



Interface Specification
Universal Gateway - Request Protocol



© 2009 SmartTrust AB. All rights reserved.

SmartTrust endeavors to ensure that the information in this document is correct and fairly stated, but does not accept liability for any error or omission. The development of SmartTrust products and services is continuous and published information may not be up to date. It is important to check the current position with SmartTrust. This document is not part of a contract or license save insofar as may be expressly agreed.

Unless otherwise noted, all names of companies, products, street addresses and persons contained herein are part of a completely fictitious scenario and are designed solely to document the use of the described product or service.

SmartTrust and SmartTrust WIB are trademarks of SmartTrust AB.

All the other trademarks are the property of their respective owners.



Contents

1 Introduction	4
1.1 Backward compatibility	4
2 References	5
3 Definitions and Abbreviations	6
4 Protocol description	7
4.1 Technical Overview	7
4.2 HTTP	7
5 HTTP Request	8
5.1 Request-Line	8
5.1.1 Method	8
5.1.2 HTTP-Version	8
5.2 Headers	8
5.2.1 Accept, Accept-Charset, Accept-Encoding	9
5.2.2 Connection	9
5.2.3 Content-Length	9
5.2.4 Content-Type	9
5.2.5 Cookie	9
5.2.6 Host	10
5.2.7 User-Agent	10
5.2.8 X-Wig-IcclId	10
5.2.9 X-Wig-Info	10
5.3 Examples	11
6 HTTP Response	12
6.1 Status-Line	12
6.1.1 Status-Code	12
6.2 Headers	13
6.2.1 Cache-Control and Expires	13
6.2.2 Connection	13
6.2.3 Content-Length	14
6.2.4 Content-Location	14
6.2.5 Content-Type	14
6.2.6 Location	14
6.2.7 Set-Cookie	14
6.2.8 X-WAP-Payment-Info	15
6.2.9 Transfer-Encoding	15
6.3 Examples	15



1 Introduction

This document defines the protocol between the Universal Gateway (UG) and the Content Provider. The UG acts as a client to the Content Provider. The protocol is used by UG to send information to the Content Provider via http. The document defines the http parameters used in the communication.

The intended audience of this document are people that configure web-servers that receive requests from and send results to UG. The document also provides an application developer with some information on what parameters are available to the application. The document does not specify how the actual application shall behave.

The version specified in this document is supported for UG version 3.3 and later which corresponds to Delivery Platform version 6.0 and later.

For backward compatibility, the UG supports earlier versions of this document, but the use of the syntax specified in this document is encouraged.

1.1 Backward compatibility

Special attention has been paid to ensuring smooth evolution of the specifications. Wherever backward compatibility with DP 5 has not been possible to achieve, it has been clearly stated in the *UG & WSM – Application Developers Guide – Development of Wib Services* [4].

The WIG WML used in the examples is supported by Delivery Platform version 6.1.



2 References

- [1] ETSI. GSM 11.14. SIM Application Toolkit (SIM-ME) Interface. Version 8.5.0. Release 1999.
- [2] RFC 2109. HTTP State Management Mechanism. February 1997.
- [3] RFC 2616. Hypertext Transfer Protocol - HTTP/1.1. June 1999.
- [4] *UG & WSM – Application Developers Guide – Development of Wib Services*, SmartTrust.
- [5] *WIG WML v.5 - Specification*, SmartTrust.



3 Definitions and Abbreviations

Term	Definition
DP	SmartTrust Delivery Platform
S@T	SIM Alliance Toolbox
UG	Universal Gateway
WIG	Wireless Internet Gateway
Wib	SmartTrust Wib™
WML	Wireless Markup Language

4 Protocol description

4.1 Technical Overview

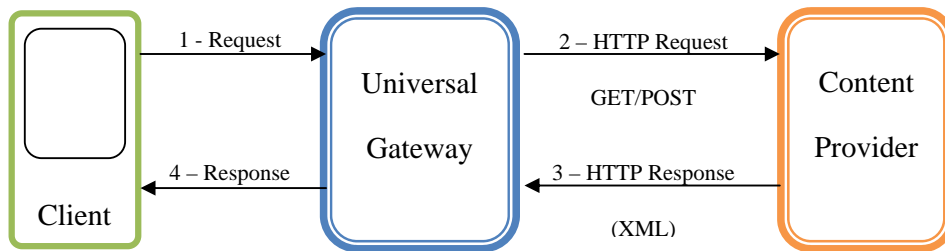


Figure 1 UG - Content Provider Communication

1. The Client sends a request containing binary data to UG.
2. UG receives the request and interprets it as an *HTTP GET or POST request*. See Chapter 5 for further details.
3. The Content Provider sends an *HTTP Response* back to UG. See Chapter 6 for further details.
4. UG parses the HTTP response and compiles the XML document into the bytecode corresponding to the client type. The Client receives the sequence of commands in byte code from UG and executes these commands, using SIM Toolkit [1], for user interactive commands.

To handle secure transactions between the UG and the Content Provider the secure socket layer (SSL) may be used.

4.2 HTTP

The protocols between the UG and the Content Provider are the standard Internet protocols HTTP and TCP/IP. Both the HTTP GET method and the HTTP POST method of the HTTP protocol [3] are supported by UG.



5 HTTP Request

This Chapter describes the HTTP Request from the UG to the Content Provider (Step 2 in

Figure 1).

5.1 Request-Line

The Request has the following structure:

```
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

Example [1]

```
GET /page.jsp?NAME=Bill+Jones HTTP/1.1
```

5.1.1 Method

The UG supports POST and GET.

5.1.2 HTTP-Version

The UG will always send the value `HTTP/1.1`.

5.2 Headers

The following headers are used by the UG in the HTTP header of the GET and/or POST request.

- Accept
- Accept-Charset
- Accept-Encoding
- Connection
- Content-Length
- Content-Type
- Cookie
- Host
- User Agent
- X-Wig-IccId



- X-Wig-Info

5.2.1 Accept, Accept-Charset, Accept-Encoding

The `accept` headers are used to specify certain media types that are acceptable for the response.

Example [2]

In the UG the following values are always used:

```
Accept: text/*  
Accept-Charset: iso-8859-1, UTF-8  
Accept-Encoding: identity
```

5.2.2 Connection

The following values are used:

- Keep-Alive
- close

Example [3]

```
Connection: Keep-Alive
```

5.2.3 Content-Length

Only used for POST requests. According to HTTP [3].

Example [4]

```
Content-Length: 39
```

5.2.4 Content-Type

Only used for POST requests. The following value is used:

- `application/x-www-form-urlencoded`

Example [5]

```
Content-Type: application/x-www-form-urlencoded
```

5.2.5 Cookie

The cookie will be added to the request if the following is true.

- The fully qualified domain name of the server matches the domain attribute of the cookie. The domain attribute was set when the cookie was set. For more details about setting cookies see Section 6.2.7 .
- The cookie Path attribute matches a prefix of the request-URI.



Only the name-value pair attribute is included in the cookie sent in the request. No other attributes are included at the moment.

Example [6]

```
Cookie: cookie1=value1;cookie2=value2
```

5.2.6 Host

The hostname and optionally a port number according to HTTP [3].

Example [7]

```
Host: www.webserver.com:8080
```

5.2.7 User-Agent

UG uses this header to communicate the Client version and the Delivery Platform version to the Content Provider. If the user MSISDN is stored in the DP database, the Client version is retrieved from the DP database, otherwise the configured default value is used.

Example [8]

```
User-Agent: WIG Browser/1.2 Gateway/6.1
```

5.2.8 X-Wig-IccId

This header contains the ICCID of the card in the mobile station if the UG is configured to send it. The name of this header is configurable in the UG, but `x-wig-IccId` is the default name.

Example [9]

```
X-Wig-IccId: 01234567890123456789
```

5.2.9 X-Wig-Info

It is possible to configure the UG to send the MSISDN of the mobile station. The MSISDN can be sent in two ways (configurable):

- **Query string:** In case of GET method, the MSISDN is placed as a name/value pair in the query string after the GET method. See Example [11]. In case of POST method, the MSISDN is placed as a name/value pair in the body. See Example [12].
- **Header:** After the `x-wig-Info` header. See Example [10] and Example [13].

The name of this header is configurable in the UG, but `x-wig-Info` is the default name.

Example [10]

```
X-Wig-Info: 0701234567
```



5.3 Examples

This Section contains some complete HTTP Request examples.

Example [11]

```
GET /t.jsp?first+name=Tom&last+name=Smith&MSISDN=0701234567 HTTP/1.1
Accept: text/*
Accept-Charset: iso-8859-1, UTF-8
Accept-Encoding: identity
User-Agent: WIG Browser/1.2 Gateway/6.1
Connection: Keep-Alive
Host: www.mywebserver.com
```

Example [12]

```
POST /mypage.jsp HTTP/1.1
Accept: text/*
Accept-Charset: iso-8859-1, UTF-8
Accept-Encoding: identity
User-Agent: WIG Browser/1.1 Gateway/6.1
Host: www.webserver.com
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 46
```

```
first+name=Tom&last+name=Smith&MSISDN=0701234567
```

Example [13]

```
GET /?VAR1=%03%23%12%EF HTTP/1.1
Accept: text/*
Accept-Charset: iso-8859-1, UTF-8
Accept-Encoding: identity
User-Agent: WIG Browser/1.0 Gateway/6.1
Connection: Keep-Alive
Host: www.smarttrust.com
X-Wig-Info: 46703213376
Cookie: name=Bob;city=Stockholm
```



6 HTTP Response

This Chapter describes the HTTP Response from the Content Provider to the UG (Step 3 in

Figure 1.)

The Response has the following structure:

```
Response = Status-Line   ; Section 6.1
           Headers       ; Section 6.2
CRLF
[ message-body ]
```

6.1 Status-Line

The Status Line has the following structure:

```
Status-Line = HTTP-Version SP Status-Code SP Reason-Phrase CRLF
```

Example [14]

```
HTTP/1.1 200 OK
```

6.1.1 Status-Code

UG supports the following values for the Status-Code:

- **200** OK. This response will be accepted. The UG will look at the headers *Content-length* and/or *Transfer-encoding: chunked* and receive the XML document in an appropriate way.
- **300** Multiple Choices. The most appropriate alternative is chosen and the request is resent. If no "redirection URL" is present, an error message will be sent to the mobile station.
- **301** Moved Permanently. An error will be logged. The request will be redirected to the "redirection URL" given in the response. If no "redirection URL" is present, an error message will be sent to the mobile station.
- **302** Found. The request will be redirected to the "redirection URL" given in the response. If no "redirection URL" is present, an error message will be sent to the mobile station.



- **303** See other. The request will be redirected to the "redirection URL" given in the response. If no "redirection URL" is present, an error message will be sent to the mobile station.
- **307** Moved Temporarily. The request will be redirected to the "redirection URL" given in the response. If no "redirection URL" is present, an error message will be sent to the mobile station.
- All other values of the Status-Code will cause an error message to be sent to the mobile station.

6.2 Headers

This section describes the headers that the UG uses.

- Cache-Control, Expires
- Connection
- Content-Length
- Content-Location
- Content-Type
- Location
- Set-Cookie
- X-WAP-Payment-Info
- Transfer-Encoding

6.2.1 Cache-Control and Expires

These two headers have to be present to make the UG cache the XML document. The Expires header, specify when the information contained in this response may change or become invalid. After that time, the document may change or be deleted.

Example [15]

```
Cache-control: public  
Expires: Thu, 04 Oct 2001 05:39:03 GMT
```

6.2.2 Connection

If set to close, the UG will always close the connection when receiving the response. If omitted or set to Keep-Alive, the UG will continue to use Keep Alive if it has been configured for it.

Example [16]

```
Connection: close
```



6.2.3 Content-Length

The `Content-Length` header specifies the length of the data (in bytes) of the transferred message-body. The `Content-length` has to be specified unless `Transfer-Encoding: chunked` is used.

6.2.4 Content-Location

The `Content-Location` header specifies the "redirection URLs" for HTTP 300 (See Section 6.1.1).

6.2.5 Content-Type

The following content-types are allowed by the UG

- `text/vnd.wap.wml`
- `text/wml`

Example [17]

```
Content-Type: text/wml
```

6.2.6 Location

The `Location` header specifies the "redirection URL" for HTTP 301-303 and 307 (See Section 6.1.1).

6.2.7 Set-Cookie

The `Set-Cookie` header allows the UG to store state information concerning a subscriber in the UG.

Cookies are supported by UG and handled according to RFC 2109, HTTP State Management Mechanism [2]. The UG also supports Netscape HTTP Cookies.

`Set-Cookie` consists of a couple of attributes described below:

```
Set-Cookie: name=value [;Options]
```

Options

Expires=date Specifies that the cookie will be invalid after the specified date. Either `expires` or the `Max-age` attribute is used. `expires` is included to support Netscape HTTP Cookies.

Max-age=*delta-seconds* Specifies the number of seconds the cookie is valid. Either `Max-age` or the `expires` attribute is used.

Path=*pathname* Specified for which URLs the cookie is valid.

Domain=*domain_name* Specifies for which domains the cookie is valid. Default value is the domain name of the Web Server.

If the response contains a cookie the path and domain attributes will be compared with the request-URI to decide if the cookie will be accepted.



The cookie is checked as is and compared with the request:

- The value of the Domain attribute has to contain embedded dots or start with a dot.
- The value of the Path attribute is a prefix of the request-URI.
- The request-host domain-match the Domain attribute.
- The request-host is an FQDN (Fully qualified domain name and not an IP address) and has the form HD, where D is the value of the Domain attribute, and H is a string that contains one or more dots. E.g. the domain attribute “strust.com” will match the request host name “aa.dp.strust.com”.

The cookie will be stored in the database and included in subsequent requests. See Section 5.2.5 . Expired cookies will be discarded and not forwarded to an origin server.

The cookies are valid as long as either the max-age or the expired attribute specifies. If none is included when the cookie is set the default value of one hour is used.

Example [18]

```
Set-Cookie: Entrykey=29377; Expires=Mon, 08-Oct-2001 12:52:23 GMT; Path=/
```

6.2.8 X-WAP-Payment-Info

The `X-WAP-Payment-Info` is an optional header and indicates the tariff class to be used for billing purposes. The value 0 is not allowed. See also *Guidelines – Development of Wib Services* [4].

This functionality of sending the tariff class can be used together with caching. The value will then be cached according to the same rules that applies to the WML document. See Section 6.2.1 .

Example [19]

```
X-WAP-Payment-Info: content-value-class=42
```

6.2.9 Transfer-Encoding

The `Transfer-Encoding` header with the value `chunked`, specifies that the following message-body is encoded as a series of chunks according to HTTP [3].

Example [20]

```
Transfer-Encoding: chunked
```

6.3 Examples

This Section contains some complete HTTP Response examples.



Example [21]

This example shows the use of the X-WAP-Payment-Info that specifies the tariff class.

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Date: Thu, 04 Oct 2001 05:39:03 GMT
Connection: close
Content-Type: text/wml; charset=ISO-8859-1
X-WAP-Payment-Info: content-value-class=23
Content-Length: 242

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
"http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
  <card>
    <p>
      This response uses a tariff class!
    </p>
  </card>
</wml>
```

Example [22]

This shows how cookie values are set. The cookie name/value pairs of information will be retained in the UG. It also shows how to specify that the connection shall be retained after retrieving the response since Connection: Keep-Alive.

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Date: Thu, 04 Oct 2001 15:40:07 GMT
Connection: Keep-Alive
Content-Type: text/wml; charset=ISO-8859-1
Set-Cookie: value=77; expires=Mon, 08-Oct-2001 12:52:23 GMT; path=/
Set-Cookie: name=bob; expires=Mon, 08-Oct-2001 12:52:23 GMT; path=/test
X-WAP-Payment-Info: content-value-class=12
Content-Length: 235

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
"http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
  <card>
    <p>
      This response sets a cookie!
    </p>
  </card>
</wml>
```



Example [23]

This shows the Expires header, used to specify when the information contained in the document may change or become invalid.

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Date: Thu, 04 Oct 2001 15:40:07 GMT
Connection: close
Content-Type: text/wml; charset=ISO-8859-1
Cache-control: public
Expires: Thu, 04 Oct 2001 05:39:03 GMT
Content-Length: 237
```

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE wml PUBLIC "-//SmartTrust//DTD WIG-WML 4.0//EN"
"http://www.smarttrust.com/DTD/WIG-WML4.0.dtd">
<wml>
  <card>
    <p>
      This response will be cached.
    </p>
  </card>
</wml>
```