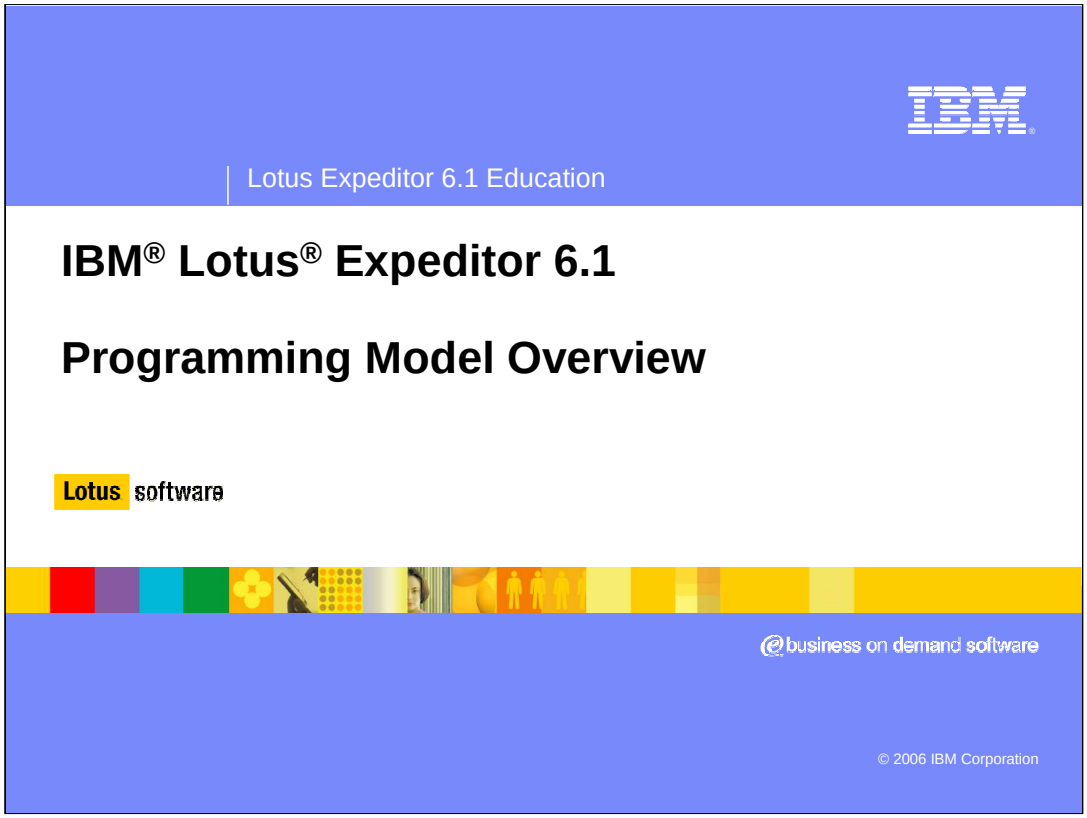
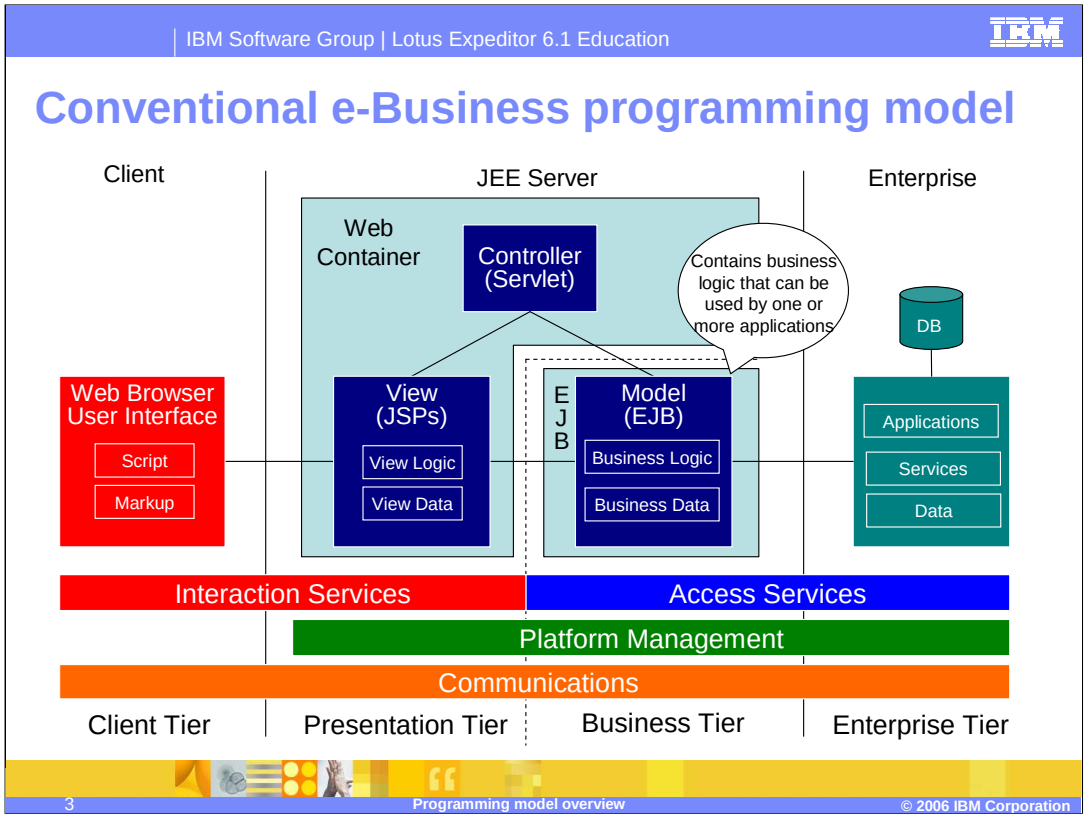


This presentation provides you with an overview of the programming model supported by Lotus Expeditor 6.1.

Goals

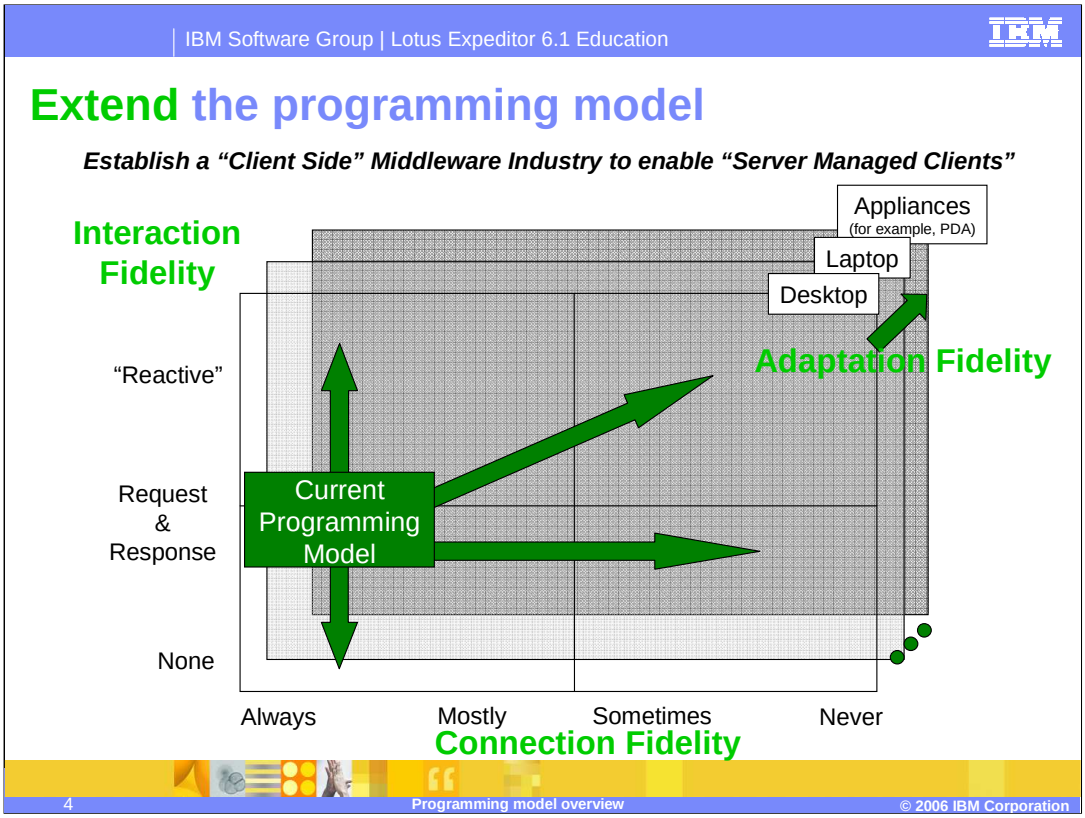
- To understand the programming model supported by IBM Lotus Expeditor 6.1
- To understand the types of applications you can develop with IBM Lotus Expeditor 6.1

The goals of this presentation are to understand the programming model supported by IBM Lotus Expeditor 6.1 and the types of applications you can develop with Lotus Expeditor as a result of this programming model.



In the conventional e-business end-to-end programming model, also known as the Java™ Enterprise Edition (or JEE) Web programming model, an end-to-end application consists of components distributed across multiple tiers and nodes, where each node is a physical platform such as a device or server. These components can be logically classified by using the popular model-view-controller (or MVC) pattern. In MVC, a *model* represents business data and the business logic that manipulates that data, a *view* interacts with the model on behalf of a user or other external interaction, and a *controller* coordinates the flow of interaction between the view and the model.

The components of the application run in *containers* located on *tiers* as defined by the JEE architecture. Web applications run in a *Web Container* on the *Presentation Tier* and Enterprise JavaBeans (or EJBs) run in an *EJB container* on the *Business Tier*. Containers can provide *Interaction Services* that enable JEE components to interact with users and *Access Services* that enable JEE components to access Enterprise applications, services, and data.



The conventional JEE programming model works fine in an environment where users interact with a Web browser on a highly functional desktop (or equivalent) device to send requests to and receive responses from Web applications over a relatively reliable, high bandwidth network. However, what about users who work in a much more diverse environment in which these fundamental assertions do not hold true?

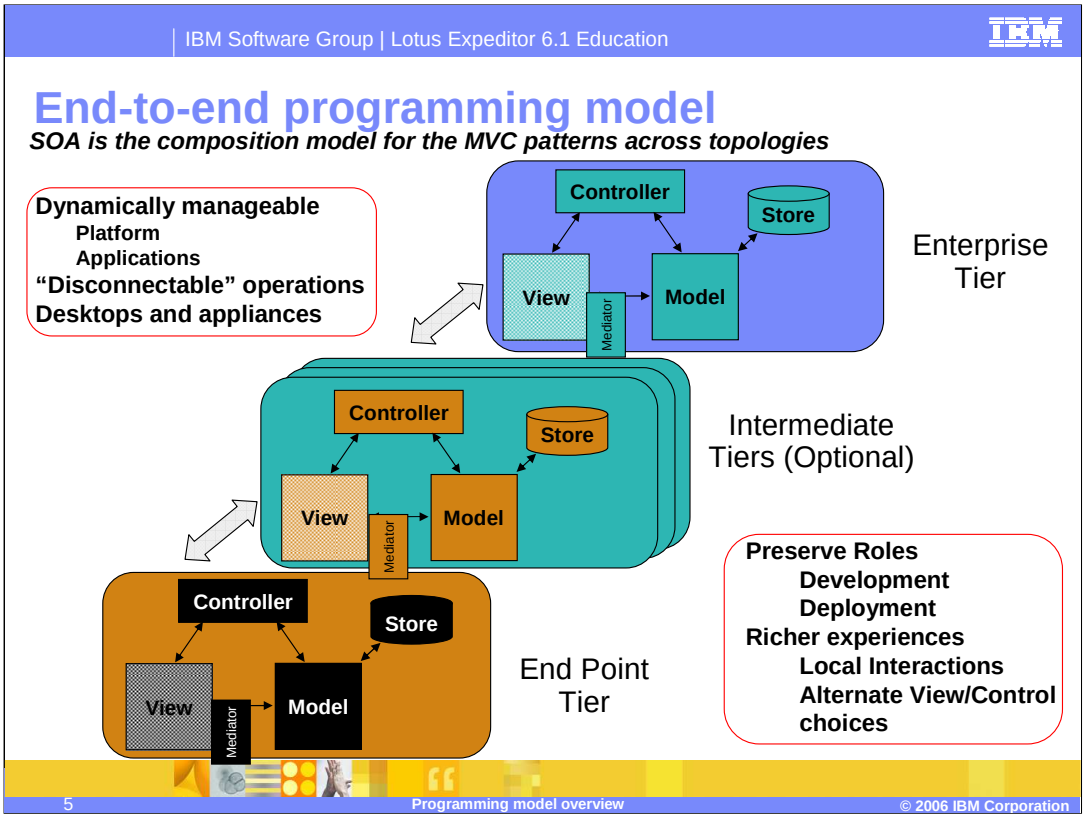
IBM's Server Managed Client technology provides a solution by extending the programming model to client devices so that JEE developers can extend their existing applications to, and create new applications for, a wide range of devices, connection states, and user experiences while allowing them to use their existing skills.

Extending the programming model leads to unique requirements in the following three areas:

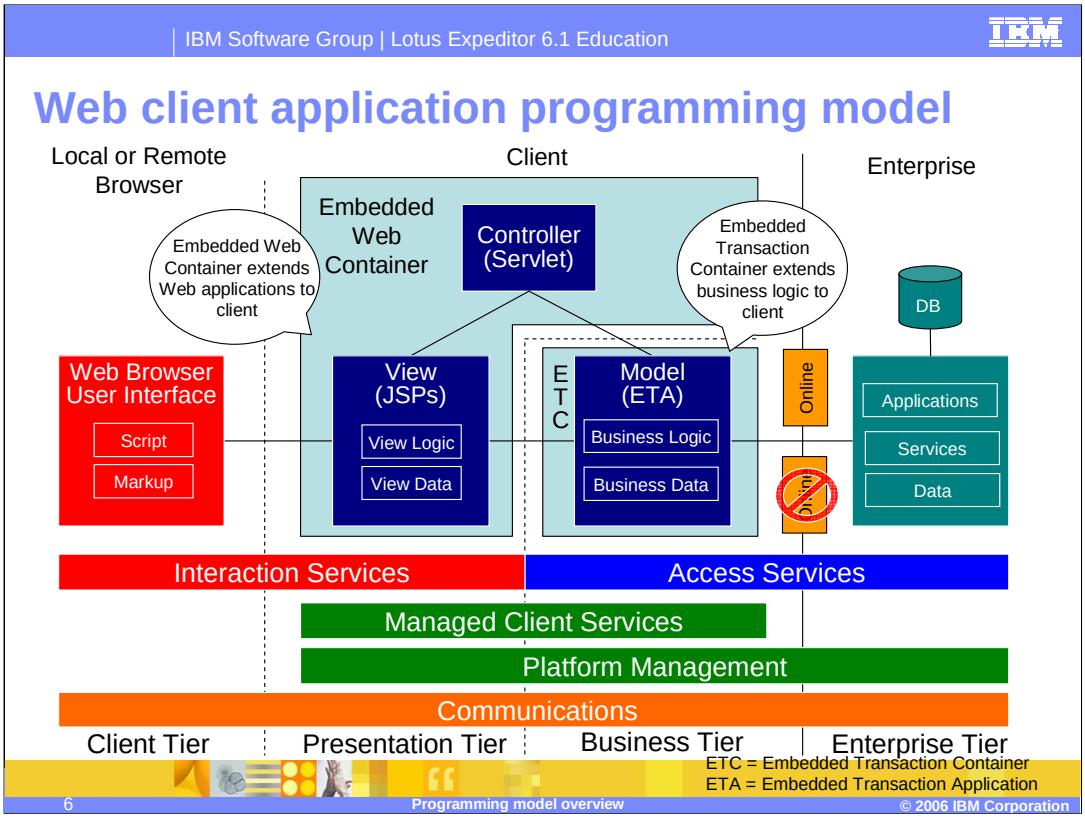
Connection Fidelity – The programming model must enable applications to operate across states that are always, mostly, occasionally, and never connected. Network characteristics determine the active connection state. These characteristics range from fully connected, high bandwidth connections such as hardwired Ethernet to very low bandwidth connections such as today's 2G cellular networks. Network characteristics also include latency, geographic coverage, and even billing models.

Interaction Fidelity – The programming model must continue to support the dominant Web browser-based request and response user experience. However, the programming model must also support other user experiences that are increasingly becoming important. Reactive user experiences include rich graphical user interfaces (GUIs) that directly utilize the native widgets of an operating system and multimodal browsers that combine visual and voice interactions in a single user experience. For embedded devices, no user experience is available; however, the programming model must enable applications to run on those devices.

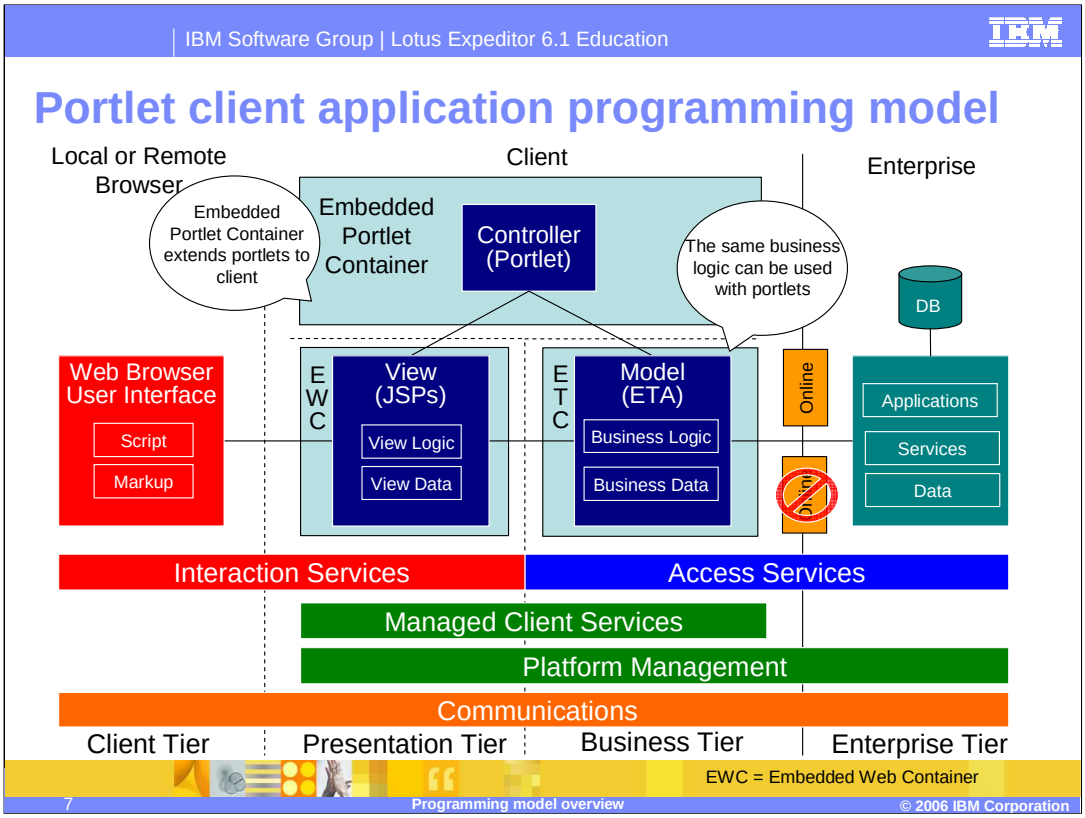
Adaptation Fidelity – The programming model must address a wide range of client devices covering desktops, laptops, tablets, and appliances, which include embedded controllers, smart phones and PDAs.



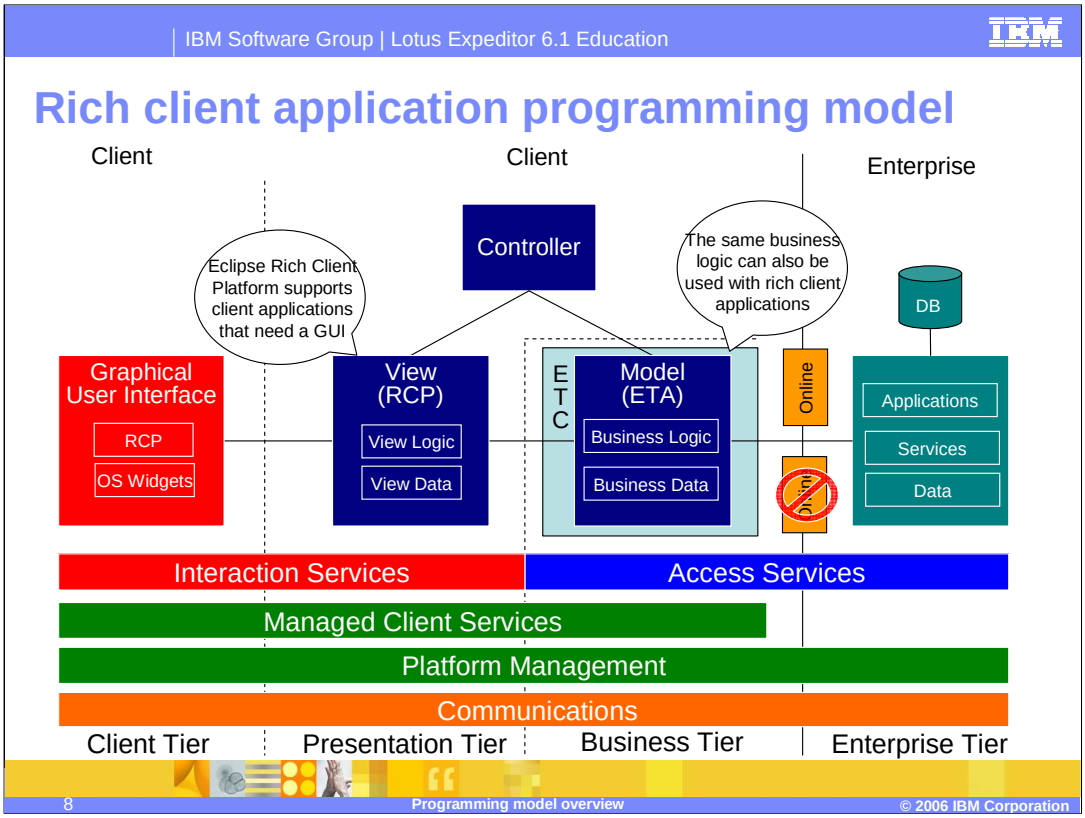
IBM's Server Managed Client technology defines a programming model that addresses these requirements through a set of containers and services that enable applications to operate in this environment. This includes the capability to distribute key components of applications to the client, and possibly intermediate tiers, so users can continue to perform selected business operations with these applications even when their device is offline. Moving application components to the client can also result in improved performance through local interactions between the end-user and their application, and alternate view/control choices, such as a graphical user interface. By preserving a selected set of JEE APIs and roles, developers can reuse their skills and experience to develop client applications. Furthermore, IBM Lotus Expeditor 6.1 enables you to compose end-to-end applications through its support for a Service-Oriented Architecture (SOA).



IBM Lotus Expeditor 6.1 implements this client programming model by providing an *Embedded Web Container* and an *Embedded Transaction Container (ETC)* with services that enable Web applications and Embedded Transaction Applications respectively to run on clients. These containers and services support selected APIs from the JEE programming model to facilitate reuse of components and skills, while also supporting online and offline operations. The Embedded Web Container extends the JEE and WebSphere® programming model to the client by supporting standard Web applications that comply with the Servlet 2.3 and JSP 1.2 specifications as well as the Servlet 2.4 and JSP 2.0 specifications. The Embedded Transaction Container further extends the symmetry of the client programming model with the JEE and WebSphere programming model by supporting a subset of the EJB 2.0 specification. You also have the flexibility to implement business logic as beans, Eclipse plug-ins, or OSGi services (not shown). As a testament to the power of this programming model, an IBM business partner had their Web application running on the client platform and supporting mobile users with laptops in a matter of weeks.



The client platform also enables you to deliver client-side portlets that conform to JSR 168 through support of an *Embedded Portlet Container*. JSR 168 standardizes the development of portal applications, called *portlets*. The specification defines a common Portlet API and infrastructure that provides facilities for personalization, presentation, and security. You may continue the use of business logic components, such as Embedded Transaction Applications, with your portlets.

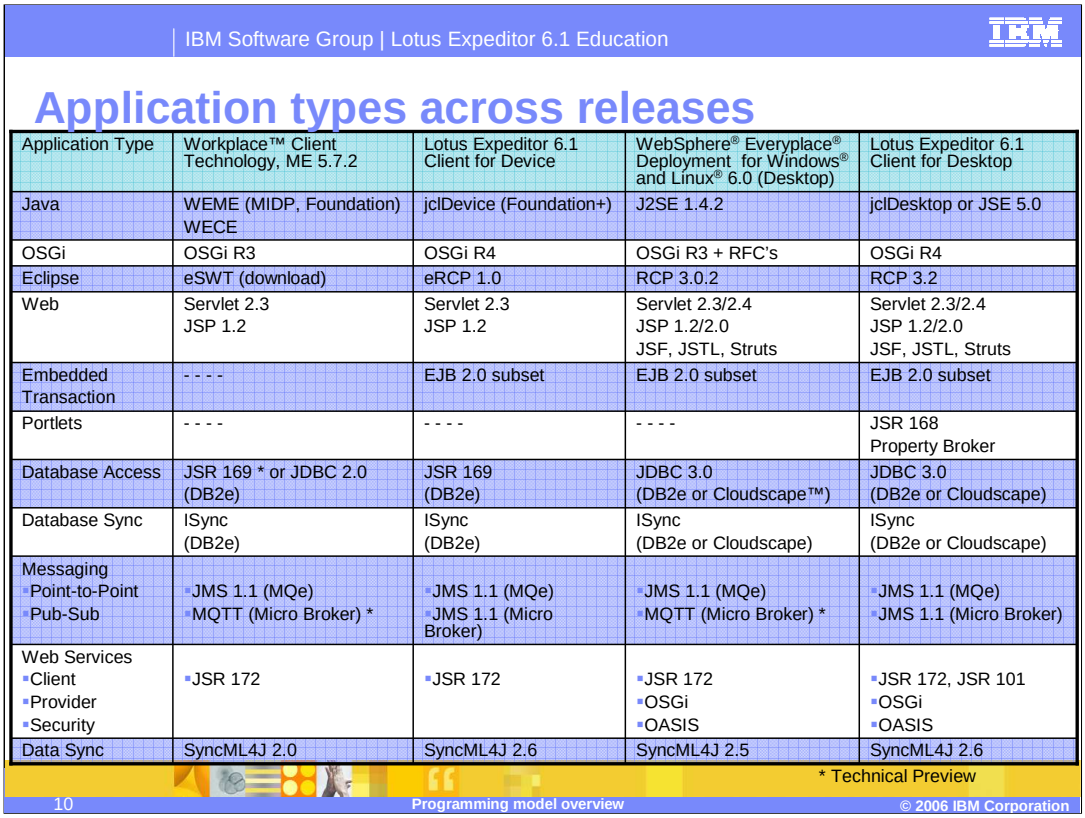


Finally, the client platform enables you to deliver client applications that provide a rich graphical user interface through support of the Eclipse Rich Client Platform (RCP) or the embedded RCP (eRCP). You can continue to use business logic components, such as Embedded Transaction Applications, with your rich client applications.

Client application types

- Java
- OSGi
- Eclipse (RCP and eRCP)
- Web
- Embedded Transaction Applications
- Portlets
- Database
- Messaging
- Web Services

As a result of the Client Services and the programming model supported by IBM Lotus Expeditor 6.1, you can develop a variety of standards-based client applications, including Java applications, OSGi applications, Web applications that provide a traditional Web browser user experience, Embedded Transaction applications for business logic, portlets for a portal user experience, Eclipse Rich Client Platform (RCP) desktop applications and embedded RCP (eRCP) device applications with rich graphical user interfaces, database applications to access and synchronize relational data, messaging applications to perform secure transactions, and Web Services applications to access consumer and business data.



This slide shows the varied types of applications and how they are supported across the managed client releases. As you can see, IBM's managed client releases support industry standard API's that enable programmer and application portability across a variety of clients.

For Windows Mobile handheld devices, Lotus Expeditor 6.1 client for device continues to support almost all of the API's implemented by the previous managed client device release, Workplace Client Technology, Micro Edition (WCTME) 5.7.2. However, Lotus Expeditor 6.1 client for device does not support the Java Micro Edition (JME) Mobile Information Device Profile (MIDP), which was implemented by IBM in WebSphere Everyplace Micro Edition (WEME). MIDP is for small consumer handheld devices. Lotus Expeditor focuses on more powerful handheld devices that are better suited for business applications.

For desktops, laptops, and tablets, Lotus Expeditor 6.1 client for desktops supports all of the API's that were implemented by the previous managed client desktop release, WebSphere Everyplace Deployment (WED) for Windows and Linux 6.0.

Lotus Expeditor 6.1 client for device and desktop releases now fully support many important standard API's. JSR 169 enables application access to local relational databases DB2 Everyplace or Cloudscape. JMS enables applications to publish and subscribe messages through the Micro Broker and the underlying publish and subscribe messaging protocol, called MQ Telemetry Transport (MQTT).

Lotus Expeditor 6.1 client for device adds support for the Eclipse embedded Rich Client Platform (eRCP), which enables applications to implement rich graphical user interfaces (GUI) by using the native GUI subsystem of the handheld device. This release also adds support for Embedded Transaction Applications, which run in an Embedded Transaction Container that implements a subset of the Enterprise Java Bean (EJB) 2.0 specification so that these applications can now run on handheld devices in addition to desktops.

Lotus Expeditor 6.1 client for desktop now supports JSR 101 for Web Services client applications, JSR 168 so portlets can run on the client, and Property Broker so portlets and Eclipse applications can communicate with other portlets and Eclipse applications.

And finally, for both the device and desktop releases, specific API's have been updated to support the latest standards including OSGi release 4 (R4) and SyncML for Java (SyncML4J) 2.6.

Programming model summary

- Developers move key application components to the client through use of standard APIs and services
  - Client applications can run on multiple client platforms
- Moving application components to run on the client can have dramatic results for business
  - Support mobile users (online/offline)
  - Improve application response time
  - Reduce network traffic
  - Distribute application workload
  - Provide the appropriate user experience
  - Reuse business logic
- Client applications access Enterprise applications, services and data

Moving application components to run on a client can have dramatic results for business. End-users benefit from improved application response time because applications perform business operations locally on the client. As a result, there is a reduction in network traffic between clients and servers, and in server workload. Furthermore, mobile end-users can continue to productively use their applications from their clients even when they are at a location that does not have network connectivity, such as a customer site. You can also utilize the local graphical user interface (GUI) capabilities of the client devices to deliver a richer user experience than can be supported by a Web browser. You can reuse your client business logic across multiple applications, thus saving on development and maintenance costs. Through support of Access Services, client applications can securely access mission-critical applications, services and data in the Enterprise.

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:

Cloudscape Everyplace IBM Lotus WebSphere Workplace

Windows, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

EJB, J2SE, Java, JDBC, JSP, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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.

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 this presentation.