Motivation

- Why Service Based Development?
  - Do we need a new methodology? The enterprise can't handle agile development. Product lines?
- What happen to Past Efforts?
  - Is OO dead?
  - Is CORBA dead?
- Who will be successors?
  - Web Services?
  - Grid Services?
  - Peer-to-Peer?
- Knowledge Engineering will replace Software Engineering?
Is SOA an implementation?

Why do we need SOA?
Monolithic
How many physical tiers?

Client-Server
How many physical tiers?
N-Tier
How many physical tiers?

Internet
How many physical tiers?
## Properties

<table>
<thead>
<tr>
<th></th>
<th>Monolithic</th>
<th>Client-Server</th>
<th>N-Tier</th>
<th>WWW</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Formats</strong></td>
<td>Proprietary</td>
<td>Proprietary</td>
<td>Open</td>
<td>Standard</td>
</tr>
<tr>
<td><strong>Protocols</strong></td>
<td>Proprietary</td>
<td>Proprietary</td>
<td>Open</td>
<td>Standard</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td><strong>Number of nodes</strong></td>
<td>Very Small</td>
<td>Small</td>
<td>Medium</td>
<td>Huge</td>
</tr>
<tr>
<td><strong>Pervasiveness</strong></td>
<td>Not</td>
<td>Somewhat</td>
<td>Somewhat</td>
<td>Extremely</td>
</tr>
</tbody>
</table>

## Software Architecture History

- Structured Development
- Object Oriented Development
- Component Based Development
- Service Based Development
Structured Development
How abstract? How coarse?

Object Oriented Development
How abstract? How coarse?
Component Based Development
How abstract? How coarse?

Service Based Development
## Properties

<table>
<thead>
<tr>
<th></th>
<th>Structured</th>
<th>Objects</th>
<th>Components</th>
<th>Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Granularity</strong></td>
<td>Very Fine</td>
<td>Fine</td>
<td>Medium</td>
<td>Coarse</td>
</tr>
<tr>
<td><strong>Contract</strong></td>
<td>Defined</td>
<td>Private/Public</td>
<td>Public</td>
<td>Published</td>
</tr>
<tr>
<td><strong>Reusability</strong></td>
<td>Low</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td><strong>Coupling</strong></td>
<td>Tight</td>
<td>Tight</td>
<td>Loose</td>
<td>Very Loose</td>
</tr>
<tr>
<td><strong>Dependencies</strong></td>
<td>Compile-Time</td>
<td>Compile-Time</td>
<td>Compile-Time</td>
<td>Run-Time</td>
</tr>
<tr>
<td><strong>Communication Scope</strong></td>
<td>Intra-Application</td>
<td>Intra-Application</td>
<td>Inter-Application</td>
<td>Inter-Enterprise</td>
</tr>
</tbody>
</table>

## SOA Entities

How do these entities interact?

1. **Service Consumer**
2. **Service Provider**
3. **Contract**
4. **Registry**

Diagram:

- Service Consumer
- Contract
- Service Provider
- Registry

Connections:
- Send and Receive
- Publish
- Find
Register, Find, Bind and Execute

- The *find, bind and execute* (Talking Blocks, 2001) paradigm allows the consumer of a service to ask a third-party registry for the service that matches its criteria.
- If the registry has a service that *matches* its criteria, then the registry gives the consumer a *contract* and an *endpoint address* for the service.
- SOA consists of *entities* (i.e., services, registries, contracts, and proxies) that are configured together to support the find, bind

Service Consumer

What is it?

- an application, service or some other type of software module that requires a service.
  - *initiates the locating of the service* in the registry, binding to the service over a transport and executing the service function.
  - *executes the service by sending it a request* that is formatted according to the format specified in the contract.
Service Provider
What types are there?

- It is the *network addressable entity* that accepts and executes requests from consumers.
  - It may be a *mainframe system*, a *component* or some other type of software component
  - *publishes its contract* in the registry for access by service consumers.
  - *executes the service request.*

Service Registry
What does it do?

- A service registry is a network-based directory that contains available services.
  - *accepts* and *stores contracts* from service providers
  - *provides* those contracts to interested service consumers.
- 3rd Party Binding
Service Contract

What does it specify?

- A contract is a specification of the way a consumer of a service will interact with the provider of the service.
  - specifies the format of the request and response from the service.
  - require a set of preconditions and post conditions. The preconditions and post conditions specify the state that the service must be in for the service to execute a particular function.

Contract

- Contract is an abstract agreement such as CORBA IDL that is used at both compile-time and run-time.
- Contains only functional aspects, not technical.
  - QoS and endpoint address is separate
Service QoS
Why are these important?

- The service may also specify quality of service (QoS) levels.
- QoS levels are specifications for the non-functional aspects of the service. For instance, a quality of service attribute is the amount of time that it takes to execute a service method.

QoS

- Availability
- Reliability
- Integrity
- Performance
- Regulatory
- Security
Interfaces

Public (defined)
Published (formal)
Contractual

Coarser grained means more formal interfaces

Granularity and Dependencies

Modularity
Optional But Crucial Entities

- Service Proxy
- Service Lease

Service Proxy

- The *service provider* supplies a service proxy to the service consumer.
- The *service consumer* executes the request by calling an API function on the proxy.

- The *service proxy*
  - *finds a contract and a reference* to the service provider in the registry.
  - *formats* the request message and executes the request on behalf of the consumer.
Service Proxy (Cont)

- Advantages
  - a convenience entity for the service consumer - *enhance performance* by caching remote references and data.
  - When a proxy caches a remote reference, subsequent service calls will *not require additional registry calls*.
  - By storing service contracts locally, the consumer *reduces the number of network hops* required to execute the service.
  - Strongly typed to strongly tagged
  - Early binding/late binding
Service Lease

- The service lease specifies the amount of time that the contract is valid.
  - The registry grants the service lease to the service consumer.
  - The contract is valid only from the time the consumer requests the contract from the registry to the time specified by the lease (Sun Jini™ Technology Core Specification 2001).
  - When the lease runs out, the consumer must request a new lease from the registry.
- Extremely loose coupling

Service Topologies

- Simple Service
- Composite Service
- Middleware Service
- Service Bus
Simple Service
What are some examples?

Middleware Service
Provides transformation of internal interfaces to external contracts.
Service Bus

Simple Component Based Service
SOA Benefits

- Reuse
- Better Return-on-Investment (ROI)
- Faster time to market
- Higher quality
- Developer Productivity
- Interoperability

SOA Drawbacks

- Performance
- Requires enterprise focus
- QA is difficult
- Security
- Service Junk Drawer
Web Services
Do you need Web Services?

- Past Efforts: DCE, CORBA, RMI, DCOM, EJB
  - Failed to provide interoperability
  - Failed to get through the firewall

### Web Services Technology Stack

<table>
<thead>
<tr>
<th>Layer/Description</th>
<th>Implementation(s)</th>
<th>Other Concerns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Messaging</td>
<td>Electronic Business XML Initiative (ebXML)</td>
<td></td>
</tr>
<tr>
<td>Service Composition</td>
<td>Business Process Execution Language for Web Services (BPEL4WS)</td>
<td></td>
</tr>
<tr>
<td>Service Registry</td>
<td>Universal Description, Discovery and Integration (UDDI)</td>
<td></td>
</tr>
<tr>
<td>Service Description</td>
<td>Web Services Description Language (WSDL)</td>
<td></td>
</tr>
<tr>
<td>Messaging</td>
<td>Simple Object Access Protocol (SOAP)/Extensible Markup Language (XML)</td>
<td></td>
</tr>
<tr>
<td>Transport</td>
<td>Hypertext Transfer Protocol (HTTP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Simple Mail Transfer Protocol (SMTP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>File Transfer Protocol (FTP)</td>
<td></td>
</tr>
</tbody>
</table>

### Standards

The best thing about standards is that there are so many to choose from

<table>
<thead>
<tr>
<th>Organization</th>
<th>Web Site</th>
<th>Description</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>World Wide Web Consortium (W3C)</td>
<td><a href="http://www.w3c.org">http://www.w3c.org</a></td>
<td>The W3C is a consortium of over 500 member organizations that sets standards for the Internet</td>
<td>XML, SOAP, HTTP</td>
</tr>
<tr>
<td>Web Services Interoperability Organization (WS-I)</td>
<td><a href="http://www.ws-i.org">http://www.ws-i.org</a></td>
<td>WS-I is a consortium of mostly vendor companies focusing on Web services standards</td>
<td>BPEL4WS, WS-Security, WS-Transaction, WS-Coordination, WS-Attachments</td>
</tr>
<tr>
<td>Organization for the Advancement of Structured Information Standards (OASIS)</td>
<td><a href="http://www.oasis-open.org">http://www.oasis-open.org</a></td>
<td>OASIS is an consortium that focuses on the development of e-business standards</td>
<td>WS-Inspection, WS-Referral, WS-Routing, UDDI</td>
</tr>
<tr>
<td>UN/CEFACT (United Nations Centre for Trade Facilitation)</td>
<td><a href="http://www.unece.org/cefact">http://www.unece.org/cefact</a></td>
<td>The UN/CEFACT and OASIS are joint committees on the ebXML standard. The UN/CEFACT is sponsored by the UN and focuses on standards for...</td>
<td>ebXML</td>
</tr>
<tr>
<td>The Internet Engineering Task Force (IETF)</td>
<td><a href="http://www.ietf.org/">http://www.ietf.org/</a></td>
<td>The IETF is a consortium of vendors, universities and individuals. It handles mostly low-level internet protocol standards</td>
<td>DIME</td>
</tr>
</tbody>
</table>

39

40
Java and Web Services

- Java API’s JAX* Pack
  - JAXP
    - XML parsing
  - JAXB
  - JAX-RPC
  - JAXM
    - Document oriented messaging
  - SAAJ
    - SOAP with attachments
  - JAXR –
    - Registries (UDDI and ebXML)

Think about these….

- Why Service Based Development?
  - Do we need a new methodology? The enterprise can’t handle agile development. Product lines?
- What happen to Past Efforts?
  - Is OO dead?
  - Is CORBA dead?
- Who will be successors?
  - Web Services?
  - Grid Services?
  - Peer-to-Peer?
- Knowledge Engineering will replace Software Engineering?
References

- Herzum, Peter. Tutorial #20: Web Services and Service Oriented Architectures. OOPSLA 2002, Seattle, WA.
References