UML2

Interfacce e Componenti

Andrea Polini

Laboratorio di Ingegneria del Software
Corso di Laurea in Informatica – L-31
Università di Camerino

Progettazione di sottosistemi

La Progettazione di sottosistemi prevede la suddivisione del sistema in più parti il più possibile indipendenti e con obiettivi ben definiti.

Le interazioni tra i sottosistemi sono definite attraverso precise interfacce che vengono definite sulla base della scomposizione scelta. I sottosistemi implementanto le interfacce mantenendo fede al loro scopo.

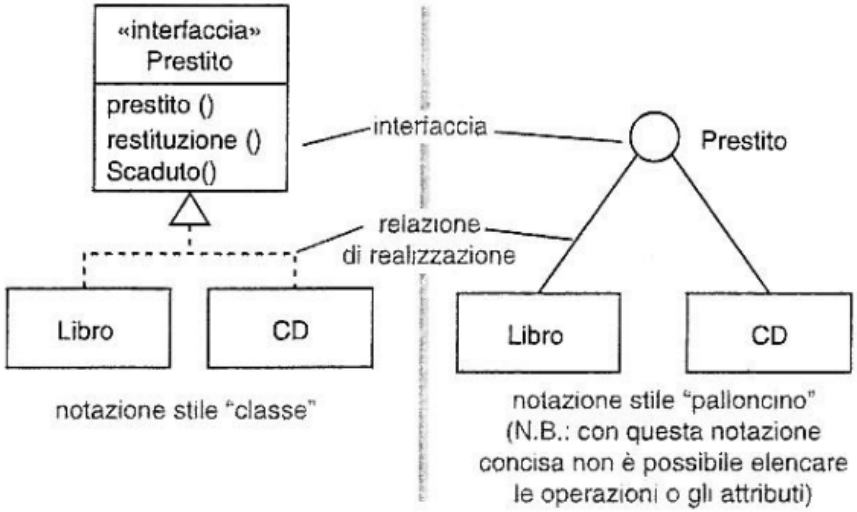
Interfacce

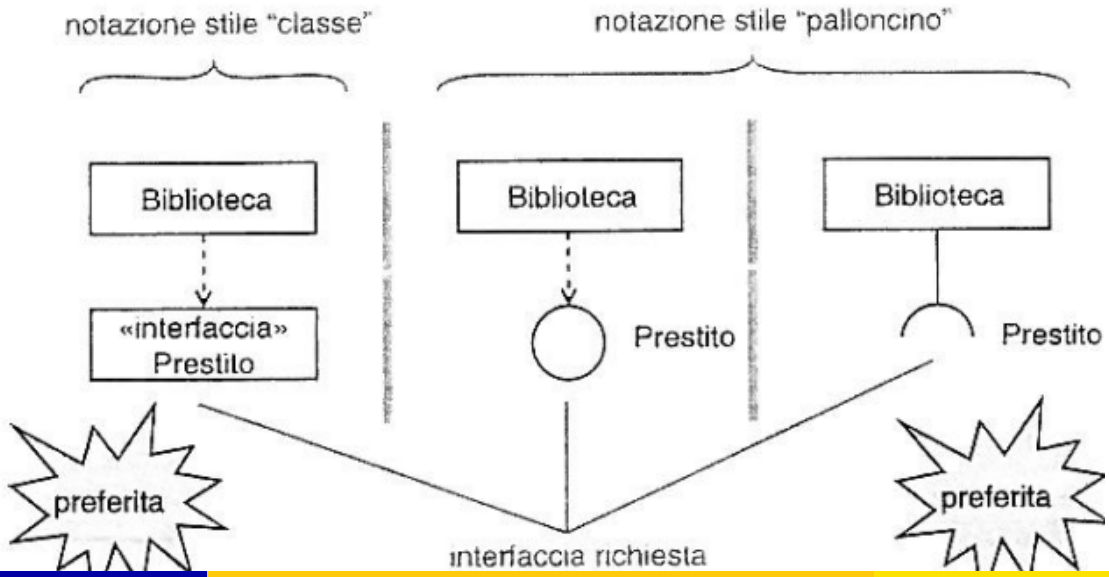
Le interfacce possono andare oltre la specifica della semplice segnatura. In generale la descrizione può essere arricchita al fine di caratterizzare semanticamente l'interfaccia in modo più preciso. A tal fine si può far uso di descrizioni testuali o si possono utilizzare linguaggi che forniscono costrutti per la logica (e.g. OCL)

- Operazione
- Attributi
- Associazioni
- Vincoli
- Stereotipo
- Valori Etichettati
- Specifica del Protocollo

Interfaccia specifica un contratto che l'implementazione si impegna a rispettare

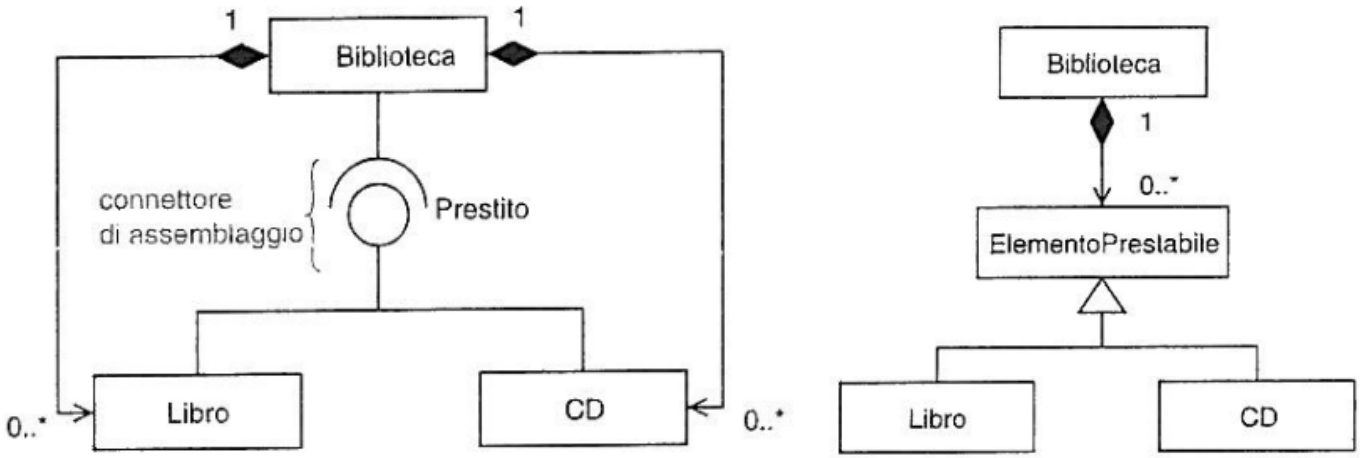
Simboli grafici





Ancora su Interfacce ed Ereditarietà...

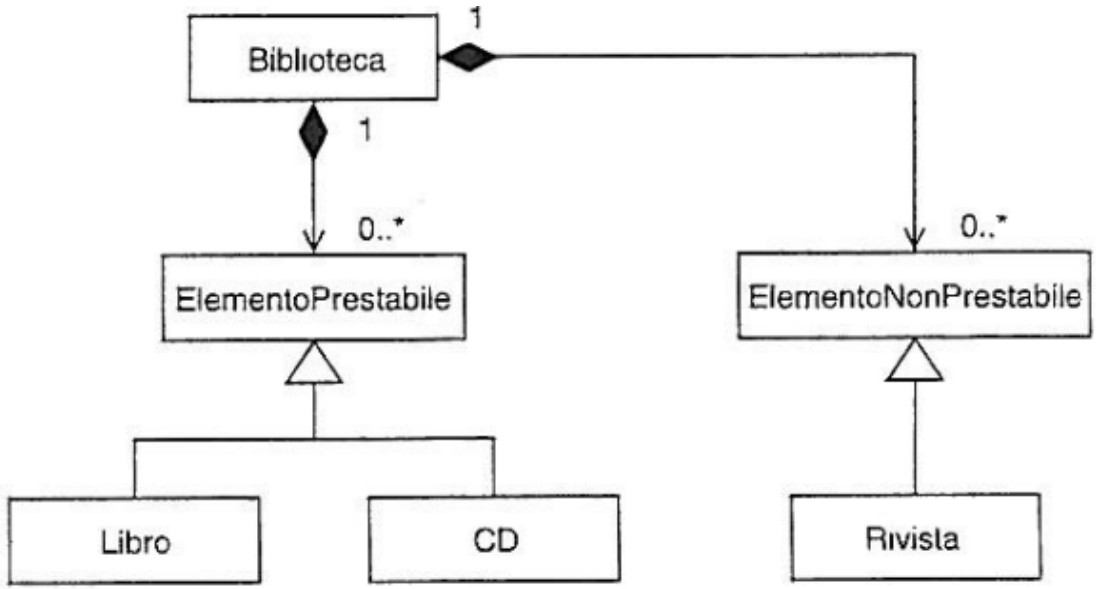
Principio di sostituibilità (Liskov) si applica parimenti a implementazioni di interfacce che a classi ereditate.
Consideriamo esempio di una frammento di progettazione per un sistema di gestione di una biblioteca:



Ad una prima vista potrebbero sembrare soluzioni grosso modo equivalenti...

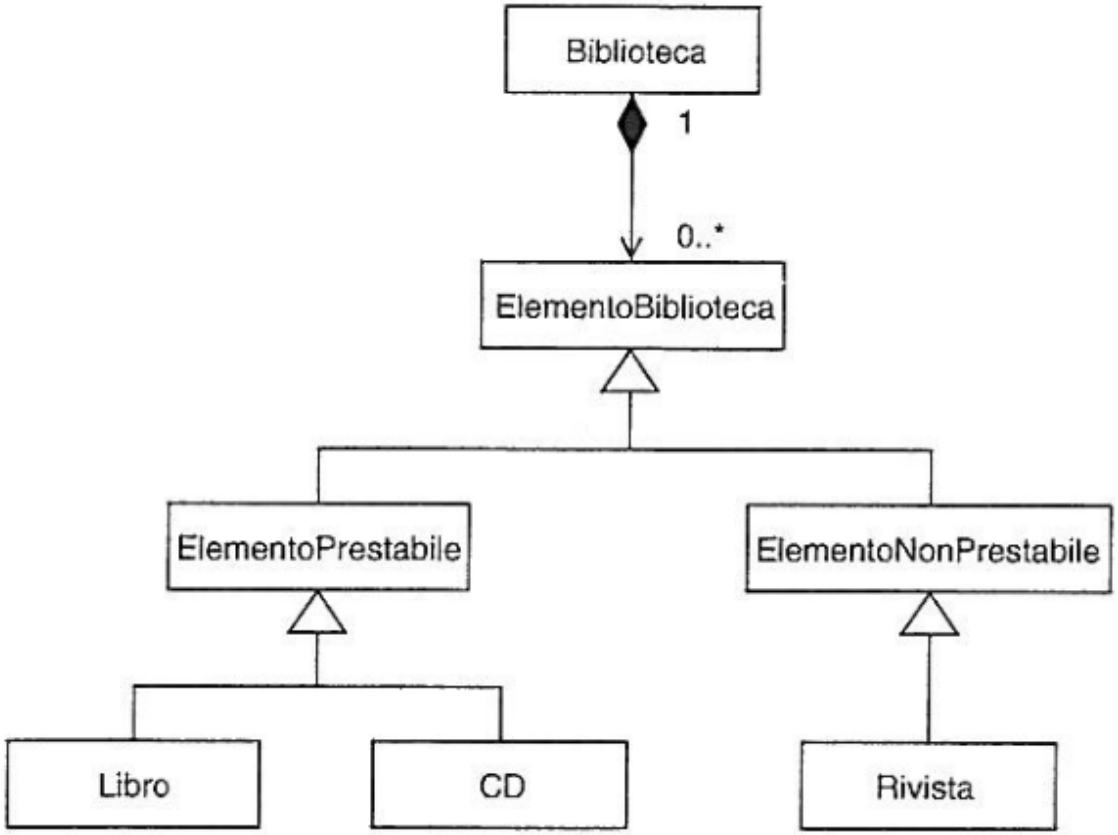
continua...

Si supponga di dover aggiungere una nuova tipologia di oggetto bibliografico quali le riviste con la sola possibilità di consultazione.



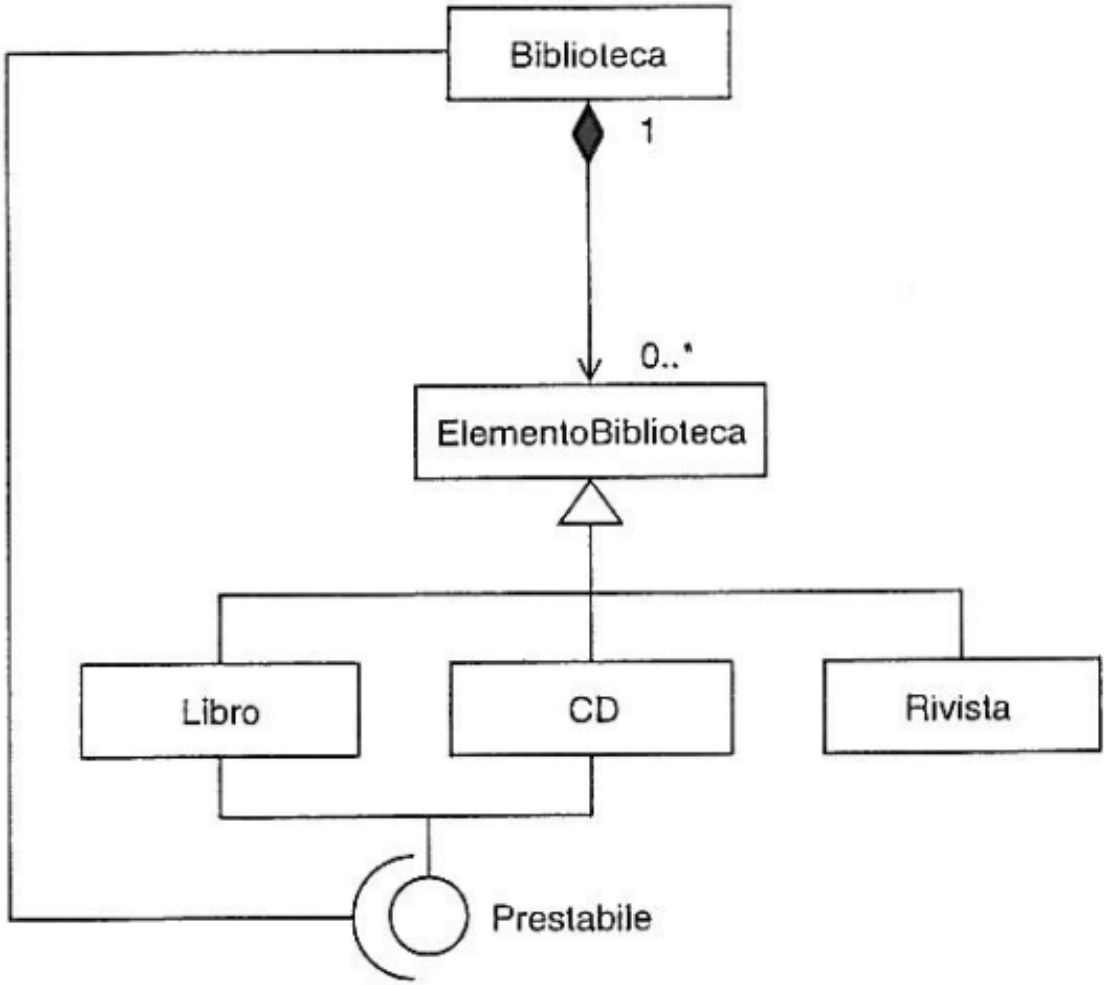
...continua...

Soluzione che scarica la biblioteca da compiti di “prestito” ma crea gerarchia di ereditarietà complessa



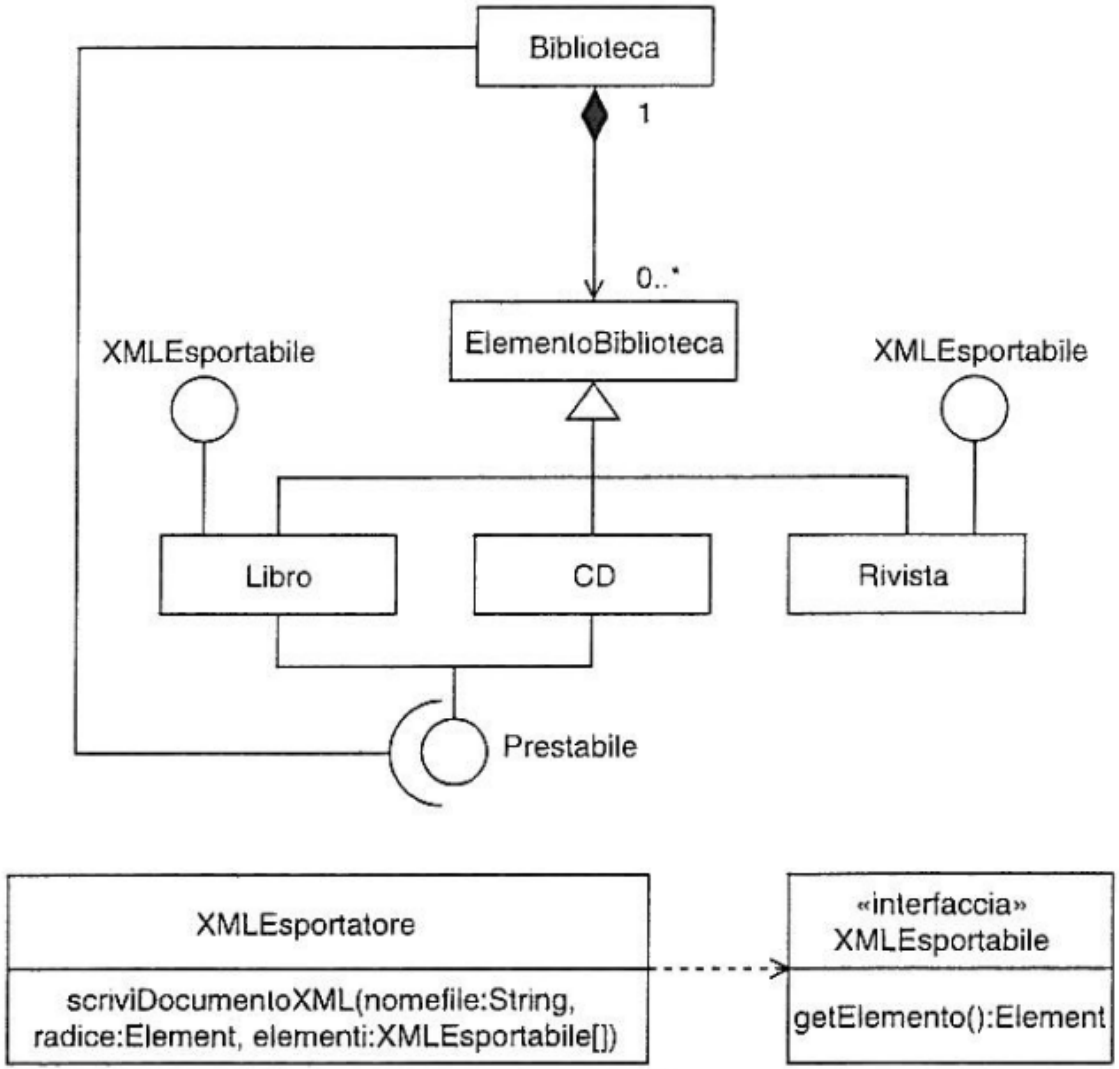
...continua...

Corretta assegnazione dei compiti tramite interfacce:



...continua

Si vuole aggiungere una funzionalità di esportazione delle informazioni



Porte

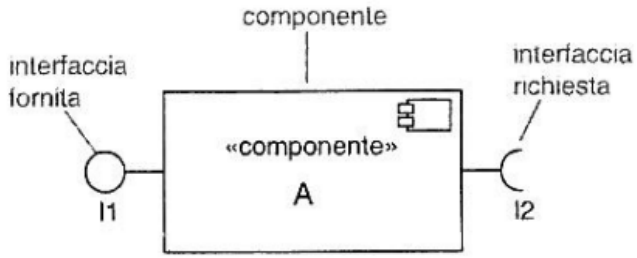
Costrutto che serve a raggruppare un insieme semanticamente coeso di interfacce fornite e richieste. Graficamente si ha un rettangolo sul bordo del classificatore. In particolare la posizione rispetto al bordo serve a rappresentare anche la visibilità della porta stessa.

Porte possono essere connesse quando sono complementari permettendo di risparmiare spazio nella visualizzazione della relazione

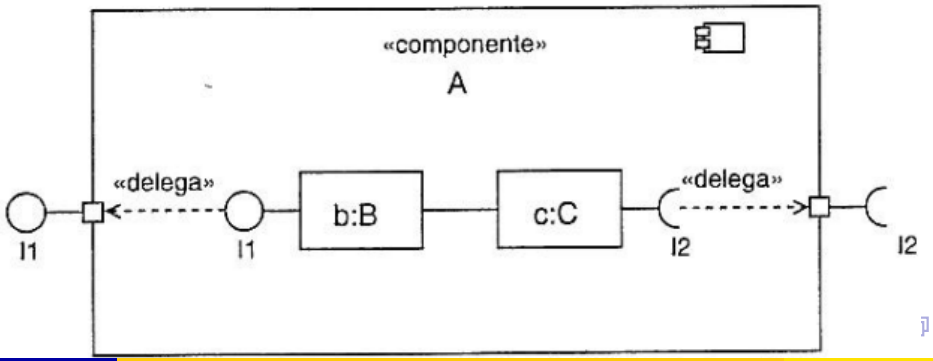
- porte pubbliche
- porte private/protette

Componenti

Componente identifica insieme di funzionalità molto coese e specifica precisi servizi offerti e richiesti (interfacce). In generale una granularità tale per cui l'elemento non potrebbe costituire sistema a se stante i.e. un componente è fatto per essere composto



È possibile mostrare l'interno di un componente e gli elementi responsabili dei servizi offerti e richiesti:



Sottosistemi

Rispetto ad un componente base un sottosistema tipicamente realizza un insieme di funzionalità coese che portano allo sviluppo di un sistema software che potrebbe prevedere utilizzi anche se non composto con altri sottosistemi, in generale comunque complessità molto più alta rispetto ad un componente base. . . ovviamente la distinzione non è netta.

Sottosistemi

Rispetto ad un componente base un sottosistema tipicamente realizza un insieme di funzionalità coese che portano allo sviluppo di un sistema software che potrebbe prevedere utilizzi anche se non composto con altri sottosistemi, in generale comunque complessità molto più alta rispetto ad un componente base. . . ovviamente la distinzione non è netta.

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Individuazione delle interfacce

È estremamente importante identificare le corrette interfacce e ridurre le dipendenze tra gli elementi di un sistema software complesso.
Tecniche applicabili:

- mettere in discussione ogni associazione
- mettere in discussione ogni messaggio inviato
- individuare gruppi di operazioni riutilizzabili da più classi
- individuare operazione ripetuto in più classi
- individuare gruppi di attributi ripetuti in più classi
- identificare classi del sistema che sembrano ricoprire ruoi simili o identici
- individuare possibilità di estensioni future del sistema stesso
- osservare ed analizzare dipendenze tra componenti

Vantaggi e Svantaggi

Le interfacce permettono di realizzare sistemi più flessibili a cui è relativamente semplice apportare modifiche. Interfacce tendono a ridurre ed a rendere più “deboli” le dipendenze tra classi/componenti, dove l’uso è di tipo black box

D’altro canto le interfacce tendono a complicare ed a riempire maggiormente il modello richiedendo tipicamente l’inclusione di più classi nel modello.