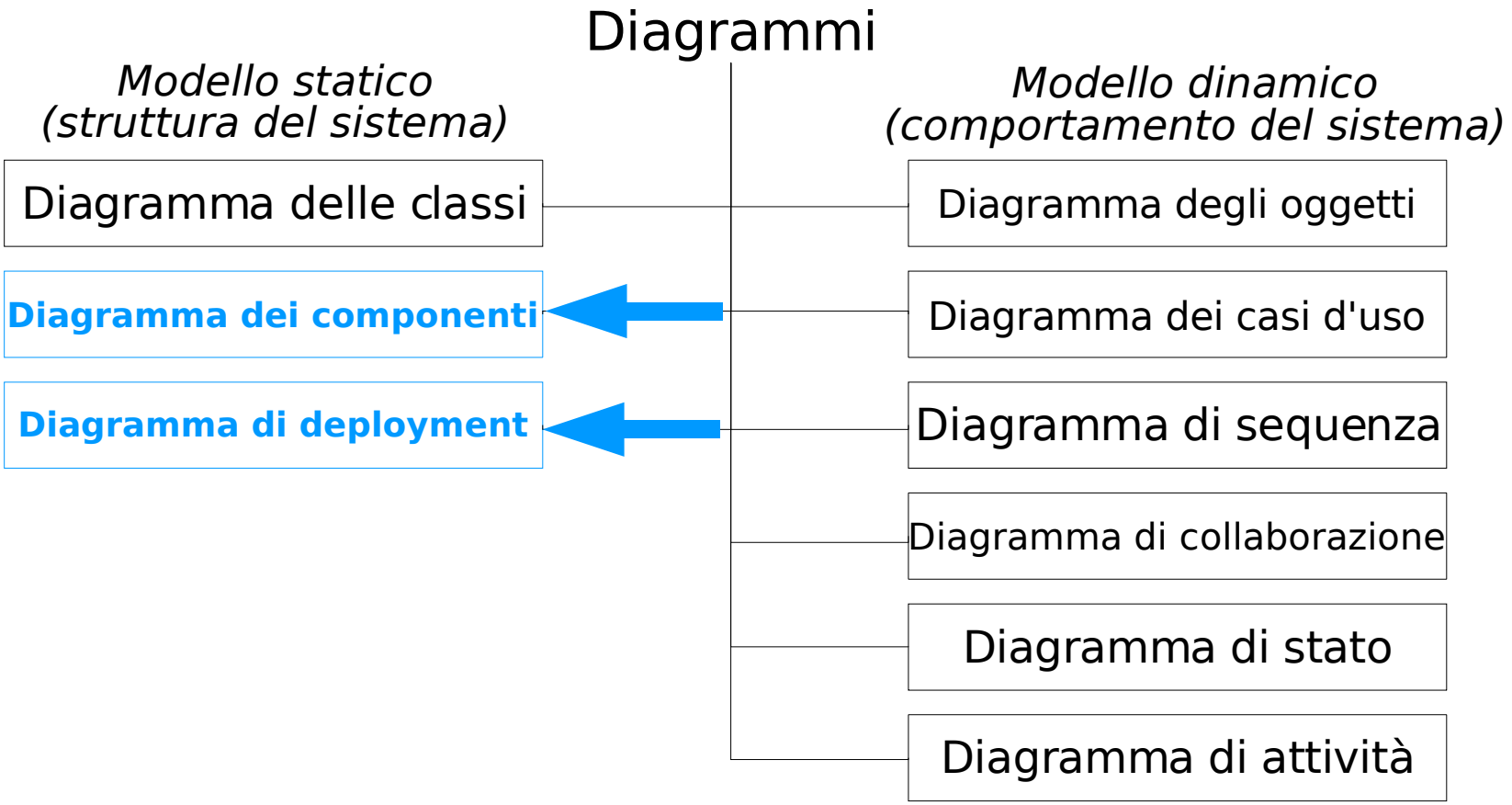
Elementi di UML (7): Diagrammi dei componenti e di deployment

Università degli Studi di Bologna
Facoltà di Scienze MM. FF. NN.
Corso di Laurea in Scienze di Internet
Anno Accademico 2004-2005

Laboratorio di Sistemi e Processi Organizzativi

Diagrammi di implementazione

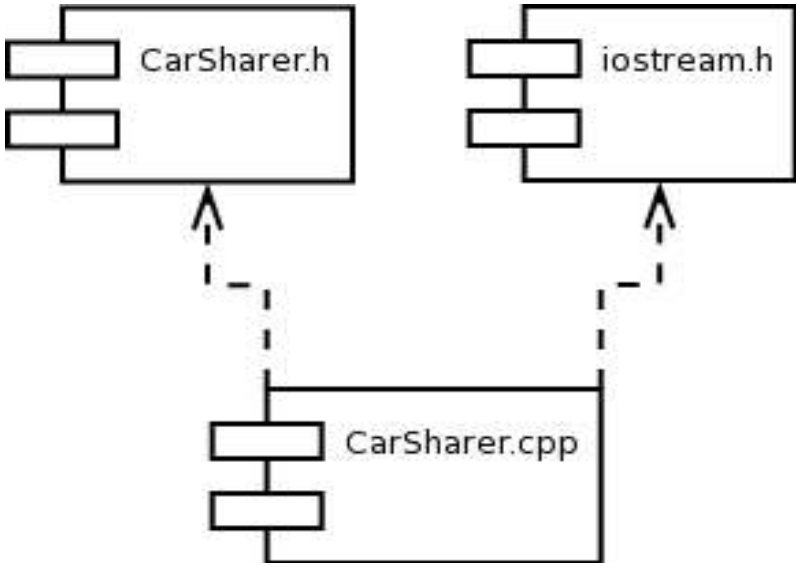


Diagrammi di implementazione

- In UML esistono due tipi di diagrammi per la modellazione degli aspetti implementativi di un sistema informatico:
  - I diagrammi dei componenti
  - I diagrammi di deployment
- *Definizioni UML di componente*: “è una parte, fisica e sostituibile di un sistema, che contiene implementazione e che rispetta un insieme di interfacce e ne fornisce una realizzazione”

Diagrammi dei componenti

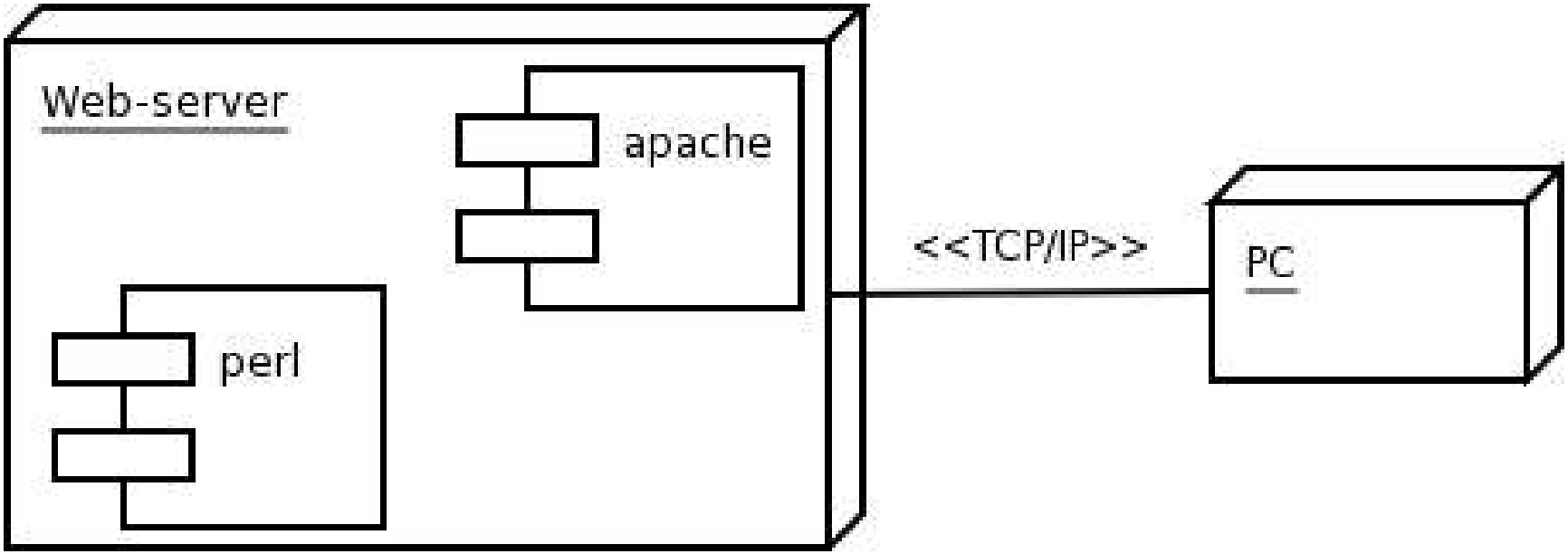
- Un diagramma dei componenti serve a modellare le relazioni tra i componenti software del sistema, che solitamente si stabiliscono
  - tra i file del codice sorgente
  - tra i componenti run-time
  - tra gli eseguibili compilati e i loro sorgenti



Diagrammi di deployment

- Un diagramma di deployment viene sfruttato per modellare l'hardware utilizzato per l'implementazione del sistema e i collegamenti tra i diversi pezzi hardware.
- In un diagramma di deployment si possono disporre anche i singoli componenti, così da mettere in evidenza la loro collocazione nei vari eleaboratori.

Diagramma di deployment con componenti (esempio)



Diagrammi dei componenti (1/8)

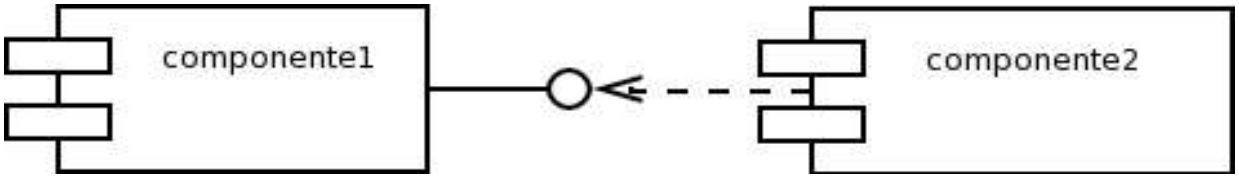
- I ***diagrammi dei componenti*** vengono utilizzati per modellare i componenti di un sistema da un punto di vista **statico**
- I *componenti* possono essere oggetti come
  - i file,
  - i programmi eseguibili,
  - i documenti,
  - le librerie,
  - le tabelle di dati

Diagrammi dei componenti (2/8)

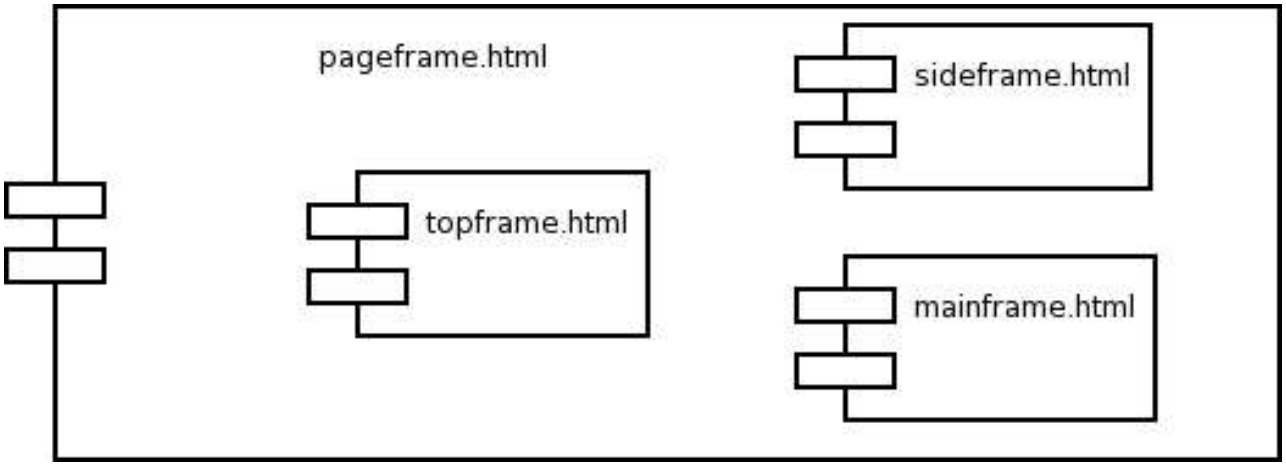
- Un diagramma dei componenti è un *grafo* di componenti collegati tra loro da *relazioni di dipendenza*
- Il *nome di un componente* è tipicamente
  - il nome di un file fisico
  - Il nome di un sottosistema di un certo tipo che è stato progettato per essere un componente
- Le dipendenze tra componenti sono indicate da frecce tratteggiate che li collegano, dal componente dipendente verso quello da cui dipende

Diagrammi dei componenti (3/8)

- Le dipendenze possono essere etichettate con uno stereotipo che indica la natura della dipendenza.
- Un componente può implementare un'interfaccia

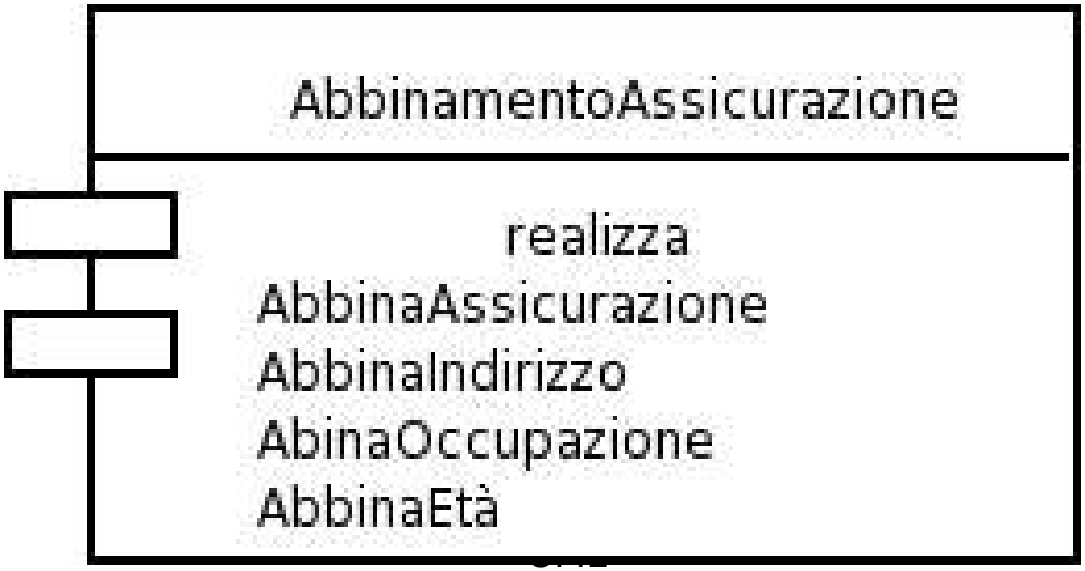


- Un componente può contenere uno o più componenti al suo interno (**composizione**)



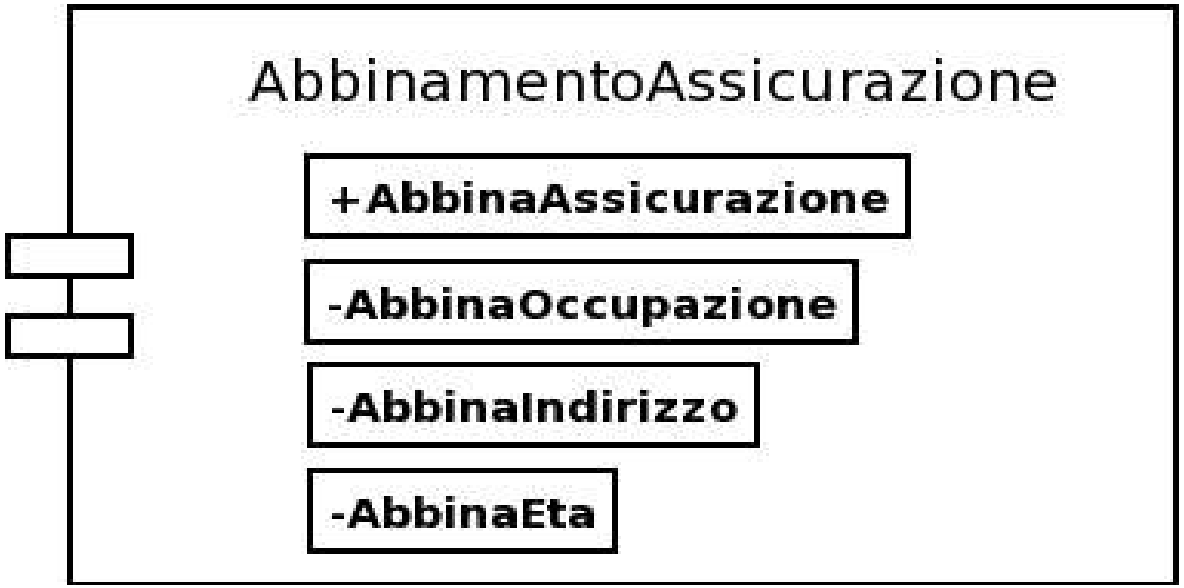
Diagrammi dei componenti (4/8)

- Trattandosi di un classificatore, un componente può avere operazioni
- Informazioni relative al componente come la classe che realizza possono essere mostrate in appositi riquadri



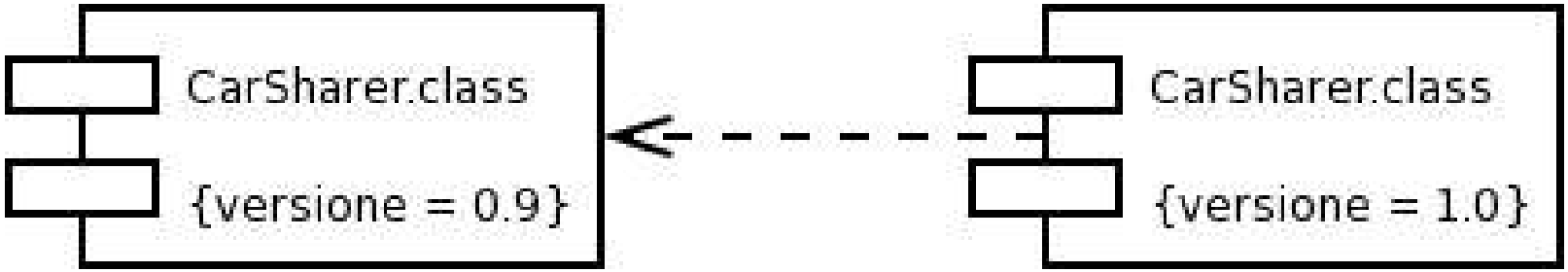
Diagrammi dei componenti (5/8)

- Le classi realizzate da un componente possono essere rappresentate anche contenute al suo interno.
- I nomi delle classi possono essere etichettati con un segno “+” o “-” per specificare la loro visibilità alle altre classi o componenti al di fuori del componente che le implementa.



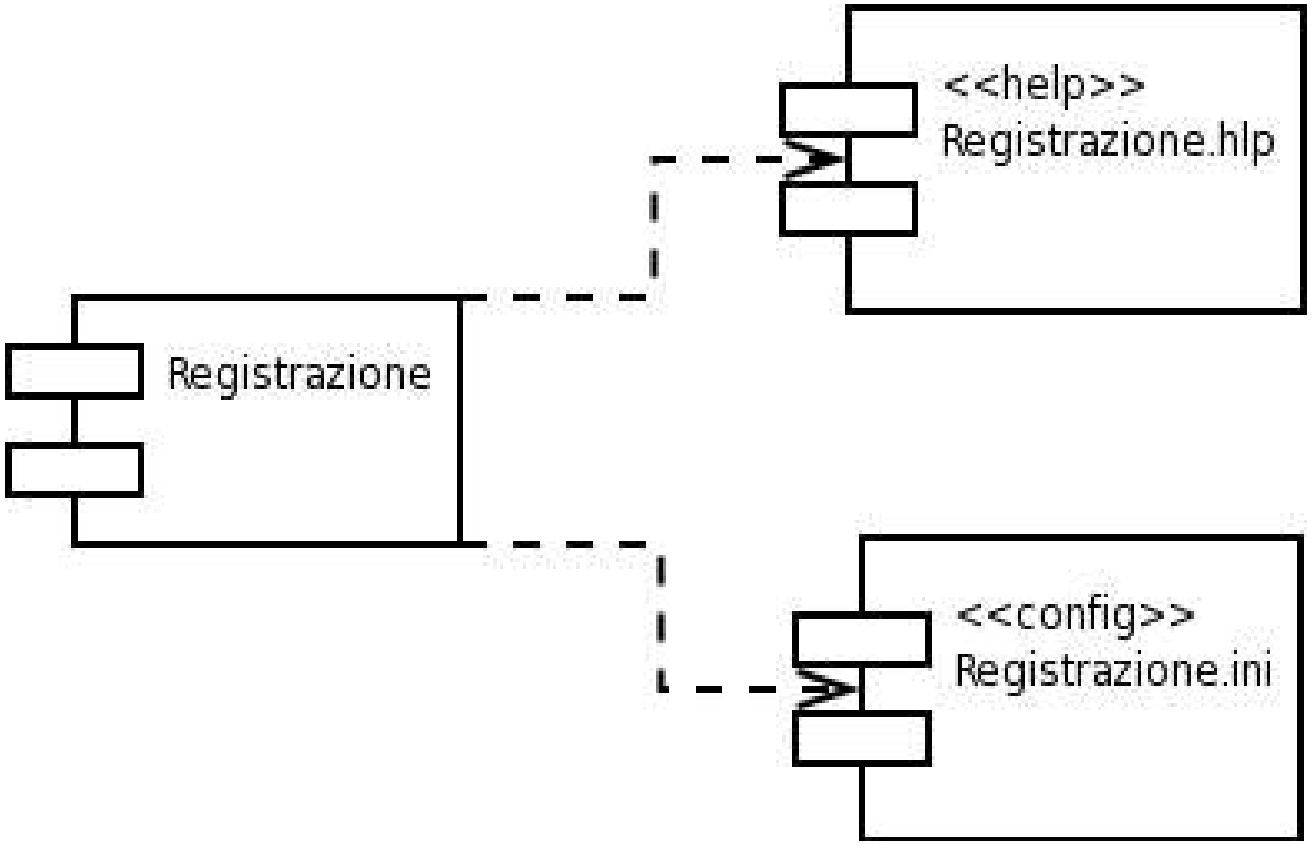
Diagrammi dei componenti (6/8)

- Ai componenti possono essere anche aggiunti **vincoli**, per mostrare ulteriori informazioni.



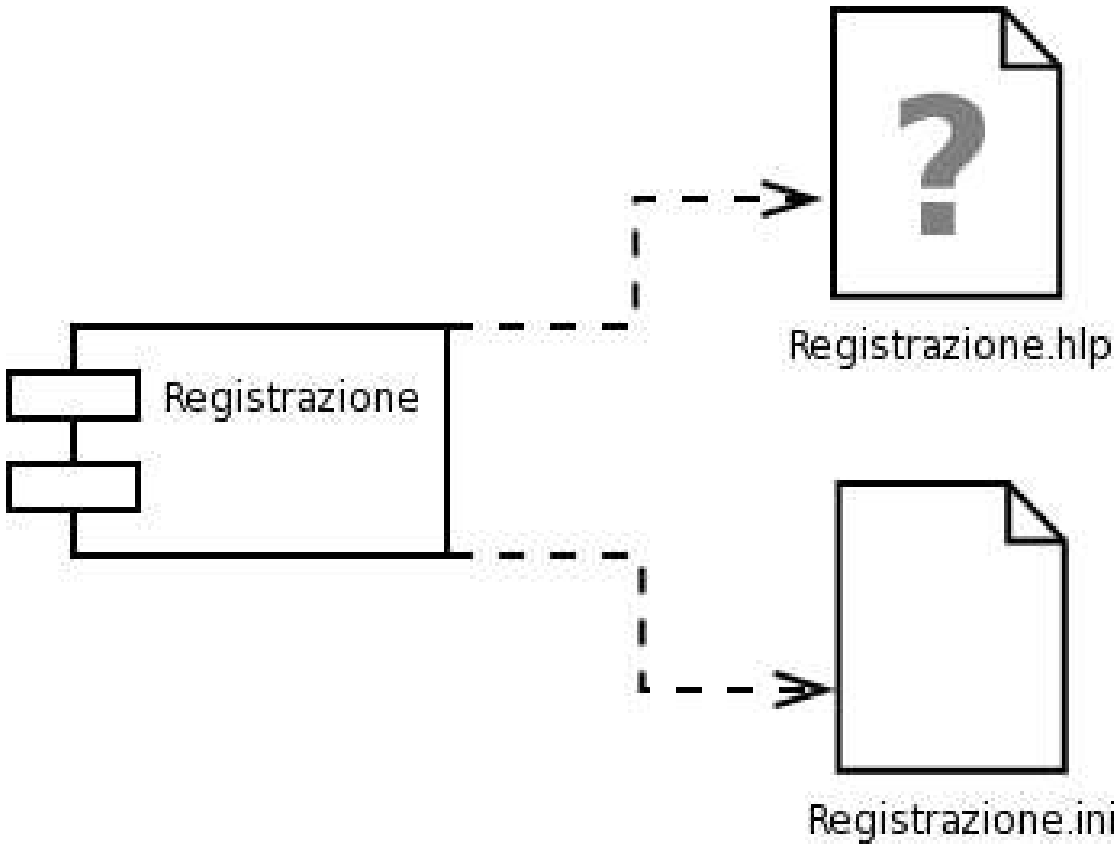
Diagrammi dei componenti (7/8)

- La notazione standard per i componenti è usata titpicamente per indicare programmi
- È possibile usare stereotipi per i componenti



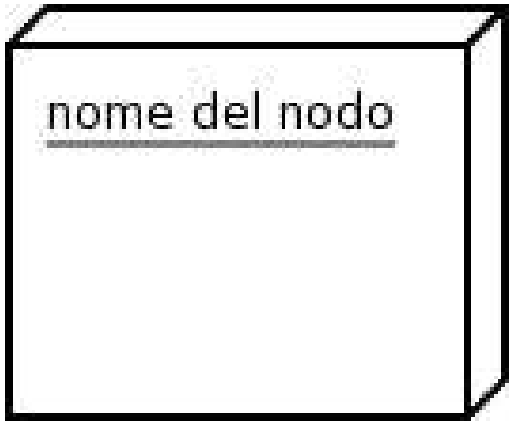
Diagrammi dei componenti (8/8)

- Se si vuole mettere in risalto che il componente non è un programma si possono usare simboli di stereotipi per i componenti al posto dei simboli standard.



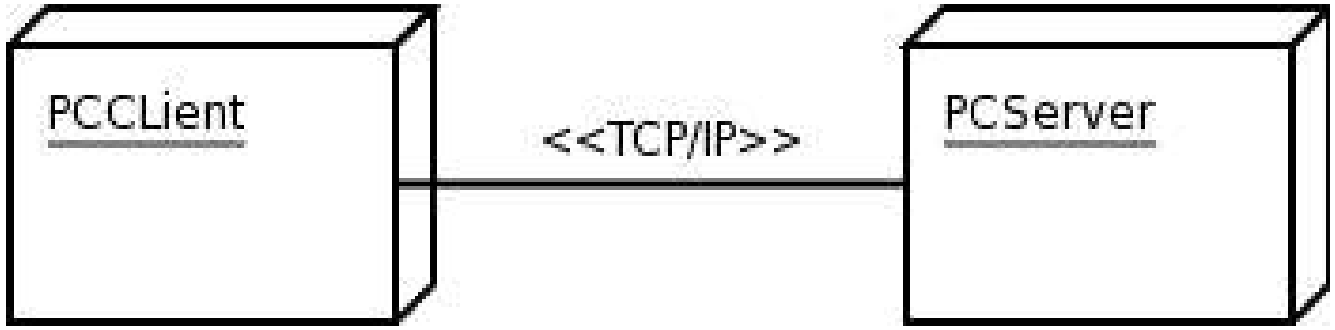
Diagrammi di deployment (1/7)

- I componenti di un diagramma dei componenti sono sempre tipi, mai istanze.
- Le istanze dei componenti vengono visualizzate nei diagrammi di deployment
- I ***diagrammi di deployment*** mostrano *nodi* connessi tra loro da *associazioni di comunicazione*



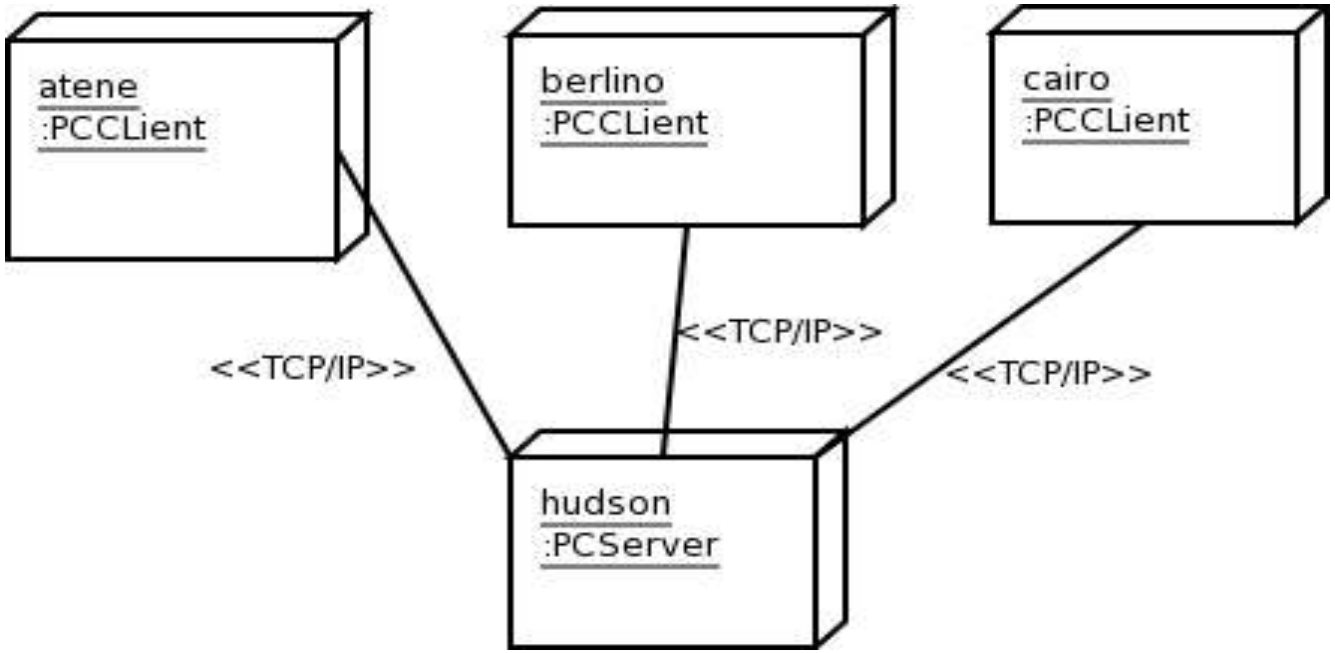
Diagrammi di deployment (2/7)

- I nodi rappresentano le risorse di elaborazione di un sistema
- I nodi sono
  - Tipicamente computer dotati di capacità di elaborazione e memoria
  - Ma anche sensori o periferiche
- Le associazioni di comunicazione possono essere stereotipate per mostrare la natura della comunicazione tra i due nodi



Diagrammi di deployment (3/7)

- I diagrammi di deployment possono essere utilizzati per rappresentare
  - tipi di nodi (come abbiamo visto nella slide precedente)
  - istanze di tipi di nodi, ad esempio:

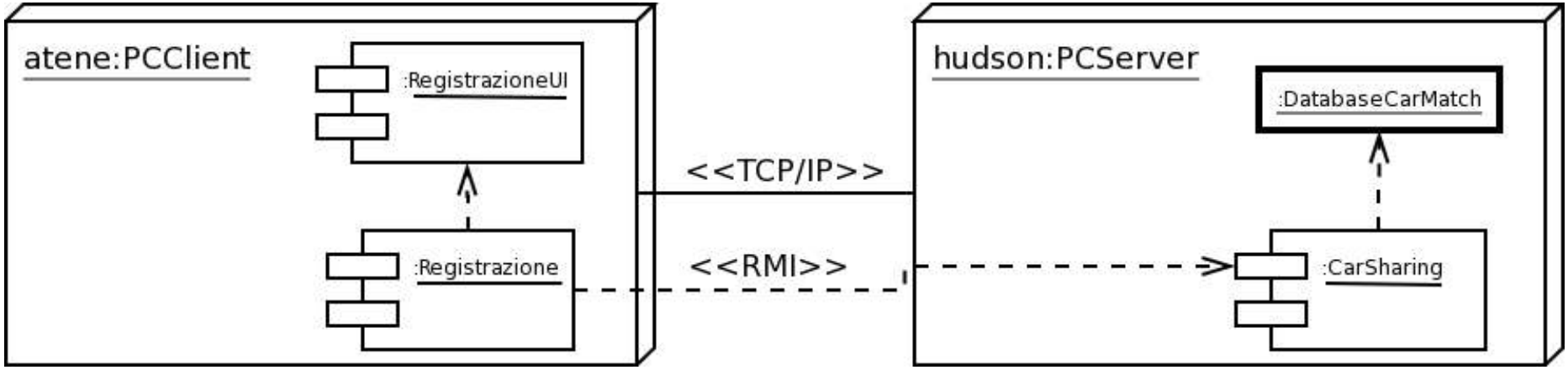


Diagrammi di deployment (4/7)

- Sebbene come abbiamo detto le istanze vengano mostrate solo nei *diagrammi di deployment*, è possibile produrre *diagrammi di deployment* senza nodi, solo con istanze di componenti
- In questo caso si parla di ***diagramma di deployment degenere***
- Ad esempio

Diagrammi di deployment (5/7)

- Il diagramma seguente
  - mostra i componenti della slide precedente su un nodo client
  - mostra un componente CarSharing che usa un oggetto attivo su un noo server

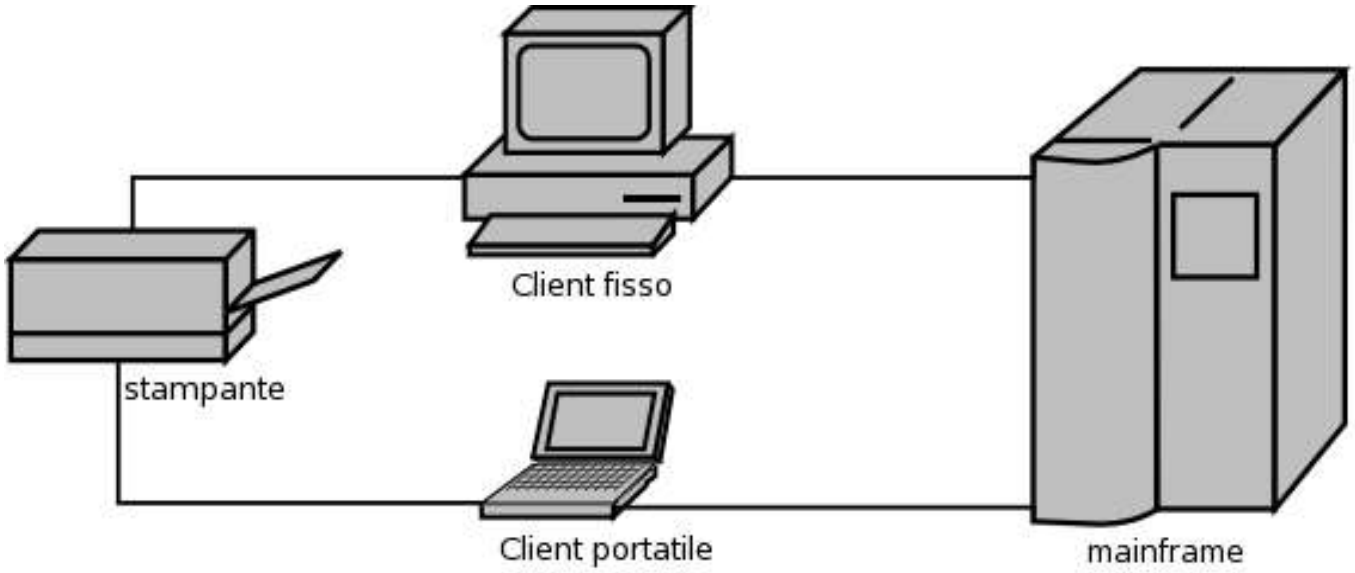


Diagrammi di deployment (6/7)

- Relativamente alla slide precedente
  - Si noti che l'associazione di comunicazione tra i nodi è stereotipata come TCP/IP,
  - mentre la dipendenza tra il componente Registrazione e il componente CarSharing è stereotipata come RMI, poiché utilizza la *Remote Method Invocation* di *Java* per chiamare le operazioni sulle istanze di CarSharer, Viaggio e Indirizzo

Diagrammi di deployment (7/7)

- I nodi possono essere mostrati tramite simboli stereotipati
- L'esempio riporta un *diagramma di deployment* completamente stereotipato



Riferimenti

- [UML 1.5] *OMG UML Specification* v. 1.5.
- [AN02] Arlow, Neustadt, *UML e Unified Process*, McGraw-Hill, 2002
- [BSL02] Bennett, Skelton, Lunn, *Introduzione a UML*, McGraw-Hill collana Schaum's, 2002
- I diagrammi sono stati realizzati con *Dia* ed *Eclipse*
  http://www.gnome.org/projects/dia/
  http://www.eclipseuml.com