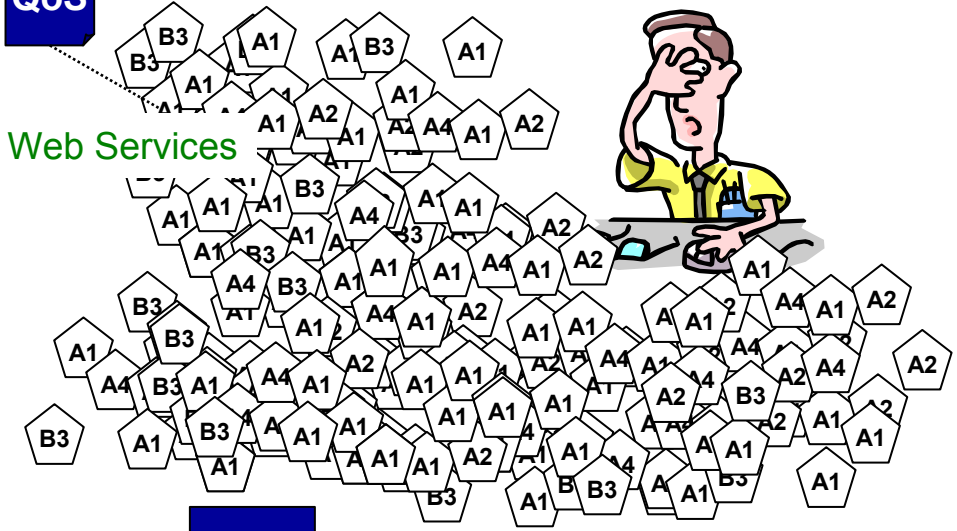


Workflow



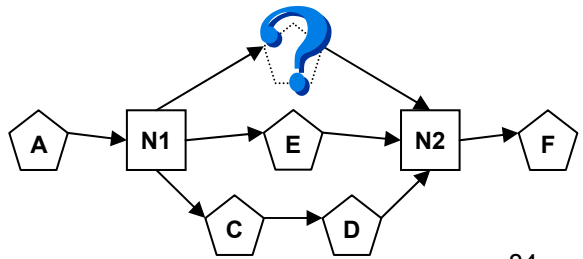
QoS

Web Services

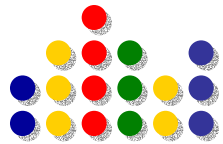


QoS

Web Process



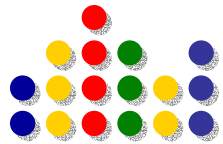
Integration
New Requirements



- Once the desired Web services have been found, mechanisms are needed to facilitate the resolution of structural and semantic differences.

- This is because the heterogeneous Web services found in the first step need to interoperate with other components present in a process.

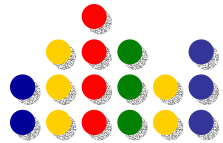
Integration
New Requirements



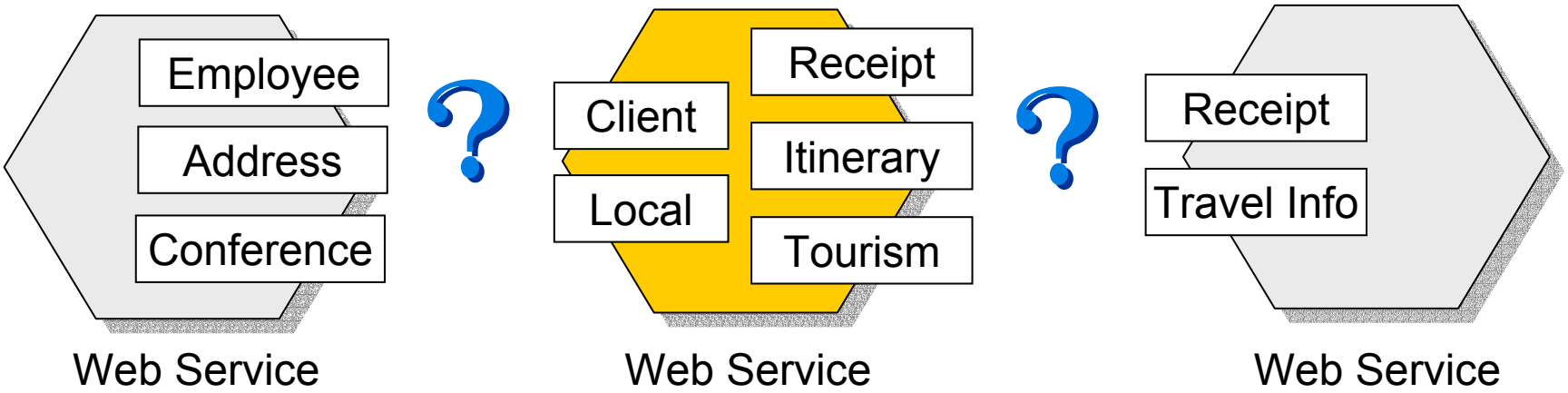
- When Web services are put together
  - Their interfaces need to interoperate.
  - Structural and semantic heterogeneity need to be resolved*.
- Structural heterogeneity exists because Web services use different data structures and class hierarchies to define the parameters of their interfaces.
- Semantic heterogeneity considers the intended meaning of the terms employed in labeling interface parameters. The data that is interchanged among Web services has to be understood.

*Kashyap and Sheth 1996

Integration
New Requirements



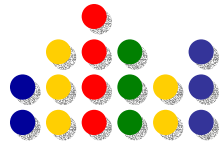
How to establish data connections between Web Services interfaces?



How to establish data connections between the different data structures and class hierarchies of the interface parameters?

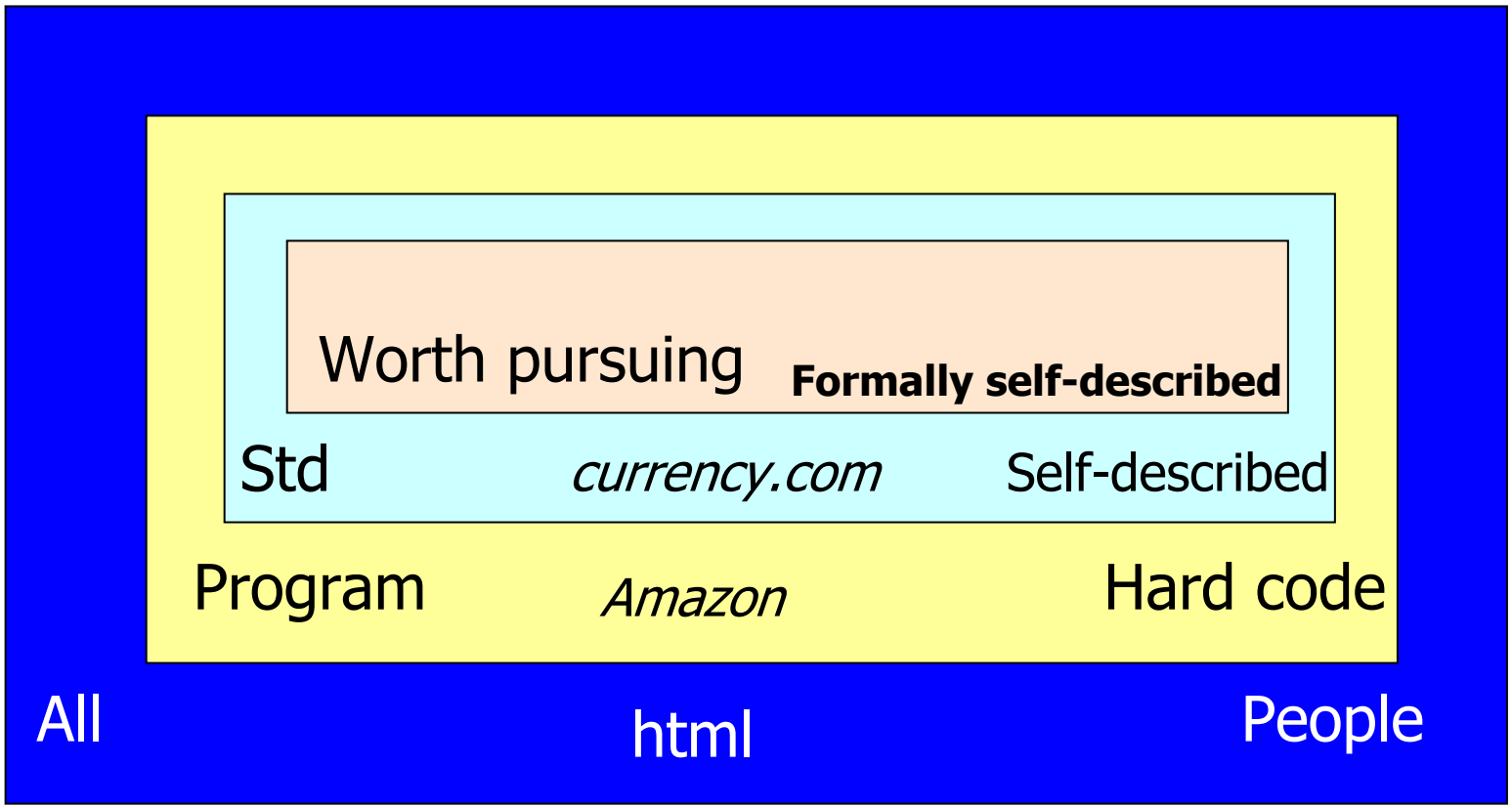
How to understand the intended meaning of the terms used in labeling interface parameters?

Our Approach

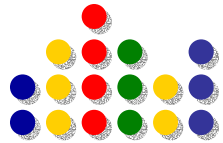


- We rely on the use of ontologies to describe Web services and their interfaces.
- Interfaces parameters can be specified with distinct ontological concepts.
- We use a QoS model to describe operational metrics.

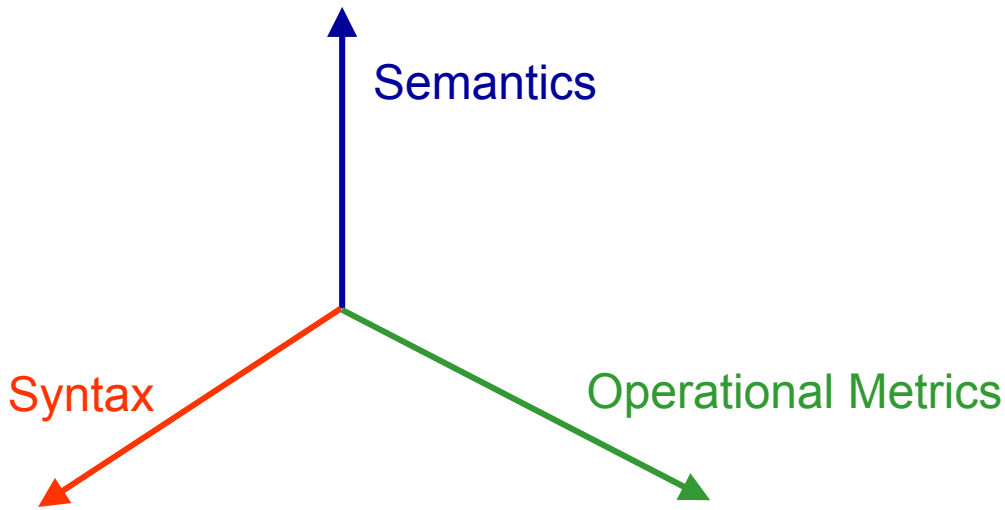
Our Approach
The use of Semantics



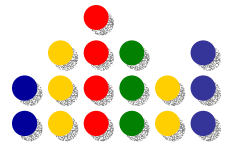
Our Approach



- Our method provides a multidimensional approach to Web service discovery and integration using syntactic, operational, and semantic information.

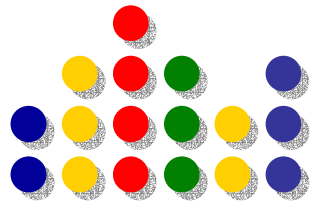


Road Map

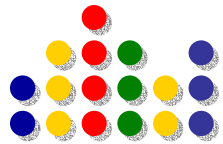


- Web Service Specification
  - Interface Specification
- Quality of Service (QoS)
- Web Process/Workflow Composition
  - Discovery
  - Integration

Web Services
Semantic Specification



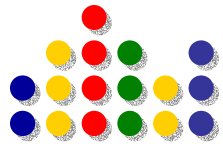
Web Services
Specification



- The importance of Web services has been recognize by the academia and by commercial organizations.
- Several efforts are being carried to develop a specification language for Web services.
- Two main approaches have been proposed.
  - One of the approaches uses declarative and structured data based purely on syntax, such as WSDL[1] and XLANG[2].
  - A second approach provides a semantic orientation to the description of Web services. This is the case in the DAML-S[3] specification.

[1]Christensen, Curbera *et al.* 2001, [2]Thatte 2001, [3]Ankolekar, Burstein *et al.* 2001

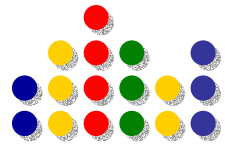
Web Services
Specification



- As with WSMF*, our approach to Web Process composition is not dependent on the method chosen to specify Web services.
- Therefore, any of the specification languages mentioned previously can be employed.
- For the system that we have developed we have selected the DAML-S specification; more precisely, we use the *Service Profile* ontology.
- The service profile ontology describes the functionality of a Web service.

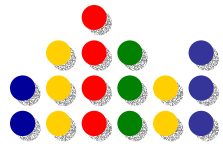
*Fensel and Bussler 2002

Web Services
Specification



- Web Services Specification
  - We use DAML-S to specify Web services.
  - Web Services interfaces are associated with ontological concepts.
  - When using DAML-S, the association of interface parameters with ontological concepts is facilitate.
- Operational Metrics Specification
  - Operational metrics are described using a QoS model represented with a suitable ontology.

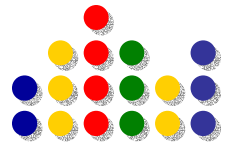
Web Services
Semantic description



- The semantic description of Web services allows
  - To better advertise and subsequently discover Web services
  - And supply a better solution for the selection, composition and interoperation of Web services.

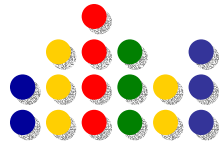
DAML-S ontologies can be used to achieve this purpose.

DAML-S
Introduction

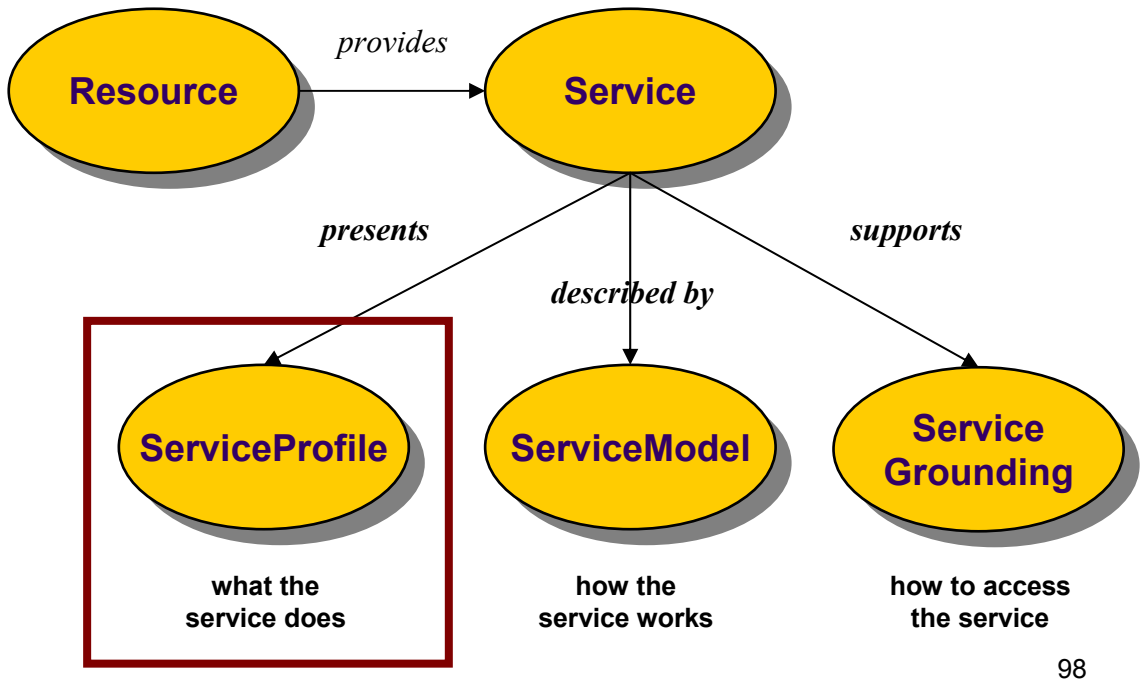


- DAML-S
  - DAML (DARPA Agent Markup Language)
  - DAML-S: Upper ontology of web services
- DAML-S provides support for the following elements:
  - Process description.
  - Advertisement and discovery of services.
  - Selection, composition & interoperation.
  - Invocation.
  - Execution and monitoring.

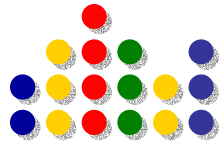
DAML-S
Ontologies



- DAML-S defines ontologies for the construction of service models:
  - Service Profiles
  - Process Models
  - Service Grounding



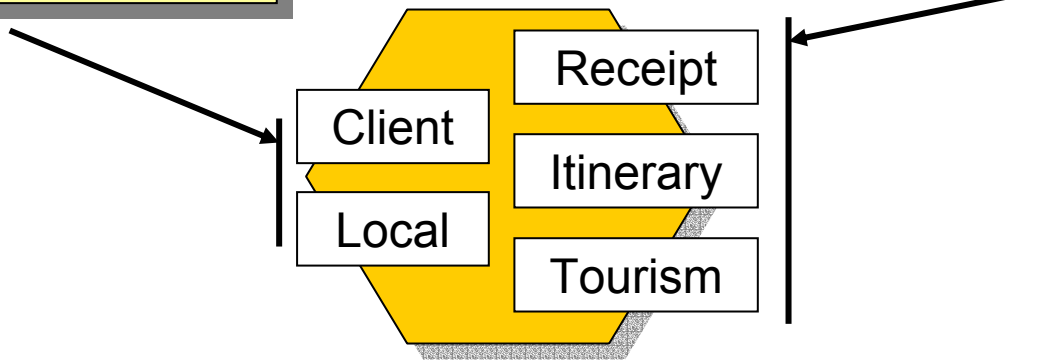
DAML-S
Service Profile



The Service Profile provides details about a service.

Inputs. Inputs that should be provided to invoke the service.

Outputs. Outputs expected after the interaction with the service.



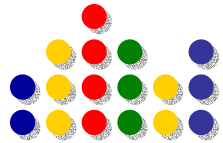
Web Service

Preconditions. Set of conditions that should hold prior to the service being invoked.

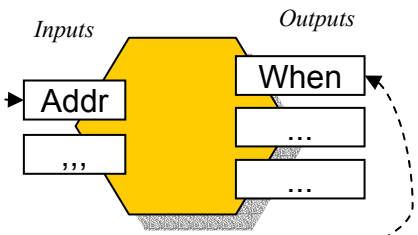
Effects. Set of statements that should hold true if the service is invoked successfully.

Service Profile

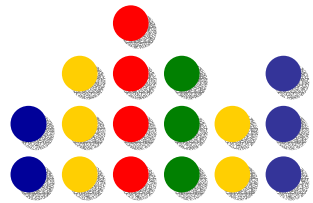
An example of Inputs and Outputs



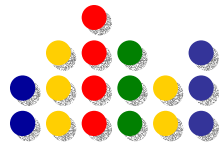
```
...
<!ENTITY temporal "http://ovid.cs.uga.edu:8080/scube/daml/Temporal.daml">
<!ENTITY address "http://ovid.cs.uga.edu:8080/scube/daml/Address.daml">
...
<input>
    <profile:ParameterDescription rdf:ID="Addr">
    <profile:parameterName> Addr </profile:parameterName>
    <profile:restrictedTo rdf:resource="&address;#Address"/>
    <profile:refersTo rdf:resource="&congo;#congoBuyReceipt"/>
    </profile:ParameterDescription>
</input>
...
<output>
    <profile:ParameterDescription rdf:ID="When">
    <profile:parameterName> When </profile:parameterName>
    <profile:restrictedTo rdf:resource="&temporal;#Date"/>
    <profile:refersTo rdf:resource="&congo;#congoBuyReceipt"/>
    </profile:ParameterDescription>
< output >
...
```



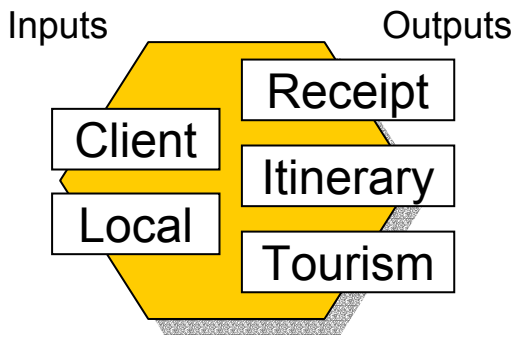
Web Services
Interface Specification



Web Services
Interfaces

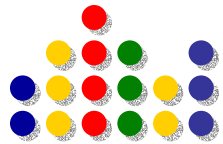


- A Web Service invocation specifies:
  - The number of input parameters that must be supplied for a proper task realization and
  - The number of outputs parameters to hold and transfer the results of the task realization to other tasks.



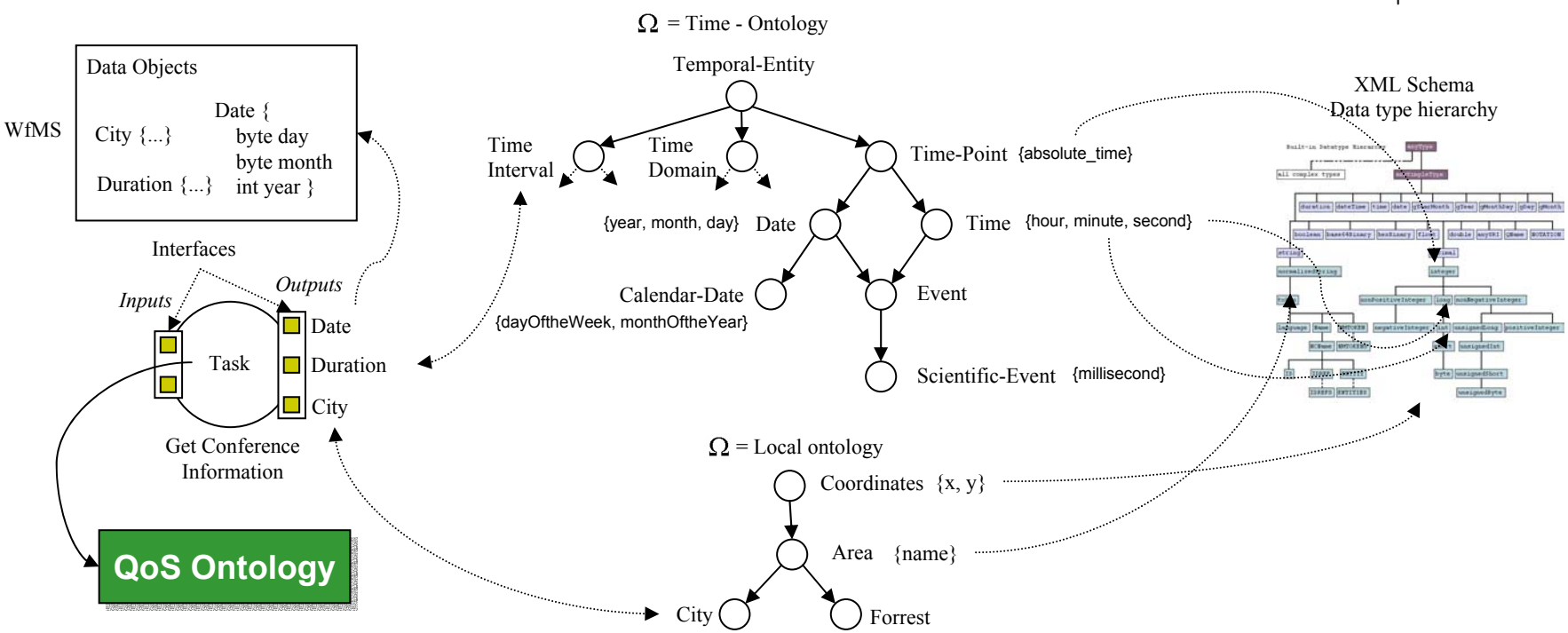
In their simplest form, the input and output parameters can be represented by attributes, or they can follow an object-oriented model represented by data components.

Web Services
Interfaces



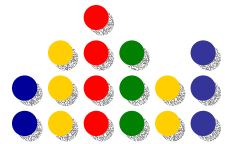
- To enhance the integration of tasks and Web services, workflow components need to have their inputs and outputs associated with ontological concepts (classes).
- This will facilitate the resolution of structural and semantic heterogeneity.

Web Services
Interfaces



Since there is a strong analogy between the attributes and data classes of an object-oriented model and the concepts classes defined in an ontology the association is facilitated.

Mapping Interfaces with Ontological Concepts

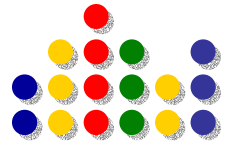


- To enhance the discovery and integration of Web services, it is necessary to increase the description of their interfaces.

- One solution is to associate the interfaces with ontological concepts.

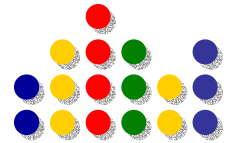
An ontology is a specification of a representational vocabulary for a shared domain of discourse.

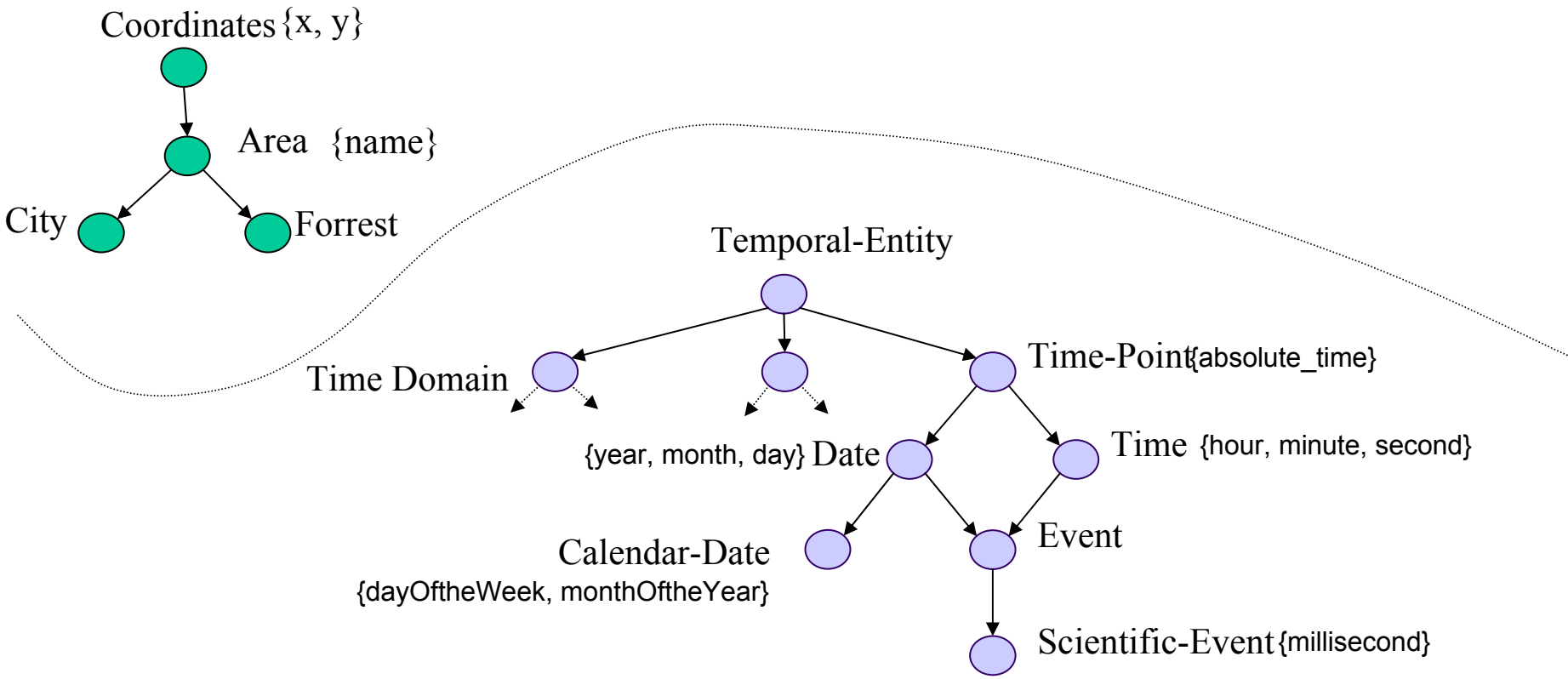
What is an Ontology



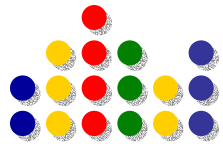
- An ontology may take a variety of forms.
- But necessarily it will include a vocabulary of terms, and some specification of their meaning.
  - This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms.
- The goal is to create an agreed-upon vocabulary and semantic structure for exchanging information about that domain.

Ontologies
Two Simple Examples





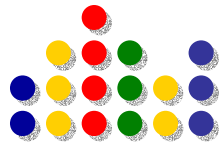
Ontologies-based approaches
Shared and non-shared ontologies



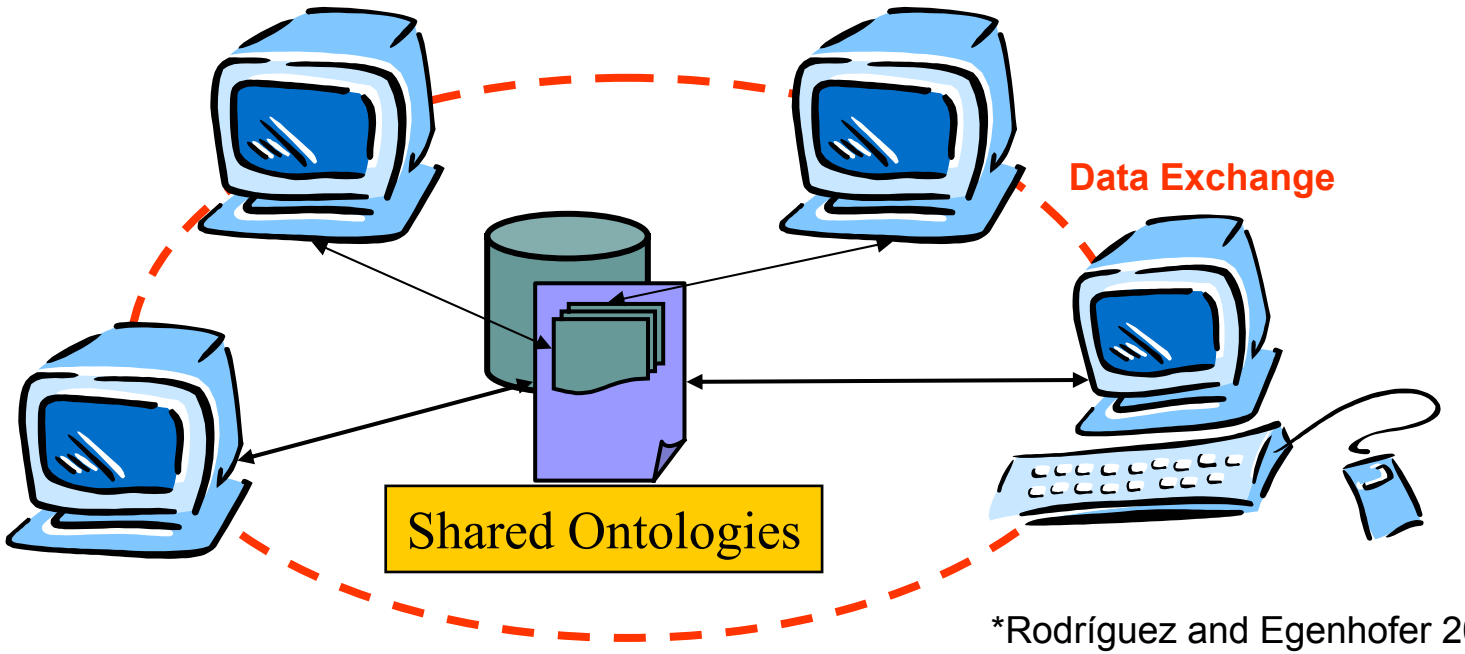
- Ontologies-based approaches have been suggested as a solution for information integration that achieves interoperability*.
- Two distinct approaches can be selected to achieve semantic integration:
  - The use of shared ontologies
  - The use of non-shared ontologies
- The general approach has been to map the local terms onto a shared ontology.

*Kashyap and Sheth 1994; Uschold and Gruninger 1996

Ontologies-based approaches
Shared Ontologies

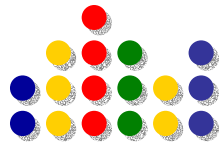


- Autonomous systems are required to commit to a shared ontology, and compromises are difficult to maintain when new concepts are added*.
- Even though a shared ontology ensures total integration, constructing such an ontology is costly, if not impractical.

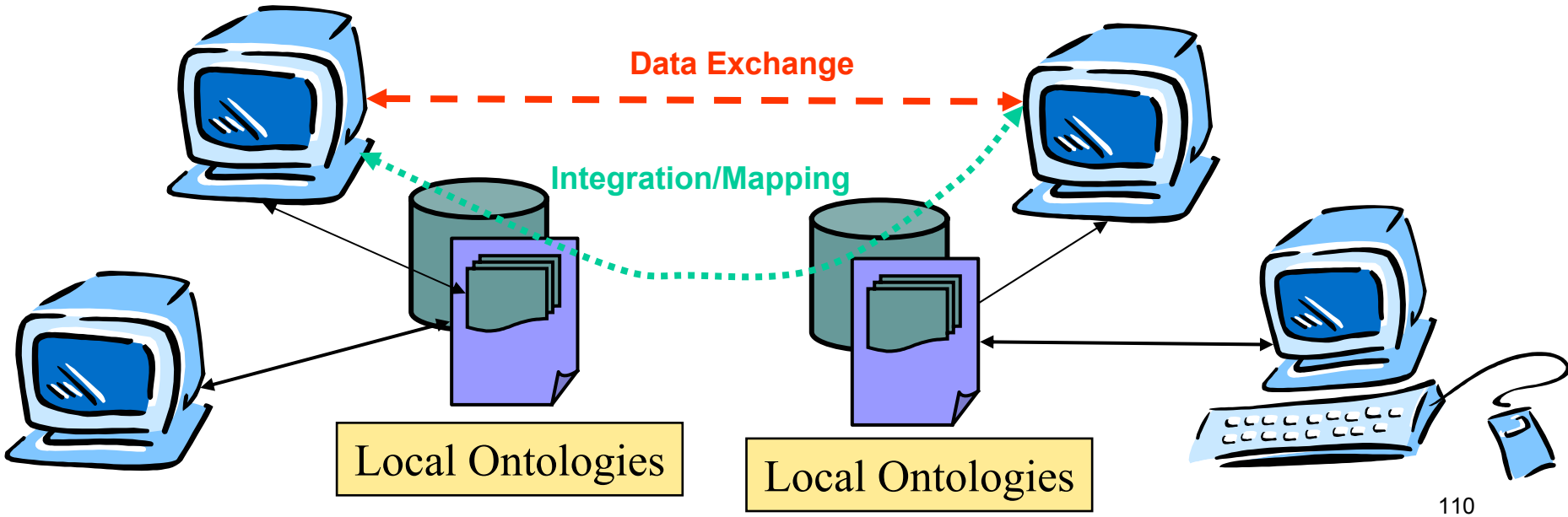


*Rodríguez and Egenhofer 2002

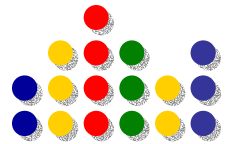
Ontologies-based approaches
Non-Shared Ontologies



- Since the Web is a distributed infrastructure with autonomous systems, it is not reasonable to expect that all the systems will commit to shared ontologies.
- Instead, autonomous systems will use non-shared ontologies.
- This will require the integration and mapping of ontologies.



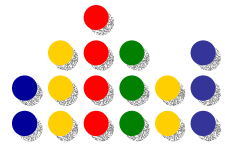
The Semantic Web



- The Web is “machine-readable” but not “machine-understandable”
- “The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation.”*

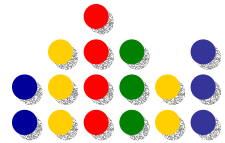
*Tim Berners-Lee, James Hendler and Ora Lassila, The Semantic Web, Scientific American, May 2001

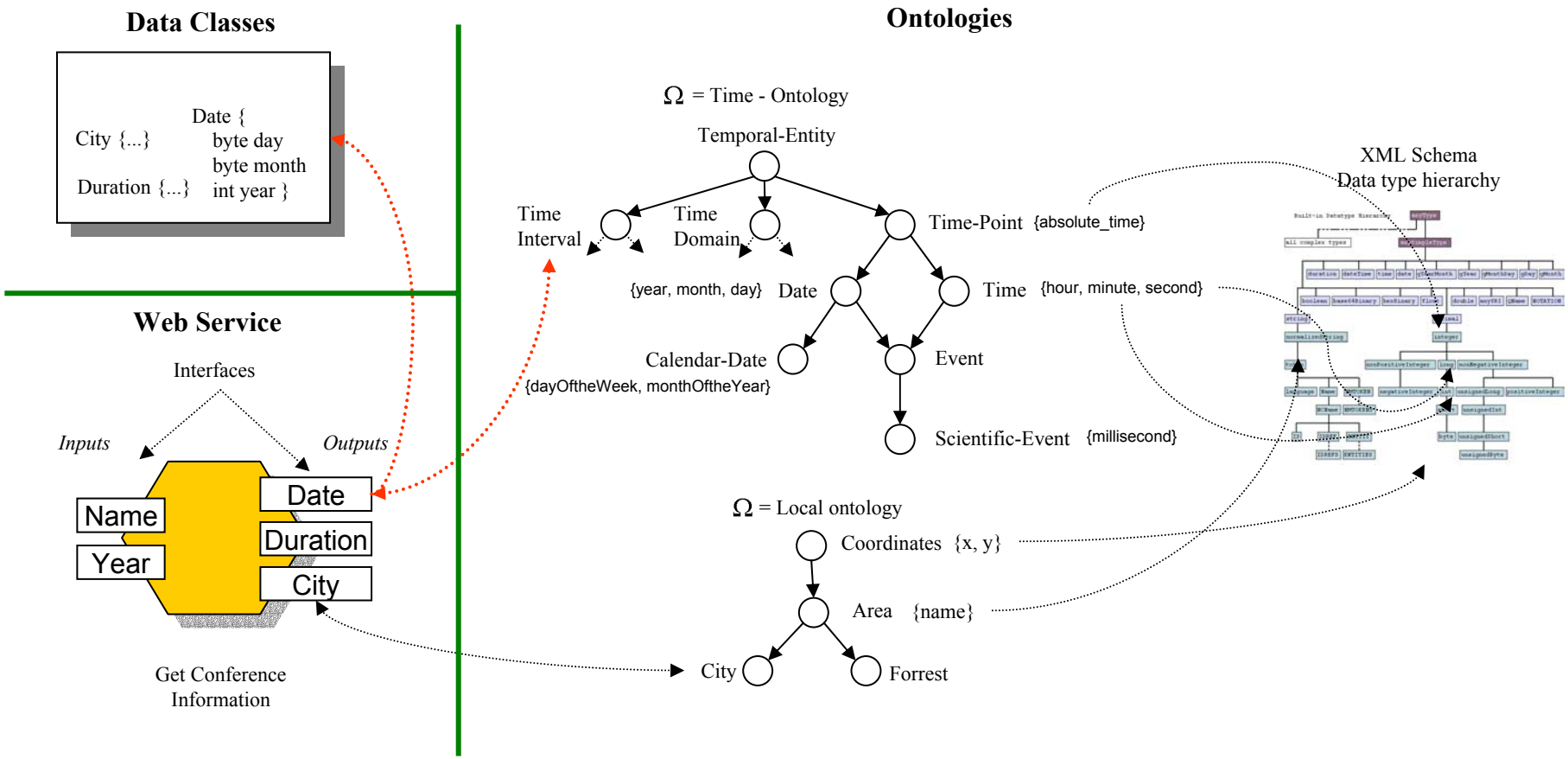
The use of semantics
Benefits



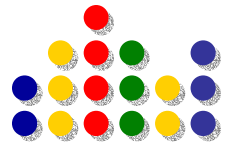
- Search engines can better “understand” the contents of a particular page
- More accurate searches
- Additional information aids precision
- Makes it possible to automate searches because less manual “weeding” is needed to process the search results
- Facilitates the integration of several Web services

Mapping Interfaces with Ontological Concepts





Building Ontologies
with Semantic Languages

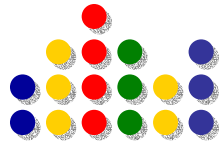


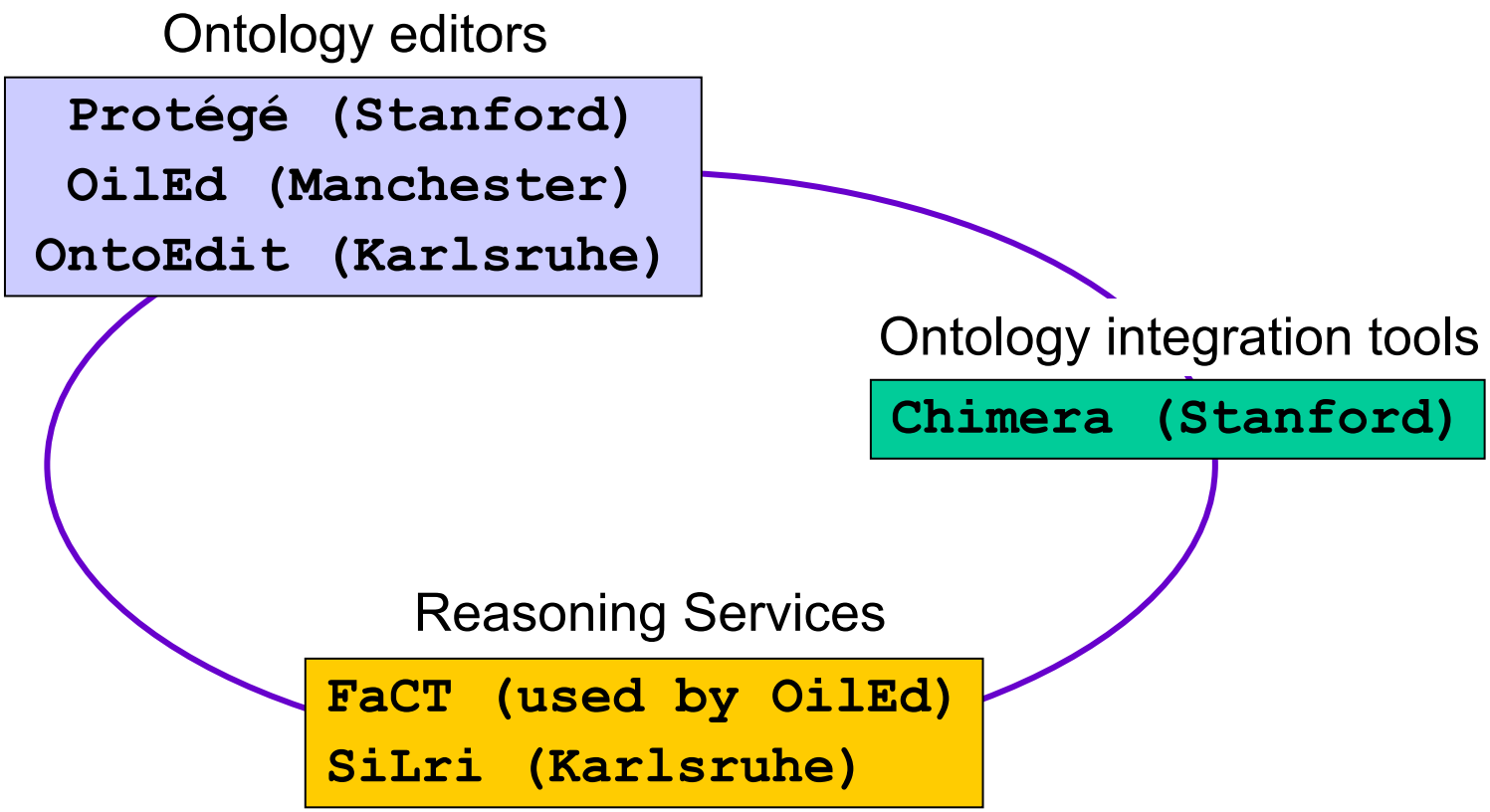
The ontologies deployed must allow the precise description of the data objects associated with Web services interfaces.

Some examples of indispensable features that ontologies must supply include:

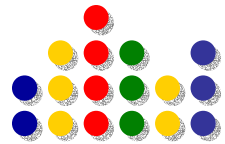
DAML+OIL	Data types, cardinality constraints, …
RDFS	Classes, inheritance, …
RDF	Nodes, relations, …

Ontologies
Tools



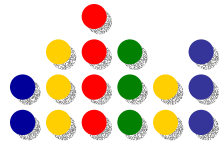


RDF
Basic features



- Provides basic ontological primitives
  - Resource Description Framework
  - An XML application
  - “Not just tags” – RDF makes use of a formal model
  - Basis for “The Semantic Web” (SW)
  - RDF provides a model for describing resources.
  - Resources have properties.

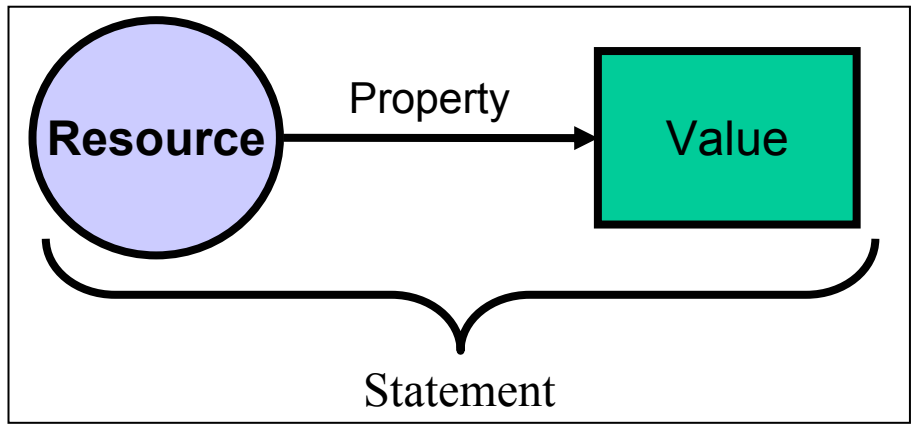
RDF
Data Model

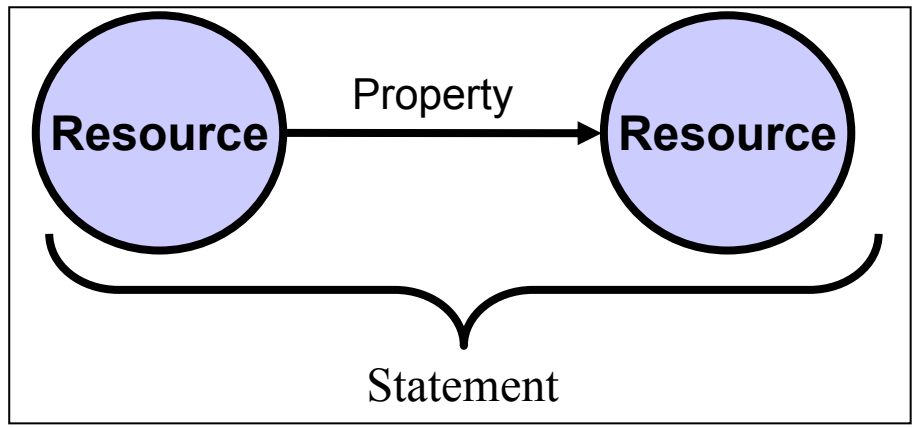


- Directed labeled graphs

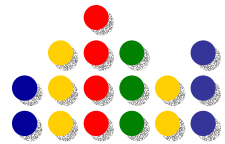
- Model elements
  - Resource
  - Property
  - Value
  - Statement

RDF triples assert facts about resources

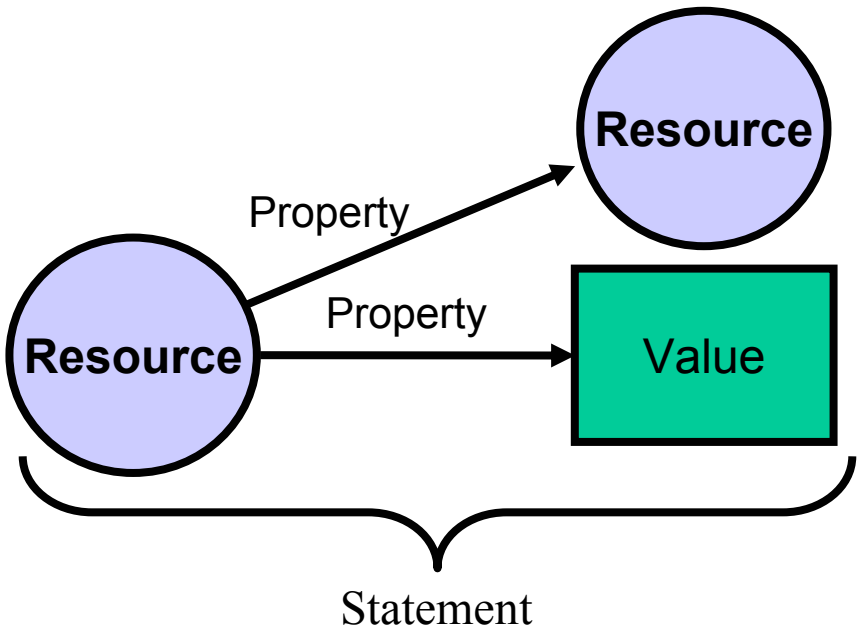




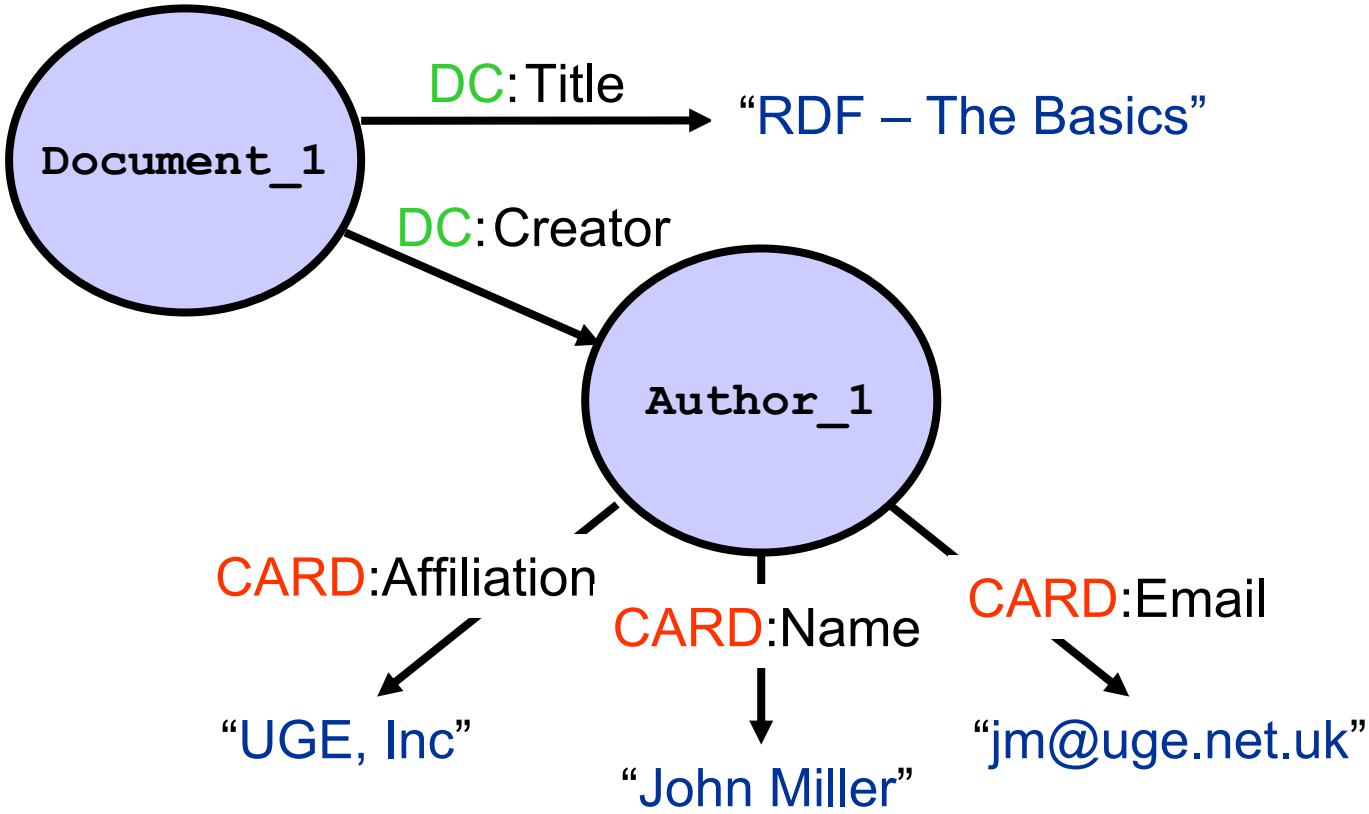
RDF
Data Model



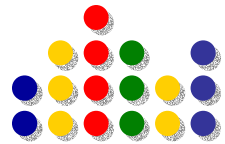
- The properties associated with resources are identified by property-types, and property-types have corresponding values.
- In RDF, *values* may be atomic in nature (text strings, numbers, etc.) or other resources, which in turn may have their own properties.



RDF Model
An Example

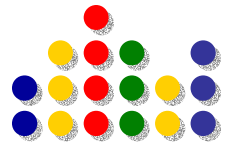


RDF
An Example - Syntax



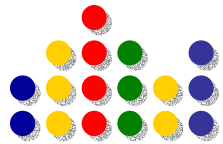
```
<RDF xmlns = "http://www.w3.org/TR/WD-rdf-syntax#"
     xmlns:DC = "http://purl.org/dc/elements/1.0/"
     xmlns:CARD = "http://person.org/BusinessCard/>

   <Description about = "Document_1">
     <DC:Title> RDF - The Basics </DC:Title>
     <DC:Creator>
       <Description>
       <CARD:Name>John Miller</CARD:Name>
       <CARD:Email>jm@uge.net</CARD:Email>
       <CARD:Affiliation>UGE, Inc.</CARD:Affiliation>
       </Description>
     </DC:Creator>
   </Description>
</RDF>
```

RDF Summary



- RDF is a general-purpose framework
- RDF provides structured, machine-understandable metadata for the Web
- RDF provides a model for describing resources. Provides basic ontological primitives
- Basis for “The Semantic Web” (SW)

RDF Schema (RDFS)
Extending the RDF



- Classes

```
<rdfs:Class rdf:ID="Staff" rdfs:comment="A Staff member at UGA ">
   <rdfs:subClassOf rdf:resource="&rdfs;Resource"/>
</rdfs:Class>
```

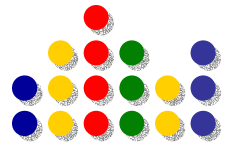
- Inheritance between classes

```
<rdfs:Class rdf:ID="Researcher" rdfs:comment="A Researcher at UGA">
   <rdfs:subClassOf rdf:resource="#Staff"/>
</rdfs:Class>
```

- Range

```
<rdf:Property rdf:ID="LName" rdfs:comment="Last Name of the Person">
  <rdfs:domain rdf:resource="#Staff"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

RDF Schema (RDFS)
Extending the RDF

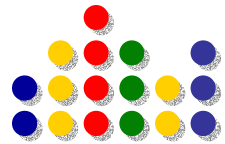


- Cardinality
  - No cardinality restrictions on properties
- Basic Datatypes
  - Only includes ‘literals’ which is the set of all strings

```
<rdf:Property rdf:ID="LName" rdfs:comment="Last Name of the Person">
  <rdfs:domain rdf:resource="#Staff"/>
  <rdfs:range rdf:resource="&rdfs;Literal"/>
</rdf:Property>
```

- Enumeration of property values
  - Not supported

DAML+OIL
Extending the RDFS

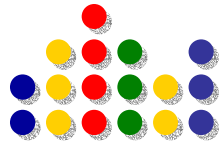


- DAML+OIL is the result of the fusion of DAML (DARPA Markup Language) developed in US and OIL (Ontology Inference Layer) funded by EU.
- DAML+OIL: a semantic markup language for Web resources which builds on earlier W3C standards such as RDF and RDF Schema, and extends these languages with richer modelling primitives. See

  http://www.w3.org/TR/daml+oil-walkthru/

  http://www.w3.org/TR/daml+oil-reference

DAML+OIL
Extending the RDFS



- Two kinds of properties are defined
  - Object Properties

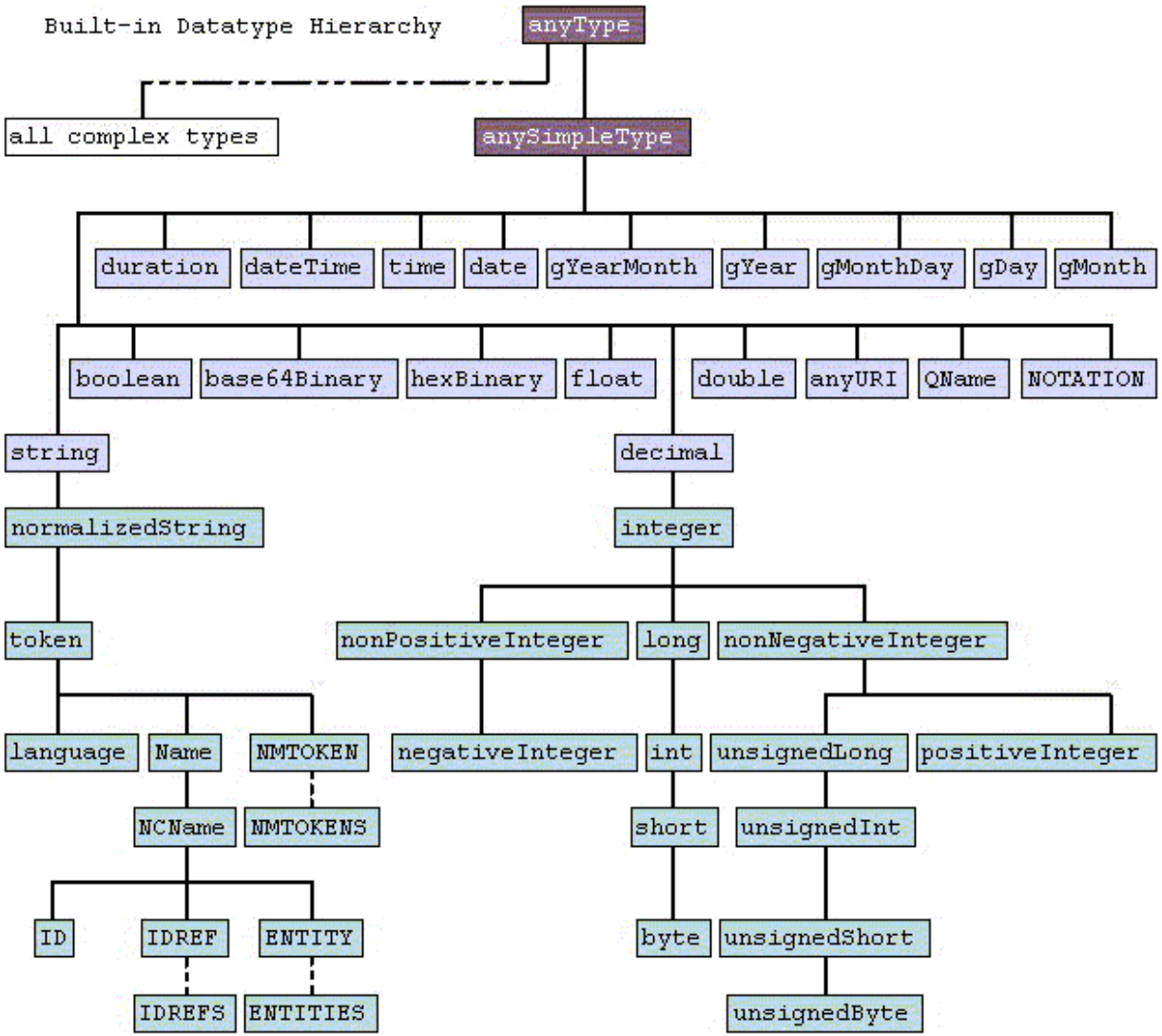
```
<!-- Relating one object to another object -->
<rdf:ObjectProperty rdf:ID="Project"/>
  <rdfs:domain rdf:resource="#Staff" />
  <rdfs:range rdf:resource="#Project"/>
</daml:ObjectProperty>
```

  - Datatype Properties

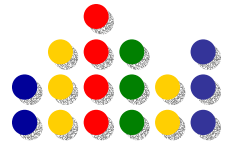
```
<!ENTITY xsd 'http://www.w3.org/2001/XMLSchema#'>
…
<!-- Relating an object to a primitive datatype -->
<daml:DatatypeProperty rdf:ID="StartDate">
  <rdfs:domain rdf:resource="#Intern" />
  <rdfs:range rdf:resource="&xsd;date"/>
</daml:DatatypeProperty>
```

XMLSchema Datatypes

Datatype hierarchy



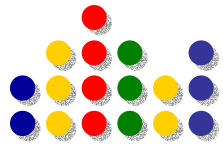
DAML+OIL
Extending the RDFS



- Cardinality (minCardinality, maxCardinality, cardinality)

```
<!-- DAML uses rdf Classes -->
 <rdfs:Class rdf:ID="Staff">

     <rdfs:subClassOf>
        <!-- Minimum 1 Email required (minCardinality) -->
         <daml:Restriction daml:minCardinalityQ="1">
             <daml:onProperty rdf:resource="#EMail"/>
             <daml:toClass rdf:resource="&xsd;String"/>
         </daml:Restriction>
     </rdfs:subClassOf>

     <!-- Restricting the cardinality of the property -->
     <rdfs:subClassOf>
         <daml:Restriction daml:cardinality="1">
             <daml:onProperty rdf:resource="#FName"/>
         </daml:Restriction>
     </rdfs:subClassOf>
 </rdfs:Class>
```

DAML+OIL
Extending the RDFS



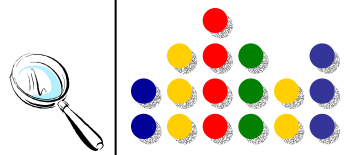
- Basic Datatypes
  - Refer to the XMLSchema URI

```
<!ENTITY xsd 'http://www.w3.org/2001/XMLSchema#'>
```

- Enumeration

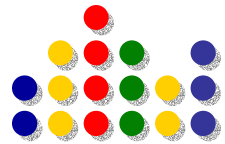
```
<daml:Class rdf:ID="ValidityType">
  <daml:oneOf rdf:parseType="daml:collection">
    <ValidityType rdf:ID="Valid"/>
    <ValidityType rdf:ID="Expired"/>
    <ValidityType rdf:ID="InvalidCCNumber"/>
    <ValidityType rdf:ID="InvalidCCType"/>
    <ValidityType rdf:ID="AuthorizationRefused"/>
  </daml:oneOf>
</daml:Class>
```

Web Service
QoS Specification



- The specification of Web services operational metrics allows the analysis and computation processes QoS.
- Therefore, processes can be designed according to QoS objectives and requirements.
- This allows organizations to translate their strategies into their processes more efficiently.

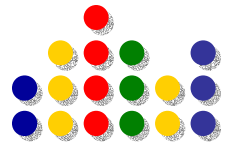
Operational Metrics



- DAML-S does not supply a QoS model that allow the automatic computation of Web processes
- The operational metrics of tasks and Web services are described using a QoS model.
- We have developed a theoretical model for the automatic computation of workflow QoS based on tasks QoS metrics*.
- Based on our model, we have developed an ontology for the specification of QoS metrics for tasks and Web services.
- This information will allow for the discovery of Web services based on operational metrics.

*Cardoso et al., 2002a, Cardoso et al., 2002b

Web Service Flow Language (WSFL)



- WSFL is IBM's XML language for describing Web Services Composition
- Can be extended WSFL to include QoS specification such as time, cost and reliability
- Constructs
  - Activity Elements

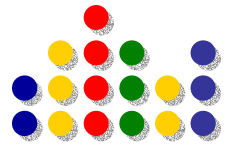
QoS specification (time,cost,reliability)

```
<activity  name="BarnesGetPrice" X1="102.0" Y1="184.0" X2="227.0" Y2="230.0"
           time="1833" cost="200" reliability=".7" type="Facility">

        <input message="BarnesInput"/>
        <output message="BarnesOutput"/>
        <performedBy serviceProvider="BarnesNobleSP">
                <implement>
                        <export>
                                <plugLink>
                                        <target PortType="BarnesNoblePT" Operation="getPrice"/>
                                </plugLink>
                        </export>
                </implement>
        </performedBy>
</activity>
```

  - Message Elements

```
<message name="BarnesInput">
        <part name="isbn"  element="string"/>
</message>
```

WSFL (contd.)



- **Service Provider Elements**

Web service information

```
<serviceProvider  name="BarnesNobleSP">
                <locator type="static"
                 service="http://www.xmethods.net/sd/2001/BNQuoteService.wsdl"/>
</serviceProvider>
```

- **Control Link Elements**

Conditional Branching

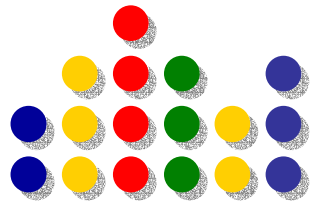
```
<controlLink  name="cl4" source="CheckCredit" target="CheckInventory"
              condition="creditResult!=0" probability=".5"
              X1="336.0" Y1="184.0" X2="368.0" Y2="154.0"/>
```

- **Data Link Elements**

Data Routing

```
<dataLink  name="dl1" X1="227.0" Y1="207.0" X2="251.0" Y2="207.0"
                      source="BarnesGetPrice" target="CheckCredit">
        <mapInfo sourcePart="price" targetPart="price"/>
</dataLink>
```

Process Specification
BPEL4WS



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

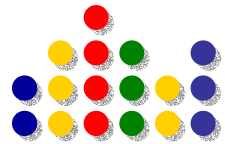
[1]LSDIS Lab, Computer Science, University of Georgia

[2]Oracle Corporation

[3] Universität Innsbruck

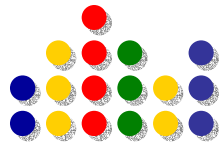
[4] Semagix, Inc

BPEL4WS
Introduction



- BPEL4WS (Business Process Execution Language for Web Services) is a process modeling language.
  - Developed by IBM, Microsoft, and BEA
  - Version 1.0, 31 July 2002
- It represents the merging of XLANG (Microsoft) and WSFL(IBM).
- It is build on top of WSDL.
  - For descriptions of what services do and how they work, BPEL4WS references port types contained in WSDL documents.

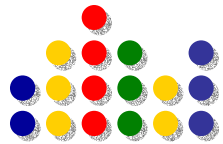
BPEL4WS
Introduction



- BPEL4WS was released along with two others specs:
  - WS-Coordination and WS-Transaction*.
- WS-Coordination describes how services can make use of pre-defined coordination contexts to subscribe to a particular role in a collaborative activity.
- WS-Transaction provides a framework for incorporating transactional semantics into coordinated activities.

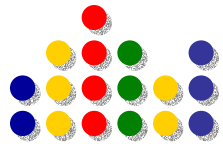
*http://www-106.ibm.com/developerworks/webservices/library/ws-coor/, http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/

BPEL4WS
Introduction



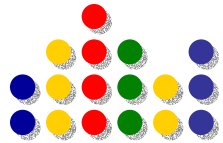
- BPEL4WS is a block-structured programming language, allowing recursive blocks but restricting definitions and declarations to the top level.
- The language defines activities as the basic components of a process definition.
- Structured activities prescribe the order in which a collection of activities take place.
  - Ordinary sequential control between activities is provided by sequence, switch, and while.
  - Concurrency and synchronization between activities is provided by flow.
  - Nondeterministic choice based on external events is provided by pick.

BPEL4WS
Introduction

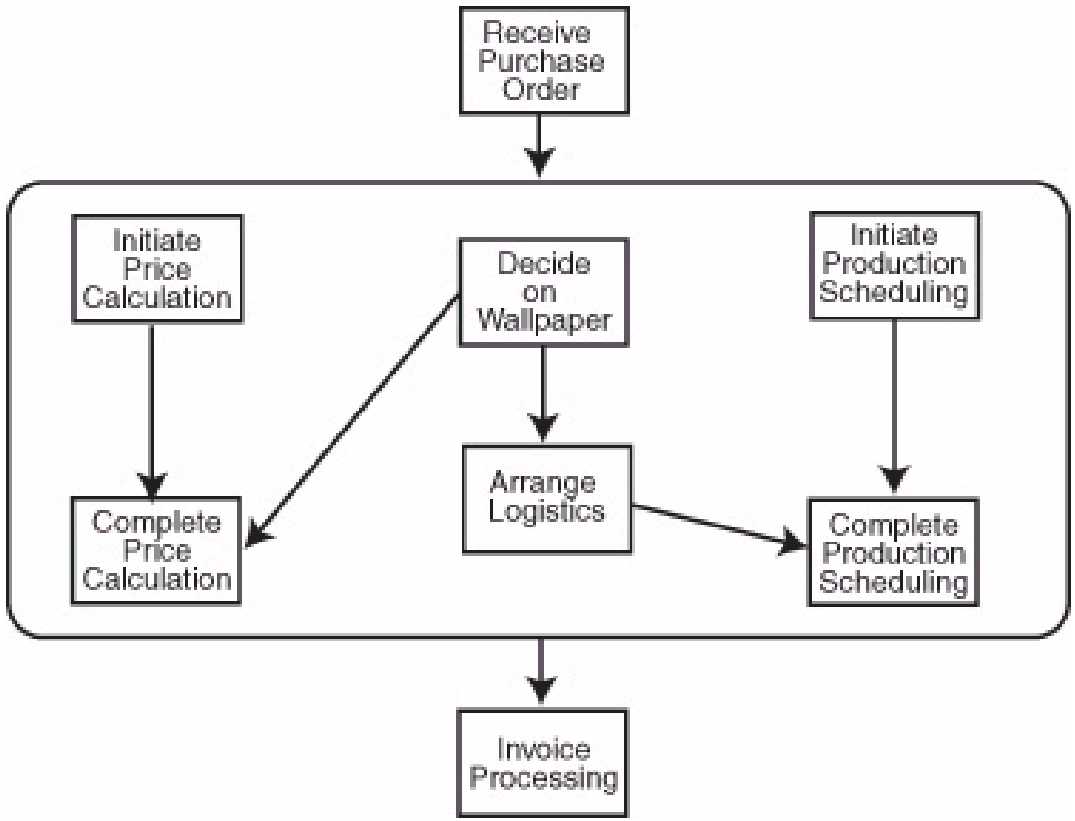


- Process instance-relevant data (containers) can be referred to in routing logic and expressions.
- BPEL4WS defines a mechanism for catching and handling faults similar to common programming languages, like Java.
- One may also define a compensation handler to enable compensatory activities in the event of actions that cannot be explicitly undone.
- BPEL4WS does not support nested process definition.

BPEL4WS
An Example



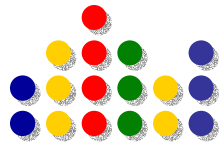
Let consider the following process.



[http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/]

BPEL4WS

An Example – WSDL definitions



```
<definitions targetNamespace="http://manufacturing.org/wsdl/purchase"
      xmlns:sns="http://manufacturing.org/xsd/purchase"
…
<message name="POMessage">
   <part name="customerInfo" type="sns:customerInfo"/>
   <part name="purchaseOrder" type="sns:purchaseOrder"/>
</message>
…
<message name="scheduleMessage">
   <part name="schedule" type="sns:scheduleInfo"/>
</message>

<portType name="purchaseOrderPT">
   <operation name="sendPurchaseOrder">
      <input message="pos:POMessage"/>
      <output message="pos:InvMessage"/>
      <fault name="cannotCompleteOrder"
             message="pos:orderFaultType"/>
   </operation>
</portType>
…
<slnk:serviceLinkType name="purchaseLT">
   <slnk:role name="purchaseService">
       <slnk:portType name="pos:purchaseOrderPT"/>
   </slnk:role>
</slnk:serviceLinkType>
…
</definitions>
```

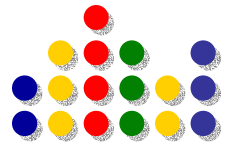
Messages

The WSDL portType offered by the service to its customer

Roles

BPEL4WS

An Example – The process



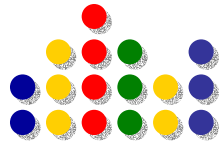
```
<process name="purchaseOrderProcess"
         targetNamespace="http://acme.com/ws-bp/purchase"
…
   <partners>
      <partner name="customer"
               serviceLinkType="lns:purchaseLT"
               myRole="purchaseService"/>
   …
   </partners>

   <containers>
      <container name="PO" messageType="lns:POMessage"/>
      <container name="Invoice"
                 messageType="lns:InvMessage"/>
   …
   </containers>

   <faultHandlers>
      <catch faultName="lns:cannotCompleteOrder"
             faultContainer="POFault">
         <reply   partner="customer"
                  portType="lns:purchaseOrderPT"
                  operation="sendPurchaseOrder"
                  container="POFault"
                  faultName="cannotCompleteOrder"/>
      </catch>
   </faultHandlers>
…
```

This section defines the different parties that interact with the business process in the course of processing the order.

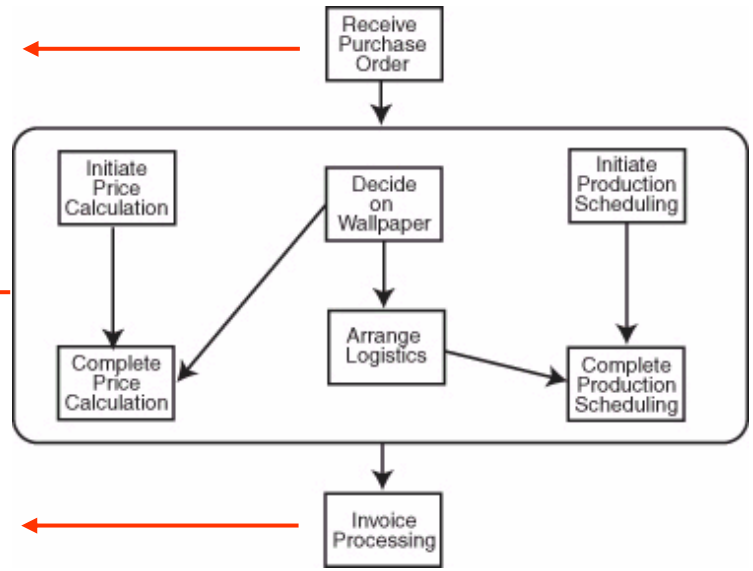
This section defines the data containers used by the process, providing their definitions in terms of WSDL message types.

This section contains fault handlers defining the activities that must be executed in response to faults.

BPEL4WS
An Example – The process

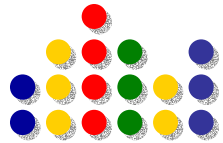


```
…
    <sequence>

      <receive partner="customer"
               portType="lns:purchaseOrderPT"
               operation="sendPurchaseOrder"
               container="PO">
      </receive>

      <flow>
      …
      </flow>

      <reply partner="customer"
             portType="lns:purchaseOrderPT"
             operation="sendPurchaseOrder"
             container="Invoice"/>
   </sequence>

</process>
```



BPEL4WS

An Example – The process



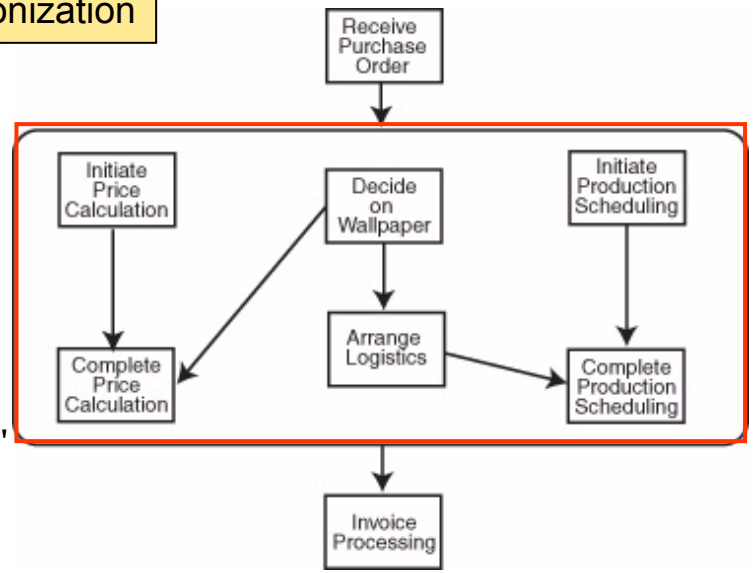
```
<flow>
        <links>
           <link name="ship-to-invoice"/>
           <link name="ship-to-scheduling"/>
        </links>

        <sequence>
         …

           <invoke  partner="shippingProvider"
                    portType="lns:shippingPT"
                    operation="requestShipping"
                    inputContainer="shippingRequest"
                    outputContainer="shippingInfo">
              <source linkName="ship-to-invoice"/>
           </invoke>

           <receive partner="shippingProvider"
                    portType="lns:shippingCallbackPT"
                    operation="sendSchedule"
                    container="shippingSchedule">
              <source linkName="ship-to-scheduling"/>
           </receive>
        </sequence>
…
<flow>
```

The flow construct provides concurrency and synchronization

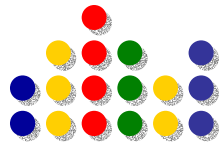
Activities are executed sequentially

Activity Call

Activity call



BPEL4WS vs. DAML-S
Comparison

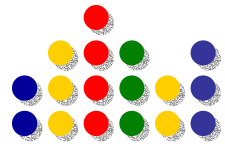


- BPEL4WS relates closely to the ServiceModel (Process Model) component of DAML-S.

- DAML-S defines preconditions and effects
  - This enables the representation of side effects of Web services.
  - It also enables a better reasoning about the composition of services.

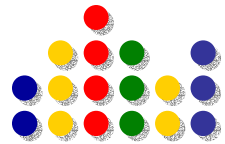
- DAML-S classes provide a richer representation of services
  - Classes allow reasoning draw properties from inheritance and other relationships to other DAML-S classes.

BPEL4WS vs. DAML-S
Comparison



- The DAML-S ServiceProfile and ServiceModel provide sufficient information to enable
  - The automated discovery, composition, and execution based on well-defined descriptions of a service's inputs, outputs, preconditions, effects, and process model.
- BPEL4WS has complicated semantics for determining whether an activity actually happens in a block.
- BPEL4WS defines mechanisms for catching and handling faults and for setting compensation handlers.
- BPEL4WS includes WS-Coordination and WS-Transaction to provide a context for pre-defined transactional semantics.

References



http://www.daml.org/services/

http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/

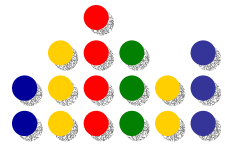
http://www.daml.org/2001/03/daml+oil-index

http://www-106.ibm.com/developerworks/webservices/library/ws-coor/

http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/

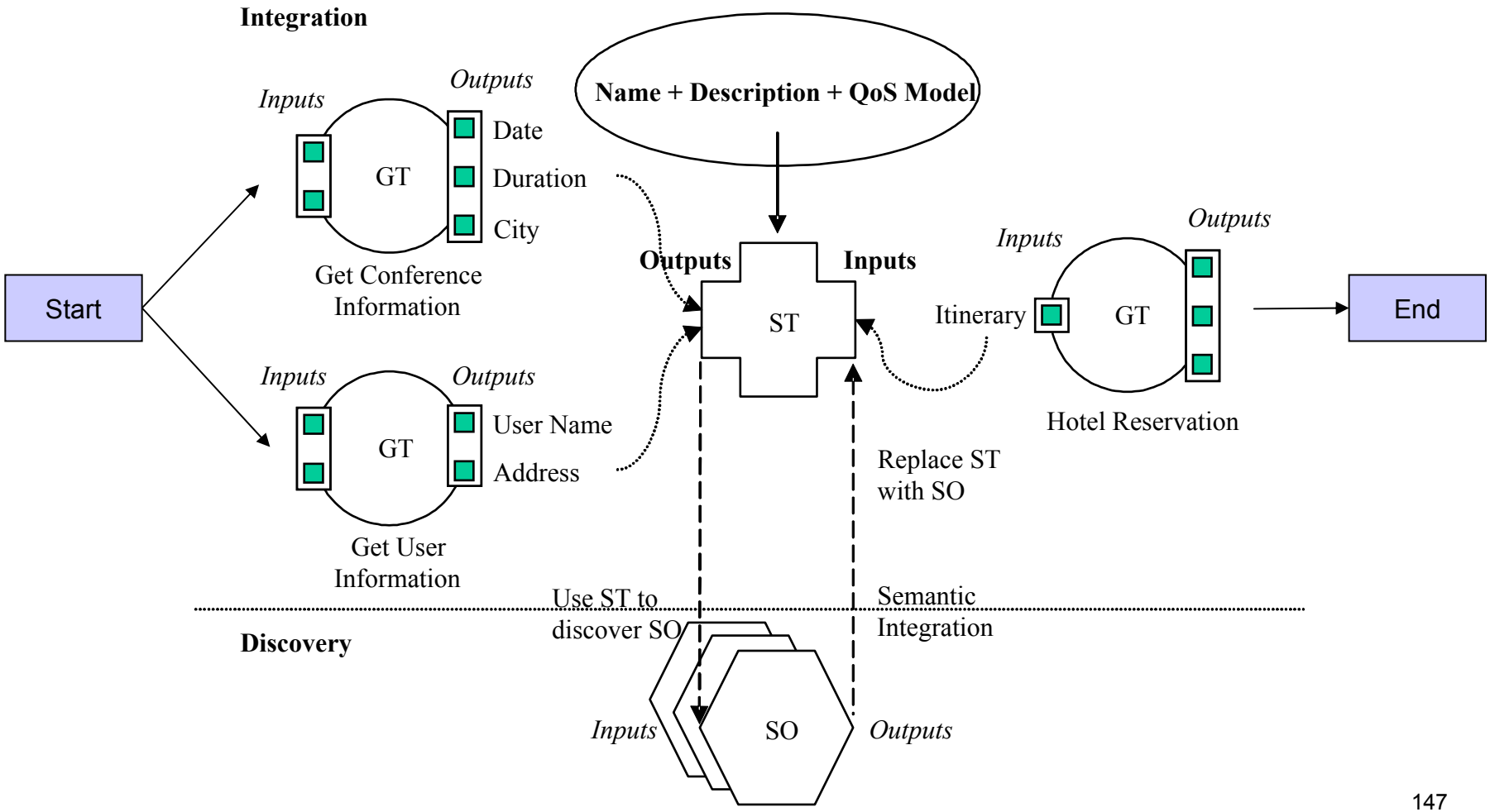
http://www.ksl.stanford.edu/projects/DAML/Webservices/DAMLS-BPEL.html

The Composition Process
Definitions

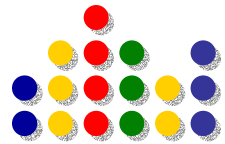


- Traditional workflow tasks and Web service tasks already associated with a process and therefore with a realization are called grounded tasks (GT).
- When the designer wishes to add a Web service to an Web process, a service template (ST) is created, indicating his intention to extend the functionality of the process.

The composition process
GT and ST Examples



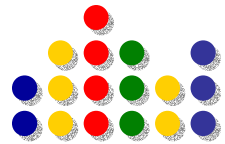
The Composition Process
Steps



- Once a ST is created, it is sent to the Web service discovery module
- The ST is employed to find an appropriate Web service.
- The discovery module returns a set of service object (SO) references that are ranked according to their degree of similarity with the service template.

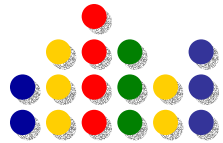
The Composition Process

Steps

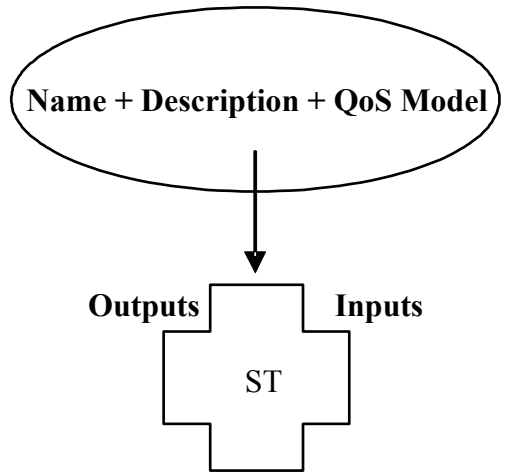


- SOs can be ranked according to a syntactical, operational, or semantic perspective.
- The designer then selects the most appropriate SO to accomplish his objectives.
- Additionally, a set of data mapping is presented to the designer suggesting a possible interconnection among the newly added task interfaces and the grounded task interfaces.

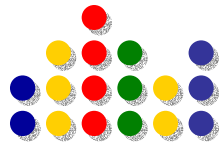
ST Structure



- A ST has five sections that need to be specified:
  - The name of the Web service to be found,
  - Its textual description,
  - Its operational metrics,
  - The set of outputs parameters from the grounded tasks that will be connected to SO inputs, and
  - The set of input parameters from the grounded tasks that a SO will be connected to.

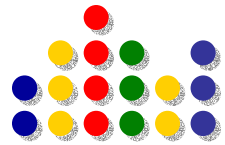


SO Structure



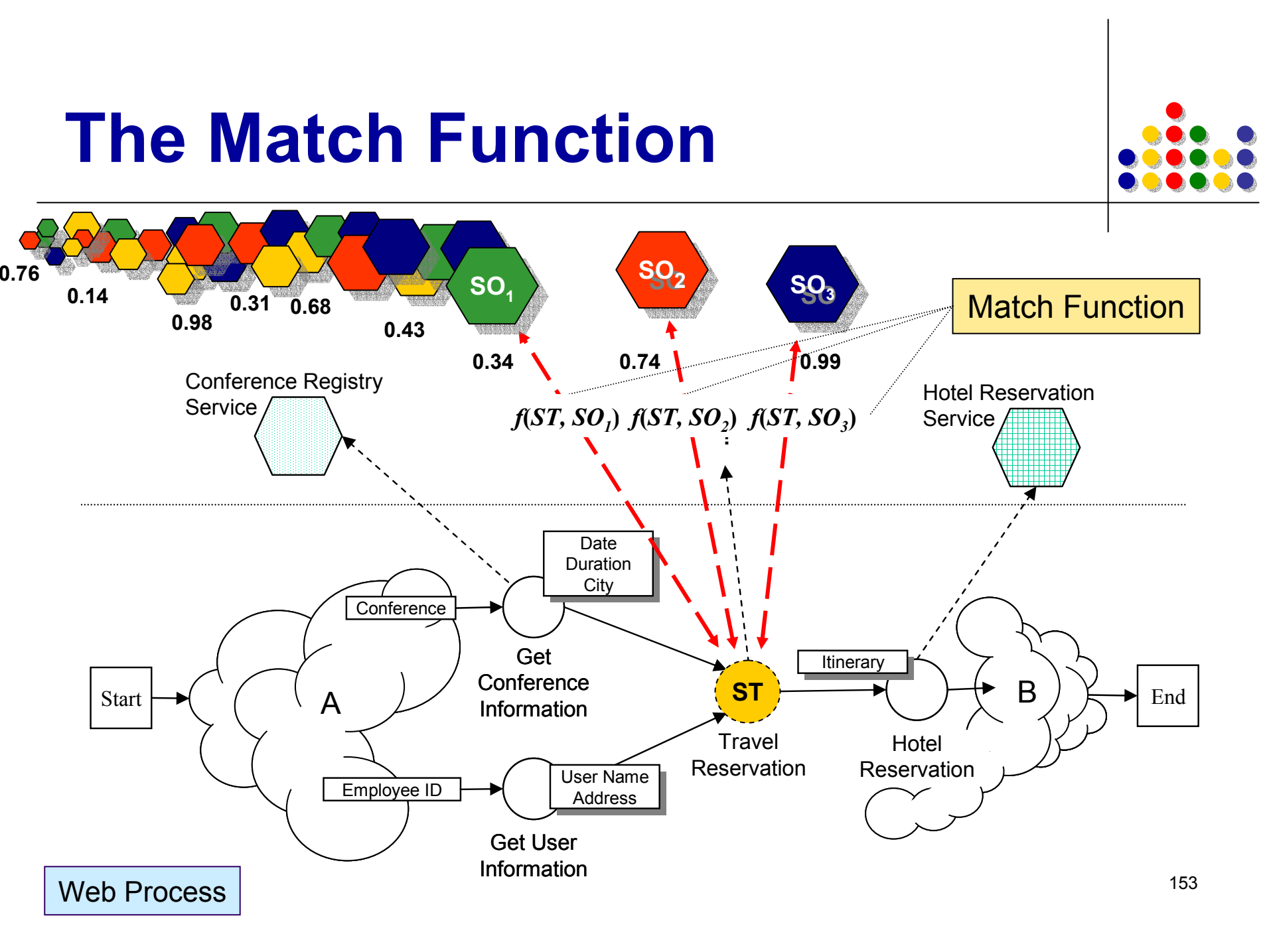
- A SO structure has also five sections:
  - Its name,
  - Its textual description,
  - Its operational metrics,
  - The set of outputs parameters, and
  - A set of input parameters.

The Match Function

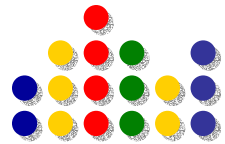


- The Web service discovery and integration process is carried out by a key operation:
  - The match function.
- The matching step is dedicated to finding correspondences between a service template (ST, *i.e.*, a query) and a service object (SO).

The Match Function

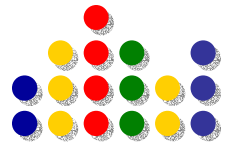


The Match Function



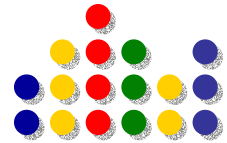
- The match function uses syntactic, operational, and semantic information as a way to increase the precision of the match.
- There types of similarity are evaluated:
  - Syntactic Similarity
  - Operational Similarity
  - Semantic Similarity

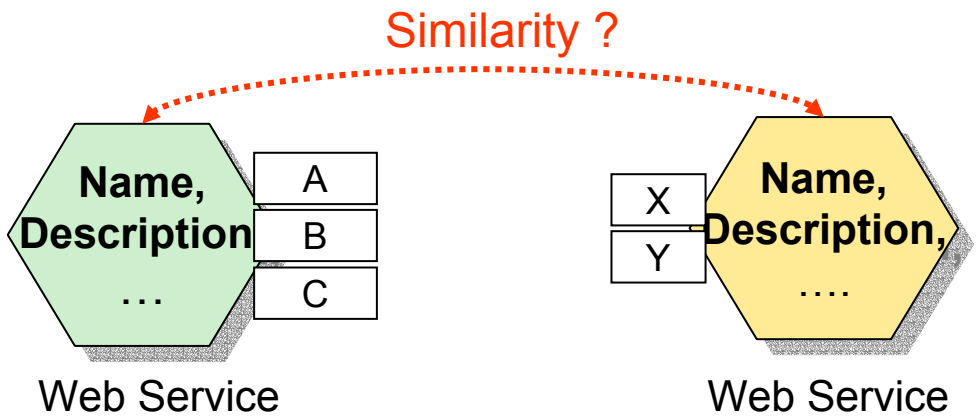
The Match Function
Syntactic Similarity



- The syntactic similarity of a ST and a SO is based on their *service names* and *service descriptions*.
- Additional fields can be compared.
- At this stage, only syntactic information is taken into account, since the fields are simply expressed using a set of words.
- No tags or concepts are attached to the words used.

The Match Function
Syntactic Similarity

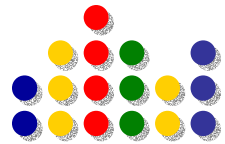




$$SynSimilarty(ST, SO) = \frac{\omega_1 SynNS(ST.sn, SO.sn) + \omega_2 SynDS(ST.sd, SO.sd)}{\omega_1 + \omega_2} \in [0..1],$$

$$\text{and } \omega_1, \omega_2 \in [0..1]$$

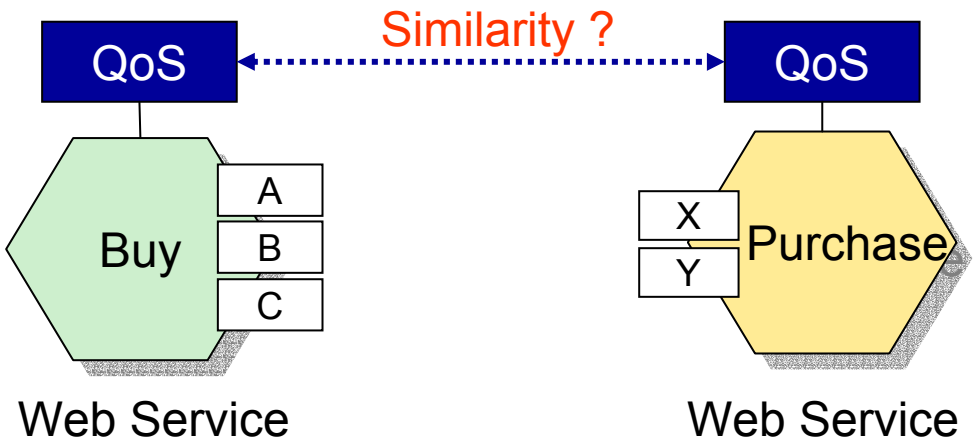
The Match Function
Operational Similarity



- Syntactic and semantic information allows for the selection of Web services based on their functionality*, but without accounting for operational metrics.
- The operational similarity of a ST and a SO is calculated based on the metrics specified in their QoS model.
- The purpose is to determine how close two Web services are, based on their operational capabilities.

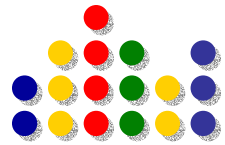
*We recognize that additional research is necessary to specify the functionally of Web services

The Match Function
Operational Similarity



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

The Match Function
Operational Similarity

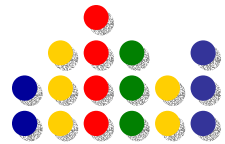


$$\text{OpSimilarity}(ST, SO) = \sqrt[3]{\text{QoSdimD}(ST, SO, time) * \text{QoSdimD}(ST, SO, cost) * \text{QoSdimD}(ST, SO, reliability)}$$

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

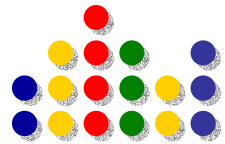
$$\text{dcd}_{\min}(ST, SO, dim) = 1 - \frac{|\min(SO.qos(dim)) - \min(ST.qos(dim))|}{\min(ST.qos(dim))}$$

The Match Function
Semantic Similarity



- Purely syntactical methods that treat terms in isolation from their contexts.
  - It is insufficient since they deal with syntactic but not with semantic correspondences
  - Users may express the same concept in different ways.
- Therefore, we rely on semantic information to evaluate the similarity of concepts that define ST and SO interfaces.
- This evaluation will be used to calculate their degree of integration.

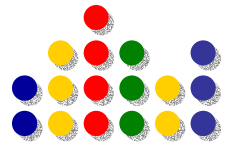
The Match Function
Semantic Similarity



- When comparing an output with an input two main cases can occur:
  - The concepts are defined with the same Ontology ($\Omega(O) = \Omega(I)$)
  - The concepts are defined in different Ontologies ($\Omega(O) \neq \Omega(I)$)

The Match Function

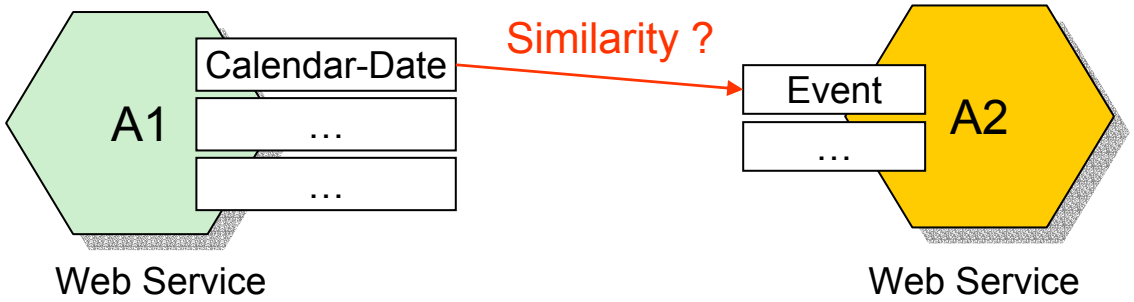
Semantic Similarity ($\Omega(O) = \Omega(I)$)

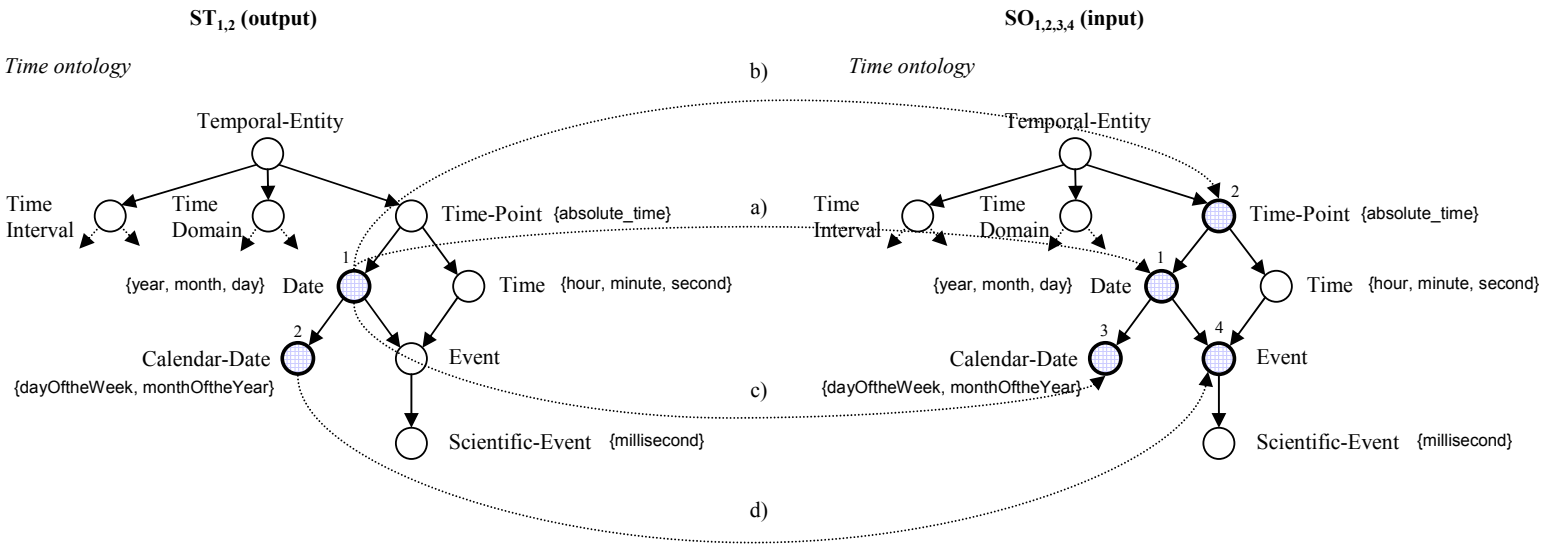


- When comparing concepts defined with the same ontology four distinct scenarios need to be considered:
  - a) the concepts are the same ($O=I$)
  - b) the concept I subsumes concept O ($O>I$)
  - c) the concept O subsumes concept I ($O<I$), or
  - d) concept O is not directly related to concept I ($O \neq I$).

The Match Function

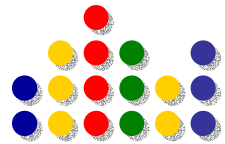
Semantic Similarity ($\Omega(O) = \Omega(I)$)





The Match Function

Semantic Similarity ($\Omega(O) = \Omega(I)$)

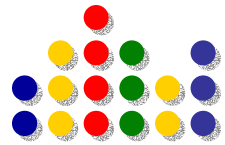


$$SemS'(O,I) = \begin{cases} 1, & O = I \\ 1, & O > I \\ \dfrac{|\,p(O)\,|}{|\,p(I)\,|}, & O < I \\ Similarity'(O,I), & O \neq I \end{cases}$$

$$similarity'(O,I) = \sqrt{\frac{|\,p(O) \cap p(I)\,|}{|\,p(O) \cup p(I)\,|} * \frac{|\,p(O) \cap p(I)\,|}{|\,p(I)\,|}}$$

The Match Function

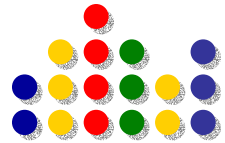
Semantic Similarity ($\Omega(O) \neq \Omega(I)$)

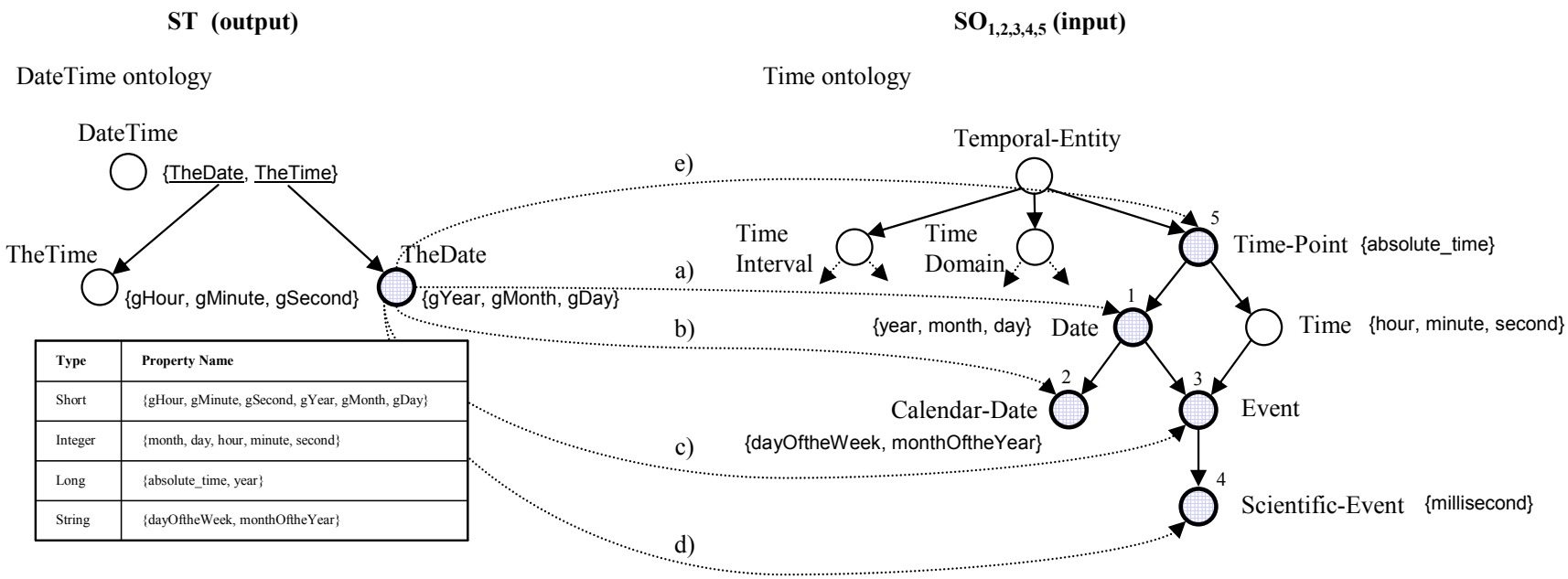


- When comparing concepts defined with different ontologies three distinct scenarios can occur:
  - The ontological properties involved are associated with a primitive data type
  - The properties are associated with concept classes, and
  - One property is associated with a primitive data type, while the other is associated with a concept class.

The Match Function

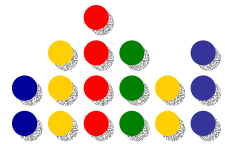
Semantic Similarity ($\Omega(O) <> \Omega(I)$)





The Match Function

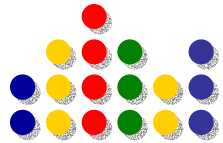
Semantic Similarity ($\Omega(O) \neq \Omega(I)$)



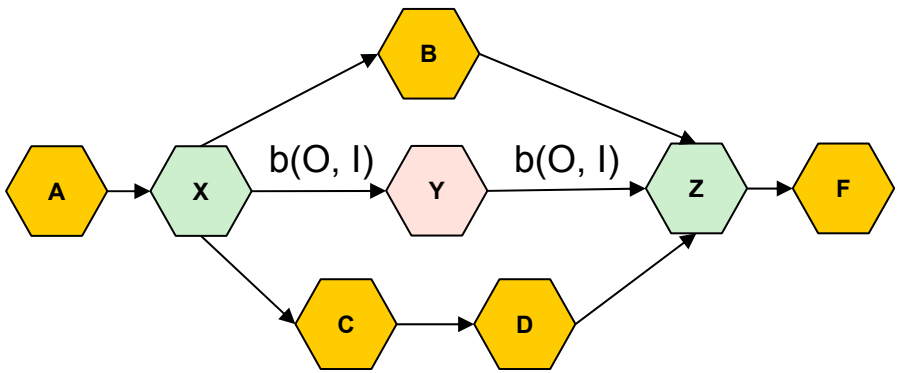
$$S(o,i) = \begin{cases} \sqrt[3]{SemDS(d(o),d(i)) * SynS(n(o),n(i)) * SemRS(r(o),r(i))}, & o \text{ and } i \text{ are primitive types} \\ SemDS(o,i), & o \text{ and } i \text{ are concept classes} \\ f(o,i), & \text{otherwise} \end{cases}$$

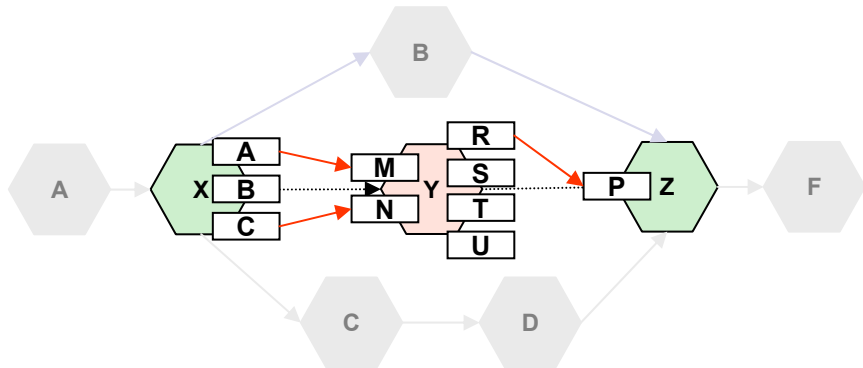
$$SemRS(or,ir) = \begin{cases} 1, & or = ir \\ 1, & or = \text{integer}, ir = string \\ 2/3, & or = \text{long}, ir = \text{integer} \\ 1/3, & or = \text{double}, ir = \text{integer} \\ 1, & or = \text{integer}, ir = \text{long} \\ 0, & \text{otherwise} \end{cases}$$

Web Services
Integration



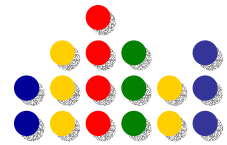
- The degree of integration of a Web service is evaluated using semantic information.
- For each interface to integrate we construct a bipartite graph with a bipartition b(O, I).
- Each edge has a weight (semantic similarity).
- We then compute the optimal matching*.

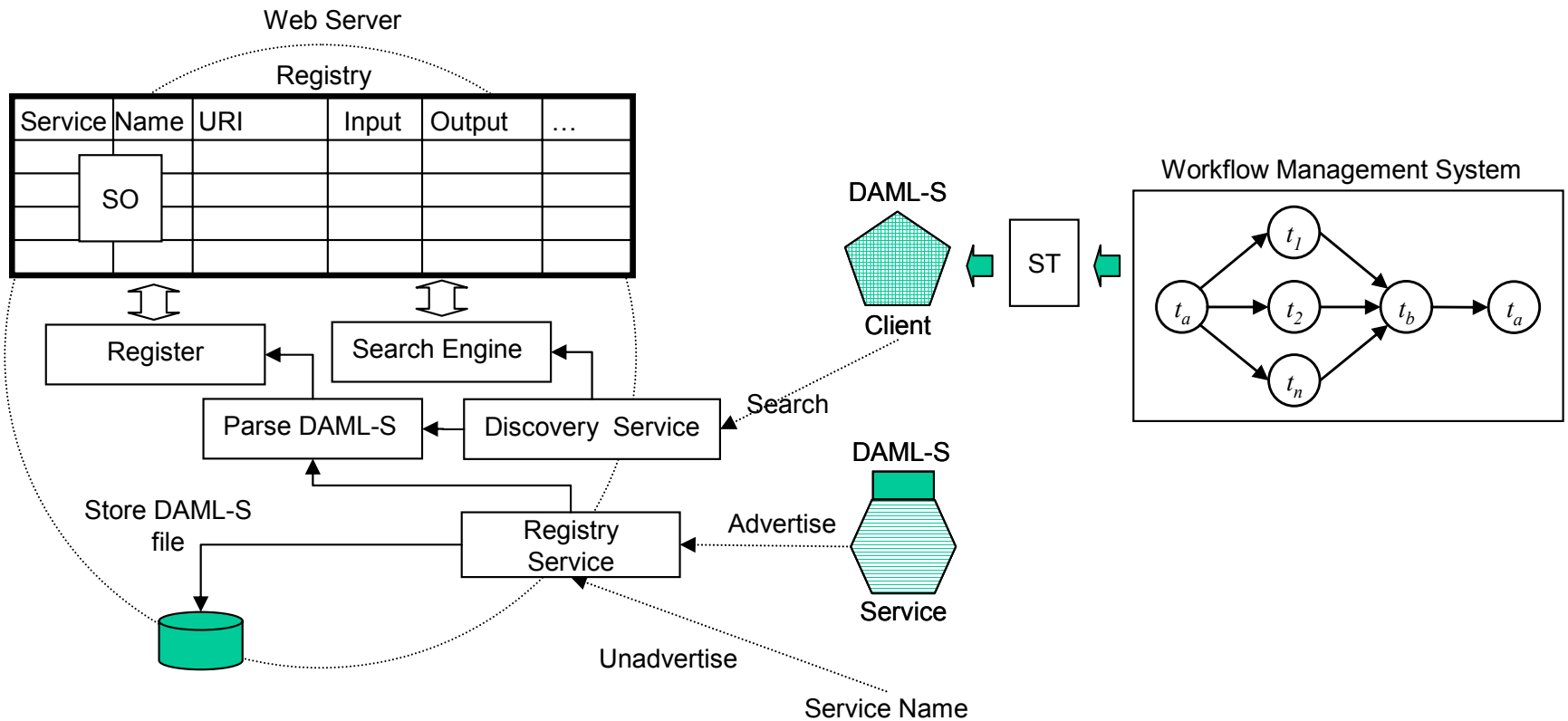




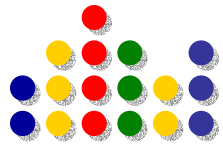
*Bondy and Murty 1976

System Architecture

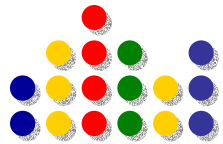


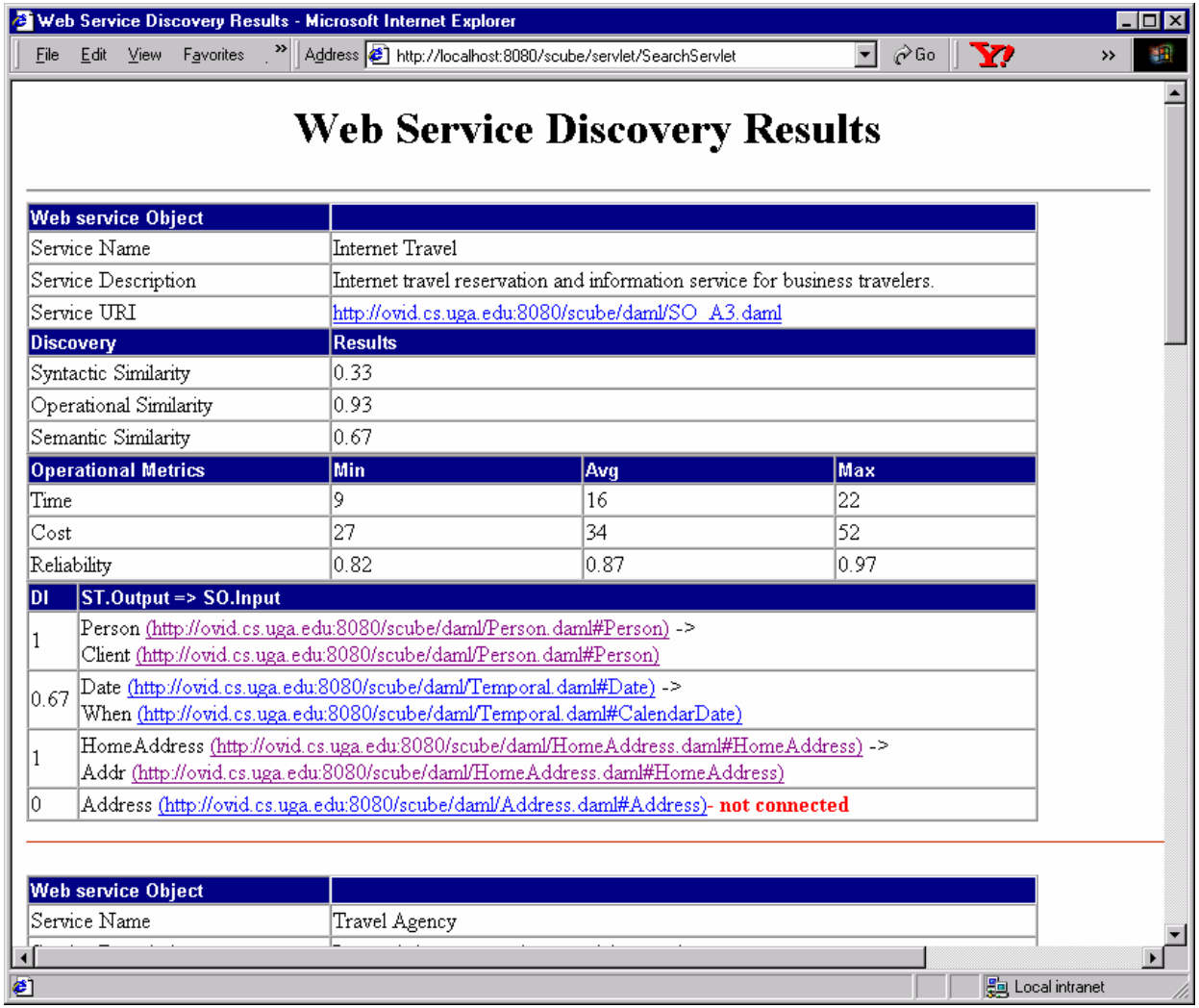


Discovery
Example of a Query

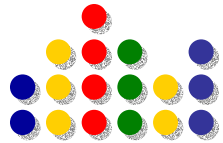


Discovery and Integration
Query Results





What's next?



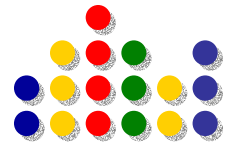
- ✓ We have found the a set of Web services.
- ✓ We have composed a process.

- Question?
  - Does the process meet operational requirements?

  - Maybe or maybe not !!!

- Solution
  - End-to-End Process Analysis

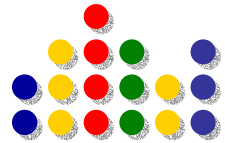
Performance Analysis



- Performance evaluation of Web services can help implementers understand the behavior of the activities in a composed process

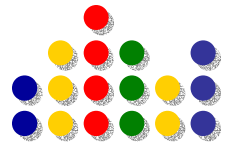
- Web services performance evaluation techniques
  - Time Analysis
  - Load Analysis
  - Process Execution Monitoring

Performance Analysis (contd.)



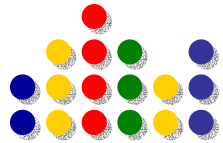
Difficulties in Conducting Performance Analysis Tests

- For conducting performance analysis tests, we require the Web services to be managed by the composer
- If the services involved are real world services (*e.g.*, Flight Booking Service), then performance analysis by conducting real tests is not feasible
- To overcome these problems, Simulation could be used as an alternative technique to do performance estimation



Questions?

References



Berners-Lee, T. (2001). Keynote presentation on web services and the future of the web. Software Development Expo 2001 Visionary Keynote, http://www.technetcast.com/tnc_play_stream.html?stream_id=616.

Bussler, C. (1998). Workflow Instance Scheduling with Project Management Tools. 9th Workshop on Database and Expert Systems Applications DEXA'98, Vienna, Austria, IEEE Computer Society Press. pp. 753-758.

Fabio Casati, Ming-Chien Shan and D. Georgakopoulos (2001). "E-Services - Guest editorial." The VLDB Journal 10(1): 1.

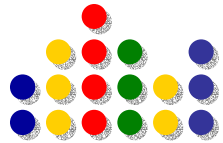
Fensel, D. and C. Bussler (2002). The Web Service Modeling Framework. Vrije Universiteit Amsterdam (VU) and Oracle Corporation, http://www.cs.vu.nl/~dieter/ftp/paper/wsmf.pdf.

Kashyap, V. and A. Sheth (1996). "Schematic and Semantic Similarities between Database Objects: A Context-based Approach." Very Large Data Bases (VLDB) Journal 5(4): 276-304. http://lsdis.cs.uga.edu/lib/download/KS95b.pdf

Kochut, K. J., A. P. Sheth and J. A. Miller (1999). "ORBWork: A CORBA-Based Fully Distributed, Scalable and Dynamic Workflow Enactment Service for METEOR," Large Scale Distributed Information Systems Lab, Department of Computer Science, University of Georgia, Athens, GA.

Miller, J. A., J. S. Cardoso and G. Silver (2002). Using Simulation to Facilitate Effective Workflow Adaptation. Proceedings of the 35th Annual Simulation Symposium (ANSS'02), San Diego, California. pp. 177-181.

References



Miller, J. A., R. Nair, Z. Zhang and H. Zhao (1997). JSIM: A Java-Based Simulation and Animation Environment. Proceedings of the 30th Annual Simulation Symposium, Atlanta, GA. pp. 786-793.

Miller, J. A., D. Palaniswami, A. P. Sheth, K. J. Kochut and H. Singh (1998). "WebWork: METEOR2's Web-based Workflow Management System." Journal of Intelligence Information Management Systems: Integrating Artificial Intelligence and Database Technologies (JIIS) 10(2): 185-215.

Miller, J. A., A. F. Seila and X. Xiang (2000). "The JSIM Web-Based Simulation Environment." Future Generation Computer Systems: Special Issue on Web-Based Modeling and Simulation 17(2): 119-133.

Paolucci, M., T. Kawamura, T. R. Payne and K. Sycara (2002). Semantic Matching of Web Services Capabilities. Proceedings of the 1st International Semantic Web Conference (ISWC2002), Sardinia, Italia.

Rodríguez, A. and M. Egenhofer (2002). "Determining Semantic Similarity Among Entity Classes from Different Ontologies." IEEE Transactions on Knowledge and Data Engineering (in press).

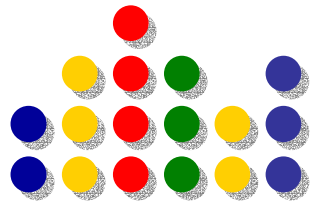
Shegalov, G., M. Gillmann and G. Weikum (2001). "XML-enabled workflow management for e-services across heterogeneous platforms." The VLDB Journal 10(1): 91-103.

Sycara, K., J. Lu, M. Klusch and S. Widoff (1999). Matchmaking Among Heterogeneous Agents on the Internet. Proceedings AAAI Spring Symposium on Intelligent Agents in Cyberspace, Stanford, USA.

Uschold, M. and M. Gruninger (1996). "Ontologies: Principles, methods and applications." Knowledge Engineering Review 11(2): 93-155.

W3C RDF Home Page. http://www.w3.org/RDF/

Process and Quality of Service



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4], Dieter Fensel[3]

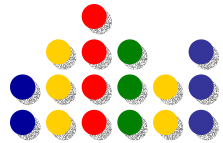
[1]LSDIS Lab, Computer Science, University of Georgia

[2]Oracle Corporation

[3] Universität Innsbruck

[4] Semagix, Inc

QoS
Introduction



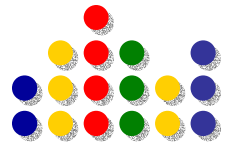
- Organizations operating in modern markets, such as e-commerce activities, require QoS management.

> QoS management is indispensable for organizations striving to achieve a higher degree of competitiveness.

- Products and services with well-defined specifications must be available to customers.

> The appropriate control of quality leads to the creation of quality products and services.

QoS
Introduction

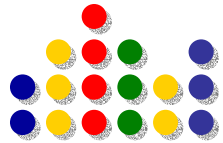


- The computation of QoS metrics allow organizations to better align workflow processes with their vision.
- These, in turn, fulfill customer expectations and achieve customer satisfaction.

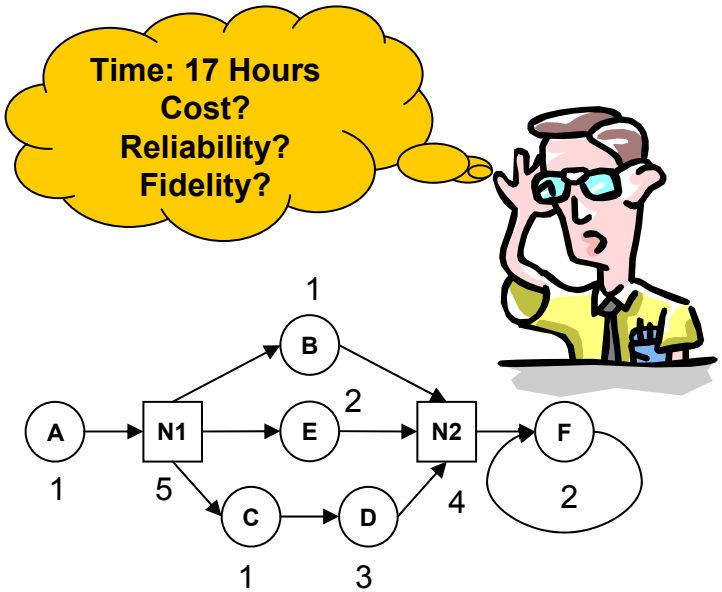
Web processes and workflow QoS can be calculated through
End-to-End Process Analysis

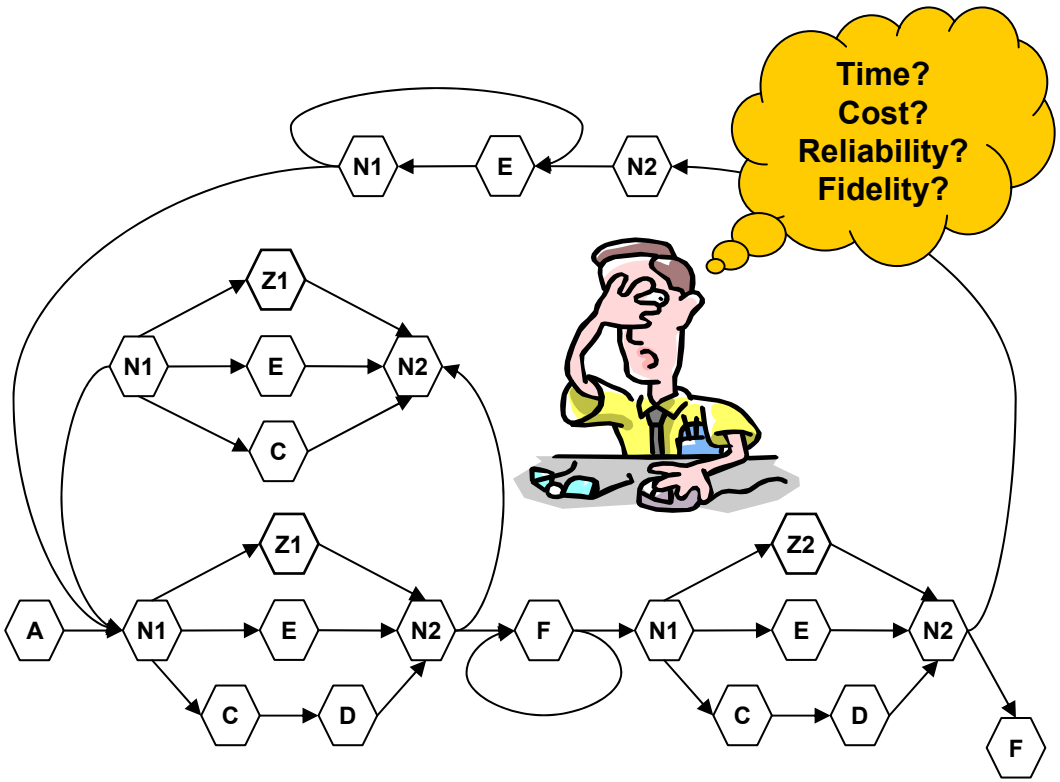
QoS

New Requirements

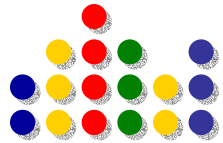


Before	Now





QoS
Benefits



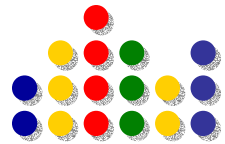
- Composition of processes according to QoS objective and requirements.

- Selection and execution of processes based on QoS metrics.

- Monitoring of processes to assure compliance with initial QoS requirements.

- Evaluation of alternative strategies when QoS requirements are violated.

QoS
Related Work

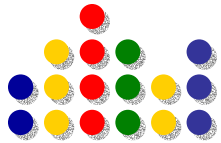


- QoS has been a major concern in the following areas:
  - Networking[1],
  - Real-time applications[2], and
  - Middleware[3].

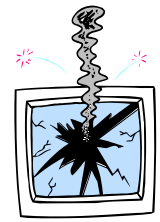
- In the area of Web services, DAML-S allows for the specification of QoS metrics of Web services.
  - It provides a basic QoS model.
  - But the model does not allow for the automatic computation of processes QoS.

[1] Cruz 1995; Georgiadis, Guerin *et al.* 1996,
[2] Clark, Shenker *et al.* 1992
[3] Zinky, Bakken *et al.* 1997; Frlund and Koistinen 1998; Hiltunen, Schlichting *et al.* 2000.

QoS
Related Work



- For workflow systems, QoS studies have mainly been done for the time dimension[1].
- Additional research on workflow reliability has also been conducted.

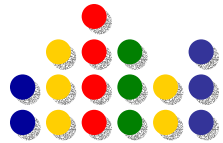


- But the work was mostly on system implementation[2].

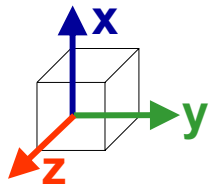
[1]Kao and GarciaMolina 1993; Bussler 1998; Eder, Panagos et al. 1999; Marjanovic and Orlowska 1999; Dadam, Reichert et al. 2000; Sadiq, Marjanovic et al. 2000; Son, Kim et al. 2001.

[2]Kamath, Alonso et al. 1996; Tang and Veijalainen 1999; Wheater and Shrivastava 2000.

QoS
Research Issues



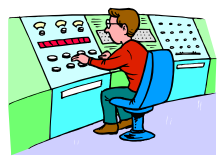
- ✓ Specification. What dimensions need to be part of the QoS model for processes?



- ✓ Computation. What methods and algorithms can be used to compute, analyze, and predict QoS?

- ✓ Monitoring. What king of QoS monitoring tools need to be developed?

- Control. What mechanisms need to be developed to control processes, in response to unsatisfactory QoS metrics?



End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

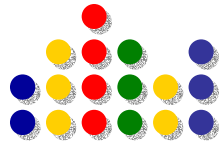
QoS Computation

Enact

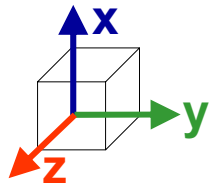
Simulation

Log

QoS



Specification



End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

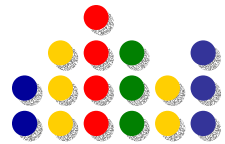
QoS Computation

Enact

Simulation

Log

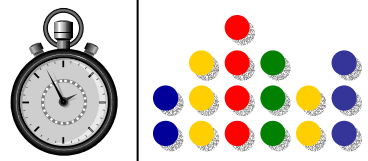
QoS Model



- QoS describes non-functional properties of a process.
- Based on previous studies* and our experience with business processes, we have constructed a QoS model composed of the following dimensions:
  - Time
  - Cost
  - Reliability
  - Fidelity

*Stalk and Hout,1990;Rommel et al.,1995;Garvin, 1988

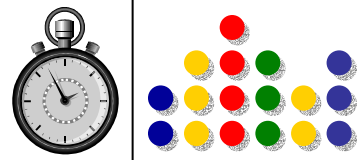
QoS Model
Web Service/Task Time



- Time is a common and universal measure of performance.
- The first measure of time is task cycle time (CT)
- For workflow systems, it can be defined as the total time needed by an task to transform a set of inputs into outputs.
- The task cycle time can be breakdown in two major components: delay time and process time.

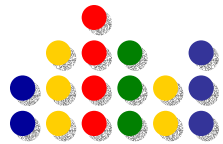
$$CT(t) = DT(t) + PT(t)$$

QoS Model
Web Service/Task Time



- The delay time can be further broken down into
  - Queuing delay
  - Setup delay
- Another time metric that may be considered is the
  - Synchronization delay

QoS Model
Web Service/Task Cost

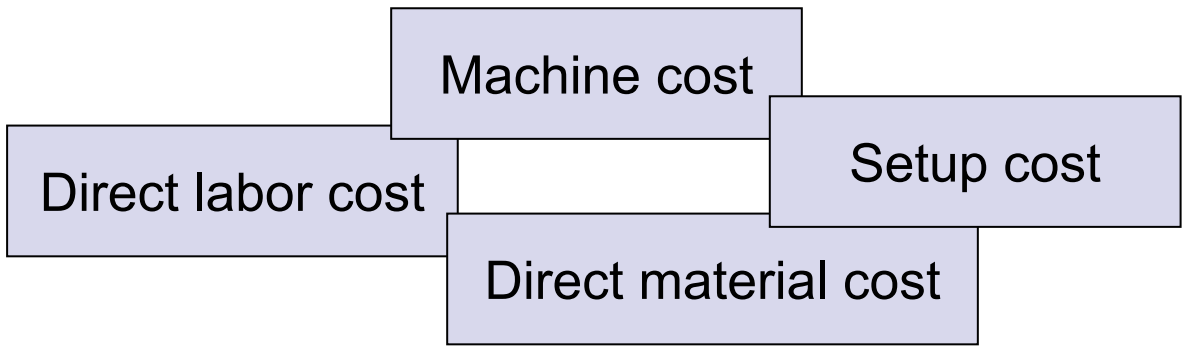


- The cost dimension represents the cost associated with the execution of Web Services or workflow tasks.
- Cost is an important factor, since organizations need to operate according to their financial plan.
- Task cost (C) is the cost incurred when a task *t* is executed; it can be broken down into two major components: enactment cost and realization cost.

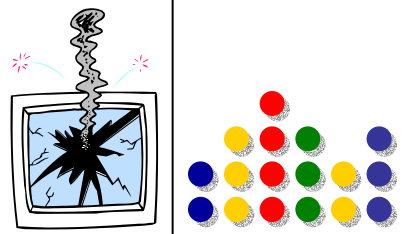
$$C(t) = EC(t) + RC(t)$$

QoS Model
Web Service/Task Cost

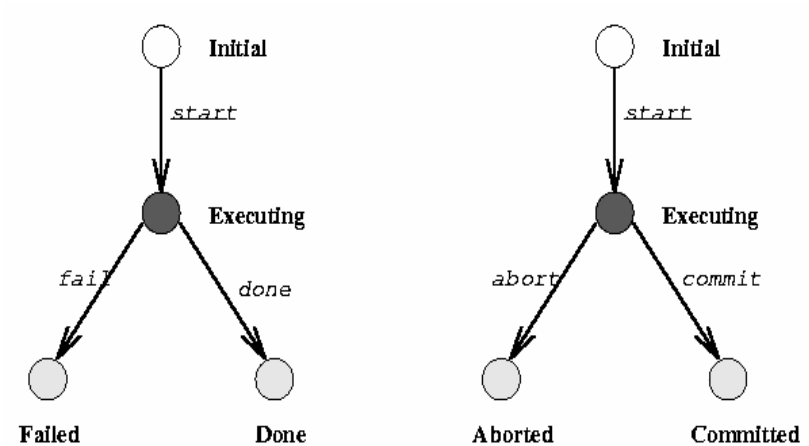
- The enactment cost (EC) is the cost associated with the management of the workflow system and with workflow instances monitoring.
- The realization cost (RC) is the cost associated with the runtime execution of the task. It can be broken down into:



QoS Model
Web Service/Task Reliability

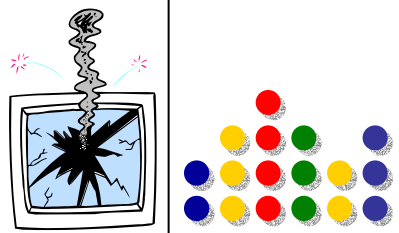


- Reliability (R) corresponds to the likelihood that a task will perform for its users when the user demands it.
- Workflow task execution can be represented using the following task structures



(Krishnakumar and Sheth, 1995)

QoS Model
Web Service/Task Reliability

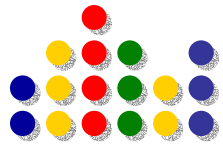


- This QoS dimension provides information concerning a relationship between the number of times the state done/committed is reached and the number of times the failed/aborted state is reached after the execution of a task.
- This dimension follows from the discrete-time stable reliability model proposed in Nelson (1973).

R(t) = 1 - failure rate

Note: Other reliability models can also be used (Goel ,1985; Ireson, Jr *et al.,* 1996).

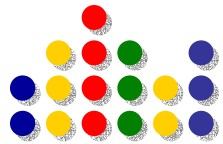
QoS Model
Web Service/Task Fidelity



- Fidelity is a function of effective design and refer to an intrinsic property or characteristic of a good produced or service rendered.
- Tasks have a fidelity (F) vector dimension composed by a set of fidelity attributes ($F(t)_{.attribute}$).

For more information on this dimension the reader is referred to Cardoso, J., J. Miller, A. Sheth and J. Arnold (2002). "Modeling Quality of Service for Workflows and Web Service Processes." LSDIS Lab Technical Report, May 2002.

QoS Model
Discussion



- Workflows can be classified in one of the following categories*:
  - *ad hoc* workflows
  - administrative workflows, and
  - production workflows.
- The QoS model presented here is better suited for production workflows since they are more structured, predictable, and repetitive.

*McCready 1992

End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

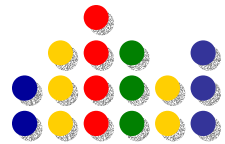
QoS Computation

Enact

Simulation

Log

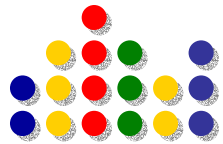
QoS
Creation of Estimates



- To analyze a process QoS, it is necessary to:
  - Create estimated for task QoS metrics and
  - Create estimated for transition probabilities

Once tasks and transitions have their estimates set, algorithms and mechanisms, such as simulation, can be applied to compute the overall QoS of a process.

QoS
Estimates for Tasks



The task runtime behavior specification is composed of two classes of information: basic and distributional.

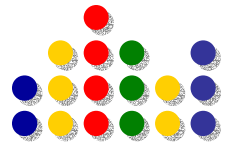
The basic class associates with each task's QoS dimension the minimum value, average value, and maximum value the dimension can take.

	Basic class			Distributional class
	Min value	Avg value	Max value	Dist. Function
Time	0.291	0.674	0.895	Normal(0.674, 0.143)
Cost	0	0	0	0.0
Reliability	-	100%	-	1.0
Fidelity.a_i	0.63	0.81	0.92	Trapezoidal(0.7,1,1,4)

Task QoS for an automatic task (SP FASTA task)

The second class, corresponds to the specification of a constant or of a distribution function (such as Normal, Weibull, or Uniform) which statistically describes task behavior at runtime.

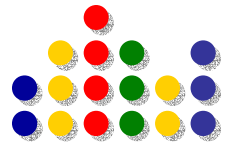
QoS
Estimates for Tasks



- The values specified in the basic class are typically employed by mathematical methods in order to compute workflow QoS metrics

- The distributional class information is used by simulation systems.

QoS
Re-Computing Estimates for Tasks

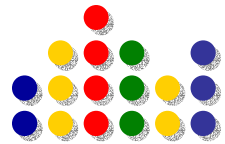


- The re-computation of QoS task metrics is based on data coming from designer specifications and from the workflow system log.

Designer Average$_{Dim}(t)$	Average specified by the designer in the basic class for dimension Dim
Multi-Workflow Average$_{Dim}$ (t)	Average of the dimension Dim for task t executed in the context of any workflow
Workflow Average$_{Dim}(t, w)$	Average of the dimension Dim for task t executed in the context of any instance of workflow w
Instance Average$_{Dim}(t, w,$ i)	Average of the dimension Dim for task t executed in the context of instance i of workflow w

Designer, multi-workflow, workflow and instance average

QoS
Re-Computing Estimates for Tasks

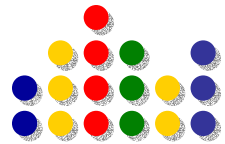


- The task QoS for a particular dimension can be determined at different levels:

a)	$QoS_{Dim}(t)$	Designer $Average_{Dim}(t)$
b)	$QoS_{Dim}(t)$	wi_1* Designer $Average_{Dim}(t)$ + wi_2* Multi-Workflow $Average_{Dim}(t)$
c)	$QoS_{Dim}(t, w)$	wi_1* Designer $Average_{Dim}(t)$ + wi_2* Multi-Workflow $Average_{Dim}(t)$ + wi_3*Workflow $Average_{Dim}(t, w)$
d)	$QoS_{Dim}(t, w, i)$	wi_1* Designer $Average_{Dim}(t)$ + wi_2* Multi-Workflow $Average_{Dim}(t)$ + wi_3* Workflow $Average_{Dim}(t, w)$ + wi_4* Instance Workflow $Average_{Dim}(t,w, i)$

QoS dimensions computed at runtime

QoS
Estimates for Transitions



- In the same way we seed tasks' QoS, we also need to seed workflow transitions.
- Initially, the designer sets the transition probabilities at design time.
- At runtime, the transitions' probabilities are re-computed.
- The method used to re-compute the transitions' probabilities follows the same lines of the method used to re-compute tasks' QoS.

End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

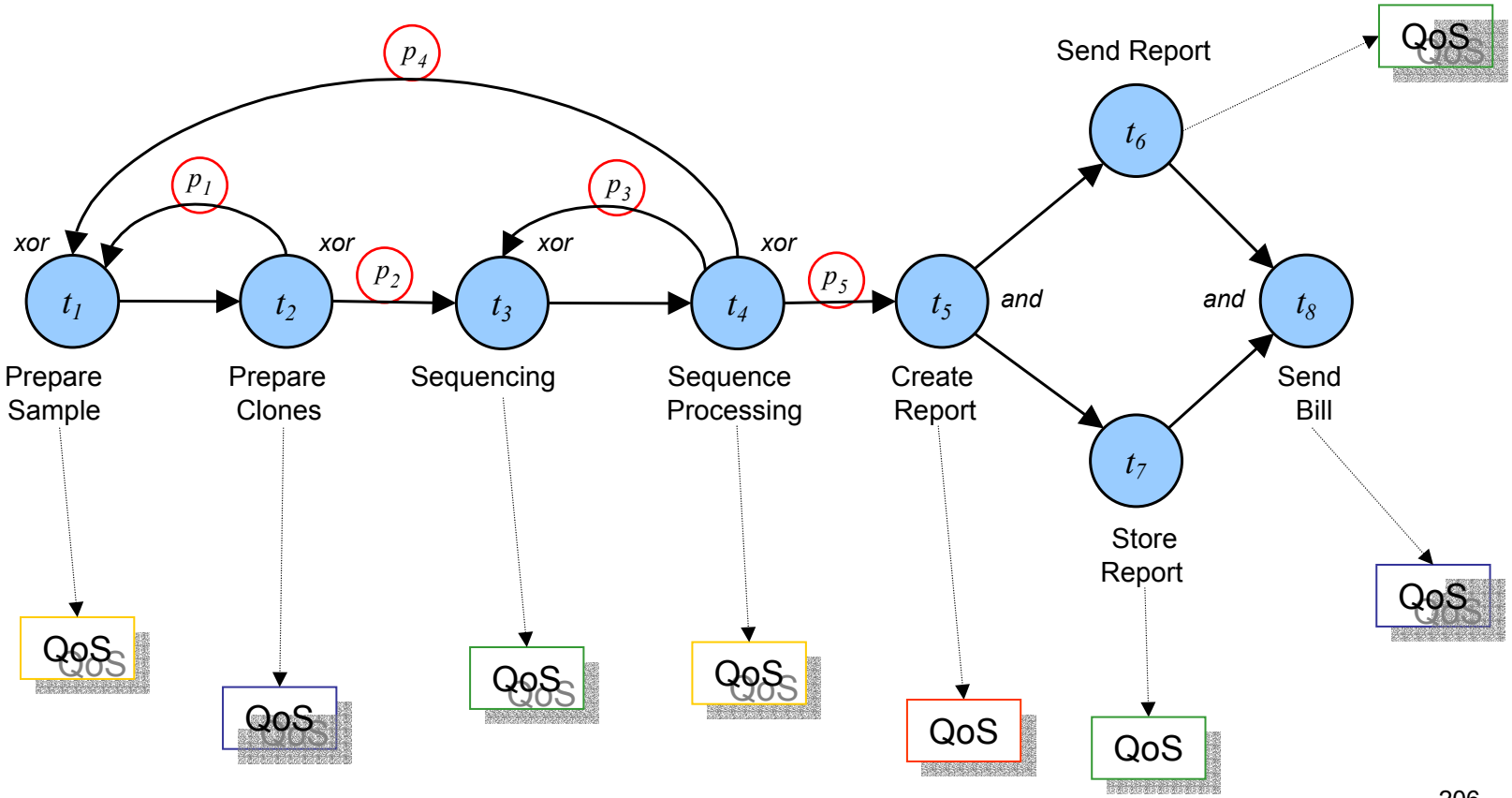
QoS Computation

Enact

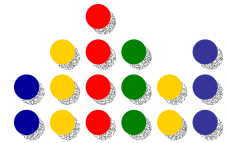
Simulation

Log

Stochastic QoS-based Process



QoS



Computation

End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

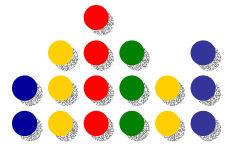
QoS Computation

Enact

Simulation

Log

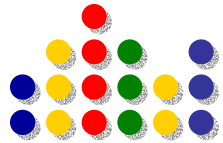
QoS
Computation



- Once QoS estimates for tasks and for transitions are determined, we can compute the overall QoS of a workflow.

- Two modeling techniques can be used to compute QoS metrics for a given workflow process: mathematical modeling and simulation modeling.

QoS Computation
Mathematical Modeling



- To compute process QoS metrics, we have developed a set of six distinct reduction systems:
  - (1) sequential,
  - (2) parallel,
  - (3) conditional,
  - (4) loop,
  - (5) fault-tolerant, and
  - (6) network.

Mathematical Modeling
Reduction of a Sequential System

(a) (b)

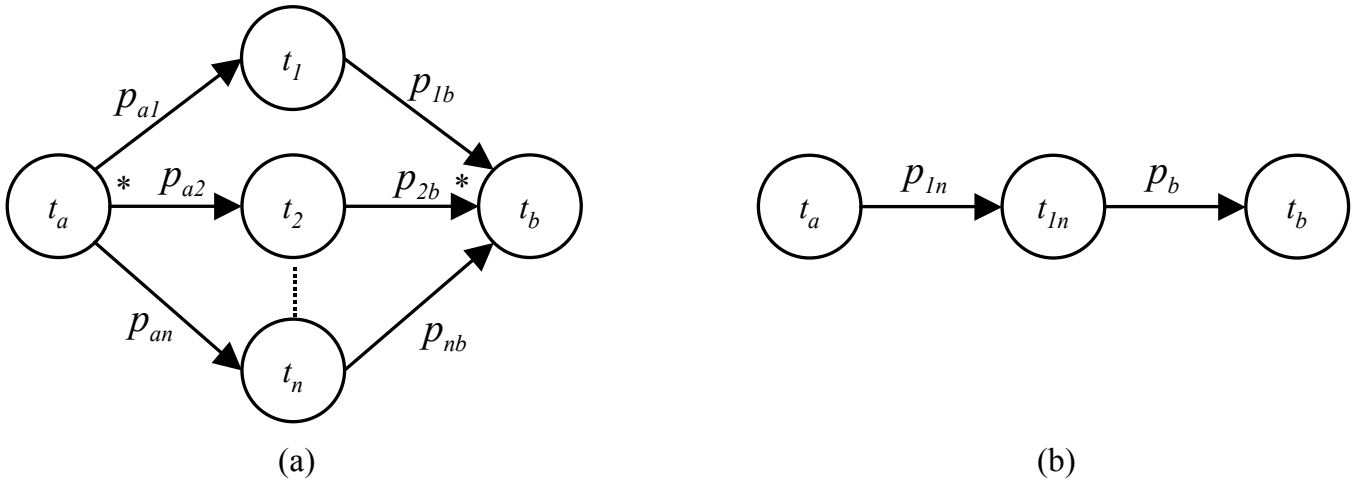
$$T(t_{ij}) = T(t_i) + T(t_j)$$

$$C(t_{ij}) = C(t_i) + C(t_j)$$

$$R(t_{ij}) = R(t_i) * R(t_j)$$

$$F(t_{ij}).a_r = f(F(t_i), F(t_j))$$

Mathematical Modeling
Reduction of a Parallel System



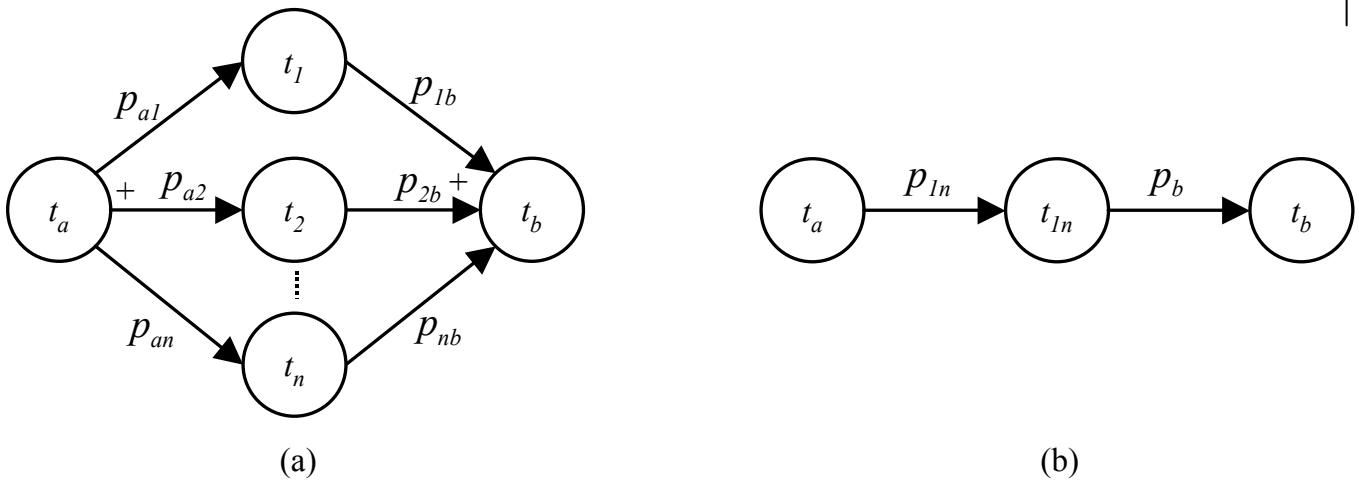
$$T(t_{1n}) = \text{Max}_{I \in \{1..n\}} \{T(t_i)\}$$

$$C(t_{1n}) = \sum_{1 \le i \le .n} C(t_i)$$

$$R(t_{1n}) = \prod_{1 \le i \le .n} R(t_i)$$

$$F(t_{1n}).a_r = f(F(t_1), F(t_2), \ldots, F(t_n))$$

Mathematical Modeling
Reduction of a Conditional System



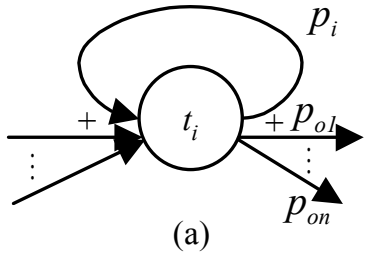
$$T(t_{1n}) = \sum_{1 \le i \le .n} p_{ai} * T(t_i)$$

$$C(t_{1n}) = \sum_{1 \le i \le .n} p_{ai} * C(t_i)$$

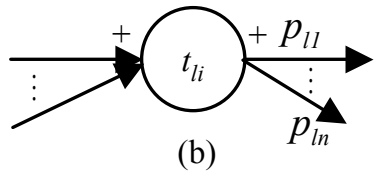
$$R(t_{1n}) = \sum_{1 \le i \le .n} p_{ai} * R(t_i)$$

$$F(t_{1n}).a_r = f(p_{a1}, F(t_1), p_{a2}, F(t_2), \ldots, p_{an}, F(t_n))$$

Mathematical Modeling
Reduction of a Loop System



(a)



(b)

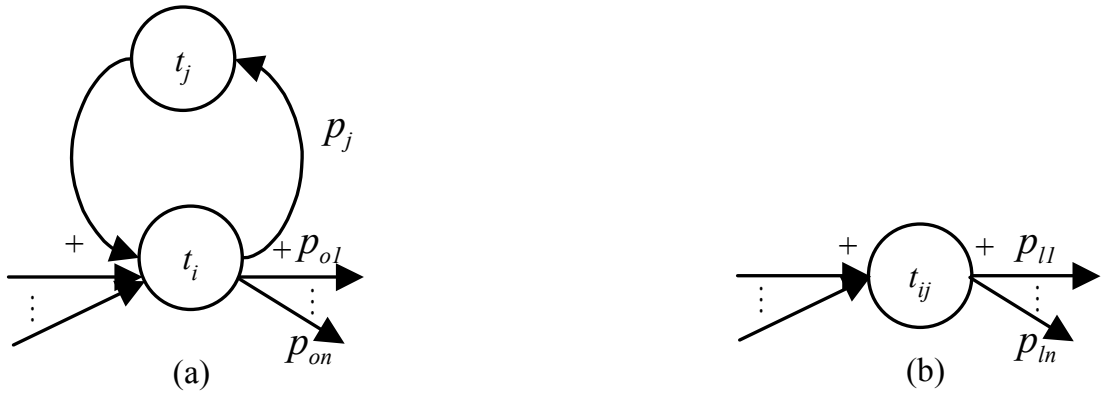
$$T(t_{li}) = \frac{T(t_i)}{1 - p_i}$$

$$C(t_{li}) = \frac{C(t_i)}{1 - p_i}$$

$$R(t_{li}) = \frac{(1 - p_i) * R(t_i)}{1 - p_i R(t_i)}$$

$$F(t_{li}).a_r = f(p_i, F(t_i))$$

Mathematical Modeling
Reduction of a Loop System



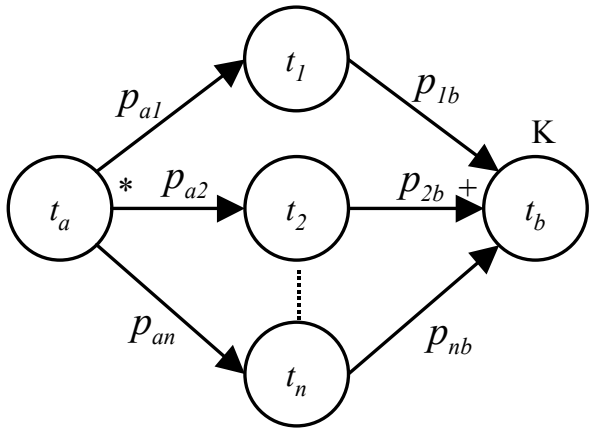
$$T(t_{ij}) = \frac{T(t_i) + T(t_j) - (1 - p_j)T(t_j)}{(1 - p_j)}$$

$$C(t_{ij}) = \frac{C(t_i) + C(t_j) - (1 - p_j)C(t_j)}{(1 - p_j)}$$

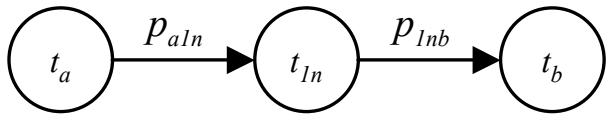
$$R(t_{ij}) = \frac{(1 - p_j) * R(t_i)}{1 - p_j R(t_i) R(t_j)}$$

$$F(t_{ij}).a_r = f(F(t_i), p_j, F(t_j))$$

Mathematical Modeling
Reduction of a Fault-Tolerant System



(a)



(b)

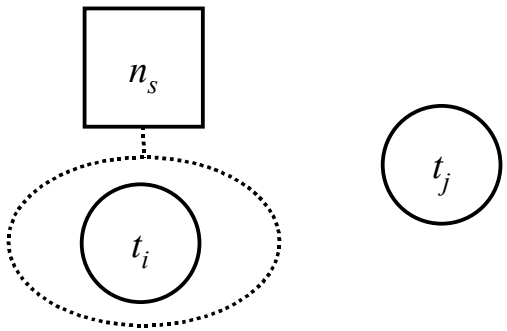
$$T(t_{1n}) = \underset{k}{Min}(\{T(t_1),...,T(t_n)\})$$

$$C(t_{1n}) = \sum_{1 \le i \le n} C(t_I)$$

$$R(t_{1n}) = \sum_{i_1=0}^{1} \dots \sum_{i_n=0}^{1} f\left(\sum_{j=1}^{n} i_j - k\right) * ((1-i_1) + (2i_1 - 1)R(t_1)) * \dots * ((1-i_n) + (2i_n - 1)R(t_n))$$

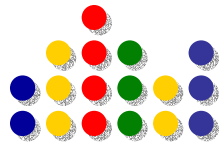
$$F(t_{1n}).a_r = f(p_{a1}, F(t_1), p_{a2}, F(t_2), \dots, p_{an}, F(t_n), k)$$

Mathematical Modeling
Reduction of a Network System



$$X(t_j) = X(t_i),\ X \in \{T, C, R, F\}$$

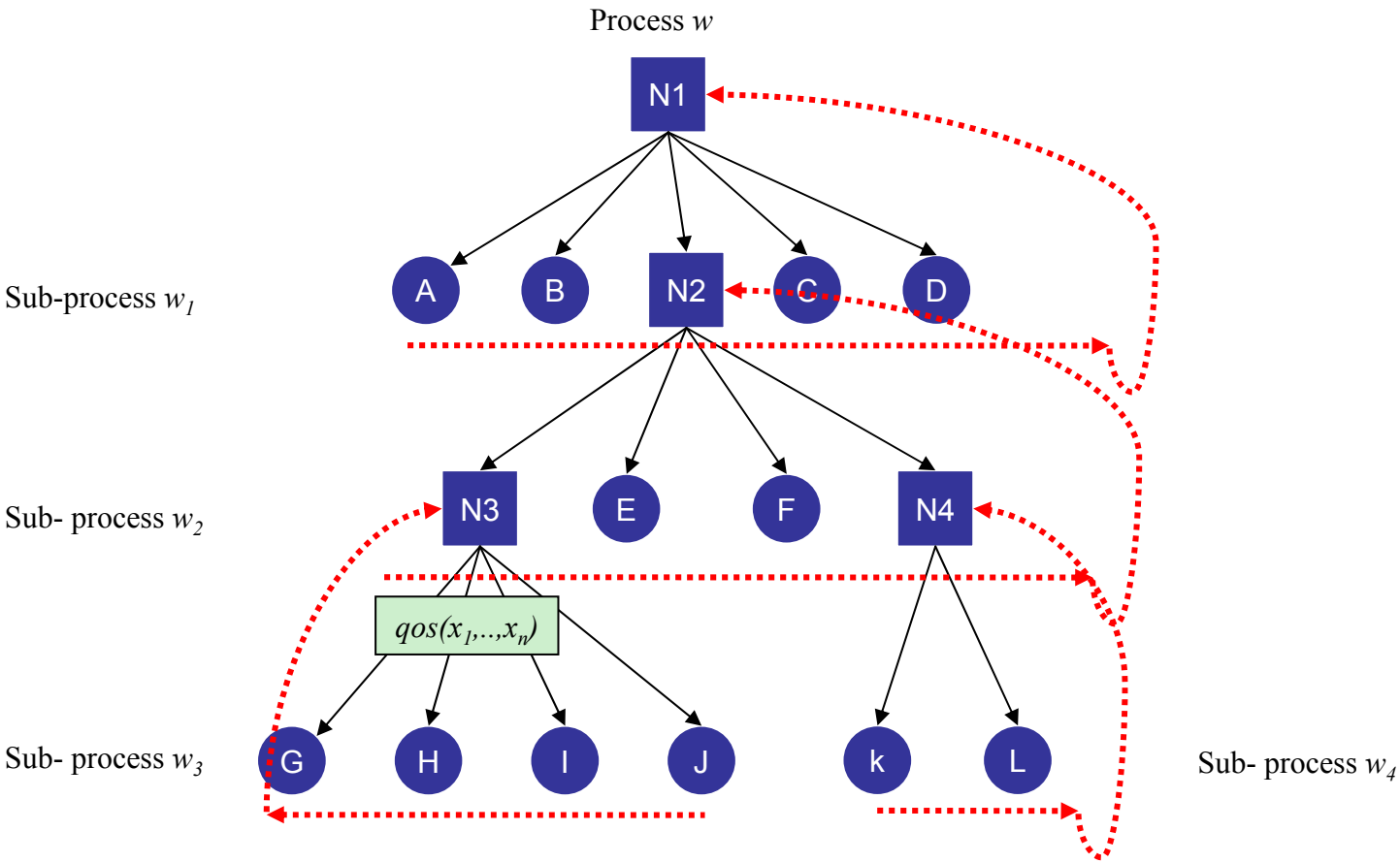
Mathematical Modeling
The SWR Algorithm



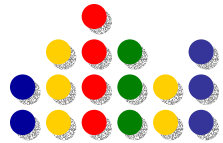
- The stochastic workflow reduction (SWR) method consists of applying the previous set of reduction rules to a process until only one atomic* task exists.
- Each time a reduction rule is applied, the process structure changes.
- After several iterations only one task will remain.
- When this state is reached, the remaining task contains the QoS metrics corresponding to the process under analysis.

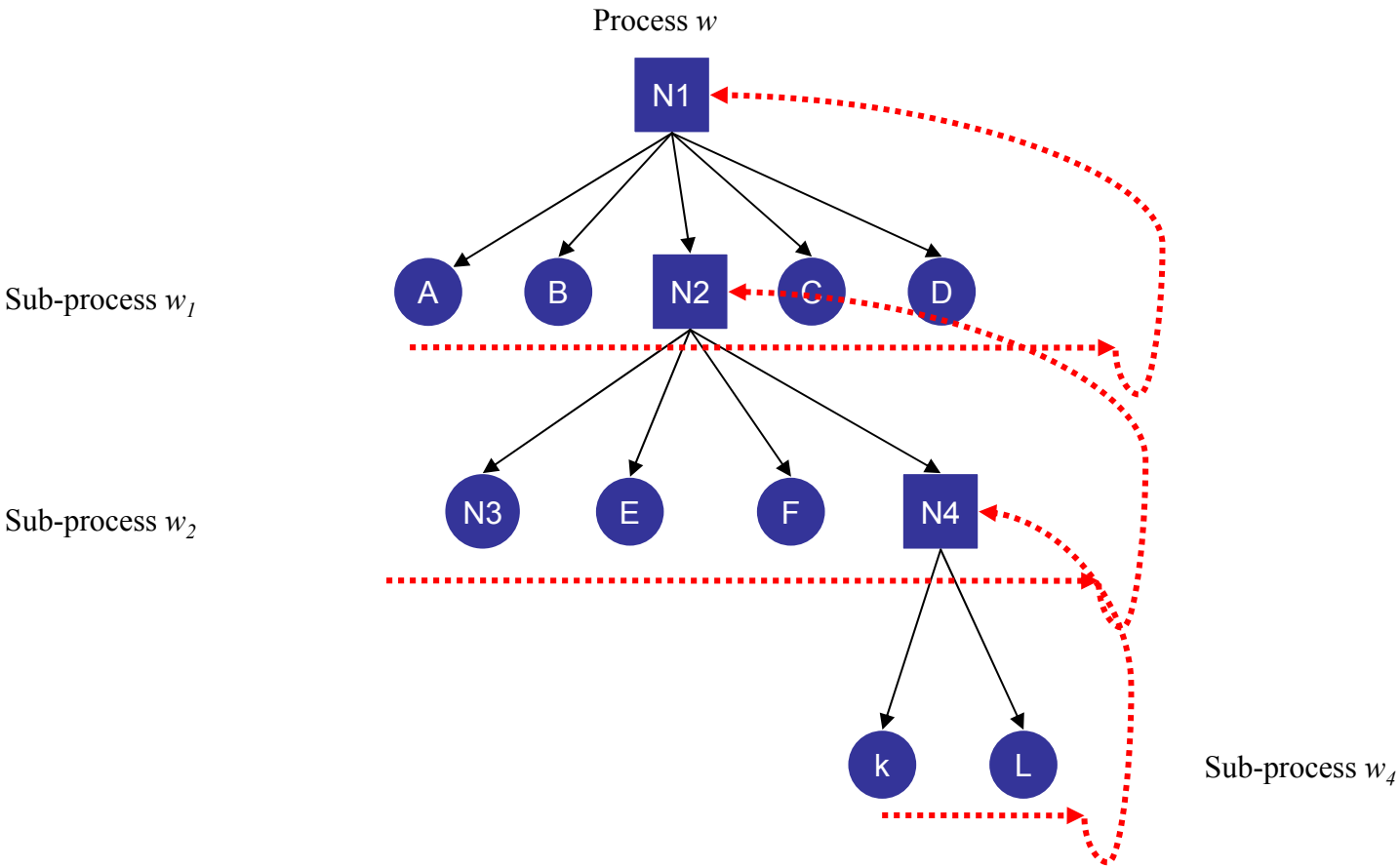
*Kochut, Sheth *et al.* 1999

SWR Algorithm

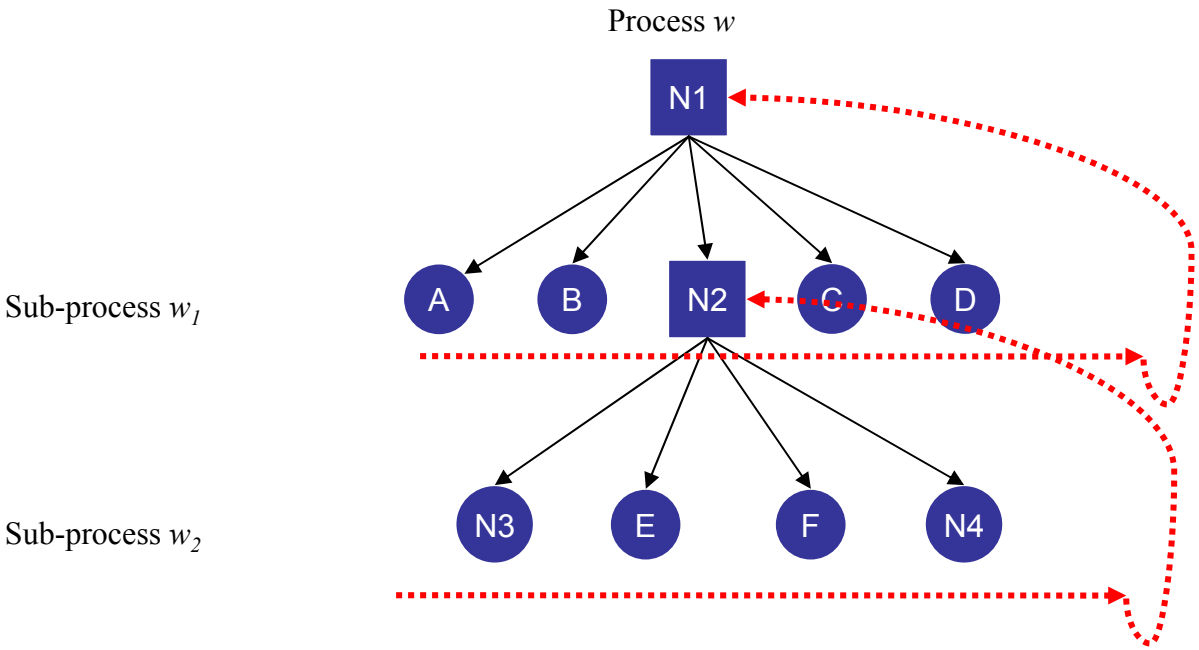


SWR Algorithm

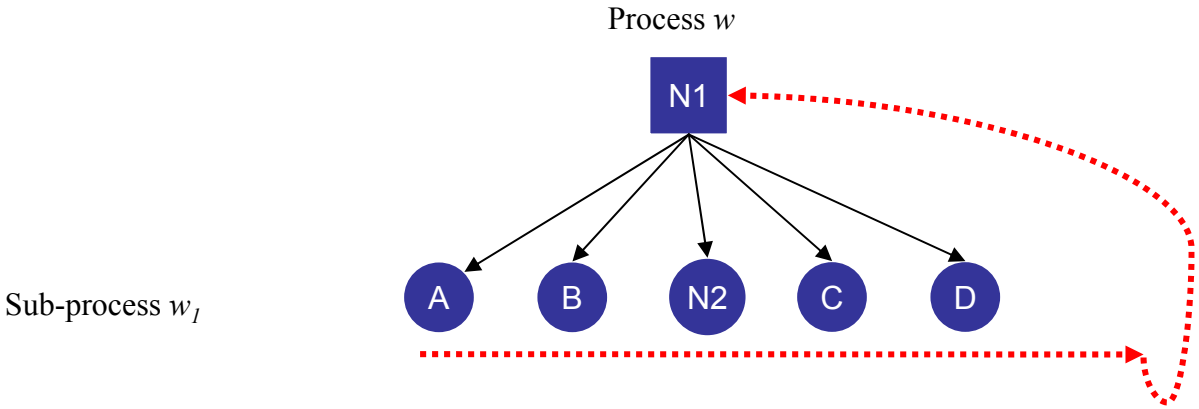




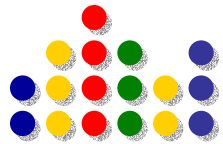
SWR Algorithm



SWR Algorithm



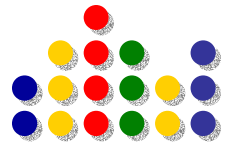
SWR Algorithm



Process *w*

N1

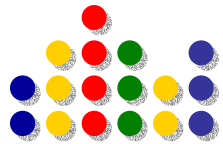
Simulation Modeling
Introduction



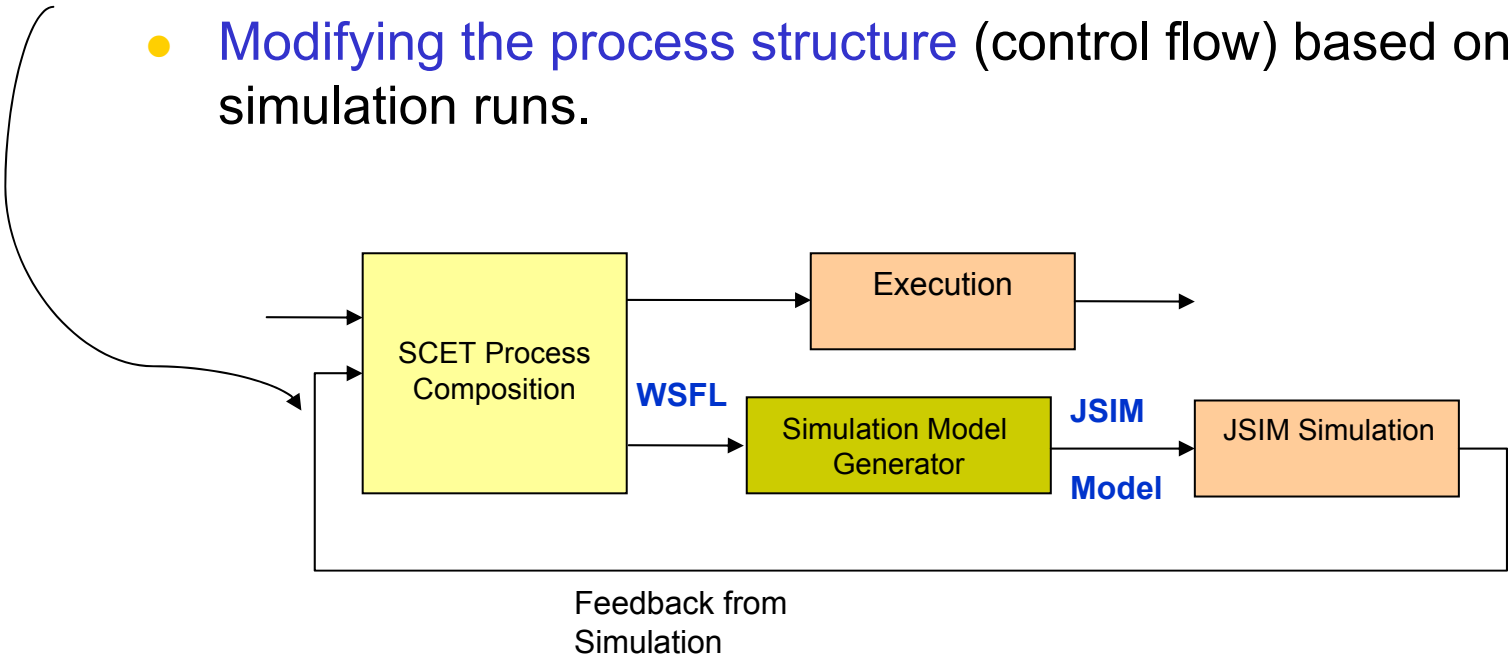
- While mathematical methods can be effectively used, another alternative is to utilize simulation analysis[1].
- Simulation can play an important role in tuning the QoS metrics of processes by exploring "what-if" questions.
- In our project, these capabilities involve a loosely-coupled integration between the METEOR WfMS and the JSIM simulation system[2].

[1]Miller, Cardoso *et al.* 2002, [2]Nair, Miller *et al.* 1996; Miller, Nair *et al.* 1997; Miller, Seila *et al.* 2000.

Web Process Simulation
Tools

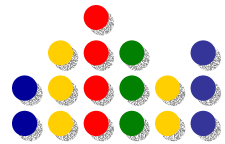


- Simulation provides feedback on processes, allowing the composer to modify his process design by
  - Replacing services which do not satisfy the expected runtime behavior with more suitable Web services.
  - Modifying the process structure (control flow) based on the simulation runs.



Senthilanand Chandrasekaran, M.Sc. Thesis presented at the Department of Computer Science of the University of Georgia.

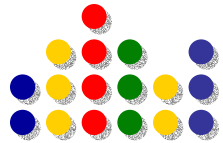
Web Process Simulation
SCET Tool



- SCET (Service Composition and Execution Tool) allows
  - to compose services statically by modeling the process as a digraph in a graphical designer
  - stores the process description as WSFL based specification
  - allows execution of the composed process using Perl
  - supports a simple execution monitoring feature
  - supports performance estimation using JSIM simulation

Senthilanand Chandrasekaran, M.Sc. Thesis presented at the Department of Computer Science of the University of Georgia.

Web Process Simulation
SCET Tool



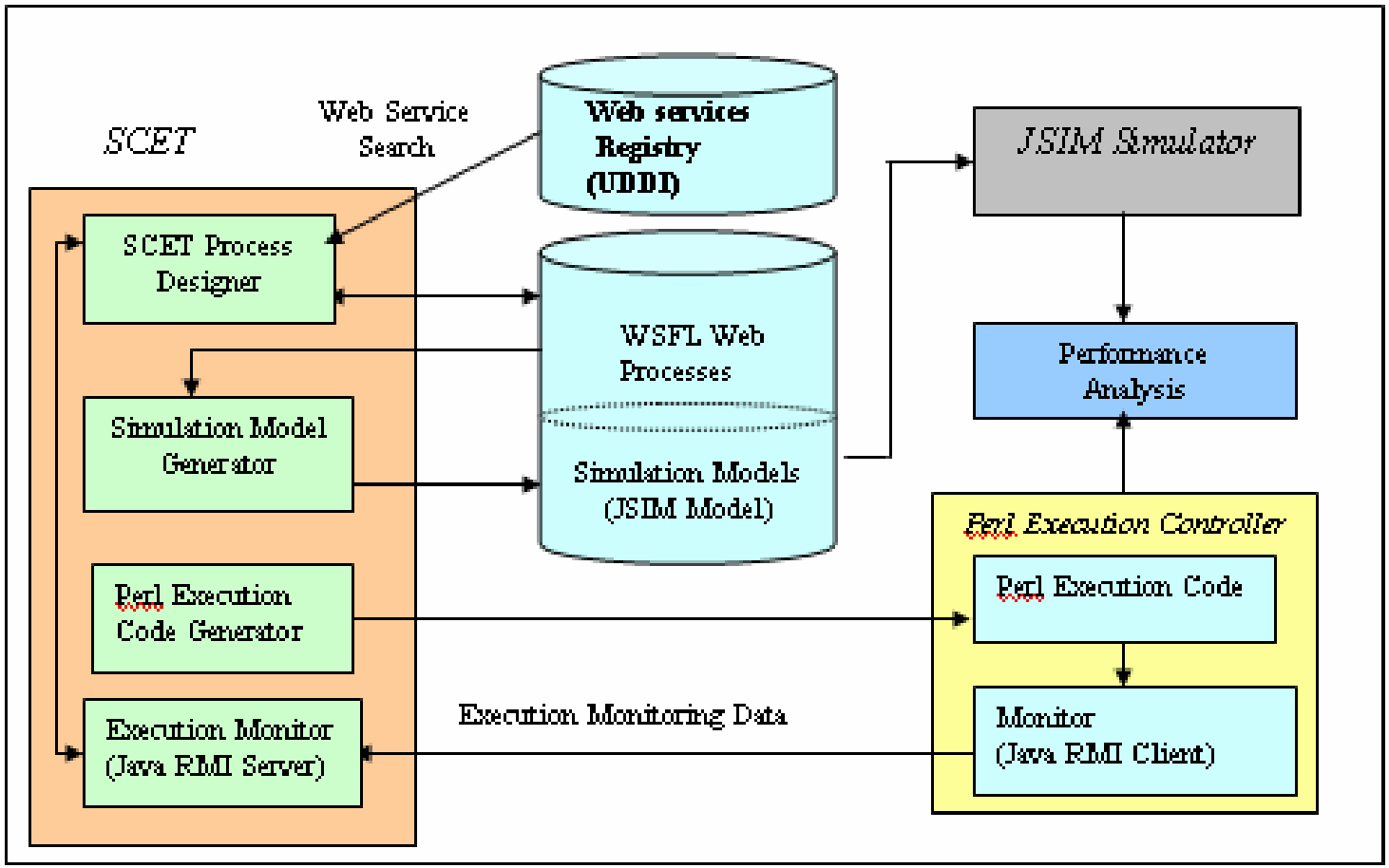
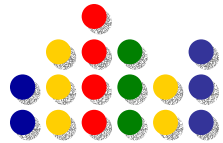


Figure 1. System Architecture for SCET

Senthilanand Chandrasekaran, M.Sc. Thesis presented at the Department of Computer Science of the University of Georgia.

QoS
Metrics of Interest



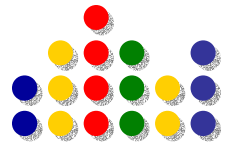
Workflow Response Time (T(*w*))

The workflow response time is the total amount of time that a workflow instance spends within a workflow process before it finishes.

Workflow Delay Time (DT(*w*))

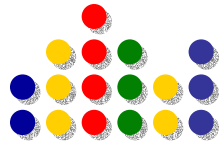
The workflow delay time, sometimes called “waiting time,” is the total amount of time that a workflow instance spends in a workflow, while not being processed by a task.

QoS
Metrics of Interest



- Minimum Workflow Response Time (min T(*w*))
  - The minimum workflow response time, sometimes called the "service time" of a workflow, is the time required for a workflow instance to be processed, not accounting for any task delay time.
- Workflow Response Time Efficiency (E(*w*))
  - is the ratio of the minimum workflow response time and the workflow response time.
  - It is instructive to compare these two measures, since instance efficiency measurement provides an indication of the time an instance is delayed during its execution and also indicates the degree a workflow process can be improved by reducing its response time.

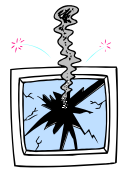
QoS
Metrics of Interest



Workflow Cost (C(*w*))

Workflow reliability corresponds to the likelihood that a workflow will perform for its users on demand.

Workflow Reliability (R(*w*))

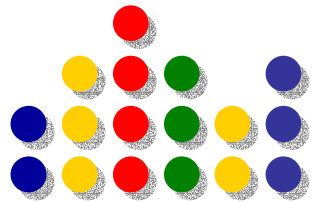


Workflow cost analysis measures the cost incurred during the execution of a workflow.

Workflow Fidelity ($F._{attribute}(w)$)

Workflow fidelity is a function of effective design; it refers to the intrinsic properties or characteristics of a good produced or a service rendered.

QoS Implementation



End-to-End Process Analysis

The Overall Idea

Design

QoS Model

QoS Estimates for Tasks/Web services

QoS Estimates for Transitions

SWR algorithm

Stochastic Process

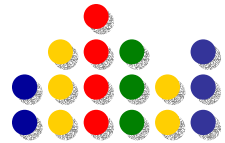
QoS Computation

Enact

Simulation

Log

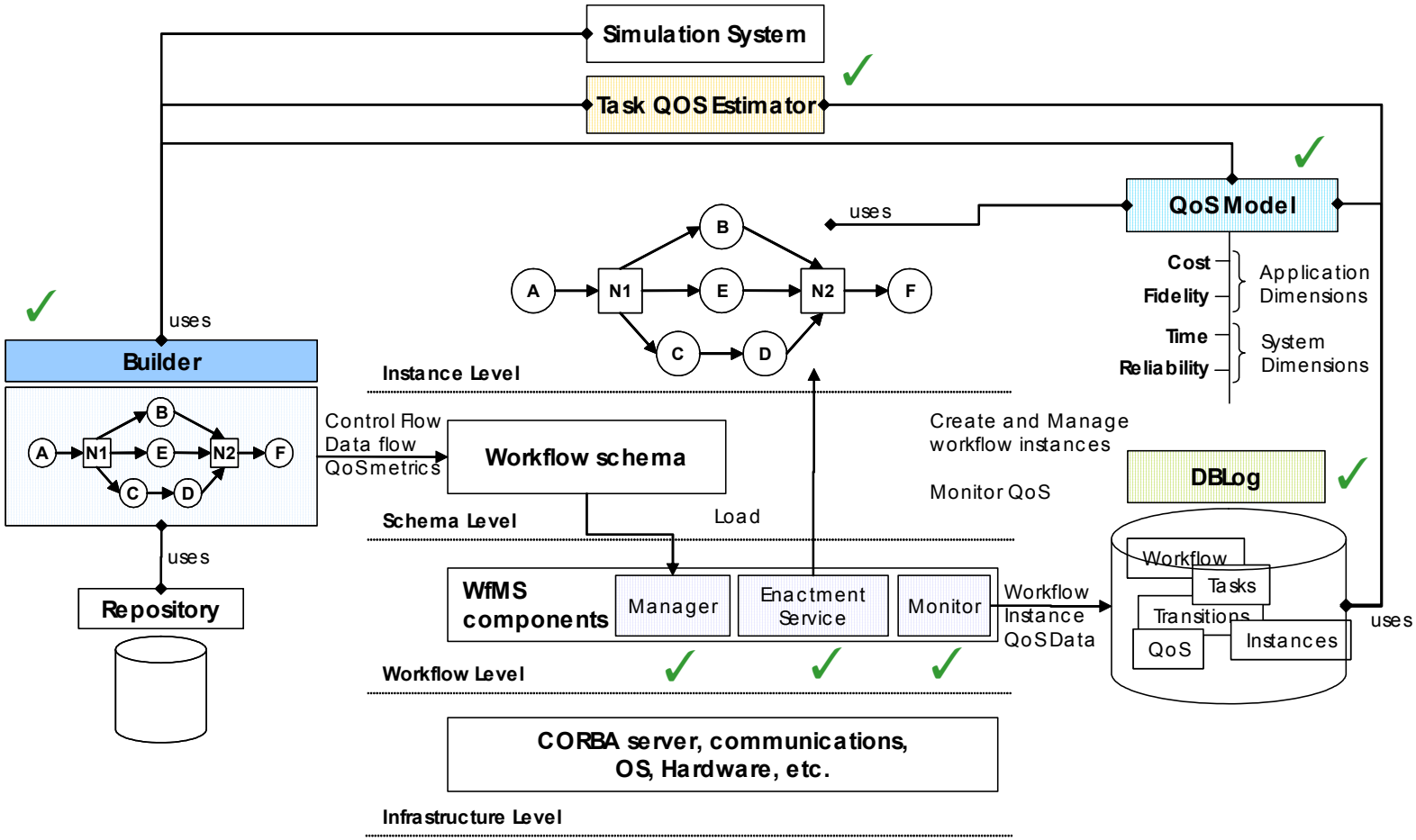
QoS Implementation



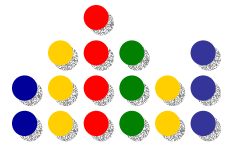
- The QoS model developed was implemented for the METEOR workflow management system.

- It was necessary to make changes to four services:
  - ✓ Enactment,
  - ✓ Manager,
  - ✓ Builder, and
  - Repository.

QoS Implementation
Architecture



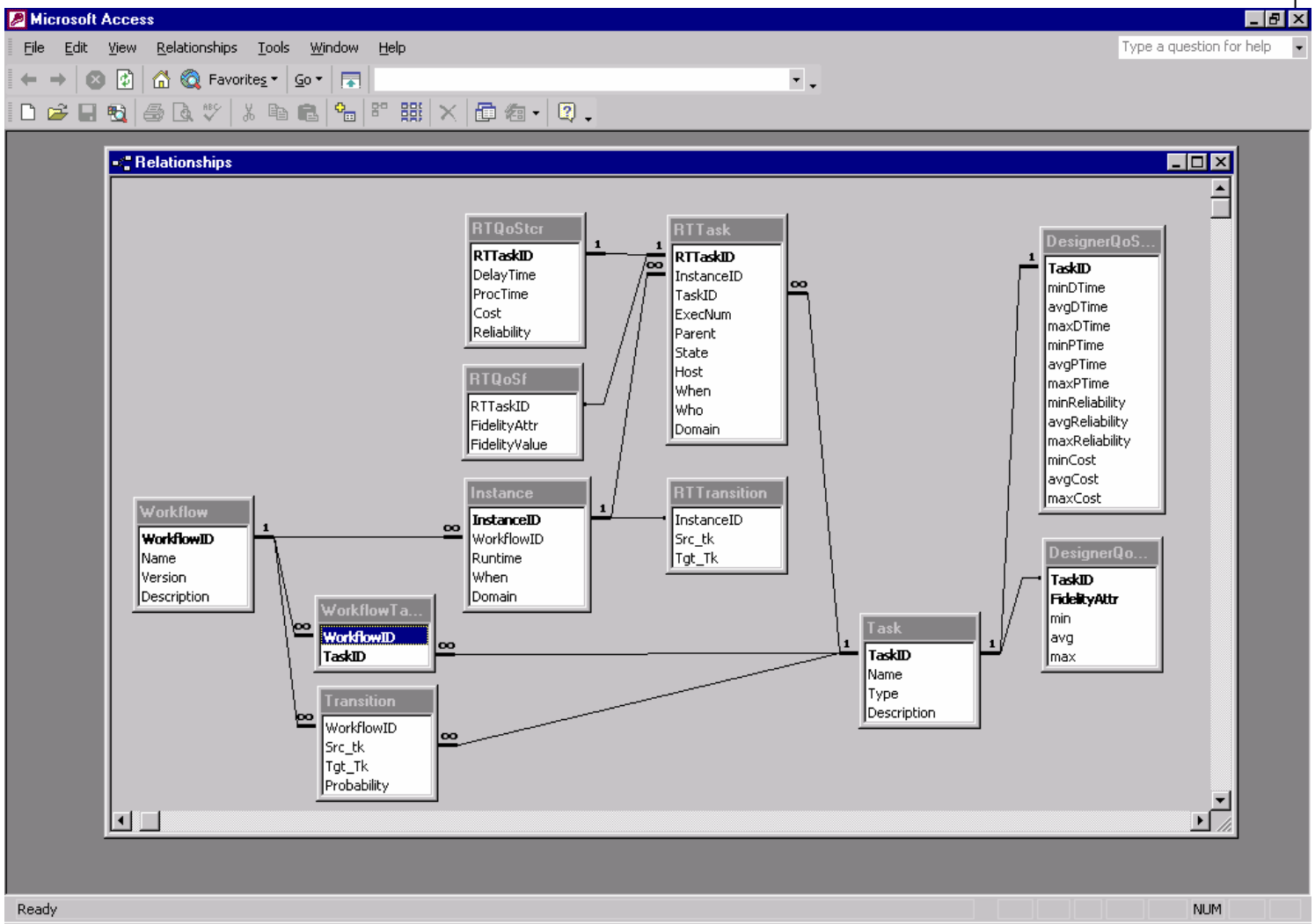
QoS Implementation
Monitor - DBLog



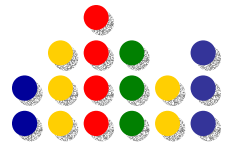
- A DBlog has been developed to store the status and QoS events generated in a relational database.
- When a process is installed and executed, task QoS estimates, runtime QoS metrics, and transition frequencies are stored in the database.
- The stored information will be later utilized to create a QoS profile for the tasks and to enable the computation of the workflow QoS.

QoS Implementation

Monitor - DBLog

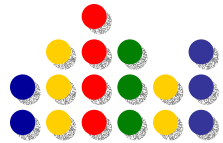


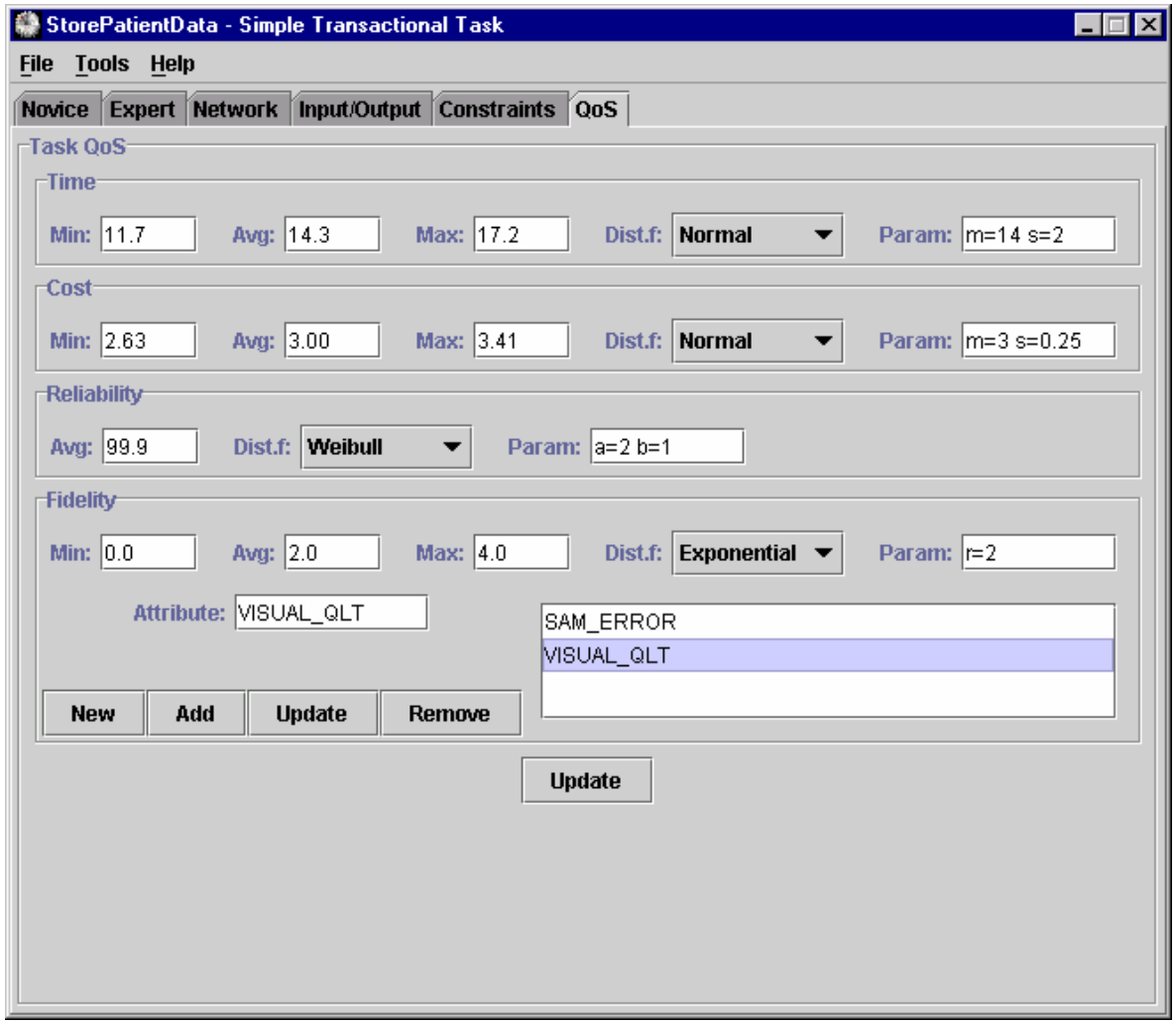
QoS Implementation
Builder



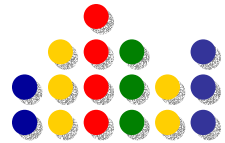
- The workflow builder tool is used to graphically design and specify a workflow.
- To support workflow QoS management the designer must be able to set estimates for transition probabilities and QoS estimates for tasks.
- The workflow model and the task model have been extended to support the specification of QoS metrics.

QoS Implementation
Setting Task QoS



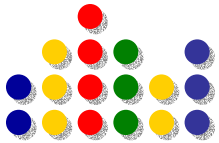


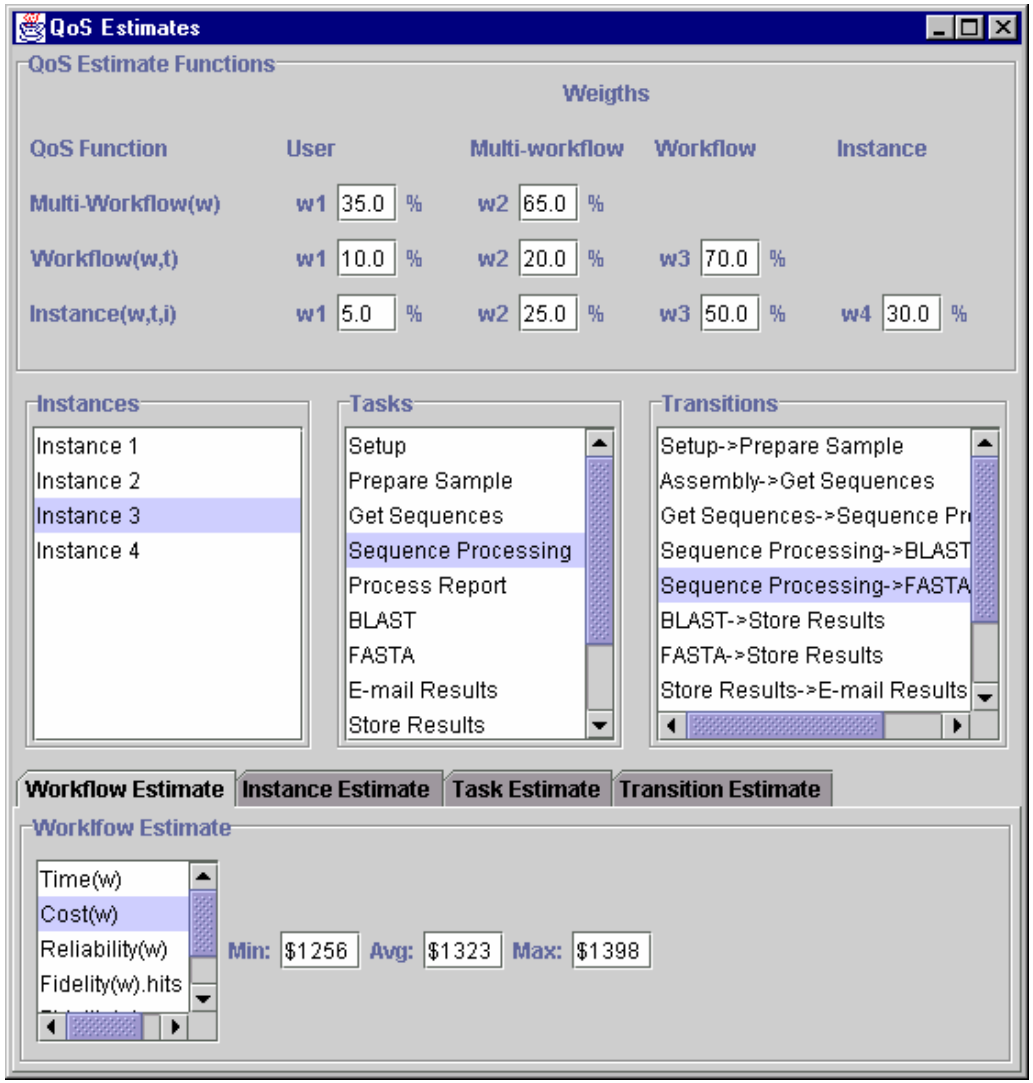
Builder



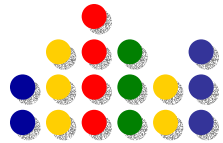
- The initial QoS specifications may not be valid over time. To overcome this difficulty we re-compute task QoS values for the basic class, based on previous executions.
- The user sets the QoS functions used to automatically re-compute QoS metrics for workflows, instances, tasks, and transitions.
- At any time, including design time and runtime, it is possible to calculate QoS estimate.
- Workflow QoS estimates are calculated using the SWR algorithm.

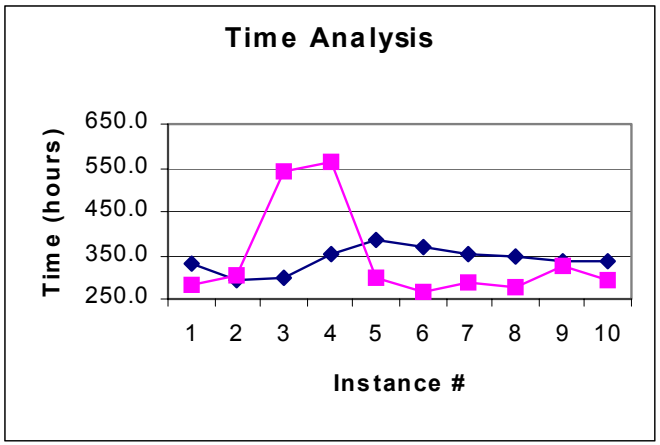
QoS Implementation
QoS Analysis

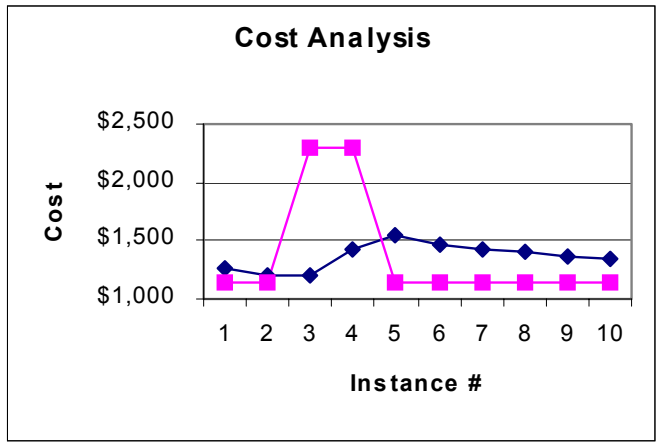


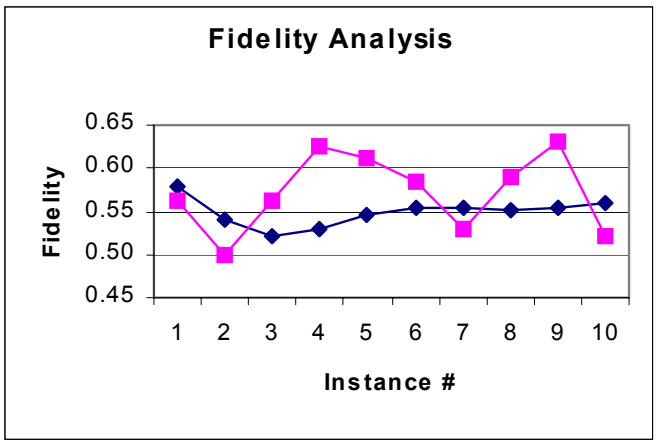


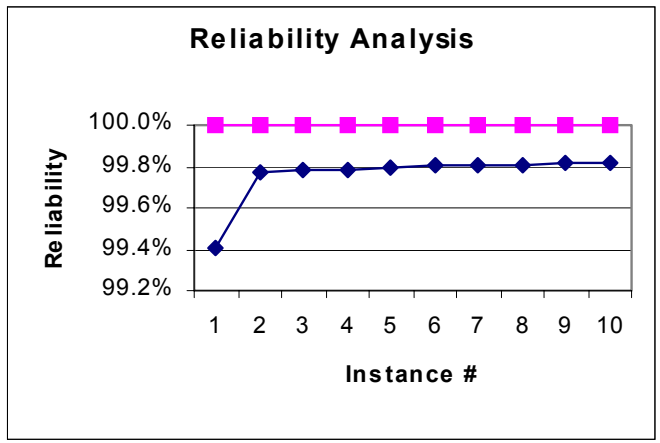
Experiments
Results

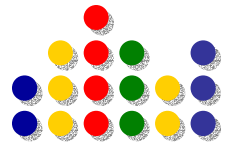






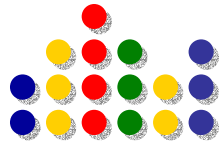






Questions?

References



Berners-Lee, T. (2001). Keynote presentation on web services and the future of the web. Software Development Expo 2001 Visionary Keynote, http://www.technetcast.com/tnc_play_stream.html?stream_id=616.

Bussler, C. (1998). Workflow Instance Scheduling with Project Management Tools. 9th Workshop on Database and Expert Systems Applications DEXA'98, Vienna, Austria, IEEE Computer Society Press. pp. 753-758.

Cardoso, J., J. Miller, A. Sheth and J. Arnold (2002). "Modeling Quality of Service for Workflows and Web Service Processes." the Very Large Data Bases Journal submitted in May 2002.

Eder, J., E. Panagos, H. Pozewaunig and M. Rabinovich (1999). Time Management in Workflow Systems. BIS'99 3rd International Conference on Business Information Systems, Poznan, Poland, Springer Verlag. pp. 265-280.

Fabio Casati, Ming-Chien Shan and D. Georgakopoulos (2001). "E-Services - Guest editorial." The VLDB Journal 10(1): 1.

Frlund, S. and J. Koistinen (1998). "Quality-of-Service Specification in Distributed Object Systems." Distributed Systems Engineering Journal 5(4).

Goel, A. L. (1985). "Software reliability models: assumptions, limitations, and applicability." IEEE Transactions on Software Engineering 11(12): 1411-1423.

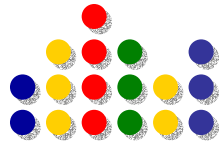
Ireson, W. G., C. F. C. Jr. and R. Y. Moss (1996). Handbook of reliability engineering and management. New York, McGraw Hill.

Kamath, M., G. Alonso, R. Guenthor and C. Mohan (1996). Providing High Availability in Very Large Workflow Management Systems. Proceedings of the 5th International Conference on Extending Database Technology, Avignon. pp. 427-442.

Kochut, K. J., A. P. Sheth and J. A. Miller (1999). "ORBWork: A CORBA-Based Fully Distributed, Scalable and Dynamic Workflow Enactment Service for METEOR," Large Scale Distributed Information Systems Lab, Department of Computer Science, University of Georgia, Athens, GA.

Marjanovic, O. and M. Orlowska (1999). "On modeling and verification of temporal constraints in production workflows." Knowledge and Information Systems 1(2): 157-192.

References



McCready, S. (1992). There is more than one kind of workflow software. Computerworld. November 2: 86-90.

Miller, J. A., J. S. Cardoso and G. Silver (2002). Using Simulation to Facilitate Effective Workflow Adaptation. Proceedings of the 35th Annual Simulation Symposium (ANSS'02), San Diego, California. pp. 177-181.

Miller, J. A., R. Nair, Z. Zhang and H. Zhao (1997). JSIM: A Java-Based Simulation and Animation Environment. Proceedings of the 30th Annual Simulation Symposium, Atlanta, GA. pp. 786-793.

Miller, J. A., D. Palaniswami, A. P. Sheth, K. J. Kochut and H. Singh (1998). "WebWork: METEOR2's Web-based Workflow Management System." Journal of Intelligence Information Management Systems: Integrating Artificial Intelligence and Database Technologies (JIIS) 10(2): 185-215.

Miller, J. A., A. F. Seila and X. Xiang (2000). "The JSIM Web-Based Simulation Environment." Future Generation Computer Systems: Special Issue on Web-Based Modeling and Simulation 17(2): 119-133.

Nelson, E. C. (1973). "A Statistical Basis for Software Reliability," TRW Software Series March.

Saavendra, R. H. and A. J. Smith (1996). "Analysis of Benchmark Characteristics and Benchmark Performance Prediction." ACM Transactions on Computer Systems 14(4): 344-384.

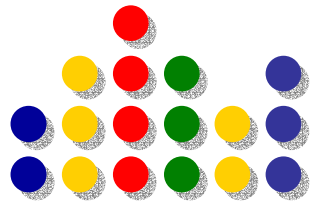
Sadiq, S., O. Marjanovic and M. E. Orlowska (2000). "Managing Change and Time in Dynamic Workflow Processes." The International Journal of Cooperative Information Systems 9(1, 2): 93-116.

Wheater, S. M. and S. K. Shrivastava (2000). "Reliability Mechanisms in the OPENflow Distributed Workflow System," Department of Computing Science, University of Newcastle upon Tyne Technical Report 31, Esprit LTR Project No. 24962 - C3DS First year Report, pp. 269-288.

Zinky, J., D. Bakken and R. Schantz (1997). "Architectural Support for Quality of Service for CORBA Objects." Theory and Practice of Object Systems 3(1): 1-20.

More at: Workflow and Semantic Web Processes: http://lsdis.cs.uga.edu/proj/meteor/SWP.htm

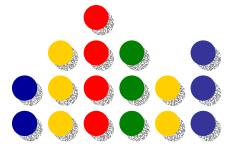
Conclusions



Jorge Cardoso[1], Christoph Bussler[2], Amit Sheth[1, 4] , Dieter Fensel[3],
[1]LSDIS Lab, Computer Science, University of Georgia
[2]Oracle Corporation
[3] Universität Innsbruck
[4] Semagix, Inc

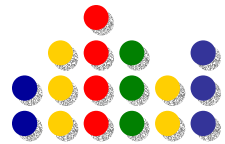
Conclusions
Putting Everything Together

Web Service
Discovery and Integration



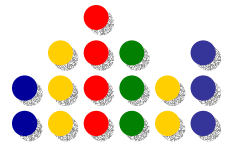
- A multidimensional approach to Web service discovery is more suitable for current requirements.
- Syntactic, operational, and semantic dimensions needs to be considered.
- The discovery has also to account for the posteriori integration of the services found.
- An Ontology-based approaches have proved to be an important solution to the discovery and integration of Web services.

Web Service
Discovery and Integration



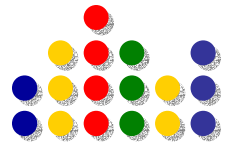
- In the area of process composition, our research has resulted in the following advances:
  - Development of a methodology for semantic process composition.
  - Development of an algorithm to compute the syntactic, operational, and semantic similarity of Web services and to assist designers in resolving interoperability issues among Web services.
  - Development of a prototype incorporating the above concepts.

Web Service
QoS



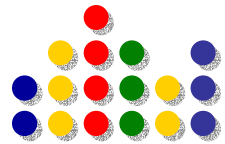
- Our efforts on workflow QoS management have resulted in the following advances:
  - Development of a comprehensive and predictive QoS model for Web processes and workflows.
  - Development of a QoS mathematical model.
  - Development of an algorithm (the SWR algorithm) to automatically compute and estimate Web processes and workflow QoS.
  - Implementation of the above elements in the METEOR workflow system.

Web Service
QoS



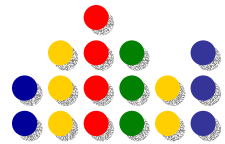
- The composition of Web-services cannot be undertaken while ignoring the importance of QoS measurements.
- The use of a QoS model allows for the description of process components from a QoS perspective.
- Based on the QoS of tasks the QoS of processes can be automatically computed.
- Mathematical models and simulation models are suitable to compute QoS metrics.

Semantic Web Service
Research Topics



Environment	Scalable, openness, autonomy, heterogeneity, evolving
Representation	Self-description, conversation, contracts, commitments, QoS
Programming	Compose & customize, workflow, negotiation
Interaction (system)	Trust, security, compliance
Architecture	P2P, privacy,
Utilities	Discovery, binding, trust-service

Semantic Web Service
Research Topics

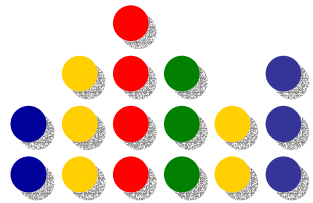


Data => services, similar yet more challenging:

- Modeling <functional and operational>
- Organizing collections
- Discovery and comparison (reputation)
- Distribution and replication
- Access and fuse (composition)
- Fulfillment
  - Contracts, coordination/negotiation versus transactions
    - Roll back, Roll forward, Exception handling, recovery
  - Quality: more general than correctness or precision
  - Compliance
- Dynamic, flexible security and trust; privacy

Web Resource for this tutorial

(incl. latest version)



http://lsdis.cs.uga.edu/lib/presentations/SWSP-tutorial-resource.htm

Contact: Amit Sheth