The Wireless Communication Transfer Protocol (WCTP)

Protocol Specification

Revision:	1.1	**Status:**	Released
Release Date:	December 29, 2000	**Document ID:**	wctp-v1r1.doc

Personal Communications Industry Association
http://www.pcia.com

Participation in WCTP Developments

Comments, contributions, and discussions may be submitted to the Personal Communications Industry Association through its Web site http://www.pcia.com. PCIA members who wish to contribute to the efforts to enhance this protocol are encouraged to join the WCTP technical sub-committee. Non-PCIA members who wish to contribute suggestions for modification of the protocol or to suggest new features and capabilities are encouraged to work with a member of the technical committee to represent their ideas to the committee.

Table of Contents

WCTP COMMITTEE ACKNOWLEDGEMENT V
REVISION HISTORY VII
REFERENCES IX
LIST OF FIGURES X

1 INTRODUCTION 1

1.1 GOALS OF WCTP 1
1.2 FUTURE OF WCTP 1

2 WCTP PROTOCOL PROCEDURE 3

2.1 WCTP FRAMEWORK 3
2.2 WCTP COMMUNICATIONS THROUGH HTTP 3
2.2.1 Pushing WCTP from the Wireline to Wireless Network 3
2.2.2 Pushing WCTP From the Wireless to Wireline Network 5
2.2.3 "Pseudo-Pushing" of WCTP from the Wireless to Wireline network 6
2.3 TWO CATEGORIES OF WCTP USERS 7
2.3.1 Enterprise Hosts 7
2.3.2 Transient Clients 7
2.4 ADDRESSING IN WCTP 8
2.4.1 General Address Identification in WCTP 8
2.4.2 transport-address Format 8
2.4.3 entity-address Format 9
2.4.4 WCTP Identifiers for Wireless Devices 10
2.4.5 WCTP Identifiers for Wireline Hosts 10
2.4.6 Identifiers For Transient Clients 10
2.4.7 Message Response Redirection to non-WCTP Destination 10

3 WCTP STANDARDS 11

3.1 XML SYNTAX 11
3.2 DATE AND TIME FORMATS 11
3.2.1 WCTP Date Format 11
3.2.2 WCTP Time Format 11
3.2.3 Subsecond Resolution 12
3.2.4 Time Zones 12
3.2.5 WCTP Date and Time Format 12
3.3 STATUS AND ERROR CODES 12
3.3.1 No HTTP Relationship 12
3.3.2 Status and Error Code Returned Text Strings 13
3.3.3 Categories of Status and Error Codes 13
3.4 WCTP OPERATION DECLARATIONS 14
3.5 WCTP PROTOCOL VERSION 14
3.6 WCTP TOKENS 15

4 WCTP OPERATIONS 16

4.1 WCTP OPERATIONS OVERVIEW 16
4.1.1 Submit Message Operations 16
4.1.2 Status Operations 16
4.1.3 Message Reply Operation 16

4.1.4 Poll Operations *17*
4.1.5 Query Operations *17*
4.2 VALID RESPONSES 19
4.3 CALL FLOW BETWEEN NETWORK ELEMENTS 19
4.4 WCTP SUBMIT REQUEST OPERATION 23
4.4.1 Message Payload *23*
4.4.2 Submit Header *24*
4.5 WCTP MESSAGE REPLY OPERATION 24
4.5.1 Response Header *24*
4.6 WCTP STATUS INFO OPERATION 25
4.7 WCTP CONFIRMATION OPERATION 26
4.8 WCTP POLL FOR MESSAGES OPERATION 26
4.9 WCTP POLL RESPONSE OPERATION 27
4.10 WCTP LOOKUP SUBSCRIBER OPERATION 27
4.11 WCTP LOOKUP RESPONSE OPERATION 27
4.12 WCTP SUBMIT CLIENT MESSAGE OPERATION 27
4.13 WCTP SUBMIT CLIENT RESPONSE OPERATION 28
4.14 WCTP CLIENT QUERY OPERATION 28
4.15 WCTP CLIENT QUERY RESPONSE OPERATION 29
4.16 WCTP VERSION QUERY OPERATION 29
4.17 VERSION RESPONSE OPERATION 30
4.17.1 Contact Information *31*
4.17.2 Failure *32*
4.17.3 DTDSupport *32*

APPENDIX A - WCTP DOCUMENT TYPE DEFINITION 34

APPENDIX B - WCTP FIELD DEFINITIONS 42

APPENDIX C - ERROR AND SUCCESS CODES 50

APPENDIX D – EXAMPLE OF WCTP BASED APPLICATIONS 54

APPENDIX E - HISTORY OF WCTP 65

1. A NEW STANDARD IS NEEDED 65
2. WCTP AND ITS RELATIONSHIP TO WAP 66

WCTP Committee Acknowledgement

The following committee members contributed to the development, enhancements and promotion of this specification.

Chairman

Edward “Ted” Sumner	Vast Solutions	edward_sumner@vast.com

Past Chairman

Wiley Wilkins	Vast Solutions	wiley_wilkins@vast.com

Originating Authors

Jay Moskowitz	RTS Wireless	jmosk@rtswireless.com
Dr. Yuri Salkinder	RTS Wireless	yuri.salkinder@rtswireless.com

Drafting Committee

Committee Member	Company
Dr. Alfred Adler	Motorola
Raj Cherukuri	Skytel
Dr. Michael Christiansen	Pagenet
Vic Cox	Weblink Wireless
John Davis	Weblink Wireless
Bruce Deer	Skytel
Don Gayton	Glenayre
Mike Johnson	Skytel
Jerry Karasz	Skytel
Anthony Klinkert	Pagenet
Brian Laird	Glenayre
Kris Latham	Arch Wireless

Committee Member	**Company**
Kyung Tae Mun	Weblink Wireless
Jay Moskowitz	RTS Wireless
Demitrius Nelon	Arch Wireless
Bijan Nowroozi	Pagenet
Garland Phillips	Motorola
Jeff Poe	Space Data Corporation
Dr. Yuri Salkinder	RTS Wireless
Karl Schlieber	Bell South Wireless Data
Bill Scott	Pagenet
Bonnie Shapbell	Skytel
Dwight Smith	Motorola
Ted Sumner	Pagenet
Benjamin Sun	Pagenet
Chris Wells	Glenayre
Shawn Welsh	Skytel
Wiley Wilkins	Pagenet

Revision History

Document Number	Date	Description
wctpv1r1.doc	December 29, 2000	Added wctp-VersionQuery. Removed wctp-BindDomainAlias. Addressing clarified and corrigenda incorporated
wctpv1r0.doc	June 6, 2000	Updated specification to reflect WCTP as a PCIA supported protocol. Update references to the WCTP DTD to show that it now exists at the PCIA Web site.
wctpv0r7.doc	November 1, 1999	Final revision of the first “official” version of the protocol presented to the PCIA. Accepted as a standard by the Paging Technical Committee of the Personal Communication Industry Association (PCIA). Use of the uuencode scheme for the transparent data transfer is discontinued, and the use of the XML “canonical” encoding is discouraged in favor of base64 encoding.
wctpv0r6.doc	June 18, 1999	Minor typos and inconsistencies are fixed in the use cases. All the use cases were validated through an XML parser.
wctpv0r5.doc	May 17, 1999	Release of drafting committee preliminary version 0.4 with a major portion of the 21 defined use cases included.
wctpv0r4.doc	April 9, 1999	Incorporated changes agreed to by the members of the drafting committee, developed during a series of meetings and conference calls. This is major update to the DTD and rewrite of the specification from the previous release. Changes span the entire document and are too numerous to list individually.
wctpv0r3.doc	February 22, 1999	Re-release of specification under the name of the Messaging Standards Committee instead of RTS Wireless, the original author.
wctpv0r2.doc	January 29, 1999	Initial Release by Jay Moskowitz, CTO, RTS Wireless, a.k.a. Real Time Strategies, Inc. Introductory specification released after a presentation to the Messaging Standards Committee, January 13, 1999. As a result of that meeting, certain aspects of WCTP have been

		eliminated from this specification so as to simplify its description and to avoid the addressing of implementation details which should be outside of the scope of this document.

References

The following documents and Web site addresses provide information that is pertinent to this specification.

- Reference information regarding XML, Web site: http://www.w3.org/XML
- RFC 822, Standard For The Format Of Arpa Internet Text Messages: http://www.ietf.org/rfc/rfc0822.txt
- RFC 1238, Uniform Resource Locators (URL): http://www.ietf.org/rfc/rfc1238.txt
- RFC1945, Hypertext Transfer Protocol -- HTTP/1.0: http://www.ietf.org/rfc/rfc1945.txt
- RFC2068, Hypertext Transfer Protocol -- HTTP/1.1: http://www.ietf.org/rfc/rfc2068.txt
- RFC2396, Uniform Resource Identifiers (URI): Generic Syntax: http://www.ietf.org/rfc/rfc2396.txt
- Motorola, FLEXsuite™ of Enabling Protocols, Document Number: 6881139B10
- Motorola, ReFLEX™ 25 Specification, Document Number: RFL25LX-033195
- Personal Communications Industry Association Web site: http://www.pcia.com
- WAP Forum Web Site: http://www.wapforum.org
- ISO8601, Web Site: http://www.iso.ch/
- WCTP Working Group Web Site: http://www.wctp.org

List of Figures

Figure 2-1 Pushing WCTP Operations to the Wireless Network 5
Figure 2-2 Pushing WCTP Operations From the Wireless Network 5
Figure 4-1 Call Flow Between Network Components 21
Figure 4-2 WCTP Submit Request Operation 23
Figure 4-3 WCTP Message Reply Operation 25
Figure 4-4 WCTP Status Info Operation 26
Figure 4-5 WCTP Confirmation Operation 26
Figure 4-6 WCTP Submit Client Message Operation 28
Figure 4-7 WCTP Client Query Response Operation 29

1 Introduction

The Wireless Communications Transfer Protocol (WCTP) is specifically aimed at creating a simple means of passing alphanumeric and binary messages between wireline systems and two-way capable wireless devices. WCTP does not require that the wireless device be capable of responding. A one-way numeric pager would be equally as accessible as any device capable of bi-directional messaging. Although introduced through the paging industry, WCTP is directly applicable for messaging to and from most other wireless technologies including PCS, GSM, and cellular.

WCTP has been created to provide a wireless industry specific standardized representation of messages that are:

- Sent from a wireline host to a wireless device.
- Sent from a wireless device to a wireline host.
- Sent from a wireless device to another wireless device on another wireless network.

1.1 Goals of WCTP

The major goals to be derived through WCTP are:

- To interconnect Internet accessible systems with wireless networks.
- To capitalize on current and emerging Internet standards which will dramatically simplify the process of implementing WCTP over other standards that are straightforwardly usable over the Internet.
- To send and receive messages containing any type of binary data content.
- To support as many wireless networks as possible.
- To create an extensible architecture allowing unlimited growth to accommodate new features and capabilities.
- To support a wide range of wireless devices.
- To support the ability for wireless devices to send messages to other wireless devices located on other wireless carriers' networks.

1.2 Future of WCTP

Many suggestions have been made to improve WCTP. It is not known which will be implemented in future versions. The primary goals are to simplify use and practical application of WCTP.

These suggestions include:

- Splitting the protocol into a suite with logical functional groupings.

- Organizing a single repository for aliasing long addresses in order to save on wireless bandwidth.
- Adding device or wireless protocol specific instructions for addressing.
- Augmenting the DTD and creating operations that support the ability to determine the behavior or parameters of a wireless gateway.

2 WCTP Protocol Procedure

2.1 WCTP Framework

The WCTP protocol defines a number of different control blocks that are transferred between a wireline system and an entry point, or gateway, into the wireless network. Such gateway understands the wireless network specifics and the native protocols used to communicate within those networks. It is a function of the gateway to map those specifics into the WCTP standardized representations in order to provide uniform appearances regardless of the wireless network over which data is being carried. The gateway "does what is necessary" to provide this mapping. This may include the maintenance of transactional information by the gateway through the "lifetime" of certain messages in order to make up for the inability of certain specific networks to carry all of the information provided within WCTP. The gateway creates WCTP messages on behalf of the wireless device or on behalf of the wireless network reporting a significant event regarding the message delivery process.

Each WCTP control block is referred to as a WCTP Operation. The Operation defines a function being requested or a response being returned. The Operation also carries mandatory and optional parameters associated with it. WCTP Operations are represented in an XML format as dictated by the format language syntax or Document Type Definition (DTD) described in Appendix A.

A WCTP Operation may be initiated by either a wireline system or a wireless gateway. An Operation may result in one, many or no direct responses to it.

WCTP is a request/response type protocol, and as such, some WCTP Operations represent protocol requests, and others represent protocol responses. Full correlation of WCTP request and response Operations is presented later is this Specification in Section 4.2.

2.2 WCTP Communications through HTTP

WCTP is called a “transfer” protocol in that it is transferring information content between wireline and wireless systems. The manner, in which the WCTP defined operations are transported between a pair of systems, is independent of WCTP syntax.

Although any number of transport protocols may be used to move WCTP operations between a pair of systems, the HyperText Transfer Protocol (HTTP) has been selected as the recommended transport protocol for WCTP information. HTTP was selected as the protocol of choice because its use already addresses the issue of security and firewall protection typically found within those corporations that allow Internet access and that are expected to be the primary users of the protocol. The firewalls maintained by the security departments of most corporations normally allow HTTP requests to be initiated from systems that are within a corporate network. Because of this, WCTP will be able to be deployed in most corporate environments without coordination with the security administration of the network.

2.2.1 Pushing WCTP from the Wireline to Wireless Network

A wireline system sends (or pushes) information to the Gateway of the wireless network through the use of the HTTP protocol. This is the same protocol used by browser software to request and receive information from the World Wide Web and to forward information to Internet based servers to perform some function based upon the data forwarded (such as a search function, a database lookup, adding a name to a mailing list or ordering goods).

HTTP is a client/server protocol. An HTTP client always makes a request to an HTTP server. HTTP supports two *major* processes. It may GET (or pull) information from a server (such as a Web page) or it may POST data to a server. Regardless of which process is requested, there is always a response sent to the request. The response always contains an HTTP protocol header field optionally followed by some returned data.

In WCTP, the POST process is used to send a WCTP formatted request (WCTP Operation) to the WCTP server. The response by the server is a WCTP XML formatted response (another WCTP control block or Operation). The types of control blocks and their formats are discussed Chapter 4.

The HTTP content type to be used in WCTP communications is text/xml.

The WCTP server MUST support HTTP version 1.0 and MAY support HTTP version 1.1.

A typical HTTP 1.0 POST header to a WCTP gateway would look like this:

```
POST /WCTP HTTP/1.0
Content-Type: text/xml
Content-Length: 503
```

A typical HTTP 1.0 response header would look like this:

```
HTTP/1.0 200 OK
Date: Wed, 04 Oct 2000 14:05:44 GMT
Server: Apache/1.3.9 (Unix)
Content-Length: 406
Content-Type: “text/xml"
```

A typical HTTP 1.1 POST header to a WCTP gateway would look like this:

```
POST /WCTP HTTP/1.1
Content-Type: text/xml
Content-Length: 503
```

A typical HTTP 1.1 response header may look like this:

```
HTTP/1.1 200 OK
Date: Wed, 04 Oct 2000 14:05:44 GMT
Server: Apache/1.3.9 (Unix)
Content-Length: 406
Connection: close
Content-Type: "text/xml"
```

or like this:

```
HTTP/1.1 100 CONTINUE
Date: Wed, 04 Oct 2000 14:05:44 GMT
Server: Apache/1.3.9 (Unix)
Content-Length: 406
Content-Type: "text/xml"
```

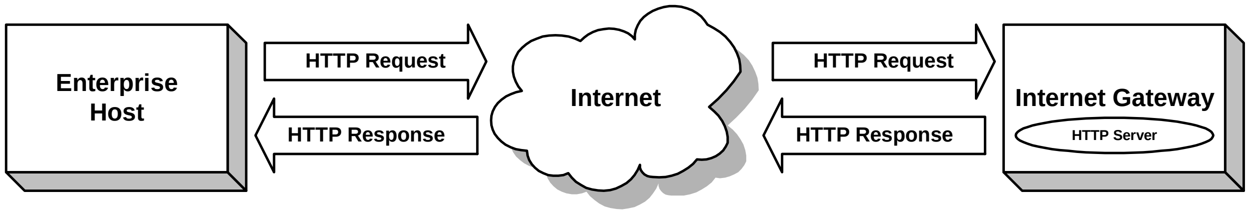


Figure 2-1 Pushing WCTP Operations to the Wireless Network

Figure 2-1 shows the normal call flow from the client to the server when a wireline client sends a request to the wireless network.

2.2.2 Pushing WCTP From the Wireless to Wireline Network

The wireless network can forward data to the wireline network in the HTTP response to a previous HTTP request as shown in Figure 2-2. But when the wireless network has unsolicited information available, it would like to push this data out to the wireline system. There are several methodologies employed within the Internet today to "Push" information from servers to client machines. Two forms of push technology are currently recommended by this specification.

The recommended means of pushing information from the wireless network to the wireline system is also through the use of the HTTP protocol. In this case, the TCP/IP connection is initiated by the wireless network and an HTTP request carries a POST operation that contains an XML formatted WCTP Operation as its data content. The WCTP Operation carried in the HTTP response from the wireline system indicates if the data sent has been accepted, processed and/or safe-stored by the wireline system. If so, the wireless network can drop the WCTP Operation from its queues knowing that the information has been properly conveyed to the wireline system. Figure 2-2 shows the call flow of WCTP Operations pushed from the wireless network to the wireline system.

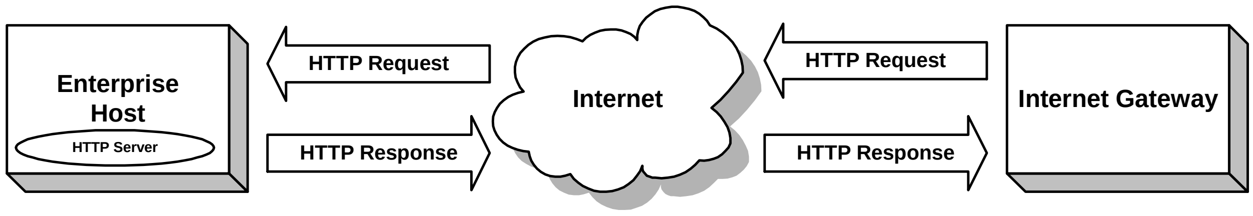


Figure 2-2 Pushing WCTP Operations From the Wireless Network

The HTTP protocol has been selected as the transport protocol of choice to push data to the wireline systems, again because of security and firewall issues. Many corporate networks are already operating Web servers where this HTTP data may be forwarded for processing. Special applications on these Web servers could process the received data without having to open special TCP ports in firewalls.

The major requirement to support this type of push technology is that the wireline system must be running some type of HTTP server software in order to accept and process the WCTP Operations pushed from the wireless network. A minimum requirement is to accept a HTTP 1.0 post in accordance with RFC 1945.

In support of this form of push, addressing information that is carried by the WCTP protocol, knows the host and domain name of the wireline server which is to process the pushed data from the wireless network. The standard WCTP message routing mechanism derives the host and domain name from the domain portion of the recipient ID.

The HTTP protocol also allows "path information" to be included as part of the request. This path can signify the location of particular applications software on the server that is to specifically process the WCTP Operation being sent. In the case of a wireless device sending a message to wireline host for processing, this path information can be forwarded by the wireless device in order to specify that a specific application should process this message.

2.2.3 "Pseudo-Pushing" of WCTP from the Wireless to Wireline network

If the wireline network is not able to run an HTTP server to accept pushed data from the wireless network, a pseudo-form of pushing is used. When this form of push is supported, the wireless network queues the data that normally would have been pushed and instead forwards this data in response to a *Poll For Messages* Operation which is sent from the wireline system through an HTTP request as shown in the Sections 4.2 and 4.8. For examples please refer to the *Use Case Companion* document. The response to the *Poll For Messages* may indicate that there is no data to return to the wireline system, or it may return one or more queued WCTP Operations in response. The *Poll For Messages* Operation can control the maximum number of possible WCTP Operations returned. If any WCTP Operations are returned, a subsequent *Poll For Messages* must carry information that indicates that the returned WCTP Operations have been acknowledged by the wireline system so that the wireless system may remove them from its queues.

For a wireline system to support this manner of push, it must be registered at the wireless network for a polling mode of operation. The wireline system employing polling has a unique host identifier in the form *host.domain* that is being used as a part of sender or recipient ID registered with the wireless network. When this host identifier is encountered in the domain portion of the recipient ID of the WCTP Operation, the wireless network will hold the data in its queues that it would have normally immediately pushed out using the mechanism of Section 2.2.2.

2.3 Two Categories of WCTP Users

There are two major categories for wireline systems that utilize the WCTP protocol. These are the Enterprise host and the Transient client. There are specific WCTP protocol operations for each type of user as will be discussed in this section.

The Enterprise host is a system that operates on behalf of one or more message senders and/or message processing applications. It is capable of receiving unsolicited messages from wireless devices. These unsolicited messages may be directed to a general or specific processing application at the Enterprise host as discussed in the following sections. The wireless network will normally push status, message replies and unsolicited messages to the wireline host as soon as such information is ready to send as described in Section 2.2.2. In those cases where security considerations or host limitations do not permit the pushing of information, the Enterprise host will use the pseudo-push mechanism described in Section 2.2.3.

The Transient client is categorized as a message sender who may not have a permanent connection to the Internet and periodically sends messages to one or more wireless devices. The Transient clients may receive message status information and replies for specific messages that they have submitted into the wireless network. They may not receive unsolicited messages from wireless devices. They cannot have data pushed to them as described in Section 2.2.2. Instead, they must use *Client Query* operation to query the wireless system for information using a polling mechanism similar to that described in Section 2.2.3, but only capable of returning responses to a given single original message.

2.3.1 Enterprise Hosts

In the WCTP environment, the Enterprise host is expected to act as a Gateway for one or more machines within the Enterprise. Messages may be originated from one or more users or applications at individual machines connected on a LAN or WAN to the Enterprise host. It is expected that the Enterprise host will accumulate these messages and submit them to the wireless network as if originating from a common host. Responses on behalf of these messages will be returned to the Enterprise host that will then direct the responses to the proper recipient or application.

Unsolicited messages from wireless devices are addressed to the Enterprise host designated to process such messages.

This release of the WCTP protocol assumes that the Enterprise host is capable of accepting messages pushed via the HTTP protocol. This may be accomplished if the Enterprise host is already running a Web Server or other application that is capable of receiving HTTP POST's. Where this is not feasible, the "pseudo-push" technique is employed (see Section 2.2.3).

Messages submitted by an Enterprise Host are assigned a Message Identifier (messageID) by the host. This Message Identifier is returned to the host in all replies and responses to the submitted message.

2.3.2 Transient Clients

A WCTP Transient client may submit messages to many different wireless devices. However, this type of user does not receive a consolidation of all of its status responses, delivery notifications and replies returned to it automatically as in the case of the Enterprise host.

Instead, the transient message sender may check for data regarding previously submitted messages. In support of such checking, the system returns a message tracking number whenever a message is submitted. This tracking number is used in subsequent *Client Query* Operations. (See Section 4.14)

2.4 Addressing in WCTP

2.4.1 General Address Identification in WCTP

Address identifiers in WCTP are presented in the following general form:

[entity-address@]transport-address

transport-address is used by the WCTP transport layer to deliver WCTP operation to a WCTP Gateway or an Enterprise host.

Optional **entity-address** is used by a WCTP server or an Enterprise host to identify the ultimate source or destination of the WCTP operation. It can represent a wireless device, a user name, an application alias or can carry any other carrier or host-specific meaning.

The character set for both transport- and entity-address portions must conform with RFC 2396, section 2 URI Characters.

> **Note:** While in most cases WCTP address identifiers appear similar to either e-mail identifiers of RFC 822 or URL/URI identifiers of RFCs 1738 and 2396, the full general address specification form in WCTP is not compliant with any one of the mentioned RFCs.

The use of aliases to represent transport-address or full WCTP address is not described, as these are converted by the specific serving WCTP gateway and associated elements into fully qualified WCTP addresses before transport occurs. Serving WCTP gateways must map aliases into the format described herein. Assignment of aliases is specific to a particular gateway implementation and beyond the scope of this specification, but may be coordinated among cooperating carriers to prevent duplicates. One possibility is for carriers to prepend a carrier designation to aliases, for example, sk! for Skytel or mc! for Metrocall. Aliases are expected to be limited by wireless carriers to less than 40 characters in total length.

2.4.2 transport-address Format

transport-address has the following general form:

[Protocol**:**[**//**]]Domain[**:**Port][**/**Path]

Domain is used to locate the WCTP server or the Enterprise host where WCTP operation is being delivered. Furthermore, in case of a delivery to an Enterprise host, the domain portion is used to determine if the message is to be pushed to the Enterprise host or held for polling by the Enterprise host.

For use in push operation the domain specification of a WCTP server or an Enterprise host should present fully qualified domain name resolvable via DNS or an IP address.

Both the optional **Port** and **Path** contain additional routing information for the WCTP transport layer (by default, HTTP). If no forward-slash "/" is present (the Path portion of the address is not specified), default path **/wctp** must be assumed. The prefix wctp. must be prepended to a valid domain. Many domains are expected to co-locate a WCTP gateway on the same Web server, but as a separate sub-domain, e.g. wctp.carrier.net. If an IP address is used, then "wctp." must not be prepended to it. If "wctp." is already prepended to a valid domain, then wctp. must not be prepended again. The double forward-slash "//" after the Protocol: is optional. The designation Protocol:// should not be used even where /Path is included in the transport-address portion in order to reduce characters transported wirelessly.

Optional **Protocol** is used to identify the alternative transport for WCTP. If Protocol is not specified, the default **http:** is assumed and means standard WCTP transport over HTTP. Other protocols can be used, for example, smtp: or tcp:. The serving WCTP gateway as well as the destination gateway must support the transport protocol specified by the sender.

For example, a recipient of the message is specified as:

inventory@http:**myenterprise.com**

or as:

myenterprise.com*/processing/request.jsp*

or as:

joe.user@**wctp.myenterprise.com**/

The domain portion of the address in the first two examples is **myenterprise.com**, and in the last one is **wctp.myenterprise.com**. However, the name of the target host in all three cases will be derived as **wctp.myenterprise.com.**

The POST command of the HTTP protocol in the first example will be formed as:

POST /wctp HTTP/1.0

because the *path* portion is omitted. If the *path* portion of the address is explicitly specified, it is used as is. Thus in the second example the POST command would be formed as:

POST /processing/request.jsp HTTP/1.0

and in the third example:

POST / HTTP/1.0

2.4.3 entity-address Format

entity-address has the following general form:

[Scheme**:**]Entity[**:**Port][**/**Path]

Entity as well as optional **Port** and **Path** present identification and possibly additional routing information meaningful to the corresponding side of WCTP communication.

Optional **Scheme** component may be used for the specification of entity identification method and is compliant with a notion of scheme from RFC 2396. Scheme, if specified, can be used by infrastructure of wireless network or an enterprise to distinguish and properly route WCTP traffic. In the absence of a specified scheme, wctp: is assumed by default.

2.4.4 WCTP Identifiers for Wireless Devices

For wireless devices, Entity portion of a WCTP identifier can represent a numeric identifier of a wireless device or its alias. In this case, Port and Path parts of entity-address are relative to the wireless device id, and may specify delivery information for an application running on a wireless device.

Examples:

1234567@abcwireless.com
8005551212.98765@abcwireless.com
wctp:john_doe@abcwireless.com
john_doe/chess_app@abcwireless.com

2.4.5 WCTP Identifiers for Wireline Hosts

In case of Enterprise hosts, use of Entity, Port and Path components of entity-address is particular to the implementation of the Enterprise host. Entity can represent a user name, an application alias or carry any other Enterprise host specific meaning. Combined with optional Port and Path, they can be used for internal message routing and processing. This specification does not define the standard behavior or representation for these components.

Examples:

inventory@myenterprise.com
joe.user@https:myenterprise.com/wctp-gateway
wctp:accounting/receivables@http://myenterprise.com:8080/process-wctp
im:johndoe@im-gateway.com

2.4.6 Identifiers For Transient Clients

The format of identifiers for Transient clients in WCTP is not fixed. Normally, uniqueness of such an identifier cannot be guaranteed. Therefore, wireless systems cannot rely on the sender identifier for messages submitted by Transient clients. In case of Transient clients, for delivery of responses the wireless system must rely on the uniqueness of an assigned tracking number or a combination of tracking number and recipient identifier of the original message. The senderID, specified on a submitted message by a Transient client, must be specified for authentication when querying for status.

2.4.7 Message Response Redirection to non-WCTP Destination

WCTP allows specification of where the responses to a particular message are to be sent via sendResponsesToID attribute of wctp-MessageControl. Wireless Carriers may support redirection of responses to non-WCTP destinations, like e-mail, faxes, phones, etc. To support this feature WCTP reserves the following prefixes that can be specified in sendResponsesToID: **mailto:**, **faxto:**, **phoneto:**.

WCTP server may reject messages requesting response redirection if the requested method of redirection is not supported.

3 WCTP Standards

This section describes aspects of the WCTP protocol that are common to all types of operations. Subsequent sections discuss the specific operations themselves.

3.1 XML Syntax

WCTP uses XML as the format to define its operations. WCTP operations must be formatted in accordance with the XML 1.0 specification [W3C-WD-xml]. Furthermore, these operations must be "well-formed" and “valid” according to the WCTP Data Type Definition. The meaning of well-formed and valid is described in the XML specifications referenced. If you are not familiar with XML, please familiarize yourself with XML before continuing to Section 4, WCTP Operations.

3.2 Date and Time Formats

This section describes the date and time format used by WCTP in all contexts where a parseable date is required. The format shown here is a selected profile or options from ISO8601:1988 (with Technical Corrigendum 1 applied), hereinafter referred to as ISO8601.

3.2.1 WCTP Date Format

The format for a date string in WCTP Date Format is:

`CCYY-MM-DD`

where CCYY is the four-digit year (century and year, as described in ISO8601), MM is a two-digit month number, DD is the two-digit ordinal number of the day within the calendar month, and the separator character is a "hyphen" ("-"). This format is the *Extended Format* described in ISO8601 Section 5.2.1.1 with the separator as described in ISO8601 Section 4.4. WCTP implementations must use this format for all date strings specified as being in WCTP Date Format.

Section 3.2.5 denotes the format string for specifying both date and time.

3.2.2 WCTP Time Format

The format for a time string in WCTP Time Format is:

`HH:MM:SS`

where HH is the two-digit hour in 24 hour notation ranging from 00 to 24 (this is not a typo), MM is the two-digit minute ranging from 00 to 59, SS is the two-digit seconds ranging from 00 to 59, and the separator character is a "colon" (:). This format is the Extended Format described in ISO8601 Section 5.3.1.1 with the separator as described in ISO8601 Section 4.4. WCTP implementations must use this format for all time strings specified as being in WCTP Time Format.

Note that midnight has two representations:

```
00:00:00
24:00:00
```

This is deliberate and in accordance with ISO8601 Section 5.3.2.

3.2.3 Subsecond Resolution

WCTP Time Format for representing subsecond granularity follows ISO8601 Section 5.3.1.3 and thus uses a "comma" (,) separator and an arbitrary number of digits representing the fraction down to the appropriate level of precision. Thus, the format for time with subsecond resolution is:

```
HH:MM:SS,S
```

where the "comma" (,) is a literal character (ISO8601 separator) and "s" after the comma is "to the right of the decimal mark" and indicates the subsecond value. The number of digits in the subsecond value, the precision of the subsecond value, and the ability of a given implementation to honor that precision are quality of implementation issues and are not specified by WCTP.

3.2.4 Time Zones

All times specified within WCTP must be specified using GMT (UTC). Implementations are expected to translate these times into the appropriate local time presentation format before utilizing the time in the manner in which it is being requested.

3.2.5 WCTP Date and Time Format

When date and time need to be specified in a single string, the WCTP Date and Time format is:

```
CCYY-MM-DDTHH:MM:SS,S
```

where "T" (uppercase T) is a literal character (ISO8601 designator). This format is the Extended Format of calendar date and time of day as described in ISO8601 Section 5.4.1 clause (a).

WCTP operations must not specify invalid combinations of fields such as February 31. WCTP implementations should reject invalid combinations of fields rather than trying to interpret them.

3.3 Status and Error Codes

WCTP standardizes all of the status and error codes returned in a WCTP Response operation. It utilizes the familiar Internet protocol paradigm of three-digit status values in responses to protocol operations. This paradigm was chosen because it is well understood and is suited to both machine-to-machine communication and human interpretation.

3.3.1 No HTTP Relationship

There is no relationship between the status codes in WCTP and the status codes at the HTTP transport level. As described above, HTTP is merely an instance of a transport mechanism for WCTP operations and any WCTP implementations utilizing HTTP must appropriately honor HTTP status or error conditions at the transport level. The semantics of completing the HTTP transport operation do not affect the semantics of the WCTP operations in any way.

Throughout the rest of this discussion, an HTTP status of "200 OK" (or "100 CONTINUE," if applicable) is implicit for the transport of the WCTP operations.

3.3.2 Status and Error Code Returned Text Strings

WCTP Response operations that return success or error codes are restricted to specifying one of the standard codes from Appendix C. The error and success responses may optionally return an error text or success text attribute. The text string returned in this attribute is implementation-specific, but must be consistent with the general description of the associated status or error code value (specified in Appendix C) that is required in status response operations. WCTP implementations must process the three-digit numeric value returned. WCTP implementations should not use the text string returned for any purposes other than as informational data with no defined semantics. A typical use for the text string is to report it to an error or call processing log file for later analysis by humans. WCTP implementations must not utilize the free-form body of the WCTP status or error response for any other purpose than as informational data with no defined semantics. Again, a typical use for the text string is to report it to an error or call processing log file for later analysis by humans. For error responses, the free-form data returned may assist in tracking down XML data encoding problems in WCTP implementations.

Error and success responses may optionally return an arbitrary text string within the error or success element. In the case of error responses, this text string could contain portions of the WCTP XML syntax where errors were found resulting in the inability to process the requested operation. For example, the following XML fragment could be returned when a date value is determined to be incorrect:

```
<wctp-Failure errorCode="404" errorText="Invalid date/time format">
      deliveryBefore="January 29, 1999" is not a valid WCTP date/time format
</wctp-Failure>
```

3.3.3 Categories of Status and Error Codes

The defined status and error codes are specified in Appendix C of this specification. Each code contains a three-digit numeric value and a general description of the status or error code.

Implementations of WCTP may treat unrecognized codes not defined in Appendix C of the WCTP specification or any 9xx series code in an implementation-specific manner.

The general categories into which a status or error definition may fall is:

- 2xx: "Success" Codes
- 3xx: "Protocol Violation" Error Codes
- 4xx: "Content Error" Codes
- 5xx: "Message Processing" Error Codes (Permanent Errors)
- 6xx: "Message Processing" Error Codes (Temporary Errors)
- 9xx: "Experimental" Codes

Implementations may experiment with new codes and new features by employing codes values in the 9xx series without fear of interfering with future versions of the WCTP specification. The 900 series codes will never be specified in an officially released WCTP specification. Presumably, once an experimental implementation is deemed to be operational, the developer of this new capability may submit a request to the appropriate WCTP Standards Committee to request that an official status or error code designation be allocated.

The set of error codes referred to as permanent errors are those error conditions that will not change if the WCTP operation is retried. The set of error codes referred to as temporary errors represent certain error conditions of a temporary nature. A re-submission of the WCTP operation at a later time may result in a successful operation.

3.4 WCTP Operation Declarations

A WCTP operation must begin with a suitable XML document type definition declaration such as:

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation
  SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
```

Any alternate form of the above XML declaration is acceptable so long as it meets the following criteria:

- The XML version must be 1.0.
- The DOCTYPE must be wctp-Operation.
- The SYSTEM keyword must appear, and must be followed by a valid URI that will return a WCTP/XML DTD that defines the version of the WCTP protocol being used to create the operation.

3.5 WCTP Protocol Version

Each wctp-Operation contains an attribute (wctpVersion) that specifies the name of the protocol version being used. Under WCTP 1.1, the only valid wctpVersion value is “WCTP-DTD-V1R1”. For future releases of WCTP, version names will be of the form WCTP-[SUBTYPE]-V<major release number>R<minor release number>. Version 2.0 is expected to implement SUBTYPEs other than DTD. Since the valid values for the wctpVersion attribute were not defined prior to this release, any unrecognized values will be treated as referring to WCTP version 1.0 (as though the user had entered “WCTP-DTD-V1R0”).

Carriers are allowed to implement protocol extensions to support operations that are not yet included in WCTP. These carrier-specific extensions must have version names of the format “<PROTOCOL>-<SUBTYPE>-V<X>R<Y>”. The <PROTOCOL> value may be any all-capital name **other than** WCTP, and the <SUBTYPE> may be any all-capital value so long as the total length of the version name is no more than 32 characters. The <X> and <Y> values refer to the major and minor version numbers as mentioned above.

The name of the WCTP Document Type Definition (DTD) file being used to define the operations appears in the SYSTEM specification as shown in Section 3.4 above. The version name may be reflected in the DTD filename. If it is, the filename will be of the form **wctp-dtd-v**<major release number>**r**<minor release number>**.dtd**. For example, wctp-dtd-v1r1.dtd is the official file name associated with this release of WCTP.

3.6 WCTP Tokens

Each wctp-Operation returned by a WCTP server can contain an optional attribute (wctpToken) specifying that server's "current" configuration status. Clients and proxies should examine these tokens to detect if the WCTP server has changed any aspects of its support (operations, configuration, or version information). As of this release, clients only read tokens and do not generate them. Beginning in release 2.0, this functionality will be expanded. As an example of using the wctpToken attribute, consider when a client receives a wctp-Confirmation with wctpToken="1AA" when he had previously received wctpToken="11AA". The client then knows that some aspect of the server's support has changed, and should perform a Version Query Operation (See Section 4.16) to determine what has changed.

4 WCTP Operations

4.1 WCTP Operations Overview

The WCTP protocol defines a number of "operations," each of which is represented as a “control block” in an XML form. The operations currently supported by the protocol are summarized below. The names of the various operations are of the form "*wctp-OperationName*."

4.1.1 Submit Message Operations

In general, a message contains any binary data. Alphanumeric messages in human readable form are a subset of this generalized format. Submission control blocks may also carry an indication of a particular form for the otherwise transparent binary information. For example, binary messages that follow the Motorola FLEXsuite™ defined form are tagged in a wctpTransparentData as type=”FLEXsuite”.

There are two types of message submission operations. One is used for Enterprise hosts (*wctp-SubmitRequest*) and the other for Transient clients (*wctp-SubmitClientMessage*). In either case, servers may impose a maximum submission rate on messages coming into the wireless network. If this rate is exceeded, an error response code is returned in lieu of accepting the message to indicate that the allowed input rate has been exceeded. If this rate is exceeded an excessive number of times, another error response code may be returned indicating that the sending system is subject to being disabled from sending further messages.

4.1.2 Status Operations

Whenever data is moved between a client and a server, a first level status operation must be returned. This status indicates if the message has been successfully or unsuccessfully received so that the sender knows that the data can safely be removed from its queue. The *wctp-Confirmation* operation is used for this purpose in response to *wctpSubmitRequest* operations. *wctp-SubmitClientResponse* is returned in response to a *wctp-SubmitClientMessage*.

Message status information may be sent upon the detection of a change of state of a message by using *wctp-StatusInfo*. This status could be in the form of a notification of the message processing state (the message has been queued or delivered, for example) if such notifications were requested in the *wctp-SubmitRequest* when the message was first sent. The status could also be in the form of a failure indication with an explanation, indicating that the message was undeliverable for any of a number of reasons. This status may be returned quickly if, for example, it was determined that the destination ID is invalid (some implementations may not be able to check validity at the point of entry into the network and may instead perform this validation step after the message goes through the first level of acceptance). The failure status could also, however, come much later if, for example, the message could not be delivered within a time interval specified.

4.1.3 Message Reply Operation

A reply contains a message generated in response to a previously submitted message. WCTP provides threading information to tie such a response (*wctp-MessageReply*) to the original message. More than one response can potentially be returned for each message submitted.

4.1.4 Poll Operations

Polling is used by an Enterprise host when messages may not be automatically pushed to it for security reasons or because of the lack of an HTTP server. A Poll operation (*wctp-PollForMessages*) is used by an Enterprise Host to request and potentially receive (via *wctp-PollResponse*) messages (*wctp-StatusInfo*, *wctp-MessageReply*, *wctp-SubmitRequest*, and *wctp-LookupResponse* operations). These may be notifications or replies for messages which were previously submitted, or they may be unsolicited messages to be processed by the Enterprise host.

In order to speed up the transfer of messages queued at the wireless network, a Poll Response operation (*wctp-PollResponse*) may return multiple messages (status responses, replies, delivery notifications or unsolicited messages). The system responding to the polling operation may limit the maximum number of messages that may be returned should a large number be queued for delivery. The maximum number of messages returned in a single Poll Response is implementation specific. The maximum number actually returned is the smaller of the number of messages queued, the implementation specific maximum number returnable and the maximum number specified with the maxMessagesInBatch attribute in the *wctp-PollForMessages*.

A sequence number is assigned to each *wctp-Message* retrieved in a *wctp-PollResponse*. Each sequence number received must be returned in a *wctp-MessageaReceived* element in a subsequent Poll operation to acknowledge its successful retrieval. The *wctp-PollForMessages* allows multiple sequence numbers to be specified so that multiple messages may be acknowledged at once. If a sequence number is not acknowledged in the next *wctp-PollForMessages*, the message will be retrieved again.

When there are no further messages queued at the wireless network, a *wctp-NoMessages* is returned inside of the *wctp-PollResponse*.

The wireless network may impose implementation-specific maximum polling rates: one rate when there are no further queued messages and another rate when messages are returned. System error response codes are returned in the *wctp-PollResponse* to indicate when a polling rate has been exceeded or when polling rates have been excessively exceeded. If excessive polling continues, the wireless network may permanently disable the polling system.

As an example, the diagram below shows the pushing of an Failure, Queued and Delivered Status, a Reply and an unsolicited message to the Enterprise host. If the Enterprise host is not capable of supporting the HTTP protocol to receive this pushed information, it would have been registered with the wireless network as supporting Polling to receive these messages. Periodically, the Enterprise host would issue a Poll to check for messages . The sequence of operations by the Enterprise could have been as follows:

Poll	→	
	←	No Messages
Poll	→	
	←	Failure Status (sequence # 15)
Poll (AK sequence # 15)	→	
	←	No Messages
Poll	→	
	←	Queued Status (sequence # 16)
	←	Delivered Status (sequence # 17)
Poll (AK sequence # 16, 17)	→	
	←	No Messages
Poll	→	
	←	Reply (sequence # 18)
	←	Submit (sequence # 19)
Poll (AK sequence # 18)	→	
	←	Submit (sequence # 19)
Poll (AK sequence # 19)	→	
	←	No Messages

When polling is being used to receive queued messages from the wireless network, the poll operations are also used to acknowledge the positive receipt of the messages returned by the wireless network. If this acknowledgement is not returned with the subsequent poll, non-confirmed messages will be returned again.

Subsequent Poll operations may be submitted by an Enterprise Host immediately after the receipt of the messages from the wireless network, unless the wireless network sent No Messages indication or explicitly specified the next poll interval.

4.1.5 Query Operations

A Client Query operation (*wctp-ClientQuery*) is a form of a poll operation used by Transient clients to check for status information or responses to a previously submitted message. The response to a Client Query operation (*wctp-ClientQueryResponse*) can indicate that no information is available, or it may return a collection of replies (*wctp-ClientMessageReply*) and message statuses (*wctp-ClientStatusInfo)*.

A Subscriber Lookup operation (*wctp-LookupSubscriber*) is used by Enterprise hosts to query the WCTP server about the capabilities of a wireless device. The response to a Subscriber Lookup operation (*wctp-LookupResponse*) specifies a set of attributes describing capabilities of the device. Such information can be used to appropriately format the subsequent messages.

Both Transient clients and Enterprise hosts use the version query operation (*wctp-VersionQuery*) to query the WCTP server about what version(s) of the WCTP it supports. The response to a version query operation (*wctp-VersionResponse*) indicates the WCTP version the server supports and may include a token that a client can use to monitor changes in the server's support.

4.2 Valid Responses

The operations specified in the previous section represent both request and response operations. The following table shows the valid response for each request. The HOST/GW column identifies whether the request can be submitted by a carrier (GW only), a transient/enterprise client (HOST only), or both (HOST or GW).

Request	**Submitted By (HOST/GW)**	**Valid Responses**
wctp-ClientQuery	HOST only	wctp-ClientQueryResponse
wctp-LookupSubscriber	HOST only	wctp-Confirmation
wctp-LookupResponse	POST only	wctp-Confirmation
wctp-MessageReply	HOST or GW	wctp-Confirmation
wctp-PollForMessages	HOST only	wctp-PollResponse
wctp-StatusInfo	GW only	wctp-Confirmation
wctp-SubmitClientMessage	HOST only	wctp-SubmitClientResponse
wctp-SubmitRequest	HOST or GW	wctp-Confirmation
wctp-VersionQuery	HOST only	wctp-VersionResponse
Any request containing recognizable XML but un-parseable WCTP	HOST or GW	wctp-Failure

4.3 Call Flow Between Network Elements

The operations specified above are used to convey information between the wireline and wireless network and possibly between elements within the wireless network itself. shows several call flows between the Enterprise Host and other network elements. The diagram is assuming that messages from the wireless network that need to be returned to the Enterprise host are being pushed directly to the host using the recommended HTTP POST methodology.

The call flows are read from the top of the diagram down showing the types of operations that may be moving between elements in normal and error cases. The diagram also shows cases in which the user of a wireless device is responding to a specific message as well as a case in which an unsolicited message is generated from the wireless device. It should be pointed out that some wireless two-way technologies support the direct association of a message response against a specific message while some other wireless devices may send such responses as new, seemingly unsolicited messages that are not tagged to the original message. Where the device itself is not capable of tagging subsequent responses, the method of associating transmissions and responses is outside of the scope of the WCTP protocol.

helps to point out several items for the WCTP application developer on the wireline side as well as to the infrastructure equipment suppliers on the wireless network side. The definitions of the WCTP control blocks (operations) themselves do not make it apparent how a call flow may proceed. In fact, the sequence of operations that may occur could be dependent upon the nature of the transport protocol being utilized as explained in the following paragraphs.

Referring to the figure, it is shown that when an operation is submitted (set #1), it is possible to get an XML formatted error response posted back. The Submit operation does carry optional parameters (referred to as *notifyWhen* type parameters) that indicate the requirement to post back one or more delivery status conditions to the message submitter. When the requested status condition occurs, it will be queued for return to the submitter. The "Acknowledged" status shown in the second Submit in the call flow (set #2) is not one that it optionally requested. This status return is provided for implementations in which an error or status response may not be immediately available to return to the submitter, but the transport protocol being used requires a return response before closing the connection. This is the case of the submit/response HTTP protocol that is expected to be used as the primary WCTP transport protocol. The second Submit operation shown in the figure (as set #2) is assumed to be operating over an HTTP type connection. Although the Submit request received was accepted by the Internet Gateway element of the wireless network, the diagram is attempting to show the generalized case in which the operation may be determined to be un-executable or in error at some later point in its processing (such as when it is processed by the WCG element within the wireless network). That determination may not be able to be made in a "timely" fashion and it may not be desirable to leave HTTP transport connections open for longer than they need to be. The "Acknowledged" response (success code 200 of Appendix C) is returned via *wctp-Confirmation* in order to acknowledge the receipt of a valid WCTP operation that it may or may not be able to successfully complete.

From the point of view of the creator of a WCTP application, it is only necessary to understand that an error response to a Submitted request may be returned to the application some time in the future. A received response status is only an intermediate response condition and does not necessarily mean that the operation was acceptable for execution.

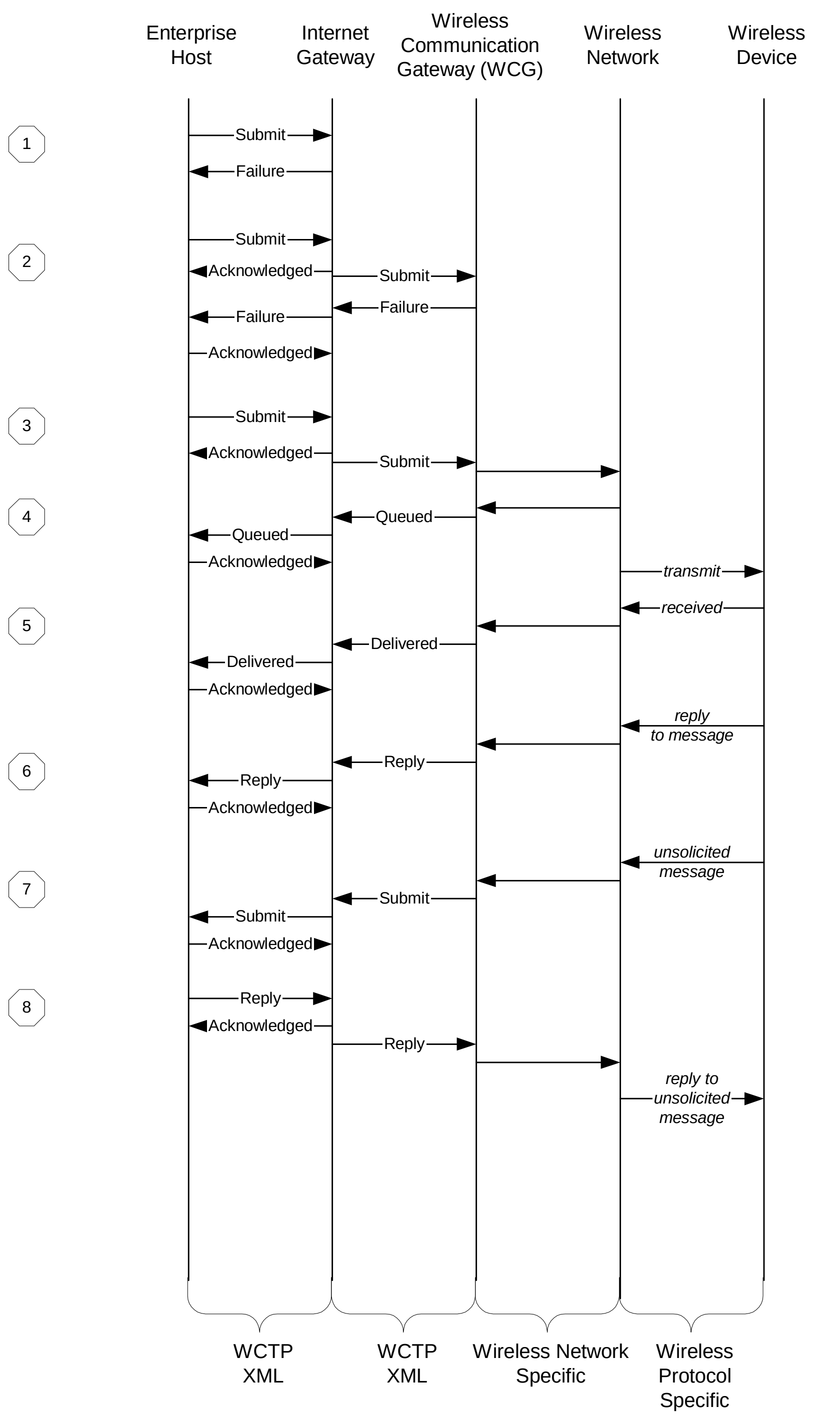


Figure 4-1 Call Flow Between Network Components

The third Submit operation in the figure (set #3) shows where a valid Submit operation was presented. When the message being submitted is validated, properly queued and safe-stored for delivery within a network specific element of the wireless Network, a "Message Queued" delivery status may be returned to the message sender (set #4 of the figure) to indicate that the requested operation is pending. The diagram shows the Queued Status being pushed to the

Enterprise host through the HTTP protocol. As required by HTTP, a response must be sent to the pushed Queued status. That response is sent as a WCTP Acknowledgement operation (*wctp-Confirmation*) that confirms the receipt of the pushed data.

The example shows the pushing of a Delivered Status notification (set #5) upon the successful delivery of the message to the wireless device because the Submit operation had requested a "*notifyWhenDelivered*" status report. When the user of the wireless device replies to the received message, a WCTP MessageReply type operation (set #6) is pushed to the Enterprise Host.

The example also shows the case where a wireless device initiates an unsolicited message (set #7). This type of message is pushed to the Enterprise Host as a WCTP Submit operation initiated by the wireless network. The Enterprise host may send a response to the unsolicited message (set #8) by initiating a MessageReply operation against the messageID assigned by the wireless network.

4.4 WCTP Submit Request Operation

The Submit Request operation of WCTP (*wctp-SubmitRequest*) is used to initiate a new message from the Wireline Enterprise Host or from the Wireless system. Figure 4-2 shows the major XML formatted elements of the Submit request. Each of these elements may specify a number of different attributes or parameters which are associated with the element.

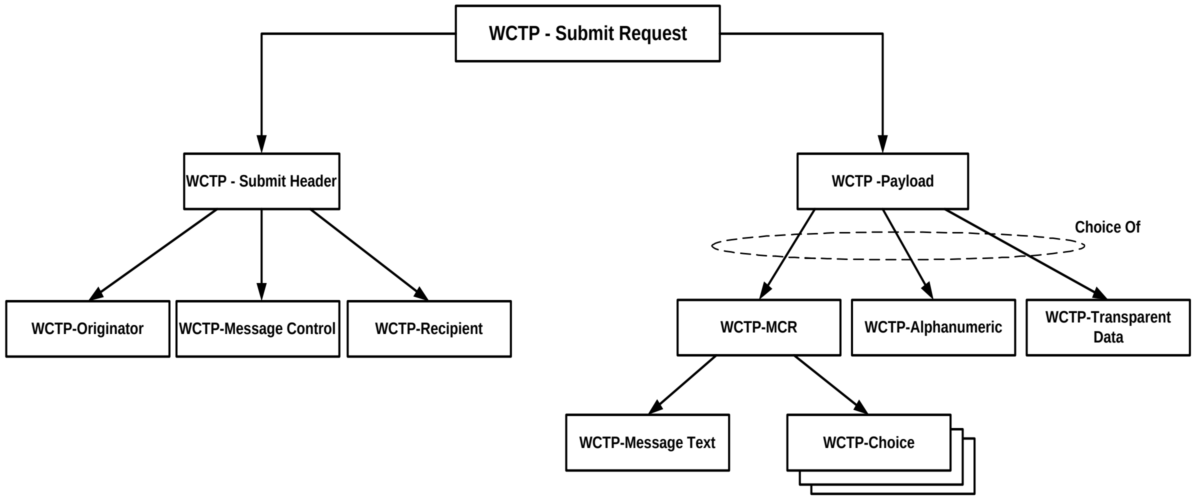


Figure 4-2 WCTP Submit Request Operation

The Submit request has two major elements, the Submit Header and the Payload.

4.4.1 Message Payload

The Payload contains the message content that is to be delivered. This content is contained within a structure that is dependent upon the data type being carried. General binary type messages, referred to as Transparent Data, may be further tagged as to a specialized form of the binary data such as FLEXsuite. XML provides a standard canonical form which can be used to specify numeric, alphanumeric and some 8-bit data. The 8-bit binary data can be represented as an ASCII string that encodes a hexadecimal value (e.g. &#xD1; represents the binary octet 11010001). However, not all octets can be encoded in XML. In particular, most of the “special characters” with hexadecimal values 0x00 to 0x1F cannot be encoded in XML and are considered illegal characters. Refer to the referenced XML documentation for further details.

For transfer of arbitrary binary data WCTP supports *base64* (RFC 1341) encoding. Specification of the encoding method is made via the *encoding* attribute with base64 representation being a default.

Multiple Choice Messages (MCR) may be specified through the Submit Message Request. An MCR message is a menu of selections. The current support of MCR in WCTP allows the wireless user or application on the wireless device to specify a reply that selects one item from the list of choices received. A subsequent Message Reply tagged back to the original message will contain the actual text selected. Since it is the actual text returned as opposed to the index of the MCR selection, the list of choices should contain unique strings of text. If the same text appears within multiple selections, the information provided in the MCR response does not provide sufficient information to know specifically which index of the MCR message was selected. If this information is required, the list of choices must contain unique strings of text. The support and implementation of MCR messages by a wireless device or application is device or application specific.

The MCR response message does permit responses to be returned that are not members of the list of choices. This permits an application to provide a list of "suggested" responses while still permitting the wireless device to return its own specific response. Some wireless devices, such as those that integrate ReFLEX™ technology, provide an over-the-air optimized methodology for supporting this capability. Other wireless devices could implement the MCR type capability through special application code operating in the wireless device or a computer directly connected to the two-way wireless transceiver. If specialized applications code is used to support the feature, the Wireless network processing the WCTP requests must be aware of manner in which the capability is implemented in order to properly activate the MCR function of WCTP. The current release of WCTP does not support the return of multiple selections at one time.

4.4.2 Submit Header

The message header provides information as to the message originator, message recipient, and many message control parameters. One of the key parameters is the Message ID. This is a unique message identifier assigned by the message originator. This identifier will be contained in any subsequent responses returned on behalf of the submitted message.

Other Message Control parameters allow the message sender to control when a message is to be delivered and what types of status notifications should be reported. It is also possible to direct responses to a different Enterprise host or application than the sender of the original message.

A transaction ID may be submitted with the message. If such an ID is included, it will be returned in subsequent responses. A transaction ID is used by the message originator to tag a group of individually submitted requests. These requests might even be sent to a number of different devices that may even span several different wireless networks. The transaction ID may be used to indicate that all of these requests where part of a particular instance of an application on behalf of a sequence of operations that the application may have been performing.

4.5 WCTP Message Reply Operation

The Message Reply operation of WCTP (*wctp-MessageReply*) is used to report message replies returned on behalf of other messages. This includes message replies returned on behalf of yet other message replies. Figure 4- shows the major XML elements of the Message Reply operation. Each of these elements may specify a number of different attributes or parameters, which are associated with the element. The Message Reply contains two main elements: the Response Header and the Payload. Specification of the message payload follows the same rules as listed in Section 4.5.1.

4.5.1 Response Header

The Response Header provides information with regard to the message originator, the message recipient, and control information to further specify how the response should be processed. The Originator portion of the Response Header indicates who is initiating the response. In the case of a response from a wireless device, this would specify the unique identification code of the device as specified by the wireless network on which this device operates.

The Recipient information of the Response Header would normally contain the address of the sender of the original message. If, when the message was submitted, it was indicated that responses should be sent to a different destination other than the message originator, this destination information will appear in the *wctp-Recipient* instead.

The Message Control element of the Response Header contains the unique Message Identifier (responseToMessageID) that ties the response back to a particular submitted message. The structure of the Message Control element is identical to its namesake from the Submit Header of the Submit Message Request operation (see Section 4.4.2).

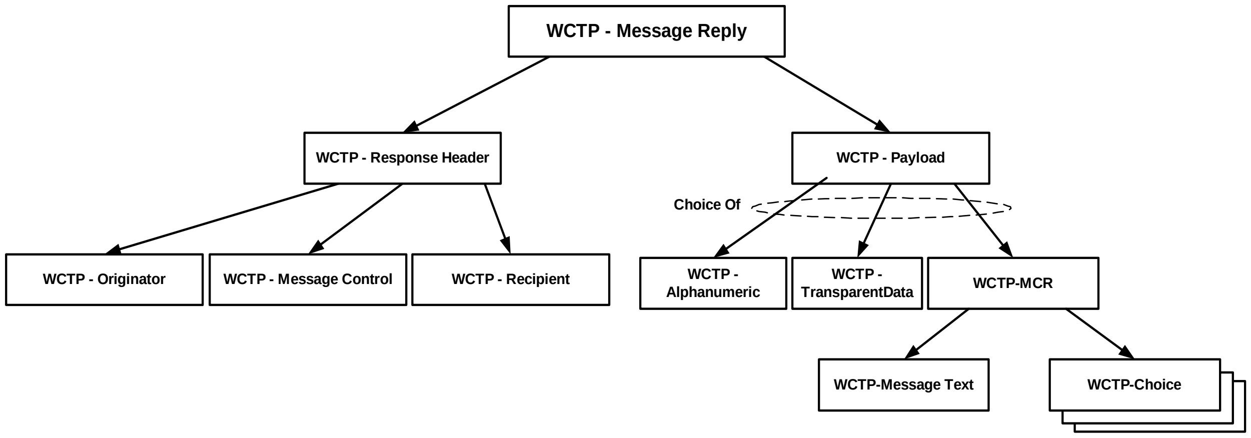


Figure 4-3 WCTP Message Reply Operation

4.6 WCTP Status Info Operation

The Status Info operation of WCTP (*wctp-StatusInfo*) is used to report both message delivery failures as well as delivery notifications returned on behalf of other messages. Figure 4- shows the major XML elements of the Status Info operation. Each of these elements specifies a number of different attributes or parameters. The Status Info contains two main elements: the Response Header and either a Failure or a Notification.

The Failure response may indicate a permanent or temporary failure of message delivery. The specifics of the failure condition are reported via a standardized error code and optional error text.

The Notification response may indicate a message delivery status condition such as "message accepted by the wireless device" or "message read or processed by the wireless device." The types of notification status responses desired must be specified when the message is created.

The Response Header provides information with regard to the message originator, the message recipient, and control information to further specify how the response should be processed. Specification of the Response Header follows the rules as listed in the Section 4.5.1.

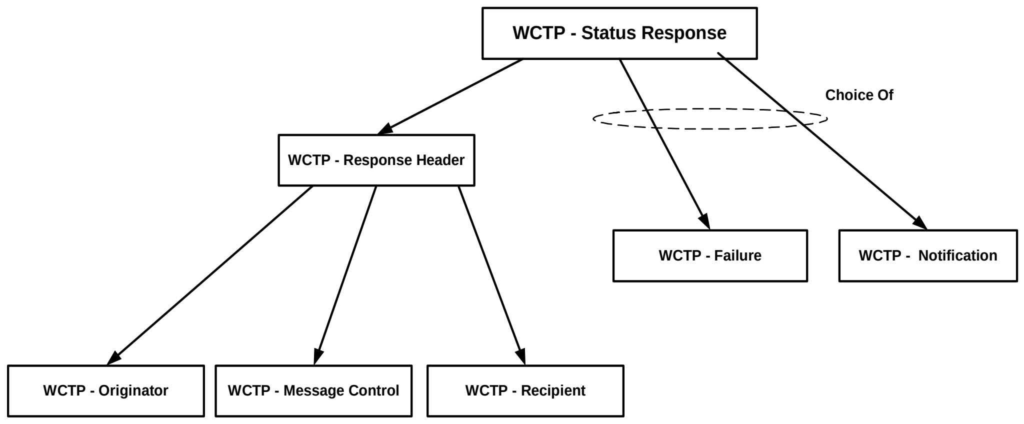


Figure 4-4 WCTP Status Info Operation

4.7 WCTP Confirmation Operation

The Confirmation operation of WCTP (*wctp-Confirmation*) is used to report the success or failure of the submission of a request for processing. Figure 4- shows the major XML elements of the Confirmation operation. Each of these elements specifies a number of different attributes or parameters associated with it.

The Confirmation is returned for each HTTP POST, and conveys positive or negative acknowledgement that the message was received for delivery.

Structurally, the Confirmation operation contains either the Success or the Failure element. *Note: WCTP-Success is not an indication that the message has been delivered, rather it indicates the gateway has accepted the message for delivery.*

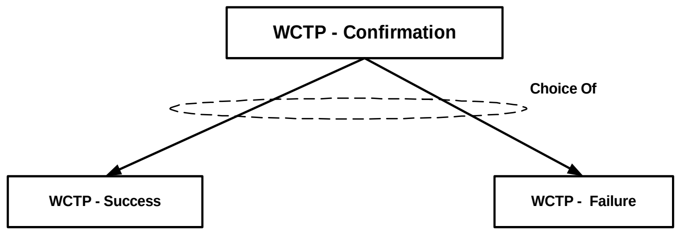


Figure 4-5 WCTP Confirmation Operation

4.8 WCTP Poll For Messages Operation

The Poll For Messages operation (*wctp-PollForMessages*) is used by wireline systems that are not able to accept pushed data from the wireless network and have to use the Polling mechanism to collect messages directed to them. This “pseudo-push” mechanism was discussed in Section 2.2.3. The Poll For Messages operation is initiated by a wireline system to solicit messages accumulated for it by a WCTP server of a wireless carrier. The response to a Poll For Messages operation, the Poll Response operation, may return a batch of messages, each identified by an assigned sequence number. The following Poll For Message operation must acknowledge the receipt of messages received in the prior poll by providing a Message Received element with the sequence number for each received message.

The sequence of Poll For Messages/Poll Response operations is repeated until the Poll Response returns an indication that there are no more messages to deliver (*wctp-NoMessages*).

To limit the number of messages being delivered, the Poll For Messages operations may specify the maximum number of messages to be returned in one Poll Response.

4.9 WCTP Poll Response Operation

The Poll Response operation (*wctp-PollResponse*) is sent by a WCTP server of a wireless carrier to a wireline system in a response to the Poll For Messages operation.

Each Poll Response operation may contain a batch of messages destined to the wireline system that have been queued on its behalf by a WCTP server of a wireless carrier. Each message in a batch is assigned a sequence number. Receipt of the messages delivered via Poll Response is acknowledged based upon this sequence number with the following Poll For Messages operation (see Section 4.8).

4.10 WCTP Lookup Subscriber Operation

The Lookup Subscriber operation of WCTP (*wctp-LookupSubscriber*) is used by an Enterprise host to ask a carrier’s WCTP server about the capabilities of a subscriber’s messaging device.

Typically, an Enterprise host would do a Lookup Subscriber operation to get information about a specific subscriber’s messaging device that the Enterprise Host could use when formulating a Submit Request operation for that subscriber.

4.11 WCTP Lookup Response Operation

The Lookup Response operation of WCTP (*wctp-LookupResponse*) is sent by a WCTP server of a wireless carrier to an Enterprise host in a response to the Lookup Subscriber operation.

Typically an Enterprise host would use the information about a specific subscriber’s messaging device to formulate a viable set of attributes for a Submit Request operation for that subscriber.

4.12 WCTP Submit Client Message Operation

The Submit Client Message operation of WCTP (*wctp-SubmitClientMessage*) is used to initiate a new message from a Transient client to the WCTP server of a wireless system. Figure 4- shows the major XML elements of the Submit Client Message operation. Each of these elements specifies a number of different attributes or parameters associated with it.

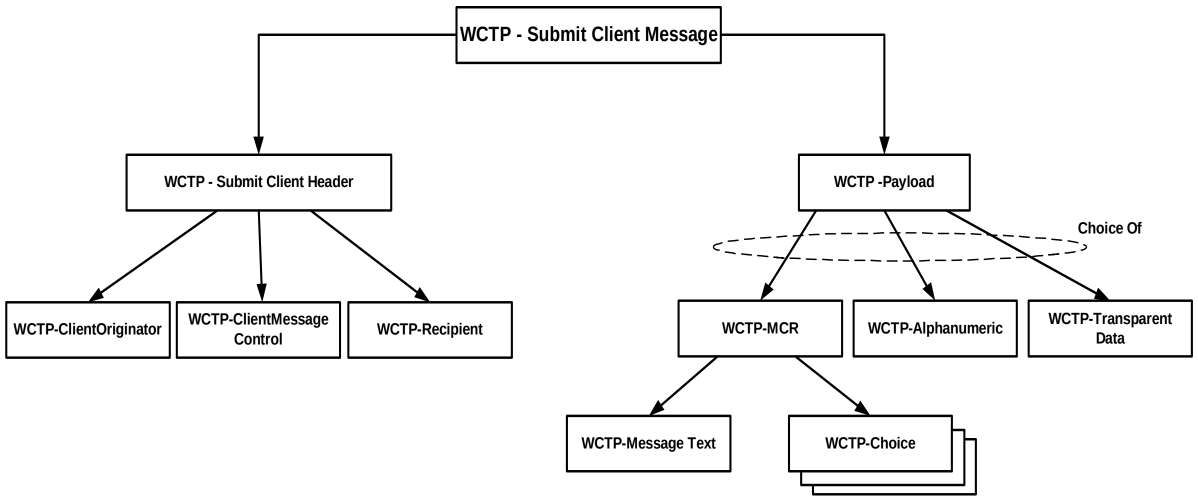


Figure 4-6 WCTP Submit Client Message Operation

The Submit Client Message has two major elements: the Submit Client Header and the Payload.

The Submit Client Header element is analogous to the Submit Header described in Section 4.4.2. However, it is tailored for use by Transient clients. In particular, there is no requirement for the sender ID to be unique. Also, there is no message ID supplied with this operation.

The structure of the Payload element is described in the Section 4.4.1.

4.13 WCTP Submit Client Response Operation

The Submit Client Response operation (*wctp-SubmitClientResponse*) is sent by a WCTP server of a wireless carrier to a Transient client in a response to the Submit Client Message operation.

The Submit Client Response operation provides positive or negative acknowledgement of the submission of the message. In the case of positive acknowledgement the WCTP server provides the message sender with a tracking number assigned to the message. This tracking number, in combination with the sender ID and the recipient ID of the message, may be subsequently used to check for replies and status notifications through a Client Query Operation (See Section 4.14).

4.14 WCTP Client Query Operation

The Client Query operation (*wctp-ClientQuery*) is used by a Transient client to query a WCTP server of a wireless carrier for replies and status notifications to the previously submitted message.

To perform the Query Client Replies operation the sender ID, the recipient ID and the tracking number of the original message must be supplied.

4.15 WCTP Client Query Response Operation

The Client Query Response operation (*wctp-ClientQueryResponse*) is sent by a WCTP server of a wireless carrier to a Transient client in response to a Client Query operation. Figure 4- shows the major XML elements of the Submit Client Message operation.

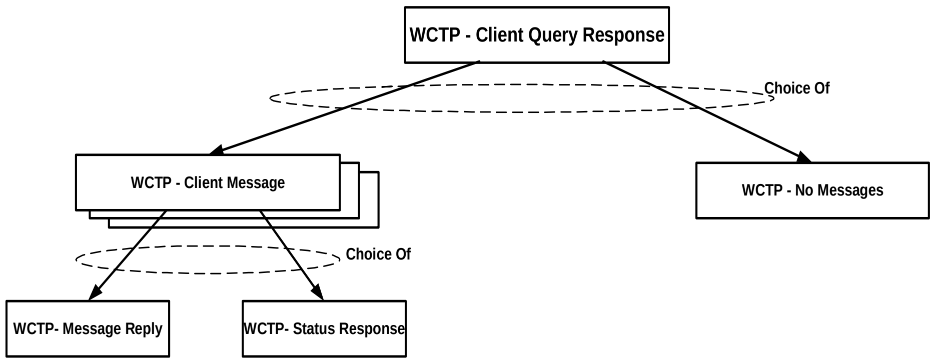


Figure 4-7 WCTP Client Query Response Operation

The Client Query Response operation provides the requestor with a set of Client Messages. Each Client Message contains either a Message Reply or a Status Info to the message identified by a combination of the sender ID, the recipient ID and the tracking number as specified in the Client Query request.

Alternatively, the No Messages element is used to indicate to the requestor that there are no replies or status information for the original message.

Note: each time the Client Query operation is submitted, the Client Query Response operation will return a set of replies and status notifications with the respect to the original message. WCTP servers will hold this information for an implementation-specific period of time.

4.16 WCTP Version Query Operation

As the WCTP protocol grows and evolves, there is a need for clients (applications requesting a WCTP service) and servers (applications providing a WCTP service) to negotiate with each other in order to establish a set of common requests and responses.

In the standard WCTP model, each WCTP gateway has a server application (that receives requests from outside clients) and a client application (that makes requests against other servers). In each of these relationships, the client needs to know what version of the WCTP the server supports before making a request against it.

In both cases, a version query (a request for information about which version(s) of the WCTP protocol is/are supported) lodged against a server should provide the client with information about the DTDs supported by the server. It is then the client's responsibility to use this information to create WCTP requests that are compatible with the server's capabilities.

The Version Query request (wctp-VersionQuery) allows clients to request information about the DTDs that a server application currently supports. In it, an inquirer identifies himself and optionally provides a time-stamp for his request.

The response to a Version Query request is a Version Response (wctp-VersionResponse) as shown below. If, however, the Version Query received by the server cannot be parsed (invalid XML characters, for example), the response will be a generic Failure.

Operation	Response Operation	Comments
VersionQuery	VersionResponse	Version support information is requested and supplied for the currently supported DTDs.
	Failure	See error table.

Table 4-1. Valid Response Operations

The requester can use the inquirer attribute to identify himself to the server. This value is for informational purposes only, and is not required.

The user can provide a date/time stamp with the query in the dateTime field. If this value is provided, it must be returned by the server in the dateTimeOfReq field in the wctp-VersionResponse element. This value is the key for connecting the response with the original request.

The listDTDs attribute is not used in version 1.1. It is placed here in preparation for WCTP version 2.0.

The listConfiguration attribute is not used in version 1.1. It is placed here in preparation for WCTP version 2.0.

4.17 Version Response Operation

The Version Response (wctp-VersionResponse) is sent to users in response to a wctp-VersionQuery request. It is not an initiating operation (a request). It may contain the following WCTP errors (see the table below). For details on errors in the range from 300 to 899, see Appendix C.

Code	Description	Code	Description
0	Undefined Error	302	XML Validation Error
200	Success	303	Version Error
300	Operation Not Supported	400	Function Not Supported
301	Cannot Parse Input		

Table 4-2. Error Codes in Failure Response

The value submitted in the inquirer attribute of the original wctp-VersionQuery will be returned here in the inquirer attribute. This attribute is provided as a key for connecting this response to the original request.

If the requester provided a timestamp in the dateTime attribute of the original wctp-VersionQuery, that value will be returned here in the dateTimeOfReq attribute. This attribute is provided as an optional key for connecting this response to the original request.

The server must return information in the responder attribute identifying the URI/URL against which this version information is valid. It should be noted that this URI may be different from that against which the wctp-VersionQuery was actually submitted.

The server may also provide a date/time stamp for the response in the dateTimeOfRsp field. If this value is provided, it is for informational purposes only. Similar to the inquirer's date-time stamp, this can be used for filing away responder capability information.

The server may optionally return a date/time value in the invalidAfter attribute. This date/time value represents the time until which the requester may rely on the response. The responder may alter this date-time at will, however, and different inquirers may obtain different time-stamps for the exact same capabilities set. If there are any doubts as to the validity of the version information, the requester should refresh by again requesting version information via a wctp-VersionQuery after the invalidAfter date-time.

The listDTDs attribute specifies whether or not the user requested a list of all of the DTDs supported by the server. This field is not used in version 1.1, and is placed here in preparation for WCTP version 2.0.

The listConfiguration attribute specifies whether or not the user requested configuration information for this server. This field is not used in version 1.1, and is placed here in preparation for WCTP version 2.0.

4.17.1 Contact Information

The wctp-VersionResponse may contain an optional wctp-ContactInfo element, providing contact information through which the inquirer can obtain assistance and additional information about the server.

The email attribute is intended to provide a valid SMTP e-mail address. Any valid messaging address may be supplied. However, note that if contact is for technical problems, it may be advisable not to the very messaging system that has a problem.

The phone attribute is intended to provide a voice telephone number. Only valid Hayes-type telephone characters may appear: left- and right-parentheses, space, hyphens and commas. No Hayes-type alpha commands, such as ATDT, etc., are allowed. An 'x' may appear at the end of the telephone number to indicate that a phone extension of one or more numeric digits follows. If 'x' is used, then alpha characters may follow up to the limit of the field. The following are valid examples for phone

“8005552113”

“1-800-555-2368”

“1(800)555-2368”

“(800) 555-2368”

"8005436789x5128"

The www attribute is intended to provide a URL or URI identifying a valid web address that may be used to find additional information about the WCTP support offered by the responder.

4.17.2 Failure

The wctp-VersionResponse may contain a wctp-Failure element if the version request fails.

4.17.3 DTDSupport

The wctp-VersionResponse will contain at least one wctp-DTDSupport element if the version request succeeds. A wctp-DTDSupport element provides support information about the DTD identified by the name that it contains. Each wctp-DTDSupport element returned in the wctp-VersionResponse provides version information about the specified DTD as it is implemented in the server/responder.

The DTD name attribute should have a format consistent with the version names defined in Section 3.5.

The verToken attribute, if provided, is an arbitrary string representing the "current" token for version information, and may be cached long-term. This value must change whenever the specified DTD support has been modified (a new feature or operation is added, a feature or function is dropped, etc.). Any change in this value (between this point and subsequent retrievals) indicates that there has been some change in the overall support for the WCTP protocol-level features or DTDs, and the submitting application must initiate a wctp-VersionQuery to determine the exact nature of the changes. Care should be exercised to assure that no token is ever re-used with a different meaning. For more information, see Section 3.6.

The exceptions attribute defines whether the specified supported DTD has exceptions (when some aspect of the DTD is unsupported or only partially supported) or not. If the exceptions attribute is Yes, then there is at least one feature of the DTD that is not fully supported. In WCTP version 1.1, any additional information about these exceptions must be obtained through the resources identified in one of the contact addreses in the wctp-ContactInfo element. In version 2.0, additional information about the exceptions will be returned in the wctp-VersionResponse.

The supportType attribute defines the type of support (if any) that is available for the specified DTD. Valid values are:

Supported	operations may be submitted under this DTD [see the exceptions attribute]
Deprecated	operations may be submitted under this DTD, but the use of this DTD is discouraged (an alternative may be available, and this DTD may be moved to NotSupported at any time)
NotSupported	operations submitted under this DTD will be rejected.

DTDs that are deprecated may also have the optional supportUntil and replacement attributes set.   supportUntil provides a date/time after which the DTD will no longer be supported, if one is known. The replacement attribute contains the name of another (more recent) DTD that should be used in place of the deprecated one.

Appendix A - WCTP Document Type Definition

```
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- file-url='http://dtd.wctp.org/wctpv1-1.dtd' -->
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->

<!-- Wireless Communication Transfer Protocol (WCTP)                 -->

<!-- ============================= -->
<!-- Declaration of wctp-Operation -->
<!-- ============================= -->
<!ELEMENT wctp-Operation          ( wctp-ClientQuery
                                  | wctp-ClientQueryResponse
                                  | wctp-Confirmation
                                  | wctp-LookupSubscriber
                                  | wctp-LookupResponse
                                  | wctp-MessageReply
                                  | wctp-PollForMessages
                                  | wctp-PollResponse
                                  | wctp-StatusInfo
                                  | wctp-SubmitClientMessage
                                  | wctp-SubmitClientResponse
                                  | wctp-SubmitRequest
                                  | wctp-VersionQuery
                                  | wctp-VersionResponse
                                  ) >
<!ATTLIST wctp-Operation
          wctpVersion             CDATA  #REQUIRED
          wctpToken               CDATA  #IMPLIED
                                  >

<!-- =============================== -->
<!-- Declaration of wctp-ClientQuery -->
<!-- =============================== -->

<!ELEMENT wctp-ClientQuery          EMPTY
                                    >
<!ATTLIST wctp-ClientQuery
          senderID                  CDATA  #REQUIRED
          recipientID               CDATA  #REQUIRED
          trackingNumber            CDATA  #REQUIRED
                                    >

<!-- ======================================= -->
<!-- Declaration of wctp-ClientQueryResponse -->
<!-- ======================================= -->

<!ELEMENT wctp-ClientQueryResponse  ( wctp-ClientMessage+
                                    | wctp-NoMessages
                                    | wctp-Failure
                                    ) >
<!ATTLIST wctp-ClientQueryResponse
          minNextPollInterval       CDATA  #IMPLIED
                                    >

<!ELEMENT wctp-ClientMessage        ( wctp-ClientMessageReply
                                    | wctp-ClientStatusInfo
                                    ) >

<!ELEMENT wctp-ClientMessageReply   ( wctp-ClientResponseHeader
                                    , wctp-Payload
                                    ) >

<!ELEMENT wctp-ClientStatusInfo     ( wctp-ClientResponseHeader
                                    , ( wctp-Failure
                                      | wctp-Notification
                                      )
                                    ) >

<!ELEMENT wctp-ClientResponseHeader ( wctp-Originator
                                    , wctp-Recipient
                                    ) >
<!ATTLIST wctp-ClientResponseHeader
          responseTimestamp         CDATA  #IMPLIED
          respondingToTimestamp     CDATA  #IMPLIED
                                    >


<!-- ================================  -->
<!-- Declaration of wctp-Confirmation  -->
<!-- ================================  -->

<!ELEMENT wctp-Confirmation       ( wctp-Failure
                                  | wctp-Success
                                  ) >

<!ELEMENT wctp-Success            (#PCDATA)
                                  >
<!ATTLIST wctp-Success
          successCode             CDATA  #REQUIRED
          successText             CDATA  #IMPLIED
                                  >

<!-- ==================================== -->
<!-- Declaration of wctp-LookupSubscriber -->
<!-- ==================================== -->

<!ELEMENT wctp-LookupSubscriber     ( wctp-Originator
                                    , wctp-LookupMessageControl
                                    , wctp-Recipient
                                    ) >

<!ELEMENT wctp-LookupMessageControl EMPTY
                                    >
<!ATTLIST wctp-LookupMessageControl
          messageID                 CDATA  #REQUIRED
          transactionID             CDATA  #IMPLIED
          sendResponsesToID         CDATA  #IMPLIED
                                    >

<!-- ================================== -->
<!-- Declaration of wctp-LookupResponse -->
<!-- ================================== -->
```

```
<!ELEMENT wctp-LookupResponse       ( wctp-Originator
                                    , wctp-Recipient
                                    , ( wctp-LookupData
                                      | wctp-Failure
                                      )
                                    ) >
<!ATTLIST wctp-LookupResponse
          responseToMessageID       CDATA  #REQUIRED
          transactionID             CDATA  #IMPLIED
                                    >

<!ELEMENT wctp-LookupData           EMPTY
                                    >
<!ATTLIST wctp-LookupData
          maxMessageLength          CDATA  #REQUIRED
          mcrSupported              ( true | false )  #IMPLIED
          canRespond                ( true | false )  #IMPLIED
                                    >

<!-- ======================================  -->
<!-- Declaration of wctp-MessageReply        -->
<!-- ======================================  -->

<!ELEMENT wctp-MessageReply       ( wctp-ResponseHeader
                                  , wctp-Payload
                                  ) >

<!ELEMENT wctp-ResponseHeader     ( wctp-Originator
                                  , wctp-MessageControl
                                  , wctp-Recipient
                                  ) >
<!ATTLIST wctp-ResponseHeader
          responseToMessageID     CDATA  #REQUIRED
          responseTimestamp       CDATA  #IMPLIED
          respondingToTimestamp   CDATA  #IMPLIED
          onBehalfOfRecipientID   CDATA  #IMPLIED
                                  >


<!-- =================================== -->
<!-- Declaration of wctp-PollForMessages -->
<!-- =================================== -->

<!ELEMENT wctp-PollForMessages    ( wctp-MessageReceived*
                                  ) >
<!ATTLIST wctp-PollForMessages
          pollerID                CDATA  #REQUIRED
          securityCode            CDATA  #IMPLIED
          maxMessagesInBatch      CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-MessageReceived    ( wctp-Failure
                                  | wctp-Success
                                  ) >
<!ATTLIST wctp-MessageReceived
          sequenceNo              CDATA  #REQUIRED
```

```
                                  >

<!-- =================================== -->
<!-- Declaration of wctp-PollResponse    -->
<!-- =================================== -->

<!ELEMENT wctp-PollResponse       ( wctp-Message+
                                  | wctp-Failure
                                  | wctp-NoMessages
                                  ) >
<!ATTLIST wctp-PollResponse
          minNextPollInterval     CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-Message            ( wctp-SubmitRequest
                                  | wctp-MessageReply
                                  | wctp-StatusInfo
                                  | wctp-LookupResponse
                                  ) >
<!ATTLIST wctp-Message
          sequenceNo              CDATA  #REQUIRED
                                  >

<!ELEMENT wctp-NoMessages         EMPTY
                                  >

<!-- ==================================  -->
<!-- Declaration of wctp-StatusInfo      -->
<!-- ==================================  -->

<!ELEMENT wctp-StatusInfo         ( wctp-ResponseHeader
                                  | wctp-Failure
                                  | wctp-Notification )
                                  ) >

<!ELEMENT wctp-Failure            (#PCDATA)
                                  >
<!ATTLIST wctp-Failure
          errorCode               CDATA  #REQUIRED
          errorText               CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-Notification       EMPTY
                                  >
<!ATTLIST wctp-Notification
          type                    ( QUEUED | DELIVERED | READ )  #REQUIRED
                                  >

<!-- ======================================= -->
<!-- Declaration of wctp-SubmitClientMessage -->
<!-- ======================================= -->

<!ELEMENT wctp-SubmitClientMessage ( wctp-SubmitClientHeader
                                   , wctp-Payload
                                   ) >

<!ELEMENT wctp-SubmitClientHeader  ( wctp-ClientOriginator
                                   , wctp-ClientMessageControl?
                                   , wctp-Recipient
                                   ) >
<!ATTLIST wctp-SubmitClientHeader
          submitTimestamp          CDATA  #IMPLIED
                                   >

<!ELEMENT wctp-ClientOriginator    EMPTY
                                   >
<!ATTLIST wctp-ClientOriginator
          senderID                 CDATA  #REQUIRED
          miscInfo                 CDATA  #IMPLIED
                                   >

<!ELEMENT wctp-ClientMessageControl     EMPTY
                                  >
<!ATTLIST wctp-ClientMessageControl
          sendResponsesToID       CDATA  #IMPLIED
          allowResponse           ( true | false ) "true"
          notifyWhenQueued        ( true | false ) "false"
          notifyWhenDelivered     ( true | false ) "false"
          notifyWhenRead          ( true | false ) "false"
          deliveryPriority        ( HIGH | NORMAL | LOW) "NORMAL"
          deliveryAfter           CDATA  #IMPLIED
          deliveryBefore          CDATA  #IMPLIED
          preformatted            ( true | false ) "false"
          allowTruncation         ( true | false ) "true"
                                  >


<!-- ======================================== -->
<!-- Declaration of wctp-SubmitClientResponse -->
<!-- ======================================== -->

<!ELEMENT wctp-SubmitClientResponse ( wctp-Failure
                                    | wctp-ClientSuccess
                                    ) >

<!ELEMENT wctp-ClientSuccess        (#PCDATA)
                                    >
<!ATTLIST wctp-ClientSuccess
          successCode               CDATA  #REQUIRED
          successText               CDATA  #IMPLIED
          trackingNumber            CDATA  #REQUIRED
                                    >

<!-- ================================== -->
<!-- Declaration of wctp-SubmitRequest  -->
<!-- ================================== -->

<!ELEMENT wctp-SubmitRequest      ( wctp-SubmitHeader
                                  , wctp-Payload
                                  ) >

<!ELEMENT wctp-SubmitHeader       ( wctp-Originator
                                  , wctp-MessageControl
                                  , wctp-Recipient
                                  ) >
<!ATTLIST wctp-SubmitHeader
          submitTimestamp         CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-Originator         EMPTY
                                  >
<!ATTLIST wctp-Originator
          senderID                CDATA  #REQUIRED
          securityCode            CDATA  #IMPLIED
          miscInfo                CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-MessageControl     EMPTY
                                  >
<!ATTLIST wctp-MessageControl
          messageID               CDATA  #REQUIRED
          transactionID           CDATA  #IMPLIED
          sendResponsesToID       CDATA  #IMPLIED
          allowResponse           ( true | false ) "true"
          notifyWhenQueued        ( true | false ) "false"
          notifyWhenDelivered     ( true | false ) "false"
          notifyWhenRead          ( true | false ) "false"
          deliveryPriority        ( HIGH | NORMAL | LOW) "NORMAL"
          deliveryAfter           CDATA  #IMPLIED
          deliveryBefore          CDATA  #IMPLIED
          preformatted            ( true | false ) "false"
          allowTruncation         ( true | false ) "true"
                                  >

<!ELEMENT wctp-Recipient          EMPTY
                                  >
<!ATTLIST wctp-Recipient
          recipientID             CDATA  #REQUIRED
          authorizationCode       CDATA  #IMPLIED
                                  >

<!ELEMENT wctp-Payload            ( wctp-MCR
                                  | wctp-Alphanumeric
                                  | wctp-TransparentData
                                  ) >

<!ELEMENT wctp-MCR                ( wctp-MessageText
                                  , wctp-Choice+
                                  ) >

<!ELEMENT wctp-MessageText        (#PCDATA)
                                  >

<!ELEMENT wctp-Choice             (#PCDATA)
                                  >

<!-- ================================ -->
<!—- Declaration of wctp-VersionQuery -->
<!-- ================================ -->

<!ELEMENT wctp-VersionQuery         EMPTY
```

```
                                  >
<!ATTLIST wctp-VersionQuery
          inquirer                CDATA #REQUIRED
          dateTime                CDATA #IMPLIED
          listDTDs                (yes | no) "no"
          listConfiguration       (yes | no) "no"
>

<!-- ======================================== -->
<!—- Declaration of wctp-VersionResponse -->
<!-- ======================================== -->

<!ELEMENT wctp-VersionResponse      ( wctp-ContactInfo?
                                    , wctp-Failure
                                    | wctp-DTDSupport+
                                    ) >

<!ATTLIST wctp-VersionResponse
          responder               CDATA #REQUIRED
          dateTimeOfRsp           CDATA #IMPLIED
          inquirer                CDATA #IMPLIED
          dateTimeOfReq           CDATA #IMPLIED
          invalidAfter            CDATA #IMPLIED
          listDTDs                (yes | no) "no"
          listConfiguration       (yes | no) "no"
>


<!-- =============================== -->
<!-- Declaration of wctp-ContactInfo -->
<!-- =============================== -->

<!ELEMENT wctp-ContactInfo    (EMPTY)     >

<!ATTLIST wctp-ContactInfo
          email               CDATA #IMPLIED
          phone               CDATA #IMPLIED
          www                 CDATA #IMPLIED
          info                CDATA #IMPLIED
>

<!-- ============================= -->
<!-- Declaration of wctpDTDSupport -->
<!-- ============================= -->

<!ELEMENT wctp-DTDSupport     (EMPTY)     >

<!ATTLIST wctp-DTDSupport
          dtdName             CDATA #REQUIRED
          verToken            CDATA #IMPLIED
          supportType         ( Supported
                              | Deprecated
                              | NotSupported ) "Supported"
          exceptions          (yes | no) "no"
          supportUntil        CDATA #IMPLIED
          replacement         CDATA #IMPLIED
>
```

```
<!-- ================================ -->
<!-- Common elements to WCTP commands -->
<!-- ================================ -->

<!ELEMENT wctp-Alphanumeric       (#PCDATA)
                                  >

<!ELEMENT wctp-TransparentData    (#PCDATA)
                                  >
<!ATTLIST wctp-TransparentData
          type                    ( OPAQUE | FLEXsuite ) "OPAQUE"
          encoding                ( standard | base64 ) "base64"
                                  >
```

Appendix B - WCTP Field Definitions

Field Tag	O/M	Field Value	Comments
Attributes of wctp-Operation			
wctpVersion	M	Char String	Version of the WCTP protocol used to communicate the WCTP operation.
wctpToken	O	Char String	An arbitrary key to indicate when the supported DTDs have changed.
Attributes of wctp-SubmitHeader			
submitTimestamp	M	Time Stamp	Date/Time when request was created.
Attributes of wctp-Originator			
senderID	M	Char String	Unique identifier of the message submitter. The length of the senderID string is limited to 254 octets.
securityCode	O	Char String	Password validating sender as being allowed to send under the specified senderID.
miscInfo	O	Char String	Information about the message sender such as a domain name. This field is intended for logging purposes only. The information provided is not utilized by the protocol in any manner.
Attributes of wctp-MessageControl			
messageID	M	Char String	Message reference number of this message submission. The message ID must be unique for a particular sender ID for the duration of the lifetime of this message transmission and the receipt of one or more responses. The length of the messageID string is limited to 254 octets.
transactionID	O	Char String	Identifier to tie together multiple message submissions into a common set primarily for logging purposes and application-level session management. There is no requirement for this value to be unique. The information content of this field is not utilized by the protocol. The length of the transactionID string is limited to 254 octets.
sendResponsesToID	O	Char String	Unique identifier where responses should be returned. Responses are returned to the originator if this field is not specified. Responses can be directed to non-WCTP destination, such as e-mail, fax, phone. Such destination is identified with one of the reserved prefixes: **mailto:**, **faxto:**, **phoneto:**.
allowResponse	O	Char String	If set to false, indicates that no recipient response should be returned.

Field Tag	O/M	Field Value	Comments
notifyWhenQueued	O	Char String	True if a notification response should be sent when the message is queued to the recipient.
notifyWhenDelivered	O	Char String	True if a notification response should be sent when the message is delivered to the recipient.
notifyWhenRead	O	Char String	True if a notification response should be sent when the message is read by the recipient. This capability may be dependent on the features of the recipient.
deliveryPriority	O	Char String	Delivery urgency of the message.
deliveryAfter	O	Time Stamp	Message should be held until the time specified. Must be smaller than the DeliveryBefore value if one is specified.
deliveryBefore	O	Time Stamp	Message should be delivered by the time specified or automatically canceled. Must be larger than the DeliveryAfter value if one is specified. If specified, the DeliveryAfter function is performed before DeliveryBefore function.
allowTruncation	O	Char String	True if Message payload can be truncated. False if payload truncation is impossible (for example, messages with binary payloads may not be truncatable).
preformatted	O	Char String	True if Message payload is preformatted and should not be modified in any way.
Attributes of wctp-Recipient			
recipientID	M	Char String	Unique identifier of a message recipient. The length of the recipientID string is limited to 254 octets.
authorizationCode	O	Char String	Password for recipients that are programmed not to accept messages without the proper authorization.
Sub-Elements of wctp-Payload			
wctp-MCR	O	Element	Multiple choice response definition.
Sub-Elements of wctp-MCR			
wctp-MessageText	O	Char String	Body of the alphanumeric message to send.
wctp-Choice	O	Element	Definition of one possible response choice.
Attributes of wctp-ResponseHeader			
responseToMessageID	M	Char String	Message Identifier of the message to which this response refers.
responseTimestamp	O	Time Stamp	Date/Time when response was created (and **not** the time it is delivered via WCTP).

Field Tag	O/M	Field Value	Comments
respondingToTimestamp	O	Time Stamp	Date/Time of message to which this response refers.
onBehalfOfRecipientID	O	Char String	Recipient ID to which this response refers.
Attributes of wctp-Notification			
type	M	Char String	Indicates the type of notification being reported.
Content of wctp-Failure			
<body of element>	O	Char String	The body of this element may contain a portion of the data to which this error refers.
Attributes of wctp-Failure			
errorCode	M	Char String	WCTP standard numeric error value representing the type of error being reported.
errorText	O	Char String	Description of the error condition.
Attributes of wctp-Success			
successCode	M	Char String	WCTP standard numeric success code of the response being reported.
successText	O	Char String	Additional information regarding the success of the operation.
Attributes of wctp-PollForMessages			
pollerID	M	Char String	Unique identifier of the system polling for messages.
securityCode	O	Char String	Password validating poller as being allowed to send poll requests under the specified pollerID.
maxMessagesInBatch	O	Char String	Maximum number of messages to be returned in response to this poll request.
Attributes of wctp-PollResponse			
minNextPollInterval	O	Char String	Minimum interval before the next Poll request can be issued by the poller where this poll response is directed.
Attributes of wctp-MessageReceived			
sequenceNo	M	Char String	Sequence number of the message received in response to the previous Poll request. Used as an acknowledgement of the receipt of the message.
Attributes of wctp-Message			
sequenceNo	M	Char String	Sequence number of the message being sent as a part of a response to the Poll request.
Attributes of wctp-SubmitClientHeader			
submitTimestamp	M	Time Stamp	Date/Time when request was created.

Field Tag	O/M	Field Value	Comments
Attributes of wctp-ClientOriginator			
senderID	M	Char String	Identifier of the message submitter. Uniqueness of this identifier for client submissions cannot be guaranteed.
miscInfo	O	Char String	Information about the message sender such as a user name. This field is intended for logging purposes only. The information provided is not utilized by the protocol in any manner.
Attributes of wctp-ClientMessageControl			
sendResponsesToID	O	Char String	Unique identifier where responses should be returned. Responses are returned to the originator if this field is not specified. Responses can be directed to non-WCTP destination, such as e-mail, fax, phone. Such destination is identified with one of the reserved prefixes: **mailto:**, **faxto:**, **phoneto:**, **pageto:**.
allowResponse	O	Char String	If set to false, indicates that no recipient response should be returned.
notifyWhenQueued	O	Char String	True if a notification response should be sent when the message is queued to the recipient.
notifyWhenDelivered	O	Char String	True if a notification response should be sent when the message is delivered to the recipient.
notifyWhenRead	O	Char String	True if a notification response should be sent when the message is read by the recipient. This capability may be dependent on the features of the recipient.
deliveryPriority	O	Char String	Delivery urgency of the message.
deliveryAfter	O	Time Stamp	Message should be held until the time specified. Must be smaller than the DeliveryBefore value if one is specified.
deliveryBefore	O	Time Stamp	Message should be delivered by the time specified or automatically canceled. Must be larger than the DeliveryAfter value if one is specified. If specified, the DeliveryAfter function is performed before DeliveryBefore function.
allowTruncation	O	Char String	True if Message payload can be truncated. False if payload truncation is impossible (for example, messages with binary payloads may not be truncatable).

preformatted	O	Char String	True if Message payload is preformatted and should not be modified in any way.
Attributes of wctp-ClientSuccess			
successCode	M	Char String	WCTP standard numeric success code of the response being reported.
successText	O	Char String	Additional information regarding the success of the operation.
trackingNumber	M	Char String	Tracking number assigned to the submitted message, which is being acknowledged.
Attributes of wctp-ClientQuery			
senderID	M	Char String	Sender of the original message for which replies and status notifications are being requested.
recipientID	M	Char String	Recipient of the original message for which replies and status notifications are being requested.
trackingNumber	M	Char String	Tracking number of the original message for which replies and status notifications are being requested.
Attributes of wctp-ClientQueryResponse			
minNextPollInterval	O	Char String	Minimum interval before the next Client Query request can be issued by the transient client.
Attributes of wctp-ClientResponseHeader			
responseTimestamp	O	Time Stamp	Date/Time when the reply was created (and **not** the time it was delivered via WCTP).
respondingToTimestamp	O	Time Stamp	Date/Time when the original message was created (and **not** the time it was delivered via WCTP).
Attributes of wctp-VersionQuery			
inquirer	M	Char String	An identifier for the agent making the version query. Generally expected to be a URI, it must correlate to the “from” field in future requests in order for any version information provided to be useful to the carrier or responder. This field has a maximum length of 255 octets.
dateTime	O	Time Stamp	The date/time stamp at which the version query was created. If it is included in the query, the value will appear in the wctp-VersionResponse in the dateTimeOfReq field. This field has a maximum length of 21 octets.
listDTDs	O	Yes|No	Specifies whether or not the inquirer wants a list of all of the DTDs supported by the server. This field is not used in version 1.1, and is placed here in preparation

			for WCTP version 2.0.
listConfiguration	O	Yes|No	Specifies whether or not the inquirer wants the configuration information for this server. This field is not used in version 1.1, and is placed here in preparation for WCTP version 2.0.
Attributes of wctp-VersionResponse			
responder	M	Char String	An identifier for the server providing the version information. Must be a valid URI/URL for making future queries/requests. This field has a maximum length of 255 octets.
dateTimeOfRsp	O	Time Stamp	The date/time stamp at which the version info was generated. This is a standard WCTP date and time field.
inquirer	C	Char String	An identifier for the agent making the version query. It must be returned here only if it was provided in the original request. This field has a maximum length of 255 octets.
dateTimeOfReq	C	Time Stamp	The date/time stamp at which the version request was sent. It must be returned here only if it was provided in the original request. This is a standard WCTP date and time field.
invalidAfter	O	Time Stamp	The date/time after which the version info provided here will no longer be valid. This is a standard WCTP date and time field.
listDTDs	O	Yes|No	Specifies whether or not the user requested a list of all of the DTDs supported by the server. This field is not used in version 1.1, and is placed here in preparation for WCTP version 2.0.
listConfiguration	O	Yes|No	Specifies whether or not the user requested configuration information for this server. This field is not used in version 1.1, and is placed here in preparation for WCTP version 2.0.
Attributes of wctp-ContactInfo			
email	O	SMTP Address	A valid SMTP email address where the querying agent can be contacted if additional version or support information is needed. This field has a maximum length of 255 octets.

phone	O	Char String	A valid telephone number where the querying agent can be contacted if additional version or support information is needed. Any of {0123456789,( )-x and space} [see text]. This field has a maximum length of 30 octets.
www	O	URI String	A valid web site URI/URL where additional version or support information can be found if needed. This field has a maximum length of 255 octets.
info	O	Char String	Any additional version or support information or details. This field has a maximum length of 255 octets.
Attributes of wctp-DTDSupport			
dtdName	M	DTD Name	The name of the DTD for which support/version information is being provided. This field has a maximum length of 30 octets.
supportType	O	Char String	Specifies what kind of support, if any, is offered for the named DTD. Valid values are Supported, NotSupported, and Deprecated. If no value is specified the default is “Supported”.
exceptions	O	Yes|No	Specifies whether or not the support for the specified DTD contains exceptions from the standard specification. This field has no meaning unless the supported field is “Yes”. Valid values are Yes and No, and if no value is specified the default is “No”.
supportUntil	O	Time Stamp	The optional date/time stamp at which the specified DTD will become unsupported, after which it is not valid for use with this server. [see WCTP date-time]
replacement	O	DTD Name	The name of the DTD (if any) which should be used in place of this DTD [see dtdName above]. This field has a maximum length of 30 octets.
verToken	O	Char String	The current version of support for this DTD. This field has a maximum length of 21 octets.
Attributes of wctp-TransparentData			
type	O	Char String	Type of transparent data being carried in the wctp-TransparentData element.
encoding	O	Char String	Encoding of transparent data being carried in the wctp-TransparentData element.

Attributes of wctp-LookupMessageControl			
messageID	M	Char String	Message reference number for this submission. The message ID must be unique for a particular sender ID for the duration of the lifetime of this message and the receipt of the response. The length of the messageID string is limited to 254 octets.
transactionID	O	Char String	Identifier to tie together multiple requests into a common set primarily for logging purposes and application-level session management. There is no requirement for this value to be unique. The information content of this field is not utilized by the protocol. The length of the transactionID string is limited to 254 octets.
sendResponsesToID	O	Char String	Unique identifier where lookup information should be returned (they will be returned to the originator if this field is not specified).
Attributes of wctp-LookupResponse			
responseToMessageID	M	Char String	Message Identifier of the request message to which this is a response.
transactionID	O	Char String	Identifier to tie together multiple requests into a common set primarily for logging purposes and application-level session management. There is no requirement for this value to be unique. The information content of this field is not utilized by the protocol. The length of the transactionID string is limited to 254 octets.
Attributes of wctp-LookupData			
maxMessageLength	M	Char String	The maximum number of bytes allowed in messages to this recipient.
mcrSupported	O	Char String	Whether or not Multiple Choice Response messages can be sent to this recipient. Valid values are true and false, and there is no default value.
canRespond	O	Char String	Whether or not this recipient can responds from his/her device. Valid values are true and false, and there is no default value.

Appendix C - Error and Success Codes

Code	Definition	Comments
		200 Series Success Codes
200	Acknowledged	The operation has been received and a first level of validation has been performed. Further validation may result in additional responses. Other responses to this message, such as delivery notifications, may occur as per the operations requested.
210	Deprecated version	The WCTP version used has been deprecated.
211	Message exceeds allowable length	Message exceeds allowable message length. Message will be delivered, but truncated at max.
212	Delivery ACK Not Supported by Device	The Device does not support Delivery Acknowledgements. Message will be delivered but no DELIVERED notification will be returned.
213	Read ACK Not Supported by Device	The Device does not support Read Acknowledgements. Message will be delivered but no READ notification will be returned.
215	Multiple Choice Response Not Supported by Device	Device does not support Multiple Choice Responses. Converted to appended text.
216	Multiple Choice Responses exceeds allowable length	Multiple Choice Response text is too long. Truncated to max
217	Max Multiple Choice Responses exceeded	Message contains too many Multiple Choice Responses. Truncated at max
218	Delivery ACK Not Supported	The Network does not support Delivery Acknowledgements. Message will be delivered, but no DELIVERED notification will be returned.
219	Read ACK Not Supported	The Network does not support Read Acknowledgements. Message will be delivered but no READ notification will be returned.
220	Multiple Choice Responses Not Supported	The Network does not support Multiple Choice Responses. Converted to appended text.

		300 Series Protocol Violation Error Codes
300	Operation not supported	The requested WCTP operation is not supported by this system.
301	Input can not be parsed	Input message is garbled. Invalid XML format. Could be caused by binary data or non-XML data being sent.
302	XML validation error	Input is not a well formed XML document and cannot be processed properly.

Code	Definition	Comments
		properly.
303	Version error	The version of the input data is not supported.
304	Cannot accept operation	The requested operation can not be performed until messages are pulled using the PollForMessages operation.

400 Series Content Error Codes		
400	Function not supported	The requested function, is not supported for this operation.
401	Invalid SenderID	The senderID is invalid.
402	Invalid security code	The security code for this senderID is invalid.
403	Invalid recipientID	The recipientID is invalid.
404	Invalid authorization code	The authorization code for this recipientID is invalid.
405	Invalid Date or Time Format	A date and/or time value is not in the WCTP Standard date and/or time format.
406	Date/Time Ranges specify an unsupported combination	A combination of a request to send a message after a specific time but before another time is incorrectly specified. The 'before' date/time value must be beyond the 'after' date/time value.
407	Invalid date/time period	The specified date and/or time is beyond the time range supported by the system.
408	Date/time specified has already expired	The date/time specified has already passed and the operation requested can not be performed as specified.
411	Message exceeds allowable length	Message exceeds allowable message length.
412	Delivery ACK Not Supported by Device	The Device does not support Delivery Acknowledgements.
413	Read ACK Not Supported by Device	The Device does not support Read Acknowledgements.
414	Multiple Choice Response Not Supported by Device	Device does not support Multiple Choice Responses.
415	Multiple Choice Responses exceeds allowable length	Multiple Choice Response text is too long.

Code	Definition	Comments
416	Max Multiple Choice Responses exceeded	Message contains too many Multiple Choice Responses.
417	Delivery ACK Not Supported	The Network does not support Delivery Acknowledgements.
418	Read ACK Not Supported	The Network does not support Read Acknowledgements.
419	Multiple Choice Responses Not Supported	The Network does not support Multiple Choice Responses.
430	Binary Message Not Supported	Device does not support Binary Messages.
431	Subscriber Limit Exceeded	Message limit to subscriber has temporarily been exceeded
432	Invalid Poller or Security Code	Invalid combination of Poller ID with Security Code

500 Series Message Processing Error Codes (Permanent)		
500	Message has timed out	The message could not be delivered within a network specific maximum delivery time period.
501	Message has expired	The message could not be delivered within the delivery time period specified when it was submitted.
502	No further responses possible	The specified message ID will no longer be able to receive responses because of network or device limitations.
503	Sender permanently disabled due to excessive polling or traffic	Manual intervention by the operator of the WCTP server is required. The system performing polling has ignored warnings to reduce the polling rate or warnings that traffic levels are over the allowed quota. Service has been barred.
504	Unknown Message Reference	Message reference (a combination of senderID, recipientID and a trackingNumber) presented in ClientQuery is not recognized.

600 Series Message Processing Error Codes (Temporary)		
600	Poll rate exceeded	Warning. Polling for messages and status is occurring too frequently. The error text, which accompanies this status, may provide a value (in seconds) for the maximum allowed polling rate.

Code	Definition	Comments
601	Excessive polling	Warning. Polling is occurring at a rate beyond the allowed limitations. If this continues, the polling system will be permanently disabled (see error 503).
602	Traffic rate exceeded	Warning. The rate at which messages are entering the system exceeds the allowed quota. Slow down the message entry rate.
603	Traffic rate excessive	Warning. The rate at which traffic is entering the system exceeds the allowed quota. If this continues the sender may be permanently disabled (see error 503).
604	Internal Server Error	The WCTP gateway encountered an unexpected condition that prevents it from processing the message.
605	Service Unavailable	The WCTP gateway is unable to handle the request due to either temporary overloading or server maintenance.

900 Series Experimental Error Codes		
900-999	Implementation specific	Implementations may experiment with new codes and new features employing these codes, without fear of interfering with future versions of the WCTP specification, by utilizing code values within this range of numbers. The 900 series codes will never be specified in an officially released WCTP specification. Presumably, once an experimental implementation is deemed to be operational, the developer of this new capability may submit a request to the appropriate WCTP Standards Committee to request that an official status or error code designation be allocated.

Appendix D – Example of WCTP Based Applications

To better understand how WCTP might be used in typical applications and to better demonstrate why the protocol is carrying certain types of information, this section describes a fictional application that could be implemented under the protocol. To show how WCTP may be used to not only move data between the wireline and wireless network, but also within the infrastructure of the network, this example also shows the message flow between elements within the wireless network.

The sample application involves the following elements:

- Wireless-equipped telemetry device capable of monitoring the operation of the Gadget machine and capable of adjusting the operation of the Gadget machine based on the command from the Telemetry server.

- Front-end WCTP-equipped Internet Gateway of the wireless network.

- Wireless Communications Gateway (WCG) providing an interface between the Internet Gateway and the protocols of a specific wireless network

- Telemetry server capable of receiving the alerts from the telemetry device, sending back correcting actions in the format understood by the telemetry device, supporting the technician on-duty database, able to communicate with technicians regarding the selection of corrective action and keeping a log of all of the activities. The telemetry server is implemented as a WCTP Enterprise Host.

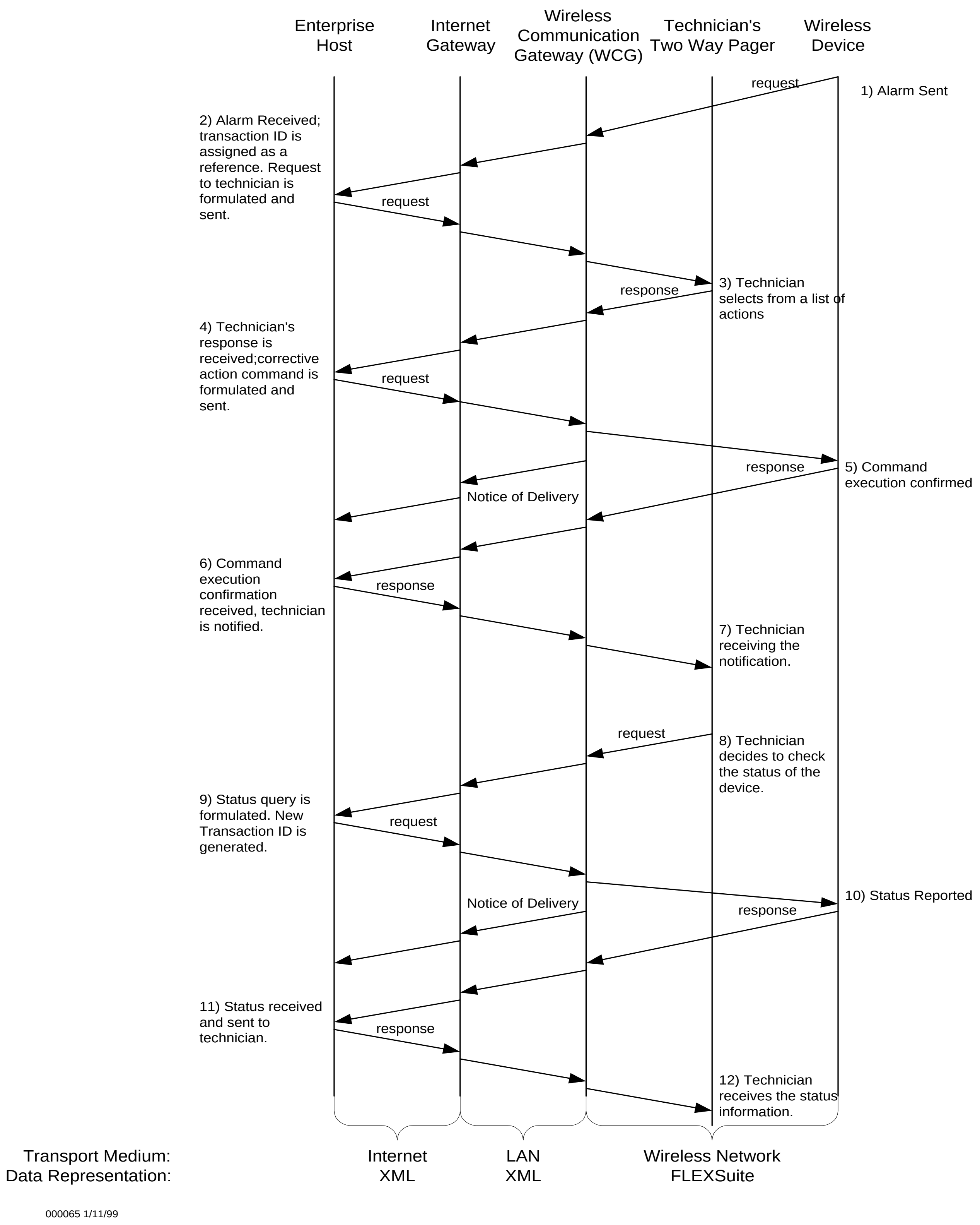
- Fleet of technicians.

The following call flow will be followed by this example:

The telemetry device registers abnormal condition of the Gadget machine and sends an alert to the Enterprise Host (Telemetry Server). The telemetry server selects potential corrective actions applicable for this specific type of failure and presents them to a technician equipped with a 2-way pager in a multiple-choice form. The technician selects the appropriate corrective action in his reply. The telemetry server encodes and sends the appropriate command to the device. The telemetry device confirms that the corrective action is executed. The telemetry server host forwards a notice to the technician. Later, the technician sends a request to the telemetry server to inquire about the state of the Gadget machine. The telemetry server formulates and encodes a query and sends it to the device. The device responds. The telemetry server delivers status information back to the technician.

The transaction flow for this scenario is shown in D-1 Call Flow of The Telemetry Application

.


Enterprise Host
Internet Gateway
Wireless Communication Gateway (WCG)
Technician's Two Way Pager
Wireless Device
request
1) Alarm Sent
2) Alarm Received; transaction ID is assigned as a reference. Request to technician is formulated and sent.
request
3) Technician selects from a list of actions
response
4) Technician's response is received;corrective action command is formulated and sent.
request
5) Command execution confirmed
response
Notice of Delivery
6) Command execution confirmation received, technician is notified.
response
7) Technician receiving the notification.
request
8) Technician decides to check the status of the device.
9) Status query is formulated. New Transaction ID is generated.
request
10) Status Reported
Notice of Delivery
response
11) Status received and sent to technician.
response
12) Technician receives the status information.
Transport Medium:
Data Representation:
Internet
XML
LAN
XML
Wireless Network
FLEXSuite
000065 1/11/99

D-1 Call Flow of The Telemetry Application

This section contains examples of specific WCTP operations executed.

1. Request From the Wireless Device to the Enterprise Host

1.1. Alarm Message Sent

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation
          SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-SubmitRequest>
    <wctp-SubmitHeader
        submitTimestamp="1999-03-31T18:15:00"
    >
        <wctp-Originator
            senderID="1234567@abcwireless.com"
            securityCode="OpenSesame"
        />
        <wctp-MessageControl
            messageID="1C936BA3"
        />
        <wctp-Recipient
            recipientID="controlcenter@myenterprise.com"
            authorizationCode="HelloWorld"
        />
    </wctp-SubmitHeader>
    <wctp-Payload>
        <wctp-TransparentData
            type="FLEXSuite"
            encoding="base64"
        >
…              - base64-encoded binary FLEXSuite message
        </wctp-TransparentData>
    </wctp-Payload>
  </wctp-SubmitRequest>
</wctp-Operation>
```

2. Request from the Enterprise Host to the Wireless Device

2.1. The Enterprise Host presents multiple choice selections to a technician

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM "http://dtd.wctp.org/ wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1">
  <wctp-SubmitRequest>
    <wctp-SubmitHeader
        submitTimestamp="1999-03-31T18:18:00">
        <wctp-Originator
            senderID="controlcenter@myenterprise.com"
            miscInfo="Enterprise Corp. Operations Control Center"
        />
        <wctp-MessageControl
            messageID="46264399"
            transactionID="19990331.PN.1234567.001"
            notifyWhenQueued="true"
            notifyWhenDelivered="true"
            notifyWhenRead="true"
            deliveryPriority="HIGH"
        />
        <wctp-Recipient
            recipientID="john.tech@skypage.com"
        />
    </wctp-SubmitHeader>
    <wctp-Payload>
        <wctp-MCR>
            <wctp-MessageText>
                pH threshold exceeded. Location: Louisville, KY.
                TankNumber: 1234. DeviceId: Abcwireless.1234567
                Select a Corrective Action
            </wctp-MessageText>
            <wctp-Choice>
                Raise temperature 1 degree.
            </wctp-Choice>
            <wctp-Choice>
                Lower temperature 1 degree.
            </wctp-Choice>
            <wctp-Choice>
                Raise pressure 1 atm.
            </wctp-Choice>
            <wctp-Choice>
                Lower pressure 1 atm.
            </wctp-Choice>
        </wctp-MCR>
    </wctp-Payload>
  </wctp-SubmitRequest>
</wctp-Operation>
```

3. Delivery notification – Gateway to Enterprise Host

3.1. The Gateway informs the Enterprise Host that the message was successfully delivered to a 2-way device.

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-StatusInfo>
    <wctp-ResponseHeader
        responseToMessageID="46264399"
        responseTimestamp="1999-03-31T18:19:00"
        respondingToTimestamp="1999-03-31T18:18:00"
    >
        <wctp-Originator
            senderID="wcg-3.abcwireless.com"
            miscInfo="WCG for Skypage Network"
        />
        <wctp-MessageControl
            messageID="32456"
            transactionID="19990331.PN.1234567.001"
        />
        <wctp-Recipient
            recipientID="controlcenter@myenterprise.com"
        />
    </wctp-ResponseHeader>
    <wctp-Notification
        type="DELIVERED"
    />
  </wctp-StatusInfo>
</wctp-Operation>
```

4. Response from the Wireless Device to the Enterprise Host

4.1. The technician selects one of the presented choices

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-MessageReply>
    <wctp-ResponseHeader
        responseToMessageID="46264399"
        responseTimestamp="1999-03-31T18:25:00"
        respondingToTimestamp="1999-03-31T18:18:00"
    >
        <wctp-Originator
            senderID="john.tech@skypage.com"
            miscInfo="John The Technician"
        />
        <wctp-MessageControl
            messageID="33"
            transactionID="19990331.PN.1234567.001"
        />
        <wctp-Recipient
            recipientID="controlcenter@myenterprise.com"
        />
    </wctp-ResponseHeader>
    <wctp-Payload>
        <wctp-Alphanumeric>
            Raise temperature 1 degree.
        </wctp-Alphanumeric>
    </wctp-Payload>
  </wctp-MessageReply>
</wctp-Operation>
```

5. Response from the Enterprise Host to the Wireless Device

5.1. The Enterprise Host sends a request to perform a corrective action

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1">
  <wctp-SubmitRequest>
    <wctp-SubmitHeader
        submitTimestamp="1999-03-31T8:30:00"
    >
        <wctp-Originator
            senderID="controlcenter@myenterprise.com"
            securityCode="HelloWorld"
            miscInfo="Enterprise Corp. Operations Control Center"
        />
        <wctp-MessageControl
            messageID="46264423"
            transactionID="19990331.PN.1234567.001"
            notifyWhenQueued="true"
            notifyWhenDelivered="true"
            notifyWhenRead="true"
            deliveryPriority="HIGH"
        />
        <wctp-Recipient
            recipientID="1234567@abcwireless.com"
            authorizationCode="OpenSesame"
        />
    </wctp-SubmitHeader>
    <wctp-Payload>
        <wctp-TransparentData
            type="FLEXSuite"
            encoding="base64"
        >
…              - base64-encoded binary FLEXSuite message
        </wctp-TransparentData>
    </wctp-Payload>
  </wctp-SubmitRequest>
</wctp-Operation>
```

6. Confirmation of message acceptance to the Enterprise Host

6.1. The Gateway informs the Enterprise Host that the message passed its protocol conformance verification.

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM "http://dtd.wctp.org/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-Confirmation>
    <wctp-Success
        successCode="200"
        successText="Message acknowledged"
    >
         Your message is being processed by the ABC wireless network
    </wctp-Success>
  </wctp-Confirmation>
</wctp-Operation>
```

7. Response from the Wireless Device to the Enterprise Host

7.1. The Wireless device informs the Enterprise Host that the corrective action has been started

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM
"http://www.pcia.com/wireres/protocol/dtd/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-MessageReply>
    <wctp-ResponseHeader
        responseToMessageID="46264423"
        responseTimestamp="1999-03-31T18:35:00"
        respondingToTimestamp="1999-03-31T18:30:00"
    >
        <wctp-Originator
            senderID="1234567@abcwireless.com"
        />
        <wctp-MessageControl
            messageID="1C936BA3"
            transactionID="19990331.PN.1234567.001"
        />
        <wctp-Recipient
            recipientID="controlcenter@myenterprise.com"
        />
    </wctp-ResponseHeader>
    <wctp-Payload>
        <wctp-TransparentData
            type="FLEXSuite"
            encoding="base64"
        >
…              - base64-encoded binary FLEXSuite message
        </wctp-TransparentData>
    </wctp-Payload>
  </wctp-MessageReply>
</wctp-Operation>
```

8. The response from the Enterprise Host to the Wireless Device

8.1. The Enterprise Host informs the technician that the corrective action has been started

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM
"http://www.pcia.com/wireres/protocol/dtd/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1">
  <wctp-MessageReply>
    <wctp-ResponseHeader
        responseToMessageID="33"
        responseTimestamp="1999-03-31T18:37:00"
        respondingToTimestamp="1999-03-31T18:25:00"
    >
        <wctp-Originator
            senderID="controlcenter@myenterprise.com"
        />
        <wctp-MessageControl
            messageID="1C936BA3"
            transactionID="19990331.PN.1234567.001"
        />
        <wctp-Recipient
            recipientID="john.tech@skypage.com"
        />
    </wctp-ResponseHeader>
    <wctp-Payload>
        <wctp-Alphanumeric>
            Corrective action (Raise Temperature 1 deg) has been
            initiated. Location: Louisville, KY.
            TankNumber: 1234. DeviceId: Abcwireless.1234567
        </wctp-Alphanumeric>
    </wctp-Payload>
  </wctp-MessageReply>
</wctp-Operation>
```

9. Request from the Wireless Device to the Enterprise Host

9.1. The technician asks the Enterprise Host to request that the telemetry device verify that corrective action was taken

```
<?xml version="1.0"?>
<!DOCTYPE wctp-Operation SYSTEM
"http://www.pcia.com/wireres/protocol/dtd/wctp-dtd-v1r1.dtd">
<wctp-Operation wctpVersion="wctp-dtd-v1r1" wctpToken="11AA">
  <wctp-SubmitRequest>
    <wctp-SubmitHeader
        submitTimestamp="1999-03-31T19:45:00"
    >
        <wctp-Originator
            senderID="john.tech@skypage.com"
            miscInfo="John The Technician"
        />
        <wctp-MessageControl
            messageID="44"
        />
        <wctp-Recipient
            recipientID="controlcenter@myenterprise.com"
        />
    </wctp-SubmitHeader>
    <wctp-Payload>
        <wctp-Alphanumeric>
            QUERY STATE Device=Abcwireless.1234567
        </wctp-Alphanumeric>
    </wctp-Payload>
  </wctp-SubmitRequest>
</wctp-Operation>
```

Appendix E - History of WCTP

1. A New Standard is Needed

The **Wireless Communications Transfer Protocol** (***WCTP***) is specifically aimed at creating an easy means of passing alphanumeric and binary messages to and from wireline systems and two-way capable wireless devices. A draft proposal was submitted to the Radio Paging Community, through an ad hoc "Messaging Standards Committee," as means to foster industry input, co-operation, study, promotion and participation in its further expansion and growth as an open, non-proprietary standard. The Messaging Standards Committee is a group of paging industry manufacturers and carriers focused on rapidly creating mutually agreeable standards to address emerging applications that can not be easily realized through the utilization of existing standards. The Committee accepted the first proposal and established a WCTP drafting sub-committee to further enhance the protocol and release it in the form of this specification. The sub-committee is continuing its efforts to refine the features and capabilities of the protocol, and further feature richness will be introduced in future revisions. It is the intent of the drafting committee to provide the greatest degree of revision compatibility between releases as possible, so as to easily allow for the incorporation of new features and to provide a means for systems operating at different protocol revisions to communicate. With the completion of the first full release of the WCTP specification, the Personal Communications Industry Association agreed to adopt the protocol as a PCIA standard and support the continued activities of the working group as an official PCIA technical sub-committee.

Although introduced through the paging industry, WCTP is directly applicable for messaging to/from most other wireless technologies including PCS, GSM, and cellular.

The wireless industry today hosts a range of different protocols of varying complexities and capabilities to submit messages into a wireless network (e.g. TAP, TNPP, TDP/TME, TME-X, WMtp™, WMapi™, UCP, I4, and SMPP). In certain cases, messages are submitted within the confines of Internet Standard protocols such as the Simple Mail Transport Protocol (SMTP/Email protocol) or the HyperText Transfer Protocol (HTTP). In these cases the extent of the information that may be conveyed to the wireless network is limited due to the nature of these generalized protocols. Sometimes carriers further offer specialized software development kits providing Application Programming Interfaces (API's) which are specific to their individual networks. Oftentimes, the implementers of "wireless enabled" applications need to find the right protocol to integrate into their application for a particular network, only to find that they need to re-implement their application under another protocol for another wireless two-way network.

WCTP has been created **to provide a wireless industry specific standardized representation of messages** which are to:

- Be sent from a wireline host to a wireless device.
- Be sent from a wireless device to a wireline host.
- Facilitate inter-carrier connectivity for messages sent from a wireless device to a wireless device on another carrier's network.

2. WCTP and its relationship to WAP

The Wireless Application Protocol forum has received worldwide attention and at the time of this writing has grown to a membership of more than 85 corporations. A set of approximately 20 specification documents currently exists detailing the architecture and protocol specifics of the various aspects of the "WAP Protocol Suite". WAP is still an evolving specification. Many people, who are not knowledgeable about all of the current WAP specifics, believe that WAP is the all-encompassing answer to two-way communications with wireless devices. In reality, the forum is presently focused on a particular set of features that operate between a "WAP gateway" and wireless devices. It has not addressed the aspect of providing a specific standardized protocol to reach the WAP gateway itself.

The initial focus of WAP has concentrated on the support of a handheld wireless Internet browser and integrated telephony/browser functions as well as a standard set of protocols between the handset, the wireless network and the WAP Gateway to support these functions. The set of protocols established by the WAP forum are designed to support much more than browser functions. WAP discusses the desire to access the various layers of the protocol suite through a set of standardized API's, but these have not yet been established.

WCTP is complementary to WAP in that it addresses a standardized manner to move data to/from the wireless network gateway over a wired network (an area noted as not being addressed). It might even represent a means of presenting data directly to the internal WAP layers. As such, WCTP could potentially be studied for integration with the WAP protocol stack. Having such discussions is anticipated.

Another area where WCTP and WAP differ is in how WCTP views the wireless network. The WAP architecture calls for all wireless devices, regardless of the over-the-air protocol details of the individual wireless network, to support the WAP protocol layers. This helps to make all devices appear the same. WCTP, on the other hand, only looks toward the use of one methodology to move between the wired network and a wireless network gateway (explained shortly). It does not require that all of the wireless networks appear uniform and can be used both by WAP compatible networks as well as today's existing non-WAP networks employing the host of protocols referred to in the WCTP introduction.