RPC methods

The notification module provides the following RPC methods:

sendMail

```
Struct sendMail(Struct email)
```

This method returns either void in case of success or a value that describes the error.

The parameter `Struct` has the following properties:

Property	Type	Optional	Description
`to`	`String` or `Array`	no	The e-mail address or several where the e-mail will be sent.
`toName`	`String`	yes	The name for the e-mail, if `to` is a `String`, otherwise ignored.
`from`	`String`	yes	The e-mail address of the sender.
`fromName`	`String`	yes	The name for the sender.
`cc`	`String` or `Array`	yes	An e-mail address or several, for 'carbon copy'.
`ccName`	`String`	yes	The name for the cc e-mail, if `cc` is a `String`, otherwise ignored.
`bcc`	`String` or `Array`	yes	An e-mail address or several, for 'blind carbon copy'.
`bccName`	`String`	yes	The name for the bcc e-mail, if `bcc` is a `String`, otherwise ignored.
`subject`	`String`	yes	The subject for the e-mail.
`message`	`String`	no	The e-mail contents.

`to`, `cc` and `bcc` can be `Array`s, in which case they contain `Struct` values (at least one in `to` case), having the properties:

Property	Type	Optional	Description
`emailAddress`	`String`	no	The e-mail address.
`name`	`String`	yes	The name of the recipient.

Note the difference in property name, depending if it's for a `to`, `cc` or `bcc` property.

EXAMPLES:

```
{
   "id": 12,
   "method":"sendMail",
   "params":[{
   "to":"email@domain.org",
   "toName":"John Doe",
   "cc":"other_email@domain.org",
   "ccName":"John Doe 2",
   "bcc":"other_email2@domain.org",
   "bccName":"John Doe 3",
   "from":"whatever@email.com",
   "fromName":"Nemo",
   "subject":"The subject",
   "message":"Message text"
   }]
}
```

Every parameter is optional except the `to` and `message`. Probably `subject` should be set, too, but it's not mandatory. The `from` and `fromName`, if not set, are taken from the configuration file (if set, otherwise they are left empty). The above example allows only one recipient for `to`, `cc` and `bcc`. One can specify more than one recipient using arrays:

```
{
   "id": 12,
   "method":"sendMail",
   "params":[{
   "to":[{
      "emailAddress":"email@domain.org",
      "name":"John Doe"
   }],
   "cc":[{
      "emailAddress":"other_email@domain.org",
      "name":"John Doe 2"
   }],
   "bcc":[{
      "emailAddress":"other_email2@domain.org",
      "name":"John Doe 3"
```

```
    }],
    "from":"whatever@email.com",
    "fromName":"Nemo",
    "subject":"The subject",
    "message":"Message text"
    }]
}
```

The above example contains only one recipient in the arrays, but one can specify more than one if needed.

sendSms

```
Struct sendSms(Struct sms)
```

This method returns either void in case of success or a value that describes the error.

The parameter `Struct` has the following properties:

Property	Type	Optional	Description
`to`	`String`	no	The recipient number (or address book entry: group, contact).
`from`	`String`	yes	The sender number or name.
`message`	`String`	no	The text to send.

The `message` text cannot be bigger than 1520 chars. Attachments will add some overhead due of the links added, so those must be taken into account. To have a single message sent, it must not be over 160 chars.

`to` can contain more than one number (or address book entry), just separate them with commas.

`from` can be either a number or some name. Can contain either 11 alphanumeric chars or 16 numeric chars. If not specified, it will be specified from the configuration file (if specified there).

Just a message:

```
{
    "id": 12,
    "method":"sendMail",
    "params":[{
    "to":"+404832490343",
    "from":"John Doe",
    "message":"Message text"
```

```
    }]
}
```

`from` is optional but probably it should be specified either in the call or in the configuration file.