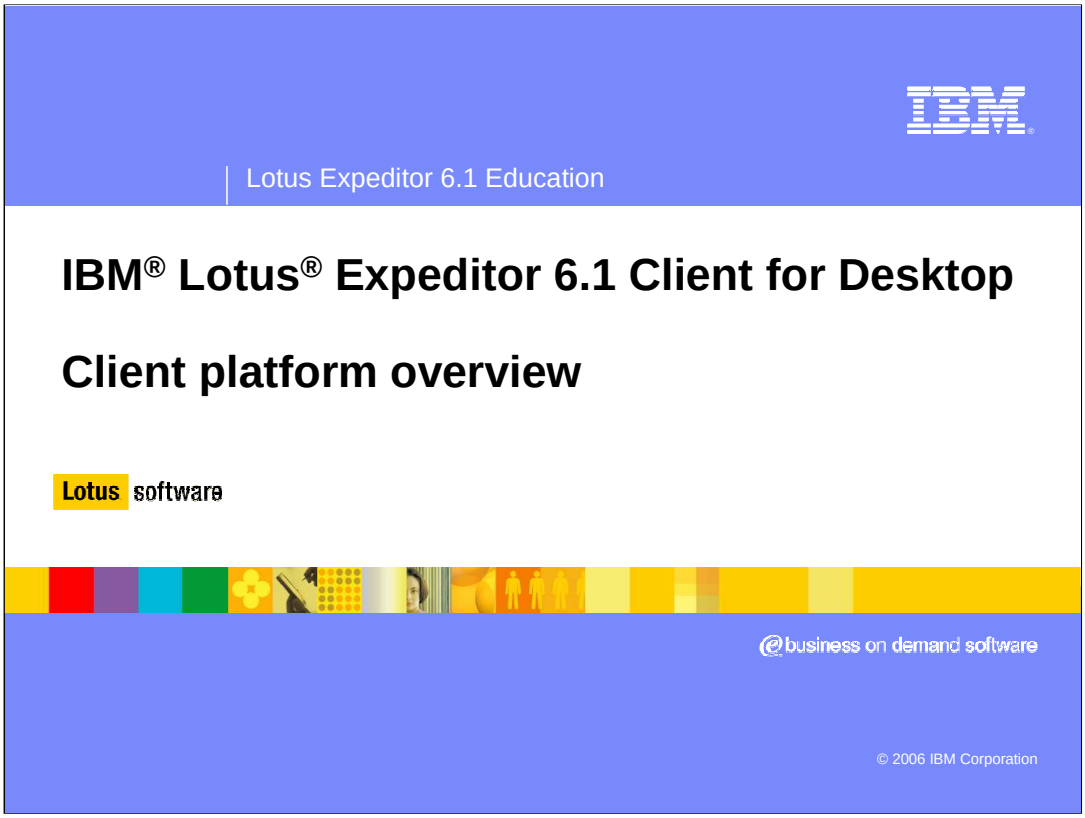


Hello, and welcome to this presentation from IBM, which is provided to give you an overview of the client platform provided by IBM Lotus Expeditor 6.1 Client for Desktop.

Goals

- Understand the client platform provided by IBM Lotus Expeditor 6.1 Client for Desktop

The goal of this presentation is to understand the client platform provided by IBM Lotus Expeditor 6.1 Client for Desktop.

Agenda

- Managed Client
- Client Services
  - Interaction Services
  - Managed Client Services
  - Platform Management
  - Access Services

The agenda of this presentation is to provide an overview of the managed client and client services provided by IBM Lotus Expeditor 6.1 Client for Desktop.

Section

Managed client

Let’s start with an overview of the Managed Client.

The managed client platform provides the following set of standards-based Client Services for the development of your managed client applications:

Managed Client Services, which include a selection of runtime environments, a robust component framework, and additional component services, all of which enable Java™ applications to run on multiple operating systems and clients.

Platform Management, which includes the Eclipse Update Manager and an Enterprise management agent to install and update applications and services on the client platform.

Access Services, which includes data and synchronization services, transactional messaging, Web Services, a Web container to run local Web applications, an embedded transaction container to run local embedded Enterprise Java Beans (EJB's), a portlet container to run local portlets, and more.

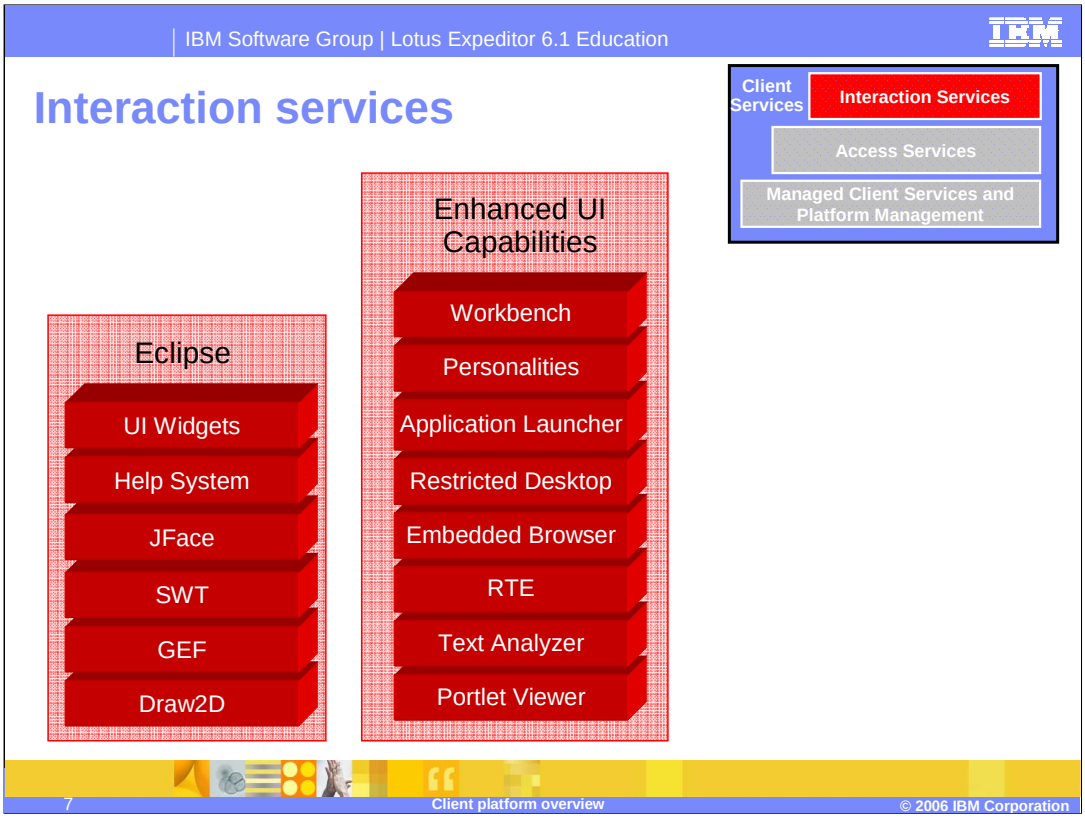
Interaction Services, which includes integrated browser controls to launch Web applications, Eclipse technology to support GUI applications, a portlet viewer to launch portlets, and a Workbench that enables end-users to install and launch one or more applications.

And **Standard-based APIs** to invoke these services.

Section

Client services

Next, let’s highlight the value, standards, and major components for each of these Client Services.

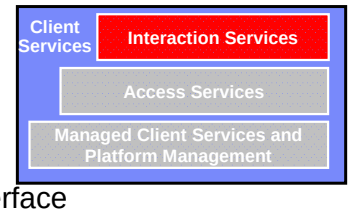


The Expeditor Desktop client platform includes key interaction services provided by Eclipse. It also adds enhanced user interface capabilities that result in an integrated Workbench window from which your end-users can install, launch and manage one or more applications.

You can deliver rich client applications that use Eclipse to present a graphical user interface and Web applications that enable a Web browser to render the user interface. You can also deliver Portlet applications and Composite applications. Composite applications are applications that enable independently-developed components which provide specific business functions to run together in a single application.

The Expeditor Desktop client platform also includes support for User Interface widgets, such as Rich Text Editor, Text Analyzer, Portlet Viewer and the Embedded browser.

Interaction services



- Value
  - Provides “fit for purpose” application user interface
  - Provides integrated workbench
- User Interface Models
  - Rich Client Application Model (“Reactive”)
  - Web Application Model (“Request and Response”)
  - Portlet Application Model s
  - Composite Application Model
- Standards
  - Eclipse
  - W3C

The client platform provides both a “fit for purpose” application user interface and an integrated workbench that supports multiple user interface models. The platform supports user interface models such as the Rich Client Application Model, Web Application Model, Portlet Application Model and Composite Application Model.

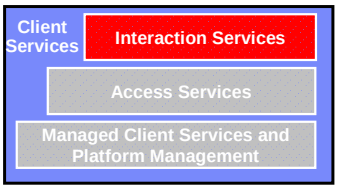
There are several application patterns that are recommended for use in the managed client platform. One pattern is the browser user interface pattern, which is supported by the Web Application Model. Web applications present their user interface through the use of generated scripting languages like HTML, which is rendered for display by a browser.

Another pattern is to build a graphical user interface using Eclipse, to aggregate display components into views, and views into perspectives. Applications are defined using extension points to define the actions, views, and perspectives that provide the user interface. This pattern is the rich client user interface pattern, which is supported by the Rich Client Application Model.

For Expeditor Client for Desktop, there are additional application patterns.

The portlet pattern is supported by the Portlet Application Model. This pattern is based on JSR 168. Another pattern is the Composite Application pattern, which is supported by the Composite Application Model. In this model, multiple applications cooperate by using inter-component communications. For example, you can use the Property Broker to communicate among portlets and Eclipse applications running in the same client process. Or, you can use the Lotus Expeditor micro broker to communicate among applications running in the same or different processes.

Interaction services:
Major components



- Workbench/Personalities
  - Integrated application desktop window
  - Browser perspective
  - Application perspective
  - Branding
  - Restricted Workbench
- Eclipse
  - Standard Widget Toolkit (SWT), JFace
  - GEF, Draw2D
  - Help System
- Preferences
  - Contribute preference dialogs for your application

The **Workbench** provides an integrated application desktop window so end-users can install, manage, and launch one or more applications within a single window. The Workbench presents each application individually in its own perspective, only one of which is visible at any given time. When an end-user selects an application from the Workbench, the Workbench launches the perspective for that application. You specify an extension point for each of your applications so the Workbench can correctly launch the perspective for your application. You can also modify the user interface of the client workbench to include your own **branding**. You can modify such elements as the title bar, splash screen, icons and images, and the About dialog.

A personality extends the concept of a WorkbenchAdvisor to allow for different WorkbenchWindow configurations to run within one running VM. The desktop client provides a personality. However, you can also provide your own personalities.

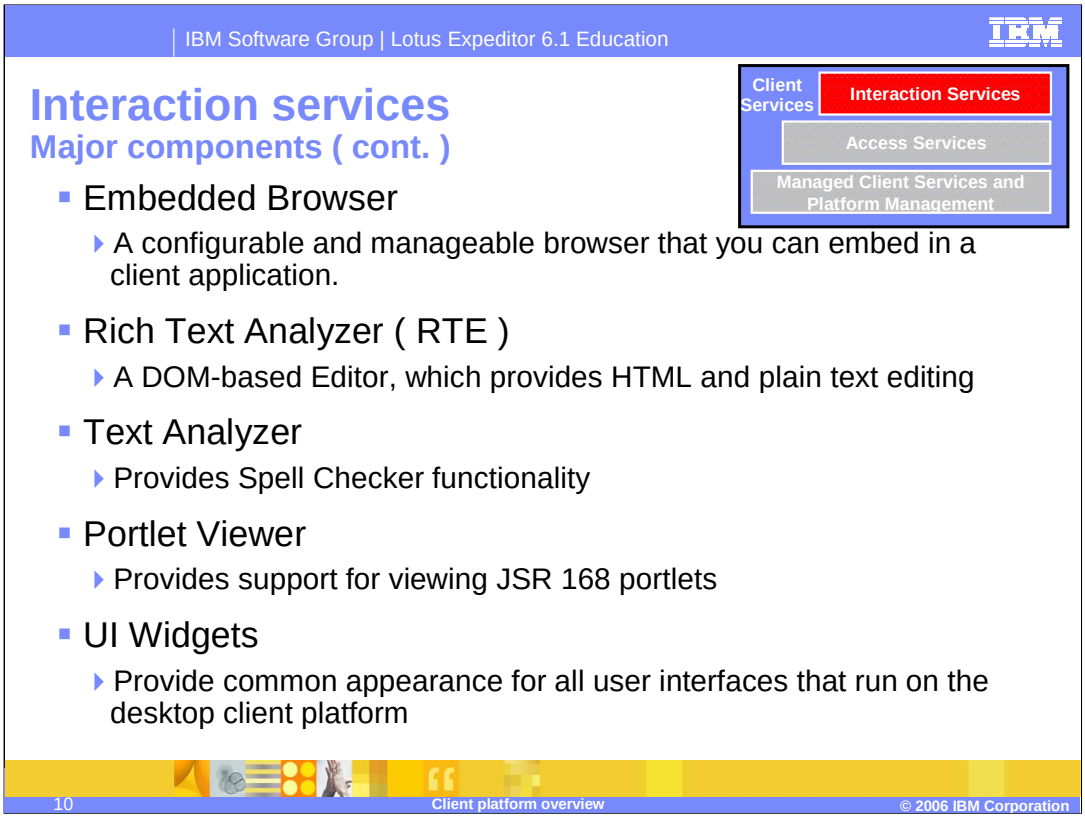
You can also enable the **Restricted Workbench** Service to provide a restricted environment in which all Expeditor Client users are limited to the applications and operating system services that the administrator has configured.

You use the Eclipse **Standard Widget Toolkit (SWT)** and the **JFace toolkit** to develop the GUI for rich client applications. SWT provides a cross-platform API that tightly integrates with the native widgets of the operating system and, therefore, gives your applications an appearance that makes them virtually indistinguishable from native applications. The JFace toolkit provides a set of components and helper utilities that simplify many of the common tasks in developing SWT user interfaces.

You use the **Help System** so end-users can select Help in the workbench to view helpful information about your applications.

You can use Eclipse **preferences** with OSGi configuration administration to persistently store your applications' preferences so that end-users can view and set these preferences from the Workbench. Your applications can also be notified about configuration changes and System Administrators can remotely query and update this configuration.

The desktop client also provides services that enable you to contribute **Helps** and **Preferences** for your applications so end-users can understand and configure your applications respectively within the Workbench.



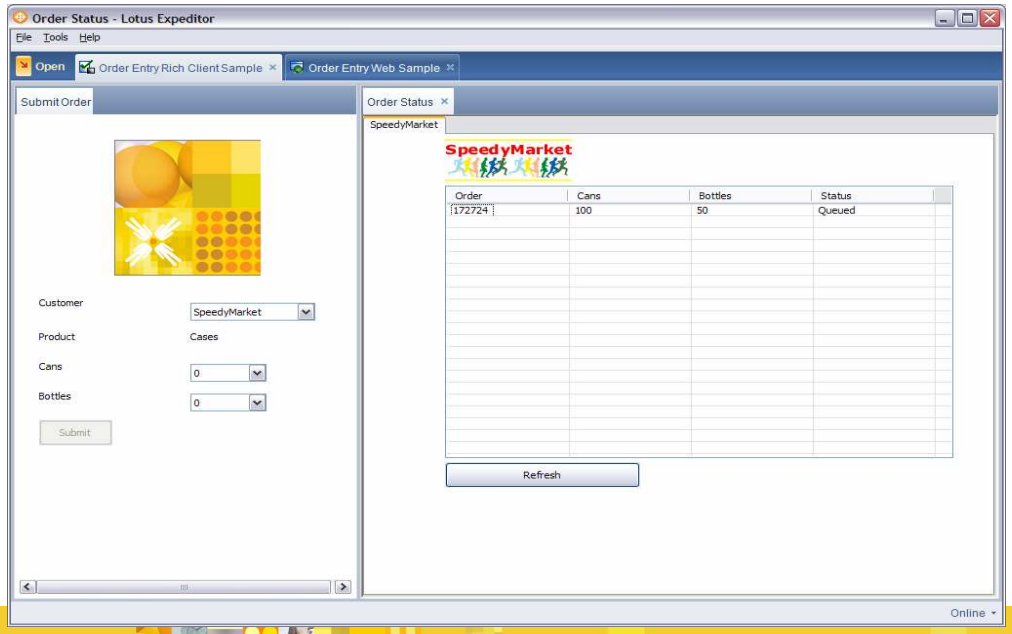
With the Lotus Expeditor Client, you can run Web browsers embedded in the client window. The Embedded Browser is based on the SWT browser widget in Eclipse, which can support Internet Explorer on Windows and Mozilla on Linux.

The provided Rich Text Editor, along with the a DOM-based Editor, allows HTML and plain text editing. The Text Analyzer framework can be extended to provide Spell Checker functionality. Spell Checker functionality is used to check misspelled words in a document and supports 26 languages. The Spell Checker functionality can be used by many editors, by implementing the document interfaces.

The Portlet Viewer provides support for viewing JSR 168 portlets on the client.

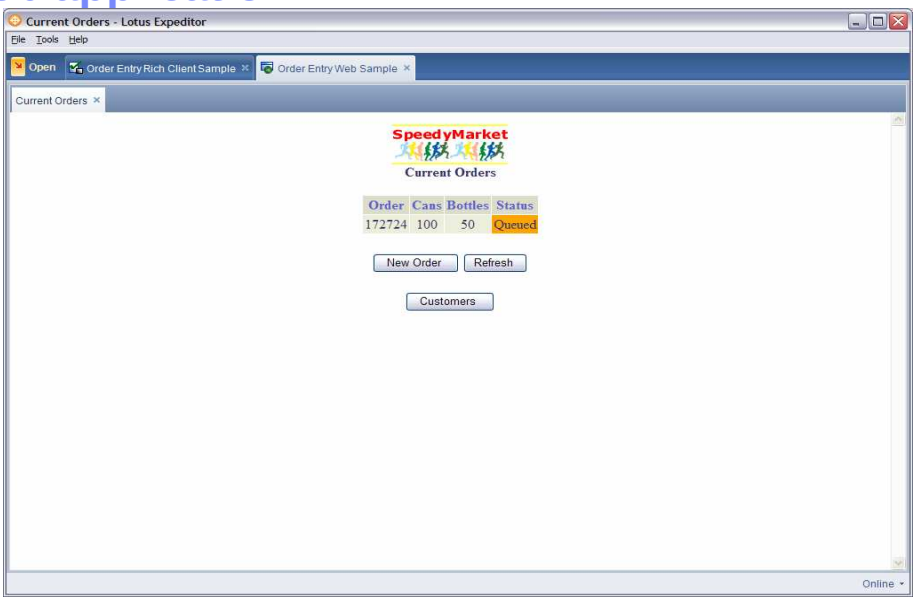
The user interface widgets provide a common appearance for all user interfaces that run on the desktop client platform. Using These widgets, you can render the toolbar skin, which means it formats the toolbar to make its style consistent with the style defined for the application in which it displays in. You can also, manage the dragging and dropping of items within the toolbar. In addition, you can save and load the last state of toolbar items using the Eclipse memento pattern, which uses the saveState and restoreState methods to remember the position of each toolbar item when the user shuts down the client. This allows the same toolbar items to display when the user restarts the client.

Rich client application



This screen shot shows a rich client application running in the workbench. The application has two views so you can submit orders while simultaneously viewing the status of orders that have already been submitted.

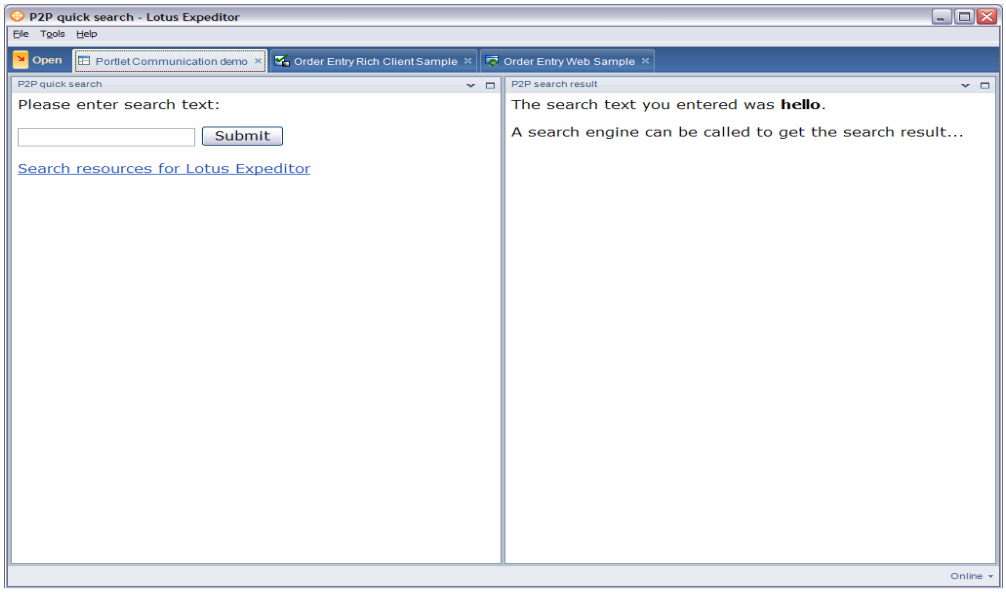
Web application



This screen shot shows a Web application running in the workbench. Notice that the user experience consists of a single view. In this example, users must go to another view to submit new orders for the current customer or to select another customer.

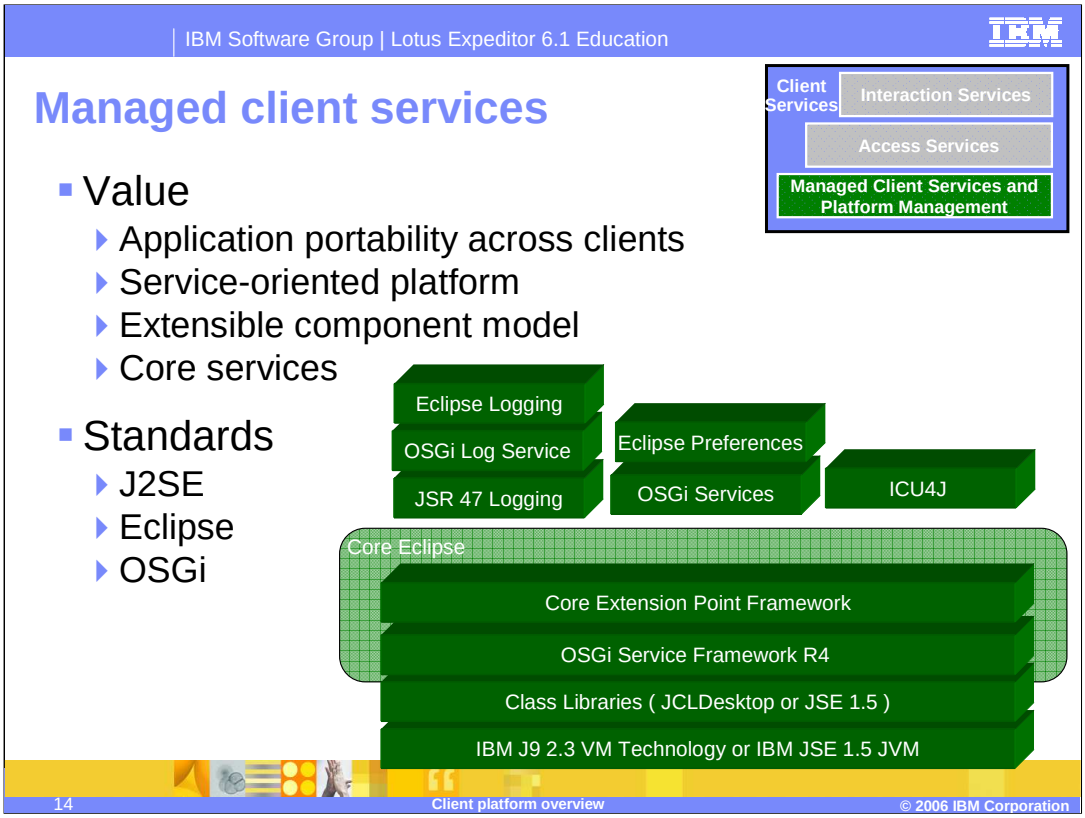
Also notice that both the Order Entry Web Application and the Order Entry Rich Client Application are running in the same Workbench as shown by the application tabs, allowing the user to switch between applications.

Portlet application

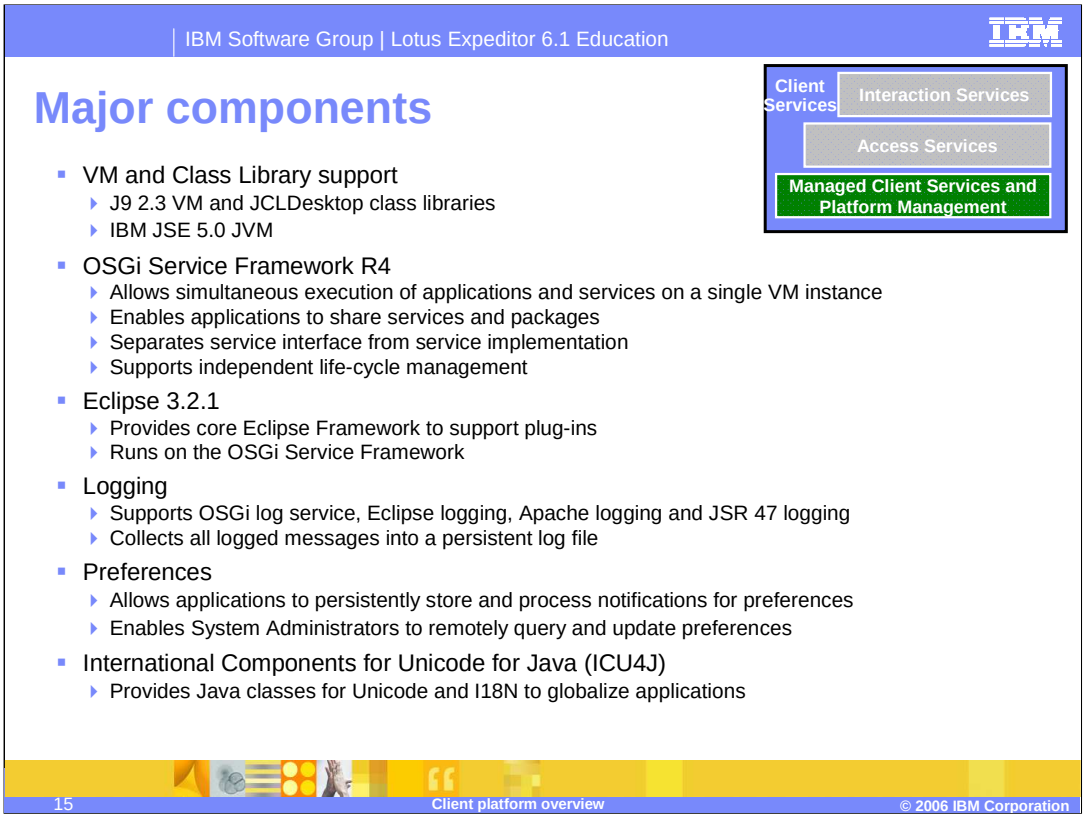


This screen shot shows a local Portlet application running on the workbench. This application demonstrates communication between two portlets. When data is entered in the search text field and Submit is clicked, the search text is updated in the second portlet window.

Also, notice that the Order Entry Web Application, the Order Entry Rich Client Application and the Portlet application are all running in the same Workbench as shown by the application tabs, allowing the user to switch between applications.



Managed Client Services enable multiple applications and services to run on the same Virtual Machine, support life cycle management of these applications and services, and provide application portability across Windows and Linux operating systems. The client provides an extensible component model through a service-oriented platform that enables you to add your own applications and services. The client platform provides additional services to assist you in the development of your applications including logging, preferences, globalization, and more.



The client platform ships IBM's **J9 2.3 VM with JCLDesktop** class library, as the default Java runtime environment. Also provided is the **Java Standard Edition (JSE) 5.0** as an optional Java runtime environment.

The client platform provides an **OSGi Service Framework** that implements the OSGi R4 framework specification and provides a service-oriented architecture on top of the VM. The OSGI Service Framework enables multiple applications and services to run on the same VM and supports life cycle management of these applications and services. OSGi Services provide the interface definitions for standard services defined by the OSGi R4 specification and OSGi Utilities provide an implementation of the OSGi utilities interface.

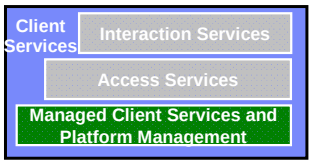
The **Eclipse** Core Extension Point Framework supports the Eclipse plug-in component model and, by running with the OSGi Service Framework, enables plug-ins to receive the corresponding benefits of OSGi. The client platform is built on Eclipse 3.2.1

The client platform enhances **Logging** by collecting all messages logged in the client platform into one persistent log file regardless of whether you use the OSGi log service, Eclipse logging, Apache Commons Logging or JSR 47 logging.

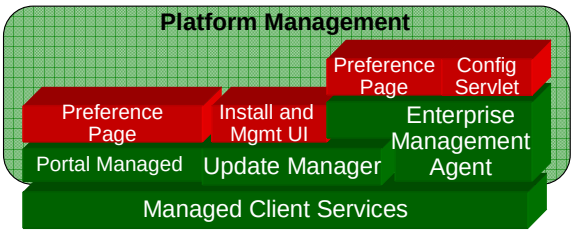
You can use **Preferences** to persistently store your applications' preferences, allow your applications to process notifications of preferences changes, and to enable System Administrators to remotely query and update these configuration values.

International Components for Unicode for Java (or ICU4J) is a set of Java classes that extend the capabilities provided by the J2SE class libraries in the areas of Unicode and internationalization support so you can globalize your applications.

Platform management

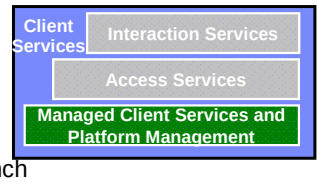


- Value
  - Enable “On Demand” management of applications
- Deployment options
  - Stand-alone - Eclipse Update Manager
  - Enterprise - Enterprise Management Agent and DMS
  - Portal Managed Client
- Standards
  - Eclipse
  - OSGi
  - Open Mobile Alliance



Platform Management installs, maintains, and configures applications and services on the client. End-users can directly deploy applications onto their Workbench, known as “stand-alone deployment” or a System Administrator may remotely deploy and configure applications on one or more clients from a central administrative console in the Enterprise which is known as “Enterprise deployment”.

Platform management:
Major components



- Update Manager (“Stand-alone Deployment”)
  - End-users can directly install applications onto their Workbench
  - Supports the Eclipse standard
  - Install User Interface - Allows end-users to install applications onto their client, locally from their Workbench
  - Application Management User Interface - Enables end-users to manage the current configuration of their client platform from their Workbench
- Enterprise Management Agent (“Enterprise Deployment”)
  - Enables a System Administrator to remotely manage the client platform
  - Supports the Open Mobile Alliance (OMA) protocol with DMS
  - Supports the OSGi standard to locally manage client components
  - Enterprise Management Agent Servlet - Allows a System Administrator to view and control the Agent with a browser
  - Preferences Page allows an end-user to locally configure Agent from their Workbench
- Portal Managed Client
  - Manage your client from a Portal server
  - Preference Page allows an end-user to locally configure the Home Portal Account from their Workbench

The **Update Manager** provides the capability to install applications and components from standard Eclipse update sites onto the Workbench.

The **Install user interface** enables end-users to use the Update Manager to install applications and components.

The **Application Management user interface** also runs with the Update Manager, and enables end-users to manage the current configuration of the client platform, including applications that are installed on the client, from the Workbench.

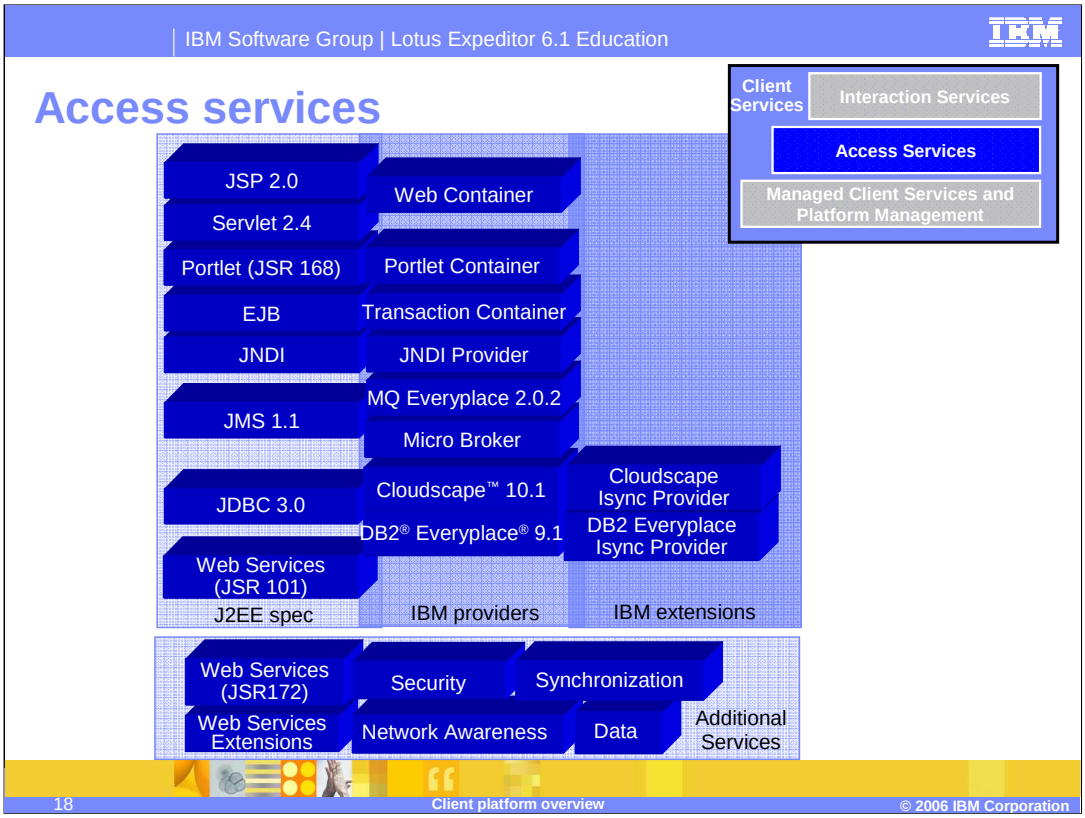
The **Enterprise Management Agent** enables a System Administrator to manage the client platform remotely by using a Device Management Server, which is provided by the server platform.

The **Enterprise Management Agent Servlet** allows you to view and modify the configuration and control the features of the Enterprise Management Agent. You can launch it from within the Expeditor client workbench or from within a web browser.

You may also configure the Enterprise Management Agent after you install the client platform by using the **Preferences Page** from the Workbench.

The Portal Managed client enables the management of applications on your client from the Portal server.

You may configure the Home Portal Account from the workbench Preference Page after the install of the client platform.



Access Services provide a familiar programming model for J2EE developers so they can reuse their skills and software components to develop applications that run on managed clients. Additionally, Access Services enable managed client applications to support offline operations. Access Services also enable you to move key components of your application to the client platform through the use of standard APIs.

The client platform provides an embedded **Web container** to run JEE Web applications that support the Servlet 2.3/2.4, JSP 1.2/2.0, JSF 1.1, JSTL, and Struts specifications. The Web container enables you to move your Web applications from the server to clients to preserve the existing browser user interface, leverage your existing Web components, and provide a richer user experience through support of local and offline operations.

The client platform also provides an **Embedded Transaction Container** to run JEE Enterprise Java™ Beans (or EJB's) that conform to any of the following specifications: 1.1 and 2.0 Stateless Session Beans, Container Managed Persistence (or CMP) Entity Beans, and Bean Managed Persistence (or BMP) Entity Beans. This container enables you to move your business logic from the server to clients, allowing you to leverage your existing beans to make business logic available to client applications, including Web applications, and support local and offline operations. These business logic components are referred to as *Embedded Transaction applications.*

In addition, the desktop client provides a portlet container to run **portlets** that support the JSR 168 specification.

There are four key services that support local and offline operations.

First, you can use the **JDBC** API with DB2 Everyplace or IBM Cloudscape as a local SQL database when more advanced data manipulations are required than can be supported by placing data in a local file store. These databases can periodically synchronize with Enterprise databases to capture data on the client for use by the client application when the user is offline. These databases can also protect local data through data encryption.

Second, you can also use the Java Message Service (or **JMS**) API with WebSphere® MQ Everyplace (or MQe) to send and receive messages. MQe provides once-only, assured messaging and supports offline operations with local message queues that hold messages when the device is offline and then sends these queued messages to Enterprise applications when the device is back online. Similarly, messages destined for client applications are held in server-side message queues and then sent to the client applications when the device is back online. MQe encrypts messages to protect content over the network. As a result, the client platform enables your users to conduct secure e-business on demand transactions.

Third, you can use the JMS API with the **micro broker**, which is suitable for applications that require messaging, notification and event services. The micro broker supports publish and subscribe messaging in which publishers generate messages containing information about a particular subject, subscribers express interest in messages containing information on a particular subject, and a broker receives messages from publishers and delivers messages on a particular subject to the subscribers registered for that subject. You can support offline operations through defined quality-of-service levels and durable subscriptions.

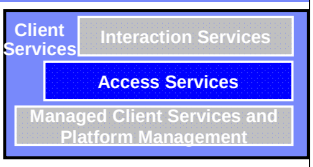
Fourth, for the desktop client, you can use the **Network Layer** API to determine the status of the network and remote resources when running your applications. You can then run your application logic accordingly.

For online operations, the client platform supports **Web Services** so client applications can consume and provide Web Services in a secure manner. As a result, your users have access to a broad range of business data and consumer information. The client platform implements Web Services similar to those defined in JSR 172 and provides support for document literal encoded streams that exchange well-typed data objects so client applications can consume Web Services. You can also develop an OSGi service and, during registration of the service, indicate that it is also available as a Web Service. For the desktop client, you can also use Axis Web Services so client applications can consume Web Services, with full support for JAX-RPC (JSR 101).

The **SyncML4J** (SyncML for Java) toolkit enables you to develop data synchronization and device management client applications based on the Open Mobile Alliance (or OMA) Data Synchronization (or DS) and Device Management (or DM) standard protocols. As a framework, SyncML4J supports user-defined data sources. Data sources can range from simple resources, such as memos and images, to complex schema-aware data types, such as relational databases or PIM databases.

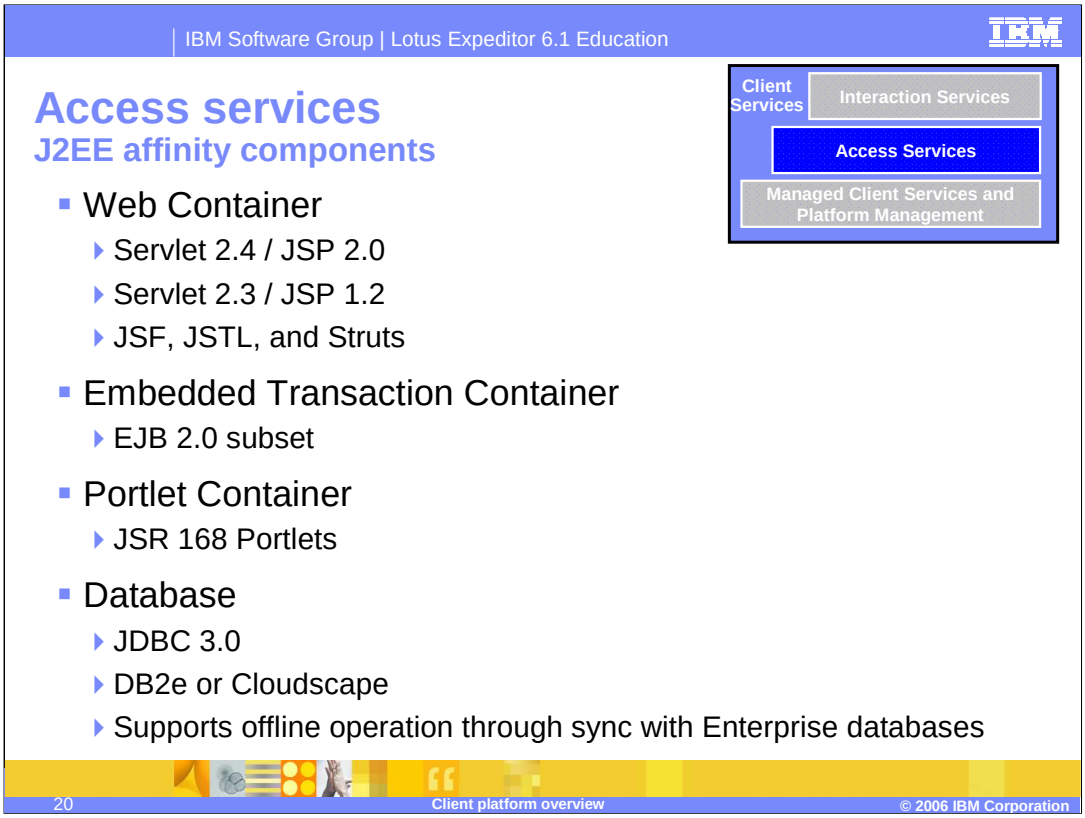
For the desktop client, **security** services support: a key store, which provides an encrypted local repository for user security information; accounts, which allow access to user account information (for example: user id and password); and single sign-on, which minimizes logon prompts. Additional services include: **database lifecycle management**, which provides uniform interoperability with different relational databases; and **synchronization manager**, which allows users and applications to initiate, control and monitor synchronization of local data stores using one or more synchronization services.

Access services



- Value
  - Extend backend programming model “out” to desktops and devices
- Services
  - Web Container
  - Transaction Container
  - JNDI
  - Messaging
  - Database
  - Web Services
  - Data Sync Framework
  - SyncManager
  - Portlet Container
  - Network Layer
  - Security
  - DB Lifecycle Management
- Standards
  - J2EE subset, J2SE, J2ME
  - W3C
  - Web Services
  - OMA

Access Services provides value by extending the backend programming model “out” to desktops and devices. Access services provides services such as web container, portlet container, security, sync manager, web services including JSR 172 support and Axis 1.3 support. The Access services also include database and messaging support and database lifecycle management.



Here are the specific Access Services that support affinity with the J2EE / WebSphere programming model.

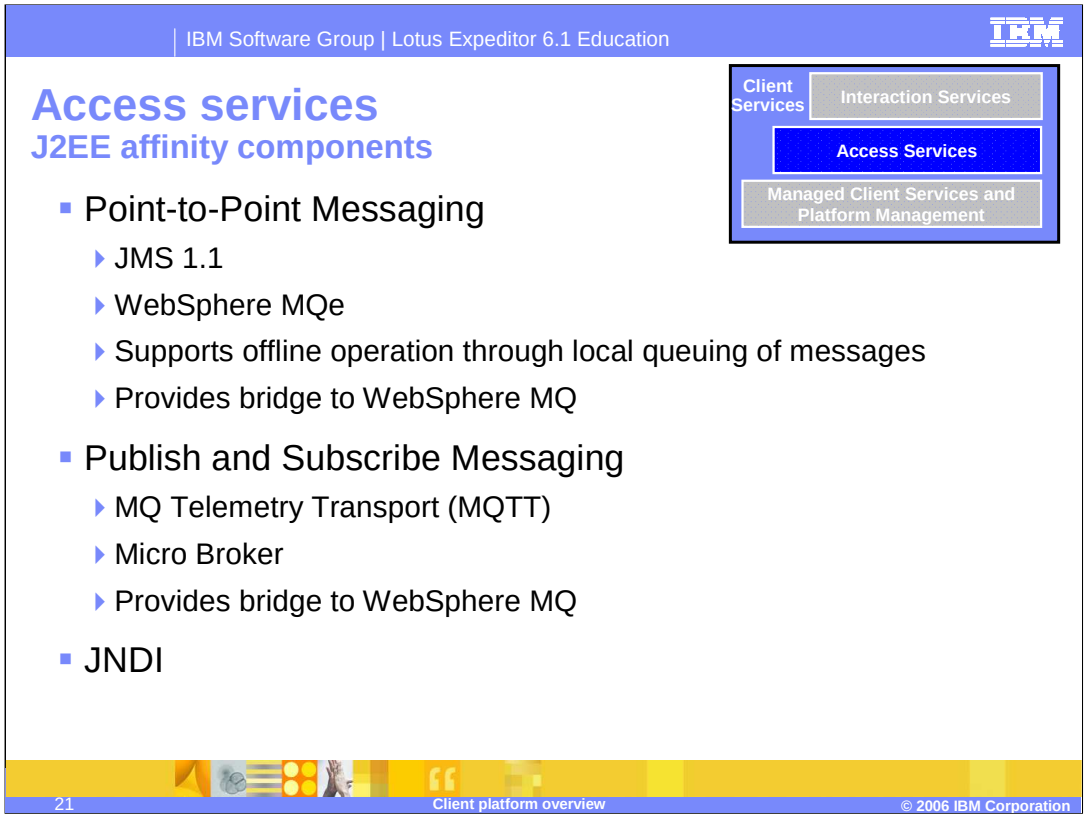
The client platform provides an embedded **Web container** from WebSphere to run Web applications that consist of JSP's and servlets. The Web container also supports applications that use Java Server Faces (or JSF), JSP Standard Tag Library (or JSTL), and Struts. As a result, the Web container enables you to move your Web applications from the server to clients to preserve the existing browser user interface, leverage your existing Web components, and provide a richer user experience through support of local and offline operations.

The client platform also provides an **Embedded Transaction Container** to run Embedded Transaction Applications that conform to a subset of the EJB 2.0 specification. This includes support for stateless session beans and entity beans. This container enables you to move your business logic from the server to clients so you can leverage your existing beans to make business logic available to client applications, and support local and offline operations.

The client platform provides a **portlet container** to run JSR 168 portlets.

There are key services that support local and offline operations.

You can use the **JDBC 3.0** API with DB2 Everyplace or IBM Cloudscape as a local SQL database when more advanced data manipulations are required than can be supported by placing data in a local file store. These databases can periodically synchronize with Enterprise databases to capture data on the client for use by the client application when the user is offline. These databases can also protect local data through data encryption.



Here are more of the specific Access Services that support affinity with the J2EE / WebSphere programming model.

You can use the Java Message Service (or **JMS**) 1.1 API with WebSphere MQ Everyplace (or MQe) to send and receive messages. MQe provides once-only, assured messaging and supports offline operations with local message queues that hold messages when the device is offline. It then sends these queued messages to Enterprise applications when the device is back online. Similarly, messages destined for client applications are held in server-side message queues and then sent to the client applications when the device is back online. MQe encrypts messages to protect content over the network. As a result, the client platform enables your users to conduct secure e-business transactions.

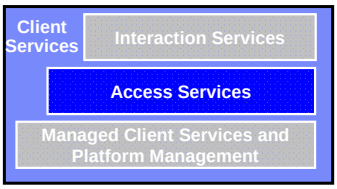
WebSphere MQe provides a bridge to WebSphere MQ to connect your client applications to the Enterprise Service Bus.

The client platform provides both enterprise class messaging through the Java Message Service (JMS) with WebSphere MQ Everyplace (MQe) and the Micro Broker with the MQ Telemetry Transport (MQTT) Java client APIs.

Micro Broker supports publish and subscribe messaging.

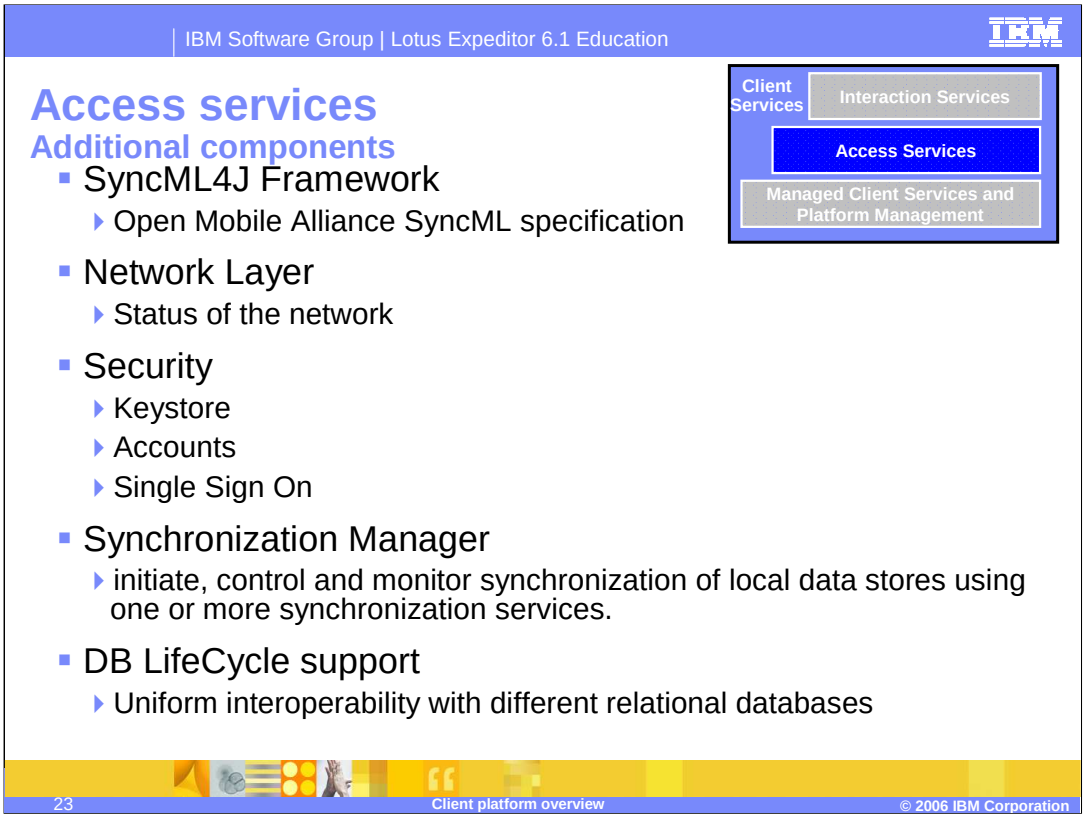
JNDI provides a name space so applications can utilize named objects.

Access services
Additional components



- Axis Web Services
  - Apache Axis 103
  - JAX-RPC compliant
- Mobile Web Services
  - Web Services client (similar to JSR 172)
  - Web Services provider (host Web Services)
  - WS-Security
  - Enables support of additional data types

For online operations, the client platform supports **Web Services** so client applications can consume and provide Web Services in a secure manner. As a result, your users have access to a broad range of business data and consumer information. The client platform implements Web Services similar to those defined in JSR 172 and provides support for document literal encoded streams that exchange well-typed data objects so client applications can consume Web Services. You can also develop an OSGi service and, during registration of the service, indicate that it is also available as a Web Service. For the desktop client, you can also use Axis Web Services so client applications can consume Web Services, with full support for JAX-RPC (JSR 101).



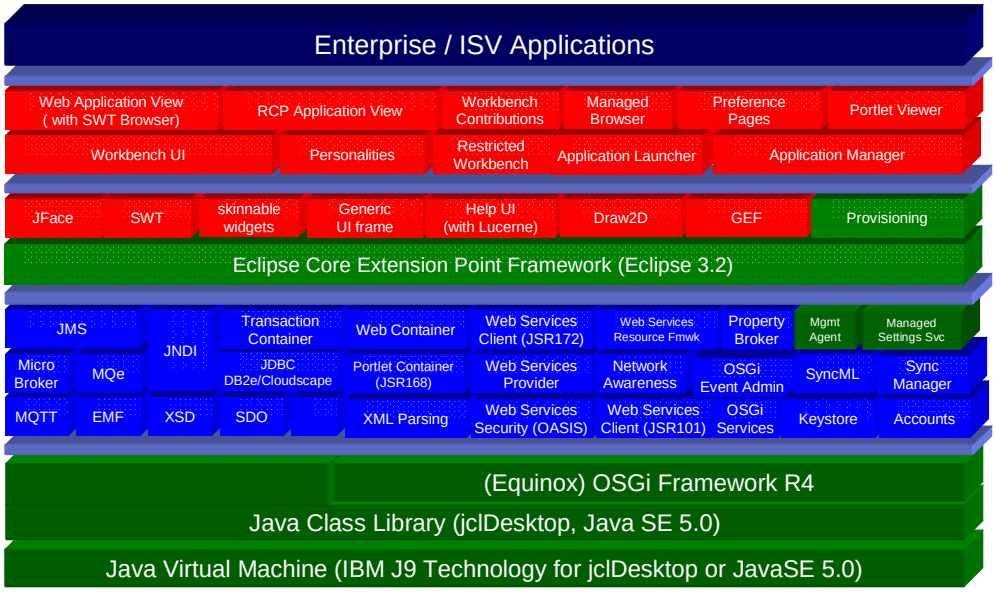
The **SyncML4J** (or SyncML for Java) toolkit enables you to develop data synchronization client applications.

The **Network Layer** API enables you to determine the status of the network and remote resources when running your applications.

Security services support includes: a key store, which provides an encrypted local repository for user security information; accounts, which allows access to user account information (for example, your user id and password); and single sign-on, which minimizes logon prompts.

Additional services include: **database lifecycle management** and **synchronization manager**.

Complete client stack



This is a view of the complete client stack including managed client services, access services and interaction services.

For more information…

- For more information on any of the Client Services, select from the following topics under the Client platform task:
  - Managed Client Services
  - Platform Management
  - Access Services
  - Interaction Services

For more information on the Client Services, select from the following topics under the Client platform task:

Managed Client Services

Platform Management

Access Services

Interaction Services

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

EJB, J2EE, J2ME, J2SE, Java, JDBC, JSP, JVM, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. 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 the presentation