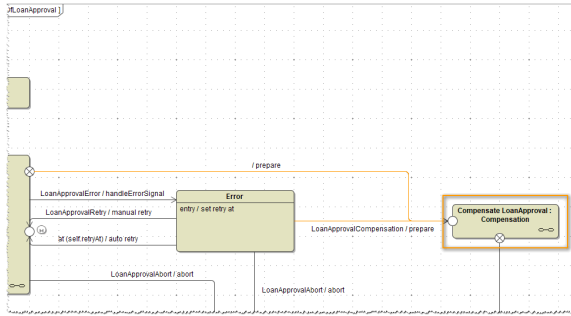


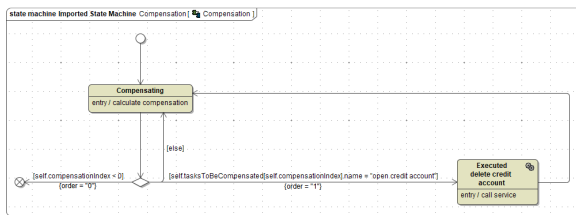
# BPMN Task Compensation

BPMN offers the possibility to assign **compensation tasks** to tasks. These compensation tasks describe the activities to be performed, if the original task has to be rolled back. Compensation tasks are triggered within the BPMN model by compensation throwing events. Having defined a compensation task on at least one task, the root state machine of the generated module contains a default compensation state machine that manages all compensation tasks of the process.

The compensation state will only be generated, if a **Compensation Boundary Event** and a corresponding task tagged to be **Is For Compensation** is defined.

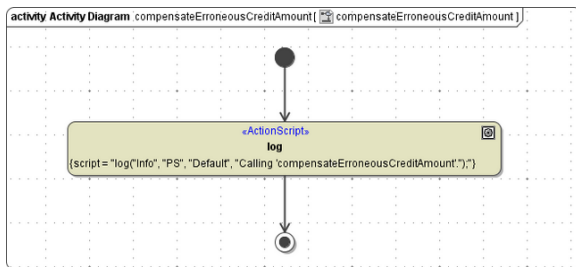


For each task needing to be compensated an appropriate transition to the corresponding state exists.



The service interface part (e.g. **BPMNLoanApprovalServices**) contains a persistent state class derived from the abstract definition generated from the BPMN. This real class contains an overridable operation for each compensation task defined in the BPMN, resp. for each compensation state entry.

The activity diagram of this operation can be amended to execute whatever is needed as a compensation of the original task.



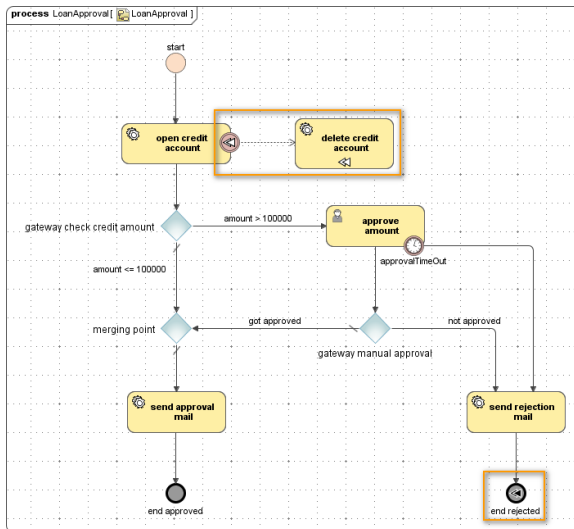
## Start Compensation via a Compensation End Event

Generally, if compensation was defined in the BPMN diagram, the UML model generated from the BPMN description is ready to perform compensation. But it is on the modelers side to decide at which point of the process the compensation should be activated.

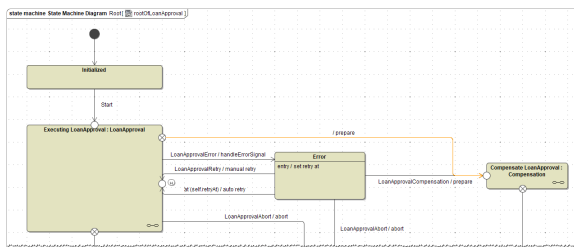
Within the **LoanApprovalExample**, it is defined in the BPMN diagram, that the compensation should be started, if the credit request is rejected (**end rejected**).

### On this Page:



- [Start Compensation via a Compensation End Event](#)
- [Start Compensation via a Compensation Throwing Intermediate Event](#)
- [Start Compensation via the Error State](#)

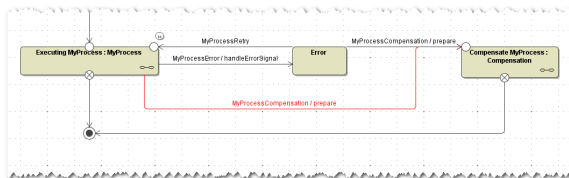
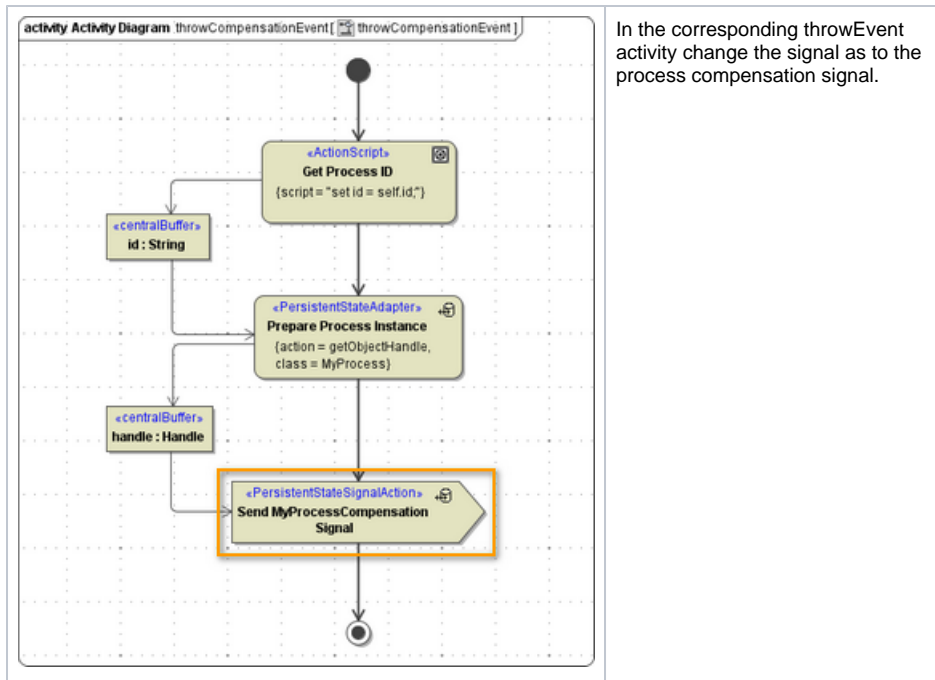


On end rejected, in the root state machine of the generated loan approval service the state of the process has to change to **Compensate LoanApproval**. The transition coming from end rejected has to be re-routed to the entry point of **Compensate LoanApproval**. Behavior type of this transition has to be the same as assigned to the **LoanApprovalCompensation** signal. The activity can be copied and pasted.



## Start Compensation via a Compensation Throwing Intermediate Event

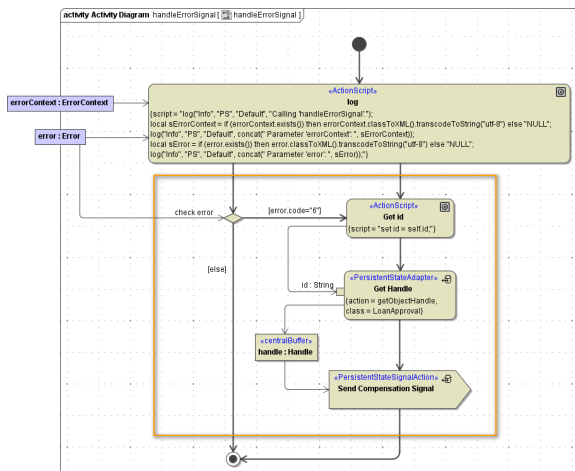
BPMN shape	BPMN description
	<p>A <b>Compensation Throw Intermediate Event</b> would be used to describe in BPMN a point where to start compensation.</p>
UML representation	UML description
	<p>Having defined a <b>Compensation Throw Intermediate Event</b>, in the UML model you will find a <b>compensationEvent</b> state (see <a href="#">Intermediate Events</a>).</p>



In the root state machine draw an additional transition from the process sub state machine to the compensation sub state machine, so that the compensation signal sent before can be handled.

## Start Compensation via the Error State

An error occurring the process changes into error state. Within activity **handleErrorSignal** the compensation can be triggered e.g. on given conditions.



In the diagram shown above compensation is started, if error "6" occurs (which is a division by zero).