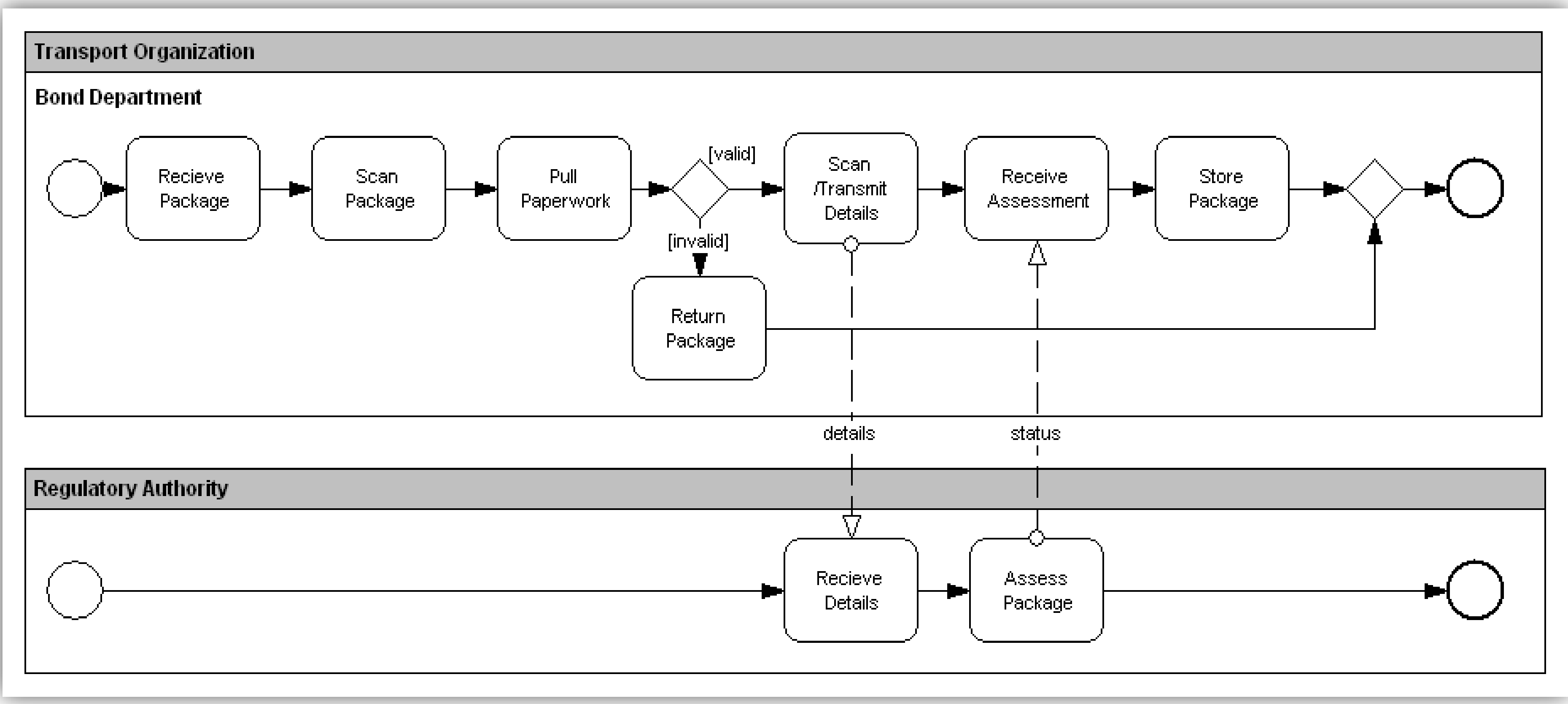
CSCI427/927 Systems Development

Semantic processes and services

BPMN


Transport Organization
Bond Department
Recieve Package
Scan Package
Pull Paperwork
[valid]
[invalid]
Scan /Transmit Details
Receive Assessment
Store Package
Return Package
details
status
Regulatory Authority
Recieve Details
Assess Package

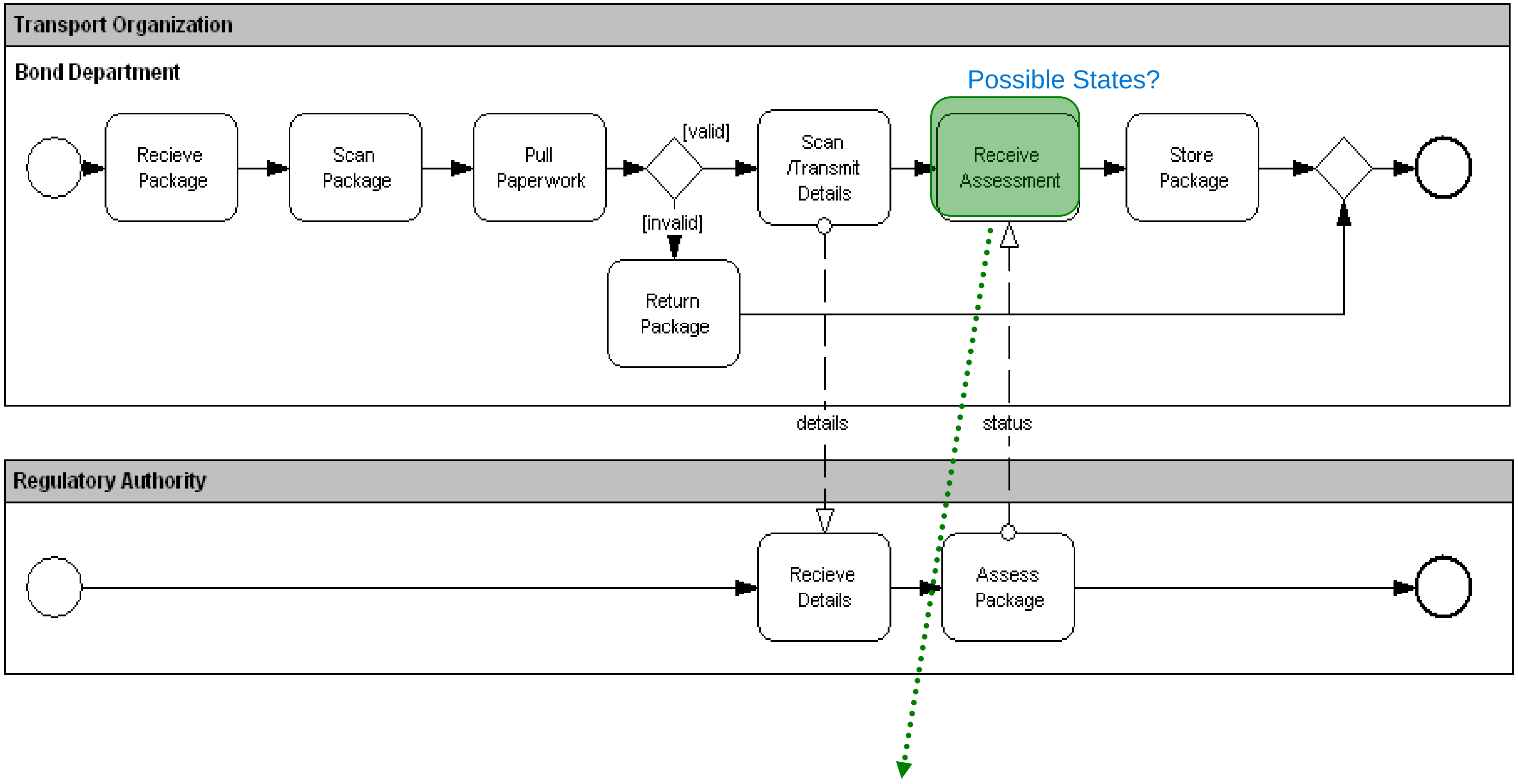
Challenges with BPMN

- BPMN is the industry standard
- Only the coordination semantics of business processes are explicitly described in BPMN.
  - E.g. Task A must precede task B.
- But the **semantics of processes** in terms of their **effects** are not explicitly described
  - Does not provide any indication of what is done (i.e. their effects) by tasks A and B (beyond what might be implicit in their name).

Challenges with BPMN (cont.)

- Unable to determine from a process model in BPMN what the effects achieved by a process might be at any point in the process model.
  - (**Functional**) effects of a process
  - (**Non-functional**) performance/Quality-of-Service factors of a process

Challenges with BPMN (cont.)



What would the effects of this process be if it reached this point?

Benefits of effect annotation

- Compliance checking: Determining whether business processes comply with regulatory or legislative requirements.
  - E.g. Whether a process model complies with a set of rules.
  - Example: "*Admit critically injured patients into consultation in no greater than 15 minutes after triage*"
    - Functional effect: patient admission
    - Non-functional objective: minimizing delay between triage and patient admission
- How to modify a process model that does not comply to obtain one that does?

Benefits of effect annotation (cont.)

- Establish inter-process relationships
  - Change impact analysis
    - a change made to one business process can potentially affect a range of other processes that are related to the process being changed
    - estimating the potential effects of changing a business process to other business processes in an organization's business process repository.
  - 6,000+ process models in Suncorp's process model repository for insurance.

BPMN: Functional effects

- Given any point in a process model, we wish to determine (at **design-time**) what the effects of the process would be if it executed up to that point
- The answer is non-deterministic (represented by a set of <u>effect scenarios</u>)
  - Process steps may "undo" the effects of prior process steps – alternative resolutions of these inconsistencies lead to alternative effect scenarios
  - Processes may take alternative paths (determined by run-time parameters) to reach a given point

Effect annotations: I

- Analysts annotate each BPMN task with *immediate effect annotations*
- These immediate effects are accumulated (usually in an automated fashion) to obtain *cumulative annotations* describing:
  - Functional effects up to that point in the process
  - Non-functional properties

Effect annotations: II

- Effect annotations can be:
  - Informal (e.g. plain English)
  - Formal (e.g., FOL, LTL, CTL etc.)
  - Controlled Natural Language: This involves specifying effects using a limited repertoire of strictly formatted natural language sentence patterns, which can be directly mapped to underlying formal assertions

Example

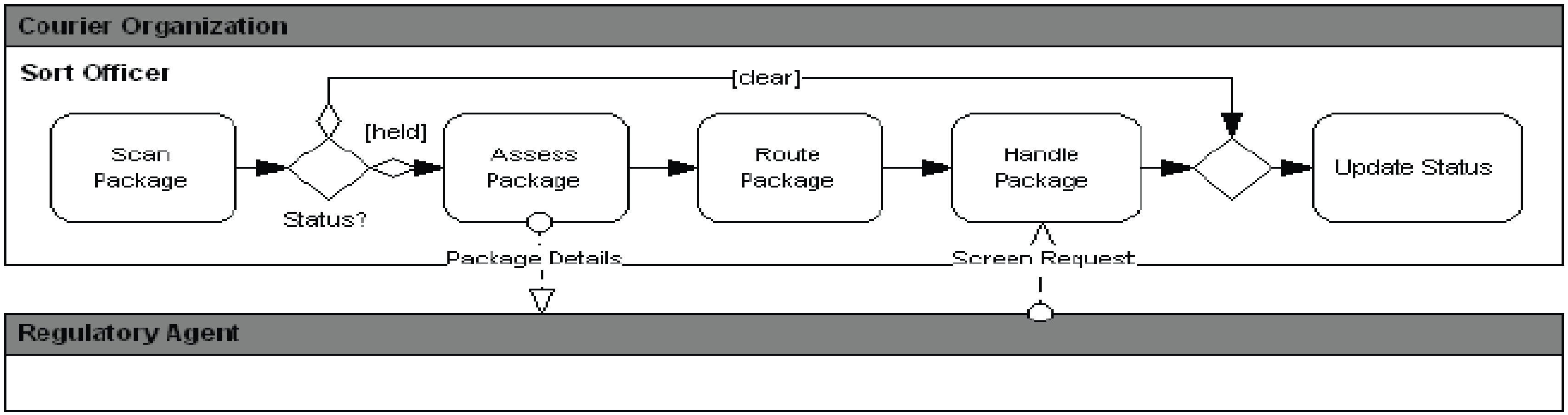
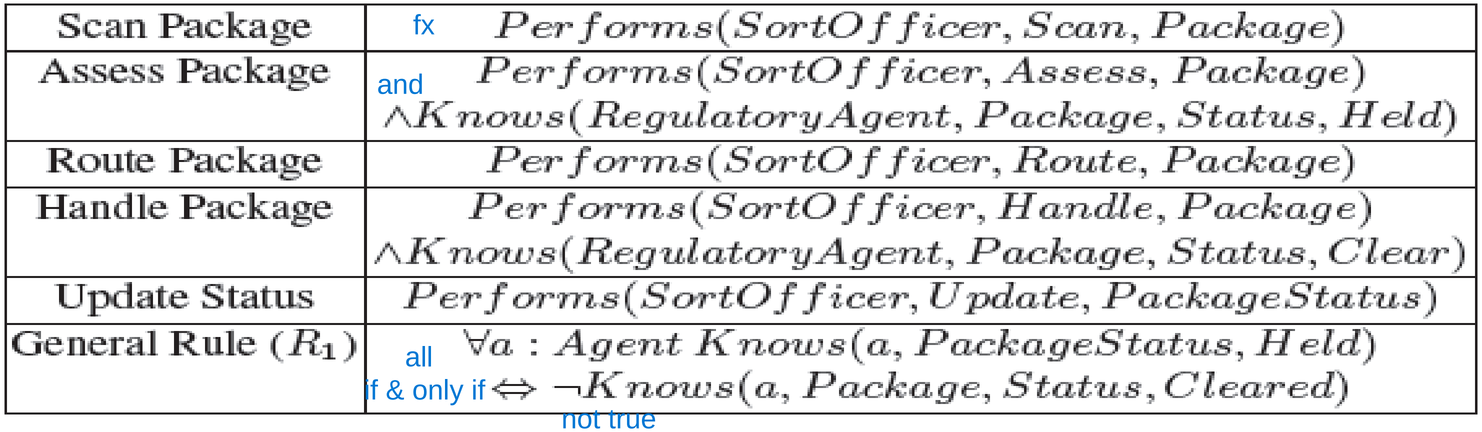


Fig. 1. Package Screening Process (O)

Table 1. Annotation of Package Screening Process (O) in Figure 1

FOL



Scan Package	fx $Performs(SortOfficer, Scan, Package)$
Assess Package	and $Performs(SortOfficer, Assess, Package)$ $\wedge Knows(RegulatoryAgent, Package, Status, Held)$
Route Package	$Performs(SortOfficer, Route, Package)$
Handle Package	$Performs(SortOfficer, Handle, Package)$ $\wedge Knows(RegulatoryAgent, Package, Status, Clear)$
Update Status	$Performs(SortOfficer, Update, PackageStatus)$
General Rule (R_1)	all $\forall a : Agent\ Knows(a, PackageStatus, Held)$ if & only if $\Leftrightarrow \neg Knows(a, Package, Status, Cleared)$ not true

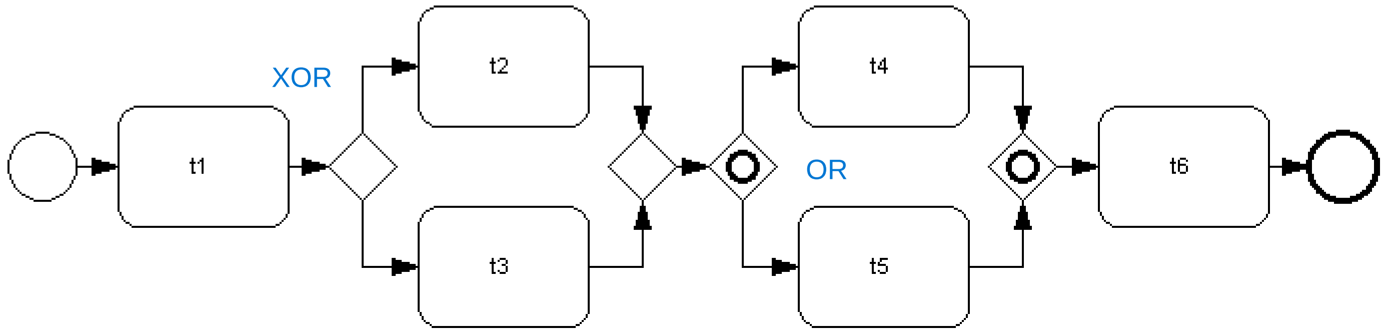
Effect scenarios

An **effect scenario** consists of:

- A single consistent **cumulative effect** assertion
- A *scenario label*: Describes the precise path through the model that would have been taken to achieve the effect scenario. Represented as a **<u>sequence</u>** consisting of either:
  - Activity identifiers
  - Sets whose elements are scenario labels (needed to deal with AND-splits)
- An *exclude set*: A set of prefixes of scenario labels
  - Used to ensure that effect scenarios spawned due to XOR-splits are never combined via AND- or OR-merges

We will occasionally use "effect scenario" to refer to the effect description component (the 1st component) of an effect scenario.

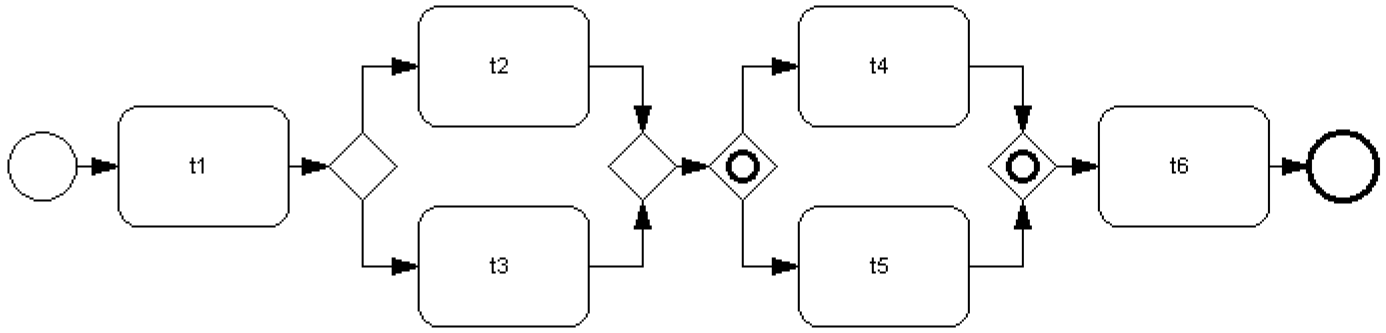
Example



Scenario Label exclude set

- $t_4 : Scenario(1) : \langle\langle t_1, t_3, \{\langle t_4\rangle\}\rangle, \{\langle t_1, t_2\rangle\}\rangle$
  $t_4 : Scenario(2) : \langle\langle t_1, t_2, \{\langle t_4\rangle\}\rangle, \{\langle t_1, t_3\rangle\}\rangle$
- $t_5 : Scenario(1) : \langle\langle t_1, t_3, \{\langle t_5\rangle\}\rangle, \{\langle t_1, t_2\rangle\}\rangle$
  $t_5 : Scenario(2) : \langle\langle t_1, t_2, \{\langle t_5\rangle\}\rangle, \{\langle t_1, t_3\rangle\}\rangle$

Example (cont.)



What are the scenario labels and exclude sets for task t6?

- $t_6 : Scenario(1) : \langle\langle t_1, t_3, \{\langle t_4\rangle, \langle t_5\rangle\}, t_6\rangle, \{\langle t_1, t_2\rangle\}\rangle$
  $t_6 : Scenario(2) : \langle\langle t_1, t_3, \{\langle t_4\rangle\}, t_6\rangle, \{\langle t_1, t_2\rangle\}\rangle$
  $t_6 : Scenario(3) : \langle\langle t_1, t_3, \{\langle t_5\rangle\}, t_6\rangle, \{\langle t_1, t_2\rangle\}\rangle$
  $t_6 : Scenario(4) : \langle\langle t_1, t_2, \{\langle t_4\rangle, \langle t_5\rangle\}, t_6\rangle, \{\langle t_1, t_3\rangle\}\rangle$
  $t_6 : Scenario(5) : \langle\langle t_1, t_2, \{\langle t_4\rangle\}, t_6\rangle, \{\langle t_1, t_3\rangle\}\rangle$
  $t_6 : Scenario(6) : \langle\langle t_1, t_2, \{\langle t_5\rangle\}, t_6\rangle, \{\langle t_1, t_3\rangle\}\rangle$

Computing cumulative effects

- We need to define the accumulation of effect scenarios over the following:
  - Pairs of contiguous tasks
  - XOR-, AND-, OR-merges
  - XOR-, AND-, OR-splits

Contiguous task accumulation

- Let <ti, tj> be a contiguous pair of tasks connected by a sequence flow (ti precedes tj)
- Let e_i be an effect scenario associated with ti and e_j be an effect scenario associated with tj.

$$e_i = \{c_{i1}, c_{i2}, \ldots, c_{im}\} \text{ and } e_j = \{c_{j1}, c_{j2}, \ldots, c_{jn}\}$$

  - Assume that effect assertions are written in **conjunctive normal form** (CNF) consisting of prime implicates
- If $e_i \cup e_j$ is **consistent**, then $acc(e_{i,} e_j) = e_i \cup e_j$
  - Else $acc(e_{i,} e_j) = e_j \cup X$ where $X=\{C \mid C \subseteq e_i$ and $C \cup e_j$ is **satisfiable**$\}$

Conjunctive normal form (CNF) is a conjunction of clauses, where a clause is a disjunction of literals; otherwise put, it is an AND of ORs

Consistent theory is one that does not contain a contradiction

The role of background knowledge

- Inconsistencies in effects can sometimes be determined only via **background knowledge**.
  - Example: The propositions "RequestApproved" and "RequestDenied" are inconsistent only because of the rule:
    - RequestApproved $\rightarrow$ $\neg$RequestDenied
- In the satisfiability checking described above, we implicitly assume the existence of an background knowledge base KB.
  - Thus: A U B is satisfiable means that A U B U KB does not contain inconsistencies

Contiguous task accumulation (cont.)

- acc(e_i, e_j) is non-unique
  - there are multiple alternative sets that satisfy the requirements for consistency.
- In other words, the cumulative effect of the two tasks consists of
  - the effects of the second task, plus
  - as many of the effects of the first task as can be consistently included.
  - We remove those clauses in the effect annotation of the first task that contradict the effects of the second task.

Example

$$KB = r \rightarrow \neg(p \wedge q)$$

$$\begin{tikzcd} {} \arrow[r] & \boxed{\textbf{T1}} \arrow[r] & \boxed{\textbf{T2}} \arrow[r] & {} \\ & p \wedge q & r & \\ & e_1 & e_2 & \end{tikzcd}$$

$$\neg(p \wedge q) \equiv (\neg p \vee \neg q)$$

What are the alternative effect scenarios describing the cumulative effects at T2?

Accumulation over AND-merges

- Let t1 and t2 be 2 tasks immediately preceding an AND-merge with sets of effect scenarios

$$E_1 = \{es_{11}, es_{12}, \ldots, es_{1m}\}$$

$$E_2 = \{es_{21}, es_{22}, \ldots, es_{2n}\}$$

- Let **t** be the task immediately following the AND-merge, with a immediate effect **e**

- The cumulative effect of task t is E and is defined as:

$$E = \{acc(es_{1i}, e) \cup acc(es_{2j}, e) | es_{1i} \in E_1 \text{ and } es_{2j} \in E_2$$

and $\{es_{1i}, es_{2j}\}$is exclusion-compatible$\}$

$ANDacc(E_1, E_2, e)$

- A pair of effect scenarios is exclusion-compatible iff the scenario label of neither matches any prefix in the exclude set of the other

Accumulation on OR-, XOR-merges

- XOR-merges $XORacc(E_1, E_2, e)$

$$E = \{acc(es_i, e) | es_i \in E_1 \text{ or } es_i \in E_2\}$$

- OR-merges

$$ORacc(E_1, E_2, e) = ANDacc(E_1, E_2, e) \cup XORacc(E_1, E_2, e)$$

While this is simple for 2 incoming flows, with n incoming flows, we would need to consider the following number of possibilities:

$$C_1^n + C_2^n + \ldots + C_n^n$$

Accumulation over XOR-, OR- and AND-splits

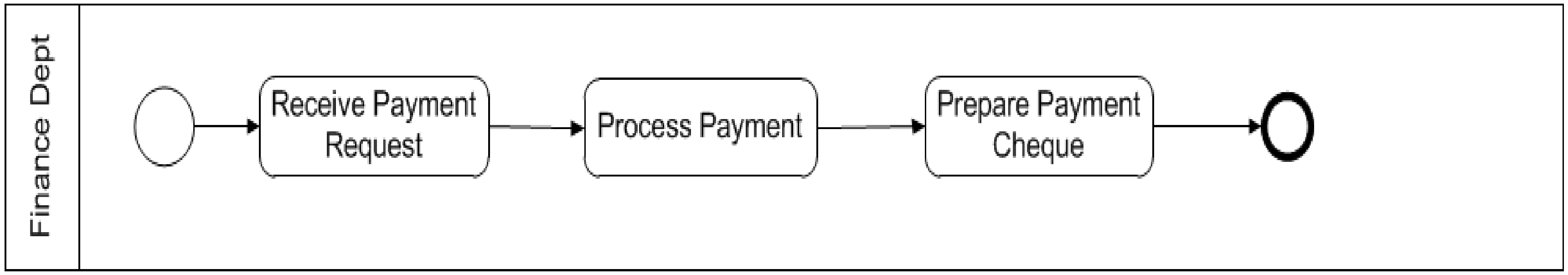
- Exclude sets: Each effect scenario of the n tasks on the n outgoing flows immediately following an XOR-split will include in its exclude set the labels of the effect scenarios of the remaining n-1 tasks. (This is also true for outgoing flows from OR-splits with mutually exclusive labels)
- Note: Guard conditions are conditions association with outgoing flows from a split gateway
- Enforce consistency of effect scenarios with guard conditions
- Guard conditions also accumulated over outgoing flows

Methodology for effect annotation

- Identify objects (often <u>business objects</u>) of interest in the domain
- For each task, identify which objects are impacted by the task
- Describe what the impact is for each business object, i.e. changing the state of the object
    - E.g. Task "Borrow a book" would impact the Book object (changing from "Available" to "On loan") and the Borrower Loan Record object (this record has now included this book)
- Pay special attention to inter-object relationships impacted by a task
    - E.g. Task "Enrolling in a subject" creates a relationship between object "Student[423432]" and object "Subject[CSCI927]"

Pen and paper exercise 1

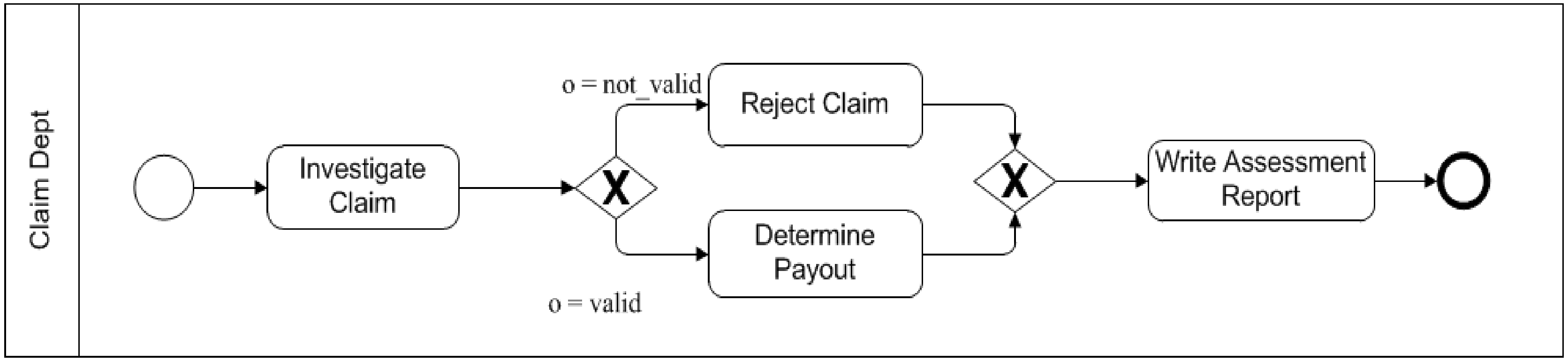
- Annotate and compute the cumulative effects of the following processes



Objects of interests: **Payment p**, **Money Amount t**
States: Payment p is authorized – **authorized(p)**
Relationships between objects:
- Payment request p of amount t received – **received(p, t)**
- Cheque is created for the payment with amount of money t – **cheque(p, t)**

Pen and paper exercise 2

- Annotate and compute the cumulative effects of the following processes

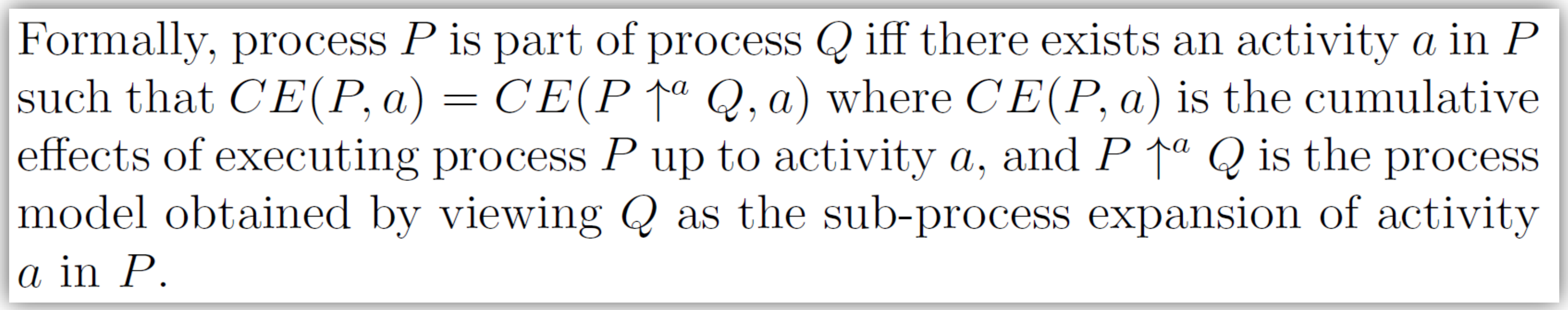


<u>Objects of interests:</u> **Claim c**, **Outcome o, Amount t**
<u>States:</u> Claim rejected or approved – **rejected(c), approved(c)**
<u>Relationships between objects:</u>

- Claim investigated with an outcome – **investigated(c, o)**
- Claim assessed with a report – **assessed(c, r)**
- Claim paid with an amount – **payout(c, t)**

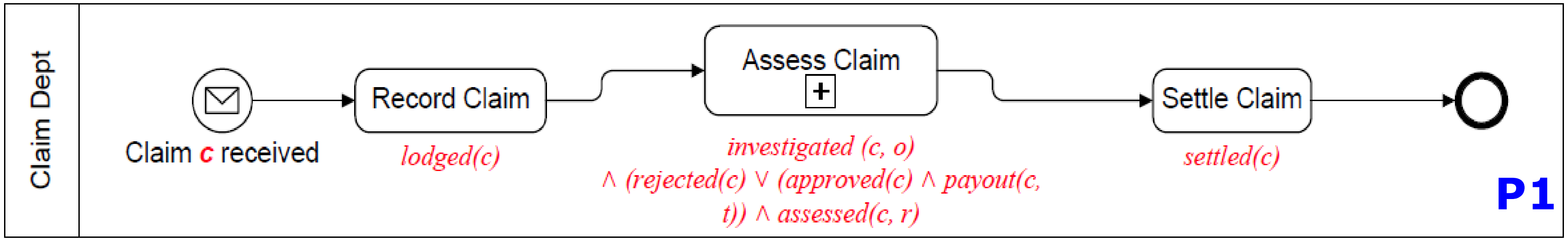
Inter-process relationships

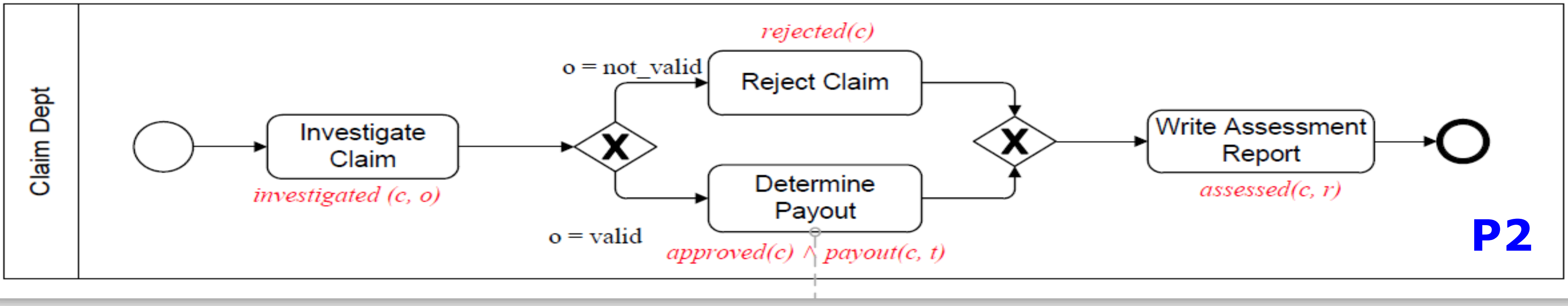
- **Part-whole**:
  - exists between two processes when one process is required by the other to fulfill some of its functionalities.
  - Specifically, the "whole" process must have an activity that represents the functionalities (in terms of cumulative effects) of the "part" process (commonly named a sub-process).



Formally, process P is part of process Q iff there exists an activity a in P such that $CE(P, a) = CE(P \uparrow^a Q, a)$ where $CE(P, a)$ is the cumulative effects of executing process P up to activity a, and $P \uparrow^a Q$ is the process model obtained by viewing Q as the sub-process expansion of activity a in P.

Inter-process relationships (cont.)

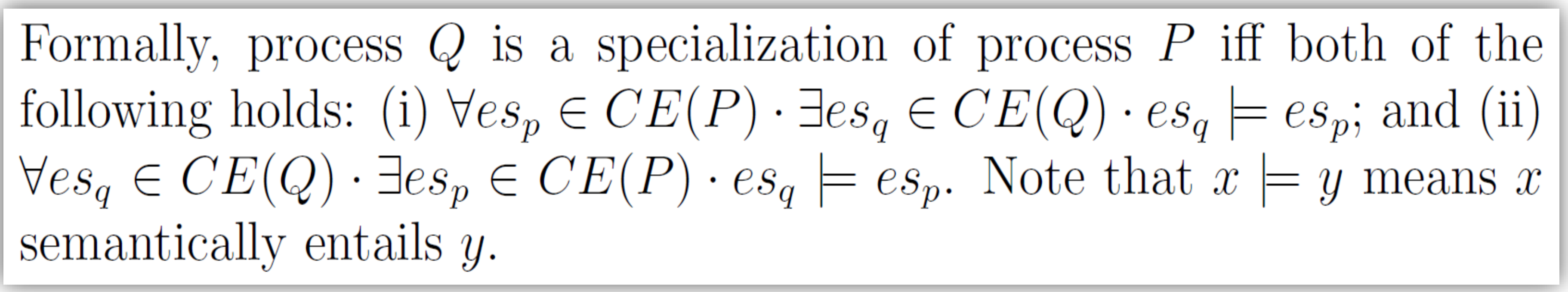




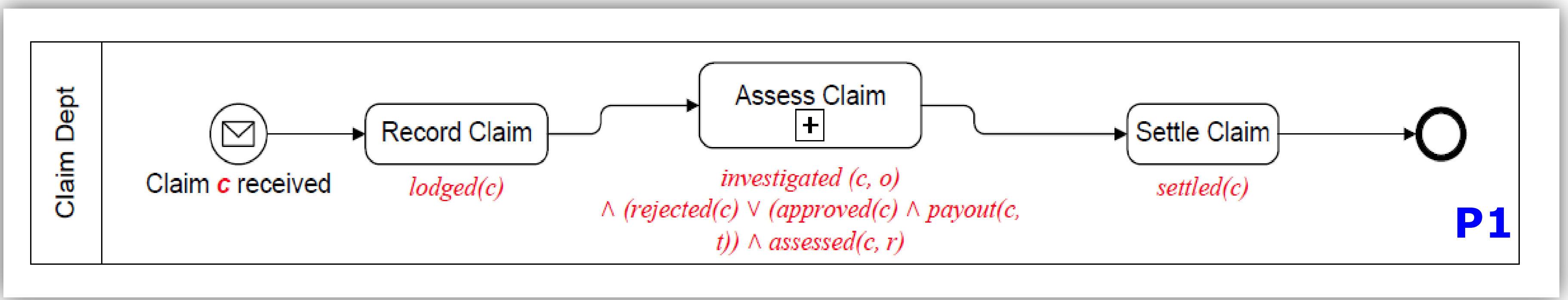
For example, the expansion of process P_2 in the sub-process activity "Assess Claim" (AC) results in process $P_1 \uparrow^{AC} P_2$, and $CE(P_1 \uparrow^{AC} P_2, AC) = \{lodged(c) \wedge investigated(c, o) \wedge (rejected(c) \vee (approved(c) \wedge payout(c, t)) \wedge assessed(c, r)\}$. Therefore, $CE(P_1, AC) = CE(P_1 \uparrow^{AC} P_2, AC)$, and thus P_2 is a part of P_1 in the part-whole relationship between the two processes.

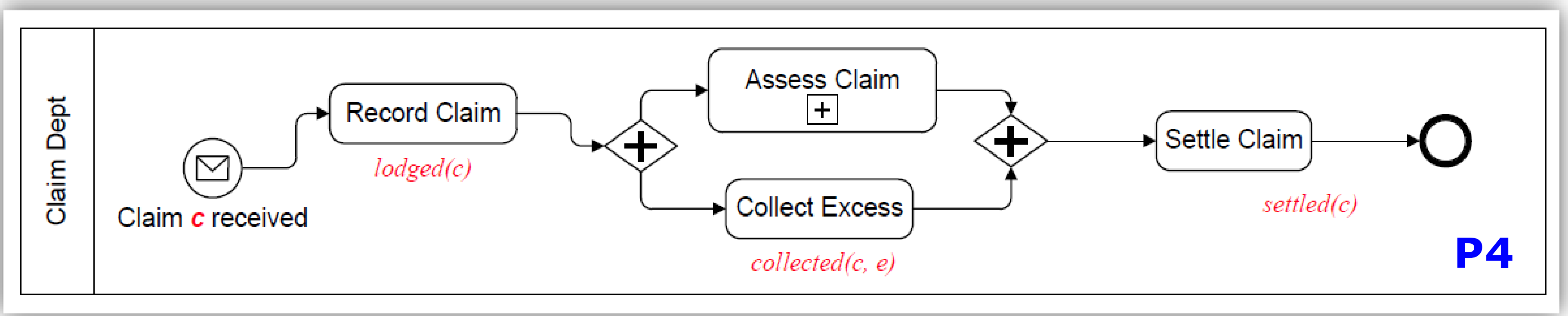
Inter-process relationships (cont.)

- **Generalization-Specialization:**
  - covers the situation when one process (i.e. the specialization) is a functional extension of the other (i.e. the generalization).
  - The specialized process not only inherits the same functionalities from its generalization but also has additional functionalities.
  - This can be realized by having some additional activities or enriching the immediate effects of the existing activities.
  - Both methods extend the indented end cumulative effects of the specialized process.


Formally, process Q is a specialization of process P iff both of the following holds: (i) $\forall es_p \in CE(P) \cdot \exists es_q \in CE(Q) \cdot es_q \models es_p$; and (ii) $\forall es_q \in CE(Q) \cdot \exists es_p \in CE(P) \cdot es_q \models es_p$. Note that $x \models y$ means x semantically entails y.

Inter-process relationships (cont.)

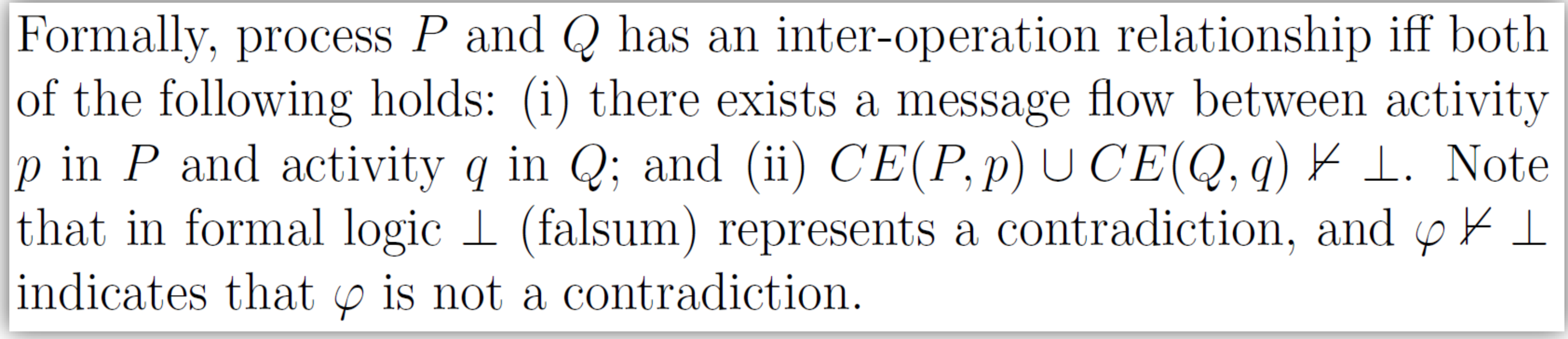




For example, the end cumulative effects of the handling claim process (P_4) is the end cumulative effects of process P_1 plus $collected(c, e)$ which is the effect of collecting the excess e associated with claim c. As a result, there is a generalization-specialization relationship between P_1 and P_4 where the former is a generalization of the latter.

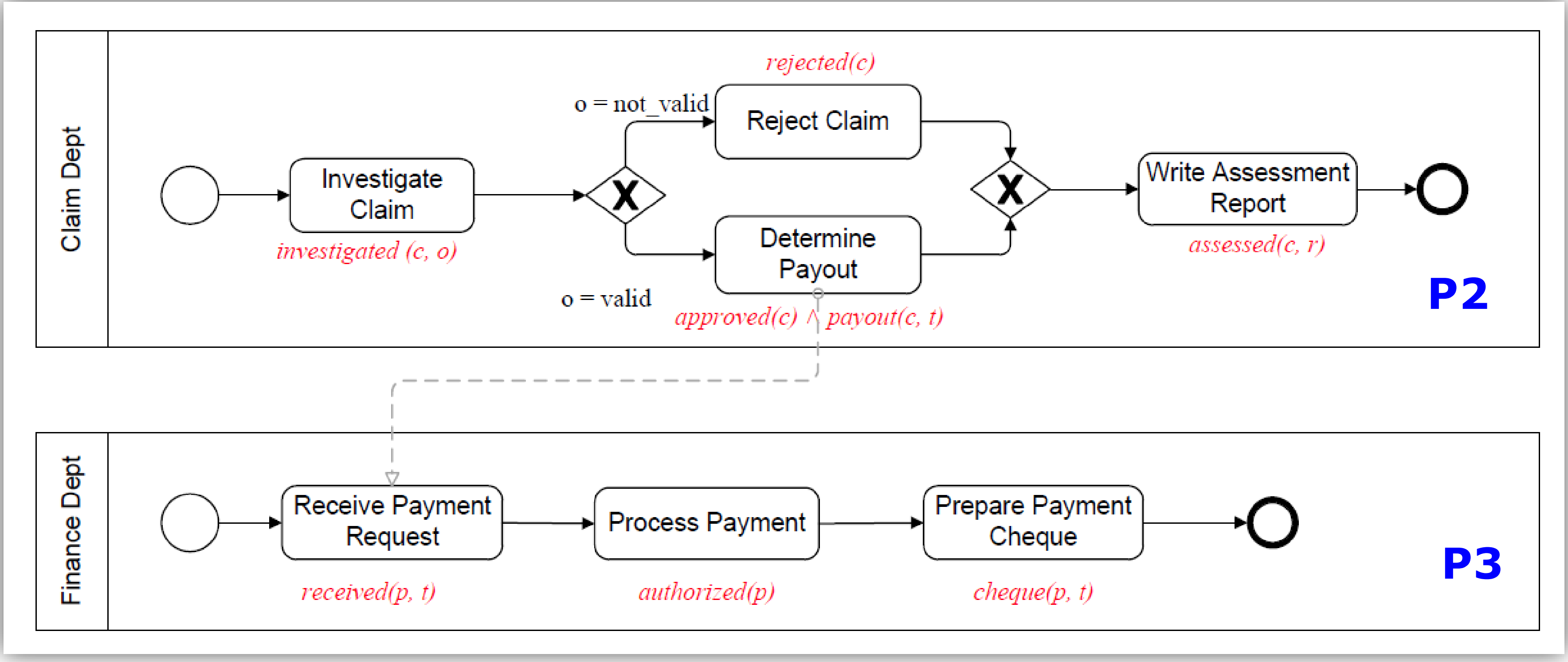
Inter-process relationships (cont.)

- **Inter-operation:**
  - exists between two processes when there is at least one message exchanged between them and there is no cumulative effect contradiction between activities that involve in the message exchanging.



Formally, process P and Q has an inter-operation relationship iff both of the following holds: (i) there exists a message flow between activity p in P and activity q in Q; and (ii) $CE(P,p) \cup CE(Q,q) \nvdash \perp$. Note that in formal logic $\perp$ (falsum) represents a contradiction, and $\varphi \nvdash \perp$ indicates that φ is not a contradiction.

Inter-process relationships (cont.)



For example, there is a message flow between activity "Determine Payout" (DP) in process P_2 and activity "Receive Payment Request" (RP) in process P_3 and there is no contradiction between $CE(P_2, DP)$ and $CE(P_3, RP)$, and thus there exists an inter-operation relationship between the two processes