IBM Software Group

IBM WebSphere Extended Deployment V6

Partitioning Facility Overview

@business on demand.

© 2005 IBM Corporation
Converted to video June 18, 2015

This presentation will cover the partitioning facility in WebSphere® Extended Deployment V6.

IBM Software Group

Agenda

- Overview / example scenario
- Partitioning concepts
- Database partitioning
- HTTP partitioning
- Performance advantages

Partitioning Facility Overview

© 2005 IBM Corporation

This presentation will first introduce the partitioning facility through an example scenario that compares a classic Java™ 2 Enterprise Edition (J2EE) application to a partitioned application. It will then cover concepts associated with partitioning, to familiarize you with the partitioning facility. Database partitioning and HTTP partitioning will also be discussed. The presentation will end by summarizing the performance advantages of partitioning, and direct you to other resources for learning about the partitioning facility.

Section

Overview

3

This section will introduce the partitioning facility.

Partitioning Facility Overview

- The Partitioning Facility (PF) enables you to develop high performance, highly scalable J2EE applications
  - ▸ Includes a programming framework and supporting runtime infrastructure
- Partitioning creates unique endpoints within your cluster (or database) to handle specific work
  - ▸ Enables more efficient large-scale clustering
- Near-linear scalability on commodity hardware

The partitioning facility is a programming framework and runtime infrastructure that enables you to create J2EE applications that scale efficiently to support very high transaction volumes. This is accomplished by creating unique endpoints, or 'partitions', within a cluster to handle requests for particular data sets, which can also communicate with a partitioned database. Requests can then be consistently routed to the single known endpoint within the cluster that will handle the specific data set being accessed. The strategy effectively reduces contention for database resources, and also avoids some of the traditional challenges of large-scale clustering, for example large-scale replication of cache data. Applications that utilize the partitioning facility can achieve near-linear scalability on commodity hardware.
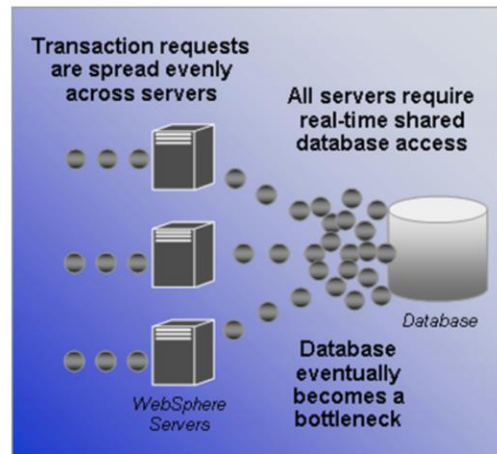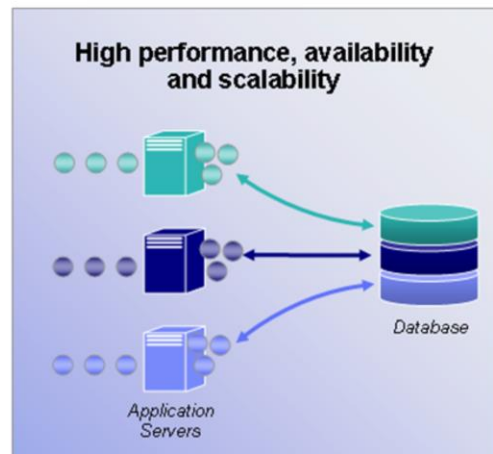
As an example, imagine a traditional environment that is hosting a very high volume application for which traffic is expected to increase over time. Requests are distributed equitably across all of the servers in the cluster. Each of these requests may require a database transaction, which is expensive, and any request that requires a database update will require a lock that prevents other servers from reading the affected data. In this scenario, the database is very likely going to be the performance bottleneck as volume increases. Also, if the servers are caching any data or require shared state information, the overhead of replicating that data can become large as the size of the cluster increases. If the quality of service declines in this environment under increased load, adding more application servers will probably not improve the situation.

Partitioning solves this scalability problem. The cluster is divided into several uniquely addressable partitions, based on the data sets they will work with, and the database is also partitioned on the same lines. When a request comes in for a particular set of data, such as a given stock symbol, it is predictably routed to a particular partition. The application server hosting that partition has exclusive access to the data related to the requested stock symbol, and exclusive access to the database instance that holds that data. This means that database read and write activities for the stock symbol 'ABC' will not cause contention with a request from another server. Additionally, the application can be designed to cache data in memory more aggressively, resulting in fewer database transactions, since no other servers will be accessing that data. As load on the system increases over time, servers can be added to the environment, spreading the application and database partitions across more servers. This gives more computing resources to each partition, as they are spread more thinly. In this scenario, adding hardware will allow the application to scale efficiently and keep up with the increased demand.

IBM Software Group

Section

Partitioning Concepts

Partitioning Facility Overview

© 2005 IBM Corporation

7

This section will discuss concepts associated with partitioning.

What is a Partition?

- A partition is a uniquely addressable endpoint within a cluster
  - Any cluster member can host a partition
  - A cluster member may host multiple partitions
- Partitions are created at startup when a partitioned application is detected
- Partition lifecycles are managed by the High Availability Manager (HA Manager)
  - Ensures availability and uniqueness of partitions

Partitioning Facility Overview

© 2005 IBM Corporation

A partition is a uniquely addressable endpoint within a cluster. Each cluster member might host multiple partitions, but each partition exists on only one cluster member. Each partition is started as a highly available singleton, managed by the High Availability Manager infrastructure. This ensures that the partitions remain constantly available while running in only one cluster member at any given time.

Request Routing

- When partitioned applications are detected, a special workload manager singleton for partitions is started

- IIOP requests are dispatched to an partition based on a 'key' (such as a passed parameter)
  - ▸ User-programmed logic determines how to map parameters to partitions
  - ▸ If no partition context is detected, request is routed using normal workload management

The WebSphere XD runtime provides a separate workload management router for partitioned applications, which is started automatically when a partitioned application is detected. This router locates and routes the request to the appropriate partition based on a partition key. The application defines partition keys and how they are mapped to partitions. The partition key could be a parameter passed to the method being called, or a hash based on such a parameter. If a partition key is not associated with the request, it will be handled by the normal workload manager, just like any other IIOP request.

What is a Partitioned Application?

- A partitioned application is a regular J2EE application that contains a Partitioned Stateless Session Bean (PSSB)

- A PSSB is a bean that uses the partitioning facility framework to create partitions according to your policy
  - Implements the PartitionHandlerLocal interface
  - Calls the HA Manager to create and activate partitions

A partitioned application contains one or more Partitioned Stateless Session Beans. These beans implement the PartitionHandlerLocal interface, and instruct the partitioning facility as to how partitions should be created and how requests should be mapped to partitions. The WebSphere XD runtime identifies partitioned applications by looking for these beans.

Deploying A Partitioned Application

- Run *ejbdeploy* against the EAR file to deploy the beans as usual

- Then run *wpfstubutil* to update the EJB stubs to reflect partition routing

- Order is important
  - Do not redeploy EJBs after running *wpfstubutil*

An extra step is required to deploy a partitioned application. Use the 'ejbdeploy' tool to deploy the enterprise beans, as you would for any other application. Then run the 'wpfstubutil' command, which modifies the EJB stubs to reflect the partitioned routing scheme. It is important not to redeploy the EJBs after calling 'wpfstubutil', because the partitioning-specific changes will be undone, requiring you to run 'wpfstubutil' again.

Controlling Partitions with wpfadmin

- Wpfadmin is a command-line utility for managing a partitioned environment using JMX
  - Wraps the wpfadmin.pty Jython script
  - Users are encouraged to use wpfadmin as an example and implement their own scripts

WebSphere XD provides a command line tool, 'wpfadmin', for managing a partitioned application environment. Wpfadmin allows you to view the location of partitions within your cluster, to move partitions from one server to another, and to configure the High Availability Manager, among other things. Wpfadmin is an interface to a Jython script, wpfadmin.pty, that is well documented and makes a useful example for implementing your own partitioning facility management scripts.

Partition Aliases

- Partition aliases are a new feature in WebSphere XD V6.0

- Using aliases gives you more flexibility in configuring and managing partitions

- Aliases can refer to one or more actual partitions

- Created programmatically using *PartitionDefinition.setPartitionAlias()*

Partition aliases are a new feature in WebSphere XD version 6. Aliases give you increased flexibility in how client requests are mapped to partitions. It allows you more flexibility in programming because partition names can be changed later if client code is using aliases. Partitions also give you the ability to group partitions within the same context, or to route requests to a partition based on a value other than the partition name.

IBM Software Group

Section

Database Partitioning

14

This section will discuss database partitioning.

Database Partitioning

- Partitioning a database can lead to increased scalability
  - Separate database instances host different subsets of the data
  - Each subset is accessed by a single WebSphere partition
  - Reduces contention
  - Increases usefulness of caching for write-heavy applications
- Database must be designed in a partitioned fashion
  - Not something that you can just 'turn on'

In addition to partitioning your application, partitioning the database that your application accesses can drastically increase scalability. Database partitioning involves creating a separate database instance for the subset of data that will be accessed by each partition. This gives each partition exclusive access to the data that it will be using, reducing database contention and giving the application the freedom to more aggressively cache values in memory. It also gives you the ability to easily scale the database across multiple servers, rather than implementing a database clustering solution. Database partitioning requires careful planning and consideration at the time of application design; it is not a feature for which there is a simple on-off switch

Data Source Configuration

- **Accessing a partitioned database requires a Proxy Data Source**
  - Special kind of Data Source that lets the application indicate which Data Source to use per-transaction
  - JNDI name of the target Data Source must be specified at the beginning of each transaction

- **Configure an actual Data Source for each database partition**

- **Create a Proxy Data Source to encapsulate the regular Data Sources**
  - Use the "Proxy JDBC Provider" to create the Proxy Data Source
  - Different implementation than the Proxy Data Source V5.1

WebSphere XD enables applications to access partitioned databases by utilizing a special Data Source, called a Proxy Data Source. A Proxy Data Source allows an application to specify the name of another Data Source to use for each database transaction. The application therefore has the ability to dynamically access different database instances based on the partition context of each request.

Database Partitioning Restrictions

- Currently only supports DB2® JDBC drivers and the Oracle type 4 (thin) driver

- CMP beans must be invoked using the local interface

- "Test connection" in Administrative Console will fail for Proxy Data Sources

WebSphere XD currently only supports database partitioning when using the DB2 JDBC drivers or the Oracle Type 4 JDBC driver. It is also important to keep in mind that beans using container-managed persistence must be invoked using their local interfaces. Also, the "test connection" feature of the Administrative Console does not support Proxy Data Sources.

This section will discuss HTTP partitioning.

HTTP Partitioning

- The On Demand Router (ODR) can integrate with the Partitioning Facility (PF) to implement partitioning for HTTP requests

- A partitioning facility module runs inside the ODR
  - Keeps an updated list of partitions
  - Gets current partition-to-server mapping from PF
  - Maps requests to partitions using regular expressions

- Partitions can be specified programmatically or by deploying partition.xml within the application

IBM Software Group

Partitioning Facility Overview

© 2005 IBM Corporation

19

WebSphere XD also provides support for partitioning of applications that are accessed over HTTP, which is particularly useful when Web module and EJB module are collocated. This means that HTTP requests are predictably routed to a single endpoint, just as IIOP requests are. To accomplish HTTP partitioning, the On-Demand Router interacts with the partitioning facility to route requests to unique partitions based on regular expressions specified by a PSSB or by a partition.xml file.

HTTP Partitioning Considerations

- Generic PSSB must always be packaged with the application, regardless of whether it has been modified

- Only cluster-scoped partitions are supported
  - ▸ Partition can only be active on one server

- Partitions can only be rebalanced manually

- Designed to work with static clusters only

- Session affinity will be overridden by partition affinity

Partitioning Facility Overview

© 2005 IBM Corporation

20

A partitioned HTTP application must always contain a Partitioned Stateless Session Bean, even if the partition mappings are defined in a partition.xml file, because the existence of the bean is what identifies an application as being partitioned. HTTP partitions can exist only on one server within a cluster, so node-scoped partitioning, which is an option for EJB partitioning, is not available for HTTP partitioning. Also, HTTP partitioning is not designed to work with Dynamic Clusters, and partitions can not be automatically rebalanced.

Many potential performance advantages are inherent in the partitioned application model. Partitioned applications can cache data very aggressively, and these caches are effective even with high write rates. The need for overqualified updates introduced by optimistic locking is reduced when using partitioning, since each partition has exclusive access to a database instance. Partitioning can also help increase parallelism within an application, because lock contention within the application server itself can also be reduced. Lastly, applications can be designed to take advantage of the fact that they exclusively handle requests for certain data sets by performing batched database writes to decrease load on the database.

Getting Started with Partitioning

- Partitioning Facility user's guide
  - Included in the WebSphere XD Information Center
  - Covers managing, programming, performance tuning, and more
- Several sample applications provided with WebSphere XD
  - Samples cover all types of partitioning
  - Found in <INSTALL_ROOT>/installableApps
- Partitioning API Javadoc
  - Found in <INSTALL_ROOT>/web/xd/apidocs/index.html

Several helpful resources are available to help you get started with the partitioning facility. The partitioning facility user's guide is included in the WebSphere Extended Deployment Information Center, and thoroughly covers developing, managing, and tuning partitioned applications. You can also find several samples of partitioned applications in the 'installableApps' directory after installing WebSphere XD. Lastly, the Javadoc installed with the product is the official documentation for the partitioning API.

Summary

- WebSphere XD includes support for 'partitioned' applications

- Partitioning eliminates some of the overhead traditionally associated with clustering
  - Data contention, caching, data replication

- Provides near-linear scalability for high transaction volume applications

In summary, the partitioning facility in WebSphere XD is a programming framework and runtime infrastructure that enables you to build high-performance J2EE applications that can scale efficiently on distributed hardware. Partitioning addresses the drawbacks typically associated with large-scale clustering, such as data contention, distributed caching, and data replication.

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	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	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, 2005. 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.