Claudio Rossi

347390020

Crossi72

Il ciclo di vita di un software si compone di diversi elementi, ognuno di questi richiede competenze specifiche.

-Analisi

-Progettazione

-Implementazione: si inizia a scrivere il codice

-Collaudo

-Distribuzione

-Manutenzione: errori futuri, modifiche, ecc…

ANALISI

La fase di analisi definisce i bisogni, ossia specifica quali input e quali output deve gestire il software, inoltre deve definire il contesto dell’applicazione (sis. Operativo di destinazione, tipo di utente, tipo di hardware)

Ogni software al mondo è un trasformatore di dati, esso prende degli input, li trasforma e produce degli output.

Un software molto grande è solo un insieme di tanti software più piccoli.

L’analisi produce il documento di specifiche funzionali.

CONTROLLO DI VERSIONE

E’ un software che consente di memorizzare più stati di un file, ossia di tenere traccia di tutta la storia dell’evoluzione del file stesso.

Consente di tornare indietro a qualunque stadio in caso di errore o necessità.

Il sistema è costituito da un’architettura client-server.

Questa architettura consente di avere diversi vantaggi:

-cloud (per il server)

-ricostruzione di stati consistenti indipendentemente dalla posizione di sviluppo

-rende semplice la scrittura collaborativa

Esistono diversi protocolli di controllo versione (regole che un client deve seguire per comunicare) i più diffusi sono GIT e SVN.

I server di controllo di versione raccolgono la storia dei file in contenitori chiamati repository.

Un repository può essere pubblico o privato e l’accesso di lettura e scrittura può essere condiviso con altri utenti.

I comandi principali sono:

-clone, duplica sul computer del client lo stato attuale dei file presenti nel repository

-commit, crea una nuova versione dei file da mettere sul server (una specie di checkpoint manuale)

-revert, annulla tutte le modifiche ai file apportate dopo l’ultima commit (torna all’ultima versione disponibile)

-pull, sincronizza la cartella locale con il server, scaricando l’ultima versione presente sul server e aggiornando i file locali.

Fork: nuovo progetto che parte da una versione già preesistente, personalizzandola in base alle necessità. Esso rimane con la stessa licenza d’uso.

PROGETTAZIONE

La progettazione del software trasforma il documento di specifiche funzionali in uno schema a blocchi che rappresenta le unità funzionali.

Un’unità funzionale è un contenitore di codice che si può interfacciare con altre unità funzionali, essa riceve uno o più input e riceve uno o più output.

Il progettista smonta in maniera più efficace possibile il software, suddividendolo in tanti piccoli pezzi, che poi verranno distribuiti ai vari sviluppatori, che si concentreranno a scrivere il codice solo di quel pezzetto.

Alla fine, se tutto sarà stato sviluppato correttamente, tutto verrà unito e sarà funzionante.

Il progetto è costituito da uno schema a più livelli, ogni livello può essere costituito da uno schema.

Suddividere il progetto in varie parti è utile anche per la manutenzione futura: se tutto è suddiviso, sarà più facile correggere bug o modificare dettagli.

La separazione della soluzione in unità funzionali consente di riutilizzare il codice già scritto.

Spesso si utilizzano software, ossia unità funzionali standard disponibili sul mercato o scritto dal team di sviluppo per altri progetti.

Per ogni unità funzionale, il progettista deve indicare tutti gli input, tutti gli output e tutte le elaborazioni che portano alla produzione degli output.

Esistono sistemi di codifica standardizzati per la produzione degli schemi, il più diffuso è il linguaggio UML (Unified Modeling Language).

Alcuni strumenti di progettazione producono (oltre allo schema concettuale) la struttura del codice sorgente del software, contenente le firme delle funzioni e la struttura delle unità.