Escalation in workflows HOWTO

The following applies to midPoint 3.6 and later. The schema for the configuration is not stable enough yet, so some changes might occur before 3.6 release.

When using workflows, it is often necessary to make sure all work items (and therefore, approval processes) are completed in a given time. In order to ensure this, two features have been introduced in midPoint 3.6:

1. work item duration and deadline,
2. timed actions.

Work item duration and deadline

The work item *duration* is an information about how long the work item is expected to "live". It does nothing by itself, however. It is used to be shown in GUI and in notifications, and it provides a convenient basis for timed actions, which deliver the actual handling of overdue (or soon to be overdue) tasks.

The duration is set like this:

Setting work item duration

```
<policyRule>
   <policyConstraints>
      ...
   </policyConstraints>
   <policyActions>
      <approval>
         <approvalSchema>
            <level>
               <duration>P7D</duration>
               ...
            </level>
         </approvalSchema>
      </approval>
   </policyActions>
</policyRule>
```

We can see that the duration for work items at this level is set to 7 days. (Currently midPoint can deal with duration specified as calendar time. Business days are not available as of now.) The deadline is therefore set to 7 days after the moment of work item creation.

Timed actions

In order to actually complete or delegate/escalate a work item when a deadline comes near, *timed actions* have to be used. They are specified again for given approval level(s). Specification of such action contains two basic parts:

1. time(s) when the action(s) are to be invoked,
2. action(s) themselves.

Automatic work item completion

Perhaps the simplest example is an automatic completion:

Setting automatic rejection at deadline

```
<timedActions>
   <!-- no time set - executes at deadline -->
   <actions>
      <complete>
         <outcome>reject</outcome>
      </complete>
   </actions>
</timedActions>
```

In this case, there's no time specified. It means that the action is executed right on work item deadline. And the action is simply: complete with the result of "reject".

Besides completion, other possible actions are:

1. escalation or delegation,
2. sending custom notification.

Automatic work item delegation/escalation

Work item delegation is a mechanism (introduced in midPoint 3.6) that changes current assignee(s) of the work item. There are two methods of delegation: *add* assignees to current one(s), and *replace* existing assignee(s) by specified one(s). The latter is a default one. When doing the delegation, other changes to a work item can be done:

1. deadline of the work item can be postponed,
2. "outcome if no approvers" flag may be changed,
3. new escalation level may be entered.

If a work item has multiple assignees (actors), it behaves in a way similar to "first decides" approval stage: whoever first approves/rejects it, decides on the whole work item. It is currently not possible to mimic "all must approve" behavior. The difference between multi-approver work item and multi-approver "first decides" approval stage is subtle but real. For example, multi-approver work item is delegated/escalated as a whole; whereas multi-approver "first decides" stage creates single independent work item for each approver; which can be escalated independently of the others.

Actually, the third point In the above list is the only difference between *delegation* and *escalation*. This deserves a little comment.

Each work item has so called *escalation level*. (This describes how big the problem has become.) Each escalation level has a *number*: 0 means no escalation, 1 is the first one, 2 the second, etc. It can also has a *name* and a *displayName* - in the same way as approval stages (originally called levels, by the way).

When a work item is created, it starts with escalation level number 0. After executing any escalation timed action, the number is increased, and the name /displayName is set to values specified in the `<escalate>` directive.

Of course, when the work item is completed (or cancelled) and a new one is created, the escalation level of the new work item is again 0.

An example of simple escalation configuration:

Simple escalation instruction

```
<escalate>
  <approverExpression>
    <script>
      <code>
        midpoint.getManagersOidsExceptUser(workItem.assigneeRef)
      </code>
    </script>
  </approverExpression>
  <duration>P9D</duration>
  <escalationLevelName>Line manager escalation</escalationLevelName>
</escalate>
```

This configuration fragment causes the following when invoked:

1. current assignees (actors) are replaced by their managers,
2. new deadline of the work item is set to be 9 days in the future (relative to the current moment),
3. new escalation level is entered (i.e. increasing level number by one); and its name is set to "Line manager escalation".

A more complex example

Here we'll describe real-life scenario:

1. After a work item is created, user has 5 days to complete it.
2. If the work item is not completed by that time, it is automatically escalated to user's managers. (But the user remains to be able to complete it even after the escalation.)
3. After escalation, there will be additional 9 days available to complete the item.
4. If the work item is not completed even in this time, it is automatically rejected.
5. Before each of these two automated actions, two reminders are sent: two and one day before it. (Of course, when the automated action occurs, a notification is sent as well.)

The configuration is like this:

Configuration for more complex auto-escalation/auto-complete scenario

```
<approvalSchema>
   <level>
      <duration>P5D</duration>
      <timedActions>
         <!-- no time set - executes at deadline -->
         <actions>
            <escalate>
               <approverExpression>
                  <script>
                     <code>
                        midpoint.getManagersOidsExceptUser(workItem.assigneeRef)
                     </code>
                  </script>
               </approverExpression>
               <duration>P9D</duration>
               <delegationMethod>addAssignees</delegationMethod>
               <notifyBeforeAction>P1D</notifyBeforeAction>
               <notifyBeforeAction>P2D</notifyBeforeAction>
               <escalationLevelName>Line manager escalation</escalationLevelName>
            </escalate>
         </actions>
      </timedActions>
      <timedActions>
         <!-- no time set - executes at deadline -->
         <actions>
            <complete>
               <outcome>reject</outcome>
               <notifyBeforeAction>P1D</notifyBeforeAction>
               <notifyBeforeAction>P2D</notifyBeforeAction>
            </complete>
         </actions>
         <escalationLevelFrom>1</escalationLevelFrom>
         <escalationLevelTo>1</escalationLevelTo>
      </timedActions>
   </level>
</approvalSchema>
```

First part of the configuration is like this:

1. First timed action executes at deadline (because there's no explicit time specified) (line 5)
2. It is an escalation, with approvers given by specified expression that returns managers of current approvers (line 11). They are added to the current ones, not replacing them (line 16).
3. New deadline is set to 9 days from then-current time (line 15).
4. There are notifications sent 2 and 1 days before the escalation occurs (lines 17 and 18).

After moving to the escalation level 1, another timed action comes into play. Elements `escalationLevelFrom` and `escalationLevelTo` (lines 32-33) restrict the second timed action to be applied only to this level. (The absence of *both of them* means "escalation level 0". If only one is absent, it means that there's no limitation at that particular side.)

It this case, the configuration can be read as:

1. The timed action executes again at deadline; this time the new one (line 24).
2. It is an automatic completion, with the outcome of *reject* (line 27).
3. There are notifications sent 2 and 1 days before the escalation occurs (lines 28 and 29).

This configuration can be seen "in action" in the Strings story test.