



IBM Software Group

IBM WebSphere Application Server V6

Last participant support and ActivitySessions



@business on demand.

© 2004, 2006 IBM Corporation
Converted to video May 13, 2015

This presentation will discuss the Last Participant Support feature and the ActivitySessions service.

Goals

- Understand Last Participant Support functionality
- Understand ActivitySession functionality
- Be able to determine under which scenarios you might use either of these services



The goals for this presentation are to be able to understand the Last Participant Support and ActivitySession Service functionality and to be able to determine under which circumstances you would want to take advantage of either of these extensions.

Agenda

- Last Participant Support
 - ▶ Configuration
- ActivitySession Service
 - ▶ Configuration
- Summary

The agenda for this presentation covers an overview of Last Participant Support and how to configure it, and an overview of the ActivitySession Service and how to configure it.

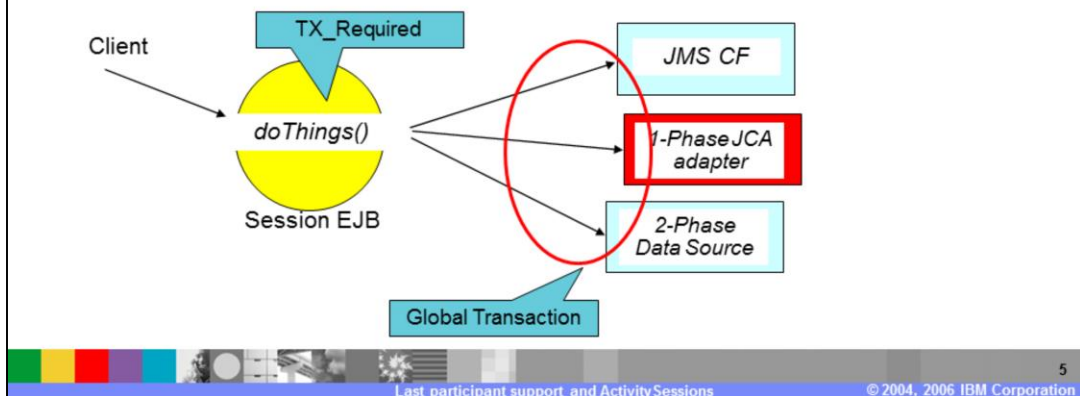
Section

Last participant support

This section will provide you with an overview of the Last Participant Support extension.

Last participant support: Overview

- Allows a single-phase resource to participate in a global transaction
 - ▶ Example: a single-phase Java™ 2 Enterprise Edition (J2EE) connector (JCA adapter)
- Support limited to *exactly one* single phase resource



Last Participant Support (LPS) allows applications to use a single, one-phase resource with one or more two-phase resources, in a global transaction. Certain applications require this functionality. For example, consider the scenario where a message is placed on a queue, an update is made to a record in an Enterprise Information System (EIS), and an update is made to a relational database. The requirement is to coordinate the Java™ Messaging Service (JMS) and relational database resources, which are two-phase resources, with the EIS record update, which is a one-phase resource, all within a single global transaction. By using Last Participant Support, you are able to do this.

How does this work? At transaction commit time, the two-phase commit resources are prepared first using the two-phase commit protocol, and if this is successful, the one-phase commit resource is then called to commit. The two-phase commit resources are then committed or rolled back depending upon the response of the one-phase commit resource.

Last participant support: Configuration

- Within the tool, enable LPS at the Enterprise Application level in the Application Deployment Descriptor
- Within the Administrative Console, enable LPS by checking “Accept heuristic hazard” for your application
 - ▶ If this is not checked, an exception will occur when trying to enlist the single-phase resource in the transaction
 - ▶ A heuristic outcome can occur if the transaction service does not receive a response from the request to commit the single-phase resource
 - ▶ Configure the heuristic completion direction (commit, rollback, manual) under the Transaction Service configuration tab
- Enablement of additional Transaction Service logging is recommended

Enterprise Applications > PlantsByWebSphere >
Last participant support extension

Extension to the transaction service to enable a single one-phase resource to participate in a two-phase transaction with one or more two-phase resources.



WebSphere Application Server Toolkit or Rational® Application Developer, allow for specifying the intent to use Last Participant Support on an Enterprise Application basis within the application deployment descriptor. Last Participant Support is declarative in nature.

On the Application Server, use the Administrative Console to enable LPS by checking the “Accept heuristic hazard” checkbox, within your application settings. To view this Administrative Console page, click Applications > Enterprise Applications > *application_name* > Last Participant Support Extension and on the Configuration tab, check the box in front of Accept heuristic hazard. A heuristic outcome can occur if the transaction service does not receive a response from the request to commit the single-phase resource. You can configure the heuristic completion direction (commit, rollback, or manual) from the Transaction Service configuration tab in the Administrative Console.

It is recommended that you enable the additional Transaction Service logging when using LPS.

Section

ActivitySessions



This section will provide you with an overview of the ActivitySessions extension.

ActivitySessions: Overview

- Provides alternate unit-of-work model to standard EJB transactions
- Allows coordinating multiple One-Phase Resources into a single logical unit of work
 - ▶ Audience: users of connectors, EIS, legacy
- Extends the EJB lifecycle to provide for better EJB performance in long transaction scenarios
 - ▶ Audience: widespread
- Extends the lifetime of Local Transactions beyond standard boundaries
 - ▶ Audience: users of JDBC, JCA,



ActivitySessions provide an alternate unit-of-work model to standard EJB transactions.

This capability allows for the coordination of multiple one-phase resources into a single unit of work. For example, the coordination of updates of two entity beans backed by two separate, one-phase resource datastores.

By extending the EJB lifecycle, Activity Sessions reduce unnecessary use of heavier-weight transactions when the only requirement is to define the activation scope of an EJB. This capability is configured using the `EJBActivationPolicy` value for activation/passivation on an Activity Session boundary.

ActivitySessions also allow for extending the life of the local transaction beyond its normal boundaries. This provides a number of benefits related to EJBs and caching of those objects, which may improve performance.

ActivitySessions: Enabling the service

- From the Application Server Administrative Console, ensure the ActivitySessions Service is enabled

[Application servers](#) > [server1](#) > Activity session service

A unit of work service that can be used to coordinate one-phase resources or for extending the activation and passivation of an enterprise bean.

Configuration

General Properties	Additional Properties
<input checked="" type="checkbox"/> Enable service at server startup	Custom Properties
* Default timeout 300 seconds	
<input type="button" value="Apply"/> <input type="button" value="OK"/> <input type="button" value="Reset"/> <input type="button" value="Cancel"/>	

9

Last participant support and ActivitySessions

© 2004, 2006 IBM Corporation

To use the ActivitySessions service on the WebSphere Application Server, ensure the service is enabled using the Administrative Console.

ActivitySession - Technical value

- Long-running transaction semantics and advantages without typical long-running transaction drawbacks
 - ▶ EJB Lifecycle influenced to avoid expensive activation/passivation cycles
 - ▶ Maximizes use of caching mechanisms
 - ▶ ActivitySession acts as a logical envelope around a number of Local Transactions
 - ▶ Provides a single point of control to establish consistency across Local Transactions

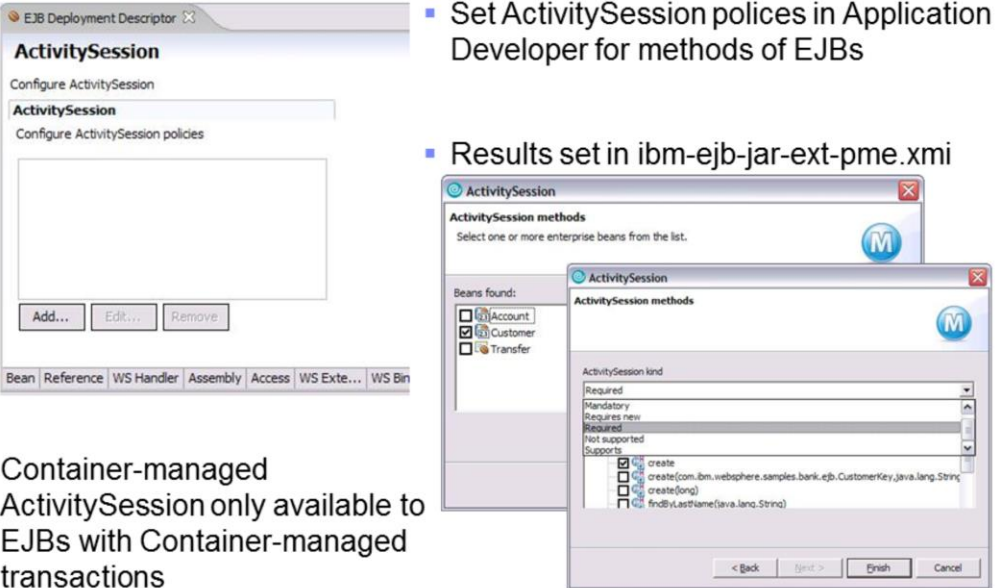


This chart articulates the fundamental technical value of ActivitySessions. Here you can see a number of single-phase, local transactions. Technically, it would not be possible to provide a single checkpoint for all these activities, although they may be logically related. For example, this scenario could correspond to a travel booking application, where different reservations may need to be made, such as car, hotel, flight, and so on. Each individual reservation may take place on different servers and the travel booking application may need to coordinate all the steps into a single unit of work. Then the user needs to confirm or cancel the entire reservation.

ActivitySessions allow you to coordinate the various transactions into a single unit of work and allow you to provide a single checkpoint for all the transactions that were initiated during the ActivitySession. Notice that at the end of the ActivitySession, you still have the choice of rolling back the local transactions, in spite of the fact that, theoretically, those changes were already committed.

The ActivitySession service of WebSphere Application Server captures the commit operations of each local transaction and upholds them. At the end of the ActivitySession, the container will return and commit (or roll back) each individual transaction. Keep in mind that the ActivitySession is not a substitute for two-phase commitment control, when it comes to data integrity. An ActivitySession may result in a mixed outcome, if some of the single phase resources successfully get committed before another resource fails to commit. In that case, the ActivitySession service will allow the programmer to retrieve the list of resources that were committed and those whose state is uncertain.

Deployment descriptor EJB attributes



- Set ActivitySession policies in Application Developer for methods of EJBs
- Results set in `ibm-ejb-jar-ext-pme.xml`
- Container-managed ActivitySession only available to EJBs with Container-managed transactions

11
Last participant support and ActivitySessions © 2004, 2006 IBM Corporation

Special extensions to the deployment descriptors allow you to specify the ActivitySession settings, using WebSphere Application Server Toolkit or Rational Application Developer.

Entity Beans are always container-managed from an ActivitySession standpoint and therefore there is no need to specify the ActivitySession attributes. Entity Beans will use any ActivitySession context that is provided to them by their callers.

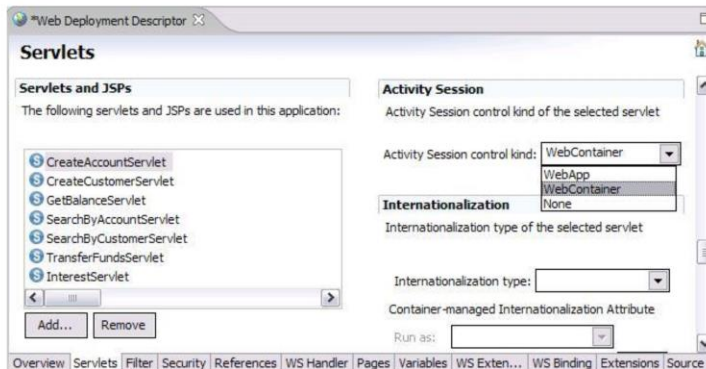
Session Beans will support an activity-session-type that matches the transaction-type. A Session Bean that is configured with container-managed transactions supports container-managed ActivitySessions. You can configure the ActivitySession policy for such a Session Bean using WebSphere Application Server Toolkit or Rational Application Developer.

Session Beans that are configured for bean-managed transactions will not support container-managed ActivitySessions. If any ActivitySession needs to be started by this type of Enterprise Java Beans, you need to use the programmatic approach.

Here, you see how you can define a container-managed ActivitySession policy by using Rational Application Developer. The policy needs to be defined on a method-by-method basis, and the dialog to define it follows very closely the semantics of the container transaction definitions.

ActivitySession definitions in the web components

- ActivitySession support can be set for Web components on the Extended Services tab of the web.xml deployment descriptor
 - ▶ Values stored in ibm-web-ext-pme.xml



12

Last participant support and ActivitySessions

© 2004, 2006 IBM Corporation

Web components can also be configured for the ActivitySession support using special deployment descriptor extensions. WebSphere Application Server Tool Kit or Rational Application Developer allow you to set the parameters for those extensions on individual servlets or Java™ Server Pages JSPs).

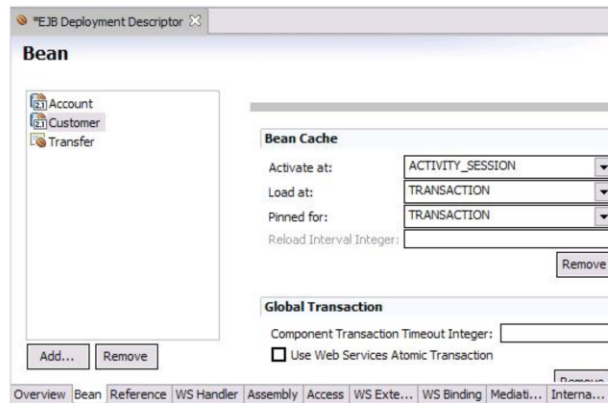
If you set the ActivitySession control to “WebApp” for a certain web component, that equates to choosing the programmatic approach. The programmer will be responsible for starting an ActivitySession from within a servlet/JSP and attaching it to the HTTP session, if the ActivitySession has to span multiple HTTP requests, by using the UserActivitySession APIs.

If you set the ActivitySession control to “WebContainer”, the container will automatically start a new ActivitySession every time a new HTTP session is created. The ActivitySession context will also be attached to the HTTP session automatically.

Setting the ActivitySession control to None equates to disabling this service for a specific web application (the default).

Influencing the EJB lifecycle using Rational Application Developer

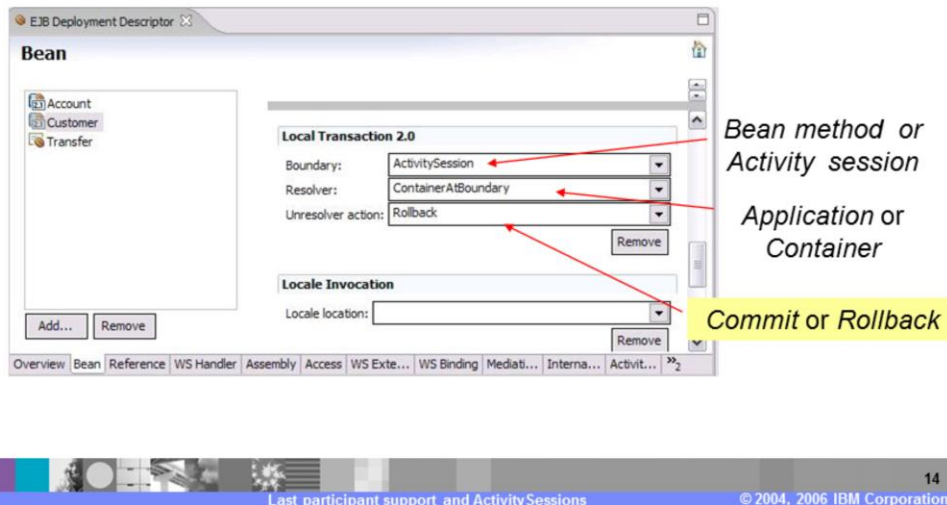
- EJB Lifecycle can be specified within the EJB Deployment Descriptor per bean



Here, you see how you can use the Rational Application Developer to set the caching options for EJBs. In particular, this example shows how to scope the EJB activation to the `ActivitySession` boundaries.

Scoping local transaction containment

- Local Transaction enlistment and containment is specified on a bean basis in EJB Deployment Descriptor editor
 - Results stored in `ibm-ejb-jar-ext-pme.xml`



You can choose between the local transaction containment scenarios and the local transaction enlistment. Rational Application Developer allows you to edit certain extended deployment descriptors that regulate the behavior of local transactions with respect to ActivitySessions.

If the local transaction boundary value is set to ActivitySession, a single local transaction can survive multiple method invocations, allowing the container to extend the normal lifecycle of a local transaction. The resolution-control can be either Container or Application. If you set this value to Container, the application server will take care of starting and committing local transactions. In this case, the ActivitySession will enlist the local transactions. If you set it to Application, it's up to the programmer to control the transaction boundaries and therefore to commit or rollback those transactions. This scenario corresponds to the Containment approach.

Notice the unresolved action parameter. This parameter can be set to either commit or rollback, and will determine what happens to any local transaction that wasn't yet completed (committed or rolled back) at the time the ActivitySession completed. Consider, for example, a travel reservation application where local transactions are controlled by the programmer. The user has made a reservation, but has not yet clicked the final confirmation button, causing the local transaction to remain uncommitted. For some reason, the connection becomes unavailable, and after a certain period of time, the ActivitySession expires. The "unresolved action" value determines how that pending reservation will be treated. If you specify "commit", the transaction will be committed and

the reservation implicitly confirmed. If you choose “rollback” the reservation will be cancelled.

Section

Summary

The following section will summarize.

Summary

- Described last participant support
- Described ActivitySessions
- Provided sample scenarios



In this presentation you learned about Last Participant Support and ActivitySessions extensions. You were also provided with usage scenarios to help you understand how you might be able to use this functionality in your own applications.

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	CICS	IMS	MQSeries	Tivoli
IBM (logo)	Cloudscape	Informix	OS/390	WebSphere
e!(logo)/business	DB2	Series	OS/400	xSeries
AlX	DB2 UniversalDatabase	Lotus	pSeries	zSeries

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

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and 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 and/or changes in the product(s) and/or program(s) 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 (e.g., 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 2004, 2006. 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.