

Synchronizing and Managing Mobile Devices

BENEFITS

- Small code base enables developers to implement Device Management in memory-limited devices such as cellular phones, smart phones and PDA's.
- Shortens time-to-market and increases return on investment by reducing development cost and testing cycle
- Enables developers to focus on applications, and system integration

As the mobile phone market matures, operators and phone vendors are demanding more sophisticated software features from chipset manufacturers and ODMs. These new features enhance the mobile user's experience but cause two key problems. First, how can the increasing bulk of personal data on the phone be accessed and modified remotely from a user's PC, an in-car telephony unit, or a corporate back-office? Second, how can the phone's various applications and configurations be updated and managed remotely?

The Open Mobile Alliance (OMA) specifies technologies that solve these problems. Data Synchronization (DS) allows PIM information to be exchanged between a client and server. Device Management (DM) allows a server to upload new settings and software to a client. Additionally, by using OMA's Client Provisioning (CP) specification, it's possible to configure a mobile phone with the information and initial configuration it needs to activate DM and DS protocols at the point of sale.

This datasheet provides more information about these technologies and explains some of the issues that device manufacturers will face as they deploy these critical features.

These technologies are broadly known as SyncML1. Figure 1 shows how SyncML fits in with some other wireless technologies.

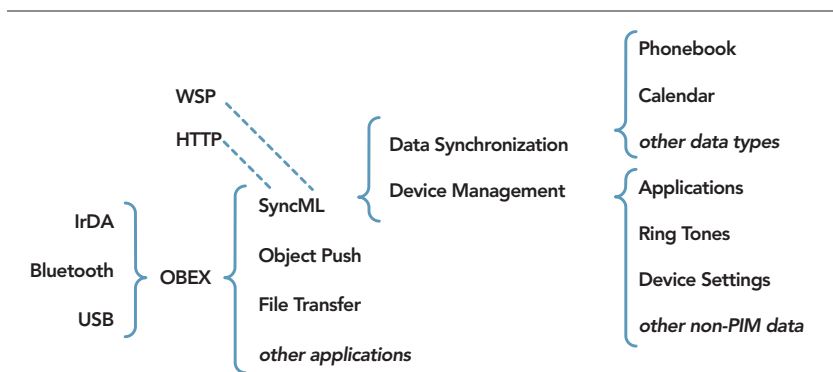


Figure 1: SyncML protocol/feature map

¹ These specifications were previously maintained by the SyncML organization, which was absorbed into the Open Mobile Alliance in late 2002. In the future these specifications will be known as OMA Data Synchronization and OMA Device Management.

TECHNOLOGY

Specification Organizations

Mobile device manufacturers first came together in 1998 to produce an open standard for synchronization of personal device information. Early efforts at standardization within the IFA produced the IrMC (Ir Mobile Communications) protocol, which specified four “levels” of synchronization stretching from basic object exchange to incremental synchronization using unique record identifiers. Based on lessons learned while implementing this protocol, many of the same manufacturers later formed the SyncML Initiative. This group was tasked with developing an open, extensible standard for data synchronization of any data type and any communications transport. The result is the XML-based SyncML Data Synchronization (SyncML-DS) protocol. Since its release, SyncML-DS has enjoyed widespread support by mobile device manufacturers and PIM synchronization technology providers alike.

Since the development of SyncML-DS another problem has arisen. As modern mobile devices incorporate more features, configuration and setup of these devices becomes more unwieldy. Most users are now unable to properly configure standard features, or to diagnose configuration problems. This means that carriers must be capable of remotely managing the devices in their network. Further complicating matters, carriers must support a variety of mobile phones from different manufacturers. This has generated a thrust by carriers for mobile device manufacturers to support a common mechanism for managing their devices. Standardizing on a device management platform appeals to carriers because it simplifies management of mobile device configurations both during and after their deployment into the network.

The SyncML Initiative, working from its experience and track record within data synchronization, took on the challenge of designing an open device management standard. The result is the XML based SyncML Device Management (SyncML-DM) protocol. This specification defines a protocol for use between a management server and a mobile device, as well as an access policy and tree based management object model.

In November of 2002, the work of the SyncML Initiative was consolidated into the Open Mobile Alliance (OMA). Ongoing work on the SyncML DM and DS protocols now continues under the rules and guidelines of the OMA. In June of 2003, the version 1.1.1 SyncML protocols were re-issued, based on OMA procedures as OMA Data Sync and OMA Device Management version 1.1.2.

While SyncML-DM provides a solution for ongoing device management needs, there is still another requirement. A brand new phone requires a minimal amount of bootstrap configuration, known as “provisioning”, before it can begin to communicate with a SyncML-DM server. This initial setup process is described by the OMA Client Provisioning specifications, which were derived unchanged from the WAP Client Provisioning specifications. At the completion of the Client Provisioning stage, the device is capable of being managed by SyncML DM server because it is configured with a trusted relationship with one or more management servers. We view these two procedures (initial provisioning and ongoing device management) as part of the same process of phone configuration. Therefore, we consider OMA Client Provisioning to be an integral component of the SyncML DM SDK (and may not differentiate between the two in some discussions).

SPECIFICATIONS

The SyncML suite of protocols is based on a number of sub-components. These sub-components can be collected into four groups as follows:

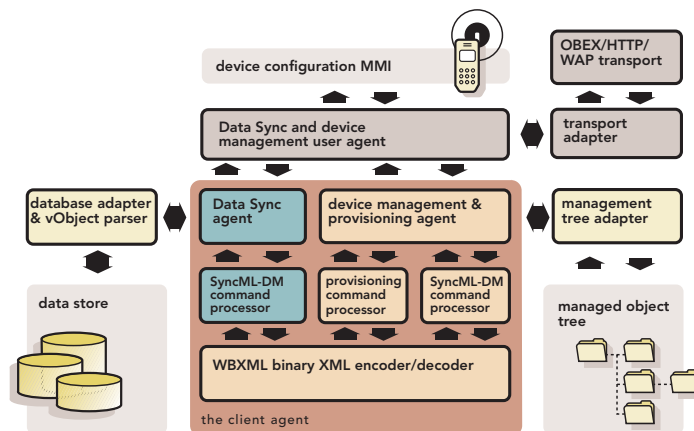
- XML documents for exchange of SyncML commands and information:
 - The SyncML Representation Protocol document
 - The SyncML Meta Information document
 - The SyncML Device Information document
 - The OMA Client Provisioning document
- SyncML Standardized Management objects:
 - The SyncML Device Information managed object
 - The SyncML Device Detail managed object
 - The SyncML DM Management object

- Specifications for exchanging SyncML messages over various transports:
 - The SyncML OBEX Transport Binding
 - The SyncML HTTP Transport Binding
 - The SyncML WSP Transport Binding
- Non-SyncML Standard object format definitions
 - The PIM data standards such as vCard, vCalendar, vNote, vMessage, vTodo
 - The Custom management objects

To support the exchange of SyncML messages the client and server must each implement a common transport protocol. In addition, they must agree upon which device will be the initiator of the session so that the transports can be set to the correct state (accepting an inbound connection or establishing outbound connection).

A SyncML session is based on the exchange of SyncML messages between the client and server. The flow of commands between the client and server is divided into multiple stages; these stages are referred to as packages. Each package requires one or more messages to be sent. The number of messages is determined by the amount of information that must be exchanged. SyncML messages include one or more commands and status on received commands. The SyncML session is complete when both devices have sent all their commands and received status for them.

SyncML messages (OMA Client provisioning included) are defined in a markup language based on the XML standard. The messages can be transmitted in either XML or WBXML format. WBXML is a binary version of XML and allows for much smaller messages (by compression) and simpler parsing. Because of its message size and parsing benefits, most mobile devices only support WBXML encoded messages.



IMPLEMENTATION

The Ideal OMA DS/DM/CP Client Architecture

Definitions:

- The User Agent initiates the management or data synchronization session by initializing the Client Agent and the transports. It also manages the exchange of packet data between the Client Agent and the Transport Adapter.
- The Data Synchronization, Device Management & Provisioning Agents manage the message exchange process, including the flow of commands and responses between the remote server and the Data Store or Managed Object Tree.
- The Object Manager Interface forms the conduit between the Management Agent and the Managed Object Framework. The Managed Object Framework is responsible for maintaining the properties and values of all managed objects. This includes maintaining and enforcing the server access control lists.

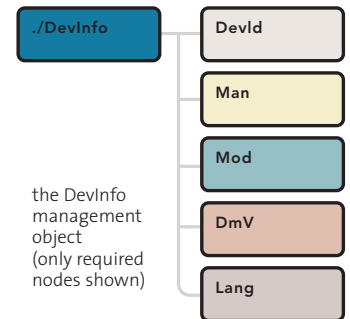
- The Database Adapter Interfaces the Data Synchronization Agent to the data store. Translation of data store records between native and MIME formats is also performed here.
- The Command Processors handle SyncML specific command building and parsing.
- The WBXML Processor handles encoding and decoding generic commands in WBXML format.

OBJECT TYPES

Device Management Objects

Device Management objects differ from synchronization objects in many ways. A device management object is comprised of a set of nodes, typically structured in a tree. This permits the values and properties of each node to be set and retrieved individually. A management object can be as small as a single integer or very large and complex. The SyncML DM protocol is agnostic about the contents, or values, of the management objects and treats the node values as opaque data.

The SyncML-DM specification defines three required management objects. They are the Device Info (shown here), Device Detail and Device management objects. The content of each of these objects are distributed across a collection of nodes, which together form the complete object. The Device Info object contains information that uniquely identifies the device. The Device Detail object contains general information about the device, and the Device Management object contains the settings for the DM client within the device. SyncML-DM is designed to permit the management of additional objects defined by other standards organizations.



The OMA Client Provisioning object is an XML based document that specifies the initial provisioning parameters for the device. This document is designed to be very flexible and can be used to provision an array of services and applications. It includes configuration parameters for proxy servers, network access points, application parameters, access rules and others. This extensible document is a popular choice for the provisioning of new and proprietary technologies. It can even be used to provision an initial SyncML-DS/DM configuration.

Data Synchronization Objects

The SyncML-DS protocol is independent of the actual data objects it synchronizes. Therefore the protocol can be used to synchronize data in any format. As long as the client and server share a common format for a given object, its data store can be synchronized. SyncML provides a mechanism for the client to convey the object formats it supports to the server. This is the Device Info document, and includes general information about device as well. Within the Device Info document, the MIME type of each object is specified, along with the content fields supported by the device.

Here is a table of the commonly used data formats by data store type:

Data Store Type	MIME Type	Description
Memo's / Notes	text/plain	Plain Text
Phonebook / Contacts	text/x-vcard	VERSIT vCard 2.1
Tasks / To-do list	text/x-vcalendar	VERSIT vCalendar 1.0, vTodo
Appointments / Events	text/x-vcalendar	VERSIT vCalendar 1.0, vEvent

CONNECTION MODELS

Client Initiated or Server Initiated?

In SyncML, a device behaves as either the client or the server. Typically, the embedded mobile device is the client and as a result the more difficult processing is pushed onto the (typically more powerful) server platform. Within the device, the transport may be implemented as a client or server or both. This enables either device to initiate the synchronization or management session. Historically the client initiated the synchronization session. Over time, this has become less prevalent and more often the server initiates the synchronization. The server does this by sending a special message to the client requesting that the client then initiate a session back to the server.

Server Alerted Notification

The SyncML-DS specification includes a definition of “Server-Initiated Synchronization” (SIS) that defines how a server can request that a client start a synchronization session. Since the client is responsible for initiating synchronization in SyncML, this message from the server to the client is unique, and is used only as a trigger. During implementation of SyncML-DS, many problems were identified in the specification of SIS. Some issues were very complex and required drastic changes to the specified mechanism. As a result, when it came time for the SyncML-DM group to address server-initiated management they chose a different path, which they call “Server Alerted Notification”.

The Server Alerted Notification (SAN) is used to instruct a client to establish a management session back to the initiating server. The format of the SAN message is unique in that it is not based on XML. This was chosen to minimize the message size so that it could be easily sent in push messages, such as SMS. This means that the SyncML-DM client device must include two parsers, one for the WBXML messages and one for the SAN message. The SAN message format addressed many of the problems with SIS and has since been adopted by SyncML-DS for use in v1.2 of the specification.

In the current environment, devices that are developed to support SIS must be considered proprietary solutions. While the general format and use of the SIS message is similar on most devices. Some amount of client or server customization is typically required to tailor support of SIS for each device.

TESTING AND INTEROPERABILITY

Limitations

While the SyncML DS/DM protocols provide the open standards necessary for uniform data synchronization and device management, there are additional issues that must be addressed. One such issue is that of data object format compatibility. The formats for objects exchanged by the DS/DM protocols are not typically specified by the protocols, but are referenced from other standards (eg. vCard/vCalendar). Widespread use of these standards has revealed numerous interoperability issues, caused by differing interpretations of the specifications. This is an area where the ODM must pay particular attention to insure that the client and server implementations interoperate. The issue of data store object interoperability often leads ODMs to select specific pairings of client and server and to work with each vendor to insure interoperability.

Another challenge to interoperable synchronization between multiple clients and servers is different devices may use different data stores for its records than another. For example, a client may decide to store contacts, tasks and events in three different databases. While a server may decide to store contacts in one database and events and tasks in another. This is a common example because events and tasks are both considered calendaring items. This may require additional logic in the client and/or the server to reconcile these inconsistencies when possible.

TEST TOOLS AND STRATEGIES

When considering a testing approach for data synchronization there are three core aspects to address. They are: protocol compliance, data communications and record interoperability.

The first, protocol compliance is best tested using the SyncML Conformance Test Suite. This tool, provided by the OMA, exercises most of the required features of the SyncML protocol and reports results by comparing the behavior of the device under test to the protocol specification. Because the SyncML core is isolated from the data transfer and object encoding modules, it can be verified independently of these other components. Most problems in this component come from differing views on how a specified feature should be used, or was intended for use. For this reason, iAnywhere has, and continues to participate in interoperability test events, to work through such issues and to insure the highest level of interoperability possible. In addition iAnywhere representatives participate in SyncML working group discussions, lending their expertise to resolve design and interoperability issues while specifications are still under development.

The data communications layer, typically OBEX, HTTP or WSP running over a variety of hardware is by nature more difficult to test. However, wide availability of established implementations of most of these protocols results in relatively few protocol layer problems. Once the SyncML specific communication settings have been worked through, interoperability testing with a variety of platforms should be conducted.

The final aspect is data object, or record compatibility testing between the client and server devices. SyncML specifies a set of common object formats that are to be used when exchanging data store records. For example, a vCard is used to exchange each contact in the phonebook. While the use of a common format enables both devices to parse the data, the interpretation of the data is still open to compatibility problems. For this reason, it is best to externally agree on the specific use of each property within an object.

Another facet of the data object exchange problem is that different capabilities and functionality on the client and server lead to differing limitations. For example, while a server running on a PC may support contact names of unlimited length, the mobile client typically will have some limit. A common feature level compatibility issue is reoccurring appointments. There are many ways to support and exchange reoccurring appointments and each device will typically have specific requirements with respect to what it can send and receive.

In the end, to effectively test data object exchange, and this is probably the most important aspect to test, one must first identify a set of synchronization targets. Once this set is identified, the requirements for each should be mapped out and compatibility plans developed. Once implemented these plans should be tested by thoroughly testing all the features implemented by each data store.

4.3 Qualification programs

When the SyncML Initiative developed the SyncML protocols, a set of test procedures were also developed to validate device interoperability. Devices that successfully completed the test procedures, which included attending a “SyncFest” event, were certified as “SyncML Compliant”. The iAnywhere SyncML-DS Client was first certified SyncML Compliant in October of 2001. With the consolidation of the SyncML Initiative into the OMA, the certification of SyncML compliance has been discontinued. Fortunately, the OMA continues to coordinate interoperability events, now referred to as “Test Fest” events and maintains the SyncML Conformance Test Suite (SCTS). However the concept of SyncML Certification and use of the SyncML Logo has been eliminated. Devices incorporating the SyncML protocol(s) may now use the OMA logo, as long as the vendor is a member of the OMA.

IANYWHERE ROADMAP

iAnywhere’s Embedded Mobile team provides portable, embeddable source code SDKs that address the connectivity needs of mobile devices. The original release of the SyncML Client SDK focused on the emerging, standardized data synchronization market. Upcoming products focus on the needs of telematics (in-car telephony units) and device management for mobile equipment OEM’s. The following road map illustrates how the SyncML SDK can grow to meet these new opportunities.

PRODUCTS

Name	SyncML SDK	SyncML Server SDK	SyncML DM Client SDK
Role	Data Sync Client	Data Sync Server	Device Management Client
Target	Mobile phone OEMs	Telematics OEMs	Mobile Phone OEMs
past releases			
Mar 2002	V1.0 – Based on spec v1.0.1, WBXML parser, HTTP client transport		
Nov 2002	V1.1 – Based on spec v1.1 added generic vObject parser, OBEX transport		
Apr 2003	V1.2 – Based on spec v1.1.1 added XCPC-compatible vObject parsing		
Sept 2003	V1.2.1 – Added server initiated synchronization		
May 2004	V1.3 – Added portable user agent, XCPC-specific sample code		
June 2004		V1.0 – Spec v1.1.2, one-way sync from client, server-initiated, SonyEricsson K700 compatibility, phonebook, OBEX/Bluetooth transport	
Sept 2004	V2.0 – Integration with Device Management SDK	V2.0 – Integration with Device Management SDK	V1.0 – Based on v1.1.2 specs, DM User Agent extensions, OBEX & HTTP transports
Dec 2004			V1.1 – Add OMA Client Provisioning (first-time device management)
2005	V2.0 – Spec v1.2 support	V2.0 – Spec v1.2 support V3.0 – Add two-way sync additional mobile devices, additional databases (calendar, message)	V1.2 – DM spec v1.2, add configuration object manager, other specs

SyncML Client SDK (Data Sync Only)

The SyncML Client SDK provides a client implementation of the OMA-DS protocol for synchronizing mobile databases with a server device. This product targets mobile phone OEMs.

SyncML Server SDK (Data Sync Only)

The SyncML Server SDK provides a server implementation of the OMA-DS protocol for synchronizing mobile client databases with an embedded server device. This product targets Telematics providers.

Device Management SDK (Client Only)

The Device Management SDK provides a client implementation of the OMA-DM and OMA Client Provisioning protocols for bootstrapping and continuous remote management of mobile devices. This product is targeted at mobile phone OEMs.

SyncML SDK Configurations

The suite of protocols offered by our SyncML SDK offer the ODM a comprehensive solution to the problems of PIM synchronization and device setup & configuration. The integration of the SyncML DS and DM protocols in one package enables code efficiencies that create a product that is smaller than the sum of its components. For users of the combined SyncML DS/DM Client SDK, this leads to reduced porting time, RAM and ROM usage, and provides a single point of contact for support and maintenance.

Feature	SyncML DS Client	SyncML DS Server	SyncML DM Client
SyncML DTD Support	✓	✓	✓
Device info DTD Support	✓	✓-optional	
Meta info DTD Support	✓	✓	✓
Provisioning DTD Support			✓
Basic & MD5 Authentication	✓	✓	✓
HMAC Data Integrity			✓
OBEX/HTTP Transports	✓	✓	✓
Database Adapter API	✓	✓	
Management Tree Adapter API			✓

When considering the three products offered there are two bundling possibilities that are particularly interesting. The combination of the SyncML-DS Client and SyncML-DS Server in one product enables peer-to-peer synchronization. This is a unique product that can only be offered by producers of an embedded SyncML-DS Server. Some of the highlights of peer-to-peer synchronization are:

- Enabling synchronization of PIM data between two or more mobile phones within a family.
- Enabling synchronization of PIM data between disparate mobile devices, such as a PDA and a phone.
- Eliminates the traditional need of a PC or network server for PIM data backup.

While the peer-to-peer synchronization solution is empowered by the combination of client and server technology in one device, it is not automatic. The research and design that must go into such a product would be similar to that of any embedded server deployment.

Another attractive bundling option is the combination of the SyncML-DS and DM Clients. With the gaining popularity of SyncML-DS, the bundling of DM to enable next generation management capabilities with minimum additional effort should resonate well with ODMs and carriers alike. Furthermore, bundling SyncML-DM with DS solves some of the issues that exist with initial SyncML-DS provisioning.

For more information and a list of distributors visit our web site: www.ianywhere.com

IANYWHERE SOLUTIONS, INC.
WORLDWIDE HEADQUARTERS
ONE SYBASE DRIVE
DUBLIN, CA 94568-7902
U.S.A.

FOR GENERAL INFORMATION:
CONTACT_US@IANYWHERE.COM
NORTH AMERICA
T 1-800-801-2069
1-519-883-6898

FOR SPECIFIC REGIONAL PRODUCT
INFORMATION:
EUROPE, MIDDLE EAST, AFRICA
+44 1628 597 100
ASIA PACIFIC
+852 2506 8700
JAPAN
+81 3 5210 6380
BENELUX
+31 (0)73 - 623 5359
FRANCE
+33 (0) 1 30 09 23 23
GERMANY
+49 (0) 7032 / 798 - 0
UNITED KINGDOM
+44 (0)17 333 9000

