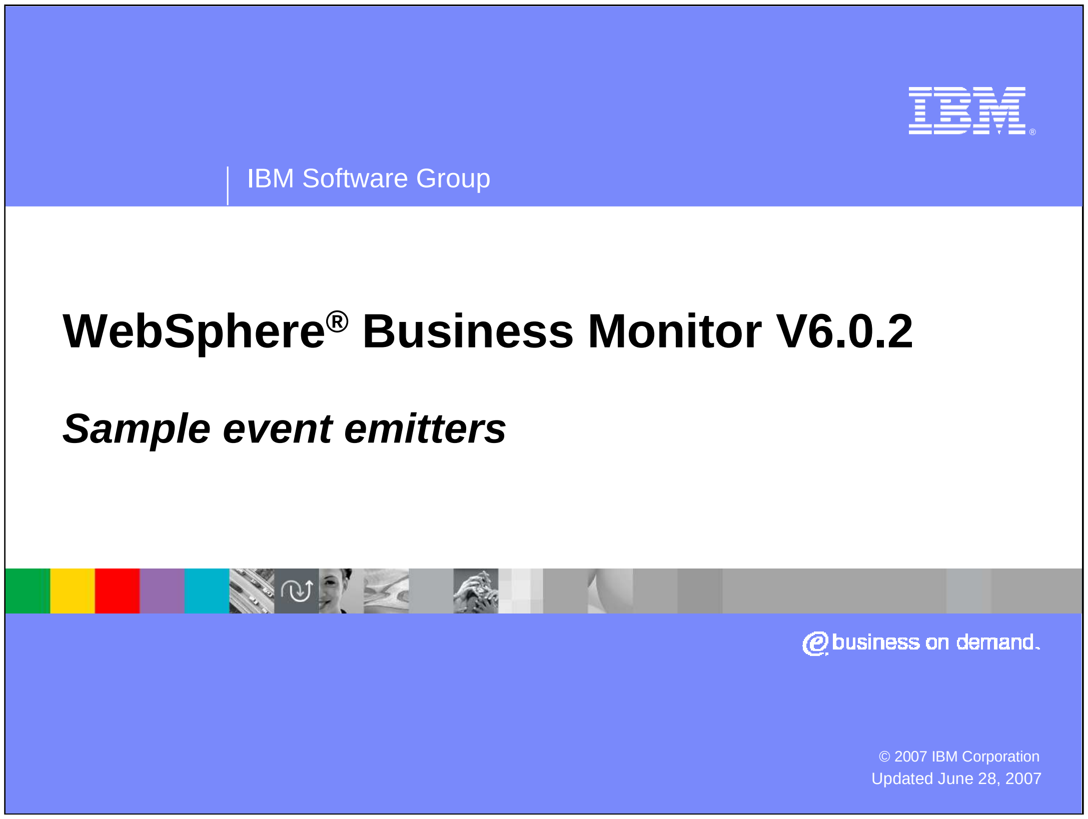


IBM Software Group

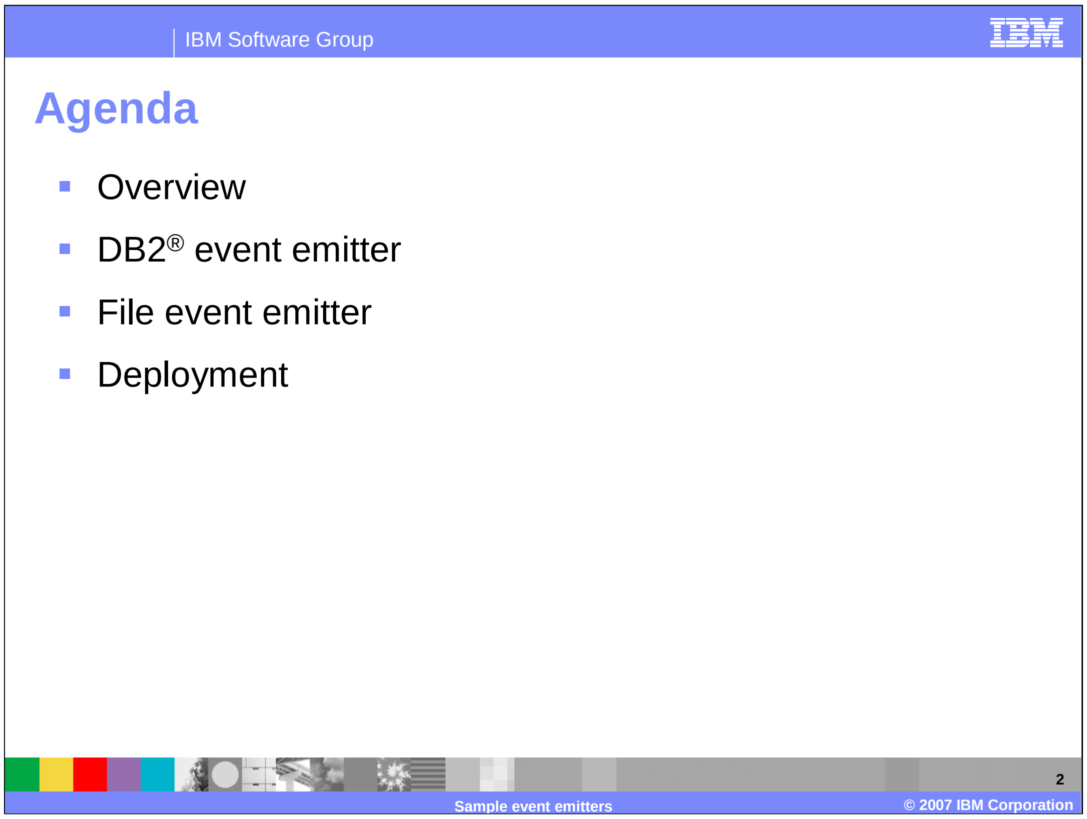
WebSphere® Business Monitor V6.0.2

Sample event emitters

business on demand.

Updated June 28, 2007

This presentation will introduce the DB2 Emitter and File Emitter components in WebSphere Business Monitor version 6.0.2.



Agenda

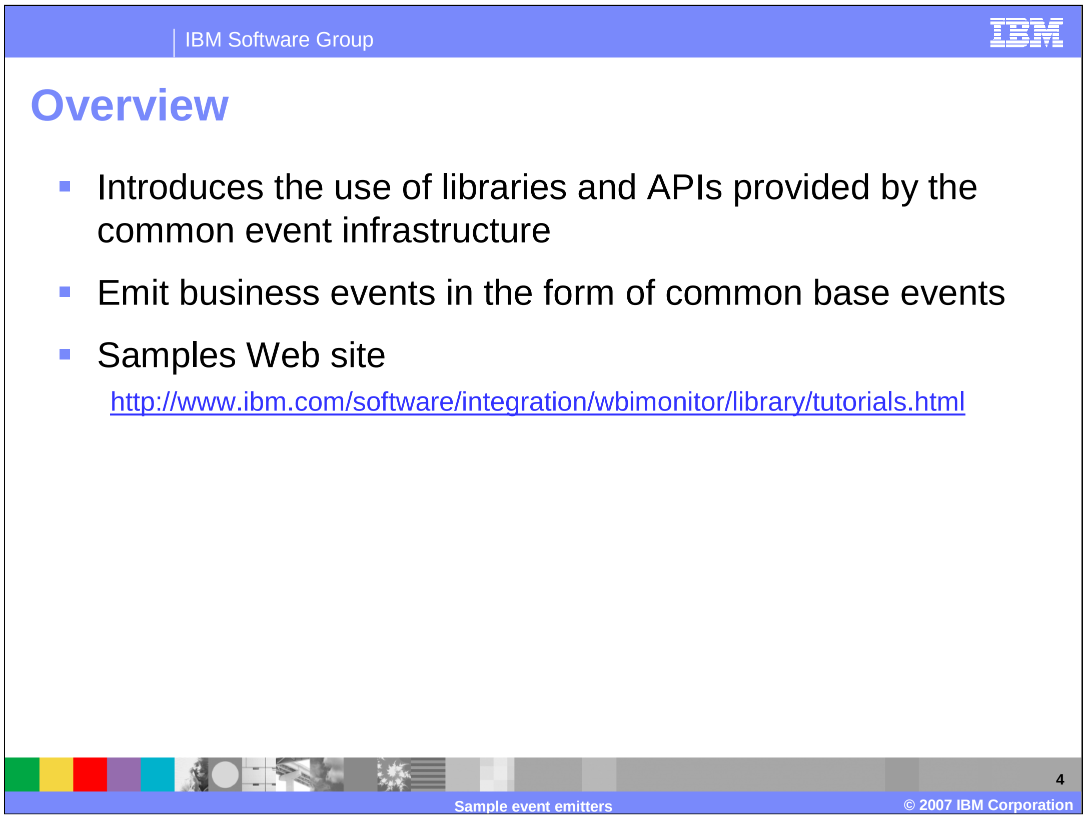
- Overview
- DB2® event emitter
- File event emitter
- Deployment

Here is the agenda for the presentation. It covers the sample emitters, then discusses the DB2 and file emitters in detail, including deployment and administration.

Section

Overview

This section discusses the overview of the sample emitters.

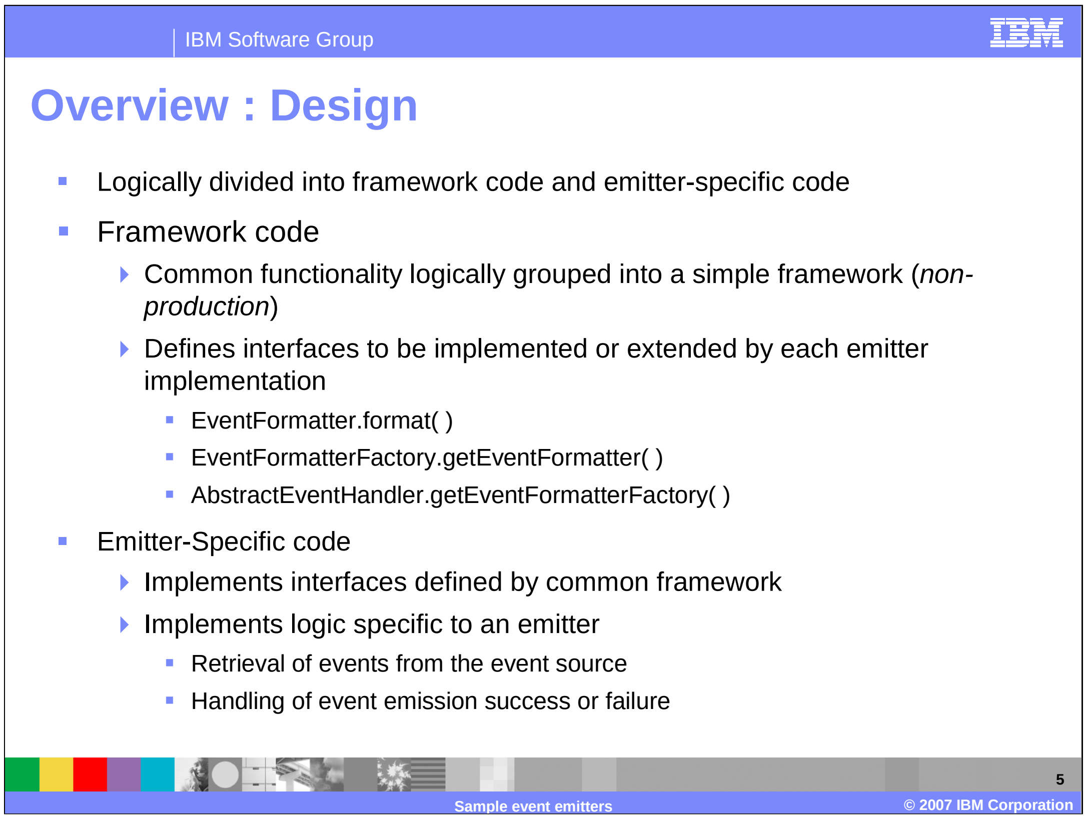


Overview

- Introduces the use of libraries and APIs provided by the common event infrastructure
- Emit business events in the form of common base events
- Samples Web site

  http://www.ibm.com/software/integration/wbimonitor/library/tutorials.html

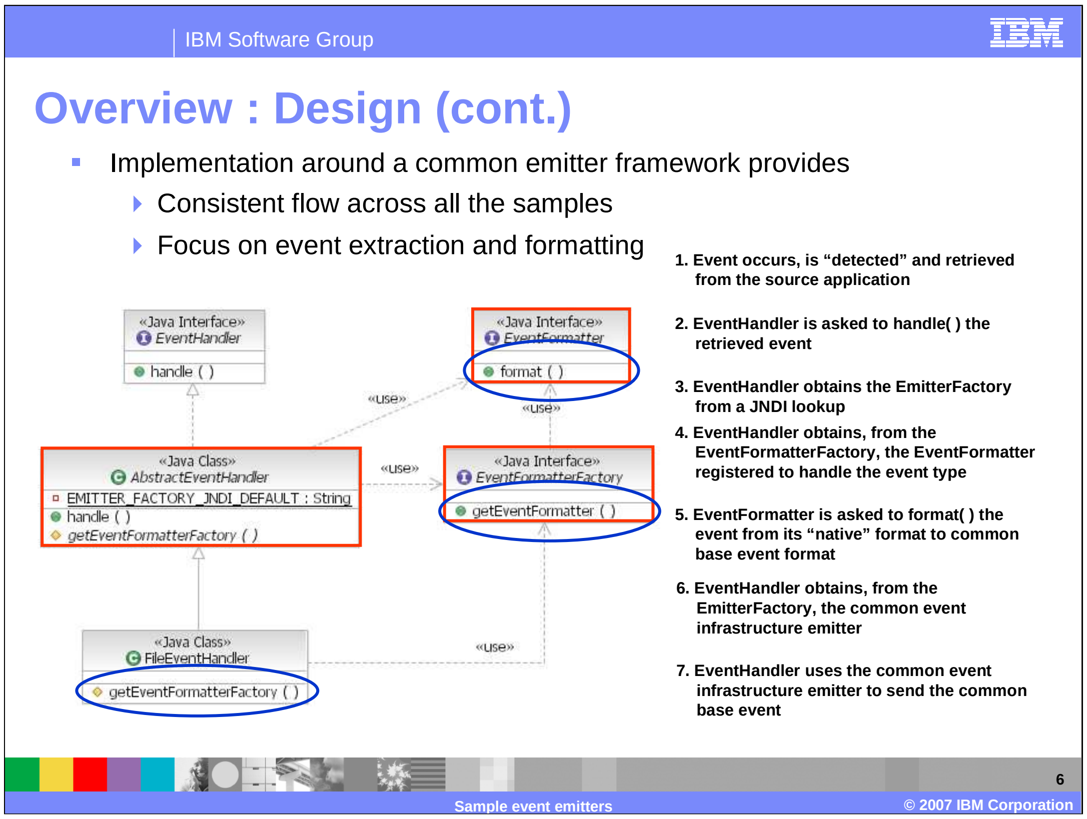
The goal of the Sample Event Emitters is to introduce the use of the libraries and APIs provided by the common event infrastructure to generate business events in the form of common base events. Common base events are the data packaging and format used by the WebSphere Business Monitor server to propagate business events.



Overview : Design

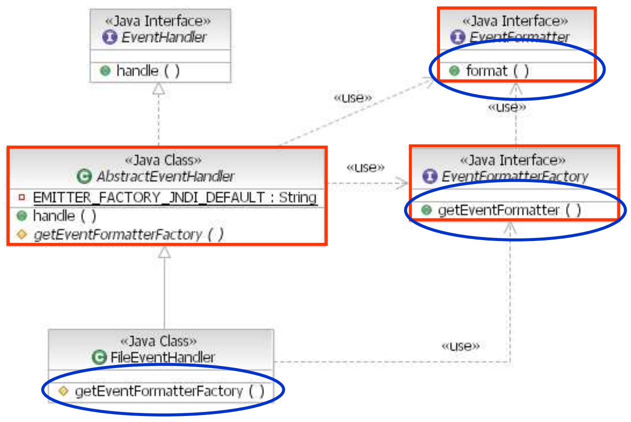
- Logically divided into framework code and emitter-specific code
- Framework code
  - Common functionality logically grouped into a simple framework (*non-production*)
  - Defines interfaces to be implemented or extended by each emitter implementation
    - EventFormatter.format( )
    - EventFormatterFactory.getEventFormatter( )
    - AbstractEventHandler.getEventFormatterFactory( )
- Emitter-Specific code
  - Implements interfaces defined by common framework
  - Implements logic specific to an emitter
    - Retrieval of events from the event source
    - Handling of event emission success or failure

The sample emitters use a design that provides a common framework, and code that is specific to the sample emitter. The framework code includes common function that any emitter could use, although it has not been optimized for production use, so you should review this code carefully before trying to implement this in a production scenario. The framework provides interfaces that you would implement if you wanted to create a new emitter. These interfaces provide functionality to get an event formatter factory, to get the event formatter and to format a common base event. The emitter specific code implements the interfaces that handle retrieving events from the event source and error handling after common base event emission.


IBM Software Group

Overview : Design (cont.)

- Implementation around a common emitter framework provides
  - Consistent flow across all the samples
  - Focus on event extraction and formatting



1. Event occurs, is "detected" and retrieved from the source application
2. EventHandler is asked to handle( ) the retrieved event
3. EventHandler obtains the EmitterFactory from a JNDI lookup
4. EventHandler obtains, from the EventFormatterFactory, the EventFormatter registered to handle the event type
5. EventFormatter is asked to format( ) the event from its "native" format to common base event format
6. EventHandler obtains, from the EmitterFactory, the common event infrastructure emitter
7. EventHandler uses the common event infrastructure emitter to send the common base event

6

Sample event emitters

© 2007 IBM Corporation

This slide shows the classes and interfaces involved in the sample emitter framework. The AbstractEventHandler class is subclassed for the particular event handler that you are implementing. The blue circles identify the three methods that you will need to update for a new emitter.

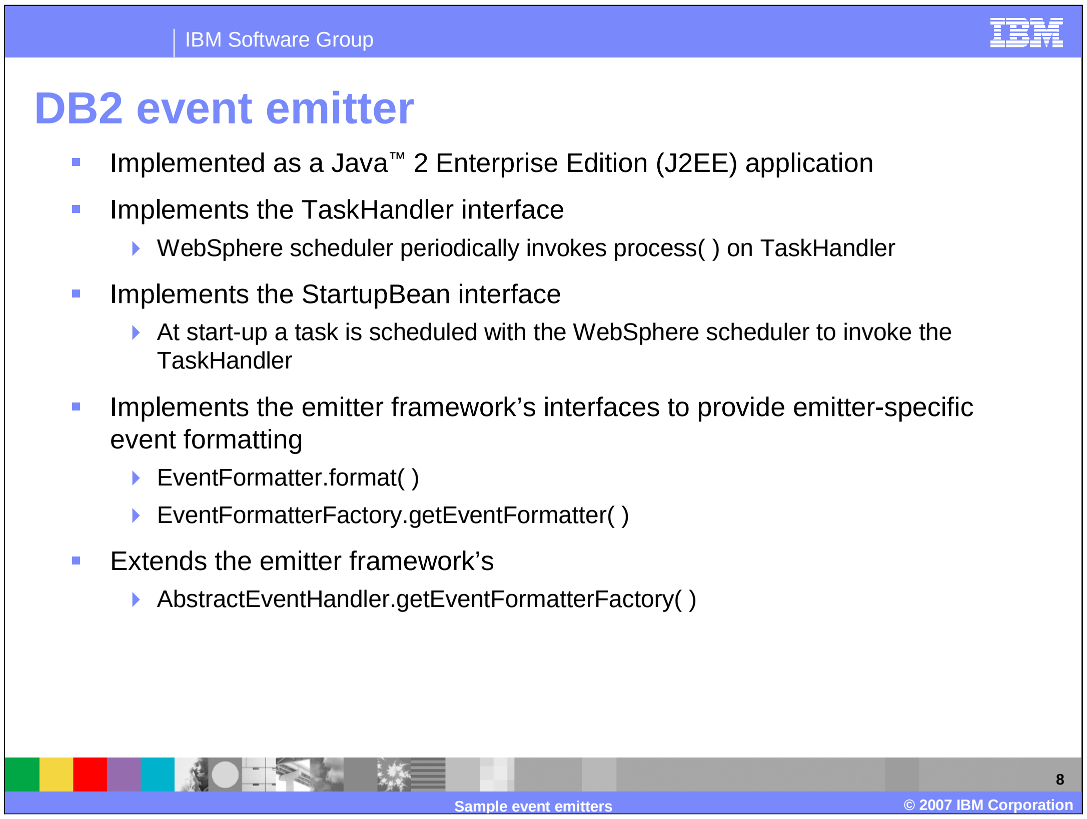
The flow of the emitter is as follows:

1) An event is retrieved from the source.
2) The event handler handles the event.
3) The emitter factory is obtained using a JNDI lookup.
4) The emitter factory provides the event formatter to handle the event type.
5) The event formatter formats the event into a common base event format.
6) The event factory provides the common event infrastructure emitter
7) The common event infrastructure emitter sends the event.

Section

DB2 event emitter

This section introduces the DB2 sample event emitter.


IBM Software Group

DB2 event emitter

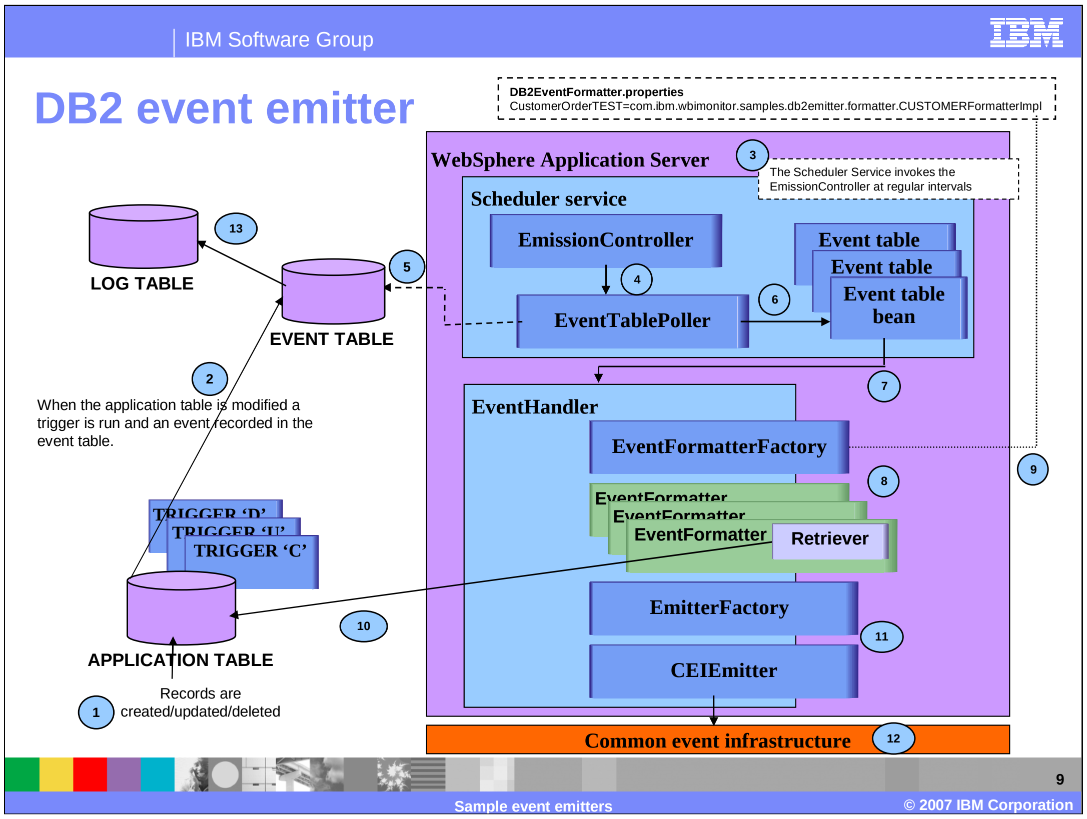
- Implemented as a Java™ 2 Enterprise Edition (J2EE) application
- Implements the TaskHandler interface
  - WebSphere scheduler periodically invokes process( ) on TaskHandler
- Implements the StartupBean interface
  - At start-up a task is scheduled with the WebSphere scheduler to invoke the TaskHandler
- Implements the emitter framework's interfaces to provide emitter-specific event formatting
  - EventFormatter.format( )
  - EventFormatterFactory.getEventFormatter( )
- Extends the emitter framework's
  - AbstractEventHandler.getEventFormatterFactory( )

8

Sample event emitters

© 2007 IBM Corporation

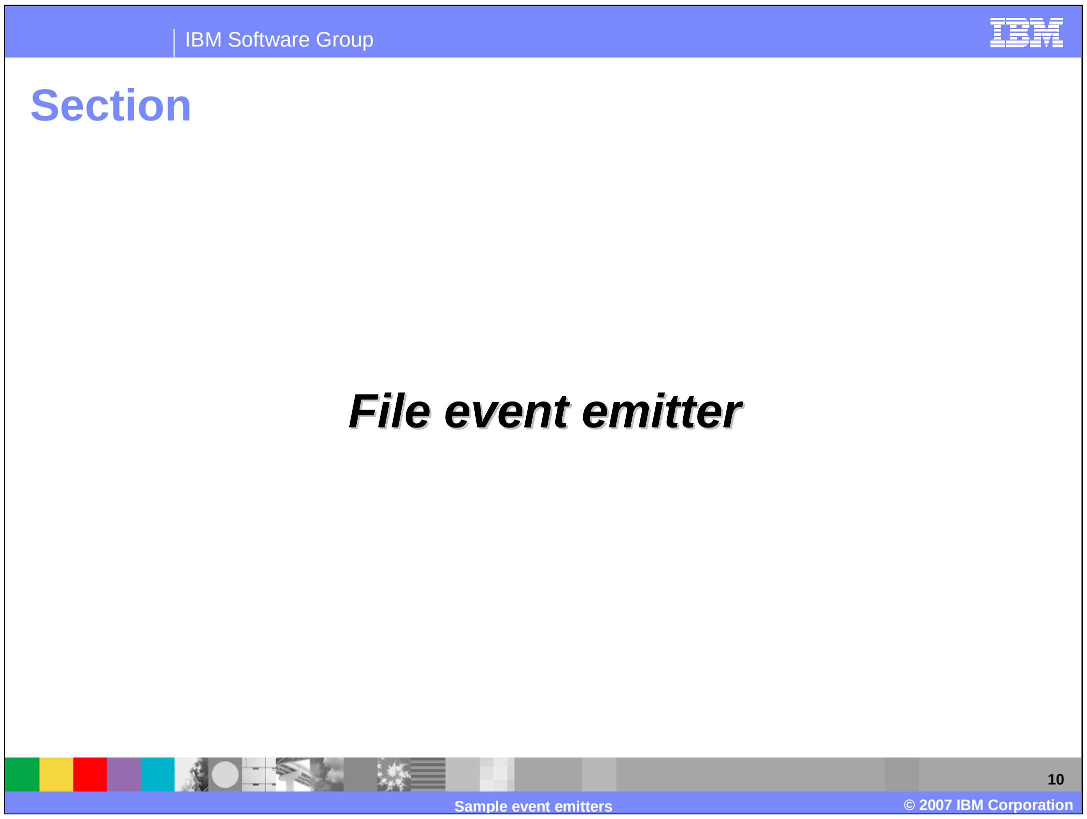
The Sample DB2 Event Emitter is a J2EE enterprise application which also implements the WebSphere TaskHandler interface, allowing it to handle invocations from a WebSphere Application Server scheduler. At start-up, a task is scheduled with the WebSphere Scheduler to invoke the TaskHandler. The DB2 Event Emitter specific code is separated from that of a common framework code. As a result, the DB2 Emitter specific code implements the Emitter Framework's interfaces to provide emitter-specific event formatting.


IBM Software Group
IBM
DB2 event emitter
DB2EventFormatter.properties
CustomerOrderTEST=com.ibm.wbimonitor.samples.db2emitter.formatter.CUSTOMERFormatterImpl
WebSphere Application Server
3
The Scheduler Service invokes the EmissionController at regular intervals
Scheduler service
EmissionController
4
EventTablePoller
6
Event table
Event table
Event table bean
7
EventHandler
EventFormatterFactory
9
8
EventFormatter
EventFormatter
EventFormatter
Retriever
EmitterFactory
11
CEIEmitter
Common event infrastructure
12
13
LOG TABLE
5
EVENT TABLE
2
When the application table is modified a trigger is run and an event recorded in the event table.
TRIGGER 'D'
TRIGGER 'U'
TRIGGER 'C'
10
APPLICATION TABLE
Records are created/updated/deleted
1
9
Sample event emitters
© 2007 IBM Corporation

This diagram shows the flow of the DB2 event emitter.

1. A record is created, updated or deleted on an APPLICATION table.
2. The DB2 TRIGGER inserts a record with the primary key and additional information to the EVENT table. The additional information includes the trigger type (Create / Update / Delete), the application data type, and the creation timestamp.
3. The Scheduler service in the application server invokes the EmissionController at a specified interval.
4. The EmissionController calls the EventTablePoller.
5. If any records have been inserted into the EVENT table, the EventTablePoller populates.
6. The populated EventTablePoller returns a collection of EventTableBean objects.
7. If the collection of EventTableBean objects is not empty, the EmissionController invokes the handle( ) method of the EventHandler for each EventTableBean object to handle the emission steps.
8. The EventHandler calls the EventFormatterFactory which returns an EventFormatter object according to the application data type specified by the user.
9. The EventFormatter registers to handle the Event Type, in this example CustomerOrderTEST
10. The Retriever class retrieves the data from the APPLICATION Table
11. The EmitterFactory invokes CEIEmitter
12. The CEIEmitter sends the CommonBaseEvent object to common event infrastructure event server.
13. Finally, the event record in the EVENT table is moved to the LOG table

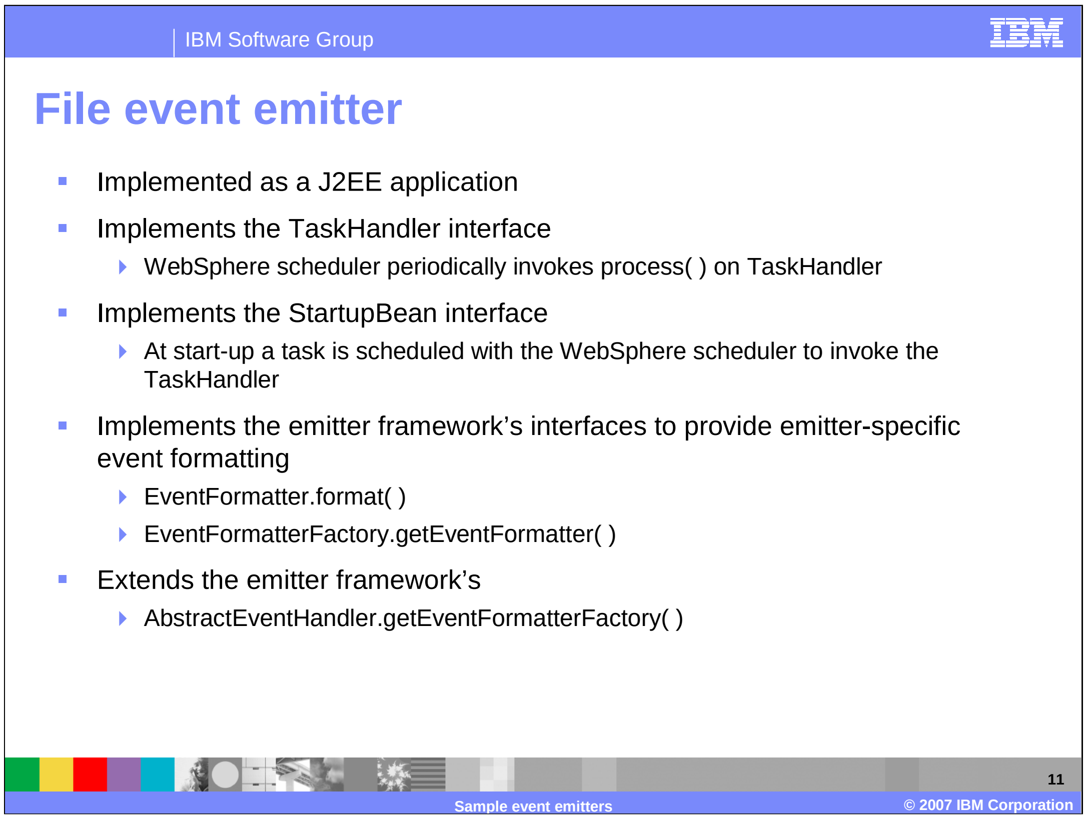
Note: Since there is a latency between the time that the trigger logs the table update and the time that the EventRetriever extracts the data from the table, it is possible for event data to get out-of-synch. Multiple updates to the application table may occur before the poller wakes up. In this case, there would be multiple events produced but they would all contain the latest application data, and any intermediate changes would be lost. Therefore, the formatter should only include data in the event that either cannot change or where the latest value is needed.



Section

File event emitter

This section introduces the file sample event emitter.



IBM Software Group

File event emitter

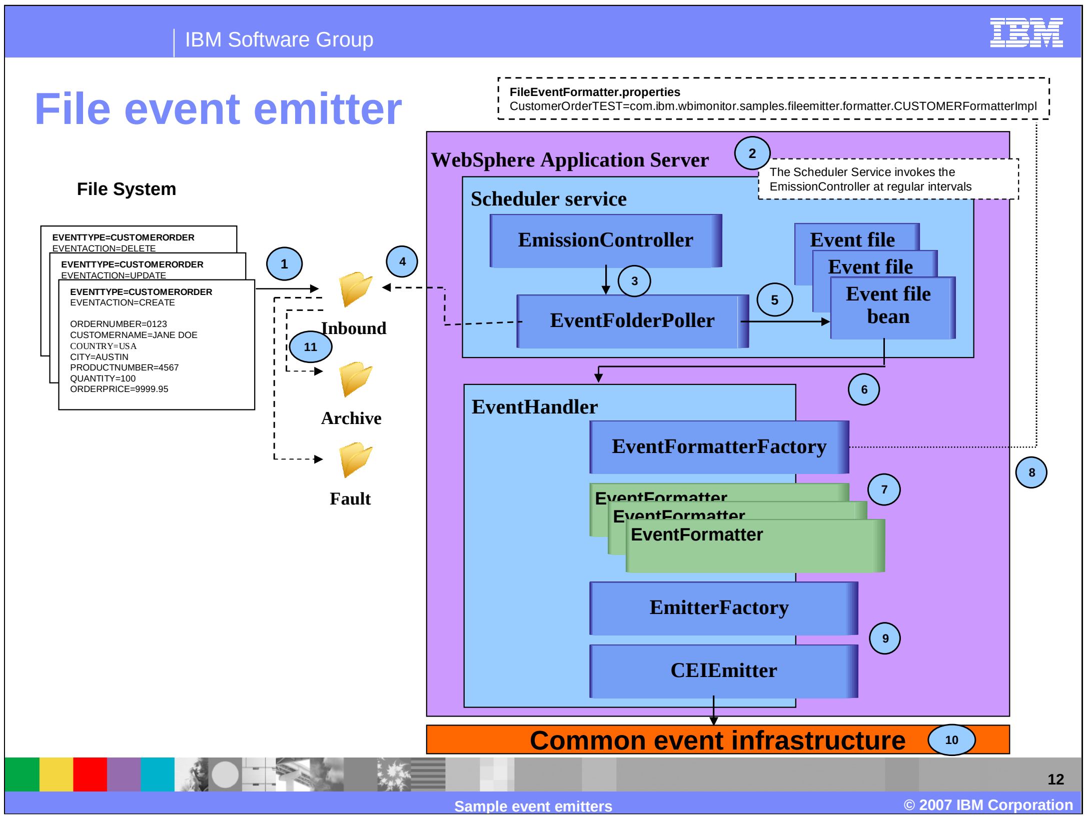
- Implemented as a J2EE application
- Implements the TaskHandler interface
  - WebSphere scheduler periodically invokes process( ) on TaskHandler
- Implements the StartupBean interface
  - At start-up a task is scheduled with the WebSphere scheduler to invoke the TaskHandler
- Implements the emitter framework's interfaces to provide emitter-specific event formatting
  - EventFormatter.format( )
  - EventFormatterFactory.getEventFormatter( )
- Extends the emitter framework's
  - AbstractEventHandler.getEventFormatterFactory( )

11

Sample event emitters

© 2007 IBM Corporation

The Sample File Event Emitter is a J2EE enterprise application which also implements the WebSphere TaskHandler interface, allowing it to handle invocations from a WebSphere Application Server scheduler. At start-up a task is scheduled with the WebSphere Scheduler to invoke the TaskHandler. The file event emitter specific code is separated from that of a common framework code. As a result, the file emitter specific code implements the emitter framework's interfaces to provide emitter-specific event formatting.



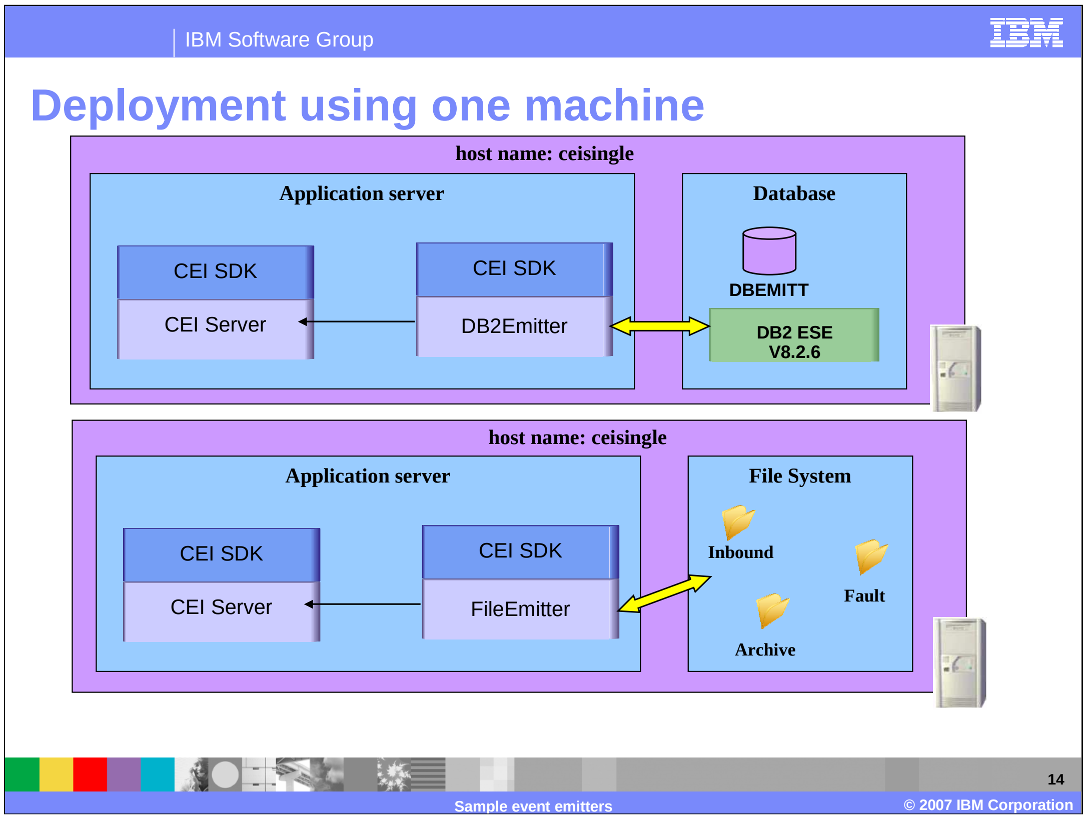
This diagram shows the flow of the file event emitter.

1. A file encapsulating information about a create, update or delete event is placed in the inbound directory. This file contains information about what data type the event pertains to, the actual event type (create, update or delete) and the attributes and their associated values.
2. The Scheduler service in WebSphere Process Server invokes the EmissionController at a specified interval.
3. The EmissionController calls the EventFolderPoller.
4. If any files are found in the inbound directory, the EventFolderPoller populates.
5. The populated EventFolderPoller returns a collection of EventFileBean objects.
6. If the collection of EventFileBean objects is not empty, the EmissionController invokes the handle( ) method of the EventHandler for each EventFileBean object to handle the emission steps.
7. The EventHandler calls the EventFormatterFactory which returns an EventFormatter object according to the application data type specified in the event file.
8. The EventFormatter registers to handle the Event Type, in this example CustomerOrderTEST
9. The EmitterFactory invokes the CEIEmitter
10. The CEIEmitter sends the CommonBaseEvent object to the event server.
11. On a successful event emission, the file that was copied to the Inbound directory is moved to the Archive directory. On a failed event emission, the file is moved to the Fault directory.

Section

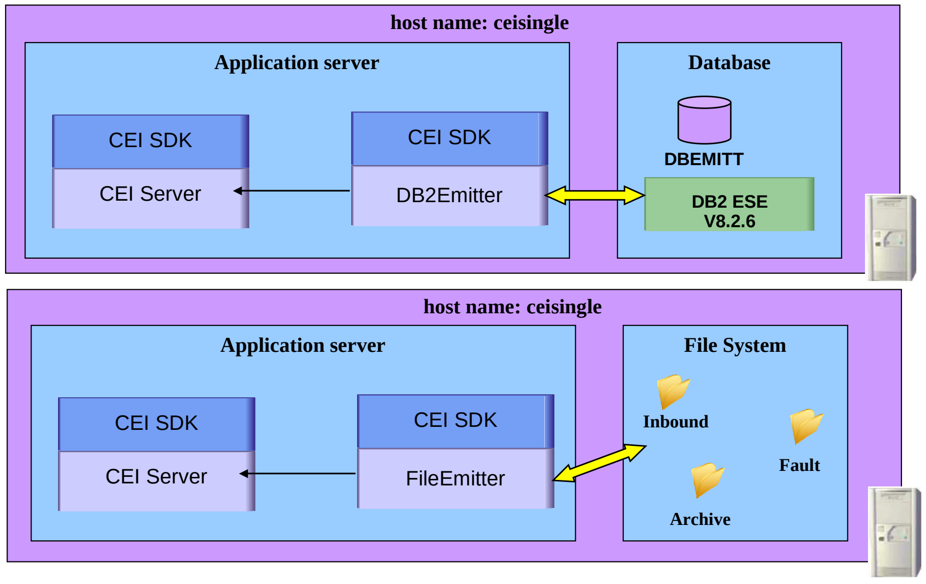
Deployment

This section reviews the deployment options for the sample event emitters



IBM Software Group

Deployment using one machine

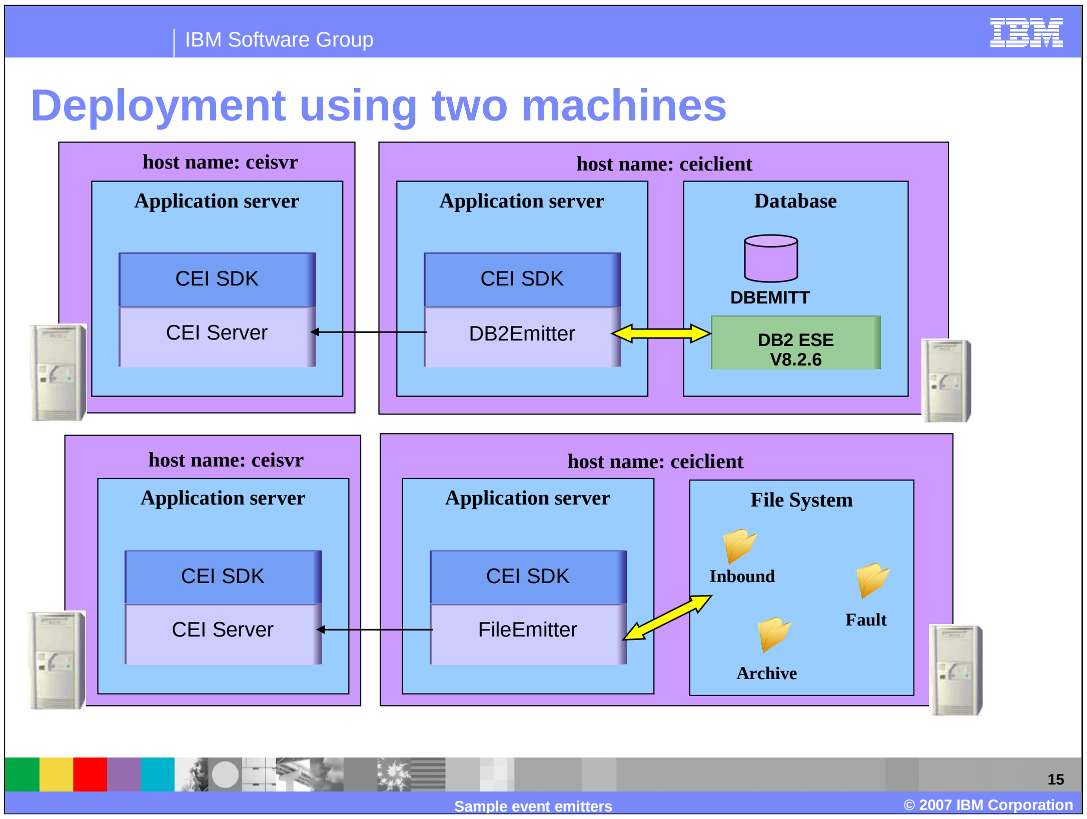


14

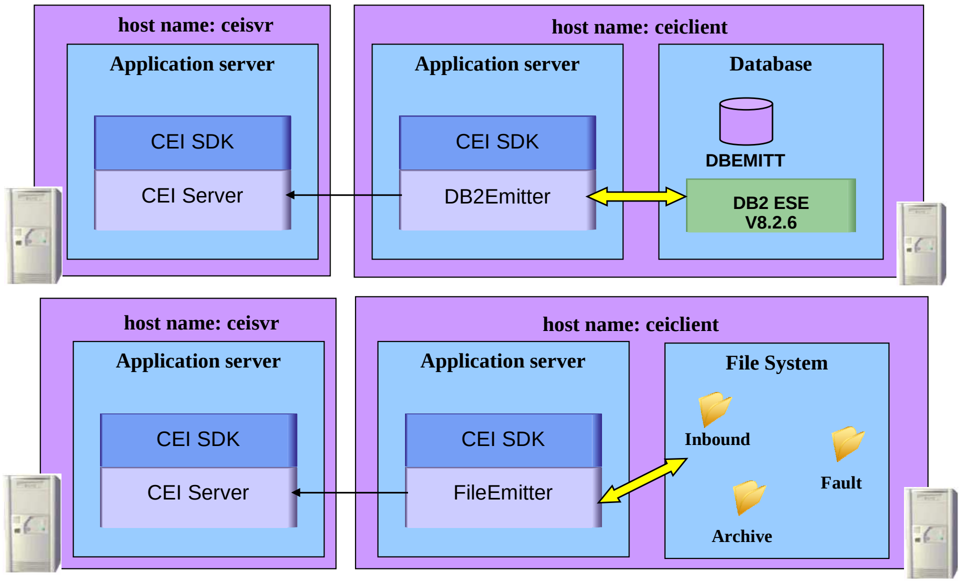
Sample event emitters

© 2007 IBM Corporation

The common event infrastructure SDK is provided in WebSphere Process Server or WebSphere Business Integration Server Foundation version 5.1 or WebSphere Application Server version 6.1. This diagram shows using the common event infrastructure server on the same machine as the sample event emitter. The sample event emitter retrieves the information from DB2 or the file system, formats the event then uses the common event infrastructure SDK APIs to send the common base events to the common event infrastructure server which is in the same JVM.

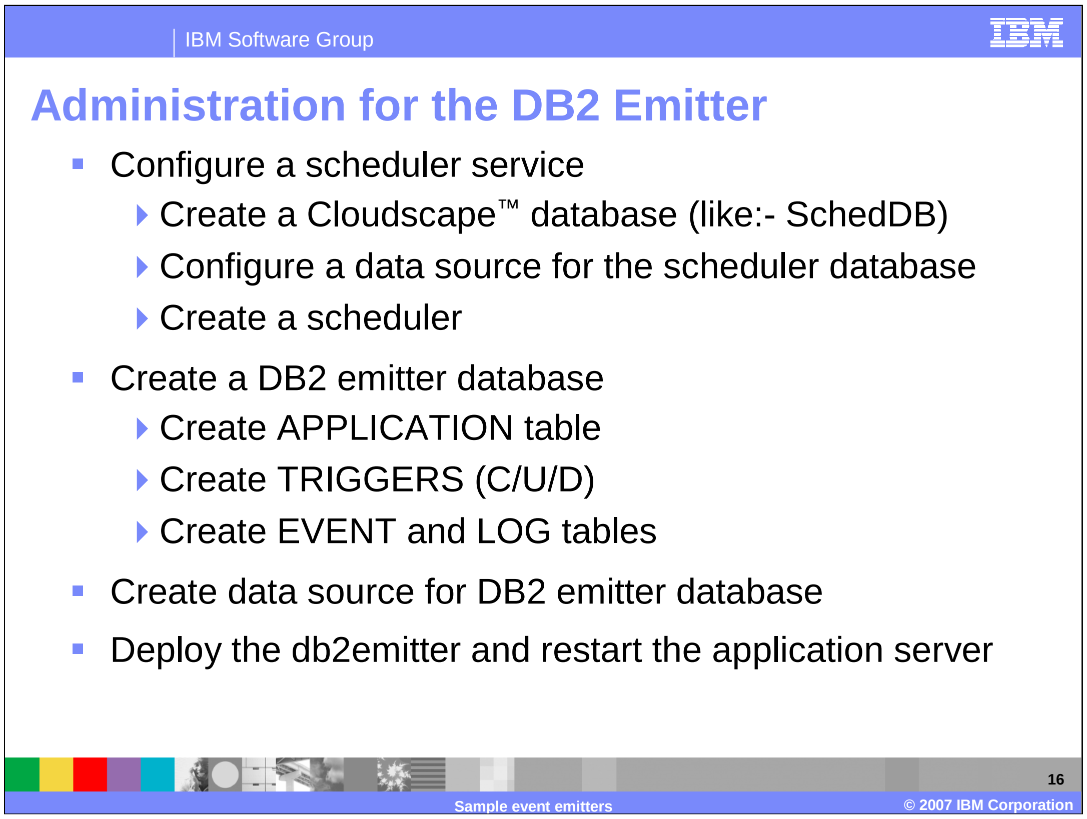


Deployment using two machines



This diagram shows using the common event infrastructure server on a remote machine from the sample event emitter. The sample event emitter retrieves the information from DB2 or the file system, formats the event then uses the common event infrastructure SDK APIs to send the common base events to the common event infrastructure server which is on a different server.

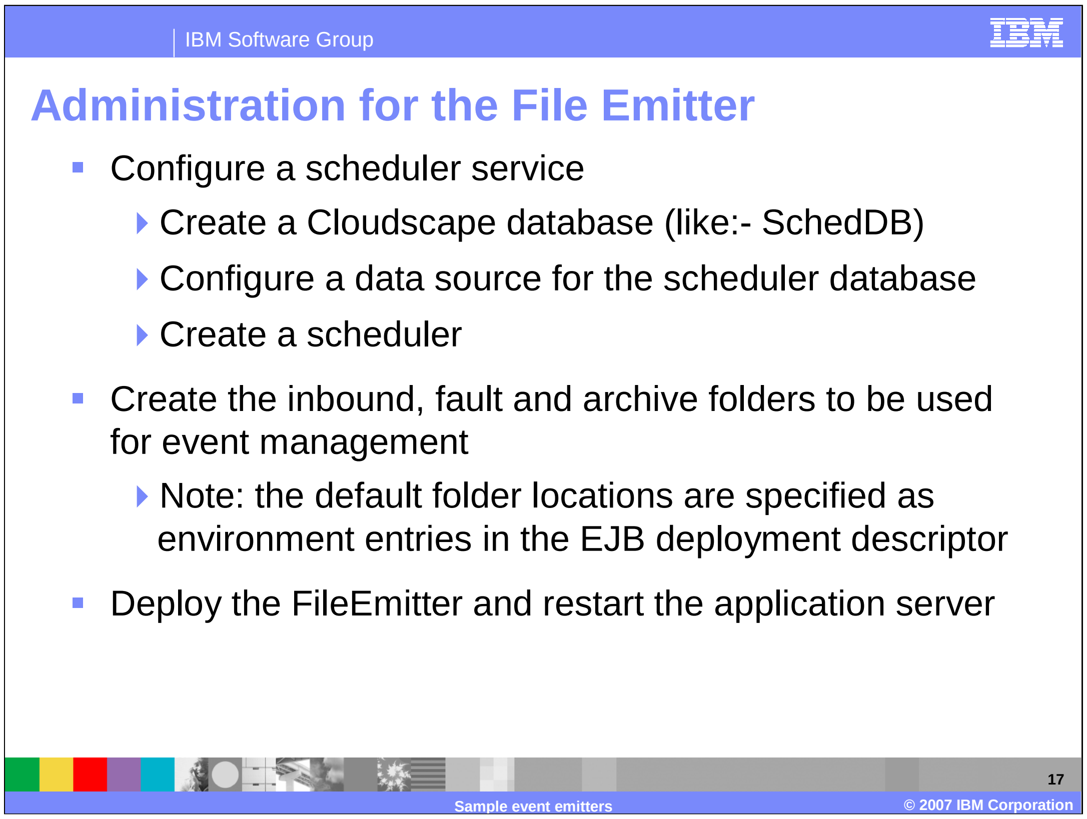
If your common event infrastructure server is on a remote server, then you will need to configure the service integration bus to handle the link between servers.



Administration for the DB2 Emitter

- Configure a scheduler service
  - Create a Cloudscape™ database (like:- SchedDB)
  - Configure a data source for the scheduler database
  - Create a scheduler
- Create a DB2 emitter database
  - Create APPLICATION table
  - Create TRIGGERS (C/U/D)
  - Create EVENT and LOG tables
- Create data source for DB2 emitter database
- Deploy the db2emitter and restart the application server

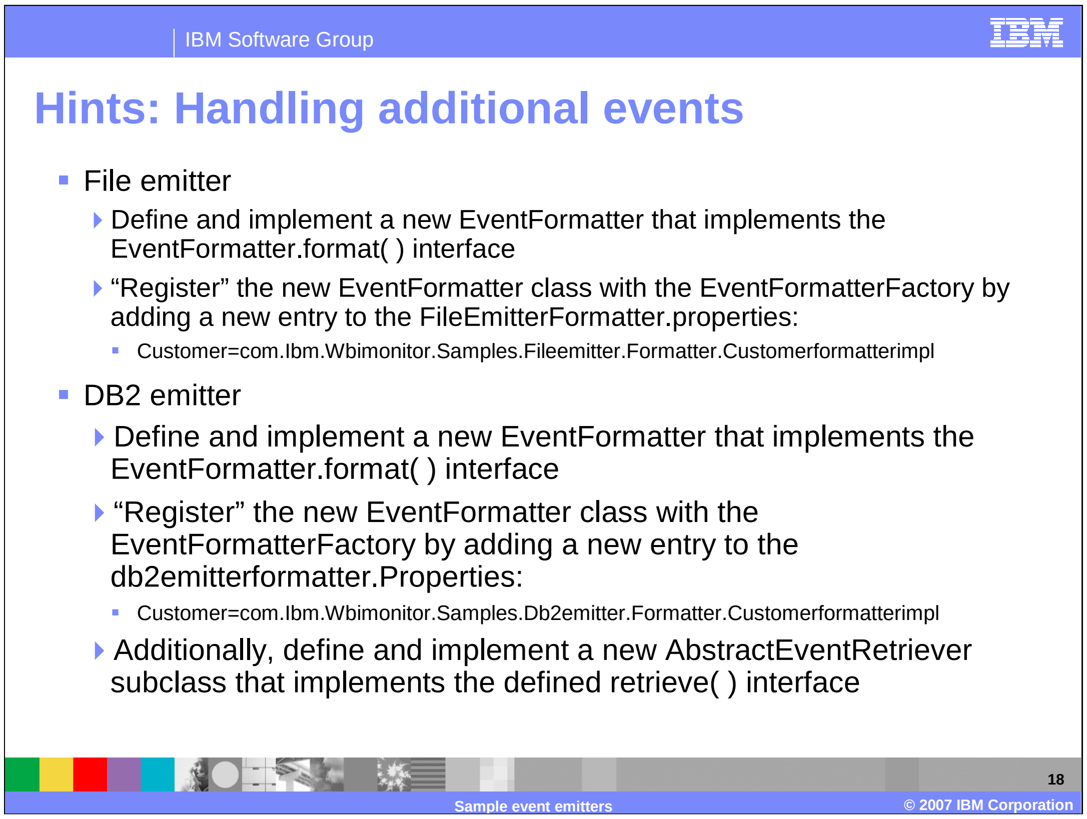
For the DB2 emitter, you will need to configure the scheduler, which includes creating a Cloudscape database, configuring the data source and creating the scheduler. You will need to create the DB2 emitter database, which includes creating the application table, the triggers, the event table, the log table, and the data source. Finally, you will deploy the DB2 emitter application and restart the server.



Administration for the File Emitter

- Configure a scheduler service
  - Create a Cloudscape database (like:- SchedDB)
  - Configure a data source for the scheduler database
  - Create a scheduler
- Create the inbound, fault and archive folders to be used for event management
  - Note: the default folder locations are specified as environment entries in the EJB deployment descriptor
- Deploy the FileEmitter and restart the application server

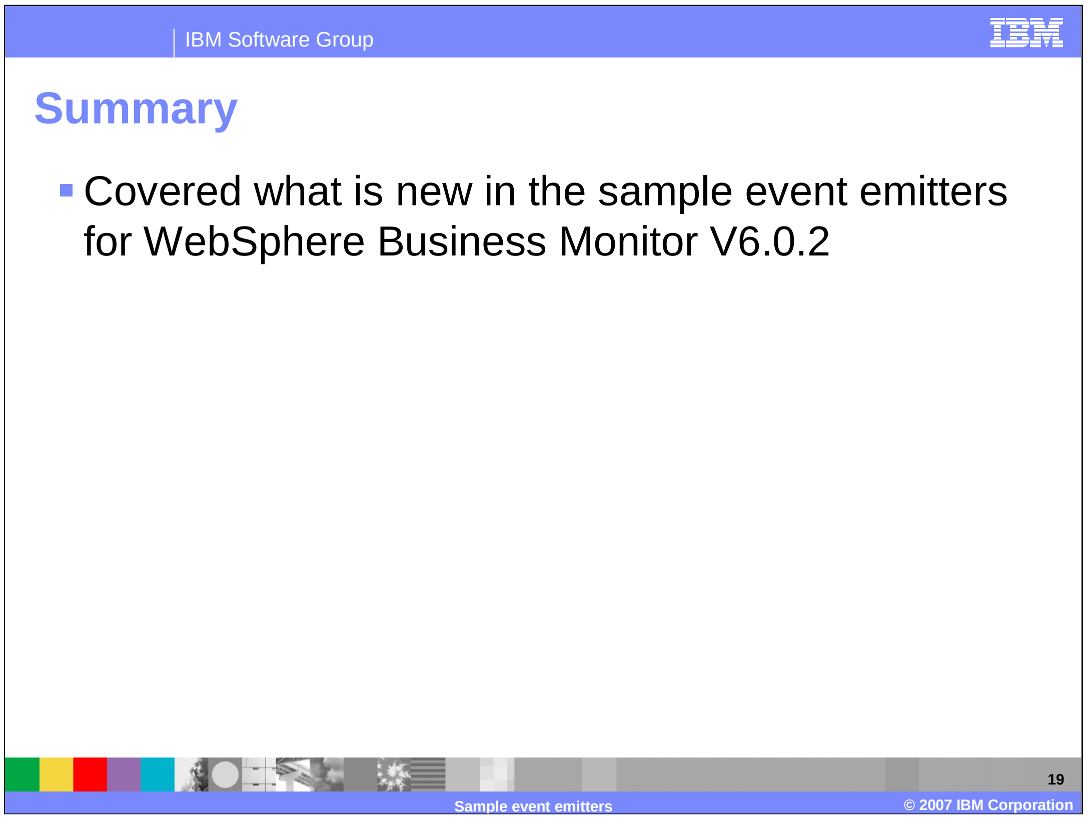
For the file emitter, you will need to configure the scheduler, which includes creating a Cloudscape database, configuring the data source and creating the scheduler. You will need to create three folders for storing the files. The folder location can be changed and is specified in the EJB deployment descriptor. Finally, you will deploy the file emitter application and restart the server.



Hints: Handling additional events

- File emitter
  - Define and implement a new EventFormatter that implements the EventFormatter.format( ) interface
  - "Register" the new EventFormatter class with the EventFormatterFactory by adding a new entry to the FileEmitterFormatter.properties:
    - Customer=com.Ibm.Wbimonitor.Samples.Fileemitter.Formatter.Customerformatterimpl
- DB2 emitter
  - Define and implement a new EventFormatter that implements the EventFormatter.format( ) interface
  - "Register" the new EventFormatter class with the EventFormatterFactory by adding a new entry to the db2emitterformatter.Properties:
    - Customer=com.Ibm.Wbimonitor.Samples.Db2emitter.Formatter.Customerformatterimpl
  - Additionally, define and implement a new AbstractEventRetriever subclass that implements the defined retrieve( ) interface

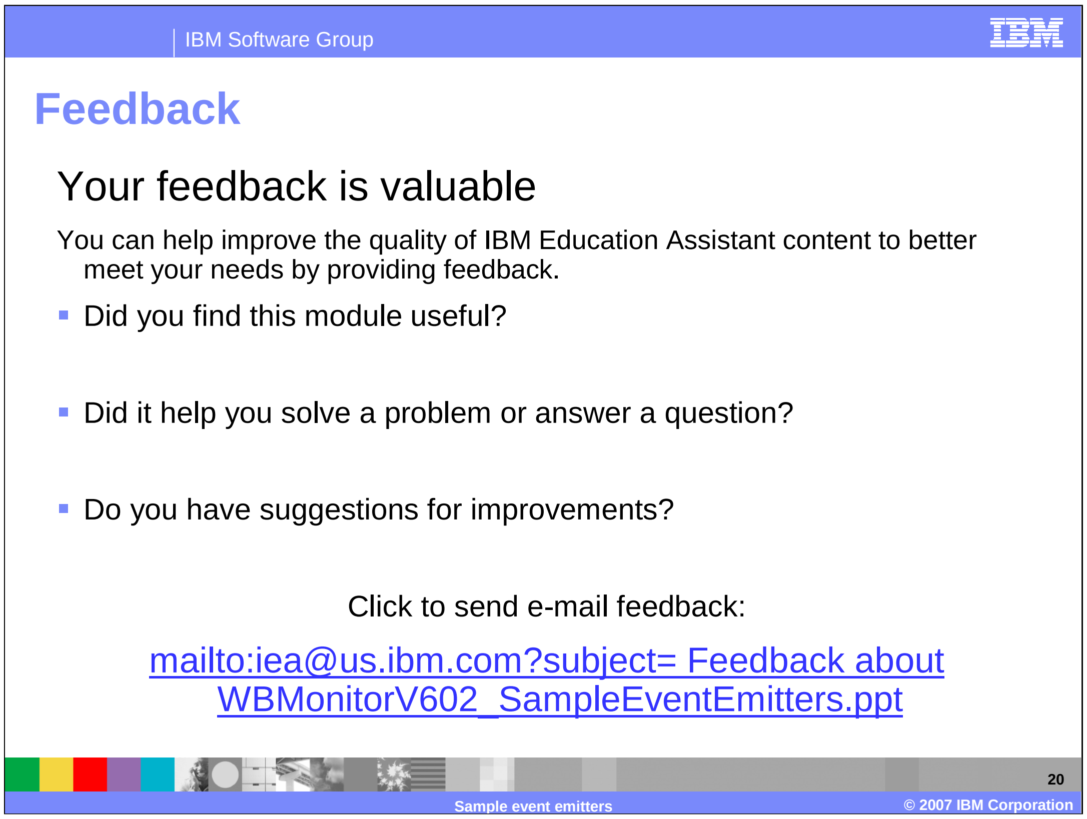
The supplied sample event emitters can be updated to handle any event types that you would like to process. For the file emitter and DB2 emitter, you will need to implement a new event formatter to format the common base event. And you will need to register your event type by adding it to the file emitter properties file. Additionally, for the DB2 emitter, you will need to implement a new retrieve interface to create the SQL to retrieve the data from your application table.



Summary

- Covered what is new in the sample event emitters for WebSphere Business Monitor V6.0.2

In summary, you have reviewed the two sample event emitters that are supplied with WebSphere Business Monitor version 6.0.2.


IBM Software Group

IBM

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

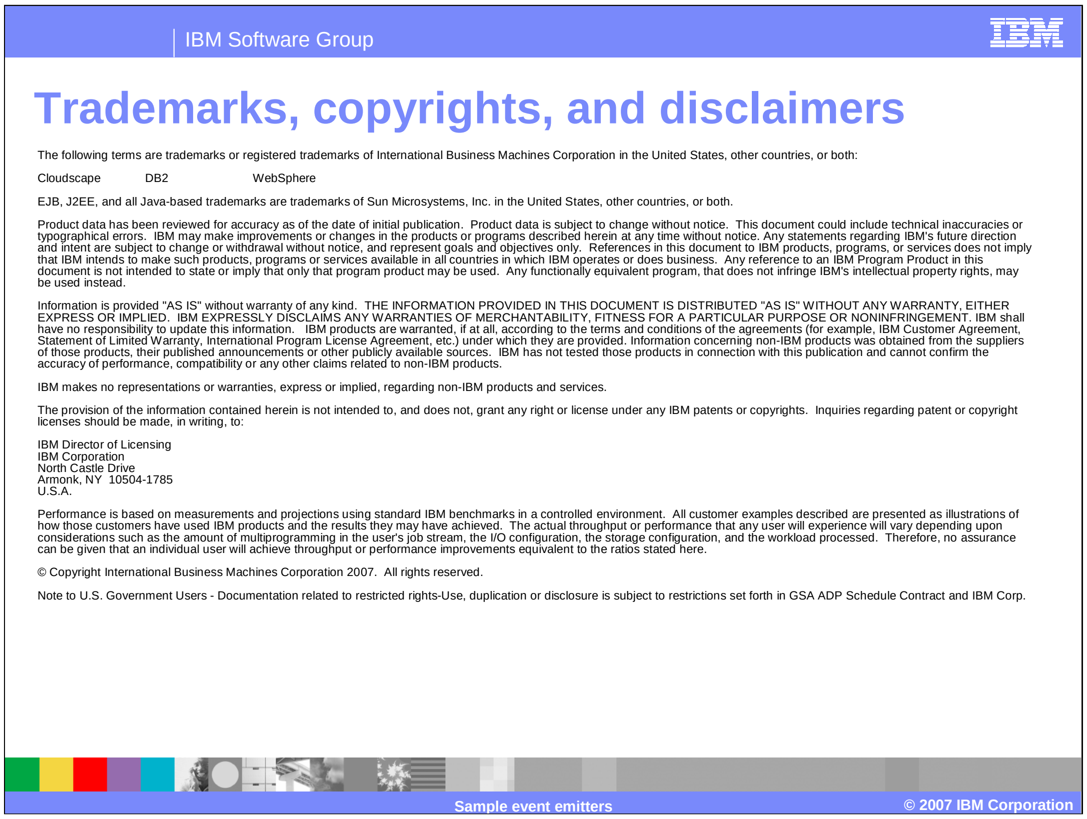
mailto:iea@us.ibm.com?subject= Feedback about WBMonitorV602_SampleEventEmitters.ppt

20

Sample event emitters

© 2007 IBM Corporation

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



Trademarks, copyrights, and disclaimers