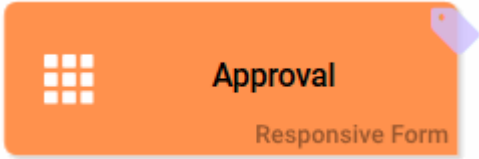


Modeling Conventions

There are no strict specifications for modeling processes in BPaaS. However if the process is supposed to be executable in an app, then some basic rules need to be followed when creating an EPC model. The modeling conventions are non-binding **Best Practice** recommendations to simplify modeling and to prevent execution errors.

Naming of Elements

When naming your elements keep the following recommendations in mind:

	<p>Chose short titles. The name should be as long as necessary and as short as possible.</p> <p>Examples:</p> <ul style="list-style-type: none">• Leave Request• Application• Approval
---	--

On this Page:

- [Naming of Elements](#)
 - [Special Cases](#)
- [EPC Modeling](#)
 - [Modeling Recommendations](#)
 - [Branching](#)
 - [AND Branching](#)
 - [XOR Branching](#)
 - [Loops](#)
 - [Nesting with Sub-EPCs](#)
 - [When is a process "too big"?](#)
 - [Practical Tips for Modelers](#)

Related Pages:

- [Modeling Processes](#)
 - [EPC Elements](#)
 - [Modeling EPCs](#)
 - [Form Combinations](#)
- [Service](#)
 - [The Processes of ACME Corp.](#)



Delivery Notification

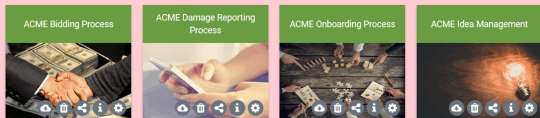
E-mail

The name should reflect the element's content. Avoid obvious name components, such as form, mail, etc.

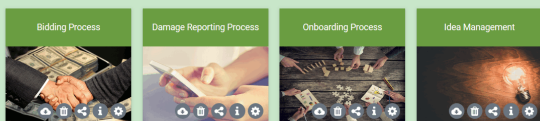
Examples:

- **Approval** (instead of Approval Form)
- **Acceptance** (instead of Acceptance Mail)
- **Delivery Notification** (instead of Delivery Notification Mail)

Not like this



Better



Do not start your element's name with the company name or with stating the element type (project, app...).

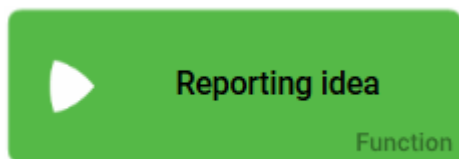
Examples:

- **Leave Request**
(instead of ACME Leave Request)
- **Idea Management**
(instead of Idea Management Project)
- **Bidding Process**
(instead of Project Bidding Process)

Avoid repetition:
The more elements start with the same description, the more chaotic the element overview becomes and the harder you have to search for your elements and projects.

Special Cases

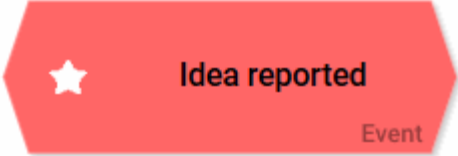
We will consider the elements **Function**, **Event** and **Process App** (start link) separately, because their naming requires consideration of their specific characteristics.



Functions display actions and should therefore receive an active name.

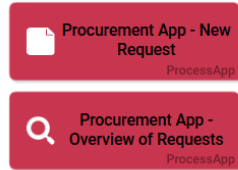


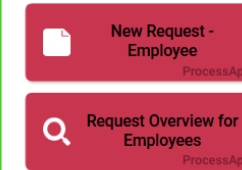

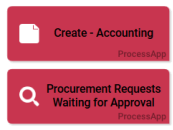
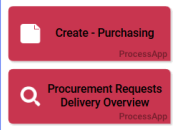
Examples:

- Reporting idea
- Sending e-mail
- Choosing product

	<p>Events indicate that an action has been completed and a state has occurred. Therefore, we recommend to use a passively formulated name.</p> <p>Examples:</p> <ul style="list-style-type: none"> • Idea (has been) reported • Proposal accepted • Product chosen
---	---



Use **noun** and **verb** for naming functions and events ("Reporting idea"). Avoid nominalization ("Listing of idea").

Process App Start Links		
<p>Not like this</p> 	<p>Better</p> 	<p>The Process App start links Create and Overview shall receive names to describe that both Create and Overview belong together.</p> <p>Positive Examples:</p> <ul style="list-style-type: none"> • Create New Request / Request Overview • Approval / Approval Overview
Role-based Process App Start Links		
<p>Not like this</p> 	<p>Better</p> 	<p>In <u>role-based</u> a pps with various users the names of the Create and Overview elements should also reflect for which users the specific start links were created.</p> <p>Positive Examples:</p> <ul style="list-style-type: none"> • Leave Request - Employee / Overview Leave Requests for Employees • Approval Supervisor / Approval Overview Supervisor
Expert Advice		
To avoid the roles names in each start link, use frames to visualize the role's start links:		
<p>Start Links EMPLOYEE</p> 	<p>Start Links ACCOUNTING</p> 	<p>Start Links PURCHASING</p> 

EPC Modeling

An EPC should be modeld in a clear fashion, the course of the process should be traceable.

Before you start modelling, decide for one approach and ask yourself:

- How would you like to display the main process?
- Where do you place which elements?

- How do you connect your elements?
- How do you display branchings and loops?
- How shall different process branches be reunited?

Once you have decided for one approach use it in all models. This increases overall transparency and user comprehension.



Visit page [Form Combinations](#) for an overview of which and how many forms can be modeled in a single process step.

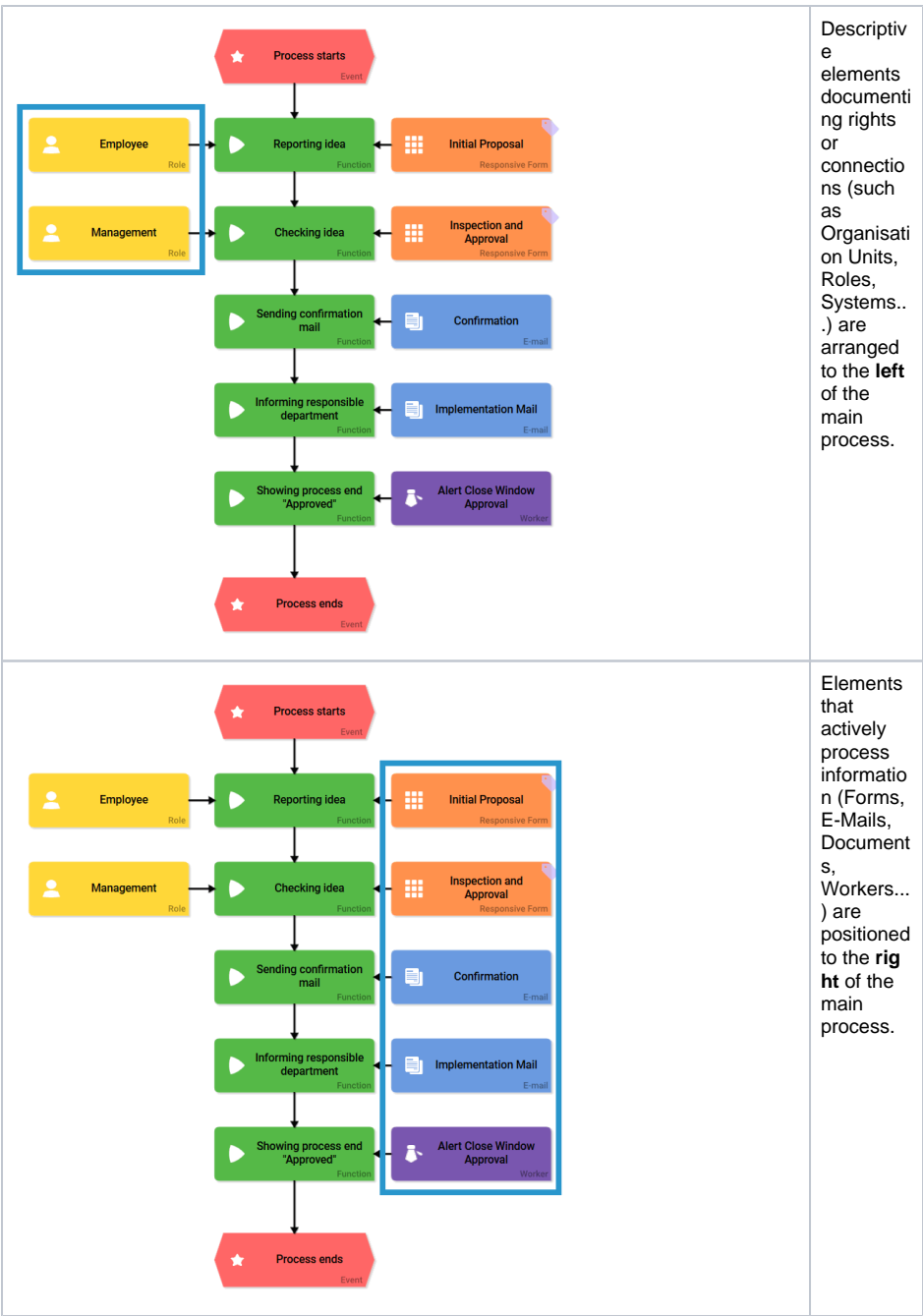


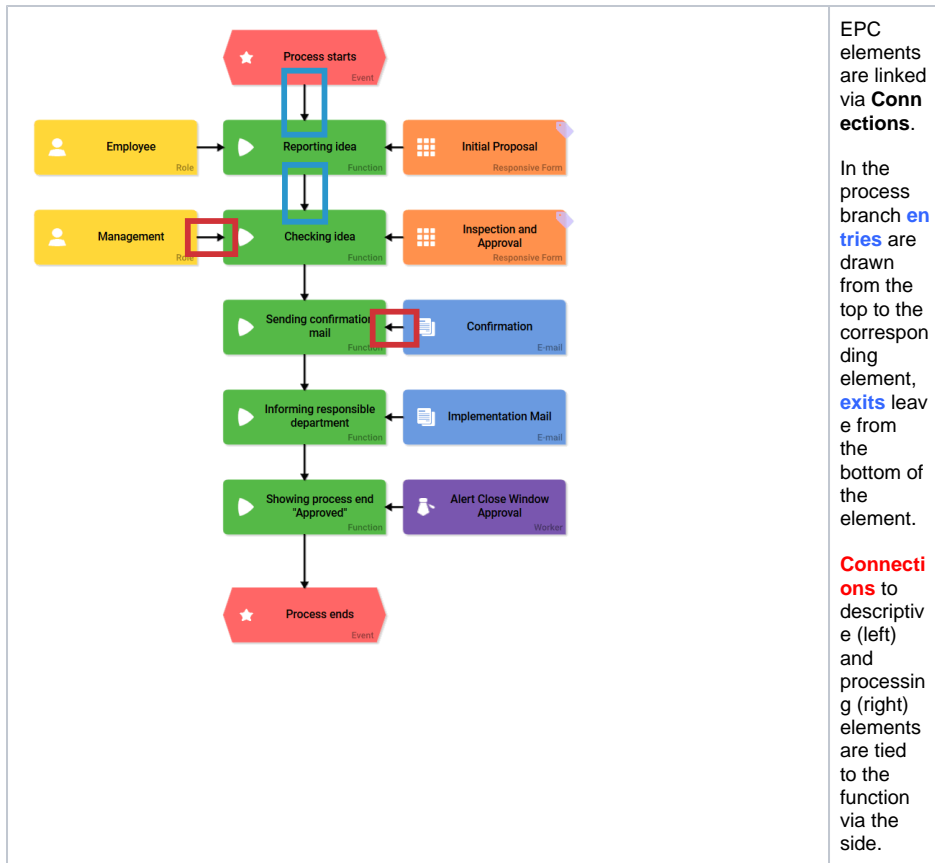
Caution when deleting process steps out of a (productive) EPC

- Instances, which are situated in this process step can not be opened again after deletion. When trying to open an affected instance an error message will be displayed in the overview. The message contains name and ID of the deleted process step.
- Every function has its own ID in the database. A new function with the same name cannot therefore replace a deleted function.

Modeling Recommendations







See chapter [EPC Elements](#) for more informations regarding the various elements, their configuration possibilities as well as available entries and exits.

Branching

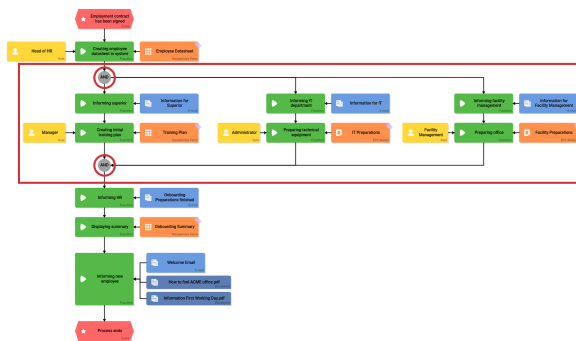
Branchings separate the process in multiple branches. Executable Process Apps may contain [AND Branchings](#) and [XOR Branchings](#).

AND Branching

In AND branchings multiple branches are processed independently at the same time.

- The parallel running branches should therefore be modeled **side by side**.
- The AND branching begins with an opening **AND Connector** and is ended by a closing **AND Connector**. These connectors should be modeled **in front of** and **after** the parallel running process branches.
- If one or more branches within the parallel execution also contain multiple steps requiring branching or nesting, then it is recommendable to use **sub-EPCs** to preserve the model's clarity.
- It is recommendable to insert a **summary** of results of the parallel steps right after reuniting the parallel branches behind the closing AND. This way everyone involved in the process or the corresponding user receives an overview over the whole parallel process.

Example: ACME Onboarding Process

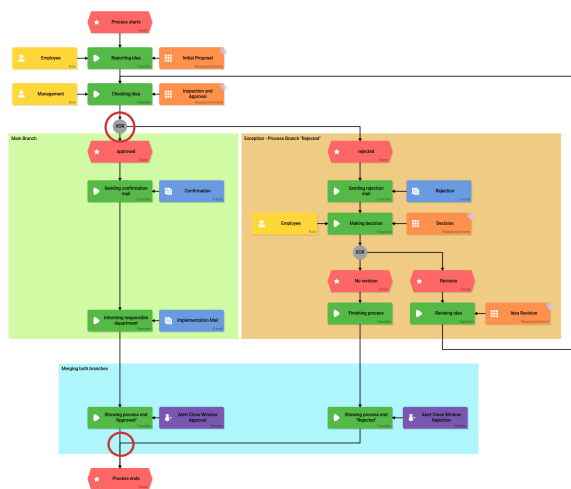


XOR Branching

XOR branching is always tied to conditions defined in the **Event** element.

- For XOR branchings we recommend to display the **general case** in the main branch (top-down modelling) and to display the **exception** to the **right** of the main process.
- To increase the main process' clarity, the merger / the next **mutual** process step should be placed **below** both branches.

Example: ACME Idea Management Process



See chapter [Configuring EPC Branching](#) for a detailed description of both branching methods.

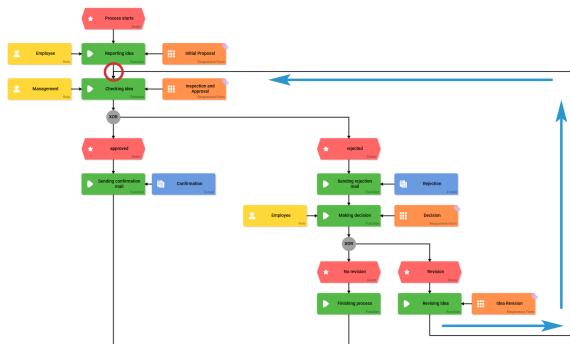
Loops

Integration of a **loop** allows you to jump back from a progressed process step to one of the earlier steps. During modeling make sure that you use the connection as entry into the function from which the process shall restart.



When modeling a loop, mark the **start event** as such (compare page [Event](#)). If the option **start event** is not activated, then the jump back within the EPC can only occur from the **second function**.

Example: ACME Idea Management Process



Nesting with Sub-EPCs

If your process model becomes too big and confusing, consider the use of sub-EPCs. Sub-EPCs help to structure a process which makes the main process clearer and keeps it manageable. Sub-EPCs are modeled in their own [EPC Model](#). You can for example outsource sub-processes in further EPC Model elements.

When is a process "too big"?

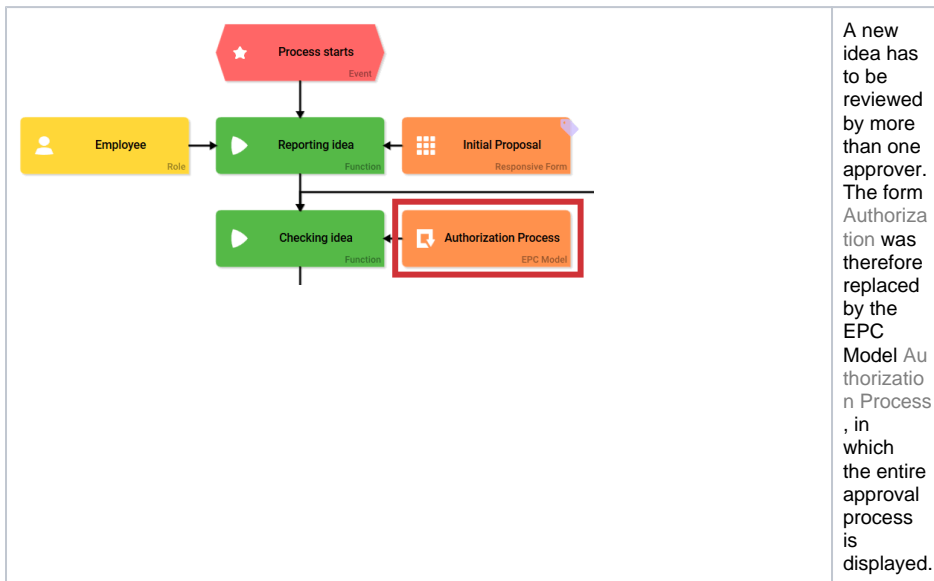
Indicators could be:

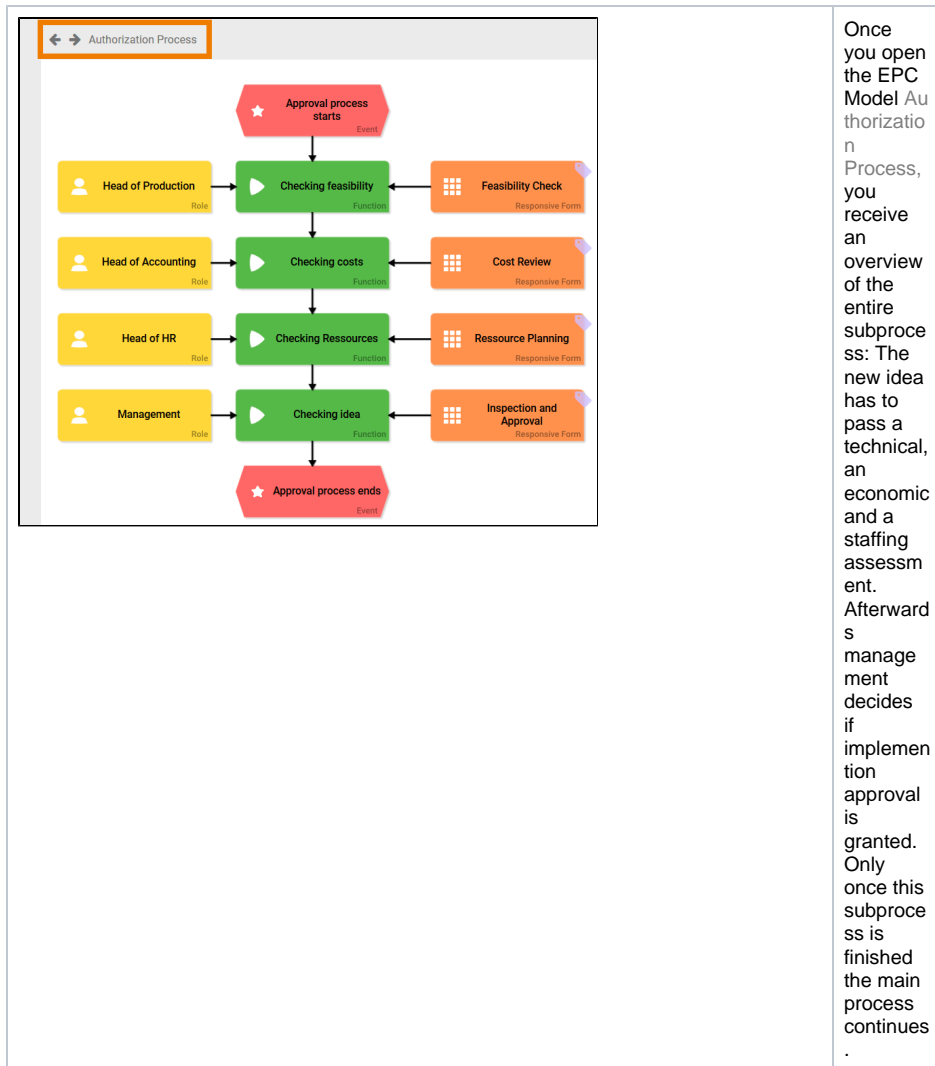
- You have to scroll a lot to find the process step you are looking for.
- Your process already contains a double digit amount of **functions**.
- The EPC contains multiple branchings and/or loops.
- Your model contains technical independent subprocesses.



How to outsource an already modeled subprocess can be reviewed at page [Nesting EPCs](#).

Example: Outsourcing a subprocess





Practical Tips for Modelers

- Use the EPC element [Note](#) to capture ideas or tips directly on the work surface. This is especially useful if multiple users have modeling authorization.
- Make sure to discourage multiple users from working simultaneously in the same model's profile - one user's changes could be unknowingly overwritten by another. **A synchronization of the work done in different tabs or windows will not occur!**
- Before starting to model take a picture of your current process:
 - Which process steps exist?
 - Which general cases and exceptions exist (and what conditions are defined)?
 - For which process steps forms need to be designed?
 - Do process steps exist that could be executed simultaneously?
 - Is it possible that you would like to add extensions to some process steps, for example adding connections to external systems, inserting automated e-mail dispatch or loops?
 - Would some subprocesses be better suited for their own sub-EPC?
- Be generous in your first model design and leave spaces when arranging the elements - this makes it easier to add elements later. You could also use placeholders, for example [Note](#) elements with corresponding remarks, that could be replaced with fitting EPC elements later on.