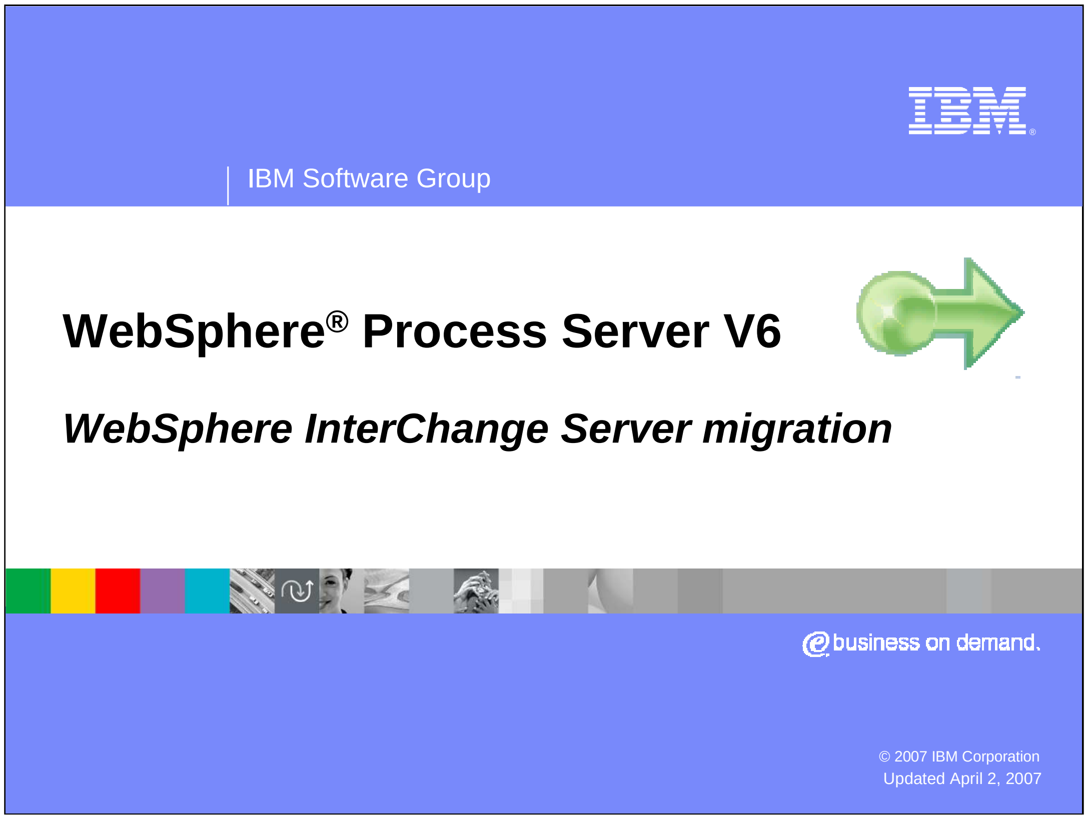
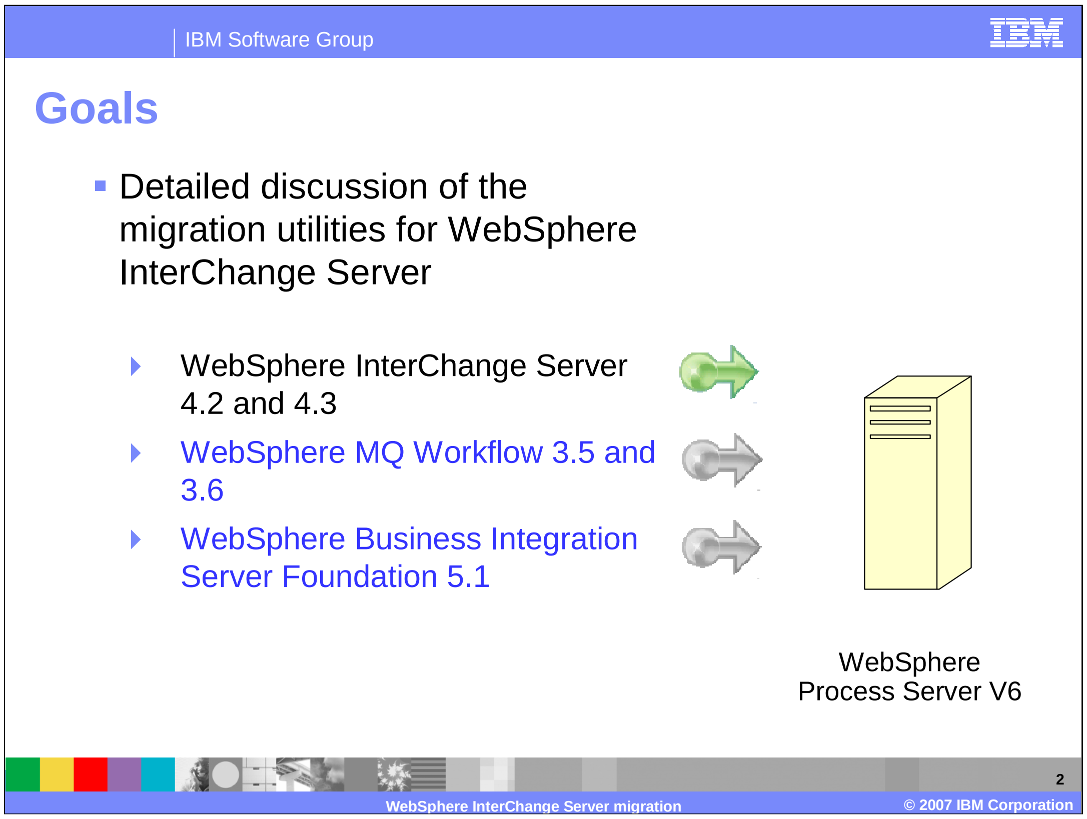


IBM Software Group

WebSphere® Process Server V6

WebSphere InterChange Server migration

This presentation will cover migrating to WebSphere Process Server V6.0 from WebSphere InterChange Server.



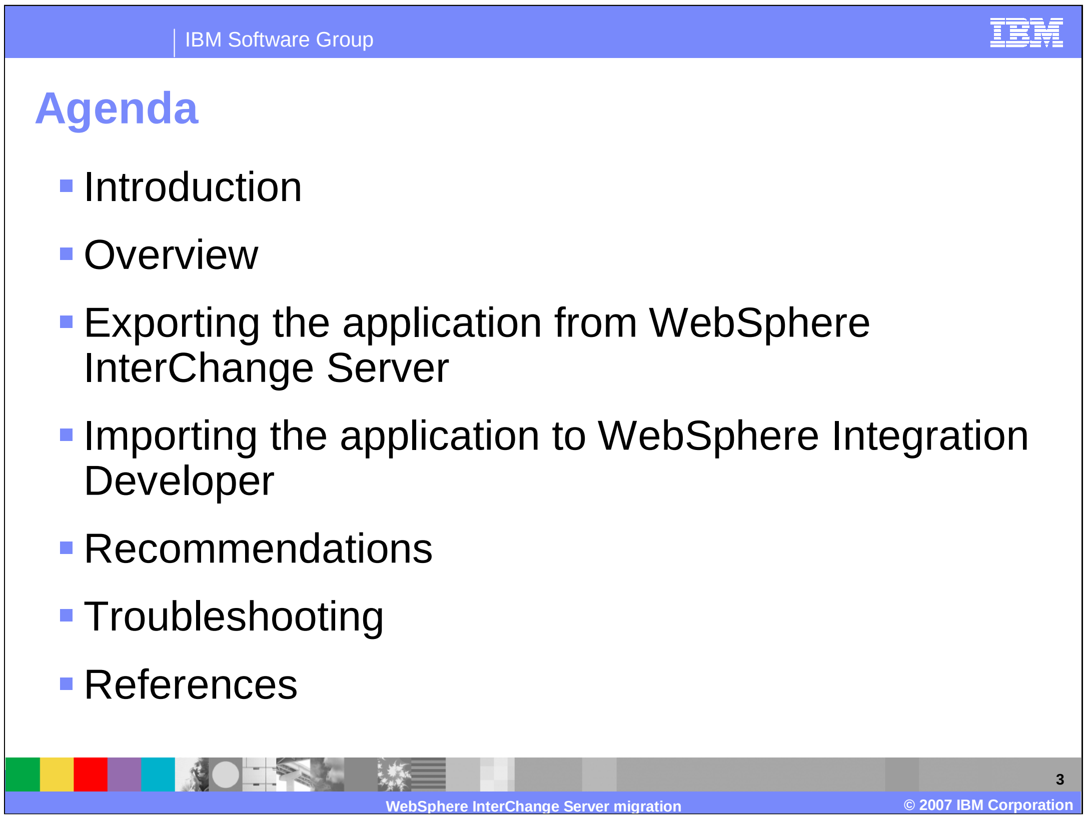
Goals

- Detailed discussion of the migration utilities for WebSphere InterChange Server
  - WebSphere InterChange Server 4.2 and 4.3
  - WebSphere MQ Workflow 3.5 and 3.6
  - WebSphere Business Integration Server Foundation 5.1

WebSphere
Process Server V6

WebSphere Process Server is the merger of three existing product lines: the WebSphere InterChange Server, the WebSphere MQ Workflow and WebSphere Business Integration Server Foundation.

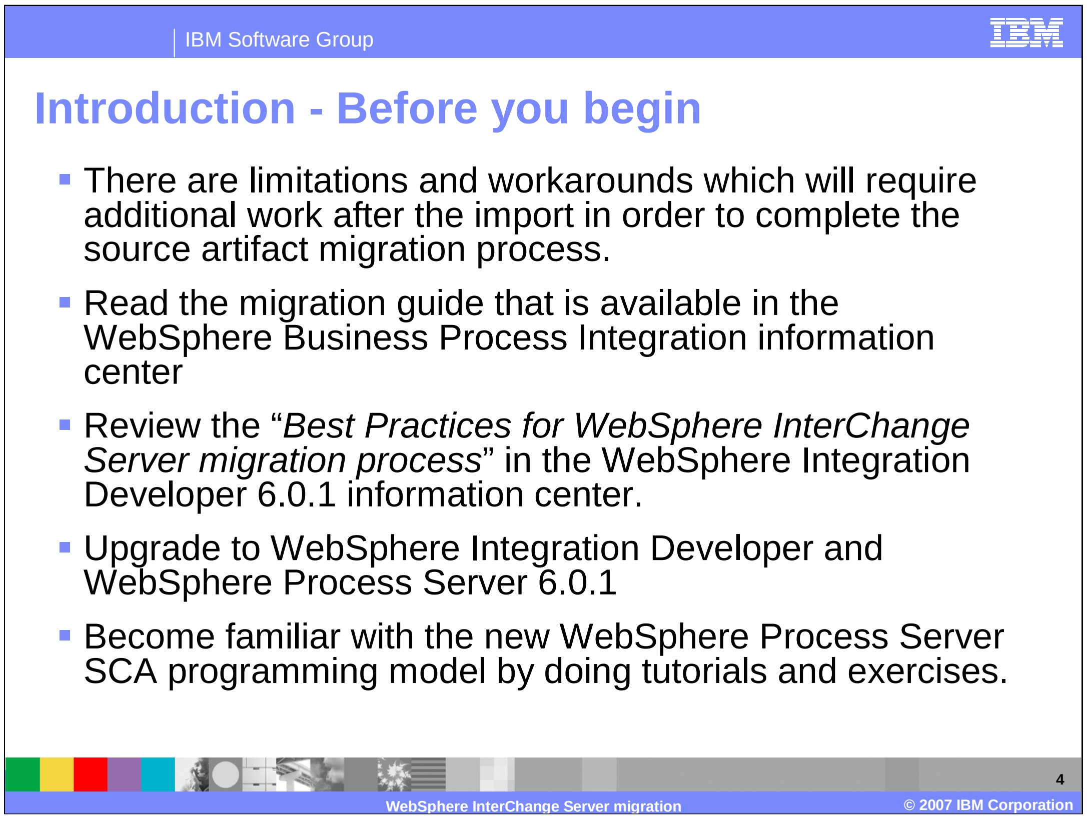
This presentation will discuss the details of the WebSphere InterChange Server migration.



Agenda

- Introduction
- Overview
- Exporting the application from WebSphere InterChange Server
- Importing the application to WebSphere Integration Developer
- Recommendations
- Troubleshooting
- References

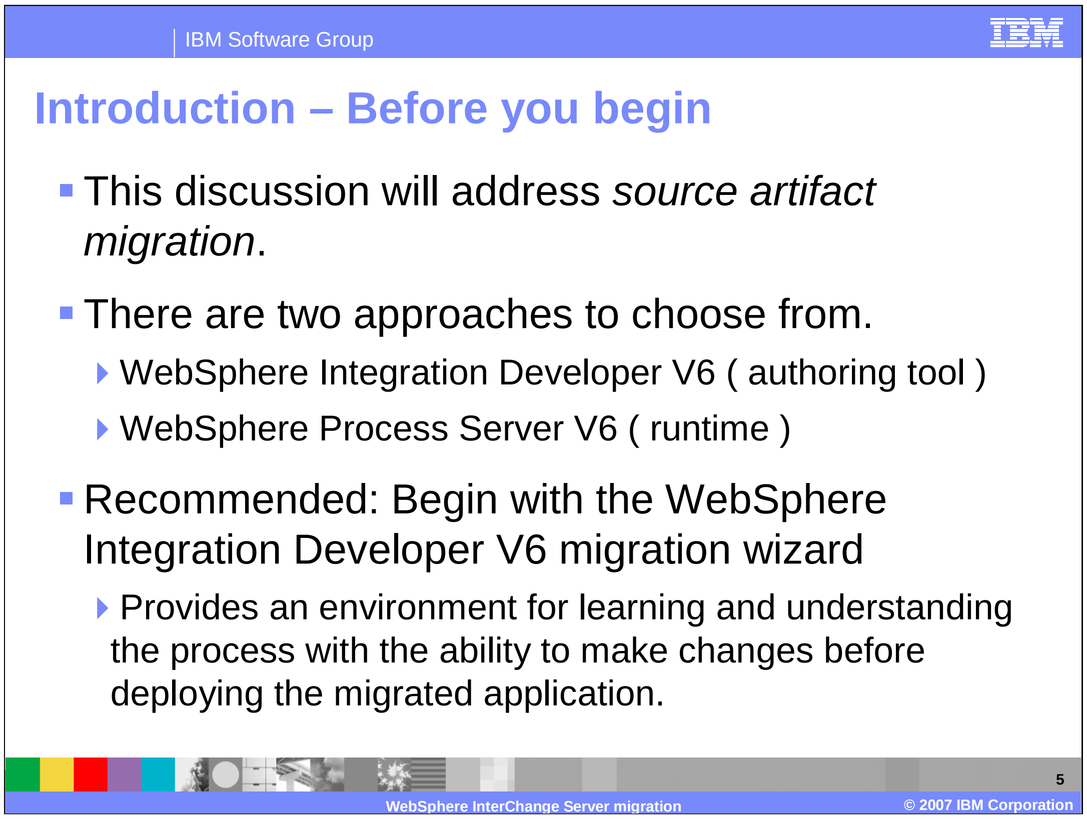
The agenda for this presentation is to focus on the steps involved in migrating from WebSphere InterChange Server to WebSphere Process Server V6.



Introduction - Before you begin

- There are limitations and workarounds which will require additional work after the import in order to complete the source artifact migration process.
- Read the migration guide that is available in the WebSphere Business Process Integration information center
- Review the "*Best Practices for WebSphere InterChange Server migration process*" in the WebSphere Integration Developer 6.0.1 information center.
- Upgrade to WebSphere Integration Developer and WebSphere Process Server 6.0.1
- Become familiar with the new WebSphere Process Server SCA programming model by doing tutorials and exercises.

Before you begin you should become familiar with the known limitations, which are listed at the end of this presentation. You should read the Migration Guide and review the Best Practices in the WebSphere Integration Developer 6.0.1 Information Center. You should also understand the guides to upgrade to WebSphere Integration Developer/WebSphere Process Server 6.0.1 and become familiar with the SCA programming model.



Introduction – Before you begin

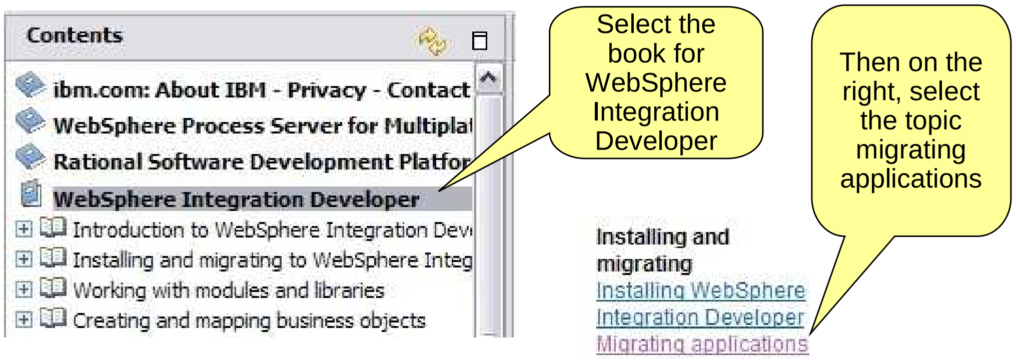
- This discussion will address *source artifact migration*.
- There are two approaches to choose from.
  - WebSphere Integration Developer V6 (authoring tool)
  - WebSphere Process Server V6 (runtime)
- Recommended: Begin with the WebSphere Integration Developer V6 migration wizard
  - Provides an environment for learning and understanding the process with the ability to make changes before deploying the migrated application.

Migration utilities are provided by both the WebSphere Integration Developer authoring tool and the WebSphere Process Server runtime. The recommended approach is to do the initial migrations using WebSphere Integration Developer, and as the migration process is understood and refined, automate it using the runtime command line tools, reposMigrate, ANT and WSADMIN.

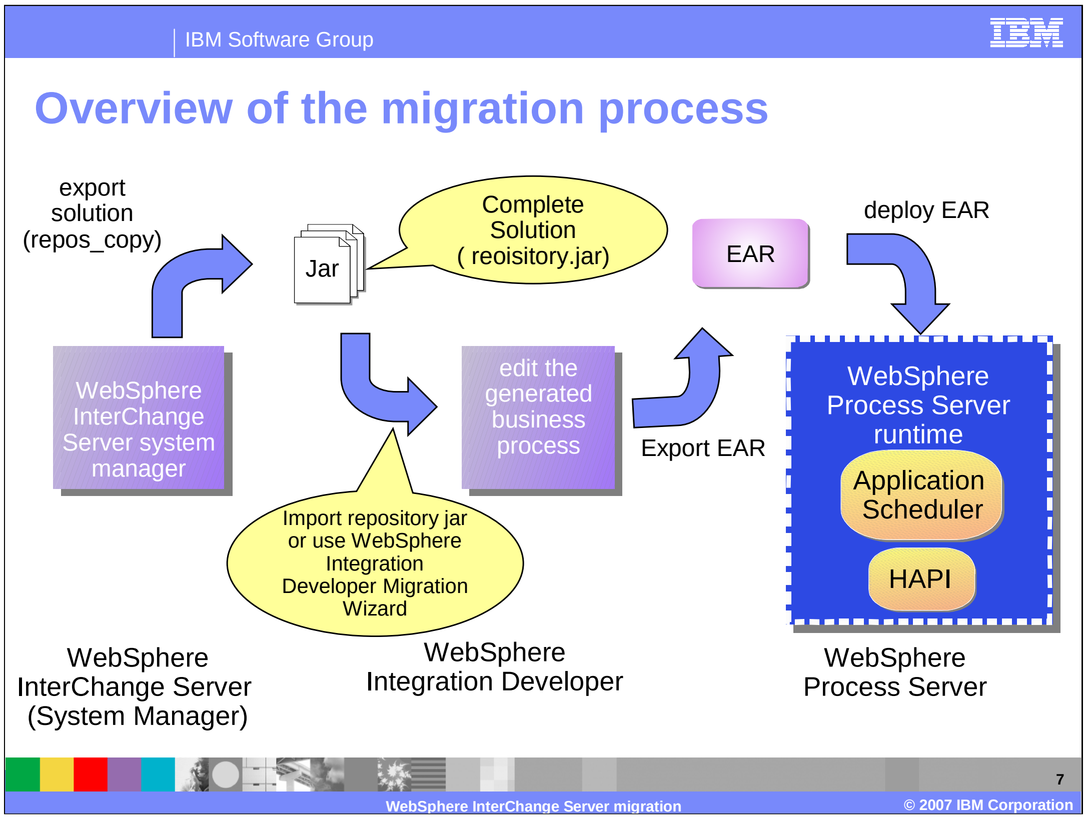
The information center

- Information center

http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp


Contents
ibm.com: About IBM - Privacy - Contact
WebSphere Process Server for Multiplat
Rational Software Development Platfor
WebSphere Integration Developer
Introduction to WebSphere Integration Devi
Installing and migrating to WebSphere Integ
Working with modules and libraries
Creating and mapping business objects
Select the book for WebSphere Integration Developer
Then on the right, select the topic migrating applications
Installing and migrating
Installing WebSphere Integration Developer
Migrating applications

The information center is an excellent place to get additional details about the migration process. A PDF version of the migration guide is also available from the welcome page of WebSphere Integration Developer V6.


IBM Software Group
Overview of the migration process
export solution (repos_copy)
Jar
Complete Solution (reoisitory.jar)
EAR
deploy EAR
WebSphere InterChange Server system manager
edit the generated business process
Export EAR
WebSphere Process Server runtime
Application Scheduler
HAPI
Import repository jar or use WebSphere Integration Developer Migration Wizard
WebSphere InterChange Server (System Manager)
WebSphere Integration Developer
WebSphere Process Server
7
WebSphere InterChange Server migration
© 2007 IBM Corporation

From a high level perspective, the migration process consists of several steps: export from the source system, import to the target system, edit, resolve errors and tune the migrated artifacts for BPEL / SCA and deploy the application to the WebSphere Process Server V6 runtime.

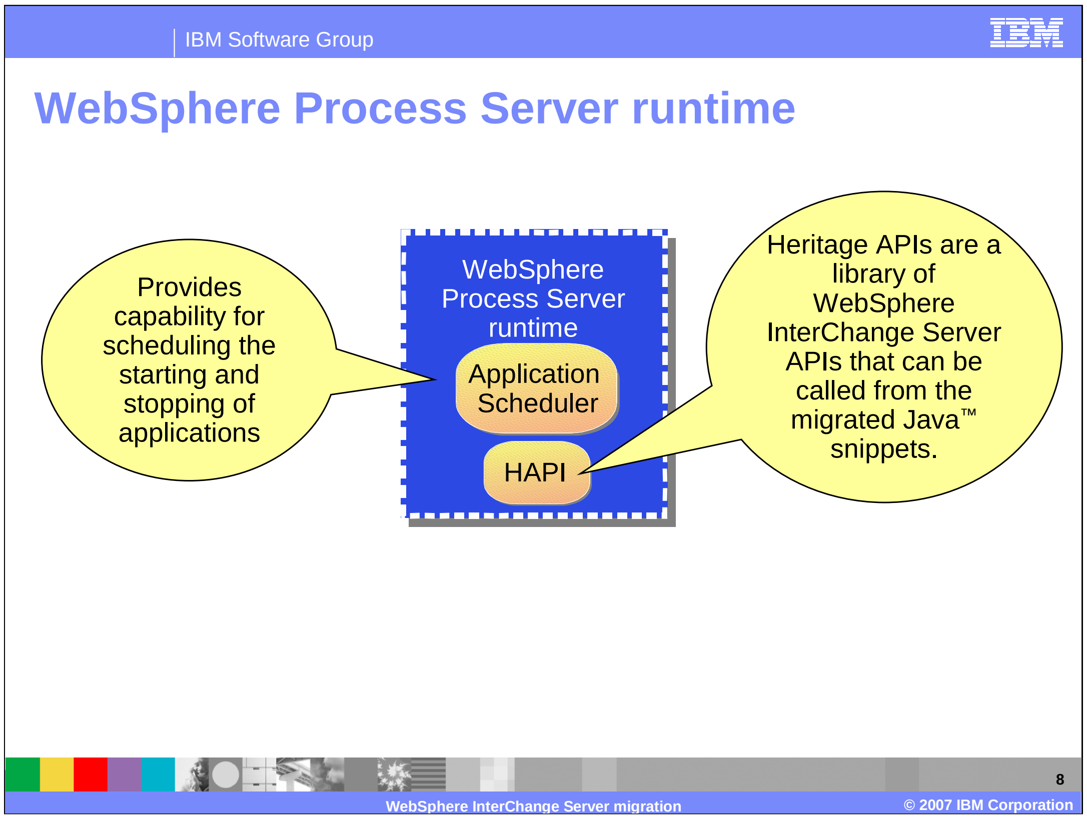
Begin by exporting the complete WebSphere InterChange Server solution from the WebSphere InterChange Server environment using the WebSphere InterChange Server system manager or the repos_copy command. This requires a complete solution, meaning that all the dependent artifacts are in the repository jar.

Then import the repository jar into WebSphere Integration Developer V6. To import the repository jar into WebSphere Integration Developer, use either the migration wizard or import the jar directly using the File -> Import menu.

The import utility will convert the WebSphere InterChange Server artifacts to WebSphere Process Server V6 artifacts and create the appropriate modules as needed. The conversion will be described in subsequent slides. (**Note** that the Wizard will call the import utility transparently.)

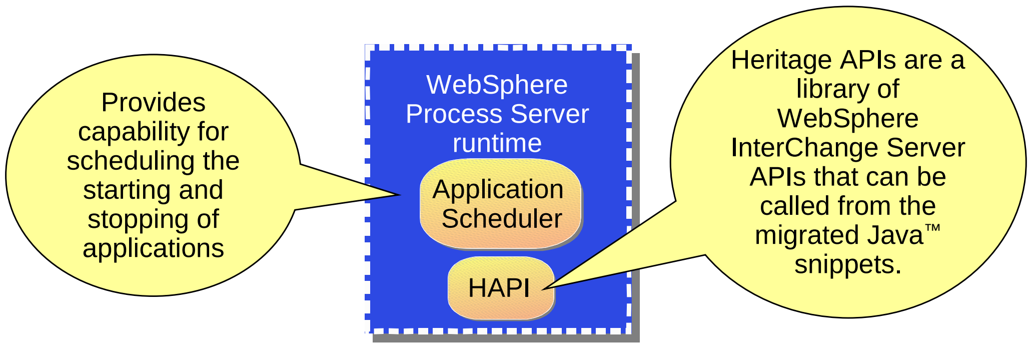
Once the WebSphere Process Server V6 source artifacts have been created in WebSphere Integration Developer, it will be necessary to review the artifacts. Based on the errors identified and the known limitations, edit the business processes to eliminate any errors.

Once the editing is complete, export the EAR and deploy it to the WebSphere Process Server V6 runtime.



IBM Software Group

WebSphere Process Server runtime


Provides capability for scheduling the starting and stopping of applications
WebSphere Process Server runtime
Application Scheduler
HAPI
Heritage APIs are a library of WebSphere InterChange Server APIs that can be called from the migrated Java™ snippets.

8

WebSphere InterChange Server migration © 2007 IBM Corporation

There are two new components in the WebSphere Process Server V6 runtime introduced specifically to support WebSphere InterChange Server migration. The **application scheduler** lets you schedule starting and stopping of applications. Refer to the WebSphere Process Server for Multi-platforms Information Center and search on "scheduler" for more information.

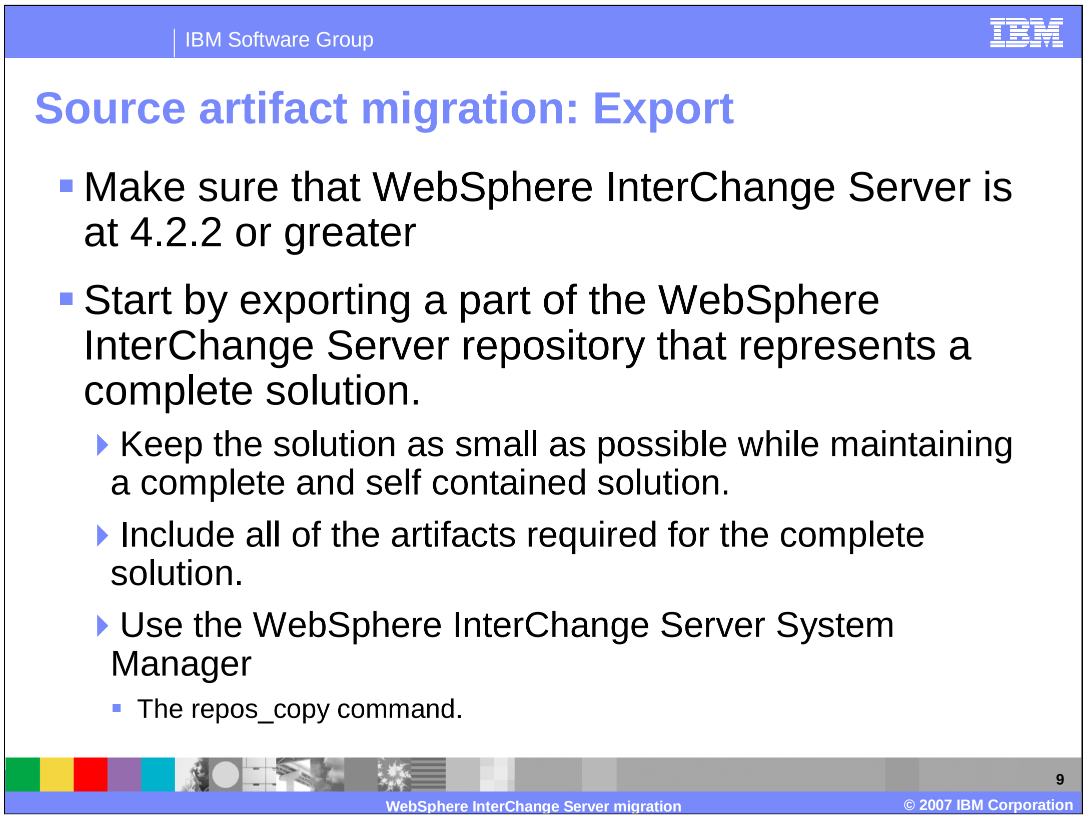
You can interact with the application scheduler using the WebSphere Process Server V6 administrative console. Additionally, you can generate scheduler entries during the migration of a WebSphere InterChange Server repository that includes WebSphere InterChange Server scheduler entries. Use the application scheduler panel in the administrative console to administer these migrated scheduler entries as well.

In a Network Deployment environment, the Application Scheduler is automatically installed for every managed server and cluster member created - no additional action is needed.

In a stand-alone server environment, the Application Scheduler is optional. While creating the stand-alone server profile, you select a check box to configure and install Application Scheduler on that server.

The **Heritage APIs** are provided to facilitate the migration process. These APIs are provided only to support migrated WebSphere InterChange Server applications until they can be modified to use the new Process Server APIs. The WebSphere InterChange Server APIs are all deprecated.

Refer to the WebSphere Integration Developer V6.0.1 information center for a detailed list of the heritage APIs.


IBM Software Group

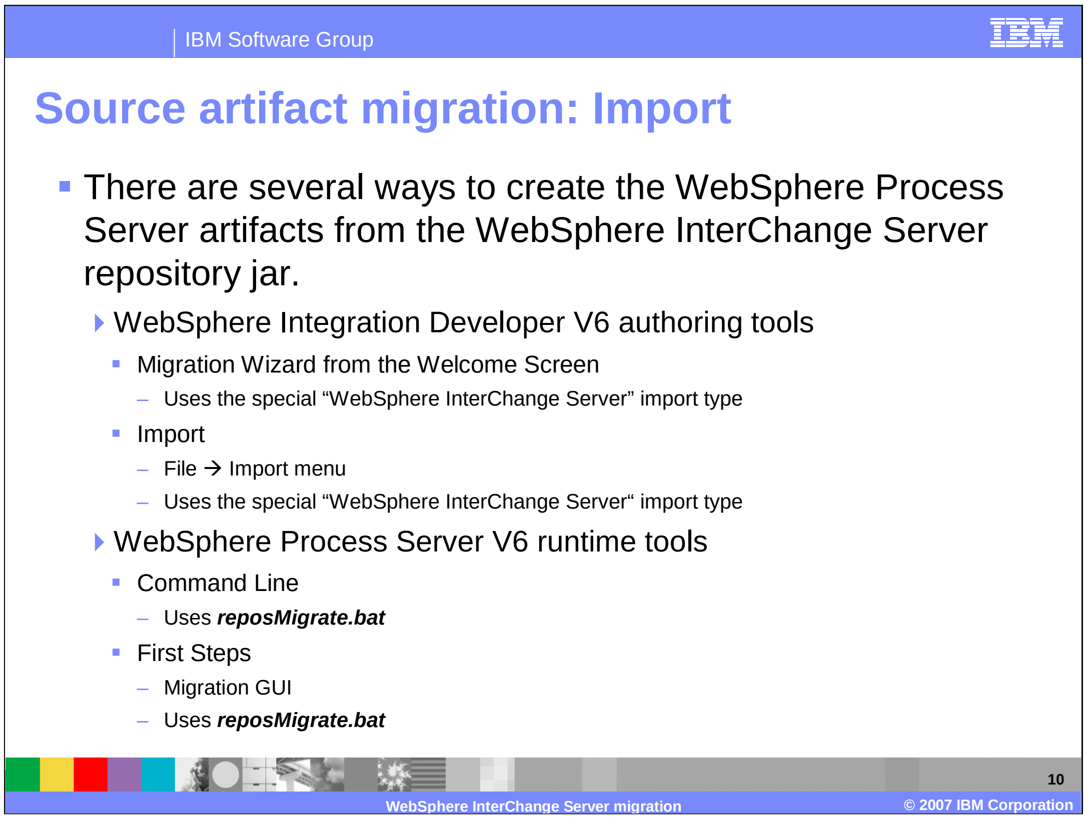
Source artifact migration: Export

- Make sure that WebSphere InterChange Server is at 4.2.2 or greater
- Start by exporting a part of the WebSphere InterChange Server repository that represents a complete solution.
  - Keep the solution as small as possible while maintaining a complete and self contained solution.
  - Include all of the artifacts required for the complete solution.
  - Use the WebSphere InterChange Server System Manager
    - The repos_copy command.

9

WebSphere InterChange Server migration © 2007 IBM Corporation

Before beginning a migration, make sure the version of WebSphere InterChange Server is supported by the migration utilities. The process of source artifact migration for WebSphere InterChange Server begins with exporting the artifacts from the WebSphere InterChange Server system to a jar file. Using the WebSphere InterChange Server System Manager, export all the artifacts that comprise a complete solution. Be sure to get everything that is referenced so that there will be no unresolved references when importing to WebSphere Process Server V6.



Source artifact migration: Import

- There are several ways to create the WebSphere Process Server artifacts from the WebSphere InterChange Server repository jar.
  - WebSphere Integration Developer V6 authoring tools
    - Migration Wizard from the Welcome Screen
      - Uses the special “WebSphere InterChange Server” import type
    - Import
      - File → Import menu
      - Uses the special “WebSphere InterChange Server“ import type
  - WebSphere Process Server V6 runtime tools
    - Command Line
      - Uses ***reposMigrate.bat***
    - First Steps
      - Migration GUI
      - Uses ***reposMigrate.bat***

The next step is to import the jar into WebSphere Integration Developer V6. The special WebSphere InterChange Server import type will recognize the artifacts and make the necessary conversions, creating new SCA artifacts.

The **Migration Wizard** provides a quick and easy way to launch the import utility and setup the new project at the same time. It can be invoked from the WebSphere Integration Developer Welcome Screen.

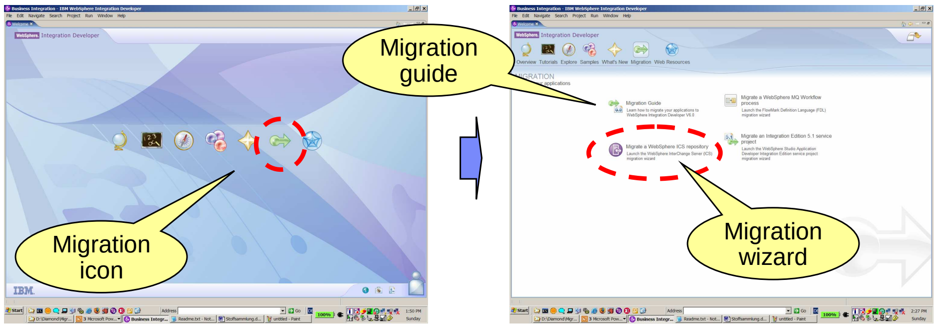
Again, its recommended that WebSphere Integration Developer be used initially. This will allow you the opportunity to understand how the conversions are made and to become familiar with the SCA components that are generated. Once a thorough understanding is achieved, the command line approach can be used to automate the process.

The “**First Steps**” and **reposMigrate.bat** are both tools provided by the WebSphere Process Server V6 runtime. The “First Steps” application is presented at the completion of the WebSphere Process Server V6 runtime installation.

reposMigrate.bat is located in the bin directory of the WebSphere runtime installation and is described in the WebSphere Integration Developer ‘Help’ under the Migration topic. It will do the source artifact conversions and can also be used with the runtime ServiceDeploy utility to create and package the EAR, bypassing WebSphere Integration Developer altogether.

Using the WebSphere Integration Developer migration wizard

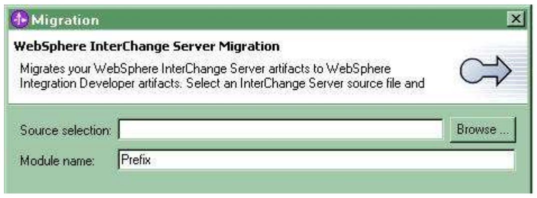
- From the Welcome page, click the migration icon to open the Migration page.
- From the Migration page, select the *"Migrate a WebSphere InterChange Server Repository"* option.



The Migration Wizard is available from the Welcome page of the WebSphere Integration Developer V6. To get to the Welcome page in WebSphere Integration Developer, select **Help → Welcome** menu. While on the Welcome page, notice the link to the migration guide, which is available in PDF format for easy printing.

Using the migration wizard

- WebSphere InterChange Server repository jar is used as the input source.



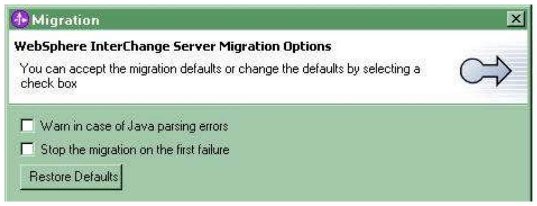
Migration

WebSphere InterChange Server Migration

Migrates your WebSphere InterChange Server artifacts to WebSphere Integration Developer artifacts. Select an InterChange Server source file and

Source selection:

Module name: Prefix



Migration

WebSphere InterChange Server Migration Options

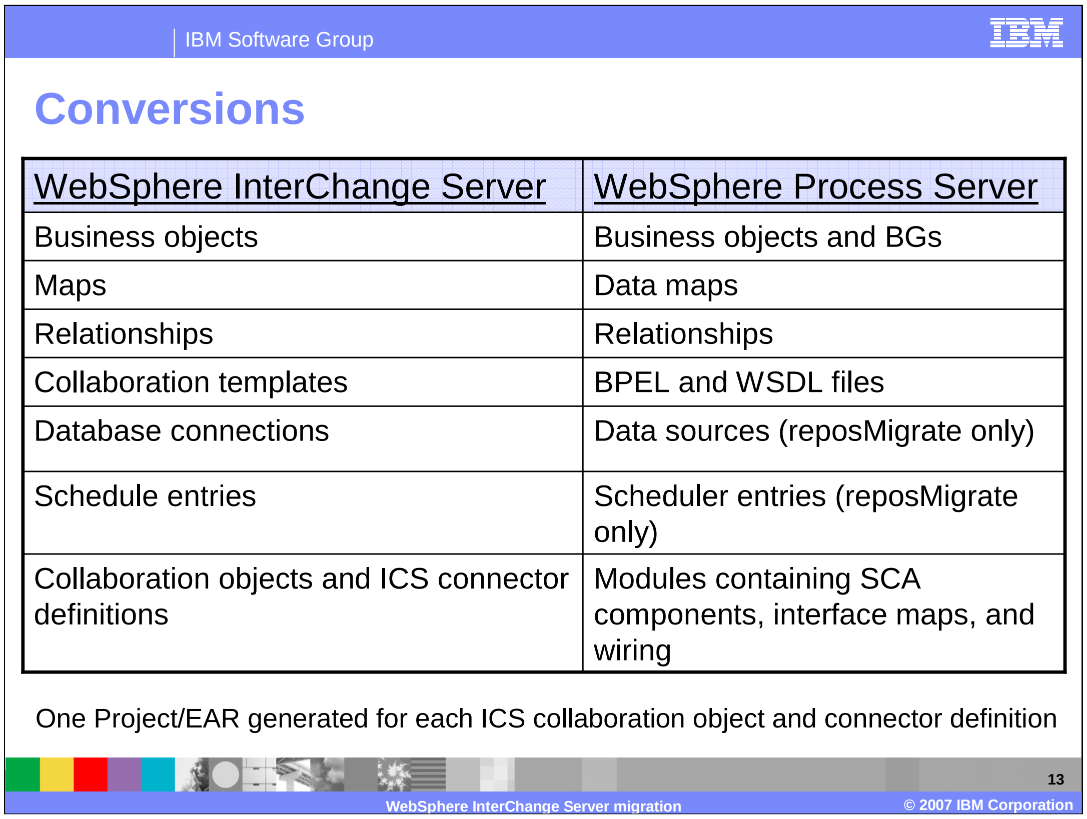
You can accept the migration defaults or change the defaults by selecting a check box

☐ Warn in case of Java parsing errors

☐ Stop the migration on the first failure

- Source Selection
  - Same as **–i** param in reposMigrate
- Module Name
  - Prefix given to all generated modules (Prefix_<ModuleName>)
- Warn in case of Java parsing errors
  - Same as **–wi** param in reposMigrate
- Stop the migration on the first failure
  - Same as **–fh** param in reposMigrate

The first panel presented by the Migration Wizard is the prompt for the module name. The module name prefix will be added to the name of the source artifact being migrated. For Example, a module name of **Simple** and a WebSphere InterChange Server connector called ClarifyConnector, will result in the Simple_ClarifyConnector WebSphere Process Server module. The second panel will provide the opportunity to specify the error handling options. The reposMigate options used by the Migration Wizard are shown on the right of the slide.



Conversions

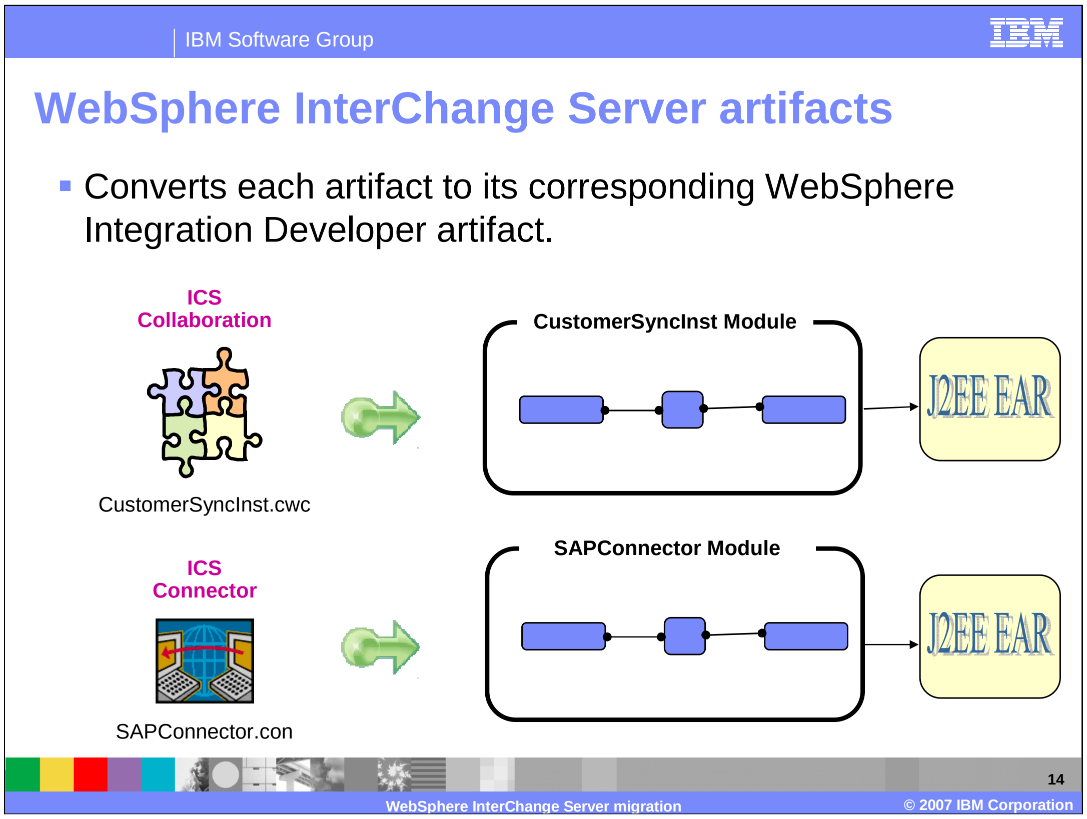
WebSphere InterChange Server	WebSphere Process Server
Business objects	Business objects and BGs
Maps	Data maps
Relationships	Relationships
Collaboration templates	BPEL and WSDL files
Database connections	Data sources (reposMigrate only)
Schedule entries	Scheduler entries (reposMigrate only)
Collaboration objects and ICS connector definitions	Modules containing SCA components, interface maps, and wiring

One Project/EAR generated for each ICS collaboration object and connector definition

Shown here is a mapping from the WebSphere InterChange Server artifacts to their corresponding WebSphere Process Server V6 artifacts. Many of the WebSphere InterChange Server artifacts, such as business objects, maps, and relationships, map directly to SCA artifacts. With WebSphere InterChange Server, business process flows are defined using collaboration templates and in WebSphere Process Server V6 flows are defined using BPEL.

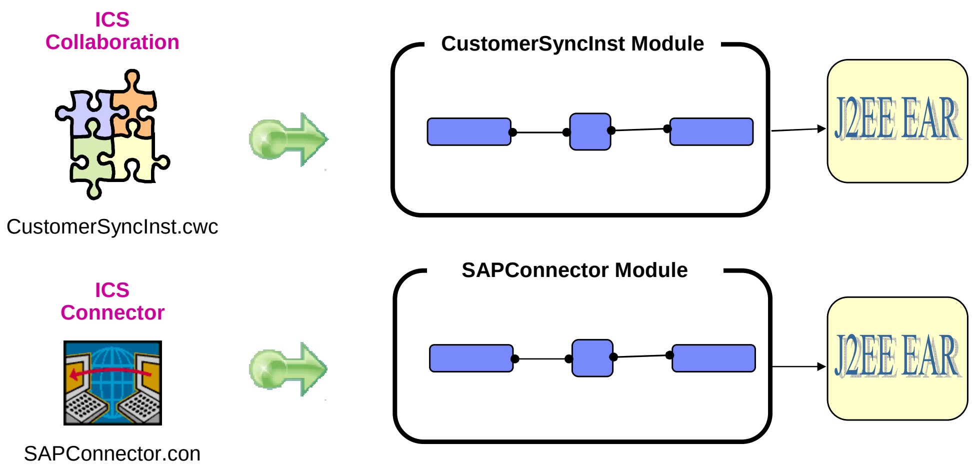
Notice that WebSphere Application Server data sources and the scheduler entries are only available using the reposMigrate command line utility. These must be manually configured when testing in WebSphere Integration Developer.

If there are Scheduler entries in the application being migrated then the WebSphere Process Server Administrative console can be used to set and configure the Application Scheduler from WebSphere Integration Developer.

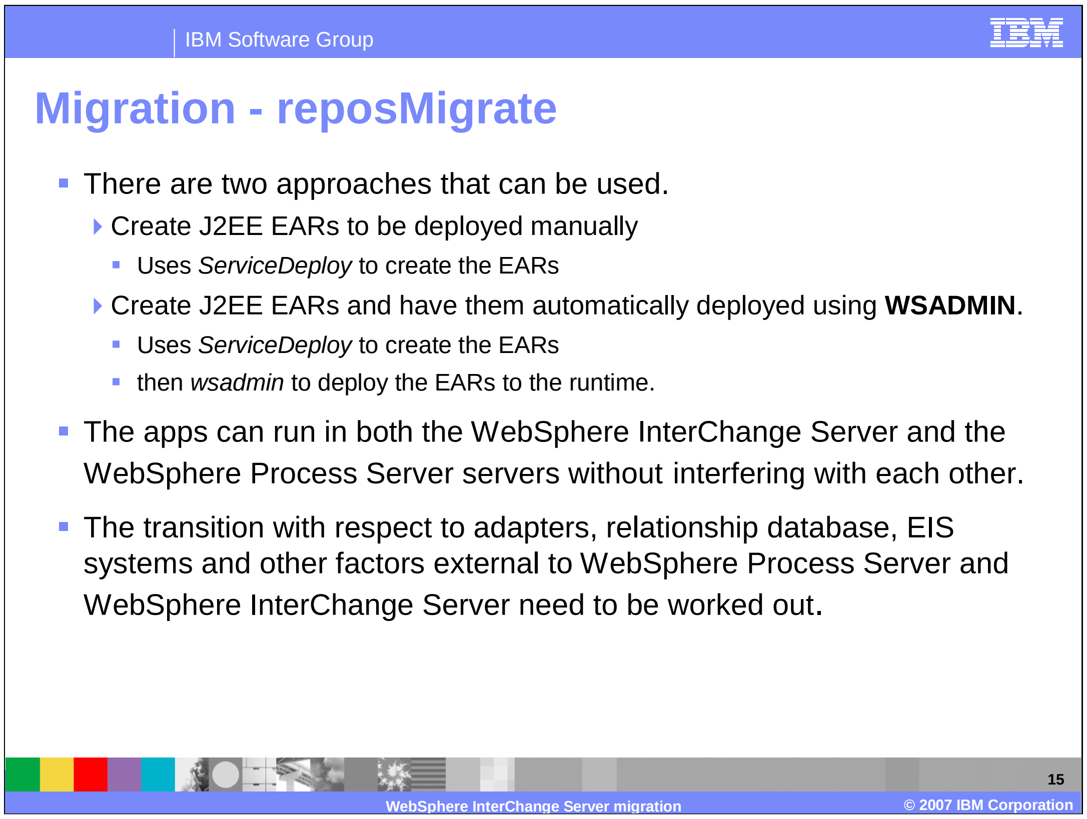


WebSphere InterChange Server artifacts

- Converts each artifact to its corresponding WebSphere Integration Developer artifact.



It is important to note that the WebSphere InterChange Server artifacts are mapped to SCA Modules, which are deployed as J2EE EARs, so there will be an EAR for each of the WebSphere InterChange Server artifacts. This has the potential for generating a lot of EAR files.



IBM Software Group

Migration - reposMigrate

- There are two approaches that can be used.
  - Create J2EE EARs to be deployed manually
    - Uses *ServiceDeploy* to create the EARs
  - Create J2EE EARs and have them automatically deployed using **WSADMIN**.
    - Uses *ServiceDeploy* to create the EARs
    - then *wsadmin* to deploy the EARs to the runtime.
- The apps can run in both the WebSphere InterChange Server and the WebSphere Process Server servers without interfering with each other.
- The transition with respect to adapters, relationship database, EIS systems and other factors external to WebSphere Process Server and WebSphere InterChange Server need to be worked out.

15

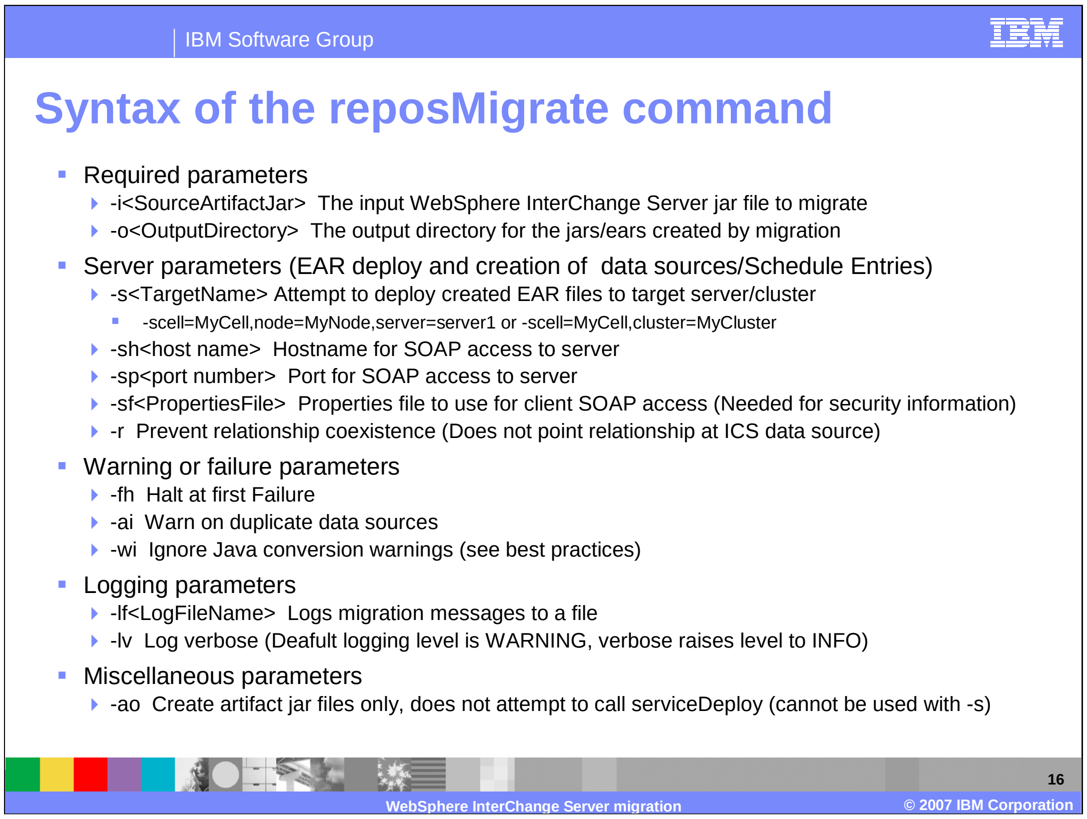
WebSphere InterChange Server migration © 2007 IBM Corporation

reposMigrate is the WebSphere Process Server command line utility for converting the WebSphere InterChange Server artifacts to WebSphere Process Server/SCA artifacts. reposMigrate can also invoke the ServiceDeploy utility to package the WebSphere Process Server/SCA artifacts into jars and EARs.

ServiceDeploy is the command utility used with WebSphere Process Server V6 to compile and package the SCA and J2EE artifacts programmatically. The reposMigrate utility receives the WebSphere InterChange Server artifact jar, creates the SCA components and then invokes the ServiceDeploy utility to create the J2EE EAR. The J2EE EAR can then be deployed manually using the WebSphere Process Server Administrative console, or programmatically using WSADMIN (the WebSphere scripting language). Additionally the reposMigrate utility includes an option to invoke the WSADMIN utility after the Service Deploy has created the EAR.

Both reposMigrate and ServiceDeploy can be used with ANT and a source code control system to automate the process even further.

Before the EARs are deployed to the runtime, whether you do it automatically or manually, the details for cutting over with respect to the adapters, relationship database and other external dependencies must be worked out.



Syntax of the reposMigrate command

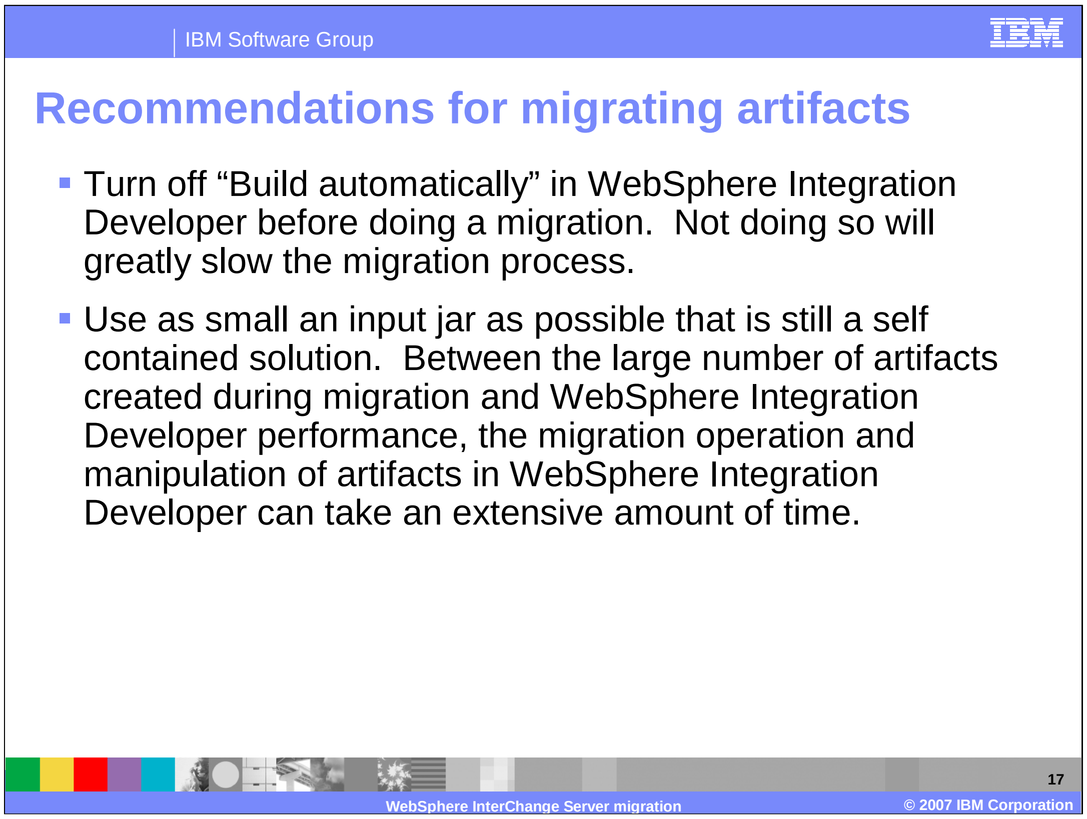
- Required parameters
  - -i<SourceArtifactJar> The input WebSphere InterChange Server jar file to migrate
  - -o<OutputDirectory> The output directory for the jars/ears created by migration
- Server parameters (EAR deploy and creation of data sources/Schedule Entries)
  - -s<TargetName> Attempt to deploy created EAR files to target server/cluster
    - -scell=MyCell,node=MyNode,server=server1 or -scell=MyCell,cluster=MyCluster
  - -sh<host name> Hostname for SOAP access to server
  - -sp<port number> Port for SOAP access to server
  - -sf<PropertiesFile> Properties file to use for client SOAP access (Needed for security information)
  - -r Prevent relationship coexistence (Does not point relationship at ICS data source)
- Warning or failure parameters
  - -fh Halt at first Failure
  - -ai Warn on duplicate data sources
  - -wi Ignore Java conversion warnings (see best practices)
- Logging parameters
  - -lf<LogFileName> Logs migration messages to a file
  - -lv Log verbose (Deafult logging level is WARNING, verbose raises level to INFO)
- Miscellaneous parameters
  - -ao Create artifact jar files only, does not attempt to call serviceDeploy (cannot be used with -s)

Shown here are the many options that can be used with reposMigrate.

It is recommended that you use the -ao flag when using reposMigrate to generate artifact jar files only.

This also eliminates the need for the -s, -sh, -sp, -sf, -r, and -ai parameters.

Once the artifacts have been edited, serviceDeploy can be called to create EAR files from the JARs.



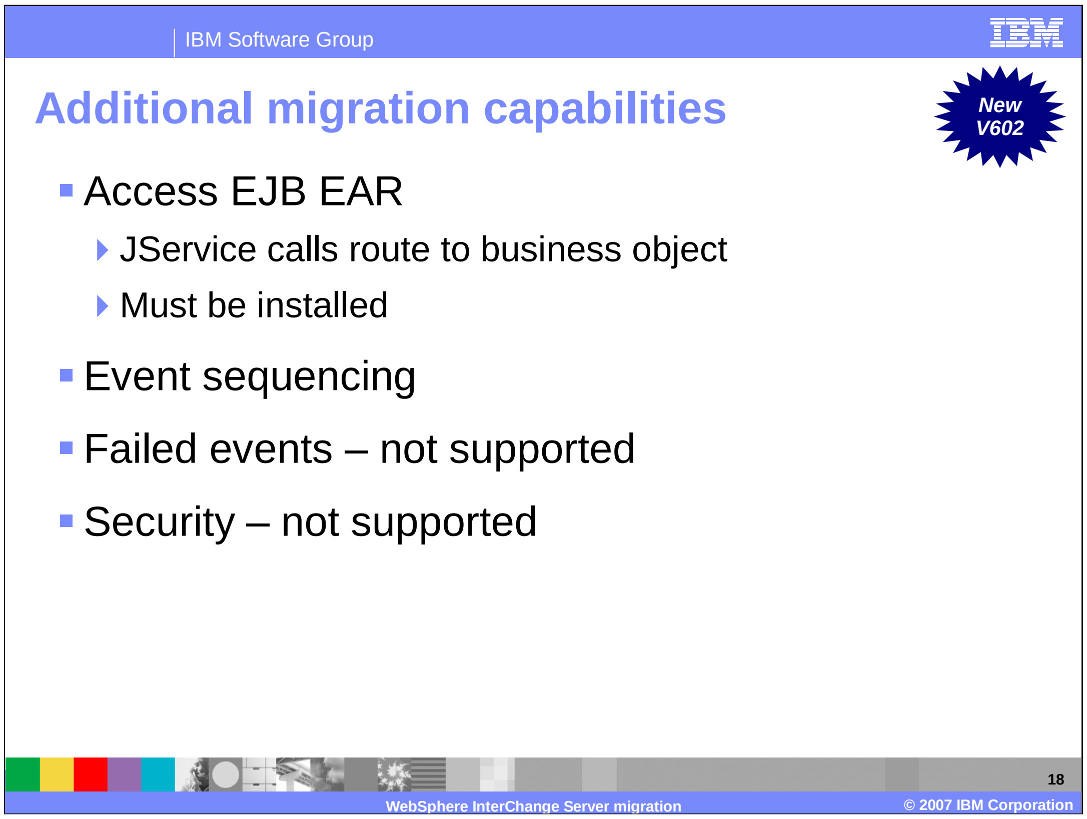
Recommendations for migrating artifacts

- Turn off “Build automatically” in WebSphere Integration Developer before doing a migration. Not doing so will greatly slow the migration process.
- Use as small an input jar as possible that is still a self contained solution. Between the large number of artifacts created during migration and WebSphere Integration Developer performance, the migration operation and manipulation of artifacts in WebSphere Integration Developer can take an extensive amount of time.

The migration process creates many new artifacts and if the auto build feature is on, a build will be started before all the artifacts have been created. As new artifacts are created, new builds are also started. The result is a degradation in the overall build performance.

Disabling the “Build Automatically” option in WebSphere Integration Developer will speed up the migration process.

Using the smallest possible input jar to contain a solution will also streamline the migration and reduce the time required to complete.



Additional migration capabilities

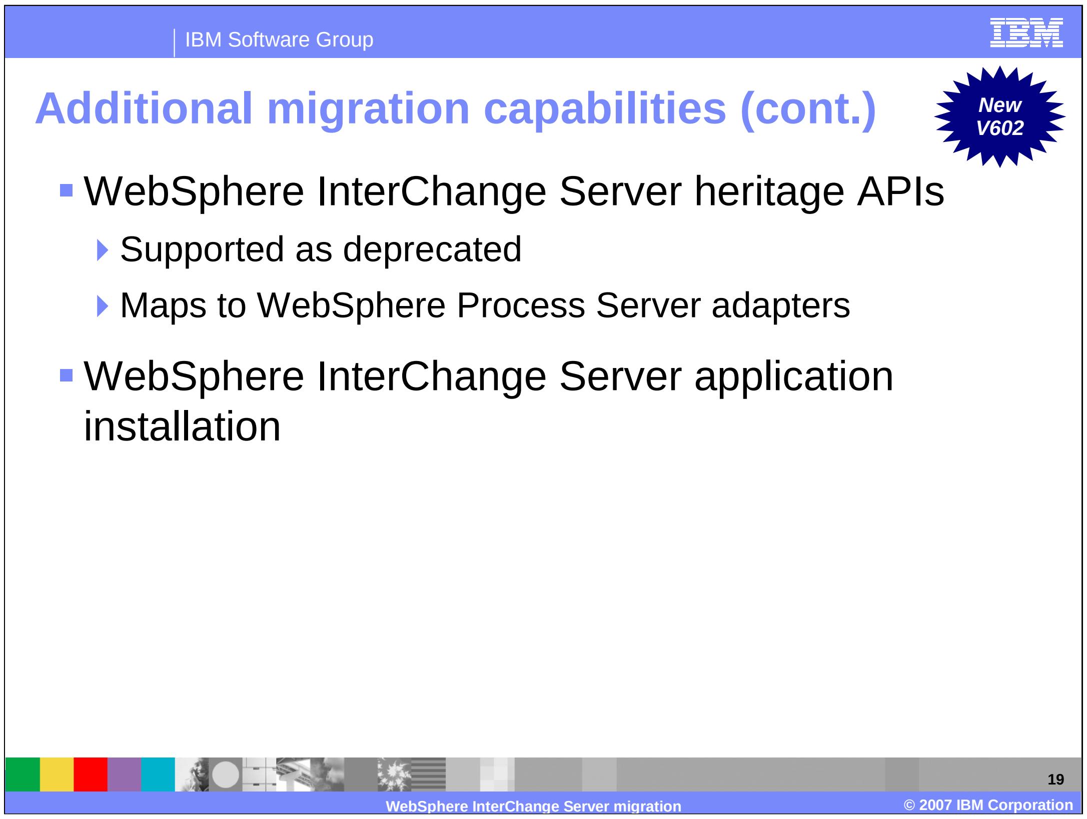
New V602

- Access EJB EAR
  - JService calls route to business object
  - Must be installed
- Event sequencing
- Failed events – not supported
- Security – not supported

WebSphere InterChange Server allowed you to not only invoke a collaboration as the result of input from an adapter, but also from standard J2EE EJB calls through the Access EJB hosted in a J2EE environment. The Access EJB will be replicated in WebSphere Process Server as an EJB that uses the migrated module naming schema to allow JService calls to route the input Business Object to the appropriate migrated module. All external clients can maintain the same EJB calls, but instead of invoking the target WebSphere InterChange Server collaboration, it will invoke the BPEL in the migrated module. The Access EJB EAR must be manually installed.

WebSphere InterChange Server provided server support for Event Sequencing for Message Driven Beans. The migrated modules will take advantage of WebSphere Process Server Event Sequencing to retain functional parity. This support is provided for SCA asynchronous invocations only.

Migration for failed events and security is not supported.



Additional migration capabilities (cont.)

New V602

- WebSphere InterChange Server heritage APIs
  - Supported as deprecated
  - Maps to WebSphere Process Server adapters
- WebSphere InterChange Server application installation

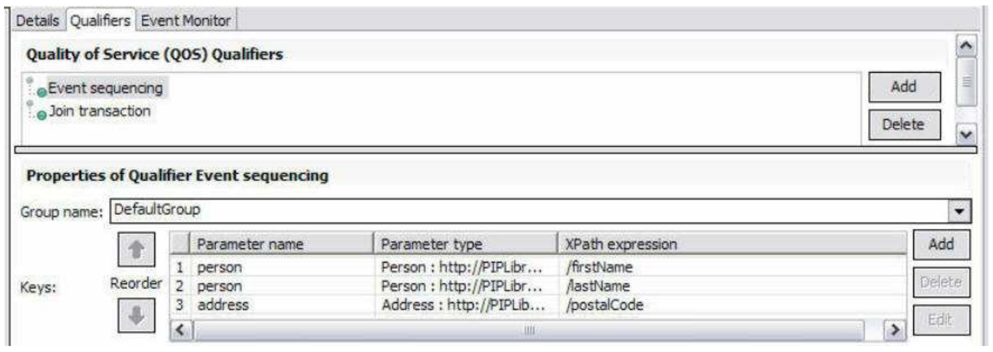
WebSphere InterChange Server Collaboration Templates and Maps can contain custom Java code that references a set of supported APIs that were delivered with WebSphere InterChange Server. Many of these APIs are implemented as deprecated in the WebSphere Process Server runtime. These APIs take advantage of the Adapter usage pattern in WebSphere Process Server.

The WebSphere InterChange Server Application installation will read the administrative artifacts data created by the migration and create administrative objects in WebSphere Process Server that match those that were found in WebSphere InterChange Server.

Qualifier event sequencing

New V602

- Set event sequencing qualifier



Quality of Service (QOS) Qualifiers

- Event sequencing
- Join transaction

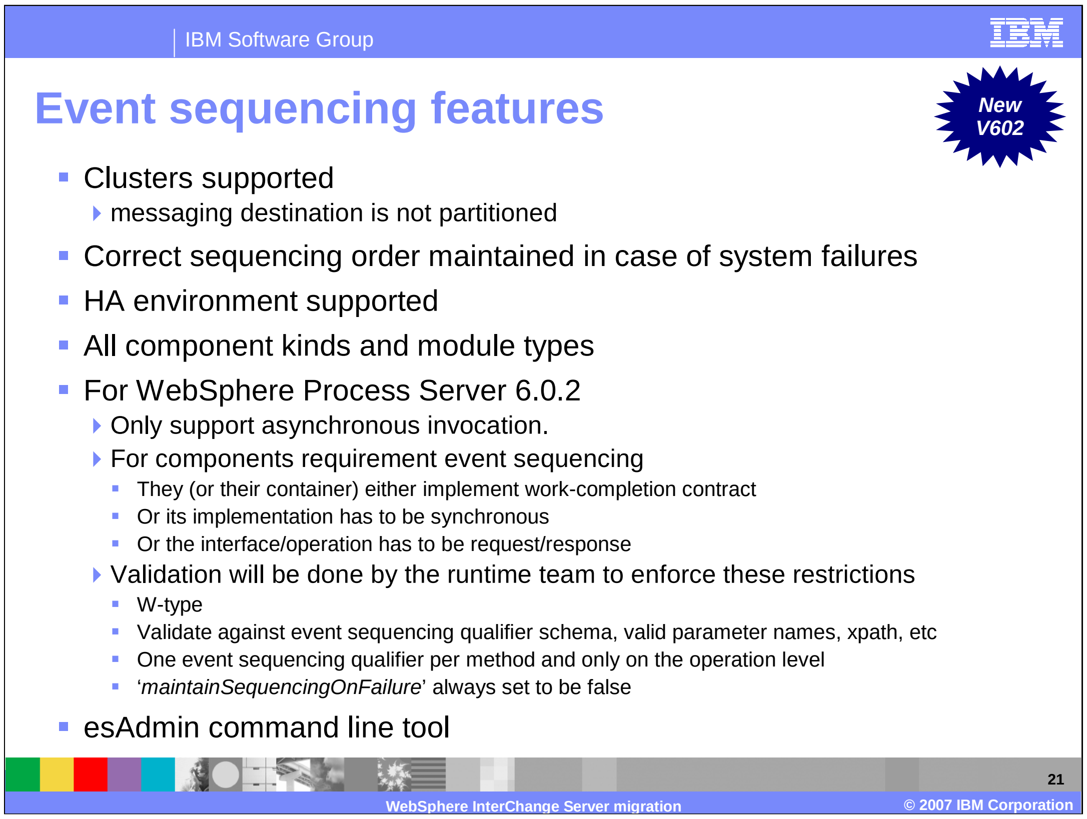
Properties of Qualifier Event sequencing

Group name: DefaultGroup

Keys:

	Parameter name	Parameter type	XPath expression
1	person	Person : http://PIPLibr...	/firstName
2	person	Person : http://PIPLibr...	/lastName
3	address	Address : http://PIPLib...	/postalCode

WebSphere Integration Developer can add the event sequencing qualifier to an interface method of an SCA component as shown here. When the event sequencing qualifier is specified, the component is enabled for event sequencing.


IBM Software Group

Event sequencing features

New V602

- Clusters supported
  - messaging destination is not partitioned
- Correct sequencing order maintained in case of system failures
- HA environment supported
- All component kinds and module types
- For WebSphere Process Server 6.0.2
  - Only support asynchronous invocation.
  - For components requirement event sequencing
    - They (or their container) either implement work-completion contract
    - Or its implementation has to be synchronous
    - Or the interface/operation has to be request/response
  - Validation will be done by the runtime team to enforce these restrictions
    - W-type
    - Validate against event sequencing qualifier schema, valid parameter names, xpath, etc
    - One event sequencing qualifier per method and only on the operation level
    - '*maintainSequencingOnFailure*' always set to be false
- esAdmin command line tool

21

WebSphere InterChange Server migration © 2007 IBM Corporation

Event sequencing will work in a clustered environment as long as the underlying messaging destination is not partitioned.

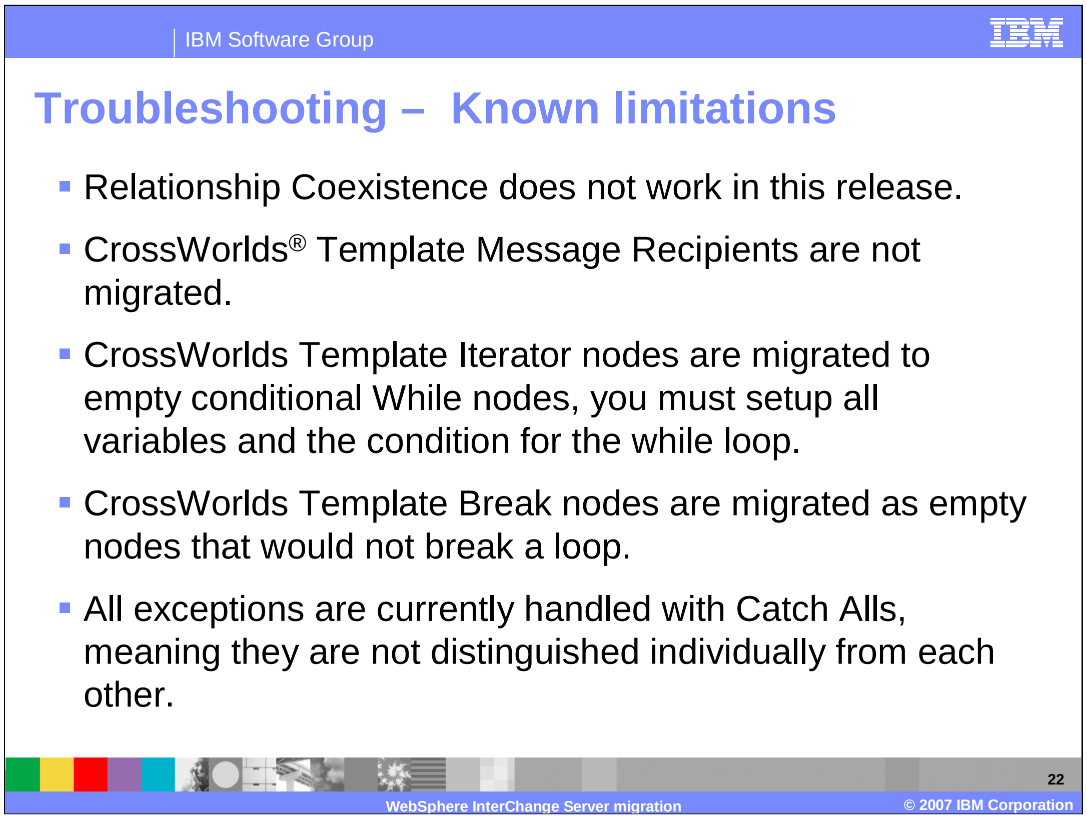
Event sequencing maintains the correct sequencing order in case of system failures. The order is kept in the system database and will continue where it left off when the system comes back up. Event sequencing works in a HA environment providing failover capability. All component kinds and module types are supported.

For the WebSphere Process Server V602 release there are some limitations because of resource constraints. For this release, Event Sequencing is supported only for components that are going to be invoked using the SCA asynchronous invocation style. Synchronous invocations to components with asynchronous implementations do require event sequencing also but will not be supported without the W-type interface and implement the work completion contract.

Validation is done when you build your project in WebSphere Integration Developer or when you run serviceDeploy. This is design time validation.

The runtime shall validate the event sequencing qualifier using the rules shown here.

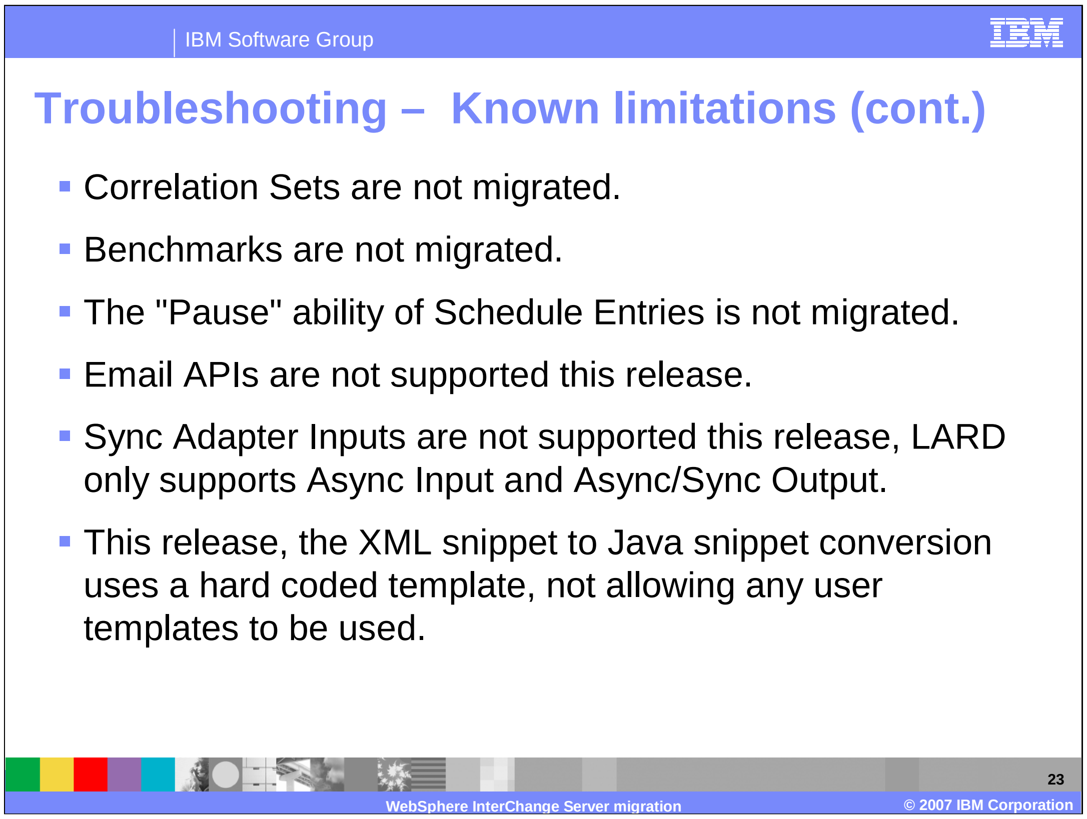
The esAdmin command line tool will list the locks that are currently in place and allow deleting locks that occur because of a circular dependency between modules.



Troubleshooting – Known limitations

- Relationship Coexistence does not work in this release.
- CrossWorlds® Template Message Recipients are not migrated.
- CrossWorlds Template Iterator nodes are migrated to empty conditional While nodes, you must setup all variables and the condition for the while loop.
- CrossWorlds Template Break nodes are migrated as empty nodes that would not break a loop.
- All exceptions are currently handled with Catch Alls, meaning they are not distinguished individually from each other.

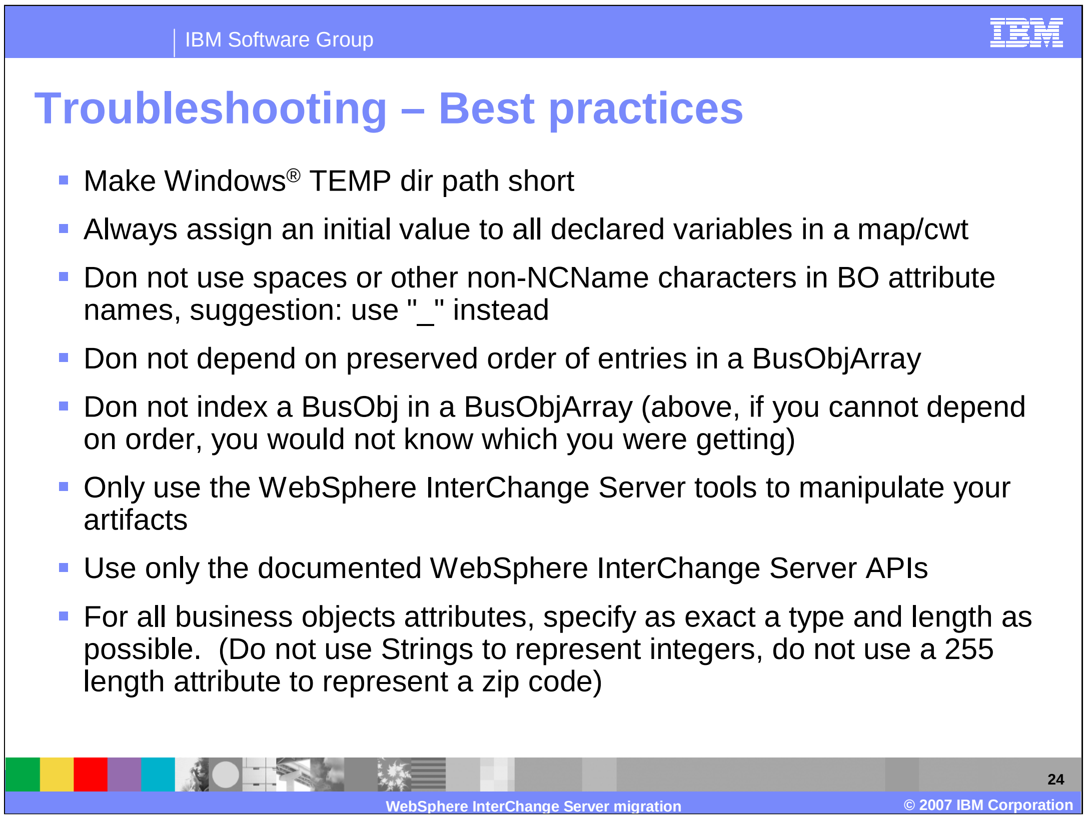
Shown here are some known limitations with respect to migration. Many of these limitations are related to CrossWorlds Templates.



Troubleshooting – Known limitations (cont.)

- Correlation Sets are not migrated.
- Benchmarks are not migrated.
- The "Pause" ability of Schedule Entries is not migrated.
- Email APIs are not supported this release.
- Sync Adapter Inputs are not supported this release, LARD only supports Async Input and Async/Sync Output.
- This release, the XML snippet to Java snippet conversion uses a hard coded template, not allowing any user templates to be used.

Shown here are some additional known limitations.

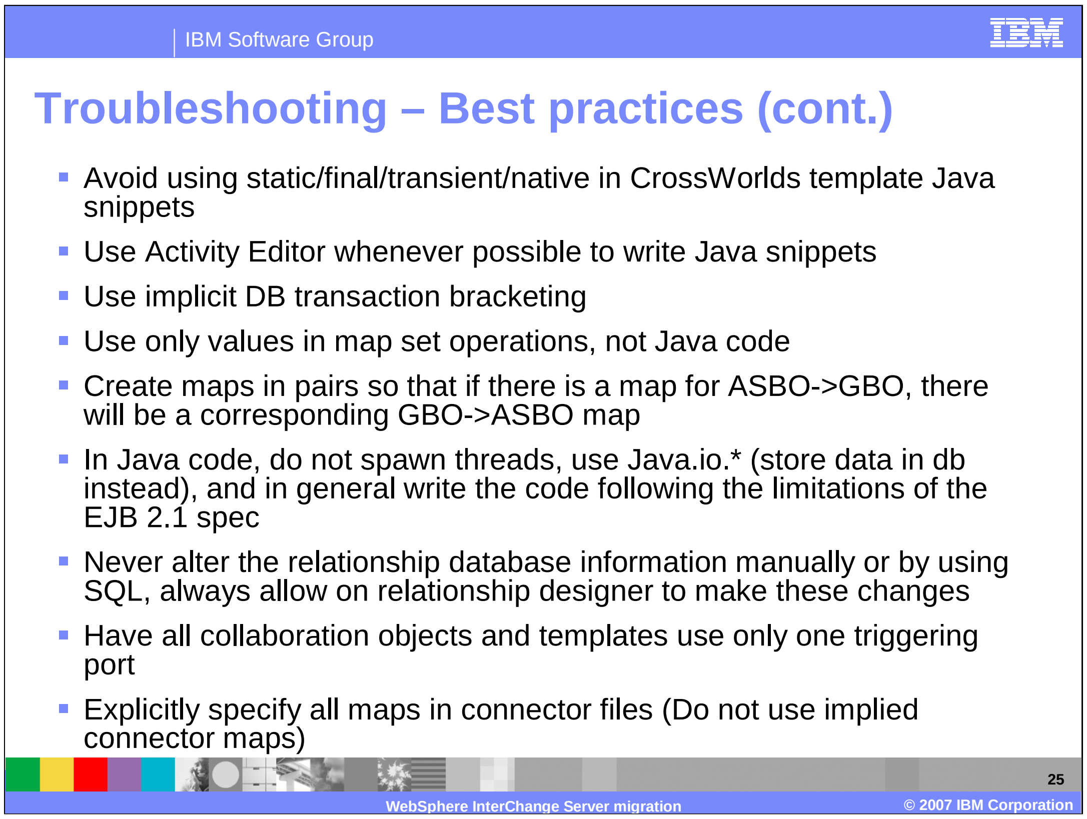


Troubleshooting – Best practices

- Make Windows® TEMP dir path short
- Always assign an initial value to all declared variables in a map/cwt
- Don not use spaces or other non-NCName characters in BO attribute names, suggestion: use "_" instead
- Don not depend on preserved order of entries in a BusObjArray
- Don not index a BusObj in a BusObjArray (above, if you cannot depend on order, you would not know which you were getting)
- Only use the WebSphere InterChange Server tools to manipulate your artifacts
- Use only the documented WebSphere InterChange Server APIs
- For all business objects attributes, specify as exact a type and length as possible. (Do not use Strings to represent integers, do not use a 255 length attribute to represent a zip code)

Shown here are some troubleshooting best practices. On the Windows platform, the length of fully qualified file names is limited to 256 characters. When the Workspace is nested in a directory structure and the objects have long names and namespaces, it is very easy to exceed the Windows 256 character limit.

Usage of the WebSphere InterChange Server tools, artifacts and APIs will help to eliminate problems.


IBM Software Group

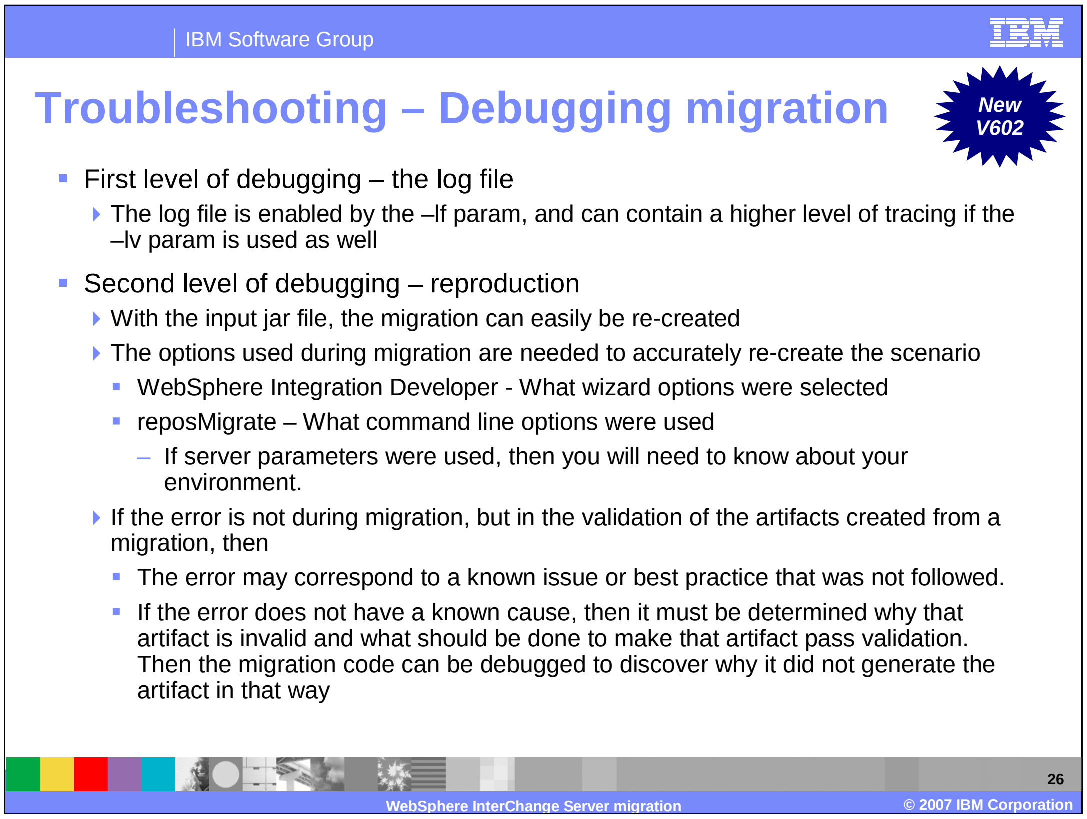
Troubleshooting – Best practices (cont.)

- Avoid using static/final/transient/native in CrossWorlds template Java snippets
- Use Activity Editor whenever possible to write Java snippets
- Use implicit DB transaction bracketing
- Use only values in map set operations, not Java code
- Create maps in pairs so that if there is a map for ASBO->GBO, there will be a corresponding GBO->ASBO map
- In Java code, do not spawn threads, use Java.io.* (store data in db instead), and in general write the code following the limitations of the EJB 2.1 spec
- Never alter the relationship database information manually or by using SQL, always allow on relationship designer to make these changes
- Have all collaboration objects and templates use only one triggering port
- Explicitly specify all maps in connector files (Do not use implied connector maps)

25

WebSphere InterChange Server migration © 2007 IBM Corporation

Continuing the list of best practices again points out that the use of WebSphere tools will help to reduce problems that may occur during migration or editing of the migrated artifacts.



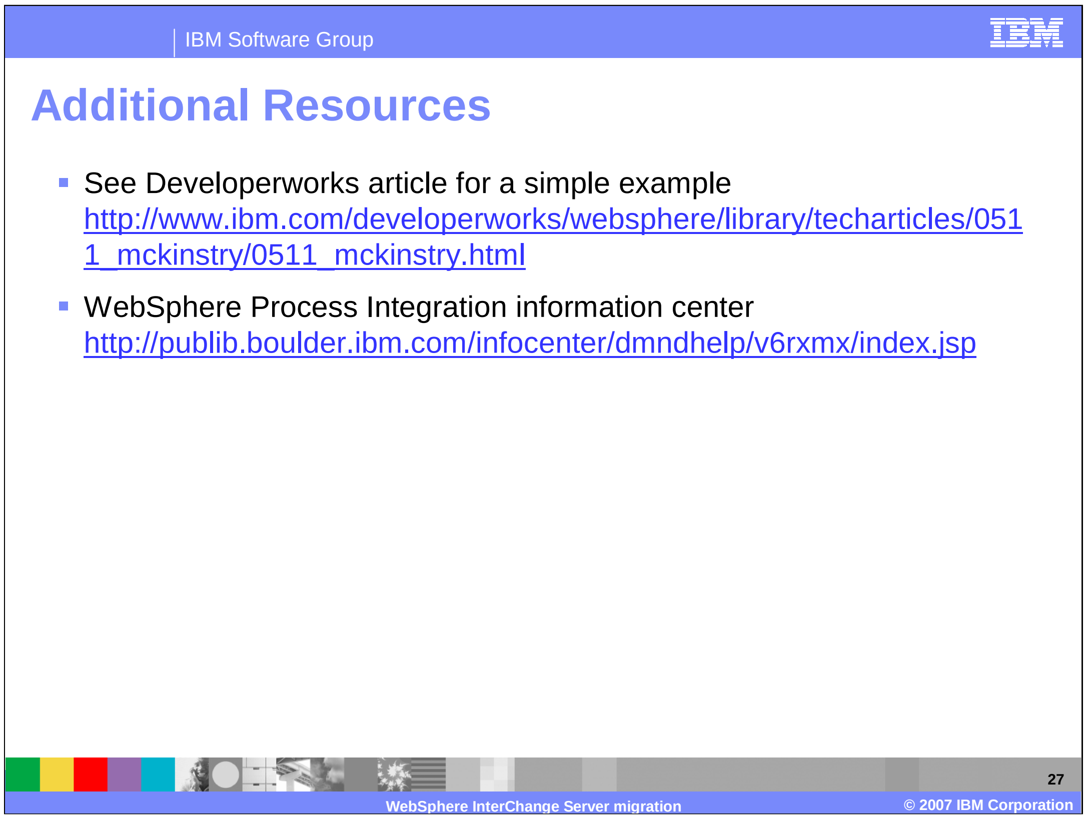
Troubleshooting – Debugging migration

- First level of debugging – the log file
  - The log file is enabled by the –lf param, and can contain a higher level of tracing if the –lv param is used as well
- Second level of debugging – reproduction
  - With the input jar file, the migration can easily be re-created
  - The options used during migration are needed to accurately re-create the scenario
    - WebSphere Integration Developer - What wizard options were selected
    - reposMigrate – What command line options were used
      - If server parameters were used, then you will need to know about your environment.
  - If the error is not during migration, but in the validation of the artifacts created from a migration, then
    - The error may correspond to a known issue or best practice that was not followed.
    - If the error does not have a known cause, then it must be determined why that artifact is invalid and what should be done to make that artifact pass validation. Then the migration code can be debugged to discover why it did not generate the artifact in that way

Shown here are some tips for troubleshooting problems encountered during migration, including the location of log files and the type of information contained in them. The log will contain any error messages that often match up to either a current know issue or best practice that was not followed. If the failure does not produce an error message, or produces the “Unknown Exception” error message, further debugging is needed.

After reproducing the problem, there are several options to be considered. If the problem occurred during migration, check the parameters used for reposMigrate. For example, if you chose to deploy to server1, check to see that server1 exists, check the host name and soap port number. If it has security enabled, check that the soap.client.properties file matches the security settings.

If the error did not occur during migration, verify that the best practices were followed. If so, follow the basic WebSphere Process Server procedure for solving these issues as if the artifacts were created brand new by themselves, rather than being created by migration.



IBM Software Group

Additional Resources

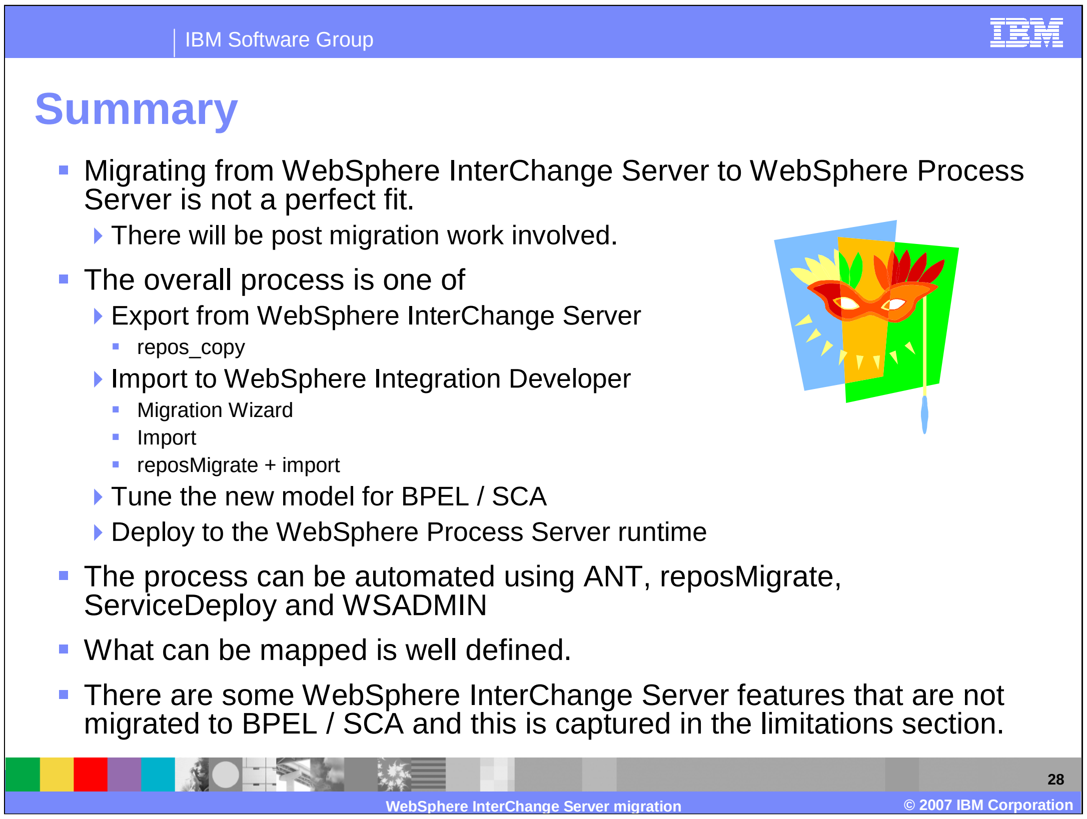
- See Developerworks article for a simple example http://www.ibm.com/developerworks/websphere/library/techarticles/0511_mckinstry/0511_mckinstry.html
- WebSphere Process Integration information center http://publib.boulder.ibm.com/infocenter/dmndhelp/v6rxmx/index.jsp

27

WebSphere InterChange Server migration

© 2007 IBM Corporation

The article on Developerworks is a tutorial that walks through a simple example. This is a great way to get started.



Summary

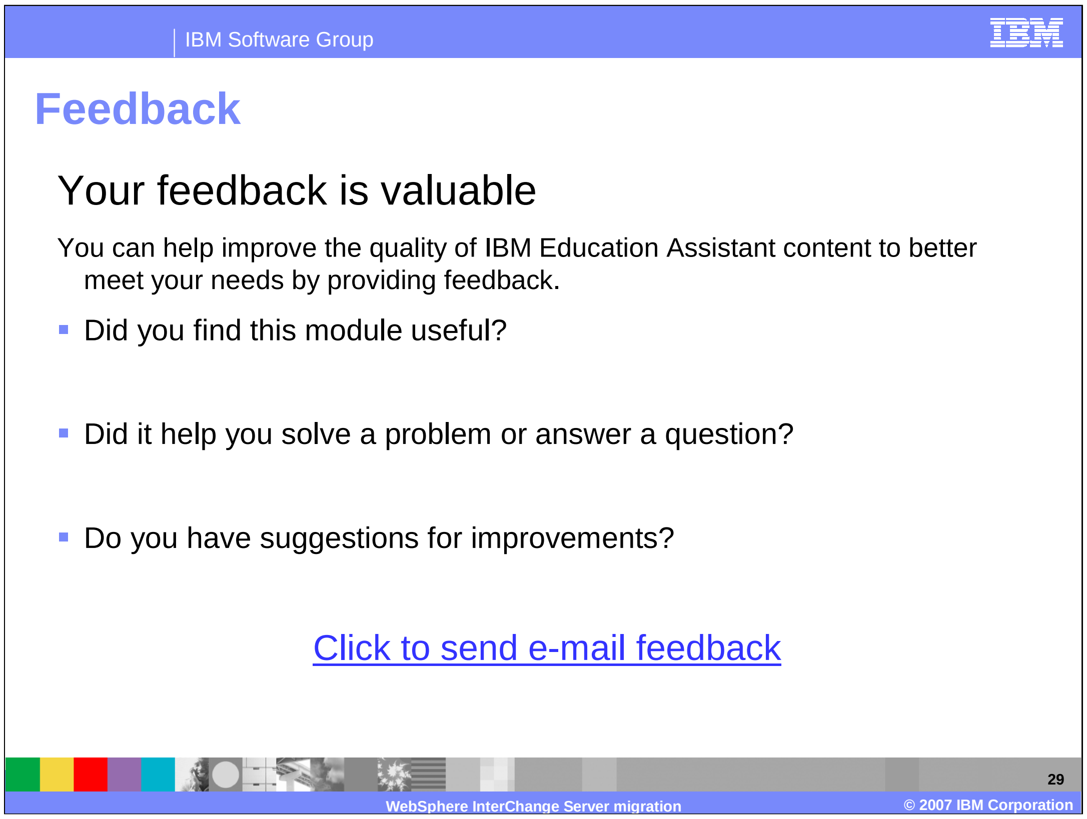
- Migrating from WebSphere InterChange Server to WebSphere Process Server is not a perfect fit.
  - There will be post migration work involved.
- The overall process is one of
  - Export from WebSphere InterChange Server
    - repos_copy
  - Import to WebSphere Integration Developer
    - Migration Wizard
    - Import
    - reposMigrate + import
  - Tune the new model for BPEL / SCA
  - Deploy to the WebSphere Process Server runtime
- The process can be automated using ANT, reposMigrate, ServiceDeploy and WSADMIN
- What can be mapped is well defined.
- There are some WebSphere InterChange Server features that are not migrated to BPEL / SCA and this is captured in the limitations section.

Depending on the complexity of the WebSphere InterChange Server application there will be some post migration work involved.

The basic process is to export the application from the source system. Then import to the target system, edit, resolve errors and tune the application for BPEL / SCA and deploy the application to the WebSphere Process Server V6 runtime.

When importing the WebSphere InterChange Server repository jar into WebSphere Integration Developer, there are several options available, the Migration Wizard, Import **or** reposMigrate and import.

Using reposMigrate and import provides more control and flexibility over the process and is useful for debugging problematic migrations.



IBM Software Group

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?

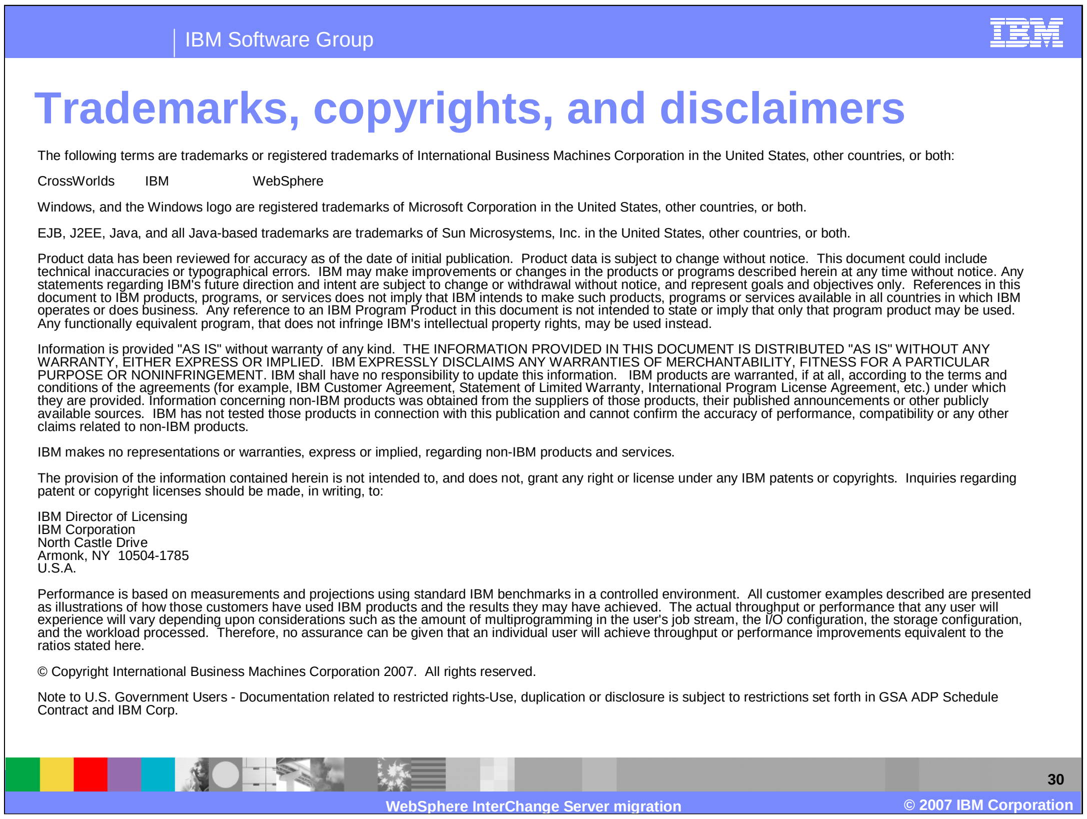
Click to send e-mail feedback

29

WebSphere InterChange Server migration

© 2007 IBM Corporation

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



Trademarks, copyrights, and disclaimers