Reinventing the Wheel?
CORBA vs Web Services

Aniruddha Gokhale, Bharat Kumar, Arnaud Sahuguet
Why Bother? (1)

- Real-life systems are complex
  - e.g. telecommunication (see previous slide),
  - E-commerce, banking and finance, healthcare, etc.
- Complex systems cannot be built as one single standalone application
- Complex systems require
  - Distributed applications
  - Interoperability
  - Location transparency
  - Ease of programming to avoid accidental complexities
Why Bother? (2)

◆ CORBA, technology of choice for distributed applications
  ■ Numerous success stories
  ■ Well accepted and active standard
  ■ Used in most mission critical applications

◆ Web Services, a new emerging technology
  ■ Unprecedented hype
  ■ Support from the major players (IBM, Microsoft, SUN)
  ■ Leverage on the XML hype
  ■ An evolution of the “Web-way” of doing things

◆ Key issues
  ■ How do both technologies compare?
  ■ When to use which?
  ■ Convergence between both technologies
Roadmap of this talk

- CORBA in a nutshell
- WS in a nutshell
- Side by side comparison
- Applicability of CORBA and WS
- CORBA / WS interoperability
- Conclusion
CORBA in a nutshell
CORBA in a nutshell (1)

- CORBA = Common Object Request Broker Architecture
  - 1.0: 1991; 2.0: 1996; 2.3: 1998; 3.0: 1999
- Open standard (Object Management Group)
- CORBA is an object bus
  - client can invoke methods on remote (server) objects
    - independently of the language the objects are written in
    - independently of the location of the objects
- Client-Server mediation via object request brokers (ORBs)
- Communication via IIOP
- Capabilities of objects defined by Interface Definition Language (IDL)
- CORBA services: naming, trading, security, persistence, events
CORBA in a nutshell (2)

◆ Life-cycle of a CORBA application
  ■ Define the service as IDL interfaces
  ■ Compile the IDL to generate stub and server skeletons
  ■ Implement the service and associate it with the skeletons via the Portable Object Adapter
  ■ Publish the service with a Naming or Trading service

◆ Client processing
  ■ Contact Naming service to get appropriate object reference
  ■ Invoke operations (static or dynamic) on the object reference via stubs
  ■ Process incoming reply or exception
CORBA in a nutshell (3)
WS in a nutshell
WS in a nutshell (1)

- Web Services are an emerging middleware technology based on a simple XML-based protocol (SOAP)
- Web Services = suite of technologies WS-xx
- Web Services described in terms of messages accepted and generated using WS-Description Language (WSDL).
- WS focuses heavily on service discovery (UDDI).
WS in a nutshell (2)
Side by Side Comparison

Keep in mind that one can implement Web Services on top of CORBA or CORBA on top of Web Services.
## Side by Side Comparison (1)

<table>
<thead>
<tr>
<th></th>
<th>CORBA</th>
<th>Web Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type System</strong></td>
<td>IDL (static + runtime checks)</td>
<td>XML Schemas (runtime checks only)</td>
</tr>
<tr>
<td><strong>Transfer Syntax</strong></td>
<td>CDR (binary)</td>
<td>XML (UTF)</td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>Stateful</td>
<td>Stateless</td>
</tr>
<tr>
<td><strong>Registry</strong></td>
<td>Interface repository</td>
<td>UDDI/WSDL</td>
</tr>
<tr>
<td></td>
<td>Implementation repository</td>
<td></td>
</tr>
<tr>
<td><strong>Service Discovery</strong></td>
<td>CORBA naming/trading service</td>
<td>UDDI</td>
</tr>
<tr>
<td><strong>Security</strong></td>
<td>CORBA security service</td>
<td>HTTPS, XML signature</td>
</tr>
<tr>
<td><strong>Firewall Tunneling</strong></td>
<td>Work in progress</td>
<td>Over HTTP</td>
</tr>
</tbody>
</table>
### Side by Side Comparison (2)

<table>
<thead>
<tr>
<th>CORBA stack</th>
<th>Web Services stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDL</td>
<td>WSDL</td>
</tr>
<tr>
<td>CORBA Services</td>
<td>UDDI</td>
</tr>
<tr>
<td>CORBA stubs/skeletons</td>
<td>SOAP messages</td>
</tr>
<tr>
<td>CDR binary encoding</td>
<td>XML UTF encoding</td>
</tr>
<tr>
<td>GIOP/IIOP</td>
<td>HTTP</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>TCP/IP</td>
</tr>
</tbody>
</table>
Applicability of CORBA and WS

Which one to use, when?
Applicability based on

- **Web interfaces**
  - XML is the Web data model $\rightarrow$ WS

- **Secure architecture with firewalls**
  - HTTP is usually accepted by firewall $\rightarrow$ WS
  - But a lot of WS related traffic on port 80 will create problems

- **Legacy components (e.g. other CORBA, EJBs, etc.)**
  - CORBA component model superset of EJB $\rightarrow$ CORBA

- **State**
  - State captured by object instances $\rightarrow$ CORBA
  - + CORBA persistence and transaction services
Applicability based on

- **Mobile environment**
  - Disconnected environments favor stateless protocols
  - SOAP has a notion of message routing $\rightarrow$ WS

- **Thin clients**
  - CORBA requires ORB libraries (all or nothing)
  - WS only require to send/receive messages $\rightarrow$ WS

- **Proxies**
  - Changes in routing of method calls requires ORB changes
  - SOAP is proxy friendly (message rewriting) $\rightarrow$ WS

- **Performance**
  - CORBA more mature + binary encoding $\rightarrow$ CORBA
  - WS are more at the level of prototypes and betas
Applicability based on

- **Human factor**
  - Learning curve ??
  - Past experience ??
  - Future will tell.

- **Maturity**
  - CORBA: > 10 years
  - WS: < 2 years
CORBA / WS
Interoperability
Why is it important

◆ Revenue Growth
  ■ Cost of phone calls is dropping
  ■ Carriers are looking for new revenue creating services
  ■ Convergence of traditional telephony services and web services is the future

◆ Motivating example
  ■ Mobile Restaurant Locator service
  ■ Location info from wireless service provider (CORBA interface)
  ■ Restaurant info from Web site (e.g. Zagat)

◆ Issues
  ■ Protocol translation
  ■ Mapping between CORBA and WS data models
Possible scenario

- SOAP request parsed
- Gateway looks up IDL description
- Gateway looks up WSDL description of the SOAP request
- A dynamic CORBA request is built and sent to the server using DII
- SOAP response is built out of the CORBA response
Conclusion
Conclusions (1)

- **Distributed systems inherently complex**
  - No one-size-fits all solution
  - No silver bullet, despite all the hype around WS

- **CORBA = mature technology (around for 10 years)**
  - CORBA value lies in CORBA services, platform and language independence, interoperability

- **WS = emerging technology (invented < 2 years ago)**
  - The only service offered by WS is UDDI

- **WS wants to replace CORBA but represents a limited subset of what CORBA already offers today:**
  - Discovery (UDDI)
  - No support for transaction, persistence, security, load-balancing, etc.
Conclusions (2)

- Danger of over simplification
  - WS as middleware layer on top of CORBA
  - There are examples where CORBA is middleware on top of WS-like layer (e.g. SIP protocol)

- XML does not mean WS
  - XML can be used with CORBA

- CORBA & WS not mutually exclusive but complementary
  - CORBA-SOAP and SOAP-CORBA gateways
  - Automatic mapping between IDL and WSDL