Articolo

Analisi accurata dei requisiti

Una best practice sempre attuale

Ercole Colonese, 2009

Sommario

Introduzione

E' noto a chiunque si occupi di sviluppo software quale ruolo importante rivesta l'analisi dei requisiti per il buon esito di un progetto. E' altrettanto noto che i requisiti iniziali possono cambiare durante il progetto e che tali cambiamenti debbano essere gestiti opportunamente. E' meno noto, invece, che requisiti, anche ben definiti, possono assumere un significato diverso per chi sviluppa il software e per gli utenti che tale software dovranno poi utilizzarlo. Il modello SERVQUAL per la qualità dei servizi indica lo scostamento ("Gap") tra il servizio atteso e quello percepito dall'utente finale come uno dei fattori di maggiore insoddisfazione. Risulta quindi di assoluta importanza "condividere" il significato dei requisiti con gli utenti finali e garantire una univocità di interpretazione. Un articolo pubblicato sulla rivista *Qualità on-line* di AICQ descrive l'applicazione del modello SERVQUAL ai progetti di sviluppo software[1].

L'interpretazione e il grado d'importanza che gli utenti danno ai requisiti dipende dal contesto nel quale essi operano: responsabilità loro assegnate, contenuti dei task da eseguire, modalità di svolgimento dei propri compiti, preferenze ed abitudini consolidate nel tempo, ambiente culturale ed organizzativo nel quale operano.

I requisiti assumono il loro corretto significato e la giusta importanza solo quando riferiti allo scenario operativo, organizzativo e culturale in cui nascono, sono percepiti e valutati.

Analogamente, la soluzione proposta a fronte dei requisiti definiti sarà valutata correttamente solo nel contesto in cui essa sarà utilizzata dagli utenti.

La pratica proposta per l'analisi dei requisiti prevede quanto segue:

- Raccolta, analisi e documentazione dei requisiti;
- Condivisione dei requisiti con il cliente e gli utenti;
- Allocazione dei requisiti nell'architettura applicativa;
- Stima e pianificazione del progetto partendo dai requisiti;
- Verifica che i requisiti siano indirizzati nella progettazione;
- Progettazione dei test partendo dai requisiti;
- Gestione delle modifiche dei requisiti durante l'intera durata del progetto.

[1] L'articolo si trova sul sito dell'autore (www.colonese.it/pubblicazioni/articoli).

Ciascuna attività risulta particolarmente efficace se svolta secondo le modalità descritte di seguito.

Ruolo svolto dal cliente nella gestione dei requisiti

Il cliente gioca un ruolo molto importante in questa pratica. Di seguito si riporta una lista di controllo (*Checklist*) delle attività, controlli e verifiche da effettuare.

- *Corretta interpretazoine dei requisiti*. Assicurare che le diverse esigenze di business e le varie caratteristiche tecniche delle soluzioni siano tutte correttamente interpretate dal fornitore. I requisiti funzionali devono descrivere di "cosa" hanno bisogno i diversi utenti delle varie organizzazioni interessate al progetto. Le caratteristiche tecniche devono descrivere "come" la soluzione deve essere realizzata in termini di prestazioni, facilità d'uso, robustezza, sicurezza, ecc.
- *Requisiti di operatività*. Tenere in debito conto i requisiti espressi dall'organizzazione responsabile della gestione applicativa nell'ambiente di esercizio. I requisiti relativi alla gestione operativa sono di tipo tecnico (coerenza con le architetture esistenti), di tipo operativo e gestionale (procedure operative) e di tipo prestazionale (tempi di completamento delle procedure, utilizzo delle risorse, ecc.). L'organizzazione responsabile della gestione applicativa deve esprimere i propri requisiti e questi devono essere opportunamente indirizzati nel progetto.
- *Condivisone dei reqisti*. Condividere i requisiti con tutte le parti ed assicurare la loro correttezza effettuando la revisione dei documenti prodotti dal fornitore (Specifica dei requisiti). E' bene che la revisione sia fatta dalle varie unità organizzative che hanno espresso i requisiti.
- *Verifica delle stime*. Verificare che le stime del progetto siano fatte basandosi sui requisiti condivisi (allocazione dei requisti nel progetto) e non in maniera astratta su modelli di progetto simili. Stime che non includano lo sviluppo di tutti i requisiti costituiscono un forte rischio per il progetto.
- *Verifica della pianficazione*. Verificare che la pianificazione del progetto sia realistica, includa le attività necessarie per indirizzare tutti i requisiti condivisi e comprenda attività di controllo e verifica della qualità del lavoro svolto (test e collaudo). In caso contrario, la pianificazione risulterà poco realistica e costituirà un forte rischio per il progetto.
- *Gestione delle modifiche*. Assicurare che le eventuali modifiche ai requisiti originali siano gestite coerentemente e correttamente: siano cioè formalizzate, condivise e prese in considerazione per rivedere le strategie di sviluppo e di test, le stime, i piani.

Modalità di svolgimento della pratica

1) Raccolta, analisi e documentazione dei requisiti

La "raccolta dei requisiti" è svolta partendo dalle diverse fonti individuate e con modalità diverse (elicitazione dei requisiti): analisi della documentazione fornita dal cliente o già in nostro possesso; interviste a persone del cliente direttamente coinvolte nel progetto o comunque direttamente interessate al successo del progetto; discussioni con il cliente in circostanze diverse ma attinenti al progetto ed ai suoi contenuti; ecc. E' bene prendere nota di tutte queste "fonti" e formalizzarne i contenuti.

L' "analisi dei requisiti" consiste nell'interpretare i requisiti raccolti, dare a ciascuno di essi un significato univoco (non ambiguo), chiaro e completo, consistente e coerente con gli altri requisiti e testabile. Occorre anche definire la tipologia del requisito ("funzionale" o "non-funzionale"), l'importanza che il requisito riveste per il business (alta, media o bassa) ed assegnare una priorità (alta, media, bassa) che permetta di stabilire quando implementare il requisito nell'ambito del progetto (nel presente rilascio o in uno successivo). Mentre un requisito "funzionale" descrive la funzionalità richiesta (cosa deve fare), i requisiti "non funzionali" identificano le caratteristiche con cui la funzionalità dovrà essere resa disponibile (affidabile, veloce, sicura, facile, intuibile, ecc.) oppure limiti e vincoli, norme e standard da rispettare. I requisiti non funzionali sono sempre presenti anche se spesso non sono esplicitamente espressi in quanto ritenuti ovvi oppure perché ci si è dimenticati di definirli. Per questo si dicono spesso

"impliciti" ma sono ugualmente richiesti, attesi e quindi valutati in fase di accettazione della soluzione finale.

La norma ISO/IEC 9126 propone un modello della qualità del software secondo tre diversi punti di vista: "interno", "esterno" e "in uso". Le caratteristiche interne ed esterne del software sono: *funzionalità*, *affidabilità*, *usabilità*, *efficienza*, *manutenibilità* e *portabilità*. Ciascuna di esse ha, a sua volta, attributi specifici. Le caratteristiche in uso del software, cioè secondo il suo utilizzo, sono invece: *efficacia*, *produttività*, *sicurezza* e *soddisfazione*. Nel sito dell'autore[2] è riportato un manuale che descrive il modello ISO/IEC 9126 e le caratteristiche del software.

La "documentazione dei requisiti" può utilizzare uno schema che permetta di gestire nel corso del progetto i requisiti, la loro implementazione ed i possibili cambiamenti. Tale documento (*Specifica dei requisiti*) rappresenta il "riferimento" più importante in tutto il progetto e ad esso si fa riferimento nelle altre best pratiche proposte. Il documento dei requisiti può essere anche un semplice foglio di calcolo con le seguenti colonne:

- *Identificativo univoco del requisito,*
- *Titolo del requisito,*
- *Tipologia del requisito,*
- *Importanza del requisito,*
- *Priorità del requisito,*
- *Descrizione dettagliata del requisito,*
- *Stato del requisito,*
- *Data in cui è stato definito o cambiato,*
- *Condivisione del requisito,*
- *Data della condivisione,*
- *Fonte del requisito,*
- *Inclusione del requisito nella progettazione,*
- *Inclusione del requisito nelle prove e collaudi (casi di test),*
- *Nota informativa con dettagli che possano aiutare a gestire il requisito.*

Una tabella così fatta è detta "Matrice di tracciabilità dei requisiti" e rappresenta uno strumento (tool) estremamente importante ai fini di una corretta gestione dei requisti.

[2] http://www.colonese.it/pubblicazioni/html

Un modo concreto ed efficace di rappresentare l'interpretazione che diamo dei requisiti è quello di tradurli in scenari operativi, detti "casi d'uso" (in inglese *Use Case*). Essi descrivono le modalità con cui gli utenti interagiscono con il sistema per svolgere determinati compiti. La modellazione dei requisiti con la tecnica degli *Use Case* permette quindi di concentrare l'attenzione sugli utenti e le modalità operative. Una funzione è perciò descritta non in maniera astratta ma come essa sarà realmente utilizzata da un particolare utente per svolgere un determinato compito.

2) Condivisione dei requisiti con il cliente

La "condivisione dei requisiti" è fondamentale per garantire l'univocità della loro interpretazione e la loro completezza.

Consiste nel condividere con il cliente e gli utenti i requisiti e/o i casi d'uso. La descrizione dei requisiti, le tipologie, l'importanza e le priorità permettono di condividere quindi l'interpretazione delle esigenze. La descrizione dei casi d'uso, inoltre, permette di condividere gli scenari operativi nei quali il software sarà utilizzato dagli utenti finali.

Solo la condivisione può garantire che la nostra interpretazione delle esigenze del cliente coincida con la sua e che, in fase di sviluppo, andremo a realizzare una soluzione come quella descritta dai casi d'uso. Sarà perciò chiaro e pacifico ad entrambe le parti che si andrà a sviluppare una soluzione che indirizzi i "requisiti condivisi" e non la "nostra" interpretazione di questi e nemmeno la "sua" aspettativa più o meno nascosta.

Le caratteristiche del software (requisiti non funzionali) non sono generalizzabili all'intera applicazione ma legate alle specifiche funzionalità e alle modalità con cui gli utenti le utilizzeranno. La condivisione dei requisiti va fatta, quindi, con diversi utenti (o categorie di utenti) a seconda delle funzionalità in discussione.

L'assegnazione di una priorità (o importanza) ad ogni requisito permette di organizzare il progetto in fasi con rilasci successivi e contenenti i requisiti secondo le priorità condivise con il cliente.

Ogni modifica ai requisiti iniziali seguirà lo stesso processo descritto: condivisione della modifica con il cliente per garantire la consistenza e la congruenza

con quanto già condiviso e indirizzato inizialmente nel progetto.

La tabella dei requisiti ("Matrice") è aggiornata con quanto condiviso e mantenuta aggiornata durante l'intera durata del progetto come vedremo nel proseguo.

3) Allocazione dei requisiti nell'archtettura software

I requisiti condivisi sono quindi collocati nella progettazione tramite l'architettura software della soluzione proposta.

Patendo dai requisiti e dai casi d'uso, il modello concettuale della soluzione software descrive il sistema nei suoi componenti, i flussi di controllo e dei dati, gli eventi, le interazioni con gli utenti, i modelli degli oggetti, ecc.

Lo scopo è di aiutare gli sviluppatori nella comprensione del probelema stesso, prima di iniziare la progettazione di dettaglio.

I modelli concettuali includono, quindi, i modelli delle entità del "dominio del problema" configurati in modo da riflettere le relazioni e le dipendenze del mondo reale.

4) Stima e pianficazione dei requisiti nel progetto

La stima del progetto deve partire sempre dall'elenco dei requisiti condivisi e alllocati nell'architettura software.

La stima tiene conto sia delle caratteristiche funzionali che di quelle qualitative (cioè non funzionali). Lo sviluppo di una funzionalità può richiedere infatti tempi e costi diversi secondo le sue caratteristiche qualitative. Una soluzione senza requisiti di sicurezza, ad esempio, costerà meno di una in cui tale caratteristica sia critica per il business. Analogamente, una funzionalità con alte prestazioni (es.: tempi di risposta inferiori ad un valore di soglia) avrà costi superiori ad un'analoga funzioni senza tali caratteristiche prestazionali, ecc.

Occorre dunque assicurare che le stime siano fatte basandosi sulla tabella dei requisiti ("Matrice") e che tutti i requisiti siano stati presi in considerazione.

Si comprende bene come ogni modifica ai requisiti iniziali possa influire sulle stime fatte e richiedere quindi che queste siano aggiornate ed approvate prima che la modifica sia inclusa nel progetto.

5) Verifica che i requisiti siano indirizzati nella progettazione

Il documento dei requisiti e l'architettura software sono la base per la progettazione di dettaglio. Il responsabile del design assicura che essi siano tutti correttamente indirizzati nella progettazione. La verifica tiene conto sia dei requisiti funzionali che di quelli qualitativi. La progettazione deve quindi garantire che le funzionalità sviluppate possano essere utilizzate dagli utenti finali nei loro contesti di lavoro: responsabilità loro assegnate, diversi compiti da svolgere, modalità di utilizzo, preferenze e criteri di valutazione.

Una funzione semplice come, ad esempio, l'inserimento di dati durante una transazione avrà caratteristiche diverse in termini di usabilità, prestazioni e sicurezza secondo lo scenario nel quale l'utente finale la utilizzerà. Una funzione realizzata tramite una transazione batch richiederà tempi di lavorazione differenti rispetto alla stessa funzione svolta in modalità on-line da un operatore che abbia davanti al suo sportello una coda di utenti in attesa. Una transazione svolta da un utente remoto sul Web ha requisiti di sicurezza diversi dalla stessa transazione eseguita da un utente autorizzato nel sistema informativo aziendale interno, ecc.

La verifica della bontà della progettazione tiene conto, dunque, del fatto che essa indirizzi correttamente e completamente i requisiti condivisi con il cliente, sia quelli funzionali che quelli qualitativi, con le priorità e le criticità definite.

La verifica della progettazione include l'aggiornamento della tabella dei requisiti ("Matrice") riportando per ogni requisito l'indicazione se esso sia stato correttamente e completamente indirizzato dalla progettazione in esame.

6) Progettazione dei test artendo dai requisiti

La documentazione dei requisiti ("Matrice") è la base di partenza per la progettazione dei test. Occorre

garantire che ogni requisito, funzionale e qualitativo, sia indirizzato da almeno un caso di test. Così si garantisce che le funzionalità e le caratteristiche del software siano verificate nella loro completezza e correttezza.

Si sa che esistono limiti pratici (oltre che terorici) nel definire un batteria di test capace di coprire l'intera gamma di funzionalità, sottofunzionalità, parametrizzazioni, condizioni speciali, anomale e eccezionali di un software. I tempi e i costi di tali test non incontrebbero mai le esigenze del "time to market" e le restrizioni imposte dai budget a disposizione. Occorre dunque fare di necessità virtù. La tecnica che si suggerisce per indirizzare tale restrizione è quella di riportare in una tabella 2x2 (a quattro quadranti) con dimensioni "Importanze-Frequenza" le funzionalità direttaente derivanti dai requisiti della "Matrice". La figura che segue ne mostra un esempio.

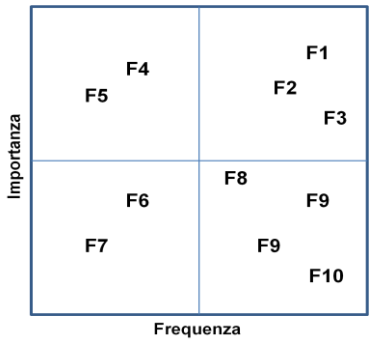

Figura 1 – Matrice Importanza-Frequenza delle funzionalità software.

Nel quadrante in alto a destra saranno collocate, dunque, le funzionalità con la massima importanza per il business e la massima frequenza di utilizzo. Queste saranno perciò tutte coperte da casi di prova adeguatamente progettati. Contrariamente, nella casella in basso a sinistra saranno collocate le funzionalità con bassa importanza e scarsa frequenza di utilizzo. Senza penalizzarle eccessivamente, queste saranno comunque coperte da un numero minimo di casi di prova. Le altre due caselle – in alto a sinistra (bassa frequenza di utilizzo ma alta importanza) e in basso a destra (minore importanza e bassa frequenza) – saranno coperte da casi di prova opportunamente selezionati.

Per valutare il grado di copertura dei requisiti da parte dei casi di prova è bene creare una "Matrice di test" che riporti l'elenco dei requisiti e l'elenco dei casi di prova. La casella di incrocio tra requisito e caso di prova indicherà se la copertura è assicurata o meno.

Tabella 1 - Matrice di copertura dei requisiti da parte dei test.

	CT-1	CT-2	CT-3	CT-4	CT-5	CT-6	CT-7	...	CT-n
REQ-1	x	x	x	x				x	
REQ-2	x		x				x	x	x
REQ-3		x		x					x
REQ-4		x		x	x	x			x
REQ-5			x	x	x	x			
...						x	x		
REQ-n							x	x	

Una tale matrice permette di verificare il grado di copertura dei requisiti e consente, allo stesso tempo, di ottimizzare i test. Un requisito che sia coperto da più casi di prova, per esempio, suggerisce di eliminare i casi duplicati che non aggiungono valore al test del requisito. Al contrario, la matrice può rilevare che alcuni requisiti non siano opportunamente indirizzati da test opportuni.

7) Gestione delle modifiche ai requisiti durante il progetto

L'importanza della gestione delle modifiche ai requisiti iniziali è ora chiara più che mai a tutti noi. Ogni modifica ai requisiti deve seguire tutte le fasi elencate e descritte in precedenza in quanto essa può avere influenza su vari aspetti del progetto: interpretazione del suo contenuto, progettazione della soluzione, stima dei tempi e dei costi, piano dei test. La modifica dei requisiti iniziali richiede l'approvazione dei responsabili del progetto, sia lato cliente che lato fornitore, prima della sua inclusione nel progetto stesso.

Risultati positivi conseguiti dall'adozione della pratica

Il risultato dell'adozione di tale pratica è importante ai fini del successo del progetto e consiste in:

- *Progettazione corretta della soluzione* che indirizzi esattamente le necessità del cliente e rispecchi il punto di vista dei vari utenti finali; riduzione drastica dei rifacimenti dovuti a ricicli in fase di design e codifica a seguito di

interpretazioni dubbie o contestate dei requisiti (per esempio, a fronte di discussioni con gli utenti finali e con il gruppo di progetto);

- *Stime più accurate e piani più realistici* con ricadute positive sulla pianificazione dei tempi e dei costi del progetto;
- *Maggiore garanzia del rispetto dei tempi e dei costi del progetto* tramite la pianificazione di rilasci successivi che includano lo sviluppo dei requisiti in base alle priorità per il business del cliente;
- *Migliore progettazione dei casi di test* che risulterà più completa, corretta ed ottimizzata; la pianificazione dei diversi tipi di test terrà conto delle funzionalità e delle varie caratteristiche del software (usabilità, performance, robustezza, sicurezza, operatività, ecc.);
- *Esito positivo del collaudo di accettazione* in quanto le funzionalità e le caratteristiche del software progettate, sviluppate e testate saranno basate sui requisiti condivisi con il cliente.

I benefici elencati sono stati personalmente verificati in molti progetti, sia semplici che complessi, in cui la "best practice" è stata adoperata.

L'adozione della pratica non richiede alcun costo aggiuntivo al progetto in quanto l'analisi dei requisiti sarà svolta "in maniera diversa" piuttosto che con attività aggiuntive.

La formazione iniziale del personale è da considerarsi un investimento dell'azienda e non un costo addebitabile al singolo progetto.

Potenziali problemi generati dalla mancata o errata adozione della pratica

I problemi derivanti da una errata analisi dei requisiti sono rilevanti, conosciuti da tutti gli addetti ai lavori e messi in luce dall'indagine in corso. I più importanti sono:

- *Bassa qualità della progettazione, dello sviluppo e del test* della soluzione che non rispecchia le reali esigenze degli utenti finali; le ripercussioni più gravi sono sui tempi di consegna, sui costi del progetto e sulla qualità della soluzione che risulterà basata sull'interpretazione unilaterale delle esigenze e senza garanzia di completezza;
- *Stime poco accurate e pianificazione non realistica* che non tengono conto delle numerose attività di rifacimento e dei ricicli dovuti alle modifiche ai requisiti non previsti o alla loro diversa interpretazione; le ripercussioni più evidenti sono sui tempi di consegna e sui costi di progetto;
- *Mancato rispetto dei tempi di consegna e superamento del budget di progetto* a causa dell'impatto negativo delle attività di rifacimento e di ricicli;
- *Rischio di mancata accettazione* della soluzione sviluppata da parte degli utenti finali a causa della contestazione sulla completezza e l'interpretazione delle loro esigenze.

L'analisi dei dati ricavati dall'indagine in corso e l'esperienza maturata in molti anni di lavoro e progetti diversi identifica nella mancata o scarsa adozione di tale "best practice" l'origine principale dei problemi evidenziati sopra.

Bibliografia

[1] Sommerville, Ian. *Ingegneria del Software*. Addison Wesley, 7ª Edizione (2005). pp. 11-136.

[2] Pressman, Roger S. *Principi di ingegneria del software.* McGram-Hill (2005). pp. 159-194.

[3] Ghezzi, Jazayeri, Mandrioli. *Ingegneria del software*. Prentice Hall 2ª Edizione (2004). pp 177- 291.

[4] Binato, Fuggetta, Sfardini. *Ingegneria del software.* Addison Wesley (2006). pp 67-108.

[5] Arlow J., Neustadt, I. *UML 2 e Unfied Process.* McGraw-Hill (2006). pp 45-121.

[6] Zuser, W., Biffl, S., Kohle, M. *Ingegnria del software con UML e Unified Process.* McGraw-Hill (2004). pp 69-105.

[7] Kruchten, Philippe. *Rational Unified Process - Introduzione.* Addison-Wesley (2000). pp 161-174.

Autore

Ercole Colonese è consulente di direzionale e servizi IT. Opera, in particolare, nell'ambito dello sviluppo software, dei servizi IT e della formazione.

La sua esperienza è maturata in moltissimi anni di lavoro presso i laboratori internazionali IBM di sviluppo software dove ha ricoperto ruoli tecnici, manageriali e dirigenziali.

E' socio APCO. E' membro del Consiglio Direttivo di AICQ-CI per il Comitato per la Qualità del Software e dei Servizi IT.

Ha realizzato Sistemi aziendali integrati di gestione per la Qualità, certificati ISO 9001, e Sistemi di gestione dei servizi IT, certificati ISO 20001:2005. Ha implementato modelli di eccellenza (EFQM, Malcom Baldrige, Six-Sigma). Ha condotto progetti di reingegne-rizzazione dei processi di sviluppo software in ottica di miglioramento delle performance e della qualità. Ha applicato il modello di maturità *Capability Maturity Model* (SEI-CMMI).

Come docente, tiene corsi di Software Engineering, Project Management e ITIL. In passato ha tenuto corsi sull'Ingegneria del software presso il Learning Center di IBM, l'Università Tor Vergata di Roma, il Politecnico di Vibo Valentia e l'Università del Sacro cuore di Roma.

Ha collaborato con CIRPS su progetti di ricerca tecnica.

Ha collaborato con CNIPA (ora DigitPA) sul tema del riuso del software applicativo.

Collabora alla pubblicazione dei Quaderni AICQ e articoli monotematici sulla qualità del software e dei servizi IT pubblicati sulla rivista Qualità On-Line di AICQ.

Articoli sull'Ingegneria del software sono pubblicati sul proprio sito all'indirizzo:

(http://www.colonese.it/pubblicazioni/htm).

e-mail: ercole@colonese.it www.colonese.it