UML: Diagramma delle classi; Diagramma degli oggetti

Roberta Gori, Laura Semini
Ingegneria del Software
Dipartimento di Informatica
Università di Pisa

Riassunto lezione precedente
Outline della lezione

- Lezioni precedente:
  - Diagramma dei casi d'uso
- Questa lezione e le prossime
  - Descrizione del dominio

Classi e oggetti

- Una classe cattura
  - un concetto nel dominio del problema o della realizzazione
- Una classe descrive
  - un insieme di oggetti con caratteristiche simili
  - cioè oggetti che hanno lo stesso tipo
- Un oggetto è un'entità caratterizzata da
  - Un'identità
  - Uno stato
  - Un comportamento

Classificatori e istanze

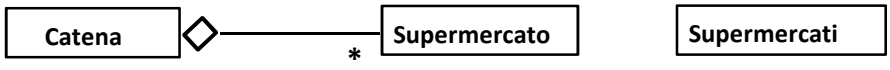
- Le classi sono classificatori
- Gli oggetti sono istanze
- Modellare a livello dei classificatori significa vincolare i modelli a livello di istanza

Diagramma delle classi

- Descrive il tipo degli oggetti che fanno parte di un sistema sw o di un dominio e le relazioni statiche tra essi.
- I diagrammi delle classi mostrano anche le proprietà e le operazioni di una classe

Classi UML vs entità (DB)

- DB: le classi sono intese come collezioni. Sottointeso che
  - ci sono più istanze.
  - ci sono operazioni per visitare tutte le istanze.
- Da cui, per esempio, nome singolare vs nome plurale.
- La differenza è significativa più in prospettiva di progettazione che di descrizione del dominio.
  - In progettazione OO e quindi in UML uso "ListaDiQcosa" come aggregato di "Qcosa"
- Un esempio dal dominio:

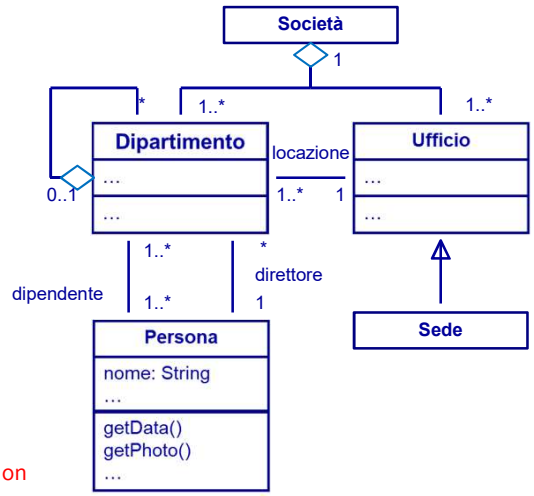


Classi UML

Entità

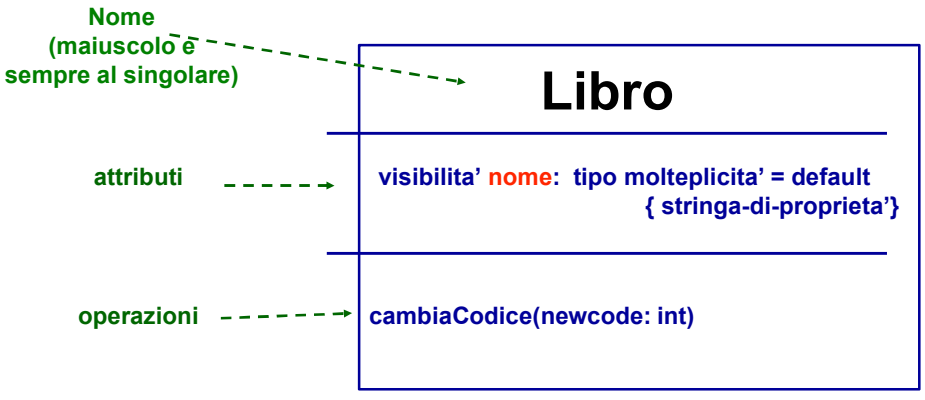
Esempio: classi

- Una società è formata da dipartimenti e uffici
- Un dipartimento ha un direttore e più dipendenti
- Un dipartimento è situato in un ufficio
- Esiste una struttura gerarchica dei dipartimenti
- Le sedi sono uffici



Se togli l'istanza, forse questo esempio non ha più senso, ma vedi tu come ti torna il discorso

Sintassi della classe

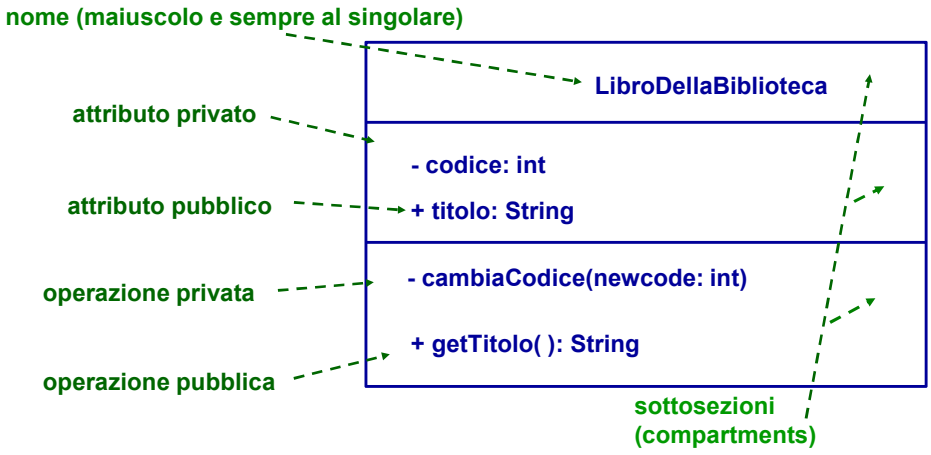


3 sottosezioni
(compartments)

Se non interessano attrib. e operazioni

Libro

Sintassi della classe



Semantica

- Un oggetto è un'entità caratterizzata da
  - Un'identità, uno stato, un comportamento
- Gli attributi definiscono lo stato dell'oggetto
- Le operazioni definiscono il suo comportamento

Visibilità

- Un elemento è visibile all'esterno dello spazio di nomi che lo contiene, in accordo con il suo tipo di visibilità
  - + public: tutti
  - # protected: i discendenti
  - - private: solo nell'elemento stesso
  - ~ package: nello stesso package

Sintassi attributi

- visibilità nome: tipo [molteplicità]= valoreIniziale {proprietà}

obbligatorio

molteplicità:

array di valori

colore: Saturazione [3]

nome: String [0..1] 0 per permettere valore null

[1] può essere omesso

proprietà

{ordered}

{>=0}

Esempi

numero di tipo intero
n: Integer

numero intero positivo
n: Integer{>= 0}

contatore positivo, inizialmente a 0
n: Integer =0 {>= 0}

Esempi

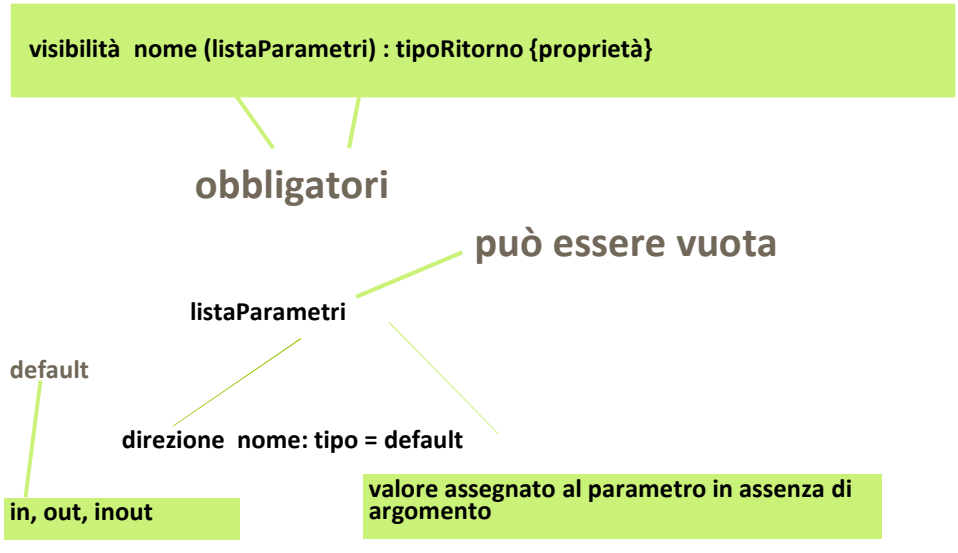
Punti del quadrante positivo, pubblico

+ puntiQuadPos : Integer [2] {>= 0}

sequenza ordinata di 10 interi compresi tra 3 e 33, privato

- seq: Integer[10] {>= 3, <= 33, ordered}

Sintassi operazioni



Visibilità di attributi e operazioni

- In modo simile alla visibilità della classe
  - \+ public: accessibile ad ogni elemento che può vedere e usare la classe
  - # protected: accessibile ad ogni elemento discendente
  - - private: solo le operazioni della classe possono vedere e usare l'elemento in questione
  - ~ package: accessibile solo agli elementi dichiarati nello stesso package

Esempi

Metodo pubblico che restituisce la somma di 2 interi

+ sum (a: Integer, b: Integer) : Integer

sum con 10 valore di default del secondo parametro

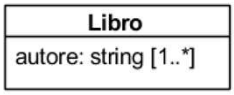
+ sum (a: Integer, b: Integer =10) : Integer

Metodo privato che restituisce un oggetto di tipo Gra

- gra () : Gra

Cosa va modellato con una classe…

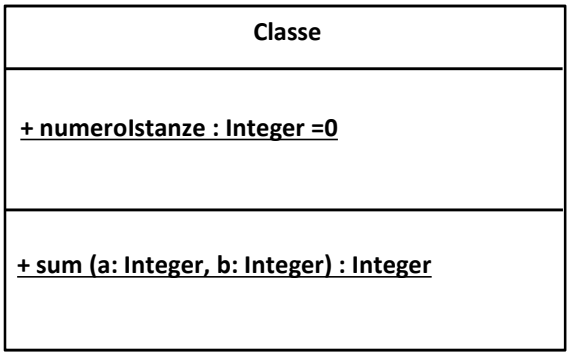
- autore come attributo



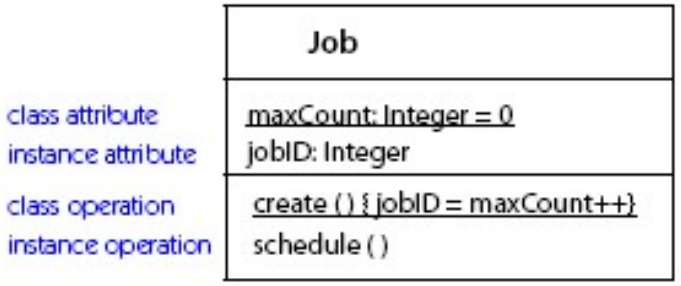
- vs autore come classe

attributi e operazioni con ambito di classe (statici)

- sottolineati



Esempio



Enumerazioni

- Le enumerazioni sono usate per mostrare un insieme di valori prefissati che non hanno altre proprietà oltre al loro valore simbolico
- In UML sono rappresentate da classi
  - etichettate dallo stereotipo <<enumeration>> (vedremo più avanti cosa è esattamente uno stereotipo, per il momento pensate a una keyword)
  - con un nome (il tipo) e l'insieme di valori che quel tipo può assumere

Relazioni

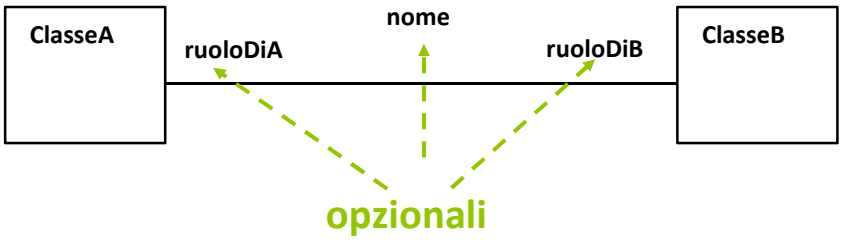
- Una relazione rappresenta un legame
  - tra due o più oggetti
  - normalmente istanze di classi diverse

Relazioni

- Tra elementi di un modello
- Vedremo le seguenti:

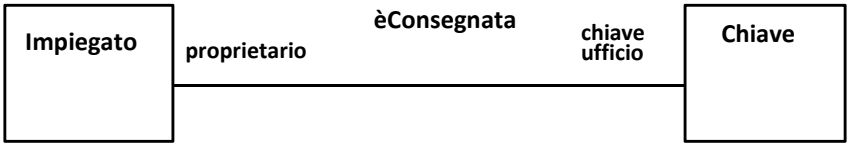
Tra Classi	Tra oggetti
Associazione Aggregazione Composizione	Collegamento Aggergazione Composizione
Generalizzazione	(non definita)
Realizzazione	(non definita)
Dipendenza (d'uso, di istanza…)	

Associazione (binaria): sintassi



- Almeno uno tra nome o ruoli, raramente entrambi.
- Serveno a distinguere associazioni multiple tra le stesse classi

Associazione (binaria): esempio

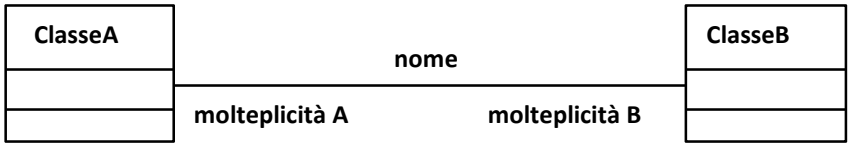


- se i è un Impiegato, e c una Chiave
- e c è stata consegnata ad i,
- si può dire che i è il proprietario di c e che c è la chiave dell'ufficio di i

Associazione (binaria): nome e ruoli

- nome e ruoli: lowercase
- nome associazione: normalmente un verbo
- ruolo: normalmente un sostantivo
- Formalmente opzionali,
  - è utile ci sia o il nome dell'associazione o l'indicazione dei ruoli
  - inutile entrambi

Associazioni: vincoli di molteplicità



Numero di oggetti coinvolti nell'associazione

in un dato istante

Molteplicita' delle relazioni

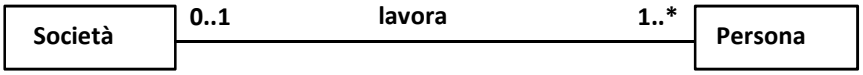
- In generale le molteplicità si possono definire indicando gli estremi inferiore e superiore di un intervallo
  - 2..4 per i partecipanti ad una partita a canasta
  - l'estremo inferiore può essere zero o un numero positivo
  - quello superiore un numero positivo o * (indefinito)
- per indicare n..n si può usare n
- per indicare o..* si può usare *

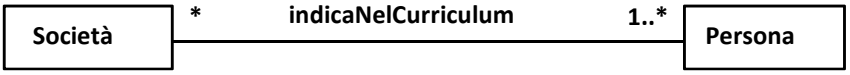
Molteplicità, un esempio

- **Un oggetto Società può essere in relazione con molti oggetti Lavoratore**
- **Un oggetto Lavoratore può essere in relazione con un solo oggetto Società**

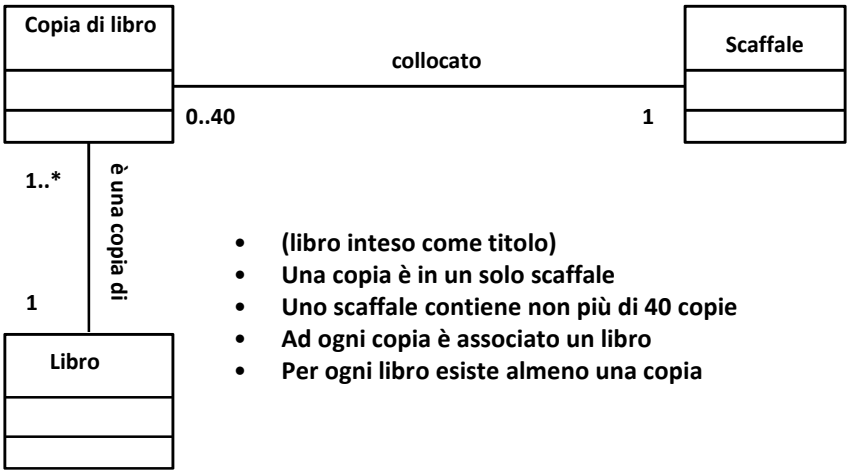
in un dato istante

Attenzione al nome dell'associazione





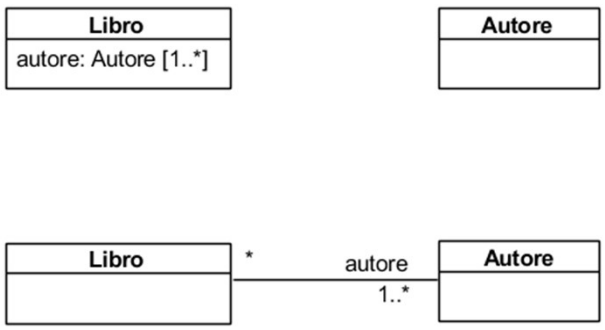
Molteplicità: esempio



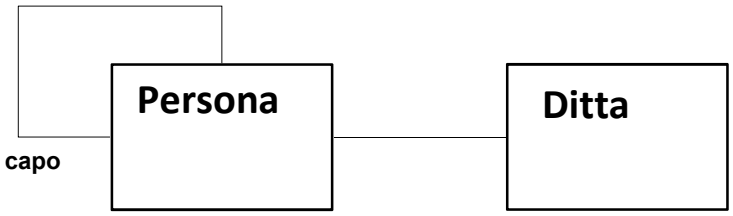
- (libro inteso come titolo)
- Una copia è in un solo scaffale
- Uno scaffale contiene non più di 40 copie
- Ad ogni copia è associato un libro
- Per ogni libro esiste almeno una copia

Associazioni e attributi

- Altro modo per rappresentare una proprieta'



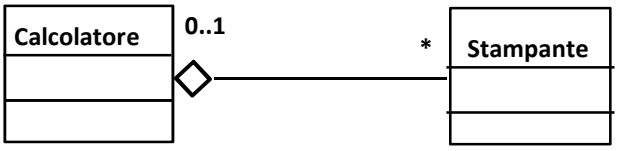
Associazione riflessiva ruoli importanti



Aggregazione e Composizione

- Aggregazione e Composizione sono tipi particolari di associazione (un raffinamento)
  - entrambe specificano che un oggetto di una classe è una parte di un oggetto di un'altra classe
- aggregazione: relazione tra oggetti poco forte, es. calcolatore con le periferiche
- composizione: relazione tra oggetti molto forte, es. albero e le sue foglie

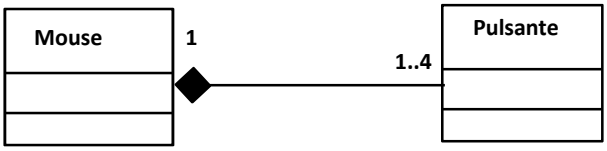
Sintassi dell'aggregazione



L'aggregazione è una relazione del tipo tutto-parte

- La stampante nel tempo può essere collegata a calcolatori diversi
- La stampante esiste anche senza calcolatore

Sintassi della composizione

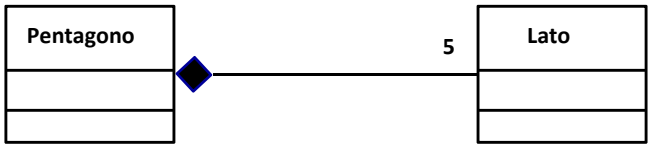


Una composizione è una forma di associazione più forte dell'aggregazione

le parti non hanno senso senza il tutto

una parte appartiene ad un solo tutto

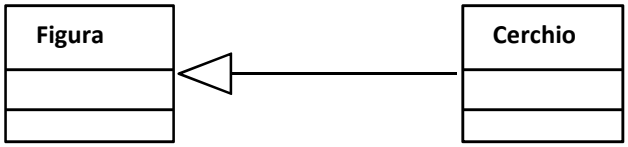
Esempio di composizione



Se cancello un poligono cancello anche i suoi lati

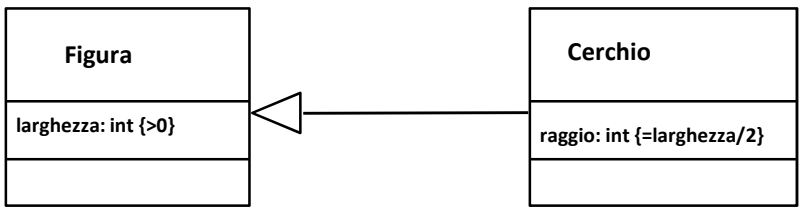
Sintassi della generalizzazione

- relazione tra un elemento generico e uno più specializzato
- elemento più specializzato completamente consistente con quello più generico ma contiene piu' informazione
- principio di sostituibilità: l'elemento specializzato può essere usato al posto dell'elemento generico
- "è un tipo di"



Ereditarieta' di classe

- Le sottoclassi ereditano tutte le caratteristiche della superasse:
  - attributi, operazioni , relazioni e ~~vincoli~~ (non se ne è ancora parlato)
- Le sottoclassi possono aggiungere caratteristiche e ridefinire le operazioni

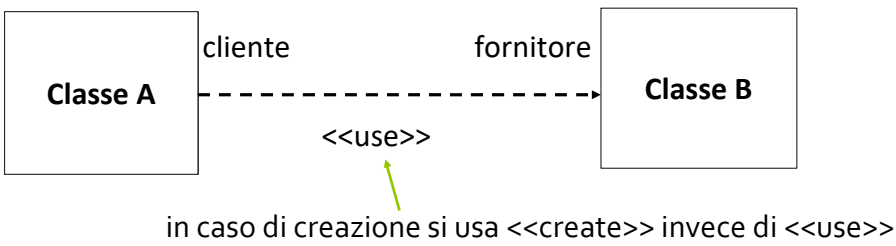


Dipendenze

- Una relazione in cui una modifica nel fornitore può influenzare il cliente
  - Il cliente dipende dal fornitore

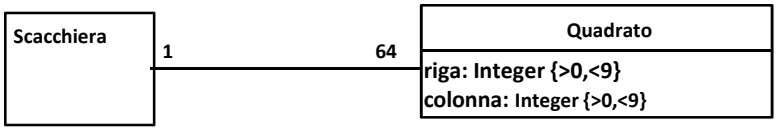
Dipendenza d'uso: esempi

- Le dipendenze più comuni
  - Un parametro di un'operazione di A è di tipo B
  - Un'operazione di A restituisce un oggetto di tipo B
  - Un'operazione di A crea dinamicamente un oggetto di tipo B



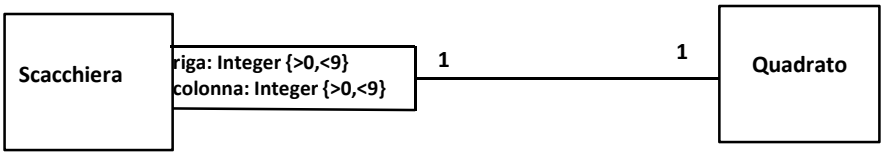
in caso di creazione si usa <<create>> invece di <<use>>

Associazioni qualificate

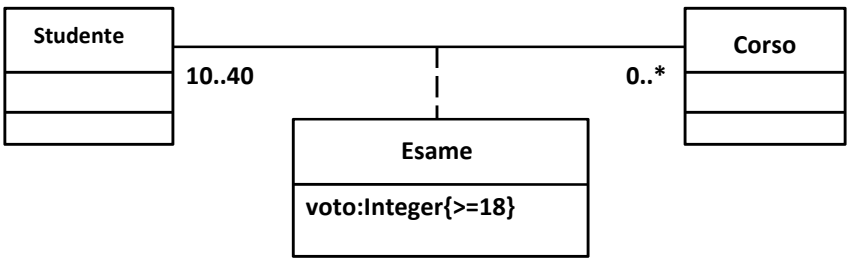


Le associazioni qualificate riducono un'associazione * a molti ad un'associazione * a 1 specificando un unico oggetto scelto nell'insieme destinazione (specificandone una chiave).

La combinazione riga,colonna specifica un unico destinatario



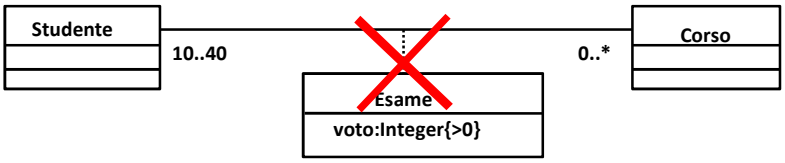
Classi associazione



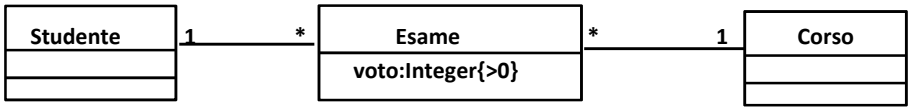
- **Un'associazione può avere attributi propri, rappresentati con una classe associazione.**
- **Le istanze sono collegamenti con attributi propri.**
- **Voto non è attributo né di Corso né di Studente**

Classi associazione (2)

Per ogni coppia di oggetti collegati tra loro può esistere un unico oggetto della classe associazione



Se vogliamo tenere traccia dei voti negativi non si possono usare le classi associazione



Individuare le classi di analisi

- Quali sono le classi
- Tecniche per individuarle
  - Nome-verbo

Cosa sono le classi di analisi

- Corrispondono a concetti concreti del dominio:
  - Per esempio i concetti descritti nel glossario
  - Normalmente, ciascuna classe di analisi sarà raffinata in una o più classi di progettazione.
- Evitare di introdurre delle classi di progettazione

Classi di analisi: caratteristiche

- Astrazione di uno specifico elemento del dominio
- Numero ridotto di responsabilità (funzionalità)
- Evitare le classi "onnipotenti"
  - Attenzione quando si chiamano "sistema", "controllore", ...
- Evitare funzioni travestite da classi
- Evitare gerarchie di ereditarietà profonde (>=3)
- Coesione e disaccoppiamento
  - Tenere responsabilità simili in una classe
  - Limitare interdipendenze tra classi

Classi di analisi: livello di dettaglio

- Operazioni e attributi solo quando veramente utili
  - Le classi di analisi dovrebbero contenere attributi e operazioni ad "alto livello"
  - Limitare la specifica di tipi, valori, etc.
  - Non inventare mai niente!

Identificazione delle classi

- Problema classico delle prime fasi di sviluppo
- Approccio data driven
  - Si identificano tutti i dati del sistema e si dividono in classi (ad esempio mediante identificazione dei sostantivi)
- Approccio responsibility driven
  - Si identificano le responsabilità e si dividono in classi

Analisi nome-verbo

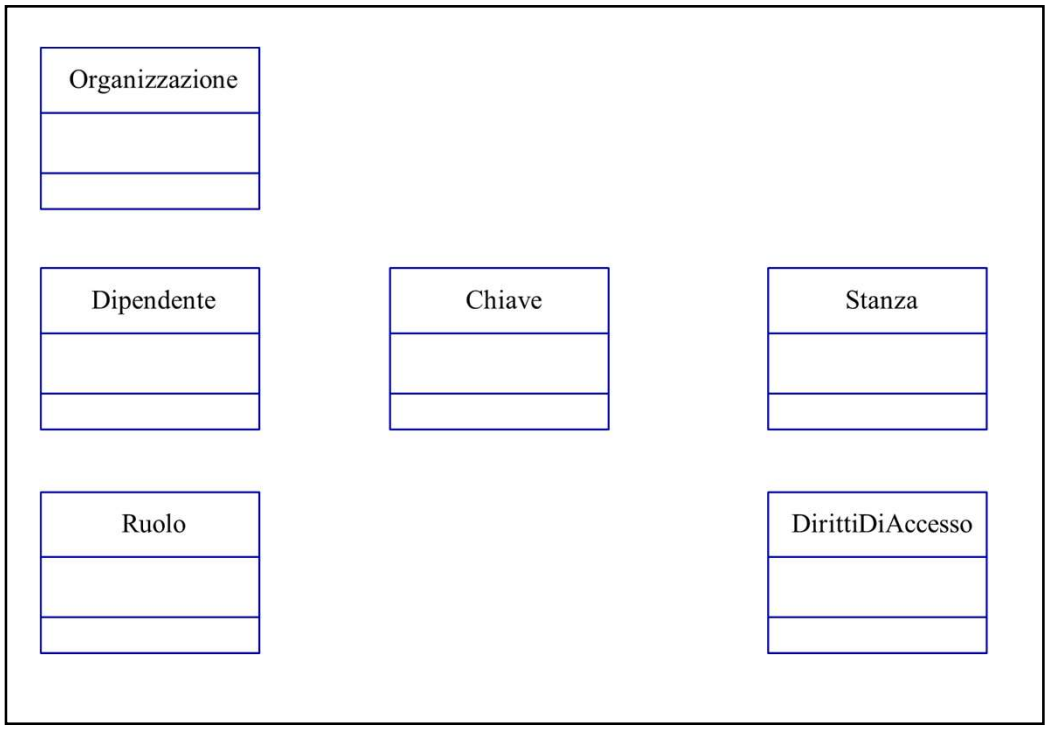
- Sostantivi → classi o attributi
- Verbi → responsabilità o operazioni
- Passi:
  - Individuazione delle classi
  - Assegnazione di attributi e responsabilità alle classi
  - Individuazione di relazioni tra le classi

Analisi nome-verbo

- Problemi ricorrenti :
  - Tagliare le classi inutili
    - Trattare i casi di sinonimia
  - Individuare le classi nascoste cioè le classi implicite del dominio del problema che possono anche non essere mai menzionate esplicitamente
    - In un sistema di prenotazione di una compagnia di viaggi si potrebbe parlare di prenotazione, richiesta, ma tralasciare il termine ordine

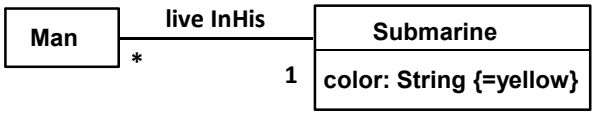
Individuazione classi: chiavi magnetiche

Per motivi di sicurezza, un'organizzazione ha deciso di realizzare un sistema secondo il quale a ogni dipendente è assegnata una chiave magnetica per accedere (aprire) determinate stanze. I diritti di accesso dipenderanno in generale dalla posizione e dalle responsabilità del dipendente. Quindi sono necessarie operazioni per modificare i diritti di accesso posseduti da una chiave se il suo proprietario cambia ruolo nell'organizzazione.

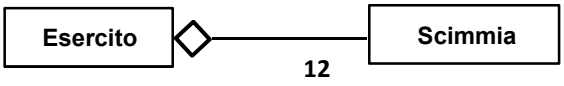


Esempi

- We all live in our yellow submarine

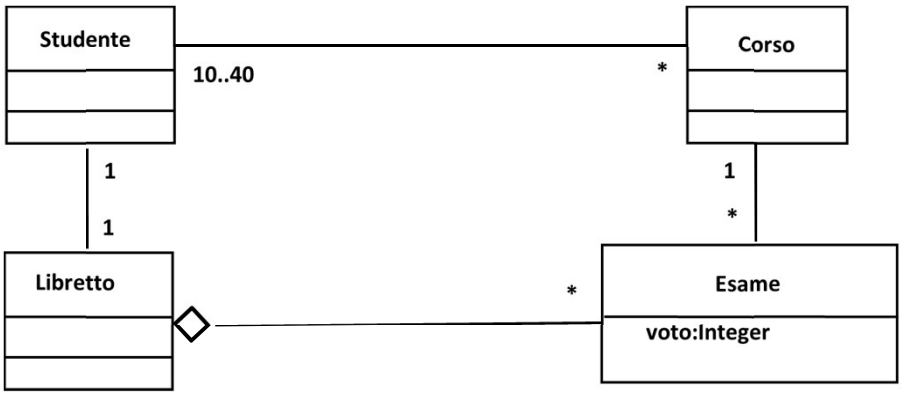


- L'esercito delle 12 scimmie



Esempio

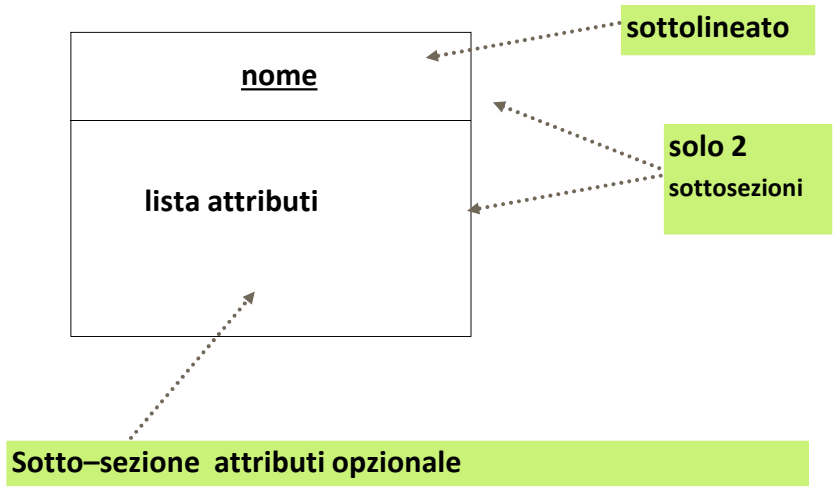
- Studente, Libretto, Esame, Corso



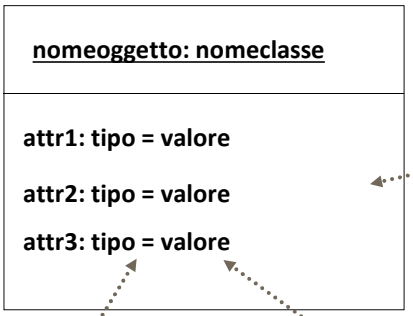
Diagrammi degli oggetti

- viene anche chiamato diagramma delle istanze
- può essere utile quando le connessioni tra gli oggetti sono complicate.

Diagrammi degli oggetti



Diagrammi degli oggetti: attributi



singoli attributi

opzionali

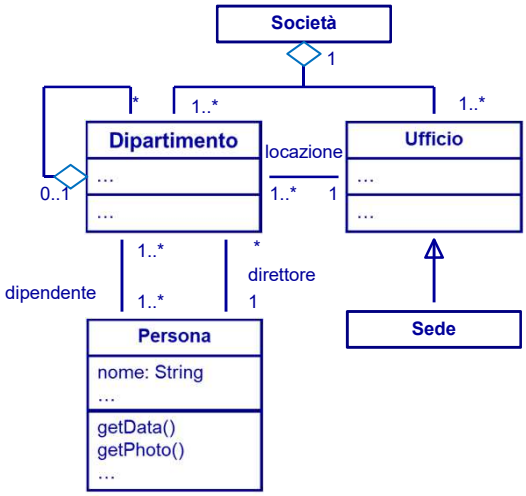
Il valore è la parte interessante può essere omesso ma allora inutile ripetere (vs classe) l'attributo

Il tipo è ridondante ed è consigliato ometterlo

Diagramma degli oggetti: collegamenti

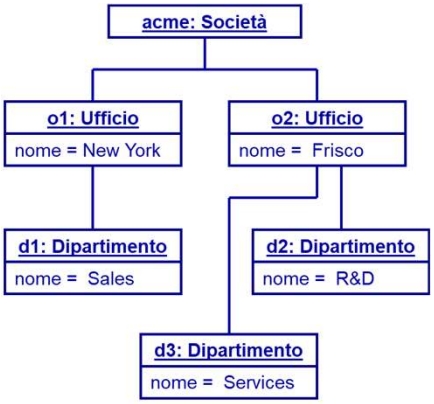
- Un collegamento è una istanza di una associazione
- Collega due (o più) oggetti
- Non ha un nome
- Se utile si possono indicare i ruoli
- Non ha molteplicità, è sempre 1 a 1
  - la molteplicità di una associazione dice quanti collegamenti ci saranno a livello di istanza

Esempio: classi

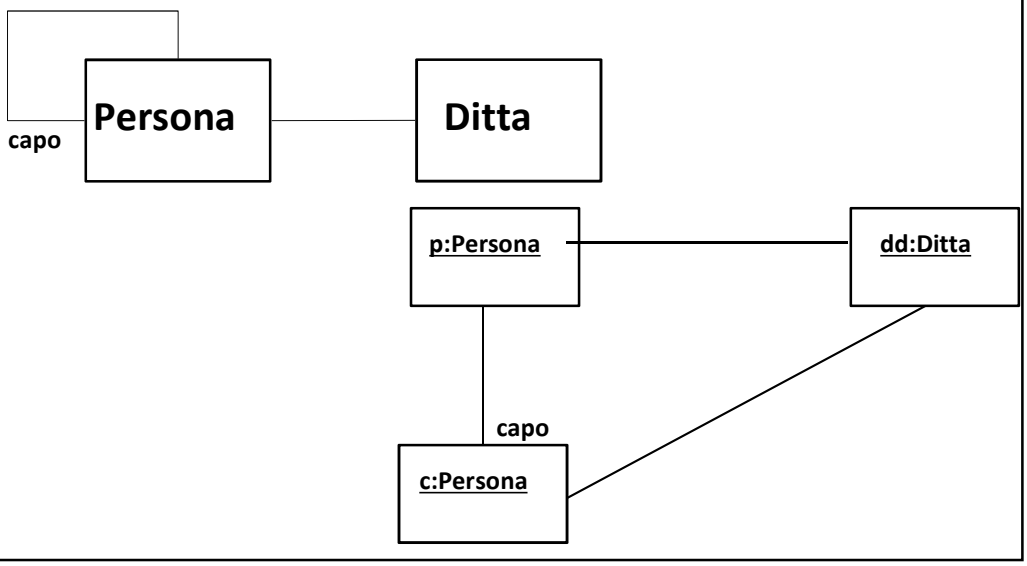


Esempio: oggetti

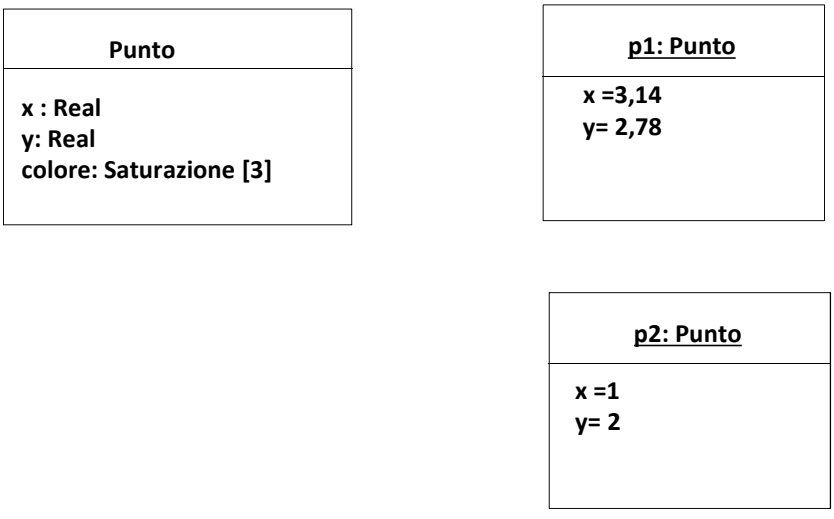
- Una società reale, ACME
- Ha due Uffici
- Il dipartimento vendite è a New York
- I dipartimenti Ricerca&Sviluppo e Servizi sono a San Francisco



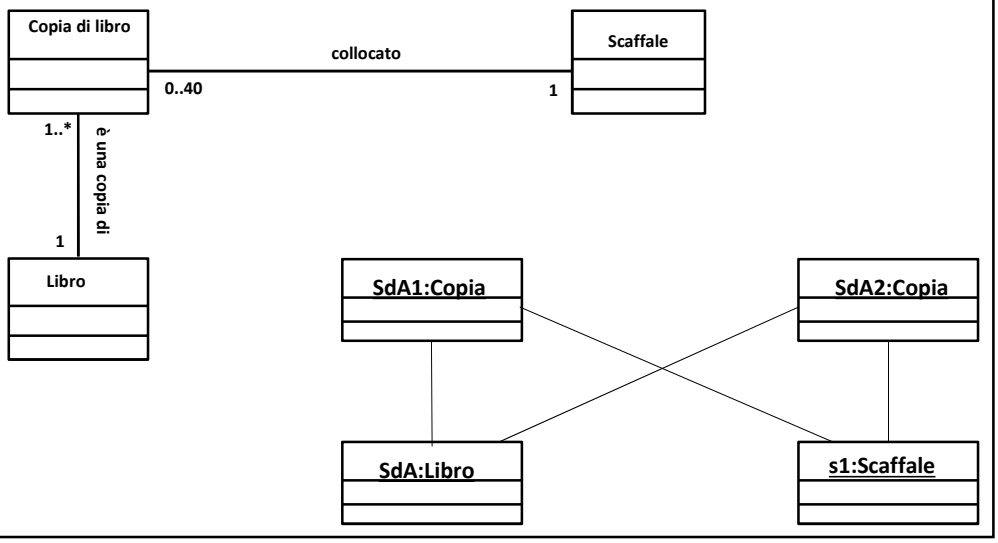
A livello di oggetti



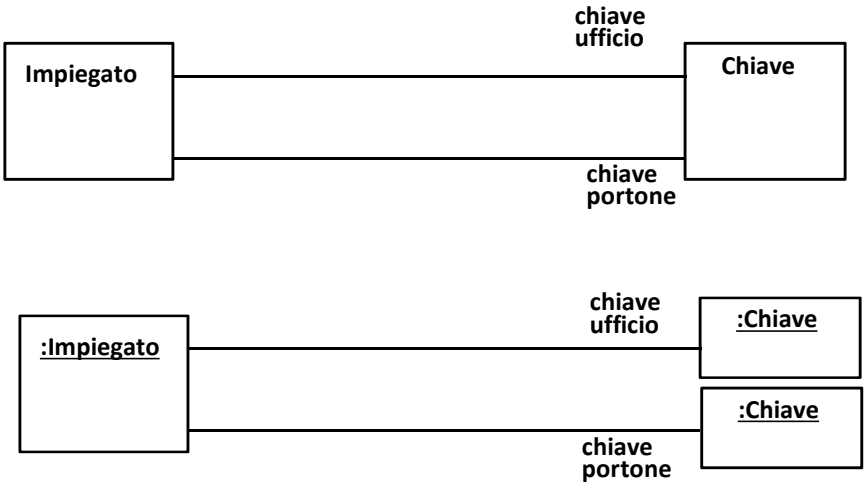
Classi e Oggetti



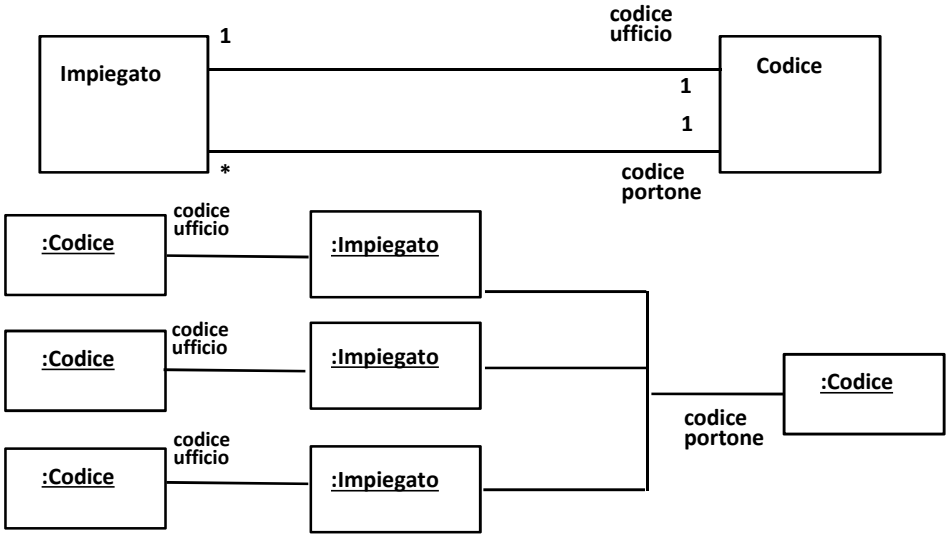
Esempio



Associazione: caso particolare ruoli importanti

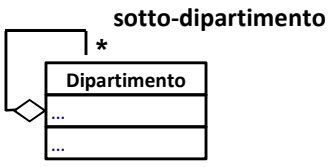


Molteplicità: esempio

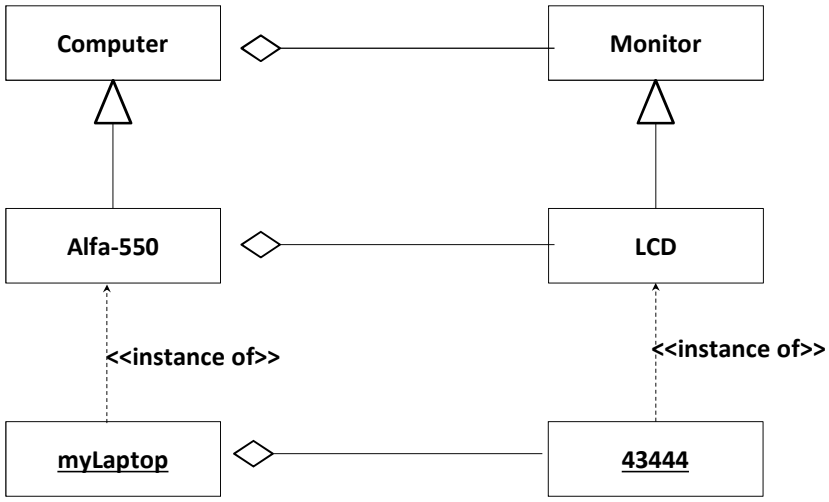


Ruoli e aggregazione: un caso speciale

- Nel diagramma delle classi

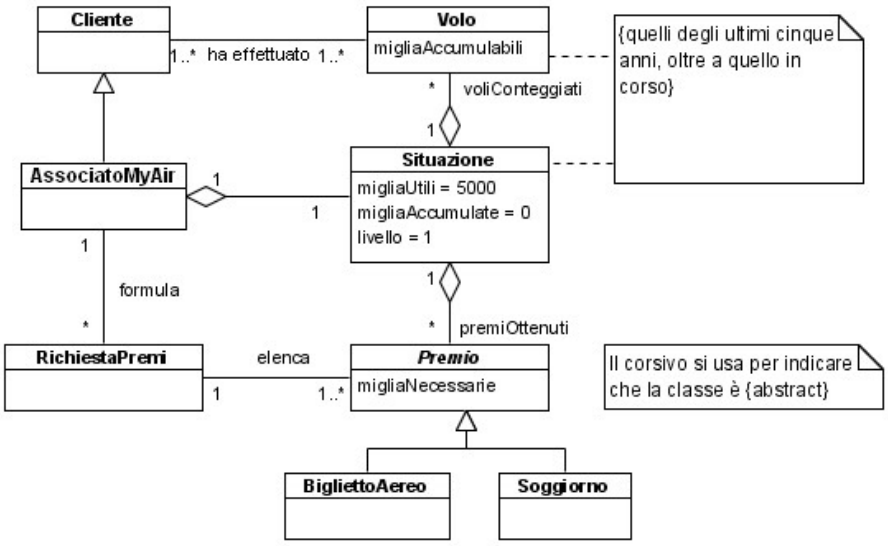


Un esempio con generalizzazione, istanza e aggregazione



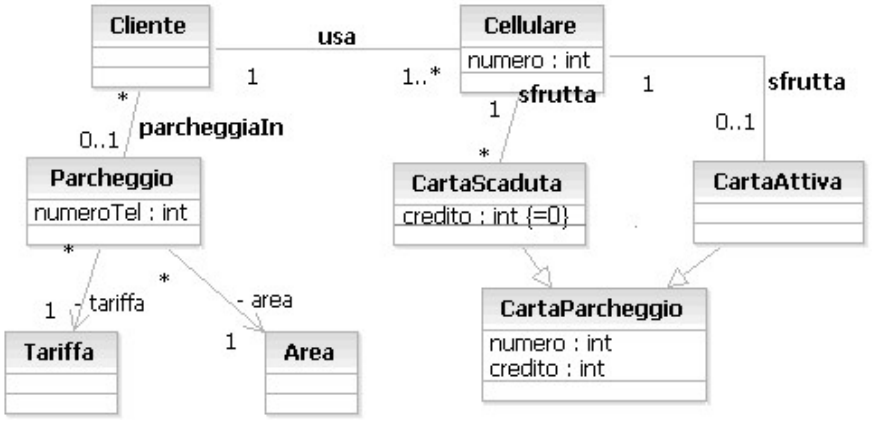
Ex: MyAir

- Iscriviti al programma e da semplice cliente diventerai un associato MyAir, guadagnando immediatamente un bonus di 5.000 miglia utili.
- Ogni volta che volerai con MyAir le miglia accumulabili del volo saranno sommate alle tue miglia utili, permettendoti di raggiungere in poco tempo le miglia necessarie per richiedere uno dei nostri premi (omaggio biglietti aereo o soggiorni in località da sogno).
- I premi riscossi danno luogo a una diminuzione immediata delle miglia utili. La situazione è aggiornata il 31 dicembre, mantenendo solo le miglia dei voli effettuati negli ultimi 5 anni.
- Inoltre se accumulerai almeno 15.000 miglia (miglia accumulate) sarai promosso dal livello standard al livello argento. Se invece accumulerai almeno 100.000 miglia entrerai a far parte del ristretto numero di associati del livello oro2.
- Tutte le condizioni si riferiscono esclusivamente alle miglia accumulate in un anno. Il passaggio da un livello all'altro è effettuato il 31 dicembre. La permanenza nel livello da un anno all'altro è soggetta al rispetto degli stessi requisiti per entrare nel livello. Il bonus iniziale non concorre al raggiungimento delle miglia richieste per cambiare o mantenere un livello.



Easy Park

- Soluzione per il pagamento del parcheggio via telefono cellulare.
- 1. Il cliente acquista una Carta Parcheggio prepagata e l'attiva indicando il proprio numero di cellulare. Durante l'attivazione, il sistema trasferisce sulla nuova carta l'eventuale credito residuo su una carta già associata al numero di telefono indicato.
- 2. Il cliente parcheggia ed espone sul cruscotto la Carta Parcheggio. Nel cartellone del Parcheggio verifica qual è il numero di telefono che identifica l'area e la tariffa. Il cliente telefona a questo numero, il cliente è identificato attraverso il proprio numero di telefono cellulare e il sistema attiva il pagamento della sosta.
- 3. Il Controllore controlla l'effettivo pagamento della sosta inserendo il numero della Carta Parcheggio in un applicativo fruibile tramite Pocket PC connesso a internet o Telefono Cellulare.
- 4. Disattivazione della sosta con chiamata via cellulare: l'utente chiama il numero associato al parcheggio, il sistema riconosce l'utente e disattiva il pagamento. Inoltre il sistema comunica vis SMS la disattivazione, la somma pagata, la durata della sosta e il residuo presente sulla Carta Parcheggio.



Modellare anche sosta, con inizio, fine costo (classe associazione)