Lezione 5

Martedì 15 Ottobre 2013
Luca De Franceschi

I requisiti sono l'elemento fondamentale nella maturazione di un progetto; spesso non sono ben espressi o capiti dal cliente o da noi. Vari passi (stadi di maturità):

- **Conceived** → i requisiti vanno concepiti, nascono nel luogo dove stanno gli stakeholder, ovvero tutti coloro che danno forma ad un prodotto. Non significa "*completamente formato*", i requisiti vanno fatti emergere.

- **Bounded** → *recintati*, evitare che i requisiti si "*espandano come il gas*", cercare di "*mettere i requisiti dentro un recinto*".

- **Coherent** → devo guardare i requisiti e devono appartenere tutti allo stesso ambito. Anche qui gli stakeholder sono gli interlocutori.

- **Acceptable** → tutti gli stakeholder possono dire "*yes!*"; designa lo stato di maturità e dice che i requisiti sono uguali per tutti e tutti sono concordi.

- **Addressed** → ho la soluzione per i requisiti marcati come *acceptable*, sono capace di dimostrare che soddisfo i requisiti. Questo va dimostrato nella progettazione. "*Sappiamo che abbiamo la soluzione*".

- **Fullfilled** → "*compiere*"; ho un prodotto che fa esattamente ciò che mi aspettavo.

Elemento **team**, chi svolge il lavoro che nel modello di *Jacobson* lo troviamo in basso. Si lavorerà in team perché la swe è *attività di gruppo*, il team è fondamentale.

- **Seeded** → si comincia a identificare di chi ho bisogno, non come persone ma come *competenze*. Dobbiamo *spersonalizzare*, importa il ruolo che ho. Nessuno è un fuoriclasse (perché il fuoriclasse non è disciplinato e sistematico, quindi dannoso), anche se in una buona organizzazione può aiutare (altrimenti fa danno).

- **Formed** → le persone ce le ho e posso lavorare; istituisco dunque tecniche e strumenti per *collaborare*.

- **Collaborating** → le forme di collaborazione vanno studiate per evitare perdite di tempo.

- **Performed** → una volta acquisite le tecniche siamo efficienti ed efficaci.

- **Adjourned** → "*liberi tutti*", avendo acquisito l'esperienza del lavoro fatto.

Suddivido il tempo nel quale un prodotto sw sosta un suo ciclo di vita. So stimare la percentuale del costo delle varie attività. Nei sistemi professionali il 40% fra verifica e validazione. Mettere moltissima attenzione a capire che sto facendo ciò che serve.

Gestione di progetto → "*project management*", non significa essere il *project manager*.

Ingredienti:

- non mi devo inventare le cose da fare ma istanziare processi *standard* e collocarli nella mia realtà.

- Avendo le attività ragiono su quanto tempo e persone mi servono per affrontare i processi sul *calendario*. Ho determinati vincoli e devo confrontare i costi con il **budjet** (tempo e denaro); in un progetto serio se finisco soldi e tempo ho fallito. Devo avere sempre il controllo completo del rapporto fra *avanzamento* e *attesa*. Controllo accurato sull'avanzamento sistematico (*milestone*). E' un segno dell'avanzamento che ci consente di dire la differenza tra attesa e realtà. Se non succede siamo *avanti* o *indietro*.

Il sw in origine è pensato come *flessibile*; questa cosa è l'inizio di tutti i mali perché sembra avere poco costo, e quindi al sw è stata data poca dignità. Ma nel tempo ha acquisito un sacco di potere. Per molto tempo si è pensato che i sw fossero *repliche uniche* ("*one off*") → esemplari unici piuttosto che di serie. E' un grave errore creare cose che siano *irripetibili*. Non va bene perché la nostra attività prevalente sarà la *manutenzione*.

I **rischi latenti** provocano problemi alla gestione di progetto:

- **Variante nel personale** → nella disponibilità e nella composizione del team. Le persone possono "*sparire*". A quel punto bisogna trovare un rimpiazzo. Questa cosa va gestita, devo fare un piano.

- **Tecnologia** → non sta mai ferma; due tecniche: chi usa solo tecnologie *consolidate* (esempio uso di *cobol* nelle banche) perché cambiare produce rischio, e chi usa *tecnologia innovativa* (per motivi di competizione) ma fortemente instabile. Questa variabilità è un grande rischio.

Alcuni rischi sono evitabili:

- **Cambio di requisiti**

- **Ritardo nelle specifiche**

Gestione di rischi, vanno identificati all'inizio e in corso d'opera, perché possono variare nel tempo; analisi delle probabilità che essi emergano.

- **Identificazione** → capire quali sono.

- **Analisi** → analizzare i rischi.

- **Pianificazione** → cosa posso fare per evitare i rischi.

Un rischio si *previene* o si *mitiga*.

- **Controllo**

Il *project manager* vive costantemente sul *risk management*.

Il team necessita di **ruoli**, che identificano capacità e compiti. Dentro un'organizzazione produttiva

ci sono raggruppamenti di ruoli. Un ruolo è attivo su un progetto, la competenza di istanza di chiama *funzione aziendale*.

- **Qualità** → ciò che ci consente di puntare al miglioramento continuo di efficienza ed efficacia. Qualità di prodotto è diverso da qualità di processo. E' più importante la *qualità di processo*.

- **Sviluppo** → insieme delle competenze per la parte di attività tecnica e realizzativa del prodotto.

- **Direzione** → fa sì che l'organizzazione possa stare in piedi.

- **Amministrazione** → ("*service manager*") erogano/gestiscono l'infrastruttura che aiuta a fare il proprio lavoro (es. manutenzione e sicurezza). Se esiste una buona infrastruttura si lavora meglio.

Competenze allo *sviluppo*:

- **Analista** → colui che serve per fare analisi dei requisiti; aiuta l'avanzamento di maturità dei requisiti; è molto importante e occorre avere competenze su molti fronti, sapere ascoltaare gli stakeholder, scrivere requisiti *bounded* e *coherent*, ragionevoli, utili, realizzabili e verificabili. E' un compito molto delicato, deve sapere se un requisito si può fare. Non può promettere cose insensate. Sono competenze così decisive che ci sono pochi analisti. L'analista pensa al problema, non alla soluzione.

- **Progettista** → pensa alla soluzione. Deve avere competenze tecnologiche e tecniche. Ricevuto il problema tira fuori la migliore soluzione possibile nel rispetto dei vincoli finanziari e temporali. Competenze tecnologiche *senza pregiudizi*, ma in relazione alle necessità. Deve sapere di architettura, mettere insieme le parti della soluzione (*divide et impera*). Rende il problema piccolo affinché sia dominabile. Le parti devono essere organizzate in modo da essere governabili. Queste parti devono funzionare in un aggregato coerente. Un solo progettista in un team.

- **Programmatori** → il programmatore non inventa niente, deve fare ciò che ha detto il progettista. Deve scrivere in modo non ambiguo. Il passaggio di consegna tra progettista e programmatore deve essere chiaro. Permette il massimo parallelismo. Ho tanti programmatori quanti posso averne.

- **Verificatori** → la validazione si fa sul prodotto finito. Impiega almeno 1/3 del tempo. E' un ruolo attivo da subito. Serve per creare l'ambiente giusto, a rendere puliti i requisiti. E' presente *sempre*, su tutto l'arco del progetto. Include anche la verifica del codice: 1) impone al programmatore stesso di verificare il suo codice, 2) lo fa verificare ad una terza persona indipendente.

- **Responsabile** → ce ne sarà 1. Rappresenta il progetto presso il fornitore e presso il committente. E' quello che fa da intermediario, che dice "*come siamo messi*"; pianifica, gestisce risorse e rischi, coordina. Responsabilità su relazioni esterne. Deve sapere ciò di cui parla. Diventa responsabile dopo avere acquisito esperienza su altri ruoli.

- **Amministratore** → ruolo molto utile, prepara l'ambiente di lavoro, strumento di collaborazione e controllo di avanzamento. Gestione della documentazione di progetto.

Risoluzione di problemi legati alla gestione di processi. Assegnazione di **tickets**, un incarico con scadenza assegnato ed accettato.