BPMN – Bussiness Process Model Notation

Este doar un standart de notatii

Este o metodologie ce contine anotatii pentru bussiness events

Bussiness process – serie de pasi repetabili facuti de echipa, om, system etc. pentru a ajunge la un rezultat concret

**Event**

Sunt chestii care au loc in timp, la un moment concret

Un event este ceva care are loc in proces si porneste o actiune cocnreta

[Start Event](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/start-event.html) – porneste procesul



[Intermediate Event](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/intermediate-event.html) – ceva ce se petrece in timpul procesului, ca trimiterea unui email



[End Event](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/end-event.html) – finisarea eventului



Putem avea mai multe start si end events intr-un proces

# Start Event

start event None Start Event

start event 1 Interrupting - Message Start Event – acest event se porneste odata ce a primit un mesaj de la un alt proces ce il va porni pe acesta. Message e doar o informatie trimisa de la un proces la altul

start event 2 Non-interrupting - Message Start Event

start event 3 Interrupting - Timer Start Event – incepe un event la o ora sau odata la un interval stabilit(repetitiv)

start event 4 Non-interrupting - Timer Start Event

start event 6 Interrupting - Conditional Start Event

start event 7 Non-interrupting - Conditional Start Event

start event 8 Interrupting - Signal Start Event – porneste un alt event si asteapta sa primeasca mesaj de la el

start event 9 Non-interrupting - Signal Start Event

start event 10 Interrupting - Multiple Start Event

start event 11 Non-interrupting - Multiple Start Event

start event 12 Interrupting - Parallel Multiple Start Event

start event 13 Non-interrupting - Parallel Multiple Start Event

start event 14 Interrupting - Escalation Start Event

start event 15 Non-interrupting - Escalation Start Event

start event 16 Interrupting - Error Start Event

start event 17 Interrupting - Compensation Start Event

# Intermediate Event

intermediate event 1 Interrupting - None Intermediate Event

intermediate event 2 Catch - Message Intermediate Event

intermediate event 3 Interrupting - Boundary - Catch - Message Intermediate Event

intermediate event 4 Non-interrupting - Boundary - Catch - Message Intermediate Event

intermediate event 5 Throw - Message Intermediate Event – pune pe pauza procesul pana nu primeste un specific mesaj de la un alt proces

intermediate event 6 Timer Intermediate Event – face ca procesul sa fie delayed pana nu se atinge un interval de timp. E bun pentru a face o pauza, gen facem un request sa fie executat peste 3 zile abea, nu deodata in proces.

intermediate event 7 Interrupting - Boundary - Timer Intermediate Event

intermediate event 8 Non-interrupting Boundary - Timer Intermediate Event

intermediate event 9 Conditional Intermediate Event

intermediate event 10 Interrupting - Boundary - Conditional Intermediate Event

intermediate event 11 Non-interrupting - Boundary - Conditional Intermediate Event

intermediate event 12 Catch - Signal Intermediate Event – porneste un event extern, din alta diagrama si va continua cand acel event va termina

intermediate event 13 Interrupting - Boundary - Catch - Signal Intermediate Event

intermediate event 14 Non-interrupting - Boundary - Catch - Signal Intermediate Event

intermediate event 15 Interrupting - Boundary - Throw – e folosit cand procesul a fost pornit de un

asa intermediate event 13 si odata ce se ajunge la el, il va anunta

intermediate event 16 Catch - Multiple Intermediate Event

intermediate event 17 Interrupting - Boundary - Catch - Multiple Intermediate Event

intermediate event 18 Non-interrupting Boundary - Catch - Multiple Intermediate Event

intermediate event 19 Throw - Multiple Intermediate Event

intermediate event 20 Catch - Parallel Multiple Intermediate Event

intermediate event 21 Interrupting - Boundary - Catch - Parallel Multiple Intermediate Event

intermediate event 22 Non-interrupting Boundary - Catch - Parallel Multiple Intermediate Event

intermediate event 23 Catch - Escalation Intermediate Event

intermediate event 24 Interrupting - Boundary - Catch - Escalation Intermediate Event

intermediate event 25 Non-interrupting - Boundary - Catch - Escalation Intermediate Event

intermediate event 26 Throw - Escalation Intermediate Event

intermediate event 27 Boundary - Catch - Error Intermediate Event

intermediate event 28 Boundary - Catch - Compensation Intermediate Event

intermediate event 29 Throw - Compensation Intermediate Event

intermediate event 30 Catch - Link Intermediate Event

intermediate event 31 Throw - Link Intermediate Event

intermediate event 32 Boundary - Catch - Cancel Intermediate Event

End Events – finiseaza un proces

end event 1 None End Event – incheie thread la procesul curent

end event 2 Message End Event – trimite un mesaj catre un event parinte, adica da throw

end event 3 Signal End Event – trimite un signal la un parinte

end event 4 Multiple End Event

end event 5 Escalation End Event

end event 6 Error End Event – eror care trimite o eroare la un alt proces

end event 7 Compensation End Event

end event 8 Cancel End Event

end event 9 Terminate End Event – incheie toate threadurile la toate procesele, deci si la events parinti

**Tasks**

Ceva ce trebuie facut, un fel de pas facut in proces

 Abstract Task -



 Service Task – chestii automatizate de system, in backed, ca Java. Fara lucru de om, doar de sytem



 Send Task



 Receive Task



 User Task - Deci, un om concret face lucrul ,de ex completeaza forma etc.



 Manual Task – facute de user, cu mana, fara IT.



 Business Rule Task



 Script Task



**Gateway**

Ne ajuta sa luam decizii, unde sa mergem in bussiness diagram

Ele separa event in mai multe branches sau bucati

 Exclusive Gateway - without Marker



 Exclusive Gateway - with Marker(either or) – alege o conditie care e true si o urmeaza, doar una



 Inclusive Gateway(Or) – se executa toate conditiile care sunt true, deci nu e neaparat ca una doar sa fie true, dar cel putin una



De ex:

cond1: if credit > 1000

cond2: if credi > 10000

credit = 10000, deci ambele sunt true si ambele se vor executa

 Parallel Gateway – se executa mai multe taskuri in paralel, toate unite la el. Pana nu termina toate executia, nu se merge mai departe. Nu au conditii



 Complex Gateway

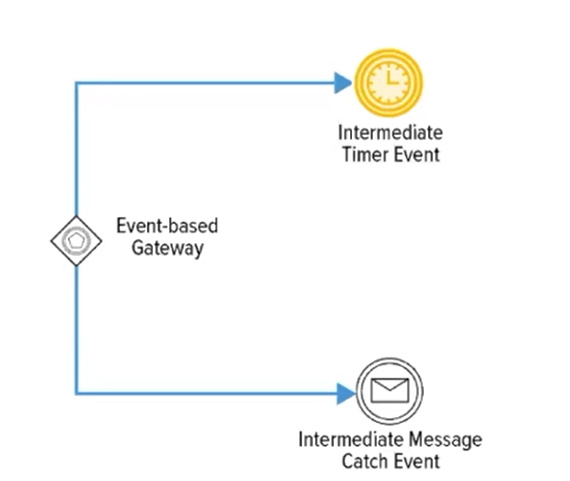


 Event-Based Gateway



 Event-Based Gateway to Start a Process – e unit la mai multe events si cel care termina primul, va anunta si cu acela se va merge mai departe





 Parallel Event-Based Gateway to Start a Process

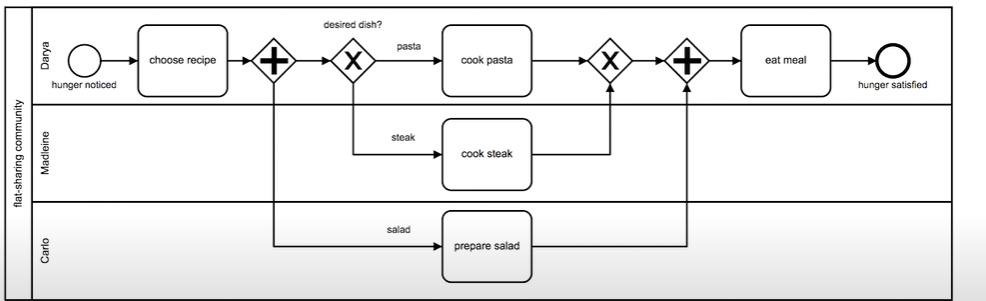


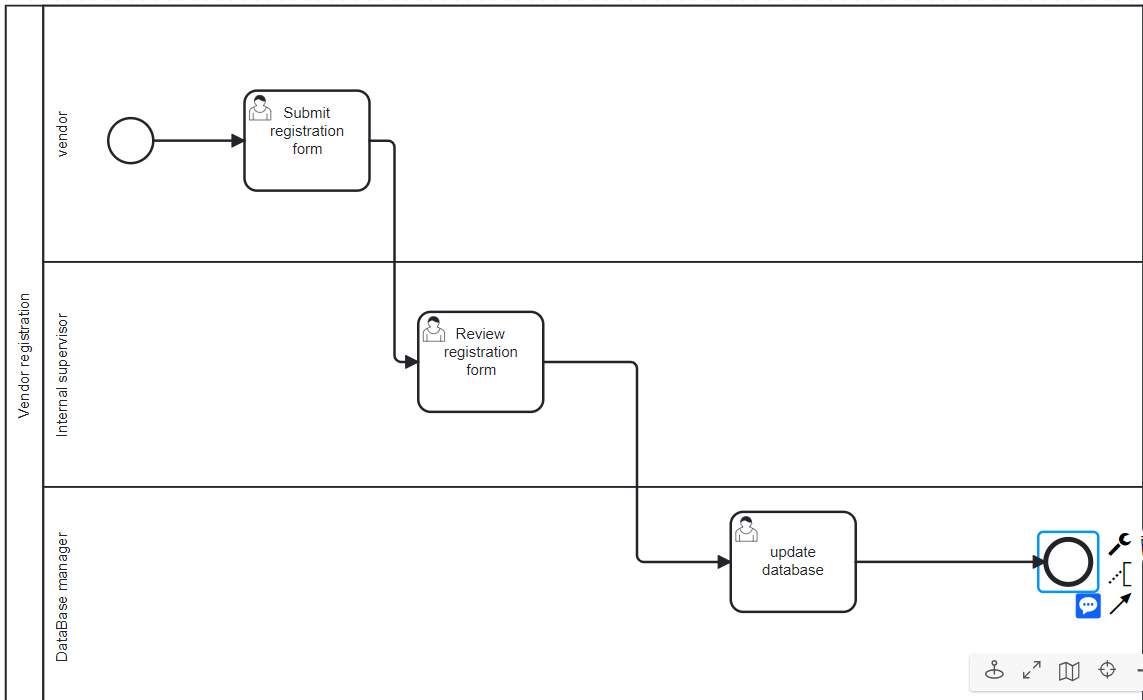
**Swimlane Basics**



**Pool** – container pentru a reprezenta un untreg proces de la A la Z, si are un participant ce are acces la intregul pool

**Line** – o parte din proces, care poate fi separat in dependenta de ceva, de ex, de userul care o foloseste, de persoana care o foloseste, sau gen un line per rol ce face o actiune din proces etc. Este doar ceva mai mult asbtract, procesul nu e influentat propriu zis nicidecum. E folosit pentru a categoriza procesul, de ex pe departamanete . Userul initial are acces la intregul pool





# Flow

[flow 1](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/sequence-flow.html) [Sequence Flow](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/sequence-flow.html)

[flow 2](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/message-flow.html) [Message Flow](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/message-flow.html)

[flow 3](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/association.html) [Association](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/association.html)

[flow 4](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/data-association.html) [Data Association](https://cloud.trisotech.com/bpmnquickguide/bpmn-quick-guide/data-association.html)

# Call Activity

Daca subprocess este o parte din proces, call activity cheama un process diferit chiar, unul extern, complet separat.

 Abstract Call Activity



 User Call Activity



 Manual Call Activity



 Business Rule Call Activity



 Script Call Activity

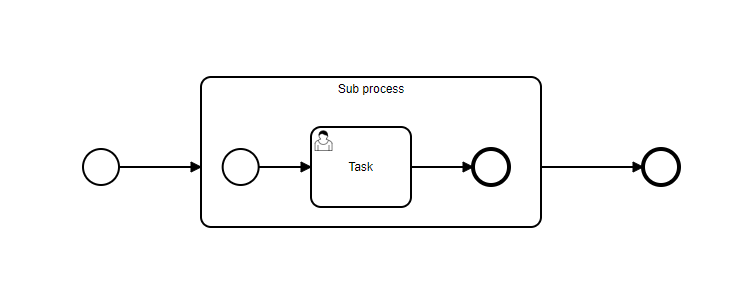


**Subprocess**

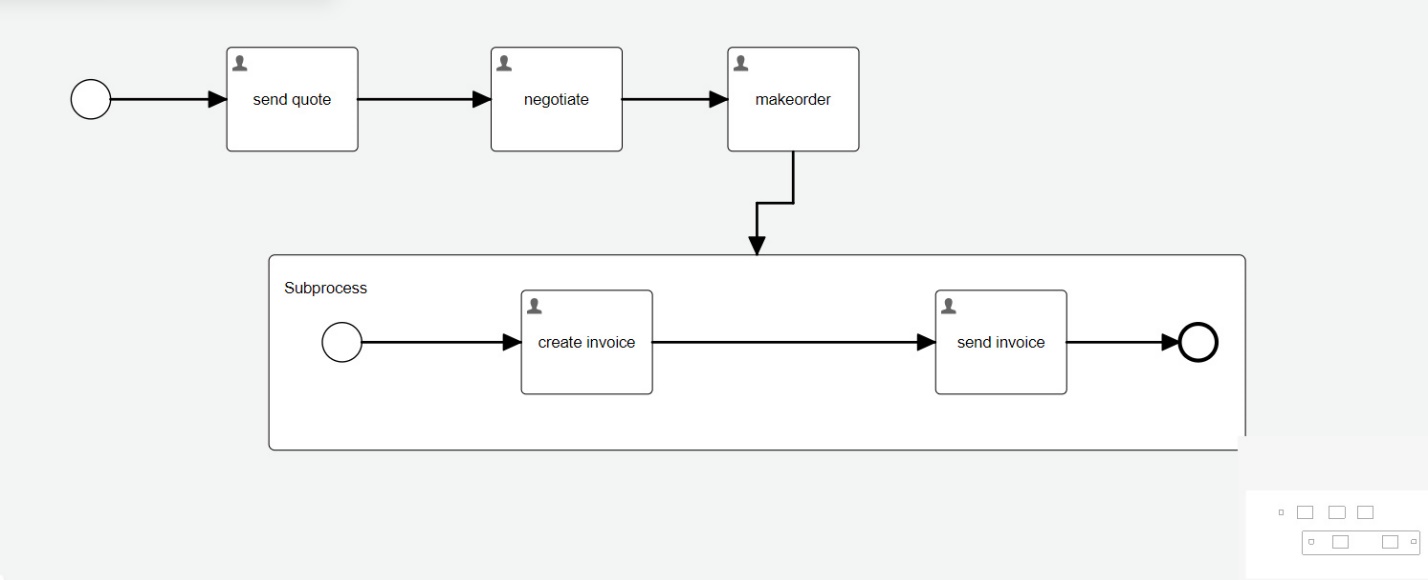
 Collapsed Sub Process



Subprocess – forms a process that is a part of a bigger process and can contain other tasks, gateways, and events. A sub-process is completely defined inside a parent process.



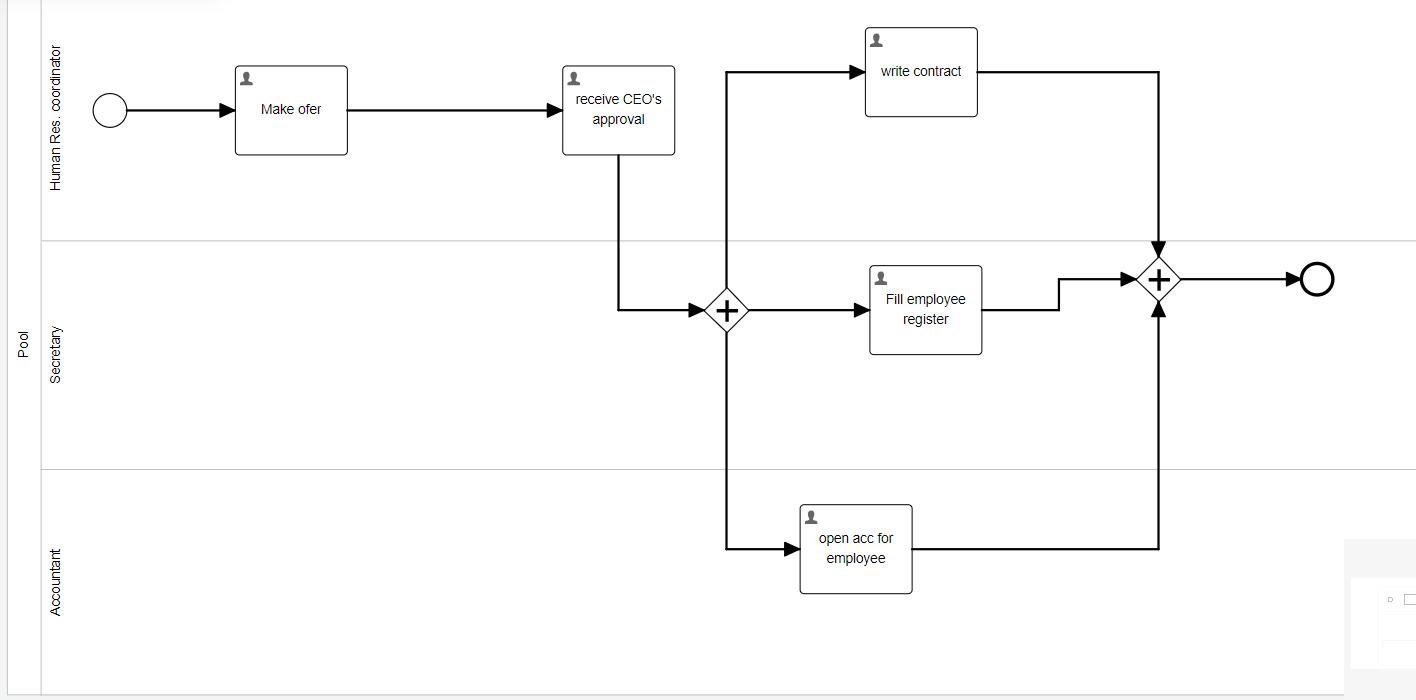
Sub proces este tot parte din acelasi proces, doar ca poate fi separat intr-un fisier sau loc aparte, doar ca sa fie procesul parinte mai clar, mai mic sa para.



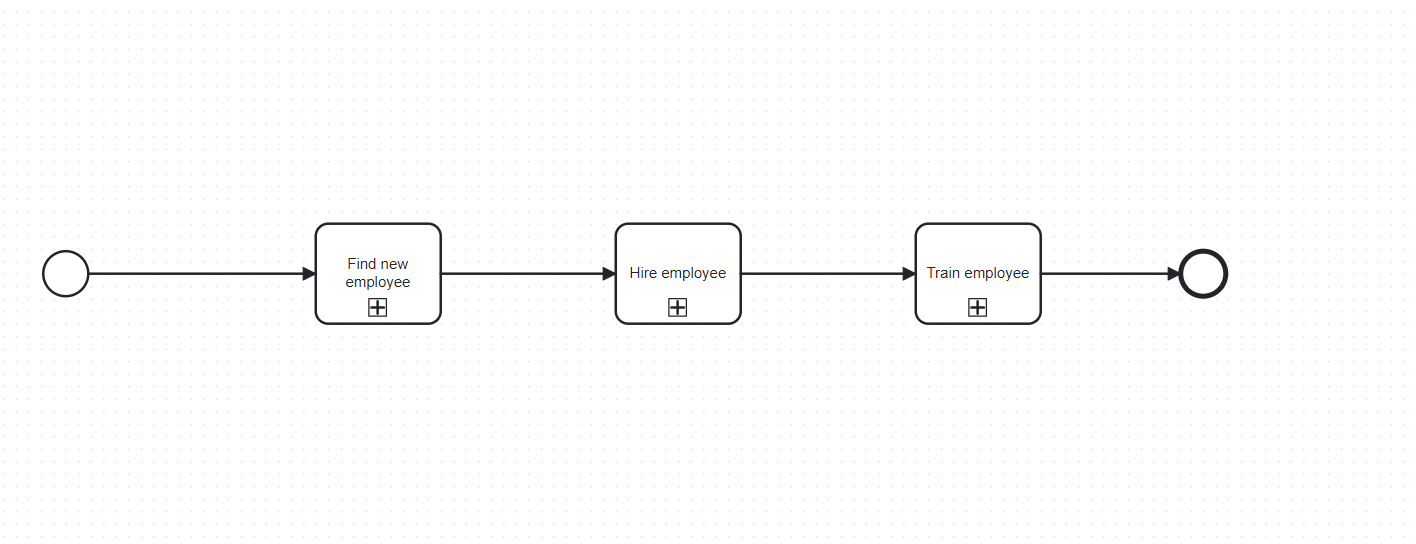
asa cam ar fi, dar subprocesul e salvat deobicei in fisiere separate

**Synch and Asynch subprocesses**

* **Synch** – procesul principal nu poate merge mai departe pana subprocesul nu termina. Deci, odata ce procesul princupal ajunge la subproces, asteapta ca el sa termine si nu merge mai departe
* **Asynch** – procesul executa subprocesele toate odata, paralel, fara sa astepte pe fiecare

****

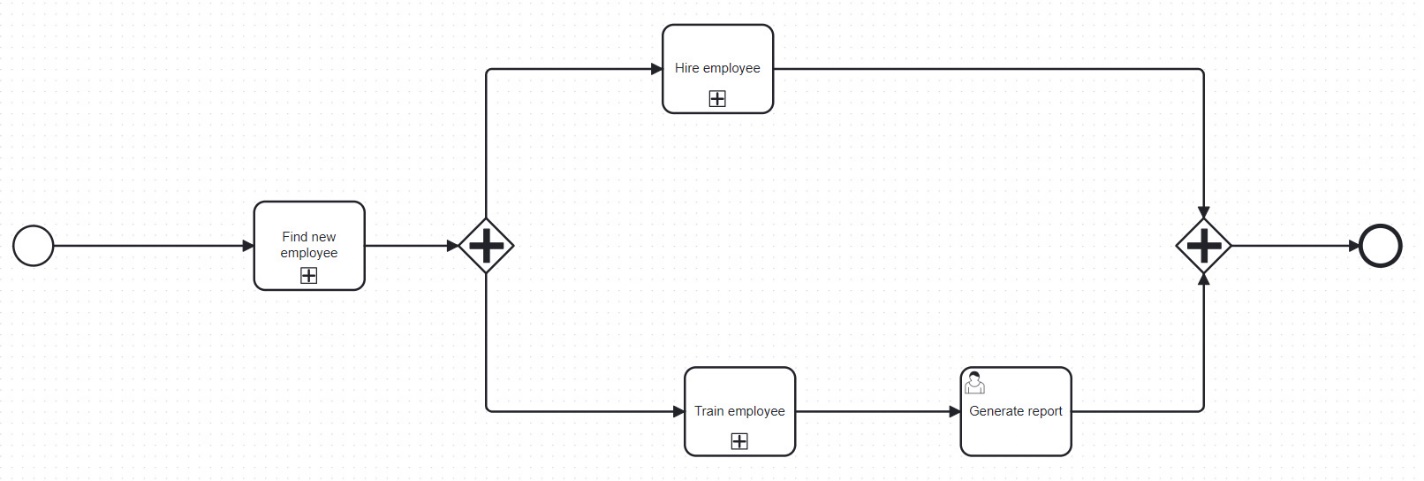
Si cam asa ar fi procesul principal, presupunand ca mai avem 2 subprocese undeva:



Deci, daca e synch, default asa e, atunci se executa fiecare subproces rand pe rand. Daca un proces e sync, odata ce se ajunge la el, procesul principal se opreste si asteapta ca executia lui sa fie finisata, apoi trece la urmatorul

Insa, in modul asynch, odata ce se ajunge la un subproces, nu se asteapta ca el se termina, se trece la urmatorul element

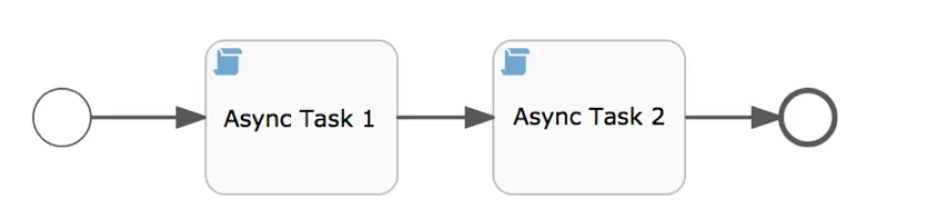
* Un mod de a obtine async comportament e cu parallel gateway:



Vedem ca generarea reportului nu va astepta ca sa se termina si Hire Employee subporcess, ci odata ce se va termina doar train employee, indiferent ca Hire Employee a terminat sau nu, se va executa Generate Report

**Asynch vs Synch flag**

* Unele BPMN ofera un buton de asynch si synch pentru fiecare task sau element
* Aici deja e diferit totul
* Synch nu inseamna ca un task va fi executat indiferent daca cel precedent a terminat sau nu
* De ex:



Chiar daca ambele taskuri au fost setate ca asynch, asta nu inseamna ca Procesul va trece in task1 si apoi deodata in task2 fara a astepta ca task1 sa termine

Asynch bifat nu modifica in niciun fel ordinea, ea tot ramane una dupa alta

* Paralelismul e obtinut cu Parallel Gateways

[Demystifying the Asynchronous Flag (flowable.com)](https://www.flowable.com/blog/engineering/demystifying-the-asynchronous-flag)

1:06:32

**Sub Event Process**

 Non-interrupting - Message - Event Sub-Process - Collapsed



 Interrupting - Message - Event Sub-Process - Collapsed



 Non-interrupting - Timer - Event Sub-Process - Collapsed



 Interrupting - Timer - Event Sub-Process - Collapsed



 Non-interrupting - Conditional - Event Sub-Process - Collapsed



 Interrupting - Conditional - Event Sub-Process - Collapsed



 Non-interrupting- Signal - Event Sub-Process - Collapsed



 Interrupting - Signal - Event Sub-Process - Collapsed



 Non-interrupting - Multiple - Event Sub-Process - Collapsed



 Interrupting - Multiple - Event Sub-Process - Collapsed



 Non-interrupting - Parallel Multiple - Event Sub-Process - Collapsed



 Interrupting - Parallel Multiple - Event Sub-Process - Collapsed



 Non-interrupting - Escalation - Event Sub-Process - Collapsed



 Interrupting - Escalation - Event Sub-Process - Collapsed



 Interrupting - Error - Event Sub-Process - Collapsed



 Interrupting - Compensation - Event Sub-Process - Collapsed



**Boundary events**

• Boundary event ofera un mod de a modela ce trebuie sa se intample cand un event are loc in timp ce o activitate este activa.

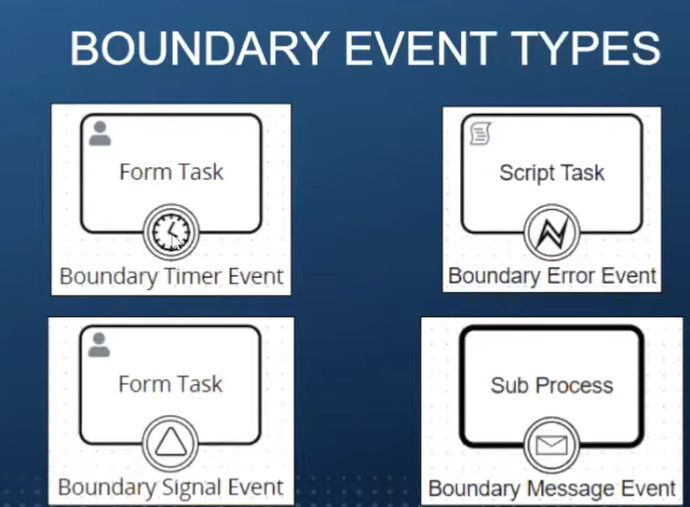
• Este un event conectat la un activity care poate fi triggered de un message, timer, signal etc.

• Este ca un handler, ce poate gen prinde erori de la alte procese pornite de un subprocess task sau chiar de la acelasi task

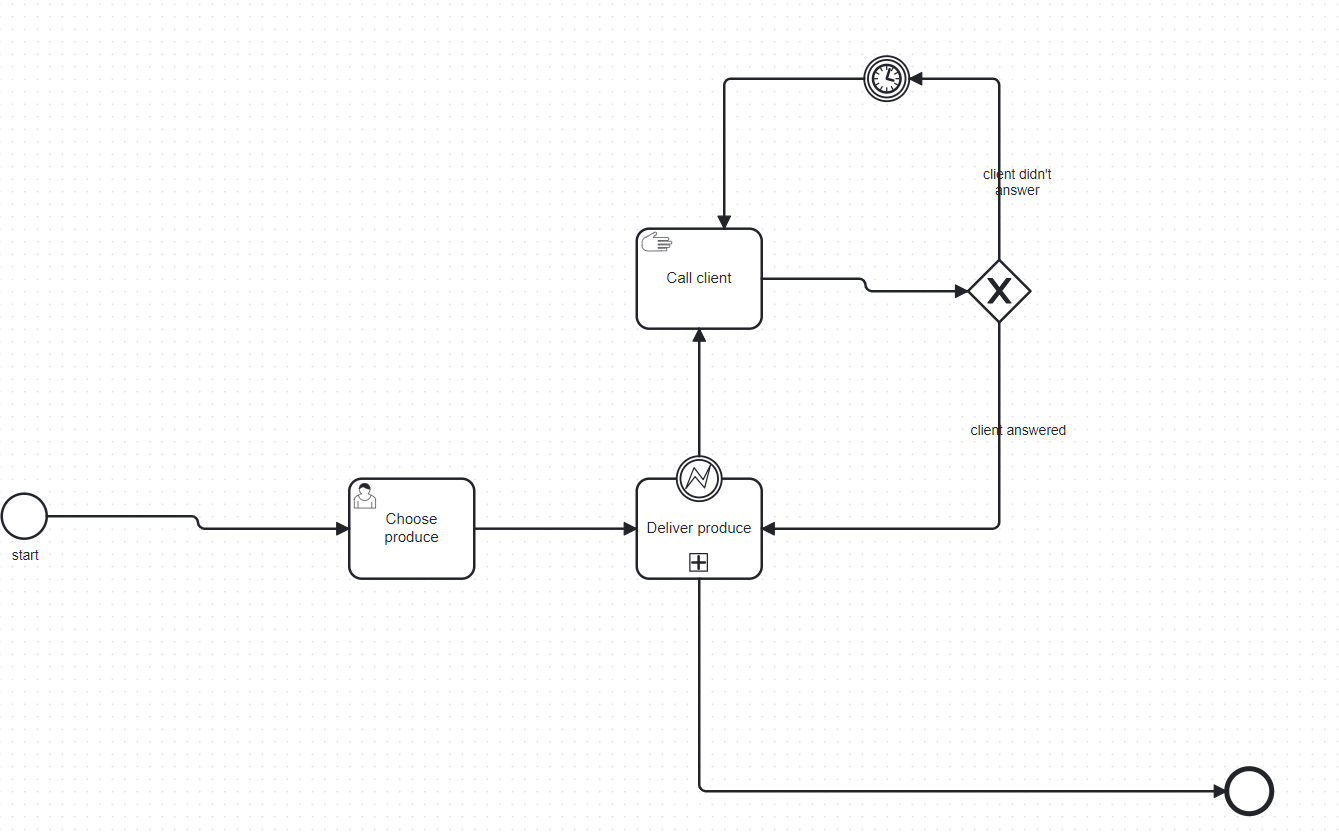
• Boundry events prind erori ce au loc in task, caci la el ajunge de ex o eroare de la alt proces

• Este conectat cu un subprocess task, caci el cheama mai multe procese ce pot returna gen messag, erori etc. dar nu mereu

• Sunt tot events, deci se noteaza la fel, doar ca stai la mirginea unui task ca notatie



* Daca e vorba de un subproces, ca ultimul din imagine, e evident ca boundary event are legatura cu un proces copil
* Daca e vorba de simple tasks, ca primele 3, boundary events au loc anume in timpul acestui task, nu depind de vren proces copil
* De ex, in prima imagine, un user completeaza o forma. Punem un boundary timer event, si daca de ex user nu completeaza forma in 20min, se v merge cu flow la timer event. Deci, e un event care poate fi declansat in timpul la task
* A 2 inseamna ca acel error boundary event se porneste daca apare vreo eroare in script



**Start vs intermediate vs end vs boundary events**

* Start event – incep la rularea procesului
* End event – finiseaza procesul, adica se executa la final
* Intermediate event – apar drept in timpul executiei procesului
* Boundary event – apar in timpul executiei unui task

**Task vs event**

A task is an action or activity that is performed by a role, and an event is something that happens or triggers a change in the process.

**Best practices**

Daca folosim un gateway pentru a intra in mai multe elemente, folosim tot unul pentru a iesi din ele, si ele sa fie identice

