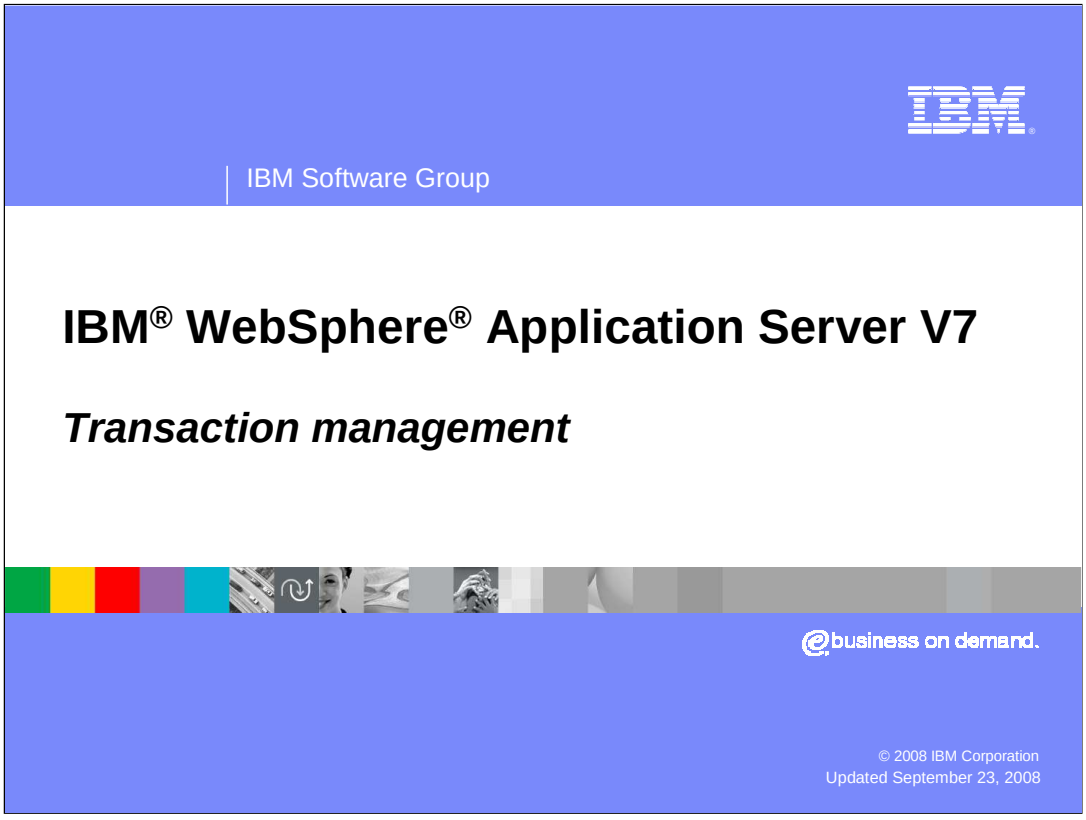


This presentation will discuss transaction management for WebSphere Application Server V7.0 on z/OS®.

Agenda

Transaction management

This presentation will briefly discuss transaction management.

Transaction management

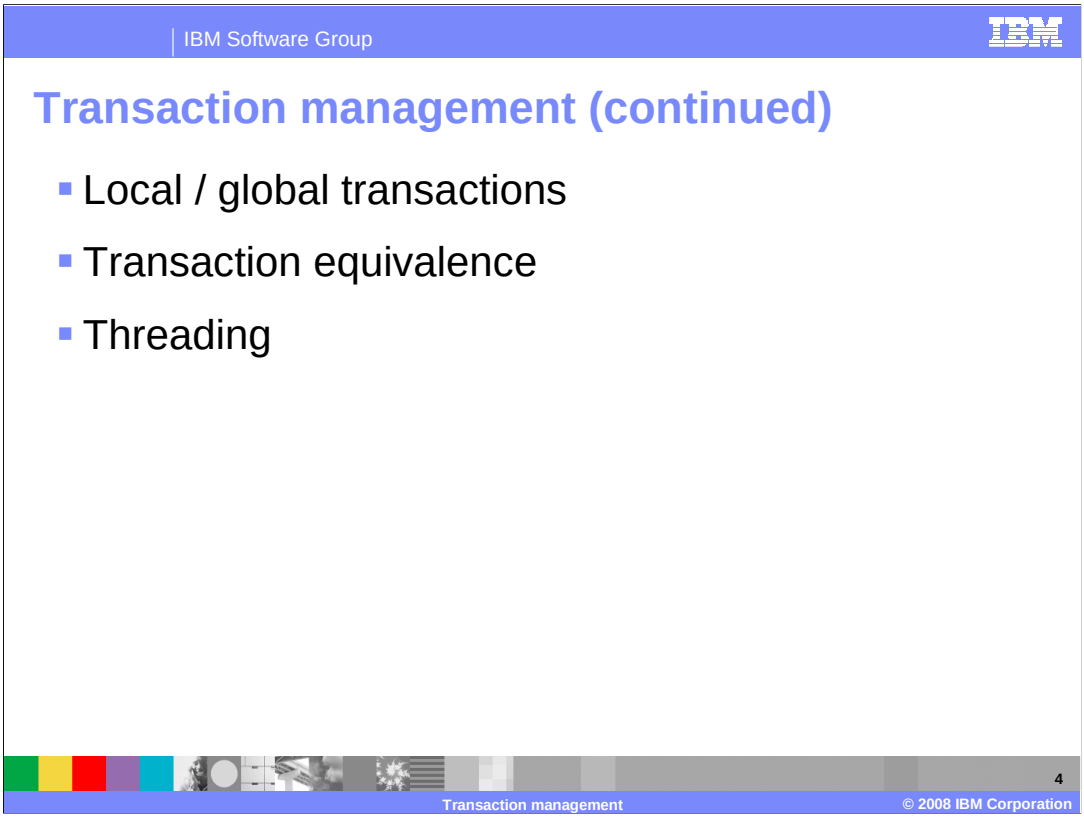
- Support Java Transaction API 1.1
  - Provides mechanism by which components can register with a transaction
  - TransactionSynchronizationRegistry interface
  - UOWSynchronizationRegistry interface
- Two additional units of work are provided
  - ActivitySessions
  - Local transaction containments

To support JTA 1.1, part of Java EE there is an implementation of the new TransactionSynchronizationRegistry interface. The new interface is a standardized alternative to some of the WebSphere Application Server-proprietary extended JTA transaction support. It provides a mechanism by which components, such as persistence managers, can register with a transaction to, for example, flush a cache upon transaction completion.

In addition to support for JTA transactions WebSphere Application Server also provides two further types of unit of work (UOW) that can be used by application components to coordinate transactional work; ActivitySessions and local transaction containments (LTCs). To ensure that support is consistent across all types of unit of work equivalent functionality to that provided by TransactionSynchronizationRegistry will also be delivered in the form of a unit of work-agnostic UOWSynchronizationRegistry interface.

To provide robust support for vendor code that wants to manipulate unit of work contexts in WebSphere Application Server a new SPI, UOWManager, is provided that provides equivalent behavior to JTA's TransactionManager interface whilst ensuring that WebSphere Application Server remains in control of its unit of work contexts at all time.

Although a full implementation of JTA 1.1 is provided, WebSphere Application Server requires applications to restrict their use of JTA APIs to the subset of APIs defined for application use by the EJB and Java EE specifications. Specifically, WebSphere Application Server does not provide access to the JTA Transaction Manager interface to applications and ISVs. WebSphere Application Server offers extended, "safe" SPIs to applications and ISVs that do not compromise the integrity of the runtime.



Local/global transactions

WebSphere Application Server specifies the EJB “unspecified transaction context” in terms of a local transaction containment (LTC). An application component always runs under a unit of work context – either a global transaction context or an LTC context. The TM ensures one or the other is present; the only exception to this is when a Web application starts an unmanaged thread, which then really does run under an unspecified transaction context.

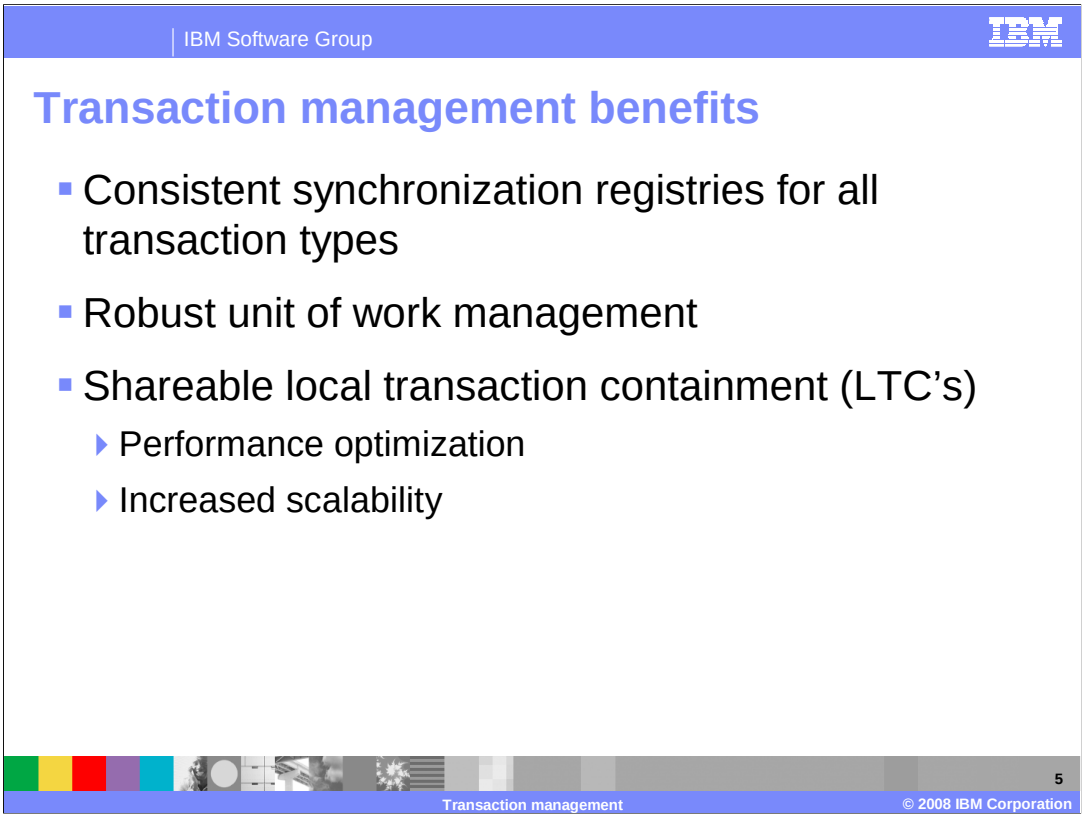
Transaction equivalence

All supported two phase commit transaction formats are normalized to a common representation by the TM implementation. An XAResource accessed under a JTA transaction, a received OTS context or a received WS-AT context is just running under a global transaction – it doesn’t matter where that transaction was received from or with what transaction protocol the remote partner is coordinated.

Threading

A transaction context can be associated with 0 or 1 threads at any given time in a WebSphere Application Server process. A transaction context is *never* associated with more than one thread concurrently. This is policed by the TM at process boundaries and any attempt by a client to send a context *T1* to a server which is concurrently processing work for *T1* is rejected. Threads begun by an application component that is running under a global transaction context (where allowed) do not themselves run under any global transaction context.

Transaction contexts might be (synchronously) loop-back into the same server but the transaction is logically suspended from the originating thread while it is blocked processing on the loop-backed thread.



Synchronization registries

By providing both a TransactionSynchronizationRegistry and UOWSynchronization registry both compliance with Java EE and maintaining a consistent story for transaction support across all unit of work types is ensured.

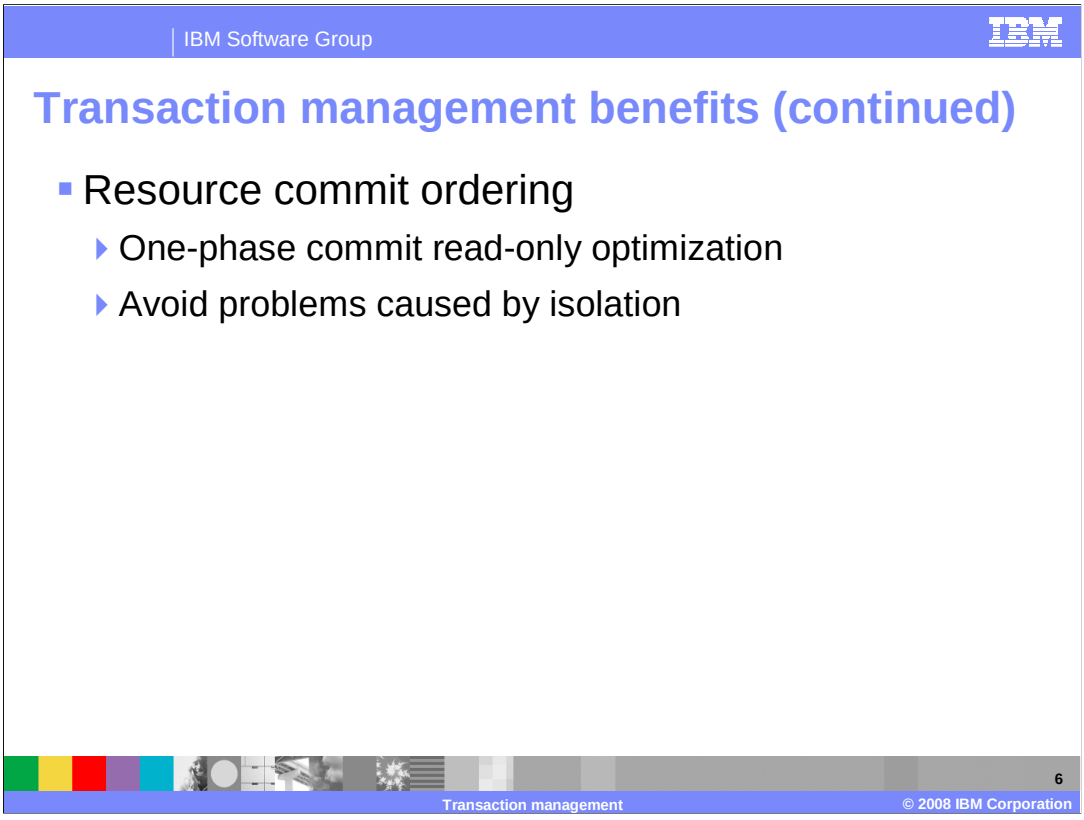
Robust unit of work management

Providing the UOWManager SPI enables vendor and stack product code to manipulate unit of works in a robust, supported fashion that does not run this risk of undermining the existing runtime. This allows you to declare support for vendor frameworks such as Spring.

Shareable LTCs

The use of shareable LTCs will allow multiple application components to share a database connection without the use of a global transaction. This sharing of connections will enable increased throughput and allow a server to support a larger number of concurrent users due to lower database connection requirements.

The new concept of shareable LTCs provides greater flexibility for resource-provider connection sharing within the application server, with benefits that include additional opportunities for performance optimization and increased scalability. A connection can only be shared within a "sharing scope"; before the introduction of shareable LTCs the only way to share a connection across multiple application components was to use a global transaction as that sharing scope. The concept of a shareable LTC brings with it a change to the concept of connection sharing because it becomes possible to share a connection between multiple application components by using a shared LTC in addition to the existing mechanism of using a global transaction.



Resource commit ordering

Allowing an application developer to control the order in which transactional resources are processed during two-phase commit provides two main benefits; it can increase the number of occasions where a one phase commit optimization occurs, and it can solve problems caused by transaction isolation.

One-phase commit read-only optimization

An application developer knows that some of the transactional resources used in the application will only be used to perform reads. The developer assigns commit priorities to all the resources in the application such that the resources performing reads are committed first. This will increase the chances that the transaction manager can perform a one-phase commit optimization giving a performance benefit.

Isolation

Isolation of resources involved in a global transaction means that updates made as part of a transaction are not visible outside the transaction until it commits. This isolation can cause problems with other application components that act upon the updates once they are committed. By allowing an application developer to specify the order in which their transactional resources are committed this isolation problem is resolved for all transactional back-end systems, (not just messaging providers and systems integration bus)

Client enablement

- No thin client support
- No Java EE client runtime support

Note: limited capability for JTA UserTransaction interface is available in the thin client and Java EE client runtimes

The transaction management capabilities that have been discussed are provided only in a server environment, not a thin client or Java EE client runtime environment, with the single exception that the JTA UserTransaction interface is available, with limited capability, in the thin client and Java EE client runtimes.

There is no LTC support in a client. There is no support for the UOWSynchronization, UOWManager or ExtendedJTATransaction interfaces on the client.

z/OS specifics

- Transactional coordination of and by z/OS Resource Recovery Services (RRS)
- Split-process design of one control region running with one or more servant regions

The WebSphere transaction manager design accommodates platform-specific features of the z/OS platform, in particular, transactional coordination of and by z/OS RRS and split-process design of one control region running with one or more servant regions.

For the most part, however, the implementation is common between z/OS and distributed platforms.

Deprecated functions

- These functions have been deprecated:
- registerSynchronizationCallbackForCurrentTran method()
- TransactionControl() interface

The method registerInterposedSynchronization on the new JTA 1.1 TransactionSynchronizationRegistry interface provides functionality that is directly equivalent to that of the registerSynchronizationCallbackForCurrentTran method on the WebSphere Application Server-proprietary ExtendedJTATransaction interface in the com.ibm.websphere.jtaextensions package. As a result the registerSynchronizationCallbackForCurrentTran method is marked as deprecated in favor of the new JTA 1.1 registerInterposedSynchronization method.

The UOWManager runUnderUOW method provides functionality that is equivalent to the extension helper TransactionControl interface. As a result the TransactionControl interface is marked as deprecated in favor of UOWManager's runUnderUOW method.

Summary

- This presentation has covered transaction management enhancements provided by WebSphere Application Server V7.0

This presentation has reviewed the transaction management feature available in WebSphere Application Server for z/OS V7.

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_WASv7_TransactionManagement.ppt

This module is also available in PDF format at: ../WASv7_TransactionManagement.pdf

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

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM WebSphere z/OS

A current list of other IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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.

Information is provided "AS IS" without warranty of any kind. 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.

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.