



Lotus Expeditor 6.1 Education

# IBM® Lotus® Expeditor 6.1 Client for Desktop

## Property broker and event administration

**Lotus** software



@business on demand software

© 2006 IBM Corporation

Welcome to this presentation, which explains the Property Broker in IBM Lotus Expeditor 6.1 Client for Desktop.

## Goals

- Understand the property broker and event administration support provided in IBM Lotus Expeditor Client for Desktop

The goal of this presentation is to help you understand the property broker and event administration provided in IBM Lotus Expeditor 6.1 Client for Desktop.

## Agenda

- Property Broker
- Event Administration

The agenda of this presentation is to explain the Property Broker and Event Administration capabilities which the client platform provides to you, the infrastructure and plug-ins that enable these capabilities, and details about the property broker and event administration supported by IBM Lotus Expeditor 6.1 Client for Desktop.

## Section

# *Property broker*

So, let's describe more details about the Property Broker

## Property broker

Property Broker

- The client property broker is a broker that allows for declarative properties, actions, and wires to be used among completely decoupled components.
- The property broker is responsible for taking changed properties and distributing the property values to the appropriate actions as defined by the wires that are registered.
- The property broker separates itself from a traditional pub/sub in that it is a controlled pub/sub that is driven by declarative markup
  - ▶ Meaning an XML or another data source defines how the two components communicate with each – which property change is passed onto which action within the component.
- The second differentiation is the chain effect that can be accomplished by taking input parameters and posting output parameters.
  - ▶ This ability allows for an infinite amount of Property > Action > Property combinations.

The client property broker is a broker that allows for declarative properties, actions, and wires to be used among completely decoupled components. The property broker is responsible for taking changed properties and distributing the property values to the appropriate actions as defined by the wires that are registered. The property broker separates itself from a traditional pub/sub in that it is a controlled pub/sub that is driven by declarative markup. Meaning an XML or another data source defines how the two components communicate with each other – which property change is passed onto which action within the component. The second differentiation is the chain effect that can be accomplished by taking input parameters and posting output parameters. This ability allows for an infinite amount of Property > Action > Property combinations.

## Property broker

Property Broker

- Components that contribute to the broker most likely do not call into the API's directly, and instead use the extension point and the declarative WSDL to declare its actions and properties.
- The preferred model allows little knowledge of the broker and its API's providing a good level of abstraction from the broker implementation.
- At most, a component calls into the broker to post a changed property and then performs an evaluation of the received property changes to complete the action on the changed property.

Components that contribute to the broker most likely do not call into the API's directly, and instead use the extension point and the declarative WSDL to declare its actions and properties. The preferred model allows little knowledge of the broker and its API's, providing a good level of abstraction from the broker implementation. At most, a component calls into the broker to post a changed property and then performs an evaluation of the received property changes to complete the action on the changed property.

## Property broker

Property Broker

- Lastly, the client side broker was designed after the IBM Portal property broker, but includes a level of flexibility above and beyond the portal broker.
- The client broker allows for different kinds of components to be contributing to the broker; different components as in SWT, AWT, Eclipse commands, OSGI Event Admin, ...
- This allows for currently established actions to participate in broker communications.
- The framework allows for new handlers to be defined using an Eclipse extension point; giving the broker complete adaptability and compatibility with other messaging systems.

Lastly, the client side broker was designed after the IBM Portal property broker, but includes a level of flexibility above and beyond the portal broker. The client broker allows for different kinds of components to be contributing to the broker; different components as in SWT, AWT, Eclipse commands, OSGI Event Admin, and so on. This allows for currently established actions to participate in broker communications. The framework allows for new handlers to be defined using an Eclipse extension point; giving the broker complete adaptability and compatibility with other messaging systems.

## Section

# ***Event administration***

So, let's describe more details about the Event Administration



## OSGi event administration

OSGi  
Event Admin

- Bundles wishing to publish events should obtain the event administration service and call one of the event delivery methods.
- Events are posted through the event administration service, either synchronously or asynchronously
- API:
  - ▶ `org.eclipse.equinox.event`
- Target feature:
  - ▶ Developer's Guide
- Reference:
  - ▶ <http://www.osgi.org>

Nearly all the bundles in an OSGi framework must deal with events, either as an event publisher or as an event handler. The Event Admin service provides an inter-bundle communication mechanism. It is based on an event publish and subscribe model, popular in many message based systems.

## 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 Lotus

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 or changes in the products or programs 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 (for example, 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 2006. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.



This concludes the presentation