Ingegneria del Software: Lezione #14

Due on Monday, November 12, 2013

Tullio Vardanega 9:30am

Luca De Franceschi

E' necessario fare un lavoro profondo sui requisiti, è un attività ingegneristica. Da essa dipende una parte cospicua del successo del progetto. Anzitutto si sminuzza la complessità in parti più piccole, quindi dobbiamo classificarli, attribuirli delle etichette per manipolarli.

- **Cosa devo fare**, funzionalità da erogare;
- **Come devo fare**, esempio tramite l'uso di una tecnologia.

Dobbiamo avere un grado ancora più fine, distinguere tra **funzionalità**, **prestazioni** e **qualità**. Sistemi che aiutano a classificare i requisiti, metterli in una gerarchia, fare un'opera di scalpellamento.

- **Sistema**, ciò che dobbiamo realizzare, l'organizzazione di cui facciamo parte;
- **processo**, il modo in cui lavoriamo, insieme di attività coese e coordinate;
- **progetto**, una commessa commerciale con certi vincoli.

I processi o ce li infliggiamo da soli o vengono espressamente richiesti, così come il progetto. Il sistema è quello più complesso e va diviso ulteriormente:

- **Requisiti funzionali**, cosa devo fare;
- **Attributo**, performance e quality;
- **Vincoli**, legati alla natura del sistema, alle cose che deve fare e all'ambiente.

Le **prestazioni** si misurano rispetto al tempo di esecuzione, alla velocità con cui effettuo le cose, l'ampiezza di dati che posso gestire e il *throughput*, ovvero quante cose posso gestire simultaneamente.

Usabilità significa che il sistema si presta alle mie abilità ed è facile da usare.
Sicurezza significa che vanno garantiti alcuni vincoli sui dati e la loro accessibilità.
Availability significa che il sistema è sempre disponibile 24h/24 e deve sempre funzionare.
Portabilità, se sviluppo un sistema per un certo protocollo voglio poterlo riusare per protocolli differenti.
Mantenibilità, quando rilascio un sistema ad un utente devo garantire di poter correggere i difetti e aggiungere ulteriori funzionalità in futuro.
Alcune applicazioni sono legate e **vincoli ambientali**.
Serve frantumare i requisiti, perchè ogni sottosistema ha bisogno di un trattamento specifico, risolvibile in tempo e costi accettabili. L'analista deve poter pensare a requisiti così atomicizzati da poter essere **facilmente verificabili**. Il modo più semplice è scrivere requisiti **elementari**.
Verificare i requisiti ha un costo che devo mettere nel budget di progetto. Devo avere rapidamente un'idea di cosa mi costerà fare una determinata verifica. Posso verificare tramite **tests** che però andranno solo a scoprire ciò che *non va bene*: questo non basta. Sulle prestazioni invece devo spesso effettuare **misurazioni**.
I requisiti di qualità non posso farli in modo automatico, possono quindi avere un costo importante. Idealmente vorremmo tecniche di verifica che dimostrino la qualità e che abbiano un basso costo. I requisiti che solitamente vengono dati sono requisiti **macroscopici**. Una volta scalpellati mi devo mettere in una situazione minima di sopravvivenza. Ci sono tre tipologie fondamentali:

- **Obbligatori**, senza non c'è un sistema;
- **Desiderabili**, "sarebbe bello fare così";
- **Opzionali**, "se lo vuoi lo paghi".

L'interesse di un fornitore è **negoziare** questi requisiti. A noi sta il compito di classificare in modo elastico, perchè il fornitore non può fare tutto ma dovrà dire "io posso fare questo". In una gara d'appalto vince chi offre allo stesso prezzo il progetto migliore. I requisiti non devono essere in contraddizione tra loro, non devono essere mai in conflitto o sovrapposti. Per ogni passo di avanzamento del progetto voglio sapere sempre dove sono, cosa sto facendo e perchè. Tutto quello che faccio deve essere documentato in modo chiaro. I requisiti nascono scritti in un linguaggio naturale, sono testi narrativi, non sono formali. I capitolati sono scritti in un linguaggio informale e noi vogliamo portarci in un linguaggio che sia il più vicino possibile ad automi. Cercheremo tecniche che rendono il più possibile automatico ciò che facciamo (es. UML o tabelle).
IEE 830-1998 è un documento che descrive le pratiche raccomandate per scrivere un'analisi dei requisiti. Ci sono 8 proprietà fondamentali:

1. **Non ambigui**, mai alcuna incertezza su che cosa significano;
2. **Correct**, non deve nascere sbagliato perchè fa danni;
3. **Completi**;
4. **Verificabili** a basso costo;
5. **Consistenti** non posso chiedere una cosa e il suo contrario;
6. **Modificabili**, serve una tecnica che renda modificabile l'insieme dei requisiti. Sui requisiti devo poter aggiungere, togliere, cercare, aggiornare: operazioni tipiche da basi di dati;
7. **Tracciabile**, deve essere univocamente identificabile;
8. **Ordinati** (ranked) per rilevanza.

Non vado a scrivere un documento, ma lo produco a partire da una base di dati preesistente.

Un verificatore deve prendere un prodotto (documento) e trovare errori e difetti. Vogliamo un verificatore che sia il più possibile automatizzato. Tecnica di ricerca *a pettine*, **walkthrough**, tecnica completamente manuale che costa un sacco di fatica, una ricerca a largo spettro; lo si fa quando non si sa esattamente cosa cercare. Oppure c'è la tecnica dell'**ispezione**, in cui c'è una lettura mirata e strutturata; questa tecnica è molto più automatizzabile. Facendo walkthrough impariamo e sviluppiamo tecniche per fare ispezione. Sui requisiti devo poter fare una buona identificazione e classificazione. Numerazione sequenziale, cpiie ¡categoria,numero¿. Dentro un progetto tutto rimanda ai requisiti, tutto è sempre in una forma tracciabile e riconducibile al *perchè è lì*. Tutto ciò deve essere fatto attraverso procedure e automatizzazioni.
SEMAT, nella struttura c'è una progressione di stati molto utili da analizzare. I requisiti hanno un ciclo di vita proprio che passa attraverso 6 stati, ciascuno con delle dipendenze. Lo stato iniziale si chiama **conceived** (concepito), si vede un'opportunità nel fare le cose e i committenti sono identificati. **Bounded**, i requisiti sono su un recinto e potrò ragionare macroscopicamente di fattibilità. **Coherent**, quando i requisiti sono classificati e quelli chiari sono distinti.
La risposta che darà il committente porterà nello stato **acceptable**, diventa un punto dal quale avanzare e dal quale non vorremmo mai retrocedere. **Addressed**, i requisiti sono collocati, ho delle soluzioni specifiche e il prodotto che sto facendo soddisfa i requisiti. Questo stato lo raggiungiamo prima del collaudo. Se il proponente dà l'ok transiamo nello stato **fullfilled** in cui "le cose sono soddisfatte", stato dell'accattazione. Le transizioni sono più vicine nel tempo nella parte iniziale.

Qualità del software

L'intuizione importante è che c'è qualità se c'è un'aspettativa valutabile, deve essere una caratteristica sulla quale si può fare una valutazione obiettiva. Qualità come concetto oggettivo. Se il prodotto ha qualità avrà più mercato e avrà meno costo in fase di manutenzione.

> Insieme delle caratteristiche di un'entità (prodotto, processo, servizio) che ne determinano la capacità di soddisfare esigenze espresse e implicite.

E' importante vedere che le esigenze possono essere **espresse** o **implicite**. Spesso la parte implicita è dominante e bisogna scoprirla. La qualità può essere guardata da 3 diversi punti di vista:

- **Visione relativa e comparativa**, "io sono meglio del mio competitore";
- **Intrinseca**, hai qualità se soddisfi i bisogni, è chiaro che deve essere così, dimensione ovvia e non comparativa;
- **Quantitativa**, anche se non ho competitività oggi, un domani potrò confrontarm. Mi dà una posizione definitiva (es. stelle degli hotel) anche senza competizione.

La qualità viene erogata con un processo che si chiama **gestione di qualità**.

> La struttura organizzativa, le responsabilità, le procedure, i procedimenti e le risorse messe in atto per il perseguimento della qualità.

Chi lavora nei principi del SWE dovrebbe essere conforme a queste caratteristiche. Ambito del sistema di qualità.

- **Pianificazione**, voglio che le attività siano sistematiche nel modo richiesto; la pianificazione è alla base di ogni sistema di qualità. Senza pianificazione è molto più facile fallire;
- **Controllo**, ogni attività svolta può introdurre errori e devo cercare di intercettarli; un sistema di qualità ha bisogno di verifica;
- **Miglioramento continuo**.

Pianificazione di qualità, è relativa a obiettivi specifici di qualità. Per raggiungere quegli obiettivi serve fare cose specifiche e aggiuntive, ma devo anche avere le risorse per poterlo fare. "Cosa mi serve per fare meglio", inteso come competenze e infrastrutture. Il **controllo** dev'essere intelligente, rapido, mirato e oggettivo. Devo avere chiaro quale attività svolgere. Cerco di farlo in modo incrementale. **Quality assurance**, un atto proattivo di garanzia, accertamento di qualità. Faccio azioni preventive che mi diano garanzie e rendano molto meno pesante il controllo.