Cairo University
Faculty of computers and information
System Analysis and Design 2

Lab 10: Process Instantiation

Guidelines

- The key point here is what are the start events that are required to execute the business process? They could be simply singular event, or multiple events combined with ∧, combined with XOR, combined with ∨, or even combinations of these.
  - In the latter case, need to check where is the last connection point that is just before execution of process instance, no know which is necessary and which one could be not-present.
  - Both XOR and ∨ are dealt similarly, since any of the incoming branches can activate the gateway to go on for next stage, while ∧ gateway must wait for all branches to be fulfilled to accomplish its logic.
- It is given to you a sequence of arrival events $\mathbf{e}_j$. It is required to check for process instance factory with respect to them.
  - The number between brackets is considered collaboration ID of process instance; i.e. something like ID of different instances
  - Example: this ID can represent different clients of the same company (i.e. following same business process but with different information).
- Process Instance Factory is creating Process instance with the collaboration ID mentioned.
  - If this process instance is already created once, it is ignored in later mentions.
  - After creating the process instance, they begin to check for other events required to activate the business process. All required events are being subscribed in this step.
  - While going on with the time $\mathbf{t}_i$, any of the required events are mentioned –regarding the same process instance- they are being unsubscribed (i.e. not waiting for them any more) and check for next step. This depends on the un-subscription approach being until consumption or until activation.
  - Once all required events are now collected, the process can be activated for execution.
  - If further events appeared for the same process instance, they are simply not used. "Ignore" state here is useless. They are just skipped without mentioning.
- We have 3 combination points to consider in solving strategy for a process instantiation problem:

  (1) When to subscribe: (All/Miss)

  a. **ALL:** Instance subscribes for all entry points that have not yet occurred (for the corresponding events). It will subscribe in all possible events, even if the possibility to happen is zero.

  b. **Miss**: Only those event subscriptions are activated that might be required later to complete the process instance properly. It will subscribe in only the needed/required events

  (2) When to unsubscribe: (Upon consumption/ Upon activation/Upon termination)

  Consuming an event, means that the process subscribed for this event previously. This doesn't necessarily means that the event is required (as in case of ALL event subscription).

a. **Upon consumption**;
   - All subscriptions for existing entry points are kept until consumed. The process can not activate without the occurrence of these events.
   - Unsubscribe from an event once it is <u>consumed</u>, but the process is not necessarily activated at the same time.

b. **Upon activation (proper termination)**; This is an additional setting for <u>upon consumption</u> policy. [Always found combined with upon consumption]
   - Subscriptions for entry points to be kept until the process instance activates.
   - Unsubscribe from the consumed event(s) only when the process is <u>activated</u>.

c. **Upon termination**;
   - The subscriptions for entry points to be kept until the process instance terminates.
   - Unsubscribe from the consumed event(s) only when the process is <u>terminated</u>.
   - This may not necessarily appear in the selected log.

(3) Who is affected: (Event type/Event instance)

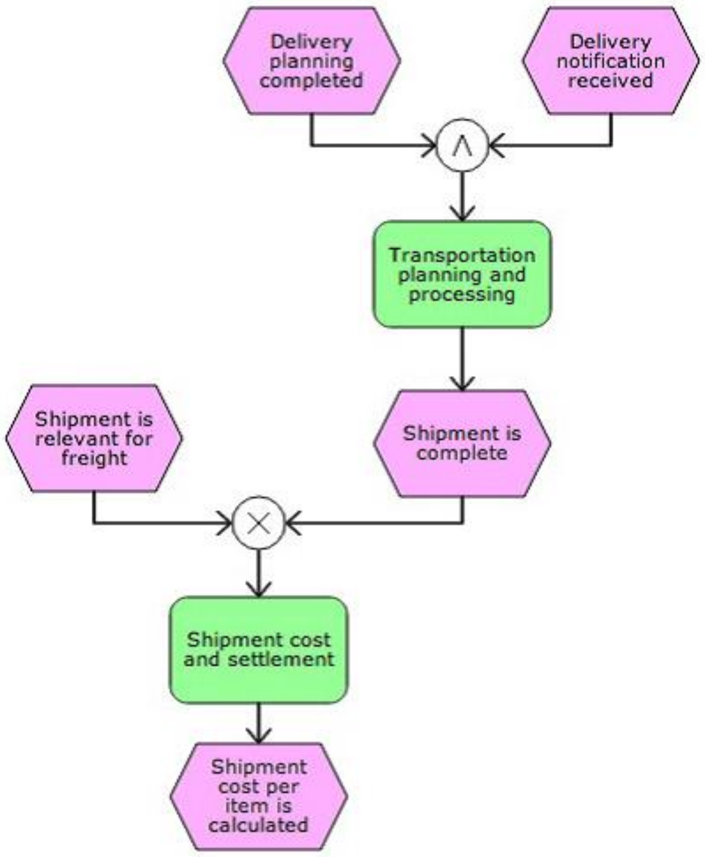
a. **Event type:**
   Whenever an event took place, notify all process instances that may consume it.
   The factory may subscribe in **e1** only without interest in process instance that assigned it, and then it will notify all process instances that may have interest in.

b. **Event instance:**
   Whenever an event took place and mentioned the process consuming it with the event type, it will notify the mentioned process only to consume it.
   The factory may subscribe in e1 with its assigned process instance (**e1(p1)**), and then it will notify only process instance (p1).

Example 1:

Consider that the process model in Figure 1 is deployed to an execution engine where "Delivery notification received", "Delivery planning completed" and "Shipment is relevant for freight" are three types of events.
We symbolize them with E1; E2; E3 respectively.
Consider the following sequence arrival of events.

t1 : e3(1000)
t2 : e1(1001)
t3 : e1(1000)
t4 : e2(1001)
t5 : e3(1001)


Delivery planning completed
Delivery notification received
Λ
Transportation planning and processing
Shipment is relevant for freight
Shipment is complete
Shipment cost and settlement
Shipment cost per item is calculated

Show the actions of the process instance factory as well as the process instances in response for the events occurrence above.
(The solution's strategy follows subscription for missing event and unsubscribes at consumption time)
Solve using the following solution strategy:
- Missing/event instance /upon consumption
- All/event instance /upon consumption

Solution 1: the strategy follows: subscription for Missing event and unsubscribes at consumption time.

- At time t1,
  - The process instance factory creates a new process instance p_{1000} with an initial activation of e3(1000).
  - At the same point in time p_{1000} turns into state running as it has all necessary input collected, because E3 is exclusive to E1 ^ E2.
- At time t2,
  - The process instance factory creates a new process instance p_{1001} with an initial activation e1(1001).
  - At the same time p_{1001} subscribes for event E2.
- At time t3 the event instance e1(1000) is ignored by the process factory. (p_{1000} already created)
- At time t4 the event instance e2(1001) is ignored by the process factory. (p_{1001} already created)
  - However, the process instance p_{1001} consumes the event (E2) and unsubscribes from E2 and turns into state running.
- At time t5 the event instance e3(1001) is ignored by the process instance factory. (p_{1001} already created).

<u>Solution 2:</u> the strategy follows: subscription for ALL event and unsubscribes at consumption time.

- At time t1,
    - The process instance factory creates a new process instance p_{1000} with an initial activation of e3(1000).
    - At the same point in time p_{1000} turns into state running as it has all necessary input collected, because E3 is exclusive to E1 ^ E2.
- At time t2,
    - The process instance factory creates a new process instance p_{1001} with an initial activation e1(1001).
    - At the same time p_{1001} subscribes for event E2 and E3.
- At time t3 the event instance e1(1000) is ignored by the process factory. (p_{1000} already created)
- At time t4 the event instance e2(1001) is ignored by the process factory. (p_{1001} already created)
    - However, the process instance p_{1001} consumes the event (E2) and unsubscribes from E2 and turns into state running.
- At time t5 the event instance e3(1001) is ignored by the process instance factory. (p_{1001} already created).
    - However, the process instance p_{1001} consumes/ignores the event (E3) and unsubscribes from E3.

Note that:

- Consuming E3 doesn't mean P1001 needs it. It just mean that P1001 is notified with E3 occurrence (since it was subscribed for at t2), and consumed or ignored.
- You can use p_1 and p_2 instead of p_{1000} and p_{1001} they are both correct in meaning.

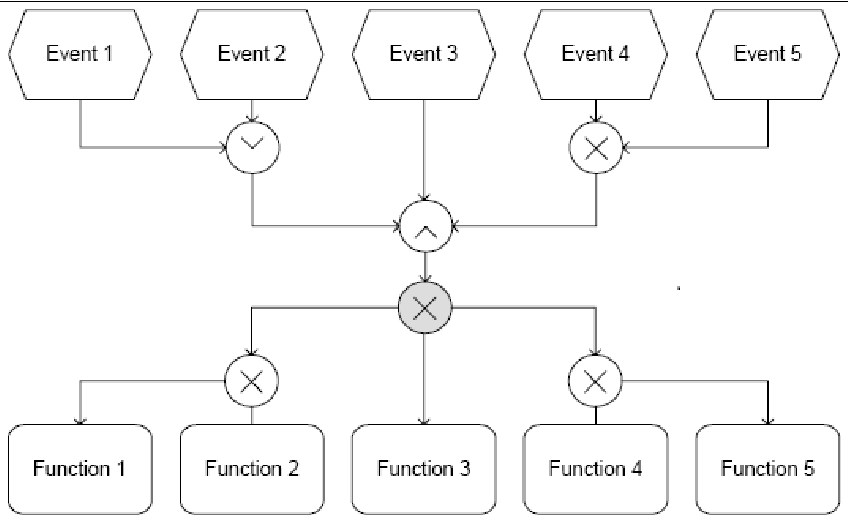
Example 2:

Consider that the process model in Figure 2 is deployed to an execution engine.

Consider the following sequence arrival of events.

t1: e2(1000)
t2: e3(1001)
t3: e4(1000)
t4: e5(1002)
t5: e5(1001)
t6: e1(1002)
t7: e1(1000)
t8: e3(1000)
t9: e3(1002)
t10: e2(1001)
t11: e4(1001)
t12: e3(1001)

Show the actions of the process instance factory as well as the process instances in response for the events occurrence above.



Solve using the following solution strategies:

1- missing/event instance /upon consumption
2- All/event instance /upon consumption & activation
3- All/event type/upon consumption & activation
4- missing/event instance /upon termination

<u>Solution 1:</u>

In this solution: we will use a combination of missing/event instance /upon consumption

- At t1: e2(1000) → Factory creates p1000 of (e2(1000)), p1000 subscribes in (e3, e4, e5).
- At t2: e3(1001) → Factory creates p1001 of (e3(1001)), p1001 subscribes in (e1, e2, e4, e5).
- At t3: e4(1000) → Factory ignores creation of p1000, p1000 consumes e4, p1000 unsubscribes from e4 (still subscribing for e3, e5).
- At t4: e5(1002) → Factory creates p1002 of (e5(1002)), p1002 subscribes in (e1, e2, e3).
- At t5: e5(1001) → Factory ignores creation of p1001, p1001 consumes e5, p1001 unsubscribes from e5 (still subscribing for e1, e2, e4).
- At t6: e1(1002) → Factory ignores creation of p1002, p1002 consumes e1, p1002 unsubscribes from e1, (still subscribing for e2, e3).
- At t7: e1(1000) → Factory ignores creation of p1000.
- At t8: e3(1000) → Factory ignores creation of p1000, p1000 consumes e3, p1000 unsubscribes from e3, business process is **activated** [process instance **p1000**], (still subscribing for e5).
- At t9: e3(1002) → Factory ignores creation of p1002, p1002 consumes e3, p1002 unsubscribes from e3, business process is **activated** [process instance **p1002**], (still subscribing for e2).

- At t10: e2(1001) → Factory ignores creation of p1001, p1001 consumes e2, p1001 unsubscribes from e2, business process is **activated** [process instance **p1001**], (still subscribing for e1, e4).
- At t11: e4(1001) → Factory ignores creation of p1001, p1001 consumes/ignores e4, p1001 unsubscribes from e4, (still subscribing for e1).
- At t12: e3(1001) → Factory ignores creation of p1001.

Solution 2:

In this solution: we will use a combination of All/event instance /upon consumption & activation

- At t1: e2(1000) → Factory creates p1000 of (e2(1000)), p1000 subscribes in (e1, e3, e4, e5).
- At t2: e3(1001) → Factory creates p1001 of (e3(1001)), p1001 subscribes in (e1, e2, e4, e5).
- At t3: e4(1000) → Factory ignores creation of p1000, p1000 consumes e4, p1000 unsubscribes from e4 (still subscribing for e1, e3, e5).
- At t4: e5(1002) → Factory creates p1002 of (e5(1002)), p1002 subscribes in (e1, e2, e3, e4).
- At t5: e5(1001) → Factory ignores creation of p1001, p1001 consumes e5, p1001 unsubscribes from e5 (still subscribing for e1, e2, e4).
- At t6: e1(1002) → Factory ignores creation of p1002, p1002 consumes e1, p1002 unsubscribes from e1, (still subscribing for e2, e3, e4).
- At t7: e1(1000) → Factory ignores creation of p1000, p1000 consumes/ignores e1, p1000 unsubscribes from e1, (still subscribing for e3, e5).
- At t8: e3(1000) → Factory ignores creation of p1000, p1000 consumes e3, p1000 unsubscribes from (e3, e5), business process is **activated** [process instance **p1000**].
- At t9: e3(1002) → Factory ignores creation of p1002, p1002 consumes e3, p1002 unsubscribes from (e2, e3, e4), business process is **activated** [process instance **p1002**].
- At t10: e2(1001) → Factory ignores creation of p1001, p1001 consumes e2, p1001 unsubscribes from (e1, e2, e4), business process is **activated** [process instance **p1001**].
- At t11: e4(1001) → Factory ignores creation of p1001.
- At t12: e3(1001) → Factory ignores creation of p1001.

Solution 3:

In this solution: we will use a combination of All/event type/ upon consumption & activation

- At t1: e2(1000) → Factory creates p1000 of (e2(1000)), p1000 subscribes in (e1, e3, e4, e5).
- At t2: e3(1001) → Factory creates p1001 of (e3(1001)), p1001 subscribes in (e1, e2, e4, e5). p1000 ignores e3(1001).
- At t3: e4(1000) → Factory ignores creation of p1000, p1000 consumes e4, p1000 unsubscribes from e4 (still subscribing for e1, e3, e5). p1001 ignores e4(1000).
- At t4: e5(1002) → Factory creates p1002 of (e5(1002)), p1002 subscribes in (e1, e2, e3, e4). p1000 ignores e5(1002), p1001 ignores e5(1002).
- At t5: e5(1001) → Factory ignores creation of p1001, p1001 consumes e5, p1001 unsubscribes from e5 (still subscribing for e1, e2, e4). p1000 ignores e5(1001).
- At t6: e1(1002) → Factory ignores creation of p1002, p1002 consumes e1, p1002 unsubscribes from e1, (still subscribing for e2, e3, e4). p1000 ignores e1(1002), p1001 ignores e1(1002).

- At t7: e1(1000) → Factory ignores creation of p1000, p1000 consumes/ignores e1, p1000 unsubscribes from e1, (still subscribing for e3, e5). p1001 ignores e1(1000).
- At t8: e3(1000) → Factory ignores creation of p1000, p1000 consumes e3, p1000 unsubscribes from (e3, e5), business process is **activated** [process instance **p1000**]. p1002 ignores e3(1000).
- At t9: e3(1002) → Factory ignores creation of p1002, p1002 consumes e3, p1002 unsubscribes from (e2, e3, e4), business process is **activated** [process instance **p1002**].
- At t10: e2(1001) → Factory ignores creation of p1001, p1001 consumes e2, p1001 unsubscribes from (e1, e2, e4), business process is **activated** [process instance **p1001**].
- At t11: e4(1001) → Factory ignores creation of p1001.
- At t12: e3(1001) → Factory ignores creation of p1001.

Solution 4:

In this solution: we will use a combination of missing/event instance /upon termination

- At t1: e2(1000) → Factory creates p1000 of (e2(1000)), p1000 subscribes in (e3, e4, e5).
- At t2: e3(1001) → Factory creates p1001 of (e3(1001)), p1001 subscribes in (e1, e2, e4, e5).
- At t3: e4(1000) → Factory ignores creation of p1000, p1000 consumes e4 (still subscribing for e3, e4, e5).
- At t4: e5(1002) → Factory creates p1002 of (e5(1002)), p1002 subscribes in (e1, e2, e3).
- At t5: e5(1001) → Factory ignores creation of p1001, p1001 consumes e5 (still subscribing for e1, e2, e4, e5).
- At t6: e1(1002) → Factory ignores creation of p1002, p1002 consumes e1 (still subscribing for e1, e2, e3).
- At t7: e1(1000) → Factory ignores creation of p1000 (still subscribing for e3, e4, e5).
- At t8: e3(1000) → Factory ignores creation of p1000, p1000 consumes e3, business process is **activated** [process instance **p1000**] (still subscribing for e3, e4, e5).
- At t9: e3(1002) → Factory ignores creation of p1002, p1002 consumes e3, business process is **activated** [process instance **p1002**], (still subscribing for e1, e2, e3).
- At t10: e2(1001) → Factory ignores creation of p1001, p1001 consumes e2, business process is **activated** [process instance **p1001**], (still subscribing for e1, e2, e4, e5).
- At t11: e4(1001) → Factory ignores creation of p1001, p1001 consumes/ignores e4 (still subscribing for e1, e2, e4, e5).
- At t12: e3(1001) → Factory ignores creation of p1001 (still subscribing for e1, e2, e4, e5).