Configuration

Setting	Default value	Description (short)
batchSizeEmail	5	The number of e-mails sent at once
sendIntervalEmail	180	The wait time period until the next batch is attempted to be sent
lockTimeEmail	60	A module instance marks the database as used with a timestamp, this is the time it's considered locked
batchSizeSms	5	The number of sms messages sent at once
sendIntervalSms	180	The wait time period until the next batch is attempted to be sent
lockTimeSms	60	A module instance marks the database as used with a timestamp, this is the time it's considered locked
cleanupEmailAfterHours	0	Delete an e-mail that was not sent only after this time has passed. Zero means it's never deleted
cleanupEmailAfterRetries	0	Delete the e-mail only after this number of resends was attempted. Zero meanns it's never deleted
cleanupSmsAfterHours	0	Delete a sms that was not sent only after this time has passed. Zero means it's never deleted
cleanupSmsAfterRetries	0	Delete the sms only after this number of resends was attempted. Zero meanns it's never deleted
emailServer		The e-mail server. Something like server-address:587
authUser		The user for authentication
authPassword		Password for authentication

Setting	Default value	Description (short)
certPassword		Needed if the server requests a client certificate and the certificate is password protected
fromName		If set and there is none supplied in the parameters, then this is used as 'from' name for the e-mail
fromEmail		If set and there is none supplied in the parameters, then this is used as 'from' e-mail address
smsServer		Sms server url
smsApikey		Sms server api key

The sms and emails are sent in batches, `batchSize(Email/Sms)` values specify how many of them are sent in 'one shot'. The reason for this is to limit the amount of sms/emails sent in a certain amount of time, because servers have such limits imposed. `lockTime(Email/Sms)` is the time that is considered for the database to be 'locked', before another batch could be sent by another instance. This is because different instances will be sending sms/emails - avoids sending the same one multiple times and also sending batches too close to one another.

`setInterval(Email/Sms)` is the time an instance waits until it tries to send another batch.

Setting `cleanup(Email/Sms)AfterHours` or `cleanup(Email/Sms)AfterRetries` to zero disables deleting old emails/sms messages. Deletion happens when both conditions are true.

After the first attempt to send the e-mail/sms (when calling the rpc method), if sending fails, the message will be in the database with a `retry count` of 1. The messages that are older than 2^`retry count`*`sendInterval(Email/Sms)` (except the first time retry is attempted) will be tried to be sent again and their `retry count` is incremented. If sending succeeds (meaning, 250 or 251 or 252 success code is returned by the server, the server could also send an e-mail and return another code, in which case the module doesn't recognize it as success), the message is deleted from the database, otherwise it's deleted only if it's old enough and with the `retry count` bigger than the limit set in `cleanup(Email/Sms)AfterRetries`. The first attempt for retry can actually be after `sendInterval(Email/Sms)` seconds (most likely after some more time, but not twice of this).

The `batchSize(Email/Sms)` limits this behaviour, so if there are many records in the database resending could take longer than what the above formula gives (for example, for the default value, the first retry should be after 360 seconds if there are only records <= `batchSize` otherwise it could take longer).

The settings refer to a single instance, so if multiple instances are running, the values should be estimated accordingly. For example, if `sendIntervalEmail` is 180 seconds and `batchSizeEmail` is 5 and 3 instances are running, expect 15 e-mails to be attempted to be resent in the 180 seconds period (if there are so many in the database, of course).