ANR PARDI

Meeting at ~t_0+12

Pascal Poizat

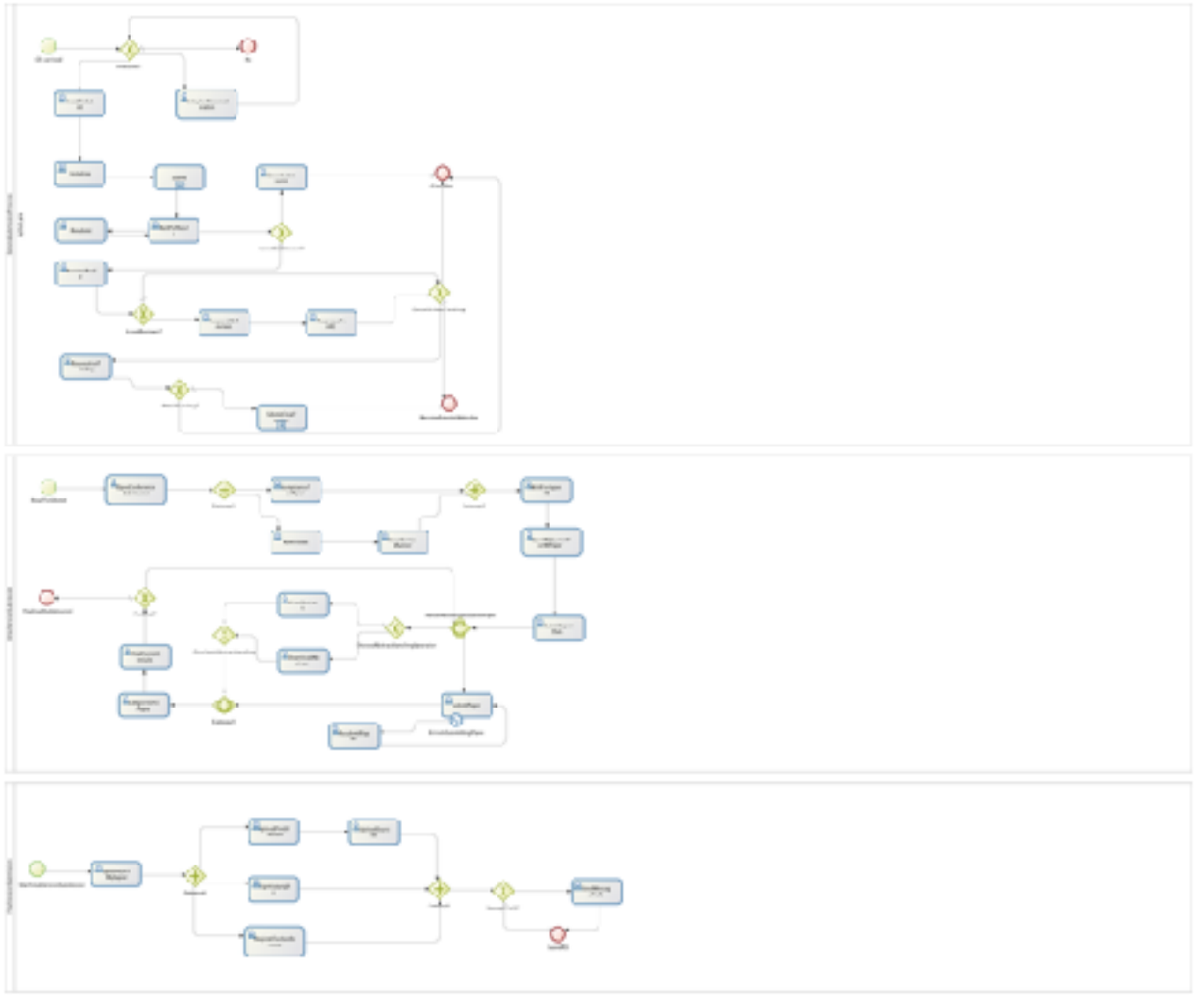
joint work with S. Baarir, L.M. Hillah, and S. Houhou

supported by ANR

Université Paris Nanterre

eConference case study

- we are late on this one



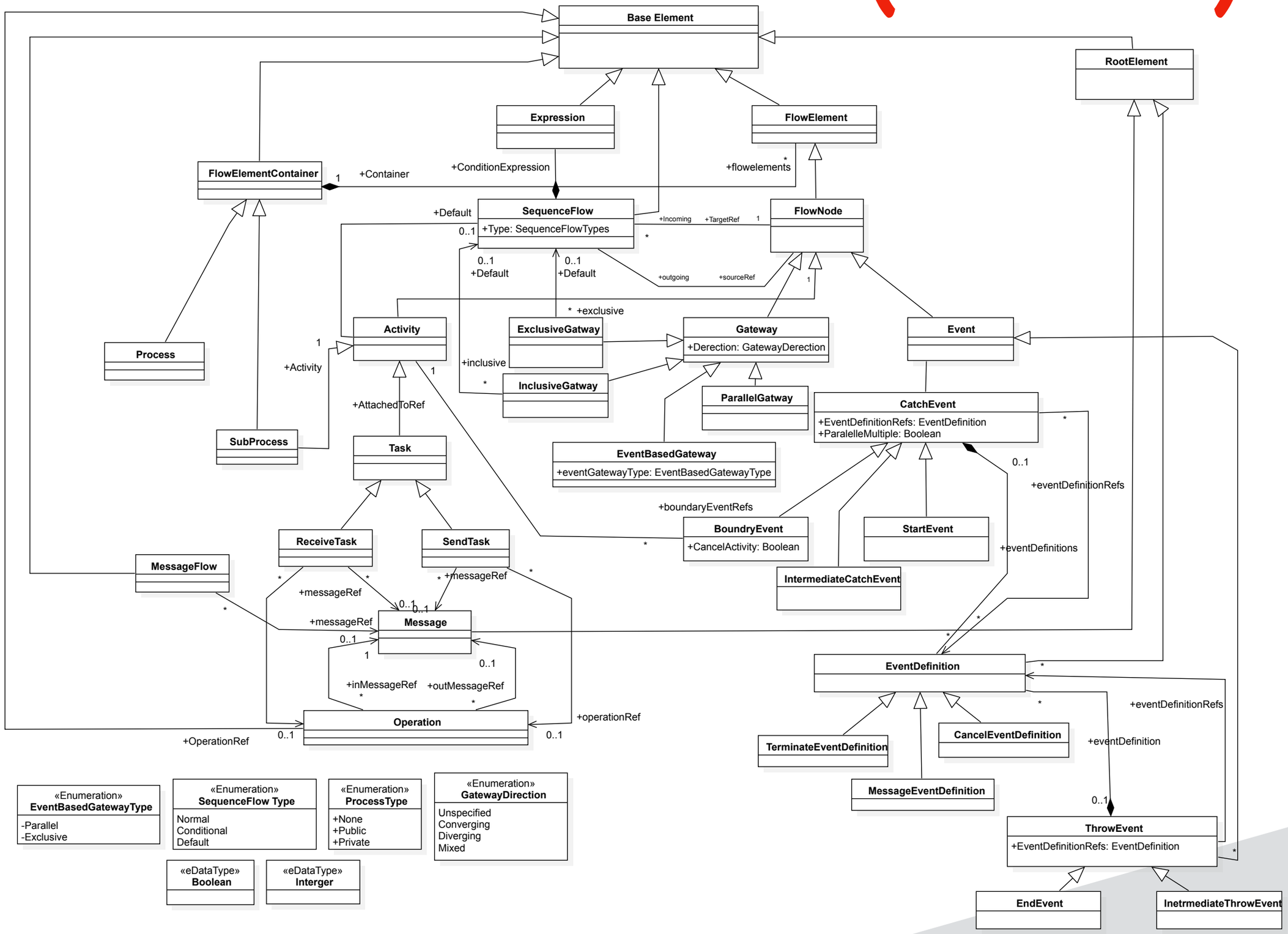
parameterized WF model/DSL

- **we chose to begin with a subset of BPMN**
  following our approach with intermediate formats (CIF, PIF)

- **this subset gives us basic building bricks**
  abstract syntax + semantics

- **it may be extended later on**
  taking inspiration from YAWL and Workflow Patterns

BPMN meta model (subset)



approach

- **colored graph** based formalization
  syntax: typed nodes and typed edges
  semantics: token-game based

- **environment as a parametric message collection (PMC)**
  simulates open or closed system
  algebraic structure with an operation to check if a message is « present »

BPMN types

- we define a set **Types** of « types » corresponding to the meta-model of our BPMN subset

$T_{SF} = \{T_{NSF}, T_{CSF}, T_{DSF}\}$ — sequence flows

$T_{CO} = T_{SF}$ — connection flows

$T_T = \{T_{AT}, T_{ST}, T_{RT}\}$ — tasks

$T_A = T_T \cup \{T_{SP}\}$ — activities

$T_{EBG} = \{T_{EEBG}, T_{PEBG}\}$ — e-based gateways

$T_G = T_{EBG} \cup \{T_{EG}, T_{IG}, T_{PG}\}$ — gateways

$T_{SE} = \{T_{NNE}, T_{MNE}\}$ — start events

$T_{CE} = T_{SE} \cup \{T_{MICE}, T_{MIBE}\}$ — catch events

$T_{EE} = \{T_{NEE}, T_{MEE}, T_{TEE}\}$ — end events

$T_{TE} = T_{EE} \cup \{T_{NITE}, T_{MITE}\}$ — throw events

$T_E = T_{CE} \cup T_{TE}$ — events

$T_{FN} = T_A \cup T_G \cup T_E$ — flow nodes

raw BPMN graphs

- **Given the set of BPMN types, a raw BPMN graph is a tuple** $(N^+, E, src, tgt, type_N, type_E, attach, Sig, V, cond, Msg, msg)$ **where:**
  - $N^+ = N \cup N^{SP}$ **are nodes,** $type_N: N \rightarrow T_{FN}$,
  - E **are edges,** $src: E \rightarrow N^+$, $tgt: E \rightarrow N^+$, $type_E: E \rightarrow T_{CO}$
  - **attach:** $N \rightarrow N^{SP}$ **relates boundary events and sub-processes**
  - Sig **is a signature,** V **are variables,** $cond: E \rightarrow T^{Sig(\mathbb{B},V)}$ **are flow conditions,**
  - Msg **are message types,** $msg: N \cup E \rightarrow Msg$
  - **plus constraints (not given here)**

+ we define helper functions, e.g., $in: N \rightarrow 2^E$ and $out: N \rightarrow 2^E$

BPMN types (extended)

- we **update** the set **Types** of « types » corresponding to the meta-model of our BPMN subset

$T_{SF} = \{T_{NSF}, T_{CSF}, T_{DSF}, \mathbf{T_{SPF}}\}$ — sequence flows

$T_{CO} = T_{SF} \cup \{\mathbf{T_{ATTCH}}\}$ — connection flows

$T_T = \{T_{AT}, T_{ST}, T_{RT}\}$ — tasks

$T_A = T_T \cup \{T_{SP}\}$ — activities

$T_{EBG} = \{T_{EEBG}, T_{PEBG}\}$ — e-based gateways

$T_G = T_{EBG} \cup \{T_{EG}, T_{IG}, T_{PG}\}$ — gateways

$T_{SE} = \{T_{NNE}, T_{MNE}\}$ — start events

$T_{CE} = T_{SE} \cup \{T_{MICE}, T_{MIBE}\}$ — catch events

$T_{EE} = \{T_{NEE}, T_{MEE}, T_{TEE}\}$ — end events

$T_{TE} = T_{EE} \cup \{T_{NITE}, T_{MITE}\}$ — throw events

$T_E = T_{CE} \cup T_{TE}$ — events

$T_{FN} = T_A \cup T_G \cup T_E \cup \{\mathbf{T_{Env}}\}$ — flow nodes

flat-env BPMN graphs

- **Given the set of BPMN types, flat-env BPMN graph is a tuple** (N_{Env}, E, src, tgt, $type_N$, $type_E$, Sig, V, cond, Msg, msg) **where:**
  - $N_{Env} = N \cup \{n_{Env}\}$ **are nodes,** $type_N: N_{Env} \rightarrow T_{FN}$,
  - E **are edges,** src: $E \rightarrow N_{Env}$, tgt: $E \rightarrow N_{Env}$, $type_E: E \rightarrow T_{CO}$,
  - Sig **is a signature,** V **are variables,** cond: E -> $T^{Sig(\mathbb{B},V)}$ **are flow conditions**
  - Msg **are message types,** msg: $N \cup E \rightarrow$ Msg
  - **plus constraints, e.g.,**

$type_N(n_{Env}) = T_{Env}$, $msg(n_{Env}) = \bigcup_{n \in N} msg(n)$

$\forall e \in E, type_E(e) = T_{ATTCH} \Rightarrow type_N(tgt(e)) = T_{MIBE}$

well formed BPMN graphs

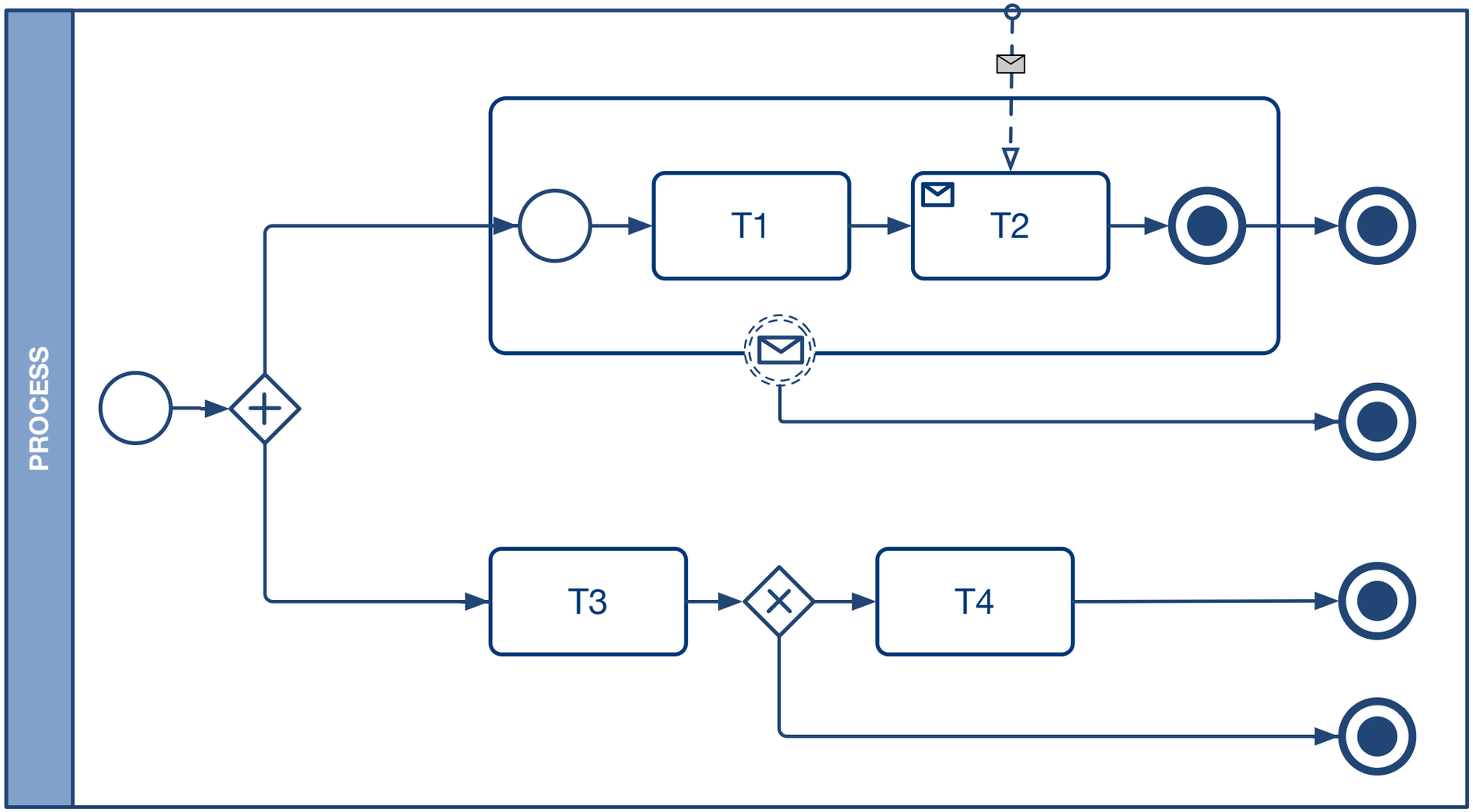
- we formalize the BPMN syntactic rules, e.g,

$$\forall\, e \in E,\ type_E(e) = T_{MF} \Rightarrow type_N(src(e)) \in \{T_{ST}, T_{MITE}, T_{MEE}\}$$

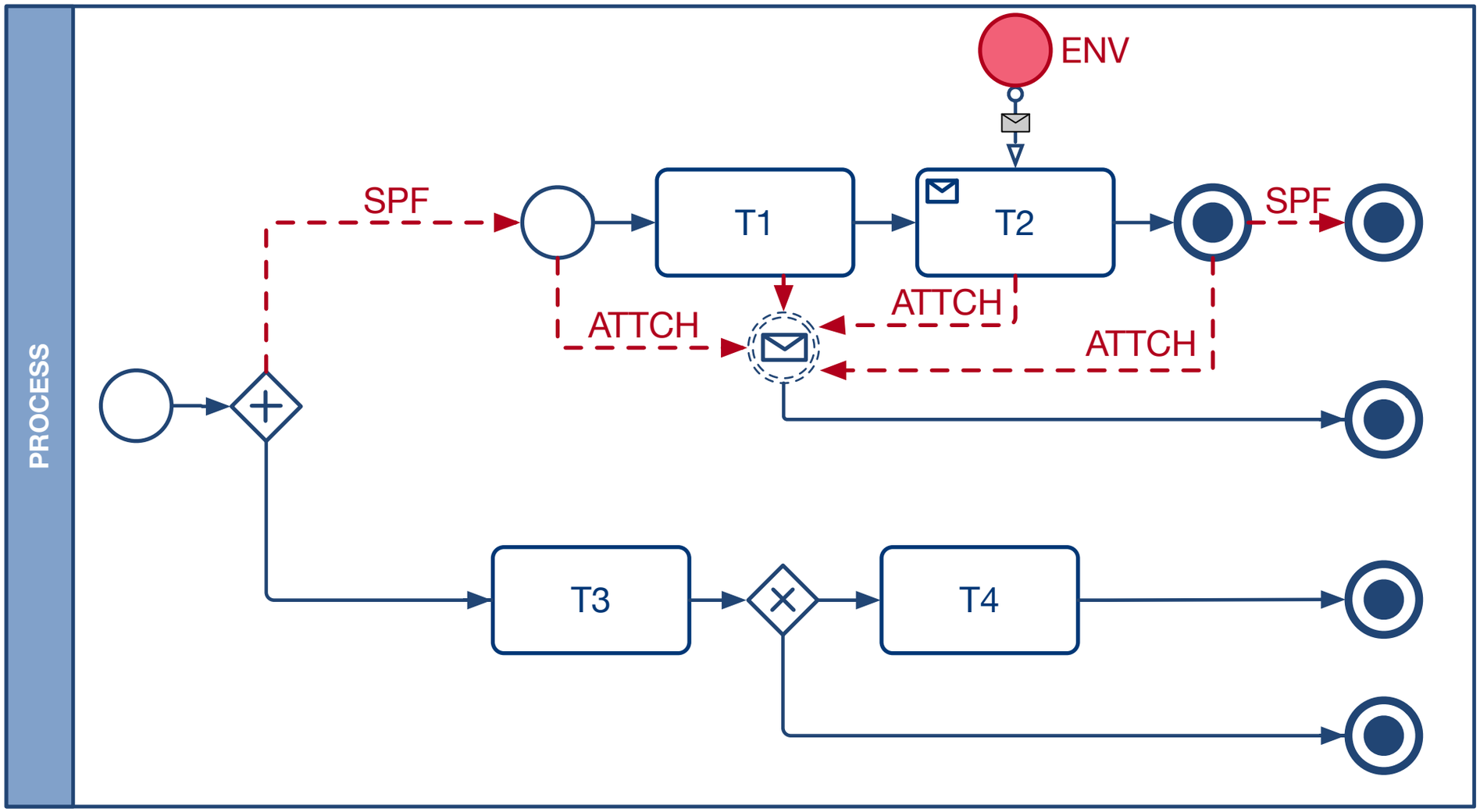
$$\forall\, n \in N,\ type_N(n) = T_{IG} \Rightarrow$$

$$(\forall\, e \in out(n),\ type_E(e) \in \{T_{CSF}, T_{DSF}\} \wedge$$
$$\exists!\, e \in out(n),\ type_E(e) = T_{DSF})$$

from raw to flat BPMN graphs



from raw to flat BPMN graphs



BPMN formal semantics (1/2)

- given a **term interpretation**, eval, and a **PMC instance**
- we define a **configuration of a process** as a function

  state: $N \cup E \rightarrow Bag(Msg \cup \{\varepsilon\})$

  such that state(n) $\in$ Bag(Msg) **if** typeN(n) $\in$ {**...**} **else** $\mathbb{N}$

- initial state: a black token for the initial node and an initial PMC configuration for n_{Env}

BPMN formal semantics (2/2)

- the **dynamics of the process** is defined as a transition between two states

- defined in terms of micro-transitions for each (kind of) node and edge in the graph

Abstract task

ready to start

- state(n) = 0
  ∧ state'(n) = 1
  ∧ ∃ e ∈ in(n),
    state(e) ≥ 1
    ∧ state'(e) = state(e) - 1
    ∧ ∀ e' ∈ in(n), e' ≠ e,
      state'(e') = state(e')

ready to complete

- state(n) = 1
  ∧ state'(n) = 0
  ∧ ∃ e ∈ out(n),
    state'(e) = state(e) + 1
    ∧ ∀ e' ∈ out(n), e' ≠ e,
      state'(e') = state(e')

Message reception (MICE)

ready to start

- $\text{state}(n) = \varnothing$

$\wedge\ \exists\ m \in \text{msg}(n),\ \textbf{check}(n_{Env}, m),$

$\text{state}'(n) = 1.m$

$\wedge\ \text{state}'(n_{Env}) = \text{state}(n_{Env}) - \{1.m\}$

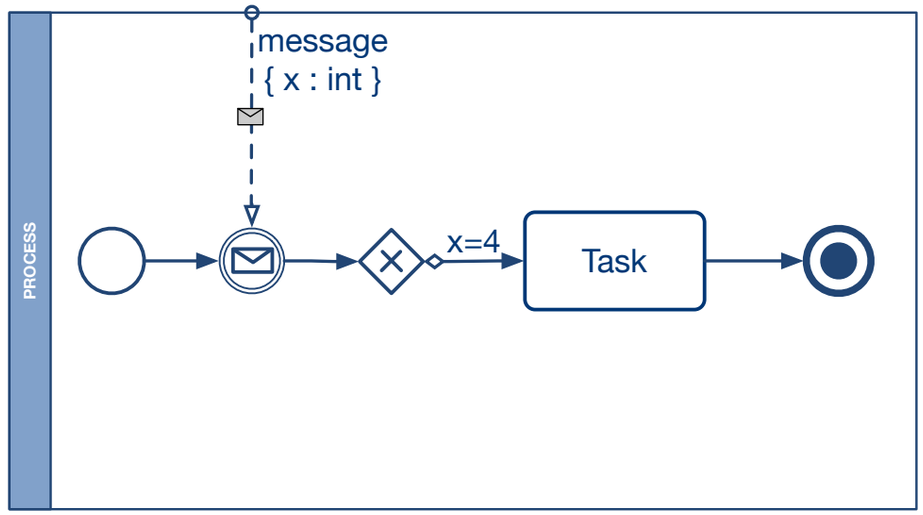
$\wedge\ \exists\ e \in \text{in}(n),$

$\text{state}(e) \geq 1$

$\wedge\ \text{state}'(e) = \text{state}(e) - 1$

$\wedge\ \forall\ e' \in \text{in}(n),\ e' \neq e,$

$\text{state}'(e') = \text{state}(e')$



any (m2)	any	✗
<m1(4),_>	head	✓
<m2,m1(_)>	head	✗
{m2,m1(4)}	in	✓
{m2,m1(3)}	in	✗
<m2,m1(4),m1(3)>	first	✓
<m2,m1(3),m1(4)>	first	✗
{m2,m1(4),m2(3)}	in	?

perspectives

- mapping of the model semantics to Why3 / TLA+
- process-oriented verification
- feed-back for the definition of the PWF DSL
- more parameters, better data, collaborations, resources