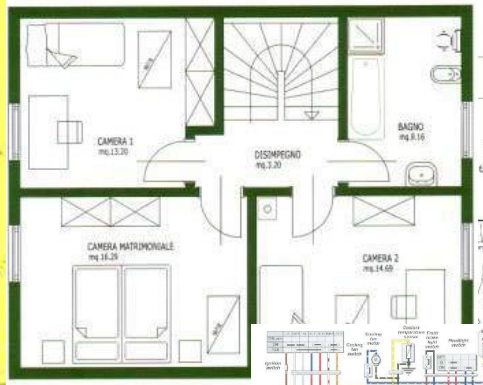
Progettazione di un database

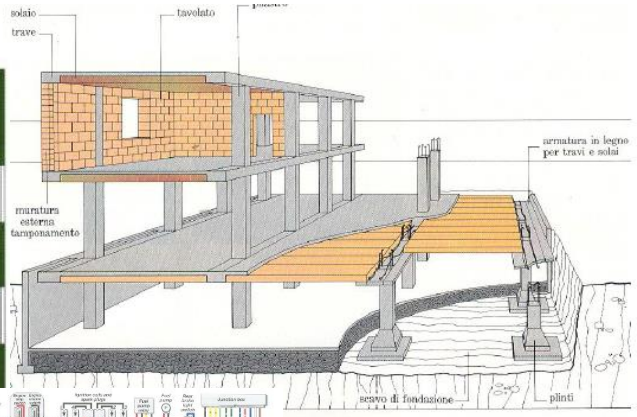
Annalisa Franco, Dario Maio

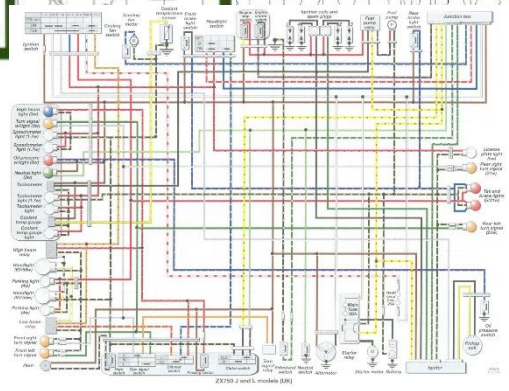
Università di Bologna

Un'analogia

- In ogni settore delle scienze applicate la progettazione è svolta in fasi, ciascuna avente un preciso obiettivo; indica in sostanza l'attività che è alla base della costruzione/realizzazione di qualsiasi manufatto complesso, sia esso materiale o soltanto concettuale attraverso la stesura di un progetto.







In ogni fase si adottano modelli di rappresentazione, si producono schemi di tipo concettuale, logico e fisico.

Scenari per la progettazione DB

Progettazione da Zero di un Sistema Informativo

- La progettazione è svolta nell'ambito della più ampia attività di sviluppo ex novo di un sistema informativo:
  - in questo scenario i task specifici relativi alle basi di dati devono essere coordinati con vari altri aspetti di design del SI, sin dalla fase di analisi, e possono condizionare anche le scelte delle configurazioni architetturali del sistema informatico.

La progettazione del database deve essere coordinata con la progettazione dell'architettura del sistema e delle applicazioni per garantire che tutte le parti funzionino insieme in modo fluido.

Progettazione di un DB esistente o no di un Sistema Informativo già presente

- Si richiede di sviluppare un nuovo DB con le relative applicazioni, o reingegnerizzare un DB esistente eventualmente aggiungendo nuove funzionalità, nel contesto di un sistema informatico già in esercizio:
  - in questo scenario l'attività di progettazione è svolta come un processo a sé stante;
  - non si esclude, tuttavia, che si rendano necessarie modifiche e/o estensioni dell'architettura del sistema informatico.

Tipologia delle applicazioni

Applicazioni orientate agli oggetti

- L'aspetto più significativo è costituito dalle informazioni
- Le funzioni svolte non sono molto complesse.

Applicazioni orientate alle funzioni

- La complessità risiede nel tipo di trasformazione input-output operata.

Applicazioni orientate al controllo

- L'aspetto più significativo da modellare è la sincronizzazione fra diverse attività cooperanti nel sistema.

Un tipico SI di media complessità richiede durante la progettazione il ricorso a più strumenti di modellazione. Il linguaggio di modellazione UML, complementato con metodi di progettazione di DB, risponde a questa esigenza.

Oggetti, funzioni e stati

Oggetti

- **ASPETTI STATICI**
- Possono essere descritti a partire da termini molto generici (es. edificio) fino ad arrivare a livello di dettaglio specifici (es. Palazzo Mazzini Marinelli, via Sacchi, 3, Cesena)

Funzioni

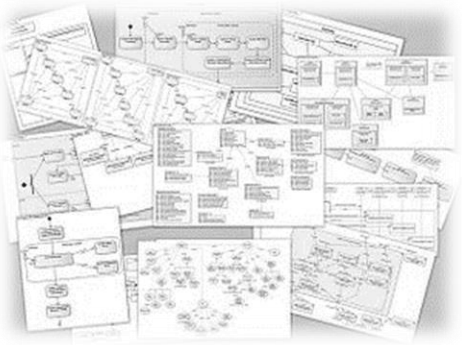
- **ASPETTI FUNZIONALI**
- Possono essere espresse inizialmente in modo vago (es. controllare il livello di gas nocivi nell'aria) e successivamente precisate (es. la programmazione del livello di soglia per l'allarme della centralina è attivata, dopo aver digitato un Pin, premendo il pulsante P)

Stati

- **ASPETTI DINAMICI**
- Possono essere decritti a un elevato livello di astrazione (es. la centralina è in stato di errore) o specificati in maggior dettaglio (es. è acceso il segnalatore d'errore nel sensore S)

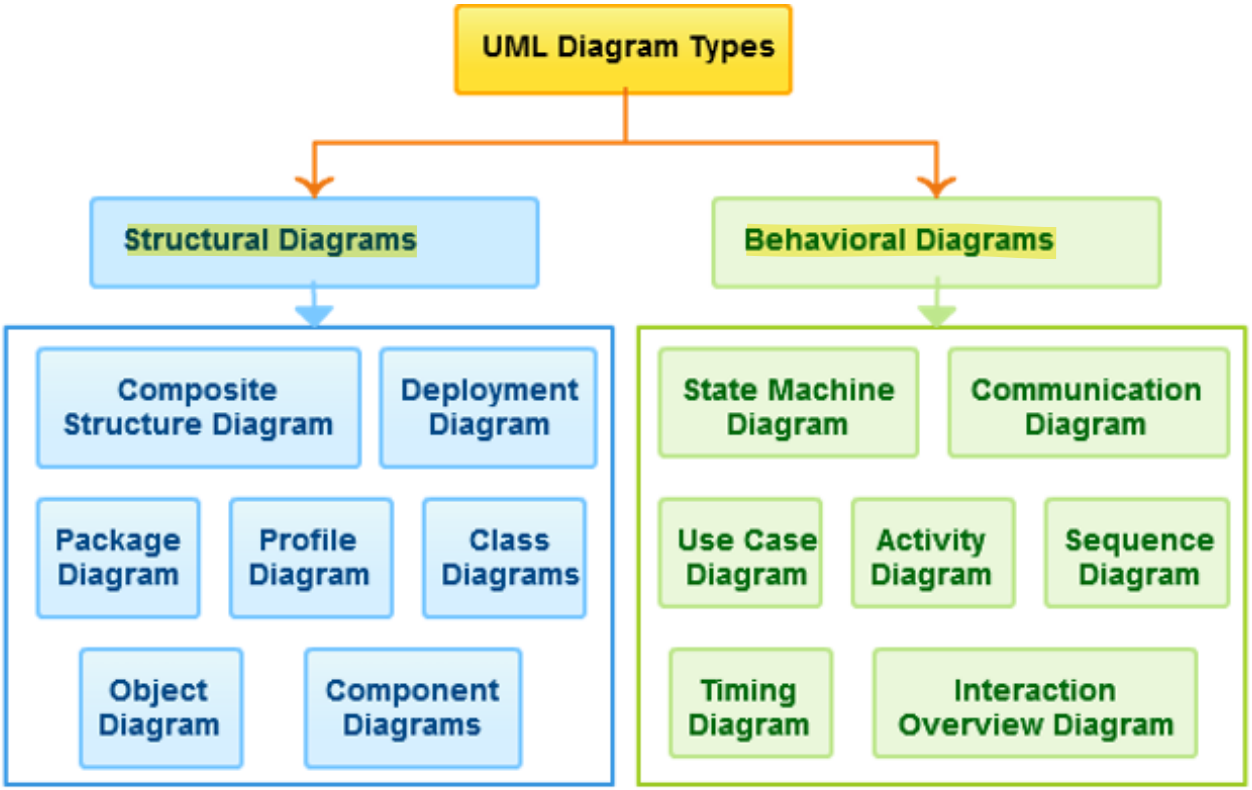
Tassonomia dei metodi di analisi

- L'orientamento di un metodo (analisi orientata agli oggetti, analisi funzionale o analisi orientata agli stati) è determinato sia dalla tipologia di SI sia dall'approccio seguito dal team che pone un diverso accento nella modellazione della realtà verso un aspetto ritenuto predominante.
- La tendenza attuale, negli approcci all'analisi e alla progettazione, consiste nell'integrare modelli di rappresentazione nati per finalità diverse; si veda ad esempio il linguaggio di modellazione UML (**U**nified **M**odeling **L**anguage), che si è affermato ormai come standard de facto.



aspetti statici, funzionali e dinamici

UML: tipi di diagrammi



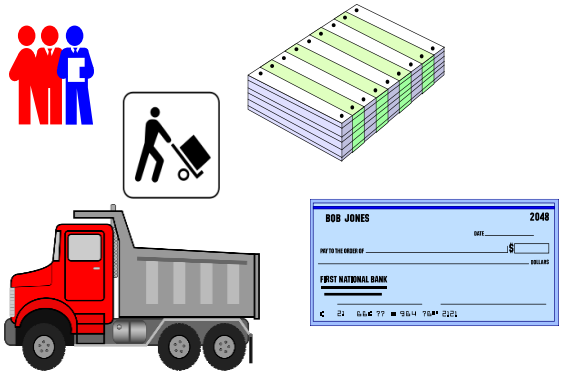
N.B. In UML la modellazione concettuale dei dati sfrutta i diagrammi delle classi nati per rappresentare sia le proprietà strutturali (attributi) degli oggetti sia i metodi che si possono invocare. Tuttavia alcuni costrutti del modello E/R appositamente sviluppato per la progettazione di DB non sono contemplati negli schemi UML.

Analisi orientata agli oggetti

si usa UML

- L'enfasi è posta:
  - sull'identificazione degli oggetti e sulla loro classificazione;
  - sulle interrelazioni tra oggetti.

- Nel tempo le proprietà strutturali degli oggetti osservati restano abbastanza stabili, mentre l'uso che degli oggetti si fa può mutare in modo sensibile.

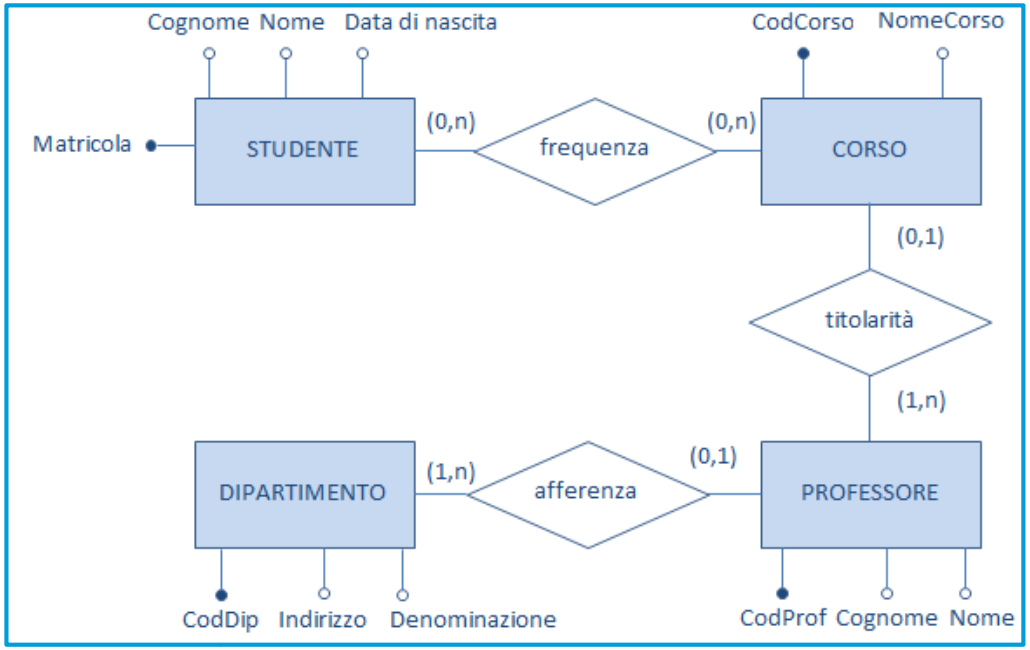


Esempio di modellazione E/R

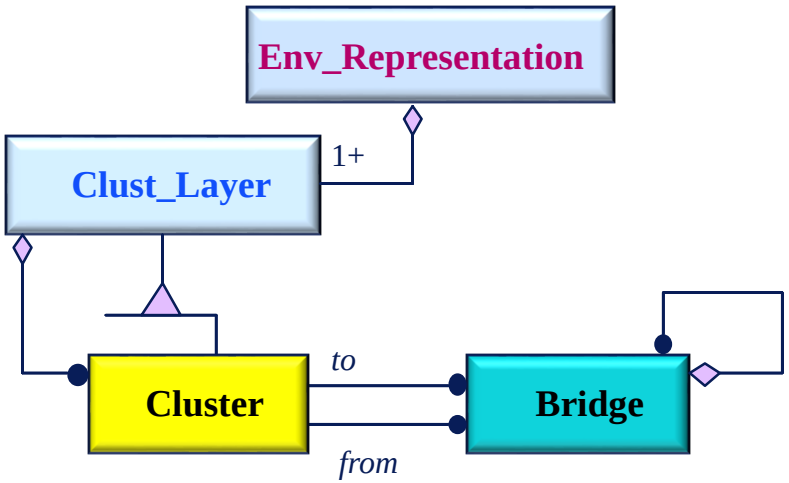
- Si fa inoltre uso di un glossario dei termini (detto anche dizionario dati), con finalità di memorizzare in modo uniforme e non ambiguo i vari tipi di dati riscontrati durante l'analisi e le relazioni che sussistono fra essi nonché le altre risorse coinvolte nel sistema informativo.
- È previsto anche l'impiego successivo o concomitante di altri strumenti di modellazione, ad esempio l'integrazione di schemi E/R con DFD.

Uno schema E/R

N.B. Snapshot di un A.A.



Un esempio di diagramma delle classi



Cluster

descr: string
position: coord
bridges: list(Bridge)
father: Cluster

init(descr, pos, level)
who: string
where: coord
which_level: integer
add_cluster(new_cluster)
add_bridge(new_bridge)
rm_bridge(old_bridge)
is_connected(to): Bridge
update_pos
ancestor(j: integer)

Bridge

from: Cluster
to: Cluster
visited: integer
cost: real
bridges: set(Bridge)

init(from, to, cost)

where_to: Cluster
where_from: Cluster
which_level: integer
how_much: real
add_bridge(new_bridge)
rm_bridge(old_bridge)
update_cost

Un esempio di diagramma delle classi per rappresentare un ambiente costituito da più livelli ciascuno formato da cluster connessi da bridge.

Env_Representation

max_level: integer
layers: list(Clust_layer)

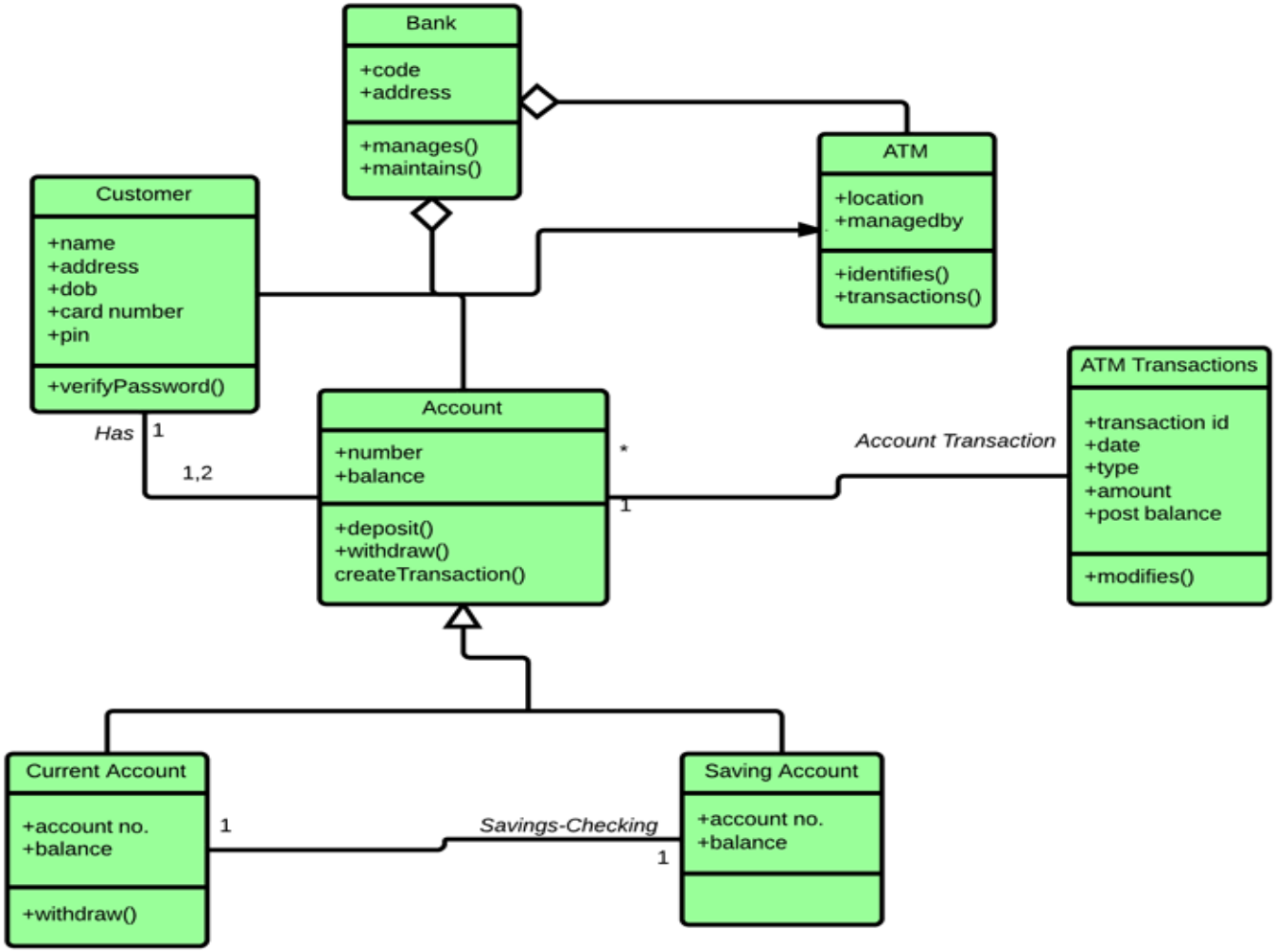
init(max_level): Env_Representation
how_many: integer
clustering(candidate): Cluster
nc_discovery(to_descr, to_pos)
c_discovery(to_descr, to_pos, cost)

Clust_Layer

clusters: set(Cluster)
level: integer

init(level): Clust_Layer
find_cluster(descr): Cluster
add_cluster(new_cluster)
add_bridge(new_bridge)
rm_bridge(old)

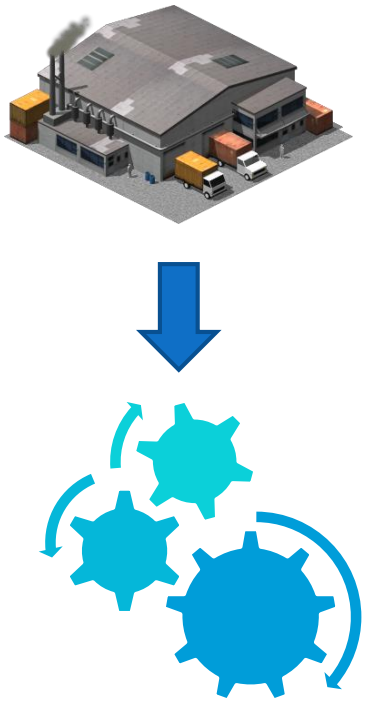
Altro esempio di diagramma delle classi



Fonte: https://www.lucidchart.com/pages/uml/class-diagram

Analisi orientata alle funzioni

- L'obiettivo è rappresentare un sistema come:
  - una rete di processi;
  - un insieme di flussi informativi tra processi.
- Ciò corrisponde alla progressiva costruzione di una gerarchia funzionale.
- Sinonimi di funzione:
  - processo, bolla, attività, trasformazione, transazione.
- Spesso in passato nella modellazione funzionale si è fatto ricorso alla rappresentazione con DFD (**D**ata **F**low **D**iagram).

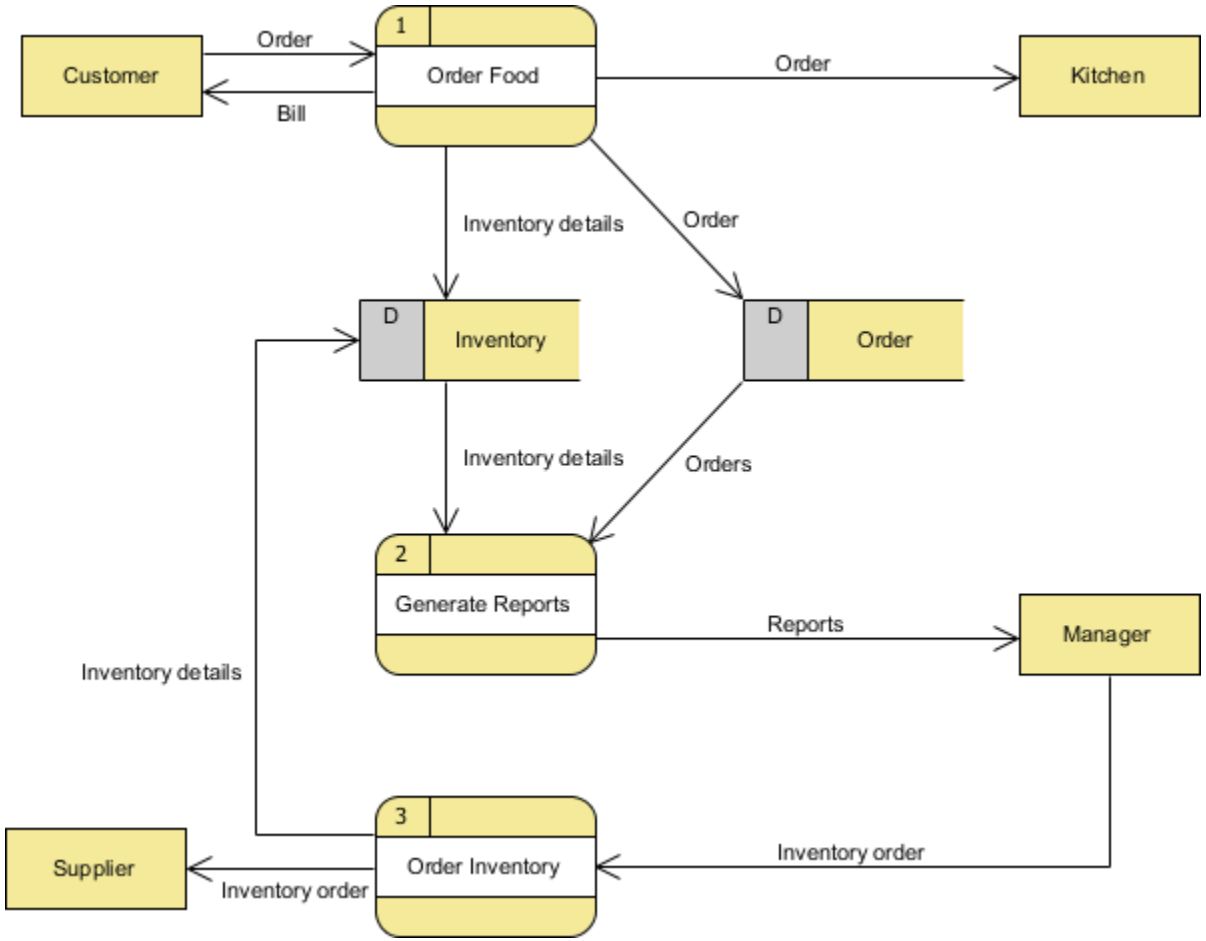


Esempio di DFD (1)

Attore

Funzione

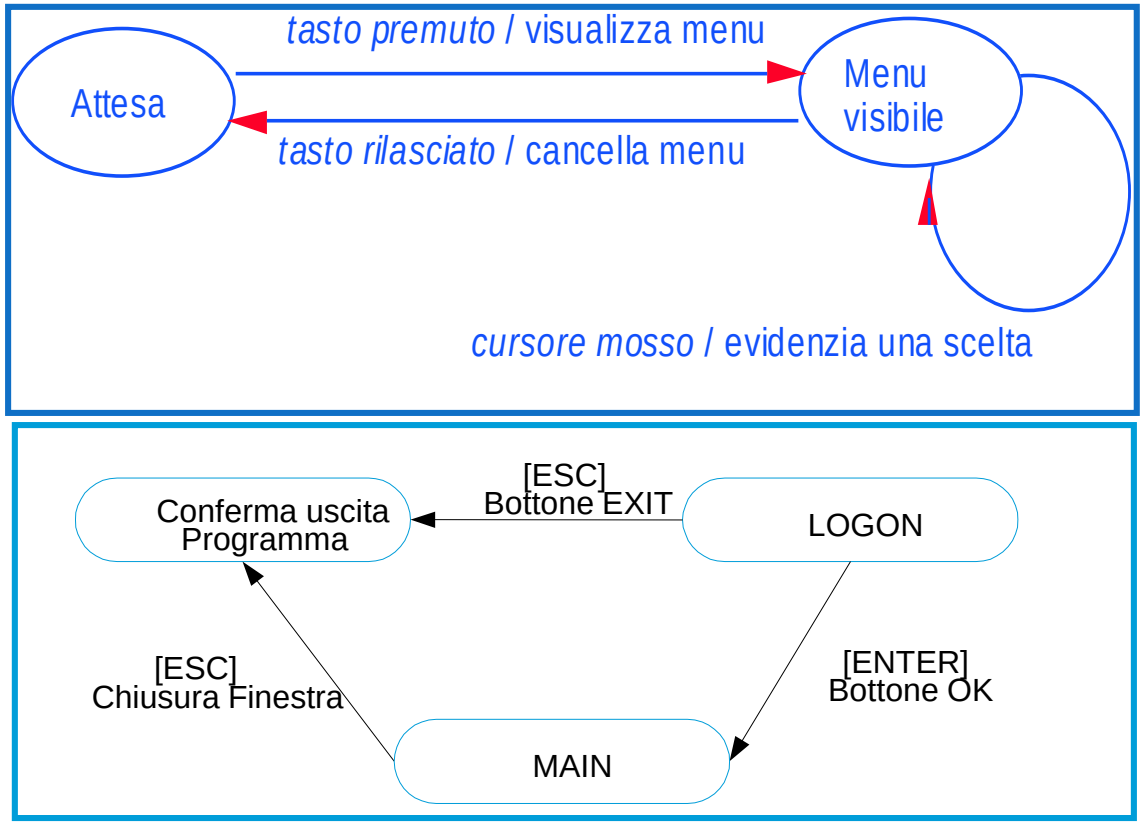
Archivio



Fonte: https://www.visual-paradigm.com/tutorials/

Analisi orientata agli stati

- Per alcune categorie di applicazioni può essere utile pensare fin dall'inizio in termini di stati operativi, in cui si può trovare il sistema allo studio, e transizioni di stato.



Meccanismi di astrazione

Molteplici sono le relazioni in gioco fra oggetti, funzioni e stati e molteplici i livelli di possibile dettaglio

- L'analista deve far ricorso a tecniche che gli consentano di organizzare e interrogare la conoscenza sul problema via via acquisita.
- I principali meccanismi di astrazione usati durante il processo di analisi per costruire una base di conoscenza sul problema sono:
  - classificazione
  - generalizzazione
  - aggregazione
  - proiezione

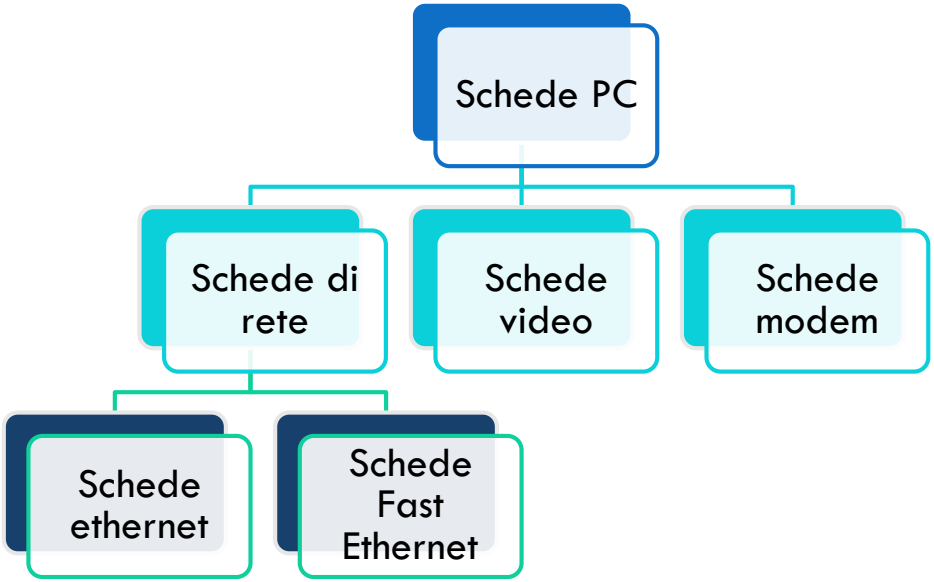
Classificazione

Raggruppa gli oggetti in classi in base alle loro proprietà

Computer

Generalizzazione

Cattura le relazioni di tipo «*è un*» ovvero permette di astrarre le caratteristiche comuni fra più classi definendo superclassi



Ciò che caratterizza una scheda per PC è comune anche a ogni suo sottoinsieme, ovvero ogni sottoclasse eredita dalla superclasse ma può anche avere caratteristiche proprie.

La specializzazione è il processo inverso della generalizzazione.

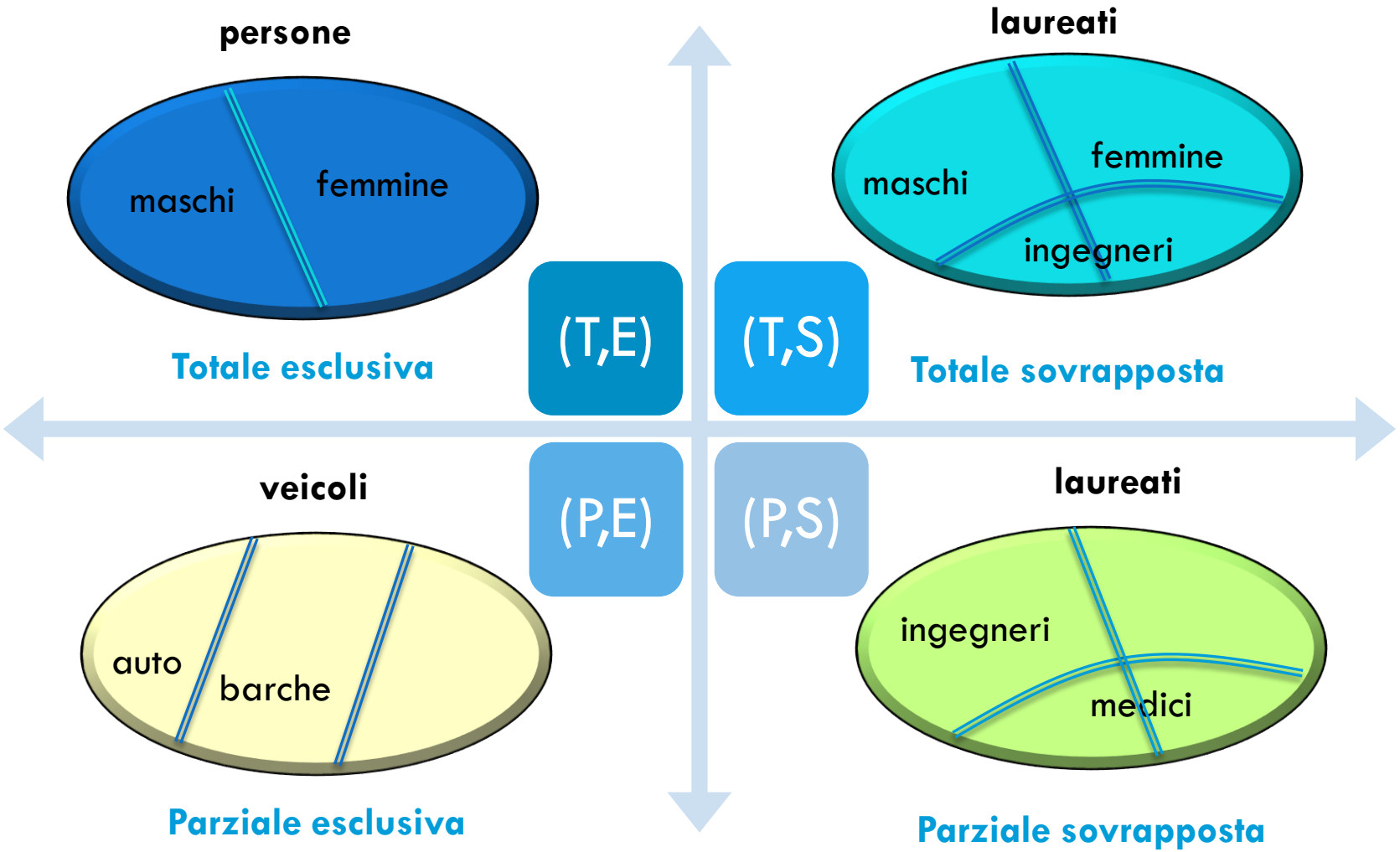
Copertura delle generalizzazioni

Confronto fra unione delle specializzazioni e classe generalizzata

Confronto fra le classi specializzate

TOTALE	PARZIALE	ESCLUSIVA	SOVRAPPOSTA
La classe generalizzata è l'unione delle specializzazioni	La classe generalizzata contiene l'unione delle specializzazioni	Gli insiemi delle specializzazioni sono fra loro disgiunti	Può esistere un'intersezione non vuota fra insiemi delle specializzazioni

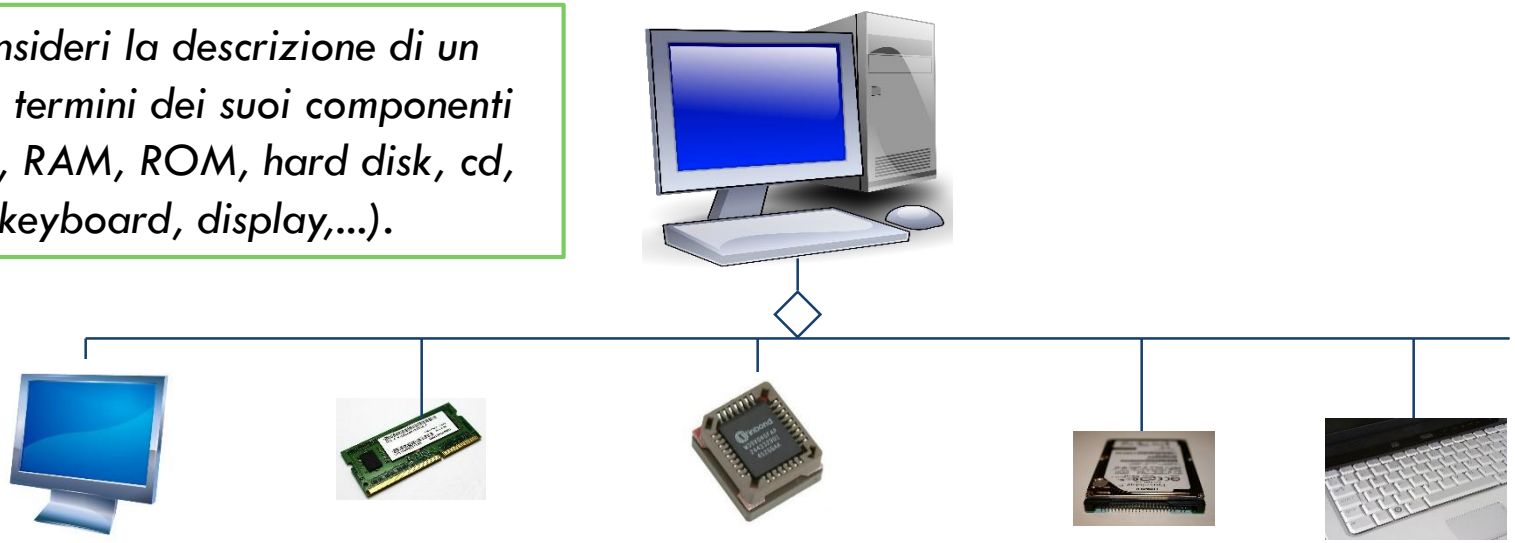
Proprietà di copertura - esempi



Aggregazione

L'aggregazione esprime le relazioni "parte di" che sussistono tra oggetti, tra funzioni, o fra stati

Si consideri la descrizione di un PC in termini dei suoi componenti (CPU, RAM, ROM, hard disk, cd, dvd, keyboard, display,...).



Proiezione

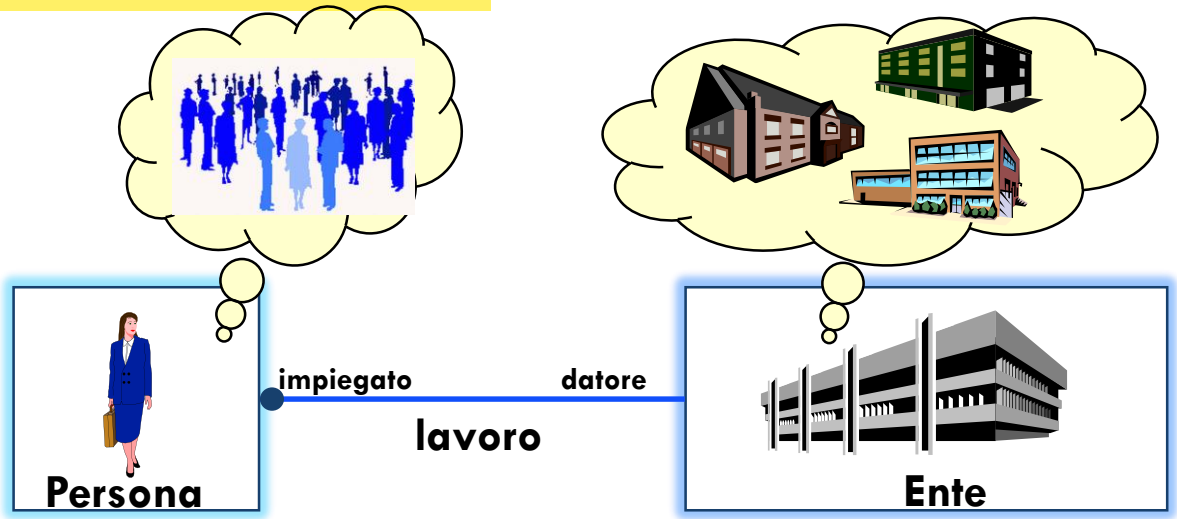
(permette di catturare e rappresentare le diverse prospettive degli utenti e delle parti interessate).

Cattura la vista delle relazioni strutturali fra gli oggetti, le funzioni, gli stati

Ad esempio, nel descrivere il funzionamento di un certo personal computer può essere necessario distinguere il punto di vista dell'operatore, dell'installatore, del programmatore.

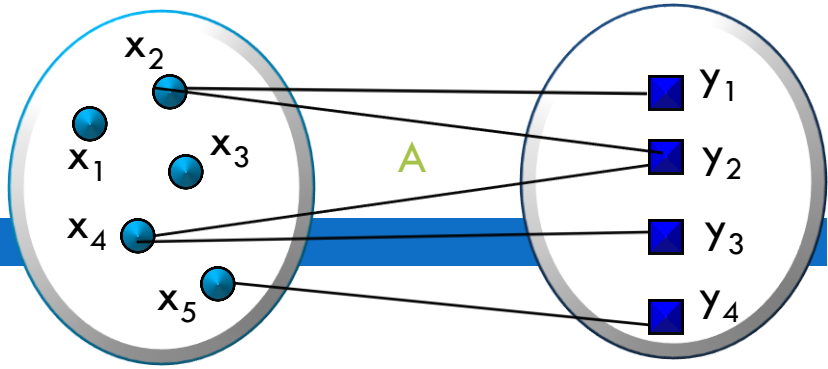
Associazioni

- Oltre ai meccanismi citati è importante modellare le associazioni che sussistono fra le varie classi.



- Le associazioni (corrispondenze tra classi) sono di fatto aggregazioni di cui le classi sono le componenti.
- Dal punto di vista della modellazione, è importante caratterizzare queste corrispondenze in termini di vincoli di cardinalità (un impiegato in quanti enti può lavorare?).

Vincoli di cardinalità



- Sia A un'associazione fra C1 e C2
  - min-card(C1,A): cardinalità minima di C1 in A
    - è il *minimo numero di corrispondenze* nell'associazione A alle quali ogni istanza di C1 deve partecipare;
  - max-card(C1,A): cardinalità massima di C1 in A
    - è il *massimo numero di corrispondenze* nell'associazione A alle quali ogni istanza di C1 può partecipare.

Indica Min/Max numero di associazioni per ogni elemento

- Vincoli di cardinalità minima
  - partecipazione opzionale: min-card(C1,A) =0
    - alcuni elementi di C1 *possono non essere associati* tramite A a elementi di C2;
  - partecipazione obbligatoria (totale): min-card(C1,A) > 0
    - a ogni elemento di C1 *deve essere associato*, tramite A, almeno un elemento di C2.

Vincoli di cardinalità: esempi

- Sia ASSEGNAZIONE un'associazione fra DIPENDENTE e MANSIONE.
- Nel particolare dominio applicativo in esame sono vere le seguenti affermazioni:
  - un dipendente svolge almeno una mansione e al massimo 3;
  - una medesima mansione può essere ancora non assegnata o essere svolta da più dipendenti, al massimo 5.
- Si ha:
  - min-card(DIPENDENTE, ASSEGNAZIONE) = 1
  - max-card(DIPENDENTE, ASSEGNAZIONE) = 3
  - min-card(MANSIONE, ASSEGNAZIONE) = 0
  - max-card(MANSIONE, ASSEGNAZIONE) = 5

Vincoli validi per qualunque stato estensionale.

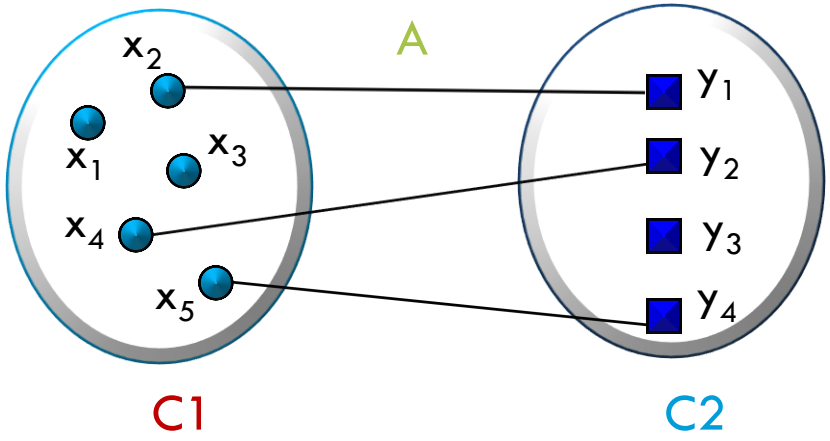
Associazioni binarie *uno a uno*

Ogni Istanza della Classe deve avere MAX una associazione

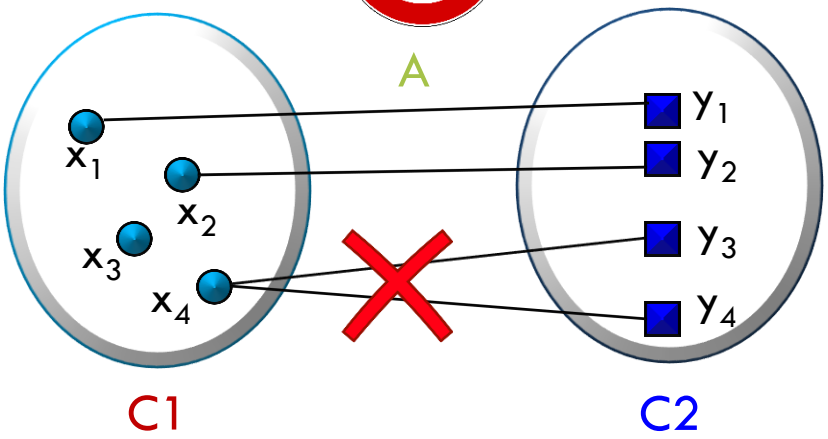
uno a uno (one-to-one)

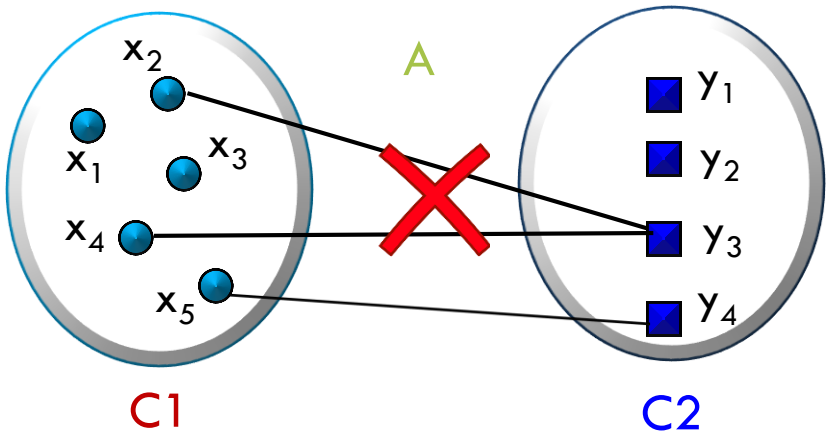
max-card(C1, A) = 1

max-card(C2, A) = 1



Associazioni che non soddisfano i vincoli





Associazioni binarie *uno a molti*

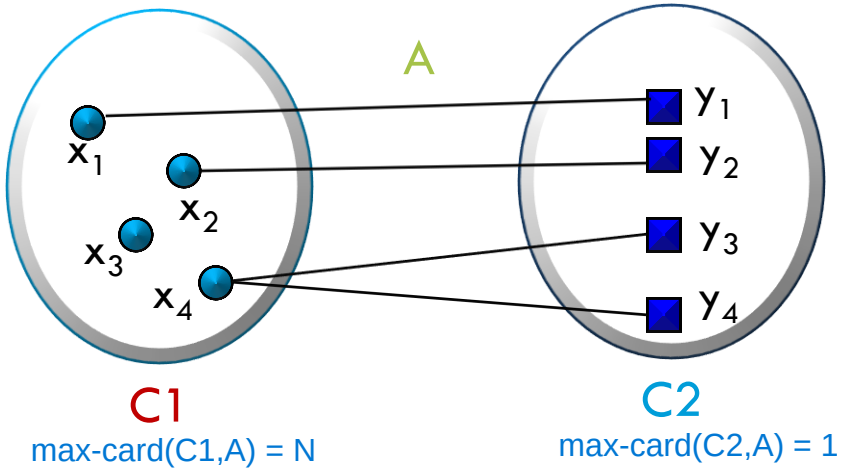
Ogni Istanza della Classe deve avere MAX una associazione, nell'altra classe Ogni Istanza deve avere MAX n associazioni

uno a molti (one-to-many)

max-card(C1, A) = N
max-card(C2, A) = 1

oppure

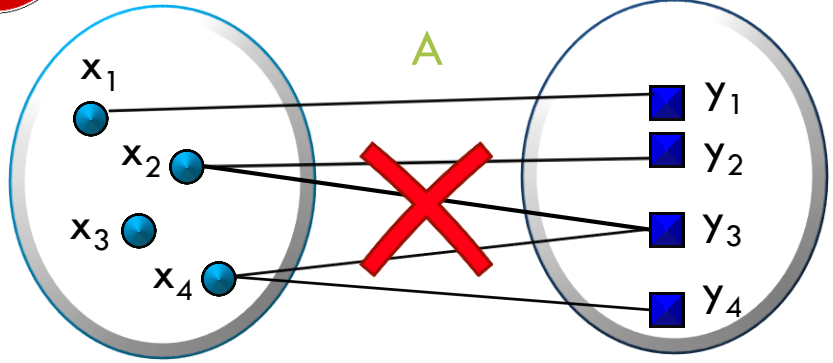
max-card(C1, A) = 1
max-card(C2, A) = N



max-card(C1,A) = N

max-card(C2,A) = 1

Associazioni che non soddisfano i vincoli



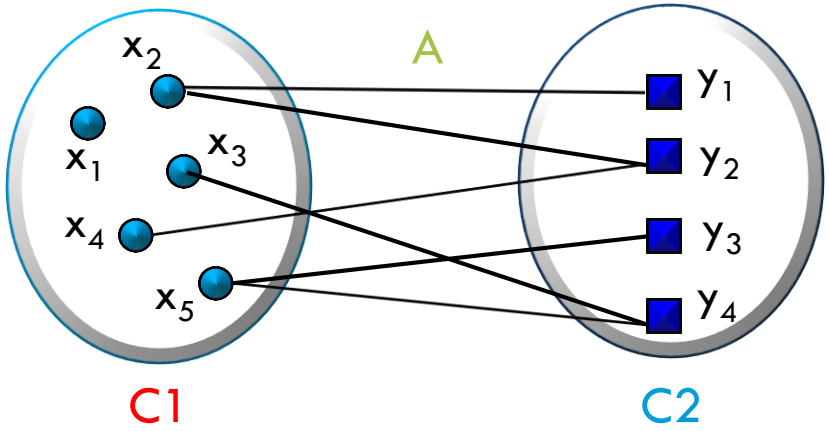
Associazioni binarie *molti a molti*

Ogni Istanza di Ogni Classe deve avere MAX n associazioni

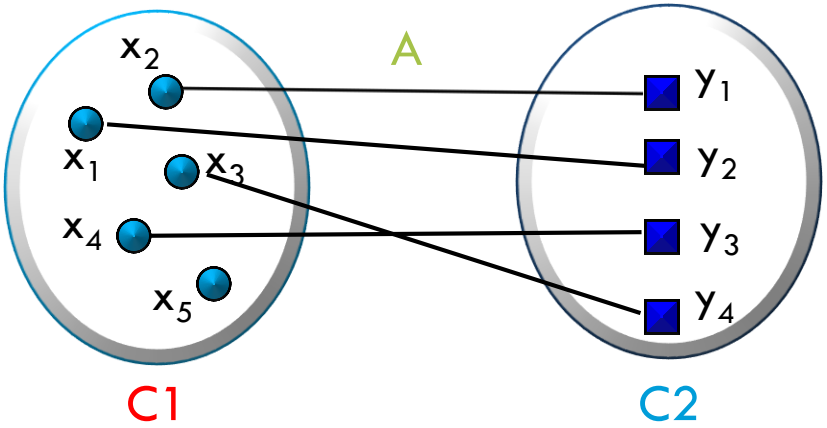
molti a molti (many-to-many)

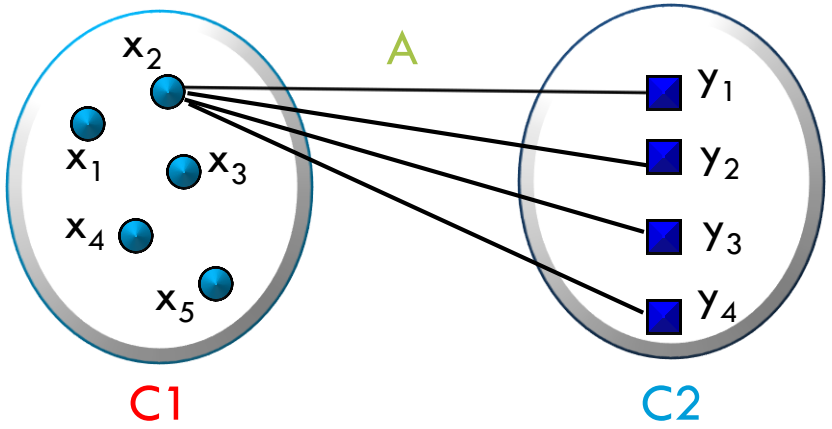
max-card(C1, A) = N

max-card(C2, A) = M



Tutte queste associazioni soddisfano i vincoli





Domande?