3GPP TS 27.103 V5.0.0 (2002-06)

Technical Specification

3rd Generation Partnership Project;
Technical Specification Group Terminals;
Wide area network synchronization standard
(Release 5)

Keywords

UMTS, terminal, WAN, synchronization

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

This specification provides a definition of a Wide Area Synchronization protocol. The synchronization protocol is based upon current synchronization industry standards.

The present document covers Wide Area Network Synchronization between current and future mobile communication end-user devices, desktop applications and server-based information servers. This is a living document and, as such, it will evaluate new technologies (e.g. XML) for inclusion as they become readily available.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] Bluetooth: Bluetooth SIG, Bluetooth Specifications, version 1.0, July 1999. (http://www.bluetooth.com/)

[2] IrMC, Infrared Data Association, "Specifications for Ir Mobile Communications (IrMC)", version 1.1, 01 March 1999, plus all applicable errata. (http://www.irda.org/)

[3] IrOBEX, Infrared Data Association, "Ir Object Exchange Protocol IrOBEX", version 1.2, April 1999, plus all applicable errata. (http://www.irda.org/)

[4] vCalendar, the Internet Mail Consortium, "vCalendar - The Electronic Calendaring and Scheduling Exchange Format - Version 1.0", 18 September 1996. (http://www.imc.org/pdi/vcal-10.doc)

[5] vCard, the Internet Mail Consortium, "vCard - The Electronic Business Card - Version 2.1", 18 September 1996.(http://www.imc.org/pdi/vcard-21.doc)

[6] WAP, WAP Forum, "WAP Technical Specifications Suite", version 1.1, June 1999. (http://www.wapforum.com/)

[7] XML, W3C, “Extensible Markup Language (XML) 1.0”, v1.0, REC-xml-19980210, Feb 1998

[8] SyncML initiative, SyncML Technical Specifications, version 1.0 (http://www.syncml.org)

[9] 3GPP TR 27.903, Discussion of Synchronization Standards

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

Bluetooth: a technology specification for short range radio links between mobile PCs, mobile phones and other portable devices. (http: //www.bluetooth.com/)

GET: the operation of requesting that the server returns an object to the client as defined in the IrDA IrOBEX specification

GSM: Global System for Mobile communications

HTTP: HyperText Transfer Protocol

IrDA: an industry consortium set up to define a set of short range Ir communications standards. (http: //www.irda.org/)

Level 1: minimum level support defined in the IrDA IrMC set of specifications

Level 2: access level support defined in the IrDA IrMC set of specifications

Level 3: index level support defined in the IrDA IrMC set of specifications

Level 4: sync level support defined in the IrDA IrMC set of specifications

MapItems: describes to the server the mapping of a local UID to a server UID

MIME: Multipurpose Internet Mail Extension

PUT: the operation of sending one object from the client to the server as defined in the IrDA IrOBEX specification

SSL: Secure Socket Layer

Sync Alert: A SyncML command for requesting a synchronization on a particular datastore.

Synchronization: the process of exchanging information between multiple physical or virtual locations for the purpose of ensuring that each location's copy of that information reflects the same information content

SyncML initiative: an industry initiative set up to define a data synchronization standard based on XML (http://www.syncml.org/)

vCalendar: a format defined by the IMC for electronic calendaring and scheduling exchange with extensions as defined in the IrDA IrMC set of specifications

vCard: a format defined by the IMC for electronic business card exchange with extensions as defined in the IrDA IrMC set of specifications

WAP: an industry consortium set up to define a set of standards to empower mobile users with wireless devices to easily access and interact with information and services. (http: //www.wapforum.com/)

Wide Area Network: a geographically-large range wireless connection between two or more devices for the purpose of transferring information. Large geographical range is typically defined as one kilometer or more in distance

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

Cookie	a method of tracking http-based information
IETF	Internet Engineering Task Force
IMC	Internet Mail Consortium
Ir	Infrared
IrDA	Infrared Data Association
IrMC	Ir Mobile Communications
IrOBEX	Ir Object EXchange
OBEX	Object Exchange
PDA	Personal Digital Assistant
PIM	Personal Information Manager
SyncML	Synchronization Markup Language.
UID	Unique Identification
URL	Universal Resource Location
WAP	Wireless Application Protocol

WBXML	Wireless Binary XML
WML	Wireless Markup Language
WSP	Wireless Session Protocol
XML	eXtensible Markup Language

4 Background

This background material is a synopsis of the material presented in 3G TR 27.903. Note that Release 99 is based on IrMC Sync. This release reflects a change to SyncML.

4.1 IrMC

The IrMC standard was developed as an extension to the IrDA standard for the purpose of providing an open standard for data exchange between mobile devices or between mobile devices and desktops or PDAs. Among other things, IrMC defines four levels of support for information exchange. By definition, each higher level must support all of the preceding levels. The four levels are: Level 1 (Minimum Level), Level 2 (Access Level), Level 3 (Index Level), and Level 4 (Sync Level). Level 4 does not require Level 3. Level 2 and Level 4 are the most relevant for synchronization. IrMC has been adopted by IrDA and Bluetooth initiatives.

4.2 Bluetooth

Bluetooth has adopted the IrMC standard as the basis for their synchronization specification.

4.3 WAP

WAP has not specified a synchronization standard. The WAP Forum is currently evaluating synchronization technologies and is expected to identify a technology later in 2001.

4.4 SyncML

SyncML is an XML-based specification for data synchronization. It accommodates not only traditional local synchronization but also the special requirements associated with remote synchronization in wide-area wireless environments with intermittent connectivity. SyncML is based on a client-server model. SyncML specifications consist of three major components: representation protocol, synchronization protocol, and transport bindings. The Representation protocol defines XML-based messages for synchronization, whereas the Synchronization protocol defines synchronization in the form of message sequence charts. The Transport binding specification defines a mechanism to carry synchronization messages over different transport mechanisms.

5 SyncML

5.1 SyncML Overview

SyncML was designed with a number of goals:

- Reduce the number of messages being sent over the air,
- Reduce the message size,
- Provide a robust authentication mechanism,
- Transport independence,
- Datatype independence.

5.1.1 Reduce the number of messages being sent over the air

To achieve a reduction in the number of messages being transmitted, the SyncML device is required to determine which of the objects to be synchronized has changed since the last synchronization for each server. This way, the SyncML device may tell the server what has changed, and then the server may tell the device what has changed and together they may determine the most appropriate course of action. In the case where there are no changes from the server or client, there could be as few as two messages exchanged. In the case where the server is sending additions to the device, there could be as few as four messages exchanged.

A detailed explanation of the messages being exchanged may be found in the protocol document [8].

5.1.2 Reduce the message size

SyncML messages can be in either XML or WBXML. The XML version is rather large, as the tags are clearly spelled out. The XML version makes for easier debugging, but is much too large for normal use over a wireless network. WBXML has the same structure as XML, but replaces the text strings with binary tags. This produces much smaller messages, at the expense of human readability. SyncML devices are required to support only XML or WBXML, not both. SyncML servers must support both XML and WBXML.

5.1.3 Provide a robust authentication mechanism

SyncML has required authentication on the message level, and optional authentication for the datastores and individual objects. The authentication is either Basic (username:password) or MD5 (username:password:nonce). Stronger authentication and encryption are work items for the year 2001. To keep the authentication free from casual viewing, the data is base64 encrypted.

It is possible to send an initial message to the SyncML server using Basic authentication, and have the server reject the message, asking instead for MD5 authentication. It is also possible authentication for every message being sent to the server and device.

5.1.4 Transport Independence

SyncML has the capability of operating over a several transport mechanisms. Version 1.0 specifies OBEX, WSP and HTTP.

To achieve this independence, SyncML has defined a set of specifications and conformance for each transport. Each transport must be able to carry a SyncML message reliably between SyncML devices. Each transport must be able to send a message to a device, and then be able to wait for a response from that device.

5.1.5 Datatype Independence

SyncML has specified a minimal set of objects for SyncML servers to support, but has not made the same requirements for SyncML clients. These objects are vCard 2.1, and vCalendar 1.0. SyncML only requires that an object be a registered MIME type for it to be synchronized. Both a client and server must support this object for it to be synchronized. The Device Information object indicates which objects are supported for any datastore. The Device Information Object must be exchanged on the first synchronization between a client and server.

Typically, SyncML servers will support a wide variety of datatypes, while a SyncML client will support only one datatype (in the interest of smaller footprint). If a new datatype is created, it is very easy for this to be supported.

5.2 Change of Client/Server Roles from Release 99

SyncML uses the following description for client/server roles:

> “A client is a device (or application) which initiates a request for a session. The server is a device that passively waits for session requests from client devices. The server can either accept the request or reject it.”

IrMC defines the client as:

> “The IrMC Client is the device that initiates communication with an IrMC Server. Typical IrMC Clients are PCs. However, in some cases, PDAs, pagers and phones may also be IrMC clients. IrMC Clients can only communicate with IrMC servers. For instance, an IrMC client may request a particular Phone Book record from an IrMC Server.”

IrMC also defines the server as:

> “The IrMC Server listens for requests from IrMC clients. Typical Servers are pagers, phones and PDAs. IrMC Servers can not initiate communication, and can only communicate with IrMC clients.“

5.3 Transport Bindings

The transport to be used should be either HTTP or WSP.

Details on the transport bindings may be found on the SyncML website. [8]

5.4 Security

Each SyncML message must exchange authentication credentials on each message level and may exchange additional authentication for each datastore as well. This authentication process will only guarantee that the client and server can rely on each other for the duration of the session. For longer duration security, the basic level of authentication is not adequate, instead MD5 authentication should be used. This will guarantee authentication not only over a session, but between sessions as well.

Encryption is not currently available in SyncML and must be supplied by the transports. Transport from the mobile device to the gateway is well protected. Transport from the gateway to the server should be sent via a secure channel, such as HTTPS.

5.5 Connection

The connect sequence sets up the connection from the mobile device to the synchronization server. The client must choose a session id that is unique between the client and server and must determine what level of authentication to use (basic or MD5). Once the authentication has been decided, the client is free to start the actual connection process. Note that the connect procedure is always invoked by the client and that the transport binding documents detail the connection process.

Persistent connections should be terminated only after the last expected response from the server has been received.

5.6 Sending and Receiving Information

Both WSP and HTTP transport bindings use the Post method for sending and receiving SyncML messages to the server.

5.7 Intermittent Connectivity

Both the WSP and HTTP transport bindings operate over networks with high latency and have timeouts built in. Both transport bindings operate as follows:

> “In the case of a server timeout, the SyncML client SHOULD establish a new HTTP session with the HTTP server and attempt to resend the current SyncML package, beginning with the first SyncML command for which the SyncML client has not received an acknowledgement. In the event that the SyncML client requested that no responses be sent, the SyncML client SHOULD begin retransmitting with the first SyncML command in the SyncML package.
>
> In the case of a client timeout during a SyncML client-initiated data synchronization, the SyncML server SHOULD clean up the TCP connection and do no further processing of the SyncML request.” [8]

5.8 Compatibility with Release 99

Compatibility between this release and Release 99 assumes that, within the network, there is a target server to act upon synchronization transactions. This target server is the destination or origin for all ME synchronization translations.

This target server has to be able to differentiate between Release 99 Sync and newer versions of Sync. If the new transport link is OBEX, then there will be different OBEX target headers for Release 99 and newer releases. If the new transport link is not OBEX, then the server will differentiate between Release 99 and newer releases by port number or MIME type. In either case, there is a distinctive method for determining how to service the Sync transaction.

Compatibility also assumes that the existing data store types may be maintained as currently defined in both the client and the server. Newer data store types may not need to maintain backwards compatibility since the older Release 99 clients would not understand the new types.

5.9 Use Case with SyncML

The user wants to synchronize data. If the user has not previously set up the remote synchronization, the user will have to enter in the URL for the server, the authentication data (user name, password, and possibly a nonce), as well as any datastores to be synchronized. The device will have to maintain this information in the local storage of the device.

The user then chooses with which remote server to synchronize and then starts the synchronization. The server will receive a message from the client with authorization information and a Sync Alert for each datastore to be synchronized. The server will send back a message to the client with a status for all of the Sync Alerts as well as the authorization. Note that the server has the option to tell the client to not send any more authentication at this point. Likewise, the server could tell the client to switch to more robust authentication and have it resend the first message with MD5 authentication.

The client will send a message to the server with all of the changed data since the last synchronization with that server. The server will respond with a message containing a status on all of the changed data and all of server-changed data since the last synchronization.

If there is any data from the server, then the client will have to send a message back to the server with status on the server's changed data requests, and possibly some MapItems. MapItems are used to tell the server how to map the local UIDs to the server's UIDs. MapItems are only sent to the server if the server has sent any new objects to the client. If the client has sent a new objectto the server, the server will take care of any UID mapping at that time.

If the client had sent any MapItems to the server, then the server will send a final message to the client with a status on all of the MapItems sent.

The user will then see a prompt telling them the synchronization is now complete and, possibly, indicate any errors.

Annex A (informative): Change history

Change history							
Date	TSG #	TSG Doc.	CR	Rev	Subject/Comment	Old	New
22/09/00	T#9	TP-000143	001	1	Introduction of PUSH and TARGET	3.0.0	3.1.0
15/03/01	T#11	TP-010028	002		Addition of SyncML	3.1.0	4.0.0
07/06/02	T#16	-	-		Upgrade to Rel-5	4.0.0	5.0.0