Volume 3, Issue 9, September 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Integration of Web Services and Routing Techniques for Distributed Applications

K. Tharun Kumar Reddy
M.Tech 2nd year
Dept. Of CSE
Annamacharya Institute of Tech., & Sci.,
(AITS), Tirupati, A.P., India

B. Ramana Reddy
Asst.Professor
Dept. Of CSE
Annamacharya Institute of Tech., & Sci.,
(AITS), Tirupati, A.P., India

Abstract -- This paper describes an approach to Enterprise Application Integration (EAI) using extensible Web services. The approach is demonstrated by building a real-world application for EAI in the financial services domain. Its main purpose is to integrate the various web services together and also assess the routing technique using load balancing and clustering mechanisms for various distributed applications. The manifestation of Web services in general and their role in EAI are discussed next. Financial services domain characteristics are presented. Business drivers that entail a strong need for functional extensibility in the financial services domain are described. Our proposed architecture for EAI which addresses functional extensibility is described. This architecture is based on the notion of extensible Web Services. We then present our implementation of the architecture and practical challenges encountered in EAI. A brief discussion of how our work relates to the current research in Service-Oriented Computing (SOC) and Semantic Web concludes the paper.

Keywords: enterprise application integration, Web Methods, Content Distribution, Content Distribution Network, Request routing, Surrogate Server, Origin Server, Domain Name Server, Internet.

I. INTRODUCTION

Typically large enterprises are supported by hundreds of applications. Many of these applications were written in COBOL on mainframe computers and are referred to as legacy systems. Enterprises critically depend on legacy systems for their day-to-day business operations. It is not unusual for large brokerage firms in the financial services sector to appropriate annual legacy maintenance budget in the order of billions of dollars.

Therefore, it is natural for the enterprises to explore ways to reduce legacy maintenance costs. Two primary approaches were pursued. The first approach involves replacing legacy systems with a new application. The latter is designed for better interoperability with other systems, and more importantly easier to maintain and evolve. Enterprise Resource Planning (ERP) systems were introduced to address this need. Though ERP systems were well received stored in different storage servers.

A. WEB SERVICES:

The term *Web services* describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. Used primarily as a means for businesses to communicate with each other and with clients, Web services allow organizations to communicate data without intimate knowledge of each other's IT systems behind the firewall.

Unlike traditional client/server models, such as a Web server/Web page system, Web services do not provide the user with a GUI. Web services instead share business logic, data and processes through a programmatic interface across a network. The applications interface, not the users. Developers can then add the Web service to a GUI (such as a Web page or an executable program) to offer specific functionality to users.

Web services allow different applications from different sources to communicate with each other without time-consuming custom coding, and because all communication is in XML, Web services are not tied to any one operating or programming language. For example, Java can talk with Perl, Windows applications can talk with UNIX applications.

Web services do not require the use of browsers or HTML.

Web services are sometimes called *application services*.

B. SERVICE-ORIENTED ARCHITECTURE

QUICKLY ASSEMBLE APPLICATIONS AND INCREASE ASSET RE-USE.

- Build new applications faster without jeopardizing quality
- Quickly automate cross-domain business processes

- Govern the life cycle of services and associated policies

Capture business capabilities as re-usable services. Rapidly assemble new applications and service-enable existing ones. Automate cross-domain processes and integrate systems. Do all of this with our Service-Oriented Architecture (SOA) and SOA governance technology. We also can guide you on how to plan and implement SOA programs.

Usedfor: multi-channel integration; enabling process automation; building new applications quickly

CAPABILITIES

SERVICE CREATION AND ENABLEMENT

- Create services including SOAP and REST-style services
- Implement new services or use our adapters to expose business logic and data from core systems
- Service-enable Oracle®, SAP® and other core apps, databases and legacy systems
- Drag-and-drop to transform services into higher-level business services

SERVICE LIFE-CYCLE GOVERNANCE

- Store and organize services, schemas and other SOA assets in a central, platform-independent SOA registry and repository
- Capture rich descriptions of your assets with owner, location, design and usage details
- Use policies to guide the design and development of good quality services
- Easy search capabilities allow you to quickly find services for re-use
- Use a graphical and interactive view of assets and their relationships
- Explore the entire asset landscape and find downstream impact of any proposed changes

SERVICE ACCESS MEDIATION

- Create virtual services from actual services shielding consuming applications from changes to underlying services
- Run multiple service versions side by side
- Improve compatibility between service consumers and service providers by switching protocols and transforming message formats “on the fly”
- Use policies to uniformly secure your services and remove the burden from consumers and providers for security
- Instead of specifying policies for each service, enforce them as a group on all services that meet specific criteria.

SERVICE MONITORING AND MANAGEMENT

- Gain real-time visibility into what’s happening with service transactions as they flow across your heterogeneous SOA infrastructure
- Capture service levels and metrics for any protocols or platforms
- Pinpoint root causes by using snapshot of the full transaction taken in the event of an error or Service Level Agreement (SLA) violation
- Discover services running in your environment and detect any rogue services or consumers

INCREASE AGILITY WHEN INTEGRATING APPLICATIONS

With web Methods, you gain a single integration backbone to connect all IT systems and application silos—from custom, mainframe and legacy apps to ERP, CRM and cloud-based systems. We can help you adopt a well-planned and effective approach to application integration and gain end-to-end application visibility across all systems.

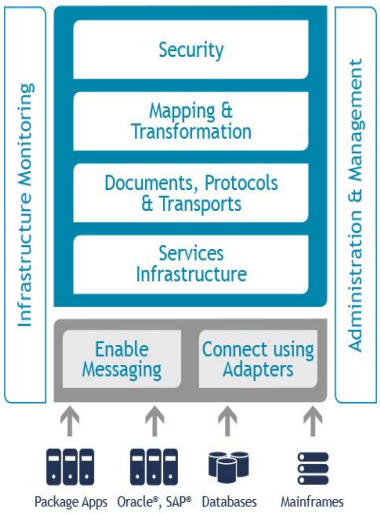


Diagram explaining the approach and monitoring of Application Integration

II. INTEGRATION SYSTEMS

A. **Application Integration**

Integrate any system across your Digital Enterprise.

- Reduce integration costs and development time
- Increase business agility
- Gain end-to-end visibility across systems

Say goodbye to the high cost and complexity of point-to-point integrations. Welcome a single integration solution that quickly connects any system or application without coding. Web Methods brings together a complete set of integration capabilities, including an analyst-recognized Enterprise Service Bus (ESB), messaging, adapters and monitoring.

Used for: enterprise application integration; integrating cloud & on-premise applications; integrating SAP® systems; integrating Oracle® systems; integrating mainframe systems

B. **CAPABILITIES**

Open, standards-based integration

- Our solution "speaks" any technology
- Integrate virtually any system, including ERP, CRM, cloud-based applications, home-grown applications and legacy systems, such as mainframes
- Secure data, encode data and communicate more easily across your Digital Enterprise

Standards-based integration is faster—web Methods supports key standards including HTTP, XML, SOAP and WDSL

C. **Reliable messaging**

- Guarantee delivery and once-and-only-once delivery of messages
- Supports all major messaging patterns, such as publish/subscribe,request/reply and synchronous/asynchronous

D. **Multiple document and protocol support**

- Supports industry-standard protocols, such as HTTP, HTTPS, FTP, FTPS, Java® Message Service (JMS) , SMTP, SNMP and file polling
- Supports document formats, including XML, flat files, .CSV files and delimited files

Enterprise Application Integration (EAI)

Enterprise application integration is an integration framework composed of a collection of technologies and services which form a middleware to enable integration of systems and applications across the enterprise Supply chain management applications (for managing inventory and shipping), customer relationship management applications (for managing current and potential customers), business intelligence applications (for finding patterns from existing data from operations), and other types of applications (for managing data such as human resources data, health care, internal communications, etc.) typically cannot communicate with one another in order to share data or business rules. For this reason, such applications are sometimes referred to as islands of automation or information silos. This lack of communication leads to inefficiencies, wherein identical data are stored in multiple locations, or straightforward processes are unable to be automated.

III. PROPOSED SYSTEM AND ARCHITECTURE

We are going to integrate web services from various applications across different platforms to design a web application called ACE and also defined routing techniques to provide high availability and scalability. Used for: Banking sectors and Domains.

ACE is a front end application where it integrates with various multiple projects to provide effective services to customers. It plays a prominent role as it deals with the clients and customers. The technical focus is on the ACE infrastructure and its connections from clients and to supporting services.

Bancorp Customer Service & Monetary Transactions Platform

- Inquiry – provides a single view of the customer including account balances, loan payoff quotes, contact history, customer relationships…
- Customer Maintenance – name & address change, account ownership…
- Service Requests – fee waiver, coupon book order, card disputes, payments errors, statement & check copy…
- Transactions – automated Teller, customer transfers & payments, internal funds transfer mechanism to move money between a customer's accounts as well as internal operating accounts (DDA & GL)

Users can access some Web services through a peer-to-peer arrangement rather than by going to a central server. Some services can communicate with other services and this exchange of procedures and data is generally enabled by a class of software known asmiddleware. Services previously possible only with the older standardized service known as Electronic Data Interchange (EDI) increasingly are likely to become Web services. Besides the standardization and wide availability to users and businesses of the Internet itself, Web services are also increasingly enabled by the use of the Extensible Markup Language (XML) as a means of standardizing data formats and exchanging data. XML is the foundation for the Web Services Description Language (WSDL).

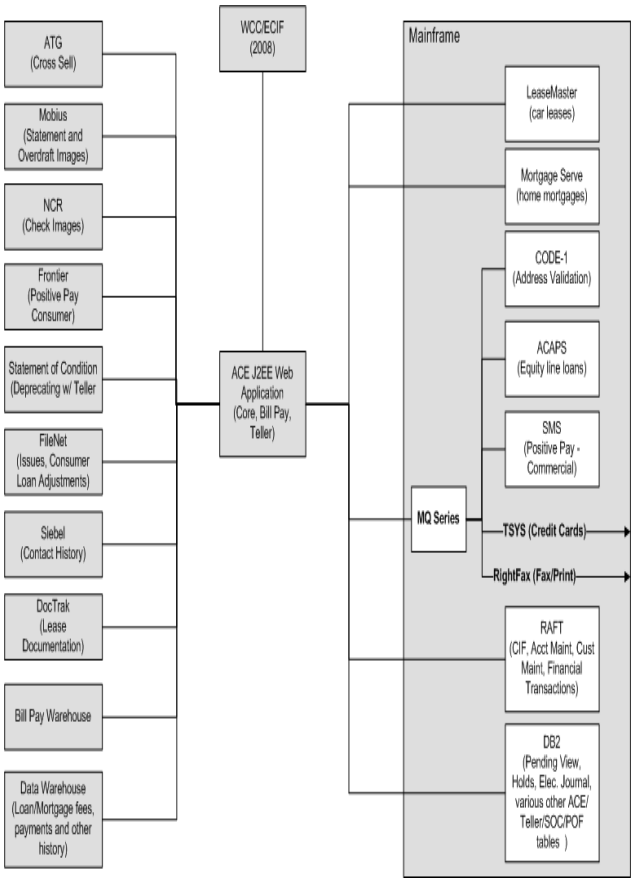


FIG: SYSTEM ARCHITECTURE

IV. COMPOSING A NEW SERVICE

A new service is composed from existing services in two ways, which are referred to as static and dynamic composite services. Users declaratively specify static composite service in the form of a script (in XML). Such a script essentially lists various services to be executed and specifies how the results from executing one service are mapped as inputs to other services. This is a powerful feature to efficiently realize functionally higher-level services tailored to a specific purpose. Dynamic composite service is similar to its static counterpart, except that the composition script is not known until the run-time.

In both types of composition, it is possible to simply execute a list of services as a unit and return multiple result sets in one XML document. Under this scenario, results of executing a service are not mapped as input parameters to other services in the unit. The resulting XML document includes information to correlate service requests with their corresponding execution results. This also enables specifying

input parameters that are common to multiple services

only once.

Web services roles

This section will describe the process of invoking web services. A web service provider describes a web service in a Web Services Description Language (WSDL) document. The web service is typically published to a Universal Description, Discovery and Integration (UDDI) registry. A web service requester finds the web service in the UDDI registry, binds to the web service, and invokes it. The web services roles are shown in Figure 1. This article will focus on the horizontal arrow (bind) from the service requester to the service provider. This article will call the requester a *client*; it can also be referred to as a *consumer*.

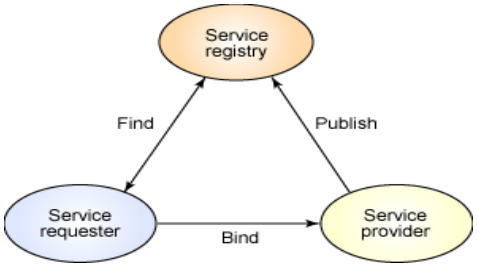


Figure 1. Web services roles

Web services standards for Java

The development of Java technology standards occurs through Java Specification Requests (JSRs) being submitted to the Java Community Process (JCP). Two JSRs cover the Java web services architecture:

1. JSR 101: Java API for XML based RPC (JAX-RPC)
2. JSR 109: Implementing Enterprise Web services.

Both specifications provide conformance and interoperability requirements for vendors' implementations.

JAX-RPC

JAX-RPC defines a simple and easy-to-use Java Application Programming Interface (API) for XML-based Remote Procedure Calls (RPC) and the Java to XML and XML to Java mapping:

- WSDL to Java and Java to WSDL mappings: For example, a WSDL port type is mapped to a Java Service Endpoint Interface (SEI).
- XML data type to Java data type and Java data type to XML data type mappings, including simple types, complex types, and arrays.

In addition to XML mappings, JAX-RPC also defines the client-side programming model and API, which I will cover in more details in later sections. JAX-RPC 1.1 adds interoperability requirements based on the Web Services Interoperability organization (WS-I) Basic Profile version 1.0.

JSR 109

JSR 109 specifies the web services programming model and architecture for the Java 2 Enterprise Edition (J2EE) environment. JSR 109 builds on SOAP 1.1 and WSDL 1.1 to cover the use of JAX-RPC in a J2EE environment (Figure 2). It also defines a deployment model to J2EE application servers. JSR 109's client-side programming model, which I will cover in sections below, is conformant to JAX-RPC.

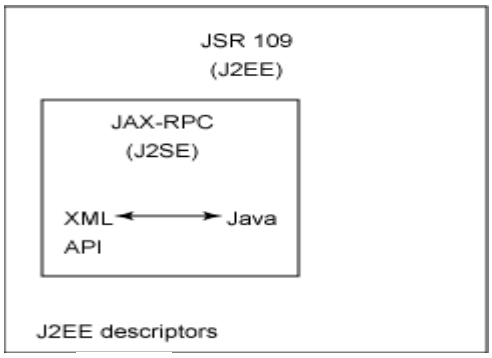


Figure2.JSR109andJAX-RPC

V. WORK LOAD MANAGEMENT AND CLUSTERING FOR REQUEST(OR) SERVICE ROUTING

Workload management is the concept of sharing requests across multiple instances of a resource. Workload management techniques are implemented expressly for providing scalability and availability within a system. These techniques allow the system to serve more concurrent requests. Workload management allows for better use of resources by distributing load more evenly. Components that are overworked, and therefore, perhaps a potential bottleneck, can be routed around with workload management algorithms. Workload management techniques also provide higher resiliency by routing requests around failed components to duplicate copies of that resource. In WebSphere Application Server, workload management is achieved by sharing requests across one or more application servers, each running a copy of the Web application. In more complex topologies, workload management is embedded in load balancing technologies that can be used in front of Web servers.

CLUSTERING

Clustering application servers that host Web containers automatically enables plug-in workload management for the application servers and the servlets they host. Routing of servlet requests occurs between the Web server plug-in and the clustered application servers using HTTP or HTTPS. This routing is based on weights associated with the cluster members. If all cluster members have identical weights, the plug-in sends equal requests to all members of the cluster, assuming no strong affinity configurations. If the weights are scaled in the range from 0 to 20, the plug-in routes requests to those cluster members with the higher weight value more often. No requests are sent to cluster members with a weight of 0 unless no other servers are available. Weights can be changed dynamically during runtime by the administrator.

A guideline formula for determining routing preference is:

% routed to Server1 = weight1 / (weight1+weight2+...+weightn)

Where there are n cluster members in the cluster. The Web server plug-in temporarily routes around unavailable cluster members

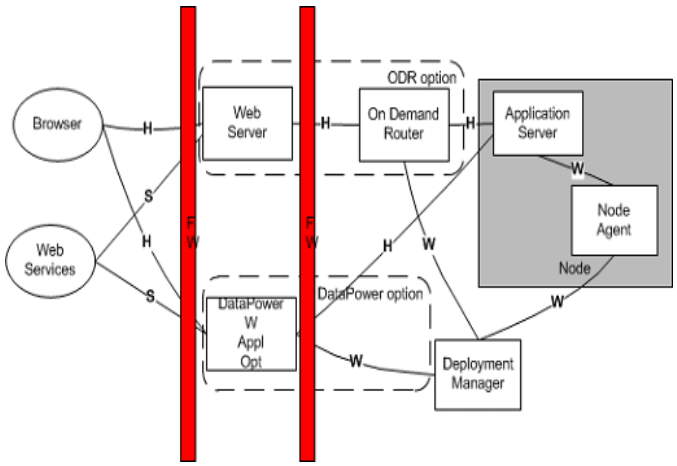
TECHNOLOGY AND APPLICATION ADAPTERS

- Rapidly connect packaged apps and databases using adapters instead of coding

- Adapters feature connection pooling, security, session management and logging
- Choose from a broad range of adapters for JDBC®, SAP®, Oracle®, Siebel CRM™, Salesforce.com® and more
- Service creation and consumption
- Create and invoke XML, SOAP, REST, JSON, Java®, Microsoft® .NET and other types of services

MAPPING AND TRANSFORMATION

- Use an Eclipse™-based integrated development environment to visually map any document type to any other
- Hundreds of transformers are available out-of-the-box to support standard and complex transformations within the maps.



INFRASTRUCTURE MONITORING

- Add web Methods Optimize for Infrastructure to centrally monitor the health of the entire integration infrastructure
- Measure and analyze metrics, such as available memory, CPU usage and number of threads used
- Predictive analysis detects potential system failures even before they occur.

DEPLOYMENT

- Web-based tools like web sphere application server can help in simulated deployments, checkpoints, rollbacks and federated deployments across environments
- Can be visually configured or scripted

VI. CONCLUSION

Web services (sometimes called application services) are services (usually including some combination of programming and data, but possibly including human resources as well) that are made available from a business's Web server for Web users or other Web-connected programs. Providers of Web services are generally known as application service providers. Web services range from such major services as storage management and customer relationship management (CRM) down to much more limited services such as the furnishing of a stock quote and the checking of bids for an auction item. The accelerating creation and availability of these services is a major Web trend. As Web services proliferate, concerns include the overall demands on network bandwidth and, for any particular service, the effect on performance as demands for that service rise. A number of new products have emerged that enable software developers to create or modify existing applications that can be "published" (made known and potentially accessible) as Web services.

FUTURE WORK

- Integration of web services with various Tools to use them as a pre-defined services.
- Future content based routing algorithms will be affected b y the desire of applications to increase the amount of expressiveness when specifying subscriptions. As a result, composite event detection will become an important way to support a more fine-grained expression of subscriber interests in many applications and will become an integral part of the basic architecture of a large-scale notification service.
- From a methodological point of view, the design of routing algorithms should be guided by real-world workloads that publish/subscribe applications experience with larger deployment of such application, sets of work-load traces should become available that researchers and engineers can use to assess the efficiency and performance of their routing algorithms and implementations.

REFERENCES

[1] Andrzej Duda, and Mark A. Sheldon, "Content Routing in a Network of WAIS Servers", Broadcast Technical Report, Esprit Basic Research Project 6360, Laboratory for Computer Science, MIT, Cambridge, MA 02139, USA.

[2] S. Arroyo, R. Lara, J. M. Gomez, D. Berka, Y. Ding, and D. Fensel. Semantic aspects of web services. In M. P. Singh, editor, The Practical Handbook of Internet Computing, pages 31–1 – 31–17. Chapman & Hall/CRC, 2005.

[3] D. Batory, C. Johnson, B. MacDonald, and D. von Heeder. Achieving extensibility through product-lines and domainspecific languages: A case study. ACM Transactions on Software Engineering and Methodology, 11(2):191–214, April 2002.
[4] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. Scientific American May 2001.
[5] F. Casati and A. Sahai. Business process: Concepts, systems, and protocols. In M. P. Singh, editor, The Practical Handbook of Internet Computing, pages 32–1 – 32–15. Chapman & Hall/CRC, 2005., 1996
[6] Kurose, James E. and Ross, Keith W. (2004). Computer Networking, Third Ed. Benjamin/Cummings. ISBN 0-321-22735-2.
[7] Medhi, Deepankar and Ramasamy, Karthikeyan (2007). Network Routing: Algorithms, Protocols, and Architectures. Morgan Kaufmann. ISBN 0-12-088588-3.
[8] Jonne Zutt, Arjan J.C. van Gemund, Mathijs M. de Weerdt, and Cees Witteveen (2010). Dealing with Uncertainty in Operational Transport Planning. In R.R. Negenborn and Z. Lukszo and H. Hellendoorn (Eds.) Intelligent Infrastructures, Ch. 14, pp. 355–382. Springer.
[9] Z. Jiang, L. Kleinrock, "An adaptive network pre-fetch scheme," in IEEE Journal on Selected Areas in Communications, vol. 16, no. 3, pp. 358-368, Apr. 1998.
[10] D.B. Terry and V. Ramasubramanian, "Caching XML Web Services for Mobility," Queue vol. 1, no. 3, pp. 70-78, May 2003.
[11] A. Myers, P. Dinda, and H. Zhang, "Performance characteristics of mirror servers on the internet," in Proc. IEEE INFOCOM, 1999, pp. 304–312.
[12] Dataplot Reference Manual. NIST Handbook Number 148, National Institute of Standards and Technology, Gaithersburg, MD, 2001.
[13] B. Noble, M. Satyanarayanan, D. Narayanan, J. Tilton, J. Flinn, and K. Walker, "Agile application-aware adaptation for mobility," in Proc. ACM Symp. Operating Systems Principles, Oct. 1997, pp. 276–287.
[14] V. Padmanabhan and J. Mogul, "Using predictive prefetching to improve World Wide Web latency," in Proc. ACM SIGCOMM, July 1996, pp. 22–36.
[15] V. Paxson, "Measurements and analysis of end-to-end internet dynamics," Ph.D. dissertation, University of California, Berkeley, 1997.
[16] J. Pitkow and P. Pirolli, "Mining longest repeating subsequences to predict World Wide Web surfing," in Proc. USENIX Symp. Internet Technologies and Systems, Oct. 1999, pp. 139–150.
[17] M. Satyanarayanan, J. Kistler, P. Kumar, M. Okasaki, E. Siegel, and D. Steere, "Coda: A highly available file system for a distributed workstation environment," IEEE Trans. Comput., vol. 39, pp. 447–459, Apr. 1990.
[18] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of internet path selection," inProc. ACM SIGCOMM, Sept. 1999, pp. 289–299.
[19] Wei Zhang and Robert van Engelen, A Table-Driven XML Streaming Methodology for High-Performance Web Services, in the proceedings of IEEE International Conference on Web Services (ICWS), 2006, pages 197-206, (best student paper award).
[20] Robert van Engelen, Madhusudhan Govindaraju, and Wei Zhang, Exploring Remote Object Coherence in XML Web Services, in proceedings of IEEE International Conference on Web Services (ICWS), 2006, pages 249-256.
[21] M. Head, M. Govindaraju , R. van Engelen, and W. Zhang, Benchmarking XML Processors for Applications *in* Grid Web Services, in the proceedings of Supercomputing 2006.
[22] M. Cafaro, D. Lezzi, S. Fiore, G. Aloisio, and R. van Engelen, The GSI plug-in for gSOAP: building cross-grid interoperable secure grid services, in the proceedings of the International Conference on Parallel Processing and Applied Mathematics (PPAM) 2007, workshop on Models, Algorithms and Methodologies for Grid-enabled Computing Environment (MAMGCE), Springer Verlag LNCS Volume 4967, pages 894-901, 2008.
[23] Giovanni Aloisio, Massimo Cafaro, Italo Epicoco, Daniele Lezzi, and Robert van Engelen, The GSI plug-in for gSOAP: Enhanced Security, Performance, and Reliability, in the ITCC conference 2005, IEEE Press, Volume I, pages 304-309.
[24] J. Hennessy and D. Patterson, Computer Architecture: A Quantitative Approach, 2nd ed. San Mateo, CA: Morgan Kaufmann, 1996.
[25] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West, "Scale and performance in a distributed file system,"ACM Trans. Comput. Syst., vol. 6, no. 1, pp. 51–81, Feb. 1988.
[26] R. Jain, The Art of Computer Systems Performance Analysis. New York: Wiley, 1991, ch. 13, pp. 179–200.
[27] A. Joseph, A. de Lespinasse, J. Tauber, D. Gifford, and M. Kaashoek, "Rover: A toolkit for mobile information access," in Proc. ACM Symp. Operating Systems Principles, Dec. 1995, pp. 156–171.
[28] J. Kistler and M. Satyanarayanan, "Disconnected operation in the coda file system," ACM Trans. Comput. Syst., vol. 10, no. 1, pp. 3–25, Feb. 1992.
[29] Michael R. Head, Madhusudhan Govindaraju, Aleksander Slominski, Pu Liu, Nayef Abu-Ghazaleh, Robert van Engelen, Kenneth Chiu, Michael J. Lewis,Benchmark Suite for SOAP-based Communication in Grid Web Services , in the proceedings of ACM/IEEE Supercomputing (SC), 2005.
[30] Robert van Engelen, Wei Zhang, and Madhusudhan, Toward Remote Object Coherence with Compiled Object Serialization for Distributed Computing with XML Web Services, in the proceedings of Compilers for Parallel Computing (CPC), 2006, pages 441-455.

