

Algoritmi e Programmi

Algoritmi e Strutture Dati

A.A. 18/19

Informatica

Università degli Studi di Bari "Aldo Moro"

Nicola Di Mauro

Sommario

- Sviluppo del software e qualità dei programmi
- Principi di progettazione e tecniche di programmazione
- Ruolo delle tecniche di astrazione nel progetto di programmi

Prerequisiti

- Principi della programmazione strutturata e un linguaggio di programmazione algoritmica
- Tipizzazione forte e principi della programmazione modulare
- Principi dell'astrazione funzionale
 - Funzioni, Procedure
- e meccanismi del passaggio dei parametri

Obiettivi

- Programmazione modulare (progettazione orientata alle funzioni)
 - Di solito top-down e basata sulla decomposizione di problemi in sotto-problemi
- Essere in grado di applicare una metodologia di progetto alternativa passando attraverso una tecnica formale basata sulla astrazione dei dati

Ciclo di Sviluppo del Software

- Studio di Fattibilità
 - Scopo: valutare costi e benefici del sistema da costruire
 - Analisi di tempi e modi di attuazione, valutazione delle alternative possibili per lo sviluppo del sistema e delle risorse necessarie
- Raccolta e Analisi dei Requisiti
 - Scopi:
 - Definire il problema
 - Specificare l'ambiente di sviluppo (sia HW che SW)
 - Risultato: un documento di analisi
 - Specifica HW e SW scelto, la struttura concettuale dei dati da elaborare, le caratteristiche generali delle operazioni da fare

Ciclo di Sviluppo del Software /2

- Progettazione
 - Scopi:
 - individuare la soluzione “informatica” del problema
 - definire l’architettura del programma
 - specificarne e organizzarne la struttura
 - specificare le funzioni delle componenti individuate
 - scegliere le strutture di rappresentazione degli oggetti
 - redigere il programma secondo il linguaggio e l’ambiente operativo scelto
 - Risultato: il codice

Ciclo di Sviluppo del Software /3

- Verifica
 - Scopo: analizzare sia la documentazione prodotta che il programma realizzato
 - Prove di correttezza sintattiche e logiche (test empirici o prove formali)
- Manutenzione
 - Scopi:
 - Controllare che il programma, durante il periodo di esercizio, produca i risultati attesi
 - Aggiornare, ove necessario, il programma

Qualità dei Programmi

- La fase di progettazione deve fornire un'analisi delle qualità che il programma deve possedere
 - Esterne
 - Si riferiscono a caratteristiche evidenziabili dal solo funzionamento del programma durante la fase di esercizio
 - Interne
 - Si riferiscono alle caratteristiche analizzabili e valutabili da esperti, attraverso uno studio delle scelte tecniche adottate

Qualità Esterne

- Correttezza
 - Capacità di eseguire precisamente i compiti individuati durante l'analisi dei requisiti
- Efficienza
 - Capacità di utilizzare in modo razionale ed economico le risorse di calcolo
- Robustezza
 - Capacità di funzionare in modo soddisfacente in condizioni limite o anomale rispetto a quelle previste in fase di analisi dei requisiti
- Usabilità
 - Capacità di consentire un'interazione semplice, naturale ed efficace con l'utente finale

Qualità Interne

- Riusabilità
 - Capacità di essere riutilizzato, in tutto o in parte, per applicazioni diverse rispetto a quella per la quale è stato prodotto
- Modularità
 - Grado di organizzazione interna del programma
 - Strutturazione delle singole parti, della funzionalità e del modo in cui cooperano per l'obiettivo generale
- Estensibilità
 - Capacità di adattarsi facilmente a modifiche nei requisiti
- Portabilità e Compatibilità
 - Facilità di trasferire il SW prodotto in ambiti diversi
- Leggibilità
 - Capacità del codice di essere autoesplicante
- Bontà della documentazione
 - Completezza ed efficacia dei documenti annessi

Fase di Progettazione

- Principi
 - Linee guida per produrre software di qualità
- Tecniche
 - Metodi per produrre programmi coerenti con i principi
- Strumenti
 - Aiuti di varia natura utilizzati nella progettazione
 - Esempio: i linguaggi di programmazione

Principi di Progettazione

- Un ruolo fondamentale è giocato dalla ASTRAZIONE
 - Dà luogo a tecniche definite di astrazione
 - Programmazione strutturata
 - Modularizzazione
 - Astrazione dati
 - ...
- che hanno segnato la programmazione negli ultimi 30 anni

Astrazione (nei sistemi software)

- Una descrizione esemplificata, o specifica, di un sistema
 - Pone enfasi su alcuni dettagli o proprietà
 - Ne elimina altri
- Fornisce un buon modo di controllare la complessità e garantire la continuità di sistemi software complessi
 - Test, Manutenzione, Estensione
- Corrisponde alla tecnica chiamata in altri campi modellazione analitica
 - La costruzione di un modello parte dalle osservazioni che sono subito seguite dalla formulazione di ipotesi circa i principi o assiomi che le spiegano
 - Tali assiomi sono usati per costruire il modello
 - Le variabili o i parametri del modello possono essere derivati dagli assiomi o stimati dalle osservazioni
 - Il modello è usato per fare previsioni

Astrazione (nei sistemi software) /2

- I requisiti o le funzionalità del sistema giocano il ruolo di osservazioni che devono essere “spiegate”
- Il processo di astrazione prevede il decidere:
 - Quali caratteristiche sono rilevanti
 - Quali parametri andrebbero inclusi
 - Quale formalismo descrittivo andrebbe adottato
 - Come validare il modello
- Come in molti altri campi, spesso si definiscono gerarchie di modelli
 - I livelli più bassi forniscono descrizioni più dettagliate di quelle che appaiono nei livelli più alti

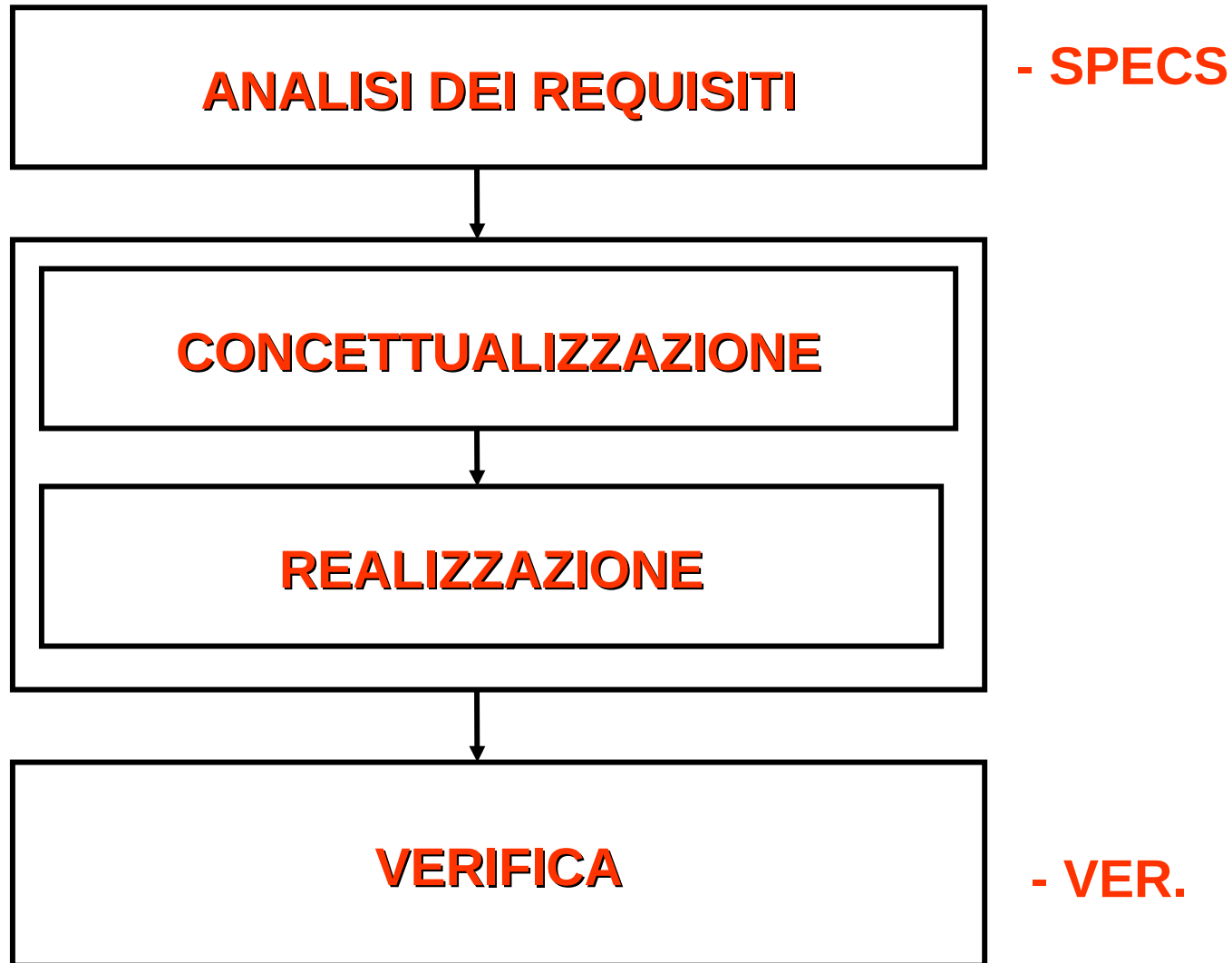
Astrazione (nei sistemi software) /3

- In programmazione ci riferiamo alla descrizione astratta fornita da un modello come alla specifica e al livello più basso nella gerarchia di modelli come alla realizzazione
 - Il procedimento di verificare se la realizzazione è coerente con la specifica è chiamato verifica
- Le astrazioni usate per il software tendono ad evidenziare gli aspetti funzionali
 - Cosa è calcolato piuttosto che come è condotto il procedimento di calcolo
- Il passo finale di questo procedimento consiste nel fare esperimenti in ambienti controllati per validare il modello
 - Determinarne accuratezza e robustezza
- Procedimento ciclico: nuove osservazioni possono mettere in crisi il modello e portare alla definizione di nuove ipotesi e nuove validazioni
 - Nello sviluppo del software il processo di astrazione è simile e si parla di ciclo di sviluppo dei programmi

Principi di Astrazione nella Progettazione del Software

- Distinzione tra il livello di concettualizzazione e quello di realizzazione
- Astrarre per decomporre, rappresentare, generalizzare
- Astrarre sui dati e sulle funzioni

Fase di Progettazione



Fase di Progettazione: Concettualizzazione

- Appartengono alla fase di concettualizzazione tutte le attività relative a
 - Individuazione della soluzione al problema
 - Specifica degli oggetti da trattare
 - Ideazione delle scelte algoritmiche
- Risultato: lo “schema concettuale di progetto”
 - La specifica di cosa deve fare il programma che ci accingiamo a costruire

Fase di Progettazione: Realizzazione

- La fase di realizzazione si riferisce a quel processo che, partendo dallo schema concettuale di progetto, permette di ottenere uno specifico programma che “realizza” gli elementi dello schema
- Questo impone :
 - La scelta della architettura del programma in termini di un insieme di parti indipendenti (moduli software)
 - La realizzazione dei moduli, tenendo conto delle caratteristiche del linguaggio di programmazione prescelto

Astrarre per Scomporre, Rappresentare, Generalizzare

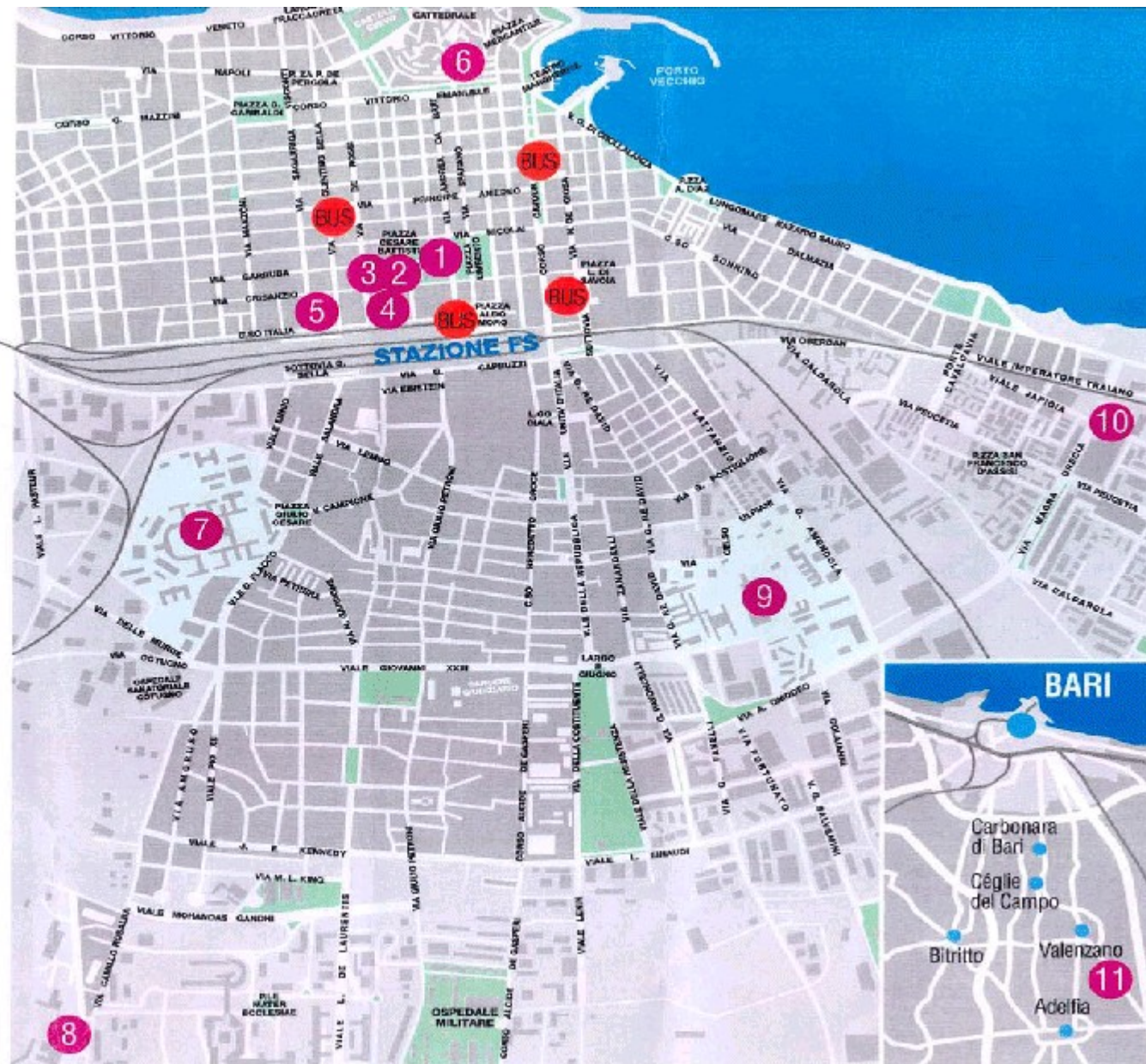
- La scomposizione di un problema in sottoproblemi è un metodo semplice e intuitivo per affrontare problemi complessi
 - Si astrae dalle caratteristiche peculiari dei sottoproblemi
 - Ci si concentra sulla loro funzionalità e interazione
- L'astrazione è tipicamente usata per cogliere gli aspetti essenziali di una situazione del mondo reale allo scopo di definire opportune strutture di rappresentazione che evidenzino alcuni aspetti e ne trascurino altri
 - mappe, grafici, immagini invece che descrizioni testuali nella comunicazione



BARI PLESSI UNIVERSITARI

FACOLTÀ E DIPARTIMENTI

- 1 **PALAZZO ATENEIO**
 - Facoltà di Lettere e Filosofia
 - Facoltà di Scienze della Formazione
 - Piazza Umberto I, 1*
- 2 **Facoltà di Lingue e Letterature Straniere**
Via Garruba, 6/B
- 3 **Facoltà di Giurisprudenza**
Piazza Cesare Battisti, 1
- 4 **Dipartimento di Linguistica, Letteratura e Filologia Moderna**
Via De Rossi, 233
- 5 **Dipartimento di Scienze Storiche e Geografiche**
Via Quintino Sella, 268
- 6 **Dipartimento di Studi Classici e Cristiani**
Strada Torretta, 6
- 7 **POLICLINICO**
Facoltà di Medicina e Chirurgia
Piazza Giulio Cesare, 11
BUS nn. 20, 27 da Via Q. Sella
- 8 **Facoltà di Economia**
Via Camillo Rosaiba, 53
BUS nn. 6, 27 da Via Q. Sella
- 9 **CAMPUS**
 - Facoltà di Scienze MM. FF. NN.
 - Via Orabona, 4*
 - Facoltà di Agraria
 - Via Amendola, 165/A*
 - Facoltà di Farmacia
 - Via Amendola, 173*
BUS n. 22 da P.zza A. Moro
- 10 **Facoltà di Medicina Veterinaria**
Istituti di:
 - Anatomia Normale
 - Anatomia Patologica
 - Patologia Aviare
 - V. Caduti di tutte le guerre, 1*
BUS n. 12/da P.zza A. Moro
BUS n. 2/da C.so Cavour
- 11 **Facoltà di Medicina Veterinaria**
Str. Prov. per Casamassima Km 3 (Valenzano)
BUS n. 4 da P.zza L. di Savoia



Astrarre per Decomporre, Rappresentare, Generalizzare

- Si astrae sempre quando si generalizza: rendere generale un metodo solutivo messo a punto per un problema particolare impone di comprendere l'essenza del metodo, prescindendo dalla peculiarità del caso
 - Si tende ad individuare un modello unico che si adatti a situazioni diverse, magari attraverso l'impostazione corretta di parametri

Astrarre sui Dati e sulle Funzioni

- Tali astrazioni portano immediatamente a due tipi di astrazione fondamentali
 - Astrazione sui dati
 - Consente di far riferimento a strutture algebrico-matematiche, caratterizzate da valori e da operazioni su tali valori
 - Si prescinde dal modo in cui queste strutture sono organizzate e realizzate mediante i costrutti di un linguaggio di programmazione
 - Astrazione sulle funzioni
 - Consente di concentrare l'attenzione su "cosa" fa una particolare operazione piuttosto che sul "come" è fatta
 - Si astrae dalle modalità di realizzazione, ci si concentra sul compito o sulle funzionalità di un segmento di programma

Tecniche di Astrazione: Evoluzione

- Le tecniche di astrazione si sono evolute nel corso del tempo, grazie a
 - Più approfondita comprensione della programmazione
 - Capacità di usare sempre di più le astrazioni come specifiche formali del sistema che descrivono
- Anni '60: attenzione, sia nelle metodologie che nei linguaggi, posta sulle funzioni e procedure
 - Riassumevano un segmento di programma in termini di un nome e una lista di parametri
 - Si era in grado di fare solo controlli di validità sintattica
 - Il termine “specifica” significava poco più che “intestazione di procedura”
- Tardi anni '60: astrazione trattata come una effettiva tecnica di organizzazione dei programmi
 - I primi linguaggi consentivano tipi di dati pre-definiti
 - interi, reali, booleani
 - Le prime strutture di dati furono trattate in modo sistematico nel 1968
 - Knuth “Fundamental Algorithms”
 - Emerse l'idea che un programmatore potesse “definire” tipi di dati adatti a un particolare problema e fu coniato il termine “Software Engineering”

Tecniche di Astrazione: Evoluzione /2

- Primi anni '70: definizione di una metodologia per costruire programmi partendo dalla definizione degli obiettivi
 - Questi venivano progressivamente specificati fino ad arrivare al vero e proprio “codice” (step-wise refinement o top-down programming)
- Insieme alla metodologia step-wise refinement fu sviluppata una “disciplina” dello scrivere programmi, la programmazione strutturata
 - Uso di strutture di controllo ideali caratterizzate da un unico ingresso e un'unica uscita
 - Risultato: relativa facilità di scoprire, in ogni punto del programma, quali ipotesi sono vere circa lo stato del programma, consentendo di verificare tanto l'aspetto statico che quello dinamico del programma
 - Questa tecnica di verifica è basata sulla definizione di asserzioni formali sul calcolo che il programma deve fare

Tecniche di Astrazione: Evoluzione /3

- Dai primi anni '70 si sono messe a punto tecniche di verifica basate sulle specifiche formali
 - Gli sforzi iniziali si concentrarono su cosa fosse utile mettere nelle specifiche formali
 - Nacque il dibattito sui tipi di dati astratti, sulla natura dei tipi e sulle connessioni con le algebre astratte
 - Di pari passo furono sviluppati linguaggi di programmazione in grado di trattare tipi di dati astratti che fornissero supporto per trattare
 - Tipi di dati
 - Moduli
 - Lo sviluppo del concetto di località
- Alcuni esempi
 - Pascal
 - Ada
 - Is Dod, Reference Manual for Ada, 1980
 - A.N. Habermann, D. Perry, Ada for experienced programmers, Addison Wesley, 1983
 - Modula
 - N. Wirth, Programming in Modula-2, Springer-Verlag, 1983
 - Euclid, Gypsy, Mesa, Clu, Alphard, Russel

Tecniche di Progettazione

- Tecniche di specifica
 - Consentono di esprimere gli elementi essenziali dello schema concettuale mediante
 - Formalismi grafici
 - Un linguaggio basato sulla logica e sull'algebra per descrivere gli aspetti concettuali dei tipi astratti di dato
 - Spesso seguono esplicitamente il principio di decomposizione
- Tecniche di programmazione (linguaggi)
 - Riguardano i metodi per la strutturazione e per la stesura dei programmi a partire dallo schema concettuale
 - Programmazione strutturata
 - Schemi iterativi e ricorsivi
 - Programmazione orientata agli oggetti
 - Programmazione logica e funzionale

Tecniche di Progettazione /2

- Modularizzazione
 - Consente di razionalizzare lo sviluppo del software costruendo programmi costituiti da parti indipendenti e interagenti
 - Un modulo:
 - è caratterizzato da una struttura interna
 - è definito con un determinato scopo
 - offre all'esterno un insieme prefissato di funzionalità utilizzabili da altri moduli
 - ha precise relazioni con altri moduli
- Tecniche di progettazione di algoritmi e strutture dati
 - Riguardano la definizione degli algoritmi e la individuazione delle strutture dati più appropriate per un determinato problema
 - tecniche enumerative, backtracking, divide et impera, etc.
- Tecniche di progettazione per specifiche classi di applicazioni
 - Specificatamente definite per applicazioni particolari, quali quelle concorrenti, quelle distribuite, quelle orientate alla gestione dei dati, etc.

Una Classificazione dei Linguaggi di Programmazione

- Linguaggi imperativi
 - programma = specifica di un insieme di istruzioni che corrispondono a precisi comandi impartiti ad una macchina che li esegue pedissequamente
- Linguaggi di programmazione Funzionale
 - programma = specifica di una funzione che, in base a un insieme di dati in ingresso, calcola il risultato secondo una legge specificabile in modo matematico
- Linguaggi di programmazione Logica
 - programma = specifica di una relazione che sussiste tra un insieme di dati, e la specifica è costruita mediante un sistema formale basato sulla logica
- Linguaggi di programmazione O.O.
 - programma = specifica di un insieme di oggetti che rappresentano gli elementi della situazione in gioco in un certo problema, e ciascun oggetto è specificato in termini di una struttura e di un insieme di operazioni tramite le quali si ottiene il comportamento voluto per risolvere il problema

Modularizzazione

- Idea di base: strutturare un programma in parti autonome e interagenti, dette moduli
 - Facilita il processo di sviluppo e le attività di analisi e di verifica.
 - Favorisce la riusabilità
 - Tra i diversi tipi di modularizzazione, grande importanza ha la modularizzazione per tipo astratto
 - Vengono sviluppati moduli che realizzano tipi astratti di dati significativi per l'applicazione
 - Concetto presente in tutte le organizzazioni
 - Banca: agenzie, ognuna con vari servizi alla clientela
 - Università: facoltà, dipartimenti e settori, con determinate funzioni e compiti

Modularizzazione /2

- Modulo: Unità di un programma con una sua struttura interna, definito per un determinato scopo, che offre all'esterno un certo insieme prefissato di servizi utilizzabili da altri moduli
 - Caratterizzato da:
 - Struttura interna, cioè l'insieme dei tipi, delle variabili e delle funzioni definiti nel modulo stesso
 - Insieme dei servizi che esporta, ovvero che offre agli altri moduli
 - Modalità con cui tali servizi possono essere utilizzati (interfaccia del modulo)
 - Insieme dei servizi che esso importa dagli altri moduli e che utilizza per le sue funzioni

Modularizzazione /3

- La qualità della modularizzazione aumenta:
 - all'aumentare della **Coesione** di un modulo
 - Esso incapsula un insieme di caratteristiche omogenee, sufficientemente indipendenti da altri moduli
 - all'aumentare dell'uso di **Information Hiding**
 - I dettagli interni al modulo non devono giocare alcun ruolo nell'utilizzo del modulo da parte di moduli esterni
 - al diminuire dell'**Accoppiamento** tra moduli
 - Non si sono create dipendenze non volute o non necessarie
 - Ad esempio, l'uso di variabili globali, visibili e utilizzabili da più moduli, va limitato al massimo poiché creano scambi di informazioni non facilmente controllabili
 - con l'utilizzo dell'**Interfacciamento Esplicito**
 - Suggerisce di rappresentare mediante parametri tutti i dati che vengono scambiati tra due sottoprogrammi

Strumenti di Progettazione

- Strumenti di specifica per la concettualizzazione
 - Formalismi (testuali, grafici) mediante i quali si descrive lo schema concettuale di progetto
- Strumenti di analisi per la concettualizzazione
 - Aiutano nell'analisi e nella verifica del documento di analisi e dello schema concettuale di progetto
 - Sistemi automatici o semiautomatici per Computer Assisted Software Engineering
- Strumenti per la programmazione
 - Supportano le fasi di scrittura del programma, compilazione e test
 - text editor, compilatori e interpreti, linker, debugger