Fix Me! This page is not fully translated, yet. Please help completing the translation.
(remove this paragraph once the translation is finished)

SignalByMailReply

Description

This timer checks mail boxes for e-mails and forwards instances if there are any e-mails which have to be replied to generate further instances. The timer has to be used in combination with a VariableDecisionHandler at a node if a decision has to be made after receiving a signal. The sender will be assigend as an actor to all tasks of the current node if the signal has been transmitted successfully. All incoming e-mails will be attached to a corresponding activity as a notice if they can be related to this activity. RegExp will be used to select a number of successful e-mails as part of a variable (with the name given *Nodename*-answer_successful).
The corresponding process instance has to be referred to as

```
-----(${processInstanceId})-----
```

in the text of the e-mail.
The answer can either be written in the first line or optionally in the following form

```
-----[Antwort: ]-----
```

.

Web service name

ProcessInstanceManager

Web service method

signalByMailReply

Parameter

email,password,port,host,limit,moveFolder,piIdRegExp,pDName,NodeNameRegExp,firstline,box,decisionRegExp,signalSuccessMail,processVariable,transitions
new version:
username,password,port,host,moveFolder,pDName,NodeNameRegExp,signalSuccessMail,processVariable,transitions,decisionRegExp

1. e-mail

This entails the complete e-mail address which will be used for generating further e-mails.

2. password

This will show the password to access the desired e-mail account.

3. Port

This provides information on the port corresponding to the e-mail address and the host.
Frequently used ports:
110 *Pop*
995 *Pop including encryption*
143 *IMAP*
993 *IMAP with encryption*

4. Host

Host, e.g., pop3.gmail.com or imap.gmail.com (very often this will entail either pop or pop3 or imap. Smtp is by far less common!). An IMAP mail box will be necesary if moving all e-mails to other folders. The decision on what kind of protocol to use is depending on the mail box employed. If both common protocols are supported by the web sevice, then IMAP should be selected as the preferred protocoll.</note>

5. moveFolder

This will not bee a mandatory field if activated: This is the name of the folder to which successfully processed e-mails for signalling should be moved.

6. pDName

Name belonging to the process definition.

7. nodeNameRegExp

NodeNameRegExp is a regular expression which provides clues on how to find the name of a node. The textual content of e-mails was well as corresponding subject headings will be scanned for this expression.
e.g., \+\+(.+)\+\+ The searched for element would be the nodeName included in brackets. For

example:

```
++Wait++
```

For further elaboration on this topic, please refer to the entry pIdRegExp.

8. signalSuccessMail

If the value is “true” then an e-mail will also be sent to the sender as a reply confirming the successful receipt of a signal. The reply can be configurated within the process by using variables. Please refer to the section provided below for further information on this topic.

9. processVariable

The value of a process variable will be set to the selected transition which then enables the VariableDecisionHandler in the XOR-Gateway to make a decision.

10. leavingTransitions

How will the pathway of the process be found if after the current activity several other activies might follow? Transitions will be included in the timer as parameters by mapping which exact terms are meant to be detected by this transition. At least one transition has to work properly which in turn means that at least one of the correct terms has to be included in the text of an e-mail. For example: Approval:ok-O.K.-okay-yes-y-to-approve-approval-accepting-accept-adopting-all-right-alright-agreed-agreeing;rejected-rejecting:-not-okay-not-ok-not-O.K.-no-never-unacceptable

The transitions according to this node are: approval and rejection To signal an approval a response to the e-mail might entail ok, okay, O.K. etc. Alternatively, to signal a rejection words would be used like “not okay”, no, never etc.

11. reasonRegExp

The decision in the e-mail which has signaled the process is supposed to be saved to be beneficial for the further course of the process. At this point it can be determined what terms should be used in searching the text of the e-mail. If the decision is meant to provide a result extending over several rows the RegExp has to use the ending '?s', for example (.*?)Replies:?s

---

Variables

The replies to invalid messages can be setup either by using a variable or the file loom.properties:
no-tim-user-found-for-signal-by-mail-reply
user-not-actor-for-signal-by-mail-reply
found-no-transition-for-signal-by-mail-reply

found-no-node-for-signal-by-mail-reply
signal-done-for-signal-by-mail-reply
found-no-processinstance-for-signal-by-mail-reply Only the configuration using loom.properties would be possible in this case.
The correct answer has to be configurated in the following way:
signal-successful-signal-by-mail-reply

The name of the folder which has to be accessed can also be configurated by using loom.properties via *inbox-name-signal-by-mail-reply*.

> Attention!
>
> > Depending on the exact stage of the procedure the following variables can be implemented:
> > *emailFrom*, This indicates the sender
> > *userNameLast*, This entry shows the surname of the T!M user
> > *userNameFirst*, This variable represents the given name of the current T!M user
> > *nodeName*, The name of the node which should be signalled is provided here
> > *nodeEndDate*, This displays the date when this node was terminated. This variable is meant to be used only after termination of a node when trying to signal
> > *nodeEndTime*, This indicates the point in time when this node was terminated. This variable is intended to be used only after termination of a node when trying to signal
> > *lastUserNameLast*, This shows the surname of the user who terminated this node. This is supposed to be used only when trying to signal afer termination of a node
> > *lastUserNameFirst*, This displays the given name of the user who terminated the node. Only after termination of a node is this meant to be used for signalling

After sending the signal the following variables are available: *nodename*-lastActor *nodename*-signalDateTime (Format: “yyyy-MM-dd HH:mm:ss”)

Procedure

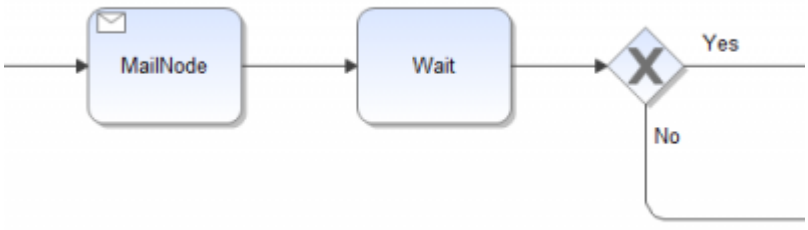
In a first step the mailbox specified in the timer will be opened to receive incoming e-mails. As a default setting the folder *Inbox* will be retrieved. This setting can be changed by configurating loom.properties with the parameter *inbox-name-signal-by-mail-reply*.
In the second step the first five e-mails to be found will be processed. This number of e-mails can be adapted by configurating loom.properties with the parameter *inbox-name-signal-by-mail-reply*. In each e-mail a search will be conducted to find a text according to the form of —–(1234)—– at first in the textual content and then in the subject heading. This text represents the identification (ID) of the T!M process instance.
The message will be ignored if neither an ID nor a corresponding process instance can be found within T!M.
In the next step a T!M user will be sought while sending an e-mail. The sender will receive an e-mail if no user can be found.
In the following stage the name of an activity will be searched for in the content as well as in the subject heading of the e-mail by using NodeNameRegExp. In case a name is found an examination will

be conducted to check if this process instance is pertaining to the delivered process definition. If this is not the case then this e-mail is not supposed to be assigned to this timer. This e-mail would then be also ignored and no e-mail would be sent as a reply.
If this timer is the proper one pertaining to this particular e-mail an examination will be started to ensure that the process is also focused on the correct activity. If this is the case the e-mail will be attached to an activity as a notice. In addition, the sender will receive a corresponding message.
Now the T!M user who sent the e-mail will be examined to ensure if he is responsible for one of the tasks related to the current activity. This user will then in turn be informed if this is not the case.
At this later stage it will be checked if there are several follow-up activities. If there is only one existindg activity then the user wiil be assigned the status of an Actor who is responsible for all tasks.
There are several possibilities to continue with the process and to check if the Mail entails a corresponding decision.
Afterwards, the user will receive the status of an actor assigned to the task. Process variables will then be generated and the process forwarded. An e-mail will also be sent if this is desired.
All e-mails which had been answered will be moved to the folder errorMessages if this folder exists.
The initial e-mail which forwarded the whole process will be moved to the especially configurated folder.

Beispiele


Parameter:

```
johnm1477@gmail.com,123456,993,imap.gmail.com,,SignalByMailReply mk4,\-
+\{(Wait.*)\}\-+,true,decision,no:abgelehnt-nein-nok;jo:genehmigt-ok-
ja,(.*?)schrieb ?s
```



Beispielprozess:

Einfaches Beispiel für den Mailtext:

```
Hallo,

für Entscheidung "Nein" antworten Sie bitte mit "abgelehnt", "nein", oder
```

```
"nok".

Für Entscheidung "Ja" antworten Sie bitte mit "genehmigt", "ja", oder "ok".


Viele Grüße
Ihr T!M-System


-----(${processInstanceId})-----{Wait4}-----
```

Abhängigkeiten

T!M Version : in dieser Konfiguration ab Version 3.6.2