

Table of Contents

Preface	4
Redesigned call park module.....	4
Intro.....	4
Configuration.....	4
How to use.....	4
Internet dial-up services provisioning, authorization, and billing	5
Description.....	5
How to configure	5
Notes.....	6
Support for caller identify and privacy flags	7
Configuration.....	7
Detailed description	7
BE behaviors	7
SIP behaviors	8
Automated discounts for subscription charges	8
Custom rounding of charged amounts for subscription	9
Custom reports.....	10
Payment Report.....	10
Multi-language web-mail interface.....	10
Localization of prompts for Recharge voucher and prepaid calling cards	10
Configuration options	10
Adding new language voice prompts.....	11
Adding Text-To-Speech module for the new language.....	14
Complete support for disk space quotas.....	16
Full mailbox auto-notification	16
Configuration options	16
How it works	16
Ability to use a local PSTN interface card for T.38 faxing	17
Network diagram and call flow	17
Required equipment.....	17
XML API.....	18
Customer class.	18
Overview.....	18
WSDL	18
Security	18
Error handling	18

Methods.....	18
Type Reference	19
Account class	26
Overview.....	26
WSDL	27
Security	27
Error handling	27
Methods.....	27
Type Reference	28

Preface

The purpose of this document is to provide some essential information needed to start beta-testing of the new features, which have been included into the Maintenance Release 14. Please note, that this document is the subject to further changes and cannot be considered as an official specification document.

Redesigned call park module

Intro

Call park is a feature that allows a person to put a call on hold at one telephone and continue the conversation from any other telephone.

The “call park” feature is activated by pressing a pre-programmed button or a special sequence of buttons. This transfers the current telephone conversation to an unused extension number and immediately puts the conversation on hold. (This is called *parking the call*, and the call is said to have *parked* onto a certain extension. Essentially, call parking temporarily assigns an extension number to an incoming call.) The telephone system will then say the extension number of the parked call so that the call can later be retrieved.

Configuration

- a) Add call parking module into modules list in /home/porta-billing/etc/porta-billing.conf file on master server.

```
LoadModules=<<EOT
...
call_parking=Porta::Modules::CallParking
...
EOT
```

- b) Restart radiusd process, on master server:

```
sudo /usr/local/etc/rc.d/000.radius.sh restart
```

- c) Enable “Call parking” for some customer on the WEB interface, and define parking and deparking prefixes. (Edit customer page-> “call feature” tab-> “call parking” drop down box)

RESTRICTION: Call parking works under the same customer only, call cannot be retrieved by account under another customer.

How to use

Example:

Customer A configuration: “Call parking”=Yes, “Park prefix” =0011, “Depark prefix”=0022

Customer A has some registered accounts 000999001, 000999002 and 000999003 which can call to each other.

- a) Account 000999001 makes a call to 000999002
- b) Account 000999002 presses a hold (flash) button and make a call to the destination 0011.
- c) Account 000999002 hears some numbers which should be called to retrieve this call.
(i.e. 00221234, where 0022 – it is depark prefix, 1234 – id of parked call (parkid))
000999002 can hang up the phone.
- d) Account 000999001 hears some music (Music on hold). Account 000999001 was being parked.
- e) Account 000999003 makes a call to 00221234 and can talk with 000999001.
Call 1234 was retrieved.

PortaBilling can park the call by two methods:

- 1) GENERATE mode: User dials parking prefix only without any extension.
- 2) ABBREV EXTENSION mode: User dials parking prefix with parkid.

PortaBilling can retrieve the call by two methods:

- 1) REFERENCE mode: User dials the deparkprefix+parkid.
- 2) QUICK mode: User dials the deparkprefix without parkid. (call can be retrieved by the parked account only).

Internet dial-up services provisioning, authorization, and billing

Description

This plugin interprets the standard CISCO RADIUS requests into call records in order to perform standard billing. The dialup is billed by a special destination prefix (DIALUP) and terminated via internal SIP-UA connection

How to configure

- a) Add to /home/porta-billing/etc/porta-billing.conf on master server:

```
[Plugins]
```

```
LoadModules=<<EOT
```

```
...
```

```
dialup=Porta::Modules::DialUp
```

```
...
```

```
EOT
```

b) Restart radius server

```
sudo /usr/local/etc/rc.d/000.radius.sh restart
```

check that module was loaded successfully (in /var/log/porta-billing.log on master server)

```
Modules: initialize dialup ...
```

```
Modules(dialup): Configured 'dialup:Framed_Protocol' => 'PPP'
```

```
Modules(dialup): Configured 'dialup:Connection_Remote_IP' => 'sip-ua'
```

```
Modules(dialup): Configured 'dialup:Dialup_CLD' => 'DIALUP'
```

c) Create destination “DIALUP”;

d) Add rate “DIALUP” to tariff assigned to “SIP-UA” connection;

e) Add rate “DIALUP” to tariff assigned to account accessibility list.

NOTES: Possible options for this module in porta-billing.conf:

```
[Dialup]
```

- Framed_Protocol=PPP
#optional, specify the dialup protocol to be matched by this module
- Connection_Remote_IP=<ip>
#optionally override the connection to be matched (sip-ua is default).
- Dialup_CLD=<the number to be matched in the tariff rates>
#optionally override the number to be used for placing the call (DIALUP is default).

f) Configure your CISCO access server send PPP authorization and accounting to Master server

Notes

1. Start accounting is mandatory for Dial UP usage. The reason is that for dialup, during auth stage is no guarantee that the session started and it also has no information to identify the future session.

Support for caller identify and privacy flags

Configuration

Find a new drop down box “Hide CLI Mode” on “VoIP to Vendor” and “PSTN to Vendor” connections:

The screenshot shows two configuration tables. The top table is for 'VoIP to Vendor' connections, and the bottom table is for 'PSTN to Vendor' connections. Both tables have a 'Hide CLI Mode' dropdown menu highlighted in red. The dropdown menu options are 'Clear caller info', 'Clear caller info', and 'Use private headers'.

Edit	Load	Node *	Port	Transl. Rule	Outgoing Rule	CLI Transl. Rule	Delete
		CISCO AS5300					
		43696 Tariff Uplo					

Edit	Load	Remote IP *	RTP Proxying	Transl. Rule	Outgoing Rule	CLI Transl. Rule	Acc
		SIP-UA	Always			s/000/444/;	None
		PortaSIP routing					100
		SIP-UA				s:000/444;	100
		PortaSIP routing					100

“**Clear caller info**” – use default PortaBilling CLIR method for this connection. (default value).

In this case PortaSIP will remove “display name” is replaced with “Anonymous”, and url.username is removed from sip filed “From” for outgoing INVITE.
example:

```
From: "John Smith" <sip:1234@sip.example.com>; tag=0099-8877
```

will be changed to:

```
From: Anonymous <sip:sip.example.com>; tag=0099-8877
```

“**Use private headers**” – use caller identify and privacy flags to interconnect with carriers regardless of their compliance with RFC3325.

1. From field is unchanged (as for “Clear caller info”);
2. The following data is added to sip packet:

```
Privacy: id
```

```
P-Asserted-Identity: <sip:1234@sip.example.com>
```

```
Remote-Party-ID:
```

```
<sip:1234@sip.example.com>;party=callid;privacy=full
```

Detailed description

BE behaviors

On “Edit customer” page, “call feature” tab.

When feature “Hide CLI” is:

- enabled for a customer

BE returns routing with parameter “clir” (Note it is a CLIR not a CLI as early!)

CLIR parameter values:

- «CLIR=A» if “Hide CLI Mode” = “Clear caller info” for this route
- «CLIR=P» if “Hide CLI Mode” = “Use private headers” for this route

NOTE: BE shouldn't (but can) write parameter CLI to route in case if CLI for this route was replaced with route CLI replacement rules, it should be ignored by PortaSIP.

NOTE: If “Hide CLI Mode” = “Use private headers” and call goes to SIP-UA or PortaUM be always set CLIR=A

When feature “Hide CLI” is disabled for customer BE returns routing without any parameters “clir” but can place “cli” parameter in case if CLI should be changed for this route.

SIP behaviors

1. If PortaSIP received routing with CLI parameter it was ignored.
2. If PortaSIP received routing with CLIR parameter:
 - If «CLIR=A» (it is concealment for UA)

Display name is replaced with “Anonymous”, and url.username is removed from sip filed “From” for outgoing INVITE.

example:

```
From: "John Smith" <sip:1234@sip.example.com>; tag=0099-8877
```

will be changed to:

```
From: Anonymous <sip:sip.example.com>; tag=0099-8877
```

- if «CLIR=P» (it is concealment for Vendor)

1. From field is unchanged;
2. The following data is added to sip packet:

```
Privacy: id
```

```
P-Asserted-Identity: <sip:1234@sip.example.com>
```

```
Remote-Party-ID:
```

```
<sip:1234@sip.example.com>;party=callid;privacy=full
```

Automated discounts for subscription charges

This feature provides certain customers with subscription charges at the rate, lower than default. Discount will be applied to the subscription amount after all the other calculations (prorating, etc.). The key parameters are:

- **Customer's discount rate** – attribute of the customer (additional info tab), possible values: empty (NULL), 0...100. (NULL and 0 are identical and mean “no discount” – the subscription charge is computed according to the price data in subscription info)
- **Subscription discount rate** – attribute of every customer's/account's subscription. Possible values: NULL (use customer's discount rate) or 0...100 (specific discount rate which overrides discount rate for a specific discount). 0 means “the standard subscription rate”, 100 means “this subscription is for free”. When a new account is created, all of his mandatory subscriptions are created with discount rate NULL. Later it is possible to edit the discount rate for any of the mandatory subscriptions (of course if ACLs permit). When adding a new optional subscription it should be possible to enter the discount rate at the creation stage (again if permitted by ACL – if not, then subscription is created with discount rate = NULL)

During the update all of the existing customers and subscriptions will be assigned NULL value.

Example:

Customer A has a discount rate of 10%. He has three accounts (1,2,3) with subscription Super Call (\$40 per month), all accounts were activated on 15th of September. Plus the customer has a subscription “IPPBX” (\$10/month) (activated back in July). Account 1 has subscription discount 20%, account 2 – NULL and account 3 – 0. On the 1st of October the following charges will be calculated:

- 1: $(\$40 * (15/30)) * 20\% = \16
- 2: $(\$40 * (15/30)) * 10\% = \18
- 3: $(\$40 * (15/30)) = \20
- A: $\$10 * 10\% = \9

and as a result 4 CDRs should be produced. These calculations should be reflected in the CDR.history, otherwise it looks like normal subscription CDRs. Calculations for progressive subscription charges are done in the same fashion.

Custom rounding of charged amounts for subscription

This feature allows customer to define rounding for subscription charges in the way it have already been made in tariffs.

Rounding is used when CDR is being inserted to the CDR_* table. All operations before CDR insert use non rounded value for calculations.

Custom reports

Payment Report

Accountants need to know the total payments received during a period distributed by Payment Types. All transaction that relate to payment received or payment charge backs.

Multi-language web-mail interface

Spread the gettext localization mechanism used in SquirrelMail to translate all PortaUM specific messages on the Web interface. User is allowed to configure web mail language in his voicemail settings.

Localization of prompts for Recharge voucher and prepaid calling cards

Configuration options

The configuration for the applications contains the following parameters in sections [Voicemail] and [Debit Card] of /home/porta-um/etc/porta-um.conf.

For Debit Card Application:

Languages N_i=<i_env>:<dnis_number>:<languages list separated by '/'>

for example:

Languages 1_i=5:*55:en/ru

Languages 2_i=1:1234567890:ru

For Voice Mail service (Recharge voucher, Play Balance):

Languages N_i=<i_env>:<languages list separated by '/'>

for example:

Languages 1_i=1:en/ru

Languages 2_i=2:ru

Max number to enter language selection attempts is 3 (not configurable option).

UM checks account ID in the database (slave).

UM finds account ID.

UM uses preconfigured account's language ('Preferred IVR Language' field).

UM doesn't find account id in the database.

UM checks Languages configuration parameters in the /home/porta-um/etc/porta-um.conf file.

Total languages per env and dnis more than 1

Application plays language selection prompts: "Please press 1 for English, press 2 for Spanish".

The language is selected when the caller presses a predefined key.

The caller is asked only once, at the beginning of the session.

Application uses the selected language until the caller disconnects.

Wrong language selected

1. Application plays "You have made an invalid selection".
2. Application returns back to the Language selection.

Maximum number of attempts to reenter language selection has been exceeded.

1. Application plays error prompt.
2. Application terminates the incoming call.

Only one language per env and dnis

Languages 2?=2:ru

or

Languages 2?=1:1234567890:ru

UM uses preconfigured language.

No languages defined per env and dnis

UM uses DEFAULT_LANGUAGE value.

Adding new language voice prompts.

1. You should record your own prompts in your language.
Prompts should be an audio file (.au) format of 8-bit, u-law, and 8-khz encoding.

The list of voice files is almost the same for all TTS packages except some language specific.

There are standard voice prompts:

Numbers:

_zero.au – “Zero”;
_one.au – “One”;
_two.au – “Two”;
_three.au – “Three”;
_four.au – “Four”;
_five.au – “Five”;
_six.au – “Six”;
_seven.au – “Seven”;
_eight.au – “Eight”;
_nine.au – “Nine”;
_ten.au – “Ten”;
_eleven.au – “Eleven”;
_twelve.au – “Twelve”;
_thirteen.au – “Thirteen”;
_fourteen.au – “Fourteen”;
_fifteen.au – “Fifteen”;
_sixteen.au – “Sixteen”;
_seventeen.au – “Seventeen”;
_eighteen.au – “Eighteen”;
_nineteen.au – “Nineteen”;
_twenty.au – “Twenty”;
_thirty.au – “Thirty”;
_forty.au – “Forty”;
_fifty.au – “Fifty”;
_sixty.au – “Sixty”;
_seventy.au – “Seventy”;
_eighty.au – “Eighty”;
_ninety.au – “Ninety”;
_hundred.au – “Hundred”;
_thousand.au – “Thousand”;

Time and currencies:

_second.au – “Second”;
_seconds.au – “Seconds”;
_minute – “Minute”;
_minutes – “Minutes”;
_hour.au – “Hour”;
_hours.au – “Hours”;
_cent.au – “Cent”;
_cents.au – “Cents”;
_dollar.au – “Dollar”;
_dollars.au – “Dollars”;

Miscellaneous prompts:

`_account_blocked.au` – “This account is currently in use.”;
`_and.au` – “And”;
`_auth_fail.au` – “You have entered an invalid card number. Please reenter your card number followed by the pound key.”;
`_author_fail.au` – “Please reenter the phone number you wish to reach”.
`_blocked.au` – “Sorry, the number you have dialed is blocked. If you feel you have reached the destination, please call the customer service number.”
`_card_expired.au` – “We are sorry. Your card has expired.”
`_dest_blocked.au` – “Sorry, the number you have dialed is blocked. If you feel you have reached the destination, please call the customer service number.”
`_dest_busy.au` – “The party you are calling is busy. Please enter a new number or hang up and try again later.”
`_dest_collect_fail.au` – “Sorry we are unable to connect your call at this time. Please hang up and dial again or call the customer service number”;
`_disconnect.au` – “Your call will be disconnected.”
`_enter_card_num.au` – “Please enter card number followed by the pound key.”
`_enter_dest.au` – “Please enter the phone number you wish to reach.”
`_final.au` – “We are having difficulties connecting your call. Please try again later.”
`_generic_final.au` – “Please hang up and try again.”
`_insufficient_funds.au` – “You have insufficient funds on your account”
`_invalid_amt.au` – “You have more than one million.”
`_invalid_digits.au` – “You have entered an invalid number of digits. Please reenter your card number followed by the pound key.”
`_lang_sel1.au` – “Please press 1 for English”;
`_lang_sel2.au` – “Please press 2 for English”;
`_lang_sel3.au` – “Please press 3 for English”;
`_lang_sel4.au` – “Please press 4 for English”;
`_lang_sel5.au` – “Please press 5 for English”;
`_lang_sel6.au` – “Please press 6 for English”;
`_lang_sel7.au` – “Please press 7 for English”;
`_lang_sel8.au` – “Please press 8 for English”;
`_lang_sel9.au` – “Please press 9 for English”;
`_no_aaa.au` – “We are having technical difficulties. Please call back later.”;
`_no_card_entered.au` – “We did not get any input. Please enter your card number followed by the pound key.”
`_no_dest_entered.au` – “We did not get any input. Please enter the destination number you are calling.”
`_no_lang_sel.au` – “We did not get your language selection.”;
`_wrong_lang_sel.au` – “You have made an invalid selection”;
`_you_have.au` – “You have”
`_zero_bal.au` – “You have zero balance on account. Please call the customer service.”

2. Place them into appropriate directory under /home/porta-um/apache/prompts
3. Encode those files into g729 format.

You can use the follow script for converting procedure:

```
#!/bin/sh

if [ "$#" != "1" ]; then
    echo "usage: `basename $0` file.au"
    exit
fi

if [ ! -f "$1" ]; then
    echo "usage: `basename $0` file.au"
    exit
fi

EXT=`echo "$1" | sed -Ee 's/.*(\..+)/\1/'`
NAME=`basename "$1" "$EXT"`
if sox "$1" -r 8000 -w -s "$NAME.raw" >/dev/null 2>&1; then
    g729client -s /tmp/g729enc.sock < "$NAME.raw" >
"$NAME.g729"
    rm "$NAME.raw"
else
    echo "ERROR: Sox cannot convert file $1"
fi
```

Adding Text-To-Speech module for the new language

1. Customer should provide and explain the language specifics rules for dynamic prompts “You have”....

We need a description about:

- 1.1) Assuming there are sound files (such as one.au, two.au, ... ten.au, twenty.au, hundred.au,) what is the correct way to pronounce
 - a) numbers between 1 and 19 (I would assume it is just playing the prompt file)
 - b) numbers between 20 and 99, are there any exceptions? (e.g. in English 25 = <twenty.au> + <five.au>, but in German 25 = <five.au> + <and.au> + <twenty.au>)

- c) numbers between 100 and 999
 - d) numbers between 1000 and 999,999
- 1.2) Is there plural/singular in the language? E.g. "one cent", but "three cents"?
 - 1.3) How the time interval is pronounced?
 - 1.4) How the date/time value (e.g. 10:05am) is pronounced?
 - 1.5) Which currencies must be supported with language?

The main idea of this is collect as much as possible information about language specific rules.

2. You should write your own Text-To-Speech module.

See `SpeechSynt::Language::EN.pm`

or `SpeechSynt::Language::RU.pm` for example

3. Define module in `SpeechSynt::UmTTS.pm`

```
package SpeechSynt::UmTTS;

# Add your language here
use SpeechSynt::Language::EN;
use SpeechSynt::Language::RU;

# And here
sub new {
    my $proto = shift;
    my $class = ref($proto) || $proto;
    my $self = {};
    bless $self, $class;

    my ($lang, $level, $path) = @_ ;
    $self->{PROMPT_PATH} = $path;
    $self->{PROMPT_LEVEL} = "$level/";

    if ($lang eq 'ru') {
        $self->{tts} = new
SpeechSynt::Language::RU($self);
    }

    # All unknown languages default to English
    else {
        $self->{tts} = new
SpeechSynt::Language::EN($self);
    }

    return $self;
}
```

}

4. Don't forget to restart apache (or /usr/local/etc/rc.d/voice-mail.sh if you're using Cisco-less setup)

Complete support for disk space quotas

The main goal of this feature is display information about mailbox quota usage under particular account.

You are currently using 7597 KB of your 9765 KB

	From	Date	
<input type="checkbox"/>	Oleg Rakitskiy	Sat, 3:44 pm	+ Second Email
<input type="checkbox"/>	Oleg Rakitskiy	Sat, 3:39 pm	+ Email
<input type="checkbox"/>	Oleg Rakitskiy	Nov 8, 2006	+ 2
<input type="checkbox"/>	Oleg Rakitskiy	Nov 8, 2006	+ 1

Quota can be configured in `~porta-um/etc/porta-um.conf`

[IMAP]

`NewuserQuota=10000000` # value is in bytes.

Full mailbox auto-notification

If account used out all of his free space or if total size of his messages became more than quota pre-defined value, a corresponding notification is sent to an external email box. External email should be configured in UM preferences.

Also a voicemail message can be played to let user know, that he have used all his free space.

Configuration options

Parameter in the [Voicemail] section is `Play Notification = yes | no`
Default: no

How it works

1. System checks user's mailbox (quota and size);

2. System compares mailbox space exhaustion and value of parameter Warning Overflow;
3. Checks Play Notification parameter;
4. Plays warning message: 'mail_box_full' voice file.

Ability to use a local PSTN interface card for T.38 faxing

A purpose of this feature is to use an intermediate Cisco VoIP/PSTN gateway to handle T.38 faxes transcoding on VoIP side and communicate with PortaUM directly via T1/E1 to avoid using Low Bitrate Codecs and T.38 on the way from PSTN to UM.

Network diagram and call flow

Fax Machine <==> PSTN <==> CiscoGW1 <==> VoIP <==> PortaSIP <==> VoIP <==> CiscoGW2 <==> PSTN <==> PortaUM

1. Caller (Fax Machine) calls to UM-enabled account using low bitrate audio codec (e.g. g.729).
2. Caller (Fax Machine) initiates fax transmission.
3. PortaUM software catches CNG tones and answers with CED ones.
4. CiscoGW1 and CiscoGW2 do REINVITE to use T.38 codec to transmit fax.
5. PortaUM receives the fax.
6. After receiving the fax call is finished.

Required equipment

1. Cisco VoIP/PSTN gateway with T1/E1 interface.
2. Digium T1/E1 PCI card ([TE110P \(One T1/E1 CAS/PRI Span\)](#) or [TE205P \(Two T1/E1 CAS/PRI Spans\)](#))

Examples how to configure gateways are available upon request.

XML API

Customer class

Overview

The PortaBilling100 Customer SOAP Interface manages the reading, insertion, update, and deletion of customer records. To access the API following parameters should be used:

Proxy (address of server): <https://pbslave-server.yourdomain.com:443/soap>

URI (namespace): <https://pbslave-server.yourdomain.com//Porta/SOAP/Customer>

To access reseller's API use 8444 port instead of 443 in address string.

WSDL

The WSDL for the SOAP Interface will be provided in the further releases.

Security

Connection to the SOAP interface is provided via HTTPS. Authentication is achieved via use of authentication pairs (login-password). Each request to a method should contain **auth_info** structure as the SOAP Header attribute.

Error handling

SOAP faults are used to carry error information within a SOAP message. If the actual response has SOAP Fault element as a body entry, then an error has occurred. In this case, any other fields on the response are not guaranteed to be accurate; only the Fault subelements should be used to identify an error. Currently these subelements are:

- **faultcode** is intended for use by client software to provide an algorithmic mechanism for identifying the fault.
- **faultstring** provides a human readable explanation of the fault and is not intended for algorithmic processing.

Methods

1. `get_customer_info`

Parameters: GetCustomerInfoRequest

Return value: GetCustomerInfoResponse

This method allows API user (administrator or reseller) to get a

customer record from database. The customer must be viewable (owned) by the user making the request.

2. **get_customer_list**

Parameters: GetCustomerListRequest
Return value: GetCustomerListResponse

This method allows API user (administrator or reseller) to get a list of customer records. The customers must be viewable (owned) by the user making the request.

3. **validate_customer_info**

Parameters: ValidateCustomerInfoRequest
Return value: ValidateCustomerInfoResponse

This method allows API user to check if supplied data is valid to create new or update existed customer record. It returns completed data in case of success.

4. **add_customer**

Parameters: AddCustomerRequest
Return value: AddUpdateCustomerResponse

This method allows API user to create new customer record with the supplied data.

5. **update_customer**

Parameters: UpdateCustomerRequest
Return value: AddUpdateCustomerResponse

This method allows API user to update existed customer record with the supplied data.

6. **delete_customer**

Parameters: DeleteCustomerRequest
Return value: DeleteCustomerResponse

This method allows API user to delete existed customer, retail or reseller, if it has no accounts, subcustomers, cdrs or managed objects.

Type Reference

1. **Account_info structure**

<i>Field</i>	<i>Type</i>	<i>Description</i>
login	string, 16 chars max	User login to PortaBilling100 web interface
password	string, 16 chars max	User password to PortaBilling100 web interface
Or		
session_id	string, 32 chars max	Unique id of previously opened soap session

2. GetCustomerInfoRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_customer	integer	Unique id of the customer database record
or		
refnum	string, 32 chars max	Reference number (custom field)
or		
name	string, 41 chars max	Name of the customer on the PortaBilling100 interface, unique within environment

3. GetCustomerInfoResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_info	CustomerInfo structure	

4. GetCustomerListRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
offset	Integer	Amount of rows to skip at the beginning of the list
limit	integer	Amount of rows to retrieve

5. GetCustomerListResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_list	array of CustomerInfo	

6. CustomerInfo structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_customer	integer	Unique id of the customer database record
refnum	string, 32 chars max	Reference number (custom field)
name	string, 41 chars max	Name of the customer on the PortaBilling100 interface, unique within environment
i_customer_type	integer	Either 1 (retail customer or subcustomer) or 2 (reseller)
i_parent	integer	Null in case of direct customer, or i_customer of reseller in case of subcustomer
Iso_4217	string, 3 chars	ISO4217 code of currency in which the customer is billed
opening_balance	decimal, 5 places after the point	Initial balance of the customer
balance	decimal, 5 places after the point	Balance of the customer
i_billing_period	integer	ID of customer's billing period, references to Billing_Period table
i_acl	integer	ID of customer's access level, references to Access_Levels table
i_routing_plan	integer	ID of customer's routing plan, references to Routing_Plans table

i_vd_plan	integer	ID of customer's discount plan, references to Volume_Discount_Plans table
i_moh	integer	ID of customer's "music on hold" option, references to Music_On_Hold table
i_customer_class	integer	ID of customer's customer class, references to Customer_Classes table
i_tariff	integer	ID of customer's tariff, references to table Tariffs, applies to resellers only
i_tariff_incoming	integer	ID of customer's incoming tariff, references to table Tariffs, applies to resellers only
i_template	integer	ID of customer's invoice template, references to table Templates
i_rep	integer	ID of customer's representative, references to table Representatives
i_time_zone	integer	ID of customer's time zone, references to table Time_Zones
i_lang	string	Code of customer's web language, references to table Locale_Languages
service_flags	string, 32 chars max	Customer's call features settings
companyname	string, 41 chars max	The customer's company name
salutation	string, 15 chars max	The customer's salutation
Firstname	string, 25 chars max	The customer's first name
Midinit	string, 5 chars max	The customer's middle name initials

Lastname	string, 25 chars max	The customer's last name
baddr1	string, 41 chars max	The 1 st line of the customer's address
baddr2	string, 41 chars max	The 2 nd line of the customer's address
baddr3	string, 41 chars max	The 3 rd line of the customer's address
baddr4	string, 41 chars max	The 4 th line of the customer's address
baddr5	string, 41 chars max	The 5 th line of the customer's address
city	string, 31 chars max	City for the customer's address
State	string, 21 chars max	Province or state
Zip	string, 13 chars max	Postal zip code
Country	string, 31 chars max	Country
Note	string, 41 chars max	Short note (description)
faxnum	string, 21 chars max	Fax number
cont1	string, 41 chars max	The main contact person
phone1	string, 21 chars max	The main phone number
cont2	string, 41 chars max	Alternative contact person
phone2	string, 21 chars max	Alternative phone number
Email	string, 99 chars max	Email address
Bcc	string, 99 chars max	BCC email address
login	string, 16 chars max	The customer's login to self-care web interface
password	string, 16 chars max	The customer's password to self-care web interface
tax_id	string, 16 chars max	Tax id
credit_limit	decimal, 5 places	The customer's credit limit

	after the point	value, null if not defined
credit_limit_warning	string, 25 chars max	The value of balance threshold to send warnings, either signed absolute value or positive relative value with % sign
send_statistics	enumeration	F – send full statistics to the customer; S – send short statistics N – do not send statistics
blocked	boolean, Y/N	Block customer's calls
ppm_enabled	boolean, Y/N	Allow customer to manage his periodic payments on his self-care
drm_enabled	boolean, Y/N	Allow customer to manage his dialing rules on his self-care
callshop_enabled	boolean, Y/N	Enable callshop features on customer's selfcare interface
bp_charge_cc	boolean, Y/N	Automatically charge customer's credit card when the billing period is closed
bill_status	enumeration	O – customer is open; S – customer is suspended due to his overdue bills; C – customer is closed due to his unpaid bills
max_abbreviated_length	integer	Maximum allowed length of customer's abbreviated numbers, applies to retail customers only
discount_rate	integer	Value of customer's subscription discount, in percents.
out_date_format	string, 16 chars max	Output format of date presentation on the customer's self-care

out_time_format	string, 16 chars max	Output format of time presentation
out_date_time_format	string, 16 chars max	Output format of full date/time presentation
in_date_format	string, 16 chars max	Input format of date presentation
in_time_format	string, 16 chars max	Input format of time presentation
cld_translation_rule		Customer's translation rule expression
cli_in_translation_rule		Customer's translation rule for incoming calls expression

7. ValidateCustomerInfoRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_info	CustomerInfo structure	
Note: Omit i_customer to check if data may be used to create new customer record.		

8. ValidateCustomerInfoResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_info	CustomerInfo structure	

9. AddCustomerRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_info	CustomerInfo structure	
Note: i_customer will be ignored; most of the fields may be omitted; mandatory fields are iso_4217 and name; for the reseller API user i_customer_type and i_parent fields will be replaced with		

predefined values

10. UpdateCustomerRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
customer_info	CustomerInfo structure	

Note:*i_customer* is mandatory;

only fields that need to be modified should be supplied;

iso_4217, *i_customer_type*, *i_parent* and *opening_balance* fields can not be updated and must be omitted**11. AddUpdateCustomerResponse structure**

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_customer	Integer	ID of the created/modified customer's record

12. DeleteCustomerRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_customer	integer	ID of the customer's record to be deleted

13. DeleteCustomerResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
result	integer	1 in case of success, 0 in case of failure

Account class**Overview**

The PortaBilling100 Account SOAP Interface manages the reading, insertion, update, and deletion of account records. To access the API following parameters should be used:

Proxy (address of server): <https://pbslave-server.yourdomain.com:443/soap>

URI (namespace): <https://pbslave-server.yourdomain.com//Porta/SOAP/Account>

To access reseller's API use 8444 port instead of 443 in address string.

WSDL

The WSDL for the SOAP Interface will be provided in the further releases.

Security

Connection to the SOAP interface is provided via HTTPS. Authentication is achieved via use of authentication pairs (login-password). Each request to a method should contain **auth_info** structure as the SOAP Header attribute.

Error handling

SOAP faults are used to carry error information within a SOAP message. If the actual response has SOAP Fault element as a body entry, then an error has occurred. In this case, any other fields on the response are not guaranteed to be accurate; only the Fault subelements should be used to identify an error. Currently these subelements are:

- **faultcode** is intended for use by client software to provide an algorithmic mechanism for identifying the fault.
- **faultstring** provides a human readable explanation of the fault and is not intended for algorithmic processing.

Methods

1. **get_account_info**

Parameters: GetAccountInfoRequest
Return value: GetAccountInfoResponse

This method allows API user (administrator or reseller) to get an account record from database. The account must be viewable (owned) by the user making the request.

2. **get_account_list**

Parameters: GetAccountListRequest
Return value: GetAccountListResponse

This method allows API user (administrator or reseller) to get a list of account records. The accounts must be viewable (owned) by the user making the request.

3. **validate_account_info**

Parameters: ValidateAccountInfoRequest
Return value: ValidateAccountInfoResponse

This method allows API user to check if supplied data is valid to create new or update existed account record. It returns completed data in case of success.

4. add_account

Parameters: AddAccountRequest
Return value: AddUpdateAccountResponse

This method allows API user to create new account record with the supplied data.

5. update_account

Parameters: UpdateAccountRequest
Return value: AddUpdateAccountResponse

This method allows API user to update existed account record with the supplied data.

6. delete_account

Parameters: DeleteAccountRequest
Return value: DeleteAccountResponse

This method allows API user to delete existed account, if there is no cdrs associated with it.

Type Reference

1. Account_info structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
login	string, 16 chars max	User login to PortaBilling100 web interface
password	string, 16 chars max	User password to PortaBilling100 web interface
Or		
session_id	string, 32 chars max	Unique id of previously opened soap session

2. GetAccountInfoRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_account	integer	Unique id of the account database record
or		
i_batch	integer	Reference to Batch record which account belongs to
control_number	integer	The sequential number of the account in the batch
or		
batch_name	string, 32 chars max	Name of the Batch which account belongs to
control_number	integer	The sequential number of the account in the batch
or		
id	string, 32 chars max	ID (PIN) of the account on the PortaBilling100 interface, unique within environment

3. GetAccountInfoResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_info	AccountInfo structure	

4. GetAccountListRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
offset	Integer	Amount of rows to skip at the beginning of the list
limit	integer	Amount of rows to retrieve
i_customer	integer	Reference to Customer record which account belongs to

i_batch	integer	Reference to Batch record which account belongs to
---------	---------	--

5. GetAccountListResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_list	array of AccountInfo	

6. AccountInfo structure

* - field cannot be updated since the account creation

** - field is read-only and cannot be updated at all

field – mandatory field

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_account **	integer	Unique id of the account database record
<u>id</u>	string, 32 chars max	ID (PIN) of the account on the PortaBilling100 interface, unique within environment
billing_model *	integer	-1 – Debit type of account 0 - Recharge voucher type 1 – Credit type
<u>i_customer</u> *	integer	Reference to Customer record which account belongs to
<u>i_batch</u> *	integer	Reference to Batch record which account belongs to
<u>control_number</u> *	integer	The sequential number of the account in the batch
iso_4217 **	string, 3 chars	ISO4217 code of currency in which the account is billed
opening_balance *	decimal, 5 places after the point	Initial balance of the account
balance **	decimal, 5 places after the point	Balance of the account

issue_date *	date, iso format	Date of the account issue
<u>activation_date</u>	date, iso format	The date from which the account is usable
expiration_date	date, iso format	The date from which the account will be unusable
first_usage	date, iso format	The date when the account was used a very first time
last_usage	datetime, iso format	The date when the account was used last time
last_recharge	datetime, iso format	The date when the account was recharged using IVR or web self-care
life_time	integer	Relative to the activation date, the account will expire on first usage date + lifetime days
redirect_number	string, 15 chars	Associated number
i_product	integer	ID of account product, references to Products table
i_acl	integer	ID of account access level, references to Access_Levels table
i_vd_plan	integer	ID of account discount plan, references to Volume_Discount_Plans table
i_moh	integer	ID of account “music on hold” option , references to Music_On_Hold table
ua_profile_id	integer	ID of UA profile
i_time_zone	integer	ID of account time zone, references to table Time_Zones
i_lang	string	Code of account web language, references to table

		Locale_Languages
iso_639_1	string, 2 chars max	Code of account preferred IVR language, references to table Languages
service_flags	string, 32 chars max	Account call features settings
companyname	string, 41 chars max	The account company name
salutation	string, 15 chars max	The account salutation
firstname	string, 25 chars max	The account first name
midinit	string, 5 chars max	The account middle name initials
lastname	string, 25 chars max	The account last name
baddr1	string, 41 chars max	The 1 st line of the account address
baddr2	string, 41 chars max	The 2 nd line of the account address
baddr3	string, 41 chars max	The 3 rd line of the account address
baddr4	string, 41 chars max	The 4 th line of the account address
baddr5	string, 41 chars max	The 5 th line of the account address
city	string, 31 chars max	City for the account address
state	string, 21 chars max	Province or state
cip	string, 13 chars max	Postal zip code
country	string, 31 chars max	Country
note	string, 41 chars max	Short note (description)
faxnum	string, 21 chars max	Fax number
cont1	string, 41 chars max	The main contact person
phone1	string, 21 chars max	The main phone number
cont2	string, 41 chars max	Alternative contact person

phone2	string, 21 chars max	Alternative phone number
subscriber_email	string, 99 chars max	Email address of the subscriber
login	string, 16 chars max	The account login to self-care web interface
password	string, 16 chars max	The account password to self-care web interface
h323_password	string, 255 chars max	VoIP password to be used to authenticate any calls made using this Account
email	string, 128 chars max	E-mail associated with the account
credit_limit	decimal, 5 places after the point	The account credit limit value, null if not defined
blocked	boolean, Y/N	Block account calls
um_enabled	boolean, Y/N	Allows the account user to access the unified messaging system
follow_me_enabled	boolean, Y/N	Activate follow-me service for this account
ecommerce_enabled	boolean, Y/N	Allows the account's owner to make online payments or setup periodical payments on the account self-care page
out_date_format	string, 16 chars max	Output format of date presentation on the account self-care
out_time_format	string, 16 chars max	Output format of time presentation
out_date_time_format	string, 16 chars max	Output format of full date/time presentation
in_date_format	string, 16 chars max	Input format of date presentation
in_time_format	string, 16 chars max	Input format of time presentation

7. ValidateAccountInfoRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_info	AccountInfo structure	
Note: Omit i_account to check if data may be used to create new account record.		

8. ValidateAccountInfoResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_info	AccountInfo structure	

9. AddAccountRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_info	AccountInfo structure	
Note: i_account will be ignored; most of the fields may be omitted;		

10. UpdateAccountRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
account_info	AccountInfo structure	
Note: i_account is mandatory; only fields that need to be modified should be supplied;		

11. AddUpdateAccountResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_account	Integer	ID of the created/modified account record

12.DeleteAccountRequest structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
i_account	integer	ID of the account record to be deleted

13.DeleteAccountResponse structure

<i>Field</i>	<i>Type</i>	<i>Description</i>
result	integer	1 in case of success, 0 in case of failure