Web Services: Delivering on the Promise of Distributed Systems

OOPSLA 2002

Alfred Z. Spector
Vice President, Services & Software Research
IBM Corporation

aspector@us.ibm.com
11/8/02
Abstract

The study of distributed systems in computer science is now many decades old. While concepts of directories, remote procedure call, distributed load balancing, etc. have been around a long time, it has been very difficult to create a successful general purpose platform that supports real distributed applications. In various generations of technology, we have seen the continual reinvention of RPC (e.g., DCE, DCOM, CORBA, RMI, and IIOP) not to mention all the variations on message passing, transaction processing, security models, etc. that seem necessary for real distributed applications. In each generation, we have improved on the flexibility and dynamic nature of distributed interactions, with an eye to increasing their generality and commercial viability. Yet each generation has also stumbled on the way to achieving a usable, open, heterogeneous, distributed environment. Some systems failed because of incompleteness; some because of complexity; some because of inadequate tools; and others because they were proprietary or could not support heterogeneity.

Finally, with the advent of Web Services from the W3C, the promise of a widely accepted, open, heterogenous, distributed environment may be at hand. Significantly, Web Services depend on the key proposition taught to us by the object oriented community - that scalability and designability depend on encapsulation (with clean interfaces) to reduce complexity. But we must also learn from the dark side of OO: we already know that good tools, useability, and performance will make or break the success of Web Services. Experiences from the history of both distributed systems and object orientation can teach us quite a lot about how to succeed this time with Web Services. This talk will place Web Services in the context of distributed computing's history and propose a roadmap for how it can succeed as the basis for the open platform for distributed computing in the future.
Outline

- Distributed Systems
- A Historical Perspective
- Web Services and Critical Success Factors
  - An Integrated Approach: Runtime & Tools
- The Tools Dimension: *Eclipse: Of the Community, By the Community, For the Community*
- A Preview of 4 Advanced Eclipse Technologies from IBM Research
- Announcing a new ~$500,000 *Eclipse Innovation Program* (2002 money)
- Conclusions
Distributed Systems

Three definitions:

- A computer science discipline aimed at reducing the cost of developing & managing applications that utilize multiple, networked systems
- Particular technology (infrastructure specifications or software) aimed at reduced development & management cost
- A complete system that implements a particular distributed application
3 Dimensions of Distributed Systems
3 Dimensions of Distributed Systems

The Distributed Systems Cube

- Function Shipping
- Data Sharing
- Lifecycle & Manage
- Create
- Programming Lang’s & Tools
- Infrastructure Runtime
## Architecture

<table>
<thead>
<tr>
<th>Function Shipping</th>
<th>Data Sharing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advantages:</strong></td>
<td><strong>Ease of Use</strong></td>
</tr>
<tr>
<td>- OO Nature</td>
<td>- Predefined capabilities</td>
</tr>
<tr>
<td>- Flexibility</td>
<td>- Distr. File Systems</td>
</tr>
<tr>
<td><strong>Examples:</strong></td>
<td>- Distr. Database</td>
</tr>
<tr>
<td>- DCE, Corba, RMI, DCOM</td>
<td>- Initial Web Appeal</td>
</tr>
<tr>
<td>- Eventual Web Appeal</td>
<td></td>
</tr>
</tbody>
</table>

- Note: There really is a continuum
  - Databases have stored procedure
  - Object-oriented have dual flavors
  - The Web today utilizes both
Goals

<table>
<thead>
<tr>
<th>Create</th>
<th>Lifecycle &amp; Manageability</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Primary focus of academic computer science</td>
<td></td>
</tr>
<tr>
<td>- Plethora of techniques now required</td>
<td></td>
</tr>
<tr>
<td>- Blending of runtime &amp; tools</td>
<td></td>
</tr>
<tr>
<td>- The greater cost</td>
<td></td>
</tr>
<tr>
<td>- Plethora of tools now required</td>
<td></td>
</tr>
<tr>
<td>- Blending of runtime &amp; tools</td>
<td></td>
</tr>
</tbody>
</table>

- Note: This community tends to focus on development
- Lifecycle & Manageability are at least equal problem
- But proper development key to manageability
- *Autonomic Computing* should be a greater academic focus
Traditionally, a community unto itself

Often overly focused on some programming issues, while ignoring essential distributed semantics

Traditionally, a community unto itself

Not focused on programming, rather on algorithms, architecture & semantics

• Note these are hard problems:
  • Programming gets far harder if one tries to include issues adaptability, reliability and availability, concurrency, ...  
  • Runtime infrastructures can be remarkably difficult to use
A Historical Perspective

- Distributed computing fundamentals known for at least 25 years
- Early great papers & collections – some favorites:
  - Using Encryption for Authentication in Large Networks of Computers, R. Needham and M. Schroeder, CACM, 21(12), 12/78
- The Distributed Systems summer course 1982-1989
1980’s Western Institute of Computer Science Distributed Systems Course Topics

- A. Spector / D. Gifford / R. Rashid, (Gray replaced Spector in late 80’s)
- Name Service
- Function shipping: Message Passing & Remote Procedure Call
- Data sharing techniques
- Authentication, authorization, privacy, Integrity/availability
  - Transactions
  - Replication
- Caching
- Related programming language issues
Results of Our Research?

- Many attempts to create a successful world-wide distributed computing standard
  - Mach Projects
  - DCE
  - CORBA/IIOP
  - DCOM
  - RMI
  - Above did not have the desired impact

- HTTP & HTML with DNS had impact!
  - Not based on breadth of distributed computing research
  - And yet, wildly successful!
Why Limited Impact of “Correct” Systems?

- Providing a complete programming, operating, & management environment for distributed computing is remarkably difficult
- Homogenized worlds unattainable
- There was minimal economic incentive to extend technology and & work out details
- Incomplete technology was not fully there
- The technical community tended to fragment along traditional lines
Why the Great Impact of HTML/HTTP?

- Simple goals → an elegant implementation
  - No security implied, paradoxically, lessened deployment resistance
  - Simple models supported use with few, if any, tools
  - Retooling of programs not required
- Significant economic value realizable without massive expense
- Worked with extant technology
- Limited need for a concerted effort by worldwide technical community
Limitations of HTTP/HTML

- Data presentation formats focused on presentation, not meaning
- Implied programming model very ad hoc
  - Difficult to use state-of-the-art programming models
  - In fact, much programming a throwback to the past
- Weak security
- Behind the curve in supporting multi-domain services models, such as IBM’s On Demand e-Business Vision
Enter Web Services

- Standards addressing most aspects of the cube
  - Greater focus on heterogeneity than previous efforts
    - Support of multiple communication protocols
    - Support for arbitrary object hierarchies
  - Extensible data model: seamless
    - From storage abstraction
    - To communication standard
  - Breadth of focus
  - Economic incentive

- Attention to protocol and programming model, alike
Web Services (partial picture)

- BPEL4WS
- Reliable Messaging
- Security
- Transactions
- Coordination
- WSDL, UDDI, Inspection
- SOAP (Logical Messaging)
- Other protocols
- XML, XSL, … (Encoding)
- Other services
- Transports
- Business Processes
- Quality of Service
- Policy
- Description
- Messaging
- Transport
Some Observations (1)

- There is greater care, honesty, and creativity in Web Services than in previous approaches:
  - E.g. Service Level Agreement
    - Distributed systems research traditionally discussed transparency of services
    - In reality, transparency of services not exactly desired: Service Level Guarantees are desired
  - E.g., The world is inevitably heterogeneous at one or more levels
  - E.g., XML is audacious and creative
  - E.g., Security standards are more complete
Some Observations (2)

- Web Services have significant complexity and success will require endurance
  - Performance not trivially achieved
  - Breadth of standards are not easily implemented nor coherently maintained
  - Complex integration required with other worlds; e.g., J2EE, Windows
  - Architecture and collection of requisite standards still growing in size
  - **Most importantly, great tools are needed**
A Tale of 3 Projects (mid-80’s):

- **Argus: Language**
  - B. Liskov, et al. (MIT)
  - Influenced by CLUE
  - Linguistic approach to reduce cost of developing reliable, OO, distributed applications – *integrated runtime*

- **Camelot: Middleware**
  - A. Spector, et al. (CMU)
  - Influenced by TABS, DRDBMS, etc.
  - Systems approach to reduce cost of developing reliable, OO, distributed applications – *language independent*

- **Avalon: A language** – Moving in the right direction
  - J. Wing, et al. (CMU)
  - Fused language with Camelot *runtime*

- **Note** (Other projects: Haskin et al. *Quicksilver*, McKendry/Allchin *Clouds*)
Combining Tools & Web Services

- At IBM, we understand this point
- Our Websphere Studio Tool makes implementing and deploying Web Services easy
- An example of a simple application
public class UpperCase {

    public String convertToUpperCase(String inputString) {
        if (inputString != null) {
            return inputString.toUpperCase();
        } else {
            return null;
        }
    }

    */
    */

    public static void main(String[] args) {
        UpperCase uc = new UpperCase();
        System.out.println(uc.convertToUpperCase("Hello World"));
    }

    * Author: John Smith

    * This is a simple example to demonstrate the Web Service capabilities of Eclipse platform
    */

    publicclass UpperCase {
    /*
    */
    public static void main(String[] args) {
        UpperCase uc = new UpperCase();
        System.out.println(uc.convertToUpperCase("Hello World"));
    }

    *
    */

    public class UpperCase {
        /*
        */
        public static void main(String[] args) {
            UpperCase uc = new UpperCase();
            System.out.println(uc.convertToUpperCase("Hello World"));
        }

        *
        */

        public class UpperCase {
            /*
            */
            public static void main(String[] args) {
                UpperCase uc = new UpperCase();
                System.out.println(uc.convertToUpperCase("Hello World"));
            }

            *
            */
```java
public String convertToUpperCase(String inputString) {
    if (inputString != null) {
        return inputString.toUpperCase();
    } else {
        return null;
    }
}
```

**Deploying Web Service From WebSphere Studio**
Web Service Creation Step

Web Service Java Bean Identity

Configure the Java bean as a Web service.

Web service URL: http://tempuri.org/UpperCase

Scope: Application

- Use static methods
- Use secure SOAP (WebSphere only)

Folder: [UpperCase]

ISD file name: webApplication/WEB-INF/isd/java/UpperCase.isd

WSDL service document name: webApplication/wsdl/UpperCase-service.wsdl

WSDL binding document name: webApplication/wsdl/UpperCase-binding.wsdl

WSDL schema document name: webApplication/wsdl/UpperCase-schema.xsd

Namespace

SOAP deployment descriptor

Web Service Descriptions

Data Mapping
Web Service has been deployed. Studio integrates Client-server for immediate testing.

Service Proxy generated automatically – simplifies usage.

Invoke the Proxy for Service

Test Client

Results Window
The Tools Dimension

Many more types of tools will be required to make Web Services easy enough:

- Service modeling and definition
- Deployment
- XML
  - Data modeling and definition
  - Transformation
- Debugging
- Performance evaluation
- Quality of service specification
- Tools specialized for particular customer segments
  - Industry-specific schemas will often have their own tools

⇒ Quality tools will be required
Because of the diversity and creativity required, we think an open-source development community is best. Eclipse Project has built a universal platform for integrating development tools. Its open, extensible architecture is based on plug-ins. Brought to market by commercial offerings. With source licensed for royalty free world-wide distribution.
Industry Support for the Eclipse Platform

- Altoweb
- Asist
- Borland
- Bowstreet
- BrowserSoft
- Cast Software
- CollabNet
- CommerceQuest
- Computer Associates
- Compuware
- Crystal Decisions
- Embarcadero
- Fujitsu
- Holosofx
- HP
- IBM
- Instantiations
- Interwoven
- LegacyJ
- LogicLibrary
- Macromedia
- Merant
- Mercury Interactive
- Midcomp International
- MKS
- NeuVis
- Peregrine
- QNX
- Rational Software
- Red Hat
- Serena
- Sitraka
- Skyva
- Starbase
- SuSE
- Sybase
- Telelogic
- TogetherSoft
- Versant
- Versata
- Wily Technology

"Rational Software has been an early adopter of the Eclipse Platform because we believe it delivers high value to software teams. We've been working closely with IBM to integrate our products into Eclipse...."

Dave Bernstein, Sr VP of Products, Rational Software

"The Eclipse community will allow developers to collaborate on enhancements to the Workbench technology and more rapidly integrate it within their own applications."

Brian Behlendorf, CTO, CollabNet

"This has long been a dream of developers, and now IBM is providing the integration framework to make it possible."

Arun Gupta, CTO, NeuVis Software

www.eclipse.org

** Board members
Download Requests

- 2.5 million since November 2001
  - 50K in 48 hours after posting Eclipse R2 distributions

- From over 300,000 unique net addresses
  - Most represent firewalls fronting multiple users
  - One download can serve an entire organization

- From over 125 countries
Eclipse Demonstrations

- Platform: Web based Eclipse
- Plug-ins for Collaboration
- Debugging and Visualization of Web Services
- Aspect Oriented Software Development
Web Eclipse

- Client-server Eclipse IDE
- Flexibility for dynamic, remote team development and testing.
Collaboration Plug-ins

- Contextual Filtering
- Attention Management
- Adaptive to the project team environment
Collaboration Plug-ins

- Contextual Filtering
- Attention Management
- Adaptive to the project team environment
Visualizing the Execution of Web Services
Profiling and Debugging
Complex Web Services

- What is happening?
- What is sent?
- How big are messages?
- Where are bottlenecks?
Problem Determination

Previously:
- Print statements
- Old-style debugging

Web Services Visualization:
- Understanding behavior
- Visual debugging
- Profiling with automated and selective tracing
- Plug-in for WebSphere Studio
Transferring Science into Products

Science:


Technology:

WebSphere Studio Application Developer: Tracing and Profiling Plug-ins
Eclipse Demonstrations

- Platform: Web based Eclipse
- Plug-ins for Collaboration
- Debugging and Visualization of Web Services
- Aspect Oriented Software Development
Aspect-Oriented Software Development

Mgr. earns more than employees

Sales
  - commission()

Research
  - salary()

Sales Manager
  - subordinates()
  - removeSub()

public float pay() {
  float result;
  // do pay computation
  result = ...;
  if (result >= manager.pay())
    // report error
  return result;
}

public void check() {
  if (this.pay() >= manager.pay())
    // report error
  }

public float pay() {
  float result = ...;
  return result;
}

public void check() {
  if (this.pay() >= manager.pay())
    // report error
}
Aspect-Oriented Software Development

Employee
- name()
- pay()

Mgr. earns more than employees

Sales
- commission()

Research
- salary()

Sales Manager
- subordinates()
- subordinates()

Cannot Remove Last employee

Research Manager
- subordinates()

Cannot Remove Last employee

Hyper/J View

PersonnelSystem
- Features
  - Personnel
    - Employee
      - name()
  - Sales
  - Payroll
    - Employee
      - pay()
- BusinessRules
  - RicherMgr
  - HasPeons

Mgr. earns more than employees
public void removeSub() {
    // Run HasPeons business rule
    ensurePeons();
    // If no exception was thrown,
    // remove the subordinate
    removeSub();
}
Papers


Contributing to the Success of Eclipse
Announcing Eclipse Innovation Grants

- Sponsored by IBM
- Grants for calendar year 2003.
- Grants to promote:
  - Adoption of the Eclipse Workbench in Academia
  - Extensibility & adoption of WebServices by academic research projects
  - Seamless integration of most innovative research technology
Eclipse Innovation Grants

- 20 awards up to $30,000 each for 2003
- Categories
  - Teaching, Community, and Research
- Examples:
  - Create a teaching-aid plug-in for Programming 101
  - Extend a compiler in Eclipse with latest research feature
- **Deadline: December 2nd, 2002 (!!!)**
- Results disseminated at Workshop sponsored by IBM in late 2003
- Encouraged to submit project results to [www.eclipse.org](http://www.eclipse.org)
- Announcement flyers at IBM booth
Conclusions

- **Distributed computing has been broadly understood for 25+ years**
  - But, the quest requires more technology than had been initially thought
  - Web Services is aimed at the breadth of the problem
  - Economic incentive sufficient now to push this over the top

- **Seamless integration of tools and runtime essential**
  - Open source seems a terrific way to go
  - Eclipse is allowing the community to create a plethora of tools:
    - IBM fully committed to success of Eclipse.

- **Our Eclipse Innovation Grants are an example of IBM’s commitment to Eclipse and Web Services success**

- **Implication for Research and Academe**
  - Our field is increasingly about the integration of things: in this case, tools and runtime
  - Build on Web Services and Eclipse, don’t reinvent the wheel
  - Rather, seize the opportunity to build on this technology & greatly advance the field