Process modeling II

PV207 – Business Process Management
Spring 2019

Jiří Kolář, Lubomír Dočkal, Lubomír Hruban

Last lecture recap

- Why process modeling?

Last lecture recap

- Why process modeling?
- BPMN L1, L2, L3

Last lecture recap

- Why process modeling?
- BPMN L1, L2, L3
- Quality aspects of process model

Last lecture recap

- Why process modeling?
- BPMN L1, L2, L3
- Quality aspects of process model
- Process interactions
  - Private process
  - Abstract process (Black box/Collapsed Pool)
  - Collaboration (Global) process

Last lecture recap

- Pool and Lane

- Task (User, Service, Abstract/None)

- Subprocess (Collapsed, Expanded)

- Start Event (None, Message, Timer)

- End Event (None, Message, Terminate)

- Gateway (Parallel, Exclusive)

- Sequence Flow

- Message Flow

- Data Object (Data store, Message)

- Text Annotation

- Link Event Pair

Lecture overview

- Information sources
- From L1 to L2
- L2: timing precision

- BPMN 2.0 Level 2:
  - Subprocess
  - Activity call
  - **Events**
    - Messages
    - Signals
    - Errors
    - Escalations
  - Gateways
  - BPMN 2.0 summary

Information sources

- BOOK: BPMN method and style / Bruce Silver
  - ISBN:9780982368107, Library FI, Amazon 33$
- BPMN 2.0 poster
  - http://www.bpmb.de/images/BPMN2_0_Poster_EN.pdf
- Signavio modeler – academic licence
  - http://academic.signavio.com/p/login
- BPMN official OMG website
  - http://www.bpmn.org

BPMN 2.0: from L1 to L2

- Level 1
  - Flowcharting
  - Business experts <=> analysts/developers
  - The goal is to express simple activity sequences
  - Minimum of nesting and interprocess interactions
  - Simple events only
- Level 2
  - Analytical BPMN model
  - Process analysts <=> Process developers
  - Precise activity execution timing
  - Subprocess nesting and interprocess interactions
  - Events and signals, exception handling

Level 2: timing precision

- Each activity has exact start and completion
- Service task
  - Starts immediately when reached
  - Being performed immediately and completed
- User task
  - Starts immediately when reached
  - Being performed once user open it in a "worklist" = task "claim"

Activity states
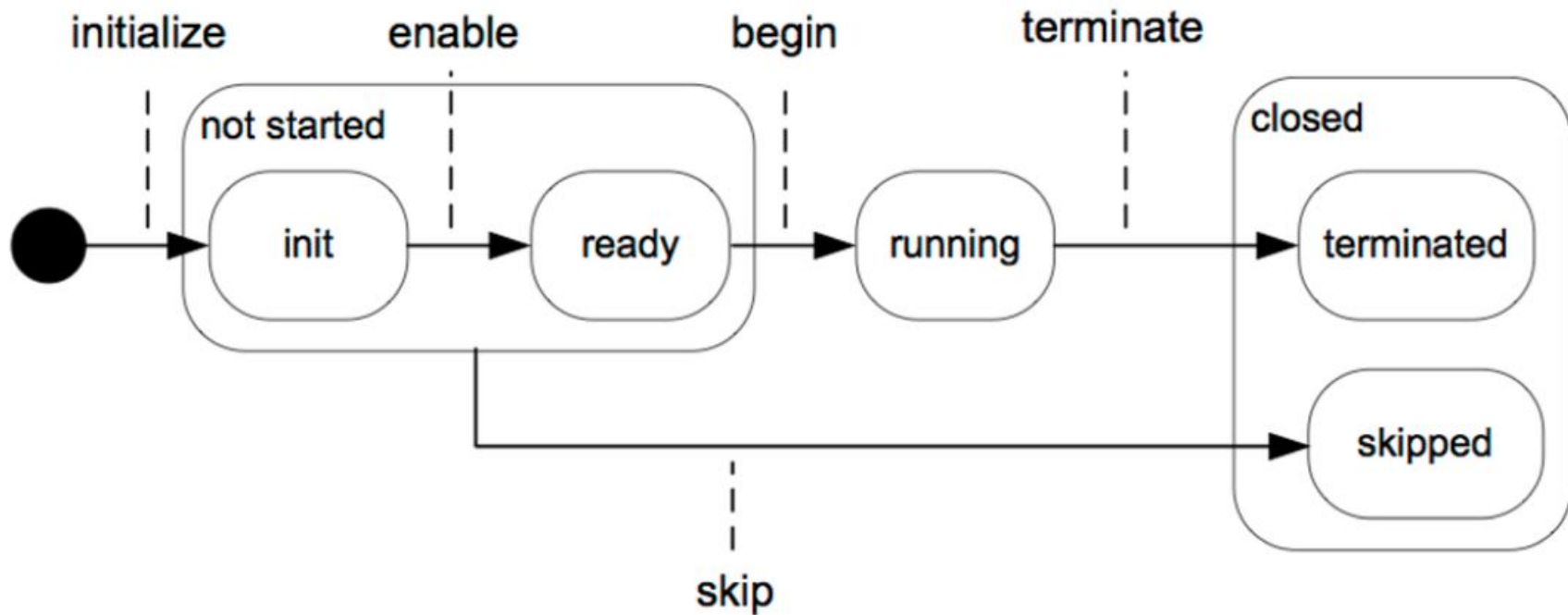
Fig. 3.9. State transition diagram for activity instances
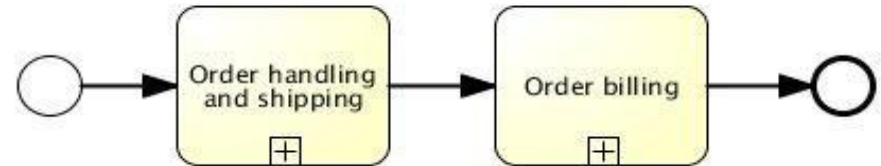
Level 2: timing precision example
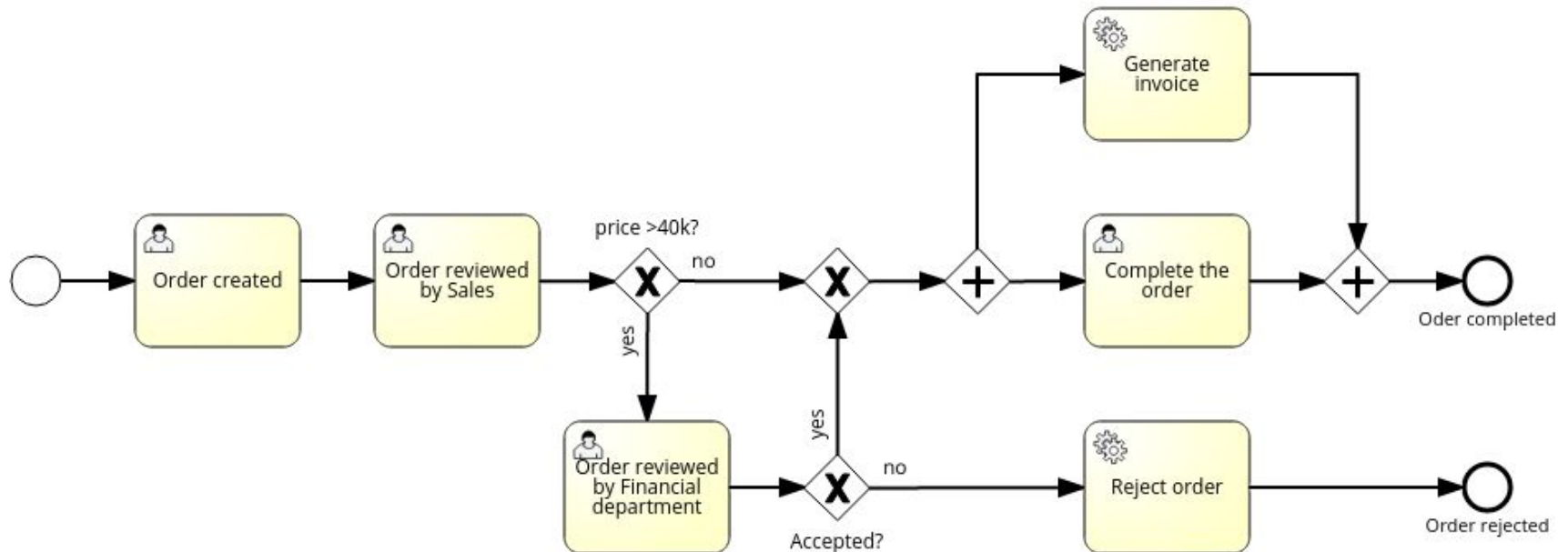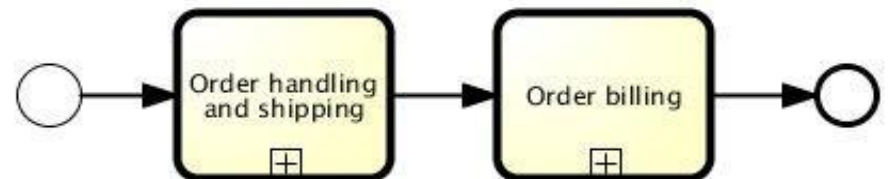
Subprocess vs Call activity

- Subprocess
  - Expandable
    (nested) part of the process
  - Defined inside process
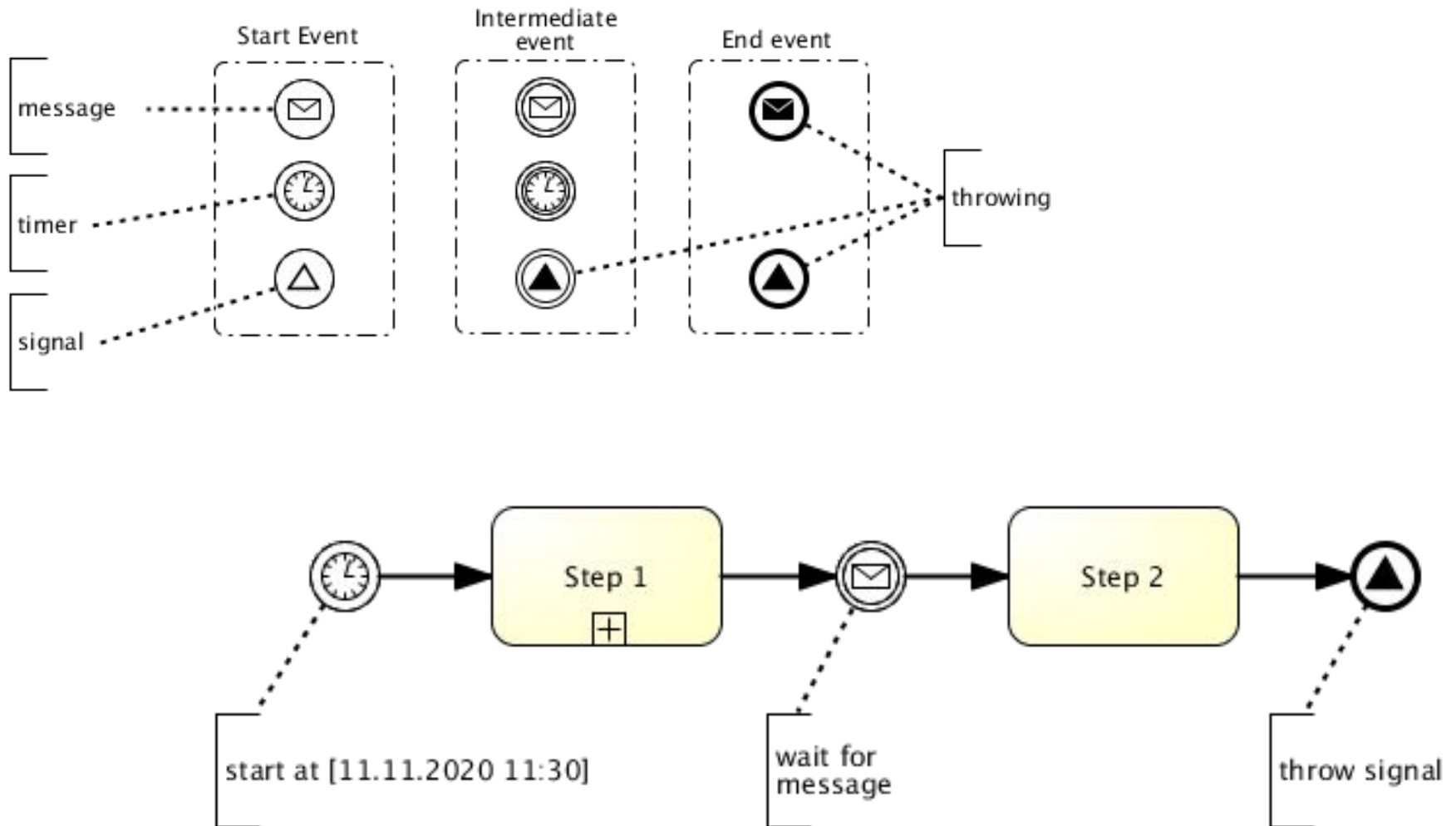 - Nested for better readability

- Activity call
  - Call of global task or process
  - Defined as a separate process, then imported
  - Reusable in other processes

Event types: Basic types

- Start events
  - Event initiate process/subprocess
  - One (or more in special cases)
  - Always catching
- Intermediate events
  - Occur during process
  - Can be throwing or catching
- End events
  - Occur at the end of process flow
  - Always throwing
  - End affect only one branch (except Terminate)

Event types - Examples

Events

Events

|	Start			Intermediate				End
	Top-Level	Event Sub-Process Interrupting	Event Sub-Process Non-Interrupting	Catching	Boundary Interrupting	Boundary Non-Interrupting	Throwing	End
None: Untyped events, indicate start point, state changes or final states.	◯						◯	◯
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	🕐	🕐	🕐	🕐	🕐	🕐		
Escalation: Escalating to an higher level of responsibility.		⌃	⌃		⌃	⌃	⌃	⌃
Conditional: Reacting to changed business conditions or integrating business rules.	▤	▤	▤	▤	▤	▤		
Link: Off-page connectors. Two corresponding link events equal a sequence flow.				⇨			⇨	
Error: Catching or throwing named errors.		⚡			⚡			⚡
Cancel: Reacting to cancelled transactions or triggering cancellation.					⊗			⊗
Compensation: Handling or triggering compensation.		⏪			⏪		⏪	⏪
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	⊕	⊕	⊕	⊕	⊕	⊕		
Terminate: Triggering the immediate termination of a process.								⬤

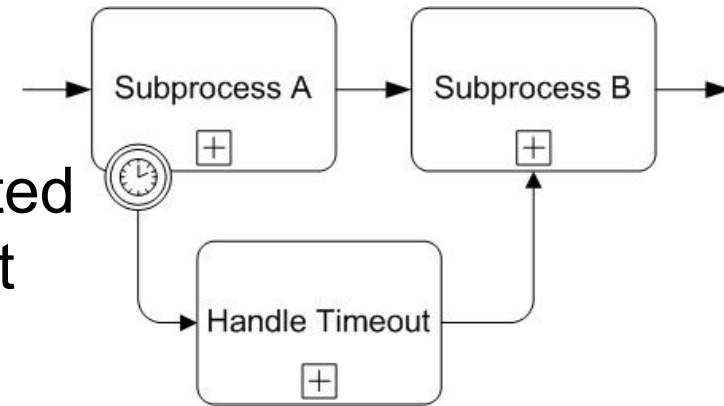# Event types: Catching vs. Throwing

- ## Throwing
  - Emits the event
  - Flow continues immediately
- ## Catching
  - Catch the event
  - Flow waits for the event
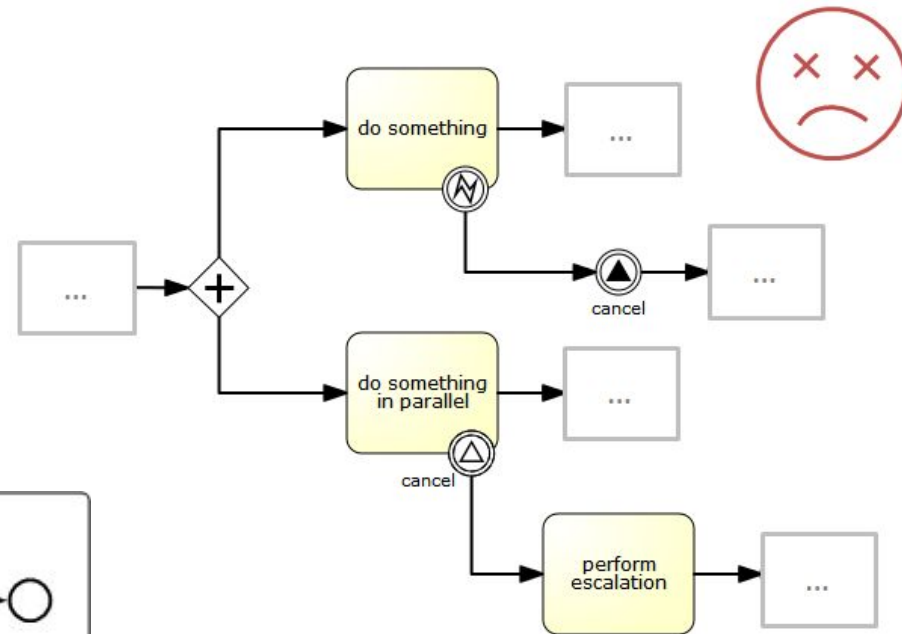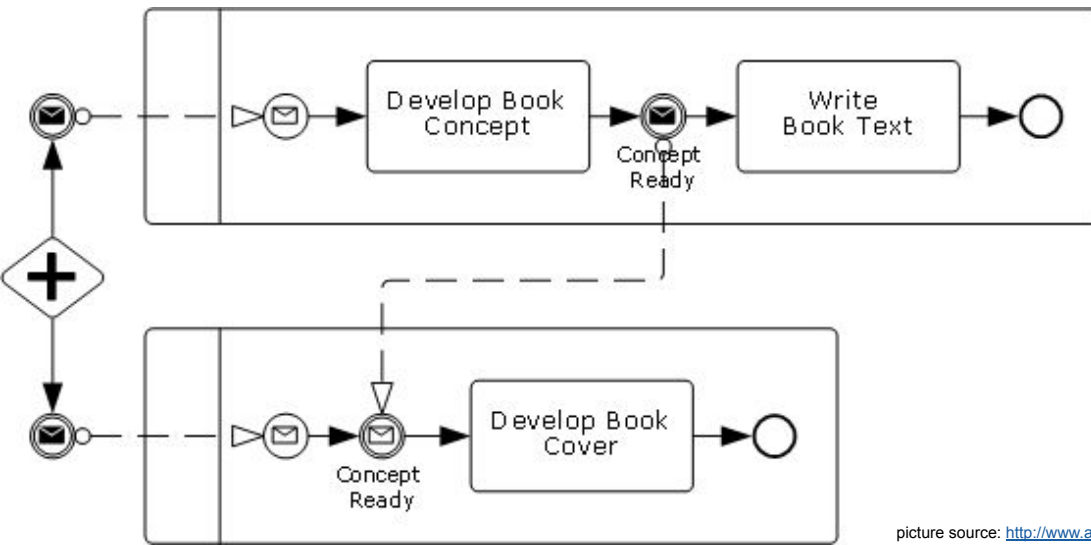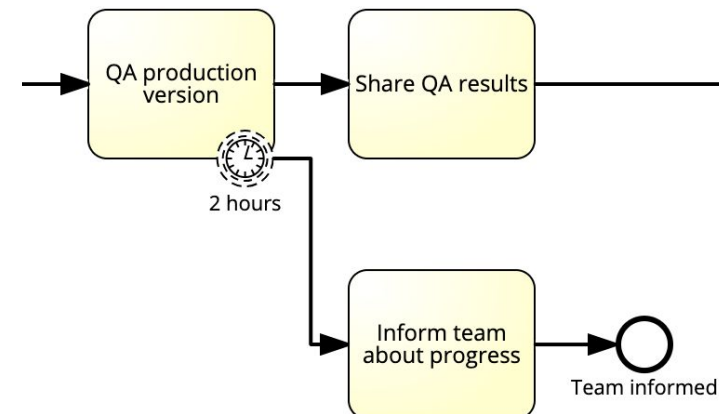
Event types:
Interrupting vs non-interrupting

- ## Interrupting
  - Standard process flow is interrupted
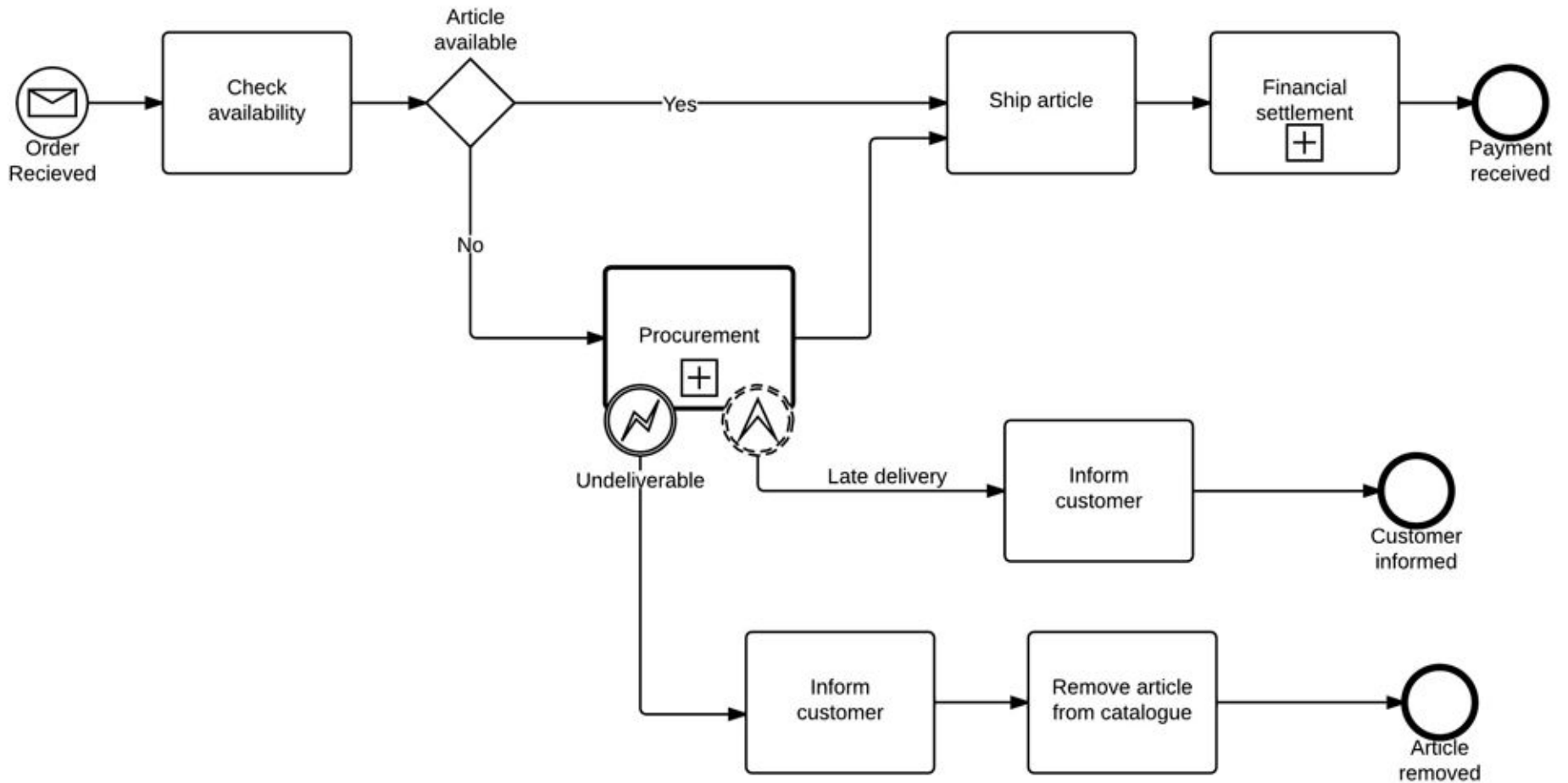 - Flow is directed through the event
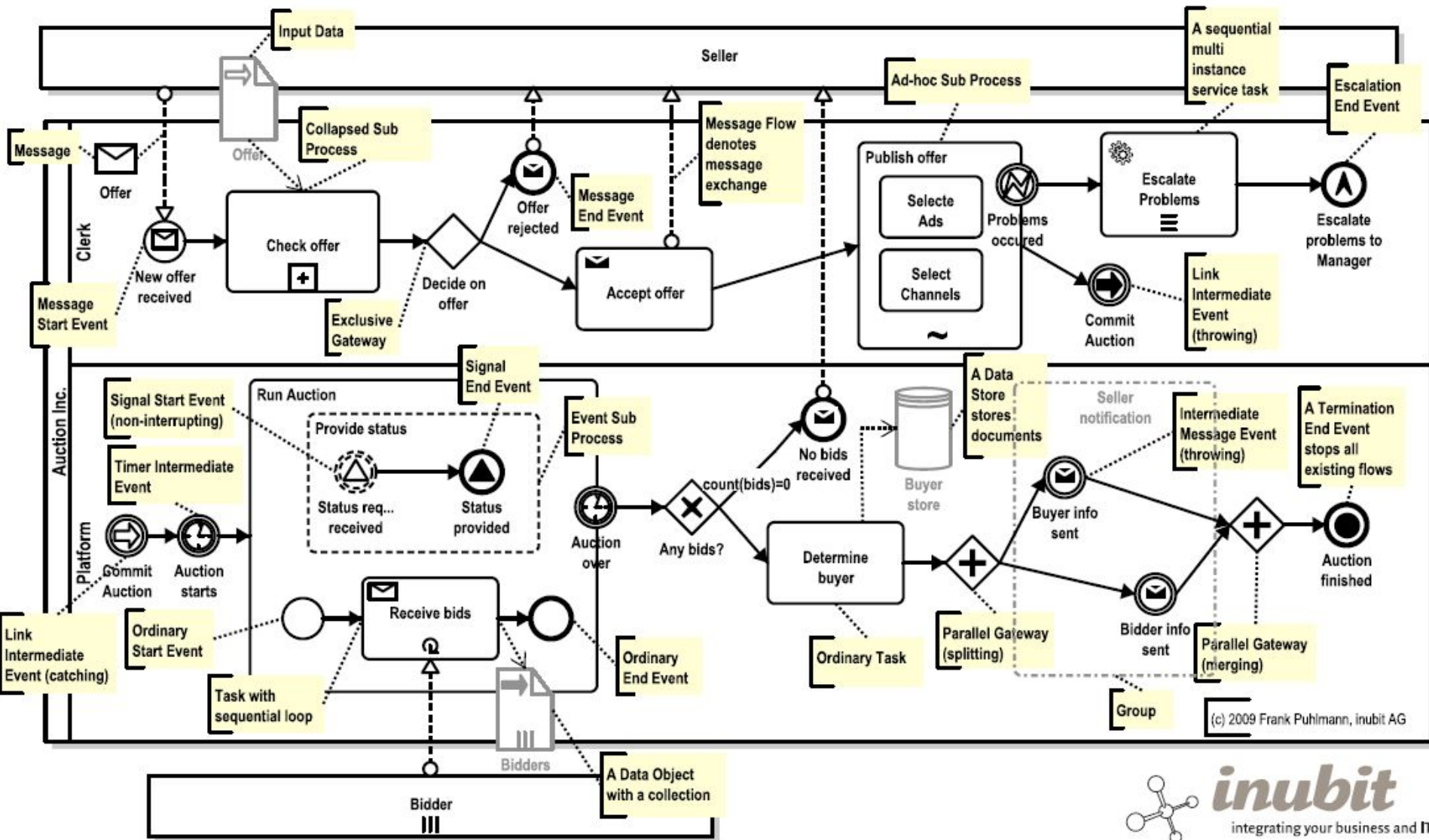

- ## Non-interrupting
  - Standard flow continues normally
  - Parallel flow is directed through the event

Event types:
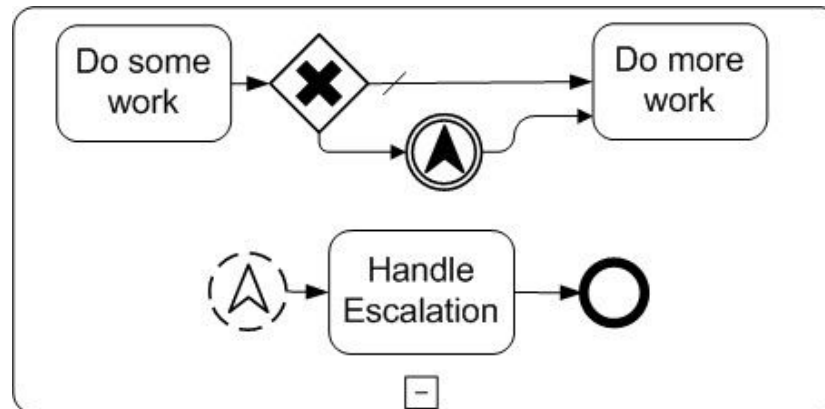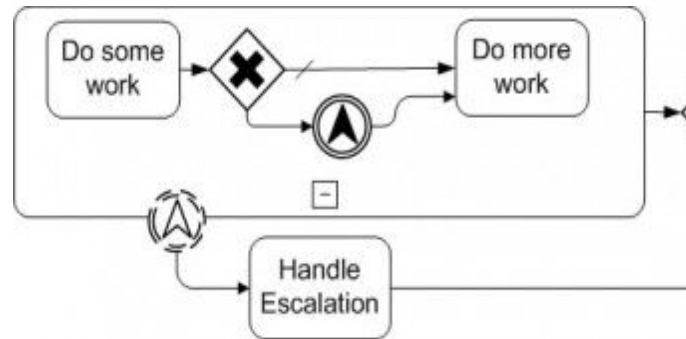# Interrupting vs non-interrupting

Break 10mins

Events

Events	Start			Intermediate				End
	Top-Level	Event Sub-Process Interrupting	Event Sub-Process Non-Interrupting	Catching	Boundary Interrupting	Boundary Non-Interrupting	Throwing	
None: Untyped events, indicate start point, state changes or final states.	○						○	○
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	🕐	🕐	🕐	🕐	🕐	🕐		
Escalation: Escalating to an higher level of responsibility.		⌃	⌃		⌃	⌃	⌃	⌃
Conditional: Reacting to changed business conditions or integrating business rules.	▤	▤	▤	▤	▤	▤		
Link: Off-page connectors. Two corresponding link events equal a sequence flow.				➡			➡	
Error: Catching or throwing named errors.		⚡			⚡			⚡
Cancel: Reacting to cancelled transactions or triggering cancellation.					⊗			⊗
Compensation: Handling or triggering compensation.		⏪			⏪		⏪	⏪
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	✚	✚	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process.								●

Event types:
Boundary vs. in-flow (event subprocess)

Events

| Events	Start			Intermediate				End
	Top-Level	Event Sub-Process Interrupting	Event Sub-Process Non-Interrupting	Catching	Boundary Interrupting	Boundary Non-Interrupting	Throwing	End
None: Untyped events, indicate start point, state changes or final states.	○						○	○
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	⏰	⏰	⏰	⏰	⏰	⏰		
Escalation: Escalating to an higher level of responsibility.		⏶	⏶		⏶	⏶	⏶	⏶
Conditional: Reacting to changed business conditions or integrating business rules.	▤	▤	▤	▤	▤	▤		
Link: Off-page connectors. Two corresponding link events equal a sequence flow.				⇨			⇨	
Error: Catching or throwing named errors.		⚡			⚡			⚡
Cancel: Reacting to cancelled transactions or triggering cancellation.					⊗			⊗
Compensation: Handling or triggering compensation.		⏪			⏪		⏪	⏪
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	⊕	⊕	⊕	⊕	⊕	⊕		
Terminate: Triggering the immediate termination of a process.								●

Event semantics: Messages

- Message represents a message send by external entity ~ Pool
  - Messaging is for interprocess communication
  - Inside the process use sequence flow instead
- Message does not have to be JMS, SOAP etc. but it can be fax, mail, SMS etc.
- A Message can be received and start process
- A message can occur as intermediate event
- A message can be sent at the end of process

Event semantics: Message - examples
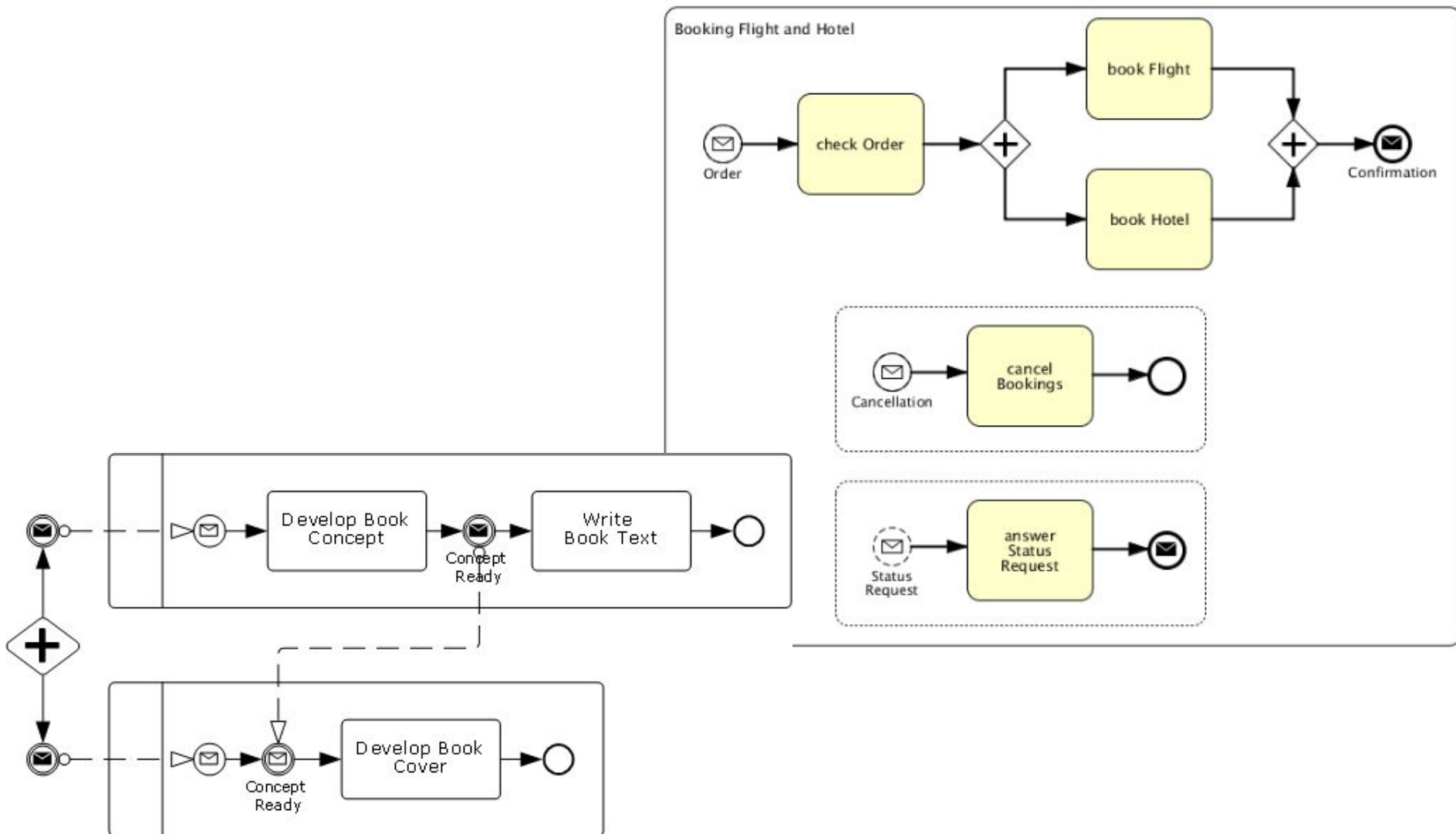
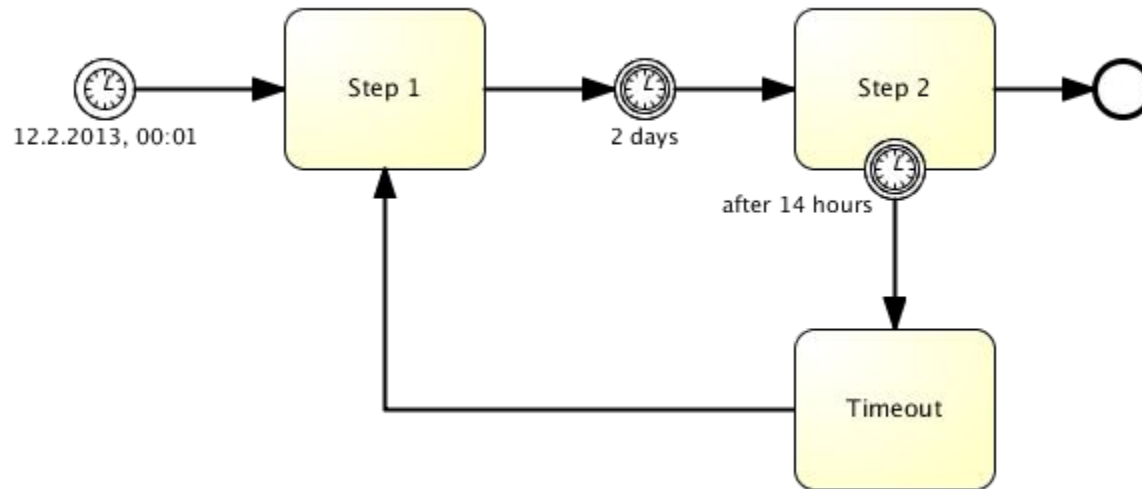# Event semantics: Signals

- Signal is similar to message, except
  - **Is not** addressed to any particular consumer
  - Entity producing signal does not "care" who is listening
  - Many instances of the same process can consume it
  - Good for loosely coupled communication
  - Signals are used often inside one process, messages not
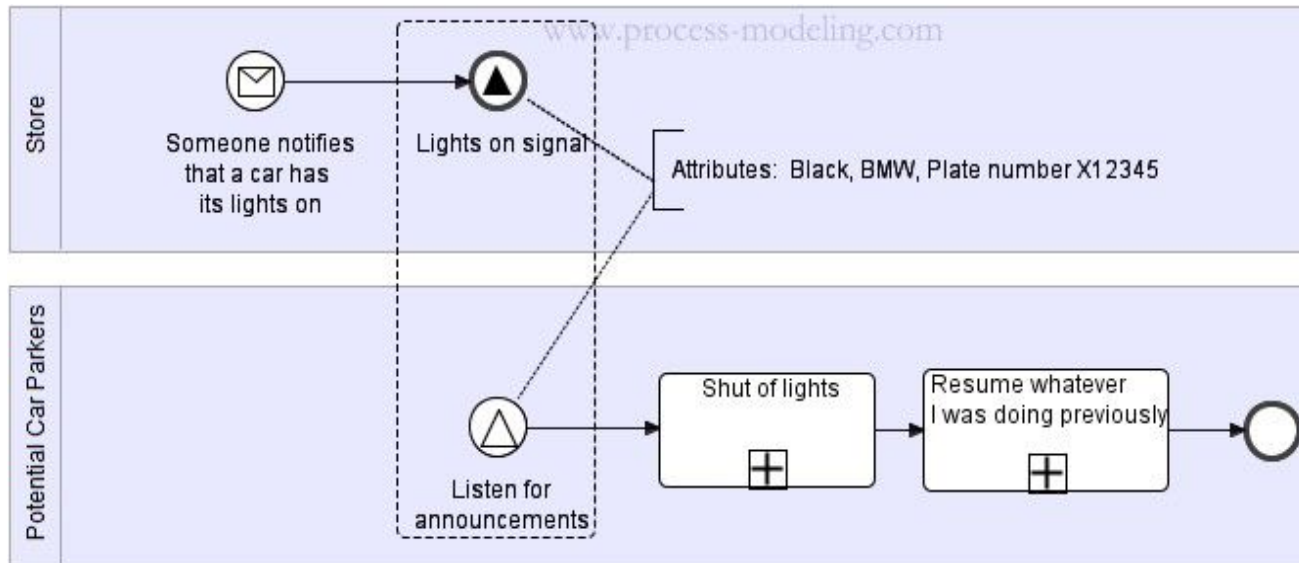
Event semantics: Timer

- Cyclic events
- Points in time
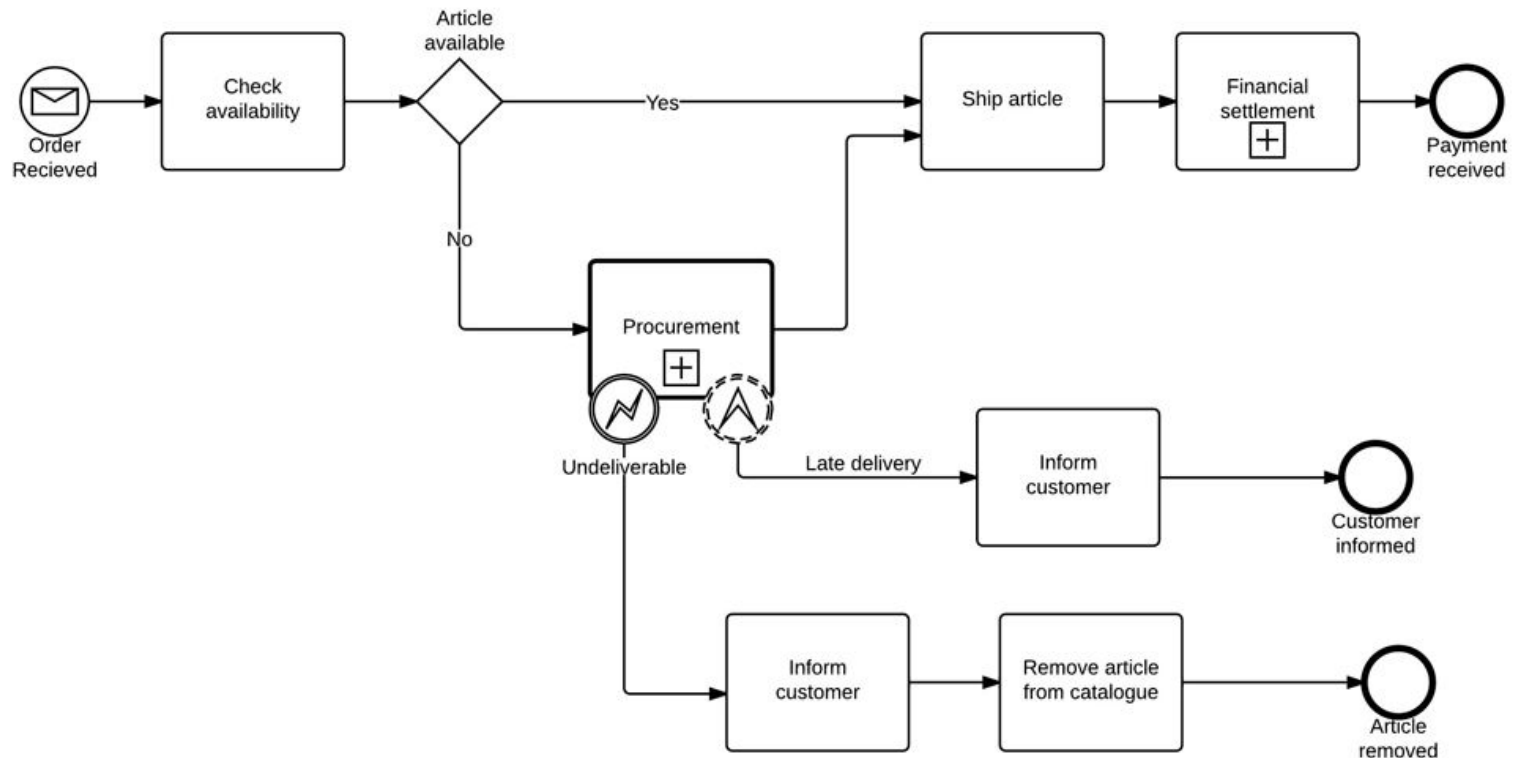- Timeouts

Event semantics: Escalations

- ## Handling unusual but **expected** behaviour
  - ○ Corrective actions (interrupting)
  - ○ Additional steps to be done in parallel (non-interrupting)

Event semantics: Errors

- Used for **serious** problem in process
- Throw - catch mechanism
  - Always interrupting
  - Always boundary event
- There should be some **error handling actions**
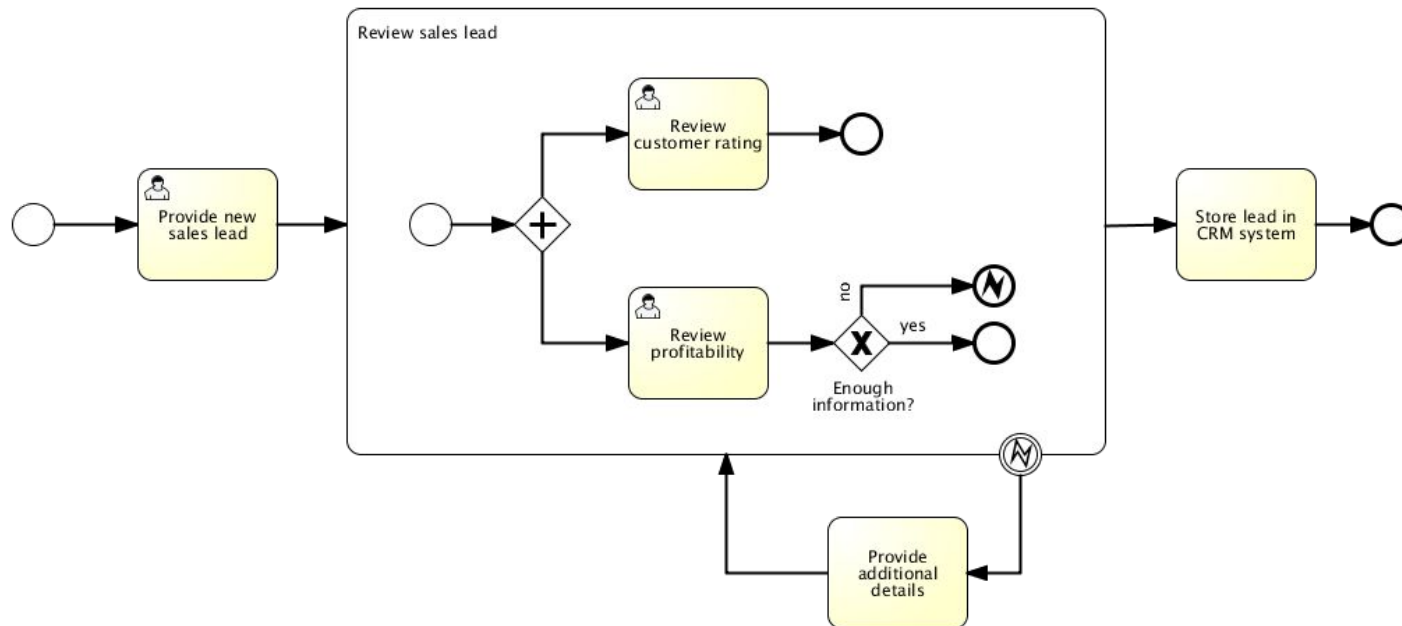
Event-based gateway
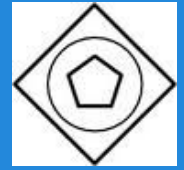
- Event-based gateway
  - Branching based on event, only one triggered
  - Different semantics – branched according to event that is placed after the gateway
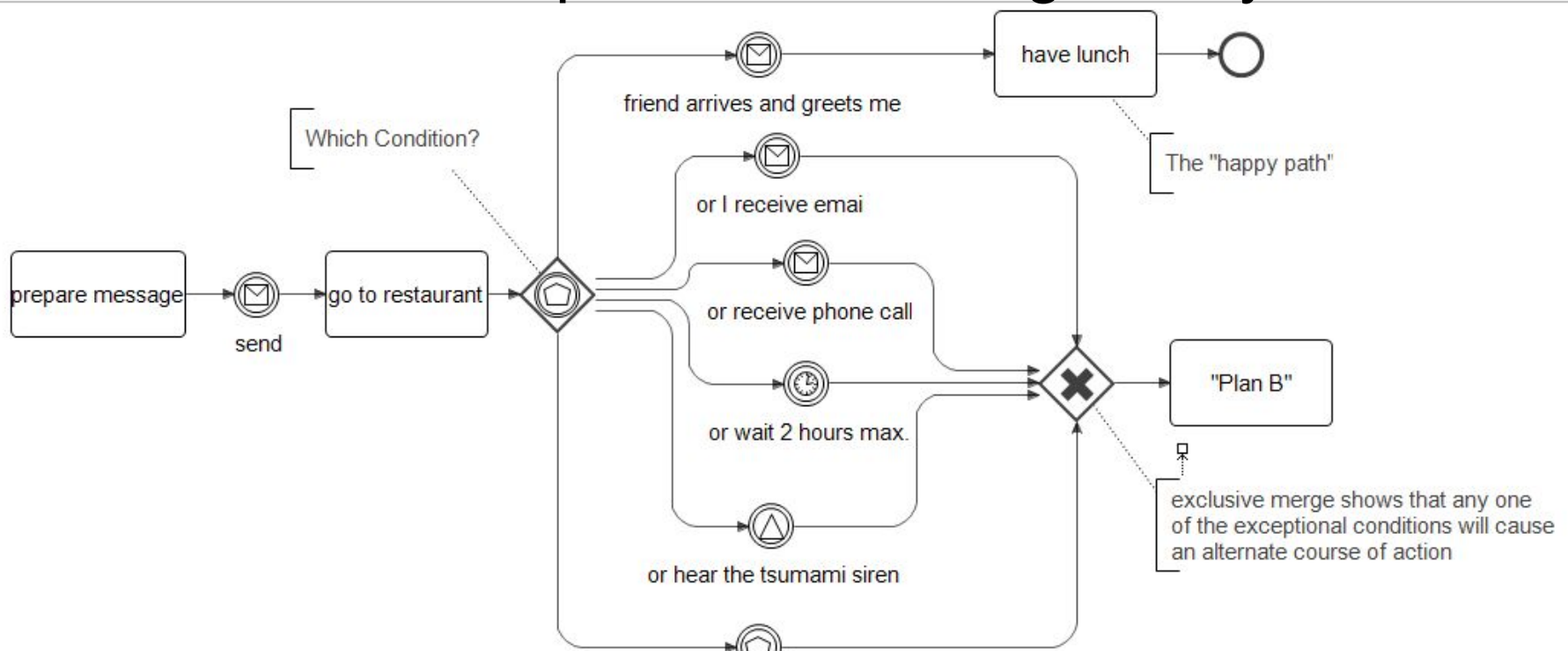
Multi-instance and Loop activity

- Multi-instance
  - Shortcut for a number (dynamically defined) of the same activities that run in parallel or in series.

- Loop
  - Shortcut for a repeating one activity until a condition is met.
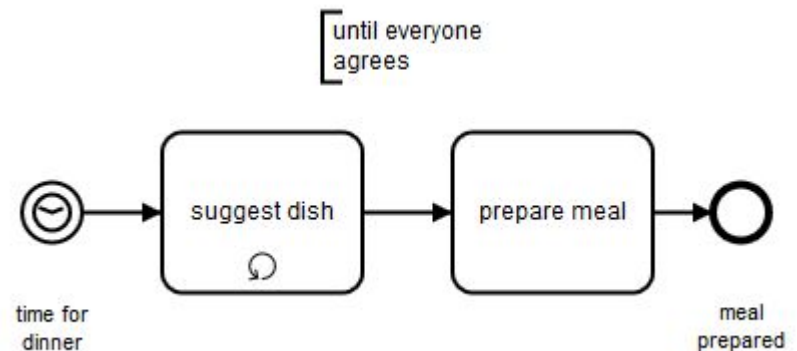
Recap: Inclusive OR-gateway
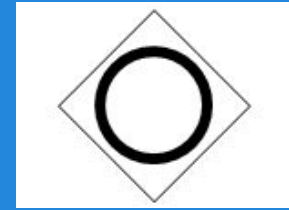
- One or more branches can be performed
- Depends on conditions
- Branches performed in parallel
- Waiting for all **activated** branches
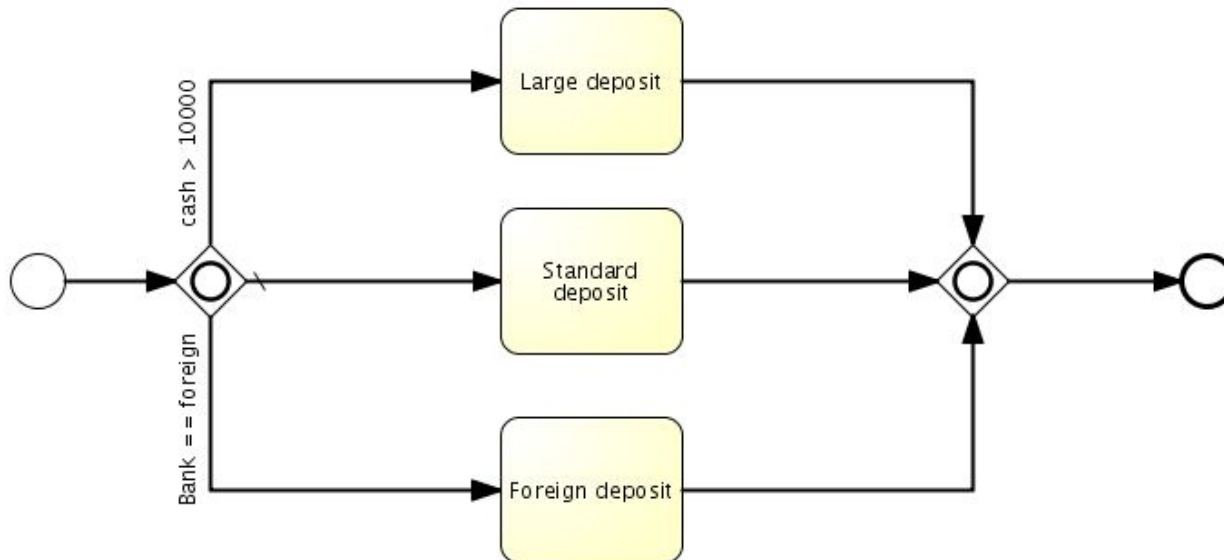
What is in not covered here

- Transactional events
  - Compensations
  - Cancellations events
  - Rollbacks
- Other diagrams covered in BPMN 2.0 specs
  - Choreography diagrams
  - Conversation diagrams

BPMN L2 summary

Seller

Input Data

Ad-hoc Sub Process

A sequential multi instance service task

Escalation End Event

Collapsed Sub Process

Message Flow denotes message exchange

Publish offer

Escalate Problems

Message

Offer

Offer

Clerk

New offer received

Check offer

Decide on offer

Offer rejected

Message End Event

Accept offer

Selecte Ads

Select Channels

Problems occured

Escalate problems to Manager

Escalate problems to Manager

Commit Auction

Link Intermediate Event (throwing)

Message Start Event

Exclusive Gateway

Signal End Event

Auction Inc.

Signal Start Event (non-interrupting)

Run Auction

Provide status

Event Sub Process

A Data Store stores documents

Seller notification

Intermediate Message Event (throwing)

A Termination End Event stops all existing flows

Timer Intermediate Event

Status req... received

Status provided

No bids received

Buyer store

Buyer info sent

Platform

Commit Auction

Auction starts

Auction over

Any bids?

count(bids)=0

Determine buyer

Auction finished

Link Intermediate Event (catching)

Ordinary Start Event

Receive bids

Ordinary End Event

Ordinary Task

Parallel Gateway (splitting)

Bidder info sent

Parallel Gateway (merging)

Task with sequential loop

Bidders

A Data Object with a collection

Group

(c) 2009 Frank Puhlmann, inubit AG

Bidder

inubit

integrating your business and IT

FIN
Questions?

PV207 – Business Process Management

Spring 2019   Jiří Kolář, Lubomír Dočkal