Session 5:
Business Process Modeling (BPMN) Events

Richard Welke
Director, CEPRIN
Professor, CIS
Robinson College of Business
Georgia State University
Atlanta, GA
Concept of a (business) event

- Definition: an event is “a noticed change in state”
  - It is an outcome or result –
    - not an activity or task
  - Two notions
    - An occasion (something that happens)
    - A result (important incident)

Examples & analogues
- Millions of events occurring in
  - Your body (handled by your brain)
  - In the room (e.g. voltage fluctuation, student entering room)
- Context – of interest to what/whom?
  - Interested in events that can effect (initiate) or affect (change how things are handled) in our business process

- Common kinds of “business events”
  - Accounting transactions (change affecting account balances)
  - Logistics events (“out-of-inventory”; delay in shipping)
  - Customer events (change in address)

- Formally: A “significant” change of state has occurred
  - The recording of this is the event
    - Typically – a timestamp plus the new state values
Some types of events

Terminology of creating/receiving an event
- “Throwing” – event message is generated as a result of an occurrence
- “Catching” – process waits to respond to an indicated event

Simple event
- A defined signal (message) captures the occurrence
  - Customer order arrives
  - Payment received
  - Credit approval task completes

Compound or “complex” event
- A conditional statement made up of simpler events
  - Create an event “IF order arrives AND not handled in five minutes”
The three basic event types

1. **Start**
   - Initiates the process (triggered by a signal)
   - Thin bordered circle (empty="none" start event)

2. **End** (normal and termination)
   - Ends the process
   - Single thick bordered circle
   - Also, the “Terminate” event

3. **Intermediate** (double-ringed)
   - Along a sequence flow …
     - Stop processing along that flow and wait for (catch) the event
     - Or … generate (throw) the event
   - On a task/activity border boundary
     - Listen for the event over duration of task
     - If attached event occurs while activity running, pass control to boundary flow
External events

External event
- Event originating from outside the process
  - Customer pool, backend system, web service
- Listened for (“caught”) by process
- Example message types include:
  - SOAP, JMS, email, fax, phone, form, etc.
Internal (to process) events

- Signal originates within the process of interest
- Example types:
  - Timer (countdown or calendar based)
  - Link (used to connect long sequence flow lines)
  - Conditional
  - Transaction failure (compensation)
  - Signal event (broadcast/listen; T/C)
  - Multiple (fires off multiple event types; T/C)
  - Message (T/C)
- In many cases, can signify either a “throw” (send) or “catch” (receive) of the event (T/C)
- Set by property of the event (“Is throw”)
BPMN 2.0 adds Events

A number of additional event types have been added in BPMN 2.0

- As well, there is a dashed-line, double-ringed (intermediate) “non-blocking” style of event that’s been added
- For more on these, see the BPMN “cheat sheet” listed on the course wiki
- We won’t cover the non-blocking style events or usage in this course
  - Or quiz on these 😊
Start event

- Creates a new process (or sub-process) instance (a single transaction moving through the business process)
  - Analogy: Baking a cake
    - Cake recipe (the process model)
    - Each person baking a cake from that recipe (the process instances)

- Sub-process start event can be omitted if no incoming sequence flow & no end event drawn
  - but this is not best practice!

- Sub-process can have multiple starts but implies parallel flows instantiated
  - also not best practice

- Only "None" start event used in a sub process
  - The ‘trigger’ is the sequence flow into the sub process (not an event signal)

Sub-process start event equivalences

Recommended
**Start event types**

- **Message start**
  - Triggered by external signal
  - Phone call, form, service request, etc.

- **Timer start**
  - Scheduled to run at specific time or cycle
  - Examples: batch processing, monthly reporting
  - Q: Who’s the customer?

- **Rules/Condition start**
  - Starts when a particular data condition or rule is met
    - E.g. Rule: (Start bounced check process) true when check amount for account exceeds account balance

- **Signal start**
  - When a signal (of type/number specified) is “published”
  - When the signal meets this signal definition, initiate a process instance (publish/subscribe model)

- **Multiple start (see next slide)**
Start event types

- Multiple event starts (triggers) to same process
  - Alternative channels that start processing but then have common processing steps afterwards
  - Sometimes called: “Bootstrapped start”
- Two ways to show this:
  - BPMN 1.0
  - BPMN 1.1

Note that either can be used; one uses an event gateway (top) while the other uses a multiple start event
**Normal end event**

- **Ends a sequence flow**
  - Process complete when **all** enabled sequence flows end

- May ‘throw’ a result signal
  - To another process pool (end event message)
  - To a parent activity in same pool (e.g. error end event)
  - In sub-processes to end, terminate or cancel

Note: there are up to three separate flows. All flows must complete before process is completed.

Q: What happens if all three don’t complete?
“Style” issue: the labeling of end events

Silver recommends that diagrams/models:

- Define the **different kinds of outcomes a process can have**
- Define end events for each outcome type (normal and non-normal end conditions of the associated service provided)
- Label each end event according to its ending type (e.g., expected result …., error type ….)
- Use arrows to them from the various paths that result in a “happy” end or an “unhappy” (error) ending.

See next slide for example of this, or Silver’s book
Example use of labeled end events
Terminate end event

- Ends the process or sub process **unconditionally**
  - Even if other active paths still not completed
  - Or, other instances of repeating activity

- **Kills** all incomplete paths in the containing process or sub-process

**Note:** In a **sub process**, a terminate only ends the sub-process (not the parent process)
End events for exception handling

- **Error end event**
  - Triggers a (named) paired error intermediate event

- **Cancel end event**
  - Aborts a business transaction and triggers a (named) paired cancel intermediate event
  - Way to do some post-processing after “terminating”

- **Compensate end event**
  - Compensation means the processing needed to back-out an incomplete transaction
  - Triggers “compensation” of a target activity

- All of these would have matching start events in another process pool
End events for exception handling

- Signal end event
  - Publishes a (named) signal when reached and ends flow

- Message end event
  - Ends the flow and sends a message to another pool

- Multiple end event
  - Ends the flow and generates multiple events (e.g., message, error)
  - These events are specified in the properties of the multiple event
Message intermediate event

*In sequence* flow can mean either
- Send/throw (publish) or wait/catch (subscribe)
- In BPMN 1.1 throw/catch use different icons
  - Throw: solid/filled vs.
  - Catch: non-solid/outline
- Indicate using right-click on event to pick type
Message intermediate event

Boundary attached event
- If message received *during* activity
- Initiate exception flow
- Message always comes from outside pool
- Can be:
  - “Blocking” (solid lines) or
  - “Non-blocking” (dashed lines)
**Timer intermediate**

- **In a sequence flow**
  - Wait until (Date/time), or
  - Wait until (Duration countdown) before proceeding to next task

- **Boundary attached**
  - Timeout at (date/time) or duration
  - Abort current activity
  - Proceed on timer “exception” path
Error event

In a sequence flow

- **Cannot** wait for an error event in sequence flow or gateway
  - In BPMN 1.1 error event only allowed from end (or outside) event

Attached to task or sub-process

- Aborts an activity on error
- On automated activities usually signifies system fault
  - The throw of an error signal is implicit; not drawn

See next slide for more detail
Error throw-catch

- Exception in a sub process “thrown” to parent process
- Show as throw from error end event in sub process to attached error event of sub process placeholder on main
  - Linkage done by ErrorCode attribute
Using a sub-process to scope an event

- Bounded activity determines scope of attached event
- So ... you wrap the process fragments (detail) in a sub-process
  - Example:
    - Assume that if a customer cancels an order while either tasks 1, 2, or 3 occur (the scope), “Handle”
Signal event

- More **flexible** than message event
  - Can signal within a pool (or between)
  - Publish-subscribe concept
- More **general** than error event
  - Can throw from a start, intermediate or end event
  - Can catch to a start or intermediate event
- Example … “milestone pattern”
Things to watch out for (1)

- Terminating process from exception in sub process
  - Sequence flow cannot cross sub process boundary

- Instead:
Things to watch out for (2)

- Terminating a process from a business exception within a sub process
  - Terminate event only ends the sub process, not the main process

Instead: