

# MDPA

giovedì 29 ottobre 2020 19:19

Questi appunti sono condivisi da Stefano Vittorio Porta (Stefa168) con licenza  
<https://creativecommons.org/licenses/by-nc-nd/4.0/>.

AA 2020-21

# Lezione Introduttiva - 23 Set

giovedì 29 ottobre 2020 19:19

Obiettivo: introdurre uno studio e mettere in pratica varie tecniche per modellare concettualmente le informazioni dei sistemi informativi.

Modellazione concettuale != ontologica

Si ha un obiettivo e la modellazione punta a raggiungerlo.

La modellazione è finalizzata a creare il sistema informativo per gestire le informazioni di un dominio e facilitare varie operazioni di un'organizzazione.

È fondamentale comprendere l'obiettivo.

I dati forniscono la base per i processi che li manipolano.

Per i processi si utilizzerà il linguaggio BPMN.

Data Modelling

1. Fact oriented approach with Object-Role-Modelling
2. CSDP, una metodologia per realizzare modelli ORM
3. Mapping relazionale

Modellazione Processi

- Business Process management
- Control-flow
- Modelling with OMG
- Reti di Petri

Obiettivo finale: mostrare correlazione tra dati e processi.

# Introduzione - 23 Set

giovedì 29 ottobre 2020 19:36

Concetto: idea o astrazione inferita da esempi o istanze della realtà e di cui estraggono le proprietà e il significato.

Sono utilizzati in genere per descrivere proprietà e relazioni tra oggetti.

## Object - Extension - Instance

Part of meaning corresponding to the effective reference.

I concetti emergono da processi di astrazione e generalizzazione dall'esperienza, utilizzati dalle persone per poter descrivere e parlare di oggetti.

La concettualizzazione è una parte della realtà come percepita e organizzata da un agente, astraendo da una situazione specifica e dal vocabolario usato.

Nella concettualizzazione ci sono anche elementi spaziali e temporali che influiscono.

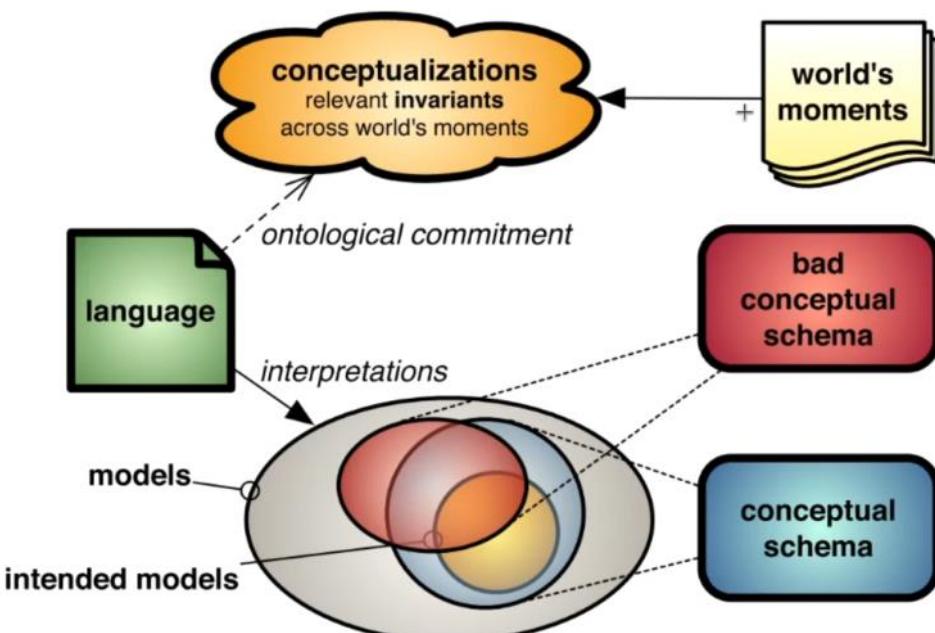
Ontologia: studio filosofico della natura e della struttura degli esseri e della realtà, di loro categorie e relazioni.

Non è presente l'agente! Coi concetti possiamo avere influenze nella definizione date dall'agente, mentre invece in un'ontologia non è così.

## -- Schema concettuale e significato inteso --

Il linguaggio impone dei vincoli sulla concettualizzazione. Dati i vincoli della concettualizzazione possiamo avere diversi modelli (interpretazioni).

Uno schema concettuale è una definizione che include anche il modello inteso.



Il sistema informativo è l'obiettivo. Dovrà poi gestire i dati per poter assistere le operazioni intraprese da un'azienda.

Funzioni di un Sistema Informativo:

- Memoria: mantenere una rappresentazione dello stato di un dominio
- Informativo: fornire informazioni dello stato di un dominio
- Attivo: eseguire azioni che cambiano lo stato di un dominio

Il SI mantiene le informazioni a livello intensionale (concetti e vincoli) ed estensionale (istanze)

L'aggiornamento del SI nella f. di memorizzazione può avvenire su richiesta (quando si modificano gli elementi o si inseriscono dati) o autonomamente quando il sistema osserva che il dominio è cambiato e deve essere aggiornato.

Un SI fornisce informazioni all'utente riguardo il dominio: di nuovo l'utente effettua richieste o vengono pubblicate in seguito a certi eventi le informazioni

Query: poste a un SI per estrarre informazioni. Devono essere poste con un linguaggio specifico, in modo che sia espressivo, comprensibile e adeguatamente complesso.

Le query possono essere estensionali quando chiedo istanze del SI; intensionale quando estraggo concetti, chiedo se valgono proprietà.

Attivo: quando il sistema delega delle operazioni al SI o quando uno stato cambia.

FUNCTIONS	+ MODES	
	On request	Autonomous
<b>Memory</b>	Change customer address.	Measure temperature.
<b>Informative</b>	Who is the nearest customer considering my current position?	Signal when the temperature is too high.
<b>Active</b>	Notify of the change all consultants working with the customer.	Turn on the heating system when the temperature is too low.

Un sistema informativo necessita di conoscere il dominio e le funzioni che devono essere eseguite. Queste sono date dal modello concettuale, uno schema concettuale e dei vincoli.

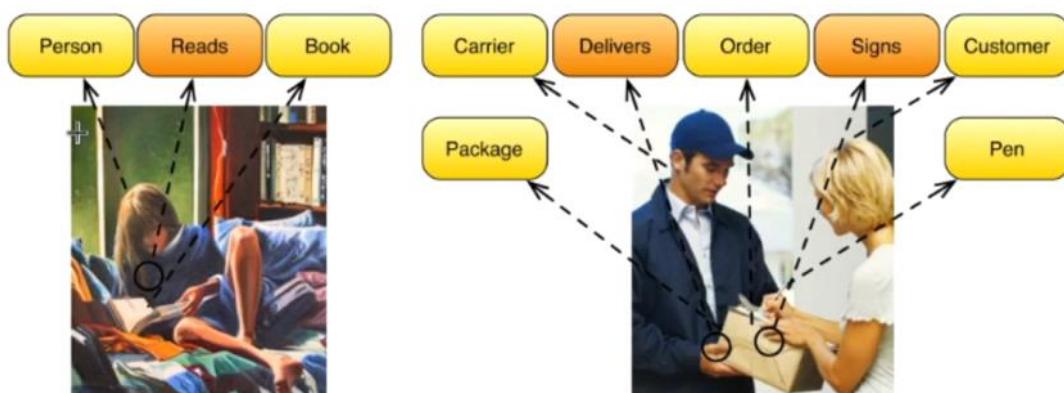
Il dominio evolve nel tempo e la rappresentazione cambia. Ci sono elementi statici e dinamici.

L'evoluzione del sistema informativo deve essere consistente con la realtà.

Ci sono anche elementi che non sono rappresentati dal sistema (età) visto che sono derivabili. Ci sono quindi informazioni elementari e informazioni derivate.

In generale per rappresentare un sistema informativo servono:

- Entità: concetti le cui istanze sono individuali e identificabili
- Relazioni: tuple tra due o più entità (in genere in FOL, fatti veri)



Lo stato consiste in proprietà interessanti che ubbidiscono a certi vincoli.

Possiamo individuare due tipi di proprietà:

- primitive/elementari: non divisibili per non perdere informazioni
- Derivate: ottenibili tramite inferenza logica

Vincoli:

- Statici: costanti dei dati
- Temporali
- Dinamici

#### Examples of static constraints

- Functional dependencies (each employee has a fixed salary).
- (Primary) keys (each employee is identified by its SSN).
- Multiplicity constraints (a car has exactly four wheels, each square can have at most one chess piece).

#### Examples of dynamic constraints

- An accepted order cannot be rejected afterwards.
- To access the cart, the user must successfully log-in.
- A chess piece can be moved in a square if the move is legal w.r.t. the piece type and does not lead to put the own king in check.
- When the auction's deadline expires, the bidder with the highest bid must be declared winner of the auction.
- When the customer closes an order, the warehouse must either refuse it or inform the customer about the expected delivery date.

Nota: vincoli statici possono imporre vincoli dinamici

Mondo: chiuso/aperto

In un mondo chiuso ogni dato non conosciuto implica falsità.

In un mondo aperto non si sa e basta.

Un modello concettuale è

#### Conceptual Model = Conceptual Schema + Information Base

- **Conceptual schema**: blueprint of the domain inside the IS.
  - Orders, employees, deliveries, cancelation, customer, gold customer, gift, payment, payment transaction ...
- **Information base** (or **conceptual database**): blueprint of a specific state of the domain inside the IS.
  - Order o-123-bzFGH, employee Mario Rossi, delivery of o-123-bzFGH via airmail,...

Sviluppare un SI richiede la definizione dello schema/modello concettuale associato.

Lo schema strutturale è invece lo studio di concetti e delle loro relazioni: quali generi di dati sono salvati nel DB, quali vincoli applicare ai dati e quali dati sono derivabili.

Lo schema comportamentale è una specifica dei cambiamenti accettabili, generalmente rappresentati da eventi del dominio risultanti dall'esecuzione di azioni o compiti.

- Prospettiva orientata ai processi: per indicare quali attività o processi sono effettuati per comprendere il modo in cui una particolare azienda opera.
- Prospettiva orientata al comportamento: come gli eventi avviano operazioni

Quindi possiamo definire lo schema concettuale anche come

Conceptual schema = fact types + constraints + derivation rules

- **Fact types:** kinds of facts used by the IS to describe a state of the domain.
  - **Object types** (student, employee, country, ...).
  - **References** to objects by value in the information base (matriculation id, SSN, country-code, ...).
  - **Relationship types** (studies in, works at, includes, ...).
- **(Integrity) Constraints:** restrict the allowed facts used by the IS to describe a state of the domain.
  - Static constraints apply to every state of the domain (every country has exactly one number representing its current population).
- **Derivation rules:** how to obtain derived facts from primitive facts.
  - Age of a person from her date of birth, ancestor relationship from parent of,...

L'information base è un'astrazione che rappresenta i fatti veri memorizzati nel sistema informativo.

Eventi e aggiornamenti: generati da un utente che agisce sul sistema e che fa cambiare il SI da uno stato ad un altro. Il cambiamento è determinato da un insieme differente di stati, richiesto da un evento di un attore.

Transazione: quando un evento è composto da più aggiornamenti, o vengono completati tutti oppure no.

Integrità: le configurazioni che il SI può assumere devono essere accettabili. In un mondo perfetto l'integrità dovrebbe essere totale. Vogliamo sia la validità, sia la completezza. Qualunque evento quindi dovrebbe portare il SI da uno stato ad un altro valido e completo.

Un vincolo di integrità è rotto quando mancano dati da cogliere o si attuano aggiornamenti non validi.

Consistenza dei fatti: la BI deve sempre soddisfare i vincoli di integrità.

Cosa si fa se un fatto non va bene?

- Lo rifiuto
- La transazione è abortita

Attenzione: non tutti gli errori possono essere rilevati, ma bisogna cercare il più possibile.

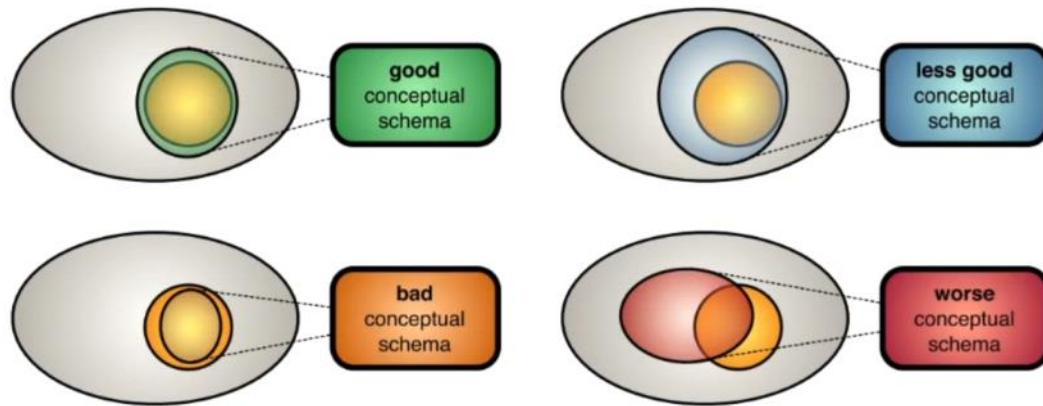
Consistenza logica: i vincoli di integrità devono essere soddisfacibili.

Soddisfacibile: deve esistere una BI che soddisfa i vincoli.

- **Satisfiable:** there must exists one information base  $\mathcal{I}$  that satisfies the constraints.
- **Strongly satisfiable:**  $\mathcal{I}$  is nonempty and finite.
- Consider the following conceptual schema:
  1. Everybody is supervised by somebody.
  2. Nobody supervises himself.
  3. If  $x$  is supervised by  $y$  and  $y$  is supervised by  $z$ , then  $x$  is supervised by  $z$ .

La consistenza è importante perché da un sistema inconsistente possiamo derivare qualsiasi affermazione (principio dell'esplosione)

## Validazione



Un buon schema concettuale deve catturare bene i modelli veri e non quelli falsi.

Un Business Process consiste in attività eseguite in coordinazione in un ambiente organizzativo e tecnico.

- IS with explicit BP support:
  - **Process-aware IS:** stores and executes BPs that manipulate the information base according to the dynamic constraints.
- IS without explicit BP support:
  - Processes hidden into software components that manipulate the information base.
  - Dynamic constraints must be reflected in the program.
  - IS understands the manipulation in terms of domain events and compound transactions.
  - The structural schema must find a counterpart in the program.
    - Transient information model.

## Strato di presentazione

Manages the interaction with external users.

- **External schema:** a view of the conceptual schema in terms of concepts and operations accessible to a particular group of users. +
  - What information can be accessed: read, access, deletion.
  - How this information must be presented to the users.
- **External database:** virtual database representing the state of the domain in terms of the external schema.
- **External processor:** exchange messages with users and enforces the prescriptions of the external schema.
  - Defines a language for communicating with the users.
  - Acts as a bridge/translator between users and the conceptual information processor.
- In general, multiple external databases/schemas/processors to deal with different groups of users.

## Strati fisici e logici

Manage the internal manipulation of data and their effective physical storage.

- **Internal (logical) schema:** expresses the conceptual schema in terms of the abstract data structures and operations supported by a concrete logical model.
  - For structural information: relational, object-oriented, ...
  - For behavioral information: executable process/program.
- **Internal database:** physical, internal storage for the actual data.
  - For relational data model: managed by a DBMS.
  - Conforms to the logical schema, realized as a physical schema (e.g., written in the specific DBMS language).
  - Focus on efficiency and conciseness.
- **Internal processor:** receives the commands from the information processor and executes them over the internal database.

#### Strato concettuale

Governs the IS at the conceptual level.

- +
  - It is completely independent from user interfaces, storage and data access techniques: **stability**.
  - **Conceptual information processor:** mediates the communication between the external users and the internal database.
    - Ensures factual consistency.
    - Transforms domain events into compound transactions over the actual data.
  - Remember: the **information base** is virtual!

# Sviluppo di sistemi Informativi - 29 Set

domenica 1 novembre 2020 12:05

L'ingegnerizzazione di un SI è un processo di problem solving:

1. **Analysis** (includes conceptual modeling): what the IS is about, what are the requirements.
2. Design (includes logical modeling): how to accomplish the requirements.
3. Implementation: coding of the design under specific architectural/technological choices.
4. Testing: check if the implementation works and meets the requirements.
5. Maintenance: assist users after release, keep the IS working.

Si possono usare le solite metodologie, a cascata, iterativa, XP...

Approccio del Modello di Dominio: *Divide et impera*. Le varie parti indipendentemente vengono sviluppate e poi si collega.

Più nel dettaglio,

1. Feasibility study: is the idea implementable?
2. Requirements analysis: what should the system do?
3. Conceptual design - data, processes: what is the conceptual schema modeling the UoD?
4. Logical design - data, processes: how can the conceptual schema be translated into a logical schema?
5. Basic physical design - data and processes: how can the logical schema be represented in a concrete management system?
6. Basic external design - data and processes: which information can be accessed by users, and how?
7. Prototyping: how does the IS look like?
8. Completion of design.
9. Implementation of production version.
10. Testing and validation: does the IS work well and satisfy the requirements?
11. Release of software, documentation, training.
12. Maintenance.

L'analisi dei requisiti comprende anche l'analisi dei rischi, che può influenzare la modifica del progetto.

Il linguaggio influenza molto sulla precisione con cui catturiamo il significato inteso.

Criteri del linguaggio che bisogna utilizzare:

- Espressività: più è ricco e più possiamo definire in maniera articolata. Troppo ricco e possiamo introdurre errori facilmente. (esempio: architetture RISC e CISC)  
Il linguaggio deve poter descrivere completamente l'intento. Questo è un goal.  
La correttezza invece è un requisito.
- Chiarezza: deve essere facile da capire ed utilizzare.
- Semantica: devono essere utilizzati solo i concetti rilevanti dal punto di vista della modellazione.

Trattabilità/espressività di un linguaggio: più è espressivo e più è possibile fare computazioni

Parsimonia/convenienza: meno concetti ma modelli più complessi da comprendere

Il linguaggio scelto è ORM. Non è vicino all'OO o al relazionale, ma si focalizza sugli elementi di

astrazione essenziali, per rendere il progetto facilmente mantenibile.

L'E-R si focalizza su entità, relazioni ed attributi. Esistono anche le indicazioni di molteplicità. L'UML (unified modelling language) è lo standard per la progettazione di software OO.

ORM



- Punta a minimizzare la ridondanza
- Gli oggetti hanno dei ruoli.
- Si possono trattare ruoli n-ari
- I vincoli esprimibili sono ricchi e sussume UML ed E-R
- Non ha attributi

Gli attributi non sono presenti perché ci possono essere errori sull'assegnazione delle responsabilità di essi.

Il modello è più semplice ma diventa più grande.

# Linguaggio ORM S1 - 30 Set

lunedì 2 novembre 2020 13:31

Si segue la metodologia Conceptual Schema Design Procedure (CSDP)

1. Transform familiar examples into elementary facts.
2. Draw the fact types, and apply a population check.
3. Check for entity types to be combined, and note any arithmetic derivations.  
+
4. Add uniqueness constraints, and check the arity of fact types.
5. Add mandatory role constraints, and check for logical derivations.
6. Add value, set-comparison, and subtyping constraints.
7. Add further constraints, do final checks.

Ad ogni passo vengono introdotti dei costrutti del linguaggio per arricchire lo schema.

## Passo 1

Trasformazione di esempi familiari in fatti elementari.

Critical step: **understanding** the UoD.

Goal: isolate relevant information to be represented in the IS.

- Every relevant piece of information: must be elementary or derivable.
- → Isolate each **elementary fact**.
  - Cannot be split into smaller units of information.
  - Simple assertion, atomic proposition about the UoD.
  - **Epistemic commitment**: people act as they believed the fact to be true.

Ogni informazione deve essere elementare o derivabile da altri fatti elementari. Un fatto elementare è l'unità più piccola informativa, che se semplificassi ancora perderebbe di significato. Quello che si crede essere vero.

Bisogna analizzare tutte le informazioni definite. Le sorgenti delle informazioni sono i report, domande ai clienti, scontrini, mappe, figure, diagrammi, video... che descrivono quello che desideriamo rappresentare.

Dai documenti dobbiamo poi estrarre quello che è il dominio del discorso. Saranno anche gli elementi utilizzati per verificare se gli elementi caratterizzanti sono stati correttamente rappresentati.

Database per le ricevute? Allora guardo le ricevute.

SI per le risposte di un questionario? Allora vedo i questionari e come vengono riempiti. Bisogna individuare gli elementi essenziali del dominio che vogliamo modellare.

- Questions: what types of information do we want from the system? Are entities well-identified? Can the facts be split into smaller units without losing information?
- Answers: by talking with domain experts about examples ("familiar information examples").
  - Reports, input forms, sample queries, maps, pictures, drawings, charts, videos, interviews, documents, books, observation, ...
- **Data use cases:** talk about processes and requirements, but to understand the data. *Then* design the processes.

Quello che dobbiamo costruire è un Data Use Case.

Fatti elementari: proprietà oppure proprietà dell'oggetto in quanto partecipi alla creazione di un altro oggetto.

- *Anna fuma*
- *Anna impiega Bob* (se la semplifico oltre non ha senso: *anna impiega ... chi?*)
- *Bob è impiegato da Anna*
- *Se Anna impiega Bob, allora Bob prende un salario.*  
(è una relazione elementare? Bob prende un salario lo è!)
- Se qualcuno è impiegato, allora prende un salario.  
(più complessa, viene creata una causalità.)
- Anna impiega Bob e John.  
(Bob e John sono necessariamente impiegati? Potremmo interpretarla con due volte che anna impiega...)
- Anna e Bob aprono una richiesta di prestito (sono due richieste o una?)
- Bob non fuma. (adesso o in genere? È da disambiguare. Si intendeva Bob è un non fumatore?)

Oggetti di base:

- **Value:** has self-identifying reference (30,  $\pi$ , 'Lee', 'E301').
  - Rigid.
  - Strings and numbers.
- **Entity/Object:** referenced by a *definite description* (Lee, E301).
  - Typically changes with time.
  - Tangible (this computer) vs abstract (this lesson).
  - **Referenced** by a *rigid* value: use/mention distinction.
    - Lee is located in E301 vs 'Lee' is located in 'E301'.
  - Just a value is not sufficient → referential ambiguity.
- Valori ("Anna", "Bob", "E301...")
  - sono immutabili, in genere stringhe e numeri che si auto-rappresentano.
- Entità e Oggetti  
Sono elementi referenziati da una descrizione.  
E301 non è più una stringa, ma un oggetto! La stringa non cambia nel tempo, mentre l'entità sì.

Come referenziare entità:

1. Value('Lee')
2. Explicit entity type (the Person 'Lee')
3. Reference mode: Il modo con cui il valore referenzia il tipo di entità (the Person with surname 'Lee')

## Definite description

1. **value** ('Lee')
2. + **explicit entity type** (the Person 'Lee')...
3. + **reference mode**: the manner in which the value refers to the entity type (the Person with surname 'Lee').

Verbalizzazione compatta:

Person(.surname) 'Lee' is located in Room (.code) 'E301'.

Ruoli

Se identifichiamo i fatti elementari come se fossero delle affermazioni vere, allora i ruoli sono come i "buchi" del predicato stesso: --- is located in ---. I buchi sono oggetti che devono essere inseriti.

La proposizione che otteniamo sostituendo i ruoli non deve essere esprimibile con una congiunzione di fatti elementari.

1. Raccogliere tutte le relazioni significative
2. Analizzarle con gli esperti del dominio usando la euristica del telefono  
Identificare i sinonimi, scegliere i termini preferiti e scrivere un glossario
3. Elaborare le informazioni verbalizzate individuando i valori differenti  
Scrivere esempi, identificare vincoli nascosti (A e B e C ma B e C sono indipendenti quindi A e B, A e C)  
Riscrivere le informazioni usando descrizioni definite per le entità e identificando i ruoli inversi  
**Elementary facts about the system as-is**
4. Fare lo stesso coi requisiti dei dati  
**Elementary facts about the system to-be**

Tute Group	Time	Room	Student Nr	Student Name
A	Mon. 3 p.m.	CS-718	302156	Bloggs FB
			180064	Fletcher JB
			278155	Jackson M
B1	Tue. 2 p.m.	E-B18	266010	Anderson AB
			348112	Bloggs FB
...	...	...	...	...

Typical verbalization by domain expert:

- Student 302156 belongs to group A and is named 'Bloggs FB'.
- Tute group A meets at 3 p.m. Monday in Room CS-718.

Notare come non vengono utilizzate tutte le informazioni fornite. Sono state scritte delle frasi elementari che descrivono le informazioni necessarie.

Student 302156 belongs to group A and is named 'Bloggs FB'.

+

- Name and surname together.
- Student name and nr in the same row refer to the same student.
- Student has only one number but could share the name with others.
  - Student number is a good identifier, student name is not.

→ Student (.nr) 302156 *has* StudentName 'Bloggs FB'.

- StudentName is a value type: no reference scheme.

→ Student (.nr) 302156 *belongs to* Tutegroup (.code) 'A'.

- **Inverse:** Tutegroup (.code) 'A' *involves* Student (.nr) 302156.
- ... (Stud.) *belongs to* ... (TuteG.)  $\leftrightarrow$  ... (TuteG.) *involves* ... (Stud.)
  - $\neq$  surface structure, = deep structure.
  - One primary (mandatory), the inverse optional.

Alcuni fatti sono indivisibili

Tute group A meets at 3 p.m. Monday in Room CS-718.

↓

TuteGroup(.code) 'A' *meets at* Time(.dhcode) 'Mon. 3 p.m.' *in* Room(.code) 'CS-718'.

- **Hp:** TuteGroups meet more than once a week.
  - Further questions (Always in the same room? Suppose not)
  - The fact is inseparable.
  - Hence elementary → a ternary predicate!
  - Need to complete the sample data with additional significant cases:
    - TuteGroup(.code) 'A' *meets at* Time(.dhcode) 'Tue. 4 p.m.' *in* Room(.code) 'CS-513'.
  - Separation → information loss!

Tute group A meets at 3 p.m. Monday in Room CS-718.

↓

TuteGroup(.code) 'A' *meets at* Time(.dhcode) 'Mon. 3 p.m.' *in* Room(.code) 'CS-718'.

- Sample questions:
  1. Does TuteGroup(.code) 'A' always meet in Room(.code) 'CS-718'?
  2. Does this hold for all TuteGroups?
  3. Do TuteGroups meet only once a week? (Note: (3) → (2)).

# Fact Type e verifica della Popolazione S2.3 - 6 Ott

martedì 3 novembre 2020 11:13

## Passo 2:

disegnare i fatti e applicare una verifica di popolazione.

1. Draw an **instance diagram** from the factual information obtained so far.
2. Generalize the instance diagram to a **conceptual schema diagram (structural schema)**.
3. Validate the correctness of the conceptual schema diagram with **sample population**  
→ **conceptual model** or **conceptual knowledge base**.

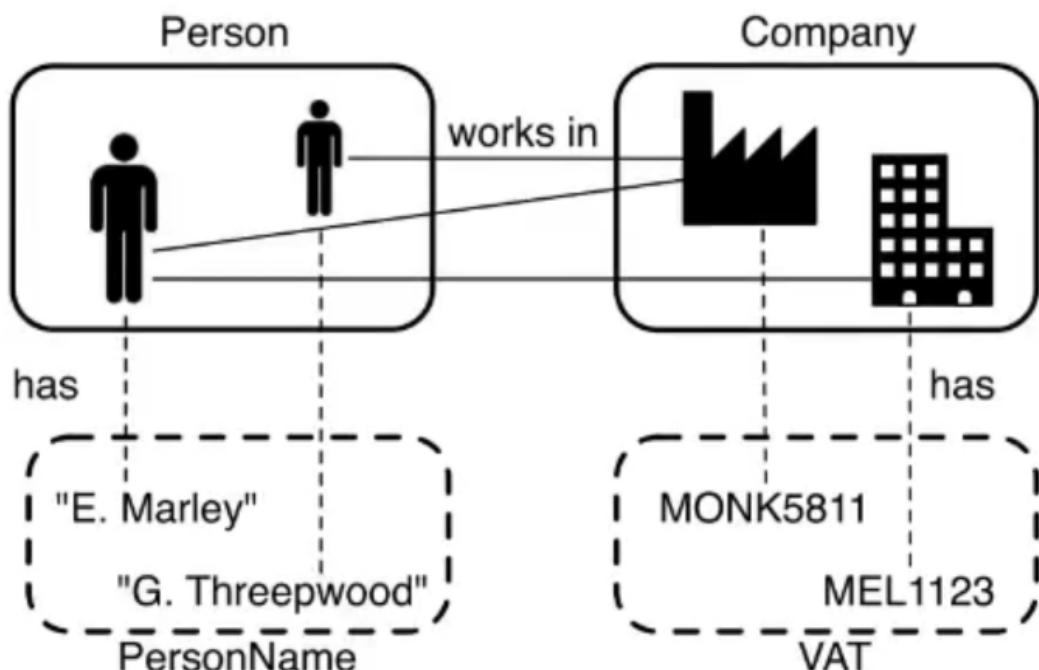
La validazione comprende l'utilizzo di query concettuali sullo schema.

Person	Company
G. Threepwood	MON5811
E. Marley	MEL1123
E. Marley	MON5811

Ricordiamo le precedenti verbalizzazioni:

- The Person named 'G. Threepwood' works in/employs + Company with VAT 'MON5811'.
- Person (.Name) 'E. Marley' works in/employs Company (VAT) 'MON5811'.

Disegniamo l'instance Diagram:

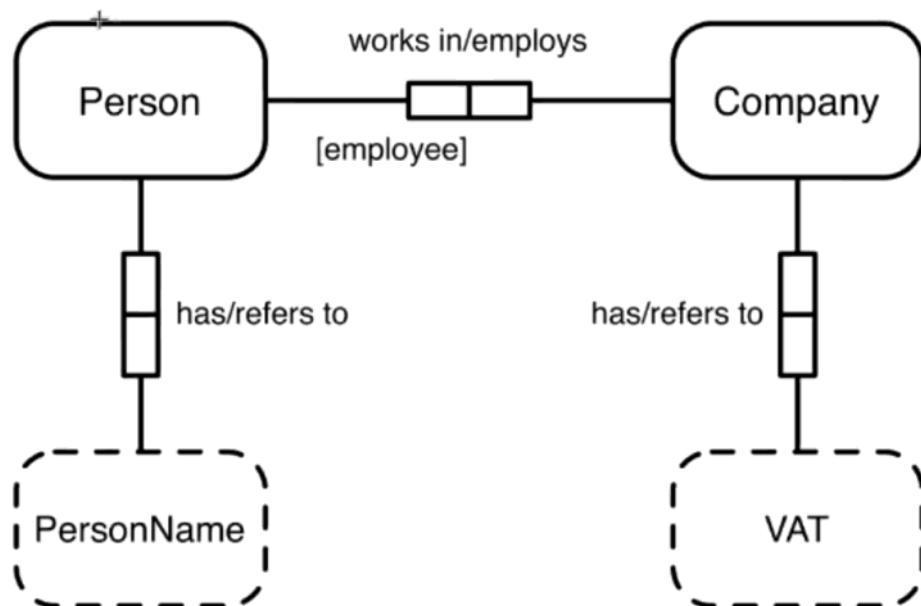


Abbiamo individuato gli:

- Object Type (rettangolo bordi arrotondati, con contorno continuo e con un nome)
- I ruoli (il "works in") è un quadrato solido. Il nome opzionale è indicato tra [ ]
- Predicato di arietà n: n ruoli contigui. Deve esserci una lettura almeno.

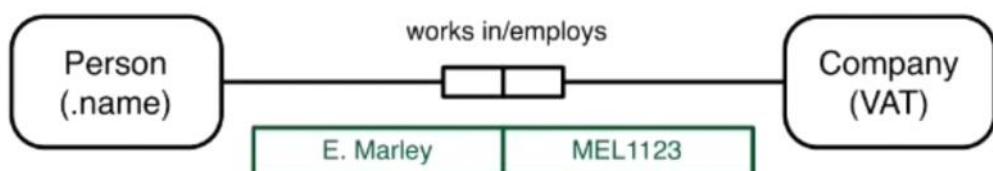
Per convenzione la lettura inversa è separata da quella principale da una '/'. La lettura inversa è opzionale. Si usa optionalmente un triangolino.

- Conventions:
  - Binary roles: optional inverse reading  
(separated from the mandatory one by '/').
  - N-ary roles ( $n > 2$ ): ellipsis '...' to represent object holes.
- How many (alias) readings for n-ary roles?
  - In general?  $n!$  (permutations)
  - Displayed? 1
  - To easily query the schema?  $n$
- Guideline: define inverse reading for binary role, alias readings for n-ary roles only when needed.



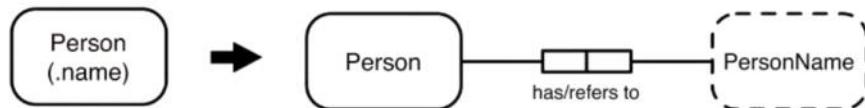
#### Tipi di Relazioni

- Elementary fact type: relationship between entities.
- Reference: relationship between entities and values.
  - E.g.: The VAT number 'MEL1123' refers to some Company.
  - Also called **existential fact** (there exists a Company that has VAT number 'MEL1123').
  - Typically used for **preferred identification scheme**.
    - **1:1 pattern**: every Company has a unique VAT number, every VAT number refers to a single Company.
    - Compact representation using parentheses inside the entity type rounded rectangle.
    - Fact tables can mention the referred values in place of the corresponding entity.

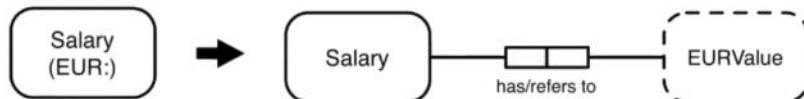


#### Tipi di Reference Mode:

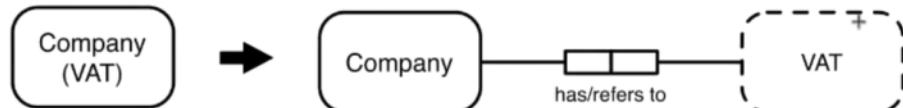
- **Popular**: predefined typical reference modes.
  - Name, code, title, nr, #, id.
  - Dot notation: Object\_type(.ref\_mode).
  - Conversion: Object\_type(.ref\_mode) → Object\_typeRef\_mode.



- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
- **General**: other reference mode types.
- **Measurement (unit-based)**: built-in (extensible) list of physical and monetary units.
  - Cm, m, kg, mile, USD, EUR, ...
  - Colon notation: Object\_type(ref\_mode:) or Object\_type(ref\_mode:unit\_type) (unit type: mass, money, ...).
  - Conversion: Object\_type(ref\_mode:) → ref\_modeValue.

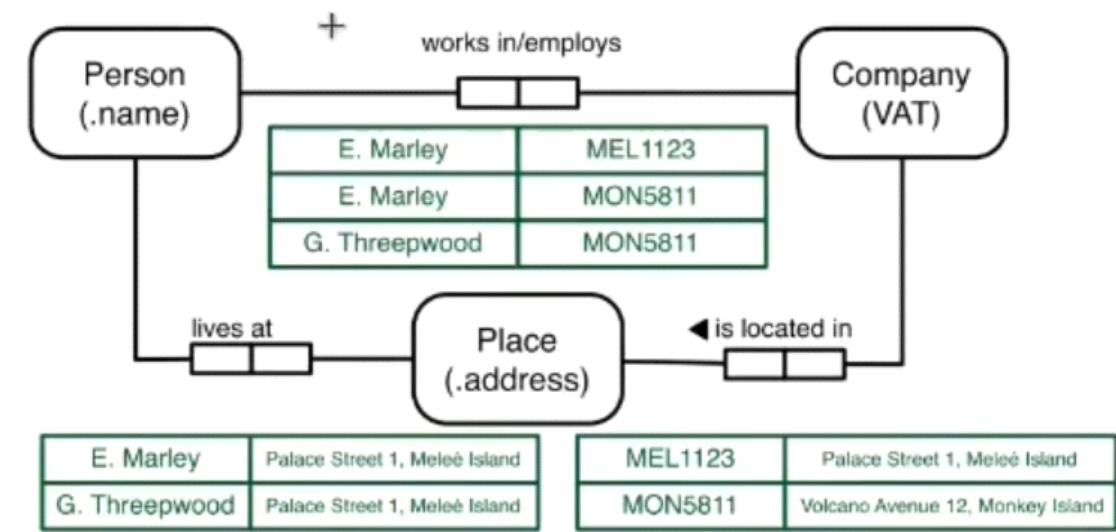


- **General**: other reference mode types.
  - Examples: VAT, SSN, ISBN, URL, ...
  - Simple notation: Object\_type(ref\_mode)
  - Conversion: Object\_type(ref\_mode) → ref\_mode.



Allo schema concettuale posso associare una fact table di fatti originali.

- **Fact table**: table with (*original*) instances of fact types.
  - For relationships: columns aligned to roles.
  - Values of reference modes identify entities.
- Why? Supports the validation of the conceptual schema diagram.
  - Identification of nonsensical diagrams.
  - Validation of constraints.
- Best practice: verbalize at least one fact from each fact table.

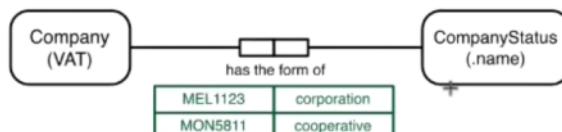


Le relazioni possono essere anche 1-arie

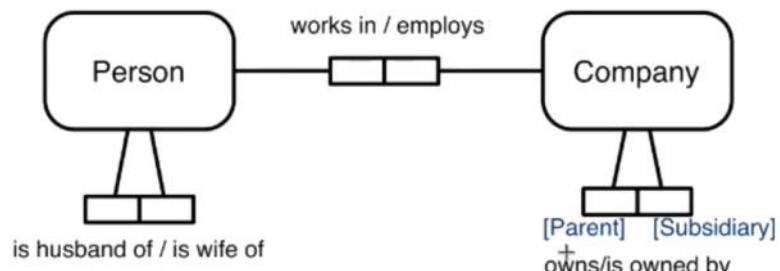
- Consider the possible types of companies: corporation, cooperative, ...
- Verbalization: Company (VAT) 'MEL1123' is a corporation.
- **Unary fact type:** only **one role** (being a corporation).



- Schema transformation: similar unaries can be factorized in a single binary.
  - "Status" object type.
  - Binary relationship between the object type and the "status" entity type.
  - Each unary becomes a value for the "status" object type.



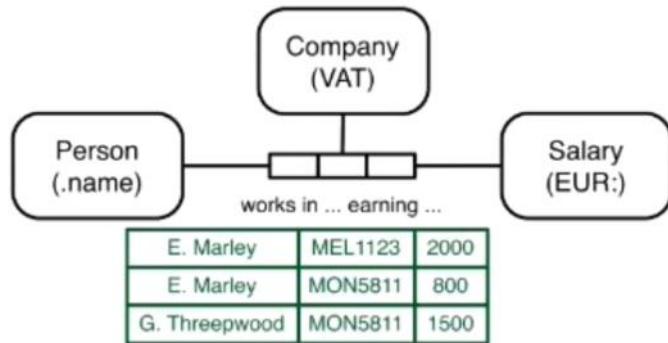
- **Heterogeneous** fact type: involves distinct object types.
- **Homogeneous** fact type: all roles played by the same object type.
  - Binary homogeneous fact type: **ring** fact type.



I fatti eterogenei coinvolgono oggetti distinti. Se tutti i ruoli sono giocati dallo stesso oggetto, allora è omogeneo.

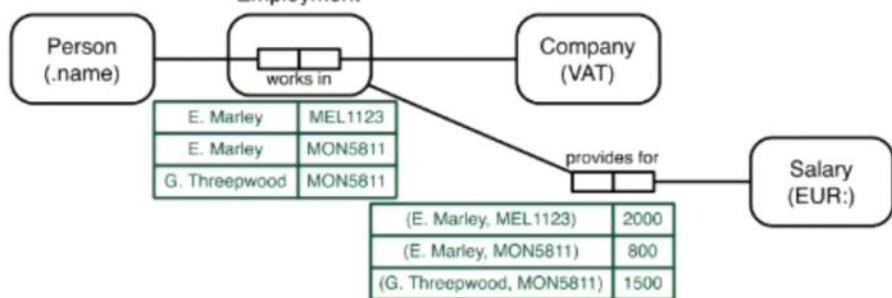
Un'ultima cosa che si può fare è la reificazione/annidamento: si tratta una relazione tra oggetti come un oggetto a sé stante.

## Flattened



(questa relazione è elementare)

## "Employment"



## Nested

L'impiego fornisce un salario.

Le due notazioni sono equivalenti solo quando il ruolo della relazione reificata è obbligatorio.

## Passo 3

È necessario verificare che gli entity type non siano combinabili.

È un passo di raffinamento dei passi 1 e 2.

Refine the conceptual schema diagram answering to:

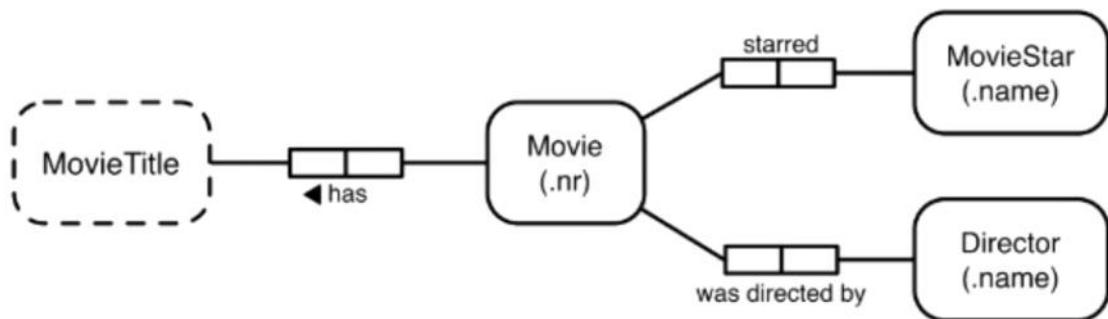
1. Can the same entity belong to two entity types?
2. Can entries of two different types be meaningfully compared? Do they have the same unit/dimension?
3. Is the same kind of information recorded for different entity types, and will you ever need to list the entities together for this information?
4. Is a fact type arithmetically<sup>1</sup>derivable from others?

Two possible actions:

- Combination of entities type in a unique type;
- Derivation rule to connect different (related) fact types.

- UoD is partitioned into exclusive and exhaustive slices:
  - Values;
  - Entities.
- Entities are again partitioned into **primitive entity types**: top-level entities never overlap.
- Top-level entity: not subtype of another entity.
- **Subtyping**: classification of objects into a more specific type.
  - Will be discussed later on in the course.
  - If object type *A* is subtype of object type *B*, then every instance of *A* is also instance of *B* (set inclusion).
  - Represented by a solid arrow in ORM notation.
  - Subtypes could overlap!
- Top-level values could overlap.
  - 'Indiana' is the name of a US state and the first name of a fictional character.
- Subtyping of values is rarely used.

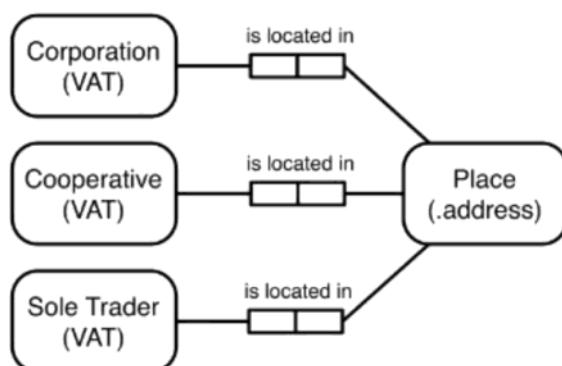
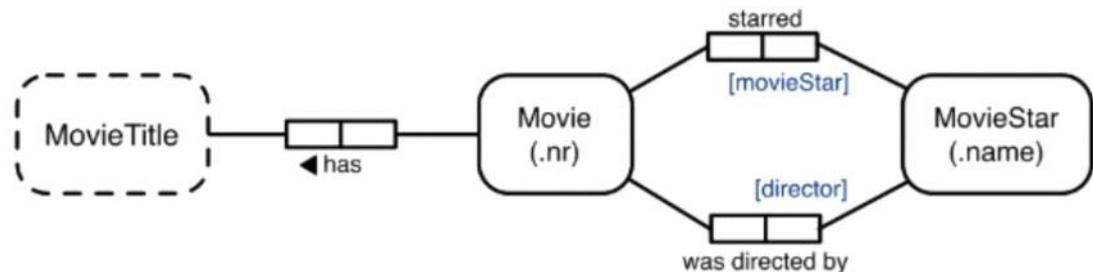
Esempio:



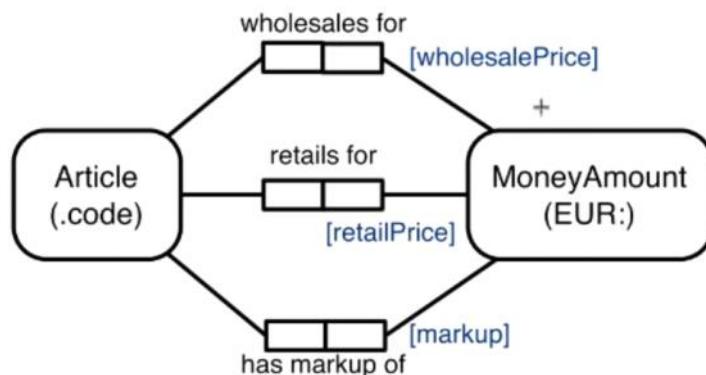
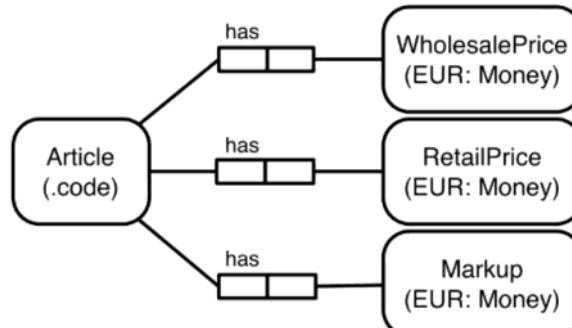
Vogliamo modellare un film identificato da un numero. Ha un titolo. Ha diverse star e alcuni direttore.

Le star e i Registi potrebbero essere elementi comuni? Attualmente non è possibile rappresentare la caratteristica. Scrivere in questo modo indichiamo come **NON SOVRAPPOSTE** le due categorie e non ci potrebbe essere una persona sia Star sia Direttore.

Soluzione:

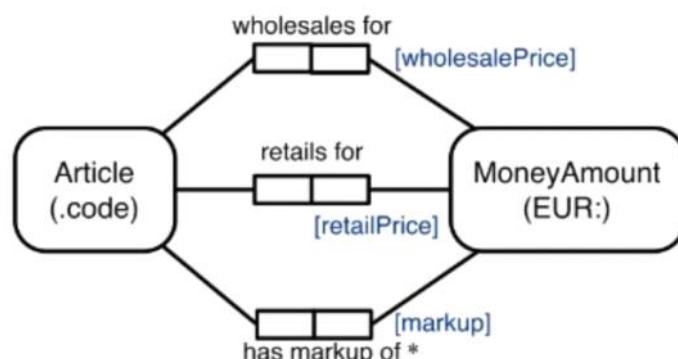


Il VAT compare in tutte e 3. Potrebbe esserci la necessità di fare delle query in funzione del CF. Come faccio a cercarla, se non so in quale delle 3 si trovi?

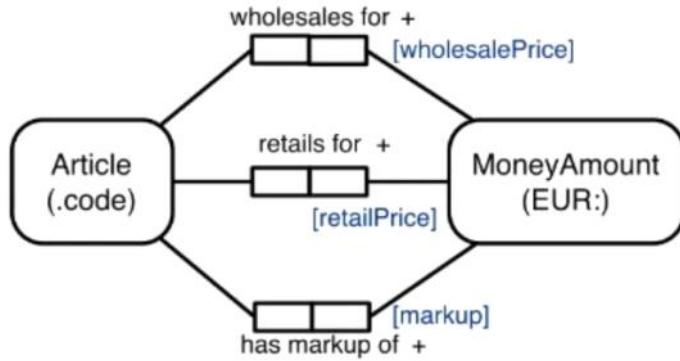


Markup = retail - wholesale.

- **Derived type:** type completely determined by other types. They must obey to a *constraint*.
- May be conceptually relevant to keep derived types in the conceptual diagram.
- Two decoration symbols for derived fact types:
  1. **derived** (\*) vs **semi-derived** (+);
  2. **derived-on-query** vs **derived-on-update** (\*\*).
- **Controlled textual annotation** to represent the derivation constraint.



Se fornisco invece le regole per tutti, possono diventare semiderivabili



$$\begin{aligned}
 \text{Markup} &= \text{retailPrice} - \text{wholesalePrice} \\
 \text{retailPrice} &= \text{markup} + \text{wholesalePrice} \\
 \text{wholesalePrice} &= \text{retailPrice} - \text{markup}
 \end{aligned}$$

- **Context** of the constraint.
  - Globally identified in the constraint (e.g., Article).
  - Locally identified: dot notation (e.g., Article.markup) vs of-notation (e.g., markup of Article).
- **Attribute** style: uses role names.

#### Example (attribute style)

```
for each Article, markup = retailPrice - wholesalePrice
```

- **Relational** style: uses predicate readings.

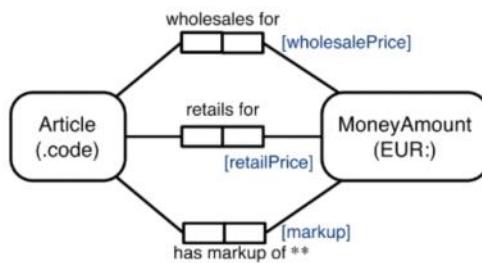
#### Example (relational style)

```

Article has markup of MoneyAmount iff
  Article retails for MoneyAmount1 and
  Article wholesales for MoneyAmount2 and
  MoneyAmount = MoneyAmount1 - MoneyAmount2
  +

```

- **Derived-on-query** (lazy evaluation): derived information is computed on request.
  - Typically part of a view of the conceptual model.
- **Derived-on-update** (eager evaluation): derived information is stored.
  - Another \* added.
  - Every time one of the primitive facts is updated, the derived fact must be updated too.
  - In databases: trigger or computed column.



# CSDP S4 - 7 Ott

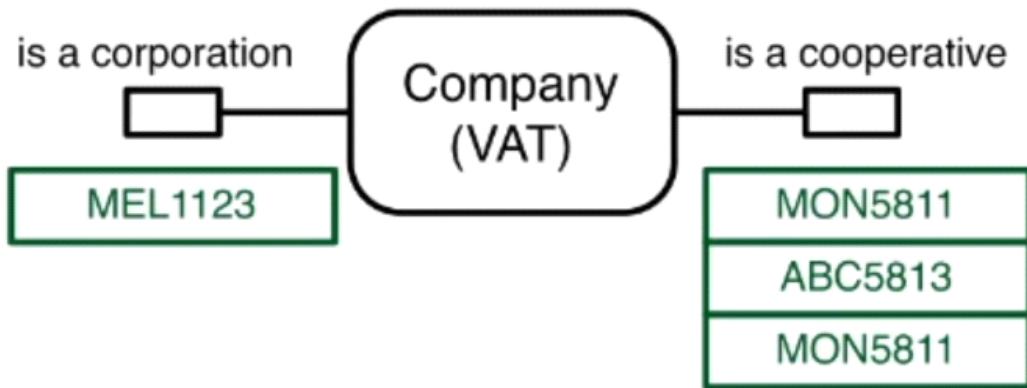
mercoledì 11 novembre 2020 16:06

L'obiettivo è l'aggiunta dei vincoli di unicità e l'arietà dei fact types

Il vincolo di unicità ci permette di individuare le chiavi che identificano i ruoli, perché uniche.  
Il vincolo di unicità permette anche di determinare se un fact type è elementare o ancora scomponibile.

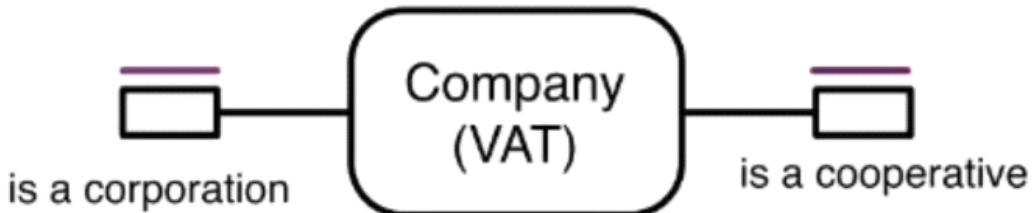
Nota: pensare ai fatti come Person has a Weight come a degli snapshot. Se vogliamo avere più pesi come fatti dovremmo aggiungere la data.

Fatti Unari



MON appare due volte. Se leggiamo il fatto Company "MON5811" is a Cooperative, notiamo un'informazione ridondante e quindi non necessaria.

Ogni fatto unario è unico e lo si indica così:



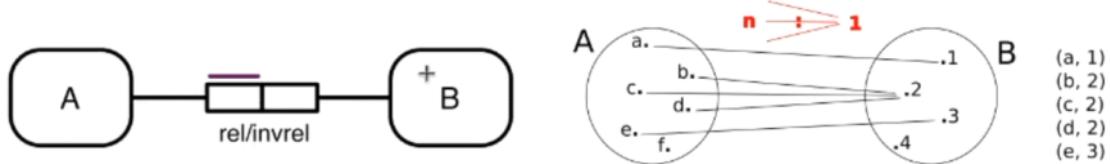
In ORM sono obbligatori tutti i vincoli di unicità, anche se ovvi.  
(la barra orizzontale può stare anche sotto)

Fatti Binari

Abbiamo 4 possibilità:

1. **Many-to-one** (n:1): each A in relation rel with **at most one** B; each B in relation rel with **many (0+)** As. (1 simple UC)
2. **One-to-many** (1:n): each A in relation rel with **many (0+)** Bs; each B in relation rel with **at most one** A. (1 simple UC)
3. **One-to-one** (1:1): each A in relation rel with **at most one** B, and **vice versa**. (2 simple UCs)
4. **Many-to-many** (n:m): each A in relation rel with **many (0+)** B, and **vice versa**. (1 composite UC)

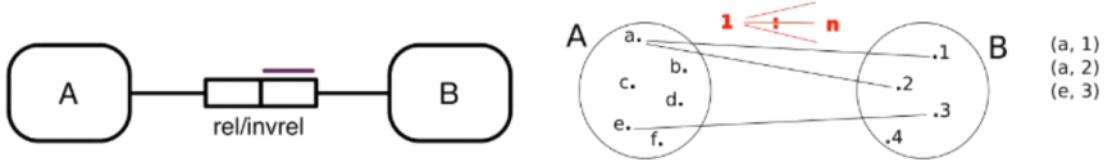
1. Molti a uno:



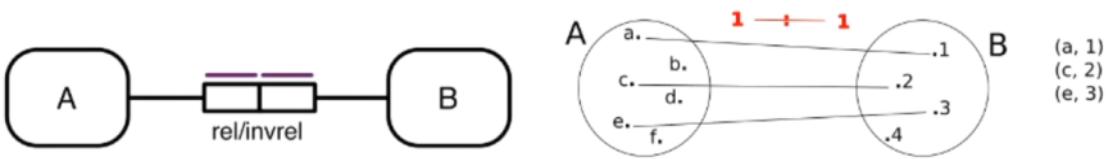
Molti elementi di A sono in relazione al più con 1 B.

Ogni elemento di A è unico e non si ripete.

2. Uno a Molti: è lo speculare.



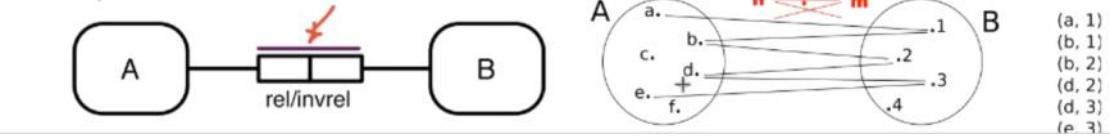
3. Uno a uno



Ogni elemento di A è associato con al più un elemento di B e vice versa.

4. Molti a Molti

- Temporization is a common case.

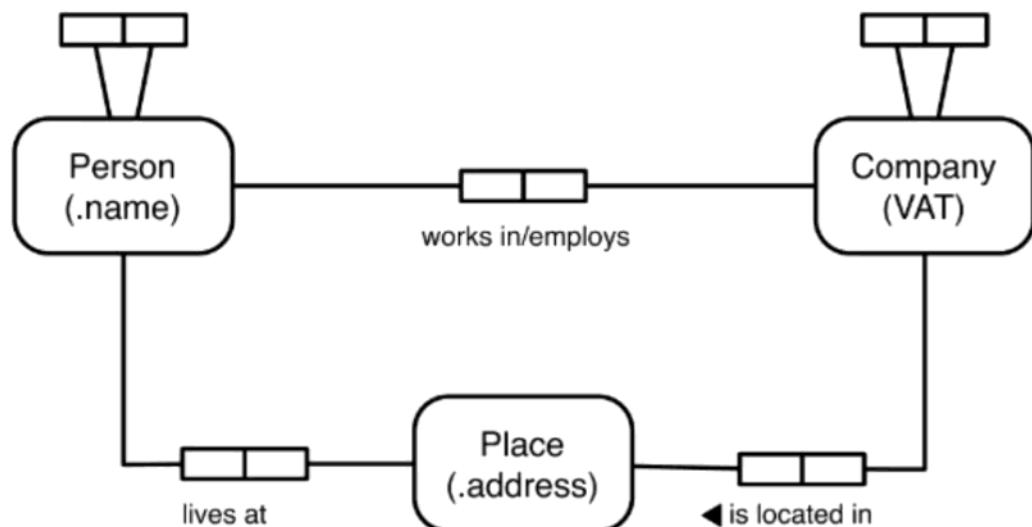


Non c'è unicità, ma ogni fatto è rappresentato una volta sola.

Come estraiamo i vincoli?

is husband of / is wife of

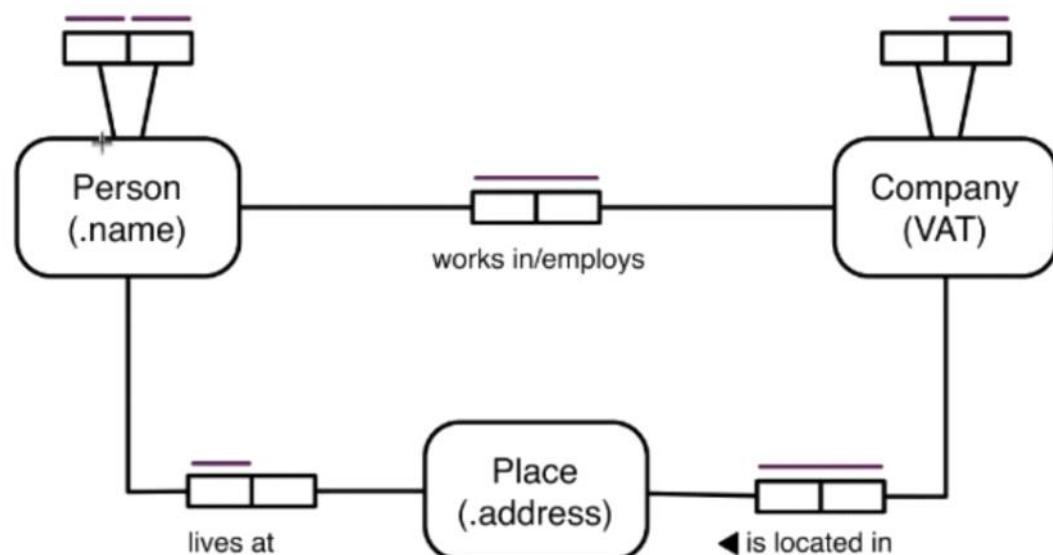
owns/is owned by



Si interagisce con gli esperti del dominio e ci si pone delle domande riguardo ogni vincolo, cercando dei controsensi. ("è possibile che un'azienda sia localizzata in più di un posto?")

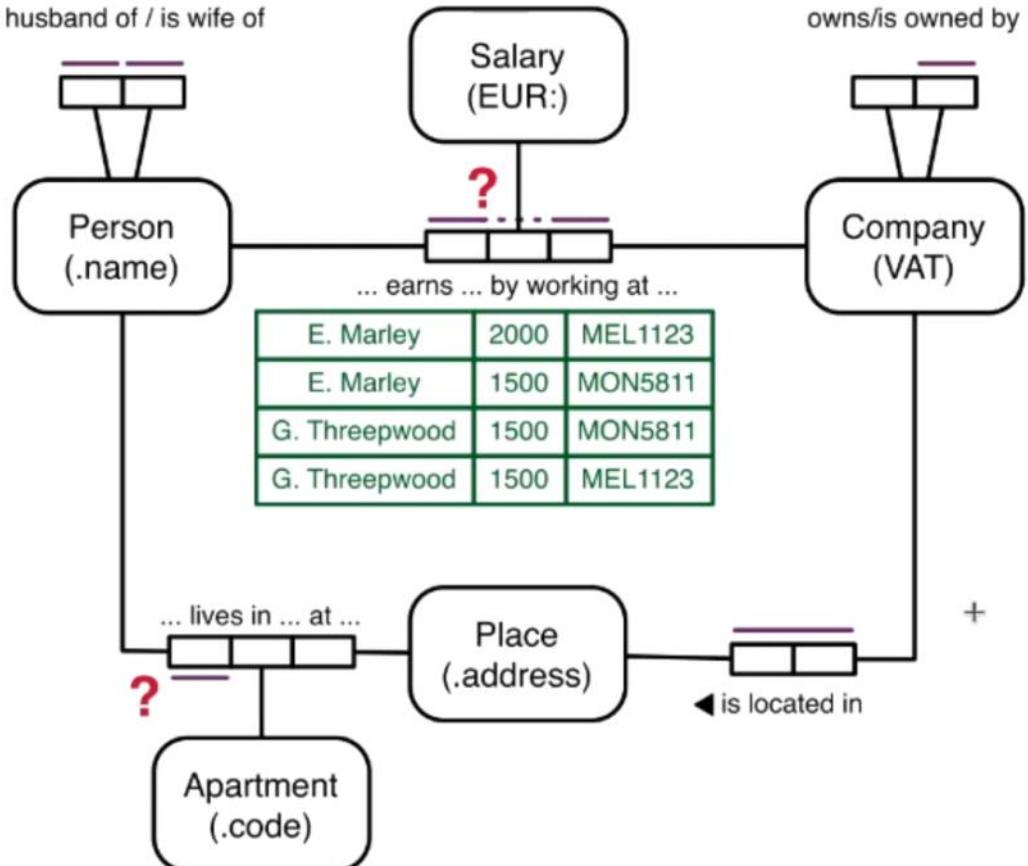
is husband of / is wife of

owns/is owned by

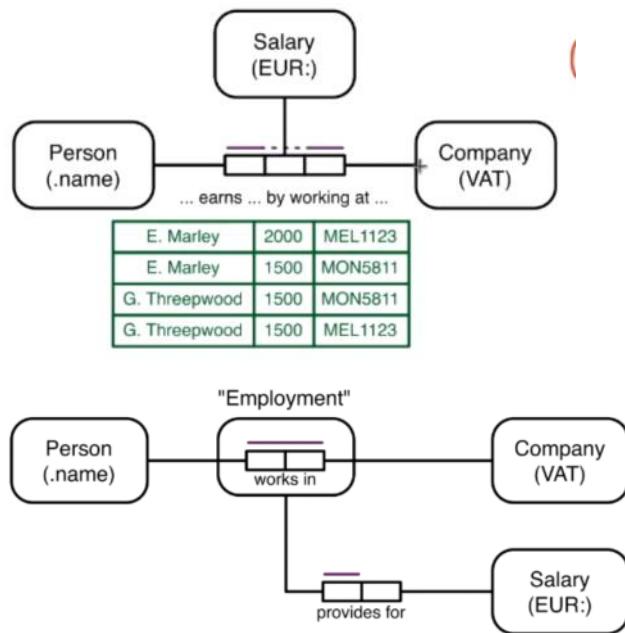


Fatti ternari

is husband of / is wife of

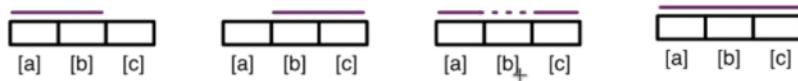


Potremmo reificare:



Possibilità di combinazione coi vincoli ternari

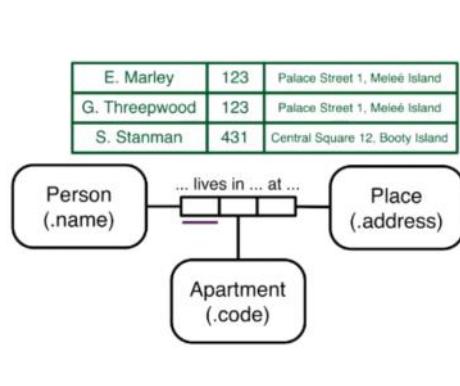
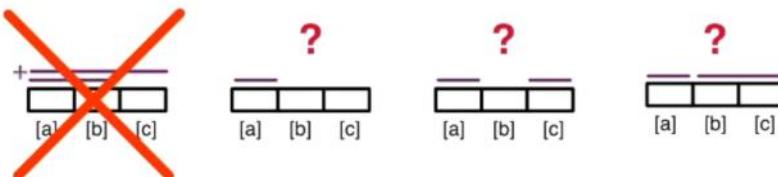
- Single UCs.



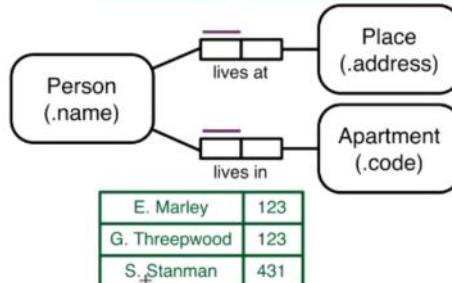
- Combined UCs.



- What about other possibilities?



E. Marley	Palace Street 1, Meleé Island
G. Threepwood	Palace Street 1, Meleé Island
S. Stanman	Central Square 12, Booty Island



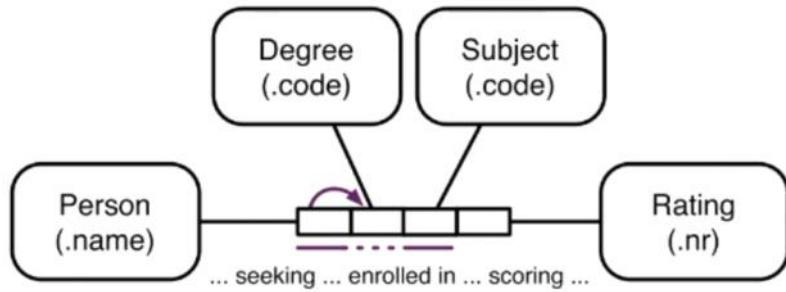
- Person acts as a pivot: determines both Apartment and Place.
- The UC *reveals* that the predicate is **non-elementary**.

La lunghezza della chiave può farci capire se le relazioni sono elementari o no.

- **Key:** minimal combination of roles spanned by an UC.
  - **Simple key:** spans one role only.
- Predicates of wrong arity.
  - Too long: non-elementary fact type → to be split.
    - Person lives in Apartment at Place
    - Person lives in Apartment; Person lives at Place.
  - Too short: information-loss → recombination.
    - Lecturer teaches Course; Lecturer teaches at Faculty
    - Lecturer teaches Course at Faculty.
- Major issue: too long predicates.

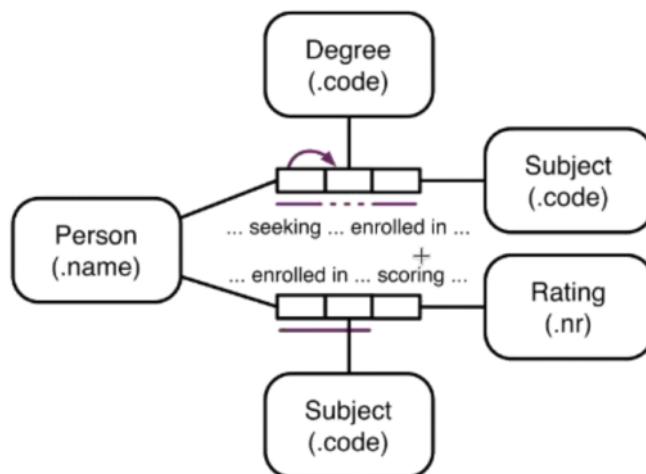
Ogni fatto n-ario deve avere un vincolo di unicità che ne coinvolga almeno n-1.

Ogni fatto di lunghezza n ha esattamente una chiave di lunghezza n e una o più chiavi di lunghezza n-1.



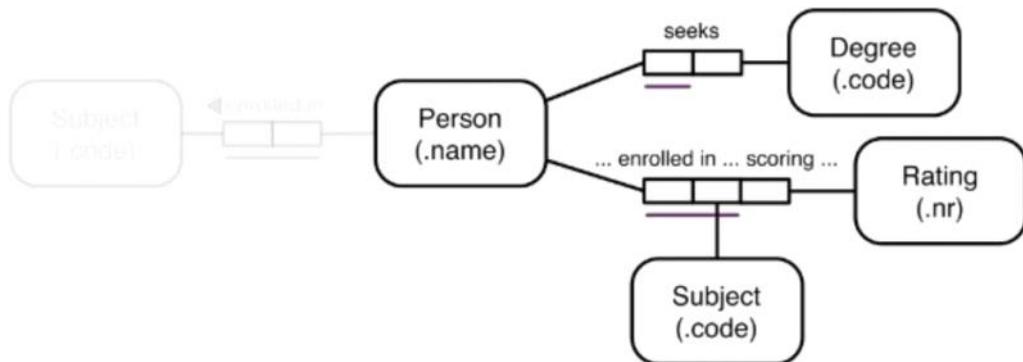
Person e Subject determinano Degree e Rating.  
Con la freccia si indica che il degree è determinato dalla persona.

La relazione non è corretta perché la chiave è di lunghezza 2.

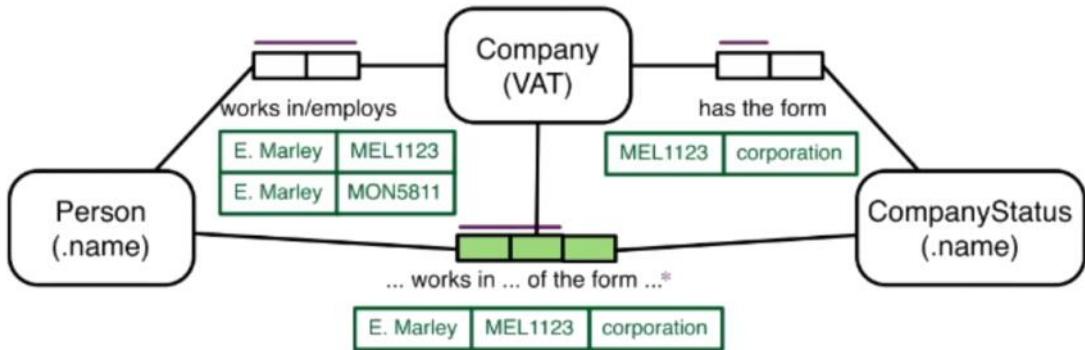


Non abbiamo ancora espresso che Person determina funzionalmente Degree.

La regola n-1 è una regola sufficiente, ma non necessaria.

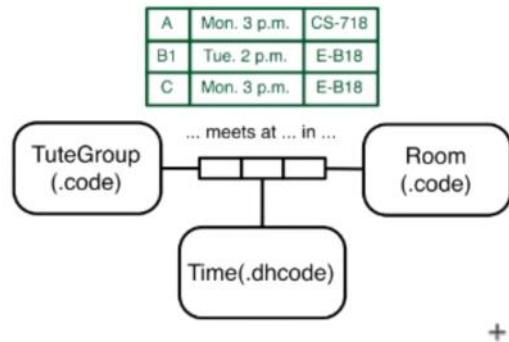


Join concettuale

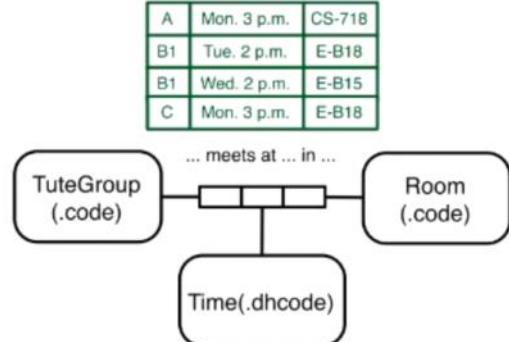


Suppose the fact table covers all possible cases.

- Is this fact type splittable?



- And this version?

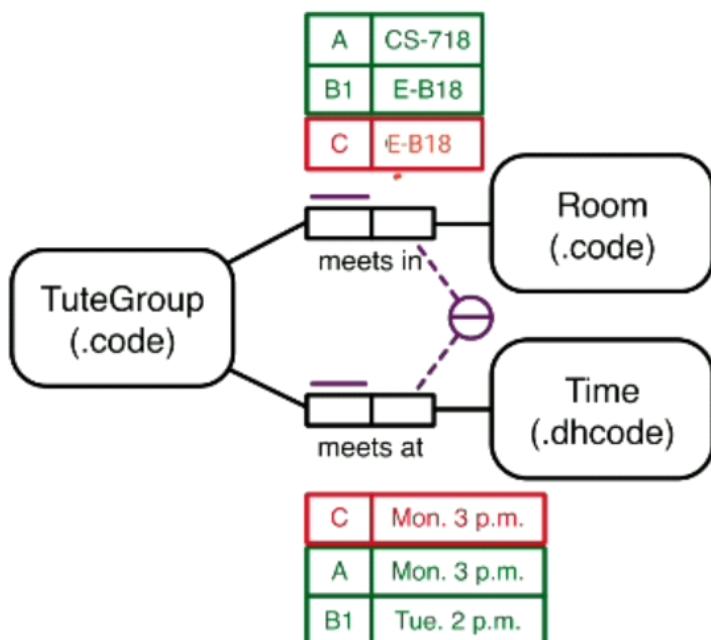
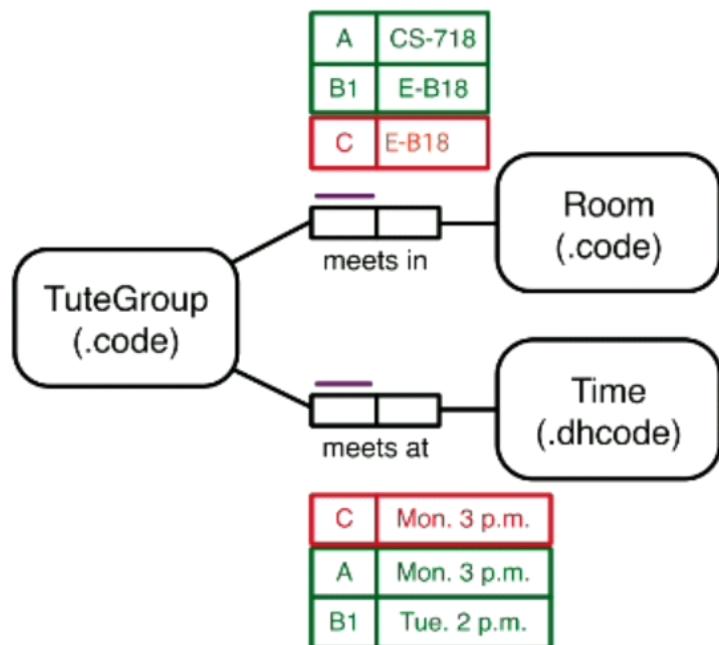


Il primo è spezzabile col vincolo di unicità a sinistra.

Il secondo no, e il vincolo è sx-centro: se scomponessimo dopo facendo il join risultano più valori del normale.

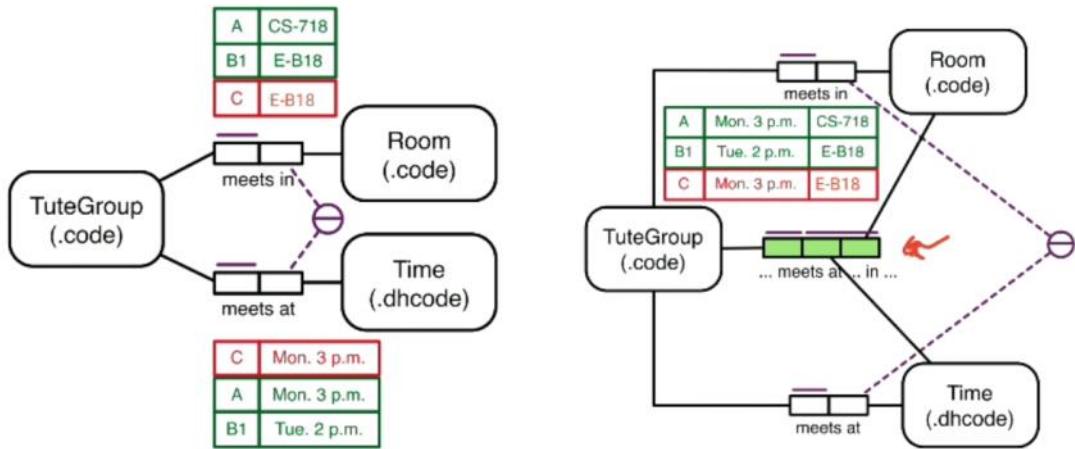
# Vincoli di unicità esterni S4 - 13 Ott

mercoledì 11 novembre 2020 17:00



Unisco i ruoli che insieme determinano funzionalmente un terzo ruolo.

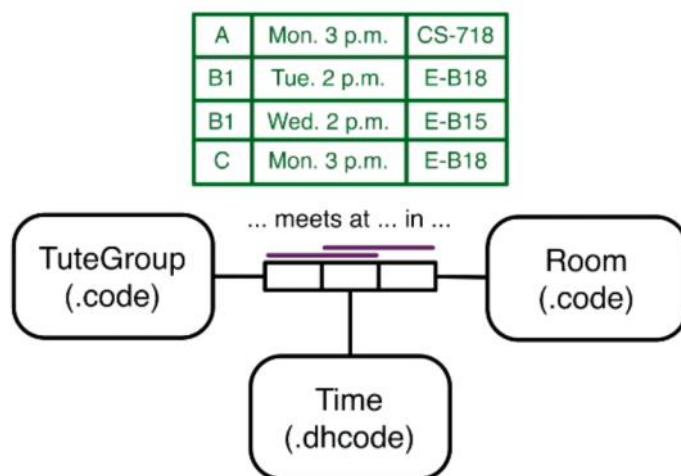
Ogni combinazione di MeetingRoom e MeetingTime è accoppiata con al massimo un TuteGroup.



L'indicazione in verde chiaro è derivata e non deve essere rappresentata.

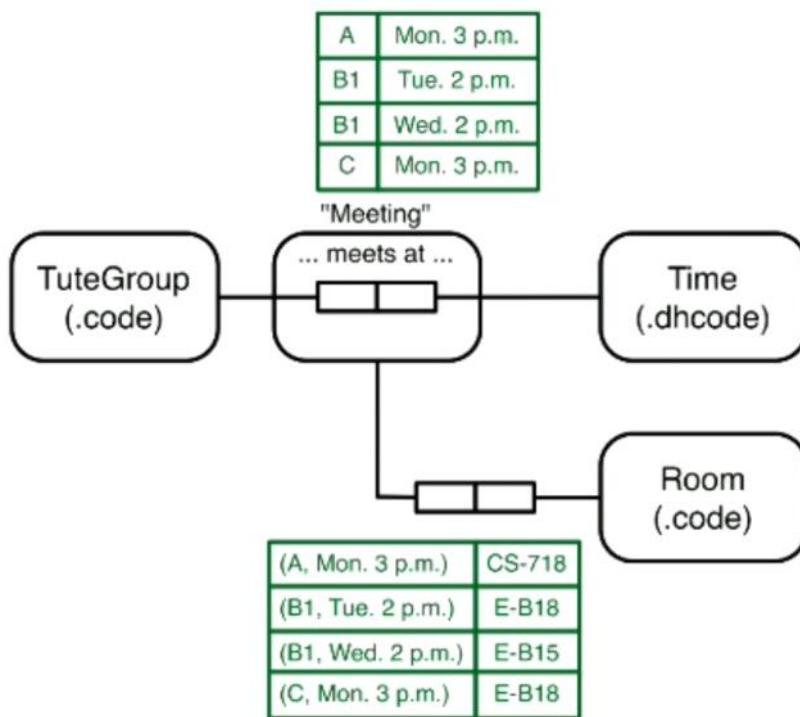
Ogni popolazione di "meets at" join "meets in" ha una coppia (Room, Time) unica.

Un gruppo e un'ora otteniamo una stanza, e data un'ora e una stanza e sappiamo il gruppo.



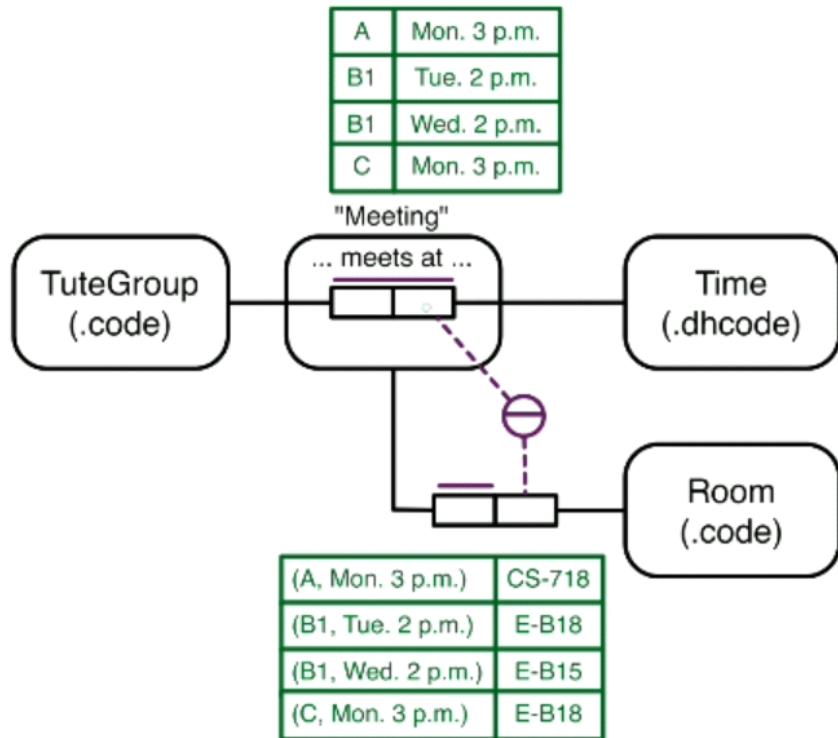
La relazione è elementare.

Si può desiderare di evidenziare il meeting.



Si perdono però delle informazioni! Non sappiamo più che data un'ora e un gruppo abbiamo una stanza.

Allora si aggiunge il vincolo di unicità esterno:



# S5 - 13-14 Ott

mercoledì 11 novembre 2020 17:14

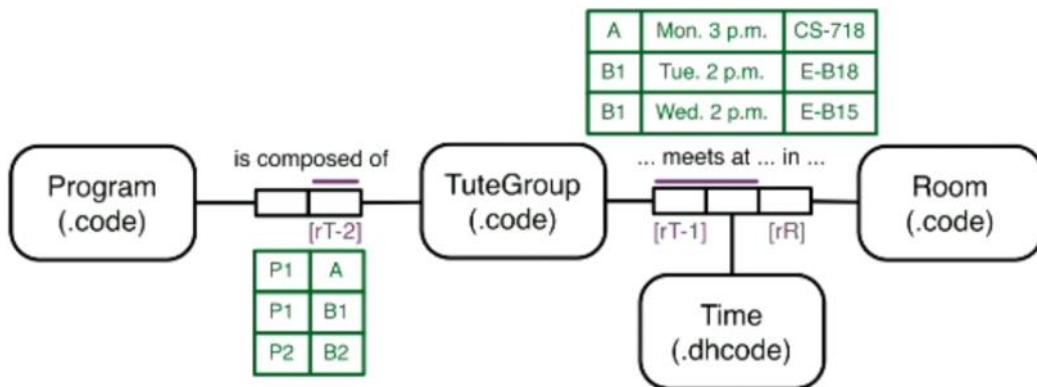
Passo 5:

Aggiunta di vincoli obbligatori sui ruoli e verifica di derivazioni logiche.

Popolazione di un entity type:

Dato un certo snapshot, la popolazione di un entity type sono le istanze di quell'entity type.

La stessa definizione si applica alla popolazione di un ruolo: l'insieme degli oggetti referenziati dai valori nella colonna di quel ruolo



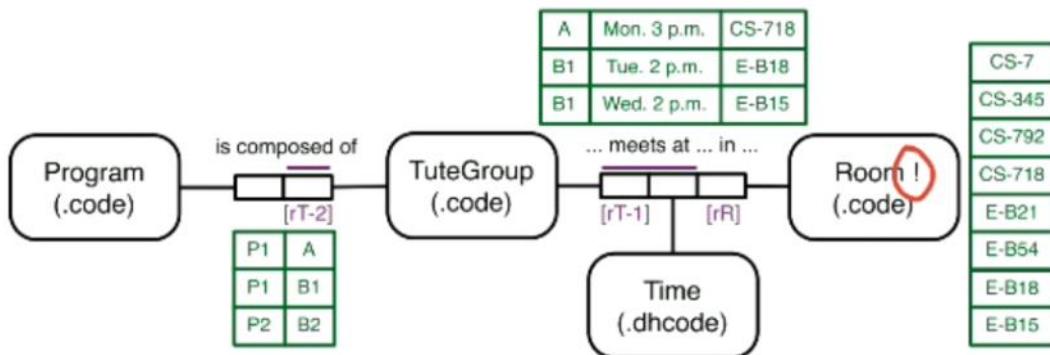
$\text{Val}(rT-1) = \{'?', 'B1'\}$

$\text{Pop}(rT-1) = \{\text{TuteGroup}(.code) 'A', \text{TuteGroup}(.code) 'B1'\}$

La popolazione di un entity type è l'unione della popolazione dei suoi ruoli.

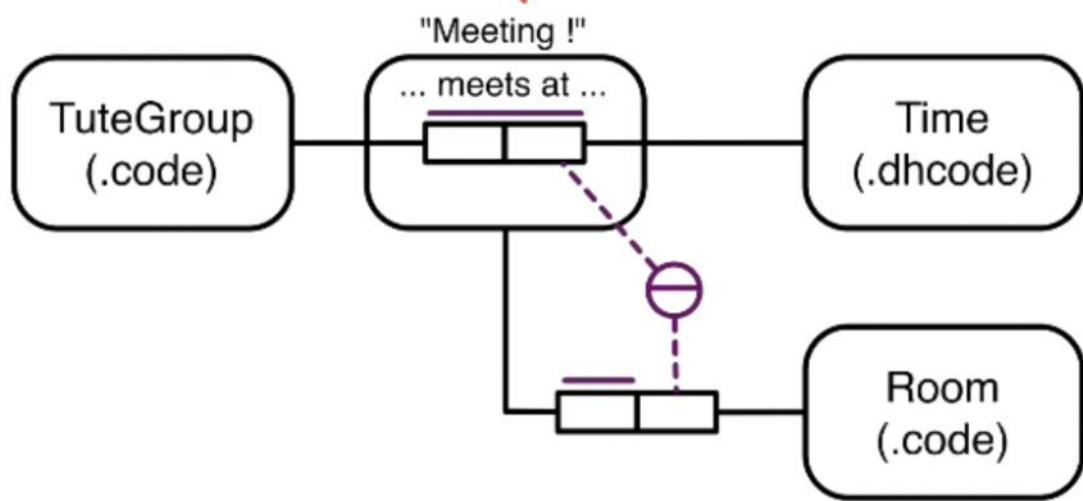
$\text{Pop}(\text{TuteGroup}) = \text{pop}(rT-1) \cup \text{pop}(rT-2) = \{A, B1\} \cup \{A, B1, B2\} = \{A, B1, B2\}$

Con il ! Indichiamo che l'entity è importante per sé

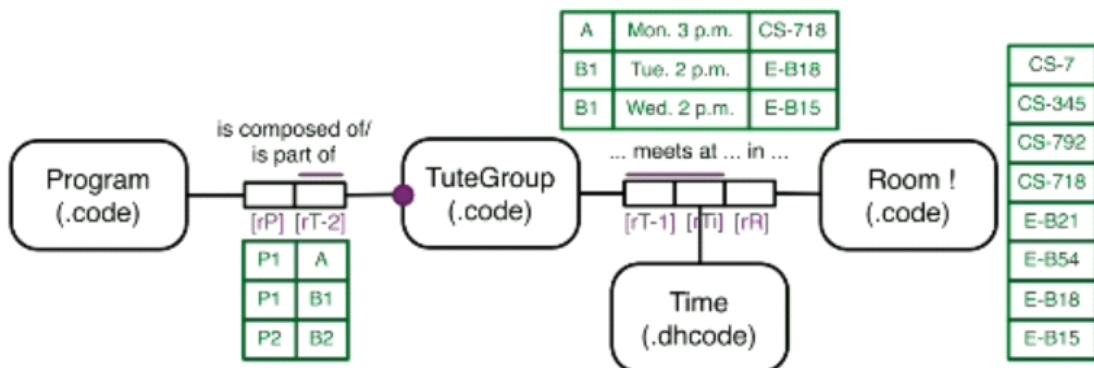


Non possiamo cambiare l'enumerazione definita con il !

L'esistenza può essere messa sulle oggettificazioni:



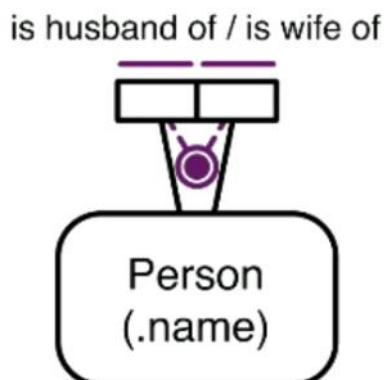
Il vincolo di obbligatorietà di una relazione lo si mette su un ruolo e indica che necessariamente tutte le istanze devono avere quella relazione.



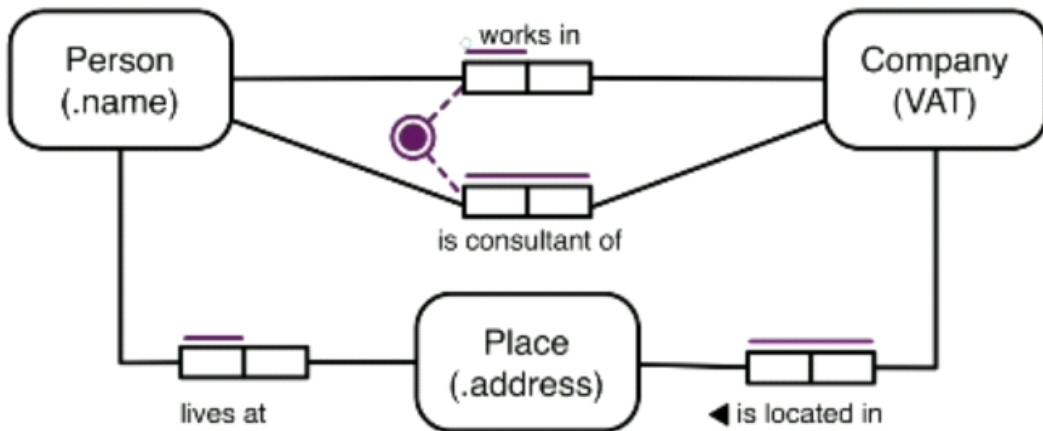
"Se esiste un gruppo di tutoraggio, deve avere un programma di tutoraggio associato".  
In genere in ORM il pallino dovrebbe essere verso la relazione, e non verso il ruolo.

I ruoli obbligatori modificano gli aggiornamenti disponibili dei dati.  
"Ogni TuteGroup è parte di al massimo un programma"  
"Ogni TuteGroup è parte di qualche Programma"  
-> Ogni TuteGroup fa parte di esattamente un Programma.

Il vincolo di obbligatorietà può anche essere esterno e può coinvolgere più entity types.



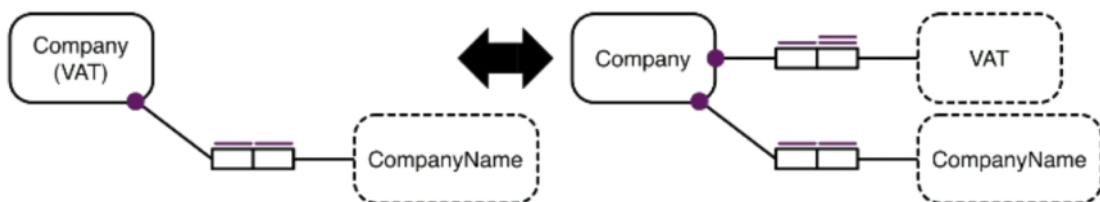
"Una persona appare come marito o come moglie di qualcuno"  
Se delle persone apparissero in altre relazioni, dovrebbero anche apparire in questa relazione.



L'unione dei lavoratori e dei datori di consulenze forma la popolazione di person.

----

Noi desideriamo che una Company abbia un solo codice fiscale, e che un codice fiscale non sia assegnato ad altre aziende. Dobbiamo anche specificare che ogni azienda deve avere un codice fiscale.



Supponiamo che il nome sia esclusivo ed obbligatorio.

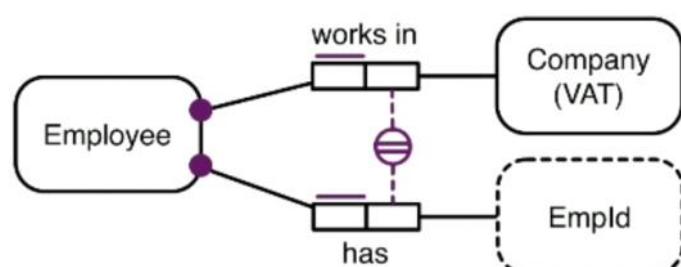


Anche il nome è una reference mode per la Company, ma il VAT è il **preferred reference scheme**, e lo si indica mettendolo sotto l'entità tra parentesi tonde. In alternativa si può utilizzare una doppia linea orizzontale nella relazione.

Nota: non tutti i codici fiscali sono associati ad un'azienda, altrimenti non potremmo introdurre altre!

----

Supponiamo questo schema:



Per ogni Company ed Empld, al massimo un Employee lavora in quella Company e ha quell'Empld.

Nulla vieta che quell'EmpID sia assegnato ad un altro impiegato in un'altra azienda!  
Non possiamo utilizzare quindi VAT per la reference mode di employee.

Il simbolo viola indica un vincolo di unicità e il fatto che sia anche lo schema preferito per rappresentare gli impiegati.

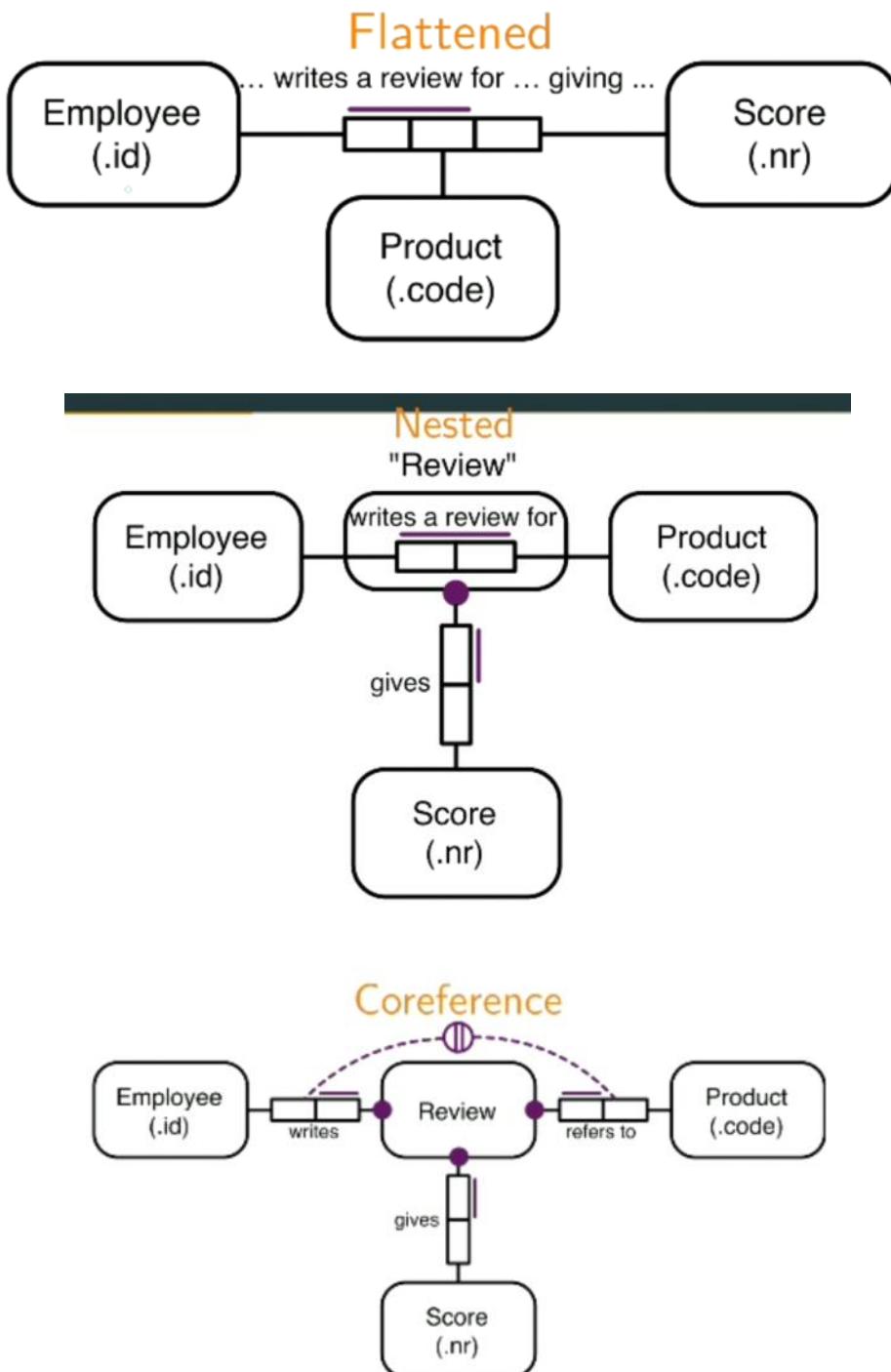
--

Vogliamo modellare la seguente richiesta:

"Ogni impiegato può scrivere una recensione per un prodotto, dandogli un certo punteggio".

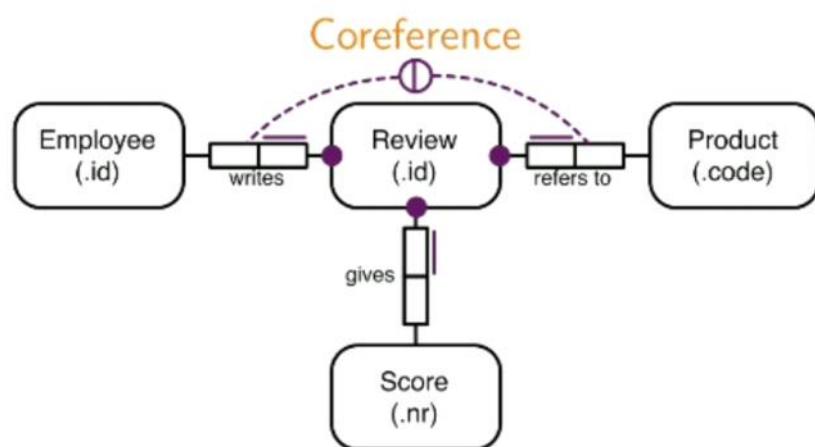
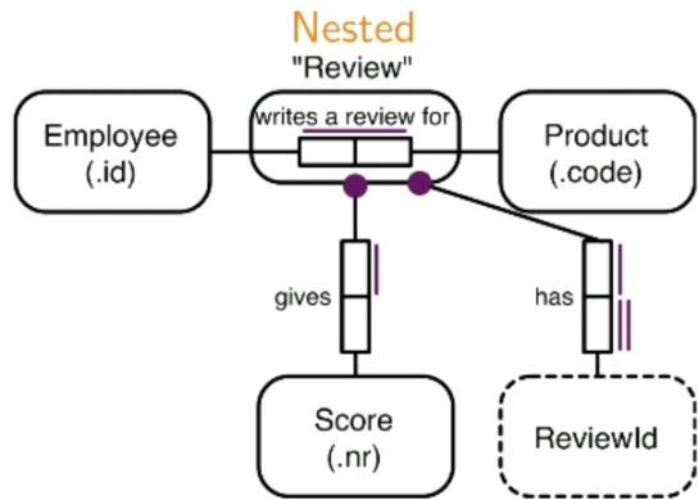
Object types: Employee, Product, Score, Review?

Core Fact type: "write" or "write a review for"?

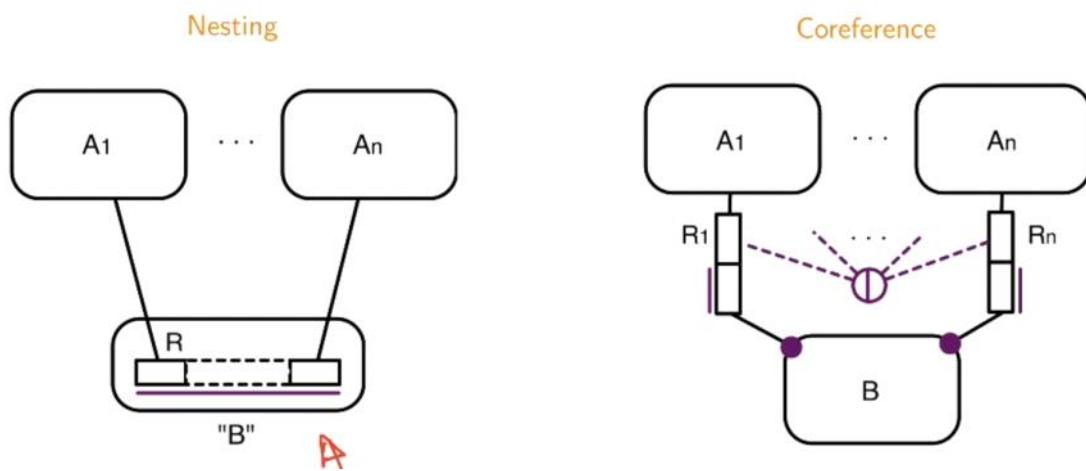


La reference mode è la copia prodotto-impiegato.

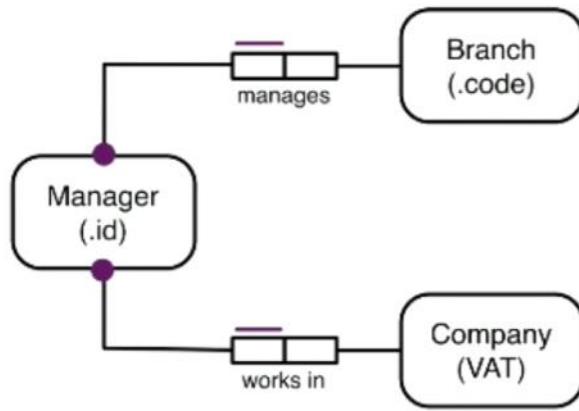
Vogliamo aggiungere un'ID alla recensione:



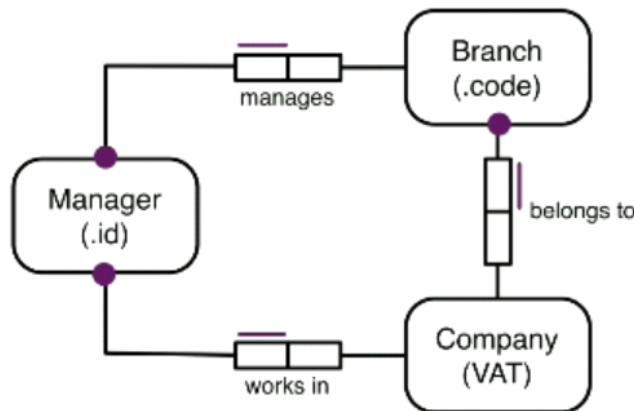
Genericamente:



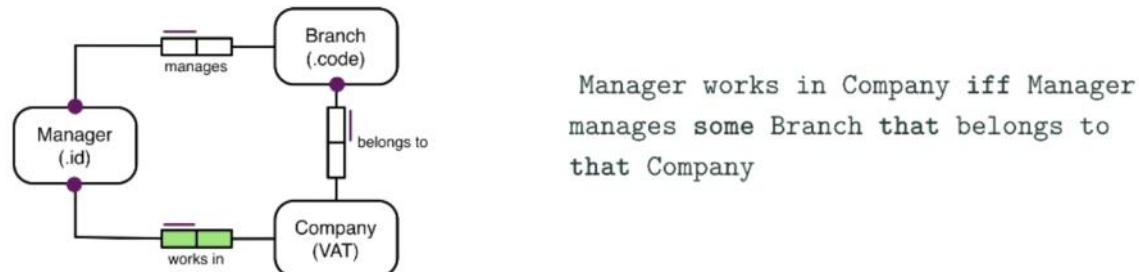
Nel passo 5 è importante anche fare attenzione alle ridondanze!  
ORM è fatto per rappresentare nel modo più preciso possibile.



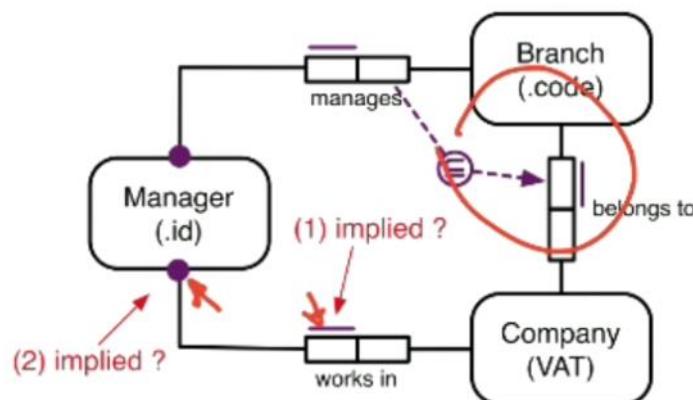
È possibile/desiderabile stabilire una relazione tra Branch e Company? È funzionale?  
Introduciamo ancora la relazione di appartenenza



È possibile derivare un fact type da due fact type?



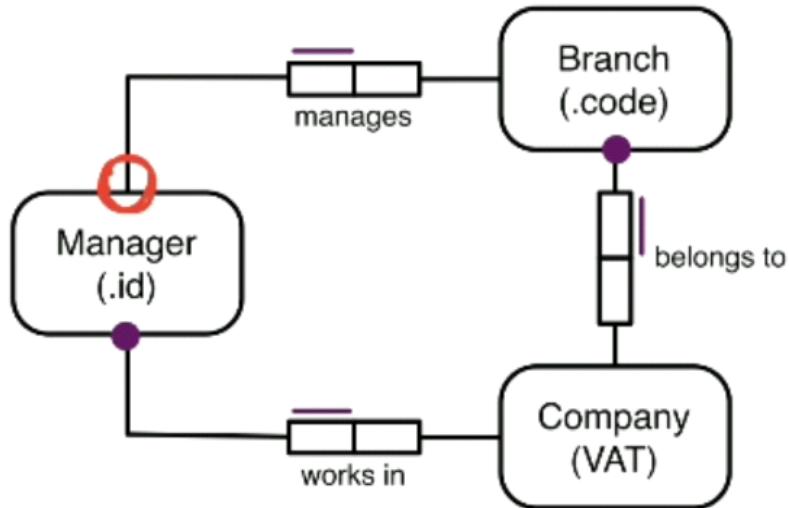
Che succede se mi manca il simbolo di obbligatorietà?



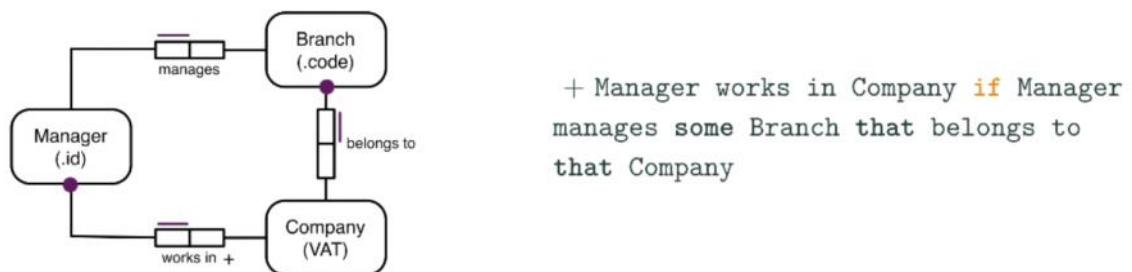
Possiamo derivare il vincolo di unicità e l'obbligatorietà!  
Va studiato se la composizione semantica è corretta oppure no.  
Questo significa che è un manager se è assegnato ad un ramo.

Se manca sul manager, potrebbe significare che il manager esiste nell'azienda ma al momento non

ha un Ramo assegnato:



In questo caso lo schema non è derivato, ma semiderivato e lo si indica con un + invece di un \*.



# S6 - 20-21 Ott

mercoledì 11 novembre 2020 18:39

Passo 6:

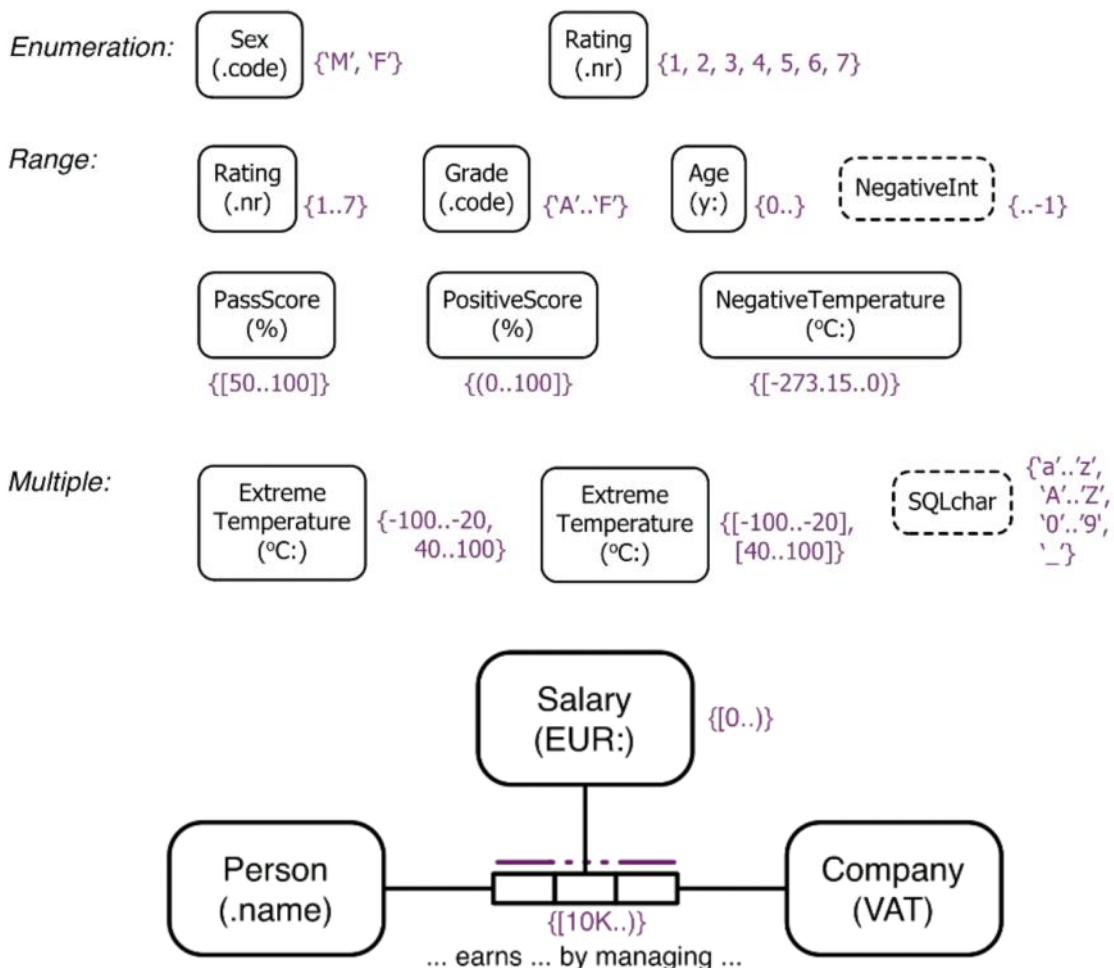
Aggiunta di vincoli di valore, sottoset, uguaglianza, esclusione e sottotipo

I vincoli che analizzeremo sono quelli di:

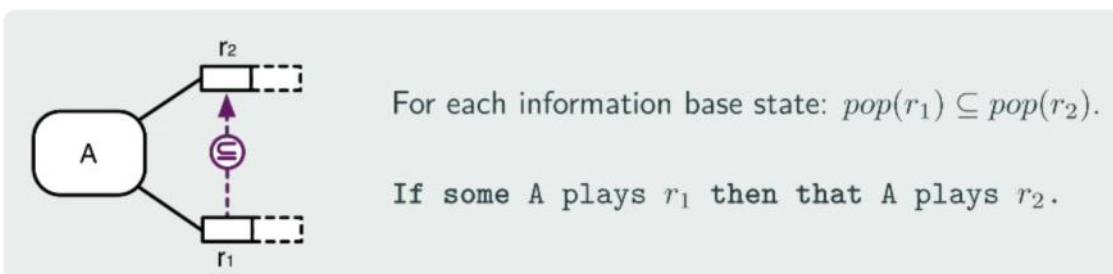
- Sottoinsieme
- Uguaglianza
- Esclusione

I vincoli limitano il numero di valori possibili che possono essere assunti da un value type o un ruolo.

Possiamo indicare un'enumerazione: {'M', 'F'} o un range



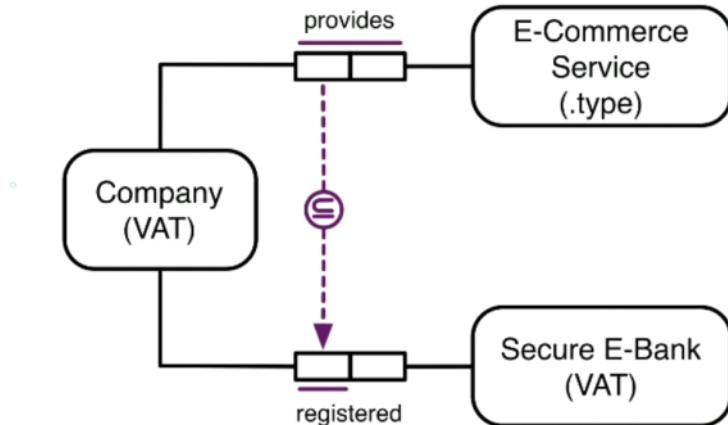
Vincoli insiemistici:



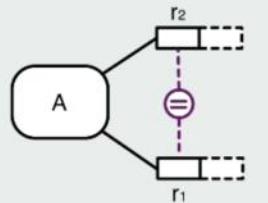
Ogni oggetto che popola  $r_1$  deve popolare anche  $r_2$ .

Nota: i duplicati non influiscono.

- A Company could be registered to at most one Secure E-Bank Service.
- A Company could provide E-Commerce Services.
- If some Company provides some Secure E-Commerce Service, that Company must be registered to a Secure E-Bank Service.



#### Vincolo di uguaglianza



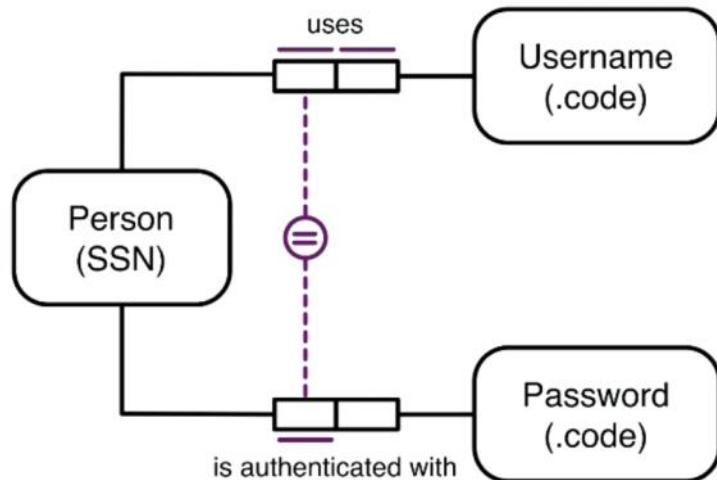
For each information base state:  $\text{pop}(r_1) = \text{pop}(r_2)$ .

For each **A**, that **A** plays **r<sub>1</sub>** if and only if that **A** plays **r<sub>2</sub>**.

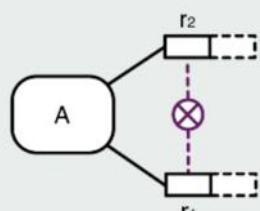
Le due popolazioni devono coincidere.

I duplicati anche in questo caso non influiscono.

- A Person could use an Username, and be authenticated with a Password.
- Either a Person does not have a Username nor a Password, or the Person has both.



#### Vincolo di Esclusione

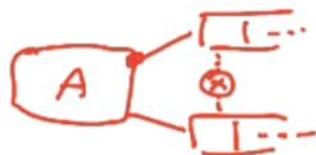


For each information base state:  $\text{pop}(r_1) \cap \text{pop}(r_2) = \emptyset$ .

For each **A**, at most one of the following holds: **A** plays **r<sub>1</sub>** ; **A** plays **r<sub>2</sub>**.

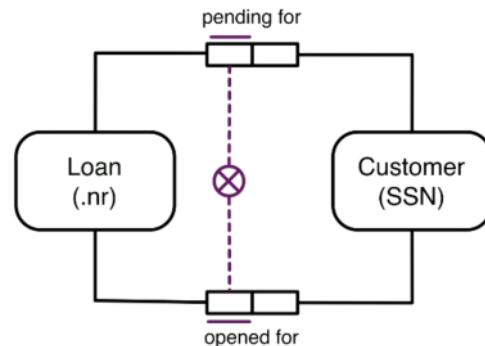
Ogni oggetto che popola r1 non può popolare r2 e viceversa.

I ruoli devono essere opzionali.



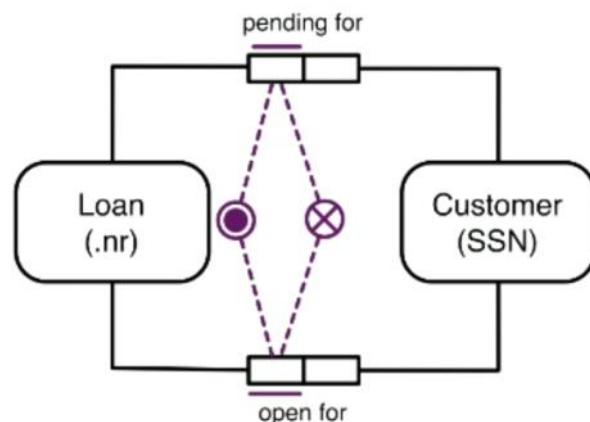
$$\text{pop}(\gamma_1) = \text{pop}(A) \quad \text{pop}(\gamma_2) = \emptyset$$

- A Loan is either pending for a (single) Customer, or open for that Customer, but not both.



Nota: così scrivendo possiamo dire che il prestito non è né aperto, né in attesa!

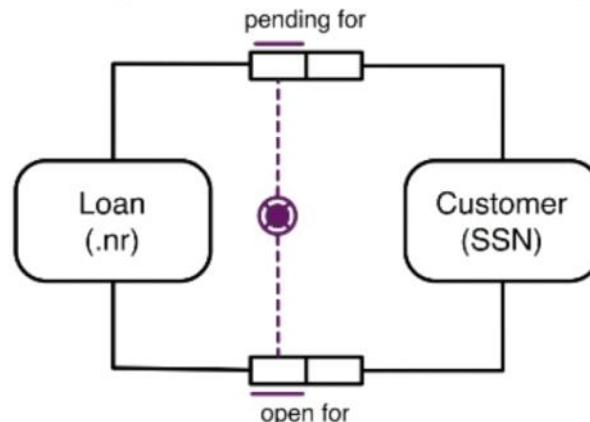
Non possiamo aggiungere dei vincoli di ruolo, ma possiamo usare un vincolo di obbligatorietà. Ricordiamo che l'unione delle popolazioni in questo modo deve essere uguale alla popolazione totale e che l'intersezione è vuota:



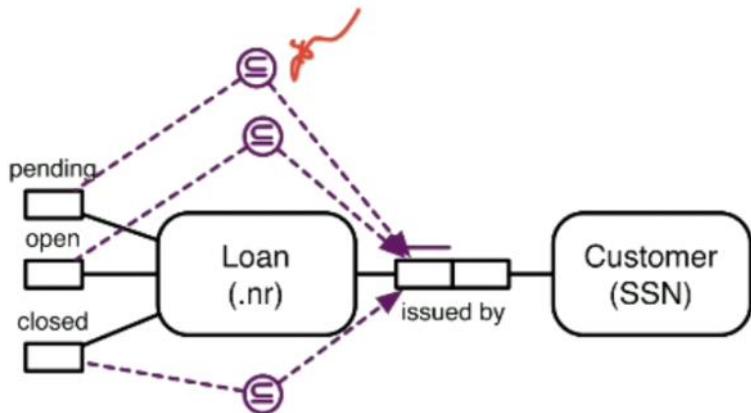
Per ogni Loan, esattamente una delle due condizioni vale:

- Loan è in attesa per un qualche Cliente
- Loan è aperta per un qualche Cliente

In ORM si possono fondere i due simboli per formare il simbolo "del salvagente":

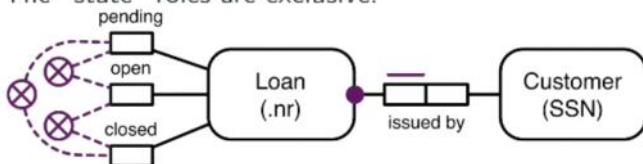


Complichiamo la situazione del prestito, supponendo che nulla escluda

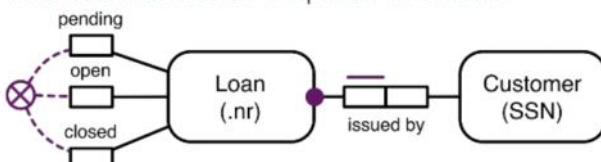


Le tre relazioni non sono esclusive tra di loro! Indicano però che se sono uno dei tre stati, allora sono richieste da un certo cliente.

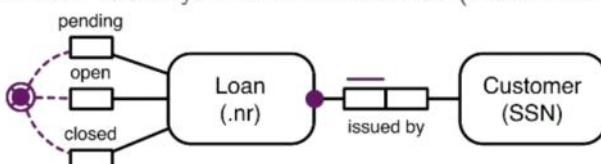
- The “state” roles are exclusive.



- This notation can be simplified as follows.

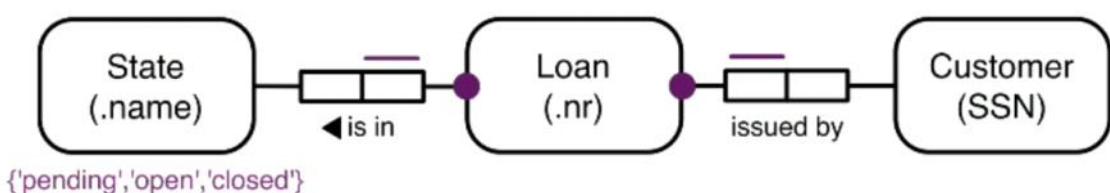


- A Loan is always in one of the states (inclusive-or, to be mixed with exclusion → ex-or).



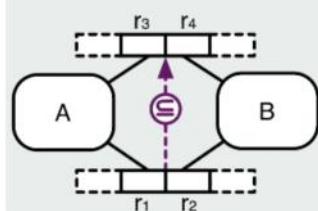
È un prestito emesso necessariamente ad un cliente e che ha solo uno dei tre stati.

Un altro metodo per rappresentare è così:



Un vincolo insiemistico può anche essere collegato a sequenze di ruoli:

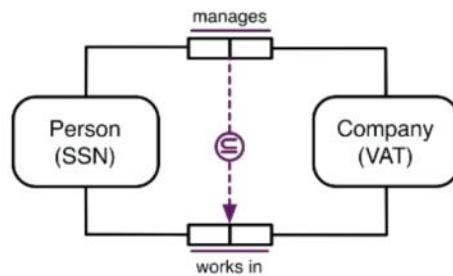
#### Pair-subset constraint



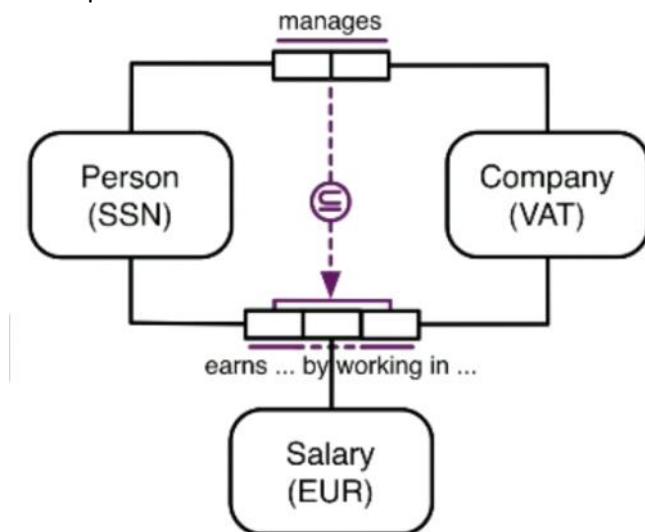
For each information base state:  $\text{pop}(r_1, r_2) \subseteq \text{pop}(r_3, r_4)$ .

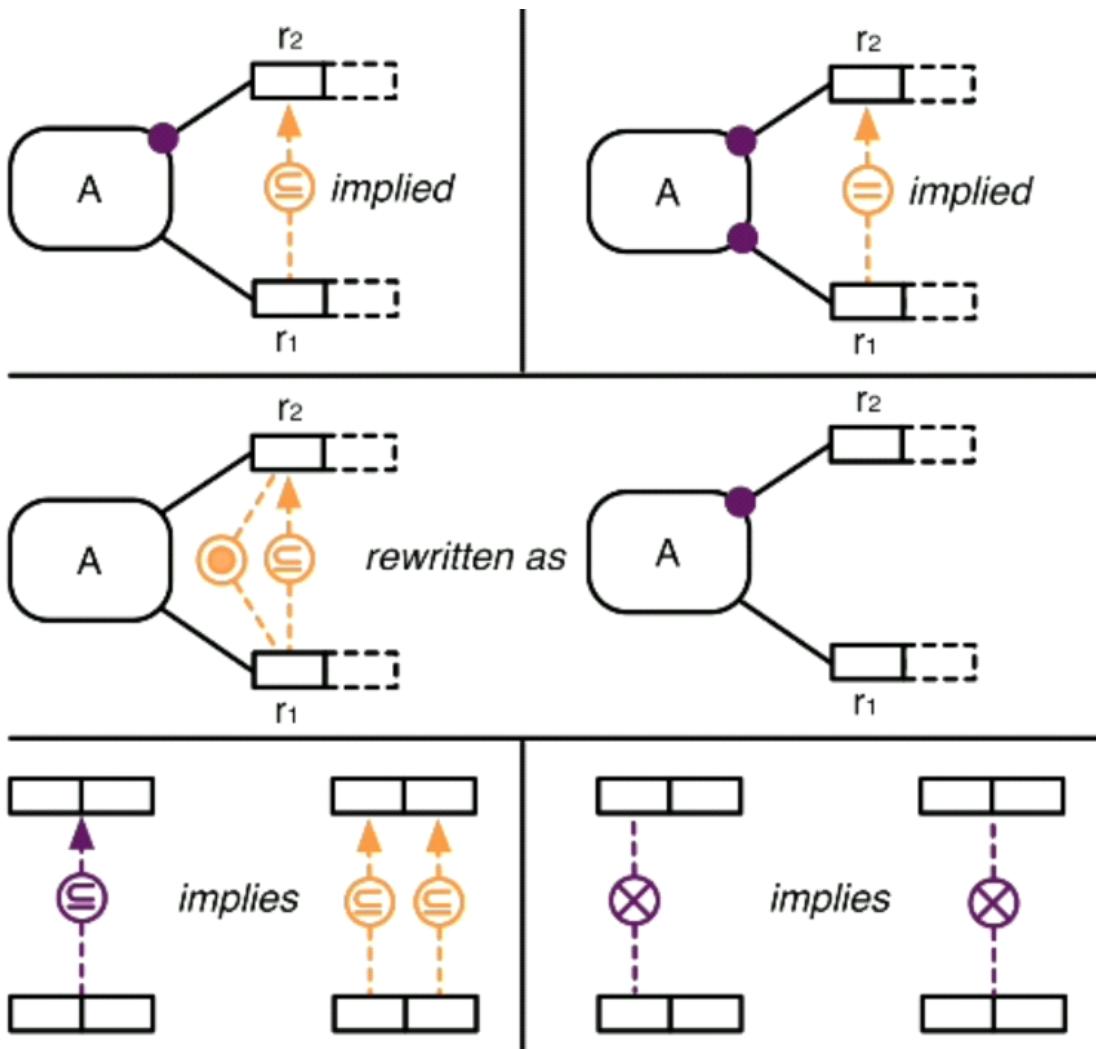
Each pair in  $\text{pop}(r_1, r_2)$  is also in  $\text{pop}(r_3, r_4)$ .

- Consider Persons who work in Companies and Persons who manage Companies.
- Clearly, Each Person who manages a Company works in that Company.



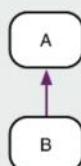
Con relazioni n-arie si usa un ponticello:





Vincolo di sottotipo:

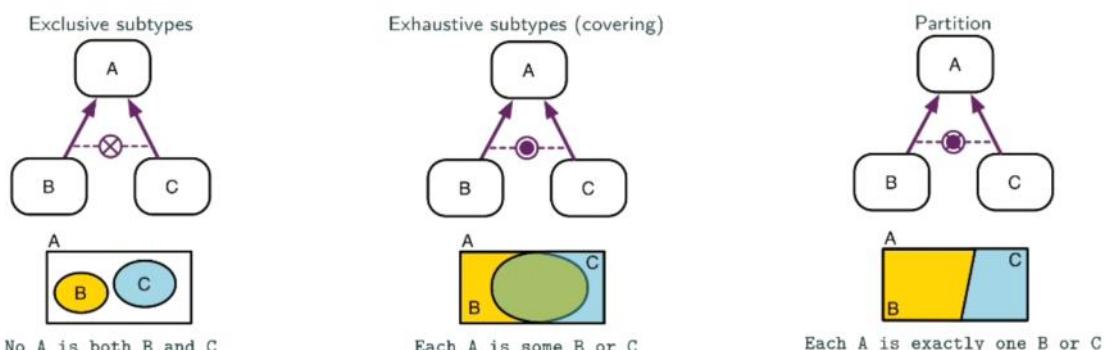
#### Proper subtype

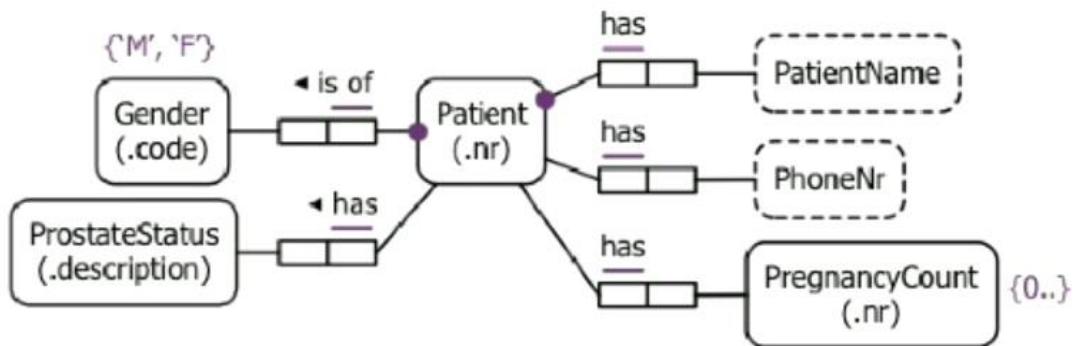
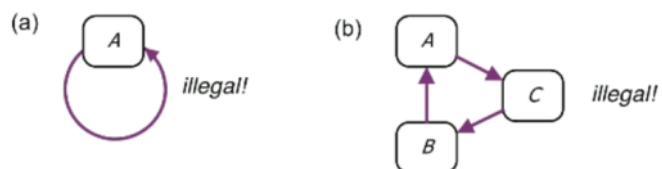


$A \neq B$  and for each information base state:  $\text{pop}(B) \subseteq \text{pop}(A)$ .

Terminology:

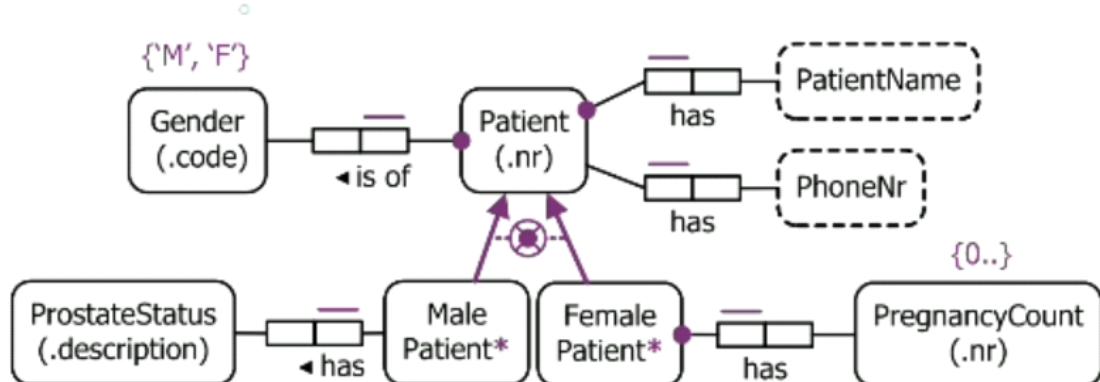
- A is a supertype, B and C are subtypes.
- Only one supertype → single inheritance.
- Multiple supertypes → multiple inheritance.
- If D subtype C, then D is an indirect subtype of A, because subtypehood relation is *transitive*.





PatientNr	Name	Gender	Phone	Prostate status	Pregnancies
101	Adams A	M	2052061	OK	-
102	Blossom F	F	3652999	-	5
103	Jones E	F	?	-	0
104	King P	M	?	benign enlargement	-
105	Smith J	M	2057654	?	-

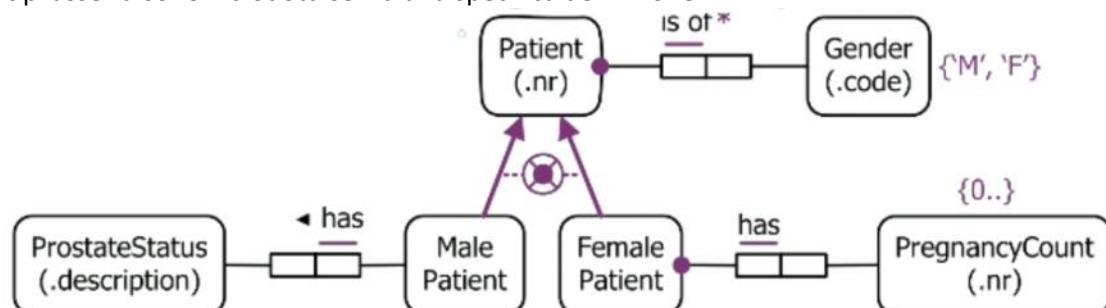
Introduciamo dei sottotipi derivati per dividere le relazioni specifiche:



\*Each MalePatient is a Patient who is of Gender 'M'.

\*Each FemalePatient is a Patient who is of Gender 'F'.

I tipi asseriti sono introdotti senza una specifica definizione:



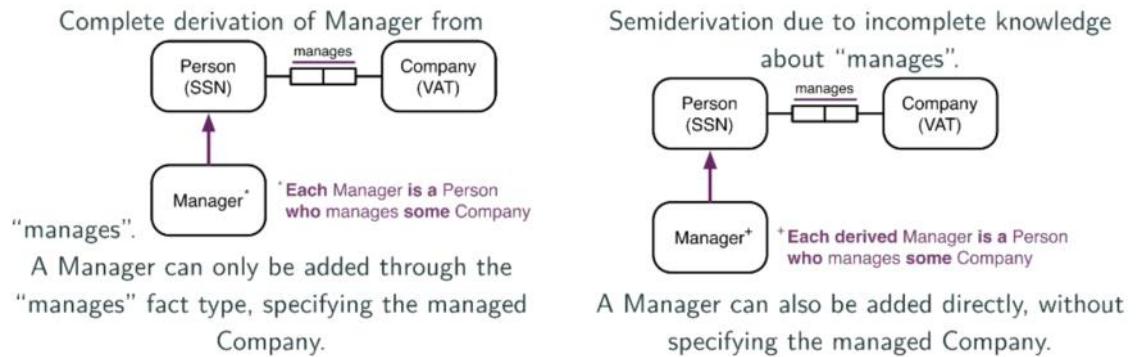
\*Patient is of Gender iff

Patient is a MalePatient and Gender = 'M'

or Patient is a FemalePatient and Gender = 'F'.

Dobbiamo immaginare che ci venga portato un foglio con le risposte indicate.

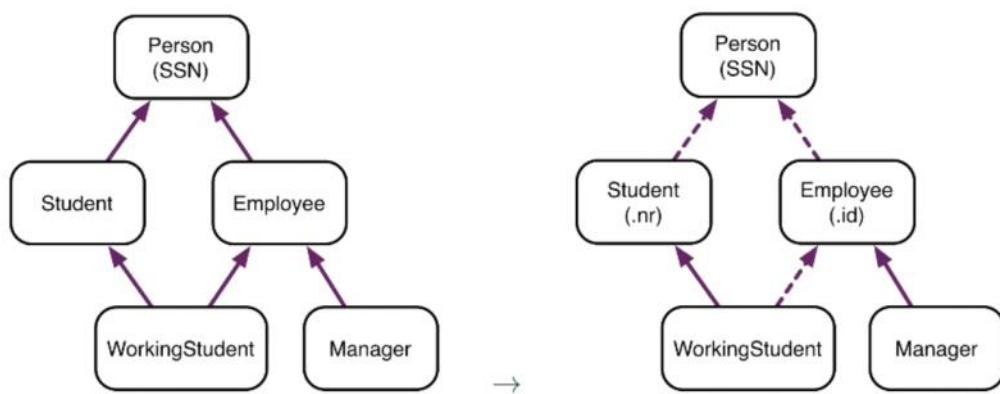
I semiderivati si usano per conoscenze incomplete.



How do we determine the preferred identification scheme of a subtype?

- Inherited from the corresponding supertype: solid arrow.
- Inherited from another supertype or autonomously determined: dashed arrow.

The second case reflects a *context-dependent reference scheme*.



Dato un object type:

1. Specificare tutti i vincoli obbligatori
2. Per ogni ruolo opzionale: **se**
  - a. È salvato solo per un sottotipo conosciuto **e**
  - b. Esiste una definizione di un sottotipo più forte che l'accompagna **oppure** un altro ruolo è registrato solo per quel sottotipo

Allora

2.
  - a. Introduco il sottotipo
  - b. Aggancio ad esso i suoi specifici ruoli
  - c. Dichiaro il sottotipo come derivato \*, asserito o semiderivato +
  - d. A seconda del \* o del +, aggiungo la regola di derivazione
  - e. Tornare al passo 1 considerando il sottotipo

Procedura di generalizzazione:

Dati due tipi di oggetti completamente separati A e B:

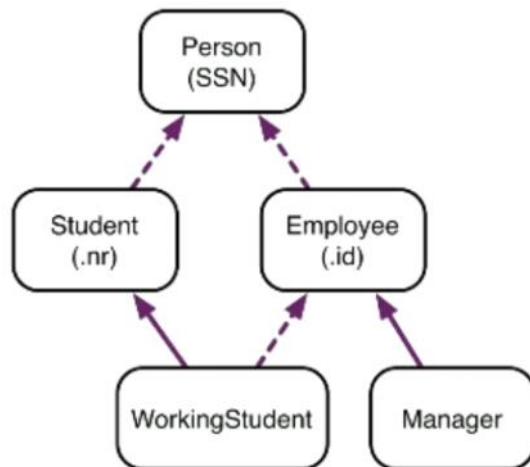
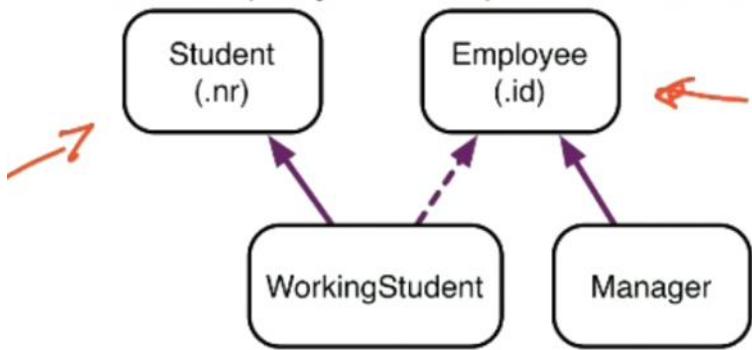
**Se**

- A e B si intersecano o sono comparabili **e** vogliamo modellare questa caratteristica
- **Oppure** A e B sono mutualmente esclusivi **e** esistono delle informazioni comuni ad entrambi **e** vogliamo elencare A e B insieme per questa informazione

Allora

1. Introdurre il supertipo A U B con il proprio schema di identificazione
2. Aggiungere i predicati di classificazione al supertipo, per identificare A e B
3. Agganciare i ruoli comuni al supertipo
4. **Se A (o B) gioca un ruolo specifico**  
**allora** definire A (B) come un sottotipo e agganciarci i suddetti ruoli.

How to query all the persons we have?



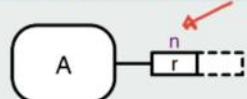
## Aggiunta dei vincoli finali e verifica finale

Si aggiungono i vincoli di frequenza, cioè quante volte un ruolo può essere giocato da una entità.

Local constraints specifying how many times an entity of a given type can/must play an attached role. Compound transactions are typically needed to populate the corresponding fact tables (i.e., more facts to be added at a time)

- **Frequency constraint:** positive integer  $n$  on top of a role - each entity of the connected type must appear *exactly n times* in the corresponding fact table.
  - If  $n = 1$ , it corresponds to an UC → the bar must be used.
- **Frequency ranges:** " $\leq n$ " (with  $n \geq 2$ ) - *at most n*; " $\geq n$ " (with  $n \geq 1$ ) - *at least n* (range can be also used when needed).
- **Compound frequency constraint** spans two or more roles.

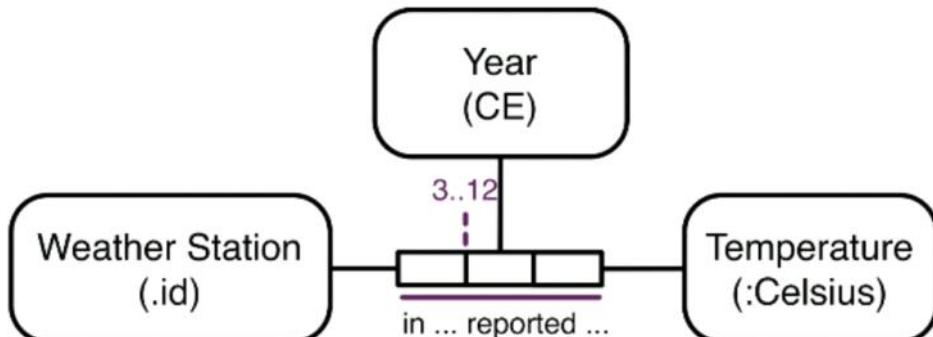
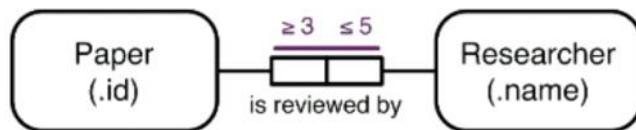
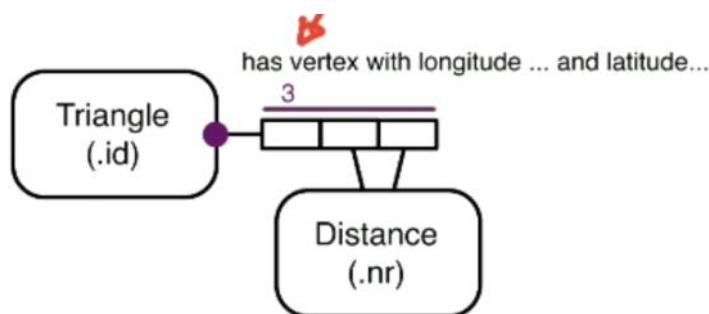
### Locality principle

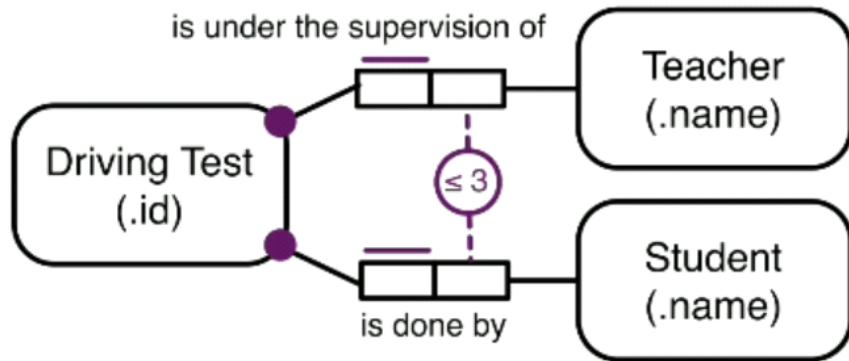


For each information base state: each member of  $pop(r)$

occurs there **exactly n times**.

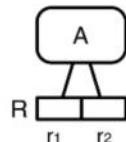
I.e., if an instance of A plays role r, it must do so **n times**.





Vincoli ad anello

- Prototypical setting:



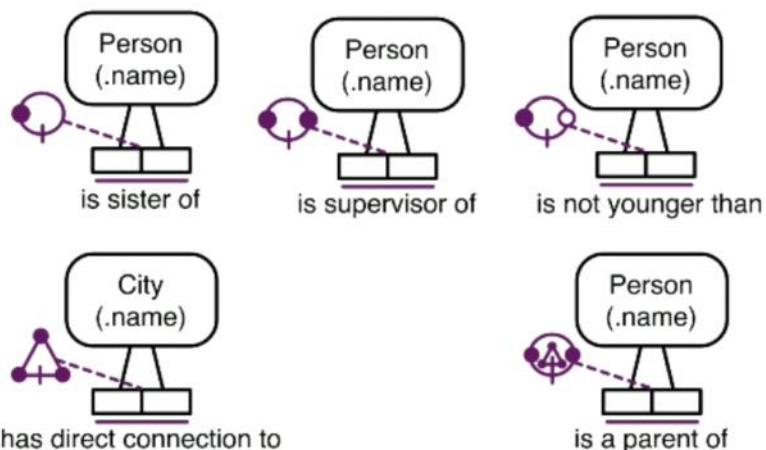
- **R** is a relation, that could enjoy different properties:

- **Reflexivity**:  $\forall x.xRx$ .
- **Symmetry**:  $\forall x,y.xRy \rightarrow yRx$ .
- **Transitivity**:  $\forall x,y,z.xRy \wedge yRz \rightarrow xRz$ .

where all variables  $x, y, z$  range over  $\text{pop}(r_1) \cup \text{pop}(r_2)$ .

- These properties are “positive”: they state how to derive *additional* information.
- Their negations can be thought as *constraints*, and are represented using stylized graphical icons remembering their semantics.
- Positive properties can also be represented by removing the “bar” attached to the icons.

- **Irreflexivity**:  $\forall x.\neg xRx$ .
- **Asymmetry**:  $\forall x,y.xRy \rightarrow \neg yRx$ .
  - Obviously stronger than irreflexivity.
- **Antisymmetry**:  $\forall x,y.x \neq y \wedge xRy \rightarrow \neg yRx$ .
  - Is asymmetry without irreflexivity.
- **Intransitivity**:  $\forall x,y,z.xRy \wedge yRz \rightarrow \neg xRz$ .
- Can be also combined.

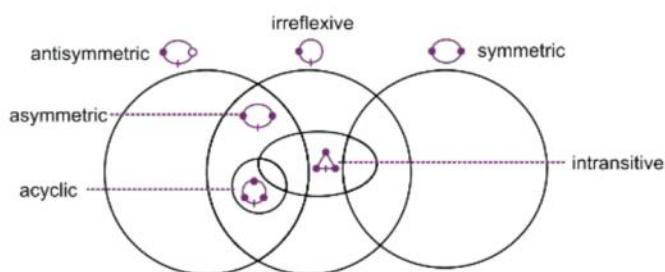


Vincoli sulla ciclicità

- What about cycles in the ring?
- There are situations in which we want to exclude that the entity is indirectly connected to itself through the repeated application of the ring relationship.
  - E.g., a Person cannot be ancestor of herself.
- In this case, we generalize asymmetry to **acyclicity**
- Can be combined with other ring constraints, e.g., with intransitivity.
- It is an expensive constraint to be checked, because it is recursive (transitive closure!)

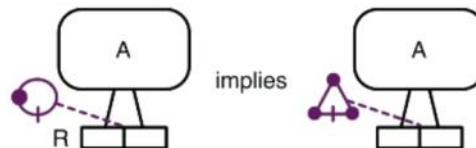


Combination of ring constraints can easily lead to redundancy or inconsistency.



The same holds when considering the combination with other constraints.

- Exclusion constraint implies asymmetry and in turn irreflexivity.
- Irreflexivity and functionality imply together intransitivity.



#### Controlli finali

Si effettuano sulla correttezza e quanto sia adatto lo schema concettuale

1. **Consistenza interna:** ogni ruolo dello schema è fortemente soddisfacibile (ad es. può essere popolato)
2. **Consistenza esterna:** lo schema concorda con i dati campione e i requisiti
3. **Mancanza di ridondanze:** verificare che nessun fatto elementare compaia più volte (o che la ridondanza sia "controllata"/"sicura").
  - a. Questo include la scelta di conservare la ridondanza nel sistema informativo o nelle viste esterne
4. **Completezza:** verificare se lo schema concettuale "copre" tutti i requisiti originali.
  - a. Effettuare un'analisi sistematica, requisito per requisito.

Iterare fino a quando lo schema non è a posto.

# Relational Mapping - 28 Ott - 3 Nov

martedì 24 novembre 2020 12:02

Uno schema concettuale serve per salvare, aggiornare e interrogare le informazioni rilevanti di un dominio.

Codd è tra i primi a mettere in evidenza la correlazione tra i database e la logica del prim'ordine.  
Lo schema dei database è costituito da relazioni, cioè tavole con un nome associato, e un insieme di attributi (colonne della tabella). Gli attributi spaziano in un dominio dei dati.

I dati sono sotto forma di tuple/record. Esistono anche vincoli e regole di derivazione.

Adesso sono supportate semantiche e tipi definiti (anche se spesso non sono supportati da altri tool)

Adottiamo la notazione degli schemi orizzontale e verticale.

I vincoli di unicità: con le chiavi.

- Layout orizzontale: sottolineate
- U<sub>k</sub> (con k → kesima chiave)

La chiave primaria è doppiamente sottolineata o sottolineata una volta nello schema verticale.

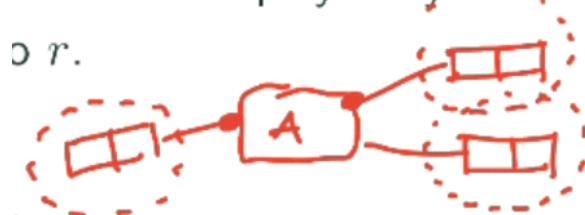
Employee	
PK	<u>empNr</u>
U1	empName
U1	deptCode
	gender
	salary
	tax

Vincoli di obbligatorietà: si possono indicare come opzionali alcuni valori che potranno ottenere valori NULL.

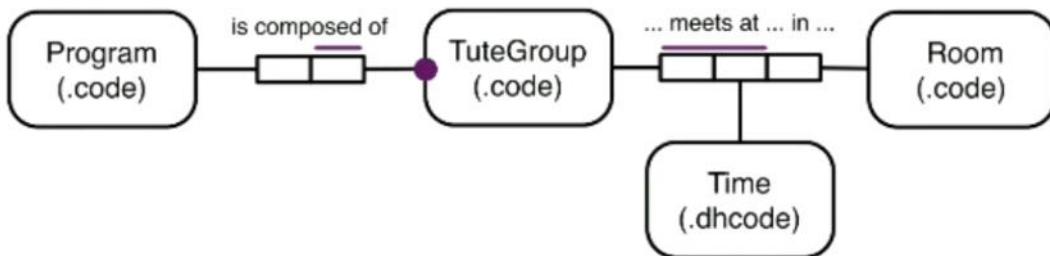
- Parentesi quadre per le opzionali
- Grassetto per le obbligatorie o [ ]

Nel diagramma ORM il principio da cui si è partiti è il concetto di fatto elementare, che non è però quello di base dello schema relazionale.

Cercheremo di introdurre una tavola per ogni relazione, per evitare la ridondanza di informazioni.



Si possono usare delle chiavi esterne per alcuni problemi.

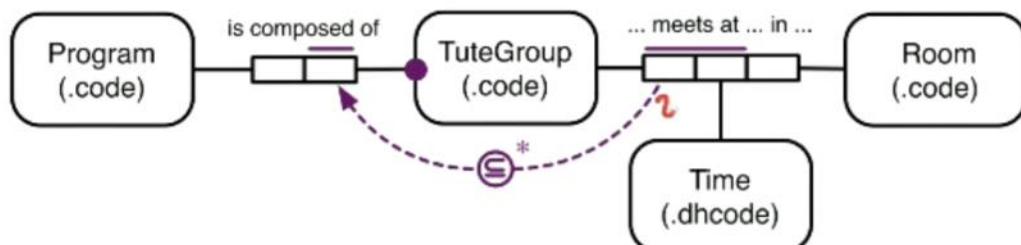


TuteGroup( tuteCode, progCode )

Meets( tuteCode, timeDHCode, roomCode )

TuteGroup	Meets
PK	PK
<u>tuteCode</u>	<u>tuteCode</u>
progCode	<u>timeDHCode</u>
	<u>roomCode</u>

Quindi la pop(composedOf) include la popolazione di meets at.



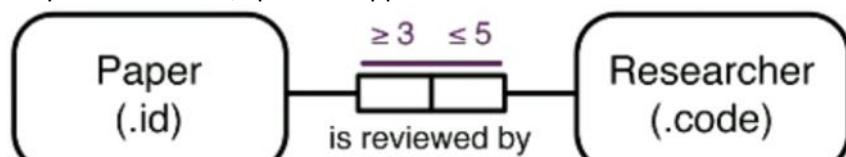
TuteGroup( tuteCode, progCode )



Meets( tuteCode, timeDHCode, roomCode )

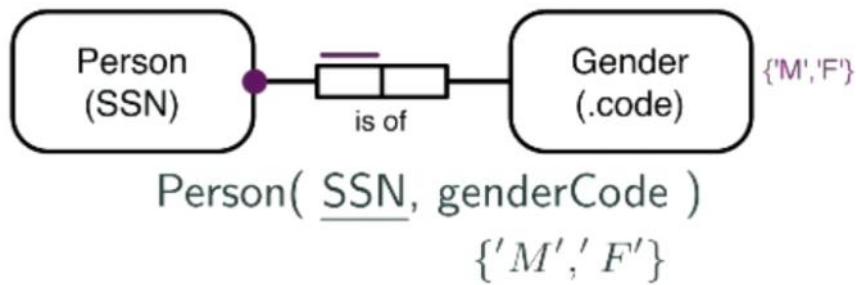
TuteGroup	Meets
PK	PK, FK1
<u>tuteCode</u>	<u>tuteCode</u>
progCode	<u>timeDHCode</u>
	<u>roomCode</u>

In ORM esistono parecchi vincoli, spesso mappabili come vincoli di chiave o referenza.



Reviewed( paperId, resCode )

$$\geq 3 \quad \leq 5$$



Alcuni vincoli sono codificabili come vincoli SQL:

```
check(not exists
      (select resCode from Reviewed
       group by resCode having count(*) > 5))
```

Per le regole di derivazione ci sono:

- Viste
- Colonne generate (tramite procedure)
- Trigger (che aggiungono valori in seguito ad un update)
- Procedure

Bisogna cercare di mantenere l'efficienza dei DB, facendo attenzione a:

- Efficienza (riducendo il numero di tabelle)
- Evitare ridondanze

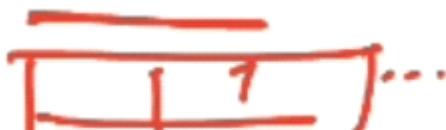
L'algoritmo RMAP cerca di ridurre il numero di conflitti.

Usa una strategia che garantisce l'evitazione delle ridondanze.

Ogni fact type è codificato in una tabella in modo che le sue informazioni appaiano solo una volta.

1. Fact type with a **compound internal UC** → mapped to dedicated table, using the UC as PK.
2. Fact types with **functional roles** on the **same object type** → grouped into a unique table, with the object type's preferred identifier as PK.

Chiave interna:



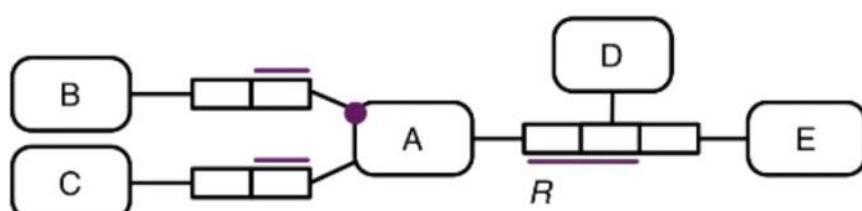
Tutti i fatti elementari di questo tipo sono mappati in una singola tabella con come chiave il vincolo di unicità.

Alla tabella si dà il nome della relazione



Si raggruppano tutti questi ruoli funzionali in un'unica tabella con A come reference e gli altri collegamenti che appaiono o rimangono NULL

In questo caso il nome della tabella è quello dell'object type.



Ci sono due ruoli opzionali, quello con C e quello con D-E.

A determina anche C.

L'altro ruolo è composto e il vincolo di unicità è diviso tra gli object type A e D

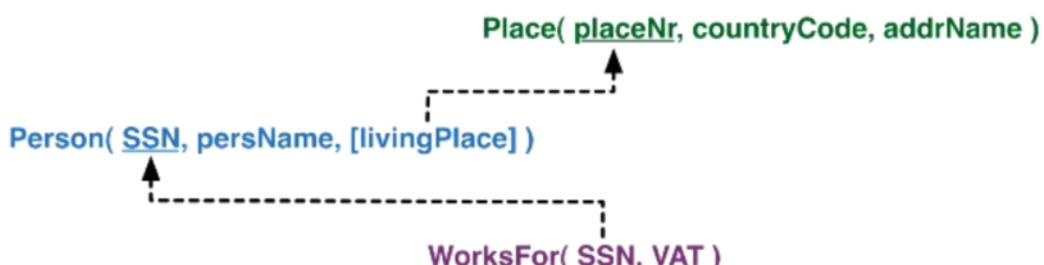
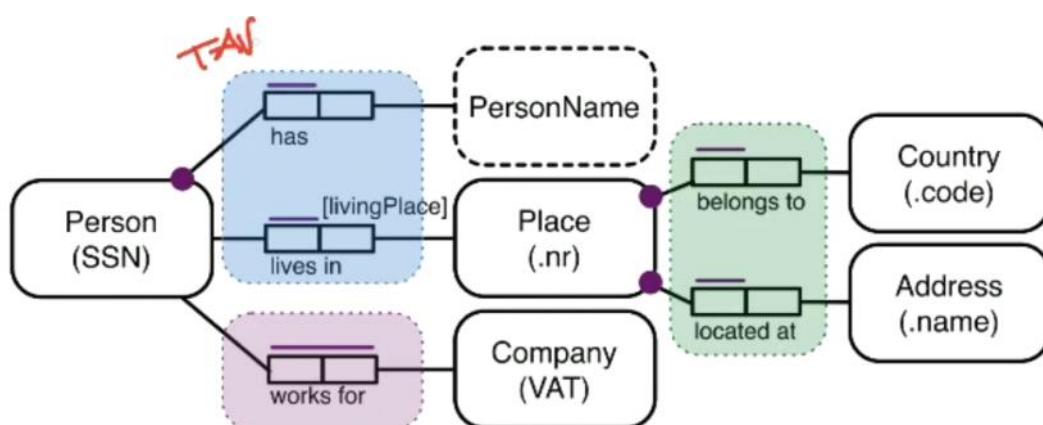
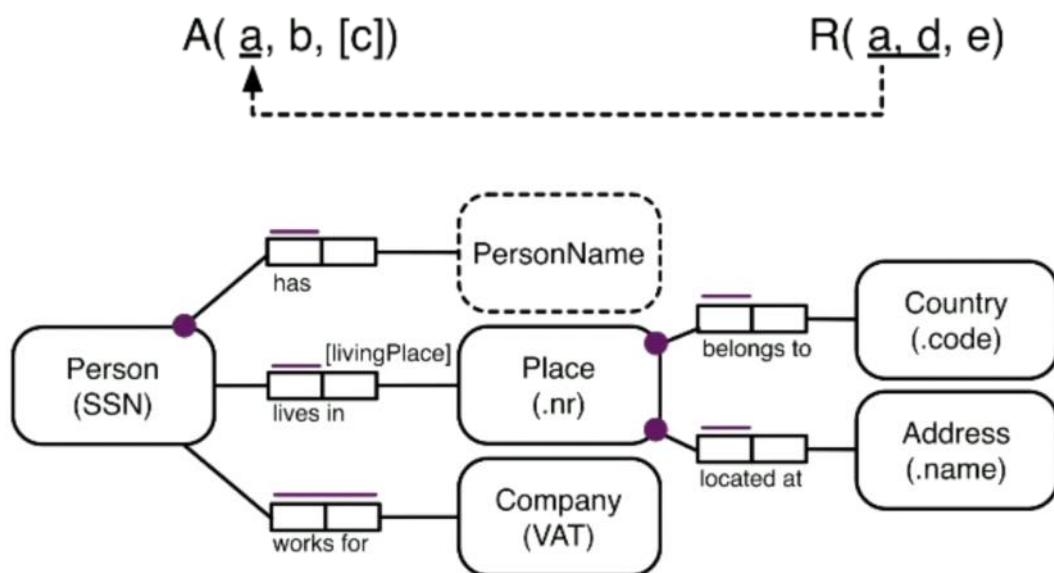
Inoltre B è determinato da A ed è obbligatorio.

a,b,c,d,e sono il set di attributi che rappresentano lo schema di identificazione preferito.

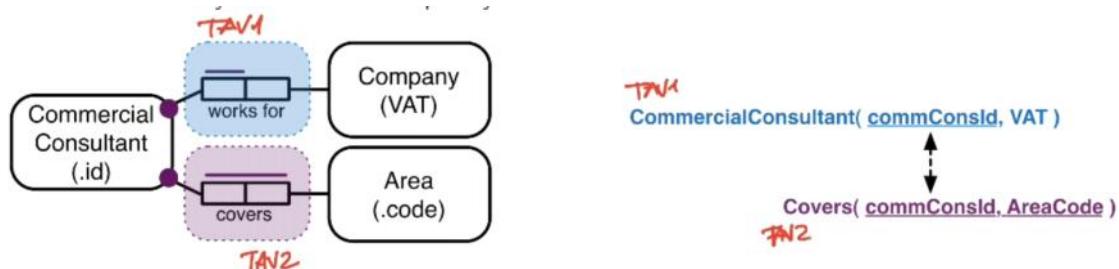
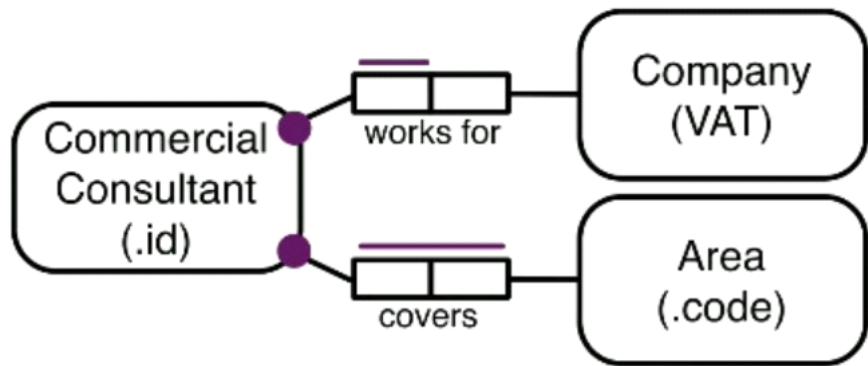
### Case 1

#### Case 2

- Applies to binary fact types with simple UCs all over roles played by the same object type.
- Mandatory dots determine mandatory columns in the grouped table.
- Applies to binary fact types with spanning UCs and to n-ary fact types with  $n > 2$  (why?).
- FKs must be added for those roles attached to object types that play mandatory roles elsewhere ( $\sqsubseteq$ ).



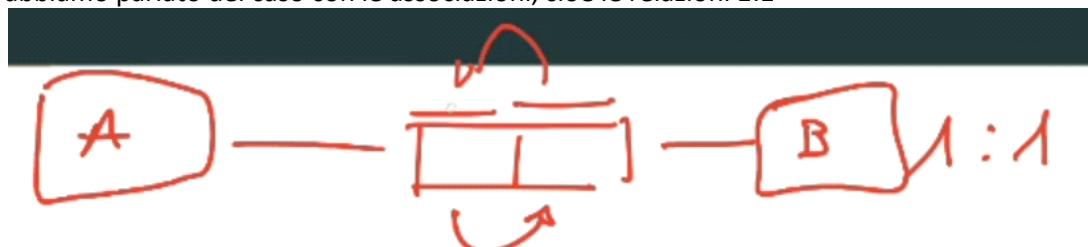
In questo caso sono giocati due ruoli:



Il problema è che abbiamo bisogno di una doppia foreign key! Il vincolo di FK su Covers non è possibile perché è referenziata parte di una chiave.

Questo genere di problemi sono risolvibili con asserzioni, procedure o trigger.

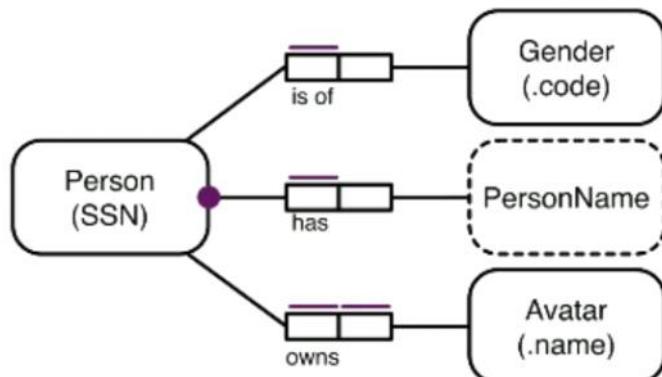
Non abbiamo parlato del caso con le associazioni, cioè le relazioni 1:1



Per scegliere quali dei due object type sarà la base del raggruppamento, è necessario considerare:

- Presence of null values must be minimized.
- Do both object types play other functional roles?
- Are the involved roles mandatory on both sides?

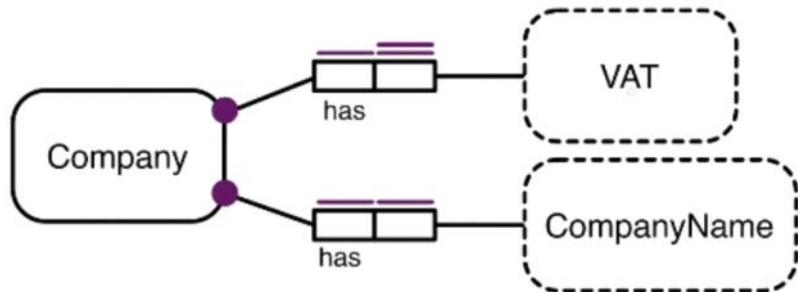
Consideriamo il caso 1:1 quando solo 1 degli endpoint è "complesso" (endpoint = object type)



Data una persona, identifico un Avatar e vice versa.

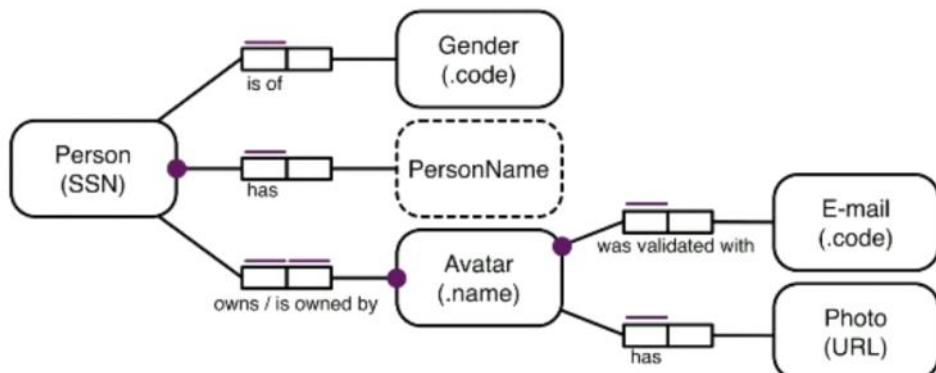
Realizziamo quindi un'unica tabella Person:

Person( SSN, [gender], persName, avName )



• Company( VAT, companyName )

Con la doppia linea indico la chiave preferita.

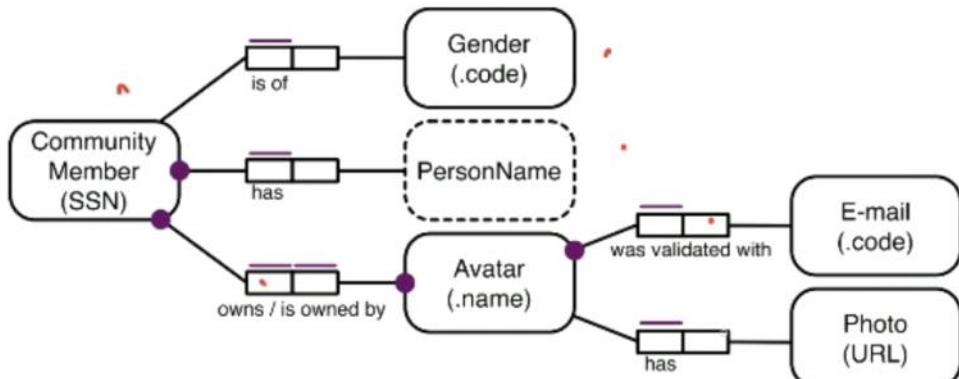


Bisogna raggruppare sul ruolo obbligatorio

Person( SSN, [gender], persName )



Avatar( avName, SSN, eMailCode, [photoURL] )



(1) CommunityMember( SSN, [gender], persName )

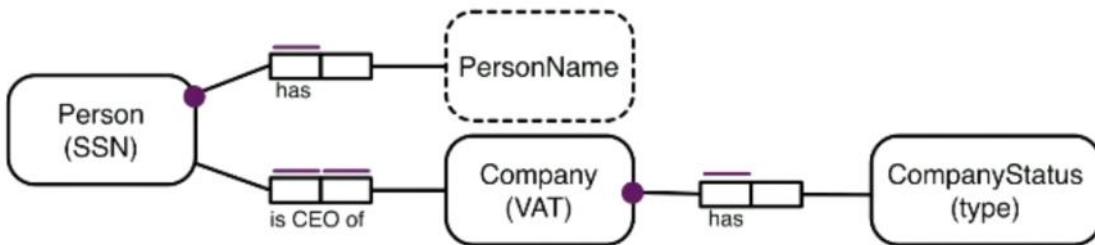


Avatar( avName, SSN, eMailCode, [photoURL] )

(2) CommunityMember( SSN, avName, [gender], persName )

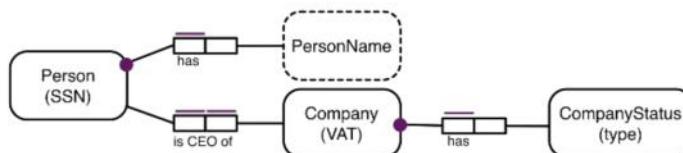


Avatar( avName, eMailCode, [photoURL] )



In questo caso potrei scegliere entrambi i raggruppamenti.

Conviene minimizzare i NULL: bisogna capire semanticamente qual è la scelta che minimizza i null. Se entrambe le soluzioni sono insoddisfacenti, bisognerebbe introdurre una terza tabella che rappresenta il ruolo 1:1



Hp: it is more likely for a Company to have a CEO than for a Person to be CEO of a Company.

Hp: it is likely that both solutions (grouping on Person or on Company) yield many null values.

Person( SSN , persName )  
 ↑  
 Company( VAT, [SSN], ComStatus)

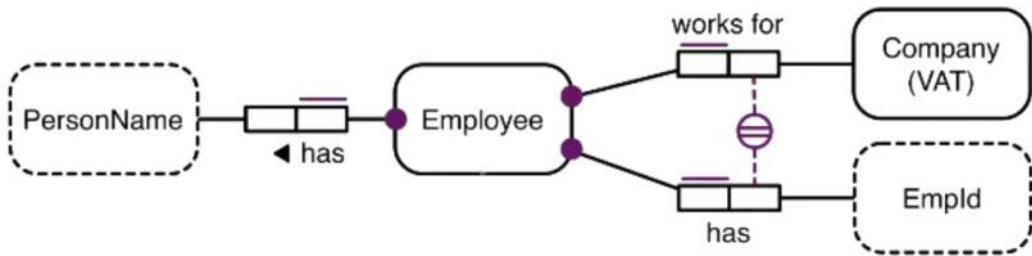
Person( SSN, persName )  
 ↑  
 IsCEOof( SSN, VAT )  
 ↓  
 Company( VAT, ComStatus )

Strategie:

1. **If only one object type in the association has another functional role then group on its side.**
2. **Else if both object types have other functional roles and only one role in the 1:1 association is mandatory then group on its side.**
3. **Else if no object type has another functional role then map 1:1 to a separate table.**
4. **Else the grouping choice is completely delegated to the modeler.**

RMAP: trasforma uno schema ORM in uno schema relazionale.

Vincoli di unicità esterni

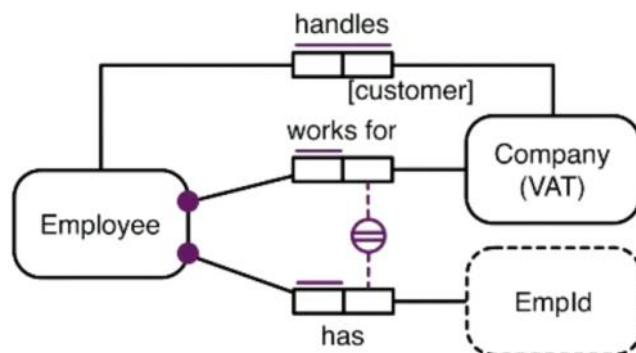


L'entity type ha un external preferred schema.

In questo caso si raggruppano gli oggetti con dei surrogati, senza considerare i predicati coinvolti.

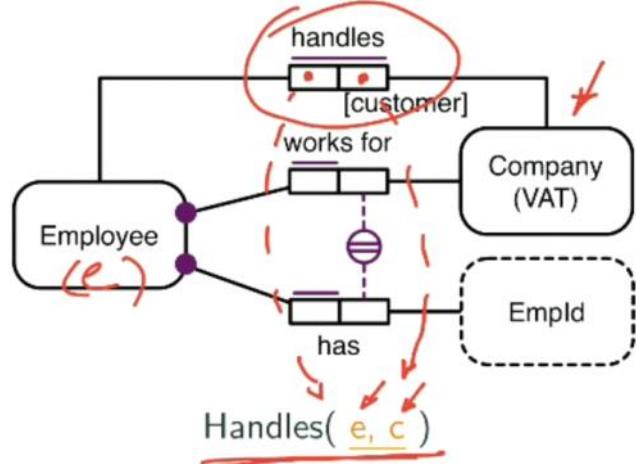
Dopo si espande la chiave primaria usando gli object types coinvolti nello schema esterno.

Employee( empld, VAT, empName )



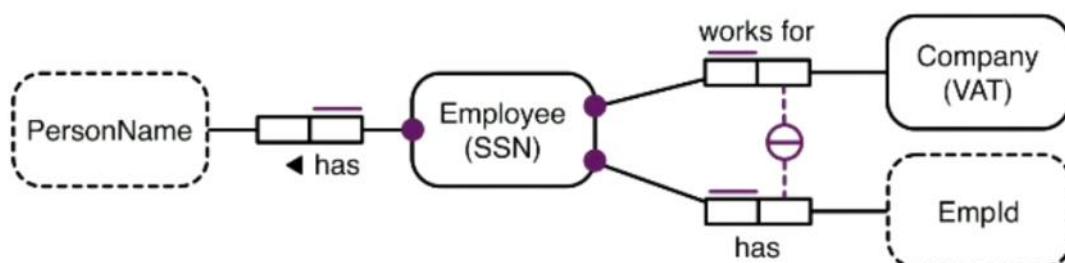
La strategia è mappare l'associazione m:n in una tabella separata, usando dei surrogati.

Poi si espande la chiave primaria utilizzando gli object types coinvolti nello schema esterno.



Handles( empld, VAT, customerVAT )

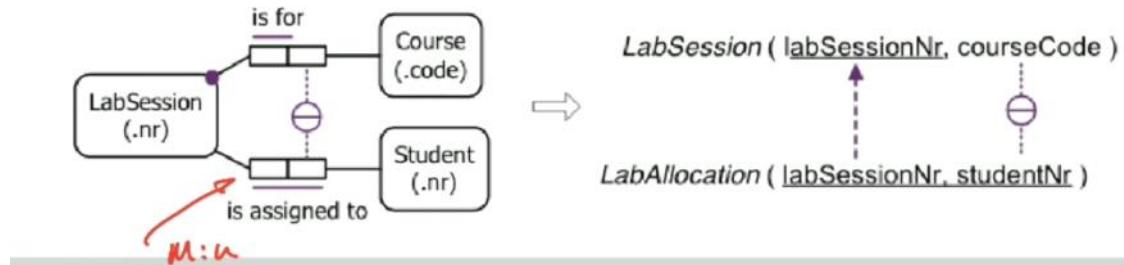
Caso: vincolo esterno di unicità e relazione funzionale.



Si segue il mapping standard e si modella il vincolo di unicità esterno come una chiave.

## Employee( SSN, VAT, emplId, persName )

Caso: Chiave di unicità esterna che coinvolge un fact type m:n

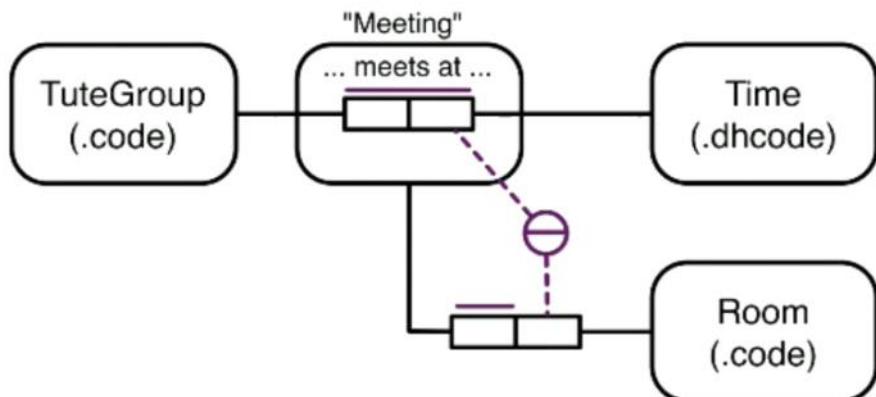


Si segue il mapping standard senza considerare il vincolo di unicità esterno, che verrà poi aggiunto come vincolo tra tabelle.

### Relazioni oggettificate

Procedure:

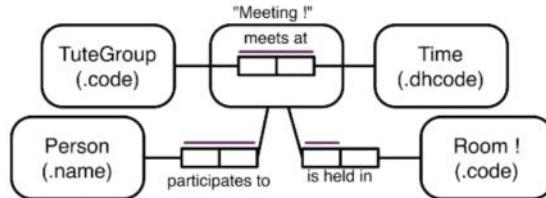
1. Do not consider the identification scheme of the objectified association.
2. Consider the objectified association as a black box.
3. Group fact types in the standard way.
4. Unpack the black box into its component attributes.
5. Deal with fine-grained constraints involving component roles of the objectified association.



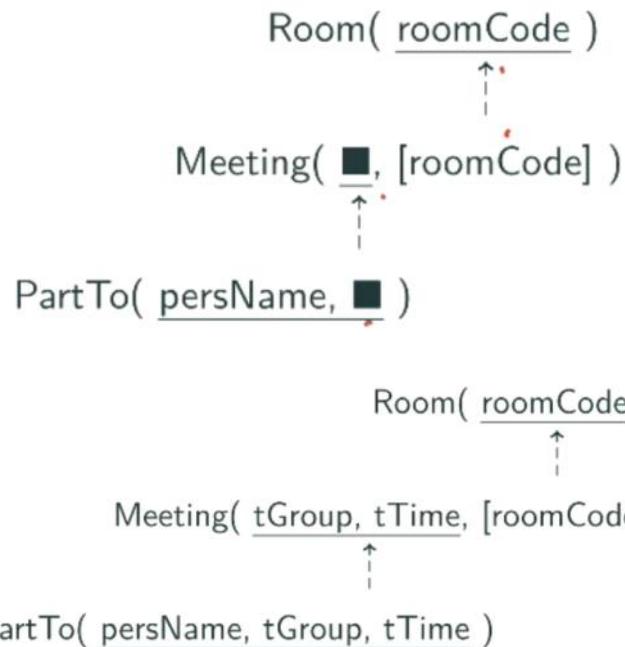
- Consider the objectified association as a black box:  
Meeting ( [■, [roomCode]] )
- Expand:  
Meeting ( tuteCode, meetingTime, [roomCode] )
- Incorporate fine-grained constraints: key meetingTime, [roomCode]

Each independent object type has its own life cycle, hence:

- it must be mapped to a dedicated table with its preferred identifier as PK, together with all fact types in which it plays a functional role;  
(note: they are all optional by construction, can you spot why?)
- every nonfunctional role will have a FK pointing to this table.



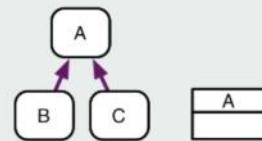
(Quelli indipendenti hanno il punto esclamativo)



**Vincoli di sottotipo:** abbiamo tre soluzioni

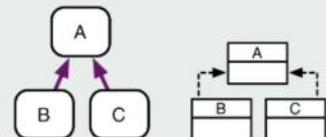
#### Absorption

Subtypes are absorbed back into their top supertype, grouping the fact types as usual and adding the subtyping constraints as textual qualifications.



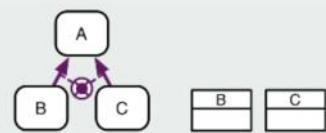
#### Separation

Each object type of the hierarchy is mapped to a separate table. FKs are added from the subtypes tables to the supertype table.



#### Partition

Supertype is removed, replicating the attached information for each subtype.

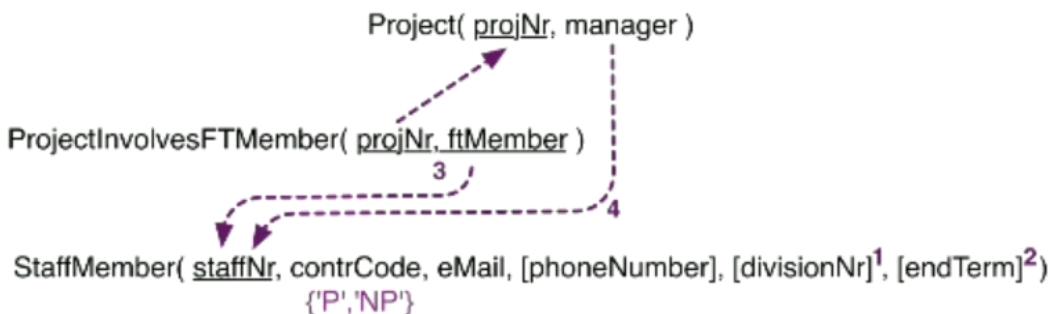
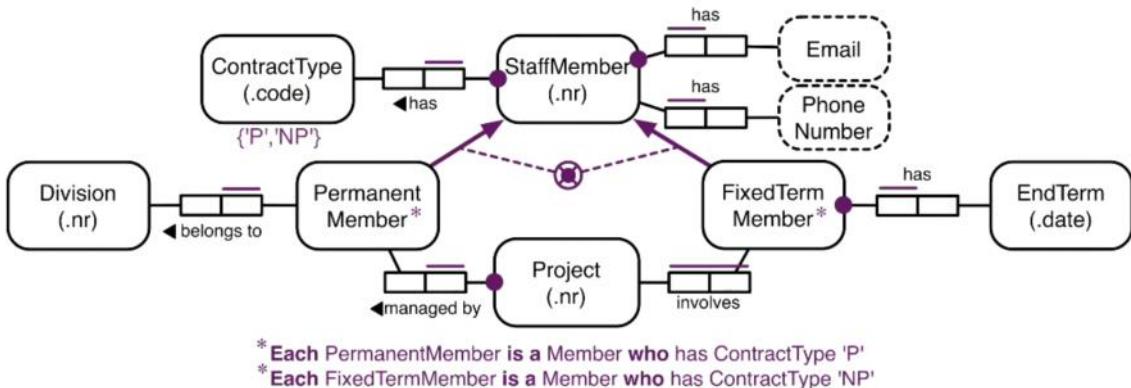


#### Assorbimento

- Viene bene quando c'è una colonna che fa da discriminazione per il tipo. Alcuni casi sono il

ruolo o il genere.

- Tutti i fatti appartenenti ai sottotipi vengono riportati come fatti del tipo padre.
- Sono aggiunti dei vincoli per discriminare in base alla colonna se gli elementi devono esistere obbligatoriamente oppure potrebbero esistere.
- Non sono quindi necessari JOIN => + efficienza, ma potremmo avere parecchi valori NULL che possono portare problemi nelle query.

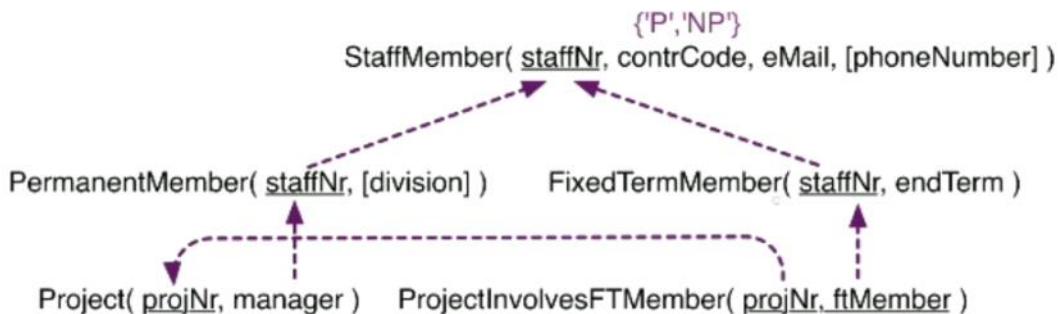


Essendo endTerm del sottotipo, non possiamo indicare che è obbligatoria anche per StaffMember o PermanentMember.

<sup>1</sup>exists only if contrCode = 'P'   <sup>2</sup>exists iff contrCode = 'NP'   <sup>3</sup>only where contrCode = 'NP'   <sup>4</sup>only where contrCode = 'P'

### Separazione

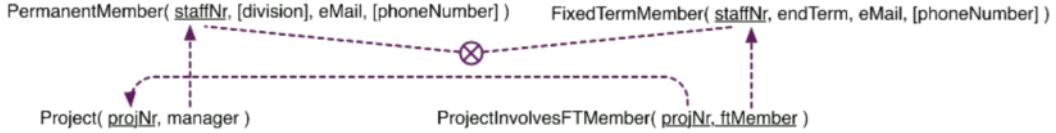
- Obiettivo: **minimizzare** i NULL e rendere più accessibili i sottotipi, a **discapito** dei JOIN necessari per avere le informazioni complete e della lentezza di multiple inserzioni.
- Ogni object type della gerarchia diventa una tavola separata, a patto che sia coinvolta in almeno un ruolo funzionale.
- La chiave primaria deve fare riferimento con una FK alla chiave primaria del supertipo
- Altri casi sono da risolvere con dei vincoli



<sup>1</sup>only where contrCode = 'P'   <sup>2</sup>exactly where contrCode = 'NP'

### Partizione:

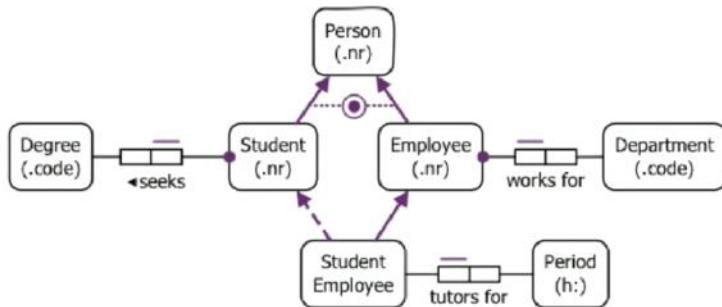
- Quando i sottotipi sono esclusivi o formano una partizione
- Si mappano solo i sottotipi
- I ruoli dei supertipi sono replicati nei sottotipi
- **Vantaggi:** minimizzare i NULL, query rapide.
- **Svantaggi:** sono necessarie delle unioni per interrogare le superclassi (è suggerita una vista)



```

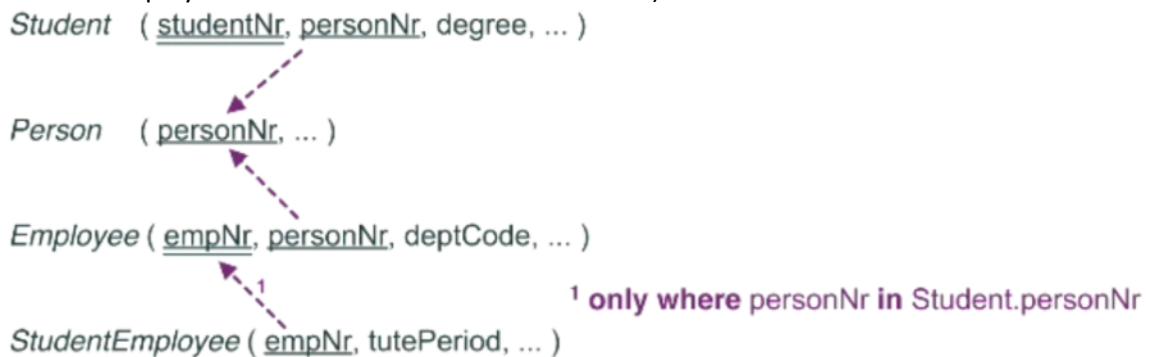
*StaffMember( staffNr ) =
  PermanentMember( staffNr ) union FixedTermMember( staffNr )
  
```

Come si mappano diversi schemi di identificazione?



- `Student`, `Employee`, `Person` mapped to separate tables.
- `Student` and `Employee` add a reference to `Person` in their total tables.
- `StudentEmployee` inherits `Employee`'s preferred id → two choices:
  - Separation strategy. In this case the functional role "tutors for" is considered mandatory in the table, because it is the only distinctive feature of `StudentEmployee`.
  - Absorption strategy. Functional role "tutors for" added as optional column in `Employee` table. Discriminator column could be introduced.

(eredito l'employee number ma non lo student number)



### Procedura di RMAP

0. Indicate any absorption-overrides (separation or partition) for subtypes, and absorb other subtypes into their top supertype.

Mentally erase all explicit preferred identification schemes, treating compositely identified object types as "black boxes".

1. Map each fact type with a compound UC to a separate table.
2. Fact types with functional roles attached to the same object type are grouped into the same table, keyed on the object type's identifier.
3. Map 1:1 cases to a single table, generally favoring fewer nulls.
4. Map each independent object type with no functional roles to a separate table.
5. Unpack each "black box column" into its component attributes.
6. Map all other constraints and derivation rules.

In case of absorption, subtype constraints on functional roles map to qualified optional columns, and those on nonfunctional roles map to qualified subset constraints.

Nonfunctional roles of independent object types map to column sequences that reference the independent table.

# Business Process Management - 10 Nov

domenica 10 gennaio 2021 18:48

L'universo del discorso si evolve.

Come descrivere in che modo si evolve l'information base? Che risultati hanno?  
Spesso chiamato "Processo Aziendale"

La gestione dell'azienda basata sui processi è diventata fondamentale.

Serve una descrizione dei processi svolti in un'azienda e quali necessità soddisfano, oltre a come sono organizzati.

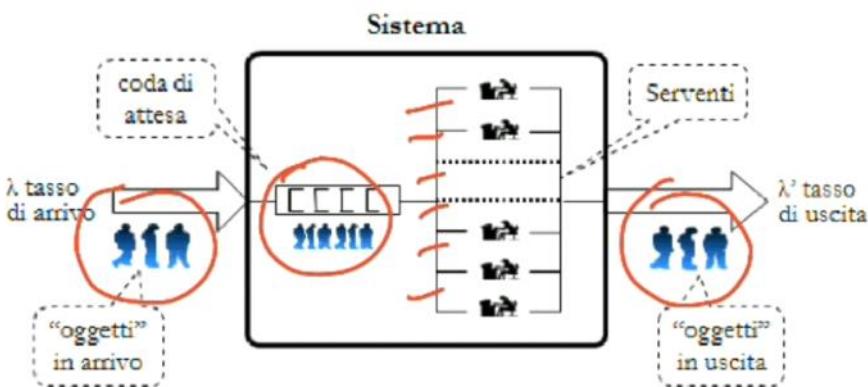
Se so come funziona un processo nella mia azienda, so anche modificarlo facilmente per poter soddisfare dei nuovi requisiti (posso conoscere costi, necessità e requisiti). Ristrutturare i processi è un requisito continuo.

I cambiamenti possono essere **esterni o interni**.

Un'organizzazione è un sistema complesso, distribuito su un territorio e su più agenti e che subisce frequentemente effetti da agenti esterni che lo mutano (complesso, distribuito, aleatorio).

Approccio sistemico

- Il sistema è considerato come una scatola chiusa, approssimata mediante una coda e dei serventi.
- Un modello del sistema piuttosto semplice basato sulla teoria delle code
- È impensabile di estendere l'analisi al processo interno modellando le attività



©A. Di Leva. La gestione dell'azienda basata sui processi e i sistemi informativi aziendali. Celid, 2014.

Leggi di Little:  $N = \lambda * \Lambda$ ,  $L = \lambda * \Lambda_c$ , dove  $N = L + s$  è il numero di oggetti nel sistema,  $S$  è il numero di serventi,  $L$  è il numero di oggetti in coda,  $\lambda$  è il tasso di arrivo,  $\Lambda = \Lambda_c + 1/\mu$  è il tempo di ciclo,  $\Lambda_c$  è il tempo di attesa,  $\mu$  è il tasso di servizio.

### Leggi di Little per la valutazione

Un'agenzia di assicurazioni tratta 120000 pratiche in un anno e in media ci sono 600 pratiche in agenzia in un qualsiasi momento. Qual è il tempo di ciclo se in un anno consideriamo 50 settimane lavorative?

- $\lambda = 120000/50 = 240$  pratiche per settimana, tasso di arrivo
- Da  $N = \lambda * \Lambda$  abbiamo  $\Lambda = 600/240 = 2.5$  settimane, tempo di ciclo
- Il gestore del processo può focalizzarsi su due parametri e trovare il terzo con le leggi di Little.
- Un sistema è stabile quando  $\lambda \sim \lambda'$ , per ridurre le code si deve ridurre il tempo di ciclo

L'approccio sistematico non è proponibile come strumento generale con cui modellare e studiare i processi aziendali, perché si considerano solo le code.

È molto meglio utilizzare l'organizzazione funzionale:

- l'azienda si struttura per raggiungere i propri obiettivi
- l'organizzazione funzionale raggruppa tutte le attività simili all'interno dell'azienda.

Le **funzioni** sono aggregazioni di uomini e mezzi necessari per lo stesso tipo di risorse e di tecnologie, e sono raggruppate in una **unità organizzativa** sotto un'unica responsabilità.

Produzione, qualità, marketing, ricerca e sviluppo...

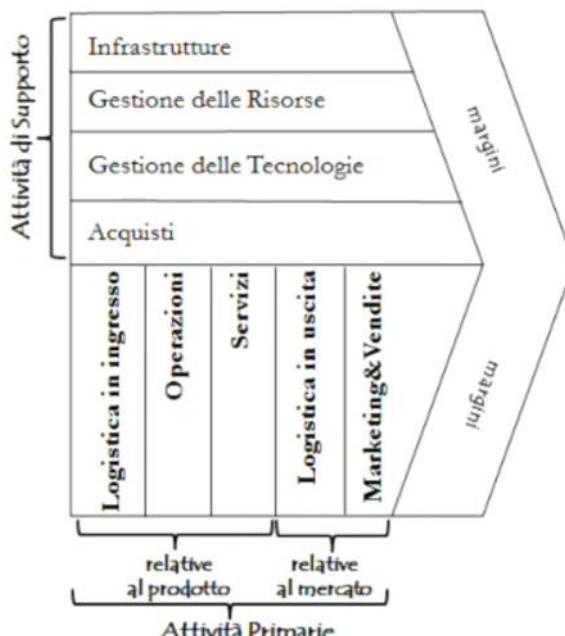
Il fine di ogni attività svolta dall'azienda è quello di creare dei prodotti che vengano incontro ai desideri dei clienti e creino valore per il cliente.

L'azienda è vista come un sistema di attività generatrici di valore.

Catena del valore di Porter

- VA aggiungono valore
- NVA non aggiungono valore
- VAA aggiungono valore per l'azienda.

Le VA e le VAA si dividono in attività primarie e di supporto.



Tutte cercano di accrescere i margini.

## Svantaggi:

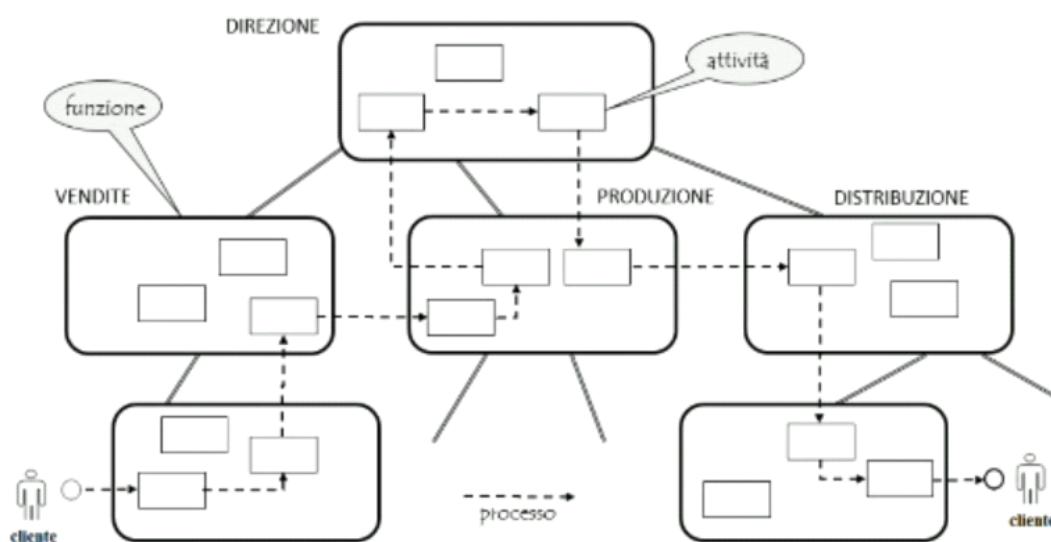
- La catena del valore rappresenta una struttura statica, gerarchica, con linee di comando e controllo rigide.
- Non permette di individuare la responsabilità verso il cliente finale
- Visione ristretta, focalizzata all'interno
- Comunicazione con le altre sezioni non efficiente

È necessaria una visione dell'azienda più ampia di quella fornita dall'architettura funzionale.

## Processi

Le funzioni aziendali, associate ad una unità organizzativa dell'azienda, sono rappresentate da rettangoli smussati contenenti le attività gestite dalla funzione stessa.

Il processo "riduce" le attività a partire da un evento esterno generato da un cliente e può "attraversare" diverse funzioni aziendali per tornare al cliente



## Processo Aziendale

Un processo aziendale come un insieme di attività (controlli ed azioni) tra loro interrelate per realizzare un risultato definito e misurabile, il prodotto o servizio che trasferisce valore al frutto (cliente) del prodotto o servizio stesso

- Risponde ad un evento esterno eseguendo delle attività
- Deve raggiungere certi obiettivi
- Deve fornire un risultato
- Consuma delle risorse aziendali
- Deve soddisfare i requisiti dei clienti
- È vincolato da regole interne ed esterne
- Ha generalmente un responsabile (process owner)

## Business Process Management

È un insieme di metodi, tecniche e strumenti per gestire nel modo migliore l'azienda.

## Obiettivi del BPM

- Ottimizzare i processi "interni" dell'azienda
- Facilitare l'allineamento dei processi gestionali e produttivi agli obiettivi strategici dell'azienda
- Migliorare la flessibilità dell'organizzazione aziendale in modo da rispondere "rapidamente" ai cambiamenti di scenario
- Facilitare il flusso del processo fra i vari partecipanti cercando di eliminare le attese

- Automatizzare il flusso di controllo e dei documenti al proprio interno
- Analizzare, modellare e misurare il processo (indicatori di efficienza)
- Utilizzare il modello per fare delle previsioni (analisi "what if")
- Migliorare l'ambiente di lavoro eliminando i passi ripetitivi mediante uso "intelligente" delle tecnologie
- Ripensare i processi avendo come obiettivo il miglioramento delle qualità e dell'efficienza

### Taylor (spilli)

Applica il metodo scientifico. Decompone il lavoro assegnandolo a chi di dovere. La gestione è fine ma non c'è concentrazione sull'organizzazione.

- Starting point: organizations are not based on solid principles.
- It is impossible to *measure* quality and efficiency of the produced outputs.
- Idea: apply the *scientific method* to organization management.
  - Decomposition of work into elementary units.
  - Scientific analysis of each element of work, fixing its boundaries and rules.
  - Train and teach workers according to the identified rules.
  - Assure that work is conducted according to the rules.
  - Divide and conquer: management plans/controls, workers perform.

### Results:

- Fine-granular division of labour.
- Emphasis on isolated activities, not on their coordination.

### Hammer & Champy

Si focalizzano sul goal di un processo. Non si individuano solo le competenze unitarie per compiere un certo lavoro, ma è importante avere una visione di insieme del processo per capire come reingegnerizzarlo al fine di migliorare una certa condizione. Non è una pratica continuativa.

Alla organizzazione funzionale viene associata un'organizzazione a processi. Si descrive l'organizzazione delle attività per renderle più efficienti.

### Davenport 1992

Vede i processi di business come una serie di compiti logicamente correlati ed eseguiti per raggiungere un goal prefissato per un particolare cliente o mercato.

### Hammer & Champy 1993

Li vede come una serie di attività che richiedono uno o più tipi di input e creano un output che è di valore al cliente

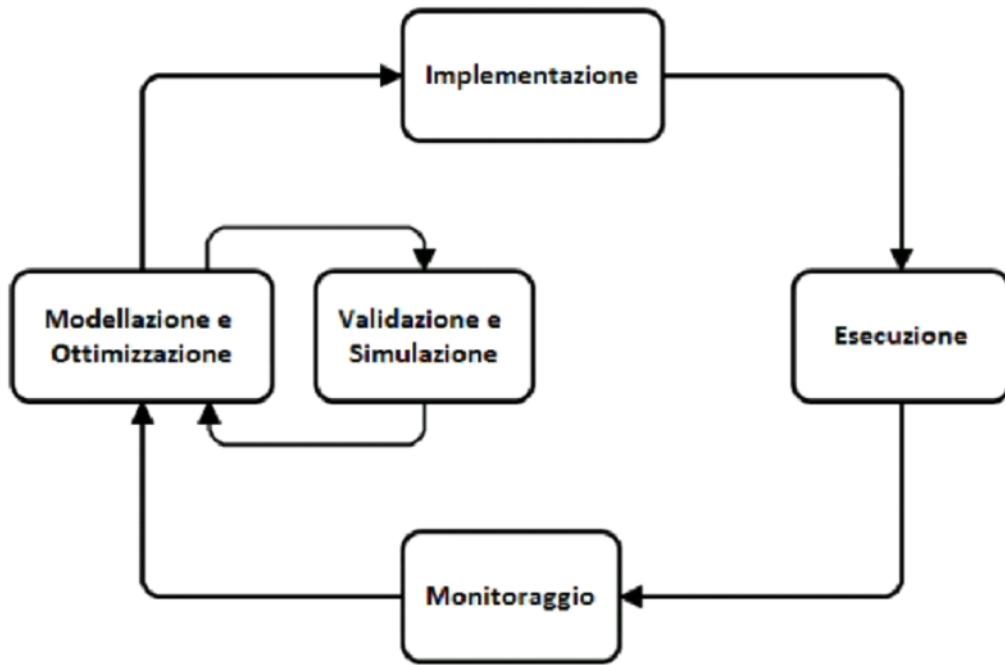
### Weske 2011

Un PB consiste in una serie di attività che sono effettuate in coordinazione in un ambiente organizzato e tecnico. Queste attività assieme realizzano un **business goal**. Ogni processo di business è attuato da una singola organizzazione, ma può interagire con i PB di altre organizzazioni.

### Il punto fondamentale diventa il BPM Lifecycle:

- Si continua a valutare e monitorare un processo
- Si continua a migliorare il processo

Un BPM (Weske) include concetti, metodi e tecniche che supportano la progettazione, l'amministrazione, la configurazione, l'attuazione e l'analisi di processi di business.



Fasi	Compiti	Attori
<u>Modellazione e Ottimizzazione</u>	- costruire il modello - definire gli indicatori - ottimizzare il modello attraverso simulazioni e analisi "what if"	analista
<u>Validazione e Simulazione</u>	- validare il modello attraverso l'animazione e la simulazione - verificare il modello sugli indicatori attuali	analista
<u>Implementazione</u>	- tradurre il modello sul motore di workflow	sviluppatore
<u>Esecuzione</u>	- eseguire il workflow	gestore
<u>Monitoraggio</u>	- controllare gli indicatori - identificare le possibilità di miglioramento - decidere la ristrutturazione	analista e gestore

Tipi di business process:

- Organizzativi e Operazionali:
  - o **Organizzativi**: BP ad alto livello che descrivono input, output, risultati attesi e dipendenze da altri BP organizzativi (con una granularità alta)
  - o Un BP organizzativo corrisponde a diversi BP **operazionali**, che rendono le attività e le relazioni esplicite.
- Intra e Inter-organizzativi:
  - o I BP **intra-organizzativi** sono interni alla compagnia e descrivono attività rilevanti che sono dirette internamente per produrre un bene o consegnare un servizio.
  - o I BP **inter-organizzativi** descrivono le interazioni tra multipli processi che vengono eseguiti in diverse organizzazioni che collaborano, in termini di **coreografie**
- **Grado di automazione** e presenza di attività umane
- **Gradi di ripetitività**: è importante verificare se la modellazione esplicita del business process vale davvero lo sforzo
- Grado di strutturatezza: varia da business process predicibili e ripetitivi (**production workflow**) a business process impredicibili e adattivi, con attività **ad-hoc** e la creatività degli esperti (**knowledge-intensive**)

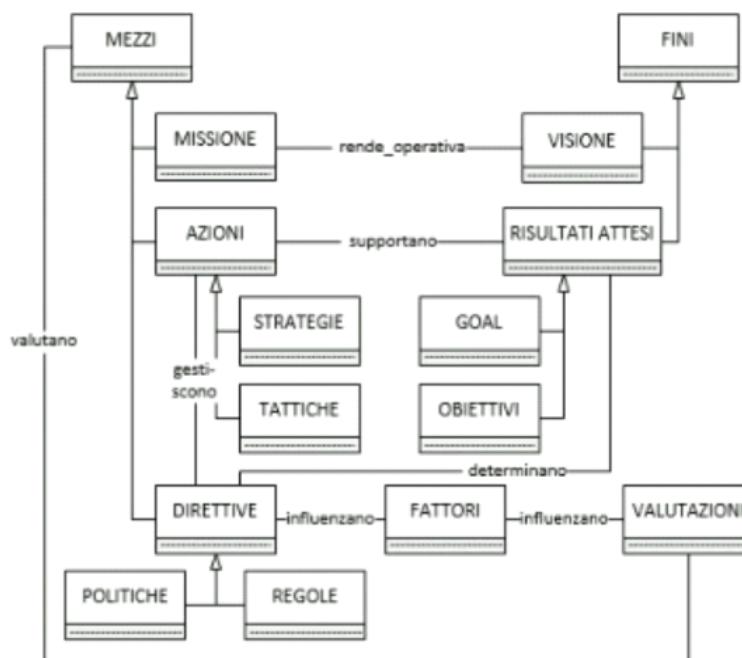
Per gli schemi, si distinguono tra livelli intensionali ed estensionali:

- Estensionali: si concentrano su una specifica esecuzione del processo, cioè una **istanza del**

### processo o un caso.

- Un caso è un'entità prodotta come tangibile (casa, auto, computer) o astratta (prestito, richiesta di risarcimento dell'assicurazione, servizio) e viene manipolata ed evoluta da attività correlate.
- Ogni caso ha un inizio e prima o poi una fine. Ha una traccia costituita dagli eventi che marcano l'esecuzione delle attività di risorse, utilizzando dati specifici in momenti precisi.
- Ogni caso è diverso dagli altri (idealmente)
- Il numero di "procedure" (ad esempio di diverse sequenze effettivamente scelte per evolvere i casi) è idealmente minore del numero di casi che deve essere gestito.
- Livello intensionale: combina tutte le procedure assieme in un'unica blueprint, chiamata **process model**, **process schema** o semplicemente **process**.

### Il modello strategico



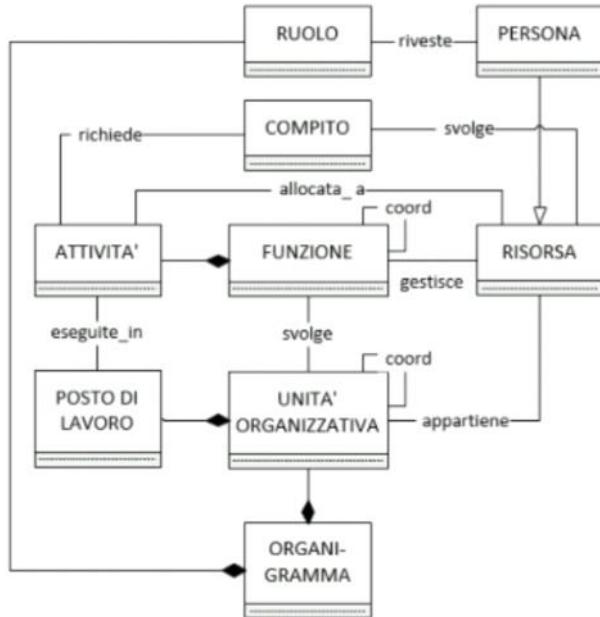
La **strategia aziendale** costituisce un punto di partenza per ogni iniziativa di miglioramento.

Esiste il **Business Motivation Model**:

- Fini: cosa un'azienda vuol essere
- Mezzi: come l'azienda intende raggiungere un suo fine
- Valutazioni: cosa o chi valuta i mezzi usati rispetto ai fini che si intendono raggiungere
- Fattori: un qualsiasi fatto che possa influenzare l'azienda nell'uso dei suoi mezzi e/o nel raggiungimento dei suoi fini

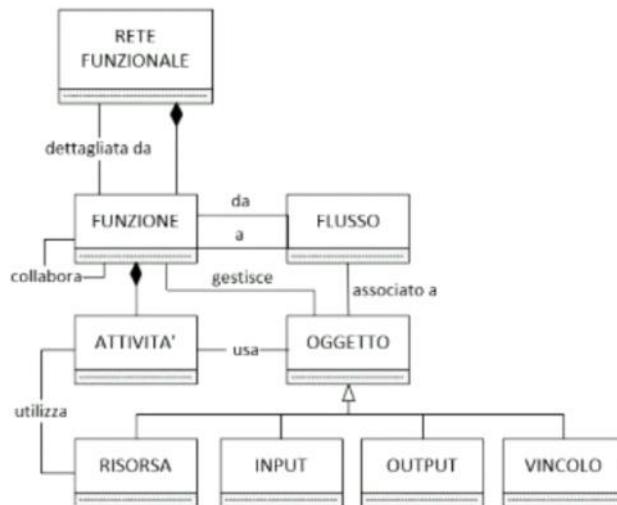
È alla base del **business plan**.

### Il modello organizzativo



Describe la struttura "statica" dell'azienda e le risorse che essa utilizza per il processo (o i processi) in esame.

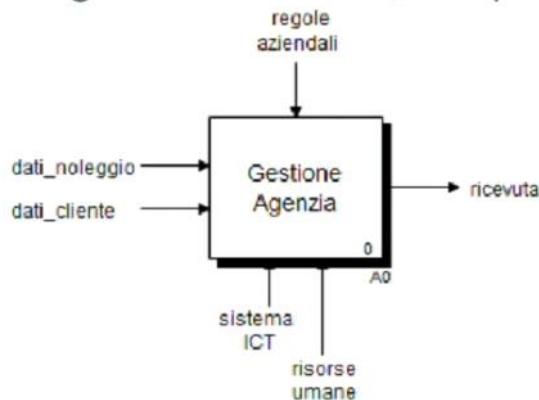
### Il modello funzionale



Describe il comportamento dell'azienda analizzando lo scambio di oggetti (informazioni, documenti, manufatti, prodotti) fra le sue funzioni, tenendo conto delle risorse impegnate e dei vincoli (o regole aziendali) da rispettare nello svolgimento delle attività.



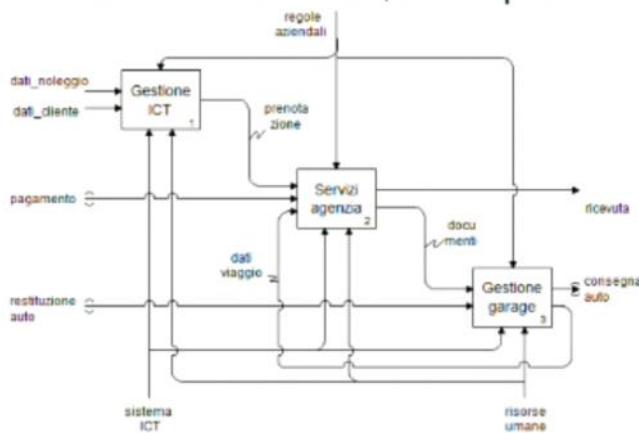
## Diagramma di contesto, esempio



È rappresentato (ad esempio) dal linguaggio **IDEFO**:

- Descrive le reti funzionali e la matrice delle attività per specificare le attività componenti la funzione in esame
- Permette di specificare l'articolazione funzionale di un'azienda mediante una rete funzionale DI (Diagramma IDEF0) organizzata gerarchicamente a livelli

## Rete funzionale, esempio



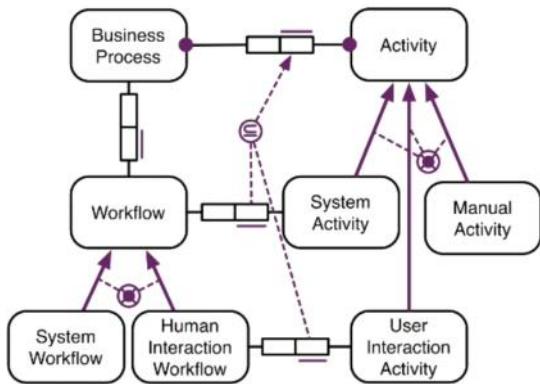
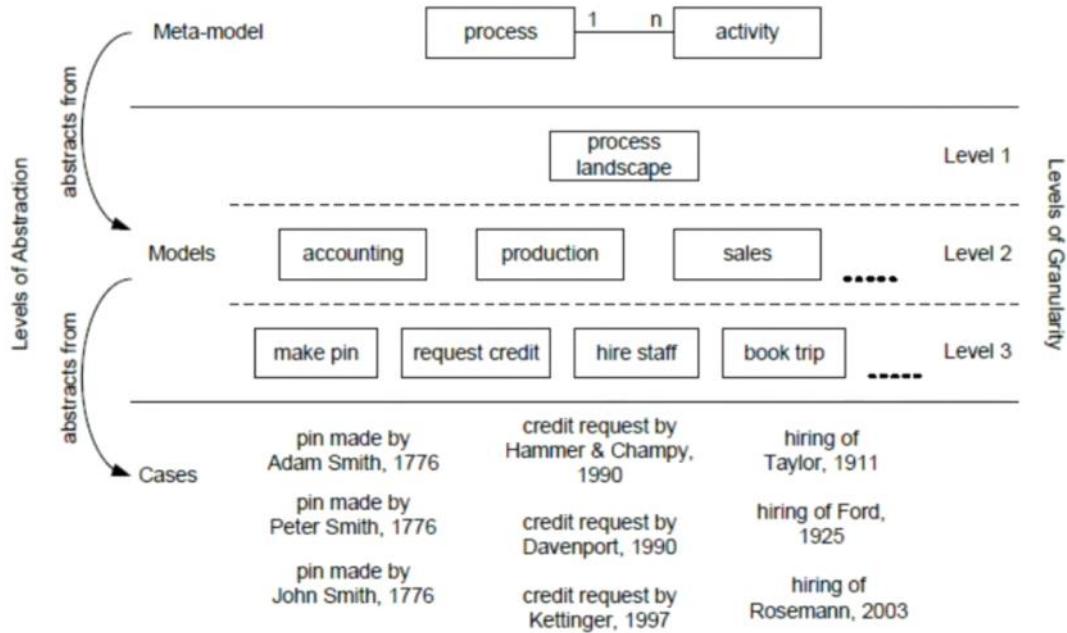
Da questi modelli non è possibile effettivamente risalire al vero comportamento dell'azienda. Bisogna poter specificare come avviene l'esecuzione dei processi e delle attività in azienda prendendo in considerazione un **Modello di Attività** che descrive l'insieme di esecuzioni possibili (istanze) dell'attività in esame. In un certo istante potrebbero esistere (sono quindi "attive") più istanze di un'attività.

Un'istanza di attività si evolve nel tempo e può passare attraverso vari stati: può essere attiva, in attesa di risorse e così via; questa dinamica può essere descritta da un **diagramma stati/transizioni** i cui nodi sono gli stati e gli archi le transizioni che permettono di passare da uno stato all'altro.

Le transizioni sono determinate da **eventi** (che identificano le transizioni stesse) e per ogni stato l'insieme delle transizioni uscenti determina i possibili stati che possono essere raggiunti a partire da quello in esame come conseguenza del verificarsi degli eventi corrispondenti.

Un evento è il verificarsi di un fatto in un certo istante di tempo che provoca una transizione, cioè il passaggio da uno stato a un altro del sistema.

Un **modello dei processi** descrive l'insieme di esecuzioni possibili (istanze) del processo in esame



Activity: unit of work performed towards the achievement of the business goal.

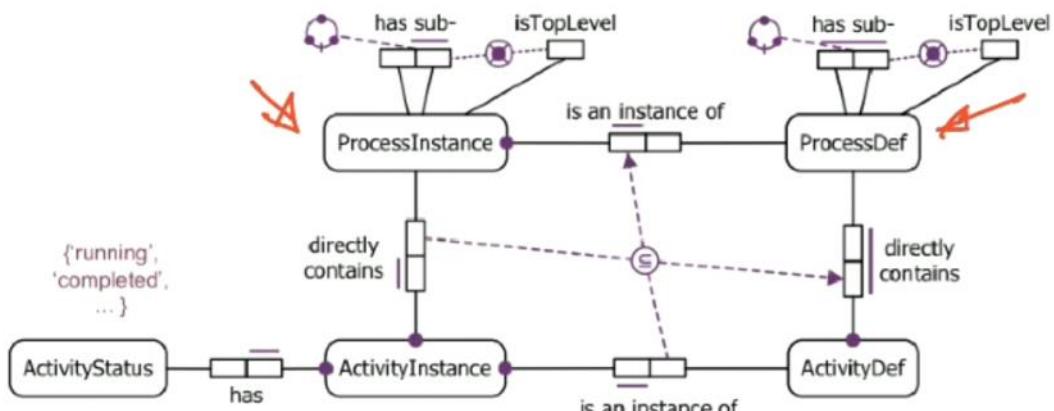
- System activity is executed by the information system (IS), in autonomy.
- Human interaction activity arises when a human interacts with the IS.
- Manual activity is not supported by the IS, unless it is attached to a notification in the form of human interaction activity.

- Certain parts of a BP can be enacted by a workflow, which ensures the desired order for activities and manages the execution of system activities as well as user interaction.

Un BP è un insieme di attività.

Possono essere:

- Attività di sistema
- Attività manuali
- Interazione dell'utente



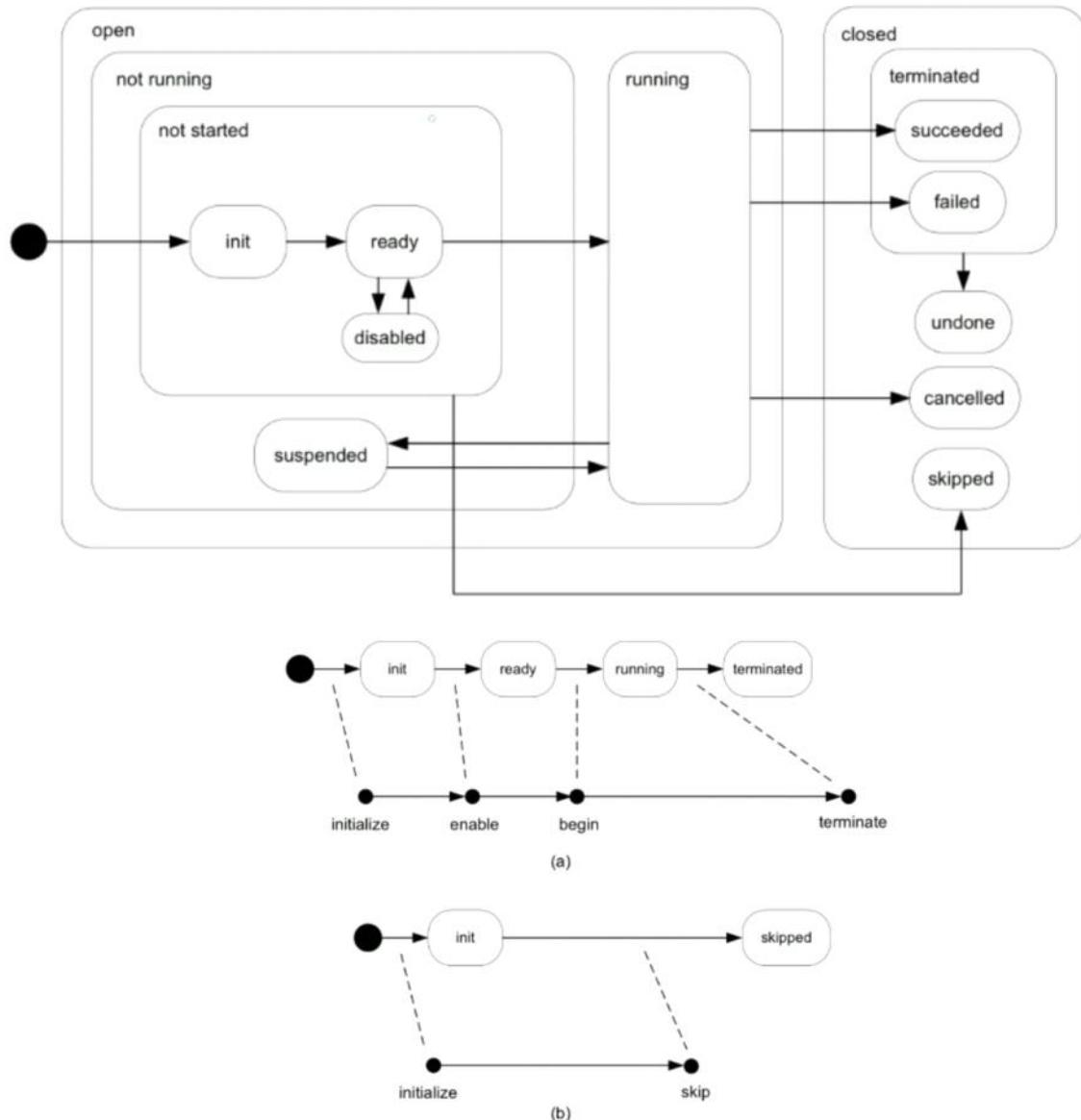
Di un processo possiamo vedere la definizione e le istanze. In BPMN si definiscono le definizioni, ma è importante ricordare che si tratta anche di istanze che descrivono il processo!

- Per ridurre la complessità del dominio e per permettere il riutilizzo, alcune attività possono

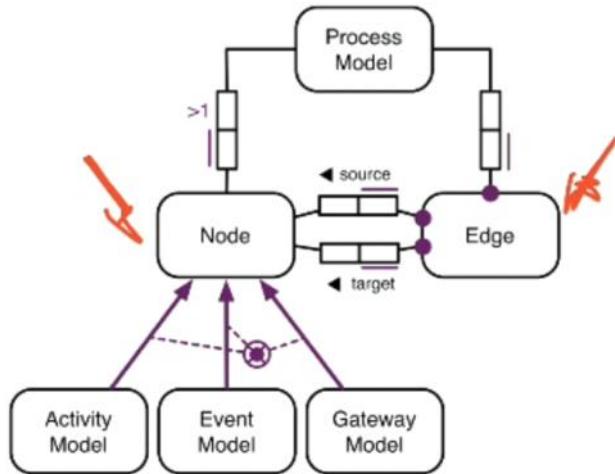
essere scomposte in sottoprocessi.

- Le attività del sistema possono essere associate a componenti o servizi del software
- Dato che il modello di un processo è una blueprint per istanze simili del processo, un modello di un'attività è una blueprint per istanze simili dell'attività
- Le istanze dell'attività sono distribuite lungo il tempo, e passano attraverso diversi stati. Le transizioni eseguite sono attivate da eventi che rappresentano collettivamente la traccia di ogni istanza del processo.

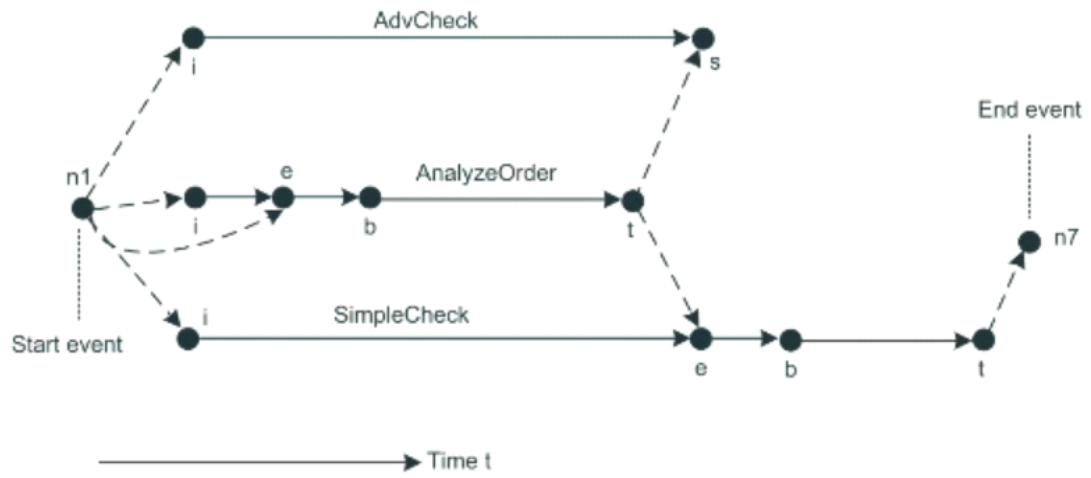
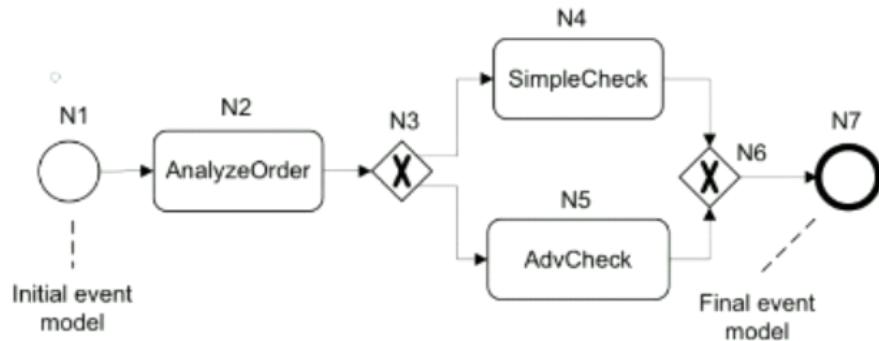
### Stati dell'attività durante il ciclo di vita



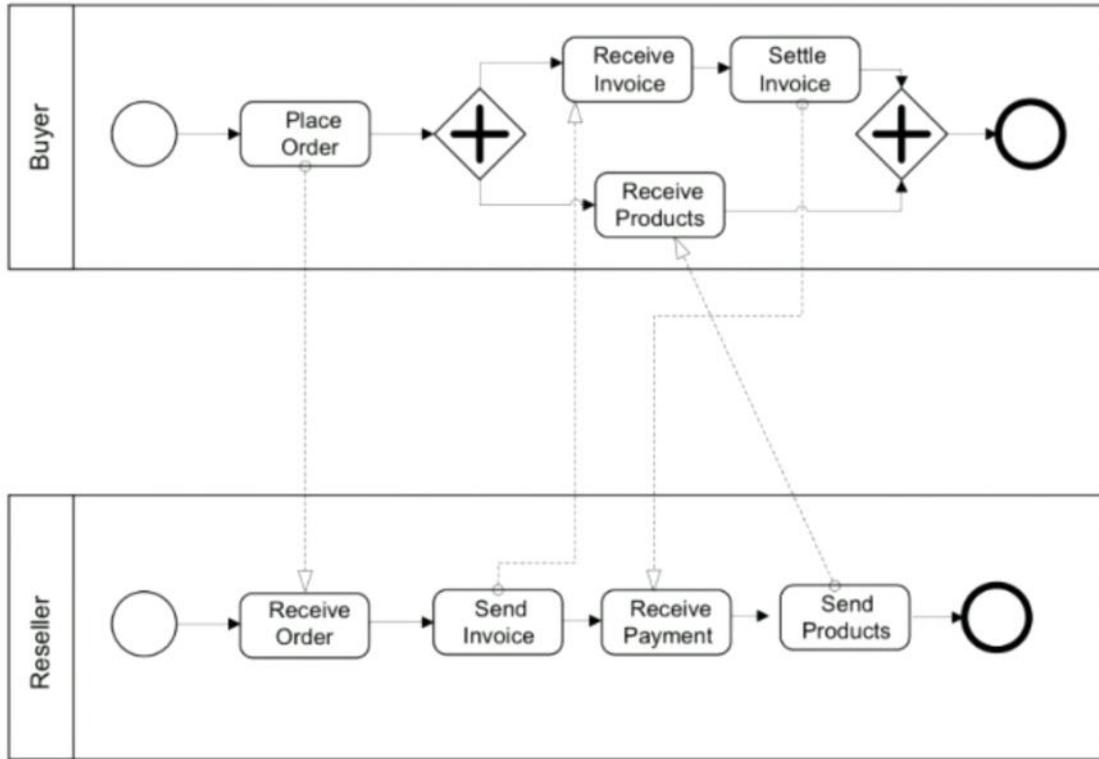
### Modelli di processo



I nodi possono essere modelli di attività, di evento o di controllo (flussi di esecuzione).



Due processi che interagiscono in parallelo:



All'interno ci sono i flussi di controllo, mentre i collegamenti sono flussi di dati.

Ci sono quindi tre elementi principali nei Business Process Models:

- **Controllo di flusso**: come le varie attività vengono coordinate tra di loro
- **Dati**: come deve navigare il flusso dei dati anche in relazione alle attività
- **Risorse**: usate per le attività. Ci sono strategie di allocazione.

Se ci sono delle attività che richiedono dei dati di input, queste saranno precedute da attività che li producono.

Ci sono anche dipendenze negli stessi dati!

I dati inoltre sono caratterizzati dal modello CRUD (Create, Read, Update, Delete) e quindi bisogna anche estrarre i vincoli rispetto alle azioni che vengono effettuate.

Le risorse sono risorse umane, dispositivi... tutto quello che può essere utile per compiere un'attività. Nel caso delle persone bisogna anche rappresentare i ruoli e i diritti/posizioni che possiede. Questo serve per poter poi capire come sono offerte le attività di lavoro a chi deve svolgerle.

Con il flusso di controllo ci sono dei punti di scelta che permettono di andare a separare il flusso, riunificarlo, reagire agli eventi o scatenare attività in funzione degli eventi.

# Business Process Model and Notation - 11 Nov

martedì 12 gennaio 2021 13:11

Fornisce un insieme di specifiche, notazioni, formati di interscambio e incentiva lo sviluppo di strumenti che li utilizzino. Standardizza un modello di esecuzione basato su processi WS-PBPEL.

Ha avuto successo perché è molto simile ai diagrammi di flusso.

Tuttavia, è diverso perché è una specifica formale:

1. Ha un meta-modello e delle regole di utilizzo. I modelli BPMN possono essere validati
2. Tiene conto di come i processi dovrebbero reagire ad eventi ed eccezioni
3. Studia come diversi processi comunicano tra di loro

BPMN ha una logica di processo.

Vengono definite tutte le possibili sequenze di attività in modo che, quando un processo sa:

1. Quali eventi sono avvenuti finora
2. Quali attività sono state completate
3. Quali dati sono stati prodotti

Sa anche cosa deve fare successivamente.

I processi modellano insiemi di esecuzioni.

Le relazioni molti-a-molti non sono esattamente rappresentabili.

Rende il flusso di controllo esplicito.

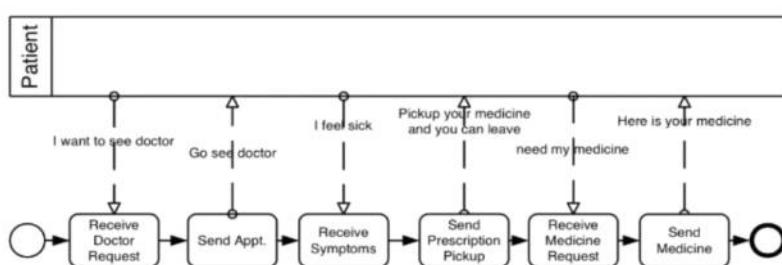
Con BPMN possiamo rappresentare:

- **Processi** (orchestrazioni): che hanno una prospettiva intra-organizzazionale e possono essere:
  - Privati, non eseguibili
  - Privati, eseguibili
  - Pubblici
- **Collaborazioni**: interazioni tra due o più entità di business:
  - Coreografie
  - Conversazioni

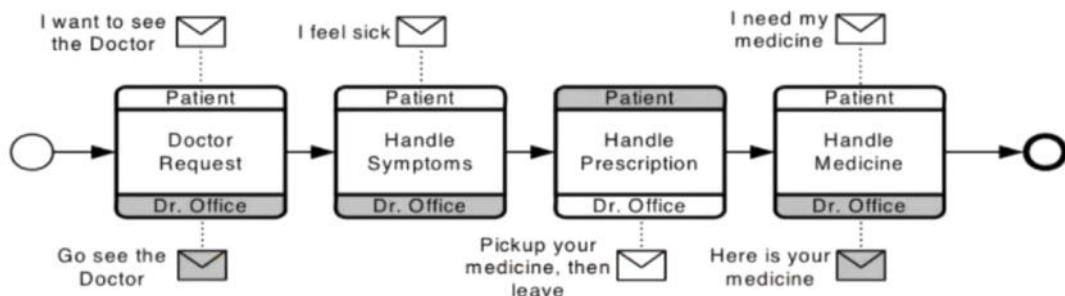
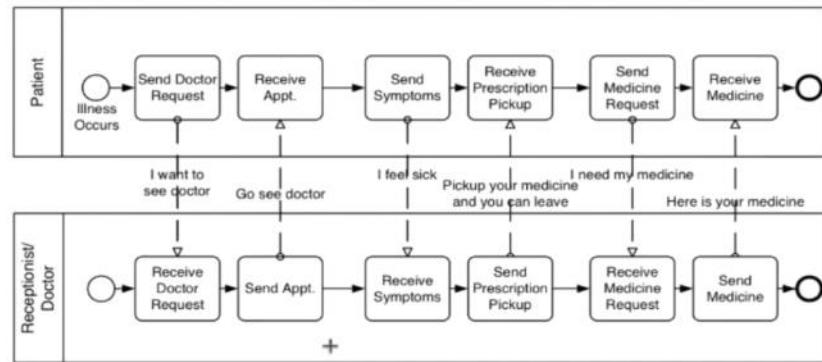
Private process.



Public process.



## Collaborative process.



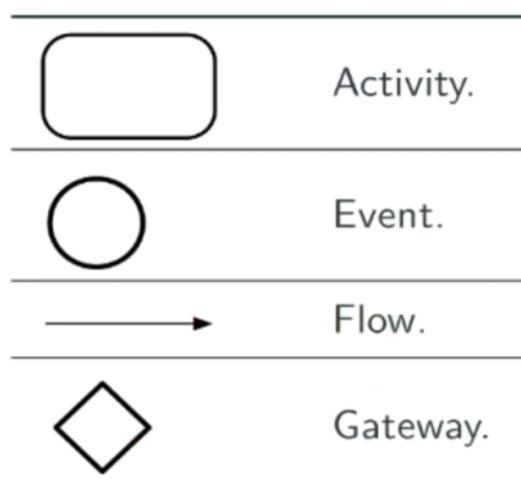
Cosa non è?

Non è una metodologia, e nemmeno uno stile. È una specifica di strumenti per annotare.

Gli elementi grafici di BPMN sono divisi in 5 categorie:

- Flow objects: comportamento del business process
- Data: Informazioni manipolate
- Connecting objects: connessioni tra flow objects e altri elementi
- Swimlanes: raggruppamento organizzazionale di elementi di modellazione
- Artifacts: informazioni addizionali.

Ogni famiglia di elementi è divisa in due strati: elementi base e avanzati.



Quando si affronta lo studio di un processo ci sono delle domande da utilizzare per comprendere bene la struttura di esso.

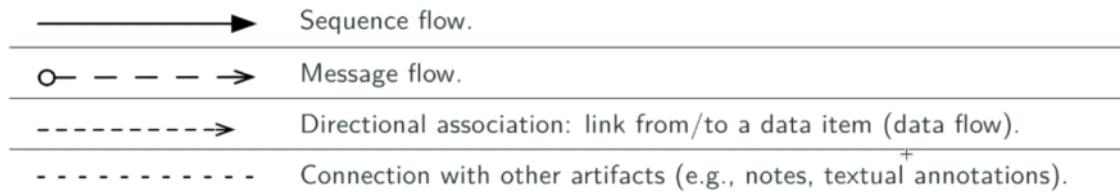
1. Qual è l'obiettivo del processo? Bisogna studiare il case (un po' come lo use case), l'istanza del processo che vogliamo generalizzare.
2. Per poterlo fare dobbiamo partire dagli eventi che scaturiscono l'inizio del processo e la sua fine.
3. Bisogna capire se ci sono dei punti di scelta o di biforcazione. Se abbiamo più inizi o

terminazioni, dovremo separare il flow.

- Quali sono gli eventi e le attività che lo compongono. Quali sono semplici e quali complesse

Un processo parte da un evento con il cerchio sottile e termina con un cerchio spesso.

Gli oggetti che connettono sono:



È importante usare la freccia corretta!!!

- Freccia principale per connettere gli elementi che costituiscono un processo
- Interazione che indica lo scambio di messaggi tra processi
- Flusso di dati da un'attività ad uno storage persistente o un documento verso una attività
- Connessione per note

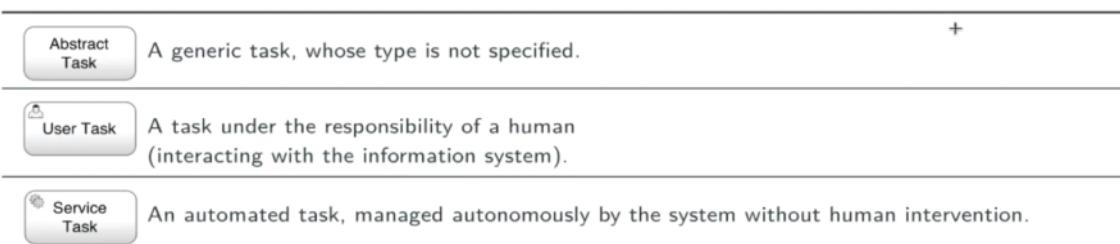
### Attività

Le attività rappresentano le unità di lavoro e vengono eseguite nel processo da un qualche performer.

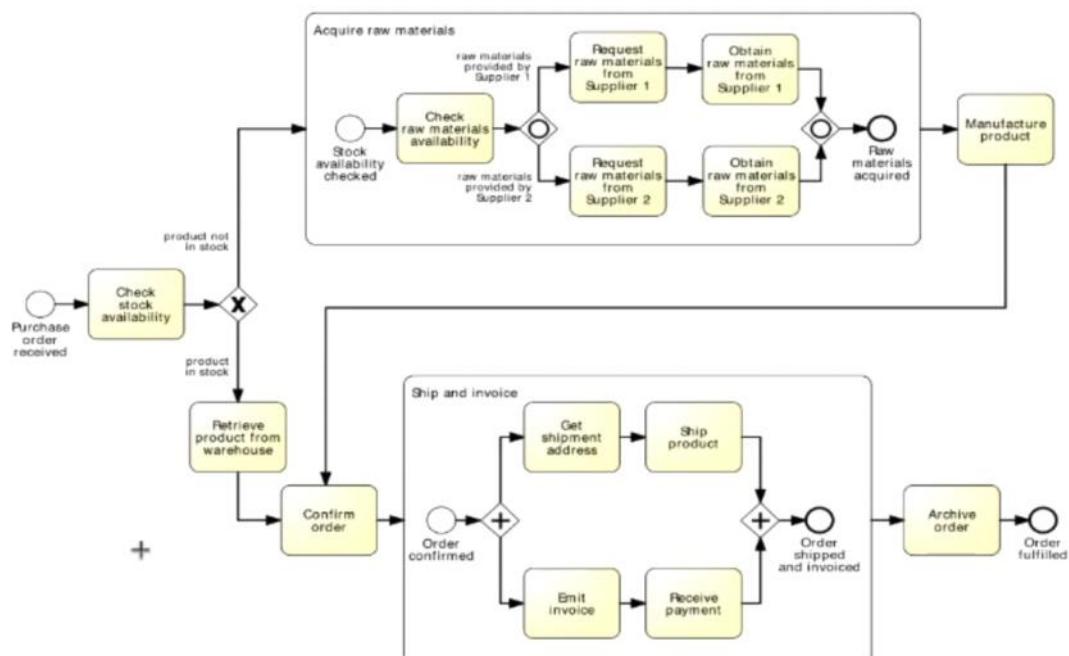
Un'attività può essere atomica (Task) o un sottoprocesso.

Le attività atomiche hanno un'etichetta del tipo **verbo-nome**

I sottoprocessi sono una composizione di altre attività.

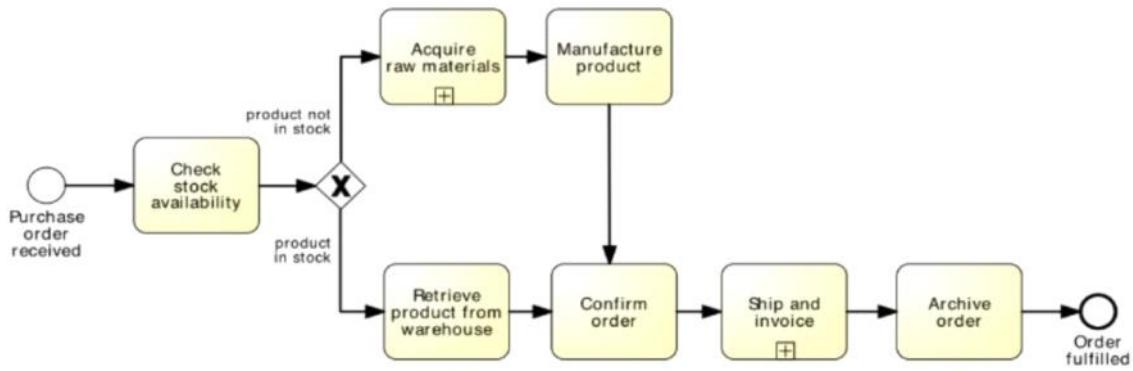


Possono essere apposte delle etichette per indicare chi sia l'attore.

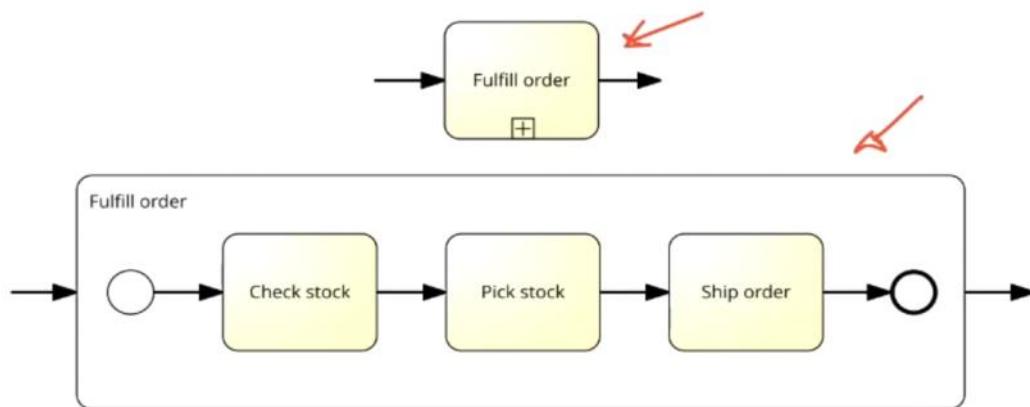


Esempio espanso di processo.

Stesso processo con i sottoprocessi collassati:

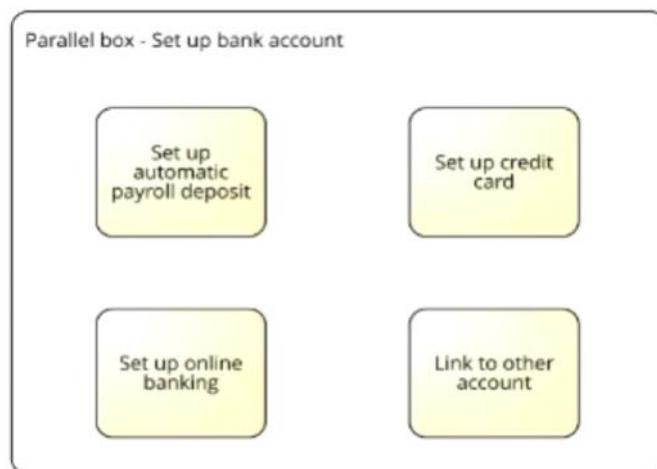


Con questa rappresentazione è possibile riutilizzare.



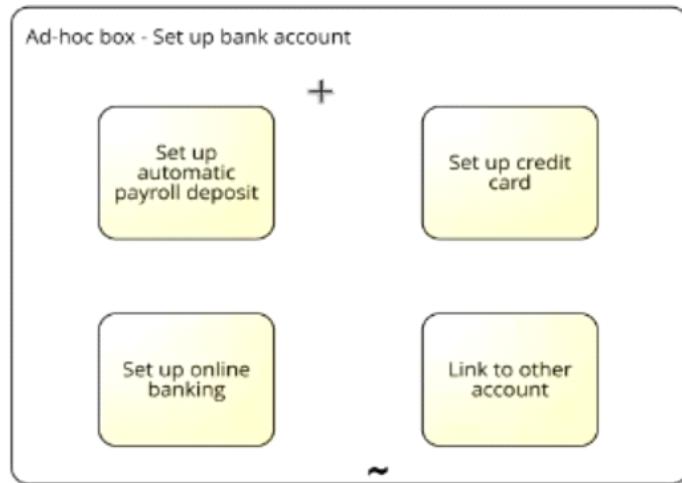
Un processo parte sempre dal suo start-event.

È possibile indicare un sottoprocesso con attività concorrenti:



Il sottoprocesso termina quando tutte le attività terminano.

Il processo ad-hoc indica un insieme di attività che devono essere eseguite, ma l'ordine non è importante.



Pro:

Modularizzare permette di rendere più leggibile e semplice un processo. Così è più condivisibile.  
Modellare in maniera top down è più facile. Le attività possono essere dettagliate in un secondo momento.

Non è possibile che ci siano messaggi o archi che superano i confini del processo. Lo scope è locale.

Gateway: rappresentati da un rombo e sono di due tipi:

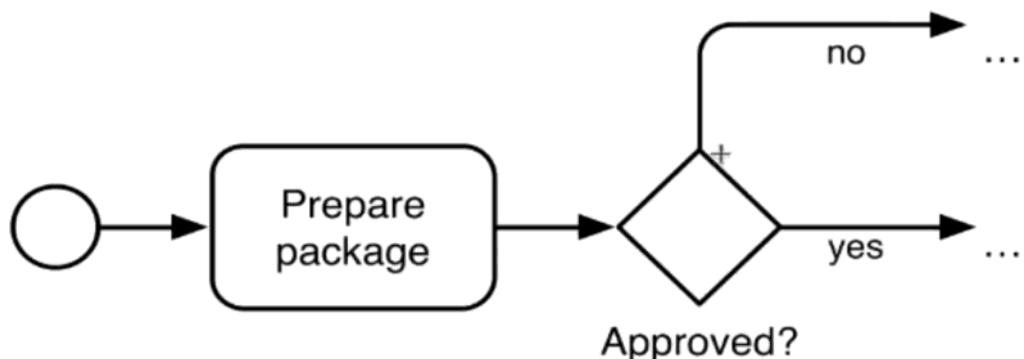
- Split: il flusso di controllo diverge in sottosequenze di flusso.
- Join: fa convergere più flussi in un unico flusso.



Se ha una X è uno XOR gateway. Il rombo vuoto è scoraggiato.

È una scelta basata su dati disponibili nell'information base in quel momento. Viene presa la decisione verificando cosa è vero.

L'uscita di default (altrimenti) è indicata da una sbarra. È importante che il modellatore si assicuri che solo una delle vie sia percorsa.



Lo XOR Join invece unisce più flussi.

AND Split



Separa l'esecuzione di un processo su più thread. È una Fork

Quando è un join, aspetterà che tutti i flussi si riuniscano.

# Eventi - 17 Nov

sabato 16 gennaio 2021 11:19

## Eventi di start

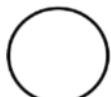
Sono rappresentati da un cerchio col bordo fine.

Indicano **come e dove** un processo parte.

Ce n'è **solo uno**.

I processi top-level possono associare un **trigger** all'evento.

### Start Generico



Indica che il trigger non è specificato o che il processo parte autonomamente con intervento manuale.

È l'**unico utilizzabile per far partire sottoprocessi**. Il significato è che il sottoprocesso parte non appena il padre lo abilita

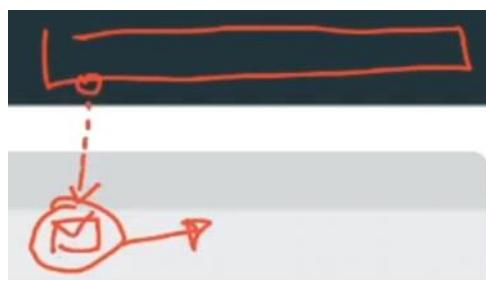
### Start con messaggio



Deve avere una lettera bianca.

Il processo **parte non appena un certo messaggio è ricevuto**. Rappresenta uno scambio di messaggi: da qualche parte c'è l'invio ma qui c'è la ricezione. Possono essere rappresentate entrambe le parti, può essere un processo collassato, oppure il flusso non è indicato.

Si può aggiungere un'etichetta per indicare il tipo di messaggio che si aspetta.



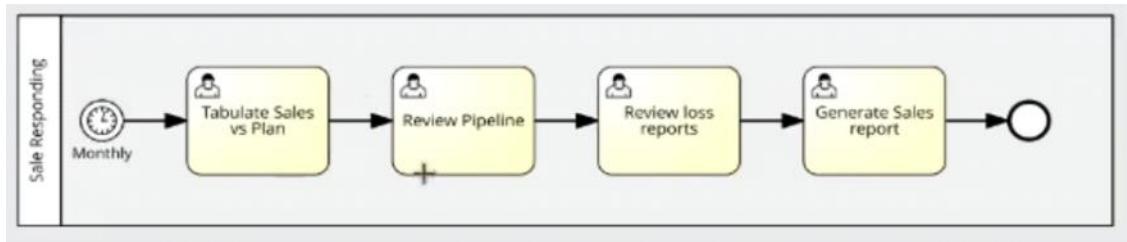
Il rettangolo identifica chi sta inviando.

### Start con timer

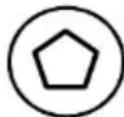


Il timer può specificare:

- Un momento specifico
- Una ricorrenza.



Start multipli

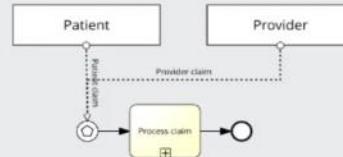


Pentagono: è uno start multiplo; il processo è iniziato ogni volta che uno dei trigger associati occorre.

Col +: Il processo è avviato ogni volta che tutti i trigger occorrono (senza un ordine specifico)

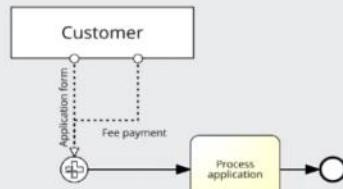
### Example

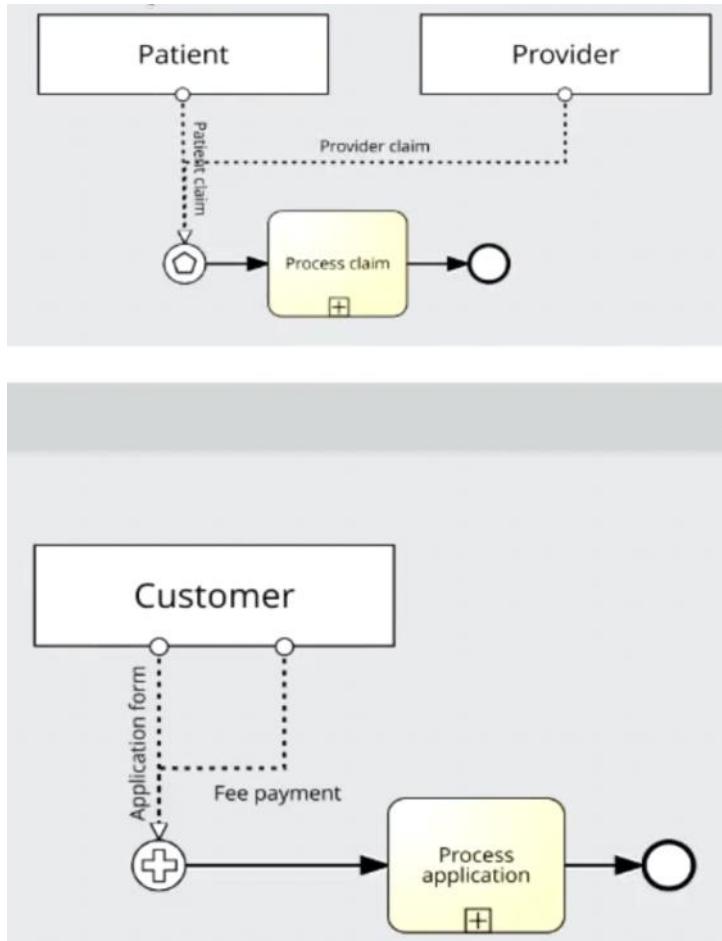
A claim is processed every time a claim is received by a customer or a provider.



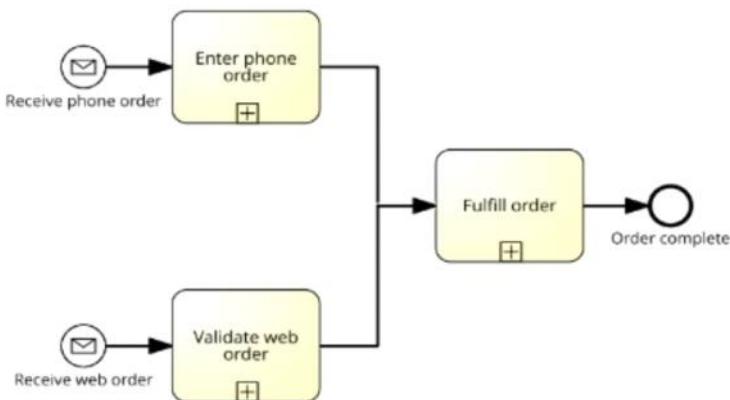
### Example

An application is processed every time the customer fills an application form and pays a corresponding fee.





#### Eventi di start alternativi



In certi casi potrei desiderare di applicare almeno parzialmente delle attività che dipendono dal tipo o dalle modalità di ricezione:

*Il refill di un prodotto nel magazzino parte quando quel prodotto scende sotto una certa soglia.*

Semplicemente si crea ogni flusso separato con un suo evento di start.

Quando uno degli eventi avvia il processo, un caso viene generato e gli altri eventi **sono ignorati**.

#### End Events

Bordo spesso. Sono la fine di un cammino di un processo.

Possono essere anche il risultato di un processo (con un'etichetta)

È molto comune finire con più possibili risultati

#### None End Event



(Cerchio spesso)

Termina quando tutti i cammini che vi entrano (o che sono collegati allo stesso case) effettivamente sono terminati.

Message End Event



(notare la lettera nera)

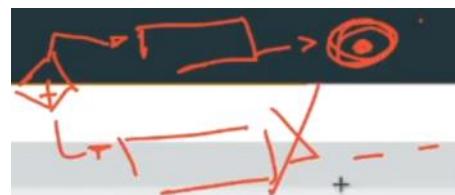
Indica che genera un messaggio!



Terminate End Event



Significa che immediatamente il processo termina.



Terminazione con eventi multipli



Rappresenta un singolo punto di terminazione dove multipli risultati sono emessi.

# Elementi di Connessione - 17 Novembre

sabato 16 gennaio 2021 15:18

## Sequence Flow



Rappresenta un flusso e connette varie attività/gateway e simili.  
La freccia non può mai attraversare il confine di un sottoprocesso.

## Message flow



Denota una comunicazione tra processi, quindi tra attività separate.  
Non possono mai collegare due parti dello stesso processo.

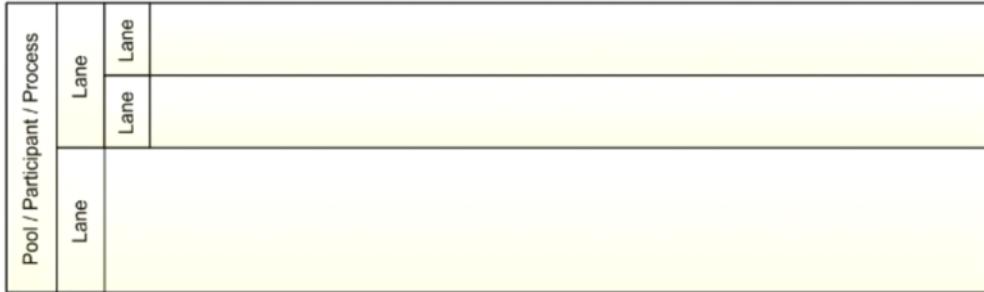
Quando è diretta verso uno start, indica la possibilità di una comunicazione.

# Modello Organizzazionale - 17 Nov

sabato 16 gennaio 2021 15:21

A seconda dei settori del processo, si va a identificare zone diverse.

Un processo è in genere associato ad un pool:

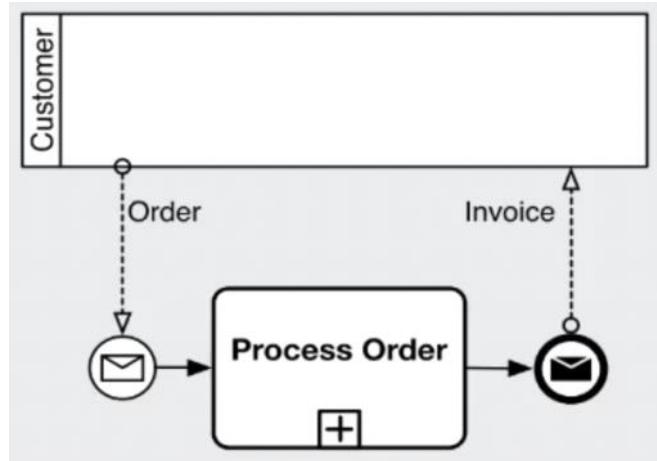
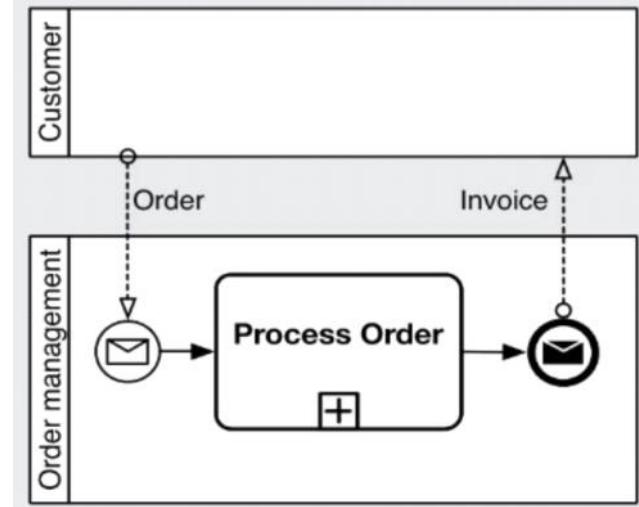


Le varie linee suddivisibili più volte, rappresentano le classi di risorse dell'organizzazione. Collocando le attività all'interno delle corsie decido anche le responsabilità.

In genere la pool ha il nome del processo.

Alle lane si dà il nome della categoria.

A volte le pool nelle collaborazioni possono essere omesse:



# Dati - 17 Nov

sabato 16 gennaio 2021 15:29

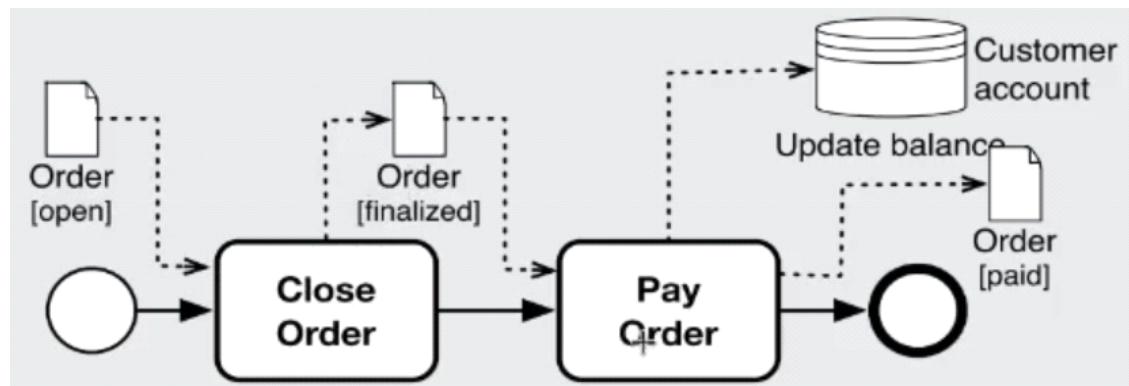
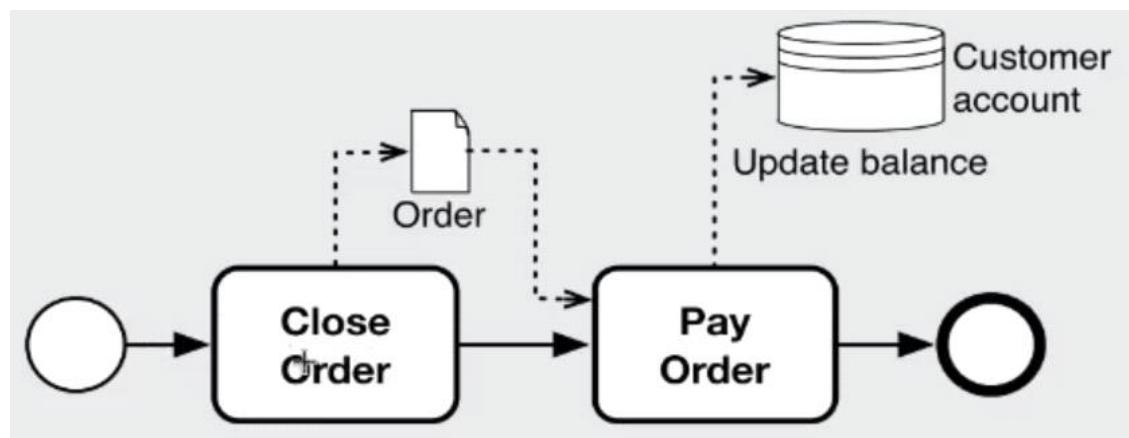
	<b>Data object:</b> local variable inside a process level, pointing to a temporary unit of information.
	<b>Data object reference:</b> refers to a data object in some state. Data Object [Object State]
	Variable representing a collection of data objects. Data Object Collection
	<b>Data store reference:</b> reference to persistent unit of information, manipulated by the process but also external entities. Data Store

Data object: i dati sono volatili/temporanei e vengono persi alla fine dell'esecuzione  
La reference è utilizzata per indicare uno stato.

La data store esiste anche al di fuori del processo e può essere usata per scambiare dati.

Si utilizza una freccia tratteggiata per collegare un Data Object con un'attività o un evento.

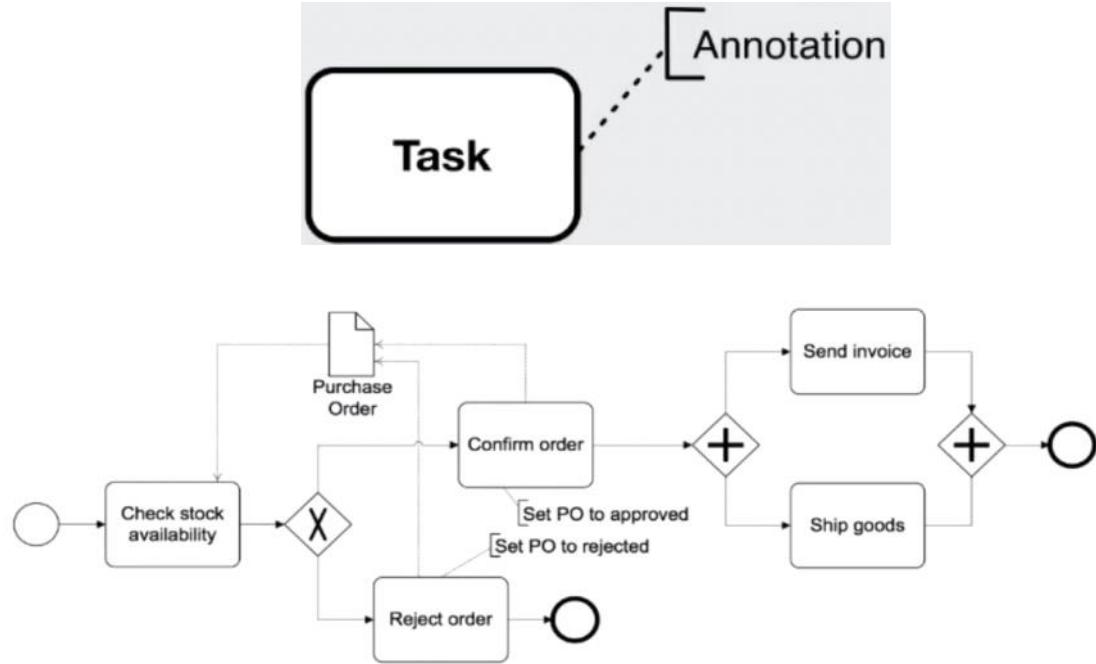
Puntare a un data store denota un aggiornamento dei dati, mentre al contrario indica una query.



# Documentazione - 17 Nov

sabato 16 gennaio 2021 15:34

Possiamo rappresentare delle note o della documentazione con una parentesi quadra assieme a una linea tratteggiata



# Metodologia per Progettare processi - 17 Nov

sabato 16 gennaio 2021 15:40

BPMN non ha una vera metodologia, ma possiamo individuare alcuni aspetti chiave.

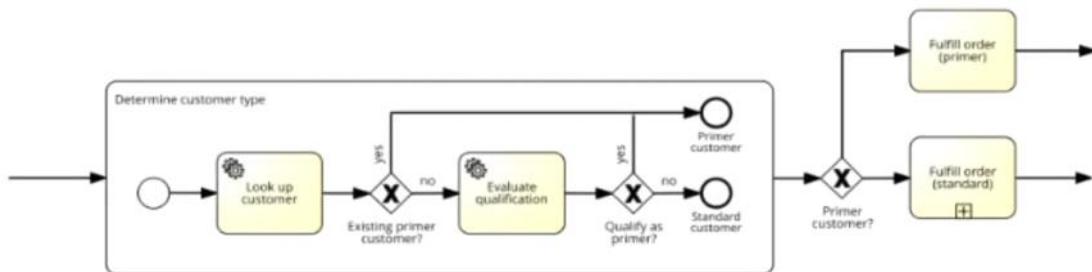
Possiamo usare un approccio gerarchico con una modellazione top-down.  
Dobbiamo unire i sottolivelli ai genitori con un'attenta scelta delle label.

Ogni diagramma deve essere in una pagina logica che deve contenere un racconto completo.  
Si devono usare solo sotto-processi collassati.

Top-down: si parte dal processo padre e poi si passa ai sottoprocessi.

È importante prestare attenzione innanzitutto ai punti terminali, per individuare i modi in cui può terminare un processo, i cammini alternativi, quelli che portano ad errori o eccezioni.  
È importante perché c'è un numero di scelte che può variare a seconda del cammino che verrà percorso.

Se in un processo padre ho una scelta sì/no, e la scelta è funzione di un'attività e voglio dettagliare l'attività, avrò bisogno di una doppia terminazione che porti ai due diversi fini (con quindi anche due gateway diversi)



-> [Determina il cliente] -> (Alta priorità) X (Bassa priorità)

Sapendo che il sottoprocesso influenza il comportamento, dovrò avere due terminazioni in esso.  
Avendo diverse terminazioni, dovrò avere delle scelte.

Passo 1:

Vedere la portata del processo. Determina anche le attività che dovrò considerare.

Chiedersi:

- Come/quando parte il processo? Lo fa su richiesta o è pianificato?
- Che cosa determina il completamento del processo?
- Quale nozione di "caso" è applicabile? (Qual è lo use case che voglio effettivamente determinare?)
- Ci sono diversi modi per terminare il processo? Sono concettualmente importanti?

## Running Example: Car Dealer Order-To-Cash

Focus: process for purchasing a new car, from the car dealer perspective.

- When does the process start?  
Well-identified moment in time: when the customer creates a purchase order.
- What information are needed to start a process?  
Detail specification of the car; customer info.
- What is the notion of a case?  
A purchase order. +
- What happens if I want to buy two cars?  
Are they part of a single *contract* and *financial transaction*?
- When does the process terminate?
  - Successful completion: transaction done.
  - (Relevant) exceptions:
    - financing unavailable;
    - delivery date unacceptable;
  - Note: they both represent that the transaction has *not* been completed, but for different reasons.
  - Different involved stakeholders. different mitigation strategies!

Passo 2:

Creare una mappa ad alto livello.

Mai trovare più di 10 attività per costituire a grandi linee lo sviluppo del processo.

Questo viene fatto andando ad individuare cos'è l'attività, chi la gestisce, come viene portata a completamento.

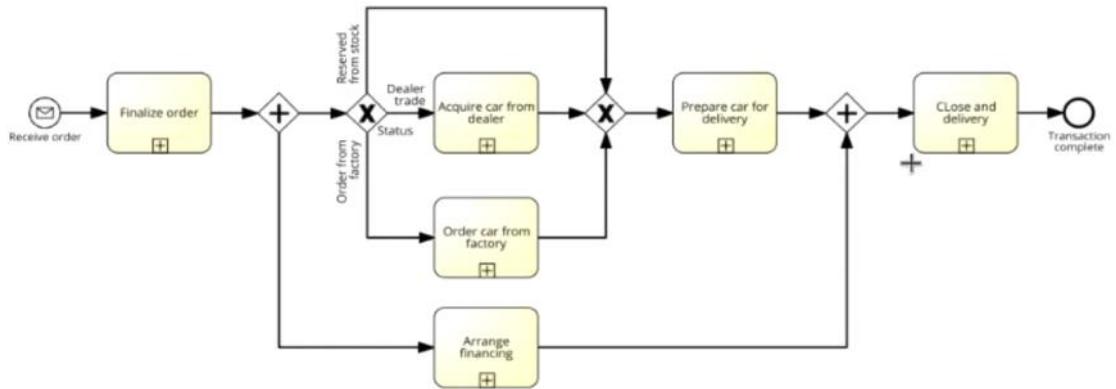
ACTIVITY	RESP.	COND.	REQUIREMENTS	END STATES
Finalize order	Sales dept.	N		Reserved from stock Dealer trade Order from factory
Get car from dealer	Sales dept.	Y	Order finalized	Car received
Get car from factory	Sales dept.	Y	Order finalized	Car received Order cancelled (dates mismatch)
Prepare car	Service dept.	N	Car obtained	Car ready
Arrange financing	Finance dept.	N	Order finalized	Financing confirmed Financing unavailable
Close and deliver	Finance dept.	N	Financing confirmed Car ready	Transaction complete
Handle order cancellation	Finance dept.	E	Order cancelled or financing unavailable	Delivery date unacceptable Financing unavailable

Passo 3:

Organizzare la mappa ad alto livello in un diagramma del processo ad alto livello.

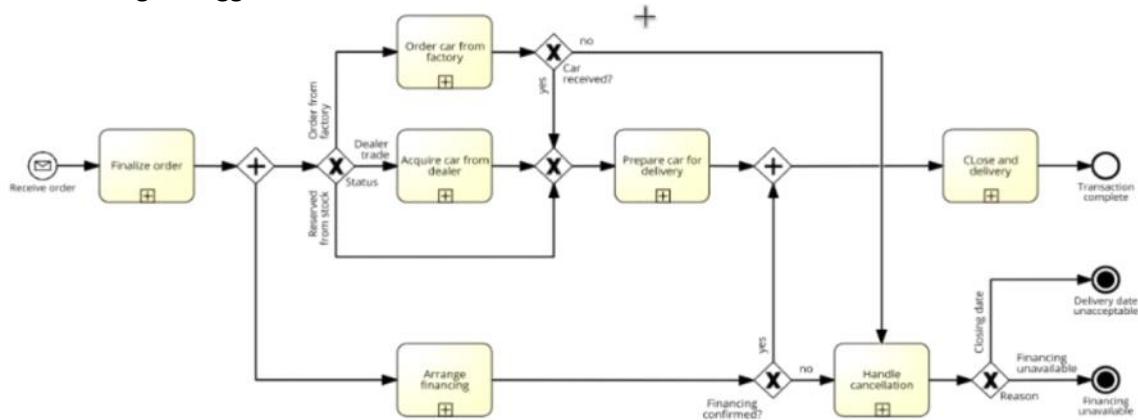
1. Determinare come inizia il processo
2. Modellare ogni attività nella mappa ad alto livello come un processo collassato
3. Collegare adeguatamente le attività utilizzando sequenze e gateway.

Suggerimento: Concentrarsi prima sui cammini felici, e poi aggiungere quelli che portano ad eccezioni!

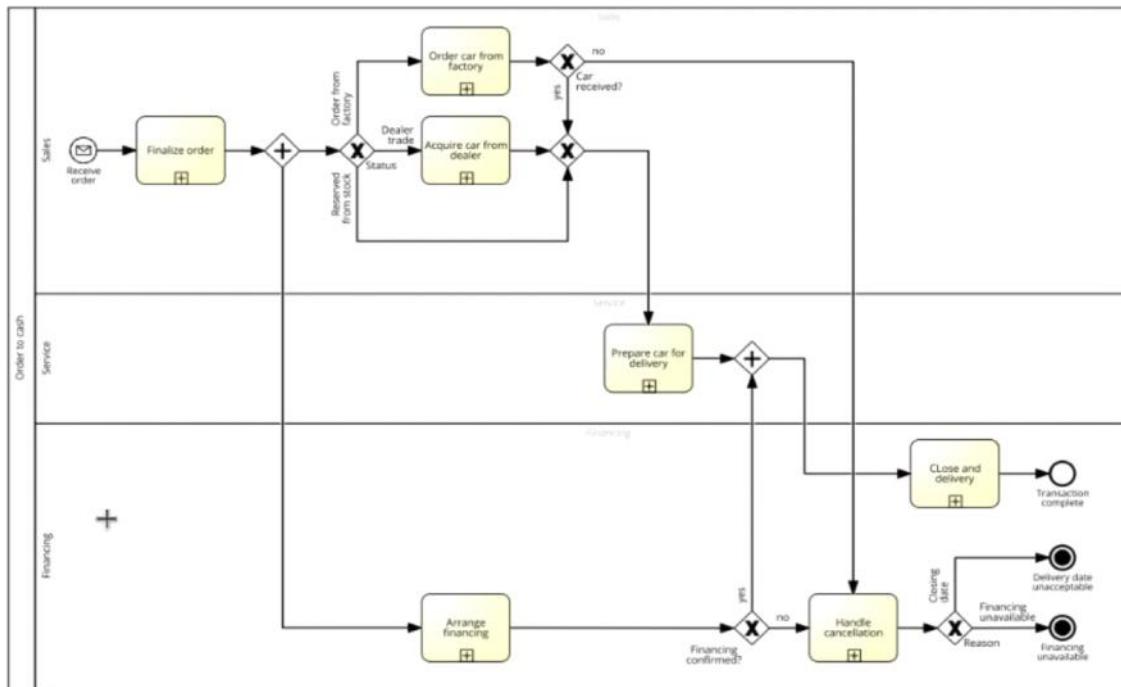


Questo è un diagramma di massima.

Adesso vengono aggiunte anche le eccezioni:



Colloco adesso in una pool il processo:



Passo 4:

Sviluppo i sottoprocessi del top-level

1. Creare un diagramma separato per ogni attività del top-level
2. Collegarlo al sottoprocesso collassato
3. Aggiungere i dettagli al livello del processo figlio, possibilmente raffinando la classificazione degli stati di terminazione (impattando anche sulla mappa del livello superiore)
4. Continuare ricorsivamente

Suggerimenti:

- Un processo figlio deve avere un unico *None Start Event*
- Gli stati di terminazione e le scelte del gateway del processo padre devono coincidere
- Se il processo figlio ha una pool, il suo nome deve corrispondere a quello del processo padre (cioè il nome del processo)
- Delle corsie possono essere aggiunte se sono "locali" ad un processo

Passo 5:

Flusso dei messaggi: aggiungere delle indicazioni chiarendo il contesto di business

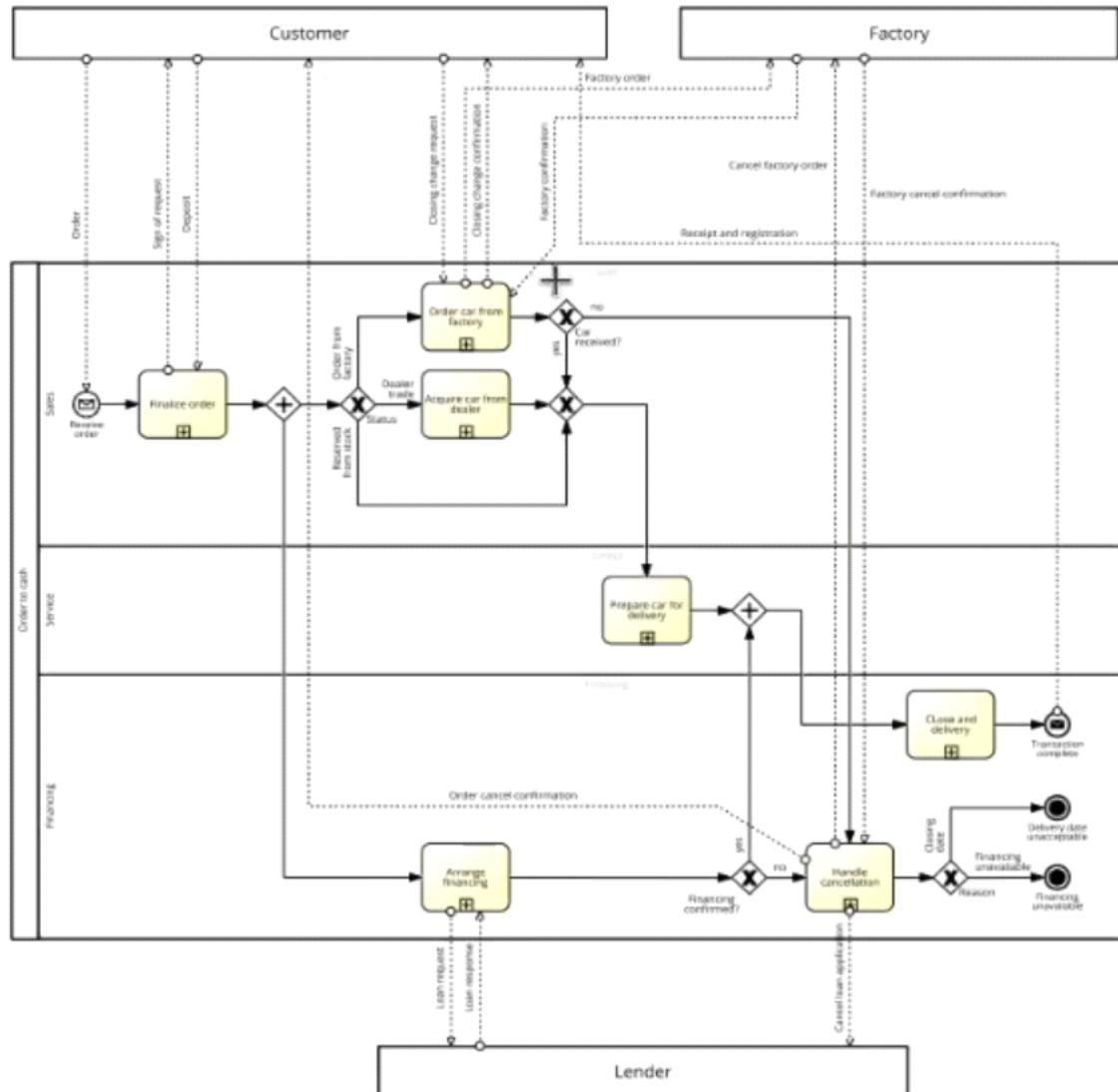
1. Inserire tutti i partecipanti esterni importanti come delle pool a scatola nera
2. Disegnare i flussi di messaggi tra le scatole nere e le attività/eventi nel process model di interesse
3. Assicurare consistenza tra i livelli: i messaggi di IO collegati ad una activity devono essere riflessi nei processi figli, tenendo conto di numeri ed etichette.

The diagram illustrates a business process flow involving two main participants: Customer and Factory. The process starts with a 'Place factory order' activity in the Customer pool, which sends a 'Factory order' message to the Factory pool. The Factory pool contains an 'Receive production completion date' activity. A 'Match original expectation?' gateway follows, with both paths leading back to the Customer pool's 'Place factory order' activity. One path leads to a 'Revise closing date' activity, and the other to a 'Reconfirm with customer' activity. Both lead to a 'Change confirmed?' gateway. If 'no', the process ends at 'Order canceled'. If 'yes', it proceeds to the Customer pool's final 'Receive car from factory' activity, which ends with a 'Car received' message. External messages include 'Closing change request' from Customer to Factory, 'Change confirmation' from Factory to Customer, and 'Factory confirmation' from Factory to Customer.

```

graph LR
    Start(( )) --> Place[Place factory order]
    Place -- "Factory order" --> FactoryPool[Factory]
    FactoryPool -- "Receive production completion date" --> Match{Match original expectation?}
    Match -- no --> Revise[Revise closing date]
    Revise --> Reconfirm[Reconfirm with customer]
    Reconfirm --> Change{Change confirmed?}
    Change -- no --> OrderCanceled((Order canceled))
    Change -- yes --> ReceiveCar[Receive car from factory]
    ReceiveCar -- "Car received" --> End(( ))
    Change --> CloseReq((Closing change request))
    CloseReq --> Conf((Change confirmation))
    Conf --> FactoryPool
    FactoryPool -- "Factory confirmation" --> Place
  
```

AA 2020-21 Pagina 94



# Token Game - 18 Nov

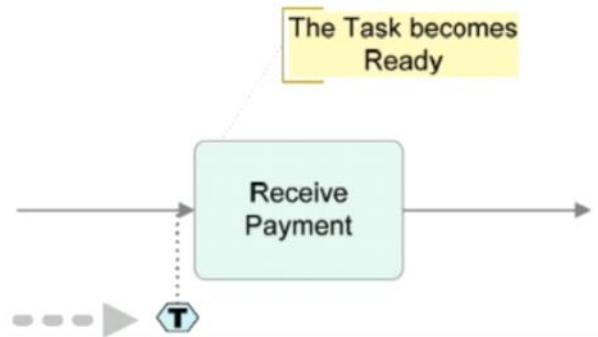
sabato 16 gennaio 2021 16:25

Il Token Game fornisce un'intuizione della semantica di esecuzione.

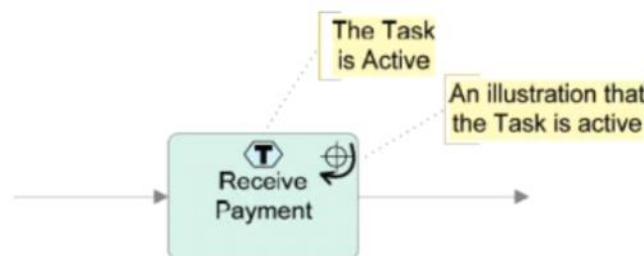
Una volta che un processo è creato (una istanza) si usa la nozione di token per identificare lo stato del progresso.

Il token è un oggetto teorico usato per una "simulazione" descrittiva.

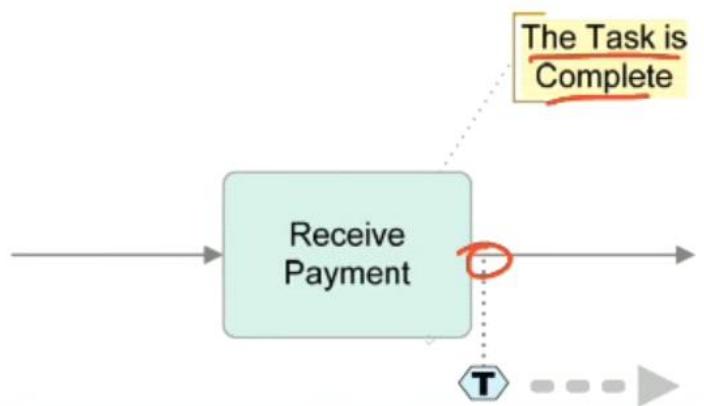
È creato con uno start event e una volta attraversato il flusso viene distrutto in un end event.  
Il token non ha un tempo associato.



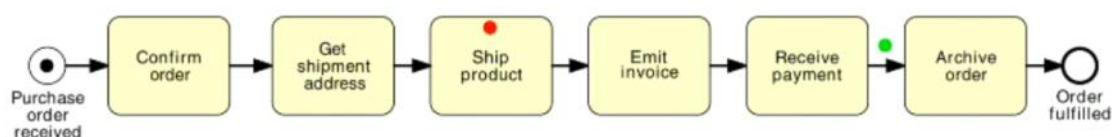
(Si trova nel punto precedente all'attività)



(A questo punto il task è attivo)



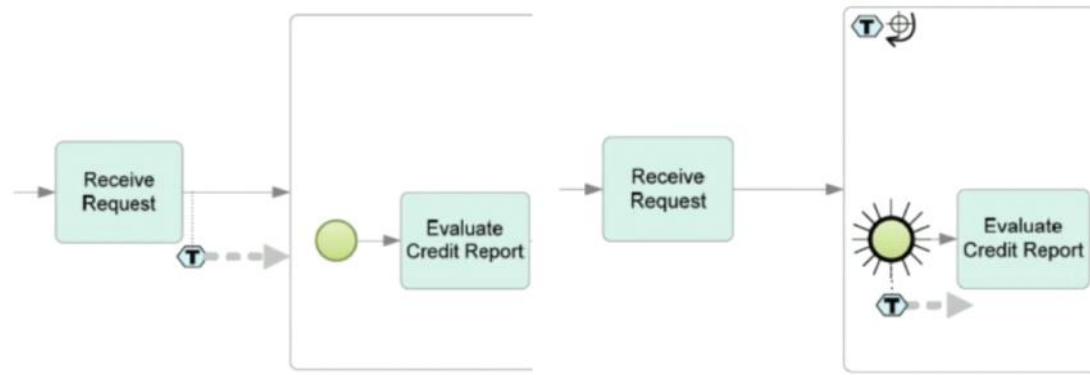
( qui invece ha completato l'attività)



(Ricordiamo che parte da uno start event e termina in un'end event.)

Possono essere presenti più istanze dello stesso processo.

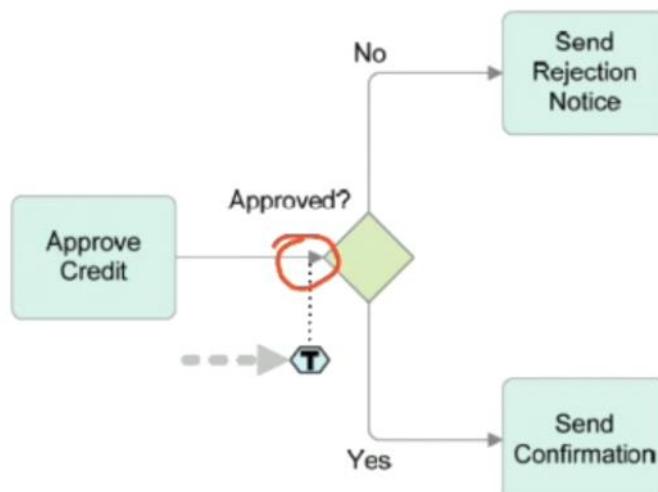
Cosa succede quando il token si trova nei pressi di un sottoprocesso?



Il token entra e un nuovo token viene associato allo start del sottoprocesso. Quando arriva alla fine, il token del sottoprocesso viene distrutto e riparte quello del sequence flow superiore, andando nell'attività successiva.

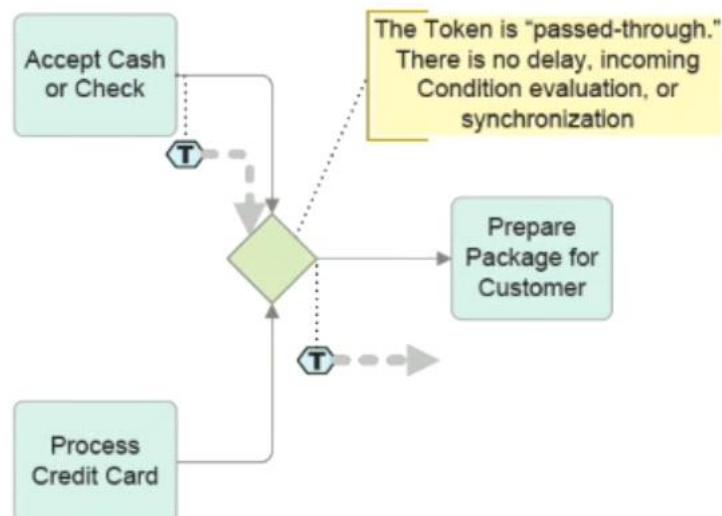
Un gateway invece implica che ci sia un meccanismo a porta.

Quando un token arriva ad un gateway, può essere unito ad altri o separato in più token

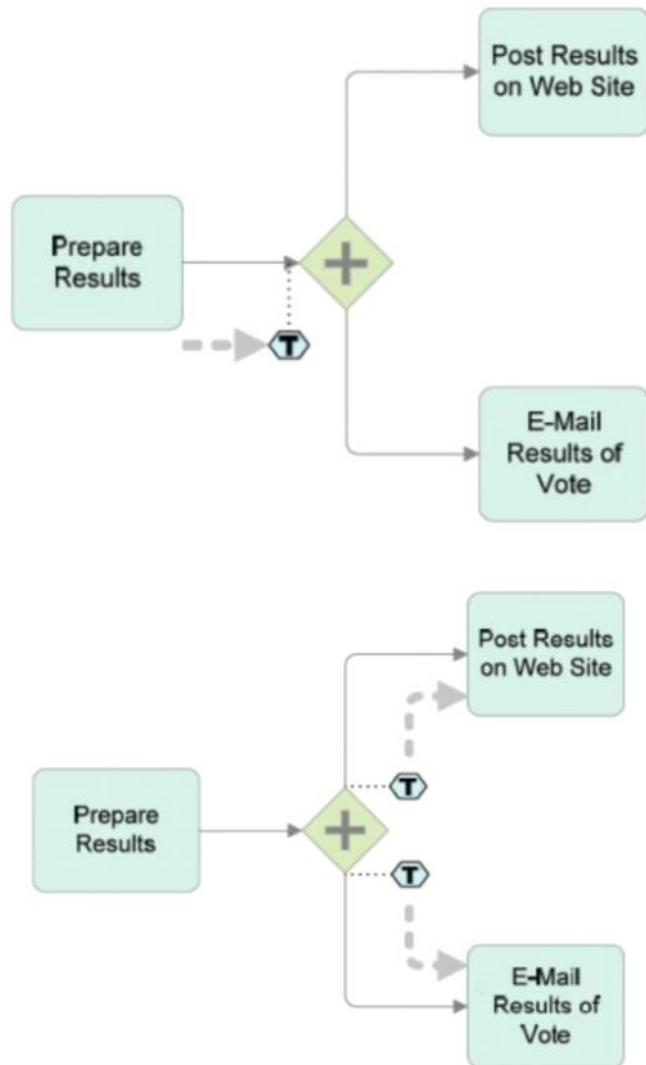


Il criterio di esclusività fa in modo che il token sia diretto su una sola delle due strade.

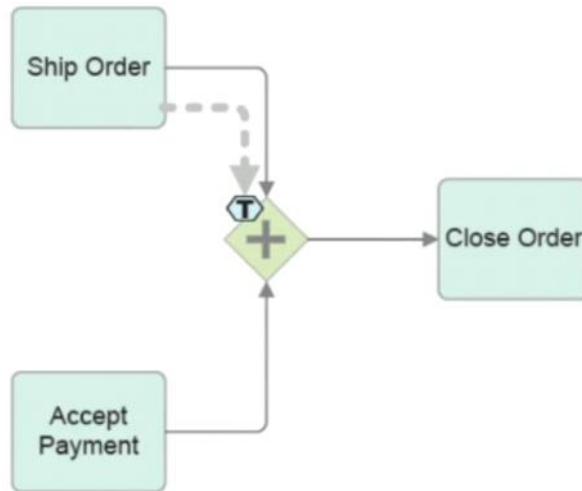
Nel caso di uno XOR join, non ci sono condizioni di valutazione quindi il token passa subito senza ritardo



AND split: viene creato un token per ogni flow parallelo

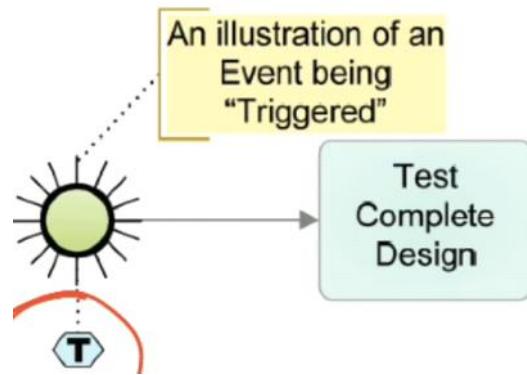


L'AND join obbliga il gateway ad aspettare per la ricezione di un token da ogni cammino:

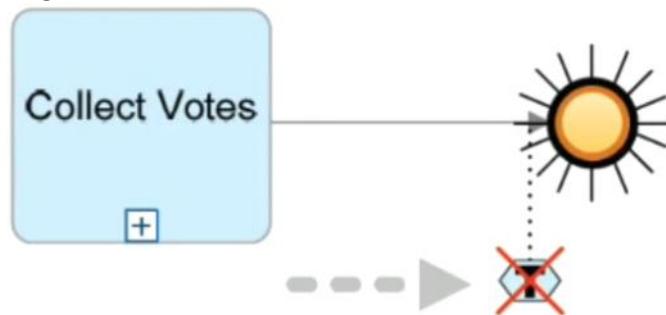


Quando finalmente arrivano tutti i token, vengono fusi e si prosegue.

Lo start event crea il token per l'esecuzione.



Quando un token raggiunge un end event viene rimosso.



Solo quando tutti i token associati alla stessa istanza del processo vengono distrutti, allora l'istanza del processo termina: è possibile avere ancora alcuni cammini che vengono eseguiti dopo la distruzione di un token.

# Linee guida stilistiche - 18 Nov

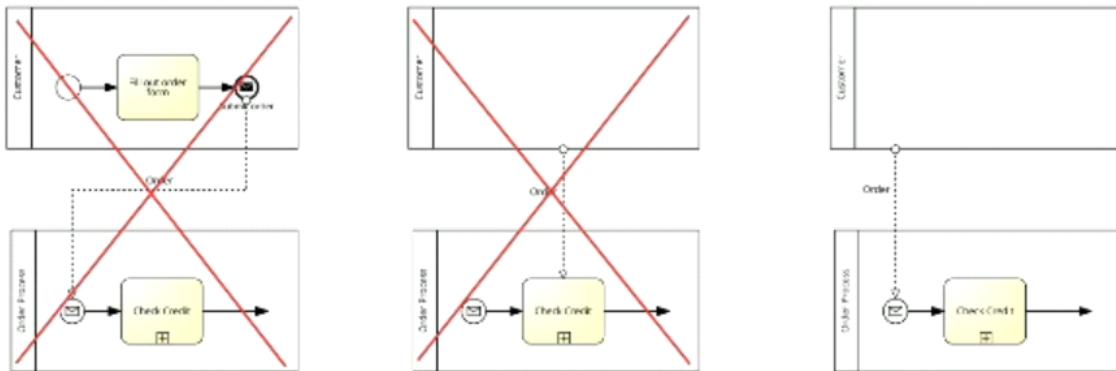
sabato 16 gennaio 2021 19:45

Si ricordi che BPMN non ha delle norme stilistiche. È solo una specifica di utilizzo degli elementi che rende disponibili.

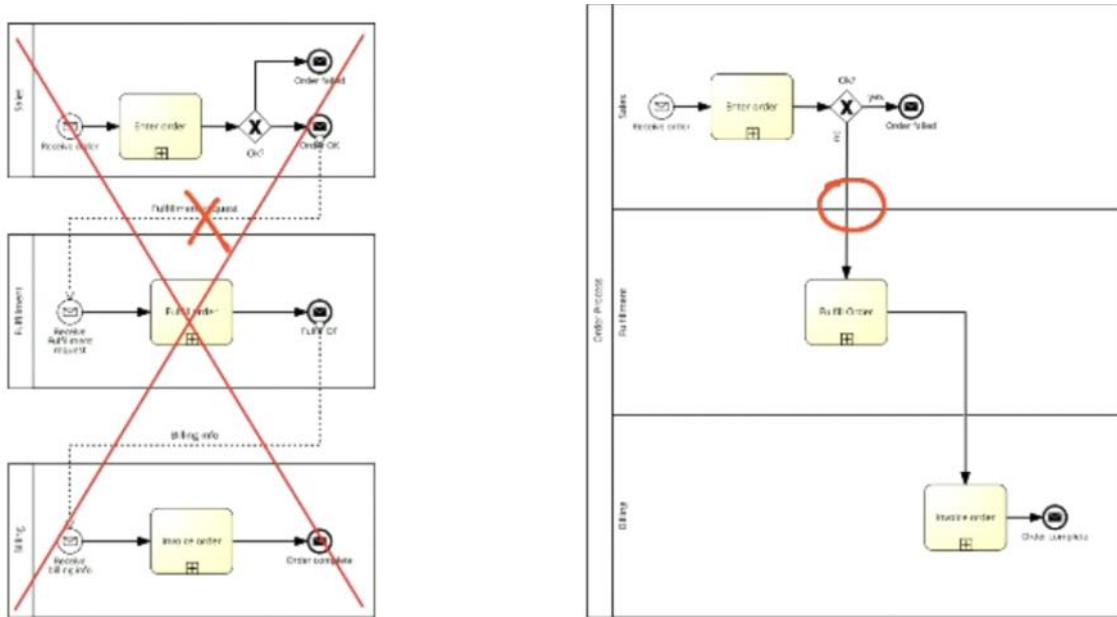
Si vuole rendere chiari, leggibili e comprensibili gli schemi.

La logica del processo deve essere comprensibile dal diagramma stesso, in maniera non ambigua.

- 1. Utilizzare icone ed etichette per rendere la logica del processo il più possibile chiara**
  - a. Tutte le attività e i sottoprocessi devono avere un'etichetta
  - b. Se ci sono più stati di termine devono essere etichettati
  - c. Le pool, i gate, i messaggi devono essere etichettati
  - d. Identificare i tipi delle attività e decorare i trigger degli eventi con icone
  - e. Se ci sono ancora delle possibili ambiguità, usare delle annotazioni
- 2. Realizzare modelli gerarchici. Ogni processo dovrebbe essere "infilabile" in una pagina**
  - a. Il diagramma top-level dovrebbe rappresentare il processo dall'inizio alla fine e come interagisce con attività esterne
  - b. I diagrammi figli devono espandere i dettagli
  - c. I processi figli forniscono un meccanismo per evolvere il processo senza toccare il top-level
  - d. Se ci sono dei diagrammi che non stanno in una pagina, utilizzare con attenzione dei connettori fuori-pagina (off-page connectors)
- 3. Utilizzare black-box pool per rappresentare il cliente o altri inquisitori esterni o fornitori di servizi**
  - a. Mostrare i processi interni dei partecipanti esterni è sbagliato: non conosciamo infatti tutti i dettagli intrinseci dei processi esterni
  - b. Non possiamo espandere un processo solo parzialmente
  - c. Non possiamo associare un message flow ai limiti di una process pool



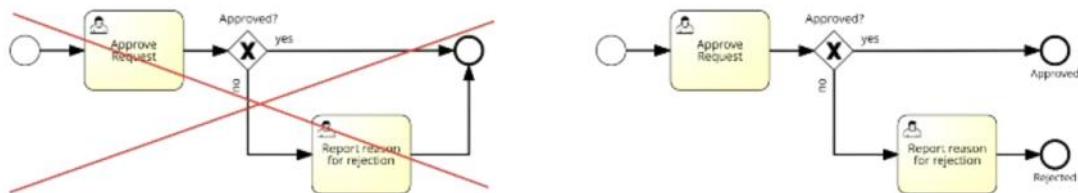
- 4. Se un processo inizia con un messaggio di start, allora è buona norma collegare il messaggio che proviene dall'esterno con l'evento di start**
  - a. Il messaggio stesso rappresenta il caso d'uso
  - b. Agganciare il messaggio ad una task indicherebbe la possibilità di uno scambio di informazione (legame più debole!)
- 5. Se l'organizzazione ha un processo che è diviso tra più ruoli, non è giusto rappresentarlo come processi separati, ma è necessario dividere i ruoli in lane della pool**



**Eccezione:** si può dividere il processo se non ci sono degli allineamenti 1:1 con i multipli casi d'uso.

(Esempio gestione ordine e ordini del mese: c'è un allineamento 1:N. dovranno essere due processi separati)

6. **Etichettare i pool che rappresentano processi.** In genere si può etichettare la pool con il nome del processo, in modo da poter aggiungere altri processi per la stessa organizzazione.
7. **Indicare separatamente gli end state di un processo/sottoprocesso che portano al successo o a un'eccezione ed etichettarli correttamente in modo da indicarne lo stato.**
  - a. Questa distinzione deve riflettere il fatto che una distinzione concettuale è importante
  - b. Particolarmente vero se il processo padre ha bisogno di distinguere il comportamento in funzione dello end state raggiunto in un sottoprocesso
  - c. Esempio solito: successo/fallimento



#### 8. Le attività dovrebbero essere sempre etichettate come VERBO-NOME

- a. Le attività denotano lavoro o azioni che vengono eseguite, NON funzioni o stati

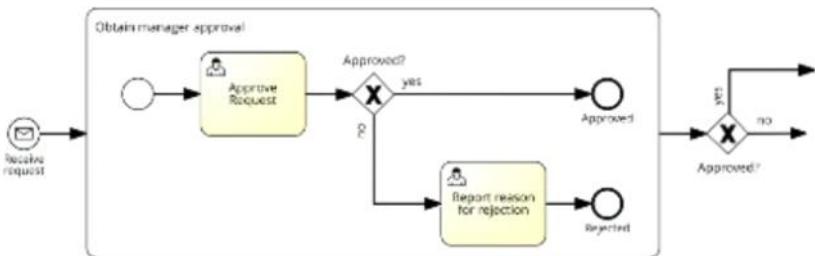
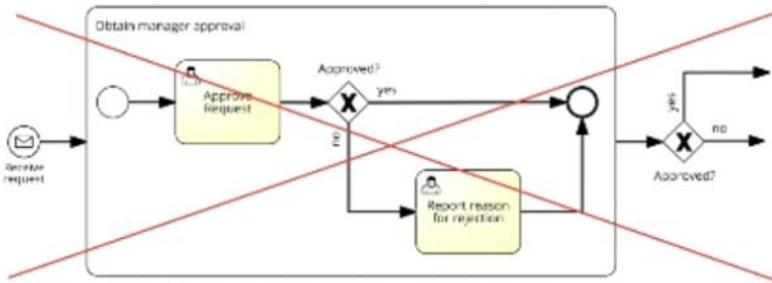
Esempio:

- Credito OK vs **Controlla credito**
- Mutuo rifiutato vs Mutuo approvato vs **Approva mutuo**
- Report ricevuto vs **Ricevi report**

#### 9. Utilizzare degli start event trigger nel processo top-level per indicare come inizia il processo.

- a. Messaggio: richiesta esterna
  - i. Etichetta: *Ricevi [nome del flusso del messaggio]*
- b. Timer: processo pianificato (ricorrente)
  - i. Etichetta: regola di pianificazione
- c. Nulla: Avvio manuale dall'esecutore del processo
  - i. Generalmente non etichettato

#### 10. Se un sottoprocesso è seguito da un gateway con una domanda, il sottoprocesso dovrebbe avere un evento terminale per ogni risposta del gateway.

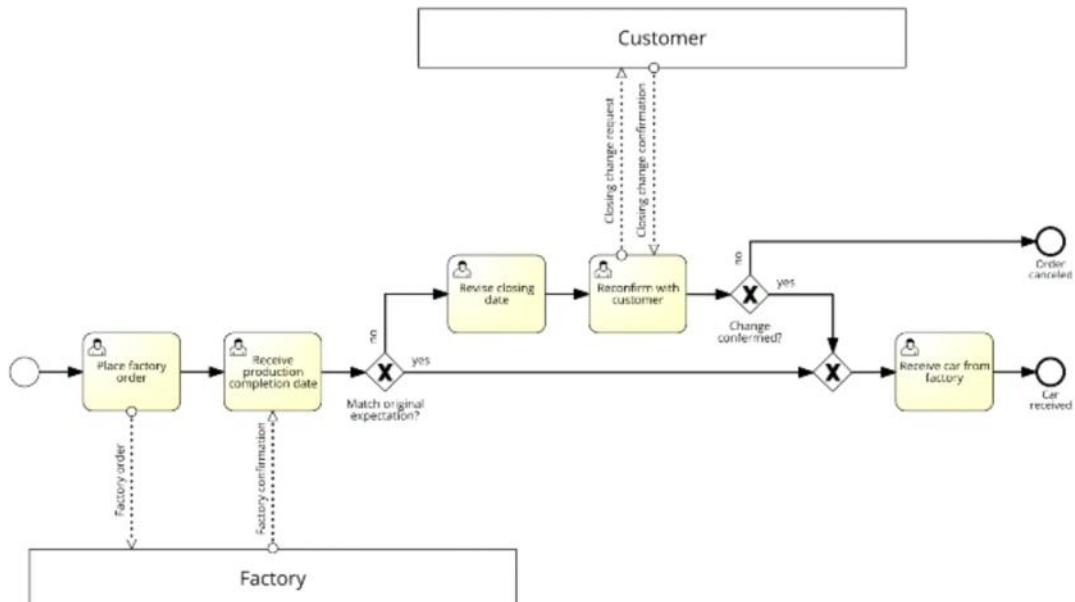


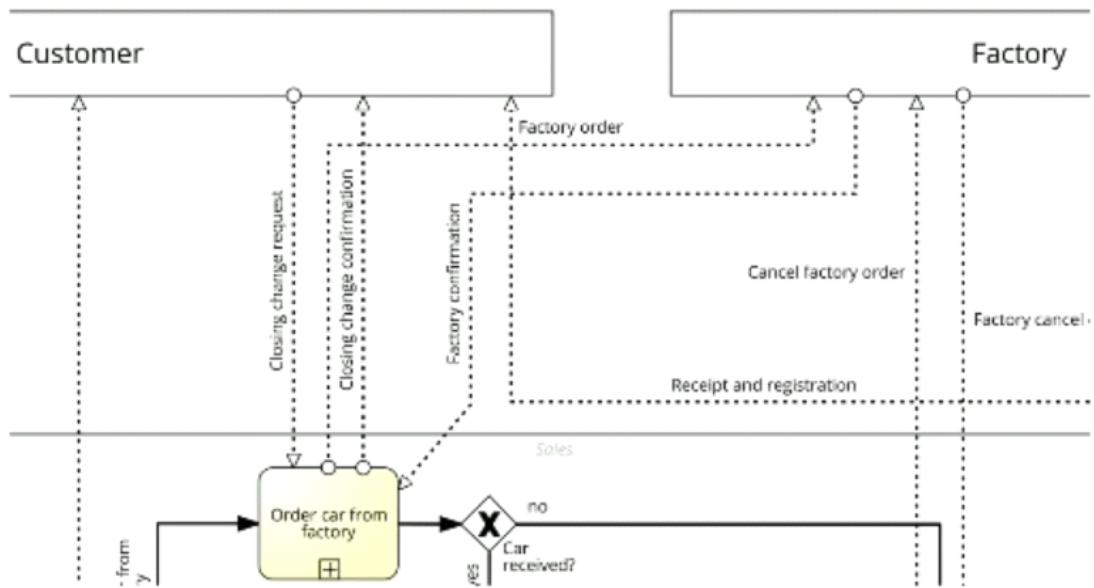
**11. Mostrare il flusso dei messaggi, con tutti gli eventi associati**

- a. Il flusso dei messaggi è opzionale in BPMN
- b. Tuttavia modellarli esplicitamente rende chiare le modalità di interazione e il contesto di business

**12. Far corrispondere il message flow dei diagrammi di alto livello con quello dei diagrammi figli**

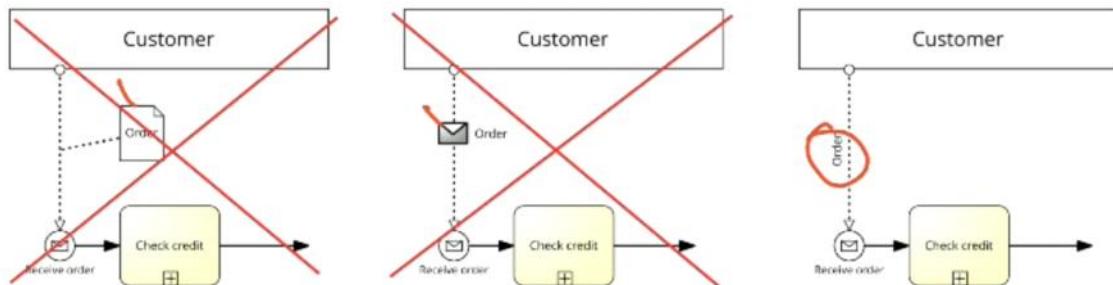
- a. Necessario per tracciabilità!
- b. La corrispondenza è intesa in termini di numero ed etichette
- c. Top-down: tutti i messaggi associati a un sottoprocesso collassato dovrebbero essere precisamente "collocati" all'interno del processo figlio
- d. Bottom-up: tutti i messaggi utilizzati in un sottoprocesso dovrebbero essere mostrati adeguatamente nel livello padre.





**13. L'etichettatura del message flow la si fa col nome del messaggio**

- Un flusso di messaggio consiste di un messaggio (ad esempio, avviso di rifiuto)
- Un messaggio non è uno stato (ad esempio, rifiutato)
- Un messaggio non è l'azione di inviarlo o riceverlo (ad esempio, invia rifiuto)
- Importante:** in BPMN non è possibile collegare un data object a un flusso di messaggio.

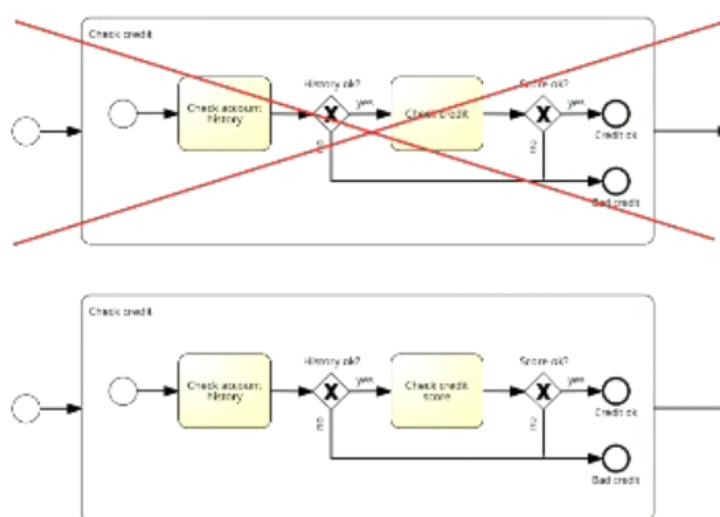


**14. Due eventi di terminazione nello stesso livello di un processo non dovrebbero avere lo stesso nome.**

- Rappresentano stati di terminazione diversi? Allora servono due etichette diverse!
- Sono effettivamente lo stesso stato? Allora combinarli.

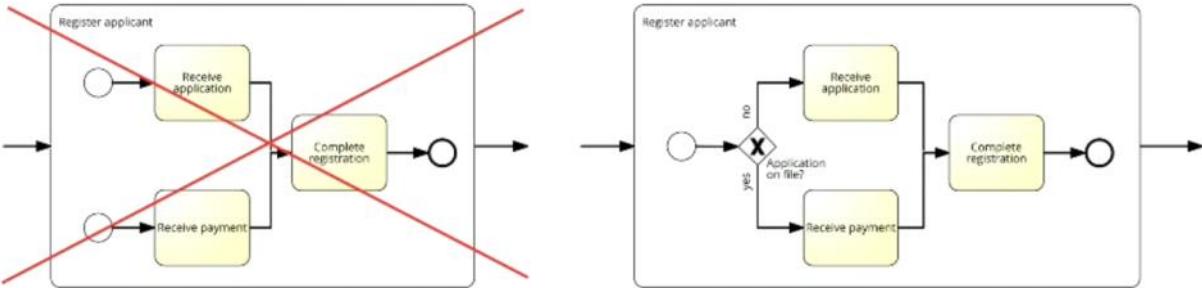
**15. Due attività in un processo non dovrebbero avere lo stesso nome.**

- Sono la stessa attività? Allora usa una **chiamata di attività** che fa riferimento alla stessa task o processo globale.
- Sono diversi? Allora usare label diverse
- Non è raro vedere il nome di un sottoprocesso ripetuto in una delle sue task

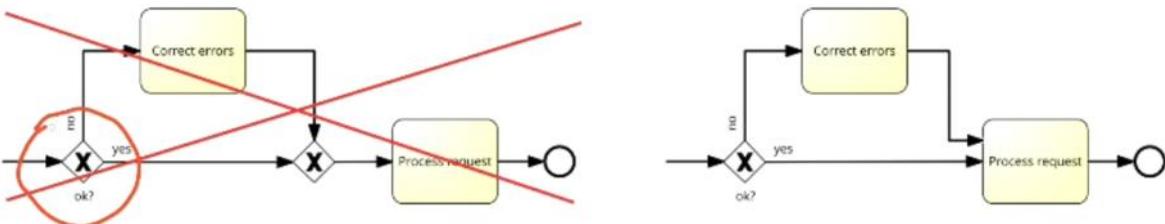


**16. Un sottoprocesso dovrebbe avere un singolo evento di partenza Nullo (none start event)**

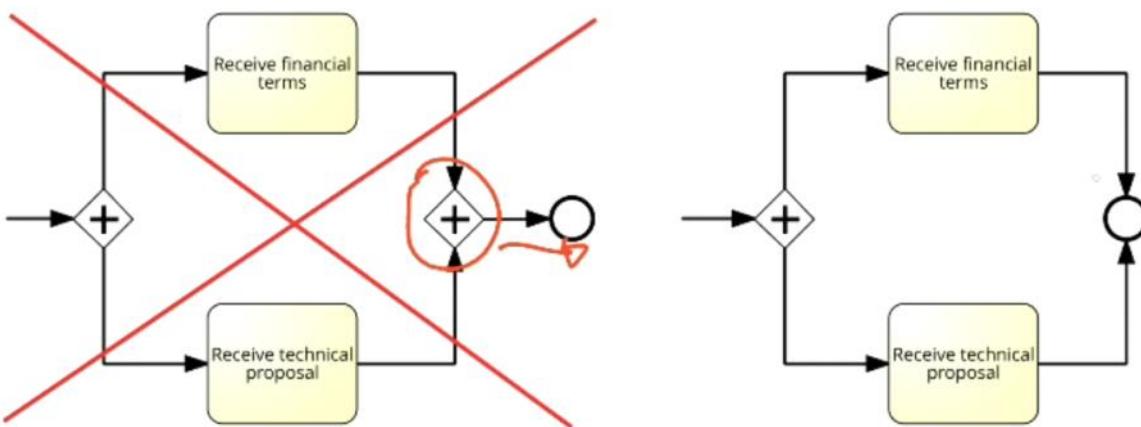
- a. Multipli start event sono permessi nel processo di top-level per distinguere diverse modalità o punti di partenza.
- b. In un sottoprocesso, multipli **none start events** creano ambiguità
  - i. Denotano punti di partenza alternativi? Allora modellare la decisione esplicitamente
  - ii. Denotano rami paralleli? Allora renderlo più chiaro.



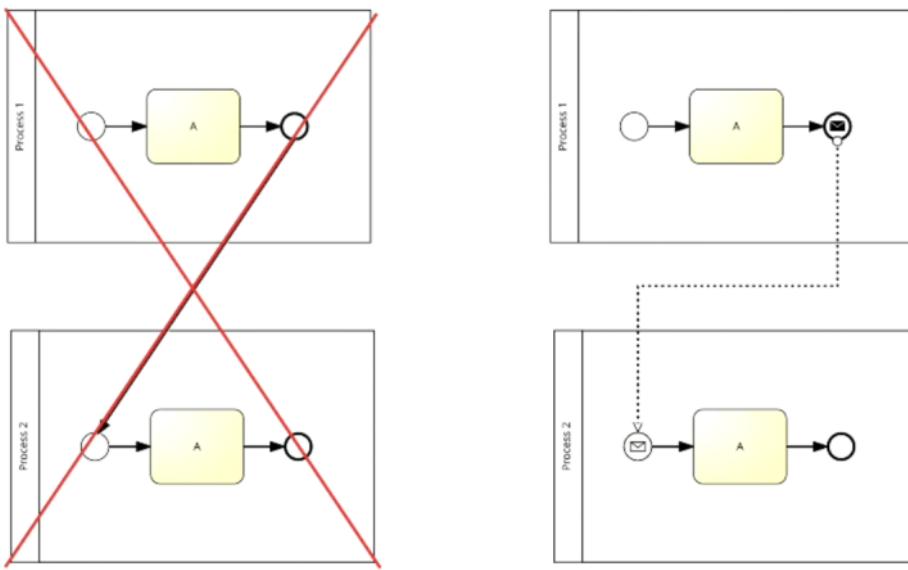
17. Se un pool è disegnato anche in un sottoprocesso, allora l'etichettatura del sottoprocesso deve rispecchiarsi nel processo padre
18. Un processo di un livello più basso non dovrebbe mai contenere elementi dei livelli più alti.
19. Non usare un gateway XOR per unire diversi percorsi alternativi, eccetto se si entra in un altro gateway. Collegare direttamente i vari flussi assieme. (per alleggerire il diagramma)



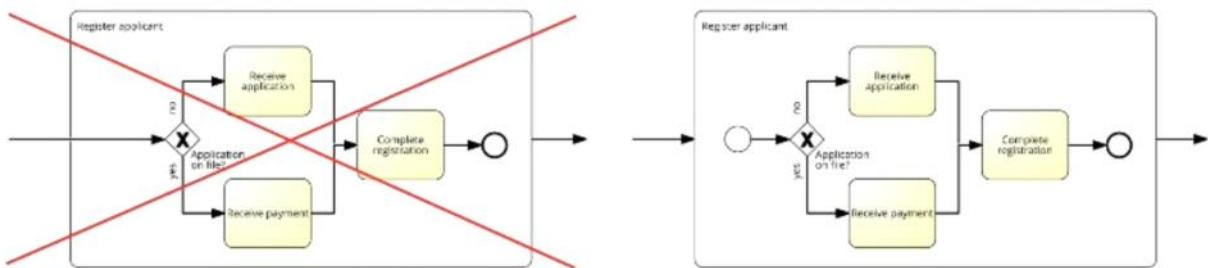
20. Non utilizzare un gateway AND per unire cammini paralleli in un evento di terminazione nullo.
  - a. Il join è sempre inteso in un evento di terminazione nullo



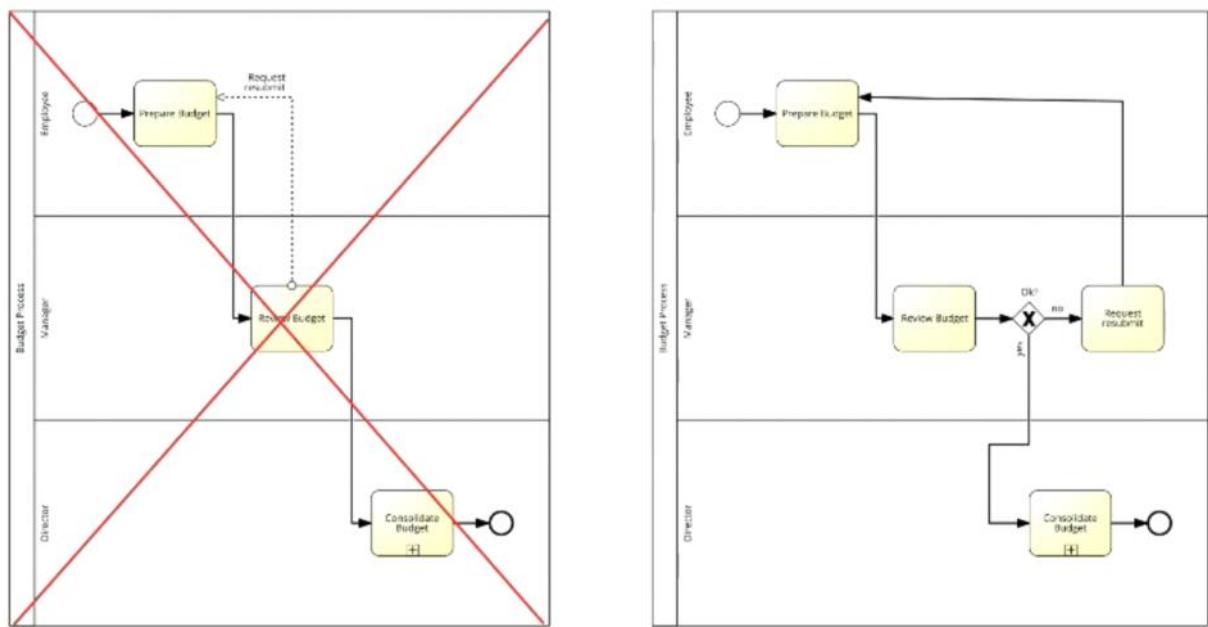
21. Un flusso di sequenza non può uscire dai limiti di una pool.



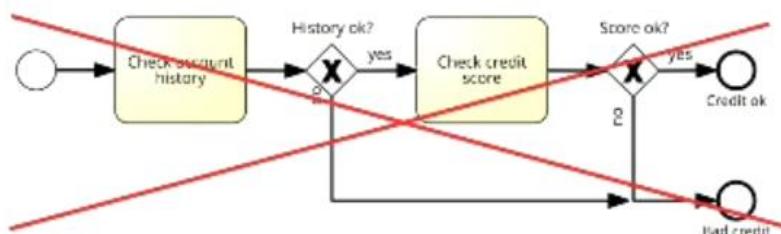
**22. Un flusso di sequenza non può entrare nei limiti**

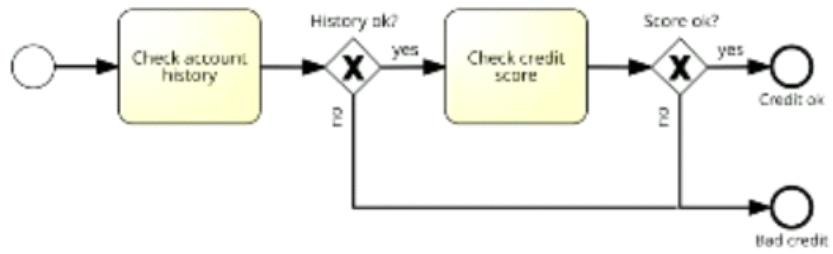


**23. I message flow non possono connettere nodi nella stessa pool.**

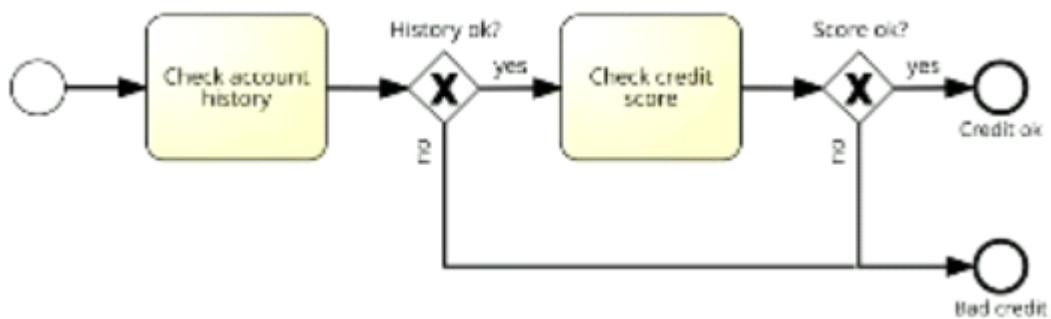


**24. Un sequence flow può solo connettere un'attività, un gateway o un evento. Entrambe le estremità devono essere connesse correttamente.**





25. Un message flow può solo connettere un'attività, un messaggio o multipli eventi o una black-box pool. Entrambe le estremità devono essere connesse correttamente.
- Nessun message flow può puntare ai limiti di una process pool, ad un data store o a un gateway.



## 2° Livello modellazione analitica - 24 Nov

venerdì 22 gennaio 2021 12:25

Il secondo livello espande il primo con le reazioni agli eventi.

Un evento è qualcosa che succede in un processo in uno specifico punto nel tempo (messaggio che arriva, errore, segnalazione...)

Gli eventi sono un modo per guidare le attività stesse.

Il diagramma di flusso non si occupa di rappresentare gli eventi che possono occorrere durante l'esecuzione, mentre il BPM sì! Molte attività sono collegate ad eventi e causano eventi.

La reazione dipende dal processo: se lo lancia o lo cattura.

Se si reagisce lanciando un segnale, il processo lo genera e rappresenta un trigger.

Se si reagisce catturando un evento, il segnale è un risultato

L'icona è un cerchio con l'evento all'interno.

Ricordiamo:

- Eventi di start: cerchio sottile
- Eventi di end: cerchio spesso
- Evento intermedio: doppio cerchio. Indica che qualcosa succede durante l'esecuzione del processo.

Con questo evento intermedio possiamo modellare reazioni ad eventi esterni, alle eccezioni, gestione parallela...

## Events

	Start	Event Sub-Process Interrupting	Event Sub-Process Non-Interrupting	Catching	Intermediate	End	
Standard					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.							

# Events

	Standard	Start	Event SubProcess Interrupting	Event SubProcess Non-Interrupting	Catching	Intermediate	Boundary Interrupting	Boundary Non-Interrupting	Throwing	End
	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard
<b>Error:</b> Catching or throwing named errors.	(W)				(W)		(W)		(W)	
<b>Cancel:</b> Reacting to cancelled transactions or triggering cancellation.					(X)		(X)		(X)	
<b>Compensation:</b> Handling or triggering compensation.	(L)				(L)		(L)		(L)	
<b>Signal:</b> Signalling across different processes. A signal thrown can be caught multiple times.	(T)	(T)	(T)	(T)	(T)	(T)	(T)	(T)	(T)	
<b>Multiple:</b> Catching one out of a set of events. Throwing all events defined	(P)	(P)	(P)	(P)	(P)	(P)	(P)	(P)	(P)	
<b>Parallel Multiple:</b> Catching all out of a set of parallel events.	(+)	(+)	(+)	(+)	(+)	(+)	(+)	(+)		
<b>Terminate:</b> Triggering the immediate termination of a process.										(Black circle)

# Events

	Standard	Start	Catching	Intermediate	Throwing	End
	Standard	Standard	Standard	Standard	Standard	Standard
<b>None:</b> Untyped events, indicate start point, state changes or final states.	(Circle)					(Circle)
<b>Message:</b> Receiving and sending messages.	(Envelope)					(Envelope)
<b>Timer:</b> Cyclic timer events, points in time, time spans or timeouts.	(Clock)					(Clock)
<b>Error:</b> Catching or throwing named errors.			(W)			(W)
<b>Terminate:</b> Triggering the immediate termination of a process.				(W)		(Black circle)

Oltre 104 eventi, non tutti approvati.

Gli eventi possono occorrere dopo la partenza del processo, ma prima della sua fine.

I contesti di uso sono 4:

1. Lancio di un evento intermedio
2. Cattura di un evento intermedio

3. Cattura di un evento intermedio in uno specifico livello di un processo, interrompendolo
4. Cattura di un evento intermedio in uno specifico livello di un processo, senza interruzione

(i livelli sono quelli dei sottoprocessi)

#### (1) LANCIO DI EVENTI INTERMEDI



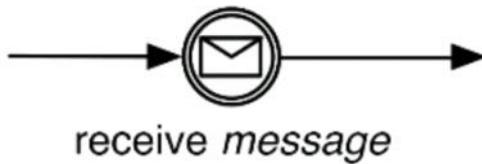
Intermedio perché si trova all'interno di un sequence flow.

Appena il processo raggiunge l'evento, il segnale è lanciato e il processo prosegue.

È da considerarsi come l'invio di un messaggio a tempo 0.



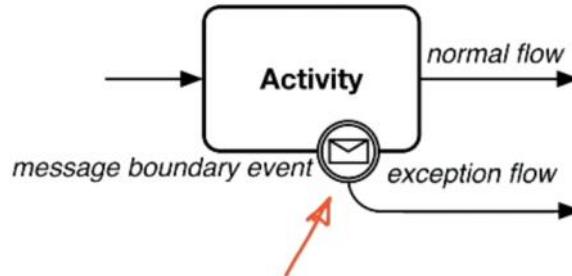
#### (2) CATTURA DI EVENTI INTERMEDI



L'icona al suo interno è bianca!!!

Qui il significato è che bisogna catturare un evento. Quando il sequence flow raggiunge l'evento, si mette in attesa. Non sono catturabili gli eventi di errore.

#### (3) Boundary Event

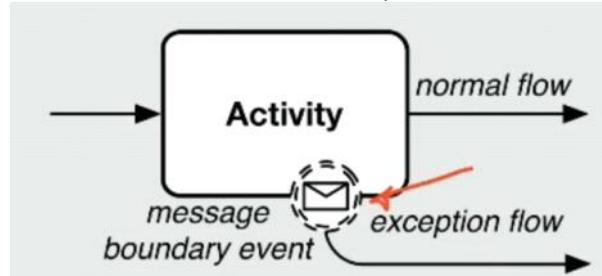


Mentre l'attività è in esecuzione, ascolta al segnale agganciato all'evento.

Se l'attività termina senza che il segnale occorra, il processo continua sul percorso standard.

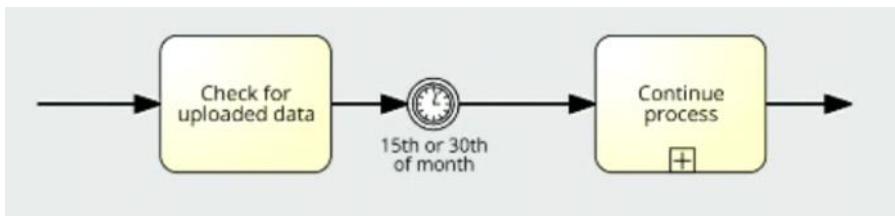
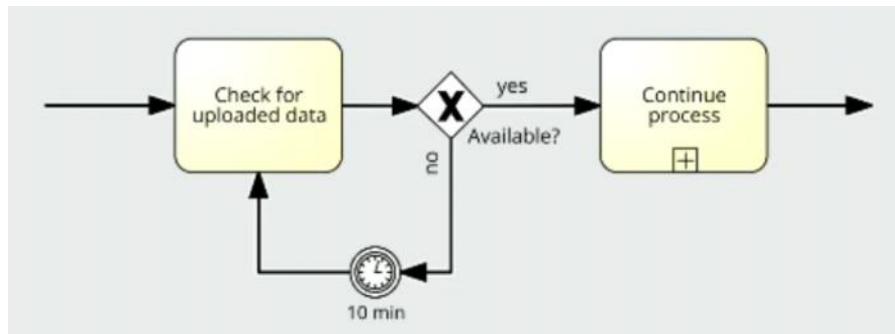
Se invece il segnale occorre prima del termine dell'attività, viene scatenata un'eccezione.

È possibile avere anche delle eccezioni che non interrompono:



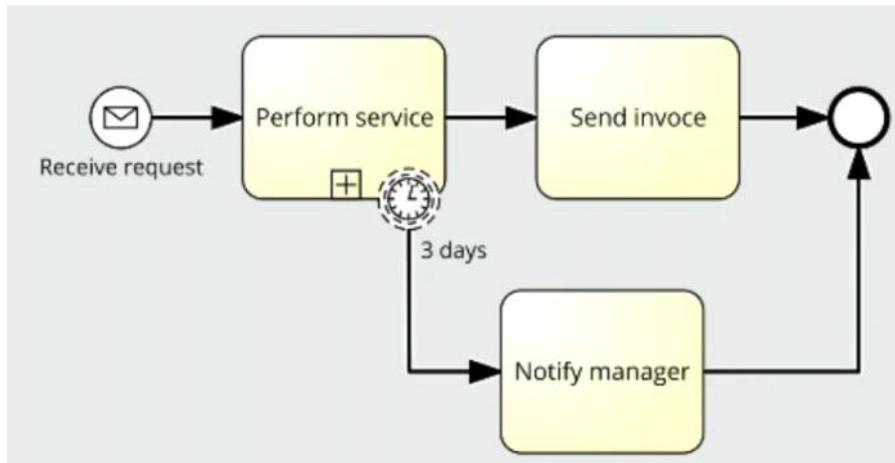
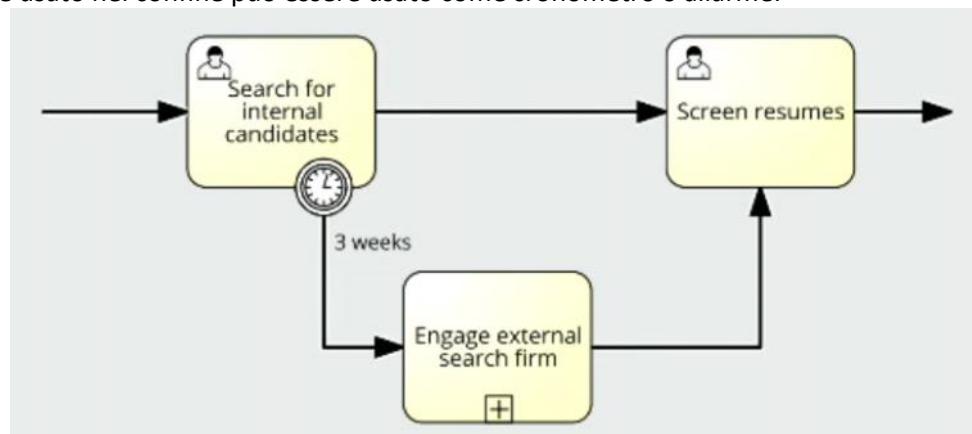
Se il segnale viene attivato durante l'esecuzione dell'attività, viene proseguita senza problemi e il flusso dell'eccezione viene attivato con un nuovo thread parallelo.

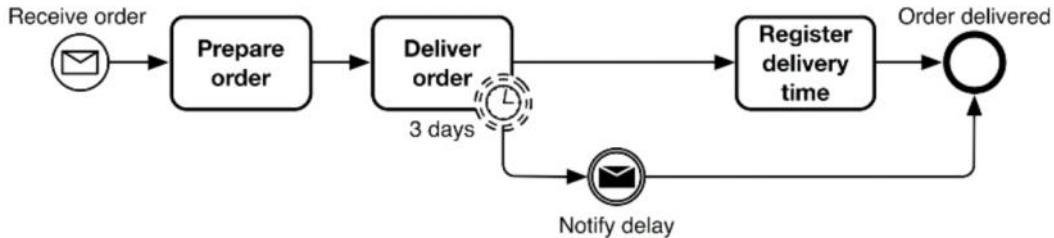
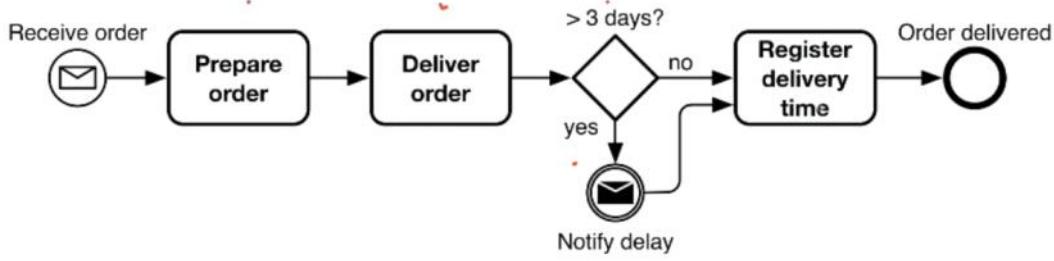
### Timer



NOTA: I timer NON devono essere usati per indicare la durata dell'attività!

Il timer se usato nel confine può essere usato come cronometro o allarme.





Il primo processo rappresenta il fatto che dobbiamo consegnare l'ordine. Una volta successo, inviamo la notifica. Il processo non rappresenta affatto l'intenzione, che è di notificare in tempo!

### Messaggi

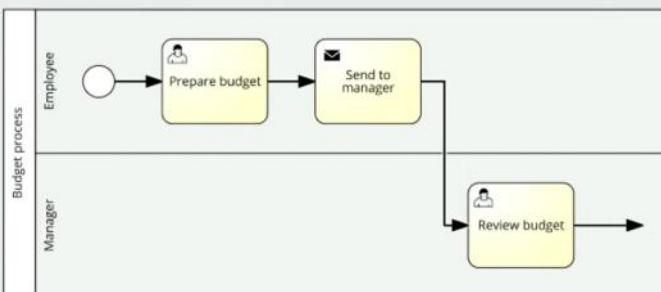
Lo scambio di un messaggio può anche rappresentare sia elementi fisici, sia immateriali (tipo del lavoro)



Un evento non rappresenta l'esecuzione di un task da una persona, mentre invece la Send Task sì.

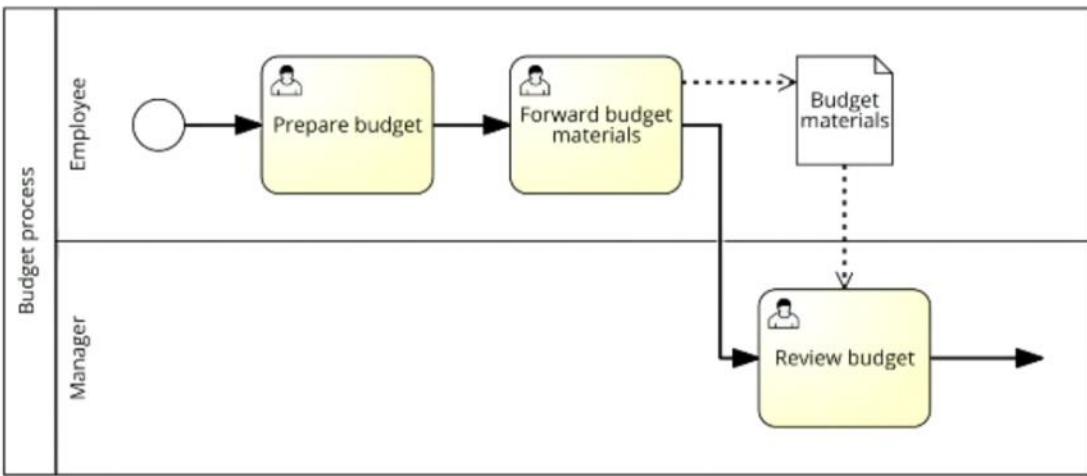
### Error

Message flow cannot be used to forward work to a downstream task within the same process.

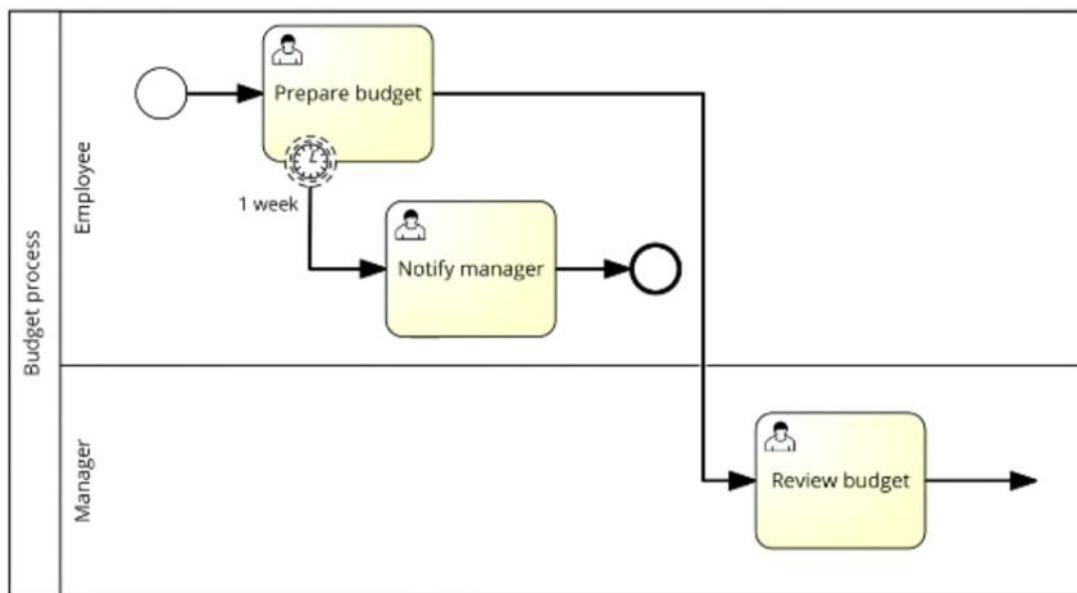


Non va usato per indicare che trasferisco delle informazioni nello stesso processo!

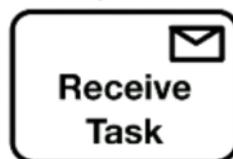
Visto che vorremmo rappresentare l'invio di del materiale, possiamo utilizzare dei simboli appositi



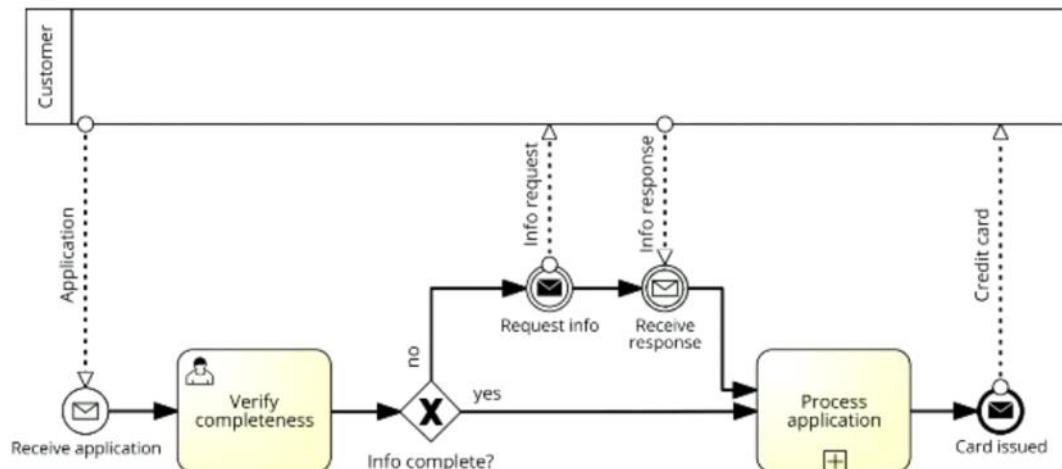
Il send va utilizzato per rappresentare una comunicazione verso l'esterno.



Nel caso della ricezione, abbiamo un responsabile che attende la ricezione del messaggio.



Come posso rappresentare il fatto che attendo una risposta SINCRONA prima di poter continuare?

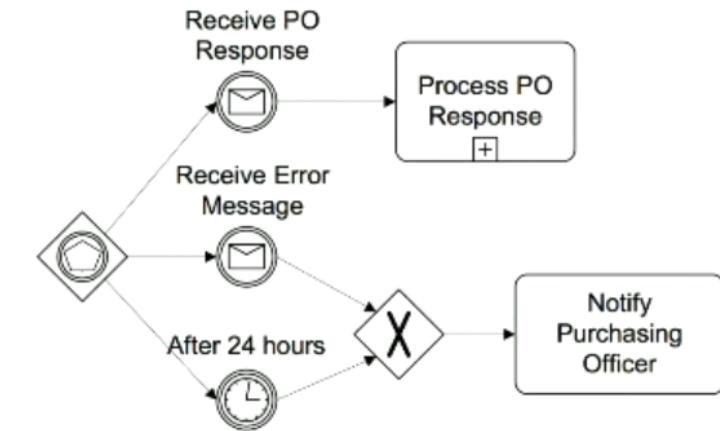


Posso utilizzare delle scelte guidate dagli eventi:

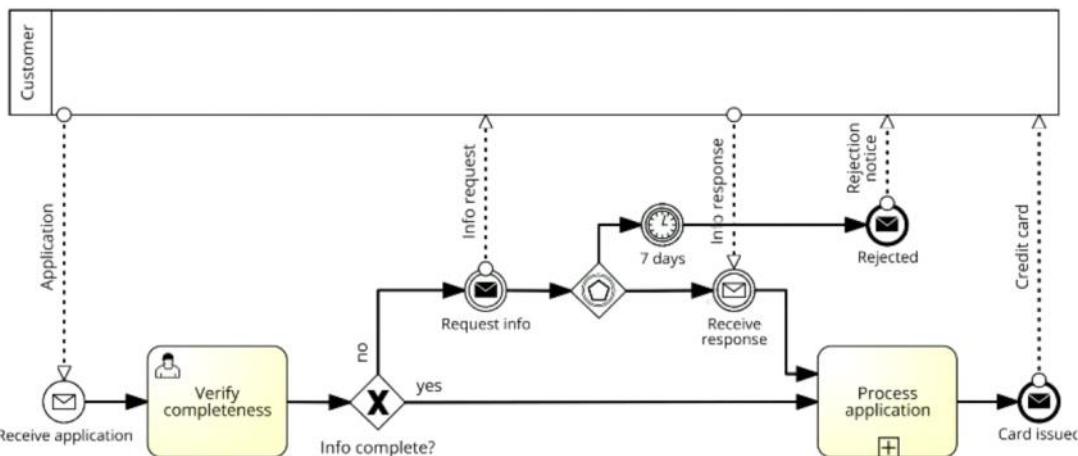
- Messaggi alternativi (potenzialmente da partecipanti diversi)
- Timeout

Non possiamo rappresentarlo con un normale choice gateway.

Quello che ci serve è un event gateway.

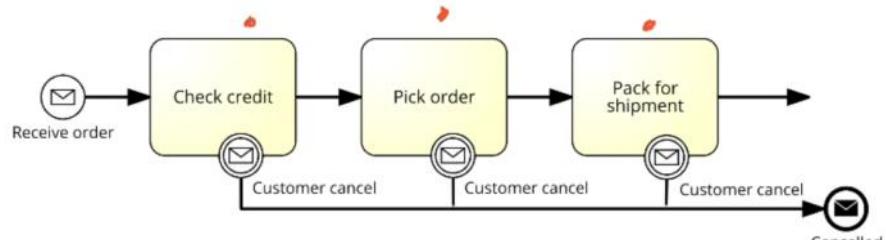


Questo gateway fa in modo che il primo degli eventi che scatta è l'unico che procede, esattamente come lo XOR.

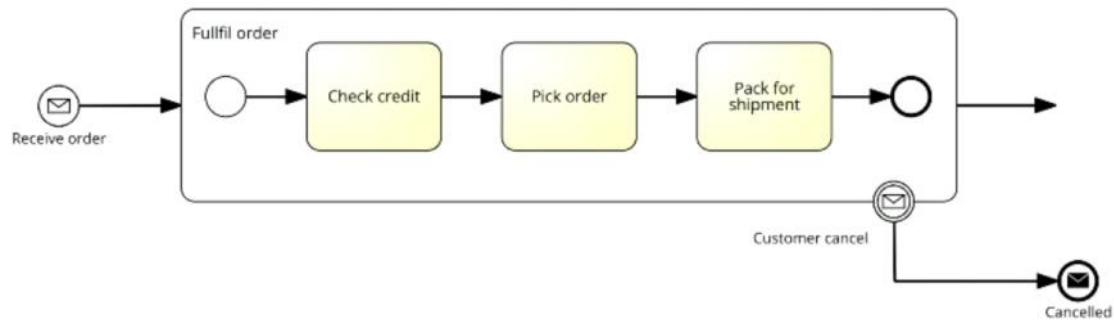


Altro esempio

Management of **unsolicited messages**.



**NO!!**



La strategia utilizzata è:

- Identificare il numero massimo di frammenti di un livello del processo per cui la gestione di un evento è la stessa
- Circondare questi frammenti con un sottoprocesso e aggiungere un boundary event con un flusso per l'eccezione, in modo da rappresentare come l'evento è gestito quando occorre in quella parte del processo.

# Evento di errore - 25 Nov

sabato 23 gennaio 2021 14:58

Uno stato di terminazione eccezionale.

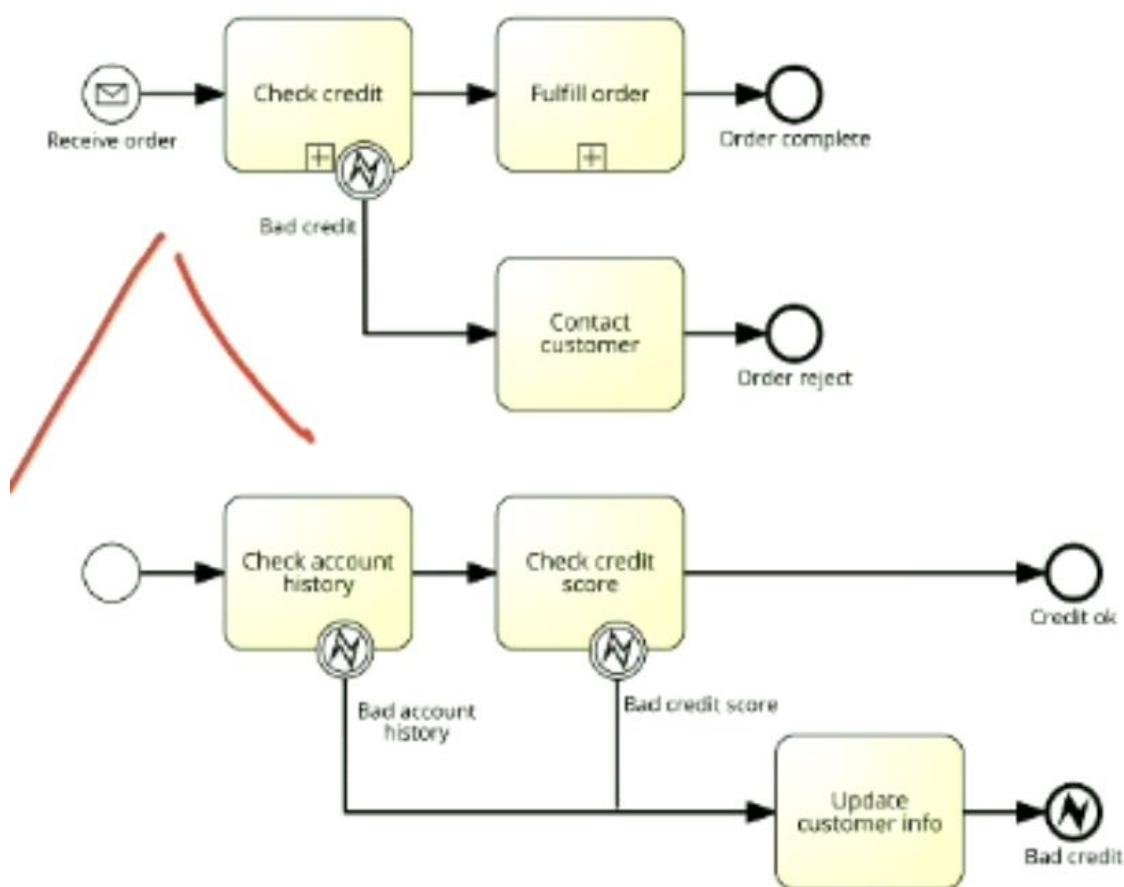
Può essere presente come boundary event, che interrompe. Sta a indicare un codice errore e quale errore è gestito se l'esecuzione dell'attività scaturisce quell'errore.

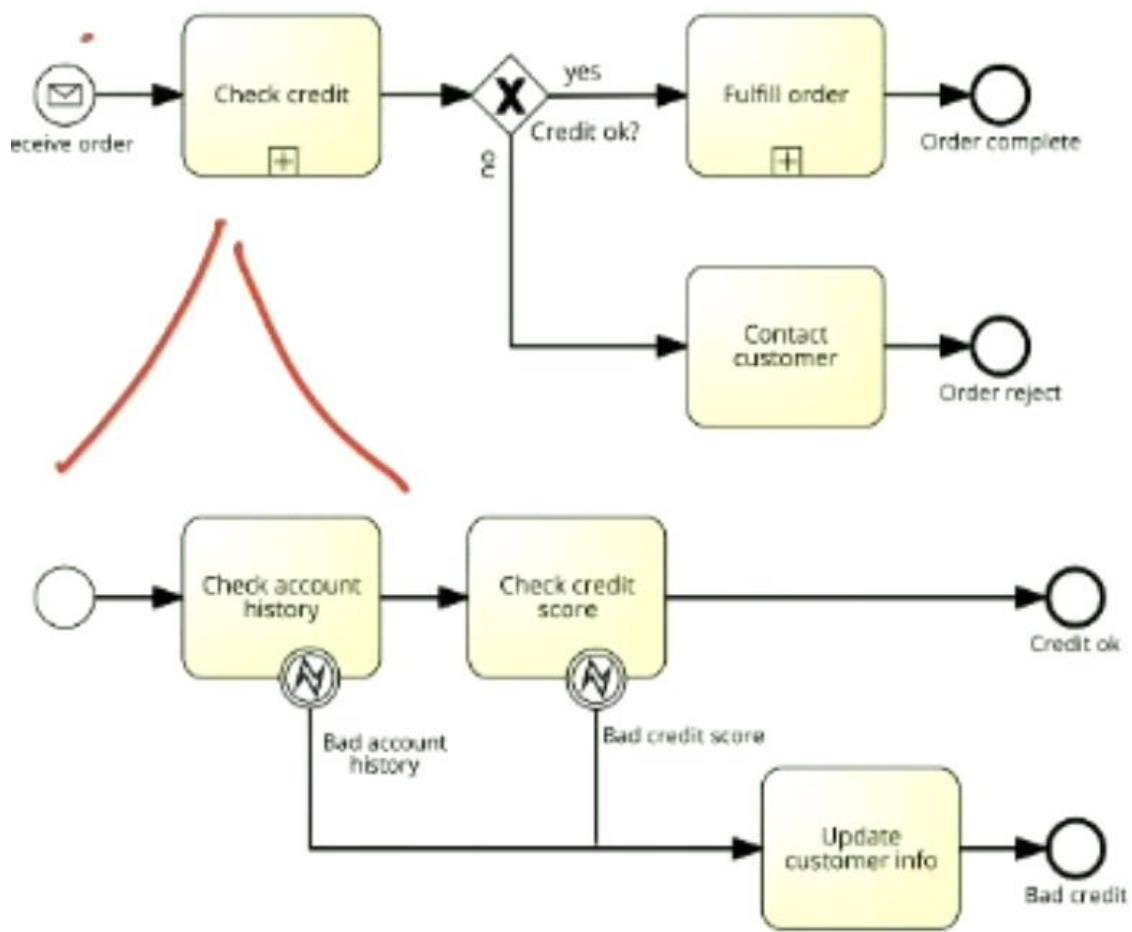
- Su una task: rappresenta il flusso dell'eccezione che deve essere seguito quando la task non termina con successo
- Su un sottoprocesso: stesso significato, ma richiede una specifica più precisa all'interno (serve un end state all'interno che indichi l'errore)

L'etichetta dell'errore è il codice errore.

Quando un processo raggiunge un livello in cui occorre un errore:

1. Tutti i thread paralleli all'interno dello stesso livello del processo sono terminati immediatamente
2. Il corrispondente segnale d'errore è generato e propagato verso il livello superiore
3. Se il livello superiore ha un error boundary con lo stesso codice d'errore, il segnale è catturato e il flusso dell'eccezione viene attivato
4. Altrimenti, il segnale è propagato ricorsivamente ai livelli superiori, finché il livello più vicino in grado di gestire l'errore alla fine lo cattura.





Possono essere entrambi giusti, ma dipende dalla semantica che vogliamo utilizzare.  
Usare il pattern throw-catch mette più enfasi sul fatto che si tratta di un errore.

# Escalation event - 25 Nov

sabato 23 gennaio 2021 16:32

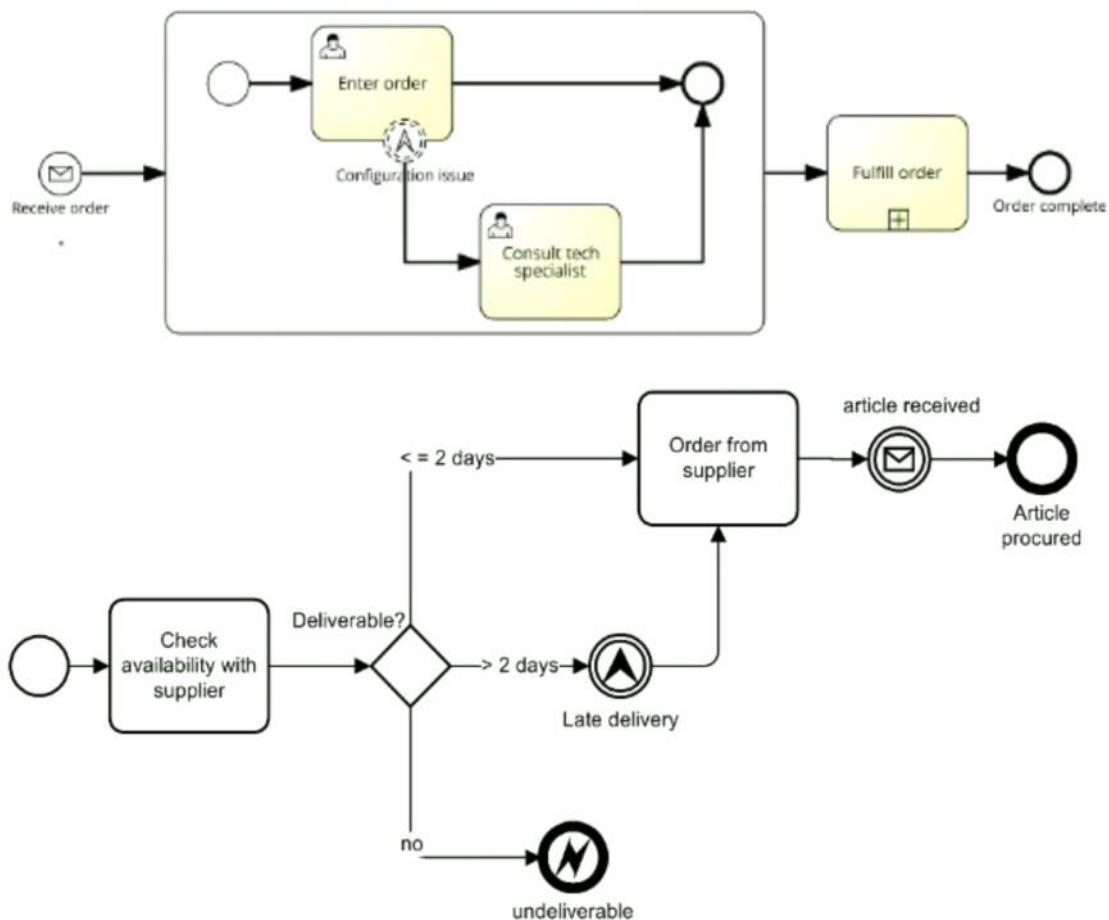
Rappresentato con una freccia che punta verso l'alto.

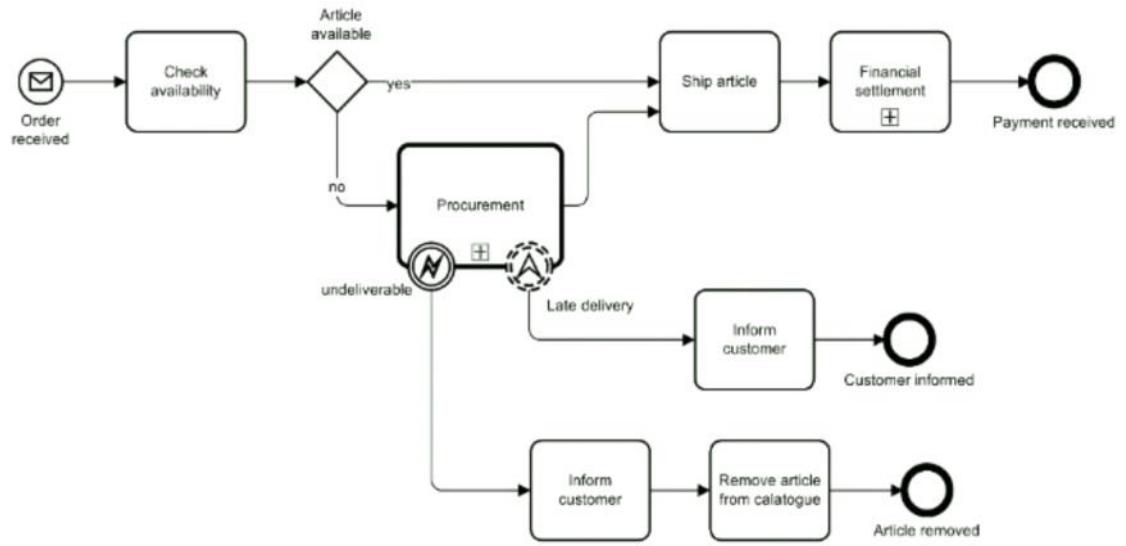


È utilizzata per rappresentare una specie di warning.

Può essere attivata nel mezzo di un livello di un processo (a differenza degli errori che possono essere attivati solo alla fine)

L'escalation è catturata, ma senza interrompere il livello attivo (in genere).





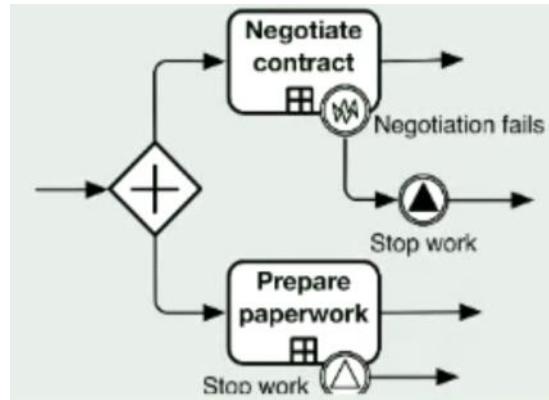
# Signal Event - 25 Nov

sabato 23 gennaio 2021 16:38

Rappresentato come un triangolo.

È come l'escalation, ma il segnale è mandato in broadcast a tutti i thread attivi e a tutti i partecipanti esterni.

- Intra-processo: permette di accoppiare debolmente i thrower e i catcher, con la possibilità di raggiungere thread paralleli.
- Inter-processo: codifica un paradigma del tipo Pubblicatore / Sottoscrittore (il segnale è inviato a tutti i partecipanti interessati).



Possiamo avere catchers più flessibili e reazioni più flessibili. Possiamo usarlo anche sullo stesso livello.

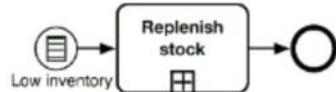
# Conditional Event - 25 Nov

sabato 23 gennaio 2021 16:47

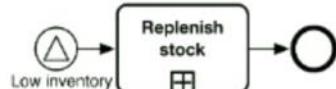
Monitora continuamente una condizione per dei dati o una regola di business.  
Genera un trigger ogni volta che delle condizioni sono vere.

Replenishing stock when inventory is low...

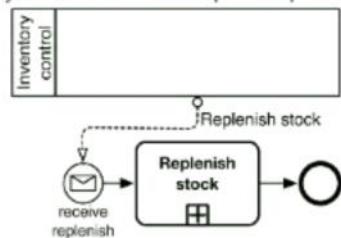
- Internal condition evaluation.



- External condition evaluation (by anybody).

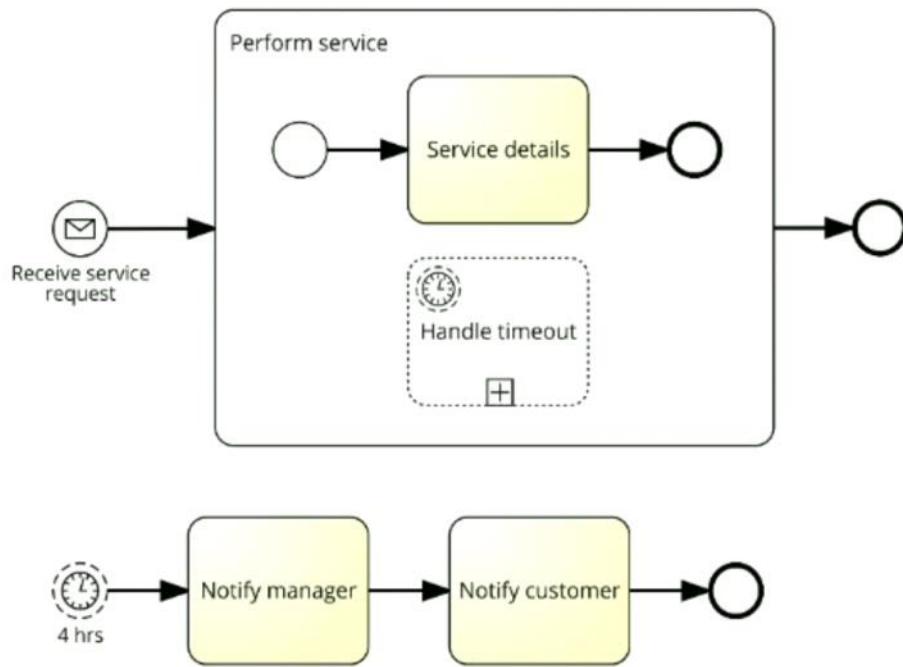


- External condition evaluation by the mentioned participant.



# Event Subprocess - 25 Nov

sabato 23 gennaio 2021 16:49



Può essere pensato come l'handler di un evento associato ad un processo.

- È attivato da un messaggio, un timer o un errore.
- Può interrompere il task
- Graficamente, ha il cerchio tratteggiato e:
  - o Collassato, ha in un angolo l'icona del tipo di trigger che si aspetta
  - o Espanso, usa l'icona dell'evento come evento di start

# Decoratori delle attività - 25 Nov

sabato 23 gennaio 2021 16:57

	Basic task.
	Compensation task.
	Loop task (looping information attached to the activity).
	Multi-instance task with parallel composition (expression attached to the activity to calculate the number of instances).
	Multi-instance task with sequential composition.

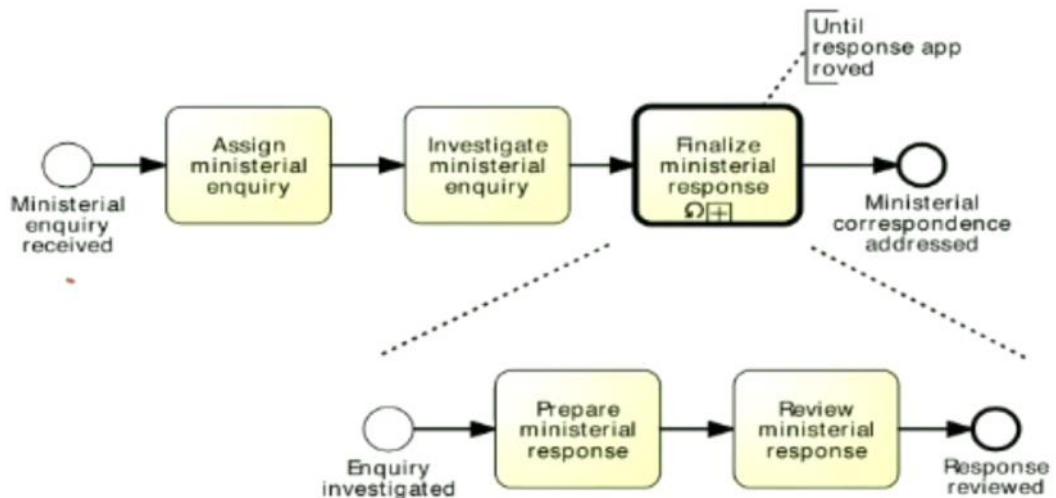
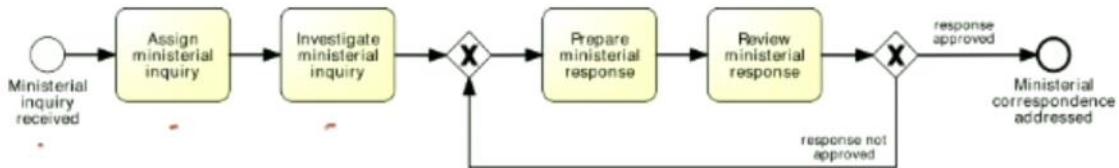
For sub-processes only: **ad-hoc** (tilde marker) - flexible execution of the inner activities, without a complete specification of the process.

Loop:

Un'attività che può essere ripetuta molte volte.

- La condizione è del tipo "until X"
- Un'annotazione testuale è utilizzabile per mostrare graficamente la condizione di loop
- Le iterazioni sono sequenziali
- Il numero di iterazioni è stabilito dinamicamente

Corrisponde a una normale attività seguita da un gateway che ricollega all'inizio se la condizione è verificata

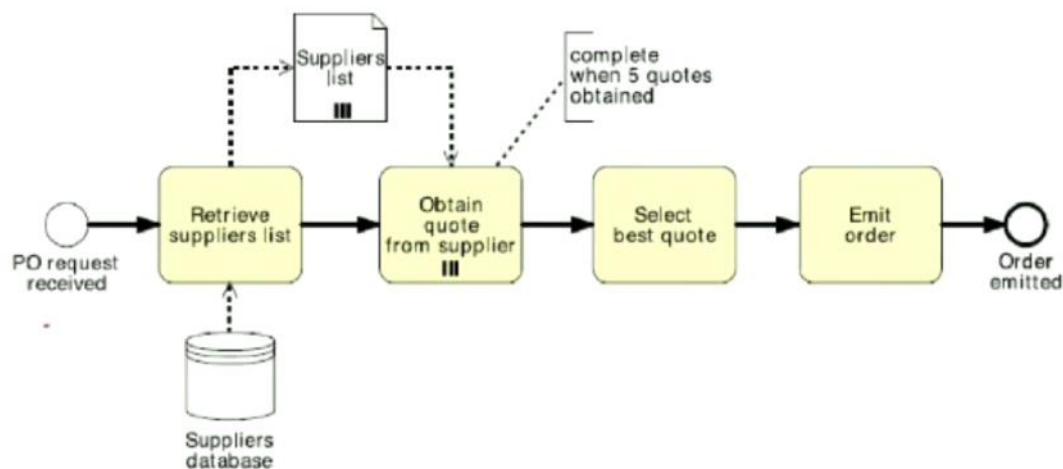


Multi-istanza

Identificato da un'attività con tre barre parallele.

Può servire per indicare per quanti elementi deve essere effettuata un'azione. Può essere effettuata sequenzialmente o in parallelo.

Termina quando tutte le istanze sono complete, oppure se occorre un'eccezione.



### Strategia per l'uno a molti

1. Identificare tutte le attività N:1
2. Circondarle con un loop, MI (attività multi-istanza) o entrambi

Problemi:

- Se in una fase di un processo non sappiamo il numero di istanze in anticipo, possiamo solo utilizzare i loop: pipelining impossibile!
- In alternativa possiamo usare MI, ma il pipelining è ancora impossibile tra le fasi.

Possiamo usare un Multipool Process.

### Example

Consider again the hiring process.

- “Post a job” is 1:1 with the job (case).
- “Evaluate a candidate” is 1:1 with an applicant.

What is the relationship between applicant and job?

- N:1 (isolation): each applicant sends CV for a specific job.
  - Single pool solution requires looping + MIs, with pipelining issues.
  - Multipool solution avoids this.
- N:M (sharing): each applicant sends CV, and the company looks whether there are matching jobs for that CV.
  - Multipool separation necessary.

## Example

We revisit the hiring process using one external pool for the applicant, and two process pools: **Evaluate Candidate** and **Hiring Process**.

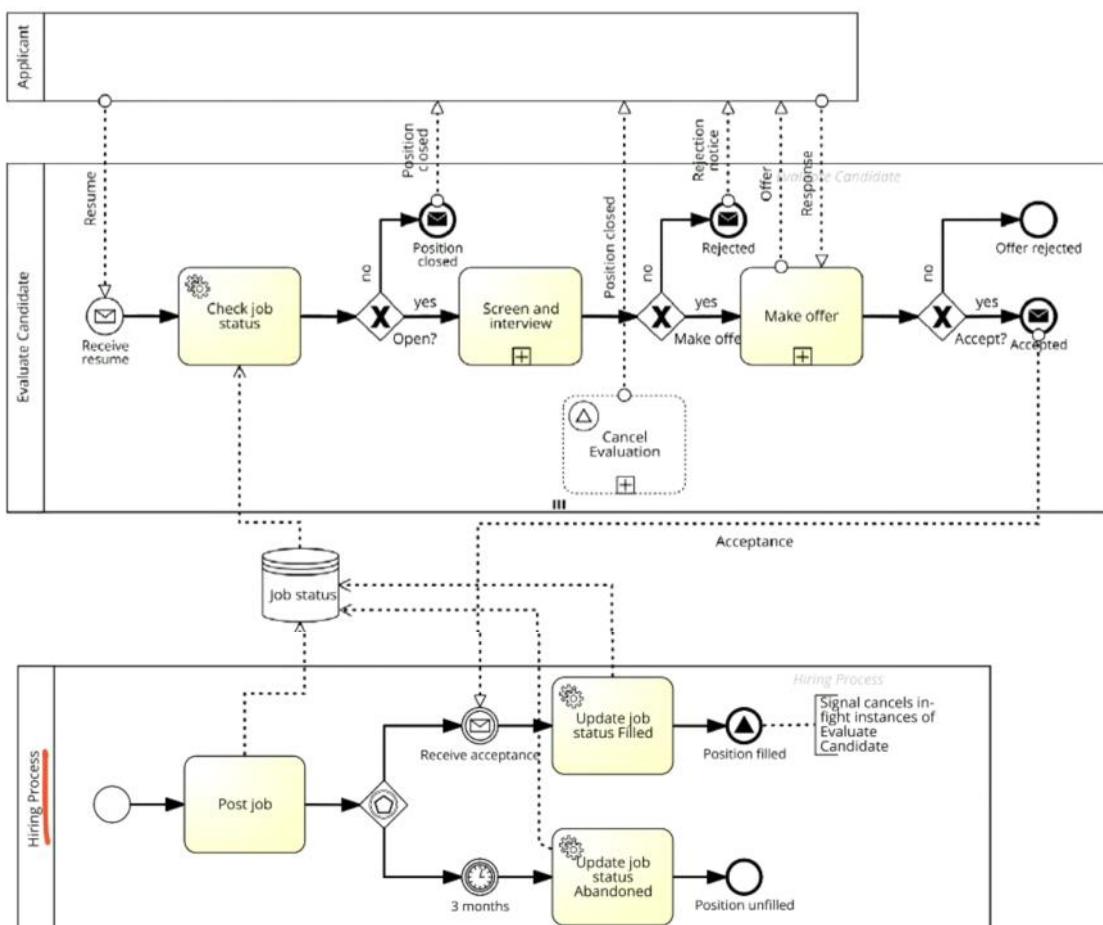
The Hiring process starts autonomously by posting a job. If the job is successfully assigned to someone, it is marked as "filled". If this is not the case within 3 months, the job is marked as "abandoned".

The Evaluate candidate process starts when a resume is received by the company from an applicant. We assume that the resume is for a specific job. If the job is not currently offered, the process terminates by notifying this to the applicant. If it is open, a subprocess "screen and interview" is invoked. Then the company decides whether to make an offer to the candidate or not. If not, this is notified to her. If so, then the offer is made to the candidate, who can then accept or reject the offer. As soon as the job is filled, the evaluation of candidates must be interrupted.

## Guideline

Hiring process could be organized as follows:

- 1 external pool for the Applicant, two process pools: Hiring Process and Evaluate Candidate.
- Job Status data store to synch on the job lifecycle.
- Message exchange to send data around and take decisions in agreement with the other pools.
- Signal event + event subprocess to interrupt sibling cases running in the other pools (e.g., to stop evaluating people when the job has been filled).



# Batch Process - 1 Dic

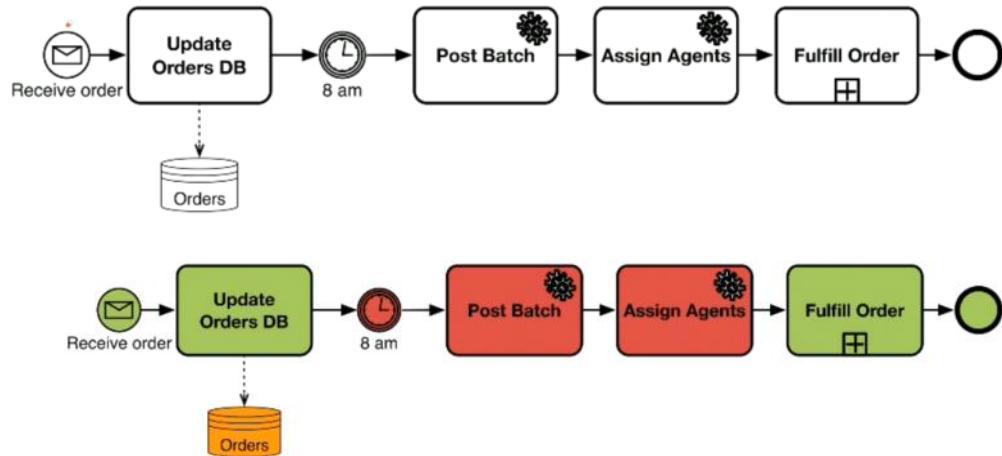
sabato 23 gennaio 2021 18:59

Un processo che opera su un lotto di elementi, trattati da un altro processo.  
Il batch process è attivato a tempo.

## Example

Once an order is received from the customer, it is registered in the "Orders" DB. Orders are posted on a daily basis: at 8am in the morning, all the orders issued during the previous 24 hours are posted. As a consequence of the posting, each order is automatically assigned to an agent, storing this information into the DB. Then the agent takes care of fulfilling the order.

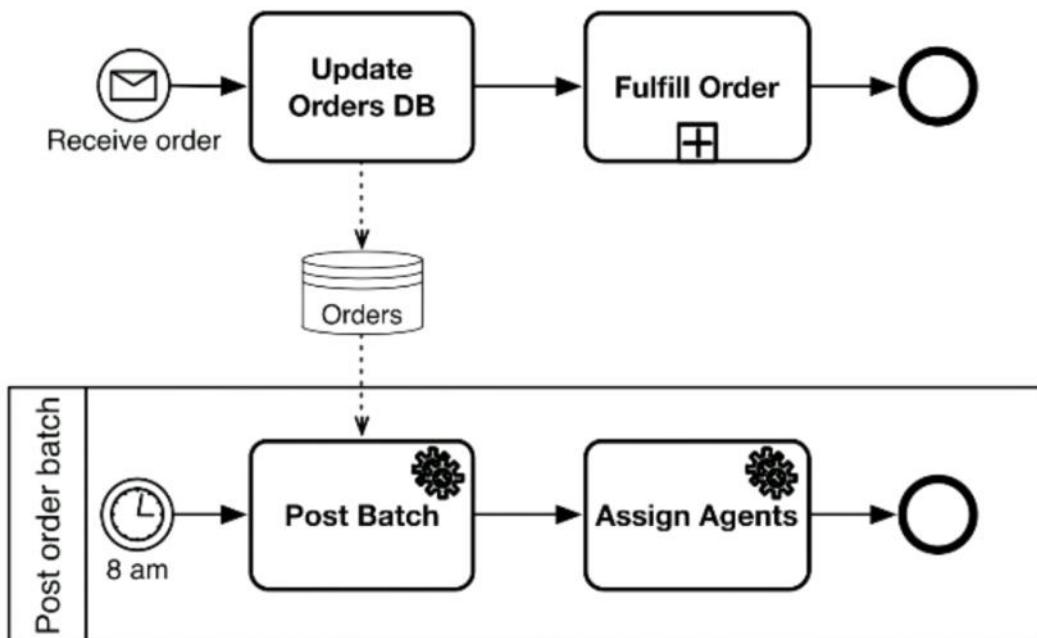
Prima "Soluzione"



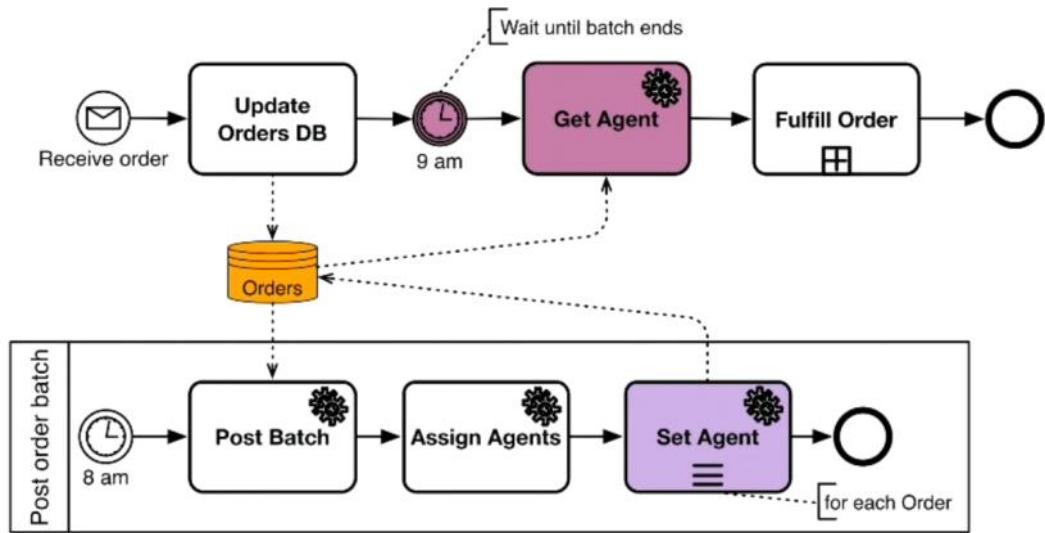
- Green: one order.
- Red: many orders.
- Orange: all known orders.

Con multipli ordini ricevuti, si avranno multipli token, e verranno triggerate multiple batch!

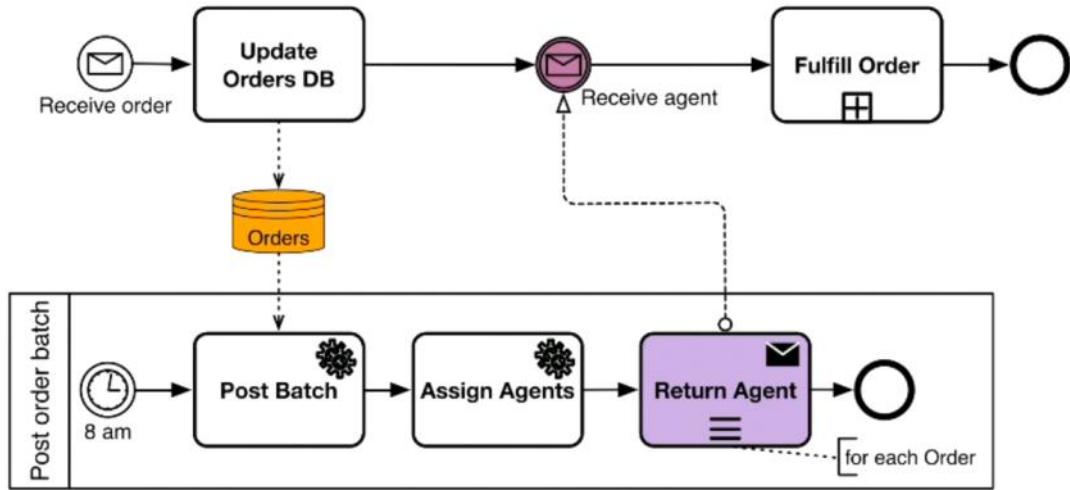
Il primo passo è disaccoppiare l'attività dell'ordine da quella della gestione del lotto di ordini!



Serve una sincronizzazione per bloccare la race condition collegata al completamento degli ordini!



Usiamo un messaggio



# Advanced Split & Merge - 1 Dic

sabato 23 gennaio 2021 19:08

## Claim handling

When a claim is received, it is registered. After registration, the claim is classified leading to two possible outcomes: simple or complex. If the claim is simple, the policy is checked. For complex claims, both the policy and the damage are checked independently.

### Questions

- Can we model the example using the constructs seen so far?
- Can we model it in a compact way?
- Is this a decision? Or an AND-split?

## Contract Check

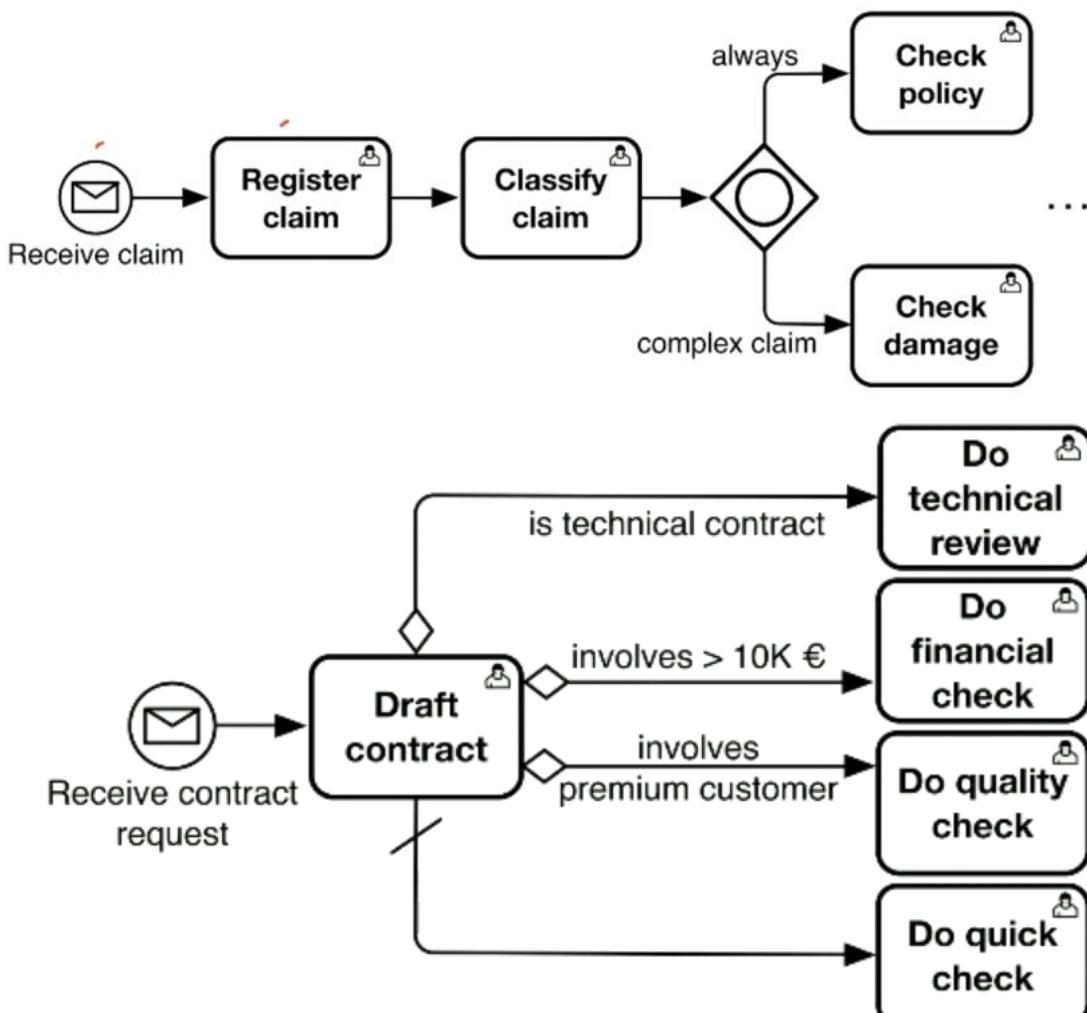
After a contract is drafted, it is subject to different checks. If it is a technical contract, a technical review is done. If it involves a financial transaction of more than 10K Euros, then a financial check is carried out. If the contract involves a premium customer, then a quality check is executed. If none of these special conditions applies, then a quick check suffices.

### Questions

- Can we model the example using the constructs seen so far?
- Can we model it in a compact way?
- Is this a decision? Or an AND-split?

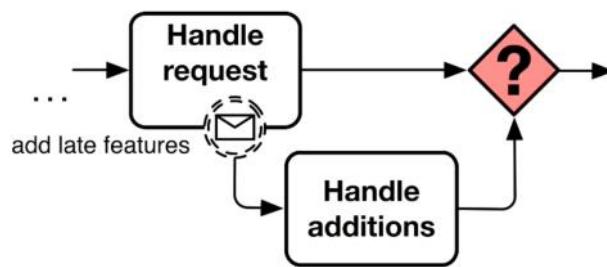
Potrei avere la necessità di attivare un set di vie, come faccio?

Si utilizza un gateway inclusivo, con una 'O' all'interno. Vengono eseguiti tutti i rami veri.



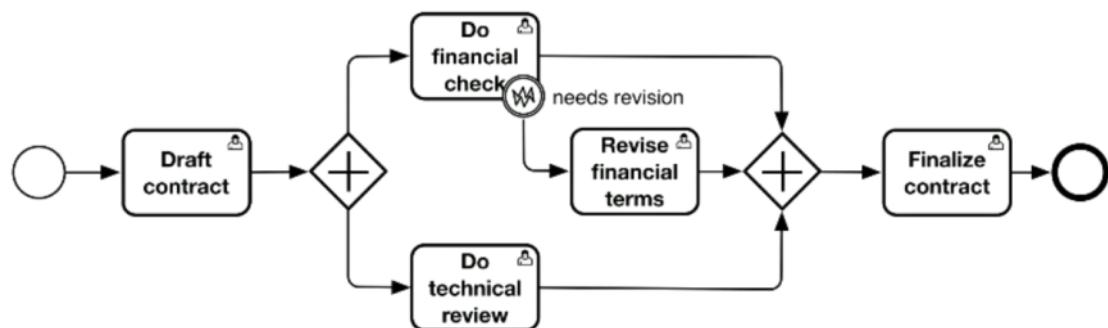
In questo caso c'è l'alternativa di default.

Nel caso dell'OR join, vengono attesi tutti i token generati dallo split.



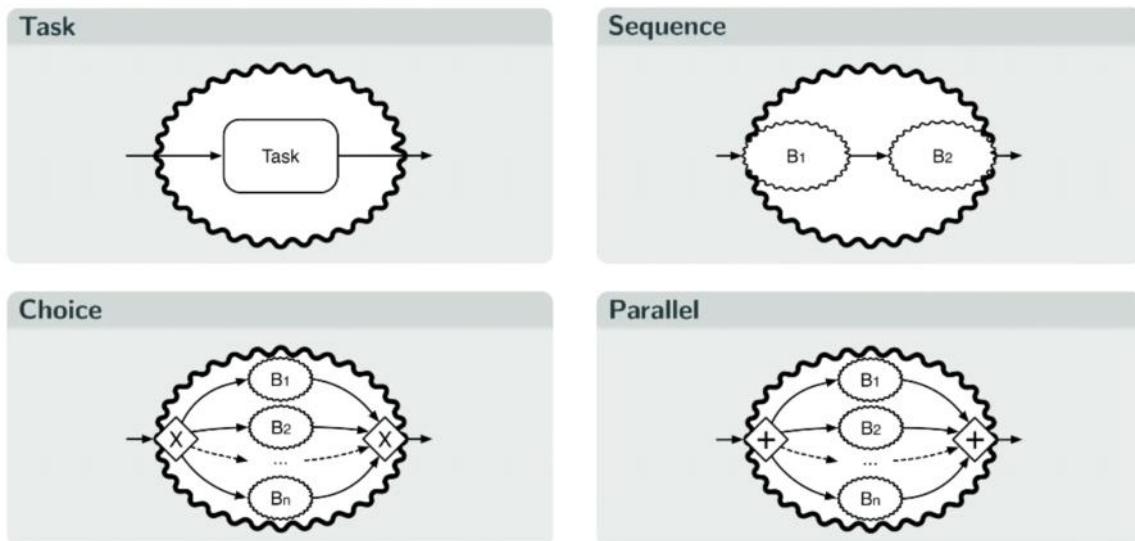
La porta di JOIN sa quanti rami deve aspettare.

Anche qui deve essere utilizzato un OR:

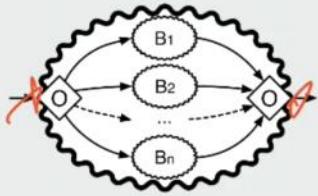


Dipende molto dal fatto se abbiamo bisogno di gestire flussi più globali.

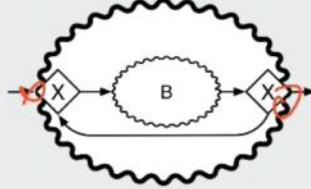
Bisogna utilizzare dei blocchi SESE (single entry single exit) strutturati: ad un certo livello di annidamento si parte con uno split e si finisce con un join.



### Inclusive



### Loop



### End-to-end process



An Inclusive Gateway is activated if:

- **At least one** incoming Sequence Flow has *at least one token* **and**
- **For every** directed path formed by sequence flow that
  - starts with a Sequence Flow  $f$  of the diagram that has a token,
  - ends with an incoming Sequence Flow of the inclusive gateway that has *no token*,
  - does not visit the Inclusive Gateway

**then there is also** a directed path formed by Sequence Flow that

- starts with  $f$ ,
- ends with an incoming Sequence Flow of the inclusive gateway that has a token, and
- does not visit the Inclusive Gateway.

Meglio evitare l'OR join se possibile.

# Discriminatore - 1 Dic

sabato 23 gennaio 2021 19:29

## Example

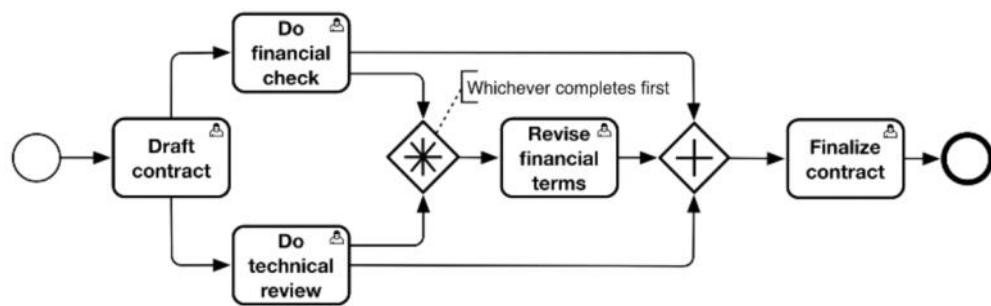
Once a contract is drafted, a financial and technical reviews are conducted in parallel. As soon as one of the two ends, an executive review must be conducted as well. When all reviews are finished, the contract is finalized.

## Question

How to enable an activity as soon as the first among two other activities completes?

- A simple merge does not work, because it would be triggered multiple times!

Lascia passare il primo flow che riceve e blocca tutti quelli che arrivano dopo.



# Reti di Petri - 2 Dic

domenica 24 gennaio 2021 20:33

Sono il fondamento della semantica di BPMN.

L'obiettivo dell'analisi formale è di derivare proprietà e fare analisi (per trovare problemi o fare osservazioni)

Originariamente una descrizione matematica dei processi chimici.

È un grafo bipartito orientato.

## Petri net

A Petri net is a tuple  $(P, T, F, W)$ , where:

- $P$  is a finite set of places;
- $T$  is a finite set of transitions, with  $P \cap T = \emptyset$ ;
- $F \subseteq (P \times T) \cup (T \times P)$  is a set of arcs forming a flow relation;
- $W : F \rightarrow \mathbb{N} \setminus \{0\}$  is an (arc) weight function.

- Graphical notation: places = , transitions =  /  , arcs = .

- Arc types:



## Multi-set

Given a set  $S$ ,  $\mathbb{B}(S)$  is the set of multi-sets over  $S$ .

$X : S \rightarrow \mathbb{N}_{\geq 1}$ ,  $X \in \mathbb{B}(S)$ , is a multi-set where, for each  $a \in S$ ,  $X(a)$  denotes the number of times  $a$  is included in  $X$ .

Un insieme di elementi dove viene annotato per ogni elemento la sua molteplicità.

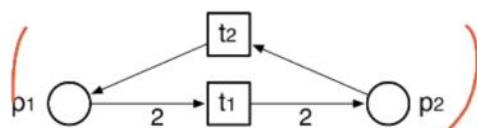
$$(a^{X(a)}): [a, a, a, b, c, b] = [a^3, b^2, c].$$

Sono anche rappresentabili con dei numeri in ordine in corrispondenza della lettera, usando un alfabeto  $[a,b,c,d] \Rightarrow [3,2,1,0]$

## Preset/postset

Given a Petri net  $(P, T, F, W)$  and  $a \in P \cup T$ :

- $\bullet a = [x^{W(x,a)} \mid W(x,a) \text{ is defined and } (x,a) \in F]$ ;
- $a \bullet = [y^{W(a,y)} \mid W(a,y) \text{ is defined and } (a,y) \in F]$ .



$$\bullet p_1 = [t_2]$$

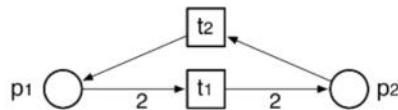
$$p_1 \bullet = [t_1^2]$$

$$\bullet t_2 = [p_2]$$

$$t_2 \bullet = [p_1]$$

Il preset/postset è definito su una rete di petri.

The Petri net  $(P, T, F, W)$  is



Petri net:

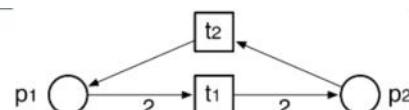
- $P = \{p_1, p_2\}$
- $T = \{t_1, t_2\}$
- $F = \{(p_1, t_1), (t_1, p_2), (\textcolor{brown}{p}_2, \textcolor{brown}{t}_2), (t_2, p_1)\}$
- $W = \{((p_1, t_1), 2), ((t_1, p_2), 2), (\textcolor{brown}{(\textcolor{brown}{p}_2, \textcolor{brown}{t}_2)}, 1), ((t_2, p_1), 1)\}$

Preset:

- $\bullet p_1 = [x^{W(x, p_1)} \mid W(x, p_1) \text{ is defined and } (x, p_1) \in F]$ , that is  
 $\bullet p_1 = [x^{W(x, p_1)} \mid x \in \{t_2\}]$ , thus  $\bullet p_1 = [t_2^1]$ , i.e.  $\bullet p_1 = [t_2]$ , since  $W(t_2, p_1) = 1$
- $\bullet p_2 = [x^{W(x, p_2)} \mid W(x, p_2) \text{ is defined and } (x, p_2) \in F]$ , that is  
 $\bullet p_2 = [x^{W(x, p_2)} \mid x \in \{t_1\}]$ , thus  $\bullet p_2 = [t_1^2]$  since  $W(t_1, p_2) = 2$
- $\bullet t_1 = [x^{W(x, t_1)} \mid W(x, t_1) \text{ is defined and } (x, t_1) \in F]$ , that is  $\bullet t_1 = [x^{W(x, t_1)} \mid x \in \{p_1\}]$ ,  
thus  $\bullet t_1 = [p_1^2]$  since  $W(p_1, t_1) = 2$
- $\bullet t_2 = [x^{W(x, t_2)} \mid W(x, t_2) \text{ is defined and } (x, t_2) \in F]$ , that is  $\bullet t_2 = [x^{W(x, t_2)} \mid x \in \{p_2\}]$ ,  
thus  $\bullet t_2 = [p_2^1]$ , i.e.  $\bullet t_2 = [p_2]$ , since  $W(p_2, t_2) = 1$

Postset

The Petri net  $(P, T, F, W)$  is



Petri net:

- $P = \{p_1, p_2\}$
- $T = \{t_1, t_2\}$
- $F = \{(p_1, t_1), (t_1, p_2), (p_2, t_2), (\textcolor{brown}{t}_2, \textcolor{brown}{p}_1)\}$
- $W = \{((p_1, t_1), 2), ((t_1, p_2), 2), ((p_2, t_2), 1), (\textcolor{brown}{(\textcolor{brown}{t}_2, \textcolor{brown}{p}_1)}, 1)\}$

Preset:

- $p_1\bullet = [y^{W(p_1, y)} \mid W(p_1, y) \text{ is defined and } (p_1, y) \in F]$ , that is  
 $p_1\bullet = [y^{W(p_1, y)} \mid y \in \{t_1\}]$ , thus  $p_1\bullet = [t_1^2]$ , since  $W(p_1, t_1) = 2$
- $p_2\bullet = [y^{W(p_2, y)} \mid W(p_2, y) \text{ is defined and } (p_2, y) \in F]$ , that is  
 $p_2\bullet = [y^{W(p_2, y)} \mid y \in \{t_2\}]$ , thus  $p_2\bullet = [t_2^1]$ , i.e.  $p_2\bullet = [t_2]$ , since  $W(p_2, t_2) = 1$
- $t_1\bullet = [y^{W(t_1, y)} \mid W(t_1, y) \text{ is defined and } (t_1, y) \in F]$ , that is  $t_1\bullet = [y^{W(t_1, y)} \mid y \in \{p_2\}]$ ,  
thus  $t_1\bullet = [p_2^2]$  since  $W(t_1, p_2) = 2$
- $t_2\bullet = [y^{W(t_2, y)} \mid W(t_2, y) \text{ is defined and } (t_2, y) \in F]$ , that is  $t_2\bullet = [y^{W(t_2, y)} \mid y \in \{p_1\}]$ ,  
thus  $t_2\bullet = [p_1^1]$ , i.e.  $t_2\bullet = [p_1]$ , since  $W(t_2, p_1) = 1$

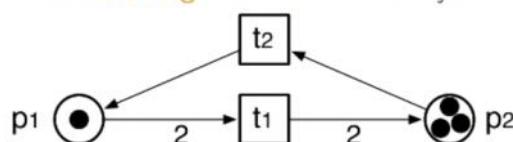
La parte computazionale è effettuata dal marking. Le transizioni rappresentano del lavoro che deve essere compiuto. I token rappresentano istanze parallele dei nostri processi.

We populate a Petri net with **tokens**.

### Marking

A marking  $M$  of a Petri net  $(P, T, F, W)$  is a multi-set over  $P$ :  $M \in \mathbb{B}(P)$ .

The **marking** identifies how many tokens are currently present in each place of the net.



$$M_0 = [p_1^1, p_2^3].$$

Vector notation:  $M_0 = (1, 3)$ .

La regola di firing decide se una certa transizione è attivabile.

Given a marking, the firing rule determines whether a transition can fire (i.e., be executed) and what is the resulting new marking.

### Firing rule

Given a Petri net  $N = (P, T, F, W)$  and a marking  $M \in \mathbb{B}(P)$ :

- a transition  $t \in T$  is enabled, denoted  $(N, M)[t]$ , if and only if  $M \geq \bullet t$ ;
- an enabled transition  $t \in T$  can fire leading to marking  $M' \in \mathbb{B}(P)$ , denoted  $(N, M)[t](N, M')$ , if and only if  $M' = (M - \bullet t) + t\bullet$ .

The notions of sub-multi-set  $\geq$ , multi-set difference  $-$  and multi-set sum  $+$  are defined following the intuition (component by component).

Given a marking, the firing rule determines whether a transition can fire (i.e., be executed) and what is the resulting new marking.

### Firing rule

Given a Petri net  $N = (P, T, F, W)$  and a marking  $M \in \mathbb{B}(P)$ :

- a transition  $t \in T$  is enabled, denoted  $(N, M)[t]$ , if and only if  $M \geq \bullet t$ ;
- an enabled transition  $t \in T$  can fire leading to marking  $M' \in \mathbb{B}(P)$ , denoted  $(N, M)[t](N, M')$ , if and only if  $M' = (M - \bullet t) + t\bullet$ .

The notions of sub-multi-set  $\geq$ , multi-set difference  $-$  and multi-set sum  $+$  are defined following the intuition (component by component).

$$M = (1, 2, 3) \geq \bullet t = (0, 1, 2) \quad t^\bullet = (2, 1, 3)$$

places      places

$$M' = (3, 1, 4)$$



We have that:

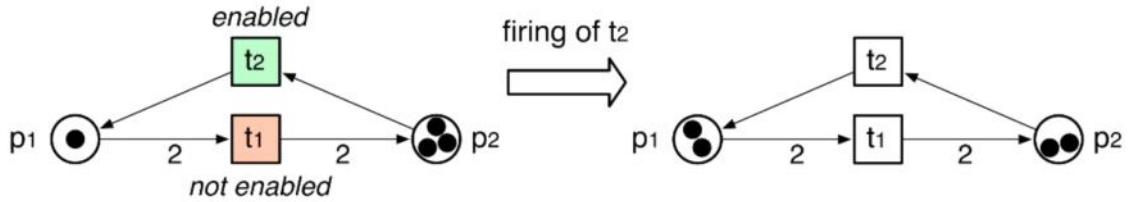
- $\bullet t_1 = [p_1^2] = [p_1^2, p_2^0] = (2, 0)$
- $\bullet t_2 = [p_2] = [p_1^0, p_2^1] = (0, 1)$

Thus:

- $M \not\geq \bullet t_1$ , i.e.  $(1, 3) \not\geq (2, 0)$ ,  $t_1$  not enabled
- $M \geq \bullet t_2$ , i.e.  $(1, 3) \geq (0, 1)$ ,  $t_2$  enabled

The firing of a transition determines an execution step of the net.

- A transition can fire if there are sufficiently many tokens in each of the input places (as required by the arcs' weights).
- The result is obtained by removing the necessary tokens from each input place, and producing the necessary tokens in each output place (as required by the arcs' weights).



We have that  $M_0 = [p_1^1, p_2^3]$ , i.e.  $M_0 = (1, 3)$ :

- $\bullet t_1 = [p_1^2] = [p_1^2, p_2^0] = \cancel{(2, 0)}$
- $\bullet t_2 = [p_2] = [p_1^0, p_2^1] = (0, 1)$
- $t_2\bullet = [p_1] = [p_1^1, p_2^0] = (1, 0)$

Thus:

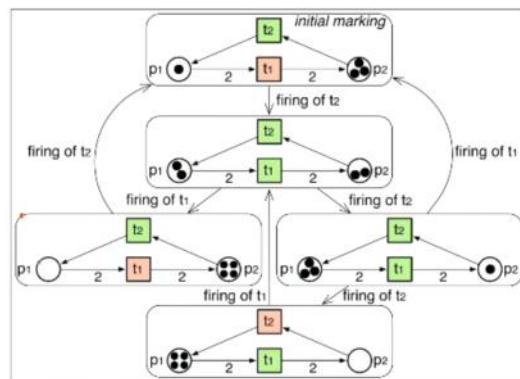
- $M \ngeq \bullet t_1$ , i.e.  $(1, 3) \ngeq (2, 0)$ ,  $t_1$  not enabled
- $M \geq \bullet t_2$ , i.e.  $(1, 3) \geq (0, 1)$ ,  $t_2$  enabled

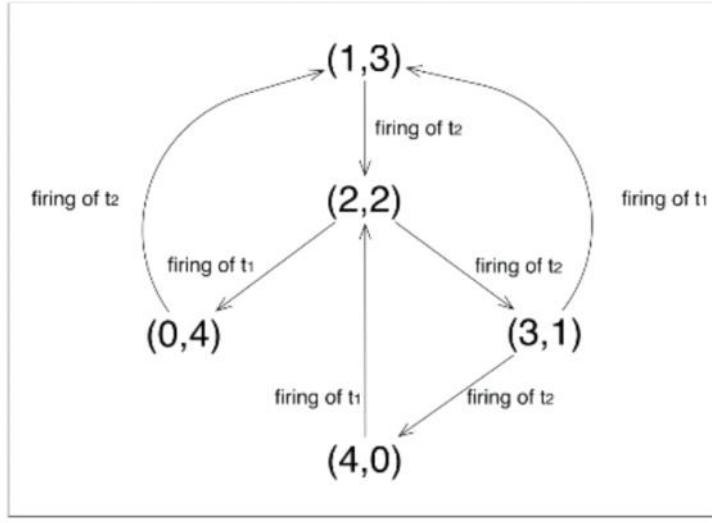
$$M_0 - \bullet t_2 = (1 - 0, 3 - 1) = (1, 2), M_1 = M_0 - \bullet t_2 + t_2\bullet = (1, 2) + (1, 0) = (2, 2)$$

Se per caso ho più possibilità, scelgo non deterministicamente

By iterating for each possible enabled transition in each produced marking, a **transition system** is obtained that represents all the possible executions.

- The transition system is in general *infinite-state*.
- The transition system includes all the *reachable* markings, and is therefore called **reachability graph**.



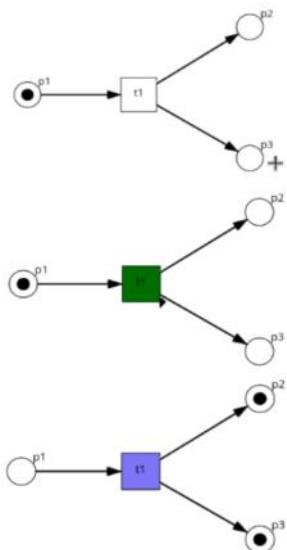


# Petri a BPMN - 2 Dic, 12 Gen

domenica 24 gennaio 2021 21:05

PETRI NET CONCEPT	BP CONCEPT
Place	State
Transition	Atomic activity/event in the activity life-cycle
Token	Object manipulated by a process instance (patient, order, item, ...)
Marking	Snapshot of a process instance
Initial marking	Initial state of a process instance
Enabled transition	Executable activity/event
Firing	Execution step of the process
Reachability graph	Transition system representing all possible executions of the process

## AND Split



We have that  $M_0 = [\dots, p_1^1, p_2^0, p_3^0, \dots]$ .

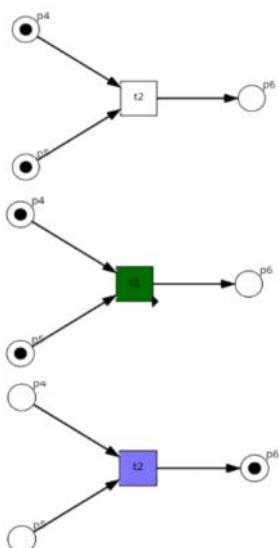
Moreover:

- $\bullet t_1 = [p_1^1] = [\dots, p_1^1, p_2^0, p_3^0, \dots]$  (all zero but  $p_1$ )
- $t_1 \bullet = [p_2^1, p_3^1] = [\dots, p_1^0, p_2^1, p_3^1, \dots]$  (all zero but  $p_2$  and  $p_3$ )

Thus:

- $M_0 \geq \bullet t_1$ ,  
i.e.  $[\dots, p_1^1, p_2^0, p_3^0, \dots] \geq [\dots, p_1^1, p_2^0, p_3^0, \dots]$ ,  
 $t_1$  enabled
- $M_0 - \bullet t_1 + t_1 \bullet = [\dots, p_1^1, p_2^0, p_3^0, \dots] -$   
 $[\dots, p_1^1, p_2^0, p_3^0, \dots] + [\dots, p_1^0, p_2^1, p_3^1, \dots] =$   
 $[\dots, p_1^0, p_2^1, p_3^1, \dots]$

## AND Join



We have that  $M_1 = [\dots, p_4^1, p_5^1, p_6^0, \dots]$ .

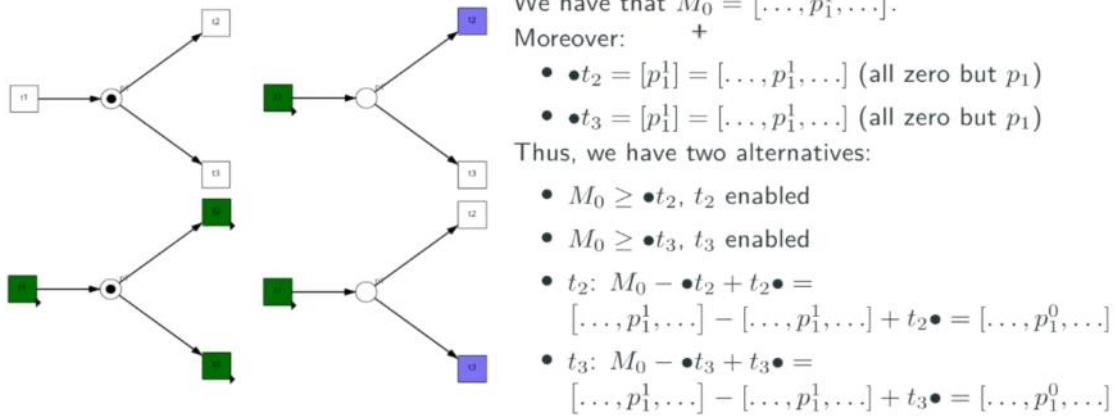
Moreover:

- $\bullet t_2 = [p_4^1, p_5^1] = [\dots, p_4^1, p_5^1, p_6^0, \dots]$  (all zero but  $p_4$  and  $p_5$ )
- $t_2 \bullet = [p_6^1] = [\dots, p_4^0, p_5^0, p_6^1, \dots]$  (all zero but  $p_6$ )

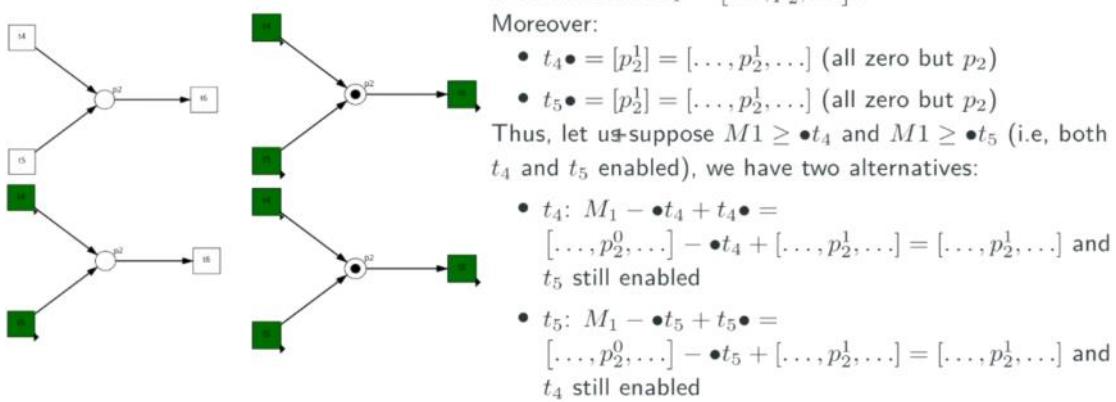
Thus:

- $M_1 \geq \bullet t_2$ ,  
i.e.  $[\dots, p_4^1, p_5^1, p_6^0, \dots] \geq [\dots, p_4^1, p_5^1, p_6^0, \dots]$ ,  
 $t_2$  enabled
- $M_1 - \bullet t_2 + t_2 \bullet = [\dots, p_4^1, p_5^1, p_6^0, \dots] -$   
 $[\dots, p_4^1, p_5^1, p_6^0, \dots] + [\dots, p_4^0, p_5^0, p_6^1, \dots] =$   
 $[\dots, p_4^0, p_5^0, p_6^1, \dots]$

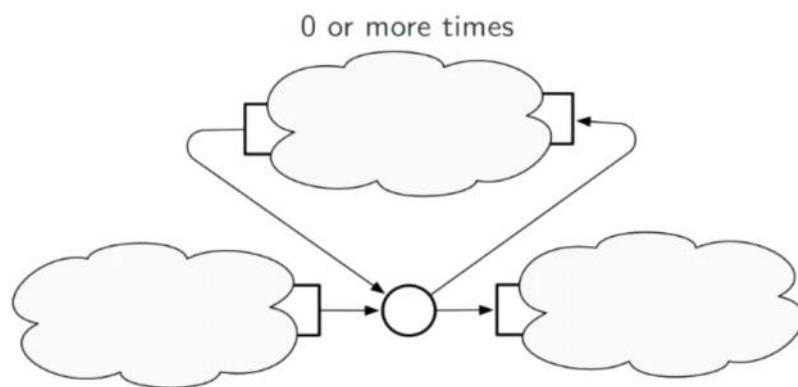
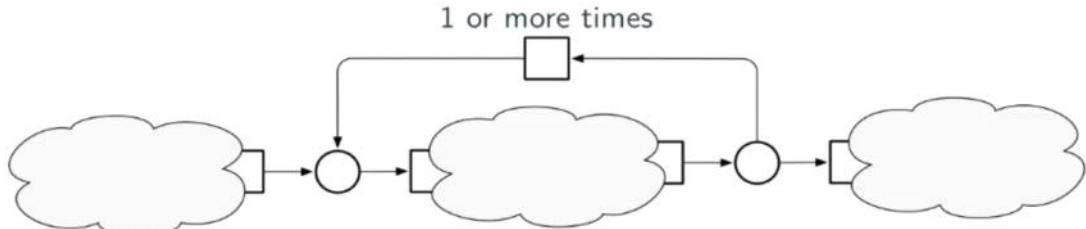
## XOR Split



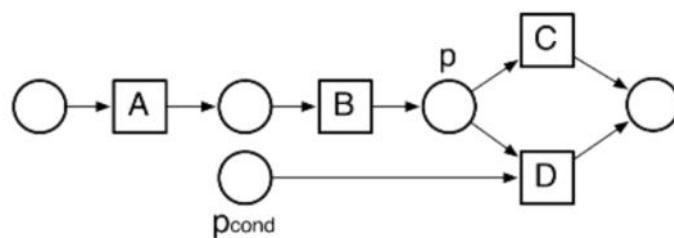
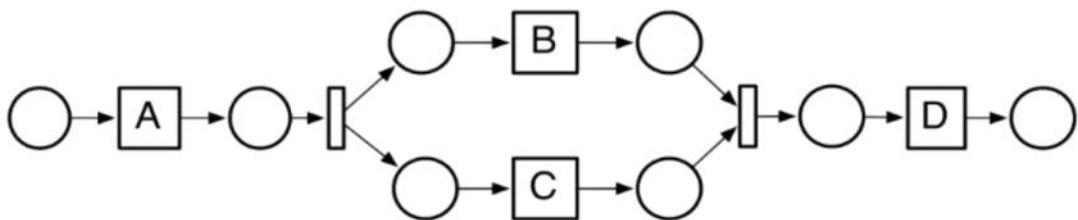
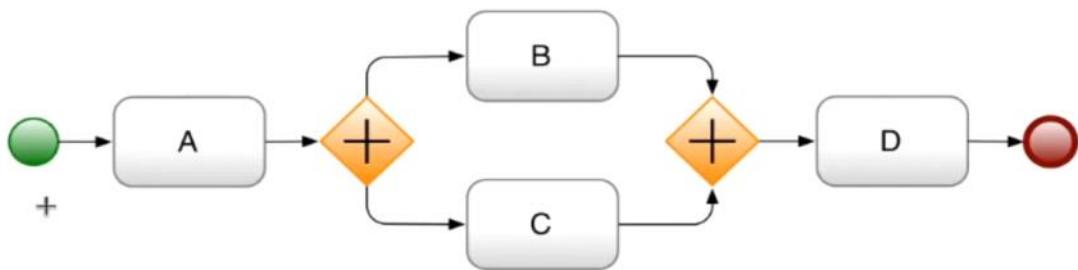
### XOR Join



### Repeat/While



### Esempio di traduzione in petri

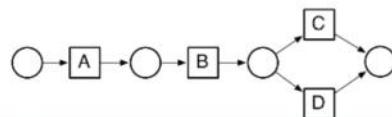


Attenzione che in questo caso la scelta non è libera!

#### Free-choice net

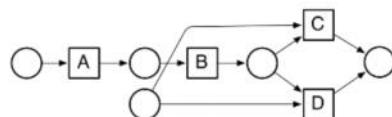
A Petri net  $(P, T, F, W)$  is *free-choice* if, for each  $f = (p, t) \in F$ :

- $|p \bullet| = 1$  ( $f$  is the unique outgoing arc from  $p$ ), or
- $|\bullet t| = 1$  ( $f$  is the unique incoming arc to  $t$ ).



#### (Extended) free-choice net

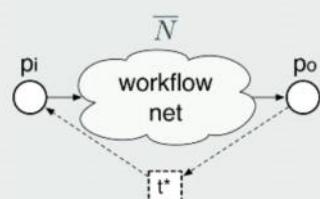
A Petri net  $(P, T, F, W)$  is *(extended) free-choice* if, for each  $p_1, p_2 \in P$ , either  $p_1 \bullet \cap p_2 \bullet = \emptyset$ , or  $p_1 \bullet = p_2 \bullet$ .



#### Workflow net

A Petri net  $N = (P, T, F, W)$  is a *workflow net* if

- There are two special places in  $P$ :
  - an **input place**  $p_i \in P$  such that  $\bullet p_i = \emptyset$ ;
  - an **output place**  $p_o \in P$  such that  $p_o \bullet = \emptyset$ .
- By adding a transition  $t^*$  from  $p_i$  to  $p_o$ , the resulting Petri net  $\bar{N}$  is **strongly connected**: every pair of nodes (transition or places) of  $N$  are connected via a direct path.



Una workflow net è

Given a Petri net  $N$  and an initial marking  $M$ :

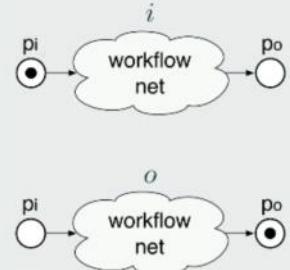
- $(N, M)$  is **terminating** iff there exists  $k \in \mathbb{N}$  such that any firing sequence from  $M$  has a length  $\leq k$ .
- $(N, M)$  is **deadlock-free** iff for every marking  $M'$  reachable from  $M$  there exists an enabled transition in  $M'$ .
- Place  $p$  of  $N$  is **k-bounded** in  $(N, M)$  iff for every marking  $M'$  reachable from  $M$ ,  $M'$  assigns to  $p$  at most  $k$  tokens.
- $(N, M)$  is **k-bounded** iff every place of  $N$  is k-bounded in  $(N, M)$ .
- $(N, M)$  is **safe** iff  $(N, M)$  is 1-bounded.
- Transition  $t$  of  $N$  is **live** in  $(N, M)$  iff for every marking  $M'$  reachable from  $M$ , there exists a marking  $M''$  reachable from  $M'$  such that  $t$  is enabled in  $M''$ .
- $(N, M)$  is **live** iff every transition of  $N$  is live in  $(N, M)$ .

Due marking interessanti

#### Input/output state

Given a workflow net  $N$ :

- The **input state**  $i$  is a marking that assigns only one token to the input place  $p_i$  of  $N$ .
- The **output state**  $o$  is a marking that assigns only one token to the output place  $p_o$  of  $N$ .



#### Soundness

A workflow net  $N$  is **sound** if and only if:

1.  $(\bar{N}, i)$  is **deadlock-free**: starting from the initial marking the only situation in which no transition is enabled is only  $o$ .
2. Starting from the input state  $i$ , the output state is **always reachable**: for every marking  $M$  reachable from  $i$ , there exists a firing sequence leading to  $o$ .
3. The output place  $p_o$  is marked only in a **clean way** by  $o$ : whenever a token is put in place  $p_o$ , all the other places are empty.

#### Theorem (van der Aalst, 1997)

A workflow net  $N$  is sound if and only if  $\bar{N}$  is live and bounded.

#### Theorem (van der Aalst, 1997)

For a free-choice workflow net it is possible to decide soundness in polynomial time.

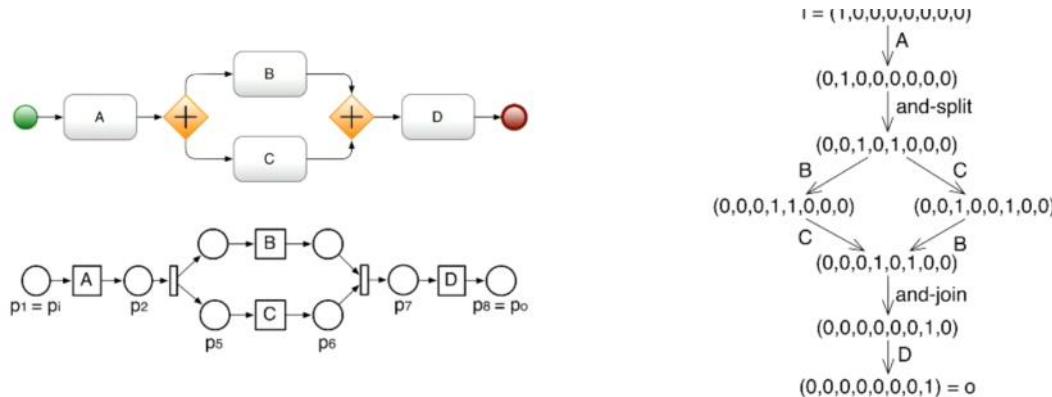
(clean way => c'è soltanto un token lì e nessuno in nessun altro place)

Costruzione del Grafo di Raggiungibilità

## Construction algorithm

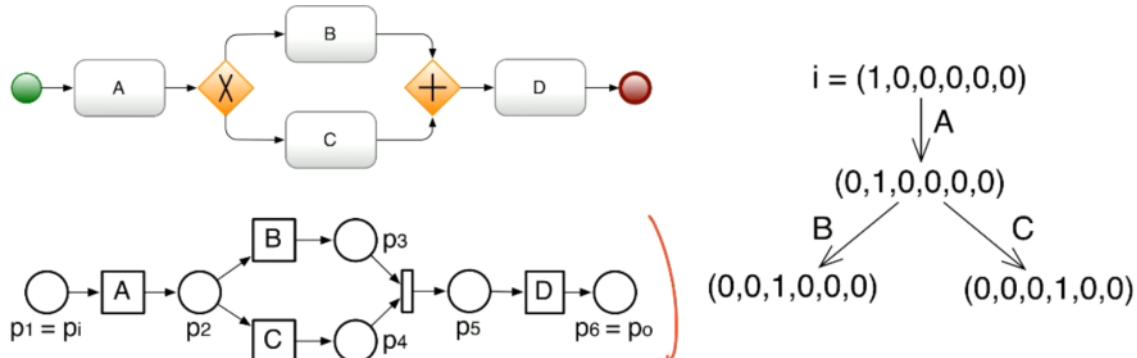
Given a Petri net  $N$  and an initial marking  $M_0$ :

1. Label  $M_0$  as the *root* and initialize set  $New = \{M_0\}$ .
2. While  $New \neq \emptyset$ :
  - 2.1 Select marking  $M$  from  $New$ .
  - 2.2 While there exists an enabled transition  $t$  at  $M$ :
    - 2.2.1 Obtain the marking  $M'$  that results from firing  $t$  at  $M$ .
    - 2.2.2 If  $M'$  does not appear in the graph add it to the graph and insert  $M'$  into set  $New$ .
    - 2.2.3 Draw an arc with label  $t$  between  $M$  and  $M'$ .
  - 2.3 Remove  $M$  from  $New$ .



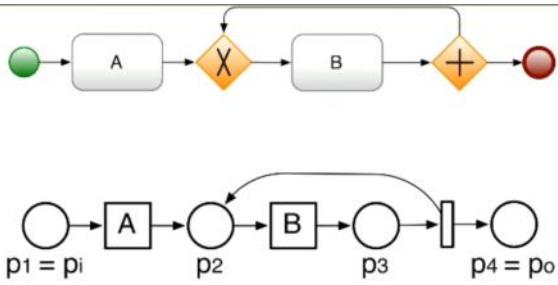
Why? Check reachability graph wrt the three properties for soundness:

1. OK! The only reachable marking without outgoing edges (i.e., no enabled transitions) is  $o$ .
2. OK! Marking  $o$  is reachable from all the other markings.
3. OK! The only reachable marking that puts a "1" in the last position (i.e., that puts a token into  $p_o$ ) is  $o$ .



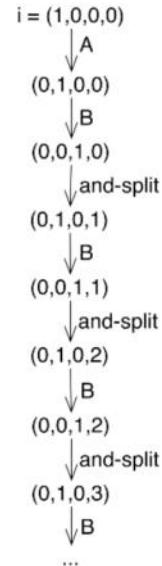
Why?

1. **NO!** There are two reachable markings different than  $o$  for which there is no enabled transition.
2. **NO!** Marking  $o$  is not reachable.
3. OK! No reachable marking exists that puts a token in  $p_o$  and at the same time tokens in other places.



Why?

1. **OK!** All reachable markings have at least one transition enabled (in fact, exactly one).
2. **NO!** Marking  $o$  is not reachable.
3. **NO!** There are reachable markings that associate a token to  $p_o$  and at the same time tokens to other places, such as  $(0, 1, 0, 1)$  and  $(0, 1, 0, 2)$ .

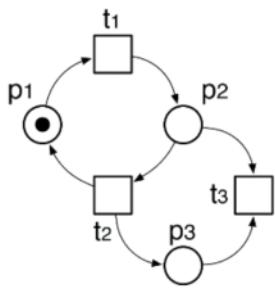


The previous example shows that we cannot always construct the reachability graph. The problem arises when the marked net is **unbounded**.

### Question

How to decide boundedness?

Consider the following example:



Fire  $t_1$  and then  $t_2$ . What happens?

- We obtain a marking that "includes" the starting one.
- The behavior of a Petri net is **monotonic**: if a transition is enabled in a marking  $M$ , it will be enabled in all those markings that include  $M$ .
- We can imagine to "accelerate" the net, by continuing to execute  $t_1$  and  $t_2$ .
- The result is that we continue to end up in the same situation, apart from  $p_3$ , which continues to accumulate new tokens  $\rightsquigarrow$  put  $\omega$  instead for the actual number.

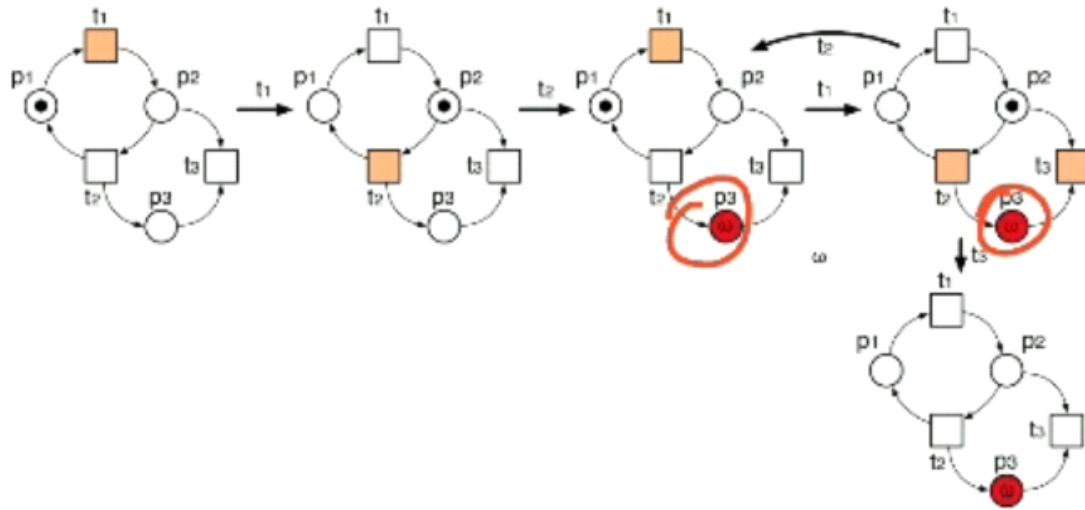
Omega è utile per indicare infinito.

$\omega$  denotes that a place is **unbounded**. Mathematically:

- Now a marking assigns to each place an element from  $\mathbb{N} \cup \{\omega\}$ .
- We extend the multiset operators accordingly:
  - $\omega \geq \omega$ , and  $\omega > n$  for every  $n \in \mathbb{N}$ .
  - An unbounded place will be unbounded forever:  $\omega + n = \omega$ ,  $\omega - n = \omega$ .

Through "acceleration", we construct a **finite abstraction** of the reachability graph that exploits  $\omega$  markings to denote unbounded places.

- Infinite parts of the reachability graph are finitely summarized.



A differenza dell'albero di raggiungibilità ora possiamo costruire un albero di copertura:

#### Construction algorithm

Given a Petri net  $N$  and an initial marking  $M_0$ :

1. Label  $M_0$  as the *root* and initialize set  $New = \{M_0\}$ .
2. While  $New \neq \emptyset$ :
  - 2.1 Select marking  $M$  from  $New$ .
  - 2.2 While there exists an enabled transition  $t$  at  $M$ :
    - 2.2.1 Obtain the marking  $M'$  that results from firing  $t$  at  $M$ .
    - 2.2.2 For every marking  $M'' \neq M'$  on a path from  $M_0$  to  $M'$ : if  $M'' \leq M'$ , then for every place  $p$  s.t.  $M'(p) > M''(p)$ , set  $M'(P) = \omega$ .
    - 2.2.3 If  $M'$  does not appear in the graph add it to the graph and insert  $M'$  into set  $New$ .
    - 2.2.4 Draw an arc with label  $t$  between  $M$  and  $M'$ .
  - 2.3 Remove  $M$  from  $New$ .

L'algoritmo termina sempre!!

Gli alberi non sono sempre intercambiabili, eccetto quando il grafo è bounded.

#### Does the coverability graph faithfully represent the reachability graph?

**NO!** When we have a marking that assigns  $\omega$  to place  $P$ , then, for any number  $n \in \mathbb{N}$ , we now that it will be possible to reach a state in which  $P$  contains at least  $n$  tokens.

Observations:

- When  $\omega$  markings are present, the coverability graph cannot be used to answer reachability queries, but only coverability queries.
- Different Petri nets could have the same coverability graph due to the abstraction.
- The same Petri net could have different coverability graphs due to non-determinism.
- Boundedness is correctly decided by checking whether the coverability graph contains  $\omega$  markings or not.
- Every run of the Petri net can be executed over the coverability graph, but not the other way around.
- Hence, liveness cannot be correctly decided by checking the coverability graph.
- A transition is dead if and only if it does not appear in the coverability graph.
- When the marked net is bounded, then the coverability and the reachability graphs coincide.

Procedura completa per verificare la soundness

Given a workflow net  $N$  (with input state  $i$ )...

1. Construct the coverability graph for  $(\bar{N}, i)$ .
2. Use the coverability graph to check whether  $(\bar{N}, i)$  (and, in turn,  $(N, i)$ ) is bounded.
3. If not  $\rightsquigarrow$  return *NO*.
4. If so (the coverability graph and the reachability graph coincide):
  - 4.1 Check whether  $(\bar{N}, i)$  is live.
  - 4.2 If so  $\rightsquigarrow$  return *YES*.
  - 4.3 If not  $\rightsquigarrow$  return *NO*.

Per verificare la boundness verifico le omega. Se ne trovo, non è bound.

Se non trovo degli omega, i due grafi coincidono.

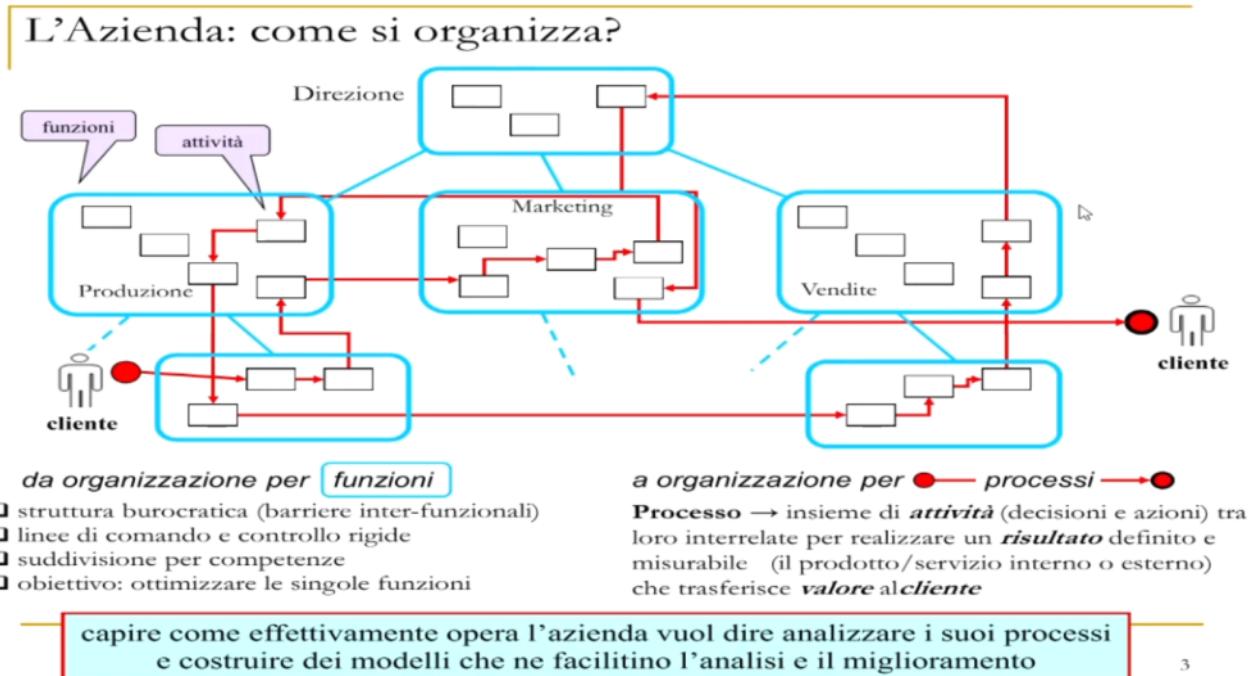
Per verificare se è live, devo vedere se raggiungo sempre la terminazione.

- Reachability graph can be infinite → **coverability graph** that uses  $\omega$ -markings to compactly represent the sources of unboundedness.
- **State-explosion** problem: the coverability graph can be **huge**  $\rightsquigarrow$  **exponential space in the size of the original net**.
- Structural analysis is used to check properties without constructing the coverability graph explicitly.
  - Place invariants, traps, ...

# BPM x reingegnerizzazione - 13 Gen

lunedì 25 gennaio 2021 21:47

Una delle principali osservazioni che abbiamo effettuato con BPM è di osservare i processi. L'azienda è divisa in vari settori, con ognuno che ha delle funzioni e svolge delle attività.



## Definizione Processo Aziendale:

Un processo è un insieme di attività (decisioni e azioni) tra loro interrelate per realizzare un risultato definito e misurabile che trasferisce valore al cliente.

### Un processo

- Risponde ad un evento esterno
- Esegue delle attività
- Raggiunge certi obiettivi
- Deve fornire un risultato
- Consuma delle risorse aziendali
- Deve soddisfare i requisiti dei clienti
- È vincolato da regole interne ed esterne
- Ha un responsabile

Descrivere un processo significa descriverne le varie parti.

Bisogna considerare il tempo, la durata delle attività, il consumo delle risorse. La descrizione dei processi deve facilitare il miglioramento degli stessi.

Usiamo una metodologia per strutturare il modello:

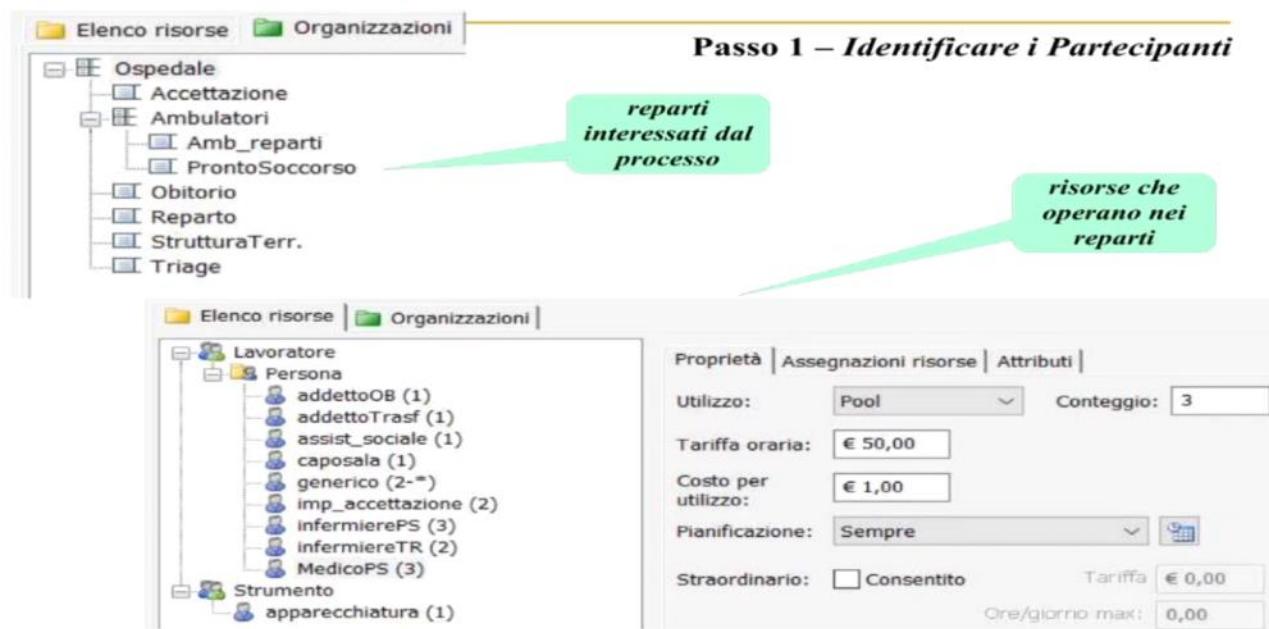
1. Identificare i partecipanti
2. Identificare gli eventi
3. Identificare le attività
4. Specificare la logica di controllo

### Pronto Soccorso (PS) Ospedaliero:

I pazienti arrivano al PS e vengono registrati. Un infermiere effettua quindi il "triage" in cui è valutata la gravità della situazione assegnando al paziente un codice: Bianco (non urgente – paga ticket), Verde (urgenza differibile), Giallo (urgenza), Rosso (emergenza), e si valuta il quadro clinico (se è il caso, si fanno i prelievi al paziente e si effettuano gli esami necessari).

La visita del medico può concludersi con: 1) dimissioni, 2) ricovero, 3) decesso.

Inoltre, alcuni pazienti possono abbandonare il PS dopo il triage (es. non intendono pagare il ticket).



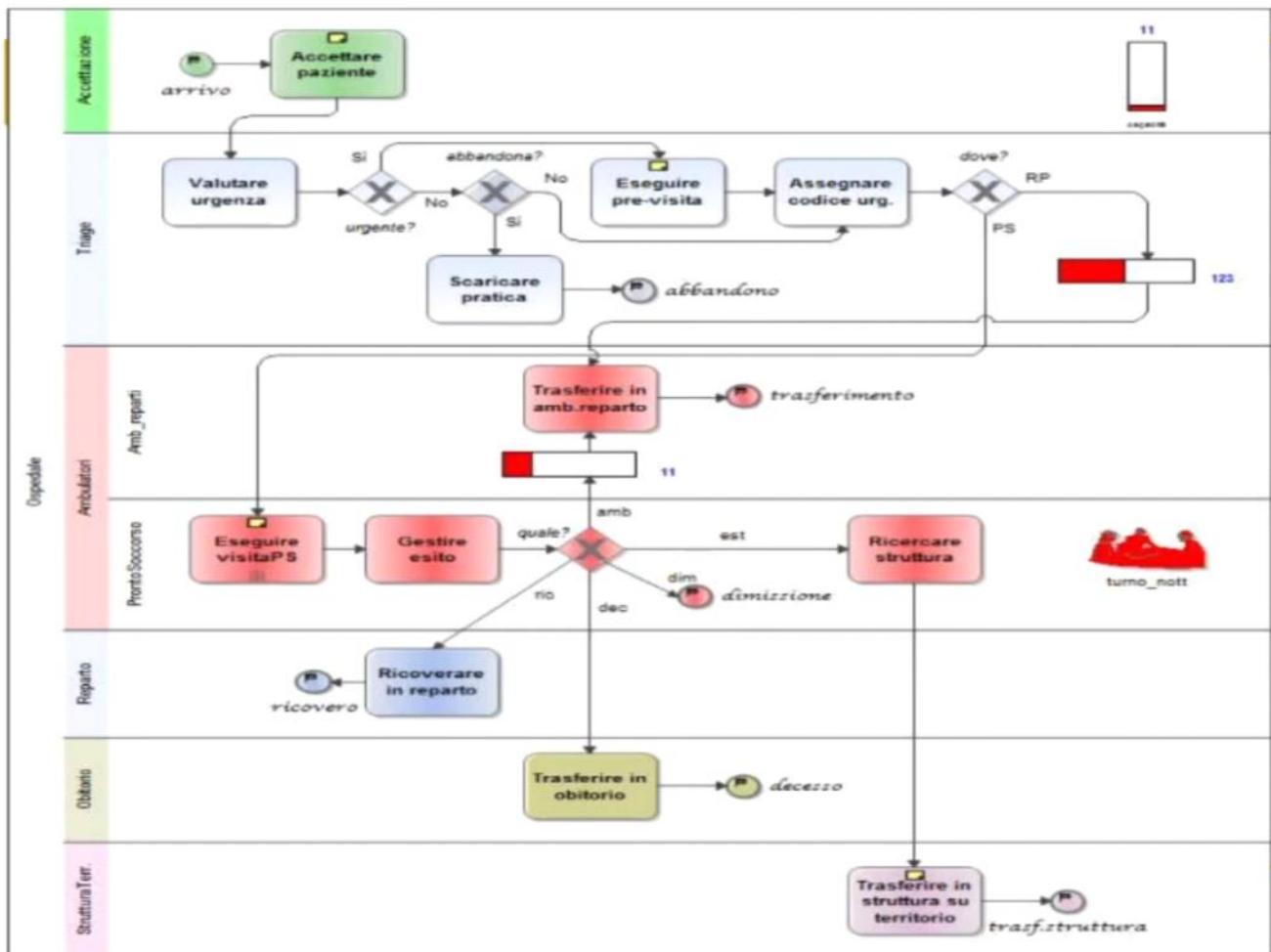
Processo Pronto Soccorso

Passo 4 – Specificare la logica di controllo

reparto	decisioni
Triage	{il paziente abbandona il PS? no(...),si(...)} {il paziente deve andare in un ambulatorio di reparto? no(...),si(...)}
Pronto Soccorso	{qual è lesito della visita? dimesso:dim(...), ricoverato:ric(...), deceduto:dec(...), trasferito:amb(...), esternalizzato:est(...),}
Pronto Soccorso	{il paziente deve fare esami? no(...),si(...)} {il paziente ha bisogno di radiografie? no(...),si(...)}
Pronto Soccorso	{qual è lesito della visita? dimesso:dim(...), ricoverato:ric(...), deceduto:dec(...), trasferito:amb(...), esternalizzato:est(...),}

Il diagramma del processo non è il modello! Mancano le risorse, le caratteristiche delle attività

(temporizzazione, risorse usate, code), delle pianificazioni...



## I generatori

➢ descrivono le transazioni (i “**pazienti**”) in ingresso al processo assegnando delle opportune frequenze di attivazione per gli eventi in ingresso al processo stesso; i pazienti possono essere specificati mediante degli **attributi**:

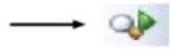
- ✓ **colore** {bianco,verde,giallo,rosso} è assegnato al paziente durante il triage, usando la funzione **Fgrav**
- ✓ **patologia**{ortop,medic,chirur,ambRP} è assegnata al paziente nel triage, con la funzione **Fpatologia**

<b>Funzioni esistenti</b> <input type="checkbox"/> Fgrav <input type="checkbox"/> Furg <input type="checkbox"/> Fesito <input type="checkbox"/> Perc_patologia_esami <input type="checkbox"/> Fpatologia <input type="checkbox"/> Perc_patologia_immagini  <input type="button"/> Aggiungi... <input type="button"/> Modifica... <input type="button"/> Elimina	<b>Tipo funzione</b> tipo_col  <b>Funzioni esistenti</b> <input type="checkbox"/> Fgrav <input type="checkbox"/> Furg <input type="checkbox"/> Fesito <input type="checkbox"/> Perc_patologia_esami <input type="checkbox"/> <b>Fpatologia</b> <input type="checkbox"/> Perc_patologia_immagini <input type="checkbox"/> Fpriorità  <input type="button"/> Aggiungi... <input type="button"/> Modifica... <input type="button"/> Elimina
N. intervalli: <input type="text" value="4"/> Intervallo: Numero: 0 - <input type="text" value="18,84"/> bianco 18,84 - <input type="text" value="92,3"/> verde 92,3 - <input type="text" value="99,59"/> giallo 99,59 - 100 rosso	N. intervalli: <input type="text" value="4"/> Intervallo: Numero: 0 - <input type="text" value="31,47"/> ortop 31,47 - <input type="text" value="53,23"/> medic 53,23 - <input type="text" value="68,88"/> chirur 68,88 - 100 ambRP

## Verificare e Simulare il modello

➤ completato il modello è necessario **verificarlo**

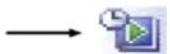
- ✓ la verifica avviene mediante l'**animazione** del processo, una esecuzione "virtuale" in cui alla mappa del processo viene sovrapposta una rappresentazione grafica (un'animazione) che «colora» i nodi della mappa all'avanzare dell'esecuzione.



	Verde: la transazione è in movimento (è "arrivata" sull'attività o la sta "abbandonando" alla sua conclusione)
	Blu: la transazione è in funzione (l'attività sta "lavorando")
	Giallo: la transazione è arrivata ma le risorse non sono disponibili (la transazione si pone in attesa e si crea una coda sull'attività)
	Rosso: la transazione è bloccata (ad esempio si attende la conclusione di un ramo parallelo)
	Grigio: la transazione è sospesa e attende a causa di una pianificazione inattiva (ad esempio le risorse sono fuori orario di lavoro)

➤ per **simulare** il modello è necessario:

- ✓ specificare i parametri della **simulazione** come **tempo di simulazione, tempo di attivazione, traccia** (animazione), ...
- ✓ il simulatore produce una completa serie di statistiche sul processo, con le quali valutare degli **indicatori di performance** sul modello da confrontare con i valori osservabili nella realtà.



Alla fine bisogna anche analizzare i rapporti delle simulazioni!

## Conclusioni

- ✓ **Caso di studio:** è stato analizzato un caso reale, il Pronto Soccorso di una cittadina in Alto Adige
- ✓ **Strumenti:** sono stati utilizzati una metodologia e degli strumenti sviluppati nell'ambito della collaborazione Proxyma - Dipartimento di Informatica dell'Università di Torino
- ✓ **Modello:** è stato sviluppato il modello di funzionamento del Pronto Soccorso, costruendo anzitutto la mappa del processo e introducendo quindi il trattamento delle risorse aziendali e dei pazienti tipizzati in base al loro «colore» e alla patologia evidenziata
- ✓ **Completamento del modello:** sono stati introdotti degli strumenti di monitoraggio del processo per ottenere in fase di simulazione le informazioni necessarie a valutare gli indicatori critici; in genere tali indicatori riguardano:
  - ✓ i tempi di permanenza dei pazienti nel reparto e nelle varie attività,
  - ✓ le code di attesa sulle attività,
  - ✓ i costi di erogazione delle prestazioni che comprendono i costi del personale, i costi legati all'ammortamento delle apparecchiature e quelli per i materiali monouso
- ✓ **Estensioni:** la flessibilità del modello permette di approfondire ulteriormente l'analisi introducendo altre caratteristiche, ad esempio diversificando le attività in base alle esigenze del paziente (ortopedia, chirurgia, internistica e altri tipi di ambulatori).

BPM\*