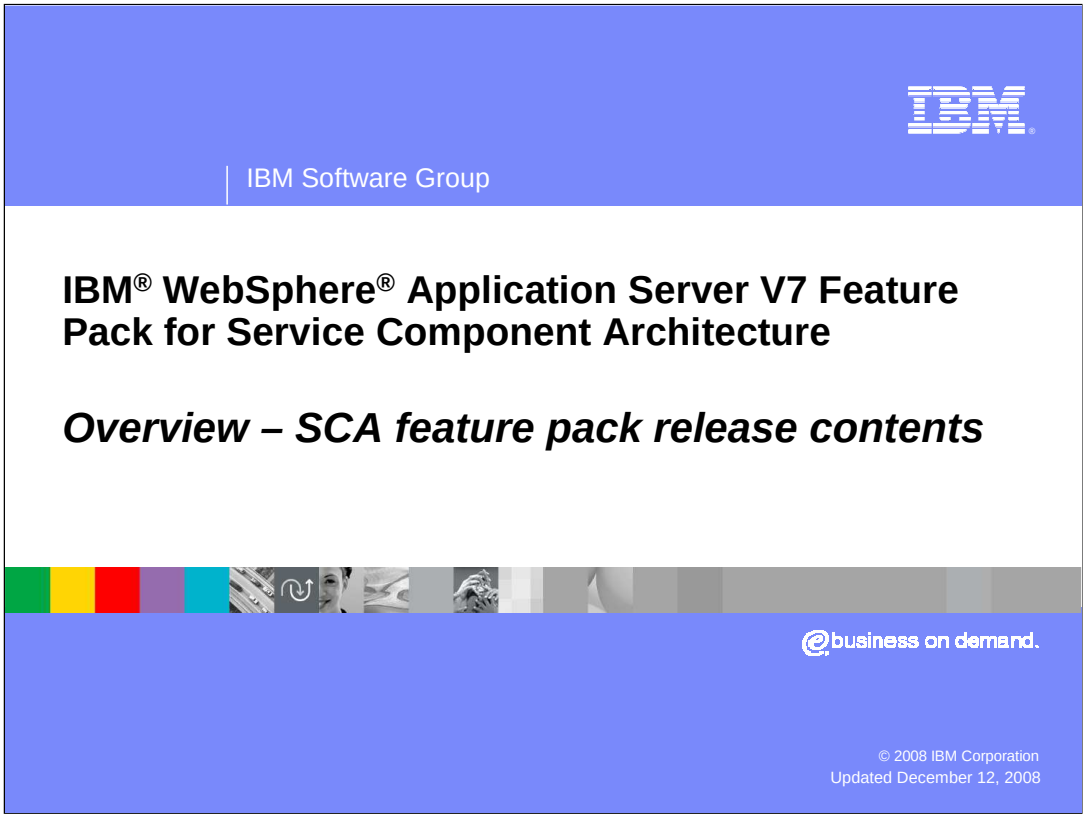


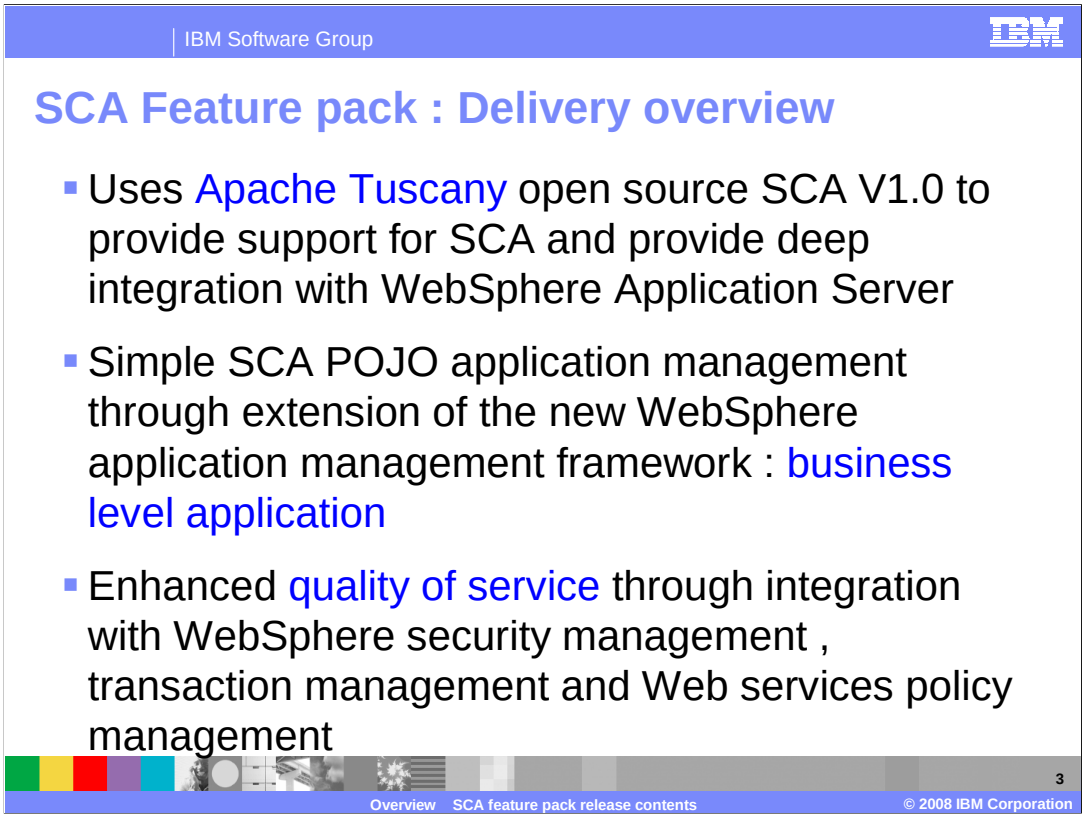
This presentation will cover an overview of the WebSphere Application Server V7.0 Feature Pack for SCA release specifically the contents of the SCA feature pack.

Agenda

- SCA FP release contents
- Summary and references

© 2008 IBM Corporation

This presentation will look at what's in the SCA feature pack at high level and finally summarize.



IBM WebSphere Application Server V7 Feature Pack for SCA delivers critical technology that enables adoption of Service-Oriented Architecture (SOA) principles.

As part of the larger SOA Foundation, which straddles all of IBM software brands, this feature pack delivers an integrated, open implementation of Service Component Architecture (SCA). SCA is a technology specified by IBM and other industry leaders through the Open SOA Collaboration (www.osoa.org).

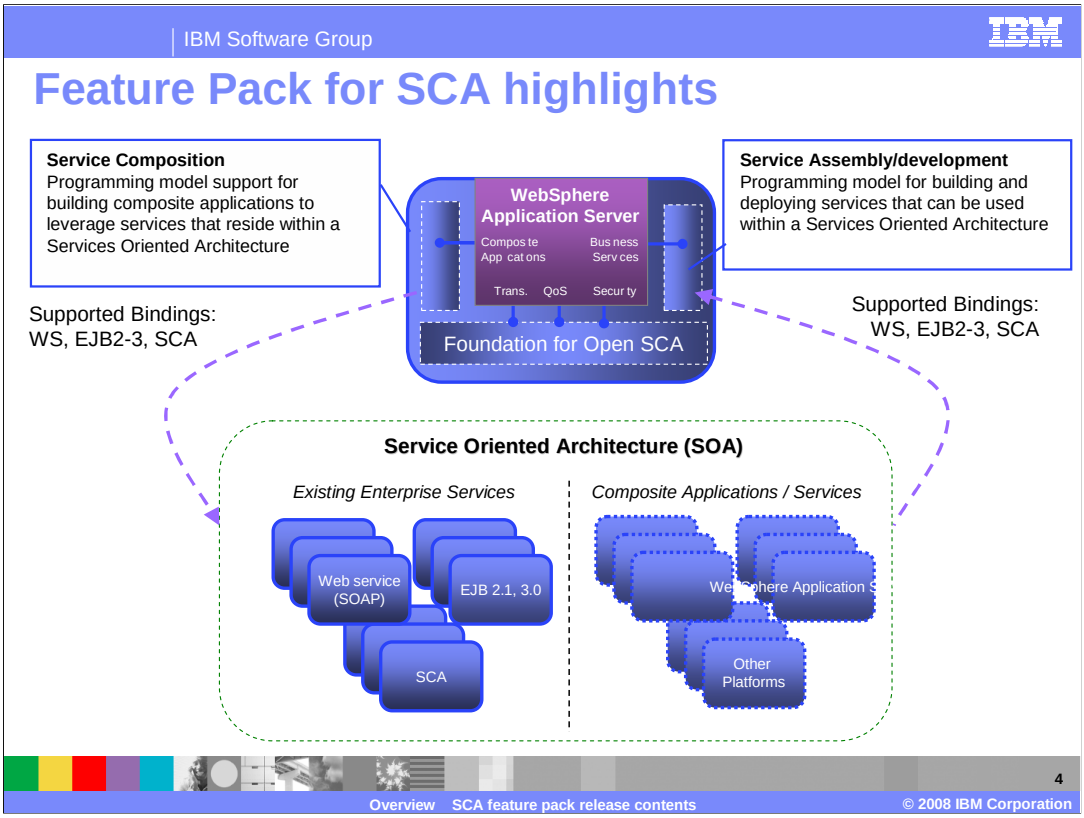
WebSphere has taken the open source SCA v1.0 implementation from Tuscany, an Apache project, and integrated it with WebSphere Application Server. This integration ensures that all of WebSphere's capabilities work together with SCA applications to provide a natural environment for both the SCA and existing applications.

IBM WebSphere Application Server V7 Feature Pack for SCA and the underlying Tuscany framework, is a "proof-point" delivery of SCA built using a plug-in concept. IBM plans to augment and enhance this initial support in subsequent releases by providing additional plug-in capabilities.

In this release there has also been additions in the application management with simple SCA POJO applications being extended into business level applications in WebSphere Application Server. Quality service has also been enhanced through the integration with WebSphere security management, transaction management and Web services policy management.

The primary objective of this initial release is to highlight usage of SCA. SCA enables users to create composite applications using new or existing assets. The key principle of SOA demonstrated by this support could be described as "Use what you have got and run it where it lives," or "Use your existing services to create new ones."

Another key objective of this initial delivery of SCA is to highlight the ease-of-use characteristics of SCA service development in Java™. This is accomplished by demonstrating annotated plain-old java-object (POJO) components deployed using simple JAR packaging schemes, an easy to use assembly model, and powerful wiring abstractions that enable service definition over different transports and protocols. It makes development and maintenance easier by moving protocols out of business logic. The key principle of SOA demonstrated by this support might be described as "Know only what you need to know to get your job done," or "Maintain proper separation of concerns."



WebSphere has chosen to embrace the open source Apache project Tuscany to provide the core of the SCA runtime and then support with the world-class enterprise application server functions inherent in the WebSphere Application Server V7 were integrated.

This picture shows the specific highlights of the SCA feature pack. What is contained in the feature pack can be classified into three groups:

(1) Features under service composition on the application server.

(2) Features under Simple Service development

(3) and features under WebSphere application server SCA foundation – agility and flexibility

Service Composition

Businesses today are challenged not only by competitors, but by social and economic pressures that directly affect their information technology systems. As businesses adopt SOA and build a growing inventory of business services, there is a real need to be able to compose, reuse, and otherwise assemble new services from those existing business services.

SCA offers a metadata assembly and composition model for assembling and constructing coarse-grained services out of software components and other services. In a sense, SCA applies the hardware circuit-board paradigm to software programming. While service implementations are vital to the functioning of the overall application, they can also be viewed as “chips” because details of their inner workings are hidden from the assembler.

SCA assemblies provide the metadata language to describe the chips, hide certain details, and provide wiring and binding semantics to the workings inside the chips, and to those exposed outside the chip. This metadata assembly language is called Service Component Definition Language (SCDL). In the hardware realm, wires have physical constraints, such a certain voltage range or frequency of operation. Software “wires” have similar constraints which are expressed as SCA policy.

Service Development

SCA has a language-neutral programming model for which there are multiple language-specific specifications defined at OSOA. The language-specific component models include Java, Spring, and C++. Being a Java runtime, WebSphere supports SCA in Java in a first-class and natural way.

The concepts of SCA apply broadly across both Java and non-Java application environments. The SCA component model has at its heart a strong focus on a proper separation of concerns. The service consumer business logic author should not need to know the details of the service implementation. For instance, a Java service consumer should not be burdened with having to know that a target service is implemented using C++ or COBOL.

A logical name can be used to identify a service, for example, “MyStockQuoteService,” and that name can be used in SCA wires to delegate the specific details of service connectivity to the SCA runtime. Essentially, the application programmer is telling SCA to use the best connectivity alternatives available to wire the service consumer to the service providing “MyStockQuoteService.”

Service Agility and Flexibility

One of the key reasons for SOA is to provide a set of patterns and best practices–formalized through infrastructural concepts and products--that allows businesses to realign and remission multiple aspects of information technology. In short, IT professionals need to be able to rewire, recast, and reuse applications in very flexible ways to keep up with business needs in the tumultuous and dynamic business climates faced today.

IBM's initial delivery of SCA in this feature pack highlights the flexibility and agility of metadata bindings, and the appropriate separation of concerns. The ability to rewire, compose, and assemble business logic without impacting the business logic itself is key.

Functional description – Highlights (1)

- Based on Tuscany SCA Java implementation
- POJO service component implementations
  - Tuscany component container
- Bindings for
  - Web services (WS feature pack), and
  - EJB2 and EJB3
  - Admin Extensions – panels, commands, application install (based on BLA), MBeans
- Native SCA2 packaging model (jars)

© 2008 IBM Corporation

The SCA feature pack is consuming a subset of Tuscany that implements **SCA V1.0. In particular it includes assembly and Java support and bindings such as**:

POJO Service component implementations.

Bindings which include Web services, EJB2 and EJB3.

The protocol used internally for default binding is currently is IIOP (for cross-JVM)

Admin Extensions – application management piece on business level applications

Native SCA2 packaging model which include jars.

Functional description – Highlights (2)

- Policy framework – interaction, implementation policy
  - Qos Extensions (security, transactions, reliability)
  - Web service profile integration
- JAXB data bindings
- RAS (log/trace), FFDC, ARM
- Technology samples

© 2008 IBM Corporation

In the area of policy framework, security policies, transaction policies, and reliability.

In databindings, JAXB databinding is implemented.

Technology samples such as multiservice, candystore, jobbank and others showing different bindings have been implemented and are also included.

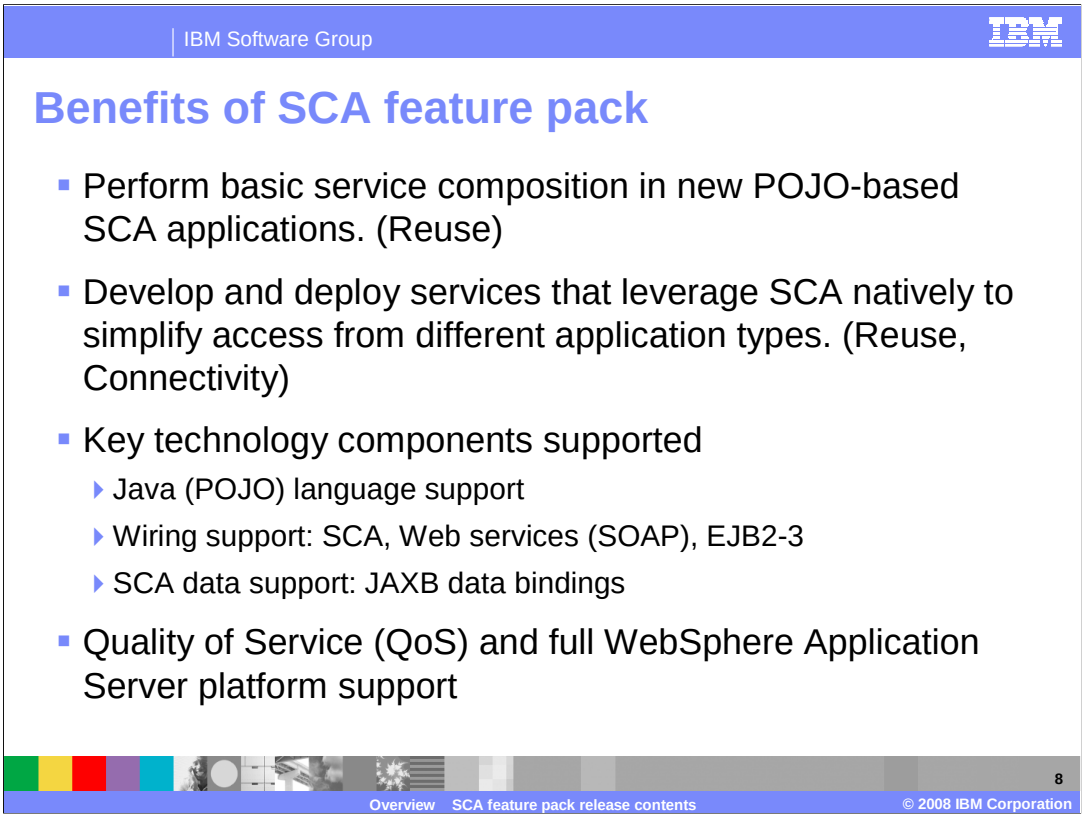
** *ARM* is an Open Group standard to measure the performance and availability of applications or business services. It's instrumented by the applications in development phase, and used at run time to analyze the transactions that are involved.

Security

- Based on SCA V1.0 Policy Framework
  - Abstract intents mapped to WebSphere Application Server PolicySets
  - Authentication, authorization, Runas, WS Policy
- Interaction policy
  - Transport security, authentication, message protection
  - WS-Security, WS-I BSP
- Implementation policy
  - Role assignment, authorization rules, runas role

© 2008 IBM Corporation

For security, intents, policy sets, authorization policies are all implemented.



The benefits of SCA Feature pack include the ability to perform basic service composition in new POJO-based SCA applications. (Reuse). Another benefit is to develop and deploy services that leverage SCA natively to simplify access from different application types. (Reuse, Connectivity)

Key technology components supported:

Java (POJO) language support

Wiring Support: SCA, Web services (SOAP), EJB2 and EJB3.0

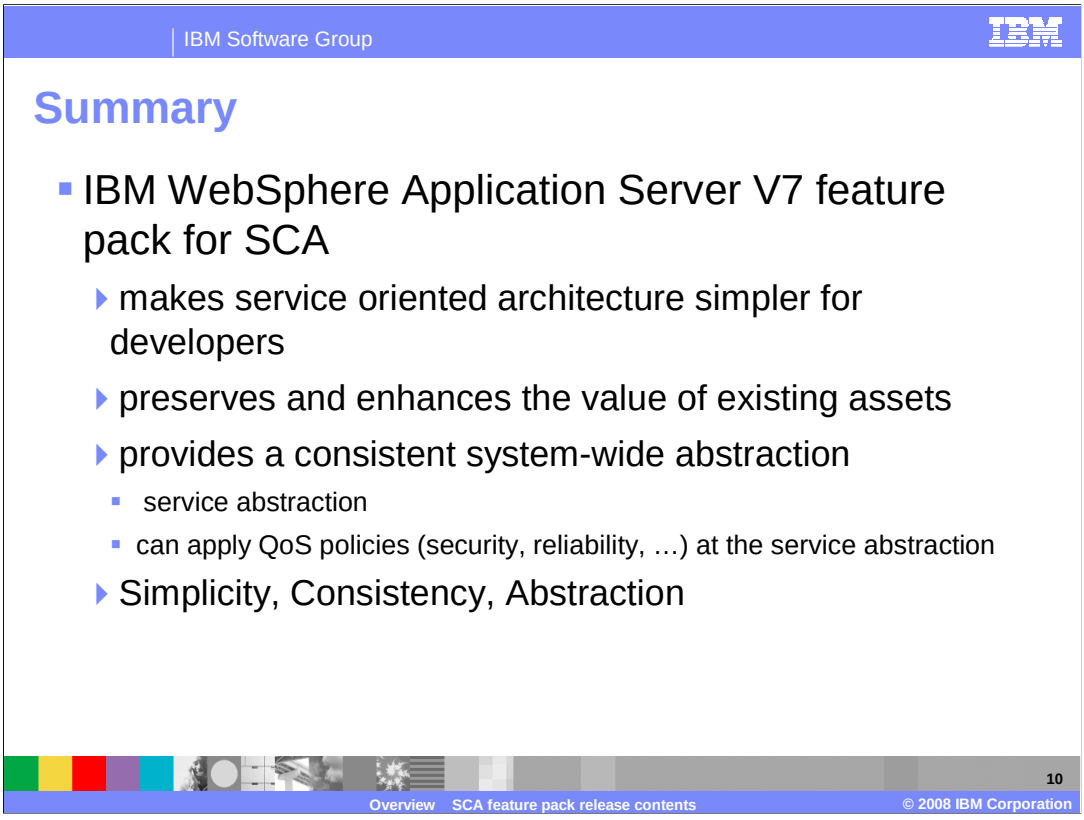
SCA Data Support: JAXB static data bindings

Quality of Service (QoS) and full WebSphere Application Server platform support

Section

Summary and references

© 2008 IBM Corporation



SCA Feature pack makes SOA simpler for developers in that application is structured as services and components. Business logic is not coupled to deployment infrastructure and there is wide choice of component kinds (use the right tool for the job).

As a result, this makes developers stay focused on solving business problems, rather than getting bogged down in the individual complexities of the technologies that connect service consumers and service providers. SCA feature pack preserves and enhances the value of existing assets which can be exposed as SCA services. SCA feature pack can model and integrate assets from a heterogeneous mix of new and existing runtime environments and provide a consistent system-wide abstraction. Service abstraction captures the SOA design and its mapping to underlying implementations. SCA feature pack can apply QoS policies (security, reliability, and others) at the service abstraction level across different runtime implementations.

References

- Open Service Oriented Architecture Web site
  http://www.osoa.org/
- Apache Tuscany Web site
  http://incubator.apache.org/tuscany/
- developerWorks®
  http://www.ibm.com/developerworks/websphere
- SCA white papers – open SCA FP beta site:

https://www14.software.ibm.com/iwm/web/cc/earlyprograms/websphere/soawas61/

© 2008 IBM Corporation

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_WASv7SCA_Overview_SCAfp_Releasecontents.ppt

This module is also available in PDF format at:
../WASv7SCA_Overview_SCAfp_Releasecontents.pdf

© 2008 IBM Corporation

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

developerWorks WebSphere

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at http://www.ibm.com/legal/copytrade.shtml

EJB, Java, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.