Sun Java System Mobile Enterprise Platform 1.0 Architectural Overview

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–3751–10
July 2008

080711@20490

Contents

Preface

This guide provides an introduction to the architecture of Sun Java System Mobile Enterprise Platform 1.0 (MEP).

MEP is a comprehensive mobility solution that enables offline data access, data synchronization, and secure access to EIS/EAI applications, such as Siebel and SAP.

MEP is based entirely upon open standards, including the following:

- Java Platform, Mobile Edition (Java ME)
- Java Platform, Enterprise Edition (Java EE)
- The dominant industry standard OMA DS, formerly known as SyncML. The specifications for Open Mobile Alliance Data Synchronization V1.1.2 and V1.2.1 are available at `http://www.openmobilealliance.org/Technical/release_program/ds_v112.aspx` and `http://www.openmobilealliance.org/Technical/release_program/ds_v12.aspx`.

Who Should Use This Book

This guide is intended for customers, developers, and administrators who will use MEP.

Before You Read This Book

This book has no prerequisites.

How This Book Is Organized

This book contains the following chapter:

- Chapter 1, "Overview of Sun Java System Mobile Enterprise Platform," describes the components of MEP and how these components communicate with each other to provide a seamless data synchronization experience.

Mobile Enterprise Platform Documentation Set

The Mobile Enterprise Platform documentation set is available at `http://docs.sun.com/coll/1780.1`. To learn about Mobile Enterprise Platform, refer to the books listed in the following table.

TABLE P–1 Books in the Mobile Enterprise Platform Documentation Set

Book Title	Description
Sun Java System Mobile Enterprise Platform 1.0 Release Notes	Late-breaking information about the software and the documentation. Includes a comprehensive summary of the supported hardware, operating systems, application server, Java™ Development Kit (JDK™), databases, and EIS/EAI systems.
Sun Java System Mobile Enterprise Platform 1.0 Architectural Overview	Introduction to the architecture of Mobile Enterprise Platform.
Sun Java System Mobile Enterprise Platform 1.0 Installation Guide	Installing the software and its components, and running a simple application to verify that installation succeeded.
Sun Java System Mobile Enterprise Platform 1.0 Deployment Guide	Deployment of applications and application components to Mobile Enterprise Platform.
Sun Java System Mobile Enterprise Platform 1.0 Developer's Guide for Client Applications	Creating and implementing Java Platform, Mobile Edition (Java ME platform) applications for Mobile Enterprise Platform that run on mobile devices.
Sun Java System Mobile Enterprise Platform 1.0 Developer's Guide for Enterprise Connectors	Creating and implementing Enterprise Connectors for Mobile Enterprise Platform intended to run on Sun Java System Application Server.
Sun Java System Mobile Enterprise Platform 1.0 Administration Guide	System administration for Mobile Enterprise Platform, focusing on the use of the MEP Administration Console.

Application Server Documentation Set

When you install MEP, it is deployed to Sun Java System Application Server 9.1 Update 1.

The Application Server documentation set describes deployment planning and system installation. The Uniform Resource Locator (URL) for Application Server documentation is `http://docs.sun.com/coll/1343.5`. For an introduction to Application Server, refer to the books in the order in which they are listed in the following table.

TABLE P–2 Books in the Application Server Documentation Set

Book Title	Description
Documentation Center	Application Server documentation topics organized by task and subject.
Release Notes	Late-breaking information about the software and the documentation. Includes a comprehensive, table-based summary of the supported hardware, operating system, Java Development Kit (JDK), and database drivers.
Quick Start Guide	How to get started with the Application Server product.
Installation Guide	Installing the software and its components.
Deployment Planning Guide	Evaluating your system needs and enterprise to ensure that you deploy the Application Server in a manner that best suits your site. General issues and concerns that you must be aware of when deploying the server are also discussed.
Application Deployment Guide	Deployment of applications and application components to the Application Server. Includes information about deployment descriptors.
Developer's Guide	Creating and implementing Java Platform, Enterprise Edition (Java EE platform) applications intended to run on the Application Server that follow the open Java standards model for Java EE components and APIs. Includes information about developer tools, security, debugging, and creating lifecycle modules.
Java EE 5 Tutorial	Using Java EE 5 platform technologies and APIs to develop Java EE applications.
WSIT Tutorial	Developing web applications using the Web Service Interoperability Technologies (WSIT). Describes how, when, and why to use the WSIT technologies and the features and options that each technology supports.
Administration Guide	System administration for the Application Server, including configuration, monitoring, security, resource management, and web services management.
High Availability Administration Guide	Post-installation configuration and administration instructions for the high-availability database.
Administration Reference	Editing the Application Server configuration file, `domain.xml`.

TABLE P–2 Books in the Application Server Documentation Set *(Continued)*

Book Title	Description
Upgrade and Migration Guide	Upgrading from an older version of Application Server or migrating Java EE applications from competitive application servers. This guide also describes differences between adjacent product releases and configuration options that can result in incompatibility with the product specifications.
Performance Tuning Guide	Tuning the Application Server to improve performance.
Troubleshooting Guide	Solving Application Server problems.
Error Message Reference	Solving Application Server error messages.
Reference Manual	Utility commands available with the Application Server; written in man page style. Includes the `asadmin` command line interface.

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–3 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
AaBbCc123	A placeholder to be replaced with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized (note that some emphasized items appear bold online)	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file.

Symbol Conventions

The following table explains symbols that might be used in this book.

TABLE P–4 Symbol Conventions

Symbol	Description	Example	Meaning
`[ ]`	Contains optional arguments and command options.	`ls [-l]`	The `-l` option is not required.
`{ | }`	Contains a set of choices for a required command option.	`-d {y|n}`	The `-d` option requires that you use either the `y` argument or the `n` argument.
`${ }`	Indicates a variable reference.	`${com.sun.javaRoot}`	References the value of the `com.sun.javaRoot` variable.
-	Joins simultaneous multiple keystrokes.	Control-A	Press the Control key while you press the A key.
+	Joins consecutive multiple keystrokes.	Ctrl+A+N	Press the Control key, release it, and then press the subsequent keys.
→	Indicates menu item selection in a graphical user interface.	File → New → Templates	From the File menu, choose New. From the New submenu, choose Templates.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Searching Sun Product Documentation

Besides searching Sun product documentation from the docs.sun.com℠ web site, you can use a search engine by typing the following syntax in the search field:

search-term `site:docs.sun.com`

For example, to search for “broker,” type the following:

```
broker site:docs.sun.com
```

To include other Sun web sites in your search (for example, java.sun.com, www.sun.com, and developers.sun.com), use `sun.com` in place of `docs.sun.com` in the search field.

Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com` and click Send Comments. In the online form, provide the full document title and part number. The part number is a 7-digit or 9-digit number that can be found on the book's title page or in the document's URL. For example, the part number of this book is 820-3751.

◆ ◆ ◆ CHAPTER 1

Overview of Sun Java System Mobile Enterprise Platform

This document describes the architecture of Sun Java System Mobile Enterprise Platform (MEP). It also describes how these components communicate with each other to provide a seamless data synchronization experience.

Introduction

MEP is an open standards-based platform that enables access between enterprise applications and mobile devices. MEP provides reliable two-way data synchronization with security, device management, and offline access. MEP helps companies maximize return on investment on their existing IT infrastructure by reducing the cost of developing and provisioning client applications. It also empowers corporate mobile workers to be more productive by providing them with access to mission-critical data, from anywhere, at any time.

MEP includes the following features:

- Out-of-the-box connectivity to many enterprise application platforms
- Support for a wide variety of mobile devices
- Offline access to data when there is no network coverage
- Encryption and data fading/wiping extends enterprise application security to mobile devices
- Support for over-the-air provisioning of mobile clients
- Tools and templates that simplify the development and customization of mobile applications
- Based on open industry standards and robust and scalable Java technologies

MEP overcomes the following challenges:

- Enterprise customers use multiple devices. MEP supports a wide variety of mobile devices.
- Enterprise solutions are proprietary. MEP provides the ability to communicate with a variety of databases and EIS/EAI systems by offering a simple API that makes it easy to develop both client applications and Enterprise Connectors.
- Mobile users need to work in both a connected and a disconnected mode. MEP supports offline access to client data on the mobile device and the ability to connect to the EIS/EAI system as needed.

MEP Deployment Scenarios

MEP can be deployed in two different ways, in a mobile provider managed scenario and in an enterprise managed scenario.

- "Mobile Provider Managed Deployment Scenario" on page 12
- "Enterprise Managed Deployment Scenario" on page 13

Mobile Provider Managed Deployment Scenario

In a mobile provider managed scenario, such as the one shown in Figure 1–1, the Gateway engine and its associated sync database form the Gateway tier in the carrier's network. The corporate network includes the MEP Enterprise tier components and the EIS/EAI system.

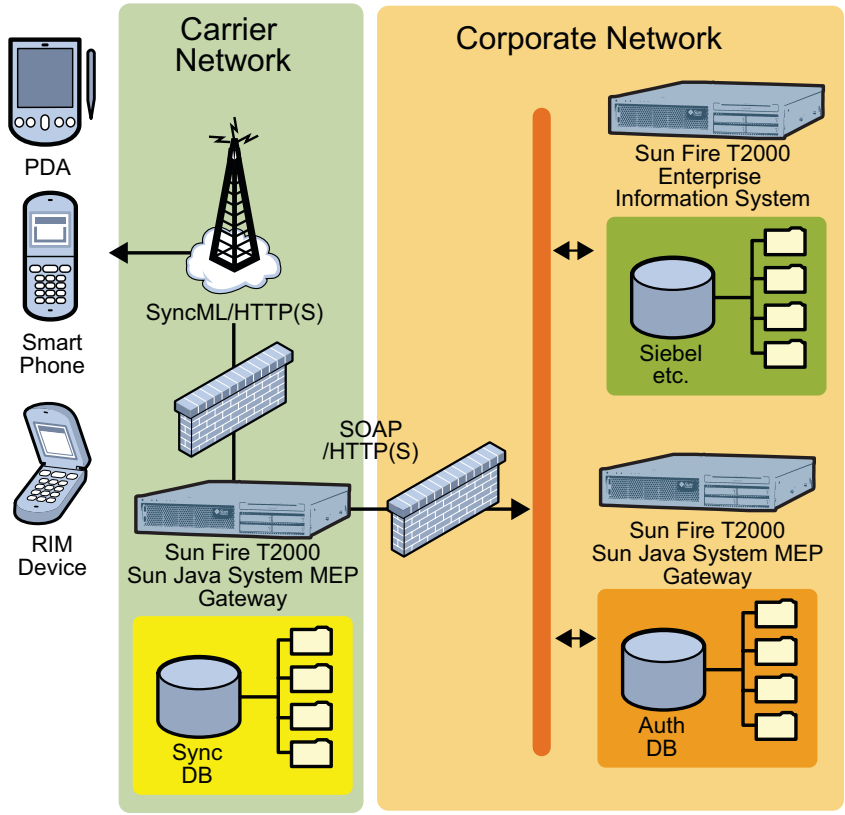

FIGURE 1–1 Mobile Provider Managed Deployment

Enterprise Managed Deployment Scenario

In an enterprise managed scenario, such as the one shown in Figure 1–2, all of the MEP components and the database or EIS/EAI system are in the corporate network.

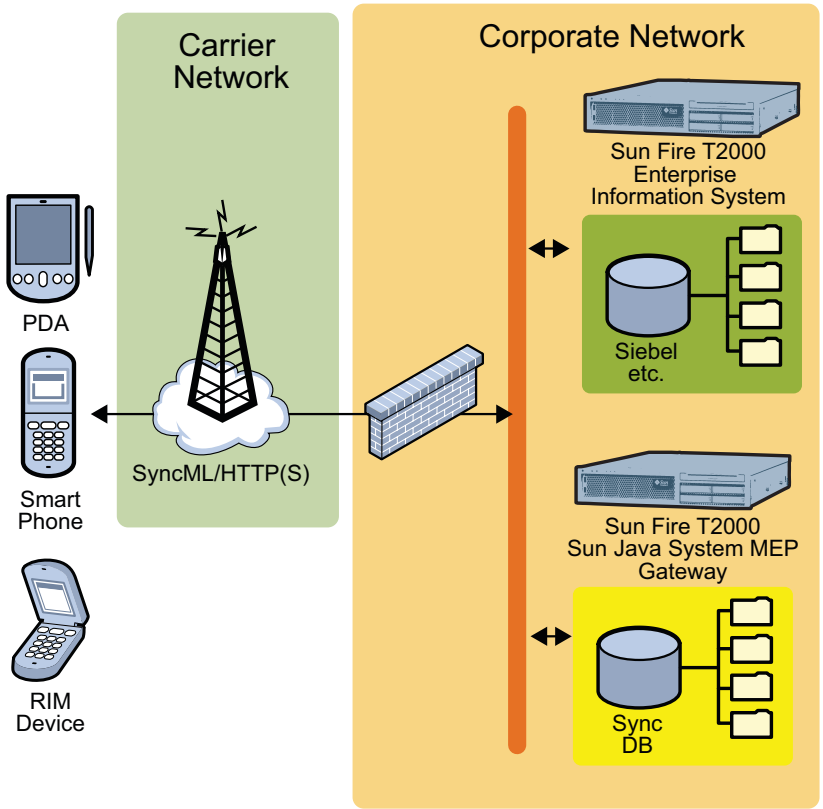

FIGURE 1–2 Enterprise Managed Deployment

The carrier's network includes no MEP components. The corporate network, in essence, controls all of the MEP components as well as the database or EIS/EAI system. In the corporate network, the MEP components can be configured in a single-tier or two-tier configuration. These configurations are described in "MEP Architecture" on page 14.

MEP Architecture

MEP supports synchronization of enterprise data between Open Mobile Alliance Data Synchronization (OMA DS) enabled mobile phones and a database or EIS/EAI system.

The MEP architecture is based entirely on open industry standards and is designed to operate a highly scalable, fault-tolerant environment tightly integrated with the existing infrastructure. MEP includes support for automatic failover and load-balancing, providing near-linear scalability. This architecture has been proven in carrier-grade deployments.

The major highlights of the MEP architecture are as follows.

Mobile Client Business Object API	The Mobile Client Business Object (MCBO) API provides an easy-to-use programming interface. It is provided as a Java ME library that the client application developer can use to develop a MEP Client application. See *Sun Java System Mobile Enterprise Platform 1.0 Developer's Guide for Client Applications* for information about using the MCBO API.
MEP Gateway	The MEP Gateway is the server component that interprets the incoming OMA DS messages and translates them into commands and data for Enterprise Connectors. The interface between the Gateway and the Enterprise Connectors is the Java Content Repository API.
Sync Database	The sync database contains the tables required by the MEP Gateway to store synchronization timestamps for client devices, mappings between client and server items, user information, configuration information, and synchronization message digest data.
Enterprise Connector Business Object API	The Enterprise Connector Business Object (ECBO) API provides an easy-to-use programming interface that makes it easy to build Enterprise Connectors. See *Sun Java System Mobile Enterprise Platform 1.0 Developer's Guide for Enterprise Connectors* for information about using the ECBO API.
Sun JCA Adapters	Sun JCA Adapters are Java EE Connector Architecture-based resource adapters that read and write data in the native format of specific databases or EIS/EAI systems. MEP includes adapters for SAP ERP, Siebel EAI, JDBC, and Oracle.

Figure 1–3 illustrates a single-tier MEP architecture. The client Sync App uses the APIs in the client library to communicate with the Gateway Engine, Enterprise Connector, and the Sun JCA Adapter. The adapter communicates with the EIS/EAI system.

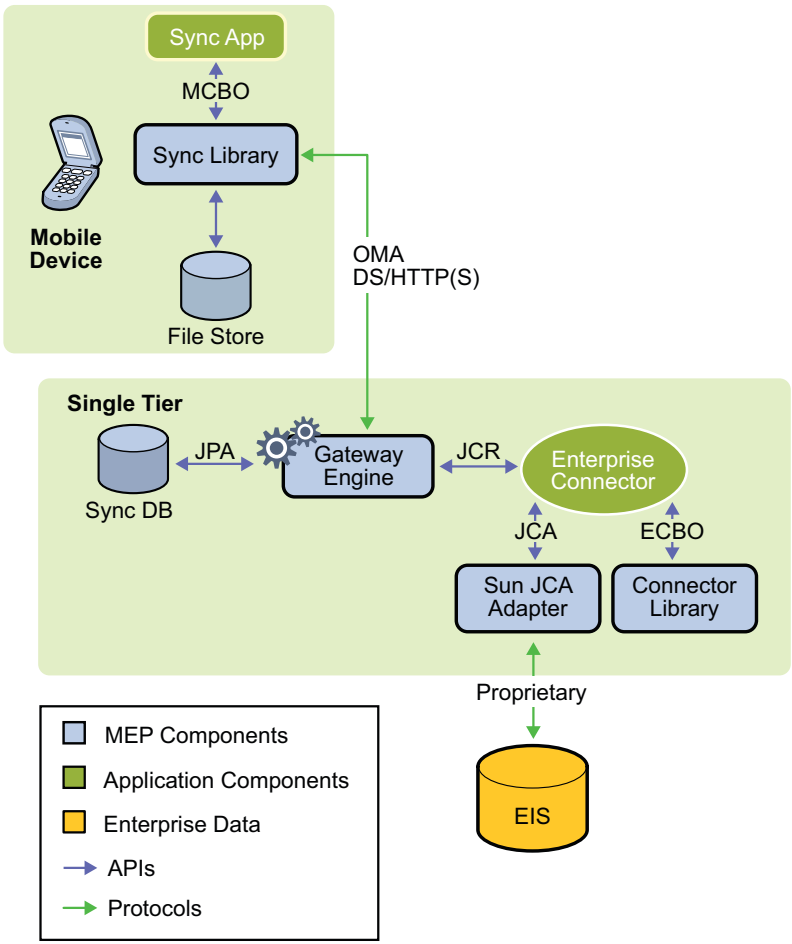

FIGURE 1–3 Single-Tier MEP Architecture

Figure 1–4 shows a two-tier architecture, consisting of a Gateway tier and an Enterprise tier. The Gateway tier includes the Gateway Engine and Web Service connector. The Enterprise tier includes the Web Service endpoint and Enterprise Connector, as well as the Sun JCA Adapter that communicates with the EIS/EAI system. The Web Service connector in the Gateway tier uses SOAP over HTTPS (or HTTP) to communicate with the Web Service endpoint in the Enterprise tier. The Enterprise Connector uses the ECBO APIs to communicate with the Connector Library and the JCA APIs to communicate with the Sun JCA adapter which, in turn, communicates with the EIS/EAI system.

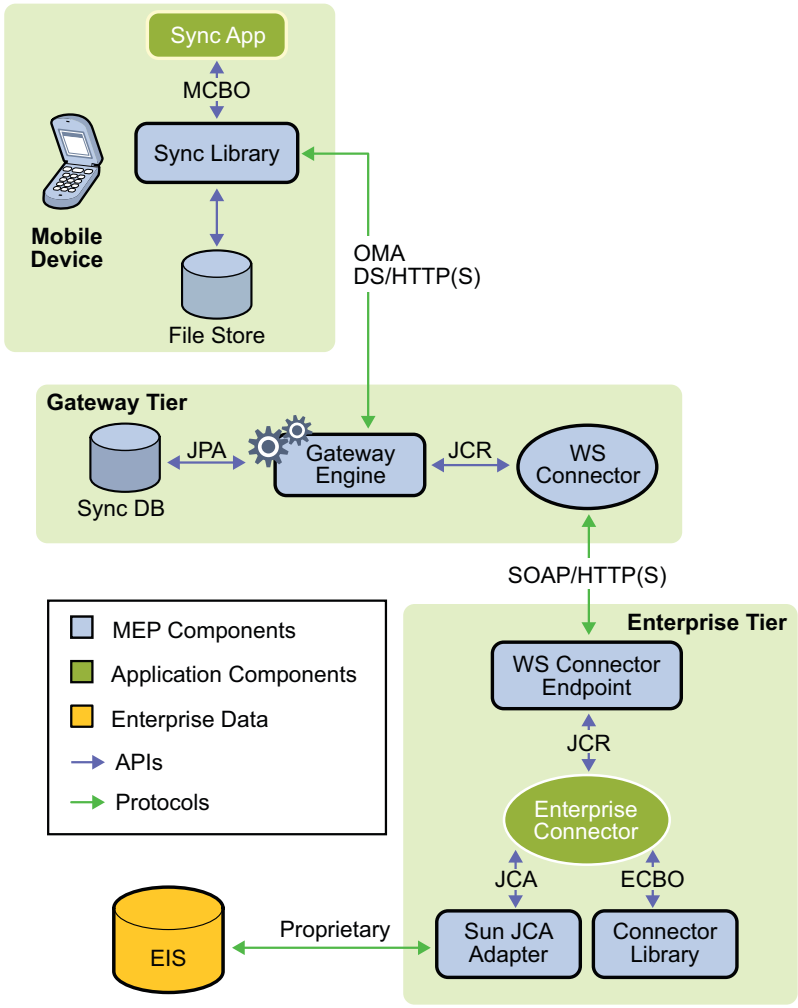

FIGURE 1–4 Two-Tier MEP Architecture

Synchronization Types

MEP supports the following types of client-initiated synchronizations:

- Both from server to client and from client to server:
  - "Two-way Sync (Fast Sync)" on page 18
  - "Slow Sync" on page 19
- From client to server only:
  - "One-way Sync from Client" on page 20

- "Refresh Sync from Client" on page 20
- From server to client only:
  - "One-way Sync from Server" on page 21
  - "Refresh Sync from Server" on page 21

There are two ways to initiate the synchronization process: the user can trigger it manually by selecting the appropriate menu item in the client device's Java ME client, or the server can initiate the process (Push). As defined by the Open Mobile Alliance (OMA), for OMA DS Push, the server sends an SMS notification message to the device. If the client has been programmed to act on this message, it can synchronize with the server to retrieve any changes.

Both the client and the server store information about changes to their respective data stores since the last successful synchronization. When the next synchronization is performed, the client and server negotiate how the changes are resolved and propagated according to the type of synchronization being performed.

The following sections describe the synchronization types.

Two-way Sync (Fast Sync)

Two-way sync, also called fast sync, is the normal synchronization mode, in which the client and the server exchange modifications to the data that they have stored. An initial slow sync is used to populate the data on the client.

The client always initiates this exchange by sending client data modifications to the server. This is called a synchronization request.

The server processes the synchronization request, comparing and unifying the data from the client with the data in the server database by means of an Enterprise Connector.

Afterwards, the server sends the modified data to the client, which updates the local data store with the information from the server. Figure 1–5 illustrates this process.

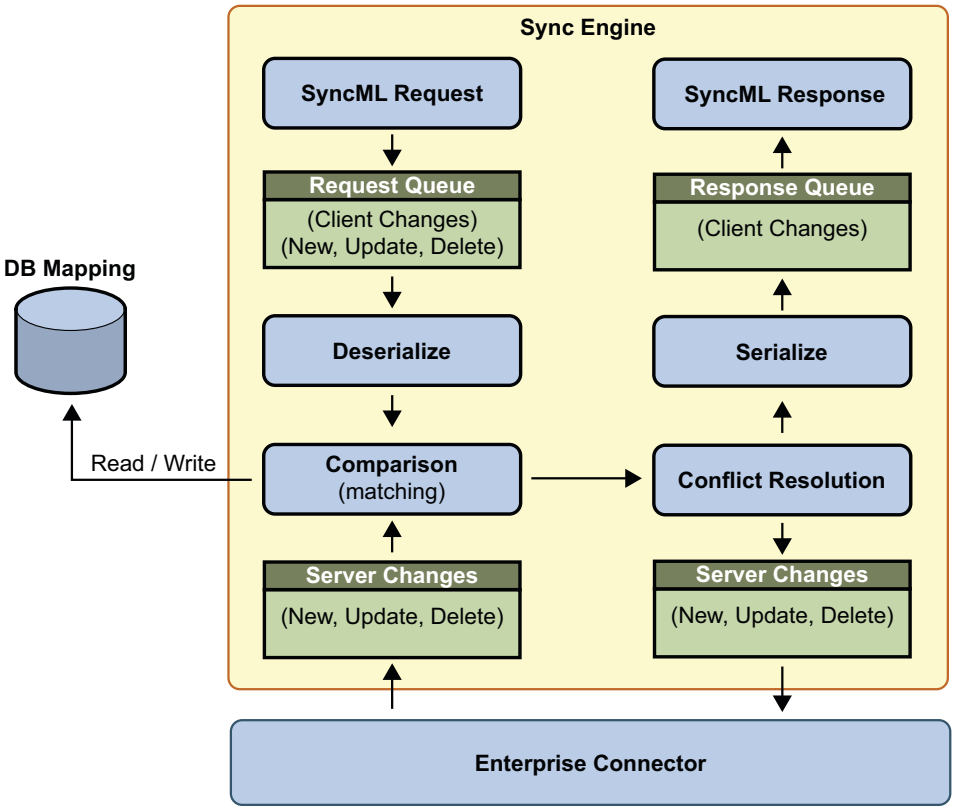

FIGURE 1–5 Two-way Sync

Slow Sync

The slow sync is similar to two-way sync, except that all the items in the client databases are compared with all the items in the server databases, on a field-by-field basis. A slow sync can be requested if the client and server data is mismatched or if the client or server loses its information.

Typically, the very first sync that a client performs is a slow sync. After that, the client performs fast syncs. If the data on the server or client is lost or corrupted, the next sync must be a slow sync.

In practice, a slow sync means that the client sends all its data to the server and the server does a field-by-field analysis, comparing its own data with that sent by the client. After the analysis, the server returns all the modification information to the client. In turn, the client returns the mapping information for all data items added by the server. Figure 1–6 illustrates this process.

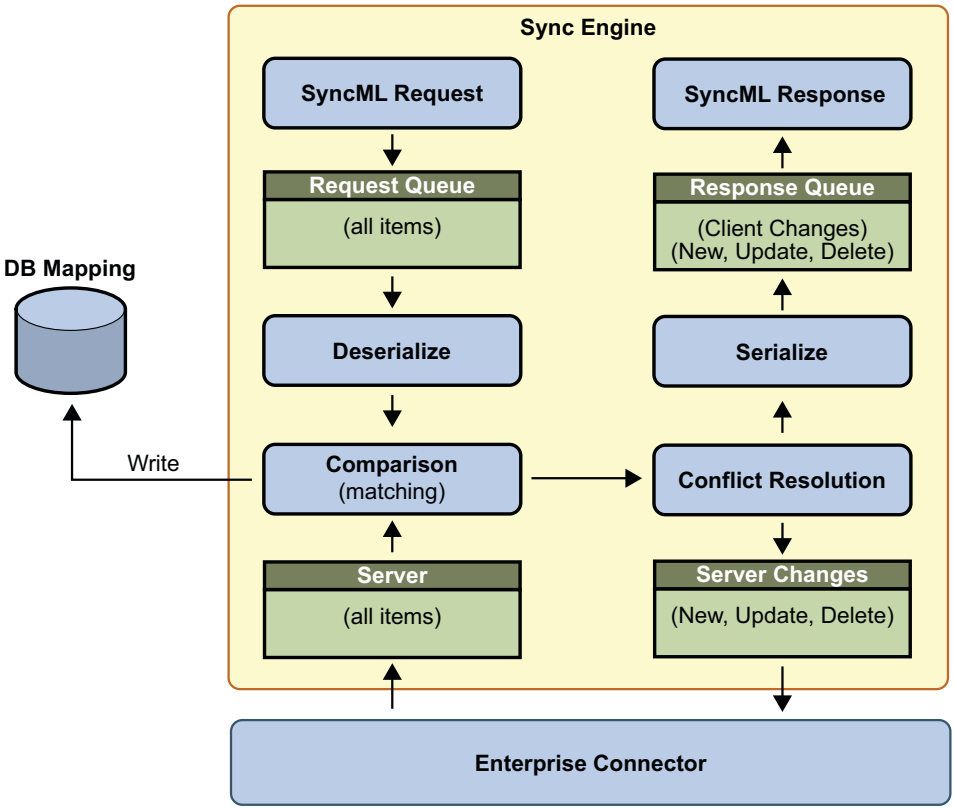

FIGURE 1–6 Slow Sync

One-way Sync from Client

This is one half of a two-way sync. In this mode, the client sends modifications of its data store to the server. The server updates its data store appropriately but does not send modifications of its data store to the client.

Refresh Sync from Client

In this mode, the client exports all its data to the server. The server is expected to replace all its data with the data sent by the client.

Note – Use this synchronization type with caution.

One-way Sync from Server

This is the other half of a two-way sync. In this mode, the server sends modifications of its data store to the client. The client updates its data store appropriately but does not send modifications of its data store to the server.

Refresh Sync from Server

In this mode, the server exports all its data from a database to the client. The client is expected to replace all its data with the data sent by the server.

Transport Layer

The transport layer for data synchronization between server and client can be HTTP or HTTPS.

- The HTTP protocol is used for insecure communication between the device and the Application Server, and for accessing the administration portal.
- The HTTPS protocol is used for a secure connection between the device and the Application Server.

MEP Security and Authentication

Sun makes every effort to ensure secure operation of MEP, which was designed with security in mind. MEP supports MD5 for encrypted authentication, and all traffic flowing through the public Internet is encrypted with SSL (HTTPS), ensuring that user data is at no time exposed to prying eyes. For security reasons, MEP does not duplicate the user's data to a local database, but only metadata required during the synchronization process.

MEP supports both client-side and server-side security:

- "MEP Client Security" on page 21
- "MEP Server Security" on page 22

MEP Client Security

MEP client security includes the following features:

- A simple PIN-based form of authentication
- A means to secure data at rest on the mobile device (data encryption)

- A means to securely synchronize with the Gateway Engine on the server (transport-layer security)
- A mechanism to destroy business data (data destruction)
- A means to prevent the client device from synchronizing (lockout)
- A means to remotely destroy all of the data on the device (poison pill)
- A means to notify the application that a certain quiet period has elapsed (data fading)
- An API that allows developers to replace the MEP default security manager implementation with their own

For details, see Chapter 3, "Client Security Architecture," in *Sun Java System Mobile Enterprise Platform 1.0 Developer's Guide for Client Applications*.

MEP Server Security

MEP server security features include the following:

- TLS/HTTPS is used to provide authentication and encryption between the device and the Gateway Engine. The OMA DS protocol requires support for basic authentication and for verification of data integrity using a message digest created with MD5. The use of transport layer security (HTTPS) is assumed.
- In a two-tier MEP installation, TLS/HTTPS is used for communication between the Web Service connector and the Web Service endpoint.
- The Gateway Engine incorporates an Application Server realm for user authentication. The default configuration uses a JDBC realm.

Client Provisioning

JSR 124, the Java EE Client Provisioning Specification, provides a framework and APIs for making client applications available on a Java EE server. MEP supports provisioning for client applications through its Administration Console. It provides a provisioning portal that offers the following features:

- **Content Management** - The provisioning portal manages the content repository and supports content versioning, a mechanism for administrators to upload applications, and the ability to search for applications.
- **Content Discovery** - The portal provides the user with a list of the available content and applications.

- **Application retrieval and installation** - Downloading the application is a two-part operation, handled by the application management system (AMS), the software in the device that manages the download, installation, execution, and removal of applications and other resources on the device. If an application description (in the form of a JAD file) exists, the AMS downloads it from the provisioning server's repository. Based on information found in the downloaded application descriptor, the AMS automatically downloads the application (the MIDlet suite JAR) from the repository. If required, the user is re-authenticated. If the application is retrieved successfully, installation is automatic.
- **Tracking** - Download status can be used to track the use of the application.

MEP Administration Console

MEP provides a web-based Administration Console that supports the following tasks:

- MEP product registration
- Addition, deletion, and modification of Gateway users
- Configuration of local and remote Enterprise Connectors
- Configuration of server push capabilities
- Management of client provisioning

The Administration Console provides different capabilities on each tier of a two-tier installation.

For details, see the *Sun Java System Mobile Enterprise Platform 1.0 Administration Guide*.

NetBeans IDE Tooling

NetBeans IDE plugin modules simplify the development of applications for MEP as follows:

- For Enterprise Connector development, NetBeans modules can generate Object Type Definitions (OTDs), which are object models that support the Sun JCA Adapters for specific databases and EIS/EAI systems.
- For Java ME client application development, NetBeans IDE with the Mobility Pack includes full integration with the Java ME Wireless Toolkit and provides a choice of UI frameworks (including LCD UI and SVG).