Gigaset

Gigaset Phonebook Search

Protocol Specification

Version 1.4.02.PUB.2

This document is provided “AS IS” – NO WARRANTY FOR NOTHING

Contact info:

http://www.gigaset.com/customercare
http://www.gigaset-blog.com

Copyright © Gigaset Communications GmbH, Germany	
Author: Dept.: Name: Tel.:	**Inspector:**
File : Date: 31 May 2012	Status: Released File:

Content

List of examples

1. Introduction

1.1 Purpose of the Document

This document describes the protocol used in Gigaset VoIP phones (client) for searching in an externally hosted (on a server) public telephone book or yellow pages directory.

1.2 Overview of the Document

This document contains the following parts:

- Interface between client and server (authorization)
- Phonebook handling

The client device exchanges information data via HTTP1.1 / XML to the server. Message examples are given for better understanding.

2. Interface between Server and Client Devices

2.1 Basic protocol principles

It is important that client and server should use:

- HTTP/1.1 protocol
- ISO8859-1 charset in requests (client to server)
- UTF-8 charset in responses:

```
Content-Type: text/xml; charset=utf-8
…
<?xml version="1.0" encoding="UTF-8"?>
```

- For backward compatibility server must ignore unknown attributes in the requests and may not respond with error.

2.2 Authorization

For stand alone server authorization is done directly at the first request sent to the server. The server responds with “401 Unauthorized” and sends the authentication data to the client. The request has to be repeated, extended with authentication data (see examples).

In the following requests client can directly use these authentication data, until it has expired (after a fixed timeout, typically 1 hour). After that time server will again respond with “401 Unauthorized” and send new authentication data to the client.

The “User-Agent” field can be used to identify different client types.

```
GET http://www.phonebook.svr.dom/shcservlet.do?command=………HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
```

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate:Digest realm="phonebook.svr.dom",
   nonce="2356561841-9a875d65f56d5c6b45a6d5d56b45a5d4",
   qop="auth,auth-int"
```

```
GET http://www.phonebook.svr.dom/shcservlet.do?command=………HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username="phonebook@phonebook.svr.dom",
   realm="phonebook.svr.dom",
   nonce="2356561841-9a875d65f56d5c6b45a6d5d56b45a5d4",
   uri="/shcservlet.do?command=………",
   qop="auth”,nc="00000001”,cnonce="78F86D5A",
   response="565a6d5455f6a45c55b6d5a45f5c5b51",
   opaque=""
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
...
```

3. Generic Interface for Phonebook Handling

This chapter describes the interface between client and server for phonebook / yellow pages handling purposes.

3.1 Get List of Phonebook Entries

For phonebook search currently only search criteria are used. All used commands and parameters have to be written in lowercase letters.

3.1.1 Request Phonebook Entries

This chapter describes the protocol definition of the retrieval of a list of phonebook entries depending on one or more search criteria from the phonebook server.

For better understanding the protocol description is explained with specific examples.

The client sends a HTTP request to the phonebook server for retrieving the list by search criteria including the following parameters:

command:

This is the primary command for the request.

get_list	get list command

type:

The type identifies which type of list shall be retrieved from the server

pb	list of public phonebook entries
yp	list of yellow pages entries
pr	list of private phonebook entries

reqid: (optional)

is a hexadecimal string with a maximum length of 32 characters. It identifies unambiguously a list that is addressed by the request. The identifier is set by the server for identification of a list. Within the first request from client the **reqid** tag is empty or omitted. All subsequent requests to the list, due to several pages in the list, are addressed by the same **reqid** given by the server in the first request.

first:

is the first item in a request. This tag is used if the request is asked for item n till n+count-1.

Note: If the tag “first” is used, a following tag “last” is not allowed.

first=1	for the first request of a list the item should begin by 1.
first>0	for subsequent request beginning from a defined offset of items for a forward search request.

last:

is the last item in a request. This tag is used if the request is asked for item n till n-count+1. Nevertheless the items are sorted in ascending order. If the tag last is used, a following tag “first” is not allowed. In this case error id 2 (syntax error) will be returned from server.

last>0	item number for which the request starts backwards
last=-1	request starts from last item in the whole list backwards

count:

is the maximum amount of items which are requested from the server.

count>0	amount of items to return to the client. If the overall size of data exceeds the “limit” the returned amount of items are shortened at the “far end”(i.e. opposite the specified first/last). Only complete items are sent.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload).

limit>=1024	amount of data in bytes

reqsrc: (optional)

Request source is a flag that indicate who starts phonebook search:

auto	auto lookup (search is started for number lookup service)
user	manual search performed by user hands

If this parameter is missing, and server supports it, server should assume **“user”** request source.

lang: (optional)

Specify client user interface language (Is used for multi language countries to indicate to the service in which language it should respond)

0	undefined
1	US
2	German
3	English International
…	full language list is available at the end of document (chapter 3.7)

If no “lang” parameter is specified, server has to respond in native language.

mac: (optional)

This parameter defines device MAC address. It will be used internally by server provider for statistical purposes. MAC address is 6 byte length value represented by 12 hexadecimal digits without any special characters inside (see example in chapter 3.1.2).

aabbccddeeff	stands for MAC address aa:bb:cc:dd:ee:ff

id: (optional)

This is a string parameter with minimum length of 2 characters and maximum length of 32 characters. It identifies unambiguously one single entry that is requested from server, therefore only exact matches are possible. Only printable subset of ASCII character set is permitted (only chars from 0x20 to 0x7F).

With “id” parameter, there is only one valid character for search criteria: “*”. If any other characters in search criteria are specified, the server should respond with error code 2 “syntax error”.

If no entry can be found, the normal response should be generated with <total=”0”>. Unknown or not handled characters should generate error response with error code 4 “invalid” (for error codes see 3.3.6).

Value for “id” can be obtained from previous search request response (from <entry id> field). See example usage of this parameter in chapter 3.1.5.

Remark, that Id=-1 is reserved for the own entry. (See chapter 3.2.1 for details)

prid: (optional)

This is a numeric parameter which determines private phonebook number (when multiple private phone books are used). If server doesn’t support multiple private phonebooks, this parameter should be ignored by server. Value range of this parameter should be between 1 and 8.

For network traffic optimize and reduce response time, there is possibility to perform one request to many private phonebooks. In this case parameter “**prid**” can be used many times.

For more details please see chapter 3.3.6.

Search criteria

There are several search criteria available, depending on server capabilities. See chapter 3.6 (Search Criteria) for currently supported search criteria. Each key may contain a wildcard character ‘*’ represented by “%2a” in HTTP request, e.g. ct=Bo%2a, which will match all cities starting with Bo (Bocholt, Bochum, Borken,…). If the wildcard is omitted, only exact matches will be found (none in this case).

To reduce the amount of returned data, only keys submitted in the query will be returned in the response, not the full phonebook entry. So, if you need the phone number in your result (which is quite common for a phonebook query), explicitly add “hm=%2a” to your request.

Remark: *First*, *last* and *count* parameters don’t change order of search results. Results list is a “constant” list specified only by search criteria. *First*, *last* and *count* parameters specify only window through we looking at the result list.

The following examples describe how the command for retrieving a list by search criteria is applicable:

Example 1: (simple search request)

For this example following assumptions are met:

The command for the request of a list of phonebook entries is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom. The client is able to process 1024 bytes of data.

○ command:	get_list	
○ type:	pb	phone book
○ ct:	Bruck%2a	keyword for city to search, with wildcard
○ nn:	krak%2a	keyword for nickname to search, with wildcard
○ hm:	%2a	home phone number is to be returned in the result
○ first:	1	first item to search for
○ count:	10	get 10 items
○ limit:	1024	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ct=Bruck%2a&nn=krak%2a&hm=%2a&reqid=&first=1&count=10&li
mit=1024  HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2(Authorization).

Example 2: (request for cities)

For the second example following assumptions are met:

The command for the request of a list of phonebook entries is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom. The client is able to process 1024 bytes of data.

○ command:	get_list	
○ type:	pb	phone book
○ ct:	Bruck%2a	search for all cities starting with “Bruck”
○ first:	1	first item to search for
○ count:	10	get 10 items
○ limit:	1024	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ct=Bruck%2a&reqid=&first=1&count=10&limit=1024 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
```

```
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2(Authorization).

Example 3: (usage of *last* parameters)

Following search criteria will generate list of 16 entries (out of 512 total) from entry 185 to entry 200:

o command:	get_list	
o type:	pb	phone book
o ln	Han%2a	search for persons with last name starting with “Han”
o ct:	Stolpen	search for city “Stolpen”
o last:	200	last item to search for
o count:	16	get 16 items
o limit:	2048	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ln=Han%2a&ct=Stolpen&hm=%2a&last=200&count=16&limit=2048
HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

To display 20 last results, the parameter last=-1 should will be used.
Example below returns 20 entries (from 14 to 32 out of 32 totals).

o command:	get_list	
o type:	pb	phone book
o ln	Hans	search for last name starting with “Hans”
o ct:	Stolpen	search for city “Stolpen”
o last:	-1	last item to search for
o count:	20	get 20 items
o limit:	2048	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ln=Hans&ct=Stolpen&hm=%2a&last=-1&count=20&limit=2048
HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

Example 4: (Yellow Pages search)

For the example following assumptions are met:

○ command:	get_list	
○ type:	yp	Yellow Pages search
○ wh:	Pizza*	search for all entries with starting with “Pizza”
○ ct:	Dortmund	search for city “Dortmund”
○ st:	*	want to know Street
○ ln:	*	want to know Last Name
○ cat:	*	want to know Category
○ nr:	*	want to know House Number
○ hm:	*	want to know Home Phone Number
○ mb:	*	want to know Mobile Number
○ first:	1	first item to search for
○ count:	16	get 16 items
○ limit:	2048	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=yp&wh=Pizza*&ct=Dortmund&st=*&ln=*&cat=*&nr=*&hm=*&mb=*&fir
st=1&count=16&limit=2048 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2(Authorization).

Response for this request is described in chapter 3.1.6 (Response for the Yellow Pages Entry request)

3.1.2 Reverse search for Phonebook entries (optional)

This chapter describes search by telephone number criteria. Request syntax is identical to described in chapter 3.1.1 (Request Phonebook Entries) but new search parameter should be used:

reqsrc:

Request source is a flag that indicate who starts phonebook search:

auto	auto lookup (search is started for name replacement service)
user	manual search performed by user hands

If this parameter is missing, and server supports it, server should assume **"user"** request source.

lang:

Specify client user interface language (Is used for multi language countries to indicate to the service in which language it should respond)

0	undefined
1	US
2	German
3	English International
…	full language list is available at the end of document (chapter 3.7)

If no "lang" parameter is specified, server has to respond in native language

Example 5: **(search by telephone number)**

For this example following assumptions are met:

command:	get_list	
type:	pb	phone book
hm:	4935912345	home phone number
nn:	*	want to know nick name
fn:	*	want to know first name
ln	*	want to know last name
first:	1	first item to search for
count:	1	get only 1 result
limit:	2048	maximum amount of data for response
reqsrc:	auto	this request is performed automatically by number lookup service
lang:	3	inform server to respond in English
mac	00:CC:CC:AC:8B:F6	MAC address of client device

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&nn=*&fn=*&ln=*&ct=*&hm=4955512345&first=1&count=1&limit=
2048&reqsrc=auto&lang=3&mac=00ccccac8bf6 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2(Authorization).

3.1.3 Response for the Phonebook Entry Request

The client receives a HTTP response from the server in succession of a request as defined in the previous chapter.

The response consists of a header part and a data part:

- The header part contains the general status information about the list.
- The data part contains the information structure depending on the list type sent in the request.

The HTTP response contains the following status information in the header part:

response:

This is the response to the primary command for the request.

get_list	get list

type:

identifies which type of list shall be retrieved from the server

pb	list of public phonebook entries
yp	list of yellow pages entries
pr	list of private phonebook entries

reqid: (optional)

is a hexadecimal string with a maximum length of 32 characters. It identifies unambiguously a list that is addressed by the request. The identifier is set by the server for identification of a list. Within the first request from client the **reqid** tag is empty or omitted. All subsequent requests to the list, due to several pages in the list, are addressed by the same **reqid** given by the server in the first request.

notfound: (optional)

If an empty list is returned, the server may use **notfound** to provide more information which search criteria failed. E.g. **notfound=”ct”** means that no cities with the given search string exist in the database. This information is optional and not provided by all servers. Currently the phone will only use the **notfound=”ct”** indication and shows dedicated error message in this case.

total:

is the overall number of items in the list. An item is for instance the collection of data describing a phone book entry.

first:

The number of the first item in the current page, based on the request

last:

The number of the last item in the current page. If the response contains the last page of the list the **last** tag is set to the last item in the result list.

<u>The data part of the response for a phonebook entry request consists of the data fields which were part of the request line:</u>

entry id: (optional)

Each entry is marked by unique entry id (for details see chapter 3.1). This ID can be used as reference in any future communication with server such as: add, edit and modify entry in private phonebook or modify own information in public phonebook.

> <u>Note:</u> Unique ID is mandatory if private pages are supported (see chapter 3.3.2). Otherwise entries can’t be copied from public phonebook to private phonebook.

Example 6: (response on phonebook search request)

For the first example following assumptions are met:

The response for a phonebook entry request is generated and sent by the server http://www.phonebook.svr.dom.

o	response:	get_list	(response for get list)
o	type:	pb	phonebook
o	reqid:	22AB2	list identifier
o	total:	3	total number of items in the list
o	first:	1	number of first item in current page
o	last:	3	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”22AB2” total=”3” first=”1”
   last=”3”>
   <entry id="52783">
      <ln>Mayer</ln>
      <fn>Alexander</fn>
      <hm>+431255512345</hm>
   </entry>
   <entry id="52793">
      <ln>Mayer </ln>
      <fn>Martina</fn>
      <hm>+435170743123</hm>
      <mb>+436648899123</mb>
      <fx>+435170752123</fx>
   </entry>
   <entry id="52805">
      <ln>Mayer-Hartl </ln>
      <fn>Wolfgang</fn>
      <hm>+435170712345</hm>
      <mb>+436769812345</mb>
      <fx>+435170712345</fx>
   </entry>
</list>
```

Note: Empty fields (e.g. “mb” and “fx” for entry 52783 in above example) will not be listed to save memory!

Example 7: (response on a phonebook search request with only “ct” information)

For the second example following assumptions are met:

The response for a phonebook entry request is generated and sent by the server http://www.phonebook.svr.dom.

response:	get_list	(response for get list)
type:	pb	phonebook
reqid:	2DF82	list identifier
total:	3	total number of items in the list
first:	1	number of first item in current page
last:	3	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”2DF82” total=”3” first=”1”
   last=”3”>
   <entry>
      <ct>Bruck an der Leitha</ct>
   </entry>
   <entry>
      <ct>Bruck an der Mur</ct>
   </entry>
   <entry>
      <ct>Bruckneudorf</ct>
   </entry>
</list>
```

Note, that in this case no entry ids are given, as the result (the cities) cannot be unambiguous assigned to a single entry.

Example 8: (response on a phonebook search request with *notfound* information)

The client sent a HTTP request to the phonebook server for retrieving the list of phonebook entries by search criteria including the following parameters:

command:	get_list	
type:	pb	public phone book
ct:	Nünchen	search for all cities starting with “Nünchen”
ln:	Mei*	search for all names starting with “Mei”
first:	1	first item to search for
count:	10	get 10 items
limit:	1024	maximum amount of data for response

As the city was misspelled in the request, the server will not find any corresponding entries. The server has now the possibility to inform the client that he can’t find the city entry by using the **notfound** parameter:

o response:	get_list	(response for get list)
o type:	pb	public phonebook
o reqid:	29AF5	list identifier
o notfound:	ct	city could not be found
o total:	0	total number of items in the list

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”29AF5” notfound=”ct”
   total=”0”/>
```

Note: This additional information is optional and not supported by all servers.

Example 9: (response on phonebook search request)

For the example following assumptions are met:

The response for a phonebook entry request is generated and sent by the phonebook server http://www.phonebook.svr.dom.

response:	get_list	(response for get list)
type:	pb	phonebook
reqid:	42632B	list identifier
total:	2	total number of items in the list
first:	1	number of first item in current page
last:	2	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”42632B” total=”2” first=”1”
   last=”2”>
   <entry id="12783">
      <ct>Bruck an der Leitha</ct>
      <nn>kraker</nn>
      <hm>+43216264123</hm>
   </entry>
   <entry id="18723">
      <ct>Bruck an der Mur</ct>
      <nn>kraki</nn>
      <hm>+43386259123/hm>
   </entry>
</list>
```

Note: Search results are sorted alphabetically by nickname.

3.1.4 Response for the Phonebook Entry Request Using Ambiguity Mechanism

Servers may also use ambiguity mechanism if a given search is not unambiguous.

For better understanding this is explained with a specific example:

Example 10: (response on phonebook search request using ambiguity mechanism)

The client sends a HTTP request to the phonebook server for retrieving the list of phonebook entries by search criteria including the following parameters:

○ command:	get_list	
○ type:	pb	public phone book
○ ct:	Boch*	search for all cities starting with “Boch”
○ ln:	Mei*	search for all names starting with “Mei”
○ first:	1	first item to search for
○ count:	10	get 10 items
○ limit:	1024	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ct=Boch*&ln=Mei*&reqid=&first=1&count=10&limit=1024
HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2(Authorization).

The server recognizes that the city is not unambigues and sends a HTTP response describing the ambiguity. This response contains the following status information in the header part:

response:

This is the response to the primary command for the request.

get_list	get list

type:

identifies which type of list shall be retrieved from the server

pb	list of public phonebook entries

reqid: (optional)

is a hexadecimal string with a maximum length of 32 characters. It identifies unambiguously a list that is addressed by the request. The identifier is set by the server for identification of a list. Within the first request from client the **reqid** tag is empty or omitted. All subsequent requests to the list, due to several pages in the list, are addressed by the same **reqid** given by the server in the first request.

ambiguity:

request was ambiguous, the given string identifies the ambiguous search criteria. The following parameters (total, first, last, entries) reference to the ambiguity list instead of the result list.

total:

The overall number of items in the ambiguity list.

first:

The number of the first item in the current page, based on the request.

last:

The number of the last item in the current page.

Example 11: (response on the previous phonebook search request)

Following assumptions are met:

response:	get_list	(response for get list)
type:	pb	public phonebook
reqid:	27AF5	list identifier
ambiguity:	ct	request was ambiguous, deliver ambiguity list of cities
total:	16	total number of items in the ambiguity list
first:	1	number of first item in current page
last:	10	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”27AF5” ambiguity=”ct”
   total=”16” first=”1” last=”10”>
   <entry>
        <ct>Bochheim</ct>
   </entry>
   <entry>
        <ct>Bochhold</ct>
   </entry>
   <entry>
        <ct>Bochin</ct>
   </entry>
   <entry>
        <ct>Bochingen</ct>
   </entry>
   <entry>
        <ct>Bochold</ct>
   </entry>
   <entry>
        <ct>Bocholt</ct>
   </entry>
   .
   .
   .
   <entry>
        <ct>Bochum</ct>
   </entry>
</list>
```

Notes:

1. For getting the next page of the ambiguity list the client has to repeat the original request with changed first or last parameter.
2. For getting the result list the client has to choose one entry of the ambiguity list as replacement of the ambiguous parameter in the original request and repeat the request with this new value.

3.1.5 Response for the Phonebook full entry request

Example 12: (full entry request)

For the example following assumptions are met:

○ command:	get_list	
○ type:	pb	Phonebook search
○ id:	000a12f500M00H01	get all data for this entry id
○ fn:	*	get first name
○ ln:	*	get last name
○ ct:	*	get city name
○ st:	*	get street name
○ nr:	*	get home number
○ hm:	*	get home phone number
○ mb:	*	get mobile phone number
○ limit:	2048	maximum amount of data for response
○ first:	1	only 1 entry expected
○ count:	1	only 1 entry expected
○ reqsrc:	user	this request is performed by user
○ lang:	3	inform server to respond in English
○ mac	00:CC:CC:AC:8B:F6	MAC address of client device

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&id=000a12f500M00H01&fn=*&ln=*&ct=*&st=*&nr=*&hm=*&mb=*&l
imit=2048&first=1&count=1&reqsrc=user&lang=3&mac=00CCCCac8bf6 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2 (Authorization).

Example 13: (full entry response)

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” total=”1” first=”1” last=”1”
   reqid=”5DC4”>
   <entry id="000a12f500M00H01">
        <fn>Franz</fn>
        <ln>Mayer</ln>
        <ct>Dortmund</ct>
        <st>Grossenstr.</st>
        <nr>28</nr>
        <hm>+492316185123</hm>
        <mb>+495551234567</mb>
   </entry>
</list>
```

Note: Empty fields (e.g. “fx”) will not be listed to save memory!

3.1.6 Response for the Yellow Pages Entry request

Please note: For Yellow Pages search, client asks for data by search criteria “**wh=**”, but server responds in “**ln**” and **“cat**” data fields. Therefore the fields “**ln=***” and “**cat=***” were added to request.

Example 14: (response on a Yellow Pages search request)

For the example following assumptions are met:

○ command:	get_list	
○ type:	yp	Yellow Pages search
○ **wh:**	**Pizza***	search for all entries with starting with “Pizza” (Depending on server implementation, this may also return all entries containing the word “Pizza”)
○ ct:	Dortmund	search for city “Dortmund”
○ st:	*	want to know Street
○ **ln:**	*	want to know Last Name
○ **cat:**	*	want to know Category
○ nr:	*	want to know House Number
○ hm:	*	want to know Home Phone Number
○ mb:	*	want to know Mobile Number
○ first:	1	first item to search for
○ count:	16	get 16 items
○ limit:	2048	maximum amount of data for response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”yp” total=”90” first=”1” last=”16”
   reqid=”5DC2”>
   <entry id="12783">
        <ln>Adria</ln>
        <ct>Dortmund</ct>
        <cat>Gaststätten u. Restaurants - Italienisch</cat>
        <st>Grossenstr.</st>
        <nr>28</nr>
        <hm>+492316181234</hm>
   </entry>
   …
   <entry>
        …
   </entry>
</list>
```

Note: Empty fields (e.g. “mb” in above example) will not be listed to save memory!

3.1.7 Local search feature (optional)

Some provider specific features are available only for registered and authorized users. During registration to provider (e.g. from website), user can specify his personal details and among other things can also specify zip code and city name.

In this case there is possibility to instruct server to perform search only in user own town. This can be useful for fast searching in Yellow Pages for entries like Pizzas, Post Office etc, without entering city name and resolve ambiguities in city names. The server then also can sort the results found by the distance between the home of the user and the location found.

As the server must be able to identify the user sending the request this feature makes only sense if the phone uses a user specific credential set which the user got during registration procedure on the provider's website. The credential set can be entered in the phone's user interface by the user.

If the local search is used without a user specific credential set, the server should response with a zero result.

To perform above search, parameter "ct=at home" must be set.

Example 15: (request and response for an YP search with local search feature)

For the example following assumptions are met:

○ command:	get_list	
○ type:	yp	Yellow Pages search
○ wh:	Pizza*	search for all entries with starting with "Pizza" (Depending on server implementation, this may also return all entries containing the word "Pizza")
○ **ct:**	**at**%20**home**	search in own city
○ st:	*	want to know Street
○ ln:	*	want to know Last Name
○ cat:	*	want to know Category
○ nr:	*	want to know House Number
○ hm:	*	want to know Home Phone Number
○ mb:	*	want to know Mobile Number
○ first:	1	first item to search for
○ count:	16	get 16 items
○ limit:	2048	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=yp&wh=Pizza*&ct=at%20home&st=*&ln=*&cat=*&nr=*&hm=*&mb=*&fi
rst=1&count=16&limit=2048 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part at the end of request definition is described in chapter 2.2 (Authorization).

Response for this request can be identical as in Example 14: (response on a Yellow Pages search request).

Note: response contain <ct> field with real city name (not “at home”).

3.1.8 Search with presence information (optional)

This chapter describes the possibility to extend the request phonebook entries procedure by the presence info. The presence info is used to indicate the status of a user. This status is managed by each user for its own entry (see xxx).

The request syntax is identical to described in chapter 3.1.1 (Request Phonebook Entries) but new search parameter should be used:

prs:

Search criteria for presence. In request message only allowed value is prs=*.

Example 16: (simple search request with presence)

For this example following assumptions are met:

The command for the request of a list of phonebook entries is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom. The client is able to process 1024 bytes of data.

command:	get_list	
type:	pb	phone book
ct:	Bruck%2a	keyword for city to search, with wildcard
nn:	krak%2a	keyword for nickname to search, with wildcard
hm:	%2a	home phone number is to be returned in the result
prs:	%2a	request presence info if available
first:	1	first item to search for
count:	10	get 10 items
limit:	1024	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ct=Bruck*&nn=krak*&hm=*&prs=*&reqid=&first=1&count=10&li
mit=1024  HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

Example 17: (response on phonebook search request)

For the example following assumptions are met:

The response for a phonebook entry request is generated and sent by the phonebook server http://www.phonebook.svr.dom.

o response:	get_list	(response for get list)
o type:	pb	phonebook
o reqid:	42632B	list identifier
o total:	4	total number of items in the list
o first:	1	number of first item in current page
o last:	4	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”42632B” total=”4” first=”1”
   last=”4”>
   <entry id="12783">
      <ct>Bruck an der Leitha</ct>
      <nn>kraker</nn>
      <hm>+43216264123</hm>
      <prs>idle</prs>
   </entry>
   <entry id="18723">
      <ct>Bruck an der Mur</ct>
      <nn>kraki</nn>
      <hm>+43216264323/hm>
      <prs>busy</prs>
   </entry>
   <entry id="13533">
      <ct>Bruck an der Mur</ct>
      <nn>krako</nn>
      <hm>+43216264521/hm>
      <prs>absent</prs>
   </entry>
   <entry id="34723">
      <ct>Bruck an der Mur</ct>
      <nn>kraks</nn>
      <hm>+43216275223/hm>
      <prs>hidden</prs>
   </entry>

</list>
```

The used values for search category field presence are in general not defined and can be used freely. But the following values are supported by standard handsets:

prs:

Default values supported by default handsets and their graphical representation

idle	user is ready to be called (green pawn)
busy	user is busy (red pawn)
dnd	do not disturb (red pawn)
absent	absent (red pawn)
hidden	use does not want to populate its presence state (no pawn)

3.1.9 Search Picture CLIP information (optional)

To retrieve the info whether a picture clip is available at server side, the request phonebook entries procedure is extended as follows.

The request syntax is identical to described in chapter 3.1.1 (Request Phonebook Entries) but new search parameter should be used:

apc:

identifies whether **a**vailability of **p**icture **cl**ip is requested or if clip picture is available (response)

yes	picture clip available
no	no picture clip available, used in response only
*****	search criteria for picture clip availability, used in request only

Example 18: (simple search request with picture clip availability info)

For this example following assumptions are met:

The command for the request of a list of phonebook entries is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom. The client is able to process 1024 bytes of data.

command:	get_list	
type:	pb	phone book
ct:	Bruck%2a	keyword for city to search, with wildcard
nn:	krak%2a	keyword for nickname to search, with wildcard
hm:	%2a	home phone number is to be returned in the result
apc:	%2a	request info about availability of picture clip
first:	1	first item to search for
count:	10	get 10 items
limit:	1024	maximum amount of data for response

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pb&ct=Bruck*&nn=krak*&hm=*&apc=*&reqid=&first=1&count=10&li
mit=1024  HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

Example 19: (response on phonebook search request)

For the example following assumptions are met:

The response for a phonebook entry request is generated and sent by the phonebook server http://www.phonebook.svr.dom.

response:	get_list	(response for get list)
type:	pb	phonebook
reqid:	42632B	list identifier
total:	2	total number of items in the list
first:	1	number of first item in current page
last:	2	number of last item in current page

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pb” reqid=”42632B” total=”2” first=”1”
   last=”2”>
   <entry id="12783">
      <ct>Bruck an der Leitha</ct>
      <nn>kraker</nn>
      <hm>+43216264123</hm>
      <apc>yes</apc>
   </entry>
   <entry id="18723">
      <ct>Bruck an der Mur</ct>
      <nn>kraki</nn>
      <hm>+43216264323/hm>
      <apc>no</apc>
   </entry>
</list>
```

Note: Instead of sending “<apc>no</apc>” its is also allowed to leave out the attribute.

The procedure to retrieve the picture clip is defined by 3.4.1.

3.2 *Change Phonebook Entry (optional)*

3.2.1 Request Change Phonebook

The users may be able to change their own entry. All used commands and parameters have to be written in lowercase letters.

command:

This is the primary command for the request.

chg_data	change data command

type:

The type identifies which type of list shall be changed on the server

pb	phonebook
yp	yellow pages
pr	private phonebook

id:

This is the entry id, which needs to be requested in a previous get_list request. The entry id = -1 is optional and can be used to address the own entry.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload). If omitted, a limit of 1024 bytes is assumed.

limit>=1024	amount of data in bytes

Data to change

Currently only nn (nickname) data field is available for changing. From protocol point of view all data fields (except entry id) can be changed.

Example 20: (changing phonebook entry)

For the example following assumptions are met:

The command for changing an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

○ command:	chg_data	
○ type:	pb	phonebook
○ nn:	“mynewnickname”	user wants to set a new nickname
○ id:	52783	id of the entry which shall be changed
○ limit	1024	optional, max. size of response in bytes

```
POST http://www.phonebook.svr.dom/phonebookservlet.do?command=
chg_data&type=pb&id=52783&limit=1024 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…

<?xml version="1.0" encoding="UTF-8"?>

<data>
        <nn>mynewnickname</nn>
</data>
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

The server has to check whether the nickname is already in use and return an appropriate error message or OK (see chapter 3.3.6).

3.2.2 Request Change Presence State

The presence of the own entry can be changed by the procedure described in this chapter.

command:

This is the primary command for the request.

chg_data	change data command

type:

The type identifies which type of list shall be changed on the server

pb	phonebook
yp	yellow pages
pr	private phonebook

id:

This is the entry id, which needs to be requested in a previous get_list request. The entry id = -1 is optional and can be used to address the own entry.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload). If omitted, a limit of 1024 bytes is assumed.

limit>=1024	amount of data in bytes

Data to change

prs (presence) data field is used to change the presence state of a given entry. If prs is equal to “hidden”, the server must not add the presence state to the response for a request phonebook entry procedure.

Example 21: (changing own presence state)

For the example following assumptions are met:

The command for changing an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

○ command:	chg_data	
○ type:	pb	phonebook
○ prs:	“absent”	user wants to set a presence state
○ id:	52783	id of the entry which shall be changed
○ limit	1024	optional, max. size of response in bytes

```
POST http://www.phonebook.svr.dom/phonebookservlet.do?command=
chg_data&type=pb&id=52783&limit=1024 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…

<?xml version="1.0" encoding="UTF-8"?>

<data>
        <prs>absent</prs>
</data>
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

Note: Default values for presence state are listed in chapter 3.1.8.

3.3 Manage private phonebook (optional)

3.3.1 Introduction

The private phone book in the network has similar functionality as the local phone book in the device. A user can browse the entries and can jump to entries by pressing a letter on the keyboard. It is possible to copy an entry from the public databases (Yellow and White Pages) and user can modify/all/delete entries using the phone or the PC (in case a web site based access to his private phone book is implemented). PC access is highly recommended as it makes tasks for managing entries much easier.

As an additional user benefit the server can store direct references from the private phone book to the public database in order to update/maintain and extend the entries in the private phone book.

As mentioned already above, the access to the private phone book is different to the access to public phone books. In case of private phone book the phone does not perform a detailed search query by providing name, city etc. Instead the phone always issues "*" searches and thus requests a list of all entries. In addition the phone can provide a character string ("set_first" attribute) which contains the starting letter(s) which define the entry point in the list. In other words the content of "set_first" advises the server where to set the "first" pointer and with it the start of the result list is given back to the phone. Still it will possible for the user to scroll up and down, across the entire list.

3.3.2 Adding entry

All used commands and parameters have to be written in lowercase letters.

command:

This is the primary command for the request.

copy_data	copy entry specified by "id" from public phone book to private phone book. Note: Because these command copy already existing data, only <nn> (nickname) data field will be processed. If <nn> is empty, or no <data> section is passed, assume that no nick name is specified.
add_data	add new entry to private phone book

type:

The type identifies which type of list shall be changed on the server

pr	list of private phonebook entries

id: (optional - valid only for "copy_data" command)

This is the entry id, which needs to be requested in a previous "get_list" request. For a new entry the ID is not known yet, therefore this parameter should be ignored when used together with "add_data" command.

Note: The features "copy an entry from public database" and "modify an entry" makes the usage of an **ID** mandatory. Furthermore the ID must be unique over the entire database (public and private) in order to allow the server to distinguish between the different entries in the private phone book.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload). If omitted, a limit of 1024 bytes is assumed.

Limit>=1024	amount of data in bytes

Data to add

From protocol point of view all data fields (except entry id) can be added. Not supported data fields should be ignored by server.

Example 22: (copy an entry to private phone book)

For the example following assumptions are met:

The command for copy an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

○ command:	copy_data	
○ type:	pr	private phonebook
○ nn:	"mynewnickname"	user wants to set a new nickname
○ id:	52783	id of the entry which shall be changed
○ limit	1024	optional, max. size of response in bytes

```
POST http://www.phonebook.svr.dom/phonebookservlet.do?command=
copy_data&type=pr&id=52783&limit=1024 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…

<?xml version="1.0" encoding="UTF-8"?>
<data>
        <nn>mynewnickname</nn>
</data>
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

The server has to check whether the nickname is already in use (in scope of private phonebook only) and return an appropriate error message or OK (see chapter 3.3.6).

3.3.3 Editing entry

Editing for nickname is done in the same way as changing own nickname in public phonebook (chapter 3.2.1) with only one difference that **"type=pr"**.

3.3.4 Deleting entry

All used commands and parameters have to be written in lowercase letters.

command:

This is the primary command for the request.

del_data	delete data command

type:

The type identifies which type of list shall be changed on the server

pr	private phonebook entries

id:

This is the entry id, which needs to be requested in a previous "get_list" request.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload). If omitted, a limit of 1024 bytes is assumed.

limit>=1024	amount of data in bytes

Data

There is no data required for this request. All data should be ignored by server.

Example 23: (delete entry from private phonebook)

For the example following assumptions are met:

The command for deleting an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

○ command:	del_data	
○ type:	pr	private phonebook
○ id:	52783	id of the entry which shall be deleted
○ limit	1024	optional, max. size of response in bytes

```
POST http://www.phonebook.svr.dom/phonebookservlet.do?command=
del_data&type=pr&id=52783&limit=1024 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

The server has to check for existence of entry id in private phonebook and return errorid=1 (no allowed) or OK (see chapter 3.3.6).

3.3.5 Scrolling across results by first letters

3.3.5.1 Introduction

Because private book is generally small book and should be accessed as fast as possible, there is no sense to ask user for search parameters (e.g. name/city). It is possible to get all entries from private phone book in one search request, simply by specify only "*" in search criteria.

A user doesn't have to scroll through all the names in his private phone book, just press the first letters of the name and the cursor will jump to that letter.

If the private phone book does not contain any entries with desired characters, the "nearest" available entry is shown. Algorithm for choosing "nearest" entry depends on provider's search engine.

To achieve above functionality new request parameter should be used:

set_first:

This is a string parameter with maximum length of 32 characters. This parameter specifies that the first item in a result should be start with those letter(s). Currently phone will send only lower case letters, but server should support all characters:

set_first=h	first returned result in request should begin from letter "h" or "H".

Note: "Set_first" is not criteria or filter parameter, this parameter causes only "scroll" of results to specified position. This tag should be used alternative to "first" and together with "count" (see 3.1.1). If the tag "set_first" is used, a following tag "last" or "first" is not allowed and server will respond with error code 2 "syntax error".

Response from server has to include "total", "first" and "last" fields to determine unambiguous current list position. These fields are required to provide correct requests in future (for example: continuous list scrolling by up/down key).

3.3.5.2 Server side sorting algorithm

Some entries in private phone book may contain incomplete data e.g.: only last name or only nick name. In this case server has to sort these entries correctly, using own sorting mechanism (see proposal below). For correct data handling by client, server returns, for each entry, additional attribute "**sc**" (sorting criteria). This attribute indicates data field which was used for sorting by server. More details you can find in Example 25: (response for request with scroll by first letters).

Proposal:

Server sorts the entries by nickname, first name or last name. Priority order for sorting the entries in the list is:

1. use the nickname for sorting (**<nn>**)
2. if no nickname is present, use the last name (**<ln>**)
3. if neither nickname nor last name available then use the first name (**<fn>**)

In the result list the client (phone) will only show the field which was marked by the server as sorting criteria. With one exception: if server indicates last name, and first name is not empty, then first name should be also displayed after comma (see table below).

Let assume that private phone book contain following entries:

Representation on phone display	**Unsorted private phone book entries**			
	Nick name	**First name**	**Last name**	**Cat**
Grandma	Grandma	Cindy	Miller	
Miller, Cindy		Cindy	Miller	
Miller			Miller	
Cindy		Cindy		
Dingo's Pizza			Dingo's Pizza	restaurant

Server should return all entries sorted by above algorithm:

Entries order on display
Cindy Dingo's Pizza Grandma Miller Miller, Cindy

Note: in above example there is no rule for determine order of **"Miller"** and **"Miller, Cindy"** entries, because from server side only last name is used for sorting.

3.3.5.3 Examples

Example 24: (request with scroll by first letters)

For the example following assumptions are met:

The command for deleting an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

o	command:	get_list	
o	type:	pr	private phonebook
o	nn:	*	include nickname field in results
o	ln:	*	include lastname field in results
o	hm:	*	include phone number (home) field in results
o	set_first:	jo	display results from first item started with “jo” letters (case insensitive)
o	count:	10	get 10 items at time
o	limit	2048	optional, max. size of response in bytes

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pr&nn=*&ln=*&hm=*&set_first=jo&count=10&limit=2048 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

Example 25: (response for request with scroll by first letters)

This is response for request parameters from above Example 24:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pr” total=”23” first=”17” last=”23”
   reqid=”5DC2”>
   <entry id="a83" sc=”nn”>
        <nn>Johny</nn>
        <ln>Bluhmen</ln>
        <hm>+492316181234</hm>
   </entry>
   <entry id="ae3f" sc=”ln”>
        <ln>John</ln>
        <fn>Wernerin</fn>
        <hm>+49865211234</hm>
   </entry>
   …
   <entry>
        …
   </entry>
</list>
```

Note: “Total”, “first” and “last” are included in response. This is required for usage with manual scrolling of the list.

Note: In this case server uses **<nn>** field for sorting. For second entry, **<nn>** not exists, therefore server uses **<ln>** for this entry.

3.3.6 Multiple private phonebooks (optional)

In common usage scenario, there is more users of the same device (One basestation device with multiple handsets). Using the same private phonebook for all may be uncomfortable, thus there is extension which introduce multiple private phonebooks for the same credentials (device is using still one account to access provider database).

Request for selected private phone book is normal request with additional search parameter “**prid**” which specify private phone book number (for complete list of parameters with detailed description please see chapter 3.1). If no “**prid**” parameter is specified, the server should treat request as request to private phone book number 1.

Response from server must contain “**prid**” attribute for each <entry> section to inform client from which private phone book entry comes (for more information about usage of this attribute please see at …).

Example 26: (request for specified private phone book)

For the example following assumptions are met:

o	command:	get_list	
o	type:	pr	private phonebook
o	nn:	*	include nickname field in results
o	ln:	*	include lastname field in results
o	hm:	*	include phone number (home) field in results
o	**prid:**	**2**	get data from 2nd private phonebook
o	count:	10	get 10 items at time
o	limit	2048	optional, max. size of response in bytes

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pr&nn=*&ln=*&hm=*&prid=2&count=10&limit=2048 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

Example 27: (response for request to specified private phone book)

This is response for request parameters from above Example 26:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pr” total=”23” first=”1” last=”10”
   reqid=”5DC2”>
   <entry id="a83" sc=”nn” prid=”2”>
        <nn>Johny</nn>
        <ln>Bluhmen</ln>
        <hm>+492316181234</hm>
   </entry>
   <entry id="ae3f" sc=”ln” prid=”2”>
        <ln>John</ln>
        <fn>Wernerin</fn>
        <hm>+49865211234</hm>
   </entry>
   …
   <entry>
        …
   </entry>
</list>
```

3.3.6.1 Reverse search in multiple private phone books

Reverse search (search by phone number) can be done in the similar way to described in 3.1.2. To reduce network usage, there is possibility to specify more than one "**prid**" in one request (as described in 3.1).

If request is performed by automatic number lookup service (search parameter: "**reqsrc=auto**"), the server will start the search in the private phone books and if this search does not find a result for one or more **prid**'s it will perform a search in the public phone book.

Results from private phone books should be marked with respective "**prid**" in each <entry> section. Results from public directory cannot be marked with "**prid**".

Example 28: (reverse search in multiple private phone books)

For the example following assumptions are met:

○ command:	get_list	
○ type:	pr	private phonebook
○ nn:	*	include nickname field in results
○ ln:	*	include lastname field in results
○ hm:	49865211234	home phone number
○ **prid:**	**1**	get data from 1st private phonebook
○ **prid:**	**2**	get data from 2nd private phonebook
○ **prid:**	**6**	get data from 6th private phonebook
○ **reqsrc:**	**auto**	this request is performed automatically by number lookup service
○ count:	10	get 10 items at time
○ limit	2048	optional, max. size of response in bytes

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pr&nn=*&ln=*&hm=4935912345&prid=1&prid=2&prid=6&reqsrc=auto
&count=10&limit=2048 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

Example 29: (response for automatic lookup from multiple private phonebooks)

This is response for request parameters from above Example 28:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pr” total=”23” first=”1” last=”10”
   reqid=”5DC2”>
   <entry id="a8d3" sc=”nn” prid=”1”>
        <nn>Johny</nn>
        <hm>+49865211234</hm>
   </entry>
   <entry id="1e2fca" sc=”ln” prid=”2”>
        <ln>Johny</ln>
        <fn>Wernerin</fn>
        <hm>+49865211234</hm>
   </entry>
   <entry id="ffaa3b5c">
        <nn>John</nn>
        <ln>Wernerin</ln>
        <hm>+49865211234</hm>
   </entry>
</list>
```

Note: For first and second entry, results are found in corresponding private phone books (there are “**prid”** marks). For third entry, result was not found in private phone book, so there is no “prid” mark, and data from public phone book are returned.

3.3.7 AES encryption extension (optional)

AES (Advanced Encryption Standard) is a block cipher adopted as an encryption standard by the US government. Because AES is a symmetric algorithm, it uses the same key for encoding and decoding. Also the key is always the same size as block to be processed. Many more details are available on Wikipedia under following URL: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard.

The client (phone device) uses the standard implementation of AES in ECB (Electronic codebook) mode, with block and key sizes of 16 bytes (AES-128). Other block and key sizes are not supported. The message is divided into blocks and each block is encrypted separetely. Current implementation is directly based on open source implementation of AES, published by Brian Gladman on his page:

http://fp.gladman.plus.com/cryptography_technology/rijndael/

Note: Because AES-128 is block oriented algorithm, message encrypted have a length which is a multiple of 16 bytes. Therefore message data have to be padded with empty bytes.

The client (phone device) and server (operator's server) uses the same encryption key which is not known to end user. The key is loaded into device during factory setup. For security reasons only server can indicate encryption usage for responses.

Note: Encryption for message body sent from client to server (in POST requests) are obligatory.

Encryption usage and algorithm type is indicated by HTTP header:

“**Content-Type: x-giga-1**”

Only message bodies (data after HTTP headers) can be encrypted, so all data passed in URL as search criteria and other parameters are still sent in clear text. More details are discovered in examples below.

Example 30: (GET request with encrypted response)

For the example following assumptions are met:

command:	get_list	
type:	pr	private phonebook
nn:	*	include nickname field in results
ln:	*	include lastname field in results
hm:	*	include phone number (home) field in results
count:	10	get 10 items at time
limit	2048	optional, max. size of response in bytes

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_list&type=pr&nn=*&ln=*&hm=*&count=10&limit=2048 HTTP/1.1
Content-Type: text/xml
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

This is response for above request:

```
HTTP/1.1 200 OK
Content-Type: : x-giga-1
Content-Length: ...

[encrypted message body with padding bytes]
```

Encrypted message can contains generic answers described in previous chapters. For example:

```
<?xml version="1.0" encoding="UTF-8"?>

<list response=”get_list” type=”pr” total=”23” first=”1” last=”10”
   reqid=”5DC2”>
   <entry id="a83" sc=”nn”>
        <nn>Johny</nn>
   </entry>
   <entry id="ae3f" sc=”ln”>
        <ln>John</ln>
   </entry>
   …
   <entry>
        …
   </entry>
</list>
```

Example 31: (POST request with encrypted data)

For the example following assumptions are met:

The command for copy an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

command:	copy_data	
type:	pr	private phonebook
id:	52783	id of the entry which shall be changed
limit	1024	optional, max. size of response in bytes

All data sent in message body will be encrypted:

nn:	"mynewnickname"	user wants to set a new nickname

```
POST http://www.phonebook.svr.dom/phonebookservlet.do?command=
copy_data&type=pr&id=52783&limit=1024 HTTP/1.1
Content-Type: x-giga-1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…

[encrypted data and padding bytes]
```

The authorization part of the request definition is described in chapter 2.2(Authorization)

In this example, encrypted data carries following message:

```
<?xml version="1.0" encoding="UTF-8"?>

<data>
        <nn>mynewnickname</nn>
</data>
```

3.4 Content Download (optional)

3.4.1 Request Picture CLIP file (optional)

This chapter describes the procedure to download the picture clip file from the server. A picture clip can only be downloaded for one entry at a time. Availability of a picture clip can be evaluated using the procedure described in chapter 3.1.9.

The client sends a HTTP request to the phonebook server for retrieving the CLIP picture by search criteria including the following parameters:

command:

This is the primary command for the request.

get_clip_file	get clip file list command

type:

The type identifies which type of list shall be retrieved from the server

pb	list of public phonebook entries
yp	list of yellow pages entries
pr	list of private phonebook entries

id:

This is the entry id derived from previous “get_list” request, for which the picture ffile is requested.

limit:

The limit defines the maximum amount of memory that the client is able to process for each server response. This includes all the data which will be returned by the server (Header and payload). If omitted, a limit of 1024 bytes is assumed.

Limit>=1024	amount of data in bytes

pcf:

This parameter defines the format of the requested picture clip file. The format defined as a value which represents the codec info and pixel size:

<codec>x<XXXX>x<YYYY> with

codec:	1	JPEG interlaced
	2	tbd

XXXX	XXXX decimal value defining number of horizontal pixel
YYYY	YYYY decimal value defining number of vertical pixel

Informational:
The following table contains pcf values needed for user interface of gigaset cordless devices.

pcf	Res. limit	Description
1x0128x0140		full screen picture CLIP (without soft key area), Diamond handset
1x0304x0192		full screen picture CLIP (without soft key area), Saturn base
1x0128x0086		Picture clip incoming call, Diamond handset
1x0144x0192		Picture clip incoming call, Saturn base
1x0032x0044		Picture Clip for list entries, Diamond handset
1x0048x0032		Picture Clip for list entries, Saturn base
tbd	tbd	Define Hawking values

Editors Note: Provide rough information about reasonable limits.

Example 32: (request with for picture clip download)

For the example following assumptions are met:

The command for copy an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

command:	get_clip_file	
type:	pb	public phonebook
id:	52783	id of the entry for which clip file is requested
limit	tbd	optional, max. size of response in bytes

```
GET http://www.phonebook.svr.dom/phonebookservlet.do?command=
get_clip_file&type=pb&id=52783&pcf=1x0128x0086&limit=1024 HTTP/1.1
User-Agent: SHC_CLIENT/1.0
Host: www.phonebook.svr.dom
Authorization:Digest username=…
```

Example 33: (response for picture clip download)

For the example following assumptions are met:

The command for copy an entry is generated by the client and sent to the phonebook server http://www.phonebook.svr.dom.

command:	get_clip_file	
type:	pb	public phonebook
id:	52783	id of the entry for which clip file is requested
limit	tbd	optional, max. size of response in bytes

This is response for above request:

```
HTTP/1.1 200 OK
Content-Type: image/bmp
Content-Length: ...

[message body with picture clip file]
```

3.5 Success/Error Response

On all requests that don't return any data (like the chg_data request), the server will return a success/error indication. The error indication may also be used in other fault cases.

The HTTP response contains the following information:

response:

identifies the primary command for the request which caused the error

type:

identifies which type of list was accessed on the server:

pb	public phonebook
yp	yellow pages
pr	private phonebook

errorid:

indicates if the previous request was successful or which kind of error occurred:

0	OK
1	not allowed (e.g. user tried to change another users entry)
2	syntax error (e.g. incorrect command parameters)
3	duplicate (e.g. new nickname already exists)
4	invalid (e.g. invalid characters in a number string)
5	imprecise (i.e. too many matching results)
6	service not available
7	requested format not available (used with content download)

message (optional):

A textual description may be given, if not used the tag is omitted.

Example 34: (errorid 0 "OK")

The error response is generated by the phonebook server http://www.phonebook.svr.dom and sent to the client.

response:	chg_data	error response for change phonebook request
type:	pb	phonebook
errorid:	0	OK

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<error response="chg_data" type="pb">
   <errorid>0</errorid>
</error>
```

Example 35: (errorid 2 "unknown command parameter")
The error response is generated by the phonebook server http://www.phonebook.svr.dom and sent to the client.

o	response:	get_list	error response for get phonebook list request
o	type:	pb	public phone book
o	errorid:	2	syntax error (e.g. incorrect command parameters)
o	message:	(optional)	textual description of the problem

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<error response="get_list" type="pb">
   <errorid>2</errorid>
   <message>unknown command parameter</message>
</error>
```

Example 36: (errorid 4 "first and last parameters not allowed")
The error response is generated by the phonebook server http://www.phonebook.svr.dom and sent to the client.

o	response:	get_list	error response for get phonebook list request
o	type:	pb	public phone book
o	errorid:	4	invalid (e.g. invalid characters in a number string)
o	message:	(optional)	textual description of the problem

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<error response=”get_list” type=”pb”>
   <errorid>4</errorid>
   <message>First _and_ last given</message>
</error>
```

Example 37: (errorid 5 “too many results found”)

The error response is generated by the phonebook server http://www.phonebook.svr.dom and sent to the client.

○ response:	get_list	error response for get phonebook list request
○ type:	pb	public phone book
○ errorid:	5	imprecise (i.e. too many matching results)
○ message:	(optional)	textual description of the problem

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: ...

<?xml version="1.0" encoding="UTF-8"?>

<error response=”get_list” type=”pb”>
   <errorid>5</errorid>
   <message>Too many results were found.</message>
</error>
```

3.6 Search Criteria

The following list shows all currently defined search criteria. Please note that these criteria may not all be searchable. Depending on the database some of them may only be valid as a field in the search results. The list reflects the current situation only and may be extended on demand.

Criteria Long Name	Abbreviation used in protocol	type and format in XML search results
Last Name (name for YP)	ln	UTF8 string
First Name	fn	UTF8 string
Nickname	nn	UTF8 string
City	ct	UTF8 string
Zipcode	zc	string (6)
Street	st	UTF8 string
House Number	nr	string (6)
Country	co	not supported yet
Home Number (phone number)	hm	UTF8 string
Fax Number	fx	not supported yet
Mobile Number	mb	UTF8 string
Sip Number (VoIP Number)	sip	UTF8 string
Prefix	pf	not supported yet
Intprefix	ipf	not supported yet
Residence Name	rn	not supported yet
Coordinates (geograph.)	cd	not supported yet
Email	em	UTF8 string
URL	url	not supported yet
Additional_info	ai	not supported yet
Province_name	pn	not supported yet
Subscriber	sc	not supported yet
Category	cat	UTF8 string
What	wh	field valid only in request
Birthday	bi	string (8) in format: YYYYMMDD
Jabber/Messenger ID	jid	UTF8 string
Company Name	cpn	UTF8 string
Presence state	prs	UTF8 string
Availability Picture Clip	apc	string (max 3) with “yes” or “no”
Picture Clip Format	pcf	string (10) field valid only in request
Internal Number	in	UTF8 string
Business number	bp	UTF8 string

3.7 Language list

Following language codes should be used with “lang” parameter:

0	undefined
1	US
2	German
3	English International (standard)
4	Spanish
5	Portuguese
6	Scandinavia
7	Italian
8	Greek
9	French
10	Dutch
11	Norwegian
12	Danish
13	Swedish
14	Finnish
15	Czech
16	Turkish
17	Polish
18	Canadian French
19	Mexican Spanish
20	Brasilien Portuguese
21	Numeric
22	Slovakian
23	Russian
24	Greek (with Greek charset)
25	Hungarian
26	Croatian
27	Slovenian
28	Romanian
29	Serbian
30	Bulgarian
31	Hebrew
32	Chinese
33	Arabic
34	Persian
35	Turkish (with Turkish charset)
36	Catalan
37	Bosnian
38	Ukrainian

4. Device specific settings

To communicate with HTTP servers, device needs to known some parameters like URL address, login and password. These parameters are stored under presets called “providers”. Each “provider” consist information about server address, credentials, supported features (capabilities) and provider specific wording to be displayed on handset’s display.

Optional:

Some devices are shipped with preprogrammed presets for common providers. Device performs all network phonebook operations only with one “provider” at time. Active provider can be selected from provider dropdown list on Web Interface “Services” page.

To allow full customization of device, there is one additional EEPROM defined “provider”. This “provider” can be freely configured by user on special hidden Web Interface page:

```
http://<device_ip_address>/settings_services_eeprom_provider.html
```

Configuration page contains following fields:

1. **Enable EEPROM Phonebook Provider [Yes/No]**

   Enabling this option causes “EEPROM defined provider” to be shown on list of all providers and this provider can be chosen from dropdown list on “Services” page.

2. **Service URL** (74 characters max including question mark at end)

   URL to the service. E.g: *http://server.domain.com/cgi-bin/pb.cgi?*

   The question mark is mandatory at the end of the Service URL, therefore it is automatically added if missing.

3. **Login name** (max. 30 characters)
4. **Password** (max. 64 characters)

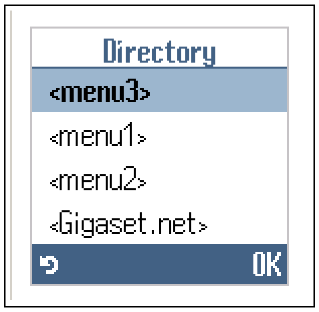
   If server access is protected by HTTP Digest authorization (only this method is supported).

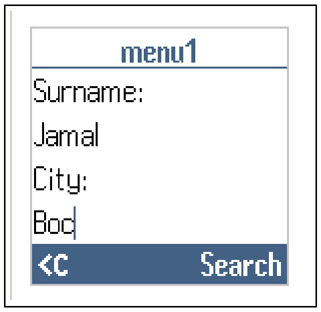
5. **Country code** (max. 2 characters)

   This code will be shown on “providers” list in Web Interface

6. **White Pages / Yellow Pages / PNAB NetDirList entry and headline** (max. 20 ch.)

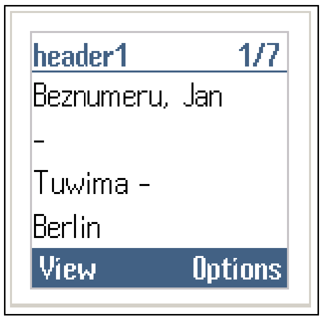
   This text is used for menu entry label and for header when search criteria editor is shown. E.g. “menu1” at pictures:





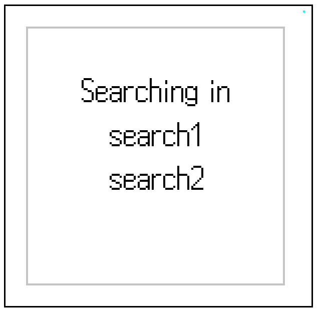
Surname:
Jamal
City:
Boc

7. **White Pages / Yellow Pages / PNAB header** (max. 20 ch.)
   This text is used for header, when list with search results are presented. E.g. “header1” at picture:



Beznumeru, Jan
-
Tuwima -
Berlin

8. **White Pages / Yellow Pages / PNAB searching in (1)** (max. 20 ch.)
9. **White Pages / Yellow Pages / PNAB searching in (2)** (max. 20 ch.)
   Field 8 and 9 define two line text displayed when search is performed. E.g. “search1” and “search2” at picture:


Searching in
search1
search2

10. **Capabilities:**
    a. **White pages search**
    b. **Yellow pages search**
       Provider supports network directory (White Pages) and Yellow pages. (Corresponding menu items are visible on handset)
    c. **Reverse lookup**
       Provider is capable to perform searches by phone number. (Additional search criteria editor is shown on handset)
    d. **Private network directory**
       Provider supports PNAB.
    e. **Autolookup**

If provider has “reverse lookup” flag and incoming call with valid CLIP number occurs, device performs search inside network directory for caller name.

If “private network directory” capability is enabled, then lookup is performed only inside PNAB.

f. Nickname for Private network directory

Provider supports nickname attribute in PNAB. This capability enables PNAB’s nickname editing from handset GUI.