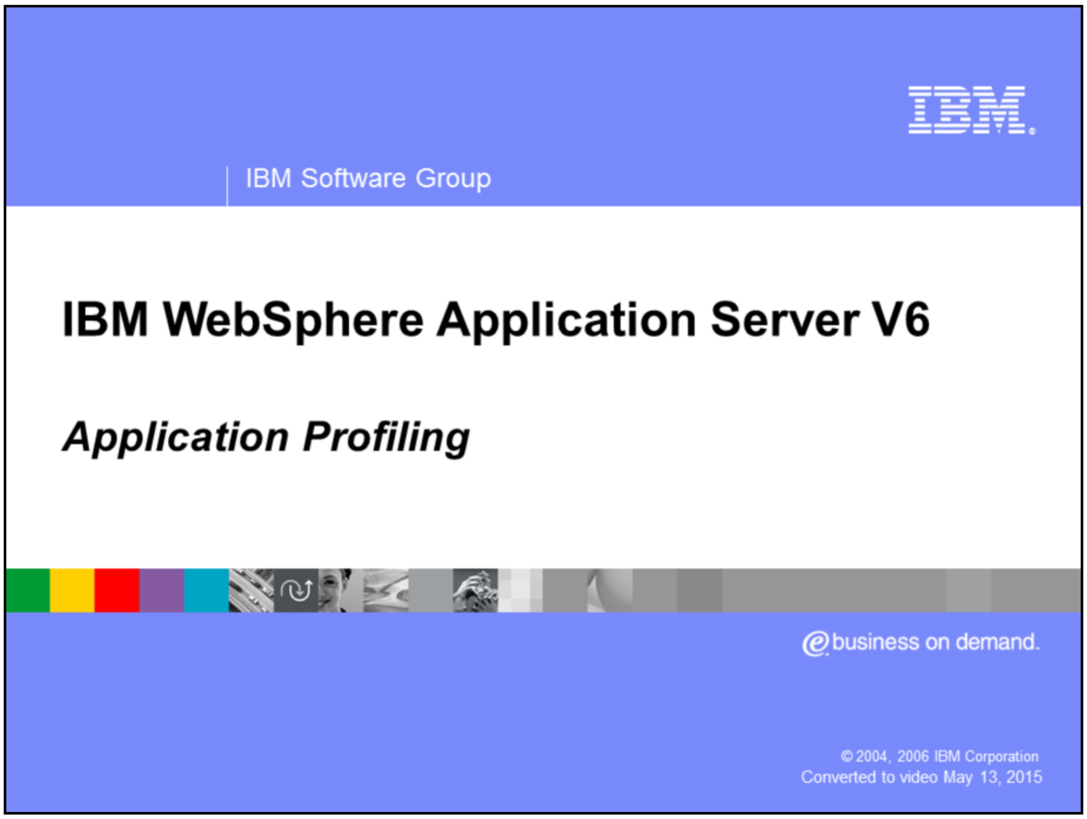


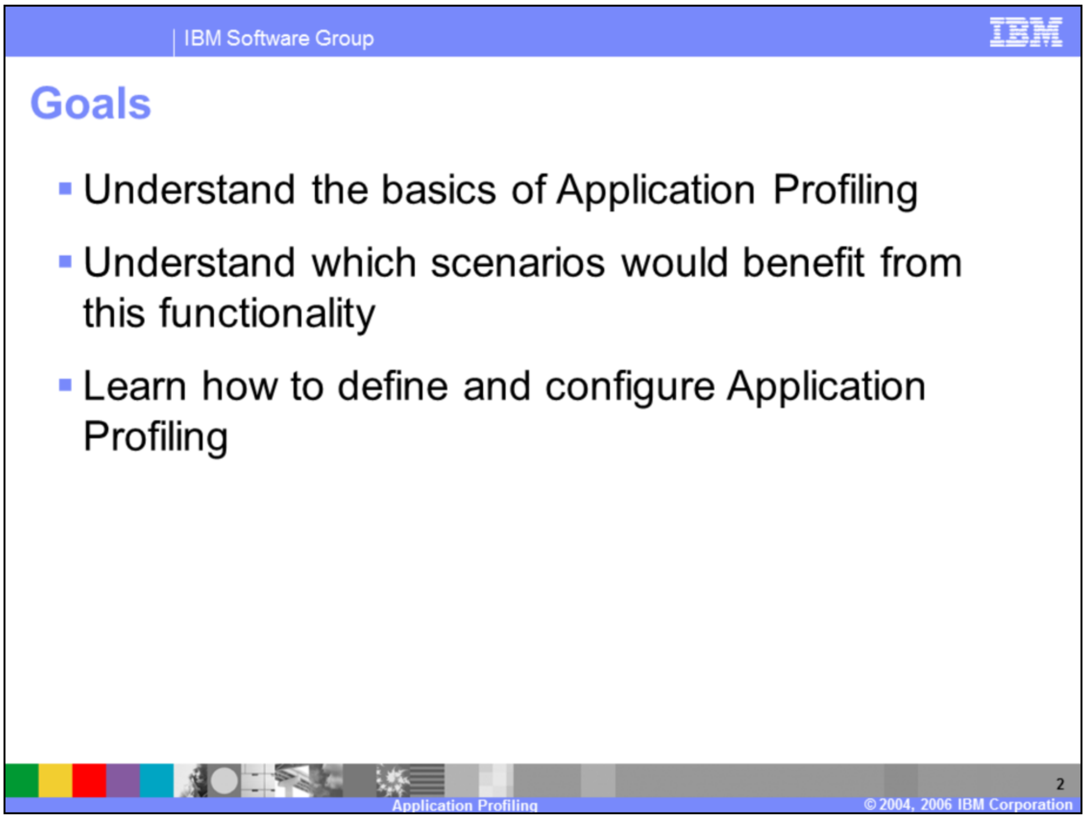
IBM Software Group

IBM WebSphere Application Server V6

Application Profiling

e business on demand.

This presentation will discuss the Application Profiling feature.


IBM Software Group

Goals

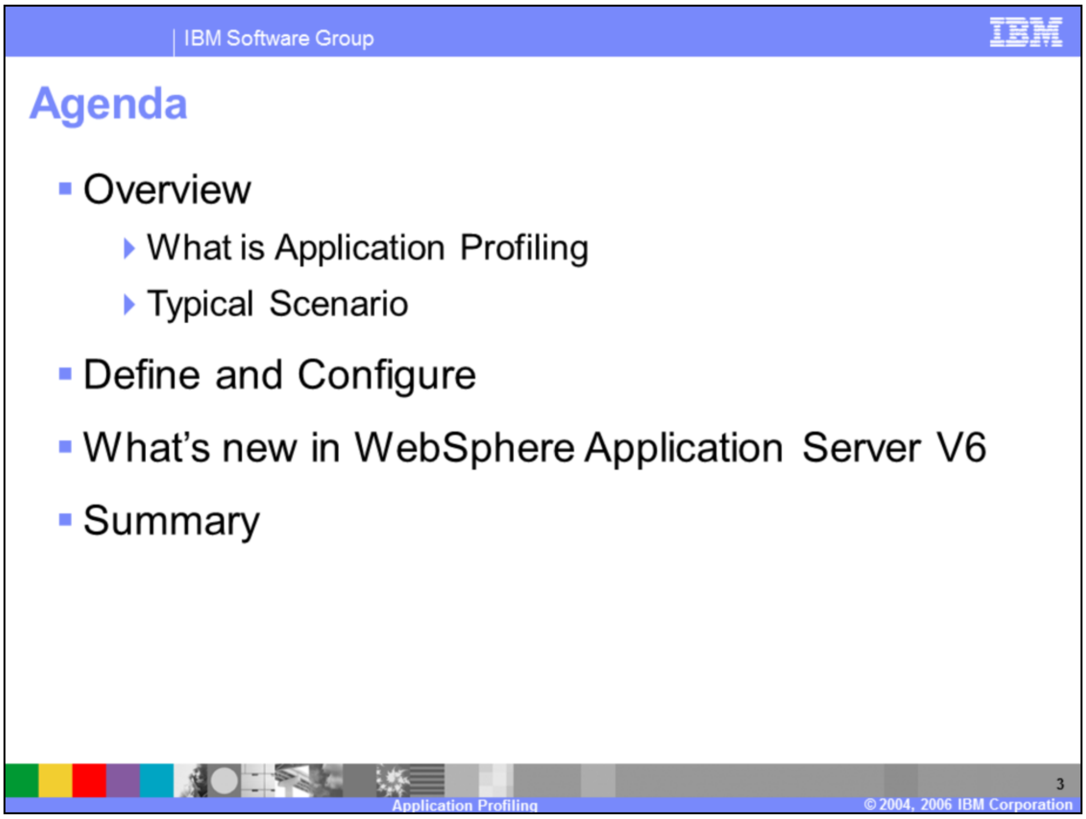
- Understand the basics of Application Profiling
- Understand which scenarios would benefit from this functionality
- Learn how to define and configure Application Profiling

2

Application Profiling

© 2004, 2006 IBM Corporation

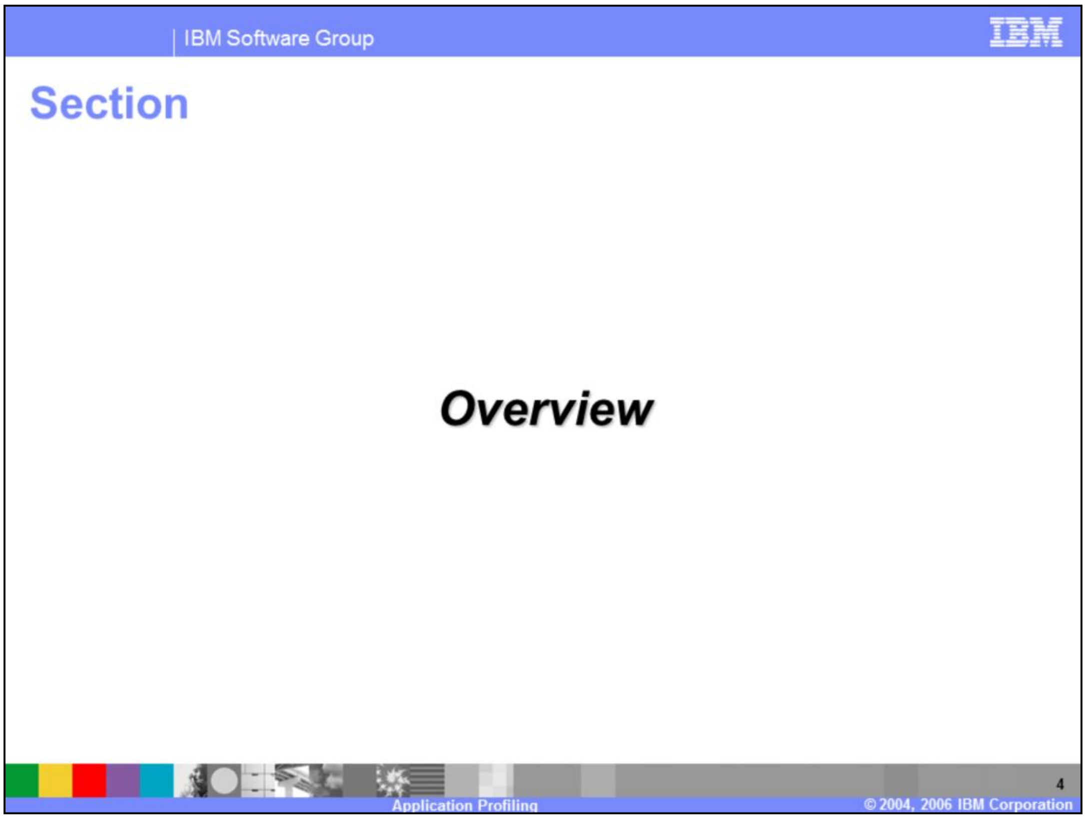
The goals of this presentation are to gain an understanding of Application Profiling, to be able to determine which scenarios could benefit from this functionality, and to become familiar with set up and configuration of Application Profiling.



Agenda

- Overview
  - What is Application Profiling
  - Typical Scenario
- Define and Configure
- What's new in WebSphere Application Server V6
- Summary

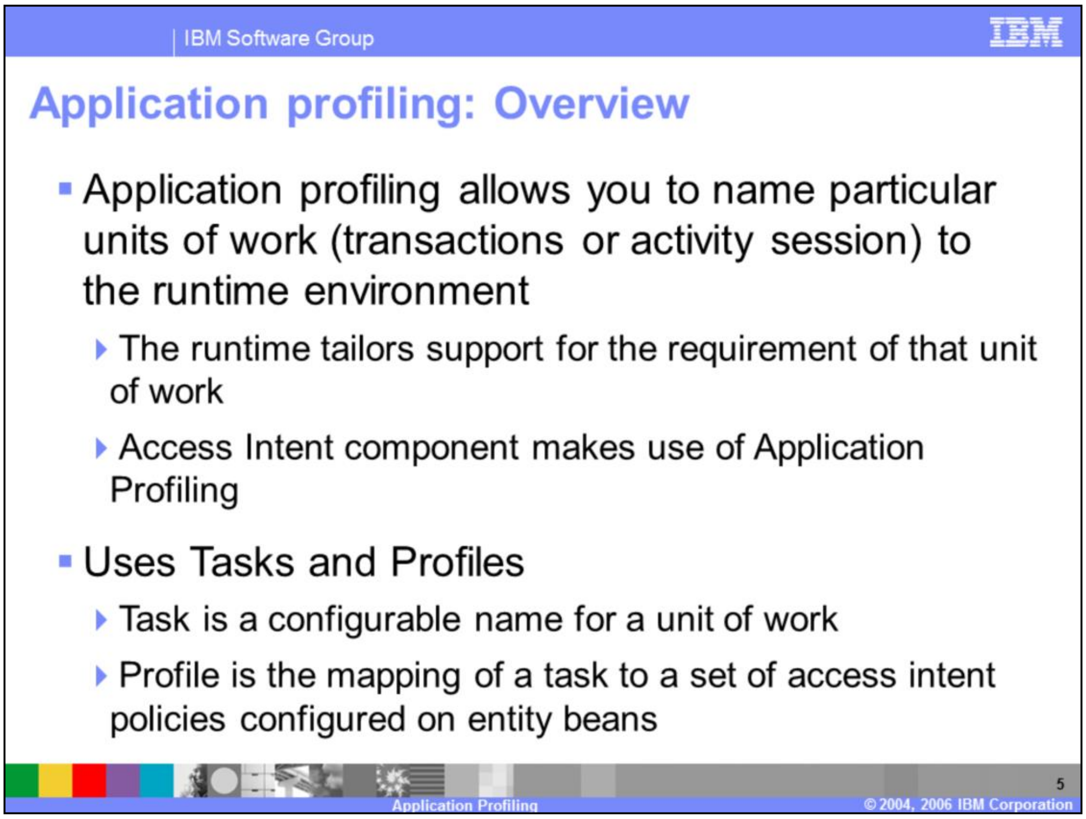
The agenda for this presentation includes an overview of Application Profiling, a brief discussion of how to define and configure the functionality, and what is new in WebSphere Application Server V6.



Section

Overview

The next section will provide an overview of Application Profiling and describe a typical scenario.



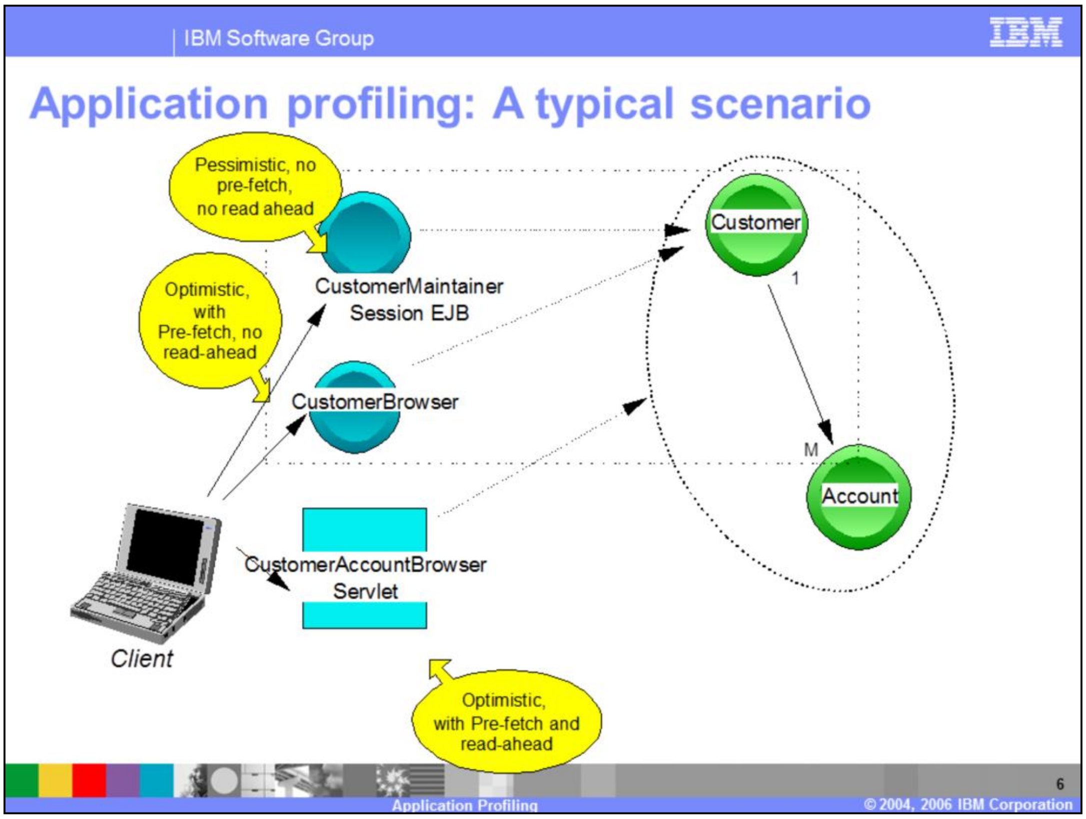
Application profiling: Overview

- Application profiling allows you to name particular units of work (transactions or activity session) to the runtime environment
  - The runtime tailors support for the requirement of that unit of work
  - Access Intent component makes use of Application Profiling
- Uses Tasks and Profiles
  - Task is a configurable name for a unit of work
  - Profile is the mapping of a task to a set of access intent policies configured on entity beans

Application Profiling provides fine-grained, application-focused Enterprise Java™ Bean (EJB) tuning capabilities. It allows you to dynamically apply access intent policies based upon the requirements of the caller. Access intent is currently the only run time component that makes use of the application profiling functionality.

Application profiling introduces two concepts in order to achieve this function: tasks and profiles.

When an invocation on a bean (whether by a finder method, a container managed relationship (CMR) getter, or a dynamic query) requires data to be retrieved from the back end system, the task of the active unit of work associated with the request is used to determine the exact requirement of the transaction. The same bean loads and behaves differently in the context of the task-to-profile mapping. Each profile provides the developer an opportunity to reconfigure the application's access intent.


IBM Software Group
IBM
Pessimistic, no pre-fetch, no read ahead
Optimistic, with Pre-fetch, no read-ahead
CustomerMaintainer Session EJB
CustomerBrowser
Customer
1
M
Account
Client
CustomerAccountBrowser Servlet
Optimistic, with Pre-fetch and read-ahead
6
Application Profiling
© 2004, 2006 IBM Corporation

Application profiling: A typical scenario

Tuning EJB applications becomes increasingly complex as EJBs are used by more than one application. Frequently, compromises must be made for the application that requires the highest isolation and locking levels when accessing data. Optimizing for one application often causes conflicts with other applications using the same EJBs.

Here, you see a typical scenario with three different clients, each with different access intent requirements to the same entity bean, Customer. Customer has a one-to-many container-managed relationship with the entity bean, Account.

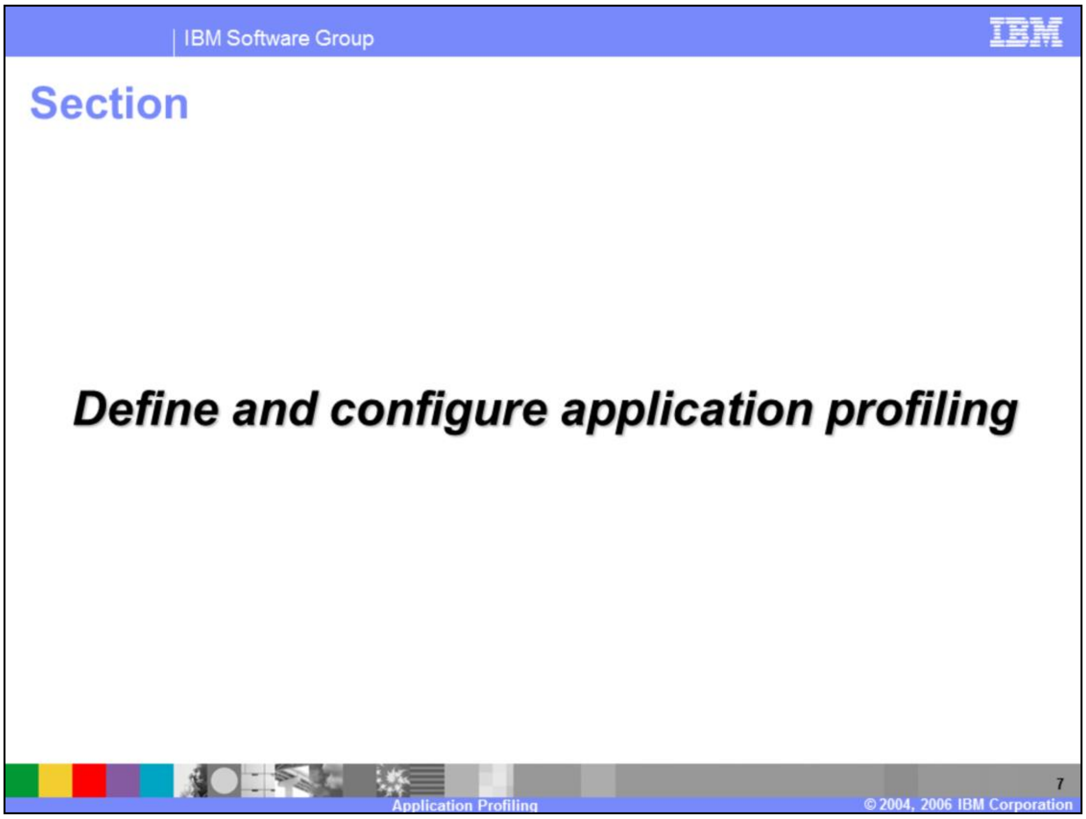
The CustomerMaintainer session bean is responsible for updating the customer application. Therefore, this client needs to access the Customer entity bean in a pessimistic manner, meaning it needs to hold a lock on the Customer object for the duration of the transaction.

The CustomerBrowser session bean is responsible for reading and presenting static data from the customer application. This client can access the Customer entity bean in an optimistic manner, not requiring it to hold locks. Here is potential for contention as both clients attempt to access the Customer object at the same time. The CustomerBrowser client may need to wait until the CustomerMaintainer client completes it’s transaction and releases it’s lock on the data.

To help optimize access to the entity beans, you can assign access intents, or “hints” to the persistence manager about how best to make decisions about isolation level, cursor management, data prefetch and caching. However these access intents are static and need to be configured for the scenario that requires the strongest level of locking. (There are two general methods for adding access intent to EJB V2 enterprise beans. There is the default method for handling access intent based on the bean level, which should be used in most cases; and there is the method-level access intent which is now deprecated in V6.0.)

The requirement in this scenario is for dynamic control over the access of the backend application. Application Profiling allows access intents to be applied more effectively by enabling the client to specify it’s access intent requirements. The container and persistence manager dynamically provide the appropriate access intent services for the caller’s request, based on the associated task defined by the Application Profile.

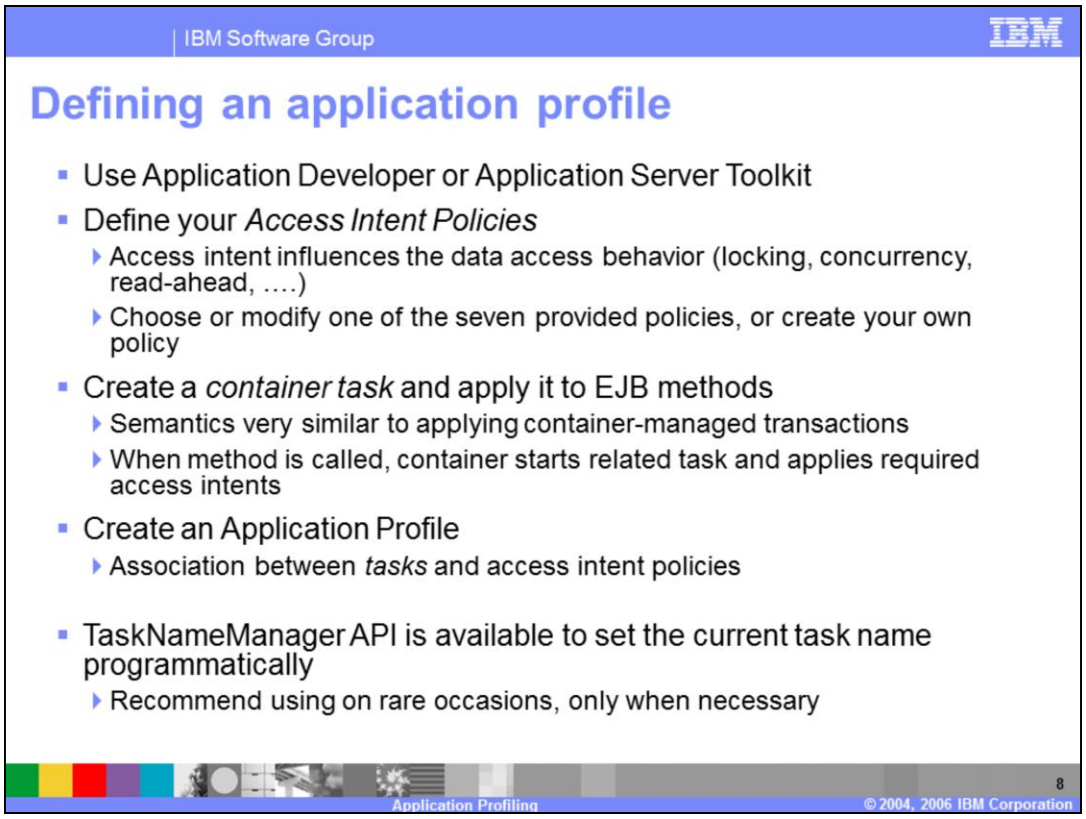
Returning to the scenario, take a look at the third client. CustomerAccountBrowser servlet is responsible for reading and displaying the full customer account portfolio, which includes account information for the customer. This client can take advantage of an optimistic, read ahead access intent. With one call to the Customer entity, you can also read ahead to the Account entity and return those account records associated with that Customer, thus avoiding the need for additional calls to the database to retrieve backend data.



Section

Define and configure application profiling

The next section will briefly describe how to define and configure Application Profiling, and touch on a few of the many different options for defining access intent.


IBM Software Group

Defining an application profile

- Use Application Developer or Application Server Toolkit
- Define your *Access Intent Policies*
  - Access intent influences the data access behavior (locking, concurrency, read-ahead, ….)
  - Choose or modify one of the seven provided policies, or create your own policy
- Create a *container task* and apply it to EJB methods
  - Semantics very similar to applying container-managed transactions
  - When method is called, container starts related task and applies required access intents
- Create an Application Profile
  - Association between *tasks* and access intent policies
- TaskNameManager API is available to set the current task name programmatically
  - Recommend using on rare occasions, only when necessary

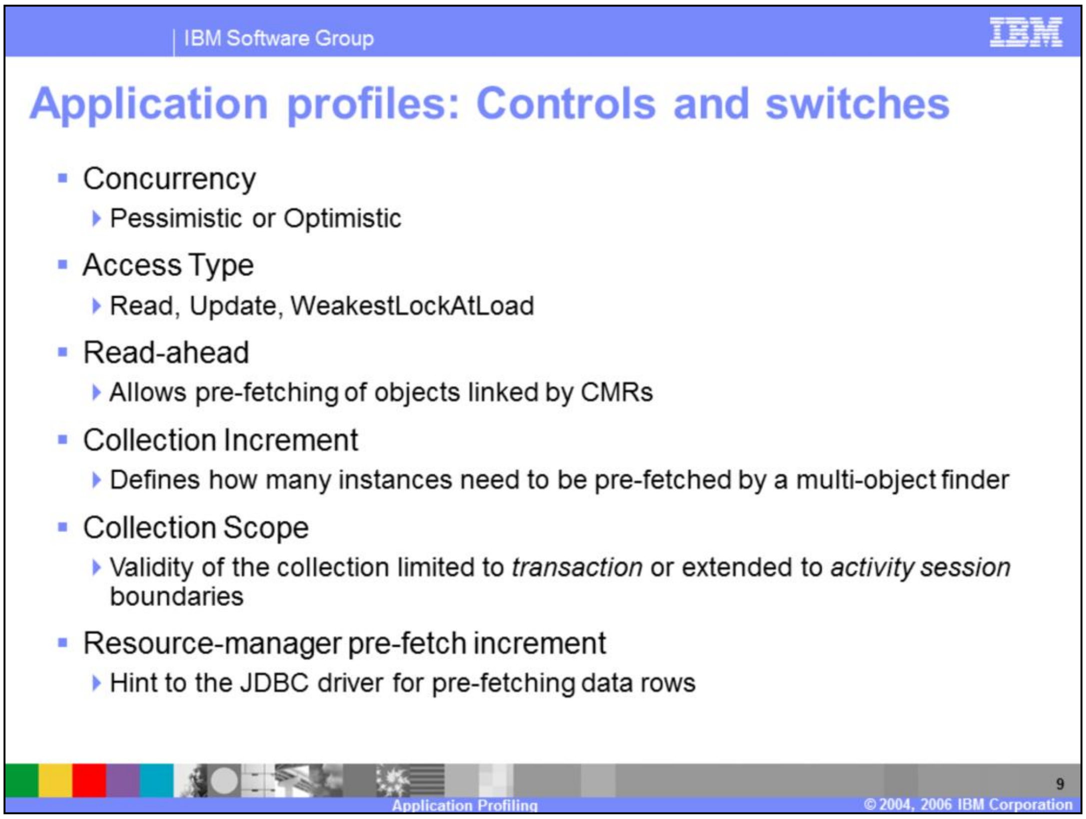
8

Application Profiling © 2004, 2006 IBM Corporation

You will use IBM Rational® Application Developer or Application Server Toolkit to define and configure Application Profiling. You may begin by defining your access intent policies. The predefined Access Intent policies include: wsPessimisticUpdate, wsOptimisticUpdate, wsOptimisticRead, wsPessimisticRead, wsPessimisticUpdate-Exclusive, wsPessimisticUpdate-NoCollision, and wsPessimisticUpdateWeakestLockAtLoad which is the default policy.

Next, create a container task, and associate the access intent policies with a task, by way of the application profile. When a method is called, if the method also begins a unit of work, the container starts the related task and applies the configured access intents. If the called method does not begin a new unit of work, any related container-managed task will be ignored. The application profile ties the task and access intent together.

On rare occasions, it may be necessary to programmatically set the current task name. Application profiling supports this requirement with the TaskNameManager interface that enables both overriding of the current task associated with the thread of execution, and resetting of the current task with the original task.


IBM Software Group

Application profiles: Controls and switches

- Concurrency
  - Pessimistic or Optimistic
- Access Type
  - Read, Update, WeakestLockAtLoad
- Read-ahead
  - Allows pre-fetching of objects linked by CMRs
- Collection Increment
  - Defines how many instances need to be pre-fetched by a multi-object finder
- Collection Scope
  - Validity of the collection limited to *transaction* or extended to *activity session* boundaries
- Resource-manager pre-fetch increment
  - Hint to the JDBC driver for pre-fetching data rows

9

Application Profiling © 2004, 2006 IBM Corporation

Application profiles influence what access intent to apply on the target bean. The access intent has the following attributes that determine how to load data for the target bean.

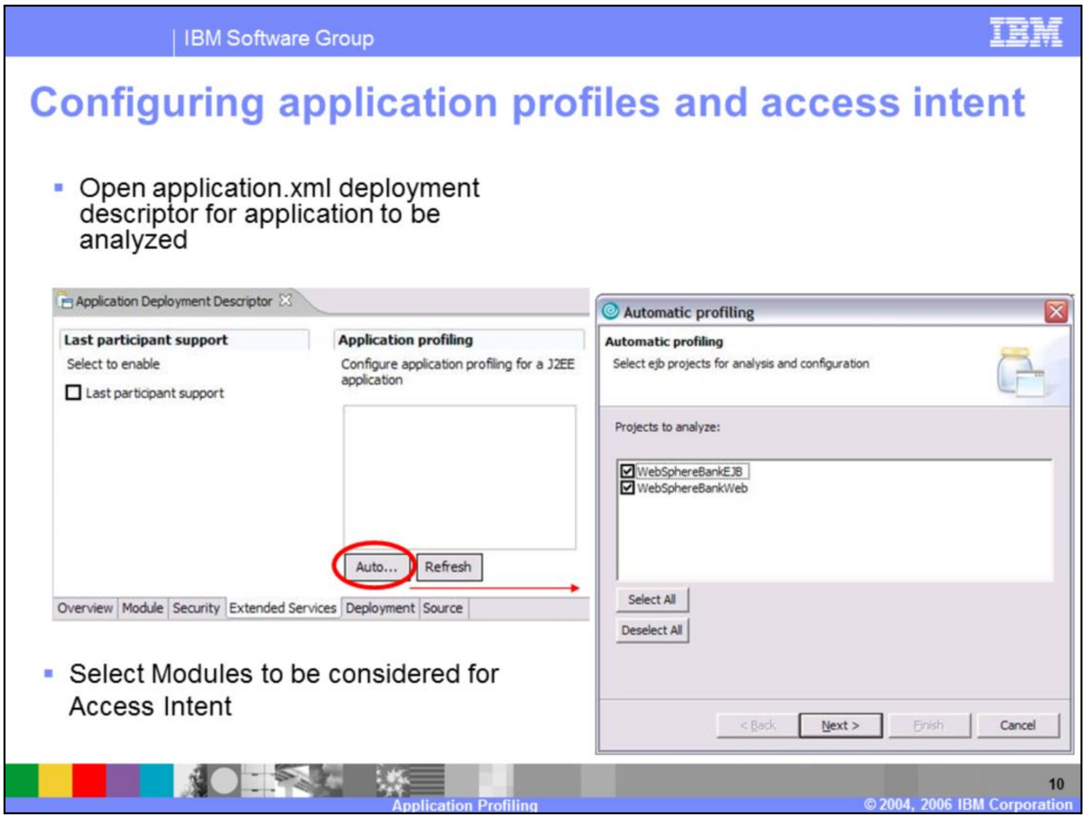
Concurrency management schemes deal with the problem of managing contention for data resources - how to allow multiple users access while maintaining accuracy and consistency of the data and preventing deadlocks. A pessimistic scheme locks a given resource at the beginning of a transaction and typically holds the lock for the duration of the transaction; while an optimistic scheme acquires a lock and typically releases it within a short period of time, after the data is read. Based upon the combination of concurrency, access type, and the different behaviors of the databases themselves, you will see different results with regards to levels of locking and duration of those locks, and isolation levels when accessing the data.

Read-ahead support is used for navigating across container-managed relationships, loading the entity bean and preloading a number of related entity beans that otherwise would have required additional calls to the database in order to retrieve the data.

Collection increment controls the number of instances that can be pre-fetched in a single operation. The number of instances loaded can have implications on your performance. If the number is set too high for your scenario, the cache may become cluttered and contention for memory with other applications and processes could adversely affect performance. If the number is set too low for your scenario, the iterator may need to wait for incoming data.

Collection Scope is used to control the lifecycle of a collection by specifying whether the collection boundary should be the transaction (the default) or extended to an activity session.

Resource-manager pre-fetch increment is a value used as a hint given to the JDBC driver. How this hint is handled is determined by the specific JDBC driver provider, whether it recognizes pre-fetch, and how pre-fetch may or may not be and implemented by the database engine.


IBM Software Group

Configuring application profiles and access intent

- Open application.xml deployment descriptor for application to be analyzed

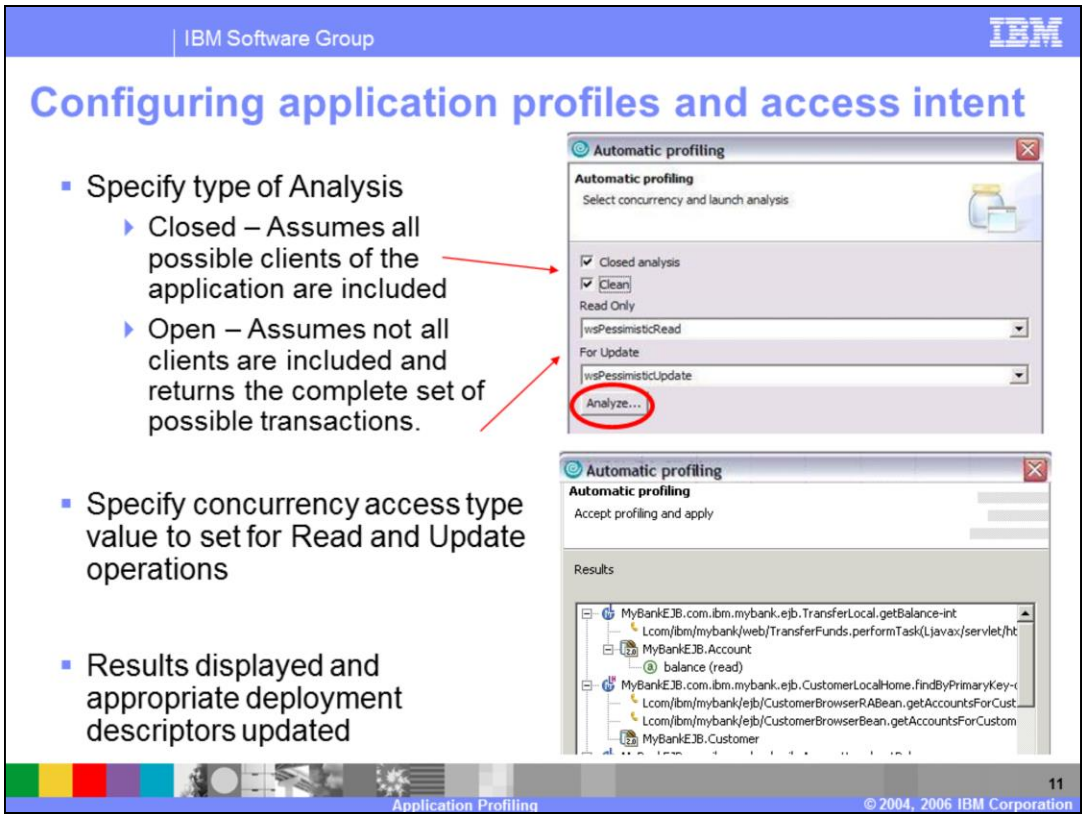
- Select Modules to be considered for Access Intent

Application Profiling 10 © 2004, 2006 IBM Corporation

It can be challenging to configure application profiles and access intents. Application Developer and the Application Server Toolkit provide an automatic application profiling analysis engine that can help with this task. The analyzer examines compiled classes and the deployment descriptors of Java™ 2 Enterprise Edition (J2EE) applications and determines the entry point of transactions, it examines the methods and determines which entities are enlisted in those transactions, and then examines the entities and determines whether those entities are read or updated during the transaction.

You can run this auto analysis wizard by opening the application deployment descriptor, selecting the Extended Services tab, then clicking on the “Auto” button from within the Application Profiling window. You will then select those EJB projects you would like included in the analysis.

The analysis engine does not support analysis when the unit of work is Activity Sessions.



IBM Software Group

Configuring application profiles and access intent

- Specify type of Analysis
  - Closed – Assumes all possible clients of the application are included
  - Open – Assumes not all clients are included and returns the complete set of possible transactions.
- Specify concurrency access type value to set for Read and Update operations
- Results displayed and appropriate deployment descriptors updated

Automatic profiling

Automatic profiling
Select concurrency and launch analysis

[✓] Closed analysis
[✓] Clean
Read Only: wsPessimisticRead
For Update: wsPessimisticUpdate

Automatic profiling

Automatic profiling
Accept profiling and apply

Results

```
MyBankEJB.com.ibm.mybank.ejb.TransferLocal.getBalance-int
    Lcom/ibm/mybank/web/TransferFunds.performTask(Ljavax/servlet/ht
  MyBankEJB.Account
      balance (read)
MyBankEJB.com.ibm.mybank.ejb.CustomerLocalHome.findByPrimaryKey-(
    Lcom/ibm/mybank/ejb/CustomerBrowserRABean.getAccountsForCust
    Lcom/ibm/mybank/ejb/CustomerBrowserBean.getAccountsForCustom
  MyBankEJB.Customer
```

11

Application Profiling © 2004, 2006 IBM Corporation

You can run the analysis in either *closed world* or *open world* mode. A closed-world analysis assumes that all possible clients of the application are included in the ear and therefore, included in the analysis, and the analysis is complete. It does not look at any other potential clients or additions. The results of a closed-world analysis report the set of all transactions that can be invoked by a web, JMS, application client, or session bean. The results do not include potential transactions that are never run.

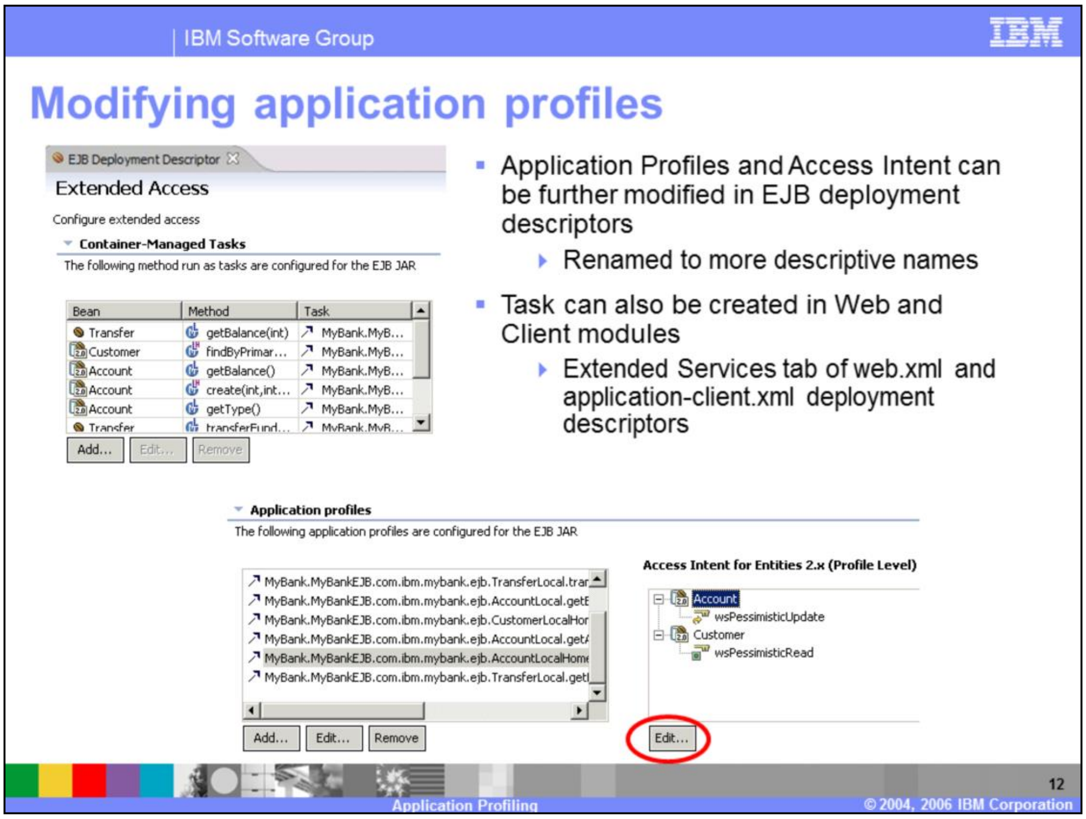
An open-world analysis assumes that not all clients are available for analysis or that the analysis cannot return complete or accurate results. An open-world analysis returns the complete set of possible transactions.

If you select the Clean attribute, the existing configuration of selected modules is removed and the new configuration is applied fresh. If you do not select this option, the new configuration is merged into the existing configuration.

Finally, you will select concurrency access type values for the defaults that should be generated. If you have a transaction that invokes only read methods on a particular object, which access type would you like, pessimistic read or optimistic read? Similarly for updates, do you want updates handled as a pessimistic update or an optimistic update?

Once analysis completes, the results will be displayed and persisted as an application profiling configuration. Container-managed tasks will be created for clients, and application profiles are constructed with the appropriate access intent for the entities in the transaction represented by the task.

You should examine the results of the analysis very carefully. In many cases you must manually modify them to meet the requirements of the application. However, the tool can be an effective starting place for most applications and may offer a quick configuration of application profiles for some applications.



IBM Software Group

Modifying application profiles

EJB Deployment Descriptor

Extended Access

Configure extended access

Container-Managed Tasks

The following method run as tasks are configured for the EJB JAR

Bean	Method	Task
Transfer	getBalance(int)	MyBank.MyB...
Customer	findByPrimar...	MyBank.MyB...
Account	getBalance()	MyBank.MyB...
Account	create(int,int...	MyBank.MyB...
Account	getType()	MyBank.MyB...
Transfer	transferFund...	MyBank.MyB...

Application profiles

The following application profiles are configured for the EJB JAR

```
MyBank.MyBankEJB.com.ibm.mybank.ejb.TransferLocal.trar
MyBank.MyBankEJB.com.ibm.mybank.ejb.AccountLocal.getE
MyBank.MyBankEJB.com.ibm.mybank.ejb.CustomerLocalHor
MyBank.MyBankEJB.com.ibm.mybank.ejb.AccountLocal.getA
MyBank.MyBankEJB.com.ibm.mybank.ejb.AccountLocalHome
MyBank.MyBankEJB.com.ibm.mybank.ejb.TransferLocal.getI
```

Access Intent for Entities 2.x (Profile Level)

```
Account
    wsPessimisticUpdate
Customer
    wsPessimisticRead
```

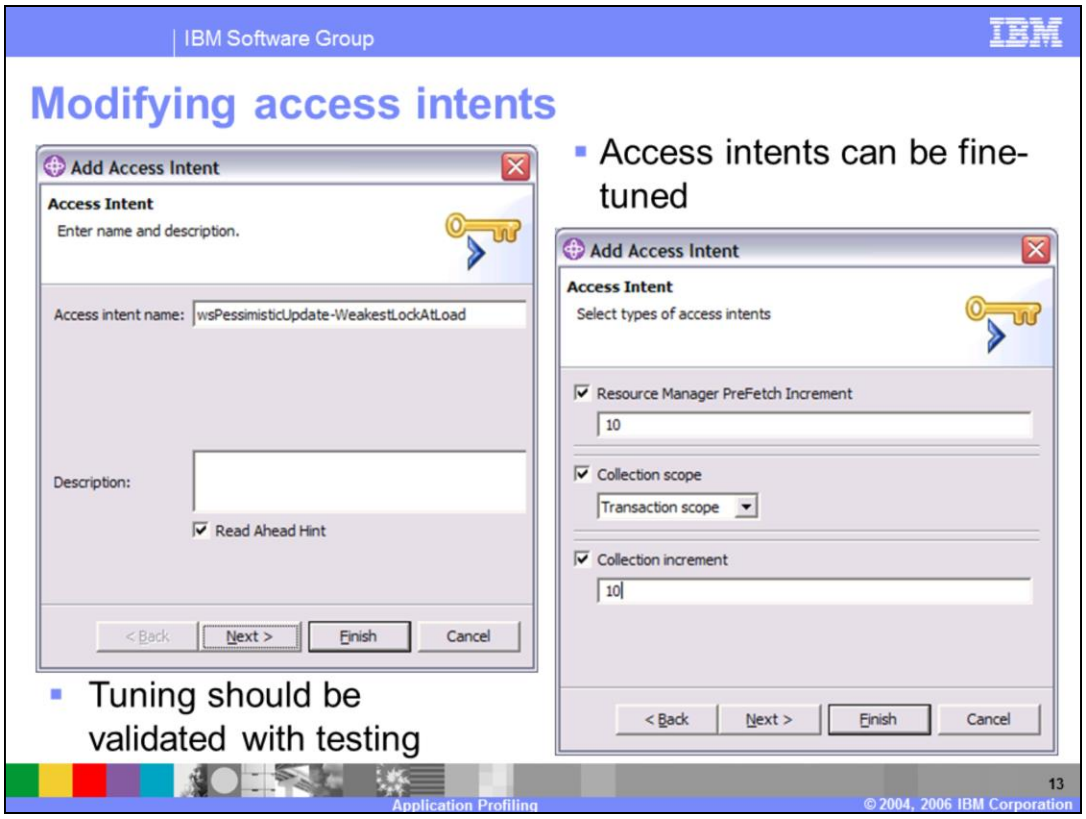
- Application Profiles and Access Intent can be further modified in EJB deployment descriptors
  - Renamed to more descriptive names
- Task can also be created in Web and Client modules
  - Extended Services tab of web.xml and application-client.xml deployment descriptors

12

Application Profiling

© 2004, 2006 IBM Corporation

As recommended, you should review the configurations and settings created by the analysis. For example, obtain a baseline test of your application, incorporate the results of the analysis, retest, and make necessary adjustments. To review the configuration, open the EJB deployment descriptor with the Deployment Descriptor Editor and select the Extended Access tab.



IBM Software Group

Modifying access intents

- Access intents can be fine-tuned
- Tuning should be validated with testing

Add Access Intent

Access Intent

Enter name and description.

Access intent name: wsPessimisticUpdate-WeakestLockAtLoad

Description:

☑ Read Ahead Hint

Add Access Intent

Access Intent

Select types of access intents

☑ Resource Manager PreFetch Increment: 10

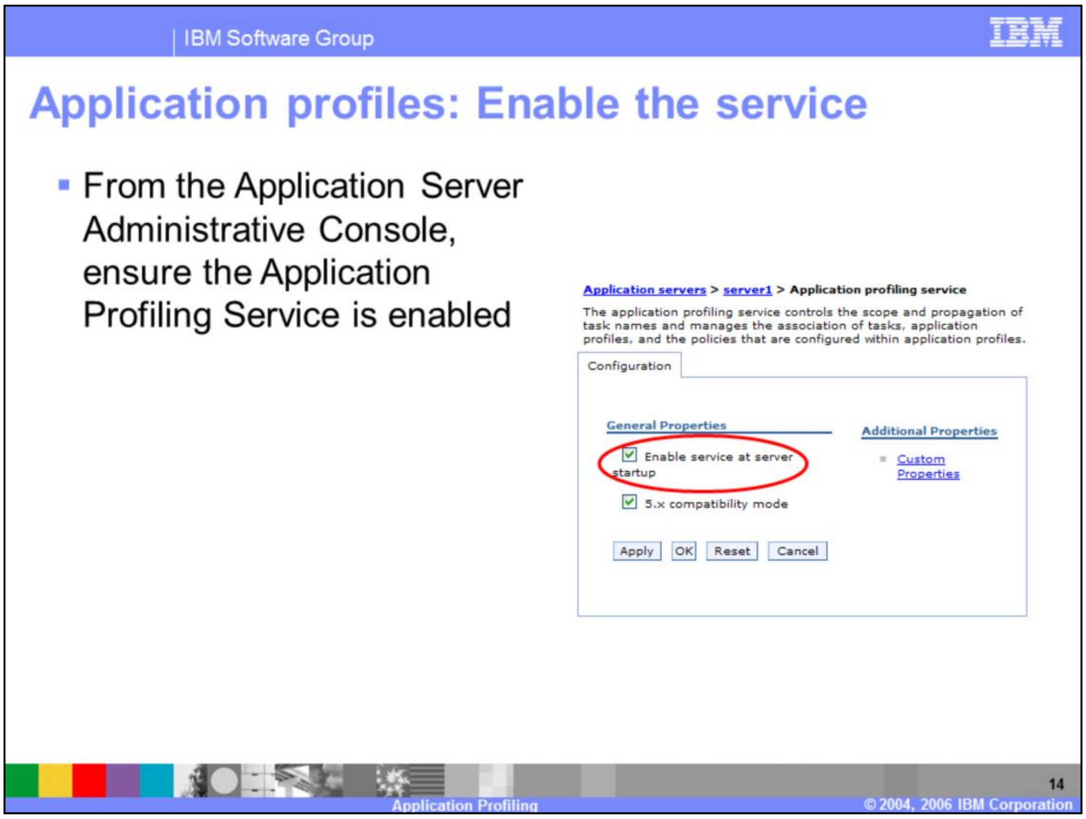
☑ Collection scope: Transaction scope

☑ Collection increment: 10

13

Application Profiling © 2004, 2006 IBM Corporation

Access Intent values can be also be modified to improve performance. Once again, all changes should be tested to insure the intended results. Some settings can have adverse affects on performance.



IBM Software Group

Application profiles: Enable the service

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

Application servers > server1 > Application profiling service

The application profiling service controls the scope and propagation of task names and manages the association of tasks, application profiles, and the policies that are configured within application profiles.

Configuration

General Properties

- [✓] Enable service at server startup
- [✓] 5.x compatibility mode

Apply | OK | Reset | Cancel

Additional Properties

- Custom Properties

14

Application Profiling

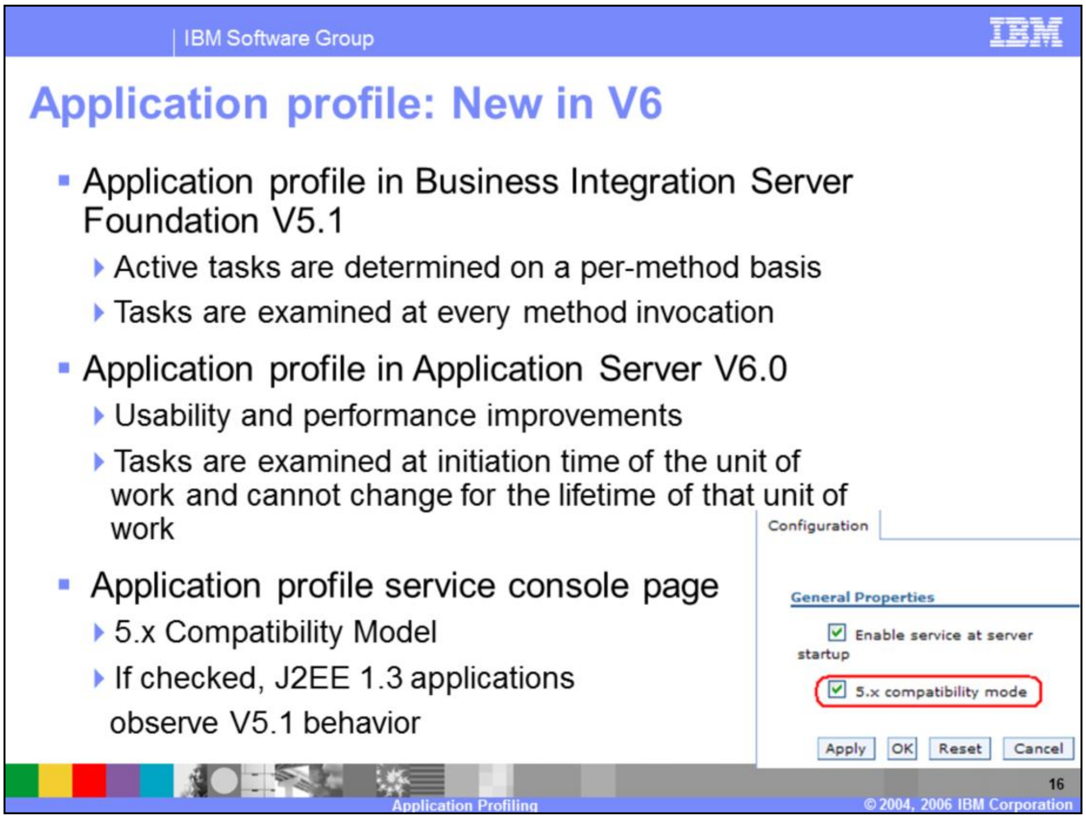
© 2004, 2006 IBM Corporation

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

Section

New in V6

The next section highlights the WebSphere Application Server V6 changes in Application Profiling.


IBM Software Group

Application profile: New in V6

- Application profile in Business Integration Server Foundation V5.1
  - Active tasks are determined on a per-method basis
  - Tasks are examined at every method invocation
- Application profile in Application Server V6.0
  - Usability and performance improvements
  - Tasks are examined at initiation time of the unit of work and cannot change for the lifetime of that unit of work
- Application profile service console page
  - 5.x Compatibility Model
  - If checked, J2EE 1.3 applications observe V5.1 behavior

Configuration

General Properties

☑ Enable service at server startup

☑ 5.x compatibility mode

16

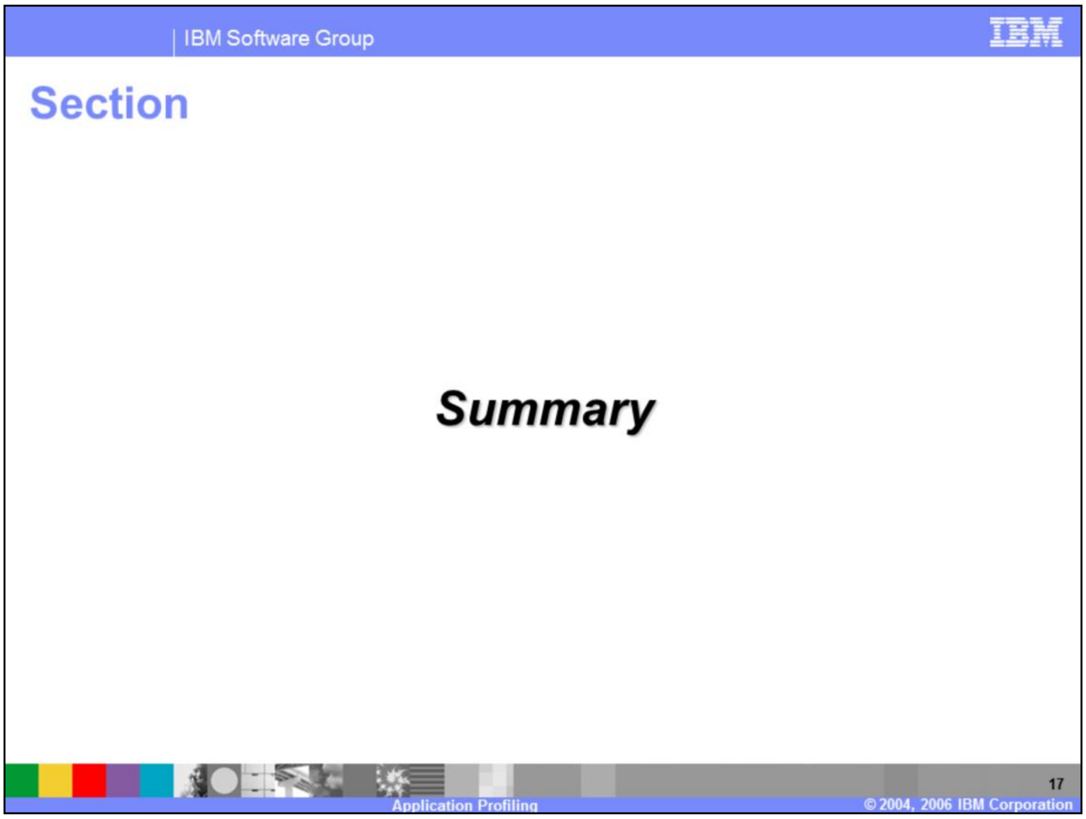
Application Profiling

© 2004, 2006 IBM Corporation

With Application Profile in WebSphere Business Integration Server Foundation 5.1, active tasks are determined on a per-method basis. When an invocation on a bean requires data to be retrieved from the backend system, the container examines the context to see if there is an active task associated. If so, the current task associated with the request is used to determine the exact requirement of the transaction. This can lead to unexpected deadlocks during database access as that task is not necessarily associated with a transaction and could be arbitrarily applied and overridden. This could also have an affect on performance as each method is examined.

With Application Profile in Application Server 6.0, there are usability and performance improvements. The concept of unit of work is incorporated. Any unit of work (transaction or activity session) that begins in the scope of a configured task is associated with that task name. A unit of work can only be named when it is initiated, and the name cannot change for the lifetime of that unit of work. A unit of work ignores any subsequent task name configurations at any point after it has begun. The task is used for the duration of its unit of work to identify configured policies specific to that unit of work. Active tasks are determined only at the initiation of a unit of work. This helps to avoid unexpected deadlocks during database access that may have occurred in previous versions. This also helps improve performance, as the check for an active task is done only at initiation of a unit of work rather than on a per-method basis.

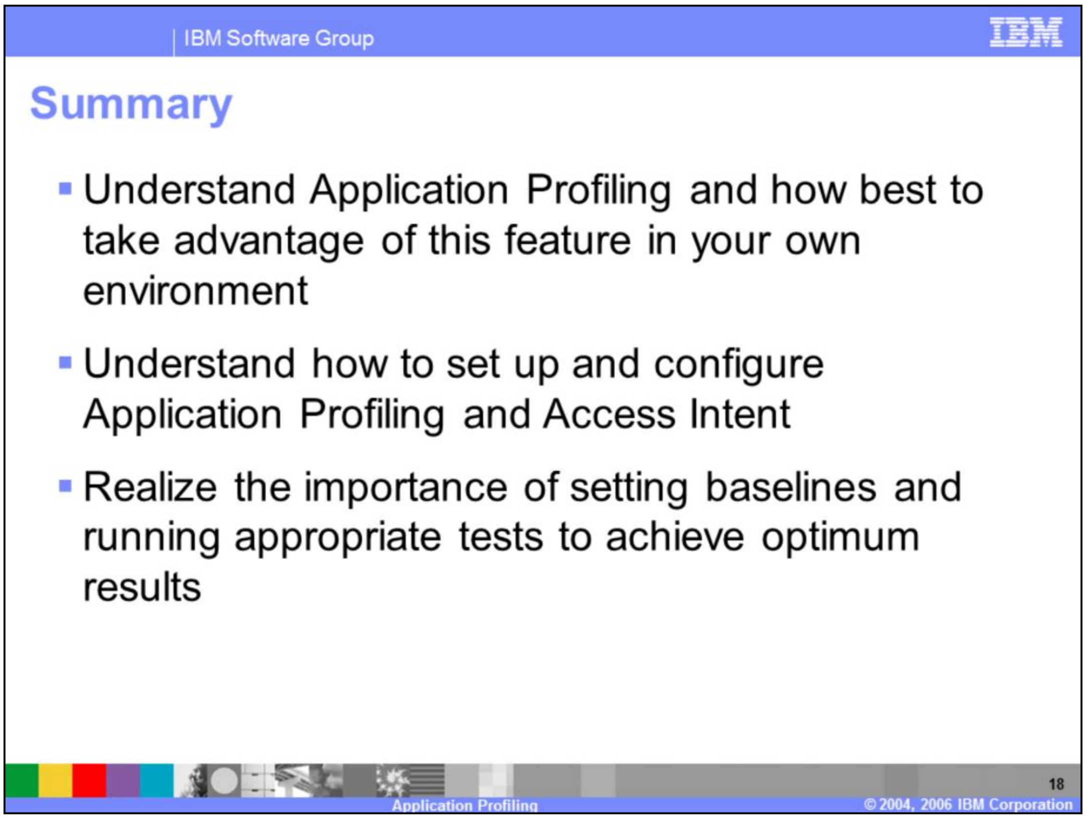
In the runtime, there is a new “compatibility” attribute in the Administrative Console that is checked by default. J2EE 1.4 applications configured with Application Profiling will automatically observe the new version 6.0 behavior; however, the behavior of J2EE 1.3 applications configured with Application Profiling is determined by this attribute. If checked, the J2EE 1.3 application will continue to use the version 5.1 behavior. For J2EE 1.3 applications, you may want to unchecked the compatibility attribute, and test the application with the new behavior. You may find you experience better results.



Section

Summary

This section will summarize the Application Profiling presentation.



IBM Software Group

Summary

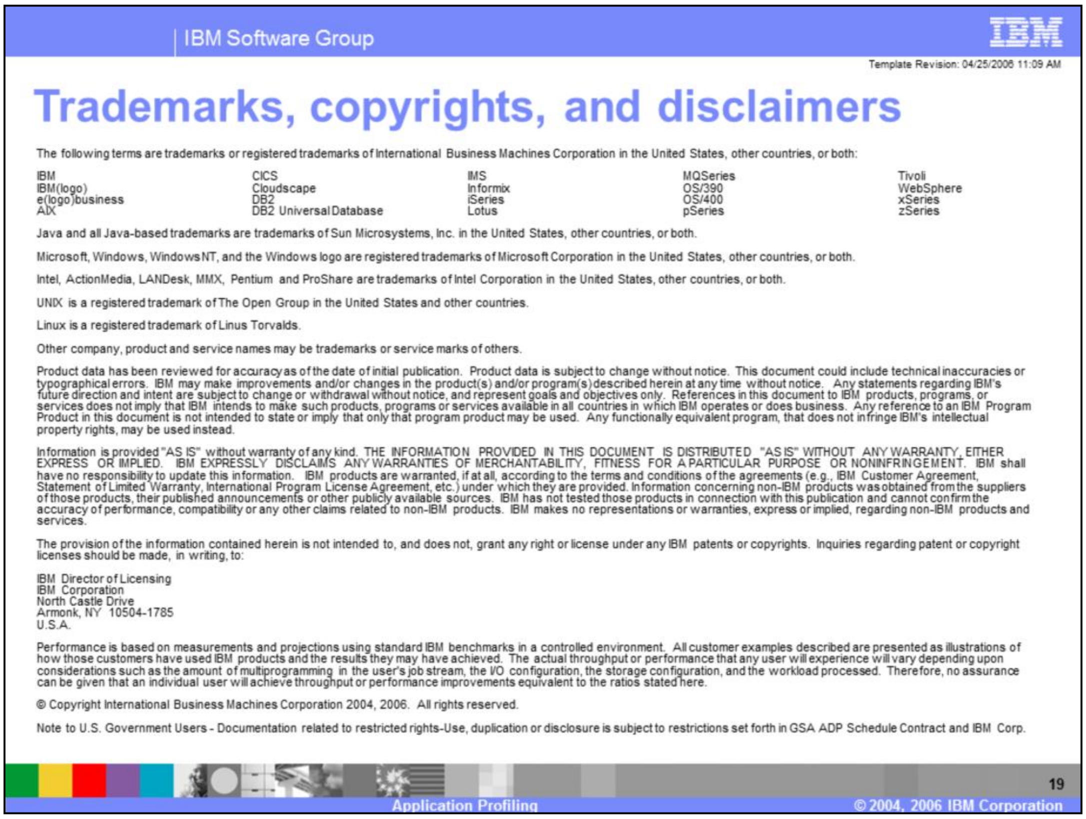
- Understand Application Profiling and how best to take advantage of this feature in your own environment
- Understand how to set up and configure Application Profiling and Access Intent
- Realize the importance of setting baselines and running appropriate tests to achieve optimum results

Application Profiling

18

© 2004, 2006 IBM Corporation

In summary, you now understand the Application Profiling feature and will be able to begin to analyze your own applications to determine how to best take advantage of this powerful function. You've also seen how to set up and configure Application Profiling and have an understanding of the various options available to you to fine-tune your EJB applications. Also emphasized is the importance of setting baseline tests for your applications and running appropriate test cycles to determine how to achieve optimum performance with regards to your unique environment.



Template Revision: 04/25/2006 11:09 AM

Trademarks, copyrights, and disclaimers