Unità di apprendimento 6

Il software: dal linguaggio alla applicazione

Unità di apprendimento 6
Lezione 3

Tecniche e strumenti per
lo sviluppo di
un programma

In questa lezione impareremo:

- le diverse fasi della produzione del software
- che cos'è la programmazione strutturata
- quali sono gli strumenti per programmare
- in che cosa consiste la manutenzione del software

Scrivere un programma

- La *scrittura di un programma* si conclude un la scrittura dell'algoritmo
- Questo è l'ultimo passo della progettazione dell'algoritmo che
  - è una operazione complessa
  - richiede l'esecuzione di più fasi.

Scrivere un programma

- Una prima scomposizione individua i seguenti “momenti”:
  - definizione e comprensione del problema (analisi del problema);
  - ricerca della soluzione (strategia risolutiva);
  - descrizione/elencazione delle operazioni da eseguire nel linguaggio “umano” o semiformale dell’algoritmo;
  - codifica in un linguaggio di programmazione.

Scrivere un programma

- La fase di analisi è la più delicata e in essa possiamo sintetizzare le seguenti operazioni:
  - comprensione del problema;
  - modellizzazione del problema.
- Nella fase di analisi si individua *cosa deve essere* fatto

Scrivere un programma

- La seconda fase prende anche il nome di progettazione
- Si cerca la strategia risolutiva, cioè *come* risolvere il problema
- Si individua l'"idea", il procedimento operativo, che permette di raggiungere la soluzione.

Scrivere un programma

- Scrittura dell'algoritmo: l'algoritmo viene dapprima descritto in uno pseudolinguaggio o linguaggio di progetto.
- Il linguaggio di progetto è un linguaggio intermedio "che non esiste"
- Ogni programmatore lo personalizza in base alla propria esperienza maturata sul campo.

Scrivere un programma

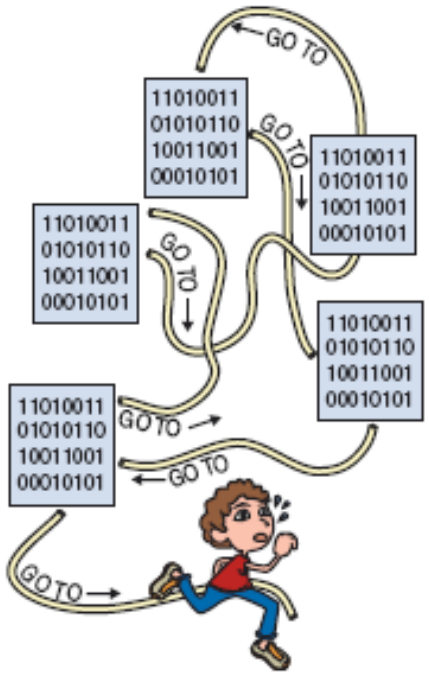
- L’ultima fase è la codifica vera e propria in un linguaggio di programmazione.
- Si traducono le istruzioni scritte nel linguaggio di progetto nella sintassi specifica del Pascal, del C, del Visual Basic o di qualunque altro linguaggio imperativo

Elementi di qualità del software

- Importanza di una specifica formalizzazione delle tecniche di sviluppo e produzione del software
- Per molto tempo il programmatore utilizzava solo la personale esperienza e/o del consiglio di colleghi e altri programmatori.
- Nasce l'ingegneria del software

Elementi di qualità del software

- Per anni si è scritto il “codice spaghetti”



1. si è programmato male
2. senza regole
3. utilizzo indiscriminato della istruzione di GO TO
4. la scrittura di programmi senza commenti
5. poca documentazione
6. codifica “artigianale per tentativi”

Elementi di qualità del software

- I primi passi dell'informatica teorica furono posti da Corrado Böhm
- Si dedicò allo studio delle tecniche per formalizzare la scrittura degli algoritmi
- Formulò con Giuseppe Jacopini uno tra i principali teoremi della programmazione strutturata.

Elementi di qualità del software

- Anticipiamo il *teorema fondamentale* che riporta il loro nome:
- Ogni algoritmo può essere descritto utilizzando unicamente tre tipologie di istruzioni:
  - la sequenza
  - la selezione
  - l'iterazione

Elementi di qualità del software

- Dimostrarono che l'uso del GOTO non è indispensabile per la scrittura di un programma.

```
...
10 IF A>B
20 THEN GOTO 80
30 <ramo ELSE>
40 GOTO 100
80 <ramo THEN>
```

L'assenza di GOTO rende il codice più facile

- da analizzare
- da seguire
- eventualmente da
- modificare e integrare.

Gli ambienti di sviluppo

- Sono stati progettati e realizzati degli strumenti che lo coadiuvano durante la realizzazione dei programmi
- Possono essere suddivisi in:
  - IDE
  - Framework
  - SDK

Gli ambienti di sviluppo : IDE

- Gli ambienti IDE (*Integrated Development Environment*) sono software che aiutano i programmatori nella codifica di un programma che consistono in :

- un editor per la scrittura del sorgente
- un compilatore,
- un generatore automatico di codice
- un debugger.

Gli ambienti di sviluppo : IDE

- Alcuni IDE sono multilinguaggio:
  - Eclipse
  - NetBeans
  - Visual Studio
- Altri sono rivolti a uno specifico linguaggio di programmazione:
  - Delphi
  - Apple XCode

Framework

- I framework vengono usati per descrivere tutta la struttura operativa nella quale viene elaborato un programma.
- Un framework realizza applicazioni senza scrivere neanche una riga di codice.
- Tipici esempi di framework:
  - all'ambiente .NET di Microsoft
  - PHP Zend

Framework

- In essi si possono disegnare pagine web senza dover scrivere il programma sorgente.
- Un *framework* consente al programmatore di risparmiare tempo evitando la riscrittura di codice già steso in precedenza.
- Il termine inglese *framework* significa *intelaiatura o struttura*, che ne definisce infatti la funzione;
- Grazie ad esso al programmatore rimane solo la creazione del contenuto vero e proprio dell'applicazione.

SDK

- SDK (*Software Development Kit*) è un termine che può essere tradotto come pacchetto di sviluppo per applicazioni
- Indica un insieme di strumenti adatti allo sviluppo e alla documentazione del software.
- Possono essere anche molto sofisticati ma hanno tutti alcuni componenti comuni

SDK

- un compilatore
- le librerie standard chiamate API (Application Programming Interface)
- la documentazione sul linguaggio di programmazione
- le informazioni sulle licenze da utilizzare per distribuire programmi creati con l'SDK

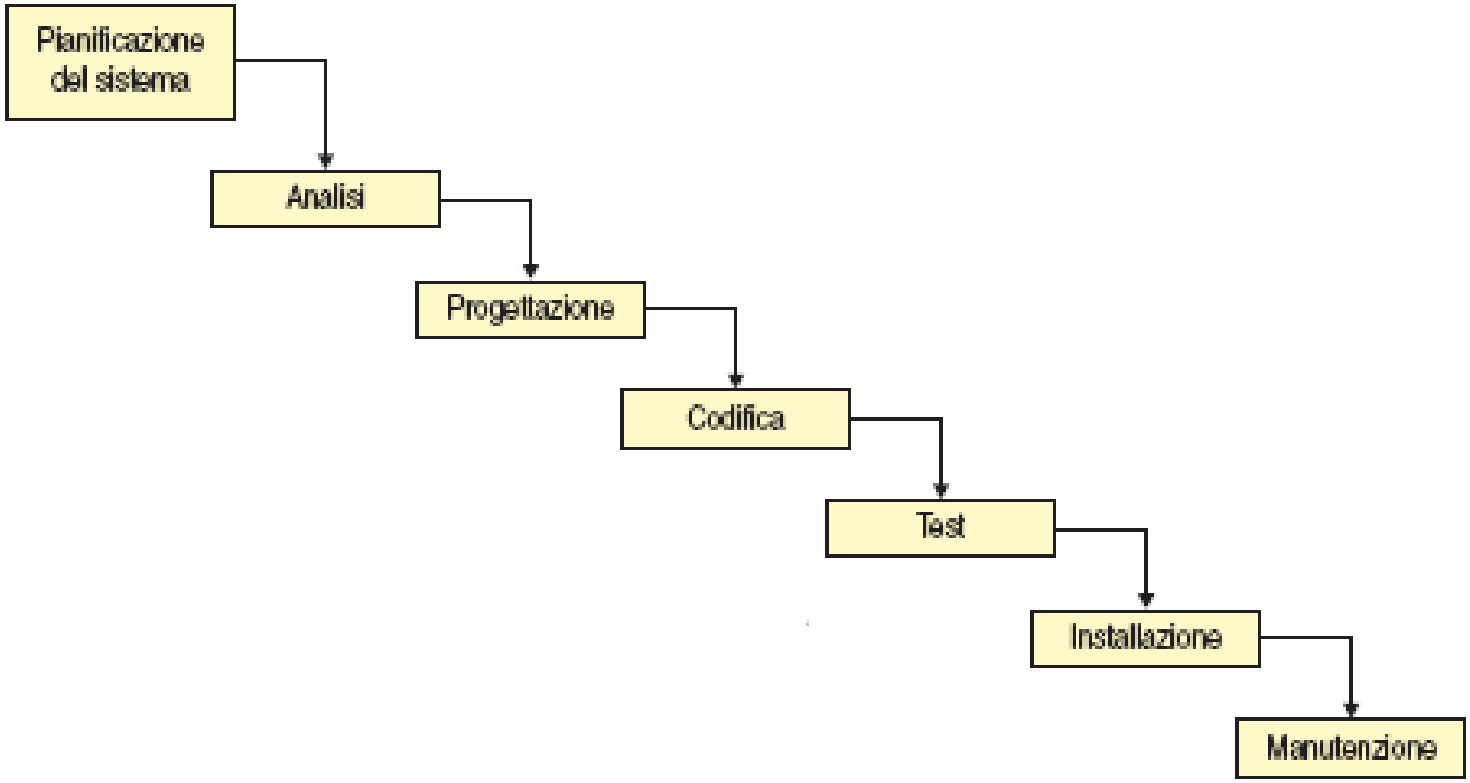
Il ciclo di vita del software

- Il programma che abbiamo scritto generalmente fa parte di un prodotto software, chiamato pacchetto software
- E' costituito da un insieme di programmi che possono essere collegati tra loro
- Oppure può anche essere una singola applicazione che condivide un database o altre risorse hardware o software.

Il ciclo di vita del software

- L'ingegneria del software ha indicato con ciclo di vita del software l'insieme di tutte le attività connesse alla produzione di un programma
- Le ha indicate secondo un modello a cascata.
- Nel modello a cascata (waterfall) ogni singola attività viene completata prima del passaggio alla successiva.

Il ciclo di vita del software



Modello a cascata

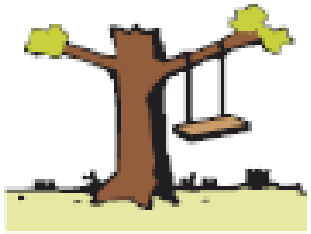
- Pianificazione del sistema
- In questa fase il programmatore identifica gli obiettivi che il software deve raggiungere.
- Si tratta di individuare le parti del sistema da realizzare con il software

Modello a cascata

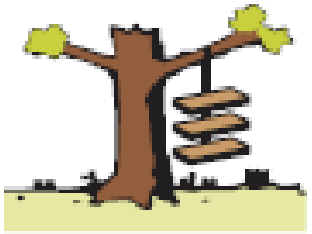
- Analisi
- Stabilisce
  - le caratteristiche dell'applicazione informatica,
  - delineandone gli aspetti di interfacciamento, di prestazione e di funzionalità;
- si decide *che cosa* il progetto dovrebbe realizzare, senza alcun riferimento diretto a come il programma realizzerà i suoi obiettivi.

Modello a cascata

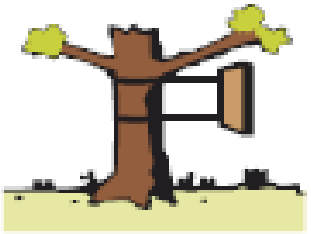
- Una analisi errata e/o incompleta porta al fallimento del progetto.



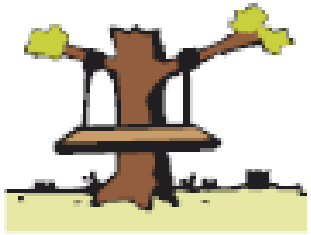
Quello che ha chiesto il cliente



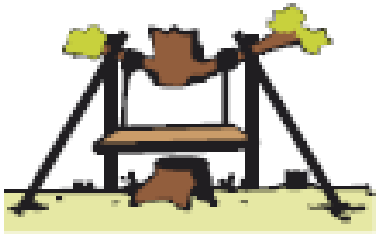
Quello che ha capito l'analista



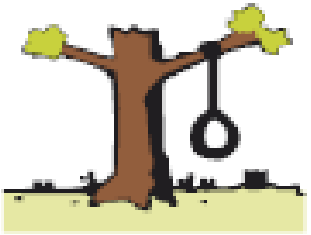
Come è stato tradotto in fase di progetto



Prototipo costruito in fase di realizzazione



Modifiche apportate in fase di test e debug



Esigenza reale che aveva il cliente

Modello a cascata

- Progettazione
- Si individuano i seguenti due obiettivi:
  - si sviluppa un piano di realizzazione del sistema, che comprende e descrive le diverse modalità operative da effettuare;
  - si individuano le strutture dati e archivi necessari alla soluzione del problema.

Modello a cascata

- Codifica (realizzazione)
- S scrive e si compila il codice sorgente eseguendo la vera e propria codifica delle istruzioni in linguaggio di programmazione.
  - si traducono in un programma le informazioni fornite dalla formalizzazione;
  - il programma viene elaborato automaticamente per produrre il codice che l'esecutore è in grado di interpretare: il linguaggio macchina.

Modello a cascata

- Test e debug
- Hanno come obiettivo quello di verificare l'assenza di errori
  - la fase di test verifica la correttezza dei risultati dell'elaborazione su un campione di dati di prova;
  - l'eventuale presenza di errori innesca la fase di debug, cioè la ricerca dell'istruzione (o segmento di codice) errata per procedere alla sua correzione e all'eliminazione del "baco".

Modello a cascata

- Installazione, verifica e collaudo
- L'installazione si intende la consegna del software al cliente e l'installazione fisica del programma sul sistema del cliente stesso;
- La verifica viene effettuata dagli utenti del programma e consiste nell'accertare che il software sia corretto rispetto alle specifiche
- Il collaudo accerta che, eseguendo prove con dati reali, il programma funzioni correttamente.

Modello a cascata

- Manutenzione
- È la fase permanente di supporto al sistema dopo la consegna all'utente.
- Riguarda in particolare:
  - l'aspetto correttivo
  - l'aspetto adattativo
  - l'aspetto migliorativo

ABBIAMO IMPARATO CHE...

- La scrittura dell'algoritmo è l'ultimo passo della progettazione dell'algoritmo che è una operazione complessa che richiede l'esecuzione di più fasi:
  - definizione e comprensione del problema (**analisi** del problema);
  - ricerca della soluzione (**strategia** risolutiva);
  - descrizione/elencazione delle operazioni da eseguire nel linguaggio "umano" o semi-formale dell'**algoritmo**;
  - **codifica** in un linguaggio di programmazione.
- **Böhm** e **Jacopini** formularono il principale teorema della programmazione strutturata: "ogni algoritmo può essere descritto utilizzando unicamente tre tipologie di istruzioni: la **sequenza**, la **selezione** e l'**iterazione**.
- L'**ingegneria del software** ha indicato con ciclo di vita del software l'insieme di tutte le attività connesse alla produzione di un programma e le ha indicate come segue, secondo un **modello a cascata** (o modello **waterfall**).
- Secondo il modello di sviluppo a cascata, il processo di realizzazione di un'applicazione informatica è scomposto in una sequenza di fasi successive:
  - pianificazione del sistema
  - analisi
  - progettazione
  - codifica (realizzazione)
  - test e debug
  - installazione, verifica e collaudo
  - manutenzione

ABBIAMO IMPARATO CHE...

- Il software ha sempre bisogno di **modifiche** per un insieme di cause:
  - innovazione tecnologica e quindi necessità di upgrade al nuovo hardware;
  - modifiche legislative e/o nuove procedure contabili;
  - aumento delle esigenze (in termini di maggiori prestazioni richieste dall'utente);
  - nuovi fabbisogni (reali e/o presunti) individuati dall'utente;
  - adempimenti fiscali e/o richieste della Pubblica amministrazione;
  - necessità di collegamento/integrazione con nuovi sistemi informatizzati;
  - semplice desiderio di innovazione.