

# AONBench: A Methodology for Benchmarking XML Based Service Oriented Applications

Abdul Waheed

Cisco Systems, Inc., 170 W. Tasman Drive, San Jose, CA 94035, USA  
Email: abwaheed@cisco.com

Jianxun Jason Ding

Cisco Systems, Inc., 170 W. Tasman Drive, San Jose, CA 94035, USA  
Email: jiding@cisco.com

**Abstract**—Service Oriented Architectures (SOA) and applications increasingly rely on network infrastructure instead of back-end servers. Cisco Systems Application Oriented Networking (AON) initiative exemplifies this trend. Benchmarking such infrastructure and their services is expected to play an important role in the networking industry. We present AONBench specifications and methodology to benchmark networked XML application servers and appliances. AONBench is not a benchmarking tool. It is a specification and methodology for performance measurements, which leverages from existing XML microbenchmarks and uses HTTP for end-to-end communication. We implement AONBench specifications for end-to-end performance measurements through public domain HTTP load generation tool ApacheBench and Apache web server. We present three case studies of using AONBench for architecting real application oriented networking products.

**Index Terms**—SOA, XML, Application Oriented Networking, Benchmarking.

## I. INTRODUCTION

Traditional network infrastructure is based on providing simple packet switching, forwarding, queuing, and routing services, while complex message-oriented tasks are left to the end hosts. This task partitioning was logical when network infrastructure had limited processing capabilities. With continuing exponential growth in processor performance, this partitioning is becoming increasingly irrelevant. Cisco Systems recently launched its Application Oriented Networks (AON) products that integrate message processing capabilities within the network infrastructure. AON modules can be added to specific switches and routers to provide application-oriented services in addition to default packet switching and forwarding functions.

A service oriented architectures with AON can offload an application server endpoint by providing network infrastructure, security, and application-specific integration services. Currently, many application servers have to provide network infrastructure services in addition to supporting their target applications. These

services include protocol processing, load balancing, security and firewalls, content caching and delivery, and integrating heterogeneous standards. Cisco AON products have pushed these services to network infrastructure and are available as switch and router embedded modules as well as appliance incarnations. We shall refer to any one of these three incarnations as an Application Oriented Network Infrastructure and Services Enabled (AONISED) node for the sake of consistency.

XML is becoming ubiquitous for web services applications. AONISED nodes use XML for message processing in the same manner that IP is used by routers for packet processing. However, in contrast to packet, message processing offers different types of challenges. Unlike IP packets, application messages are typically stateful and belong to specific flows. An AONISED node has to deal with message content instead of merely processing the protocol headers in individual packets. For instance, parsing an XML message for pattern matching is more expensive than packet header manipulation. In addition, security operations on XML message content are inherently compute-intensive. All of these challenges point to the criticality of AONISED node performance. Networked services providers and businesses using their services need a consistent performance measurement methodology to benchmark a growing number of AONISED products.

There are multiple benchmarks available for web services such as caching, proxying, content-delivery, and storage networking using HTTP and HTTPS. Similarly, multiple XML microbenchmarks are also available that target specific functions such as parsing, Document Oriented Models (DOM) construction, pattern matching, transformation, validation, and security. A benchmark for an AONISED node needs to merge the functions of a web services benchmark and an XML functions microbenchmark. However, due to AONISED node specific nature of XML interfaces, such a merger will not extend to any other product having a different interface. There are three requirements that an AONISED node benchmark is expected to meet:

- 1) Benchmark should be vendor and product neutral to cover AONISED nodes in all different incarnations;
- 2) It should be open in terms of both its specifications as well as its code to make it accessible to anyone interested in using it; and
- 3) It should exercise both web services as well as XML content processing. Web services refer to basic services of content delivery and caching. XML content processing includes common functions of parsing, pattern matching, schema checking, transformation, and secure communication.

We have developed the specifications of AONBench to meet these requirements. We have been using our in-house tools to implement these specifications for the last two years. However, to conform to the spirit of above requirements, we have implemented AONBench using ApacheBench (AB) as client and Apache Web Server (AWS) as server endpoint [1]. These tools are open-source and neutral to AONINED systems from different vendors.

In the context of specifying, we have identified eight basic use cases for web services and XML content processing. We identify forwarding HTTP requests (FR) as the basic web service. FR is also referred to as HTTP proxying. HTTP caching and load balancing may be included among other basic web services. Although Cisco AON products support these function, we are excluding them from our specifications to remain product and vendor neutral as many other products may not support them. In addition, we have identified seven basic XML content processing use cases for AONBench specifications: content based routing (CBR), schema validation (SV), transformation (TR), encryption (EN), decryption (DE), signing (SI), and signature verification (VE). We shall specify AONBench using these eight cases in terms of an input XML message that arrives at an AONISED node and the format of the processed message that emerges from that node. Our objective is to provide a level playing field to compare the performance of service oriented network infrastructure products from different vendors. An AONBench implementation exercises seven XML functions and proxy performance in an end-to-end communication setting.

We have successfully used our benchmark specifications to internally evaluate Cisco Systems' AON products at different phases of their development. In addition, we have used these specifications with our customers and partners to allow them to independently evaluate and verify AON performance. With expanding role of service oriented infrastructure in enterprise networking, vendor- and product-neutral benchmarking is expected to become increasingly important.

In Section II, we present background of Cisco AON in terms of its features and use cases. In Section III, we present an overview of the state-of-the-art in benchmarking XML services to motivate the need for end-to-end benchmarking of service oriented network infrastructure. We describe our benchmarking methodologies in Section IV followed by detailed specifications of individual benchmark cases in Section V

and present an AWS/AB based implementation of these specification in Section VI. Using a few case studies in Section VII, we demonstrate the efficacy of this benchmarking approach for networked XML applications. We conclude in Section VIII with a discussion of the scope and future directions of this benchmarking initiative.

## II. APPLICATION ORIENTED NETWORKING

Application Oriented Networking (AON) is a relatively new paradigm. In this section, we describe the background and scope of AON and its benefits to enterprise applications. Additionally, we discuss the usage scenarios of AON.

### A. Scope of AON for the Enterprise

For most organizations, the costs of building and maintaining an application integration infrastructure is huge. As many IT organizations expand, the ability to evolve an application or business process is often hampered by the inability to quickly adapt the application infrastructure. With AON, these challenges are significantly reduced.

A majority of enterprise applications and software infrastructure is distributed and heterogeneous. The cost and complexity of maintaining and integrating these infrastructures and middleware pressures is significantly high. Many businesses consider architectural consolidation to deal with this complexity. However, consolidation alone can cost many years and still not lead to a substantive increase in IT or business flexibility. With AON, the network can play the role of the standard intermediary. Beyond reducing the server and software component, Cisco AON can normalize messages and content from different application bases and provide centralized administration and management to facilitate rapid implementation and modification.

Widely distributed enterprise is forcing IT to discover new ways to incorporate remote operations while reducing costs and maintaining control. As growing volume of business and business process happens remotely, often in places ill-suited to servers and complex application middleware, the option is generally a complex and custom-built integration project. Cisco AON overcomes this challenge by providing a low-cost and low-maintenance platform that can be fully managed remotely.

Organizations increasingly recognize the value in adopting a service-oriented infrastructure in providing reusability and a lower cost of integration. However, most organizations are heavily dependent on their legacy software and infrastructure. IT typically likes to move forward incrementally, without the discontinuity implied by a new architectural approach. Cisco AON leverages the existing network infrastructure and enables this incremental platform. Because AON understands and can control the vast majority of application middleware protocols, IT gains a common control mechanism across the extended enterprise.

### B. AON Features and Benefits

AON is implemented as a set of network devices that understand, process, and control application protocols and messages that fully complement traditional network-layer services. These application protocols can be traditional middleware (MQ Series, Tibco, BEA) or more general purpose (JMS, MMQ, SOAP). AON devices are located in multiple points in the network, including switches, routers, and appliances and can process application messages either in-band (control) or out-of-band (reporting).

AON incorporates a feature-rich policy management engine to enable a variety of functions. These functions include: message content analysis, protocol transformation, message based routing, and message security. AON includes comprehensive toolkits to develop and manage policy based operations. The AON Development Studio (ADS) is used to create Policy Execution Plans (PEP). PEP includes a set of operations specific to a business policy to be applied to application messages.

### C. AON Use Cases

The Cisco AON platform enables a variety of different usage scenarios across a customer application network. Typical deployment scenarios are as follows:

1. AON is deployed at a remote office as an infrastructure consolidation device. A single AON device can provide all the services required by the branch office to effectively communicate with the central office. Cisco AON enables these services by bridging disparate applications and optimizing network usage at the application level.
2. At the enterprise edge, Cisco AON can act as an application-security gateway. In this context, it can intercept and analyze traffic in message formats such as Extensible Markup Language (XML).
3. In the enterprise core, Cisco AON provides transparent inter-application communication and application message delivery; it provides a network-embedded communication bridge between protocols and applications. Cisco AON optimizes application delivery by offloading application infrastructure functions such as message-level load balancing to the network where they can scale effectively and by enabling intelligent application message routing at the network level.

Remote office, enterprise edge, and core based description of AON usage represents broad categories of its target domains. There is a large number of solutions within each of these domains, which are based on Cisco AON using its flexibility to customize.

## III. STATE OF BENCHMARKING XML AND WEB SERVICES

There are many benchmarks for XML as well as web services. We subdivide the XML benchmarks into microbenchmarks, management benchmarks, and security

benchmarks. We discuss these benchmarking efforts in the following subsections.

### A. XML Microbenchmarks

These benchmarks target the basic XML processing functions, such as XML parsing, generating document oriented models (DOM), using XPath to match patterns, etc. XML documents can be parsed using SAX or DOM frameworks depending on the need of serial vs. random access to the content, respectively. The XML Parsing Benchmark claims to evaluate SAX, DOM, or a mix of these two types of frameworks [13]. Such microbenchmarks are suitable to compare different types of XML parsers, which constitute an important component of an XML processing system. Kuznetsov and Dolph describe XSLTMark, which focuses on XML-XML and XML-HTML transformation test cases and its performance scoring mechanism [7]. This benchmark assesses an XML processor in four areas: XSLT template pattern matching, XSLT control structure, XPath selections, and XPath functions. A simple Perl XML benchmark that targets XML parsing, extraction, replacement, and processing is described in [11]. All of these microbenchmarks can be used to compare the performance of various hardware and software systems for XML processing.

### B. XML Management Benchmarks

Many benchmarks specifically measure XML storage and database management system performance. Databases are part of a typical backend for an AONISED infrastructure. Bressan et al. describe the X007 Benchmark for this type of comparisons [4]. Runapongsa et al. describe using The Michigan Benchmark, which is a microbenchmark for XML query processing, to compare the performance of various database systems [9]. Schmidt et al. also describe a similar benchmarking project, called XMark [10]. Böhme et al. describe XMach-1 to evaluate native and relational XML databases [2]. These benchmarks do not target any complete applications and narrowly focus on XML database performance.

### C. XML Security Benchmarks

Several security function implementations are available in public domain as java packages or libraries. For instance, XMLSec implements encoding, decoding, signature, and verification functions using libxml [14]. Such implementations usually provide test programs (unit tests) to evaluate the performance of these functions. XMLSec provides unit tests for all four XML security library functions. Obviously, these tests have a very narrow scope and cannot be used beyond XMLSec library evaluation. AONBench specifications also target these four XML security functions as they are becoming increasingly important in service oriented network infrastructure.

Clearly none of the existing XML benchmark can accommodate the performance measurement scope and needs for application oriented network infrastructure and services enabled systems. AONBench is an effort to address these needs.

#### D. Benchmarking Web Services

There are many existing HTTP benchmarks that target different types of web services: proxying, caching, SSL termination, streaming, and dynamic web pages. Currently benchmarking for web services is not targeted to XML based application services at network edges. Web services based benchmarks can be adapted to XML processing network infrastructure with following three characteristics: (1) XML server should accept a message in XML and should forward same or different message in XML format; (2) XML processing functions are offered as modular services through appropriate configuration; and (3) XML processing functions are externally accessible through HTTP or other open protocols. When these characteristics are available in an XML application server, a web services benchmark can be used. In this case, benchmark should be capable of uploading the messages (HTTP POST) in addition to regular downloads (HTTP GET).

#### IV. BENCHMARKING METHODOLOGY

In this section, we target both web services as well as XML functions provided by the network infrastructure. We use HTTP POST method to carry XML messages from client to an AONISED node, which can forward it to a server endpoint depending on the use case. It is immaterial whether the message arrives at the AONISED node through explicit forwarding or implicit forwarding using, for instance, Web Cache Control Protocol (WCCP). These HTTP messages use standard HTTP protocol and headers. An AONISED node processes the message according to the use case.

We use unique URIs in the POST method to select a specific service from the AONISED node. AONBench specification defines eight use cases. Therefore, AONISED node should support eight URIs to uniquely select a service. Load generation client can employ these URIs to exercise each service without using any product-specific interfaces.

Rest of this section presents a test harness required for measurements with AONBench. This will be followed by a description of content size and performance metrics for benchmarking.

##### A. Test Harness

AONBench test harness consists of at least one load generating client and two server endpoints. We require two server endpoints to distinguish a successfully executed use case from an unsuccessful execution. AONBench specification (see Section V) depends on incoming and outgoing XML message contents. Therefore, success and failure of a use case can be decided by changing the incoming content and observing the server endpoint and contents of XML message arriving at those endpoints. In addition to client and server endpoints, test harness will consist of at least one AONISED node and a switch to connect client, server, and AONISED systems.

AONBench test harness described above is an overly simplified configuration of an AONISED system. In real

deployments, multiple AONISED nodes are used with ingress and egress on different nodes. However, for AONBench use case specification, this simple setup suffices.

##### B. Content Size

We need to choose a message size that represents a majority of XML message traffic between clients and application servers. Such messages are typically of small sizes. Based on our interactions with enterprise applications that interact through XML messages, we consider a 5 Kbytes XML message as one that is most often encountered. Although any message size is permissible, we frequently use 5 Kbytes messages with AONBench to measure and report the performance of our AON products.

##### C. Performance Metrics

AONBench uses two metrics to measure the performance of an AONISED node: throughput and latency. Using the test harness, these measurements are obtained at the client end. End-to-end latency of a transaction is an approximate measure of the transaction processing delay of the AONISED node in an isolated network with no bottlenecks.

Using the test harness, content sizes, and metrics presented in this section, we are now ready to specify the AONBench use cases in terms of XML content.

#### V. SPECIFICATIONS FOR AONBENCH USE CASES

We specify eight AONBench use cases with respect to ingress and egress XML message contents without any dependence on client and server endpoints beyond generating and consuming HTTP messages. These use cases are grouped as network infrastructure services and XML message content based services.

##### A. Network Infrastructure Services

Proxying for HTTP requests is a simple infrastructure level web service. This case represents the overhead of forwarding a message to a default destination. Unless there are configuration or execution errors in the AONISED system under test, this case should result in highest throughput compared to all other use cases because it does not involve any content processing. Thus, this use case is valuable to validate the baseline for the test harness as well as the system under test. FR uses the same XML message that will be used for CBR specifications in Section V.

Caching and load balancing are additional FR related use cases that we regularly employ for Cisco AON products that are integrated in routers and switches. However, we do not include these variants in this specification as many other AONISED solutions do not support network infrastructure related services.

##### B. XML Message Content Based Services

We now consider seven use cases that require the processing of XML message content. In each case, the content processing function can be verified either by the

XML message contents at egress or the destination server endpoint that consumes the message.

1) *Content Based Routing (CBR)*. CBR performs three operations on the incoming message content: (1) content lookup; (2) content routing based on a pattern match or lack thereof; and (3) forwarding of message to selected destination endpoint. Content lookup is performed using Xpath evaluation. In this case, we use an XML message with SOAP envelope containing a <quantity> element. Using Xpath expression //quantity/text(), we determine whether this tag is found and is equal to string "1". If this condition is evaluated to true, the message is forwarded to the intended endpoint; otherwise, it is forwarded to a default error handling endpoint. The XML message that we use for CBR uses filler text elements to increase the overall message size to conform to the specified 1, 5, or 500 Kbytes sizes. Figure 1 illustrates the CBR use case. This specification uses only one Xpath. Number of Xpath expressions can be increased with content selection such that a subset of them finds matches while the rest are unmatched to vary computational complexity.

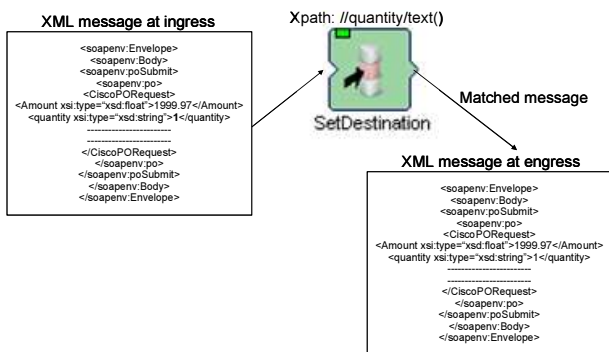


Figure 1. Content Based Routing use case.

2) *Schema Validation (SV)*. SV use case compares the incoming XML message to a pre-stored schema to determine whether it conforms to specified grammar. If the message turns out to be valid according to the schema, it is sent to one destination; otherwise, it is sent to the error endpoint. A modified input message can verify whether the AONISED node is executing this use case correctly. The objective of this case is to measure the performance during schema validation process. This is a compute-intensive task and its complexity will depend on the incoming XML message and the schema. The name of the XSD schema file is referenced in our reference XML message. An AONISED node uses this information to validate the message. Figure 2 illustrates this use case.

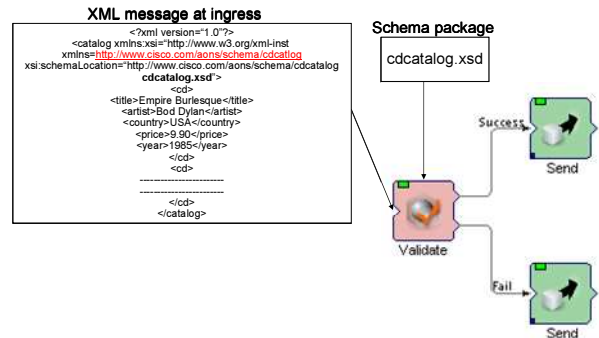


Figure 2. Schema Validation use case.

3) *Transformation (TR)*. TR use case consists of applying an XSL style sheet to transform an incoming XML message to a desired format (see Figure 3). The resulting format could be any one of CSV, XML, HTML, plain text, etc. Complexity of this operation depends on the incoming XML message as well as the style sheet. Similar to SV case, style sheet is saved off-line on an AONISED node.

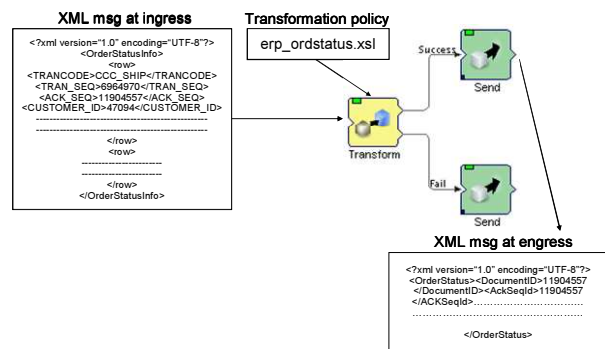


Figure 3. Transformation use case.

4) *Encryption (EN) and Decryption (DE)*. EN and DE use cases perform encryption on an XML/SOAP message and its decryption, respectively. Message resulting from the EN is used as incoming XML/SOAP message for the DE case. We use an XML message with SOAP envelope for encryption as well as signature use cases. It represents a purchase order with some user specific XML content. The EN use case encrypts the body of the message and forms another XML/SOAP message with header containing the information about SSL key, encryption algorithm, and body of message that includes encrypted original message content. The performance for this use case depends on key length, encryption algorithm, and size of the message. EN is illustrated in Figure 4.



TABLE I. COMPARISON OF SPEEDUPS FOR TR AND SV USE CASES ON AONBENCH FOR ALTERNATIVE HARDWARE PLATFORMS UNDER EVALUATION.

Hardware Platform	Speedup	
	TR	SV
2x3.2 GHz CPUs, 1 MB L2, 2 MB L3, 667 MHz FSB, and 4 GB DRAM	Baseline-1	Baseline-2
(4x3.2 GHz CPUs, 1 MB L2, no L3, 667 MHz FSB, and 4 GB DRAM	1.25x	1.1x
2x3.2 GHz CPUs, 1 MB L2, 8 MB L3, 667 MHz FSB, and 4 GB DRAM	1.6x	1.8x

### C. Evaluating Accelerator Hardware

Hardware accelerators are frequently used to enhance the performance of targeted applications on a general-purpose host platform. We regularly use AONBench to evaluate the performance impact of various types of hardware accelerators. Table II shows that a particular hardware accelerator-X improves the throughput of selected AONBench use cases with 5 Kbytes messages by an average of about 10% on a selected host platform. However, more significant throughput enhancement for CBR and EN use cases is obtained with 500 Kbytes message sizes. These performance measurements show the value of using acceleration hardware.

TABLE II. PERFORMANCE IMPACT OF USING A HARDWARE ACCELERATOR FOR SELECTED AONBENCH USE CASES.

Use Case	Performance Speedup due to Hardware Accelerator-X	
	5 KB message	500 KB message
CBR	1.08	3.08
SV	1.12	1.11
TR	1.00	1.00
EN	1.06	1.44
DE	1.39	1.18
SI	1.11	1.02
VE	1.03	1.02

In this section, we demonstrated the value of AONBench for comparing architecture and software design alternatives. As AONBench simply specifies the test cases and does not depend on the choice of client and

server endpoints, it has been used by multiple groups inside Cisco AON team as well as by our partners.

### VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the specifications of AONBench as well as some case studies where we effectively used it. Moving various common application processing capabilities to network infrastructure is becoming increasingly important for a variety of businesses. As new products and solutions are being introduced by multiple vendors, the need for a vendor and product independent benchmarking mechanism is there. AONBench is an effort to fill this gap as well as to motivate others to develop intelligent and efficient network infrastructure products. Additionally, end users can implement these specifications in terms of using our specified XML message sizes and contents, schemas, and style sheets to benchmark the basic message processing functions of these products.

In addition to eight basic use cases that we specified in this paper, we are using several other application-specific use cases. These include reliable messaging, message brokering, caching, server virtualization, message distribution, logging, and data base access. However, some of these cases depend on customized client and server endpoints. As suitable public domain tools become available for exercising such functions, AONBench specifications can be enhanced to include these cases.

Another application of AONBench is to conduct architectural evaluation [3,6]. Such evaluations guide the selection of efficient processors for building systems that host XML based service oriented applications.

### ACKNOWLEDGMENT

The authors acknowledge and wish to thank the current and former AON team members for their direct or indirect contributions to the AONBench.

### REFERENCES

- [1] Apache HTTP Server Project. On-line from: <http://httpd.apache.org/>
- [2] T. Böhme and E. Rahm, "Multi-User Evaluation of XML Data Management Systems with XMach-1," *Lecture Notes in Computer Science (LNCS) 2590*, pp. 148-159, Springer, 2003.
- [3] Dileen Bhandarkar and Jason Ding, "Performance Characterization of the Pentium Pro Processor, in *Proceedings of the 3rd IEEE Symposium on High-Performance Computer Architecture (HPCA-3)*, pp. 288-297, 1997.
- [4] S. Bressan, M. Lee, Y. Li, Z. Lacroix and U. Nambiar, "The XOO7 Benchmark," in *proceedings of the first VLDB Workshop on Efficiency and Effectiveness of XML Tools, and Techniques (EEXTT)*, Hong Kong, China, August 2002.
- [5] Cisco Application Oriented Networking. On-line from: [http://www.cisco.com/en/US/products/ps6692/Products\\_Sub\\_Category\\_Home.html](http://www.cisco.com/en/US/products/ps6692/Products_Sub_Category_Home.html)
- [6] Jason Ding and Abdul Waheed, "Dual Processor Performance Characterization for XML Application-Oriented Networking," in *proceedings of the 2007*

*International Conference on Parallel Processing (ICPP 2007)*, Xi'An, China, Sept. 2007.

- [7] Eugene Kuznetsov and Cyrus Dolph, XSLT Processor Benchmarks, March 2001. Available on-line from: <http://www.xml.com/pub/a/2001/03/28/xsltmark/>
- [8] David Mertz. Understanding ebXML: Untangling the Business Web of the Future. Available on-line from: <http://www-128.ibm.com/developerworks/xml/library/x-ebxml/>
- [9] Kanda Runapongsa, Jignesh M. Patel, H.V. Jagadish, Yun Chen, and Shurug Al-Khalifa, "The Michigan benchmark: Towards XML Query Performance Diagnostics," in *proceedings of the 29th VLDB Conference*, Berlin, Germany, 2003.
- [10] A. R. Schmidt, F. Waas, M. L. Kersten, D. Florescu, I. Manolescu, M. J. Carey, R. Busse, "The XML Benchmark Project," Technical Report INS-R0103, CWI, Amsterdam, The Netherlands, April 2001. Available on-line from: <http://www.cwi.nl/htbin/ins1/publications?request=abstract&key=ScWaKeFlMaCaBu:TR-CWI:01>
- [11] Simple Perl XML Benchmark. Available on-line from: [http://www.xmltwig.com/article/simple\\_benchmark/](http://www.xmltwig.com/article/simple_benchmark/)
- [12] XMark—An XML Benchmark Project. Available on-line from: <http://www.xml-benchmark.org/>
- [13] XML Parsing Benchmark. Available on-line from: <http://www.devsphere.com/xml/benchmark/index.html>
- [14] XMLSec Library. Available on-line from: <http://www.aleksey.com/xmlsec/>

**Abdul Waheed** received a B.Sc. degree with honors in electrical engineering from University of Engineering and Technology, Lahore, Pakistan in 1991. He received MS and PhD in electrical engineering from Michigan State University (MSU), East Lansing, Michigan, USA in 1993 and 1997, respectively.

He has a track record of ten years of working in research, industry, and academia after graduating from MSU. He worked at NASA Ames Research Center, Moffett Field, California, USA as a member of research staff for three years. He then moved to industry and worked for one year at Inktomi Corp., in Foster City, California, USA. Subsequently, he served as an Assistant Professor in the Computer Engineering Department of King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia for three years. For last three years, he has been

associated with the Application Oriented Networking business unit of Cisco Systems, Inc. in San Jose, California, USA as a Sr. Performance Engineer. His professional interests span performance evaluation, high performance computing and networking, performance tools, and hardware accelerators for XML processing.

Dr. Waheed is a member of IEEE Computer Society. He has published more than thirty papers in peer reviewed conferences and journals.

**Jianxun Jason Ding** received a Ph.D. degree in computer science from Texas A&M University, College Station, Texas, U.S.A. in 1994. He received M.Sc. and B.Sc. degrees in computer science from Fudan University, Shanghai, China, in 1987 and 1984, respectively.

He is currently a Sr. Manager of Cisco Systems' Application Oriented Networking Business Unit (AON BU) where he leads the AON Systems Architecture and Performance (ASAP) team in San Jose, USA. Prior to joining Cisco in May 2005, He was a Sr. Manager at Ariba Inc. (2000-2005), where he managed performance and scalability for 10+ product releases, initially for Ariba's e-Procurement core product and later on for all Ariba's enterprise applications. He also worked at Intel (1994-2000) as a Computer Architect on performance of Pentium, Xeon and Itanium products. He was a Computer Science Faculty Member at Fudan University and participated in design and implementation of China's very first OSI network. He also conducted research at British Telecom Research Labs in UK, summer job in NEC Systems Labs, and Research Assistant work in Texas A&M University on NSF sponsored projects. His professional interests focus on solving performance issues in both computer hardware and software products.

Dr. Ding is a member of IEEE and ACM. He has served as Industry Relations Editor of IEEE Computer Architecture Newsletter from 1997 to 1998 and as organizing committee member for IEEE High Performance Computer Architecture (HPCA-4), ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VIII), ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS-2006). He has published more than ten papers in peer reviewed conferences and journals.