Fundamentals

# Sybase Unwired Platform 2.1

DOCUMENT ID: DC01204-01-0210-01
LAST REVISED: October 2011

# Contents

# CHAPTER 1 Enterprise Mobility

Extending your enterprise business processes, data, and information away from the office entails many technical challenges. To successfully mobilize your enterprise, it is important to balance device user requirements with enterprise requirements — Sybase® Unwired Platform helps you accomplish both.

Mobility solutions present complexity to the IT organization. These complexities may include, but not be limited to, lack of integrated solutions, heavy upfront costs for large-scale customization requirements, and the lack of repeatable development and deployment models. Sybase Unwired Platform supports development and deployment of two broad categories of mobile applications.

- **Online mobile applications –** This type of mobile application is device user-driven. Information access is adhoc in nature and users use a request-reply or lookup pattern. Customer lookup is an example of an online application.
- **Offline mobile applications –** This type of mobile application is driven by IT or Line-of-Business and is intended to help device users with overall organization efficiency. Information access is more formalized and task oriented. Information availability is mission critical regardless of connectivity. Sales process enablement is an example of an offline application.

Common IT and management capabilities are also required, covering the following areas:

- User onboarding services
- Device, data, and transport security
- Error reporting and troubleshooting support
- Remote device management support

Both mobile application types, complimented by common IT and management capabilities help to meet enterprise mobility requirements and device user requirements.

## Enterprise Mobility Requirements

Enterprise requirements must be met with mobile applications that are developed and the runtime into which they are deployed.

Sybase Unwired Platform, with Sybase Mobile SDK and Unwired Platform Runtime, provides functionality and features to address:

*Connectivity*

- Connecting and interacting with various enterprise information systems and applications that exist in any large or medium size corporate IT infrastructure that supports customers, employees, sales, partners, suppliers, and executives.

*Security*

- Data partitioning for mobile applications; securing data while in transport and while accessed by the application.
- Ensuring secure access to data and application functionality that is available only to authorized mobile users.

*Device and App Management*

- Supporting the rapidly expanding and changing selection of devices and device operating systems on the market.
- Implementing a data model that fits the application requirements and physical restrictions and capabilities of the mobile device, such as network bandwidth, storage capacity, connectivity, security, and availability of consistent data.
- Reliably propagating user transactions initiated on a device that may or may not have network connectivity.
- Managing device and application life cycles, including device upgrades and turnover, deployment of application updates and new applications, with limited interruption to productivity.

*Development Tools to Minimize Total Cost of Development*

- Using development tools to leverage existing developer expertise, technologies, and commonalities, such as data models and objects, to effectively produce mobile applications for a variety of device types and device platforms.

## Device User Requirements

To maximize productivity, user requirements are important considerations when mobilizing the enterprise. Balancing user requirements with enterprise requirements is also key.

Common device user requirements may include:

- Device choice. More organizations now support this as well as the ability to configure a personal device for work productivity and vice-versa.
- Ease of use. Devices and applications that are easy to configure and use. App interfaces must be intuitive and integrate with device features, such as automatic layout, touch screen or trackball navigation, etc.

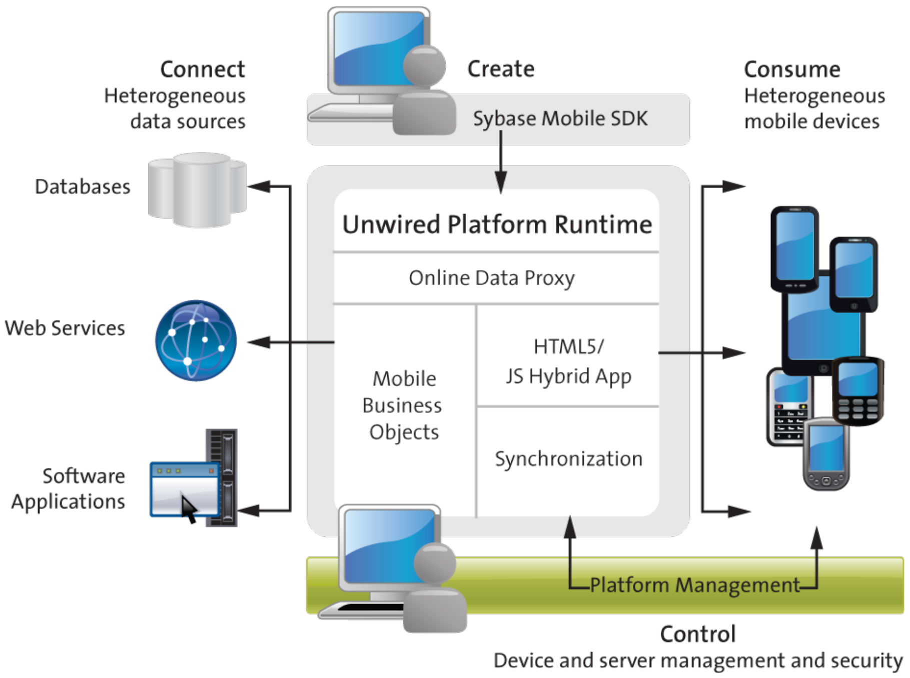
- Flexibility. The ability to load and upgrade applications, receive notifications, and perform work when connectivity is unavailable.

# CHAPTER 2 Platform Overview

Sybase Unwired Platform provides an integrated platform solution to extend your enterprise applications to mobile workers in the field.

Sybase Unwired Platform acts as the hub that connects the back-end enterprise systems and data sources to mobile devices. Features for mobile application development, deployment, security, and ongoing device and application management provide a complete end-to-end solution. The platform is made up of:

- **Sybase Mobile SDK –** The platform development tool set used to build mobile applications that meet your mobility needs.
- **Sybase Unwired Platform Runtime –** The deployment and management architecture and services used to run and manage mobile applications.



This solution allows you to:

- **Connect –** During development and deployment, connect to your heterogeneous data sources and back-end enterprise systems.

- **Create –** Use the development tools included with the Sybase Mobile SDK to build and test mobile applications that meet your mobility needs.
- **Control –** Deploy to and manage Unwired Platform Runtime, including the runtime environment, end-to-end security, and device applications.
- **Consume –** Mobile applications install to devices allowing device users to work online and offline. Enterprise data is accessed from a variety of mobile devices.

The chapters in *Fundamentals* guide you through details of this mobility platform.

- *Modeling Enterprise Data for Mobile Application Development*

  There are two ways to model enterprise data so that mobile applications can be developed to access data in your enterprise: Mobile Business Objects via the Sybase Mobile SDK and OData via SAP® Gateway.
- *Sybase Mobile SDK*

  Sybase Mobile SDK provides development support for Native Object API Applications, HTML5/JS Hybrid Applications, and OData SDK Applications.
- *Unwired Platform Runtime*

  Unwired Platform Runtime provides the platform infrastructure that allows you to deploy and manage your mobile applications.
- *Documentation Roadmap*

  Use the documentation roadmap to review the documentation available for Sybase® Unwired Platform and determine where to start based on your role.

CHAPTER 3

# Modeling Enterprise Data for Mobile Application Development

There are two ways to model enterprise data so that mobile applications can be developed to access data in your enterprise: Mobile Business Objects via the Sybase Mobile SDK and OData via SAP® Gateway.

- *Mobile Business Objects*

  The cornerstone of the solution architecture is the concept of the mobile business object (MBO). For native Object API applications and mobile workflows, mobile business objects form the business logic by defining the data you want to use from your back-end system and exposing it through your mobile application or workflow.
- *OData*

  The SAP NetWeaver Gateway exposes OData with extensions specific to SAP. Service documents, that describe service interaction and data, allow users to interact with the Gateway runtime, which then interacts with the SAP application suite.

## Mobile Business Objects

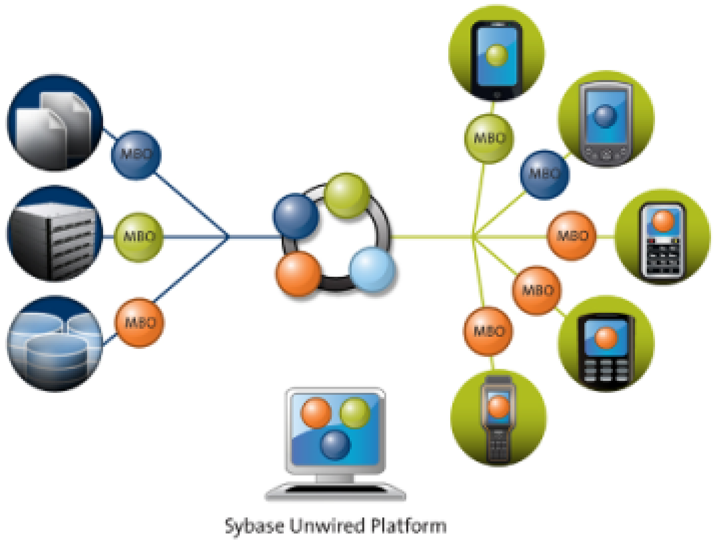
The cornerstone of the solution architecture is the concept of the mobile business object (MBO). For native Object API applications and mobile workflows, mobile business objects form the business logic by defining the data you want to use from your back-end system and exposing it through your mobile application or workflow.

MBO development involves defining object data models with back-end EIS connections, attributes, operations, and relationships that allow filtered data sets to be synchronized to mobile devices. MBOs are built by developers familiar with the data and transactional requirements of the mobile application, and how that connects to the existing EIS data sources.

A mobile business object (MBO) is derived from a data source (such as a database server, Web service, or SAP® server). MBOs are deployed to Unwired Server, and accessed from mobile device application clients. MBOs:

- Are created using the Unwired WorkSpace graphical tools. These tools simplify and abstract back-end system connections, and provide a uniform view of transactional objects
- Are reusable, allowing you to leverage business logic or processes across multiple device types.
- Future-proof your application; when new device types are added, the same MBO can be used.

- Provide a layer of abstraction from Unwired Server's interaction with heterogenous back ends/devices, as shown in the following diagram.



MBOs are developed to include:

- Implementation-level details – metadata columns that include information about the data from a data source.
- Abstract-level details – attributes that correspond to instance-level properties of a programmable object in the mobile client, and map to data source output columns. Parameters correspond to synchronization parameters on the mobile client, and map to data source arguments. For example, output of a SQL SELECT query are mapped as attributes, and the arguments in the WHERE clause are mapped as synchronization parameters, so that the client can pass input to the query.

  MBO operations include parameters that map to data source input arguments. Operation parameters determine information a client passes to the enterprise information system (EIS).
- Relationships – defined between MBOs by linking attributes and parameters in one MBO, to attributes and parameters in another MBO.

Developers define MBOs using either a top-down approach—first designing attributes and parameters, then binding them to a data source; or a bottom-up approach—first specifying a data source, then automatically generating attributes and parameters from it.

A mobile application package includes MBOs, roles, and data source connection mappings, and other artifacts that are delivered to the Unwired Server during package deployment.

When the data model is complete, code artifacts are generated. The MBO package, containing one or more MBOs is deployed to Unwired Server. Other MBO artifacts are used to develop a mobile application using Native Object API or HTML5/JS Hybrid App API — when the application is deployed to a device, the MBO data set resides on the device. On device data changes are synchronized to the MBO on the server and then to the EIS back end. Back end changes are communicated to the device via the MBO on the server that sends a notification to the device and updates the MBO data on the device.

The following sections cover MBOs from a high level; for more detail, see:

- *Sybase Unwired WorkSpace - Mobile Business Object Development*
- *Developer Guide: Unwired Server*

**See also**

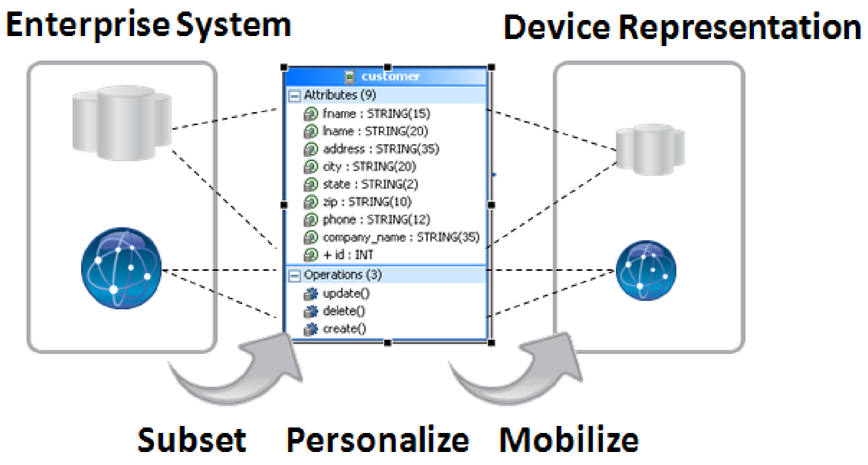
- *Native Object API Applications* on page 17
- *HTML5/JS Hybrid Apps* on page 18

## Data Modeling

After connecting to back-end data sources using connection profiles from Unwired WorkSpace, the MBO developer defines the mobile interaction pattern, which typically involves selecting data subsets.

The MBO is developed to define each data subset, describe the data and the operations on the data.

- The data subset is deployed with the MBO package to Unwired Server where the MBO manages synchronization between the EIS and Unwired Server.
- Artifacts generated from the MBO are then used to develop the mobile application, typically further defining the data subset for data representation on the device.



## Mobility Patterns

MBO design allows developers to build in support for mobility patterns—interactions between the EIS, the MBO, and the device cache. These patterns include data virtualization, publication, subscription, and operation replay.

- Virtualization – normalizes the data and semantics for interacting with different enterprise information systems (EISs), each with its own set of connection interface, data, operation, and type structures.

  An MBO provides a layer between your enterprise databases or applications, and the remote database on the device client. Data virtualization utilizes a cache database (CDB) to optimize device client access and minimize back-end resource utilization.
- Publication – data is synchronized between EIS data stores and the Unwired Server cache, to ensure that the content is consistent between systems.
- Subscription – links the client with a publication and allows the data described by the publication to be transferred to the device.
- Operation replay – supports execution of the client-initiated transactions that result in data changes on the EIS.

## Attributes

Attributes define the structure for the data associated with the MBO instance on the mobile device.

Attributes define the scope of the device-side data store. Attributes and parameters in an MBO define the server cache. The server cache and device data store are populated by reading data from the EIS, such as a SQL select statement for a database data source.

Attributes also have additional metadata, such as specification of a synchronization parameter option that can be used to provision user-specific data. For example, a sales representative in the Eastern region may be interested only in seeing data for that region. Developers can build the application to map those preferences to MBOs to drive the data filtering specific to the user. The same data obtained from the EIS is then further partitioned and used to serve all users, thereby optimizing requests for data to the EIS and improving performance of the mobile application.

## Operations

MBOs may incorporate operations that change the data retrieved from the enterprise information system (EIS).

- Create, Update, Delete (CRUD) operations – an operation definition contains parameters that map to the arguments of the EIS operation, and can create, update, or delete data. These operations cause state change.
- Other operation – an operation definition for operations other than create, update, or delete. These operations do not cause state change.

The operation definition supports validation and error handling.

## Relationships

Relationships define the data association between two MBOs by linking attributes and parameters in one MBO to attributes and parameters in another MBO.

Relationships help provision related data as one unit, and properly sequence the operations on the related MBOs based on real-time detection of the object instances used. Relationships can be one-to-many, many-to-one, or one-to-one.

From an attribute standpoint, bidirectional relationships are supported; and from an operation execution standpoint, composite relationships are supported.

## Other Key Concepts

Other key concepts for understanding mobility include object queries, synchronization parameters, result set filters, result checkers, and personalization keys.

- Object queries – a SQL statement associated with a mobile business object (MBO) that runs on the client against data that was previously downloaded to the device. The object query returns a filtered result set (such as a single row of a table). Object queries enable discrete data handling.
- Synchronization parameters – metadata that defines how the values provided by the device application client are used to filter data, which is downloaded to the device, to provide data of interest to the user. Synchronization parameters may be used to filter the cached data, or mapped to load parameters to filter the data that is cached for the MBO and then served to the client application.
- Result-set filters – a Java API used to customize the data returned by an enterprise information system (EIS) operation. Developers can use result-set filters to alter or manipulate returned data and its structure before it is placed in the server's cache.
- Result checkers – a Java API that implements operation execution checks, and error handling logic for the MBO operation. Developers can use result set checkers to implement customized error checking for MBOs, since not all MBOs use a standard error reporting technique.
- Personalization keys – metadata that enables users to store their search preferences on the client, the server, or by session. The preferences narrow the focus of data retrieved by the mobile device (also known as the filtering of data between client and Unwired Server).

  Often personalization keys are used to hold backend system credentials, so that they can be propagated to the EIS. To use a personalization key for filtering, it must be mapped to a synchronization parameter.

  The developer can also define personalization keys for the application, and can use built-in personalization keys available in Unwired Server. Two key built-in personalization keys —username and password—can be used to perform single sign-on from the device application to the Unwired Server, authentication and authorization on Unwired Server, as well as connecting to the back-end EIS using the same set of credentials. The password is never saved on the server.

## Data Caching

Data caching is initial loading (or filling) the cache database (CDB, or Unwired Server cache) with enterprise information system (EIS) data, then continuing to refresh the CDB with changes from the EIS or mobile device on an ongoing basis.

Since continual synchronization of the data between the EIS and device puts a load on the EIS, Unwired Platform provides several options for loading and refreshing the data cache.

Options include narrowing the EIS data search so that only specific data is retrieved, identifying effective policies for handling data updates once operations are performed, scheduling periodic updates to occur when system usage is low, updating only changed data in the CDB, and so forth.

You can use multiple options to load and refresh the right data at the right time, and to deliver the smallest, most focused payload to the mobile device.

The primary loading schemes provide a trade-off between time and storage space. For example, bulk loading takes more time because data is loaded for all users, but once loaded, the data can be shared between users.

- Bulk loading – data for all users is loaded in bulk.
- Partitioning data loading – only user relevant data, or partition, is loaded.
- On-demand loading – data is not filled, until the client performs synchronization.
- Scheduled loading – data is filled periodically according to a schedule you set.

Data change notification (DCN) facilitates propagation of data changes from the back-end enterprise information system (EIS) to the Unwired Server or HTTP/HTTPS interface for any MBO. The DCN payload containing changed data is applied to the cache and the change gets published to subscribing device users based on a change detection interval time. DCN is used for cases where the cache refresh may be expensive due to volume. This option is the least intrusive and most optimal for addressing high-load environments and optimizing the load on the EIS data sources to keep the Unwired Server cache consistent.

## Object Code Generation

To access and integrate MBOs in a device application, developers generate object code for the target device platform, and then use their IDE of choice to build the native device application. The object code generation step is the bridge from the Unwired Server server-side development (MBOs) to client-side development (device applications).

The generated object code for each MBO follows a standard pattern of properties for attributes, operations, and abstractions. Object code generation is supported in the native language for each target platform. Unwired Server client libraries complement and are required for the generated object code, which together are used in the device application.

The generated code is built upon the Unwired client libraries integrated part of the server, which combine to support reliable transmission of data and transactions, security of data while in transit or on device, sending notifications when data changes occur in the back-end application, consistent interface on all platforms, all of which abstract developers from aforementioned mobility related complexities.

Code generation is supported for these platforms: BlackBerry (Java), iOS (Objective-C), Windows Mobile (C#), and Windows (C#).

See *Supported Hardware and Software* for the most current version information.

### Deployment

The last step of mobile business object (MBO) development is to deploy the MBO definitions to Unwired Server as a deployment unit generated from a design-time deployment package using Sybase Unwired WorkSpace.

When you deploy MBOs to the Unwired Server, you are deploying:

- MBO definitions including attributes, operations, connections, role mappings, schedule groups, cache groups as defined in the package.
- MBO custom code related to object filters, result-set filters, and result checkers.
- Appropriate generated server-side artifacts that support the interaction with the EIS back-ends and device application.
- Other functionality captured in the MBO model.

MBOs are deployed using a deployment wizard through which you can make the choices that are appropriate for application requirements. Developers use Unwired WorkSpace to deploy a package.

The production administrator can deploy from a wizard using the web-based management console, or from the command line. Deployment-time tasks include choosing:

- Target domain – logical container for packages.
- Security configuration – used for authentication and authorization of users accessing the package.
- Role-mappings – to map logical roles to the physical roles of the back-end repository.
- Server connections mapping – to bind MBOs design-time data sources to production data sources.

## OData

The SAP NetWeaver Gateway exposes OData with extensions specific to SAP. Service documents, that describe service interaction and data, allow users to interact with the Gateway runtime, which then interacts with the SAP application suite.

Where an OData source is used as the back-end the application developer does not need to create any model elements (MBO’s) in the tooling but rather inherits a service model from the service document published from SAP Gateway. These OData service documents contain all the information the device developer needs to parse and interact with these data streams.

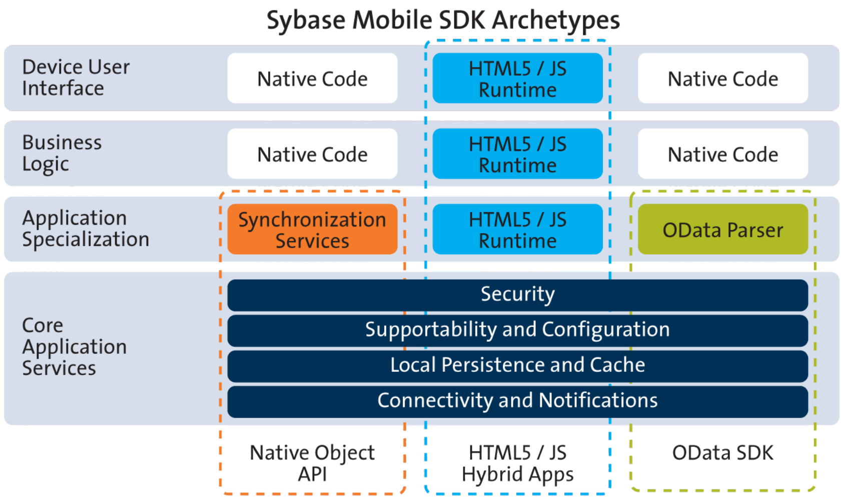
Sybase Unwired Platform provides administrative facilities for discovering OData service documents and allowing devices to communicate with those enterprise services.

See documentation for SAP NetWeaver and for SAP Gateway for additional information.

# CHAPTER 4 Sybase Mobile SDK

Sybase Mobile SDK provides development support for Native Object API Applications, HTML5/JS Hybrid Applications, and OData SDK Applications.

The end result of the development process is a packaged mobile application ready for provisioning to devices. The following diagram shows the device application structure and services provided for each of the development approaches with Sybase Mobile SDK.



Dashed outlines in the diagram represent the services provided by Sybase Mobile SDK for each application type. Native code development outside of the Mobile SDK uses APIs that support core services and application specialization, in addition to leveraging in-house device platform development expertise.

- **Core Application Services –** The set of common services provides a consistent foundation upon which mobile apps are developed. These common device services make development easier and more secure.

  Security features are embedded into the SDK to support secure storage of certificates, use of the artifacts in authentication, single sign-on, and other features related encryption of the database. APIs are supplied to support certificate store, logon certificate, and the data vault. Each application type makes use of the same set of security features.
- **Application Specialization –** Application specialization is the differentiator between development approaches. Based on these distinct application services and your business needs, develop one or more application types.

| Develop Apps with | When your business needs |
|---|---|
| Native Object API SDK | Offline applications: Mission-critical and more complex. Users work on data offline and synchronize updates to the server when connected. |
| HTML5/JS Hybrid App SDK | Online applications with Push: Simple queries and workflows that follow a request-reply or lookup pattern. |
| OData SDK | Online applications with Push that use OData protocol: Data, modeled in SAP Gateway, is accessed in a request-reply or lookup pattern. |

Synchronization implies an intermediary store of cached data and the process by which data changes are propagated between the server, the cache, and the device storage with the goal of ensuring that the data is replicated in each location.

Push leverages a proprietary message transport to move data between device and server. The data change is initiated, or pushed, from where ever the data change takes place: from the device if the change takes place there, or from the server if the change takes place there.

- **Business Logic and Device User Interface –** The user interface is developed:
    - Using the developer IDE of choice for the target device platform and either the Native Object APIs or the OData SDK APIs.
    - Using the Mobile Workflow Forms Editor to design HTML5/JS.
- *Native Object API Applications*

  Native Object API applications are customized, full-featured mobile applications that use mobile business objects (MBOs) to facilitate connection with a variety of enterprise systems and leverage synchronization to support offline capabilities.
- *HTML5/JS Hybrid Apps*

  HTML5/JS Hybrid Apps support simple business processes, such as approvals and requests, and also use mobile business objects (MBOs) to facilitate connection with a variety of enterprise systems. With this approach, a container is developed and deployed to a device, then one or more workflows are deployed to the container. This approach supports mobile workflow enablement, which enables mobile device users to operate as workflow participants, allowing the mobile user to start and respond to back-end enterprise requests within a generic framework.
- *OData SDK Applications*

  OData SDK Applications are simple, mostly online mobile business applications that leverage standards-based approaches—OData protocol and RESTful Web Services. This approach provides instant quick value applications meant for end user productivity.
- *Development Life Cycle*

  The "development life cycle" takes developers through the process of designing, developing, testing, deploying and maintaining mobile business objects, device applications, and workflow packages.

## Native Object API Applications

Native Object API applications are customized, full-featured mobile applications that use mobile business objects (MBOs) to facilitate connection with a variety of enterprise systems and leverage synchronization to support offline capabilities.

The native Object API application model enables the developer to write custom code — C#, Java, or Objective-C, depending on the target device platform — to create a device application.

Development of this type of application provides the most flexibility in terms of leveraging the common application services, but each application must be provisioned individually after being compiled, even for minor changes or updates.

Development involves both server-side—MBO development—and client-side—native application—components. Unwired Server brokers data synchronization and transaction processing between the server and the client components.

- Server-side components address the interaction between the enterprise information system (EIS) data source and the data cache. EIS data subsets and business logic are encapsulated in artifacts, called mobile business objects, that are packaged and deployed to Unwired Server.
- Client-side components are built into the mobile application and address the interaction between the data cache and the mobile device data store. This can include synchronization frequency, data change notifications, and storage of transactions created offline.

These applications:

- Allow users to connect to data from a variety of EIS systems, including SAP systems.
- Build in more complex data handling and logic.
- Leverage data synchronization to optimize and balance device response time and need for real-time data.
- Ensure secure and reliable transport of data.

### See also

- *Mobile Business Objects* on page 7

## Native Object API Development

The client object APIs form the core building block of Native Object API Applications and provide the common set of APIs for consistency across device platforms, thus following the "design once and deploy anywhere" paradigm of building applications.

Client Object API categories for Native Object API Application development include:

- Object APIs that support access to MBOs attributes, operation execution, retrieving data, state information, and so forth. The APIs offer abstraction for accessing data from the EIS,

sending operation executions to the EIS, updating and deleting data, and accessing persistent store in the object-oriented paradigm, among other things.
- Object APIs that offer consistent access to core application services across all device platforms. The APIs provide abstraction for the complex technical details of reliable communication between the device and Unwired Server. This includes a variety of network conditions for optimal transmission of data and state information, connection APIs to the back-end infrastructure, secure access to the Unwired Server data, publishing of data notifications and processing of such notifications, data protection, and access to UI management APIs and custom UI controls, such as signature control, among other things.

The common development task flow for Native Object API Applications includes these steps:

1. Verify that mobile business objects have been developed and are available.
2. Configure the platform-appropriate development environment, such as BlackBerry JDE, Xcode for iOS, and C# for Windows and Windows Mobile.
3. Generate Object API code for mobile business objects and import the artifacts—libraries and code—to the development environment.
4. Develop the device application.
5. Package the application code and prepare the package for deployment.
6. Test the device application using a simulator/emulator.
7. Deliver the device application package to the Sybase Unwired Platform Administrator for production testing and provisioning.

Complete details of this development approach and Native Object API are documented in *Developer Guide: <Device Platform> Native Applications*.

## HTML5/JS Hybrid Apps

HTML5/JS Hybrid Apps support simple business processes, such as approvals and requests, and also use mobile business objects (MBOs) to facilitate connection with a variety of enterprise systems. With this approach, a container is developed and deployed to a device, then one or more workflows are deployed to the container. This approach supports mobile workflow enablement, which enables mobile device users to operate as workflow participants, allowing the mobile user to start and respond to back-end enterprise requests within a generic framework.

These applications:

- Manage a low data volume
- Provide a simple user experience
- Implement simple business logic
- Avoid the need for long-lasting, offline, stateful transactions
- Ensure secure and reliable transport of data

With Hybrid App development, the server-side of the application is metadata-driven and the client-side of the application is a fully generated Web application package. The focus is on how data is rendered to the device user; data is made available using a request-response pattern, without synchonization. Light weight applications and mobile workflows are developed to provide business logic and interaction with MBOs.

A native container application is packaged with a Web Browser plug-in and built in core application services such as connectivity, guaranteed messaging, caching, and security. The container is provisioned to mobile devices. Data transport and access relies on messaging protocol between the server and the container on the device, invoking either online operations to the back end, or cached MBO data on the Unwired Server.

The mobile workflow package is compiled —consisting of platform-independent HTML, JavaScript and CSS resources— and can be deployed automatically to the container, without writing any code. Device and application services include offline cache, reliable messaging, and secure store.

The container is deployed to a device and provides the runtime from which request-response decision patterns are executed. It supports three basic patterns where one, or a combination of these patterns is applied to implement each use case:

- **Notifications –** A server-initiated action, performed in the EIS in the context of a business process, sends a notification to device users
- **Lookup –** Device users initiate a request for information from the EIS
- **Action/Decision Forms –** Device users submit a form to make a decision, such as an approval, that results in some EIS business process state transition

The container hosts an embedded browser and launches the individual mobile workflow applications. The workflows are assigned to users by administrators. Once assigned, those workflows can be initiated by the user (client-initiated) or automatically triggered as a result of a back-end event that is sent to the Unwired Server as a data change notification request (server-initiated).

### See also

- *Mobile Business Objects* on page 7

## HTML5/JS Hybrid App API Development

An HTML5/JS App includes both the Mobile Workflow— business logic, the data itself, and associated metadata that defines data flow and availability, and the Container — device-resident presentation and logic. Within Sybase Unwired Platform, development tools enable both aspects of HTML5/JS Hybrid App development.

Mobile Workflow applications can reference one or more MBOs and can include load parameters, personalization, and error handling.

Once you have developed MBOs and deployed them to Unwired Server, develop device-resident presentation and logic for your Mobile Workflow application using the Mobile Workflow Forms Editor.

The common development task flow for HTML5/JS Hybrid Apps includes these steps:

1. Verify that mobile business objects have been developed and are available.
2. Start the developer environment: launch Sybase Unwired WorkSpace, import the project containing MBOs, connect to Unwired Server and deploy the project.
3. Open the Mobile Workflow Forms Editor and design your worklfow.
4. Package the workflow code and prepare the package for deployment.
5. Test the workflow using a simulator/emulator.
6. Deliver the workflow package to the Sybase Unwired Platform Administrator for production testing and provisioning.

Complete details of this development approach and HTML/JS Hybrid App API are documented in:

- *Sybase Unwired WorkSpace - Mobile Workflow Package Development*
- *Developer Guide: Mobile Workflow Packages*

.

## OData SDK Applications

OData SDK Applications are simple, mostly online mobile business applications that leverage standards-based approaches—OData protocol and RESTful Web Services. This approach provides instant quick value applications meant for end user productivity.

OData SDK applications are developed to leverage a proxy connection to SAP EIS systems, such as SAP Business Suite, and are deployed using the Online Data Proxy runtime option, essentially making realtime SAP data available to device users. The proxy connection uses the RESTful OData protocol to connect through Unwired Server and SAP Gateway to SAP enterprise information systems. Data is made available using a request-response pattern, without synchronization.

The proxy connection uses OData with SAP additions:

- OData is the Microsoft-owned open protocol —a resource-based web protocol for querying and updating data. It defines operations on resources using HTTP verbs (GET, PUT, POST, and DELETE) and identifies those resources using a standard URI syntax. Data is transferred over HTTP using the Atom or JSON format.
- OData for SAP Products provides SAP Extensions to the OData protocol that enable users to build user interfaces for accessing the data published via OData. This allows interfaces to define human-readable, language-dependent labels for all properties and free-text search within and across collections of similar entities (OpenSearch).

OData SDK applications running on mobile devices use semantic annotations to tell the client which of the OData properties contain specific data, such as a phone number, a part of a name or address, or something related to a calendar event. This allows seamless integration with contacts, calendar, and telephony on the mobile device. A metadata document defines whether an entity set is searchable, which properties may be used in filter expressions, and which properties of an entity will always be managed by the server.

These applications:

- Connect mobile devices more directly to SAP business systems, essentially making OData SDK applications online. Data is not persisted, but can be cached.
- Use SAP data and business process models
- Leverage subscription and push notification features
- Implement SAP single sign-on (SSO)
- Ensure secure and reliable transport of data.

### OData SDK Development

The OData SDK development approach uses an OData source as the back end and allows device users to query and update data in an essentially online application.

Instead of being based on data model elements (MBOs), the OData SDK developer inherits a service model from the service document published from SAP Gateway. These OData service documents contain all the information the developer needs to parse and interact with these data streams.

Developers use APIs related to components for parsing, caching, persistence, connectivity, and configuration to develop OData SDK Applications.

The common development task flow for OData SDK Applications includes these steps:

1. Verify that service documents have been published from SAP Gateway and are available.
2. Configure the platform-appropriate development environment, such as Android SDK, BlackBerry JDE, and Xcode for iOS.
3. Develop the device application.
4. Package the application code and prepare the package for deployment.
5. Test the device application using a simulator/emulator.
6. Deliver the device application package to the Sybase Unwired Platform Administrator for production testing and provisioning.

Complete details of this development approach and OData API are documented in *Developer Guide: OData SDK*.

## Development Life Cycle

The "development life cycle" takes developers through the process of designing, developing, testing, deploying and maintaining mobile business objects, device applications, and workflow packages.

Sybase Mobile SDK is intended to work with your existing development paradigm. Use Sybase Mobile SDK as you would use any development tool—integrated with your existing change control and versioning mechanisms, and with your development, test, and production environments.

## Life Cycle Stages

Life cycle refers to the distinct stages of application development and upgrade. In the context of this document, life cycle refers to the application life cycle changes, focusing especially on initial release of the application, additional releases, and patches to the production system.

This is not a comprehensive description of the application development life cycle, but focuses on important information for making decisions about whether to create a new release, or simply patch the production system.

**Figure 1: Life Cycle Development for Milestone Development**

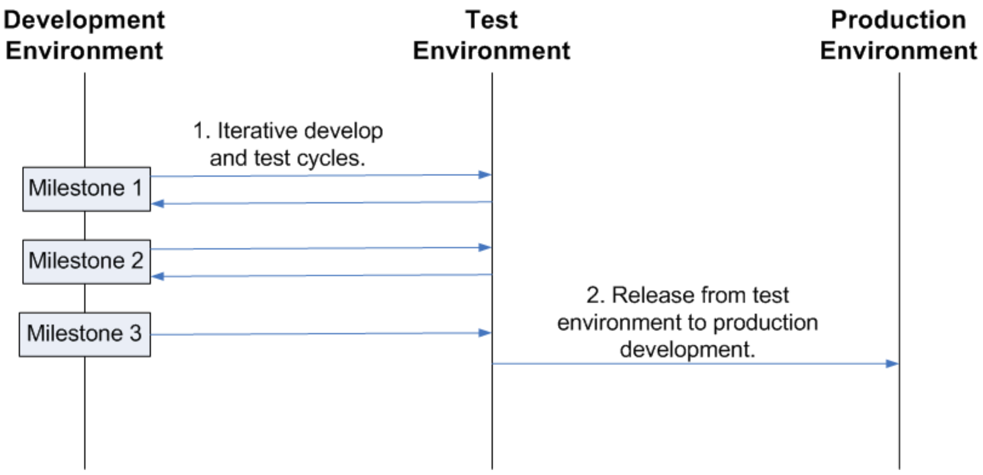
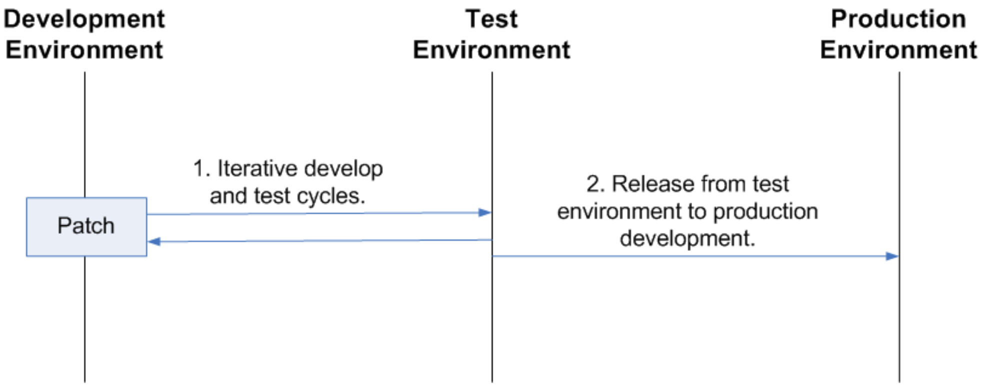

Development Environment
Test Environment
Production Environment
1. Iterative develop and test cycles.
Milestone 1
Milestone 2
Milestone 3
2. Release from test environment to production development.

**Figure 2: Life Cycle Development for Patches**


Development Environment
Test Environment
Production Environment
1. Iterative develop and test cycles.
Patch
2. Release from test environment to production development.

## Stage 1: Development

During development, developers use their tools to perform requirements analysis, design and model a solution, and develop application code.

Developers change code on an ongoing basis to develop their projects over time. This development process may be iterative, with defined milestones. For each milestone, developers work from stable milestone base on a local workstation and make additions or changes. Typically code is checked in and out of change control, and may be merged into a main code line periodically.

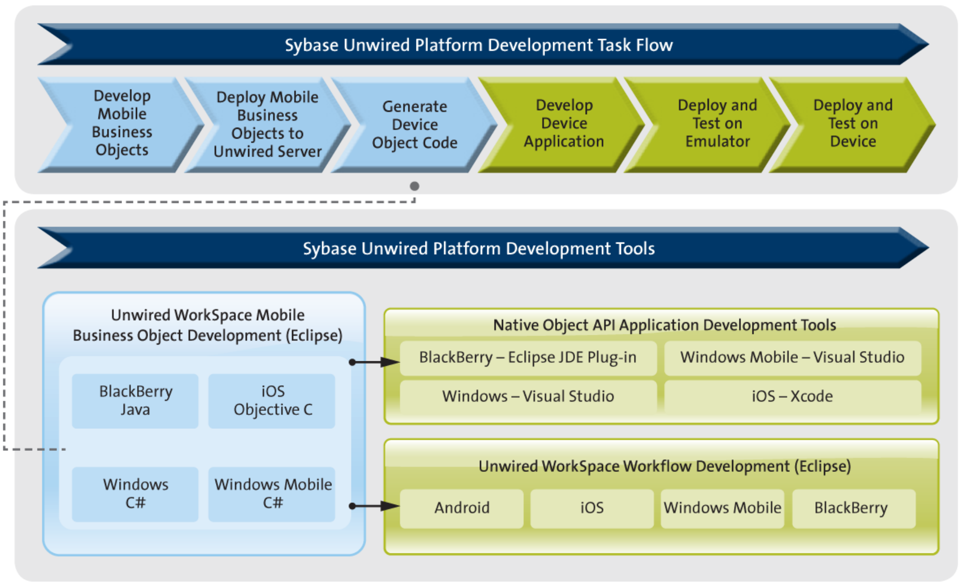
Characteristics of the development stage:

- The model is typically deployed into a dedicated single or shared development Unwired Server for testing.
- During an ongoing project, the base for a development cycle is typically the base from a previous cycle.
- The result of a development cycle is an intermediate release—or deployment package. An intermediate release could be a release or patch for a production system, or the base for more development.

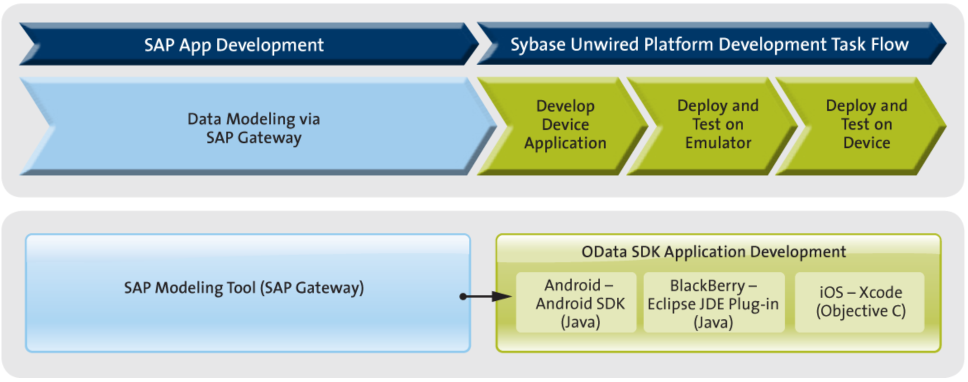
### Development Task Flow

These diagrams represent the task flows, development life cycles, and tools for developing mobile applications using Sybase Mobile SDK.

For Native Object API Applications or HTML5/JS Hybrid Apps:


Sybase Unwired Platform Development Task Flow
Develop Mobile Business Objects
Deploy Mobile Business Objects to Unwired Server
Generate Device Object Code
Develop Device Application
Deploy and Test on Emulator
Deploy and Test on Device
Sybase Unwired Platform Development Tools
Unwired WorkSpace Mobile Business Object Development (Eclipse)
BlackBerry Java
iOS Objective C
Windows C#
Windows Mobile C#
Native Object API Application Development Tools
BlackBerry – Eclipse JDE Plug-in
Windows Mobile – Visual Studio
Windows – Visual Studio
iOS – Xcode
Unwired WorkSpace Workflow Development (Eclipse)
Android
iOS
Windows Mobile
BlackBerry

For OData SDK Applications:


SAP App Development
Sybase Unwired Platform Development Task Flow
Data Modeling via SAP Gateway
Develop Device Application
Deploy and Test on Emulator
Deploy and Test on Device
SAP Modeling Tool (SAP Gateway)
OData SDK Application Development
Android – Android SDK (Java)
BlackBerry – Eclipse JDE Plug-in (Java)
iOS – Xcode (Objective C)

## Stage 2: Testing

During the development process, code may be tested on an iterative basis—typically at specific milestones—by quality assurance engineers.

The test environment should emulate the production environment, to ensure quality throughout the product life cycle. Any defects are reported back to developers, to be fixed. This iterative development and testing cycle continues as long as necessary to produce a high-quality product.

As part of the testing phase, you may need to upgrade the test environment itself as the project develops. During the transition, you may need to consider upgrade scenarios as part of the testing process.

Characteristics of the testing stage:

- Installing into and upgrading the test landscape is equivalent to installing into and upgrading the production landscape.
- The base for a testing cycle is typically an earlier release.
- The result of a testing cycle is a release candidate, or another iteration of development and testing.

### Testing in the Development Environment

Test each native application and workflow package on a simulator or emulator, and make adjustments. Then, test in a test environment that emulates the production environment for complete end-to-end testing.

Download emulators for devices such as Windows Mobile, and use them from the Sybase Unwired WorkSpace development environment. Alternatively, you can use Active Sync to connect the device to your development computer, and deploy and test the application on the device.

Download simulators for devices such as BlackBerry and iOS:

- For BlackBerry, device simulators and the BlackBerry Email and MDS Services Simulator Package are available.
- For iOS, download the iOS simulator environment.

After testing is complete on simulators or emulators, work with administrators to complete end-to-end testing in an environment that emulates the production environment.

## Stage 3: Production

The production system changes on an ongoing basis. After testing, the release can be delivered to the production environment for use.

Typically this is a phased approach, where the system administrator installs a new version, or installs the latest version, and then migrates users individually to the new version. A provisioning tool, such as Afaria, may be used to help with the roll out to users.

Characteristics of the production stage:

- During production, code changes are not allowed.
- Changes to the production system require a new development cycle (in some cases, patches, or hot fixes, can be applied to the production system).
- The base for a production cycle is typically an earlier release.

## Stage 4: Maintenance

Changes to the production system require a new development effort. In some cases, a patch, or hot fix, may be applied to the production system.

Characteristics of the maintenance stage:

- During production, or even during testing, a fix for a defect is issued.
- The base for a patch is an intermediate release, or a release.

## Best Practices

Implement best practices for developing, testing, and deploying device applications using the application life cycle.

### Implement Three Environments

To produce robust, efficient, and high-quality mobile applications, implement Sybase Unwired Platform using three environments—development, test, and production.

The test environment should replicate your production environment as closely as possible. Use the same components, such as an administrative console, security and authentication cluster configuration, and eventual use of Afaria, relay server, and clustering; network communications and protocols; and device simulators and actual devices.

It is important that you test mobile applications on devices and device simulators and emulators to ensure quality, but it is equally important to test the entire data handling experience between the device, Unwired Server, and the enterprise information system (EIS) datasource.

During the test cycle, test engineers should work closely with developers to correct defects. This is typically an iterative process resulting in a production-quality product.

Sybase Unwired Platform does not provide specific tools to follow the data path through the system. Use an administrative console, log files, and error messages to help validate your testing.

Sybase highly recommends that you use automated tests for repeatability. To help isolate problems, you may want to code test scripts.

## Implement Change Control

Implement change control as part of your development process. Integrate your change control system with Sybase Unwired Platform. Ensure that all your development artifacts are routinely checked into your change control system, so you can revert back to a known version if necessary.

Eclipse includes a built-in client for the Concurrent Versions System (CVS). See the Eclipse FAQ to learn more:*http://wiki.eclipse.org/index.php/CVS_FAQ*. You can also integrate other source control systems, such as Subversion.

When implementing change control:

- Set up the repository for the shared development environment.
- Identify the artifacts to be maintained as source code, for example:
  - Model files (the Unwired WorkSpace project file)
  - Native application code, and generated and custom client code
  - Unwired WorkSpace artifacts, which are the MBO projects, custom result checkers, enterprise information system (EIS), and model files
  - Custom ResultSet filters and error checkers
  - EIS connection profiles
  - Projects
- Identify and communicate change control policies and procedures to the development team.

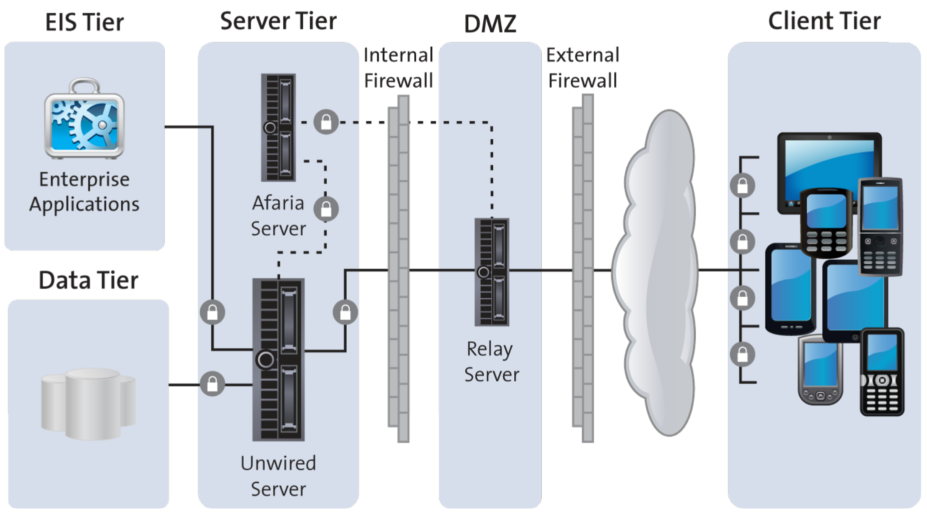
# CHAPTER 5 Unwired Platform Runtime

Unwired Platform Runtime provides the platform infrastructure that allows you to deploy and manage your mobile applications.

This platform infrastructure is installed alongside other corporate assets in one of two deployment options: Unwired Platform Runtime or Online Data Proxy.

Tiered architecture is common to both options.

- **Server tier –** Integrates the server components with back-end enterprise systems, data access and transaction services, device and application deployment, and system management functionality. Unwired Server and Afaria are part of the server tier, and depending on your deployment strategy may or may not reside on the same server host.
- **Data tier –** Stores data retrieved from the back-end data sources and other runtime related metadata. Multiple databases that assist with monitoring and user agent tracking make up the data tier. For mobile applications that use synchronization and data caching, a separate database is designated in the data tier.
- **DMZ –** As a plugin to an Apache Web server of Microsoft Internet Information Server (IIS), Relay Server resides in the DMZ to act as the single point of contact for devices and as a specialized reverse proxy that avoids opening inbound ports in the firewall to Unwired Server.
- **Client tier –** Consists of device applications built using the Sybase Mobile SDK and provisioned to devices.



The following sections describe key aspects of the runtime landscape. For more detailed information, see:

- *System Administration*
- *Security*
- *Quick Start: Online Data Proxy*

- *Runtime Landscape*

  The runtime landscape includes multiple components that are part of or interact with Unwired Platform components to provide the platform mobility solution. These components include: Unwired Server, the Data tier, Relay Server, and Afaria Server.
- *Security*

  An Unwired Platform deployment introduces a multilayer approach to corporate security designed for mobility.
- *Application Deployment*

  During development the developer tests the device application by deploying mobile business object (MBO) packages to Unwired Server, and device application binaries and their dependencies to emulators or actual devices. Deployment to devices can be achieved by physically connecting the device to the developer's machine, and copying binaries using device-supported tools. Applications should also be tested in a test environment that emulates the production environment for complete end-to-end testing.
- *System Management*

  The Unwired Platform management console offers integrated, Web-based access to all Unwired Servers running in the corporate network.
- *System Life Cycle*

  The "system life cycle" takes administrators through the process of planning and installing Unwired Platform Runtime, and integrating it with your enterprise environment; determining software deployment requirements; and setting up the Unwired Platform production environment. Understanding the system lifecycle helps establish best practices for managing and monitoring Sybase Unwired Platform.

## Runtime Landscape

The runtime landscape includes multiple components that are part of or interact with Unwired Platform components to provide the platform mobility solution. These components include: Unwired Server, the Data tier, Relay Server, and Afaria Server.

## Unwired Server

As the runtime server, Unwired Server handles enterprise data source and application access, communication between the backend data source and the mobile device, security, transaction processing, and scheduling.

Unwired Server supports:

- Optimized access to back-end systems, messaging, security services, multitenancy capabilities, and monitoring and development activities.
- Data delivery to and from device applications by applying transactions to the consolidated database (cache), and propagating transactions to back-end systems respectively.

After developing and packaging a mobile application or workflow, the package is deployed to Unwired Server. From Unwired Server, the device application can then be deployed to a mobile device, or provisioned to multiple devices using Afaria® Frontline Management, available as a separately licensed and installed product.

The Unwired Server Customization and Managment APIs support advanced requirements of complex data handling, transaction execution, security customization, exception handling, and systems management automation, all of which are intrinsic to complex corporate and multitenant-based mobile application deployment environments.

## Data Tier

The data tier comprises multiple databases installed for and used by Unwired Server and the management console within the runtime landscape.

Other than planning for installation of the data tier, among other deployment planning activities covered in the *Runtime Installation Guide*, and typical DBA maintenance, Unwired Platform administrators have limited involvement with the data tier. It is important to understand the purpose of each of the databases that make up the data tier.

- **Cache Database (CDB) –** Installed only with the Unwired Platform Runtime option and used only for applications that use synchronization, this relational database management system maintains records of device synchronization activity along with any cached data that may have been retrieved from the back end or pushed from the device.

  The application package hosted in Unwired Server communicates to the CDB through JDBC connection pools that are configured from the administration console. The MBO parameters and the relationships between MBOs define the shape of the cache tables. The internal implementation of these tables and the associated queries are not public and may change from release to release.
- **Cluster Database –** Maintains knowledge of and coordination between Unwired Server clusters.
- **Monitoring and Domain Log Database –** Stores monitoring, tracking, and log messages for application activities.

- **Messaging Database –** For applications that employ messaging services, HTML5/JS Hybrid Apps, OData SDK and applications that use notification mechanisms, this database stores in-transit asynchronous messages until they are processed by the mobile application infrastructure layers. Message data is encrypted on the device and in transit.

## Relay Server

Relay Server enables secure, load-balanced communication between mobile devices and Unwired Server. Across-the-firewall deployment occurs without opening any internal firewall ports for enterprise mobilization and ensures that data is secure while in transit.

Relay server is used in the runtime architecture as a component of the enterprise demilitarized zone (DMZ), to securely integrate mobile devices into your system landscape. You need not install any Unwired Platform components on the carrier network or outside your firewall; they can all be installed and controlled within your corporate networks.

Relay Server:

- Provides a single point of contact for devices and is a specialized reverse proxy that avoids opening inbound ports in the firewall to Unwired Server.
- Accepts and forwards requests from remote clients to Unwired Platform components.
- Is implemented as a pair of Web extensions that run in a Web server. Relay Server supports two Web servers: IIS on Windows and Apache on Linux.
- Integrates with existing security for Web and enterprise infrastructure, eliminating the need for changes to existing corporate firewalls and IT policies.

Unwired Platform synchronization uses Relay Server end-to-end encryption and HTTPS to secure communication. Devices synchronize with Unwired Platform securely on the device client side.

## Afaria Server

As a separately licensed and installed component, Afaria Server provides enterprise level device application deployment and device management. Integrated with Unwired Server, the solution supports end-to-end management and security of devices, and over-the-air (OTA) and push-based deployment of device applications.

Afaria extends Unwired Platform functionality by providing additional device client management features for remote and mobile computing devices, including laptops, desktops, and handheld devices. When Afaria is licensed and implemented with Sybase Unwired Platform, the Afaria management console is launched from the Sybase Control Center console.

Use the Afaria tools to:

- **Deploy client and runtime infrastructure components –** Using OTA and push-based deployment, mobile devices automatically receive notifications, application updates, etc. Afaria administrators can add, update or remove applications, data and content without the users' involvement and can ensure mobile workers have the correct software and data in the field. Users can be confident that the data on their devices is up-to-date and reliable.

- **Track assets –** Identify and track assets being used by your mobile workforce. Afaria Administrators have full control of the range of devices and applications deployed and can view information from a single console.
- **Secure devices and data –** Data and content is backed up and can be deleted if a device is lost or stolen.

## Security

An Unwired Platform deployment introduces a multilayer approach to corporate security designed for mobility.

This approach ensures that:

- Internal and external device users can securely connect to enterprise information systems.
- Every network link that transfers corporate information and every location that stores enterprise data guarantees confidentiality.

Unwired Platform security features cover:

- **Component Security –** Unwired Platform consists of multiple components that are installed on internal networks, primarily on the corporate LAN and the demilitarized zone (DMZ). Each component requires specific administration tasks to secure it.
- **Communication Security –** Secure Unwired Platform component communications to prevent packet sniffing or data tampering. Different combinations of components communicate with different protocols and different ports.

  End-to-end data encryption support is based on Transport Layer Security (TLS) and Secure Sockets Layer (SSL), which secures client/server communication using digital certificates and public-key cryptography.
- **Authentication and Access Security –** Authentication and access control is a core feature supported by all application types to control access to enterprise digital assets. All applications, except Workflow, use a DataVault to store secrets, including those used to authenticate user (for example, certificates).

Security applies to components of the runtime landscape:

- **Server Security –** The Unwired Server provides data services to device clients by interacting with the data tier on behalf of the client. The data tier is installed with the server tier components to the internal corporate LAN. Communications with device clients are routed through the an open port on the internal firewall to the Relay Server.

  Each runtime service uses its own communication port (secured and unsecured). Security for this tier secures both the server components that provide these services and service communications.

- **Data Tier Security –** The data tier consists of multiple databases that need protection. Each database contains sensitive enterprise data. This level of security involves securing the data infrastructure, then securing the databases by managing DBA passwords, granting DBA permissions, as well as encrypting data and logs.
- **DMZ Security –** DMZ security involves controlling internet traffic to private networks. Key to this control is Relay Server, which resides between inner and outer firewalls.
- **Device Security –** Multiple mechanisms can be combined to fully secure devices. In addition to using the built-in security features of both the device or Unwired Platform, Sybase recommends that you also use Afaria so security features can be remotely initiated as required.

  Key Unwired Platform security features for devices include the encryption of data, the implementation of login screens, and the use of Data Vault to store sensitive data.

  Application security is based mainly on the mapping of a mobile business object (MBO) package to a security configuration. A security configuration defines the authentication, authorization, attribution, and auditing security provider for an application package's access control and activities. For example, for an application, an administrator may create a security configuration that points to the LDAP server for authentication and authorization, and does not associate any provider for attribution and auditing.

  Single sign-on (SSO) security providers provide an alternative to user name and password authentication. These security providers add support for token and certificate-based authentication, such as X.509 certificates. SSO enables mobile device application users to enter credentials only once to gain access to all resources, including servers, packages, and data sources related to that application.

  Unwired Platform supports Afaria device management and security functionality, which includes features such as remote device locking, remote data cleanup, data fading (a feature that enables the IT administrator to lock, wipe, or reset a device that has not communicated with the corporate network or Afaria server after a predetermined number of days), and password expiration management. Even without Afaria, the Unwired Server administrator can lock or unlock devices from accessing applications deployed to the server.
- **EIS Security –** Secure interactions with EIS data sources. If you are using DCN, those notifications can be communicated to Unwired Server securely.

## Application Deployment

During development the developer tests the device application by deploying mobile business object (MBO) packages to Unwired Server, and device application binaries and their dependencies to emulators or actual devices. Deployment to devices can be achieved by physically connecting the device to the developer's machine, and copying binaries using device-supported tools. Applications should also be tested in a test environment that emulates the production environment for complete end-to-end testing.

In a production environment, device application deployment can be achieved by using Afaria or similar third-party product. Afaria offers enterprise-class device application deployment and management features.

### Device Testing

Device applications are typically tested by developers in an emulator environment. All target platforms provide emulator support from their native IDE, which enables deployment and execution of the application to the emulator.

### Large-Scale Device Deployment

Some device platforms require signing the binaries before they can be deployed. For production scale deployment, application deployment is done using proven products such as Afaria.

### Device Registration

Application users and their devices are registered in Unwired Server for licensing and monitoring.

Replication-based application user devices are automatically registered after the initial successful authentication with a security configuration.

Messaging-based application user devices must be registered manually. Use predefined templates to register messaging devices. Optionally, you can use Afaria to automate device registration.

## System Management

The Unwired Platform management console offers integrated, Web-based access to all Unwired Servers running in the corporate network.

This enables you to view server status; perform start, stop, restart operations; configure server settings; deploy packages; configure package settings (including cache group schedule, and synchronization settings); subscriptions management (including unsubscribe, recover, suspend, and resume); push notification settings; perform role mapping; cluster management tasks; and device lifecycle and licensing tasks (including registration, unregistration, locking, unlocking, among others).

In addition, the Web console is designed for multi-tenancy in mind to support two views:

- Platform administrator view – for the administrator managing the whole environment.
- Domain administrator view – for the customers of the Unwired Server hosting provider where they can perform package management, subscription management, log viewing, role mapping and server connections management operations in one or more domains assigned to them. The domain forms the logical container and unit of separation of the

artifacts contained in it. The Unwired Server administrator manages the lifecycle of the domains including users who have access to it.

Extending the operational and multi-tenancy capabilities of the platform is the monitoring component of the management console where the platform administrator can view synchronization activities, user access activities, key performance indicators for various activities in the server, cache performance, and other data being gathered. The administrator can set the monitoring configuration to enable/disable monitoring on selected domains and packages, and also perform monitoring data cleanup and export it to share with others for reporting and auditing purposes.

## Administration and Monitoring

Unwired Platform uses a Web-based administrative console to manage and administer Unwired Servers, device application deployment, and mobile devices running across the corporate network.

The administration console, Sybase Control Center, allows administrators to centrally manage, secure, and deploy servers, data applications and devices.

This systems management command and control functionality assists with:

- Core management activities
  - **Server configuration –** Configure server parameters including performance, network ports, messaging, SSL security, and log levels among others.
  - **Cluster management –** Register and monitor clusters to ensure that Unwired Server clusters and servers work smoothly, and scale over time.
  - **Device registration and security –** Facilitate onboarding. Control and monitor devices registered. Registration, activation, and provisioning of devices allows device users to access applications and to be tracked and managed by the system.
  - **Security configuration, administration, and application –** Configure and manage permissions that allow execution of operations on registered servers.
  - **Monitoring –** View historical application usage data, view performance statistics, and manage monitoring data.
  - **Multitenancy management –** Deploy a single production environment to service multiple client organizations, known as Tenants. Administrators can view domains that are created to support the tenancy strategy.
  - **Log viewing –** Review logs directly from the console.
  - **Troubleshooting –** Use information gathered by monitoring Sybase Unwired Platform components to identify and troubleshoot issues.
- Application management activities
  - **Device application deployment –** Manage and monitor device applications deployed to Unwired Server. Afaria Frontline Management is required.
- MBO Package management activities

- **Subscription management –** Configure, manage, and monitor subscriptions related to MBO packages and device user profiles. Subscriptions define what subset of data the device user wants to receive when the device is synchronized with the server.
- Proxy Connection management activities: Manage the endpoint for the proxy connection

In addition to the console, the systems management framework can integrate with Simple Network Management Protocol (SNMP) management tools to monitor specific server events. You can use the Unwired Server management API to automate and extend administration, management, and monitoring features.

## System Life Cycle

The "system life cycle" takes administrators through the process of planning and installing Unwired Platform Runtime, and integrating it with your enterprise environment; determining software deployment requirements; and setting up the Unwired Platform production environment. Understanding the system lifecycle helps establish best practices for managing and monitoring Sybase Unwired Platform.

Detailed information about these activities is provided in:

- *Installation Guide for Runtime*
- *System Administration*

In addition to learning about the system lifecycle, administrators should also understand the development life cycle. The knowledge about the interaction between the two lifecycles helps establish best practices for managing and monitoring Sybase Unwired Platform.

1. *Planning*

   The planning phase includes analyzing system requirements, and creating a high-level design for your system topology and production systems.
2. *Installing*

   The installation phase of the system life cycle includes installing and configuring both Sybase Unwired Platform and the system topology.
3. *Deploying Software on the Network*

   Deploying new or updated software—including new or updated mobile applications and mobile workflow packages, and software upgrades and patches—is an ongoing part of the system lifecycle.
4. *Ongoing Operations*

   An ongoing part of the system lifecycle is to develop operations, administration, and maintenance (OAM) processes that keeps Sybase Unwired Platform running smoothly.
5. *Upgrades and Patches*

   Ongoing maintenance of Sybase Unwired Platform includes upgrading the software, and applying patches.

## Planning

The planning phase includes analyzing system requirements, and creating a high-level design for your system topology and production systems.

Detailed information about these planning activities is provided with *System Deployment* in the *Installation Guide for Runtime*.

**See also**

- *Installing* on page 36

### Requirements Analysis

Analyze your requirements before designing your Sybase Unwired Platform system.

Considerations include identifying data sources, security and access policies, data concurrency, scalability and availability, number of current and projected users, device platforms, device connectivity, provisioning, and service-level agreements.

### System Topology Design

Analyze requirements to plan the system topology required to support security, failover, high availability, system load, load balancing, and database access.

Requirements information helps identify the number and specifications for machines. Sizing the system and system load enables you to determine requirements for the number of Unwired Server nodes or farms, Relay Server nodes or farms, data-tier nodes, and Afaria nodes integration.

Include a test environment in your design that emulates the production environment as closely as possible, including components such as SCC, Afaria if used, Relay Servers, clusters, network communications and protocols, security. This enables you to test data flow through the system, and communication between EIS, Unwired Server, and devices.

### Production Design

As part of system planning, you must also consider how to manage the production system. Some of these considerations may influence your initial ideas about system design and topology.

Considerations include device provisioning, how to manage users and devices, the manual or automated processes that are required, how to implement layers of security, and how to monitor and ensure system health.

## Installing

The installation phase of the system life cycle includes installing and configuring both Sybase Unwired Platform and the system topology.

See *System Administration* and the *Installation Guide*.

**See also**

- *Planning* on page 36

## Deploying Software on the Network

Deploying new or updated software—including new or updated mobile applications and mobile workflow packages, and software upgrades and patches—is an ongoing part of the system lifecycle.

See *System Administration*.

## Ongoing Operations

An ongoing part of the system lifecycle is to develop operations, administration, and maintenance (OAM) processes that keeps Sybase Unwired Platform running smoothly.

- System monitoring – develop processes to monitor Sybase Unwired Platform using the Sybase Control Center, error messages, log files, and server up and down states; third-party tools; and operating system commands.
- Device provisioning – develop processes for handling existing and new users, device changes, and mobile application or mobile workflow package upgrades.
- Data subscriptions – create subscriptions, synchronization groups (messaging and replication), and device notifications (replication) to customize how updated data in the cache is delivered to the device user.
- Disaster recovery plan – develop troubleshooting processes, and solutions for routine problems. Develop a full disaster recovery plan for more serious situations.
- Backup schedule – develop a comprehensive plan for backing up the system and databases, and restoring them in case of disaster.
- Publications – develop publication subscriptions.

See *System Administration* and *Troubleshooting*.

## Upgrades and Patches

Ongoing maintenance of Sybase Unwired Platform includes upgrading the software, and applying patches.

Unwired Platform is continuously under development, with new features being added and existing features being improved. Sybase recommends that you routinely apply upgrades and keep current with the latest software version.

Periodically, Sybase issues software patches to fix problems or improve performance. These patches are posted on the Sybase download Web site.

# CHAPTER 6 Documentation Roadmap

Use the documentation roadmap to review the documentation available for Sybase® Unwired Platform and determine where to start based on your role.

Bookmark *Sybase Unwired Platform Product Documentation* at *http://sybooks.sybase.com/nav/summary.do?prod=1289* to quickly access the complete set of online documentation.

- *Documentation for Developers*

  If you are a developer, identify documentation based on your development focus.
- *Documentation for Administrators*

  If you are an administrator, identify documentation based on your administrative focus.
- *Documentation Roadmap for Unwired Platform*

  Learn more about Sybase® Unwired Platform documentation.

## Documentation for Developers

If you are a developer, identify documentation based on your development focus.

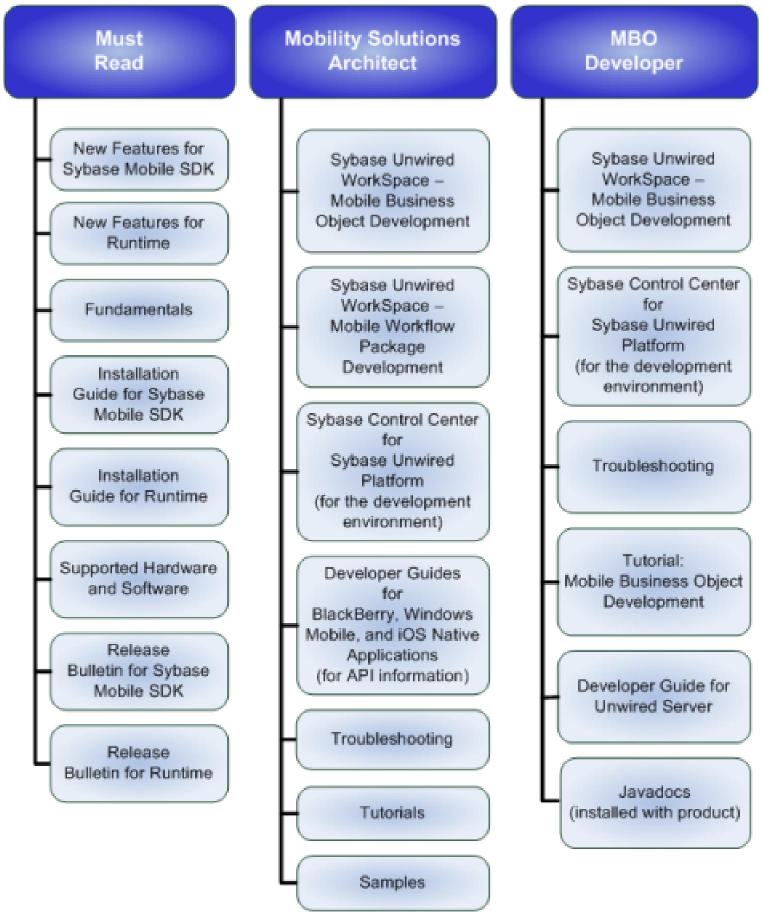
Convenient diagrams are provided to guide you through documentation for the following development categories:

- *Mobile Business Object Development Documentation*
- *Native Object API Application Development Documentation*
- *HTML5/JS Hybrid App Development Documentation*
- *OData SDK Application Development*

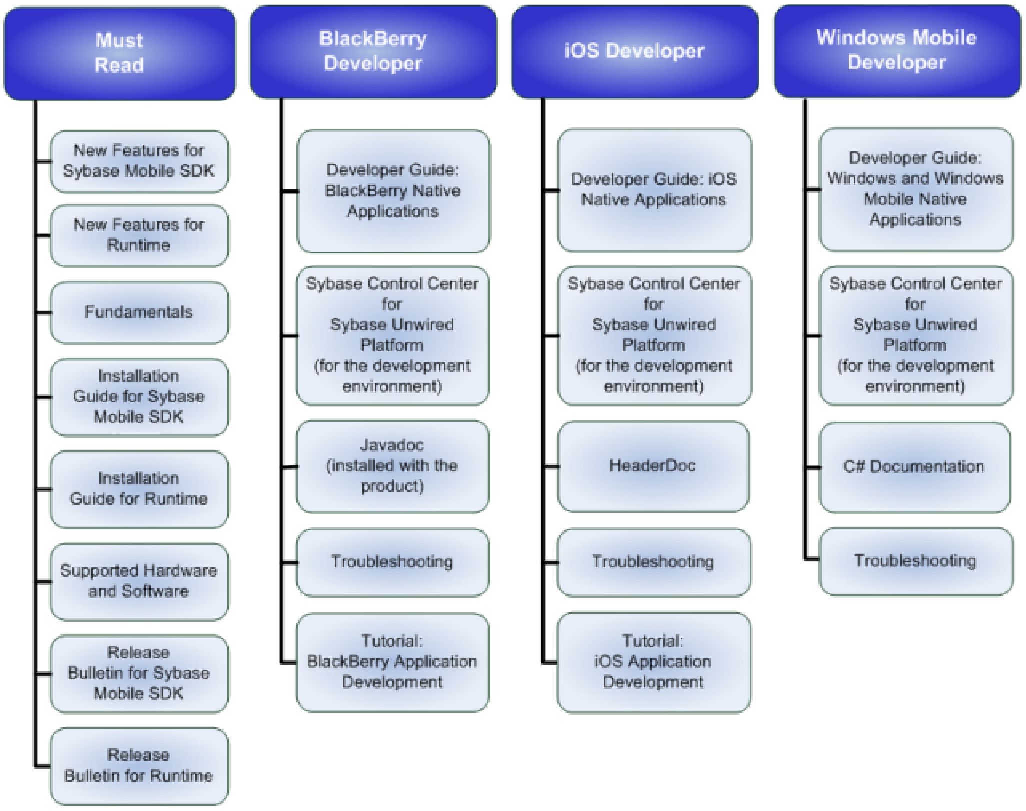
### Mobile Business Object Development Documentation

Two roles emcompass development of MBOs:

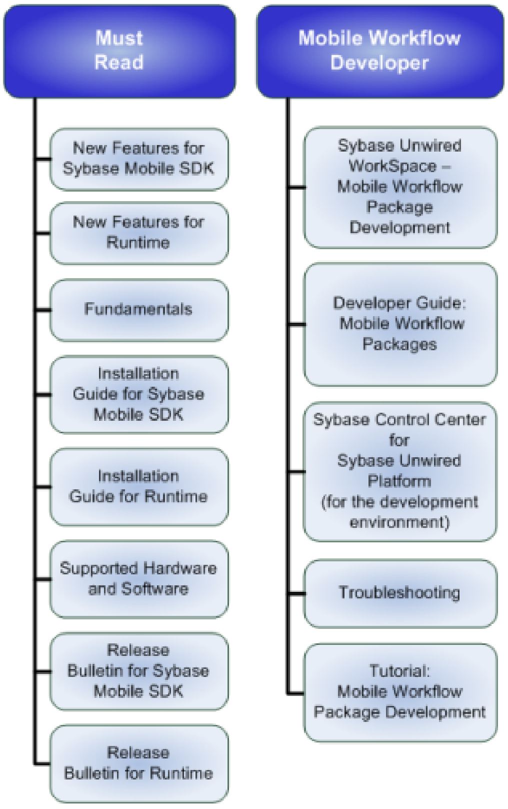
- **Mobility Solutions Architect –** Uses the Unwired WorkSpace tools to model the interaction between the mobile applications and existing EIS, services, and applications, or integrate mobile applications developed with other tools. Understands mobility concepts.
- **MBO Developer –** Develops mobile business objects using Unwired WorkSpace, and the Unwired Server API. Generates native application code for BlackBerry, Windows Mobile, and iOS.


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
Mobility Solutions Architect
Sybase Unwired WorkSpace – Mobile Business Object Development
Sybase Unwired WorkSpace – Mobile Workflow Package Development
Sybase Control Center for Sybase Unwired Platform (for the development environment)
Developer Guides for BlackBerry, Windows Mobile, and iOS Native Applications (for API information)
Troubleshooting
Tutorials
Samples
MBO Developer
Sybase Unwired WorkSpace – Mobile Business Object Development
Sybase Control Center for Sybase Unwired Platform (for the development environment)
Troubleshooting
Tutorial: Mobile Business Object Development
Developer Guide for Unwired Server
Javadocs (installed with product)

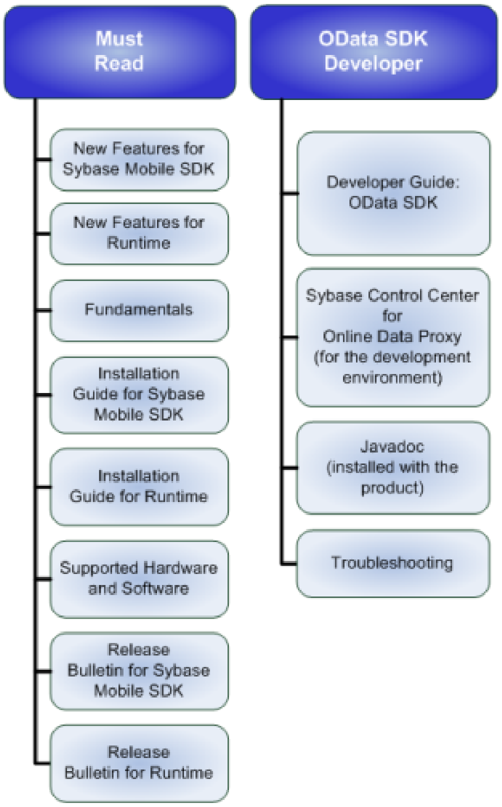
## Native Object API Application Development Documentation


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
BlackBerry Developer
Developer Guide: BlackBerry Native Applications
Sybase Control Center for Sybase Unwired Platform (for the development environment)
Javadoc (installed with the product)
Troubleshooting
Tutorial: BlackBerry Application Development
iOS Developer
Developer Guide: iOS Native Applications
Sybase Control Center for Sybase Unwired Platform (for the development environment)
HeaderDoc
Troubleshooting
Tutorial: iOS Application Development
Windows Mobile Developer
Developer Guide: Windows and Windows Mobile Native Applications
Sybase Control Center for Sybase Unwired Platform (for the development environment)
C# Documentation
Troubleshooting

## HTML5/JS Hybrid App Development Documentation


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
Mobile Workflow Developer
Sybase Unwired WorkSpace – Mobile Workflow Package Development
Developer Guide: Mobile Workflow Packages
Sybase Control Center for Sybase Unwired Platform (for the development environment)
Troubleshooting
Tutorial: Mobile Workflow Package Development

### OData SDK Application Development


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
OData SDK Developer
Developer Guide: OData SDK
Sybase Control Center for Online Data Proxy (for the development environment)
Javadoc (installed with the product)
Troubleshooting

## Documentation for Administrators

If you are an administrator, identify documentation based on your administrative focus.

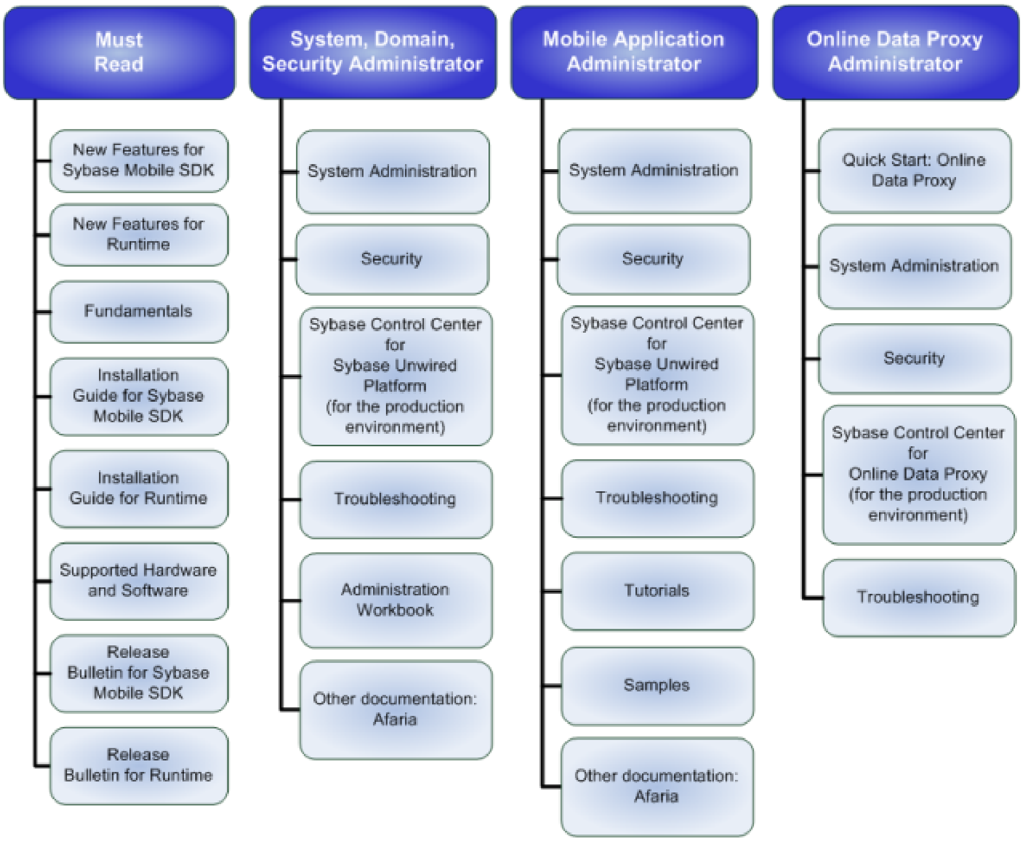
Convenient diagrams are provided to guide you through documentation for the following administration categories:

- *Runtime Administration Documentation*
- *External Systems Administration Documentation*
- *Administration Automation Development Documentation*

### Runtime Administration Documentation

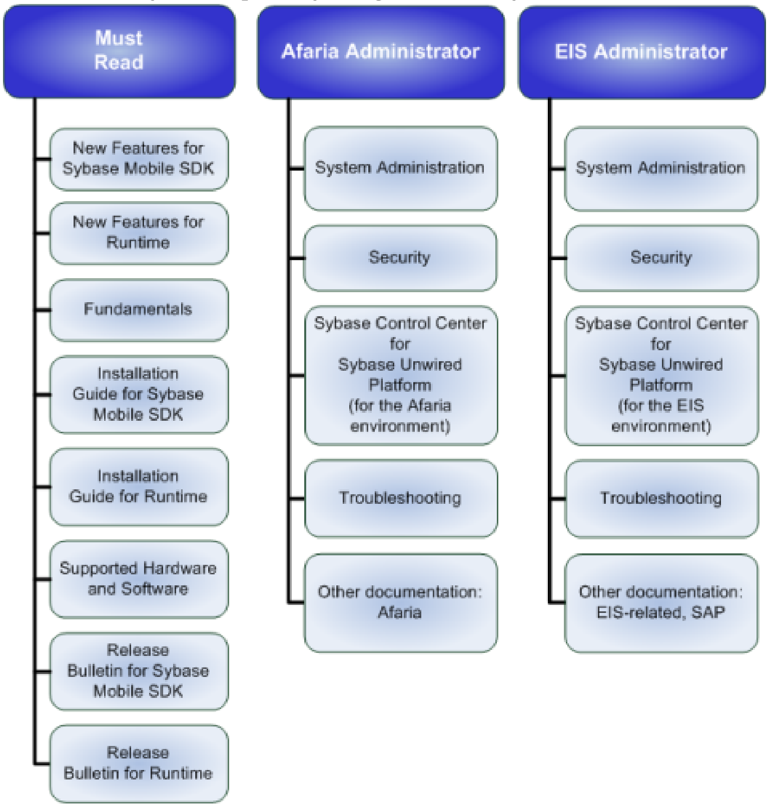
Runtime administration encompasses multiple roles:

- **System Administrator; Domain Administrator; Security Administrator –** These responsibilities may be carried out by one or more individuals, depending on the size and complexity of the enterprise:
  - Plans deployment of production system, and monitors and maintains system health. Understands system configuration.
  - Plans, creates, configures, and administers one or more hosted domain. Each domain may have multiple tenants.
  - Performs ongoing monitoring and maintenance of the cache database (CDB) and other runtime databases.
  - Plans and implements enterprise level security – access, roles, policies.
- **Mobile Application Administrator –** Responsible for operation of the mobile applications running inside the system, and in the field. Understands mobility concepts.
- **Online Data Proxy Administrator –** Administers the Online Data Proxy option.


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
System, Domain, Security Administrator
System Administration
Security
Sybase Control Center for Sybase Unwired Platform (for the production environment)
Troubleshooting
Administration Workbook
Other documentation: Afaria
Mobile Application Administrator
System Administration
Security
Sybase Control Center for Sybase Unwired Platform (for the production environment)
Troubleshooting
Tutorials
Samples
Other documentation: Afaria
Online Data Proxy Administrator
Quick Start: Online Data Proxy
System Administration
Security
Sybase Control Center for Online Data Proxy (for the production environment)
Troubleshooting

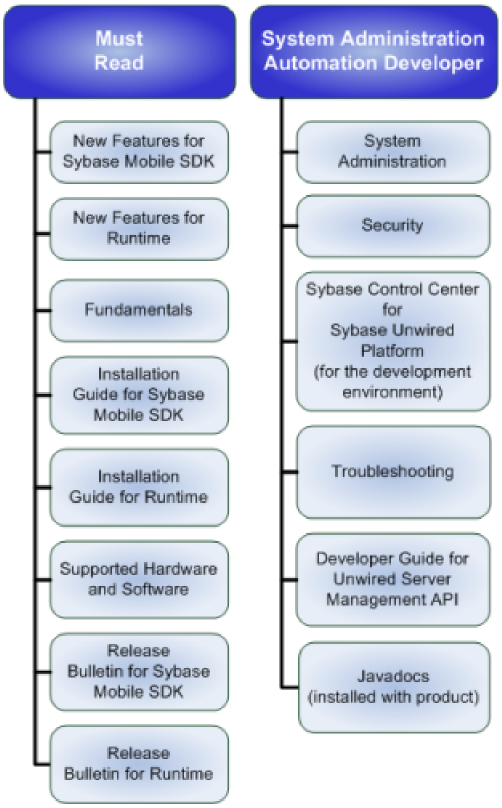
## External Systems Administration Documentation

Administrators for systems external to Sybase Unwired Platform need to understand aspects of how their systems specialty integrates with Sybase Unwired Platform.


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
Afaria Administrator
System Administration
Security
Sybase Control Center for Sybase Unwired Platform (for the Afaria environment)
Troubleshooting
Other documentation: Afaria
EIS Administrator
System Administration
Security
Sybase Control Center for Sybase Unwired Platform (for the EIS environment)
Troubleshooting
Other documentation: EIS-related, SAP

## Administration Automation Development Documentation

Within the context of administration, the administration automation developer fills a development role.


Must Read
New Features for Sybase Mobile SDK
New Features for Runtime
Fundamentals
Installation Guide for Sybase Mobile SDK
Installation Guide for Runtime
Supported Hardware and Software
Release Bulletin for Sybase Mobile SDK
Release Bulletin for Runtime
System Administration Automation Developer
System Administration
Security
Sybase Control Center for Sybase Unwired Platform (for the development environment)
Troubleshooting
Developer Guide for Unwired Server Management API
Javadocs (installed with product)

## Documentation Roadmap for Unwired Platform

Learn more about Sybase® Unwired Platform documentation.

| Document | Description |
|---|---|
| *Installation Guide for Sybase Mobile SDK*<br><br>Installation Guide for Runtime | Describes how to install or upgrade Sybase Unwired Platform Runtime or Mobile SDK. Check the corresponding *Sybase Unwired Platform Release Bulletin* for additional information and corrections.<br><br>Audience: IT installation team, training team, system administrators involved in planning, and any user installing the system.<br><br>Use: during the planning and installation phase. |

| Document | Description |
|---|---|
| *Release Bulletin for Sybase Mobile SDK*<br><br>*Release Bulletin for Runtime* | Provides information about known issues, and updates. The document is updated periodically.<br><br>Audience: IT installation team, training team, system administrators involved in planning, and any user who needs up-to-date information.<br><br>Use: during the planning and installation phase, and throughout the product life cycle. |
| *New Features for Sybase Mobile SDK*<br><br>*New Features for Runtime* | Describes new or updated features.<br><br>Audience: all users.<br><br>Use: any time to learn what is available. |
| *Fundamentals* | Describes basic mobility concepts and how Sybase Unwired Platform enables you to design and implement mobility solutions.<br><br>Audience: all users.<br><br>Use: during the planning and installation phase, or any time for reference. |
| *System Administration* | Describes how to plan, configure, manage, and monitor Sybase Unwired Platform. Use with the *Sybase Control Center for Sybase Unwired Platform* online documentation.<br><br>Audience: installation team, test team, system administrators responsible for managing and monitoring Sybase Unwired Platform, and for provisioning device clients.<br><br>Use: during the installation phase, implementation phase, and for ongoing operation, maintenance, and administration of Sybase Unwired Platform. |
| *Security* | Describes how to plan, configure, and manage end-to-end security.<br><br>Audience: installation team, test team, system administrators responsible for managing security.<br><br>Use: during the installation phase, implementation phase, and for ongoing operation to secure the system, data, access, and authorization. |

| Document | Description |
|---|---|
| *Quick Start: Online Data Proxy* | Guides the administrator through the set of tasks to install, configure, manage, and monitor Online Data Proxy.<br><br>Audience: system administrators responsible for managing proxy connections and Online Data Proxy mobile applications in the Sybase Unwired Platform runtime environment.<br><br>Use: while getting started with Online OData Proxy |
| *Sybase Control Center for Sybase Unwired Platform*<br><br>*Sybase Control Center for Online OData Proxy* | Describes how to use the Sybase Control Center administration console to configure, manage and monitor Sybase Unwired Platform. The online documentation is available when you launch the console (**Start > Programs > Sybase > Sybase Control Center**, and select the question mark symbol in the top right quadrant of the screen).<br><br>Audience: system administrators responsible for managing and monitoring Sybase Unwired Platform, and system administrators responsible for provisioning device clients.<br><br>Use: for ongoing operation, administration, and maintenance of the system. |
| *Troubleshooting* | Provides information for troubleshooting, solving, or reporting problems.<br><br>Audience: IT staff responsible for keeping Sybase Unwired Platform running, developers, and system administrators.<br><br>Use: during installation and implementation, development and deployment, and ongoing maintenance. |

| Document | Description |
|---|---|
| Tutorials | Tutorials for trying out basic development functionality.<br><br>Audience: new developers, or any interested user.<br><br>Use: after installation.<br><br>• Learn mobile business object (MBO) basics, and create a mobile device application:<br>  • *Tutorial: Mobile Business Object Development*<br>• Create native mobile device applications:<br>  • *Tutorial: BlackBerry Application Development*<br>  • *Tutorial: iOS Application Development*<br>  • *Tutorial: Windows Mobile Application Development*<br>• Create a mobile workflow package:<br>  • *Tutorial: Mobile Workflow Package Development* |
| *Sybase Unwired WorkSpace – Mobile Business Object Development* | Online help for developing MBOs.<br><br>Audience: new and experienced developers.<br><br>Use: after system installation. |
| *Sybase Unwired WorkSpace – Mobile Workflow Package Development* | Online help for developing mobile workflow applications.<br><br>Audience: new and experienced developers.<br><br>Use: after system installation. |
| Developer guides for native and mobile workflow device application customization | Information for client-side custom coding using the Client Object API.<br><br>Audience: experienced developers.<br><br>Use: to custom code client-side applications.<br><br>• *Developer Guide: BlackBerry Native Applications*<br>• *Developer Guide: iOS Native Applications*<br>• *Developer Guide: Mobile Workflow Packages*<br>• *Developer Guide: Windows and Windows Mobile Native Applications* |

| Document | Description |
|---|---|
| *Developer Guide: OData SDK* | Information for developing OData applications using the API.<br><br>Audience: experienced developers.<br><br>Use: to customize mobile applications using a proxy connection to an SAP OData source. |
| Developer guide for Unwired Server side customization – *Developer Guide: Unwired Server* | Information for custom coding using the Server API.<br><br>Audience: experienced developers.<br><br>Use: to customize and automate server-side implementations for device applications, and administration, such as OData handling.<br><br>Dependencies: Use with *Fundamentals* and *Sybase Unwired WorkSpace – Mobile Business Object Development*. |
| Developer guide for system administration customization – *Developer Guide for Unwired Server Management API* | Information for custom coding using administration APIs.<br><br>Audience: experienced developers.<br><br>Use: to customize and automate administration at a coding level.<br><br>Dependencies: Use with *Fundamentals* and *System Administration*. |

# Index