Università
Ca' Foscari
Venezia

Progettazione del software

Università Ca' Foscari Venezia

Progettazione del Software

- Software Design = derivare soluzioni che soddisfino il documento dei requisiti
- Fasi del processo di progettazione
- Strategie di progettazione:
  - approccio funzionale
  - approccio orientato ad oggetti
- Attributi di qualità della progettazione software (coesione e accoppiamento)

IEEE 610.12.1990

- **design phase --** The period in the software life cycle during which definitions for architecture, software components, interfaces, and data are created, documented, and verified to satisfy requirements.
- **design** -- (1). The process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements. (2). The results of the design process.
- **design analysis** -- (1). The evaluation of a design to determine correctness with respect to stated requirements, conformance to design standards, system efficiency, and other criteria. (2). The evaluation of alternative design approaches. [ANSI/IEEE Std 830-1984]

- **architectural design** -- In system/software system engineering, (1) The process of defining a collection of hardware and software components and their interfaces to establish a framework for developing a system/ software system. (2) The result of the architectural design process. *Also called high-level design, internal specification, preliminary design, system design, and top-level design.*
- **detailed design** -- (1). The process of refining and expanding software architectural designs to more detailed descriptions of the processing logic, data structures, and data definitions. This continues until the design is sufficiently complete to be implemented. (2). The result of the detailed design process. *Also called design, low-level design, module design, and program design*. [ANSI/IEEE Std 610.12-1990]

Università Ca' Foscari Venezia

Fasi della progettazione

- Comprensione del problema
  - Guardare al problema da differenti angolature
- Identificare una o più soluzioni
  - Valutare le soluzioni possibili e scegliere la più appropriata rispetto all'esperienza del progettista e dalle risorse disponibili
- Descrivere astrazioni delle soluzioni
  - Usare notazioni grafiche, formali o d'altro tipo per descrivere le componenti del progetto
- Ripetere il processo per ogni astrazione identificata, finché la progettazione non è espressa in termini primitivi

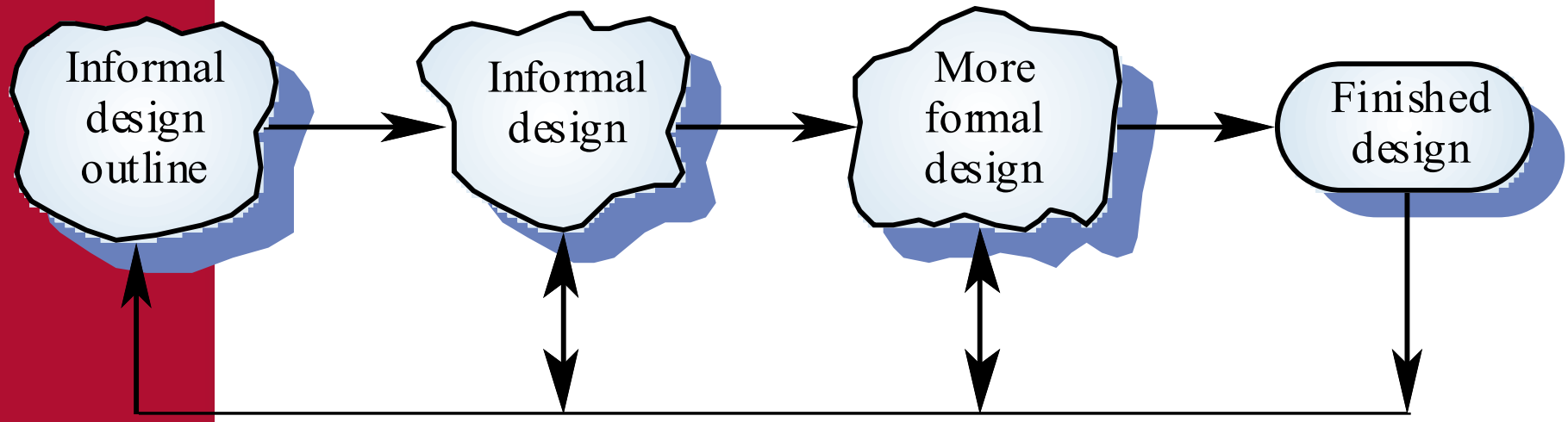
Università Ca' Foscari Venezia

Il processo della progettazione

- Un qualsiasi progetto può essere modellato da un grafo orientato, a partire da entità poste in relazione e dai loro attributi
- Il sistema deve essere descritto a diversi livelli di astrazione
- La progettazione si attua in tappe difficilmente distinguibili; ciò nonostante la strutturazione è utile.

Università
Ca' Foscari
Venezia

Dall'informale al formale



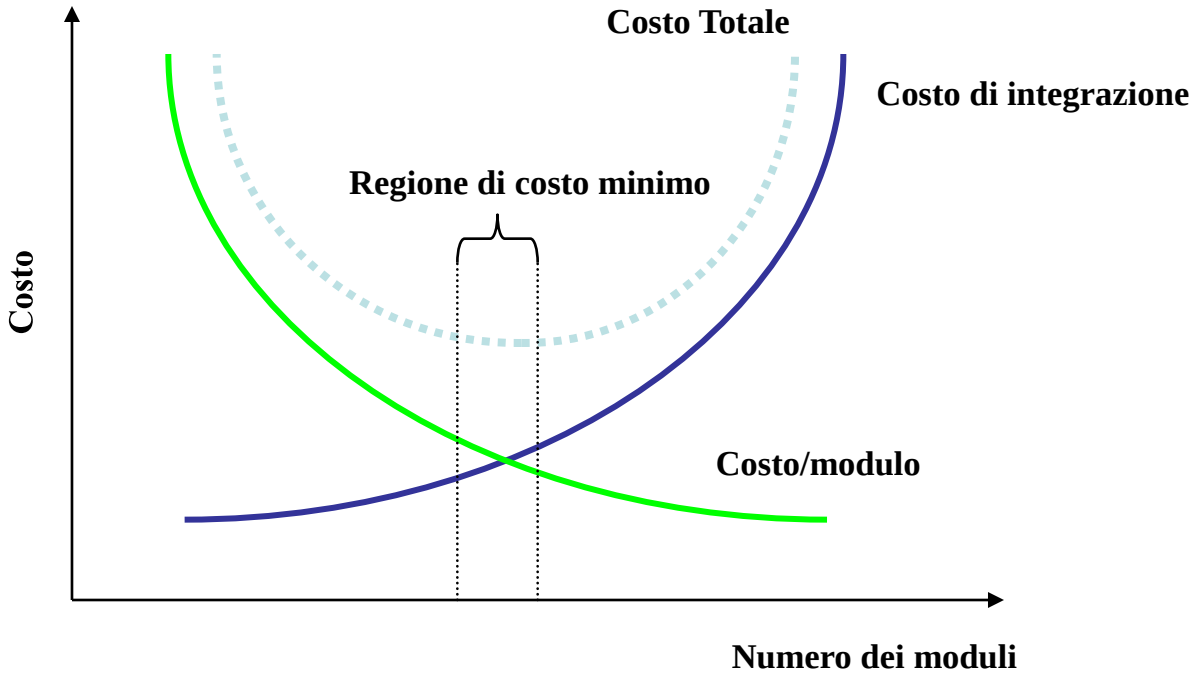
Università Ca' Foscari Venezia

Concetti di base

- Astrazione: procedere per livelli di dettaglio
  - astrazione procedurale
  - astrazione dei dati
  - astrazione del controllo
- Raffinamento:
  - decomposizione
  - strutturazione
  - elaborazione
- Modularità:
  - Vantaggi dell'approccio "divide et impera": osservando il costo C relativo alla progettazione di sistemi software, si ha che $C(p_1+p_2) > C(p_1)+C(p_2)$.
  - Il prezzo da pagare: i costi di integrazione

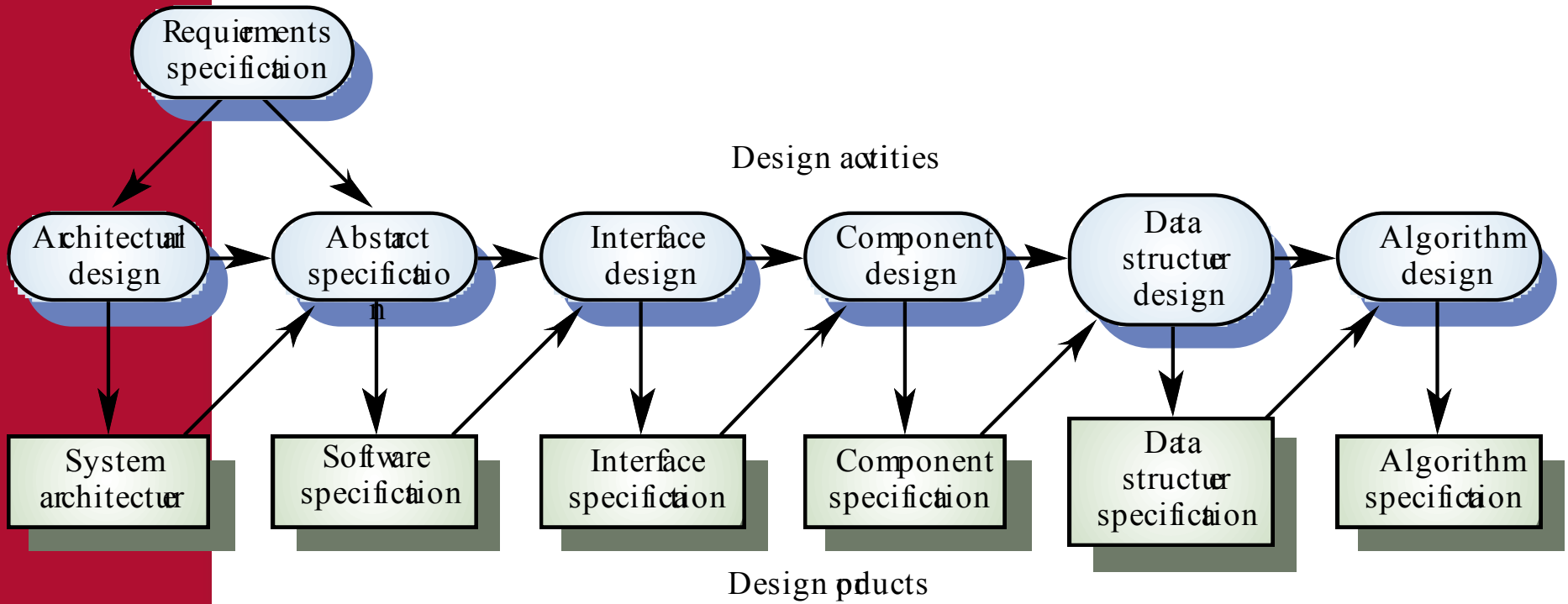
Università Ca' Foscari Venezia

Modularità e costo del software



Università Ca' Foscari Venezia

Fasi della progettazione

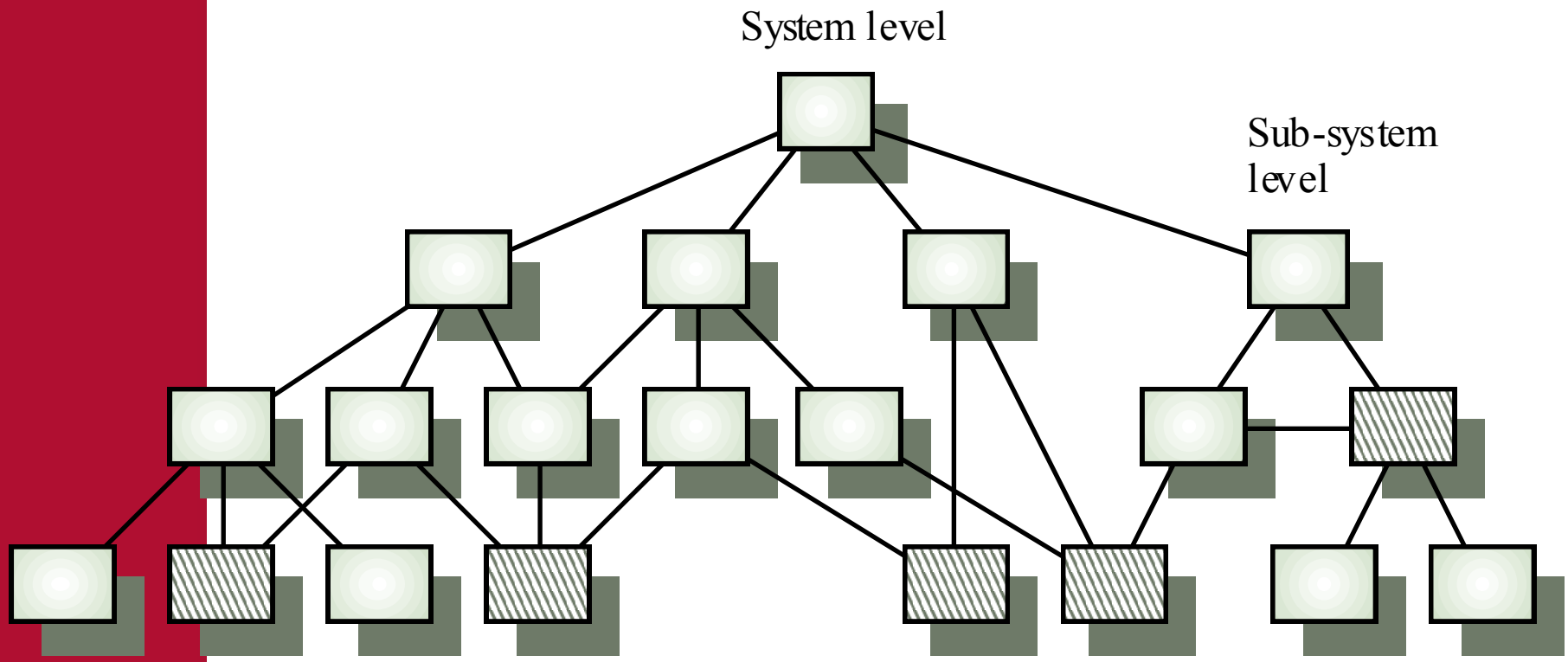


Fasi della progettazione

• *Architectural design*	Identifica i sottosistemi
• *Abstract specification*	Progetta i sottosistemi
• *Interface design*	Progetta le interfacce dei sottosistemi
• *Component design*	Decomponi i sottosistemi
• *Data structure design*	Progetta le strutture dati per gestire i dati del problema
• *Algorithm design*	Progetta algoritmi per le funzionalità del problema

Università
Ca' Foscari
Venezia

Struttura gerarchica di progetto



Università Ca' Foscari Venezia

Progettazione top-down

- In teoria, progettazione top-down significa partire dalle componenti più generali nella gerarchia e scendere nella gerarchia lavorando livello per livello
- In pratica, la progettazione di sistemi di grandi dimensioni non è mai veramente top-down.
  Alcune componenti sono progettate prima di altre.
  I progettisti riusano esperienza (e componenti) durante il processo di progettazione

Metodi di progettazione

- I metodi strutturati sono insiemi di notazioni per esprimere un progetto software e linee guida per creare un progetto
- Tra i metodi più conosciuti: Structured Design (Yourdon), e JSD (Jackson Method)
- Il linguaggio "standard" per la progettazione: UML
- I metodi strutturati possono essere supportati da tools CASE
- Offrono linee-guida piuttosto che "metodi" in senso matematico: progettisti diversi possono creare progetti di sistema alquanto diversi
- Non sono d'aiuto nella parte iniziale, quella creativa, della progettazione. Ma sono utili al progettista per strutturare e documentare meglio il lavoro.

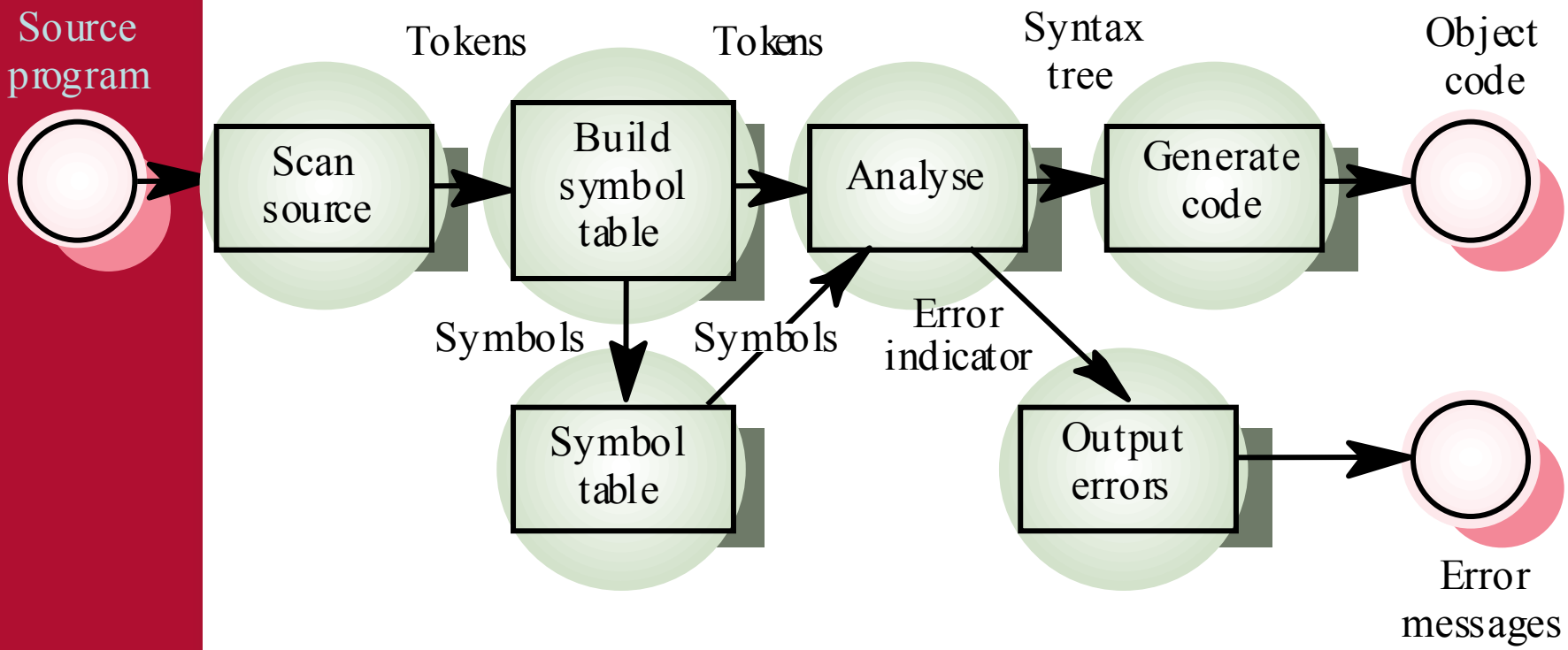
Università Ca' Foscari Venezia

Strategie di progettazione

- Progettazione Funzionale
  - Lo stato del sistema è centralizzato e condiviso tra funzioni che operano su quello stato
- Progettazione Object-Oriented
  - Il sistema è visto come collezione di oggetti che interagiscono.
    Il sistema è de-centralizzato ed ogni oggetto gestisce il proprio stato. Gli oggetti possono essere istanze di una classe e comunicano scambiando attraverso i propri metodi

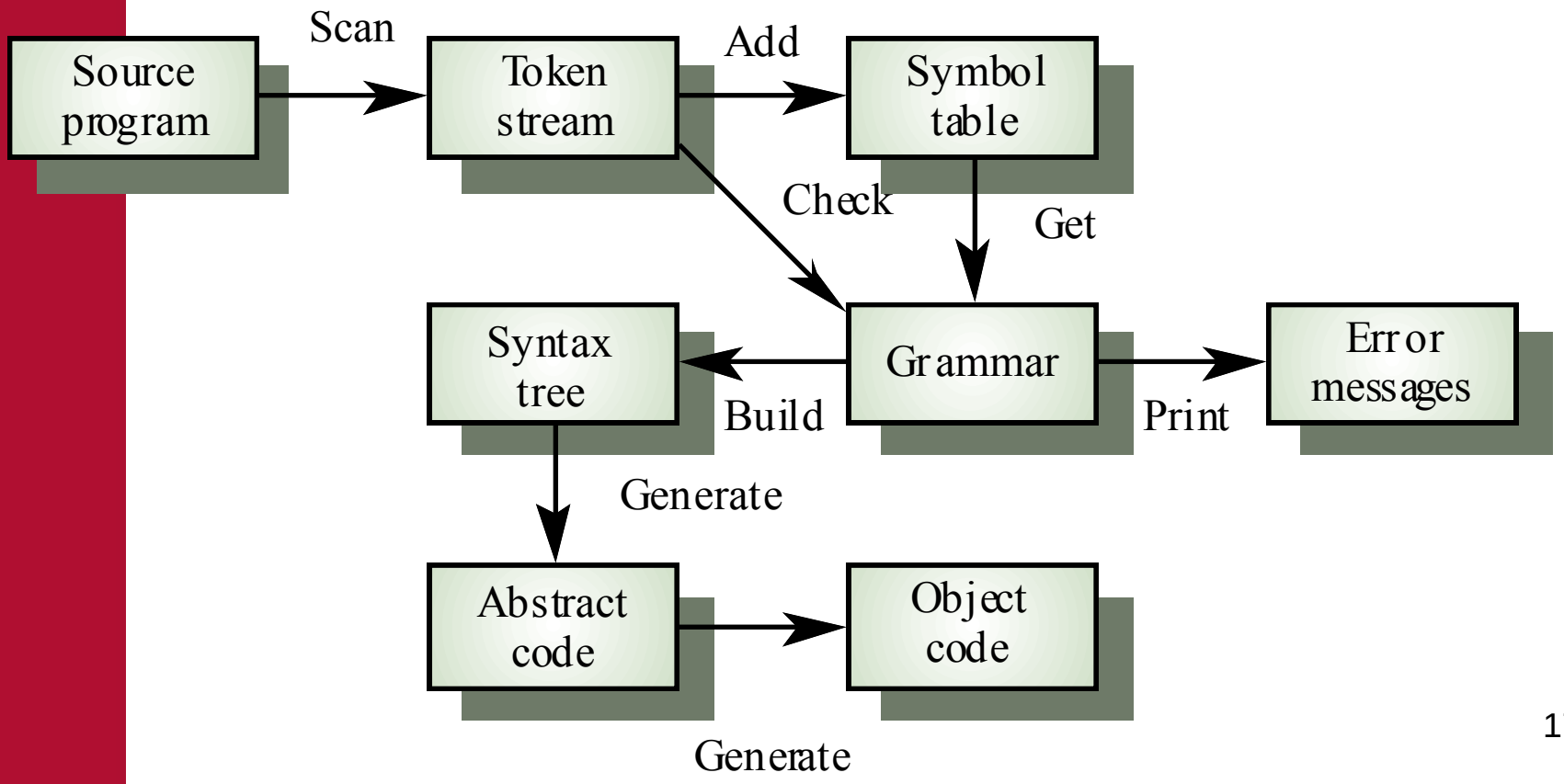
Università Ca' Foscari Venezia

Visione funzionale di un compilatore



Università Ca' Foscari Venezia

Visione object-oriented di un compilatore



Progettazione mista

- C'è una complementarietà tra approccio funzionale e approccio object-oriented
- Di volta in volta un buon ingegnere del software dovrebbe scegliere l'approccio più appropriato per il sottosistema che sta progettando.

Qualità della progettazione

- La qualità di un progetto è difficile da stabilire. Dipende da specifiche priorità
- Un "buon progetto" deve essere il più efficiente, il più economico, il più mantenibile, il più affidabile, ecc.
- Noi sottolineeremo gli attributi legati alla mantenibilità del progetto: coesione, accoppiamento, comprensibilità, adattabilità
- Le stesse caratteristiche di qualità si applicano sia alla progettazione funzionale che a quella orientata ad oggetti

Coesione

- Misura di quanto le parti di una componente "stanno bene assieme"
- Una componente dovrebbe implementare una singola entità logica o una singola funzione
- La coesione è un attributo importante in quanto, quando si dovesse effettuare un cambiamento al sistema, permette di rendere il cambiamento *locale* ad una singola componente
- Si possono individuare livelli diversi di coesione

Coesione

- Proprietà interna al singolo componente
  - Funzionalità "vicine" devono stare nello stesso componente
  - Vicinanza per tipologia, algoritmi, dati in ingresso e in uscita
- Vantaggi di un alto grado di coesione
  - Vantaggi rispetto al riuso e alla manutenibilità
  - Riduce l'interazione fra componenti
  - Migliore comprensione dell'architettura del sistema

Università Ca' Foscari Venezia

Tipologie e livelli di coesione

- Coesione accidentale (debole)
  - Parti di una componente sono semplicemente raggruppate assieme
- Associazione logica (debole)
  - Vengono raggruppate componenti che svolgono azioni simili
- Coesione temporale (debole)
  - Vengono raggruppate componenti che sono attivate contemporaneamente
- Coesione procedurale (debole)
  - Gli elementi in una componente costituiscono assieme un'unica sequenza di controllo

Università Ca' Foscari Venezia

Tipologie e livelli di coesione

- Coesione di comunicazione (media)
  - Tutti gli elementi di una componente operano su di uno stesso input o producono lo stesso output
- Coesione sequenziale (media)
  - L'output di una parte di una componente è l'input di un'altra parte
- Coesione funzionale (forte)
  - Ogni parte di una componente è necessaria solo per l'esecuzione di una singola funzione di quella componente
- Coesione d'oggetto (forte)
  - Ogni operazione offre funzionalità che permettono di modificare o ispezionare attributi dell'oggetto

Coesione come attributo di progetto in progettazione OO

- Se si ereditano attributi da una superclasse si diminuisce la coesione:
- Per comprendere una classe, bisogna esaminare sia tutte le sue superclassi che le componenti della classe

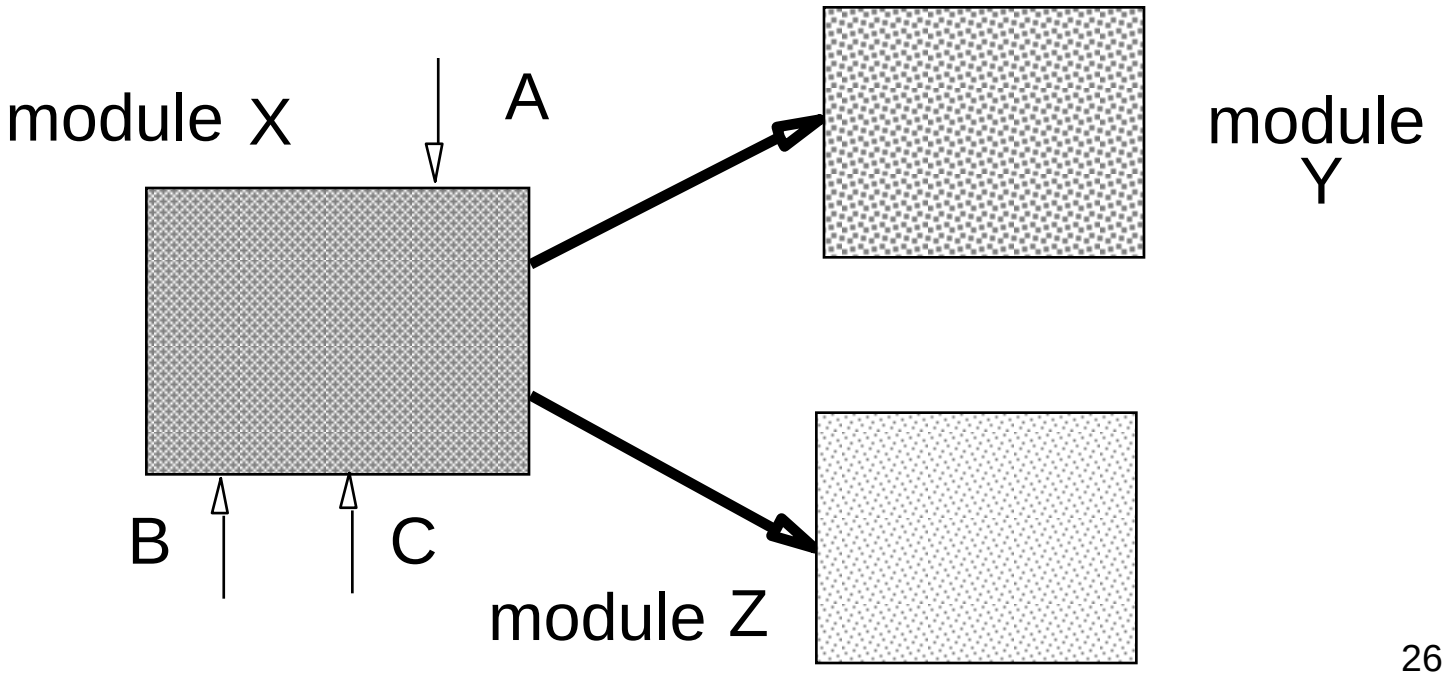
Coupling (accoppiamento)

- Misura la strettezza di connessione tra le componenti di un sistema: quanto le componenti "si usano" tra di loro.
- Loose coupling (accoppiamento lasco) implica che cambiamenti di una componente non hanno forti effetti sul comportamento delle altre.
- La presenza di variabili condivise o lo scambio di informazione di controllo porta ad accoppiamento stretto (tight coupling)
- L'accoppiamento lasco può essere ottenuto decentralizzando gli stati e realizzando la comunicazione con passaggio di parametri

Università Ca' Foscari Venezia

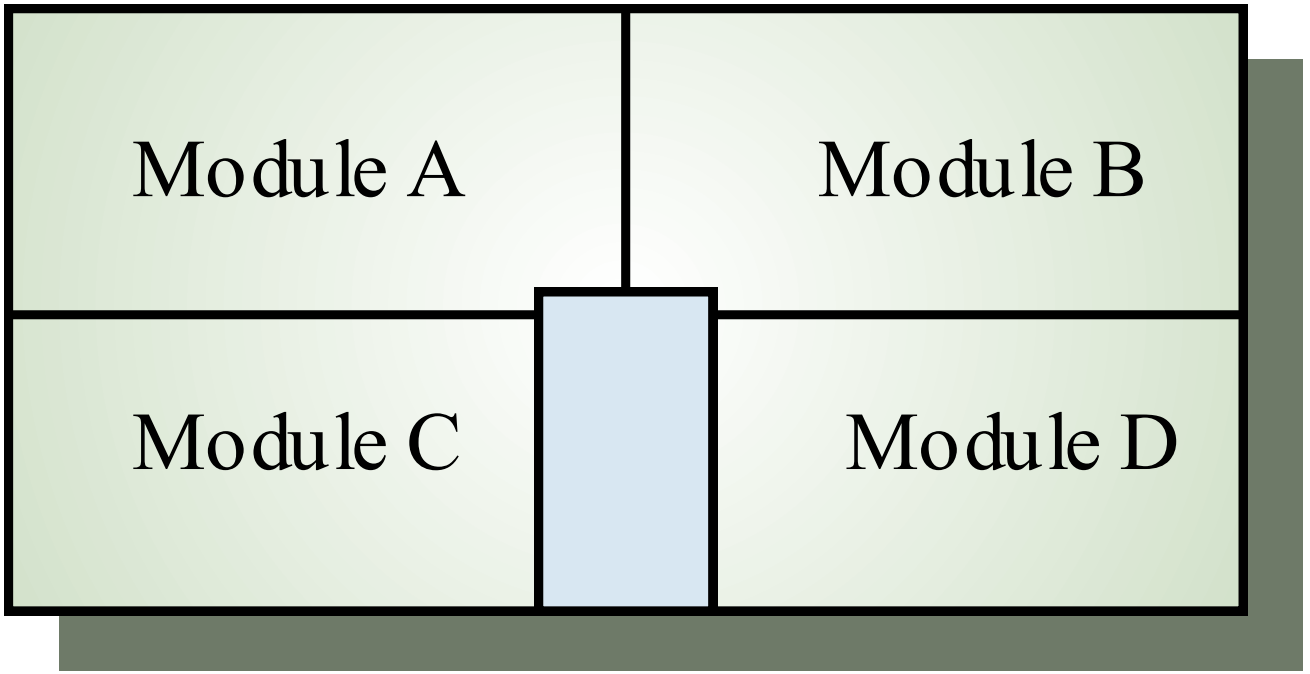
Coupling

- Proprietà fra componenti: quanto i componenti si usano fra di loro (M = numero dei moduli/componenti)
  - U = M x M massimo accoppiamento
  - U = ∅ minimo accoppiamento

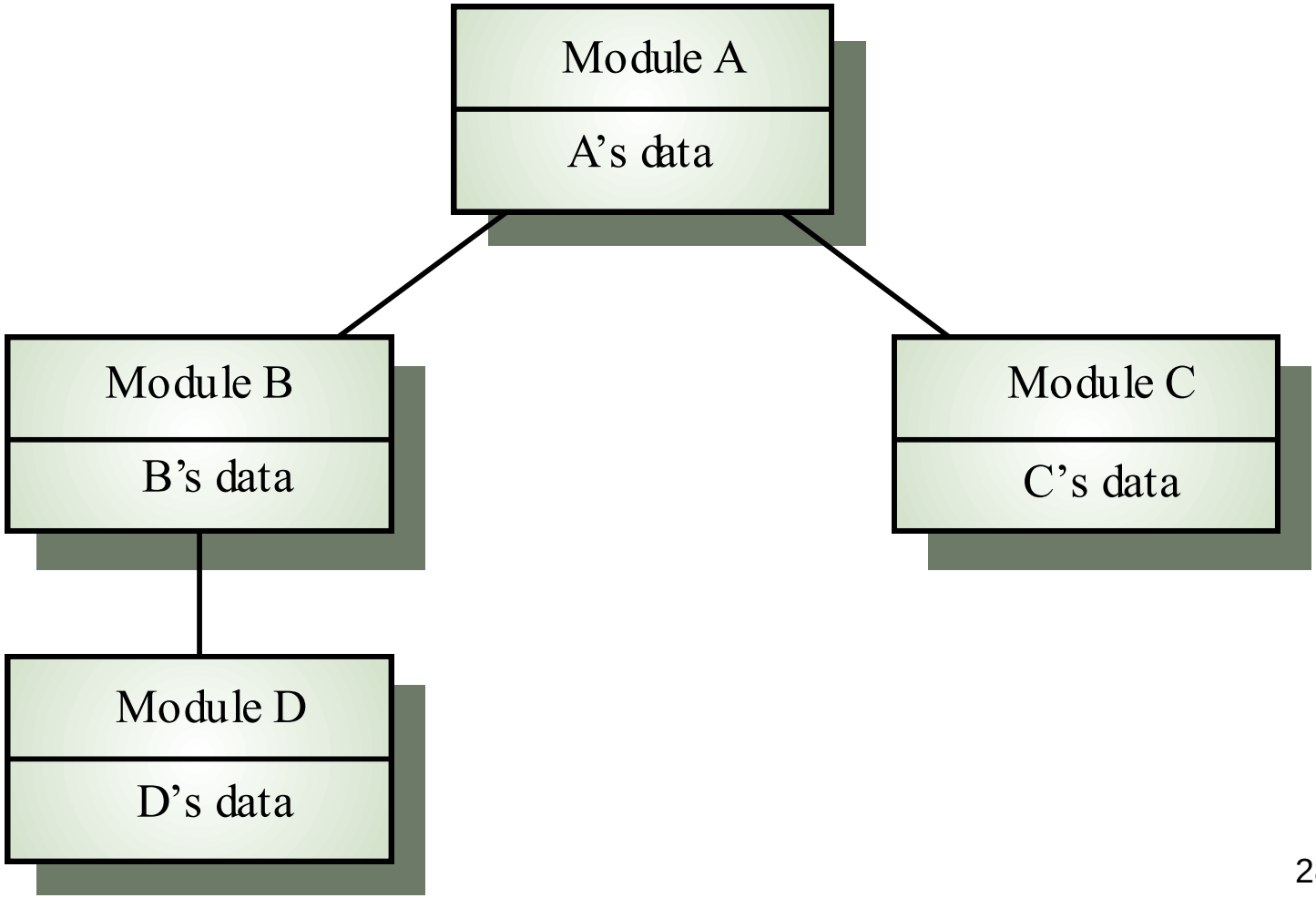


Università
Ca' Foscari
Venezia

Accoppiamento stretto



Accoppiamento lasco



Università Ca' Foscari Venezia

Accoppiamento ed ereditarietà

- I sistemi orientati ad oggetti sono “loosely coupled”
  - non condividono uno “stato”
  - gli oggetti comunicano passando messaggi attraverso i metodi
- Tuttavia una classe è accoppiata con la sua superclasse.
  - I cambiamenti effettuati su una classe si propagano a tutte le sottoclassi

Comprensibilità

- Legata a diverse caratteristiche delle componenti

  - *Naming*. I nomi usati sono significativi?
  - *Documentazione* Il progetto è ben documentato?
  - *Complessità* Si usano algoritmi complessi?

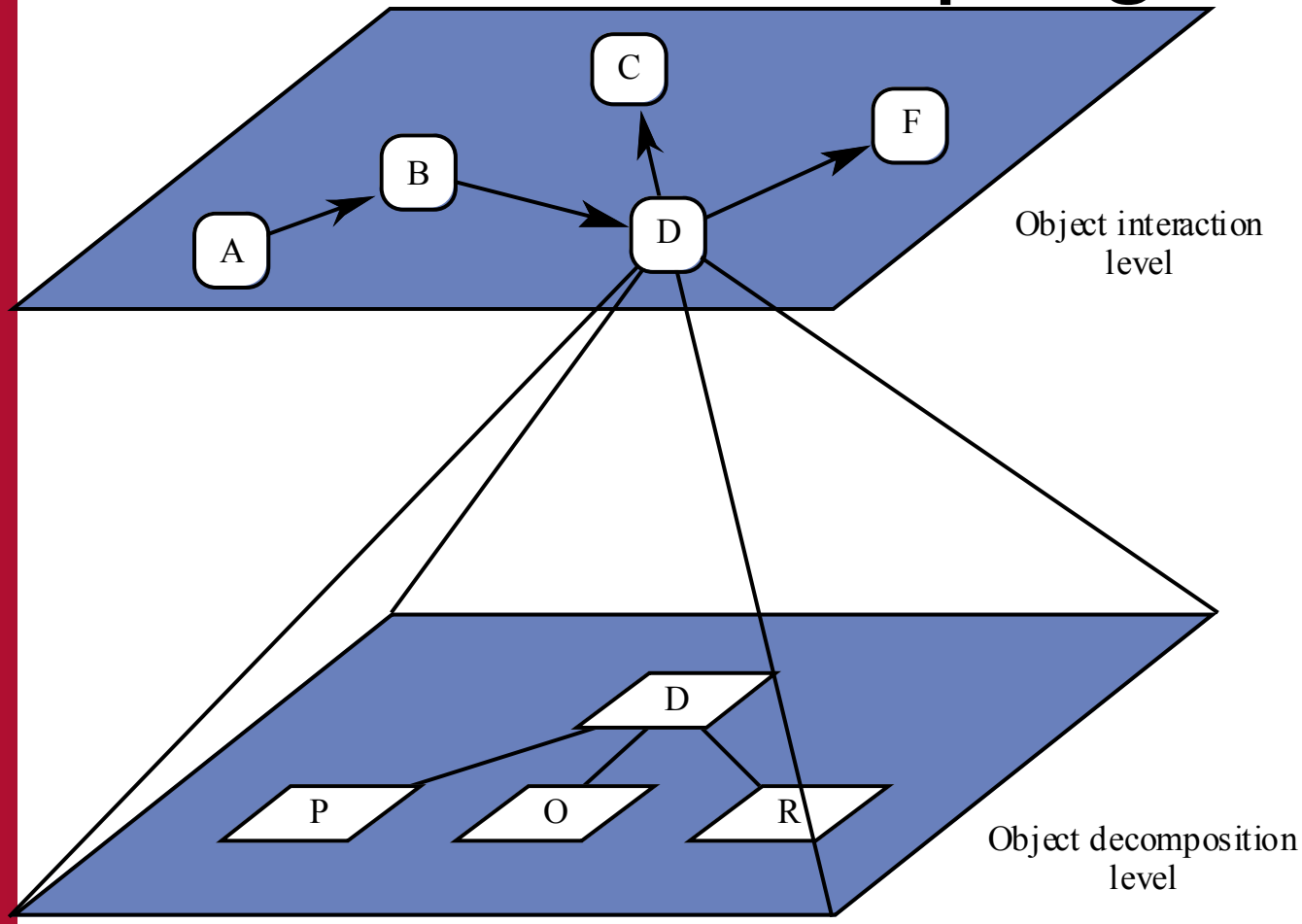
- Un'elevata complessità significa molte relazioni tra diverse parti del sistema, quindi scarsa comprensibilità
- Metriche di qualità della progettazione: misurano la complessità.

Adattabilità

- Un progetto è adattabile se:
  - Le sue componenti sono accoppiate lascamente
  - E' ben documentato e la documentazione è aggiornata
  - C'è corrispondenza stretta tra livelli della progettazione
  - Ogni componente è una entità "self-contained"
- Per adattare un progetto deve essere possibile tracciare i legami tra componenti del progetto, per poter analizzare le conseguenze di ogni cambiamento.

Università
Ca' Foscari
Venezia

Tracciabilità del progetto



Università Ca' Foscari Venezia

Adattabilità ed Ereditarietà

- L'ereditarietà è un elemento che induce adattabilità: le componenti possono essere modificate senza cambiamenti derivando una sotto-classe e modificando solo questa classe derivata
- Tuttavia, quando la profondità dell'ereditarietà aumenta, adattare un progetto diventa sempre più complesso. Il progetto richiede di essere rivisto e ristrutturato.