CPET 581 Cloud Computing: Technologies and Enterprise IT Strategies

Lecture 7
SOA and Distributed Computing

Spring 2015
A Specialty Course for Purdue University’s M.S. in Technology Graduate Program: IT/Advanced Computer App Track
Paul I-Hai Lin, Professor
Dept. of Computer, Electrical and Information Technology
Purdue University Fort Wayne Campus

References

2. CPET 545 Service-Oriented Architecture & Enterprise Applications, Paul I-Hai Lin, Fall 2008, Purdue University, M.S. in Technology IT/Advanced Computer App Track, http://www.etcw.iit.edu/~lin/CPET545_SOA/cpet545-F08/cpet545home_F08.html
CPET 545 Course Description
CPET 545 Service-Oriented Architecture & Enterprise Applications, Class 3, Cr. 3

This course provides an intensive and comprehensive introduction to all essential aspects of the Service-Oriented Architecture (SOA) and enterprise applications including modeling, methodologies, processes, enabling technologies in support of enterprise computing applications. Topics include XML and Web services, enterprise application and data integration issues, service-oriented architecture, SOA analysis and design (SOAD), enterprise SOA realization, SOA-driven project management, and case studies. Student participation in presenting technical papers from the recent literature, class discussion, and team-based mid-term and final projects and presentation are expected.

Evolution of Enterprise Applications & Data Integration

- Enterprise Applications
  - Communications and networking infrastructure
  - 2-tier and N-tier applications
  - Distributing computing
  - E-business and E-commerce
  - Enterprise Resource Planning
  - Supply-Chain Management
  - Customer Relationship Management
Evolution of Enterprise Applications & Data Integration

- Enterprise Applications Integration
  - Web services for applications integration
  - Middleware, Web

- Data integration
  - Linking databases to applications and other databases
  - Database
  - Data warehouse
  - Data center
  - Data mining

- Knowledge Management & Business Intelligence

Services and Service-Oriented Architecture

- Technologies and System Architectures used resources
  - Collecting, Accessing, Processing, Integrating, Exchange, storage, Programming
  - Resources: web pages, business info, messages, pictures, video files, etc

- Distributed, Loosely coupled, and Heterogeneous information and computer systems

- Web services
  - Internet access
  - Web based
  - Multi-tier
  - Service producer, broker, consumer

Courtesy of Geoffrey Fox, 2010
Services and Service-Oriented Architecture (SOA)
The W3C defines SOA as a form of distributed systems architecture characterized by the following properties, http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/:

- **Logic View:**
  - The SOA is an abstracted, logic view of actual programs, databases, business processes and so on, defined in terms of what it does, typically carrying out a business-level operation.
  - The service is formally defined in terms of the messages exchanged between provider agents and request agents.

- **Message Orientation**
- **Description Orientation**
  - **Granularity**
  - **Network orientation**
  - **Platform-neutral**

---


- **Figure 1-1** The General Process of Engaging a Web Service

---
Services and Service-Oriented Architecture

- SOAP-based Web Services
- A definition, [http://www.w3.org/2005/Talks/1115-hh-k-ecows/#(10)y](http://www.w3.org/2005/Talks/1115-hh-k-ecows/#(10)y)

“A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.”

Service-Oriented Architecture: The Evolution

Figure 1.23 The evolution of service-oriented architecture: Grids of Clouds and Grids where SS refers to Sensor Service and Fks to a filter or transforming service.

Courtesy of Geoffrey Fox, 2010
Service-Oriented Computing
(Knowledge & Skills Inventory)

- Web Service Architecture, 2004,
  http://www.w3.org/TR/ws-arch/
- Programming languages, design, & development tools:
  C/C++, Java, C#, UML, NetBeans, Visual Studio, etc
- Computer systems, application servers, Web servers
- Network & communications: TCP/IP protocols, IP,
  network programming
- Web programming tools and protocols: HTTP, HTTPS,
  HTML, XML, Javascript, SAP.NET, Java Applet and
  Servlet, Perl, etc
- Databases: MySQL, IBM DB2, Oracle, MS SQL

Service-Oriented Computing

- Web Service Protocols:
  - XML RPC (Remote Procedure Call)
  - CORBA (Common Object Request Broker Architecture),
  - SOAP 1.2: Simple Object Access Protocol, 2007,
    http://www.w3.org/TR/soap/
  - WSDL 1.1: Web Service Description Language, 2001,
    http://www.w3.org/TR/wsdI
  - UDDI OASIS, Universal Description, Discovery, Integration,
    http://uddi.xml.org/
  - REST
General Layered Architecture for Distributed Entities

Fig. 1.21 General layered architecture for distributed entities

Layered Architecture for Web Services and Grids

Figure 1.22 Layered architecture for web services and grids
REST (Representational State Transfer) and Systems of Systems

REST Web Services

- Web Service Protocols:
  - REST (Representational State Transfer)
  - Originally, a software architectural style for networked hypermedia applications in distributed systems
  - Primarily used to build HTTP-based web services that are lightweight, maintainable, and scalable.
  - A service based on REST is called RESTful services

- References
RESTful Services

- Features of REST (Representational State Transfer)
  - Representations, Messages, URIs, Uniform interface, Stateless, Links between resources, Caching

- Four principles
  - Resource Identification through URIs
  - Uniform, Constrained Interface
  - Self-Descriptive Message
  - Stateless Interactions

Representation of a resource
- JSON (JavaScript Object Notation), [http://json.org/](http://json.org/)
- XML based


```json
{
  "ID": "1",
  "Name": "M Vaqqas",
  "Email": "m.vaqqas@gmail.com",
  "country": "India"
}
```

XML Example
- `<Person>` `<ID>` 1 `</ID>` `<Name>` M Vaqqas `</Name>` `<Email>` m.vaqqas@gmail.com `</Email>` `<Country>` India `</Country>` `</Person>`
### Table 5.1 REST Architectural Elements

<table>
<thead>
<tr>
<th>REST Elements</th>
<th>Elements</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data elements</td>
<td>Resource</td>
<td>The intended conceptual target of a hypertext reference</td>
</tr>
<tr>
<td></td>
<td>Resource identifier</td>
<td>URL</td>
</tr>
<tr>
<td></td>
<td>Representation</td>
<td>HTML document, JPEG image, XML, etc.</td>
</tr>
<tr>
<td></td>
<td>Representation metadata</td>
<td>Media type, last-modified time</td>
</tr>
<tr>
<td></td>
<td>Resource metadata</td>
<td>Source link, alternates, vary</td>
</tr>
<tr>
<td></td>
<td>Control data</td>
<td>If-modified-since, cache-control</td>
</tr>
<tr>
<td>Connectors</td>
<td>Client</td>
<td>libwww, libwww-perl</td>
</tr>
<tr>
<td></td>
<td>Server</td>
<td>libwww, Apache API, NSAPI</td>
</tr>
<tr>
<td></td>
<td>Cache</td>
<td>Browser cache, Akamai cache network</td>
</tr>
<tr>
<td></td>
<td>Resolver</td>
<td>Bind (DNS lookup library)</td>
</tr>
<tr>
<td></td>
<td>Tunnel</td>
<td>SSL after HTTP CONNECT</td>
</tr>
<tr>
<td>Components</td>
<td>Origin server</td>
<td>Apache httpd, Microsoft IIS</td>
</tr>
<tr>
<td></td>
<td>Gateway</td>
<td>Squid, CGI, Reverse Proxy</td>
</tr>
<tr>
<td></td>
<td>Proxy</td>
<td>CERN Proxy, Netscape Proxy, Gauntlet</td>
</tr>
<tr>
<td></td>
<td>User agent</td>
<td>Netscape Navigator, Lynx, MOMspider</td>
</tr>
</tbody>
</table>

### Example 5.1 RESTful Web service in Amazon S3 Interface (S3 - Simple Storage Service)

- **S3 keep fundamental entities**
  - “Object” which are named pieces of data accompanied by some metadata to be store in containers called “bucket”
  - Unique key

- **S3 provides three types of resources accessible through https://s3.amazonaws.com/{name-of-bucket}/{name-of-object}**
  - A list of user buckets
  - A particular bucket
  - A particular S3 object

- **HTTP standard operations**: GET, HEAD, PUT, and DELETE to create, retrieve, or manipulate S3 resources

- **S3 APIs**
**Example 5.1 RESTful Web service in Amazon S3 Interface (S3 - Simple Storage Service)**

**Sample REST Request-Response for Creating an S3 Bucket**

<table>
<thead>
<tr>
<th>REST Request</th>
<th>REST Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT [/bucket-name] HTTP/1.0</td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td>Date: Wed, 15 Mar 2010-14:45:15 GMT</td>
<td>x-amz-id-2:</td>
</tr>
<tr>
<td>Authorization:</td>
<td>VjzdTviQorQtSjcgLshzCZSzn+7CnewvHA</td>
</tr>
<tr>
<td>AWS [aws-access-key-id]:</td>
<td>+6sNxR3VRcUPyO5fmSmo8bWnlIS52qa</td>
</tr>
<tr>
<td>[header-signature]</td>
<td>x-amz-request-id: 91A8CC60F9FC49E7</td>
</tr>
<tr>
<td>Host: s3.amazonaws.com</td>
<td>Date: Wed, 15 Mar 2010 14:45:20 GMT</td>
</tr>
<tr>
<td></td>
<td>Location: [/bucket-name]</td>
</tr>
<tr>
<td></td>
<td>Content-Length: 0</td>
</tr>
<tr>
<td></td>
<td>Connection: keep-alive</td>
</tr>
</tbody>
</table>

**SOAP: Simple Object Access Protocol**

**UDDI: Universal Description Discovery & Integration**

**WSDL: Web Service Description Language**

![Diagram of SOAP and UDDI interaction](image)

Figure 5.2

A simple web service interaction among provider, user, and the UDDI registry.

- BPEL4WS (Business Process Execution Language for Web Services), XML-based
- JMS – Java Message Service, CORBA’s IIOP (Internet Inter-ORB) protocol

Sample SOAP Request-Response for Creating an S3 Bucket

<table>
<thead>
<tr>
<th>SOAP Request</th>
<th>SOAP Response</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;soap:Body&gt;</code></td>
<td><code>&lt;soap:Body&gt;</code></td>
</tr>
<tr>
<td><code>&lt;Bucket&gt;SampleBucket&lt;/Bucket&gt;</code></td>
<td><code>&lt;Bucket&gt;SampleBucket&lt;/Bucket&gt;</code></td>
</tr>
<tr>
<td><code>&lt;AWSAccessKeyId&gt;1B9FVRAYCP1VJEXAMPLE=&lt;/AWSAccessKeyId&gt;</code></td>
<td><code>&lt;AWSAccessKeyId&gt;1B9FVRAYCP1VJEXAMPLE=&lt;/AWSAccessKeyId&gt;</code></td>
</tr>
<tr>
<td><code>&lt;Timestamp&gt;2010-03-15T14:40:00.165Z&lt;/Timestamp&gt;</code></td>
<td><code>&lt;Timestamp&gt;2010-03-15T14:40:00.165Z&lt;/Timestamp&gt;</code></td>
</tr>
<tr>
<td><code>&lt;Signature&gt;Iuyz3d3P0aTou39dzbqaEXAMPLE=&lt;/Signature&gt;</code></td>
<td><code>&lt;Signature&gt;Iuyz3d3P0aTou39dzbqaEXAMPLE=&lt;/Signature&gt;</code></td>
</tr>
<tr>
<td><code>&lt;/soap:Body&gt;</code></td>
<td><code>&lt;/soap:Body&gt;</code></td>
</tr>
</tbody>
</table>
| `</soap:Envelope>` | `</soap:Envelope>`
Ten Areas Covered by the Core WS-* Specifications

<table>
<thead>
<tr>
<th>WS-* Specification Area</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Core Service Model</td>
<td>XML, WSDL, SOAP</td>
</tr>
<tr>
<td>2. Service Internet</td>
<td>WS-Addressing, WS-MessageDelivery, Reliable WSRM, Efficient MOTM</td>
</tr>
<tr>
<td>3. Notification</td>
<td>WS-Notification, WS-Eventing (Publish-Subscribe)</td>
</tr>
<tr>
<td>4. Workflow and Transactions</td>
<td>BPEL, WS-Choreography, WS-Coordination</td>
</tr>
<tr>
<td>6. Service Discovery</td>
<td>UDDI, WS-Discovery</td>
</tr>
<tr>
<td>7. System Metadata and State</td>
<td>WSRF, WS-MetadataExchange, WS-Context</td>
</tr>
<tr>
<td>8. Management</td>
<td>WSDM, WS-Management, WS-Transfer</td>
</tr>
<tr>
<td>9. Policy and Agreements</td>
<td>WS-Policy, WS-Agreement</td>
</tr>
<tr>
<td>10. Portals and User Interfaces</td>
<td>WSRP (Remote Portlets)</td>
</tr>
</tbody>
</table>

Enterprise Multi-tier Architecture: from 3-tier to N tier

FIGURE 5.4
Three-tier system architecture.
OGSA (Open Grid Services Architecture) Service Areas

- **Service Areas**
  - Infrastructure Services
  - Execution Management Services
  - Data Management Services
  - Resource Management Services
  - Security Services
  - Information Services
  - Self-Management Services

- **WSRF (Web Service Framework) Related Specifications**
WSRF (Web Services Resource Framework - OASIS) and Its Related Specifications

<table>
<thead>
<tr>
<th>Specification</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSRF Specifications</td>
<td></td>
</tr>
<tr>
<td>WS-ResourceProperties</td>
<td>Standardizes the definition of the resource properties, its association with the WS interface, and the messages defining the query and update capability against resource properties</td>
</tr>
<tr>
<td>WS-ResourceLifetime</td>
<td>Provides standard mechanisms to manage the life cycle of WS-resources (e.g., setting termination time)</td>
</tr>
<tr>
<td>WS-ServiceGroup</td>
<td>Standard expression of aggregating Web services and WS-Resources</td>
</tr>
<tr>
<td>WS-Basefault</td>
<td>Provides a standard way of reporting faults</td>
</tr>
<tr>
<td>WSRF-Related Speci-</td>
<td></td>
</tr>
<tr>
<td>fications</td>
<td></td>
</tr>
<tr>
<td>WS-Notification</td>
<td>Proposes a standard way of expressing the basic roles involved in Web service publish and subscribe for notification message exchange</td>
</tr>
<tr>
<td>WS-BrokeredNotifica-</td>
<td>Standardizes message exchanges involved in Web service publish and subscribe of a message broker</td>
</tr>
<tr>
<td>tion</td>
<td></td>
</tr>
<tr>
<td>WS-Topics</td>
<td>Defines a mechanism to organize and categorize items of interest for a subscription known as &quot;topics&quot;</td>
</tr>
<tr>
<td>WS-Addressing</td>
<td>Transport-neutral mechanisms to address Web service and messages</td>
</tr>
</tbody>
</table>

Example 5.3 US DoD Net-Centric Services

- Global Information Grid (GIG), [http://www.globalsecurity.org/intell/systems/gig.htm](http://www.globalsecurity.org/intell/systems/gig.htm)

Table 5.6 Core Global Information Grid Net-Centric Services

<table>
<thead>
<tr>
<th>Service or Feature</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise services management</td>
<td>Life-cycle management</td>
</tr>
<tr>
<td>Security, information assurance (IA)</td>
<td>Confidentiality, integrity, availability, reliability</td>
</tr>
<tr>
<td>Messaging</td>
<td>Publish-subscribes important</td>
</tr>
<tr>
<td>Discovery</td>
<td>Data and services</td>
</tr>
<tr>
<td>Mediation</td>
<td>Agents, brokering, transformation, aggregation</td>
</tr>
<tr>
<td>Collaboration</td>
<td>Synchronous and asynchronous</td>
</tr>
<tr>
<td>User assistance</td>
<td>Optimize Global Information Grid user experience</td>
</tr>
<tr>
<td>Storage</td>
<td>Retention, organization, and disposition of all forms of data</td>
</tr>
<tr>
<td>Application</td>
<td>Provisioning, operations and maintenance</td>
</tr>
<tr>
<td>Environmental control services</td>
<td>Policy</td>
</tr>
</tbody>
</table>
### Example 5.4 Services and Standards used in CICC – A Chemical Informatics Grid

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow/Monitoring/Management services</td>
<td>Uses Taverna from the UK e-Science Program/OMII or mashups written in scripting languages.</td>
</tr>
<tr>
<td>Authentication/Authorization</td>
<td>Currently all services are openly available.</td>
</tr>
<tr>
<td>Registry and Discovery</td>
<td>Will inherit registry services from other grids.</td>
</tr>
<tr>
<td>Portal and portlets</td>
<td>Use a JSR 168-based portal.</td>
</tr>
<tr>
<td>File Services</td>
<td>No specialized service. URLs are used for naming files and simple remote download. Services developed for grids can be used for uploads.</td>
</tr>
<tr>
<td>NIH DTP Database Services</td>
<td>Access to the NIH Developmental Therapeutics Program (DTP)'s database of molecular screens against 60 cancer cell lines, a free service by ChemiGrid.</td>
</tr>
<tr>
<td>PubMed Search Service</td>
<td>Searchable online database of medical journal articles. CICC develops harvesting services of the abstracts combined with text analysis applications such as OSCAR3.</td>
</tr>
<tr>
<td>SPRESI Services</td>
<td>Clients/service proxies to the commercial SPRESI service (<a href="http://www.spresi.com/">www.spresi.com/</a>). This scientific database houses molecular and reaction data and references and patents.</td>
</tr>
</tbody>
</table>

### Services and Standards used in CICC

(Continued)

<table>
<thead>
<tr>
<th>Service Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varuna Database Service</td>
<td>Molecular structure and more detailed information (such as force fields).</td>
</tr>
<tr>
<td>VOTables Data Tables Web Service</td>
<td>CICC-developed Web service based on the National Virtual Observatory's VOTables XML format for tabular data.</td>
</tr>
<tr>
<td>Specific applications: BCL, OpenEye, Varuna, AutoGEFF</td>
<td>CICC inherits job management services from other grids (including one based on Apache Ant) for managing the execution of both commercial and in-house developed high-performance computing applications.</td>
</tr>
<tr>
<td>Condor and BirdBath</td>
<td>Examine the use of Condor and its SOAP interface (BirdBath) as a super-scheduler for Varuna applications on the TeraGrid.</td>
</tr>
<tr>
<td>ToxTree Service</td>
<td>Wraps an algorithm for estimating toxic hazards in a particular compound. Useful in combination with other clustering programs in a workflow.</td>
</tr>
<tr>
<td>OSCAR3 Service</td>
<td>Based on OSCAR3 by the WWMM group, performs text analysis on journal articles and other documents to extract (in XML) the chemistry-specific information. SMILES assigned to well-known compounds. Works with traditional database and clustering algorithms.</td>
</tr>
<tr>
<td>CDK Services</td>
<td>CICC has developed a number of simple services based on the Chemistry Development Kit (CDK). These include similarity calculations, molecular descriptor calculations, fingerprint generators, 2D image generators, and 3D coordinate molecular generators.</td>
</tr>
<tr>
<td>OpenBabel Service</td>
<td>Converts between various chemical formats (such as between InChI and SMILES).</td>
</tr>
<tr>
<td>InChI Google</td>
<td>For a given InChI (a string specification of a molecular structure), performs a Google search to return a page-ranked list of matches.</td>
</tr>
<tr>
<td>Key interfaces/standards/software used</td>
<td>WSDL, SOAP (with Axis 1.x), CML, InChI, SMILES, Taverna SCUFl, JSR-168 JDBC Servlets, VOTables</td>
</tr>
<tr>
<td>Unused interfaces/software</td>
<td>WS-Security, JSDL, WSRF, BPEL, OGSA-DAI</td>
</tr>
</tbody>
</table>
5.2 Message-Oriented Middleware

- Enterprise Bus
  - Enterprise Service Bus (ESB),
- Publish-Subscribe Model
- Queuing and Messaging Systems

Two Message Bus Implementation

- Message Styles: SOAP, REST, Java RMI
- MQSeries, WebSphereMQ

FIGURE 5.5
Two message bus implementations between services or using a broker network.
Queuing and Messaging Systems

- JMS (Java Message Service)
- AMQP (Advanced Message Queuing Protocol)
- WebSphereMQ
- MuleMQ
- Cloud-based
  - Amazon Simple Queue
  - Azure Queue

Table 5.8 Comparison of Messaging and Queuing Systems

<table>
<thead>
<tr>
<th>System Features</th>
<th>Amazon Simple Queue</th>
<th>Azure Queue</th>
<th>ActiveMQ</th>
<th>MuleMQ</th>
<th>WebSphere MQ</th>
<th>Narada Brokering</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMQP compliant</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>JMS compliant</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Distributed broker</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Delivery guarantees</td>
<td>Message retained in queue for four days</td>
<td>Message accessible for seven days</td>
<td>Based on journaling and JDBC drivers to databases</td>
<td>Disk store uses one file/channel, TTL purge messages</td>
<td>Exactly-once delivery supported</td>
<td>Guaranteed and exactly-once</td>
</tr>
<tr>
<td>Ordering guarantees</td>
<td>Best effort, once delivery, duplicate messages exist</td>
<td>No ordering, message returns more than once</td>
<td>Publisher order guarantee</td>
<td>Not clear</td>
<td>Publisher order guarantee</td>
<td>Publisher–time-order by Network Time Protocol</td>
</tr>
<tr>
<td>Access model</td>
<td>SOAP, HTTP-based GET/POST</td>
<td>HTTP REST interfaces</td>
<td>Using JMS classes</td>
<td>JMS, Adm. API, and JMS</td>
<td>Message Queue Interface, JMS</td>
<td>JMS, WS-Eventing</td>
</tr>
<tr>
<td>Max. message</td>
<td>8 KB</td>
<td>8 KB</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Buffering</td>
<td>N/A</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Time decoupled delivery</td>
<td>Up to four days; supports timeouts</td>
<td>Up to seven days</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Security scheme</td>
<td>Based on HMAC-SHA1 signature Support for WS-Security 1.0</td>
<td>Access to queues by HMAC SHA256 signature</td>
<td>Authorization based on JAAS for authentication</td>
<td>Access control, authentication, SSL for communication</td>
<td>SSL, end-to-end application-level data security</td>
<td>SSL, end-to-end application-level data security, and ACLs</td>
</tr>
<tr>
<td>Support for Web services</td>
<td>SOAP-based interactions</td>
<td>REST interfaces</td>
<td>REST</td>
<td>REST</td>
<td>SOAP interactions</td>
<td>WS-Eventing</td>
</tr>
<tr>
<td>Transports</td>
<td>HTTP/HTTPS, SSL</td>
<td>HTTP/HTTPS</td>
<td>TCP, UDP, SSL, HTTPS, Multicast, In-MVM, JXTA</td>
<td>Mule ESB supports TCP, UDP, RMI, SSL, SMTP, and FTP</td>
<td>TCP, UDP, Multicast, SSL, HTTP/S</td>
<td>TCP, Parallel TCP, UDP, Multicast, SSL, HTTP/S, IPsec</td>
</tr>
<tr>
<td>Subscription formats</td>
<td>Access is to individual queues</td>
<td>Access is to individual queues</td>
<td>JMS spec allows for SQL selectors also access to individual queues</td>
<td>JMS spec allows for SQL selectors also access to individual queues</td>
<td>JMS spec allows SQL selectors, access to individual queues</td>
<td>SQL selectors, regular expressions, rgs, value-pairs, XQuery and XPath</td>
</tr>
</tbody>
</table>
Example 5.5 Environmental Monitoring and Internet Conference using NaradaBrokering

- The GOAT project at Clemson University, [http://www.clemson.edu/public/impacts/07summer/env/environmental_sensors_track_impact_of_development.html](http://www.clemson.edu/public/impacts/07summer/env/environmental_sensors_track_impact_of_development.html)
  - Environmental sensor track impact of development
- The Program of Integrated Study for Coastal Environmental Sustainability (PISCES)
- The NaradaBrokering Project, Indiana University, [http://www.naradabrokering.org/](http://www.naradabrokering.org/)
- Anabas.com (Internet meeting software), [http://www.anabas.com/](http://www.anabas.com/)

Example 5.6 QuakeSim Project for Earthquake Science

- [http://quakesim.org](http://quakesim.org)
5.3 Portals and Science Gateways

FIGURE 5.8
A gateway component software stack for scientific applications.

The HUBzero Architecture

FIGURE 5.9
The HUBzero architecture and its major functional components.
5.4 Discovery, Registries, Metadata, and Databases

Information about the relationship between two parties

Business Entity

Technical information about the service interfaces

Business Service

URL

Model

Model

Binding Template

Model

Binding Template

Model

Human readable information about the service publisher

Business Entity

Descriptive information about a particular service

Business Service

Binding Template

URL

Service descriptions

FIGURE 5.10
UDDI entities and their relationship.

Databases and Publish-Subscribe Model

Oracle publish-subscribe model.

Application

Publish

Oracle

Subscribe

Application

Rules and transformation

Advanced Queue

Application

Publish

Subscribe

Application

Subscribe

Application

FIGURE 5.11
Oracle publish-subscribe model.
Semantic Web and Grid

Semantic grid-related concepts and technologies.
(Courtesy of Goble and Roue, 16th European Conference on Artificial Intelligence (ECAI-2004), Valencia, Spain, 2004 [35])

Semantic Grid Architecture

Semantic grid architecture.
(Courtesy of Goble and Roue, 16th European Conference on Artificial Intelligence (ECAI-2004), Valencia, Spain, 2004 [35])
5.5 Workflow in SOAs
The Grid of Grids of Services

![Diagram showing the grid of grids of services]

**Figure 5.14**
The concept of the grid of grids of services.

---

Basic Workflow Concepts

![Diagram showing the transition from traditional programming to workflow]

**Figure 5.15**
Hierarchical computing, data, and programming abstraction.
<table>
<thead>
<tr>
<th>Standard</th>
<th>Link</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WS-CDL</strong> Web Service Choreography Description Language (W3C)</td>
<td><a href="http://www.w3.org/TR/ws-cdl-10/">http://www.w3.org/TR/ws-cdl-10/</a></td>
<td>November 2005, not final</td>
</tr>
<tr>
<td><strong>WS-CDLI</strong> Web Service Choreography Interface V 1.0 (W3C)</td>
<td><a href="http://www.w3.org/TR/ws-cdl/">http://www.w3.org/TR/ws-cdl/</a></td>
<td>August 2002, note only</td>
</tr>
<tr>
<td><strong>WSCL</strong> Web Services Conversation Language (W3C)</td>
<td><a href="http://www.w3.org/TR/wscl10/">http://www.w3.org/TR/wscl10/</a></td>
<td>March 2002, note only</td>
</tr>
<tr>
<td><strong>WSFL</strong> Web Services Flow Language</td>
<td>replaced by BPEL</td>
<td></td>
</tr>
<tr>
<td><strong>XLANG</strong> Web Services for Business Process Design (Microsoft)</td>
<td><a href="http://msdn.microsoft.com/library/wes-wf2/">http://msdn.microsoft.com/library/wes-wf2/</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Standard</th>
<th>Link</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>WS-Coordination</strong> Web Services Coordination (BEA, IBM, Microsoft at OASIS)</td>
<td><a href="http://docs.oasis-open.org/ws-tv/wscoord/2006/06">http://docs.oasis-open.org/ws-tv/wscoord/2006/06</a></td>
<td>February 2009</td>
</tr>
<tr>
<td><strong>WS-AtomicTransaction</strong> Web Services Atomic Transaction (BEA, IBM, Microsoft at OASIS)</td>
<td><a href="http://docs.oasis-open.org/ws-tv/wsat/2006/06">http://docs.oasis-open.org/ws-tv/wsat/2006/06</a></td>
<td>February 2009</td>
</tr>
<tr>
<td><strong>WS-BusinessActivity Framework</strong> (BEA, IBM, Microsoft at OASIS)</td>
<td><a href="http://docs.oasis-open.org/ws-tv/wsba/2006/06">http://docs.oasis-open.org/ws-tv/wsba/2006/06</a></td>
<td>February 2009</td>
</tr>
</tbody>
</table>
A Typical (Load and Merge) Workflows

FIGURE 5.16
Two typical (Load and Merge) workflows from the Pan-STARRS astronomy data processing area.

(Courtesy of Berga et al., 2021)

Example 5.10 Pan-STARTS Workflow

FIGURE 5.17
The workflow in the Taverna System showing.

(Courtesy of D. Gothe, European Conf. on Research and Advanced Technology for Digital Libraries, 2006 [1.3.2])
A Workflow Graph

FIGURE 5.18
A workflow graph that includes subgraphs illustrating pipelines and loops.

Example 5.11 Swift Workflow System Architecture

FIGURE 5.19
Swift workflow system architecture. (Courtesy of www.ci.uchicago.edu/swift/ (2003))
Cloud Computing Service Provider Priorities

- Ensure confidentiality, integrity, and availability in a multi-tenant environment.
- Effectively meet the advertised SLA, while optimizing cloud resource utilization.
- Offer tenants capabilities for self-service, and achieve scaling through automation and simplification.

Figure 5.20
Application of the Swift to bioinformatics.
**Guiding Principles in Using Clouds for Data-Intensive Applications** (G. Fox, 2011)

- Clouds may not be suitable for everything but they are suitable for majority of data intensive applications
  - Solving PDEs on 100,000 cores needs classic MPI clusters
- Cost effectiveness, elasticity and quality programming model will drive use of clouds in many areas such as genomics
- Need to solve issues of
  - Security-privacy-trust for sensitive data
  - How to store data – “data parallel file systems” (HDFS), Object Stores, or classic HPC approach with shared file systems with Lustre etc.
- Programming model which is likely to be MapReduce based
  - Look at high level languages
  - Compare with databases (SciDB)
  - Must support iteration to do “real parallel computing”
  - Need Cloud- HPC Cluster Interoperability

---

**SOA Security**

- SOA security is essentially the web services security
- Three core specifications
  - WS-Security, XML-Signature, XML-Encryption
  - WS-* Security offers a new technology for SOA security
- Single sign-on (SSO) is a form of centralized security mechanism
- Related specifications for SOA security
  - WS-Security, WS-SecurityPolicy, WS-Trust, WS-SecureConversation,
  - WS-Federation, XACML, Extensible Rights Markup Language, XML Key Management,
  - XML, Signature, SAML, .NET Passport, Secure Socket Layer, WS-I Basic Security Profile
Example: Geo-Information Service (GIS) Cloud

Major Tasks to Build a GIS Cloud for Community Services

- Must build fast mining engine to support data-mining and data aggregation
- Community or virtual community discovery
- Support of bandwidth demand for video, voice, data, etc.
- Providing GISs and mashup services with GIS
- Handling of massive registered users and their payments
- Interoperability among different GISs in the industry
- Centralized, domain-oriented, and multi-granularity services, make GIS centers public and specialized.
Conclusion