# Progettazione di dettaglio

## Diagrammi di struttura composita

Ingegneria del Software

C. Montangero L. Semini

a.a. 2012/2013

# Classificatore strutturato

- È un classificatore di cui si vede la struttura interna, data in termini di *parti*, *porti* e *connettori*.
- Un classificatore strutturato definisce l'implementazione di un classificatore:
  - Così come le interfacce del classificatore definiscono *cosa* deve fare
  - La sua struttura interna definisce *come* viene fatto il lavoro.
- I tipi che definiscono le parti contenute in un classificatore strutturato, possono a loro volta essere strutturati, ricorsivamente.

# Parte

- Una parte ha un nome, un tipo e una molteplicità (anche se sono tutti facoltativi):

  nomeParte : Tipo [molt]

- Una parte p:T descrive il ruolo che una istanza di T gioca all'interno dell'istanza del classificatore la cui struttura contiene p.
- La molteplicità indica quante istanze possono esserci in quel ruolo.
- Un'istanza di p è un'istanza di T (e quindi di un qualunque sottotipo).

# Connettore

- Può essere di due tipi:
  - di assemblaggio (assembly connector)
    - Esprime un legame che permette una comunicazione tra due istanze nei ruoli specificati dalla struttura.
  - di delega (delegation connector)
    - Identifica l'istanza che realizza le comunicazioni attribuite a un porto

# Connettori: notazione

- Di assemblaggio
  - Collega i porti delle due parti le cui istanze devono comunicare
  - Si usa la notazione lollipop.

- Di delega
  - Si usa una semplice linea tra il porto della parte e il porto della struttura composita.

# Diagrammi di struttura composita

- Mostrano la struttura di dettaglio (o struttura interna) di un classificatore a tempo di esecuzione
  - Di solito  componente
  - Ma anche  di una classe
- Sono definiti in termini di:
  - Parti (eventualmente con molteplicità)
  - Porti (eventualmente con interfacce fornite e/o richieste) per mostrare il comportamento visibile all'esterno
  - Connettori
- Esplicitano le interazioni fra le parti e punti di interazione (porti) con l'esterno

# Dalla specifica all'implementazione passando per le strutture composite

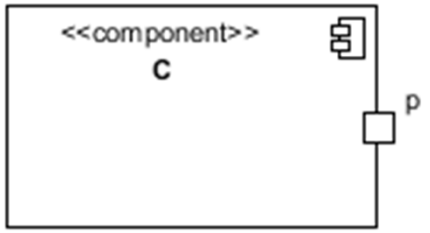



Fig. 1

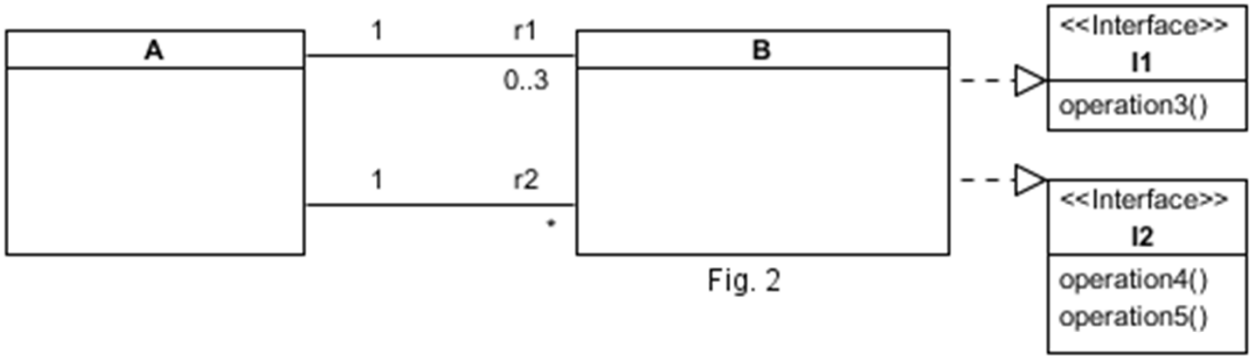



Fig. 2


Fig. 3

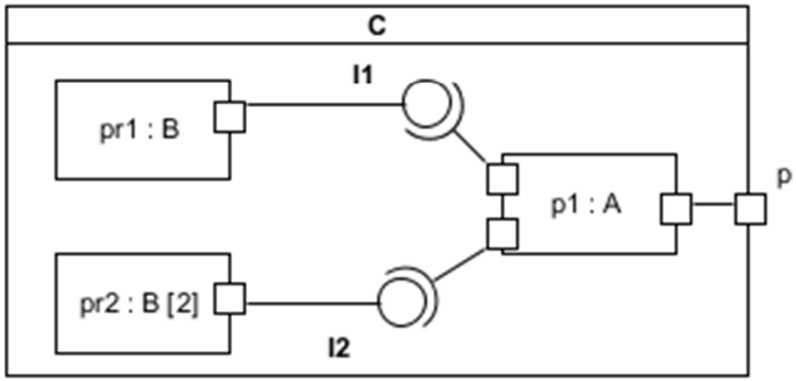



Fig. 4

# Legenda

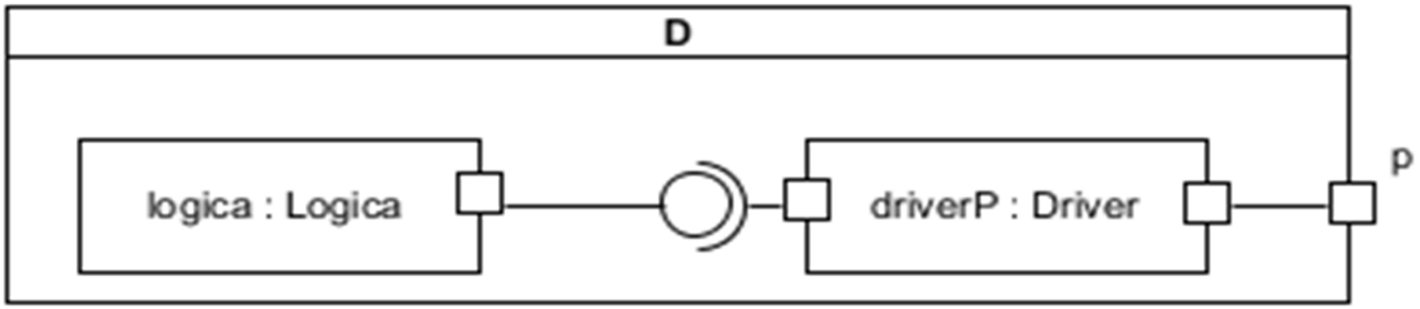
- In Fig. 1 si mostra la componente C
- In Fig. 2 si specificano le classi A e B, con associazioni, ruoli e interfacce. Si assume che un'istanza di A usi l'interfaccia I1 (I2) di B quando connessa con un'istanza di B in ruolo r1 (r2 risp.)
- In Fig. 3 si mostra un diagramma degli oggetti conforme al diagramma in Fig. 2
- In Fig. 4 si mostra la struttura di C, che fornisce una astrazione del diagramma in Fig. 3. Introduce vincoli rispetto al diagramma delle classi: dice che uso istanze di A e di B, collegate in un certo modo, con dato ruolo nel contesto di C, e date molteplicità.

# Metodo: come si struttura una componente

- Un modo conveniente di strutturare una componente prevede di separare gli aspetti di comunicazione da quelli di realizzazione delle funzionalità richieste.
  - favorisce modificabilità e comprensibilità della componente.

# Il diagramma minimo (1/2)

- Supponiamo di avere una componente D con un porto p.
- La struttura di D dovrà avere almeno due parti
  - driverP, che realizza la parte di comunicazione richiesta per implementare il porto.
  - logica, che realizza la funzionalità richiesta alla componente.
- e due connettori
  - di delega tra driverP e P
  - di asseblaggio tra driverP e logica (il verso del lollipop non è fissato a priori)



# Il diagramma minimo (2/2)

In generale, data una componente con n porti, la sua struttura minima dovrà prevedere:

- n parti col ruolo di driver
  - collegate ognuna a un porto, con un connettore di delega
- una parte che realizza la logica della componente
  - collegata a tutti i drivers con connettori di assemblaggio

I nomi "driver" e "logica" per le parti di una componente non sono standard di UML. Sono usati per scopi didattici, per aiutare a distinguere le loro responsabilità.

# Il diagramma non minimo

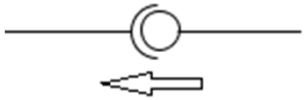
E’ ovviamente possibile, quando necessario, dettagliare maggiormente la struttura di una componente:

- Raffinando la "logica" in un insieme di parti interconnesse
  - Con connettori di assemblaggio
  - Con dipendenze
- Introducendo dei "proxy", per realizzare comunicazioni con sistemi remoti o chiamate al sistema operativo.
  - Connessi alle parti che descrivono la logica, con connettori di assemblaggio
  - Non sono collegati ai porti, perché non realizzano la comunicazione con altre componenti del sistema che si sta progettando

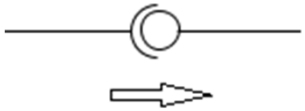
Il nome "proxy" è introdotto per scopi didattici, per distinguere il ruolo da quello di un driver.  Non è corretto pensare sia simile al proxy di RMI.

# Verso del lollipop e verso dell'informazione

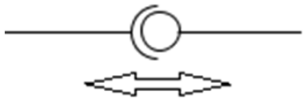
- Il verso del lollipop non ha alcun legame con il verso in cui viaggiano i dati. Ha solo a che vedere con chi ha il controllo e con chi, interrogato, risponde.
- Si pensi:
  - a un'interfaccia con solo operazioni di read



  - a un'interfaccia con solo operazione di write



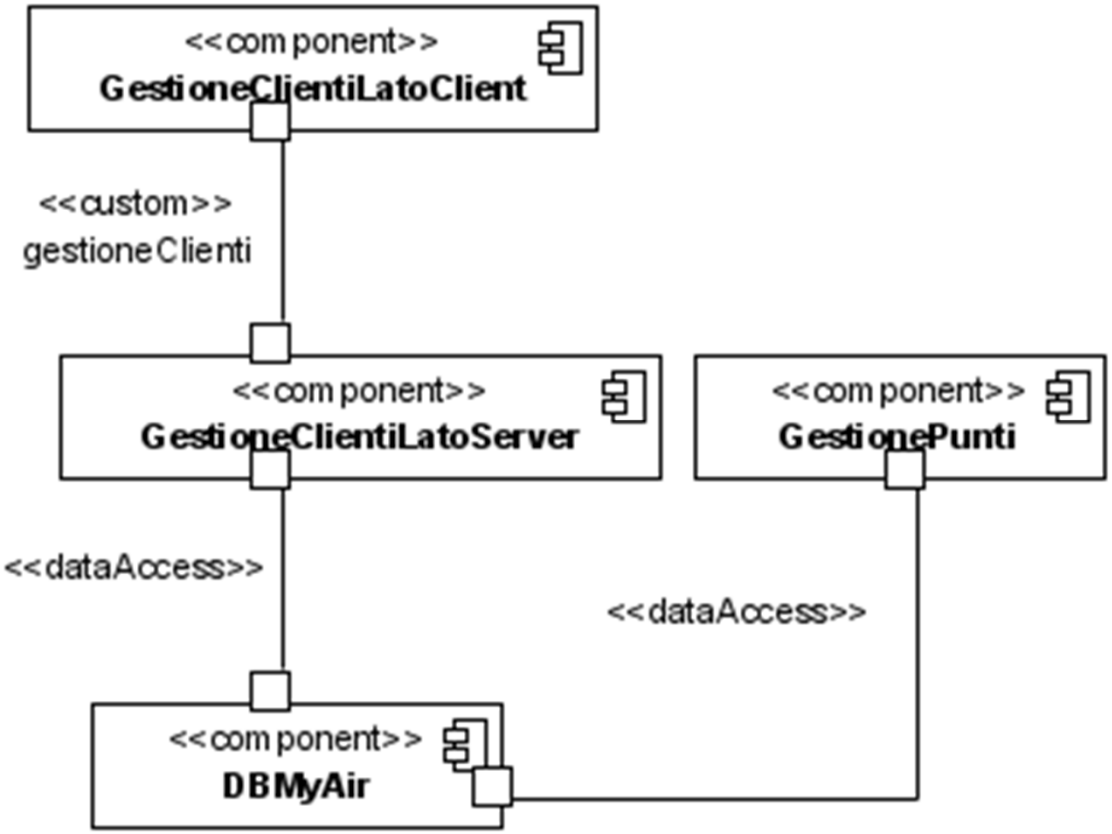
  - a un'interfaccia con operazioni di read e write



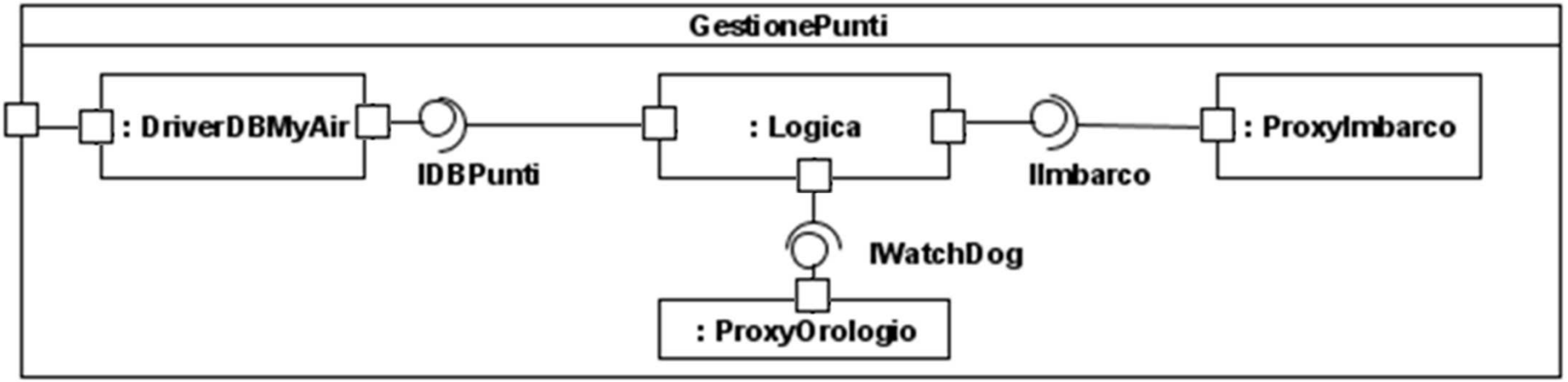
# Esercizio: MyAir (1/2)

- Si consideri la seguene vista C&C

- La componente GestionePunti realizza i casi d'uso AccumuloPunti e AggiornamentoAnnuale



# Esercizio: MyAir (2/2)




| Parte | Responsabilità |
|---|---|
| DriverDBMyAir | Realizza l'interfaccia IDBPunti che permette a Logica di accedere tramite il solo porto della componente al DBMyAir. |
| ProxyImbarco | Realizza la connessione con il sistema Imbarco, passando alla Logica la lista degli imbarcati. |
| ProxyOrologio | Sveglia la logica alla data e ora richiesta tramite IWatchDog. |
| Logica | Realizza i casi d'uso, sfruttando le interfacce introdotte. |

# Vecchia notazione

- Le dispense di esercizi sono state scritte prima dell'ultimo rilascio di UML (2.4.1). Troverete una notazione leggermente diversa:
  - Connettori tra parti collegati alle parti invece che ai loro porti
  - L'associazione unidirezionale ( → ) per indicare la direzione del flusso di dati.
  - Dipendenze al posto di forchette