Certified Tester

Syllabus Advanced Level

Test Manager

Versione 2012 (versione italiana 03a aggiornata a novembre 2022)

International Software Testing Qualifications Board

ITAlian - Software Testing Qualifications Board

Copyright © International Software Testing Qualifications Board (di seguito indicato come ISTQB®).

Storico delle review

Versione	Data	Note
V2007	12/10/2007	Certified Tester Advanced Level syllabus version 2007
V2012	01/01/2013	Certified Test Manager Advanced Level syllabus version 2012

Storico delle Review di Traduzione per ITA-STQB

REV.	AUTORI	DESCRIZIONE DELLE MODIFICHE	DATA DI APPROVAZIONE DEL COMITATO SCIENTIFICO
01	M.Sogliani	Traduzione del syllabus TM Advanced level	23 Novembre 2012
02	S. Reale	Review di errata corrige	19 Agosto 2013
03	S. Reale e C. Sobrero	Correzioni e allineamento al Glossario	Agosto 2022
03a	S. Reale e C. Sobrero	Errata corrige versione 03	Novembre 2022

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Indice

Storico delle review 2
Indice 3
Ringraziamenti 5
0. Introduzione a questo Syllabus 6
0.1 Scopo di questo Documento 6
0.2 Panoramica 6
0.3 Obiettivi di Apprendimento Esaminabili 6
1. Processo di Test – 420 minuti 7
1.1 Introduzione 7
1.2 Pianificazione dei Test, Monitoraggio e Controllo dei Test 8
1.2.1 Pianificazione dei Test 8
1.2.2 Monitoraggio e Controllo dei Test 9
1.3 Analisi dei Test 10
1.4 Progettazione dei Test 12
1.5 Implementazione dei Test 12
1.6 Esecuzione dei Test 13
1.7 Valutazione dei Criteri di Uscita e Reporting 13
1.8 Attività di Chiusura dei Test 14
2. Test Management – 750 minuti 16
2.1 Introduzione 17
2.2 Test Management nel Contesto 17
2.2.1 Capire gli Stakeholder del Testing 17
2.2.2 Ulteriori Attività e Prodotti di Lavoro del Ciclo di Vita dello Sviluppo Software 18
2.2.3 Allineamento delle Attività di Test e delle altre Attività del Ciclo di Vita 19
2.2.4 Gestire il Testing Non-Funzionale 22
2.2.5 Gestire il Testing Basato sull'Esperienza 22
2.3 Testing basato sul Rischio e altri Approcci per la Prioritizzazione dei Test e l'Allocazione dell'Effort 23
2.3.1 Il Testing basato sul Rischio 23
2.3.2 Tecniche di Test Basate sul Rischio 27
2.3.3 Altre Tecniche per la Selezione dei Test 30
2.3.4 Prioritizzazione dei Test e Allocazione dell'Effort nel Processo di Test 31
2.4 Documentazione di Test e altri Prodotti di Lavoro 31
2.4.1 Politica di Test 32
2.4.2 Strategia di Test 32
2.4.3 Master Test Plan 34
2.4.4 Level Test Plan 35
2.4.5 Gestione dei Rischi di Progetto 35
2.4.6 Altri Prodotti di Lavoro del Testing 36
2.5 Stima del Test 36
2.6 Definizione e Utilizzo delle Metriche di Test 38
2.7 Valore di Business del Testing 42
2.8 Testing Distribuito, Outsourced o Insourced 43
2.9 Gestire l'Applicazione degli Standard Industriali 44
3. Review (review) – 180 minuti 46
3.1 Introduzione 46
3.2 Management Review e Audit 47
3.3 Gestione delle Review 48
3.4 Metriche per la Review 50
3.5 Gestire le Review Formali 51
4. Defect Management – 150 minuti 52

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

4.1 Introduzione 52
4.2 Il Ciclo di Vita del Difetto e il Ciclo di Vita dello Sviluppo Software 52
4.2.1 Workflow e Stati del Difetto 53
4.2.2 Gestire Defect Report Invalidi e Duplicati 54
4.2.3 Defect Management Cross-Funzionale 54
4.3 Informazioni del Defect Report 54
4.4 Valutare la Capacità del Processo in Base alle Informazioni del Defect Report 56
5. Miglioramento del Processo di Test - 135 minuti 57
5.1 Introduzione 57
5.2 Processo di Miglioramento del Test 58
5.2.1 Introduzione al Process Improvement 58
5.2.2 Tipi di Process Improvement 58
5.3 Migliorare il Processo di Test 59
5.4 Migliorare il Processo di Test con TMMi 60
5.5 Migliorare il Processo di Test con TPI Next 60
5.6 Migliorare il Processo di Test con CTP 61
5.7 Migliorare il Processo di Test con STEP 61
6. Strumenti di Test e Test Automation – 135 minuti 62
6.1 Introduzione 62
6.2 Selezione dello Strumento 62
6.2.1 Strumenti Open Source 62
6.2.2 Strumenti Custom 63
6.2.3 Ritorno sull'Investimento (ROI, Return on Investment) 64
6.2.4 Processo di Selezione 65
6.3 Ciclo di Vita dello Strumento 66
6.4 Metriche dello Strumento 67
7. Competenze delle Persone – Composizione del Team - 210 minuti 68
7.1 Introduzione 68
7.2 Competenze Individuali 69
7.3 Dinamiche del Team di Test 70
7.4 Adattamento del Testing all'Interno di un'Organizzazione 72
7.5 Motivazione 73
7.6 Comunicazione 74
8. Riferimenti 75
8.1 Standard 75
8.2 Documenti ISTQB 75
8.3 Marchi registrati 75
8.4 Libri 75
8.5 Altri Riferimenti 76

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Ringraziamenti

Questo documento (versione 2012) è stato prodotto da un team dedicato appartenente al gruppo di lavoro dell'International Software Testing Qualifications Board Advanced Level: Rex Black (Chair), Judy McKay (Vice Chair), Graham Bath, Debra Friedenberg, Bernard Homes, Paul Jorgensen, Kenji Onishi, Mike Smith, Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Il team desidera ringraziare il gruppo dei reviewer e tutte le organizzazioni nazionali per i suggerimenti e gli input.

Al momento in cui l'Advanced Level Syllabus è stato completato, il gruppo di lavoro Advanced Level era costituto dai seguenti membri (in ordine alfabetico):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Vice Chair), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Chair), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto

Le seguenti persone hanno partecipato alla review, commento e votazione di questo Syllabus:

Chris van Bael, Graham Bath, Kimmo Hakala, Rob Hendriks, Marcel Kwakernaak, Rik Marselis, Don Mills, Gary Mogyorodi, Thomas Mueller, Ingvar Nordstrom, Katja Piroué, Miele Posthuma, Nathalie Rooseboom de Vries, Geoff Thompson, Jamil Wahbeh, Hans Weiberg.

Questo documento è stato formalmente rilasciato dall'Assemblea Generale dell'ISTQB® il 23 Ottobre 2012.

Traduzione italiana: si ringraziano Marco Sogliani, Salvatore Reale e Cristina Sobrero

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

0. Introduzione a questo Syllabus

0.1 Scopo di questo Documento

Questo Syllabus è la base per l'International Software Testing Qualification Advanced Level per Test Manager.
ISTQB® fornisce questo Syllabus con le seguenti modalità:

1. Ai Board nazionali, per consentirne la traduzione nella loro lingua locale e per accreditare i Training Provider. I Board nazionali possono adattare il Syllabus alle loro specifiche esigenze linguistiche e aggiungere i riferimenti bibliografici per adattarlo alle pubblicazioni locali.
2. Agli organismi di certificazione, per consentire la traduzione delle domande d'esame nella loro lingua adattandole agli obiettivi di apprendimento di ogni modulo.
3. Ai Training Provider, per produrre materiale didattico e determinare i metodi d'insegnamento appropriati.
4. Ai candidati alla certificazione, per prepararsi all'esame di certificazione (come parte di un corso di formazione o autonomamente).
5. Alla comunità internazionale di ingegneria dei sistemi e del software, per promuovere la professione del testing del software e di sistema come base di riferimento per libri e articoli.

ISTQB® permette inoltre ad altre entità di usare questo Syllabus per altre finalità, a condizione che richiedano e ottengano autorizzazione scritta da parte dello stesso ISTQB®.

0.2 Panoramica

La certificazione Advanced Level Core si compone di tre Syllabi separati relativi ai seguenti ruoli:

- Test Manager
- Test Analyst
- Technical Test Analyst

ISTQB® Advanced Level Overview [ISTQB_AL_OVIEW] è un documento separato che include le seguenti informazioni:

- Business Outcome per ogni Syllabus
- Riassunto di ogni Syllabus
- Relazioni fra Syllabi
- Descrizione dei Livelli di Conoscenza (Livelli K)
- Appendici

0.3 Obiettivi di Apprendimento Esaminabili

Gli Obiettivi di Apprendimento supportano i Business Outcome e vengono utilizzati per creare gli esami per il conseguimento della certificazione Advanced Level Test Manager. Tutti i paragrafi di questo Syllabus sono esaminabili a livello K1. Questo significa che il candidato dovrà riconoscere, ricordare e richiamare un termine o un concetto. Gli obiettivi di apprendimento per i livelli K2, K3 e K4 sono riportati all'inizio di ogni relativo capitolo.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

1. Processo di Test – 420 minuti

Parole Chiave:

Chiusura dei test, condizioni di test, controllo dei test, criteri di uscita, esecuzione dei test, implementazione dei test, pianificazione dei test, test procedure, progettazione dei test, test case, test log, test report, test script.

Obiettivi di Apprendimento per il Processo di Test

1.2 Pianificazione dei Test, Monitoraggio e Controllo dei Test

TM-1.2.1 (K4) Analizzare le esigenze di test per un sistema al fine di pianificare le attività di test e i prodotti di lavoro che permetteranno di raggiungere gli obiettivi di test.

1.3 Analisi dei Test

TM-1.3.1 (K3) Utilizzare la tracciabilità per verificare la completezza e la consistenza delle condizioni di test rispetto agli obiettivi di test, alla strategia di test e al test plan.

TM-1.3.2 (K2) Spiegare i fattori che potrebbero influenzare il livello di dettaglio in cui le condizioni di test possono essere specificate, nonché i vantaggi e gli svantaggi di specificare le condizioni di test a determinati livelli di dettaglio.

1.4 Progettazione dei Test

TM-1.4.1 (K3) Utilizzare la tracciabilità per verificare la completezza e la consistenza dei test case progettati rispetto alle condizioni di test definite.

1.5 Implementazione dei Test

TM-1.5.1 (K3) Utilizzare i rischi, l'assegnazione delle priorità, l'ambiente di test, le dipendenze dai dati e i vincoli per sviluppare una schedulazione di esecuzione dei test, che sia completa e consistente rispetto agli obiettivi di test, alla strategia di test e al test plan.

1.6 Esecuzione dei Test

TM-1.6.1 (K3) Usare la tracciabilità per monitorare l'avanzamento dei test per la completezza e la consistenza rispetto agli obiettivi di test, alla strategia di test e al test plan.

1.7 Valutazione dei Criteri di Uscita e Reporting

TM-1.7.1 (K2) Spiegare l'importanza della raccolta di informazioni accurata e tempestiva durante il processo di test per supportare un reporting accurato e una valutazione dei criteri di uscita.

1.8 Attività di Chiusura dei Test

TM-1.8.1 (K2) Riassumere i quattro gruppi delle attività di chiusura dei test.

TM-1.8.2 (K3) Implementare una retrospettiva di progetto per valutare i processi e identificare aree di miglioramento

1.1 Introduzione

Il Syllabus ISTQB® Foundation Level descrive un processo di test fondamentale che comprende le seguenti attività:

- Pianificazione e controllo
- Analisi e Progettazione

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Implementazione ed Esecuzione
- Valutazione dei criteri di uscita e Reporting
- Attività di chiusura dei test

Il Syllabus Foundation Level indica che, sebbene logicamente sequenziali, le attività del processo possono sovrapporsi ed essere eseguite in modo concorrente. Normalmente viene richiesto di adattare (tailoring) queste attività all'interno del contesto del sistema e del progetto.

Per il Syllabus Advanced Level alcune di queste attività sono considerate separatamente, al fine di fornire ulteriore affinamento e ottimizzazione dei processi, che meglio si adattino con al ciclo di vita dello sviluppo software, e facilitare un efficace monitoraggio e controllo dei test.

Le attività di test sono perciò considerate come segue:

- Pianificazione, monitoraggio e controllo
- Analisi
- Progettazione
- Implementazione
- Esecuzione
- Valutazione dei criteri di uscita e Reporting
- Attività di chiusura dei test

1.2 Pianificazione dei Test, Monitoraggio e Controllo dei Test

Questo paragrafo si focalizza sui processi di pianificazione, monitoraggio e controllo dei test.

Come discusso nel Foundation Level, queste sono attività previste per il ruolo di Test Manager.

1.2.1 Pianificazione dei Test

Per ogni livello di test, la pianificazione dei test inizia quando parte il processo di test per quel livello, e continua durante tutto il progetto fino al completamento delle attività di chiusura per quel livello. Questo comprende l'identificazione delle attività e delle risorse necessarie per soddisfare la mission e gli obiettivi identificati nella strategia di test. La pianificazione dei test include anche l'identificazione dei metodi di raccolta e di tracciamento delle metriche che verranno utilizzate per guidare il progetto, per determinare l'aderenza al piano e per valutare il raggiungimento degli obiettivi. Per poter determinare in fase di pianificazione le metriche utili, possono essere selezionati gli strumenti, può essere schedulata la formazione e possono essere stabilite le linee guida della documentazione.

Le strategie selezionate per il progetto di testing aiutano a determinare i compiti che si dovrebbero svolgere durante la fase di pianificazione. Per esempio, quando si utilizza la strategia di test basata sul rischio (si veda capitolo 2), l'analisi del rischio viene utilizzata per guidare il processo di pianificazione dei test relativamente alle attività di mitigazione necessarie per ridurre i rischi di prodotto identificati e per supportare la pianificazione della contingency. Se viene rilevato un insieme di difetti potenziali probabili e critici relativi alla sicurezza, un notevole effort dovrebbe essere speso nella progettazione ed esecuzione dei test di sicurezza. Allo stesso modo, se i difetti più critici vengono rilevati normalmente nelle specifiche di progettazione, il processo di pianificazione dei test potrebbe prevedere ulteriore testing statico (review) delle specifiche di progettazione.

Le informazioni sui rischi possono essere anche utilizzate per determinare le priorità delle diverse attività di test. Per esempio, quando le prestazioni del sistema sono ad alto rischio, il performance testing può essere svolto non appena è disponibile il codice integrato. Allo stesso modo, se occorre applicare una strategia reattiva, può essere garantita la pianificazione della creazione di Test Charter e l'adozione di strumenti per le tecniche di test dinamico, come il testing esplorativo.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Inoltre, la fase di pianificazione dei test è quella dove il Test Manager definisce chiaramente l'approccio al testing, che include quali livelli di test saranno utilizzati, lo scopo e gli obiettivi di ogni livello, e le tecniche di test utilizzate in ogni livello. Ad esempio, nel testing basato sul rischio dei sistemi avionici, una valutazione del rischio prescrive quale livello di copertura del codice sia richiesto e, quindi, quali tecniche di test dovrebbero essere utilizzate.

Possono esistere relazioni complesse tra la base di test (p.e., requisiti o rischi specifici), le condizioni di test e i test case che li coprono. Spesso esistono relazioni molti-a-molti tra questi prodotti di lavoro, che devono essere comprese sia per consentire un'efficace implementazione della pianificazione e del monitoraggio e controllo dei test, sia per definire e decidere l'adozione degli strumenti di test.

Le relazioni possono esistere anche tra i prodotti lavoro generati dal team di sviluppo e quelli creati dal team di test. Ad esempio, la matrice di tracciabilità può essere necessaria per tracciare le relazioni tra gli elementi nella specifica di progettazione dettagliata dai progettisti di sistema, i requisiti di business definiti dai business analyst, e i prodotti di lavoro definiti dal team di test. Se devono essere progettati e utilizzati test case di basso livello, può esistere un requisito, definito in fase di pianificazione, per cui i documenti di progettazione dettagliata definiti dal team di sviluppo debbano essere approvati prima di iniziare la progettazione dei test case. In uno sviluppo Agile, sessioni informali di trasferimento dell'informazione possono essere utilizzate per scambiare informazioni tra i diversi team prima dell'inizio del testing.

Il Test Plan può anche elencare le funzionalità specifiche del software che rientrano nel suo ambito (in base all'analisi del rischio, se appropriata), così come può esplicitamente identificare le funzionalità che non rientrano nel suo ambito. In base ai livelli di formalismo e di documentazione appropriati al progetto, ogni funzionalità che sia in ambito può essere associata ad una corrispondente specifica di progettazione dei test.

In questa fase può anche essere richiesto al Test Manager di lavorare con gli architetti di progetto per definire le specifiche iniziali dell'ambiente di test, per verificare la disponibilità delle risorse richieste, per garantire che le persone che configureranno l'ambiente siano impegnate a farlo, e per comprendere i costi / tempi di consegna che il lavoro richiesto per completare e rilasciare l'ambiente di test.

Infine, dovrebbero essere identificate tutte le dipendenze esterne e i relativi Service Level Agreement (SLA) e, se richiesto, dovrebbe essere preso un contatto iniziale. Esempi di dipendenze sono le richieste di risorse di team esterni, dipendenze con altri progetti (se si lavora all'interno di un programma), con fornitori o partner esterni per lo sviluppo, con il team di rilascio e con i database administrator.

1.2.2 Monitoraggio e Controllo dei Test

Affinché un Test Manager possa fornire un efficace controllo dei test, deve essere stabilito un framework per la schedulazione e il monitoraggio del testing, che consentano di poter tracciare lo stato di avanzamento dei prodotti di lavoro e delle risorse di test rispetto al piano. Questo framework dovrebbe includere le misure e gli obiettivi dettagliati, necessari per mettere in correlazione lo stato delle attività e dei prodotti di lavoro del test rispetto al piano ed agli obiettivi strategici.

Per progetti di piccole dimensioni e meno complessi, può essere relativamente facile mettere in relazione i prodotti di lavoro e le attività di test con il piano e gli obiettivi strategici, ma generalmente devono essere definiti obiettivi più dettagliati per raggiungere questo risultato. Questo può includere misure e obiettivi per raggiungere gli obiettivi del test e la copertura della base di test.

Di particolare importanza è la necessità di mettere in relazione lo stato dei prodotti di lavoro e delle attività con la base di test, in modo che sia comprensibile e rilevante per gli stakeholder di business e di progetto.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

La definizione degli obiettivi e la misura dell'avanzamento in base alle condizioni di test e ai gruppi di condizioni di test possono essere utilizzati come mezzi per raggiungere questo, mettendo in relazione altri prodotti di lavoro del test con la base di test, attraverso le condizioni di test. Una tracciabilità configurata in modo appropriato, inclusa la capacità di riferire sullo stato della tracciabilità stessa, rende più trasparenti e comprensibili le complesse relazioni che esistono tra i prodotti del lavoro dello sviluppo, la base di test e i prodotti di lavoro del test.

A volte, le misure dettagliate e gli obiettivi richiesti dagli stakeholder per essere monitorati, non si riferiscono direttamente a funzionalità del sistema o a una sua specifica, specialmente se la documentazione formale è carente o addirittura assente. Ad esempio, uno stakeholder di business può essere più interessato a raggiungere una copertura rispetto ad un ciclo di business operativa, anche se la specifica è definita in termini di funzionalità del sistema. Il coinvolgimento di stakeholder di business in una fase iniziale di un progetto può aiutare a definire tali misure e obiettivi, che non solo possono essere utilizzati per aiutare a fornire un migliore controllo nel corso del progetto, ma possono anche aiutare a guidare e influenzare le attività di test durante il progetto. Ad esempio, le misure e gli obiettivi degli stakeholder possono portare a strutturare i prodotti di lavoro della progettazione e dell'implementazione dei test, e/o la schedulazione dell'esecuzione dei test, per facilitare un monitoraggio accurato dell'avanzamento del testing rispetto a queste misure. Questi obiettivi possono inoltre aiutare a fornire la tracciabilità per uno specifico livello di test e hanno il potenziale per aiutare a fornire la tracciabilità delle informazioni attraverso i diversi livelli di test.

Il controllo dei test è un'attività continuativa, che confronta l'avanzamento attuale rispetto al piano e implementa azioni correttive quando necessarie. Il controllo dei test guida il testing in modo da soddisfare la mission, le strategie e gli obiettivi, inclusa la rivisitazione delle attività di pianificazione dei test quando necessario. Le reazioni appropriate ai dati di controllo dipendono dalle informazioni della pianificazione di dettaglio.

Il contenuto dei documenti di pianificazione dei test e delle attività di controllo dei test sono trattate nel Capitolo 2.

1.3 Analisi dei Test

Piuttosto che considerare insieme l'analisi e la progettazione dei test, come descritto nel Syllabus Foundation Level, i Sillabi Advanced Level li considerano come attività separate, pur riconoscendo che possono essere implementate come attività parallele, integrate o iterative per facilitare la realizzazione dei prodotti di lavoro della progettazione dei test.

L'analisi dei test è l'attività che definisce "cosa" deve essere testato, sotto forma di condizioni di test. Le condizioni di test possono essere identificate dall'analisi della base di test, degli obiettivi di test e dei rischi di prodotto. Questi possono essere visti come misure e obiettivi dettagliate per il successo (p.e., come parte dei criteri di uscita) e dovrebbero essere tracciabili rispetto alla base di test e agli obiettivi strategici definiti, compresi gli obiettivi del test e gli altri criteri di successo del progetto o degli stakeholder. Le condizioni di test dovrebbero essere tracciabili anche rispetto ai test progettati e agli altri prodotti di lavoro di test, non appena questi sono stati creati.

L'analisi dei test per un dato livello può essere eseguita non appena la base per il testing è stata definita per quel livello. Tecniche di test formali e altre tecniche analitiche generali (p.e., strategie analitiche basate sul rischio e strategie analitiche basate sui requisiti) possono essere utilizzate per identificare le condizioni di test. Le condizioni di test possono o non possono specificare valori o variabili in base al livello di test, alle informazioni disponibili al momento di effettuare l'analisi e al livello di dettaglio selezionato (cioè il grado di granularità della documentazione).

Esistono diversi fattori da considerare quando si decide il livello di dettaglio con cui specificare le condizioni di test, che includono:

- Livello di test
- Livello di dettaglio e qualità della base di test

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Complessità del Sistema / Software
- Rischi di progetto e di prodotto
- Relazioni tra le basi di test, quello che deve essere testato e come deve essere testato
- Ciclo di sviluppo del software in uso
- Strumento di test management utilizzato
- Livello di come la progettazione dei test e degli altri prodotti di lavoro del test devono essere specificati e documentati
- Competenze e conoscenze dei Test Analyst
- Livello di maturità del processo di test e dell'organizzazione stessa (si noti che una maturità più alta può richiedere maggiore livello di dettaglio, o permettere un minore livello di dettaglio)
- Disponibilità di altri stakeholder di progetto per la consultazione

Specificare le condizioni di test in modo dettagliato porterà a un numero maggiore di condizioni di test. Ad esempio, potrebbe esistere una singola condizione di test generale, "Test del checkout", per un applicativo e-commerce. Successivamente, in un documento dettagliato delle condizioni di test, questa potrebbe essere suddivisa in diverse condizioni di test, con una condizione per ognuno dei metodi di pagamento supportati, una condizione per ogni possibile nazione destinataria, e così via.

Alcuni vantaggi nello specificare le condizioni di test a un livello dettagliato includono:

- Facilita una maggiore flessibilità nel relazionare gli altri prodotti di lavoro del test (p.e., i test case) con la base di test e gli obiettivi di test, fornendo al Test Manager un migliore e più dettagliato monitoraggio e controllo
- Contribuisce alla prevenzione dei difetti, come evidenziato nel Foundation Level, svolgendo l'attività nelle fasi iniziali di un progetto per i livelli più alti di test, non appena la base di test è stata definita e potenzialmente prima che l'architettura di sistema e la progettazione di dettaglio siano disponibili
- Mette in relazione i prodotti di lavoro del test con gli stakeholder, con modalità che questi ultimi sono in grado di comprendere (spesso test case e altri prodotti di lavoro non sono significativi per gli stakeholder di business; semplici metriche, come il numero di test case eseguiti, non sono significative per la copertura dei requisiti degli stakeholder)
- Aiuta a dirigere e influenzare non solo le altre attività di test, ma anche le altre attività di sviluppo
- Facilita la progettazione, l'implementazione e l'esecuzione dei test, insieme ai risultanti prodotti di lavoro, per essere ottimizzati da una più efficiente copertura di misure e obiettivi dettagliati
- Fornisce la base per una tracciabilità orizzontale più chiara all'interno di un livello di test

Alcuni svantaggi nello specificare le condizioni di test a livello dettagliato includono:

- Potenzialmente time-consuming
- La manutenibilità può diventare difficile in un ambiente che cambia frequentemente
- Il livello di formalità richiede che siano definiti e implementati dal team

La specifica delle condizioni di test dettagliate può essere particolarmente efficace nelle seguenti situazioni:

- Si utilizzano metodi di documentazione "leggera" ("lightweight") della progettazione dei test, come checklist, per adattarsi al ciclo di sviluppo, a vincoli di tempo e/o costi, o per altri fattori
- Siano disponibili, come base di test, pochi o nessun requisito formale o altro prodotto di lavoro dello sviluppo
- Il progetto è complesso, di ampie dimensioni o di alto rischio, e richiede un livello di monitoraggio e controllo, che non può essere soddisfatto semplicemente correlando i test case ai prodotti di lavoro dello sviluppo.

Le condizioni di test possono essere specificate con un dettaglio minore, quando la base di test può essere facilmente e direttamente correlata ai prodotti di lavoro di progettazione dei test. Questo è più probabile nel caso di:

- Testing di componente

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Progetti meno complessi, in cui esistono semplici relazioni gerarchiche tra quello che deve essere testato e come deve essere testato
- Testing di accettazione, dove gli use case possono essere utilizzati per aiutare nella definizione dei test case

1.4 Progettazione dei Test

La progettazione dei test è l'attività che definisce "come" qualcosa debba essere testato. Questo comporta l'identificazione dei test case attraverso l'elaborazione graduale delle condizioni di test o delle basi di test identificate, utilizzando tecniche di test definite nella strategia e/o nel test plan.

In base agli approcci utilizzati per il monitoraggio e controllo e la tracciabilità dei test, i test case possono essere correlati direttamente (o indirettamente attraverso le condizioni di test) alla base di test e agli obiettivi definiti. Questi obiettivi includono obiettivi strategici, obiettivi di test e altri criteri di successo di progetto o degli stakeholder.

La progettazione dei test per un determinato livello di test può essere eseguita dopo aver identificato le condizioni di test e quando sono disponibili informazioni sufficienti a consentire la produzione di test case di basso o alto livello, in base all'approccio utilizzato per la progettazione dei test. Per i livelli di test più alti, è più probabile che la progettazione dei test sia un'attività separata successiva alla precedente analisi dei test. Per i livelli più bassi di test, è probabile che l'analisi e la progettazione dei test vengano svolte come attività integrate.

È anche probabile che alcune attività che normalmente sono svolte durante l'implementazione dei test siano integrate nel processo di progettazione dei test, quando si utilizza un approccio iterativo di creazione dei test richiesti per l'esecuzione, ad esempio la creazione dei dati di test. Infatti questo approccio può ottimizzare la copertura delle condizioni di test, creando durante il processo sia test case di basso che di alto livello.

1.5 Implementazione dei Test

L'implementazione dei test è l'attività durante la quale i test sono organizzati e prioritizzati dai Test Analyst. In contesti formalmente documentati, l'implementazione dei test è l'attività in cui la progettazione dei test viene implementata come test case concreti, procedure di test e dati di test. Alcune organizzazioni che seguono lo standard IEEE 829 [IEEE829] definiscono gli input e i risultati attesi associati ai test case nelle specifiche dei test case, e i passi operativi nelle specifiche delle procedure di test. Più comunemente, gli input, i risultati attesi e i passi dei test sono documentati insieme. L'implementazione dei test può anche includere la creazione dei dati di test memorizzati (p.e., i file o le tabelle di database).

L'implementazione dei test prevede anche verifiche finali per assicurare che il team di test sia pronto per l'esecuzione dei test. Le verifiche potrebbero includere la garanzia di consegna dell'ambiente di test richiesto, dei dati di test e del codice (possibilmente eseguendo alcuni test di accettazione dell'ambiente di test e/o del codice) e la verifica che tutti i test case siano stati scritti, sottoposti a review e siano pronti per essere eseguiti. Questo può includere anche la verifica dei criteri di ingresso (espliciti e impliciti) per il livello di test considerato (si veda il paragrafo 1.7). L'implementazione dei test può richiedere anche lo sviluppo di una descrizione dettagliata dei dati di test e dell'ambiente di test.

Il livello di dettaglio e la associata complessità del lavoro svolto durante l'implementazione dei test possono essere influenzati dal dettaglio dei prodotti di lavoro del test (p.e., test case e condizioni di test). In alcuni casi, specialmente dove i test case devono essere archiviati per un riutilizzo a lungo termine nel testing di regressione, i test possono fornire descrizioni dettagliate dei passi necessari per eseguire un test, per garantire un'esecuzione affidabile e consistente indipendentemente dal tester chelo esegue. Se si applicano delle normative, i test dovrebbero fornire la prova della conformità alle norme applicabili (si veda il paragrafo 2.9).

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Durante l'implementazione dei test, l'ordine in cui i test manuali e automatizzati devono essere eseguiti dovrebbe essere incluso in una schedulazione dell'esecuzione dei test. I Test Manager dovrebbero controllare attentamente i vincoli, compresi i rischi e le priorità, che potrebbero richiedere l'esecuzione dei test in un ordine particolare o su apparati particolari. Le dipendenze rispetto all'ambiente di test o rispetto ai dati di test devono essere conosciute e verificate.

Esistono alcuni svantaggi in una implementazione dei test anticipata. Nello sviluppo Agile, ad esempio, il codice può cambiare pesantemente da iterazione a iterazione, rendendo obsoleto molto del lavoro di implementazione svolto. Anche in assenza di un ciclo di vita orientato al cambiamento come quello Agile, qualsiasi ciclo di vita iterativo o incrementale può causare modifiche significative ad ogni iterazione, rendendo i test script inaffidabili o soggetti ad elevati sforzi di manutenzione. Lo stesso si verifica nei cicli di vita sequenziali non gestiti adeguatamente, dove i requisiti cambiano frequentemente, anche in fasi avanzate del progetto. Prima di dedicare un effort consistente nell'implementazione di un test esteso, è consigliabile comprendere le caratteristiche del ciclo di vita dello sviluppo software e la possibilità di prevedere quali funzionalità software saranno disponibili per il testing.

Esistono alcuni vantaggi nell'implementazione anticipata dei test. Ad esempio, i test concreti forniscono esempi di come il software dovrebbe comportarsi, se scritto in accordo alla base di test. Esperti di dominio di business possono probabilmente trovare la verifica di test case concreti più semplice rispetto alla verifica di regole astratte di business, e quindi possono individuare ulteriori punti di debolezza nelle specifiche software. Tali test verificati possono fornire evidenze illuminanti del comportamento richiesto per i progettisti e gli sviluppatori software.

1.6 Esecuzione dei Test

L'esecuzione dei test inizia quando l'oggetto del test è stato consegnato e i criteri di ingresso per l'esecuzione dei test sono stati soddisfatti. I test dovrebbero essere progettati o almeno definiti prima della loro esecuzione. Dovrebbero essere installati strumenti, in particolare per il test management, il tracciamento dei difetti e (se applicabile) il test automation. Il tracciamento dei risultati dei test, che include il tracciamento delle metriche, dovrebbe essere attivo, e i dati tracciati dovrebbero essere compresi da tutti i membri del team. Dovrebbero essere disponibili e pubblicati gli standard per il logging dei test per il reporting dei difetti. Una volta assicurato che questi elementi siano attivi prima dell'esecuzione dei test, l'esecuzione può svolgersi in modo efficiente.

I test dovrebbero essere eseguiti in base ai test case, anche se il Test Manager dovrebbe consentire una certa flessibilità in modo che il tester possa coprire interessanti scenari aggiuntivi di test e comportamenti osservati durante il testing. Quando si segue una strategia di test almeno in parte reattiva, dovrebbe essere riservato del tempo per sessioni di test che utilizzino tecniche basate sull'esperienza e basate sui difetti. Naturalmente, qualsiasi failure rilevato durante tale testing senza script predeterminati (testing unscripted) deve descrivere le variazioni dal test case documentato, in modo da consentire di riprodurre il failure. I test automatizzati seguiranno le relative istruzioni definite senza deviazioni.

Il ruolo principale di un Test Manager durante l'esecuzione dei test è di monitorare l'avanzamento in base al test plan e, se richiesto, di avviare e implementare azioni di controllo per guidare il testing verso una conclusione positiva in termini di mission, obiettivi e strategia. Per fare questo, il Test Manager può utilizzare la tracciabilità dei risultati di test rispetto alle condizioni di test, alla base di test e infine, agli obiettivi di test, e anche dagli obiettivi di test verso i risultati del test. Questo processo viene descritto in dettaglio nel paragrafo 2.6.

1.7 Valutazione dei Criteri di Uscita e Reporting

La documentazione e il reporting per il monitoraggio e il controllo dell'avanzamento dei test sono discussi in dettaglio nel paragrafo 2.6.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Dal punto di vista del processo di test, è importante garantire che siano in atto processi efficaci nel fornire le informazioni necessarie a valutare i criteri di uscita e il reporting.

La definizione dei requisiti e dei metodi di raccolta delle informazioni sono parte della pianificazione dei test, del monitoraggio e controllo dei test. Durante l'analisi, la progettazione, l'implementazione e l'esecuzione dei test, il Test Manager dovrebbe garantire che i membri del team di test responsabili di tali attività stiano fornendo le informazioni richieste in modo accurato e tempestivo in modo da facilitare un'efficace valutazione e reporting.

La frequenza e il livello di dettaglio richiesto per il reporting dipendono dal progetto e dall'organizzazione. Questi dovrebbero essere negoziati durante la fase di pianificazione dei test e dovrebbero includere la consultazione con gli stakeholder rilevanti del progetto.

1.8 Attività di Chiusura dei Test

Una volta che è stato determinato che l'esecuzione dei test è completata, gli output principali dovrebbero essere raccolti, successivamente forniti alla persona rilevante oppure archiviati. Complessivamente, queste sono le attività di chiusura dei test, che si dividono in quattro gruppi principali:

1. Verifica del completamento dei test – assicurare che tutto il lavoro del test sia davvero concluso. Ad esempio, tutti i test pianificati dovrebbero essere eseguiti o deliberatamente non eseguiti, e tutti i difetti conosciuti dovrebbero essere corretti con verificati con il test confermativo, posticipati in una futura release, oppure accettati come restrizioni permanenti.
2. Rilascio degli artefatti di test - consegnare i prodotti di lavoro a chi ne ha bisogno. Ad esempio, i difetti noti differiti o accettati dovrebbero essere comunicati a coloro che utilizzeranno e forniranno supporto all'utilizzo del sistema. Gli ambienti di test e i test dovrebbero essere consegnati ai responsabili del testing di manutenzione. L'insieme dei regression test (sia automatizzato sia manuale) dovrebbe essere documentato e rilasciato al gruppo di manutenzione.
3. Lessons Learned – Organizzare o partecipare a riunioni Retrospettive in cui possono essere documentate importanti lessons learned (sia all'interno del progetto di test sia durante il ciclo di vita dello sviluppo software). In queste riunioni possono essere predisposti piani per assicurare che le buone pratiche possano essere ripetute e che le pratiche inadeguate non siano ripetute o, nel caso in cui i problemi non possano essere risolti, che siano inseriti nei piani di progetto per un'analisi futura. Le aree da considerare includono:
    a. La rappresentazione utente nelle sessioni di analisi dei rischi di qualità è stata sufficientemente ampia? Ad esempio, a causa della tardiva scoperta di cluster di difetti imprevisti, il team potrebbe aver scoperto che una rappresentazione più ampia di rappresentanti utente dovrebbe partecipare in progetti futuri a sessioni di analisi dei rischi di qualità.
    b. Le stime sono state accurate? Ad esempio, le stime possono essere state calcolate in modo errato e quindi le stime future dovranno tener conto delle problematiche evidenziate con le ragioni del problema come, ad esempio, che il testing è stato inefficiente o che la stima attuale è stata inferiore a quella che avrebbe dovuto essere.
    c. Quali sono i trend e i risultati dell'analisi causa ed effetto dei difetti? Ad esempio, valutare se change request tardive hanno influito sulla qualità dell'analisi e dello sviluppo, cercare trend che indichino cattive pratiche, (ad esempio, avendo la percezione di risparmiare tempo, saltare un livello di test che avrebbe rilevato difetti in anticipo e/o in modo più economico), Verificare se il trend dei difetti potrebbe identificare aree relative a nuove tecnologie, cambiamenti di personale, o carenza di competenze.
    d. Esistono potenziali opportunità di process improvement?
    e. Esiste una qualsiasi variazione imprevista dal piano che dovrebbe essere adattata in una pianificazione futura?
4. Archiviare i risultati, i log, i report e altri documenti e prodotti di lavoro nel sistema di Configuration Management. Ad esempio, il test plan e il piano di progetto dovrebbero essere

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

memorizzati in un archivio di pianificazione, con un collegamento chiaro al sistema e alla versione utilizzati.

Queste attività sono importanti, spesso omesse, e dovrebbero essere esplicitamente incluse come parte del test plan. È comune l'omissione di una o più di queste attività, di solito a causa della prematura riassegnazione o congedo dei membri del team di progetto, di pressioni sulle risorse o sulle schedulazioni dei progetti futuri, o di scioglimento del team. Nei progetti realizzati sotto contratto, come uno sviluppo custom, il contratto dovrebbe specificare i compiti richiesti.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

2. Test Management – 750 minuti

Parole Chiave:

Analisi del rischio, approccio di test, condizione di test, controllo del test, identificazione del rischio, level test plan, livello di rischio, livello di test, master test plan, mitigazione del rischio, monitoraggio dei test, test plan, politica di test, rischio, rischio di prodotto, rischio di progetto, rischio di qualità, risk management, stima del test, strategia di test, testing basato sul rischio, valutazione del rischio, test director, test leader, test management, Wide Band Delphi

Obiettivi di Apprendimento per il Test Management

2.2 Test Management nel Contesto

TM-2.2.1 (K4) Analizzare gli stakeholder, le circostanze e le esigenze di un progetto o programma software, considerando il modello del ciclo di vita dello sviluppo software, e identificare le attività di test ottimali
TM-2.2.2 (K2) Capire come le attività del ciclo di vita dello sviluppo software e i relativi prodotti di lavoro influenzino il testing, e come il testing influenzi le attività del ciclo di vita dello sviluppo software e i prodotti di lavoro
TM-2.2.3 (K2) Spiegare come gestire i problemi di test management con il testing basato sull'esperienza e con il testing non-funzionale

2.3 Testing basato sul Rischio e altri Approcci per la Prioritizzazione dei Test e l'Allocazione dell'Effort

TM-2.3.1 (K2) Spiegare le diverse modalità con cui il testing basato sul rischio risponde ai rischi
TM-2.3.2 (K2) Spiegare, attraverso esempi, le diverse tecniche per l'analisi del rischio di prodotto
TM-2-3.3 (K4) Analizzare, identificare e valutare i rischi di qualità del prodotto, riassumendo i rischi e i relativi livelli di rischio, valutati dalla prospettiva degli stakeholder chiave di progetto
TM-2-3.4 (K2) Descrivere come i rischi di qualità del prodotto identificati possano essere mitigati e gestiti, in base al livello di rischio valutato, durante il ciclo di vita e il processo di test
TM-2.3.5 (K2) Fornire esempi di diverse opzioni per la selezione dei test, la prioritizzazione dei test e l'allocazione dell'effort.

2.4 Documentazione di Test e altri Prodotti di Lavoro

TM-2.4.1 (K4) Analizzare esempi di politiche di test e strategie di test, e creare master test plan, level test plan e altri prodotti di lavoro del test che siano completi e consistenti con questi documenti
TM-2.4.2 (K4) Per un determinato progetto, analizzare i rischi di progetto e selezionare le più appropriate opzioni di risk management (ad esempio, mitigazione, contingenza, trasferimento e/o accettazione)
TM-2.4.3 (K2) Descrivere, fornendo esempi, come le strategie di test influenzano le attività di test
TM-2.4.4 (K3) Definire le regole e i template della documentazione per i prodotti di lavoro del test, che si adegueranno all'organizzazione, al ciclo di vita e alle esigenze del progetto, adattando i template disponibili standard, dove applicabili.

2.5 Stima del Test

TM-2.5.1 (K3) Per un determinato progetto, creare una stima di tutte le attività del processo di test, utilizzando tutte le tecniche di stima applicabili
TM-2.5.2 (K2) Comprendere e fornire esempi dei fattori che possono influenzare le stime del test

2.6 Definizione e Utilizzo delle Metriche di Test

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

TM-2.6.1 (K2) Descrivere e confrontare le tipiche metriche relative al testing
TM-2.6.2 (K2) Confrontare le diverse dimensioni del monitoraggio dell'avanzamento dei test
TM-2.6.3 (K4) Analizzare e documentare i risultati dei test (report) in termini di rischio residuo, stato dei difetti, stato di esecuzione dei test, stato di copertura dei test, e confidenza, per fornire indicazioni e raccomandazioni che consentano agli stakeholder di progetto di prendere decisioni sul rilascio

2.7 Valore di Business del Testing

TM-2.7.1 (K2) Fornire esempi per ognuna delle quattro categorie che determinano i costi della qualità
TM-2.7.2 (K3) Stimare il valore del testing in base ai costi della qualità, insieme ad altre considerazioni di tipo quantitativo e qualitativo, e comunicare il valore stimato agli stakeholder del testing

2.8 Test Distribuito, Outsourced e Insourced

TM-2.8.1 (K2) Comprendere i fattori richiesto per l'utilizzo efficace delle strategie (distribuite, outsourced e insourced) di staffing del team di test

2.9 Gestire l'Applicazione degli Standard Industriali

TM-2.9.1 (K2) Riassumere le fonti e gli usi degli standard per il testing del software

2.1 Introduzione

A Livello Advanced, ha iniziato a diffondersi una specializzazione nella carriera dei professionisti del test. Questo capitolo si focalizza sulle aree di conoscenza richieste ai professionisti del test che si orientano verso posizioni di Test Leader, Test Manager e Test Director. In questo Syllabus chiameremo collettivamente tutti questi professionisti Test Manager, comprendendo che differenti organizzazioni avranno differenti definizioni per i titoli e i livelli di responsabilità delle persone in tali posizioni.

2.2 Test Management nel Contesto

Una delle responsabilità fondamentali di un manager è proteggere e utilizzare le risorse (persone, software, hardware, infrastrutture, ecc.) per eseguire dei processi che forniscano valore aggiunto. Per gli IT Manager e software Manager i processi sono spesso parte di un progetto o di un programma volti a rilasciare un software o un sistema per uso interno o esterno. Per i Test Manager, i processi sono quelli coinvolti nel testing, in particolare le attività del processo fondamentale di test descritte nel livello Foundation e nel capitolo 1 del presente Syllabus. Poiché i processi di test forniscono valore solo contribuendo al successo complessivo del programma o progetto (o prevenendo tipi più gravi di failure), il Test Manager deve di conseguenza pianificare e controllare i processi di test. In altre parole, il Test Manager deve organizzare in modo appropriato i processi di test, includendo le attività e i prodotti di lavoro associati, in base alle esigenze e circostanze di altri stakeholder, alle loro attività (p.e., il ciclo di vita dello sviluppo software in cui avviene il testing), e ai loro prodotti di lavoro (p.e., le specifiche dei requisiti).

2.2.1 Capire gli Stakeholder del Testing

Una persona è uno stakeholder del testing quando ha un interesse nelle attività di test, nei prodotti di lavoro del test o nella qualità del sistema finale o di un rilascio. L'interesse dello stakeholder può essere un coinvolgimento diretto o indiretto nelle attività di test, la ricezione diretta o indiretta dei prodotti di lavoro del test o un effetto diretto o indiretto della qualità dei rilasci prodotti dal programma o dal progetto.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Gli stakeholder del testing variano a seconda del progetto, del prodotto, dell'organizzazione e di altri fattori, e possono includere i seguenti ruoli:

- Sviluppatori, team leader e responsabili dello sviluppo. Questi stakeholder implementano il software da testare, ricevono i risultati dei test e spesso devono (re)agire in base a tali risultati (ad es. correggere i difetti segnalati).
- Architetti di database, architetti di sistema e progettisti. Questi stakeholder progettano il software, ricevono i risultati dei test e spesso devono (re)agire in base a tali risultati.
- Business analyst e marketing analyst. Questi stakeholder definiscono le funzionalità e il livello di qualità relativo a tali funzionalità, che devono essere presenti nel software. Sono spesso coinvolti nella definizione della necessaria copertura dei test, nella review dei risultati dei test e nel prendere decisioni in base ai risultati dei test.
- Senior management, product manager e sponsor del progetto. Questi stakeholder sono spesso coinvolti nella definizione della necessaria copertura dei test, nella review dei risultati dei test nel prendere decisioni in base ai risultati dei test.
- Project manager. Questi stakeholder sono responsabili di gestire con successo i loro progetti, che richiede di bilanciare le priorità di qualità, schedulazione, funzionalità e budget. Spesso forniscono le risorse necessarie per le attività di test e collaborano con il Test Manager nella pianificazione e controllo dei test.
- Supporto tecnico, assistenza clienti e staff di Help Desk. Questi stakeholder supportano gli utenti e i clienti che traggono beneficio dalle funzionalità e dalla qualità del software rilasciato.
- Utenti diretti e indiretti. Questi stakeholder utilizzano il software direttamente (cioè, sono gli utenti finali) oppure ricevono output e/o servizi prodotti o supportati dal software.

Per ulteriori informazioni sugli stakeholder del testing, si veda il capitolo 2 di [Goucher09].

L'elenco degli stakeholder non è completo. Il Test Manager deve identificare gli stakeholder del testing specifici per il proprio progetto o programma: IL Test Manager deve anche comprendere l'esatta natura della relazione degli stakeholder con il testing, e come il team di test risponde alle loro esigenze. Oltre a identificare gli stakeholder, come descritto sopra, il Test Manager dovrebbe identificare le altre attività del ciclo di vita dello sviluppo software e i prodotti di lavoro che influenzano e/o sono influenzati dal testing. In caso contrario, il processo di test potrebbe non raggiungere un'efficacia e un’efficienza ottimali (si veda il paragrafo 2.2.3).

2.2.2 Ulteriori Attività e Prodotti di Lavoro del Ciclo di Vita dello Sviluppo Software

Poiché il testing del software è una valutazione della qualità di uno o più prodotti di lavoro creati al difuori delle attività di test, viene svolto generalmente nel contesto di un insieme più ampio di attività del ciclo di vita dello sviluppo software. Il Test Manager deve pianificare e guidare le attività di test, con la consapevolezza di come queste altre attività e i loro prodotti di lavoro influenzino il testing, come già discusso nel Syllabus Foundation Level, e di come il testing influenzi queste altre attività e i loro prodotti di lavoro.

Ad esempio, nelle organizzazioni che utilizzano pratiche di sviluppo Agile, gli sviluppatori spesso eseguono uno sviluppo test-driven, creano unit test automatizzati e applicano il continuous integration del codice (e del testing per quel codice) nel sistema di configuration management. Il Test Manager dovrebbe lavorare con il responsabile dello sviluppo per assicurare che i tester siano integrati e allineati con queste attività. I tester possono eseguire la review degli unit test, sia per contribuire con suggerimenti ad una maggiore copertura ed efficacia di questi test, sia per acquisire una più profonda comprensione del software e della sua implementazione. I tester possono valutare modi per integrare i propri test automatizzati, in particolare i regression test funzionali, nel sistema di configuration management.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Mentre la relazione specifica tra le attività di test, gli altri stakeholder del testing, le attività del ciclo di vita dello sviluppo software e i prodotti di lavoro variano in base al progetto, al ciclo di vita dello sviluppo software adottato e una varietà di altri fattori, il testing è strettamente interconnesso e correlato ai seguenti fattori:

- Requirement Engineering e Requirement Management. Il Test Manager deve prendere in considerazione i requisiti durante la definizione e la stima dell'effort del test, rimanere informato sulle modifiche ai requisiti ed esercitare azioni di controllo dei test per adattarsi a tali cambiamenti. I Test Analyst e Technical Test Analyst dovrebbero partecipare alle review dei requisiti.
- Project Management. Il Test Manager, in collaborazione con i Test Analyst e Technical Test Analyst, deve fornire al Project Manager i requisiti di schedulazione e di risorse. Il Test Manager deve lavorare con il Project Manager per comprendere le modifiche al piano di progetto e svolgere le azioni di controllo dei test per adattarsi a tali modifiche.
- Configuration management, release management e change management. Il Test Manager, in collaborazione con il team di test, deve stabilire i processi e i meccanismi di rilascio dell'oggetto del test e inserirli nel test plan. Il Test Manager può chiedere ai Test Analyst e ai TechnicalTest Analyst di creare i test di verifica della build e di garantire il controllo delle versioni durante l'esecuzione dei test.
- Sviluppo e manutenzione del software. Il Test Manager dovrebbe lavorare con i responsabili dello sviluppo per coordinare il rilascio degli oggetti di test, compresi i contenuti e le date di ogni rilascio di test, nonché di partecipare al defect management (si veda il Capitolo 4).
- Supporto tecnico. Il Test Manager dovrebbe lavorare con il responsabile del supporto tecnico per garantire il corretto rilascio dei risultati del test durante la chiusura dei test, in modo che coloro che sono coinvolti nel supportare il prodotto dopo il rilascio siano consapevoli dei failure noti e dei workaround. In aggiunta, il Test Manager dovrebbe lavorare con il responsabile del supporto tecnico per analizzare i failure in produzione per implementare i miglioramenti del processo di test.
- Produzione della documentazione tecnica. Il Test Manager dovrebbe lavorare con il responsabile della documentazione tecnica per garantire il rilascio della documentazione per il testing in modo tempestivo, nonché la gestione dei difetti rilevati in questi documenti.

Oltre a identificare gli stakeholder del testing, come descritto sopra, Il Test Manager deve identificare le altre attività del ciclo di vita dello sviluppo software e i prodotti di lavoro che influenzano e/o sono influenzati dal testing. In caso contrario, il processo di test non raggiungerà la massima efficacia ed efficienza.

2.2.3 Allineamento delle Attività di Test e delle altre Attività del Ciclo di Vita

Il testing dovrebbe essere parte integrante del progetto, a prescindere dai modelli di sviluppo software utilizzati. Questi includono:

- Modelli sequenziali, come il modello a cascata (waterfall), il V-modelli e W-model. In un modello sequenziale, tutti i prodotti di lavoro e le attività di una determinata fase (p.e., requisiti, progettazione, implementazione, testing di componente (unit testing), testing di integrazione, testing di sistema e testing di accettazione) devono essere completati prima dell'inizio della fase successiva. La pianificazione, l'analisi, la progettazione e l'implementazione dei test procedono con sovrapposizioni con la pianificazione di progetto, l'analisi di business/requisiti, la progettazione del software e del database, e la codifica, con eventuali sovrapposizioni in base al livello di test considerato. L'esecuzione dei test procede in modo sequenziale in base ai livelli di test discussi nel Syllabus Foundation Level e in questo Syllabus.
- Modelli iterativi o incrementali, come Rapid Application Development (RAD) e Rational Unified Process (RUP). In un modello iterativo o incrementale, le funzionalità che devono essere realizzate sono raggruppate insieme (p.e., in base alla priorità di business o al rischio), quindi le varie fasi di progetto, comprese le loro attività e i prodotti di lavoro, sono svolte per ciascun gruppo di funzionalità. Le fasi possono essere svolte sia in modo sequenziale o con

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

sovrapposizioni, e le iterazioni stesse possono essere sequenziali o con sovrapposizioni. Durante l'avvio del progetto, la pianificazione e l'analisi dei test di alto livello sono svolte in parallelo con la pianificazione di progetto e l'analisi di business/requisiti. La pianificazione, l'analisi, la progettazione e l'implementazione dettagliata dei test sono svolte all'inizio di ogni iterazione, con sovrapposizioni. L'esecuzione dei test comporta spesso la sovrapposizione dei livelli di test. Ogni livello di test inizia il prima possibile e può continuare dopo l'inizio dei successivi livelli di test più alti.

- Modelli di sviluppo Agile, come Scrum ed Extreme Programming (XP). Questi sono cicli di vita iterativi, dove le iterazioni sono molto brevi (spesso da due a quattro settimane). I prodotti di lavoro e le attività di ogni iterazione vengono conclusi prima dell'inizio della successiva iterazione (cioè le iterazioni sono sequenziali). Il testing procede in modo similare ai modelli iterativi, ma con un maggior grado di sovrapposizione delle attività di test con le attività di sviluppo. Tutte le attività di una iterazione, comprese le attività di test, dovrebbero essere completate prima dell'inizio della successiva iterazione. In un progetto Agile, il ruolo del Test Manager di solito cambia, da un ruolo manageriale diretto ad un ruolo di autorità tecnica/advisory.
- Modelli a spirale. In un modello a spirale, vengono utilizzati dei prototipi nelle prime fasi del progetto per confermare la fattibilità e per sperimentare decisioni di progettazione e implementazione, utilizzando il livello di priorità di business e il livello di rischio tecnico per decidere l'ordine con cui creare i prototipi. Questi prototipi vengono testati per determinare quali aspetti dei problemitecnici rimangono irrisolti. Una volta che i principali problemi tecnici sono stati risolti, il progettoprocede applicando un modello sequenziale oppure iterativo.

Per allineare in modo appropriato le attività di test all'interno di un ciclo di vita, il Test Manager deve avere una comprensione dettagliata dei modelli del ciclo di vita utilizzati nella propria organizzazione. Ad esempio, nel V-model, il processo fondamentale di test ISTQB, applicato a livello di test di sistema, potrebbe allinearsi come segue:

- Le attività di pianificazione dei test avvengono in parallelo con la pianificazione di progetto, e il controllo dei test continua fino a quando l'esecuzione dei test e la chiusura dei test sono state completate.
- Le attività di analisi e progettazione dei test avvengono in parallelo con la specifica dei requisiti, con la specifica di progettazione di sistema e architetturale (di alto livello), e con la specifica di progettazione di componente (di basso livello).
- Le attività di implementazione dei test potrebbero iniziare durante la progettazione di sistema, anche se in genere la maggior parte di queste attività avvengono in parallelo con la codifica e il testing di componente, e continuano fino a pochi giorni prima dell'inizio dell'esecuzione dei test di sistema.
- Le attività di esecuzione dei test iniziano quando i relativi criteri di ingresso sono tutti soddisfatti (o espressamente non considerati), e questo in genere significa che almeno il testing di componente e il testing di integrazione sono stati completati. L'esecuzione dei test continua fino a quando i relativi criteri di uscita sono soddisfatti.
- La valutazione dei criteri di uscita e il reporting dei risultati dei test di sistema vengono svolti durante tutta l'esecuzione dei test di sistema, in genere con maggiore frequenza e urgenza in prossimità delle scadenze di progetto.
- Le attività di chiusura dei test si svolgono dopo che i criteri di uscita sono soddisfatti e l'esecuzione dei test di sistema è stata dichiarata completa, anche se a volte possono essere ritardate fino al completamento dei test di accettazione e di tutte le attività di progetto.

In un ciclo di vita iterativo e incrementale, devono essere eseguite le stesse attività, ma la loro tempistica ed estensione possono variare. Ad esempio, invece di dover implementare l'ambiente di test completo all'inizio del progetto, può essere più efficiente implementare solo la parte necessaria per l'iterazione corrente. Con qualsiasi modello di ciclo di vita iterativo o incrementale, più avanti nel tempo si svolge la pianificazione, maggiore è la possibilità di estendere l'ambito del processo fondamentale di test.

In aggiunta alle fasi di pianificazione svolte per ogni progetto, anche l'esecuzione e il reporting dei test possono essere influenzati dal ciclo di vita utilizzato dal team. Ad esempio, in un ciclo di vita iterativo,

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

può essere efficace produrre dei report completi e condurre sessioni di review post-iterazione, prima dell'inizio della successiva iterazione. Trattando ogni iterazione come un mini-progetto, il team ha la possibilità di correggere e modificare in base a quello che è accaduto durante l'iterazione precedente. Poiché le iterazioni possono essere brevi e di durata limitata, può avere senso ridurre il tempo e l'effort spesi per il reporting e la review, ma tali attività dovrebbero essere condotte per tracciare l'andamento generale del testing e per identificare eventuali aree problematiche il prima possibile. I problemi del processo in una iterazione possono facilmente influenzare e persino ripetersi nella successiva iterazione, se adeguate misure correttive non sono state prese.

Informazioni generali su come allineare il testing con le altre attività del ciclo di vita possono essere catturate nella strategia di test (si veda il paragrafo 2.4.2). Durante la pianificazione dei test e/o la pianificazione di progetto, il Test Manager dovrebbe eseguire un allineamento di progetto, per ogni livello di test e per ogni combinazione di ciclo di vita dello sviluppo software e processo di test.

In base alle esigenze dell'organizzazione, del progetto e del prodotto, possono essere richiesti livelli di test aggiuntivi, oltre a quelli definiti nel Syllabus Foundation Level, come:

- Testing di integrazione hardware-software
- Testing di integrazione del sistema
- Testing di interazione funzionale
- Testing di integrazione del prodotto del cliente

Ogni livello di test dovrebbe avere chiaramente definiti i seguenti elementi:

- Obiettivi di test, con obiettivi raggiungibili
- Ambito del test ed elementi di test
- Base di test, con i mezzi per misurare la copertura di tale base (ad es. tracciabilità)
- Criteri di ingresso e di uscita
- Prodotti del test, incluso il reporting dei risultati
- Tecniche di test applicabili, insieme a una modalità che garantisca i livelli appropriati di copertura usando queste tecniche
- Misure e metriche rilevanti per gli obiettivi di test, per i criteri di entrata e di uscita, per il reporting dei risultati (compresa la misura della copertura)
- Strumenti di test da utilizzare per specifiche attività di test (se e ove applicabili)
- Risorse (ad es. ambienti di test)
- Singoli individui e gruppi responsabili, sia interni che esterni al team di test
- Conformità agli standard aziendali, normativi o altri (se e ove applicabile)

Come discusso più avanti in questo capitolo, la best practice è di definire questi elementi in modo coerente per tutti i livelli di test, per evitare inutili e pericolose lacune tra diversi livelli di test similari.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

2.2.4 Gestire il Testing Non-Funzionale

Errori nella pianificazione dei test non-funzionali possono portare alla rilevazione di problemi di qualità gravi, a volte disastrosi, dopo il rilascio di un sistema. Tuttavia, molti tipi di test non-funzionali sono costosi, quindi il Test Manager deve selezionare quali test non-funzionali eseguire in base al rischio e ai vincoli. Inoltre, esistono molti tipi diversi di test non-funzionali, alcuni dei quali potrebbero essere non appropriati per uno specifico applicativo.

Poiché il Test Manager può non avere sufficiente esperienza per gestire tutti gli aspetti della pianificazione, ha bisogno di delegare una parte delle responsabilità di pianificazione dei test ai Technical Test Analyst (e in alcuni casi ai Test Analyst) assegnati delle attività di test non-funzionale. Il Test Manager dovrebbe chiedere agli Analyst di considerare i seguenti fattori:

- Requisiti degli stakeholder
- Strumenti necessari
- Ambiente di test
- Fattori organizzativi
- Sicurezza

Per ulteriori dettagli, si veda il Capitolo 4 del Syllabus Advanced Technical Test Analyst [ISTQB ATTA SYL].

Un'altra importante considerazione per i Test Manager è come integrare i test non-funzionali nel ciclo di vita dello sviluppo software. Un errore comune è quello di attendere che tutti i test funzionali siano stati completati, prima di iniziare i test non-funzionali, con il risultato di scoprire tardivamente difetti critici non-funzionali. Invece, ai test non-funzionali bisognerebbe assegnare una priorità e una sequenza in base al rischio. Esistono spesso modalità per mitigare i rischi non-funzionali durante i livelli iniziali di test o anche durante lo sviluppo. Ad esempio, le review di usabilità dei prototipi della user interface durante la progettazione di sistema possono essere molto efficaci nel rilevare difetti di usabilità, che potrebbero creare seri problemi di schedulazione se scoperti verso la fine del testing di sistema.

Nei cicli di vita iterativi, il ritmo delle modifiche e delle iterazioni può rendere difficile il focus sui test non-funzionali, che richiedono la costruzione di framework di test sofisticati. Le attività di progettazione e implementazione dei test che richiedano tempi più lunghi di una singola iterazione, dovrebbero essere organizzate come attività separate al di fuori delle iterazioni.

2.2.5 Gestire il Testing Basato sull'Esperienza

Benché il testing basato sull'esperienza fornisca benefici nel rilevare in modo efficace difetti che altre tecniche di test possono non rilevare, e nel verificare la completezza di tali tecniche, fornisce alcune sfide per il test management. Il Test Manager dovrebbe essere consapevole delle sfide e dei benefici delle tecniche basate sull'esperienza, in particolare del testing esplorativo. È difficile determinare la copertura raggiunta durante questo testing, a causa del logging "leggero" e della minima preparazione anticipata dei test. La riproducibilità dei risultati dei test richiede una particolare attenzione nella gestione, soprattutto quando sono coinvolti più tester.

Un modo per gestire il testing basato sull'esperienza, soprattutto il testing esplorativo, è quello di suddividere il lavoro del testing in periodi brevi di test, da 30 a 120 minuti, chiamati a volte sessioni di test. La durata time-boxed limita e focalizza il lavoro da svolgere in una sessione e fornisce un adeguato livello di schedulazione e monitoraggio. Ogni sessione copre un test charter, che il Test Manager comunica al tester in forma verbale o scritta. Il test charter fornisce le condizioni di test che devono essere coperte nella sessione di test, il che aiuta a mantenere il focus ed evita sovrapposizioni quando il testing esplorativo è svolto contemporaneamente da più persone.

Un'altra tecnica per gestire il testing basato sull'esperienza è di integrare questo testing auto-gestito e spontaneo in sessioni di test pre-progettate e più tradizionali. Ad esempio, ai tester può essere concesso (con un tempo allocato) di esplorare oltre ai passi, gli input e i risultati attesi specificati nei loro test predefiniti. Ai tester possono anche essere assegnate tali sessioni di test auto-gestite come parte del

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

loro testing giornaliero prima, durante o dopo lo svolgimento dei test predefiniti. Se tali sessioni di test rilevano difetti o aree di interesse per un futuro testing, i test predefiniti possono essere aggiornati.

All'inizio della sessione esplorativa, il tester rileva ed esegue i compiti di set-up necessari per i test. Durante la sessione, il tester impara e ottiene informazioni sull'applicazione da testare, progetta ed esegue i test in base alla tecnica applicata e a quanto appreso sull'applicazione, indaga su eventuali difetti e documenta i risultati in un test log (Se fosse richiesta la ripetibilità dei test, i tester dovrebbero anche tracciare gli input, le azioni e gli eventi). Dopo la sessione, può essere svolto un debriefing, che meglio indirizzi le sessioni successive.

2.3 Testing basato sul Rischio e altri Approcci per la Prioritizzazione dei Test e l'Allocazione dell'Effort

Una sfida comune del test management è una appropriata selezione, allocazione e prioritizzazione dei test. Questo significa che, partendo da un numero praticamente infinito di condizioni di test e combinazioni di condizioni di test che potrebbero essere coperte, il team di test deve selezionare un insieme finito di condizioni, determinare un'adeguata quantità di effort da allocare per coprire ogni condizione con test case, e definire una sequenza dei test case risultanti in un ordine prioritizzato che ottimizzi l'efficacia e l'efficienza del testing da eseguire. Il Test Manager, per aiutare a risolvere questo problema, può usare, insieme ad altri fattori, l'identificazione e l'analisi del rischio, anche se i numerosi vincoli e variabili interagenti possono richiedere soluzioni di compromesso.

2.3.1 Il Testing basato sul Rischio

Il rischio è la possibilità che si verifichi un risultato o evento negativo o indesiderato. I rischi esistono ogni volta che si possa verificare un problema che faccia diminuire la percezione della qualità del prodotto o del successo del progetto da parte del cliente, utente, partecipante o stakeholder. Nel caso in cui l'effetto principale del problema potenziale sia sulla qualità del prodotto, vengono indicati come rischi di qualità, rischi di prodotto o rischi di qualità del prodotto. Nel caso in cui l'effetto principale del problema potenziale sia sul successo del progetto, vengono indicati come rischi di progetto o rischi di pianificazione.

Nel testing basato sul rischio, i rischi di qualità vengono identificati e valutati durante l'analisi dei rischi con gli stakeholder. Il team di test successivamente progetta, implementa ed esegue i test per mitigare i rischi di qualità. La qualità include la totalità delle funzionalità, caratteristiche, comportamenti e attributi che influenzino la soddisfazione del cliente, utente e stakeholder. Quindi, un rischio di qualità è una potenziale situazione in cui i problemi di qualità potrebbero essere presenti in un prodotto. Esempi di rischi di qualità per un sistema includono: calcoli errati nei report (rischio funzionale relativo all'accuratezza), risposta lenta agli input utente (rischio non-funzionale relativo all'efficienza e ai tempi di risposta), difficoltà nella comprensione delle schermate e dei campi (rischio non-funzionale relativo all'usabilità e comprensibilità). Quando i test rivelano difetti, il testing ha mitigato il rischio di qualità, fornendo la consapevolezza dei difetti presenti e le opportunità di gestirli prima del rilascio. Quando i test non rilevano difetti, il testing ha mitigato il rischio di qualità assicurando che, nelle condizioni testate, il sistema funzioni correttamente.

Il testing basato sul rischio utilizza i rischi di qualità per selezionare le migliori condizioni di test, allocare l'effort del test per queste condizioni, e prioritizzare i test case risultanti. Esistono una varietà di tecniche per il testing basato sul rischio, con variazioni significative sia per il tipo e il livello di documentazione utilizzata, sia per il livello di formalizzazione applicato. Il testing basato sul rischio ha l'obiettivo, esplicito o implicito, di utilizzare il testing per ridurre il livello complessivo di rischio di qualità e, in particolare, per ridurre il livello di rischio a un livello accettabile.

Il testing basato sul rischio è costituito da quattro attività principali:

- Identificazione del rischio

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Valutazione del rischio
- Mitigazione del rischio
- Gestione del rischio

Queste attività si sovrappongono. I seguenti paragrafi descrivono ognuna di queste attività.

Per essere più efficace, l'identificazione e la valutazione dei rischi dovrebbero coinvolgere i rappresentanti di tutti i gruppi di stakeholder di progetto e di prodotto, anche se a volte le diverse realtà progettuali comportano che alcuni stakeholder agiscano come sostituti di altri stakeholder. Ad esempio, nello sviluppo software per il mercato di massa, potrebbe essere chiesto a un piccolo campione di potenziali clienti di identificare i potenziali difetti che potrebbero impattare più pesantemente l'utilizzo del software; in questo caso il campione di potenziali clienti serve come sostituto per una eventuale customer base completa. A causa della loro particolare esperienza con i rischi di qualità e i failure, i tester dovrebbero essere attivamente coinvolti nel processo di identificazione e valutazione dei rischi.

2.3.1.1 Identificazione del Rischio

Gli stakeholder possono identificare i rischi attraverso una o più delle seguenti tecniche:

- Interviste di esperti
- Valutazioni indipendenti
- Utilizzo di modelli di rischio
- Retrospettive di progetto
- Workshop sui rischi
- Brainstorming
- Checklist
- Riferimenti a esperienze passate

Coinvolgendo il più ampio campione possibile di stakeholder, il processo di identificazione del rischio ha più alta probabilità di identificare la maggior parte dei rischi di qualità significativi.

L'identificazione del rischio spesso produce degli effetti secondari, ad esempio l'identificazione di problemi che non sono i rischi di qualità. Esempi includono domande generali o dubbi sul prodotto o sul progetto, problemi nei documenti di riferimento come i requisiti e specifiche di progettazione. I rischi di progetto sono spesso identificati anche come un effetto secondario dell'identificazione dei rischi di qualità, ma non sono l'obiettivo principale del testing basato sul rischio. Tuttavia, il risk management di progetto è importante per tutto il testing, non solo per il testing basato sul rischio, ed è trattato nel paragrafo 2.4.

2.3.1.2 Valutazione del Rischio

Una volta eseguita l'identificazione del rischio, può iniziare la valutazione del rischio, che è lo studio dei rischi identificati. In particolare, la valutazione del rischio include la categorizzazione di ogni rischio e la definizione della probabilità e dell'impatto associato ad ogni rischio. La valutazione del rischio può anche valutare o assegnare altre proprietà di ogni rischio, come il proprietario del rischio.

La categorizzazione del rischio posiziona ogni rischio in un'appropriata tipologia, come prestazioni, affidabilità, funzionalità e così via. Alcune organizzazioni utilizzano le caratteristiche di qualità dello standard ISO-9126 [ISO9126] (che sarà sostituito dallo standard ISO-25000 [ISO25000]) per la categorizzazione, ma molte organizzazioni utilizzano altri schemi di categorizzazione. La stessa checklist utilizzata per l'identificazione è spesso usata anche per categorizzare i rischi. Esistono checklist generiche dei rischi di qualità e molte organizzazioni personalizzano queste liste. Quando si utilizza una checklist come base per l'identificazione dei rischi, la loro categorizzazione è spesso svolta durante l'identificazione.

La definizione del livello di rischio comporta in genere la valutazione, per ciascun elemento di rischio, della probabilità di occorrenza e dell'impatto al suo verificarsi. La probabilità di occorrenza è l'eventualità che il potenziale problema si manifesti nel sistema sotto test. In altre parole, la probabilità è una

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

valutazione del livello di rischio tecnico. I fattori che influenzano la probabilità di rischi di prodotto e di progetto includono:

- Complessità della tecnologia e del team
- Problemi di personale e di training (formazione) tra i business analyst, i progettisti e i programmatori
- Conflitti all'interno del team
- Problemi contrattuali con i fornitori
- Team geograficamente distribuito
- Prodotti software legacy oppure innovativi
- Strumenti e tecnologia
- Debole leadership manageriale o tecnica
- Pressione sui tempi, sulle risorse, sul budget e sul management
- Mancanza di attività anticipate di quality assurance
- Alta frequenza di modifiche
- Alta frequenza di difetti nelle fasi iniziali
- Problemi di interfaccia e di integrazione

L'impatto del verificarsi del rischio è la severità degli effetti sugli utenti, clienti o altri stakeholder. I fattori che influenzano l'impatto dei rischi di progetto e di prodotto includono:

- Frequenza di utilizzo della funzionalità impattata
- Criticità della funzionalità per realizzare un obiettivo di business
- Danni alla reputazione
- Perdita di business
- Potenziali perdite o responsabilità finanziarie, ecologiche o sociali
- Sanzioni legali civili o penali
- Perdita di licenze
- Mancanza di workaround ragionevoli
- Visibilità di failure con conseguente pubblicità negativa
- Sicurezza fisica (safety)

Il livello di rischio può essere valutato sia quantitativamente sia qualitativamente. Se la probabilità e l'impatto possono essere definiti quantitativamente, si possono moltiplicare i due valori per calcolare un valore numerico di priorità del rischio (risk priority number). Normalmente, il livello di rischio può essere definito solo qualitativamente. Questo significa che si può parlare di probabilità molto alta, alta, media, bassa o molto bassa, ma la probabilità non può essere espressa come percentuale con una precisione reale. Allo stesso modo, si può parlare di impatto molto alto, alto, medio, basso o molto basso, ma l'impatto non può essere espresso in termini finanziari in modo preciso o completo. Questa valutazione qualitativa dei livelli di rischio non dovrebbe essere vista come inferiore agli approcci quantitativi; anzi, quando le valutazioni quantitative dei livelli di rischio sono utilizzate in modo inappropriato, i risultati possono ingannare gli stakeholder merito alla misura in cui si comprende effettivamente il rischio e può essere gestito. Le valutazioni qualitative dei livelli di rischio sono spesso combinate, attraverso moltiplicazioni o addizioni, per creare un punteggio aggregato del rischio. Questo punteggio aggregato del rischio può essere espresso come un risk priority number, ma dovrebbe essere interpretato come una valutazione relativa su una scala ordinale.

Inoltre, a meno che l'analisi del rischio non si basi su un insieme ampio e statisticamente valido di dati del rischio, l'analisi del rischio sarà basata sulle percezioni soggettive degli stakeholder relativamente alla probabilità e all'impatto. I project manager, i programmatori, gli utenti, i business analyst, gli architetti e i tester in genere hanno percezioni diverse e conseguentemente opinioni differenti sul livello di rischio relativo ad ogni elemento di rischio. Il processo di analisi del rischio dovrebbe includere una modalità di raggiungimento del consenso o, nel peggiore dei casi, stabilire un livello di rischio concordato (p.e., attraverso un'imposizione del management o prendendo il valore medio, la mediana o la moda dell'elemento di rischio). Inoltre, dovrebbe essere verificato che i livelli di rischio abbiano una buona distribuzione nel range dei valori ammessi, per garantire che i valori di rischio forniscano significative

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

indicazioni in termini di sequenza di test, prioritizzazione e allocazione del test. Altrimenti, i livelli di rischio non possono essere utilizzati come guida per le attività di mitigazione del rischio.

2.3.1.3 Mitigazione del Rischio

Il Testing basato sul rischio inizia con una analisi dei rischi di qualità (identificazione e valutazione dei rischi di qualità del prodotto). Questa analisi è il fondamento del master test plan e degli altri test plan. Come specificato nel test plan, i test sono progettati, implementati ed eseguiti in modo da coprire i rischi. L'effort associato allo sviluppo e all'esecuzione di un test è proporzionale al livello di rischio, e questo significa che le tecniche di test più meticolose (come il testing pairwise) vengono utilizzate per i rischi più critici, mentre le tecniche di test meno meticolose (come il partizionamento di equivalenza o il testing esplorativo time-boxed) vengono utilizzate per i rischi meno critici. Inoltre, la priorità dello sviluppo e dell'esecuzione dei test si basa sul livello di rischio. Alcuni standard relativi al safety (p.e., FAA DO-178B/ED 12B, IEC 61508), prescrivono tecniche di test e grado di copertura basati sul livello di rischio. Inoltre, il livello di rischio dovrebbe influenzare decisioni, quali l'uso delle review dei prodotti di lavoro del progetto (incluso il testing), il livello di indipendenza e il livello di esperienza del tester, e il grado di copertura del testing confermativo (re-testing) e di regression testing eseguito.

Durante il progetto, il team di test dovrebbe venire a conoscenza di informazioni aggiuntive che modifichino l'insieme dei rischi di qualità e/o il livello di rischio associato ai rischi di qualità conosciuti. Dovrebbe essere eseguito un adeguamento periodico dell'analisi dei rischi di qualità, che causa un adeguamento dei test. Questi adeguamenti dovrebbero avvenire almeno in occasione delle principali milestone di progetto. Includono l'identificazione di nuovi rischi, una rivalutazione del livello dei rischi esistenti e la valutazione dell'efficacia delle attività di mitigazione del rischio. Per fare un esempio, se una sessione di identificazione e valutazione del rischio è stata eseguita durante la fase dei requisiti, basandosi sulle specifiche dei requisiti, una volta finalizzata la specifica di progettazione, dovrebbe essere svolta una rivalutazione dei rischi. Per fare un altro esempio, se durante il testing di componente si rilevano un numero di difetti notevolmente più alto rispetto a quello previsto, si può concludere che la probabilità di difetti in questa area era maggiore del previsto e quindi occorre un adeguamento aumentando la probabilità e il livello complessivo di rischio. Questo potrebbe causare un aumento nella quantità di testing da eseguire per questo componente.

I rischi di qualità del prodotto possono anche essere mitigati prima che inizi l'esecuzione dei test. Ad esempio, se durante l'identificazione del rischio si rilevano problemi con i requisiti, il team di progetto può svolgere approfondite review delle specifiche dei requisiti come azione di mitigazione. Questo può ridurre il livello di rischio, e quindi sono necessari meno test per mitigare il rischio di qualità rimanente.

2.3.1.4 Risk Management nel Ciclo di Vita

Idealmente, il risk management si svolge durante l'intero ciclo di vita. Se un'organizzazione ha un documento di politica del test e/o un documento di strategia di test, questi dovrebbero descrivere il processo generale attraverso il quale i rischi di prodotto e di progetto vengono gestiti nel testing, e mostrare come il risk management sia integrato e influenzi tutte le fasi del testing.

In un'organizzazione matura dove la consapevolezza dei rischi pervade il team di progetto, il risk management viene svolto a molti livelli, e non solo per il testing. I rischi importanti vengono indirizzati non solo il prima possibile in particolari livelli di test, ma anche nei livelli di test precedenti. (Per esempio, se le prestazioni vengono identificate come un'area chiave di rischio di qualità, il performance testing non solo inizierà in anticipo durante i test di sistema, ma verrà eseguito anche durante il testing di componente e il testing di integrazione.) Le organizzazioni mature non solo identificano i rischi, ma anche le relative sorgenti di rischio e le conseguenze di questi rischi. Per quei difetti che si manifestano, la root cause analysis viene usata per comprendere in modo più approfondito le sorgenti del rischio e per implementare i miglioramenti del processo che prevengano difetti futuri. La mitigazione avviene in tutto il ciclo di vita dello sviluppo software. L'analisi del rischio è basata su informazioni che considerano la relativa attività di lavoro, l'analisi del comportamento del sistema, la valutazione dei rischi basata sui costi, l'analisi dei rischi di prodotto, l'analisi dei rischi dell'utente finale, l'analisi dei rischi di responsabilità

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

legale. L'analisi del rischio trascende il testing, in quanto il team di test partecipa e influenza un'analisi del rischio a livello aziendale.

Molti metodi di testing basato sul rischio includono anche tecniche che utilizzano il livello di rischio per dare una sequenza e prioritizzare il testing, garantendo così fin dalle prime fasi una copertura delle aree più importanti e la scoperta dei difetti più importanti durante l'esecuzione dei test. In alcuni casi, tutti i test di rischio più elevato vengono eseguiti prima dei test di rischio inferiore, e i test vengono eseguiti secondo un ordine rigoroso basato sul livello di rischio (spesso chiamato "depth-first"); in altri casi viene usato un approccio di campionamento per selezionare un campione di test tra tutti gli elementi di rischio identificati, utilizzando il rischio per pesare la selezione, garantendo allo stesso tempo la copertura almeno una volta di ogni elemento di rischio (spesso chiamato "breadth-first").

Indipendentemente che il testing basato sul rischio sia "depth-first" o "breadth-first", è possibile che il tempo assegnato per il testing possa esaurirsi senza che tutti i test vengano eseguiti. Il testing basato sul rischio consente ai tester di riportare al management in termini di livello di rischio residuo, e permette al management di decidere se estendere il testing o trasferire il rischio residuo sugli utenti, clienti, supporto help desk/tecnico e/o staff di operations.

Durante l'esecuzione dei test, le più sofisticate tecniche di test basate sul rischio, che non devono necessariamente essere più formali o pesanti, permettono ai partecipanti del progetto, ai project manager e ai product manager, all'executive management, ai senior manager e agli stakeholder di progetto, di monitorare e controllare il ciclo di vita dello sviluppo software, incluse le decisioni di rilascio, in base al livello di rischio residuo. Questo richiede al Test Manager di riportare i risultati dei test in termini di rischio, in modo che possano essere compresi da ogni stakeholder del test.

2.3.2 Tecniche di Test Basate sul Rischio

Esistono un certo numero di tecniche per il testing basato sul rischio. Alcune di queste tecniche sono altamente informali. Esempi includono approcci in cui il tester analizza i rischi di qualità durante il testing esplorativo [Whittaker09]. Questo può aiutare a indirizzare il testing, ma può portare ad un eccessivo focus sulla probabilità dei difetti e non sul loro impatto, e non prende in considerazione gli input di stakeholder cross-funzionali. Inoltre, tali approcci sono soggettivi e dipendono dalle competenze, dall'esperienza e dalle preferenze del singolo tester. Come tale, questi approcci raramente raggiungono tutti i benefici del testing basato sul rischio.

Nel tentativo di catturare i benefici del testing basato sul rischio riducendo al minimo i costi, molti professionisti utilizzano approcci "leggeri" ("lightweight") basati sul rischio. Questi mescolano la reattività e la flessibilità degli approcci informali con la potenza e la costruzione del consenso degli approcci più formali. Esempi di approcci "leggeri" includono: Pragmatic Risk Analysis and Management (PRAM) [Black09], Systematic Software Testing (SST) [Craig02], e Product Risk Management (PRisMa) [vanVeenendaal12]. In aggiunta agli attributi usuali del testing basato sul rischio, queste tecniche hanno tipicamente i seguenti attributi:

- Evoluto nel tempo, in base all'esperienza di settore con il testing basato sul rischio, in particolare in settori dove gli aspetti di efficienza sono importanti
- Impostato sull'ampio coinvolgimento di un team cross-funzionale di stakeholder, che rappresentano sia la prospettiva del business che tecnica, durante l'identificazione e la valutazione iniziale dei rischi
- Ottimizzato se introdotto durante le prime fasi di un progetto, quando le opzioni per mitigare i rischi di qualità sono massimizzate e quando i principali prodotti e sottoprodotti di lavoro dell'analisi del rischio possono aiutare a influenzare la specifica e l'implementazione del prodotto in modo da minimizzare il rischio
- Utilizzo degli output generati (matrice di rischio o tabella di analisi del rischio) come base per il test plan e le condizioni di test, e per tutte le successive attività di test management e analisi dei test

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Supporto al reporting dei risultati dei test in termini di rischio residuo verso tutti i livelli di stakeholder del test.

Alcune di queste tecniche (per esempio SST) richiedono le specifiche dei requisiti come input nell'analisi del rischio, e non possono essere utilizzate se non quando le specifiche dei requisiti siano state fornite. Altre tecniche (p.e., PRisMa e PRAM) incoraggiano l'uso di un strategia mista (blended) basata sui requisiti e sul rischio, utilizzando i requisiti e/o altre specifiche come input per l'analisi del rischio,ma può funzionate basandosi interamente sull'input degli stakeholder. L'utilizzo dei requisiti come input aiuta infatti a garantire una buona copertura dei requisiti, ma il Test Manager deve garantire che non vengano dimenticati rischi importanti non suggeriti dai requisiti, soprattutto nelle aree non-funzionali. Quando vengono forniti in input requisiti validi e prioritizzati, si riscontra in genere una forte correlazione tra i livelli di rischio e la priorità dei requisiti.

Molte di queste tecniche incoraggiano anche l'uso del processo di identificazione e di valutazione dei rischi come un modo per creare consenso tra gli stakeholder sull'approccio di test. Questo è un beneficio molto potente, ma richiede agli stakeholder di dedicare tempo per partecipare sia a sessioni di brainstorming di gruppo sia a interviste individuali. Una partecipazione insufficiente degli stakeholder causa lacune nell'analisi del rischio. Naturalmente, gli stakeholder non concordano sempre sul livello di rischio; quindi, la persona che guida l'analisi dei rischi di qualità deve lavorare con gli stakeholder in modo creativo e positivo per raggiungere il miglior livello possibile di accordo. Tutte le competenze di un moderatore qualificato per condurre review meeting sono applicabili alla persona che guida una analisi dei rischi di qualità.

Come le tecniche più formali, le tecniche "leggere" permettono di pesare i fattori di probabilità e impatto per enfatizzare i rischi tecnici e di business (ad esempio, in base al livello di test). A differenza di tecniche più formali, le tecniche "leggere":

- Utilizzano solo due fattori, probabilità e impatto e,
- Utilizzano giudizi e scale semplici e qualitative.

La natura leggera di questi approcci fornisce la flessibilità, l'applicabilità ad un ampio insieme di domini, e l'accessibilità tra team di tutti i livelli di competenza ed esperienza (anche con persone non-tecniche e junior).

In opposizione, le tecniche "pesanti" e formali forniscono al Test Manager un insieme di opzioni:

- Hazard analysis, che estende il processo di analisi a monte, cercando di rilevare i pericoli (hazard) che sono alla base di ogni rischio.
- Cost of exposure, dove il processo di valutazione dei rischi consiste nel determinare, per ogni elemento di rischio di qualità, tre fattori: 1) la probabilità (espressa in percentuale) di un failure relativo all'elemento di rischio; 2) il costo di una perdita (espressa come quantità finanziaria) associata al failure relativo all'elemento di rischio, nel caso dovesse verificarsi in produzione, e 3) il costo del testing per tali failure.
- Failure Mode and Effect Analysis (FMEA) e le sue varianti [Stamatis03], dove vengono identificati i rischi di qualità, le loro cause potenziali e i loro possibili effetti, per poi assegnare la severità, la priorità e la frequenza di accadimento.
- Quality Function Deployment (QFD), che è una tecnica di gestione dei rischi di qualità con implicazioni di testing, che tratta in particolare i rischi di qualità derivati da una comprensione errata o insufficiente dei requisiti dei clienti o degli utenti.
- Fault Tree Analysis (FTA), dove i failure realmente osservati (durante il testing o in produzione), o i potenziali failure (rischi di qualità), vengono sottoposti a una root cause analysis a partire dai difetti che potrebbero causare il failure, poi considerando gli errori o difetti che potrebbero causare tali difetti, continuando fino a quando le diverse root cause sono state identificate.

Le tecniche specifiche che dovrebbero essere utilizzate per il testing basato sul rischio, e il grado di formalità di ogni tecnica, dipendono da considerazioni sul progetto, sul processo e sul prodotto. Ad esempio, un approccio informale come la tecnica esplorativa di Whittaker, potrebbe applicarsi a una patch o ad un quick fix. Nei progetti Agile, l'analisi dei rischi di qualità è completamente integrata nel

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

primo periodo di ogni sprint, e la documentazione dei rischi si mescola al tracciamento delle user story. I sistemi di sistemi richiedono l'analisi del rischio per ogni sistema e per sistema di sistemi complessivo. Progetti safety-critical e mission-critical richiedono livelli più alti di formalità e di documentazione.

Gli input, i processi e gli output per il testing basato sul rischio tenderanno ad essere determinati dalla tecnica selezionata. Input comuni includono le percezioni degli stakeholder, le specifiche e i dati storici. Processi comuni includono l'identificazione dei rischi, la valutazione dei rischi e il controllo dei rischi. Output comuni includono un elenco dei rischi di qualità (con l'associato livello di rischio e la raccomandata allocazione dell'effort del test), i difetti rilevati nei documenti di input, come le specifiche, domande o problemi relativi agli elementi di rischio, e i rischi di progetto che influenzano il testing o l'intero progetto.

Di solito il fattore di successo più critico per il testing basato sul rischio è il coinvolgimento del giusto team di stakeholder nell'identificazione e nella valutazione dei rischi. Tutti gli stakeholder hanno una propria opinione di quello che costituisce la qualità per il prodotto, e hanno un proprio insieme di priorità e preoccupazioni sulla qualità. Tuttavia, gli stakeholder tendono a cadere in due ampie categorie: stakeholder di business e stakeholder tecnici.

Gli stakeholder di business includono i clienti, gli utenti, lo staff di operations, lo staff di help desk e di supporto tecnico. Questi stakeholder comprendono il cliente e l'utente e quindi possono identificare i rischi e valutarne l'impatto da una prospettiva del business.

Gli stakeholder tecnici includono gli sviluppatori, gli architetti, i database administrator e i network administrator. Questi stakeholder comprendono gli elementi di base per cui il software può fallire e possono quindi identificare i rischi e valutarne la probabilità da una prospettiva tecnica.

Alcuni stakeholder hanno entrambe le prospettive, di business e tecnica. Ad esempio, gli esperti in materia (subject matter expert) che hanno ruoli nel testing o nella business analysis hanno spesso una visione più ampia sui rischi, a causa della loro competenza tecnica o di business,

Il processo di identificazione degli elementi di rischio genera una lista sostanziale di rischi. Non è necessario per gli stakeholder argomentare su un elemento di rischio; ogni volta che uno stakeholder percepisce qualcosa come un rischio per la qualità del sistema, è un elemento di rischio. Tuttavia è importante che gli stakeholder raggiungano il consenso sul valore del livello di rischio. Ad esempio, negli approcci "leggeri" che utilizzano probabilità e impatto come fattori di valutazione, parte del processo deve includere l'identificazione di uno schema comune per la valutazione della probabilità e dell'impatto. Tutti gli stakeholder, compreso il team di test, devono utilizzare questo stesso schema, ed essere in grado di convergere ad un unico valore di probabilità e impatto per ogni elemento di rischio di qualità.

Se il testing basato sul rischio deve essere utilizzato a lungo termine, allora il Test Manager lo deve promuovere e attivare con successo insieme agli stakeholder. Il gruppo cross-funzionale deve vedere il valore aggiunto dell'analisi del rischio per garantire l'uso continuativo della tecnica. Questo richiede che il Test Manager comprenda le esigenze, le aspettative e le disponibilità temporali degli stakeholder per partecipare al processo.

Un buon coinvolgimento degli stakeholder nel processo di analisi del rischio qualità offre un importante beneficio al Test Manager. Questo beneficio è che, per progetti minori con requisiti deboli o mancanti, gli stakeholder possono ancora identificare i rischi quando guidati da una checklist appropriata. Questo beneficio si manifesta quando, dopo l'implementazione del testing basato sul rischio, migliora l'efficacia della rilevazione dei difetti da parte del team di test. Questo accade perché viene utilizzata una base di test più completa, in questo caso la lista degli elementi dei rischi di qualità.

Durante la chiusura dei test nel testing basato sul rischio, i team di test dovrebbero misurare l'entità dei benefici ottenuti. In molti casi, questo richiede di rispondere ad alcune o a tutte le seguenti quattro domande, attraverso metriche e la consultazione con il team:

- Il team di test ha rilevato una maggiore percentuale di difetti importanti rispetto a quelli meno importanti?

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Il team di test ha rilevato la maggior parte dei difetti importanti all'inizio del periodo di esecuzione dei test?
- Il team di test è stato in grado di spiegare i risultati dei test agli stakeholder in termini di rischio?
- I test non eseguiti dal team di team (se presenti) avevano un livello di rischio associato più basso rispetto a quelli eseguiti?

Nella maggior parte dei casi, il successo del testing basato sul rischio dipende da una risposta affermativa a tutte e quattro le domande. Nel lungo termine, il Test Manager dovrebbe fissare obiettivi di process improvement per queste metriche, insieme allo sforzo di migliorare l'efficienza del processo di analisi dei rischi di qualità. Naturalmente possono essere applicate altre metriche e criteri di successo al testing basato sul rischio, e il Test Manager dovrebbe considerare con attenzione la relazione tra queste metriche, gli obiettivi strategici del team di test e i comportamenti che deriveranno da una gestione basata su un particolare insieme di metriche e di criteri di successo.

2.3.3 Altre Tecniche per la Selezione dei Test

Mentre molti Test Manager utilizzano il testing basato sul rischio come elemento della loro strategia di test, molti includono altre tecniche.

Una delle tecniche alternative più importanti per sviluppare e prioritizzare le condizioni di test è il testing basato sui requisiti. Il testing basato sui requisiti può utilizzare diverse tecniche come: review delle ambiguità, analisi delle condizioni di test e grafi causa-effetto. Le review delle ambiguità identificano ed eliminano le ambiguità nei requisiti (che servono come base di test), utilizzando spesso una checklist dei difetti comuni dei requisiti (si veda [Weigers03]).

Come descritto in [Craig02], l'analisi delle condizioni di test comporta una lettura attenta delle specifiche dei requisiti per identificare le condizioni di test da coprire. Se a questi requisiti è stata assegnata una priorità, questa può essere usata per allocare l'effort e prioritizzare i test case. In assenza di una priorità assegnata, è difficile determinare l'effort appropriato e l'ordine del testing, senza mischiare il testing basato sui requisiti e il testing basato sul rischio.

Il grafo causa-effetto è discusso nel Syllabus Advanced Test Analyst, nel contesto di coprire le combinazioni di condizioni di test come parte dell'analisi dei test. Tuttavia, ha un uso più ampio, in quanto può ridurre un problema di test molto grande in un numero gestibile di test case e fornire comunque il 100% di copertura funzionale della base di test. Il grafo causa-effetto identifica anche le lacune nella base di test durante la progettazione dei test case, che possono rilevare i difetti nelle fasi iniziali del ciclo di vita dello sviluppo software, quando la progettazione dei test inizia con una versione draft dei requisiti. Il maggior impedimento all'adozione del grafo causa-effetto è la complessità nel creare questi grafi. Esistono strumenti per supportare questo metodo che può essere complicato eseguire manualmente.

Un ostacolo generale al testing basato sui requisiti è che spesso le specifiche dei requisiti sono ambigue, non testabili, incomplete o non esistono affatto. Non tutte le organizzazioni sono motivate a risolvere questi problemi, per cui i tester che devono affrontare tali situazioni devono scegliere un'altra strategia di test.

Un altro metodo che viene qualche volta utilizzato per aumentare l'utilizzo dei requisiti esistenti è la creazione di profili operativi o di utilizzo, un approccio model-based (basato sui modelli), che utilizza un mix di use case, utenti (a volte chiamati Personas), input e output, in modo da descrivere con accuratezza l'utilizzo reale del sistema. Questo consente il testing non solo della funzionalità ma anche dell'usabilità, interoperabilità, affidabilità, sicurezza e prestazioni.

Durante l'analisi e la pianificazione dei test, il team di test identifica i profili di utilizzo e tenta di coprirli con i test case. Il profilo di utilizzo è una stima, sulla base delle informazioni disponibili, dell'utilizzo realistico del software. Questo significa che, come nel testing basato sul rischio, i profili di utilizzo non possono modellare perfettamente la realtà finale. Tuttavia, se sono disponibili informazioni sufficienti e anche l'input degli stakeholder, allora il modello sarà adeguato (si veda [Musa04]).

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Alcuni Test Manager applicano anche approcci metodici, come le checklist per determinare cosa testare, quanto e in quale ordine. Per i prodotti molto stabili, una checklist delle principali aree funzionali e non-funzionali da testare, combinata con un repository di test case esistenti, può essere sufficiente. La checklist fornisce le euristiche per l'allocazione dell'effort e la sequenza dei test, di solito in base al tipo e alla quantità di modifiche applicate. Tali approcci tendono a diventare meno validi quando usati per il testing di modifiche importanti rispetto a quelle minori.

Infine, un altro metodo comune è quello di utilizzare un approccio reattiva. In un approccio reattivo, vengono svolte pochissime attività di analisi, progettazione o implementazione prima dell'esecuzione dei test. Il team di test è focalizzato sulla reattività al prodotto attualmente rilasciato. Cluster di difetti, non appena vengono rilevati, diventano il focus di ulteriore testing. La prioritizzazione e l'allocazione sono completamente dinamiche. L'approccio reattivo può funzionare come complemento di altri approcci, ma quando viene utilizzata in modo esclusivo tende a dimenticare aree dell'applicativo che sono importanti ma non sono sofferenti di un gran numero di difetti.

2.3.4 Prioritizzazione dei Test e Allocazione dell'Effort nel Processo di Test

Il Test Manager, qualunque sia la tecnica o il mix di tecniche che utilizza, deve incorporare quella tecnica nei processi di test e nel progetto. Ad esempio, in cicli di vita sequenziali (p.e., V-model), il team di test seleziona i test, alloca l'effort di test e prioritizza i test durante la fase dei requisiti, con aggiustamenti periodici, mentre nei cicli di vita iterativi o Agile viene richiesto un approccio ripetuto iterazione per iterazione. La pianificazione e il controllo dei test deve considerare il grado con cui i requisiti, i rischi e i profili di utilizzo evolveranno, e rispondere di conseguenza.

Durante l'analisi, la progettazione e l'implementazione dei test, devono essere applicate l'allocazione e la prioritizzazione determinate durante la pianificazione dei test, per eseguire un'analisi e/o modellazione accurata solo per quelle informazioni che non devono essere usate successivamente per guidare il processo di test. Questo si verifica in genere durante la progettazione e l'implementazione.

Durante l'esecuzione del testing, la prioritizzazione definita durante la pianificazione dei test dovrebbe essere seguita, sebbene sia importante aggiornare periodicamente la prioritizzazione sulla base delle informazioni acquisite dopo la prima stesura del test plan. Il Test Manager, oltre a valutare e tracciare i risultati dei test e lo stato dei criteri di uscita, deve anche valutare e fare reporting in base ai rischi, ai requisiti, ai profili di utilizzo, alle checklist e alle altre guide utilizzate per selezionare e prioritizzare i test. Se necessario, la schedulazione dei test dovrebbe basarsi sullo schema di prioritizzazione dei test.

Il Test Manager può anche misurare il grado di completezza del testing come parte del reporting dei risultati e della valutazione dei criteri di uscita. Questo dovrebbe includere la tracciabilità dei test case e dei difetti rilevati verso la base di test. Ad esempio, nel testing basato sul rischio, il tester, in base ai test eseguiti e ai difetti rilevati, può valutare il livello di rischio residuo. Questo supporta l'utilizzo del testing basato sul rischio per determinare il momento giusto per il rilascio. Il test reporting dovrebbe indirizzare i rischi coperti e ancora aperti, così come i benefici ottenuti e quelli non ancora raggiunti. Per un esempio di reporting dei risultati dei test basato sulla copertura rischio, si veda [Black03].

Infine, durante la chiusura dei test, il Test Manager dovrebbe valutare le metriche e i criteri di successo che sono pertinenti alle esigenze e alle aspettative degli stakeholder del testing, incluse le esigenze e le aspettative dei clienti e degli utenti in termini di qualità. Solo quando il testing soddisfa queste esigenze e aspettative, un team di test può dichiararsi veramente efficace.

2.4 Documentazione di Test e altri Prodotti di Lavoro

La documentazione è spesso prodotta come parte delle attività di test management. Mentre i nomi specifici dei documenti di test management e l'ambito di ogni documento tendono a variare, i seguenti sono i tipi più comuni di documenti di test management che si trovano nelle organizzazioni e nei progetti:

- Politica di test: descrive gli obiettivi dell'organizzazione e del testing.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Strategia di test: descrive in generale i metodi per il testing dell'organizzazione, indipendentemente dai singoli progetti
- Master test plan (o test plan di progetto): descrive l'implementazione della strategia di test per un particolare progetto.
- Level test plan (o test plan di fase): descrive le particolari attività da svolgere all'interno di ogni livello di test.

L'organizzazione fisica di questi tipi di documenti può cambiare da contesto a contesto. In alcune organizzazioni e in alcuni progetti, possono essere combinati in un unico documento; in altri, possono essere documenti separati, e in alcuni il loro contenuto può essere intuitivo, non scritto o parte di una metodologia tradizionale per il testing. Le organizzazioni e i progetti più grandi e più formali tendono ad avere tutti questi tipi di documenti come prodotti di lavoro scritti, mentre le organizzazioni e i progetti più piccoli e meno formali tendono ad avere un minor numero di tali prodotti di lavoro scritti. Questo Syllabus descrive ognuno di questi tipi di documenti separatamente, anche se in pratica il contesto organizzativo e progettuale determina il corretto utilizzo di ogni tipo.

2.4.1 Politica di Test

La politica di test descrive perché l'organizzazione esegue il testing. Definisce gli obiettivi generali per il testing che l'organizzazione vuole raggiungere. Questa politica dovrebbe essere sviluppata dallo staff di test management senior nell'organizzazione, in collaborazione con i senior manager dei gruppi di stakeholder del testing.

In alcuni casi, la politica di test sarà complementare oppure un componente di una politica della qualità più ampia. La politica della qualità descrive i valori e gli obiettivi generali del management relativi alla qualità.

In presenza di una politica di test scritta, può essere un breve documento di alto livello che:

- Riassume il valore che l'organizzazione deriva dal testing
- Definisce gli obiettivi del testing, come aumentare la confidenza nel software, rilevare difetti nel software, e ridurre il livello di rischio di qualità (si veda il paragrafo 2.3.1)
- Descrive come valutare l'efficacia e l'efficienza del testing nel raggiungere questi obiettivi
- Delinea il processo tipico di test, magari utilizzando come base il processo fondamentale di test ISTQB
- Specifica come l'organizzazione migliorerà i propri processi di test (si veda il capitolo 5).

La politica di test dovrebbe indirizzare le attività di test per nuovi sviluppi e per la manutenzione. Può anche fare riferimento a standard interni e/o esterni per i prodotti di lavoro del testing e per la terminologia da utilizzare in tutta l'organizzazione.

2.4.2 Strategia di Test

La strategia di test descrive la metodologia generale di test dell'organizzazione. Include il modo in cui il testing viene utilizzato per gestire i rischi di prodotto e di progetto, la suddivisione del testing in livelli e le attività di alto livello associate al testing. (La stessa organizzazione può avere differenti strategie per differenti situazioni, come differenti cicli di vita dello sviluppo software, differenti livelli di rischio o differenti requisiti normativi.) La strategia di test, e i processi e le attività in esso descritti, dovrebbe essere consistente con la politica di test. Dovrebbe fornire i criteri generici di ingresso e uscita del test per l'organizzazione o per uno o più programmi.

Come riportato sopra, le strategie di test descrivono metodologie generali di test, che tipicamente includono:

- Strategie analitiche, come il testing basato sul rischio, in cui il team di test analizza la base di test per identificare le condizioni di test da coprire. Ad esempio, nel testing basato sui requisiti, l'analisi dei test deriva le condizioni di test dai requisiti, i test vengono poi progettati e implementati per coprire queste condizioni. I test vengono successivamente eseguiti, spesso

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

utilizzando la priorità dei requisiti coperti dai test per determinare l'ordine in cui i test saranno eseguiti. I risultati dei test vengono poi tracciati in termini di stato dei requisiti, ad esempio: requisito testato e superato, requisito testato e fallito, requisito non ancora completamente testato, testing del requisito bloccato, ecc.

- Strategie model-based, come i profili operativi, dove il team di test sviluppa un modello (basato su situazioni reali o previste) dell'ambiente in cui esiste il sistema, gli input e le condizioni a cui il sistema è sottoposto, e di come il sistema dovrebbe comportarsi. Ad esempio, nel performance testing model-based di un'applicazione mobile in rapida evoluzione, si potrebbero sviluppare modelli di traffico di rete in entrata e in uscita, utenti attivi e inattivi con il carico di elaborazione risultante, sulla base dell'utilizzo attuale e della crescita del progetto nel tempo. In aggiunta, i modelli potrebbero essere sviluppati considerando l'attuale hardware, software e capacità di dati, la rete e l'infrastruttura dell'ambiente attuale di produzione. I modelli possono anche essere sviluppati per valori ideali attesi e minimi di throughput, tempi di risposta e allocazione delle risorse.
- Strategie metodiche, come quella basata sulle caratteristiche di qualità, dove il team di test utilizza un insieme predeterminato di condizioni di test, basandosi su uno standard di qualità (p.e., ISO 25000 [ISO25000] che sta sostituendo ISO-9126 [ISO9126]), su una checklist o su una collezione di condizioni di test generici e logici relativi ad un particolare dominio, a un'applicazione o a un tipo di test (p.e., testing di sicurezza). Utilizza quindi quell'insieme predeterminato di condizioni di test da una iterazione alla successiva o da una release a quella successiva. Ad esempio, nel testing di manutenzione di un sito web e-commerce semplice e stabile, i tester potrebbero utilizzare una checklist che identifica le funzionalità chiave, gli attributi e i link di ogni pagina. I tester coprirebbero gli elementi rilevanti di questa checklist ogni volta che viene effettuata una modifica al sito.
- Strategie di conformità a un processo o uno standard, come i sistemi medicali soggetti agli standard U.S. FDA (Food and Drug Administration), in cui il team di test segue un insieme di processi definiti da uno standard committee o da un altro gruppo di esperti, dove i processi indirizzano la documentazione, l'appropriata identificazione e l'utilizzo della base di test e degli oracoli del test, nonché l'organizzazione del team di test. Ad esempio, in progetti che seguono lo sviluppo Agile Scrum, in ogni iterazione i tester analizzano le user story che descrivono particolari funzionalità, stimano l'effort del test per ogni funzionalità come parte del processo di pianificazione dell'iterazione, identificano le condizioni di test (spesso chiamate criteri di accettazione) per ogni user story, eseguono i test che coprono tali condizioni, e tracciano lo stato di ogni user story (non testata, fallita, superata) durante l'esecuzione dei test.
- Strategie reattive, come l'utilizzo di attacchi basati sui difetti, dove il team di test aspetta di progettare e implementare i test fino a quando non viene rilasciato il software, reagendo poi al sistema attuale sotto test. Ad esempio, quando si utilizza il testing esplorativo su un applicativo menu-based, potrebbe essere definito un insieme di test charter che corrispondono alle funzionalità, alle schermate e alle selezioni di menu. Ad ogni tester viene assegnato un insieme di test charter, che utilizza per strutturare le proprie sessioni di testing esplorativo. I testers riportano periodicamente i risultati delle sessioni di test al Test Manager, che può rivedere i charter in base ai risultati.
- Strategie consultive, come il testing guidato dall'utente, dove il team di test si affida all'input di uno o più stakeholder chiave per determinare le condizioni di test da coprire. Ad esempio, nel testing di compatibilità di un applicazione web eseguito in outsourcing, un'azienda può fornire al Service Provider del testing in outsourcing una lista prioritizzata di versioni del browser, di software anti-malware, di sistemi operativi, di tipi di connessione e di altre opzioni di configurazione che devono essere valutate rispetto all'applicazione web. Il Service Provider del testing può quindi utilizzare tecniche come il testing pairwise (per le opzioni ad alta priorità) e il partizionamento di equivalenza (per le opzioni con priorità più bassa) per generare i test.
- Strategie avverse alla regressione, come ad esempio l'automazione estesa, dove il team di test utilizza diverse tecniche per gestire il rischio di regressione, in particolare con l'automazione di regression test funzionali e/o non-funzionali ad uno o più livelli. Ad esempio, per il regression testing di un applicazione web-based, i tester possono utilizzare uno strumento GUI-based di

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

test automation per automatizzare gli use case tipici e di eccezione. Questi test vengono quindi eseguiti ogni volta che l'applicazione viene modificata.

Strategie differenti possono essere combinate fra loro. Le specifiche strategie selezionate dovrebbero essere appropriate alle esigenze e ai mezzi dell'organizzazione, e le organizzazioni possono adattare le strategie per adeguarsi a particolari situazioni e progetti.

La strategia di test può descrivere i livelli di test da eseguire. In tali casi, dovrebbe essere una guida sui criteri di ingresso e di uscita di ogni livello e sulle relazioni tra i livelli (p.e., la divisione degli obiettivi di copertura del test).

La strategia di test può anche descrivere i seguenti:

- Procedure di integrazione
- Specifiche tecniche di test
- Indipendenza del testing (che può variare in base al livello)
- Standard obbligatori e opzionali
- Ambienti di test
- Test automation
- Strumenti di test
- Riusabilità dei prodotti di lavoro software e dei prodotti di lavoro di test
- Test confermativo (re-testing) e regression testing
- Controllo e reporting del test
- Misure e metriche di test
- Defect management
- Approccio di configuration management del testware
- Ruoli e responsabilità

Potrebbero essere necessarie differenti strategie di test per il breve e lungo termine- Differenti strategie di test sono adatte per diverse organizzazioni e progetti. Ad esempio, in presenza di applicazioni security-critical o safety-critical, una strategia più intensiva può essere più appropriata rispetto ad altri casi. Inoltre, la strategia di test può anche variare in base ai diversi modelli di sviluppo.

2.4.3 Master Test Plan

Il master test plan copre tutto il lavoro del testing da svolgere in un particolare progetto, compresi i particolari livelli da eseguire e le relazioni tra questi livelli, e tra i livelli di test e le corrispondenti attività di sviluppo. Il master test plan dovrebbe descrivere come i tester implementeranno la strategia di test per il progetto considerato (cioè, l'approccio del test). Il master test plan dovrebbe essere consistente con la politica e la strategia di test e, in specifiche aree dove non lo è, dovrebbe spiegare queste deviazioni ed eccezioni, includendo qualsiasi potenziale impatto derivante dalle deviazioni. Ad esempio, se la strategia di test di un'organizzazione prevede di effettuare un regression testing completo per un sistema non modificato, immediatamente prima del rilascio, ma il progetto attuale non ha alcun regression testing, il test plan dovrebbe spiegare perché è stato pianificato e cosa verrà fatto per mitigare i rischi derivanti dalla modifica della strategia usuale. Il test plan dovrebbe anche includere una spiegazione di eventuali altri effetti attesi da questa variazione. Ad esempio, saltare il regression testing potrebbe richiedere la schedulazione di una release di manutenzione un mese dopo il primo rilascio del progetto. Il master test plan dovrebbe essere complementare al piano di progetto o alle guide di manutenzione in produzione, poiché dovrebbe descrivere l'effort del test che fa parte del progetto odella manutenzione (operation).

Mentre il contenuto specifico e la struttura del master test plan variano in base all'organizzazione, agli standard di documentazione e alla formalità del progetto, i contenuti tipici di un master test plan includono:

- Elementi da testare e da non testare
- Caratteristiche di qualità da testare e da non testare

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Schedulazione e budget del testing (che dovrebbe essere allineato al budget di progetto o al budget di manutenzione)
- Cicli di esecuzione dei test e loro relazione con il software release plan
- Relazioni e rilasci tra il testing e le altre persone o unità organizzative
- Definizione di quali elementi di test sono in ambito (in scope) o fuori ambito (out of scope) per ogni livello descritto
- Specifici criteri di ingresso, di continuazione (sospensione/ripresa) e di uscita per ogni livello, e le relazioni tra i livelli
- Rischi di progetto del test
- Governance globale dell'effort del test
- Responsabilità per l'esecuzione di ogni livello di test
- Input e output di ogni livello di test

Per progetti piccoli o di manutenzione, dove viene formalizzato solo un livello di test, il master test plan e il test plan per quel livello formalizzato saranno spesso combinati in un unico documento. Ad esempio, se il test del sistema è l'unico livello formalizzato, con il testing di componente e il testing di integrazione eseguiti in modo informale dagli sviluppatori, e con il testing di accettazione eseguito in modo informale dai clienti come parte di un processo di beta test, in questo caso il test plan di sistema può includere tutti gli elementi menzionati in questo paragrafo.

In aggiunta, il testing dipende in genere da altre attività di progetto. Se queste attività non dovessero essere sufficientemente documentate, in particolare per quanto riguarda la loro influenza e la loro relazione con il testing, gli argomenti relativi a queste attività possono essere coperti nel master test plan (o in un level test plan appropriato). Ad esempio, se il processo di configuration management non è documentato, il test plan dovrebbe specificare come gli oggetti di test devono essere rilasciati al team di test.

2.4.4 Level Test Plan

Il level test plan descrive le particolari attività da eseguire all'interno di ogni livello di test o, in alcuni casi, di tipi di test. Il level test plan espande, dove necessario, il master test plan per lo specifico livello o tipo di test da documentare. Fornisce dettagli sulla schedulazione, sulle attività e sulle milestone che non sono necessariamente descritti dal master test plan. In aggiunta, nel caso in cui specifici standard e template si applichino alla specifica dei test a livelli differenti, questi dettagli sarebbero coperti nei level test plan.

Nei progetti o nelle manutenzioni meno formali, un singolo test plan è spesso l'unico documento di gestione del testing che viene scritto. In tale caso, alcuni degli elementi informativi menzionati nei precedenti paragrafi potrebbero essere inseriti in questo documento.

Per i progetti Agile, i test plan di sprint o di iterazione possono sostituite i level test plan.

2.4.5 Gestione dei Rischi di Progetto

Una parte importante di una pianificazione appropriata include la gestione dei rischi di progetto. I rischi di progetto possono essere identificati utilizzando processi similari a quelli descritti nel paragrafo 2.3. Quando i rischi di progetto sono stati identificati, dovrebbero essere comunicati e gestiti dal project manager. Non sempre tali rischi possono essere sotto la responsabilità dell'organizzazione del testing. Tuttavia, alcuni rischi di progetto possono e dovrebbero essere mitigati con successo dal Test Manager, come:

- Readiness dell'ambiente di test e degli strumenti di test
- Disponibilità e qualifica dello staff di test
- Mancanza di standard, regole e tecniche per l'effort del test

Gli approcci alla gestione dei rischi di progetto includono la preparazione anticipata del testware, il pre-testing degli ambienti di test, il pre-testing delle versioni iniziali del prodotto, l'applicazione dei criteri di

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

ingresso di test più severi, requisiti di testabilità, partecipazione alle review delle versioni iniziali dei prodotti di lavoro del progetto, partecipazione al change management, e monitoraggio dell'avanzamento e della qualità del progetto.

Una volta identificato e analizzato il rischio del progetto, esistono quattro principali opzioni per gestire tale rischio:

1. Mitigare il rischio attraverso misure preventive per ridurne la probabilità e/o l'impatto
2. Predisporre contingency plan per ridurre l'impatto se il rischio si concretizza
3. Trasferire il rischio a un altro gruppo per la gestione
4. Ignorare o accettare il rischio

La selezione dell'opzione migliore dipende dai benefici e dalle opportunità create dall'opzione, nonché dal costo e, potenzialmente, dai rischi aggiuntivi associati all'opzione. Quando viene identificato un contingency plan per un rischio di progetto, la best practice è identificare un trigger (che determinerà quando e come il contingency plan viene invocato) e un proprietario (che realizzerà il contingency plan).

2.4.6 Altri Prodotti di Lavoro del Testing

Il testing prevede la creazione di una serie di prodotti di lavoro, quali i defect report, le specifiche dei test case e i test log. La maggior parte di questi prodotti di lavoro sono creati dai Test Analyst e Technical Test Analyst. Nel Syllabus Advanced Test Analyst, sono discusse considerazioni per la produzione e la documentazione di questi prodotti di lavoro. Il Test Manager dovrebbe garantire la consistenza e la qualità di questi prodotti di lavoro attraverso le seguenti attività:

- Definire e monitorare le metriche che monitorano la qualità di questi prodotti di lavoro, come la percentuale di defect report rifiutati.
- Lavorare con i Test Analyst e Technical Test Analyst per selezionare e personalizzare i template appropriati per questi prodotti di lavoro.
- Lavorare con i Test Analyst e Technical Test Analyst per definire gli standard per questi prodotti di lavoro, come ad esempio il grado di dettaglio necessario nei test, nei log e nei report.
- Eseguire la review dei prodotti di lavoro del testing, utilizzando le tecniche appropriate e coinvolgendo i partecipanti e gli stakeholder appropriati.

L'estensione, il tipo e la specificità della documentazione di test possono, tra le altre considerazioni, essere influenzati dal ciclo di vita dello sviluppo software adottato, dalle normative e standard applicabili, dai rischi di progetto e dai rischi di qualità associati con il particolare sistema da sviluppare.

Esistono diverse sorgenti di template per i prodotti di lavoro del testing, come IEEE 829 [IEEE829]. E' importante per il Test Manager ricordare che i documenti IEEE 829 sono stati progettati per l'utilizzo in qualsiasi settore industriale. Come tale, i template contengono un alto livello di dettaglio che possono o meno essere applicabili a una particolare organizzazione. È best practice adattare (fare il tailoring) i documenti IEEE 829 per creare template standard da utilizzare in una particolare organizzazione. Una consistente applicazione dei template riduce i requisiti di formazione e aiuta a unificare i processi in tutta l'organizzazione.

Il testing comporta anche la creazione di test report, che sono generalmente prodotti dal Test Manager e sono descritti più avanti in questo capitolo.

2.5 Stima del Test

La stima, come attività di gestione, è la creazione di un obiettivo approssimato dei costi e delle date di completamento associati alle attività coinvolte in un particolare progetto o nella manutenzione (operation). Le migliori stime:

- Rappresentano la conoscenza collettiva di esperti professionisti e hanno il supporto dei partecipanti coinvolti
- Forniscono cataloghi specifiche e dettagliate di costi, risorse, attività e persone coinvolte

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Forniscono, per ogni attività stimata, il costo, lo sforzo e la durata più probabili

La stima nel software engineering e nel system engineering è nota da tempo per essere irta di difficoltà, sia tecniche sia politiche, anche se sono state definite best practice di project management. La stima del test è l'applicazione di queste best practice alle attività di test associate a un progetto o a una manutenzione.

La stima dovrebbe coprire tutte le attività coinvolte nel processo di test come descritto nel Capitolo 1. Le stime del costo, dell'effort e, soprattutto, della durata di esecuzione dei test sono spesso quelle di maggior interesse per il management, in quanto l'esecuzione dei test si trova tipicamente sul cammino critico del progetto. Tuttavia, le stime di esecuzione dei test tendono ad essere difficili da generare e inaffidabili, quando la qualità complessiva del software è bassa o sconosciuta. In aggiunta, la familiarità e l'esperienza con il sistema probabilmente influenzeranno la qualità della stima. Una pratica comune è di stimare il numero di test case richiesti durante l'esecuzione dei test, ma questa funziona solo se si può assumere che il numero di difetti nel software da testare sia basso. Le assunzioni fatte durante la stima dovrebbero essere sempre documentate come parte della stima.

La stima del test dovrebbe considerare tutti i fattori che possono influenzare il costo, l'effort e la durata delle attività di test. Questi fattori includono (ma non sono limitati a) i seguenti elementi:

- Livello di qualità del sistema richiesto
- Dimensioni del sistema da testare
- Dati storici sul testing da progetti di test precedenti, che possono essere arricchiti con dati di settore o dati di riferimento di altre organizzazioni
- Fattori di processo, che includono la strategia di test, il ciclo di vita dello sviluppo o della manutenzione, maturità del processo, e la precisione della stima del progetto
- Fattori legati ai materiali, che includono il test automation e gli strumenti, gli ambienti di test, i dati di test, gli ambienti di sviluppo, la documentazione di progetto (p.e., requisiti, progettazione, ecc.), e prodotti di lavoro del test riutilizzabili
- Fattori legati alle persone, che includono aspettative e commitment dell'executive management, del senior management, dei manager e dei leader tecnici; esperienza, competenza, attitudini, stabilità e relazioni del team di progetto; supporto dell'ambiente di test e di debugging, disponibilità di consulenti competenti qualificati; conoscenza del dominio
- Complessità del processo, tecnologia, organizzazione, numero di stakeholder del testing, composizione e posizione geografica dei sotto-team
- Necessità significative di formazione, orientamento e coaching
- Inserimento o sviluppo di nuovi strumenti, tecnologie, processi, tecniche, hardware customizzato, oppure grande quantità di testware
- Requisiti di specifiche di test con un alto livello di dettaglio, in particolare per la conformità con uno standard di documentazione non familiare
- Schedulazione complessa della ricezione dei componenti, in particolare per il testing di integrazione e lo sviluppo dei test
- Volatilità dei dati di test (ad esempio, dati sensibili nel tempo)

Anche la qualità del software rilasciato per il testing è un fattore importante che i Test Manager dovrebbero prendere in considerazione nella loro stima. Ad esempio, se gli sviluppatori hanno adottato delle best practice come lo unit testing automatizzato e il continuous integration, almeno il 50% dei difetti verrà eliminato prima del rilascio del codice al team di test (si veda [Jones11] per l'efficacia nella rimozione dei difetti con queste pratiche). Alcuni hanno riportato che le metodologie Agile, incluso il test-driven development, forniscono livelli di qualità più elevati ai rilasci per il testing.

La stima può essere fatta sia bottom-up sia top-down. Le tecniche che possono essere utilizzate nelle stime del test, utilizzate singolarmente o in combinazione, includono:

- Intuizione, supposizioni ed esperienza passata
- Work Breakdown Structure (WBS)
- Sessioni di stima del team (p.e. Wide Band Delphi)

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Standard e normative aziendali
- Percentuali dell'effort di progetto complessivo o dei livelli di staff (p.e., numero di tester rispetto agli sviluppatori)
- Storia e metriche dell'organizzazione, che includono modelli metrics-based che stimano il numero di difetti, il numero di cicli di test, il numero di test case, l'effort medio di ogni test, e il numero di cicli di regression test
- Medie e modelli predittivi di settore, come i function point, le linee di codice, l'effort stimato di uno sviluppatore, o altri parametri di progetto (p.e., si veda [Jones07]).

Nella maggior parte dei casi, le stime, una volta preparate, devono essere consegnate al management, insieme ad una giustificazione (si veda paragrafo 2.7). Spesso, ne derivano delle negoziazioni, che si traducono in una rielaborazione della stima. Idealmente, la stima finale del test rappresenta il miglior bilanciamento possibile tra gli obiettivi dell'organizzazione e gli obiettivi di progetto, nelle aree della qualità, della schedulazione, del budget e delle funzionalità.

È importante ricordare che qualsiasi stima si basa sulle informazioni disponibili al momento in cui è stata preparata. Nelle fasi iniziali di un progetto l'informazione può essere molto limitata. In aggiunta, informazioni disponibili a inizio progetto possono cambiare nel tempo. Per rimanere accurate, le stime dovrebbero essere aggiornate per riflettere informazioni nuove e modificate.

2.6 Definizione e Utilizzo delle Metriche di Test

Un luogo comune del management dice che ciò che viene misurato viene fatto. Inoltre, ciò che non si misura non viene fatto, perché ciò che non viene misurato è facile da ignorare. Pertanto, è importante che venga definito l'insieme appropriato di metriche per ogni attività, compreso il testing.

Le metriche di test possono essere classificate come appartenenti a una o più delle seguenti categorie:

- Metriche di progetto, che misurano l'avanzamento rispetto a criteri di uscita di progetto stabiliti, come la percentuale di test case eseguiti, superati e falliti.
- Metriche di prodotto, che misurano alcuni attributi del prodotto, come l'estensione di quello che è stato testato o la densità dei difetti.
- Metriche di processo, che misurano la capacità del processo di test o di sviluppo, come la percentuale di difetti rilevati dal testing.
- Metriche delle risorse, che misurano la capacità di singoli individui o gruppi, come l'implementazione di test case in una schedulazione definita.

Ogni metrica può appartenere a due, tre o anche quattro categorie. Ad esempio, un trend chart che visualizza il tasso giornaliero di rilevazione dei difetti può essere associato ad un criterio di uscita (zero nuovi difetti rilevati in una settimana), alla qualità del prodotto (il testing non può rilevare ulteriori difetti), e alla capacità del processo di test (il testing rileva un gran numero di difetti all'inizio del periodo di esecuzione dei test).

Le metriche selle persone sono particolarmente sensibili. I manager a volte confondono metriche che sono principalmente metriche di processo con metriche delle risorse, causando risultati disastrosi quando le persone agiscono per influenzare le metriche in un modo a loro favorevole. La motivazione e la valutazione appropriata delle persone del test viene discussa nel Capitolo 7 di questo Syllabus e nel Syllabus Expert Test Management [ISTQB ETL SYL].

Nell'Advanced Level, viene data particolare enfasi all'uso di metriche per misurare l'avanzamento del testing, cioè le metriche di progetto. Alcune delle metriche di progetto utilizzate per misurare l'avanzamento del test si riferiscono anche al prodotto e al processo. Per ulteriori informazioni sulla gestione delle metriche di prodotto e di processo si consideri il Syllabus Expert Test Management. Ulteriori informazioni sull'utilizzo delle metriche di processo sono presenti nel Syllabus Expert Improving the Test Process [ISTQB ITP SYL].

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

L'utilizzo di metriche consente ai tester di documentare i risultati (report) in modo consistente, e facilita un monitoraggio coerente dell'avanzamento nel tempo. Ai Test Manager viene spesso richiesto di presentare metriche durante diversi meeting, dove possono partecipare livelli multipli di stakeholder, che vanno dallo staff tecnico all'executive management. Poiché le metriche sono a volte utilizzate per determinare il successo complessivo di un progetto, dovrebbe essere dedicata grande attenzione nel determinare cosa tracciare, quanto spesso documentarlo e quale metodo utilizzare per presentare le informazioni. In particolare, un Test Manager deve considerare quanto segue:

- Definizione delle metriche. Dovrebbe essere definito un insieme limitato di metriche utili. Le metriche dovrebbero essere definite in base a specifici obiettivi per il progetto, per il processo e/o per il prodotto. Le metriche dovrebbero essere definite in modo bilanciato, poiché una singola metrica può dare una falsa impressione dello stato o del trend. Una volta definite queste metriche, deve essere concordata la loro interpretazione da tutti gli stakeholder, per evitare confusione durante la discussione delle misure. Esiste spesso una tendenza a definire troppe metriche, invece di concentrarsi su quelle più pertinenti.
- Monitoraggio delle metriche. L'integrazione e il reporting delle metriche dovrebbero essere il più possibile automatizzate, per ridurre il tempo trascorso nel raccogliere ed elaborare le misure. Variazioni delle misure nel tempo per una specifica metrica possono riflettere informazioni differenti dall'interpretazione concordata in fase di definizione della metrica. Il Test Manager dovrebbe essere preparato ad analizzare attentamente eventuali divergenze nelle misure rispetto alle aspettative, e le motivazioni di tale divergenza.
- Reporting delle metriche. L'obiettivo è di fornire al management una comprensione immediata delle informazioni. Le presentazioni possono mostrare uno snapshot di una metrica in un certo momento, o mostrare l'evoluzione di una metrica nel tempo in modo che possa essere valutato il trend.
- Validità delle metriche. Un Test Manager deve anche verificare la validità delle informazioni da fornire. Le misure effettuate per una metrica possono non riflettere lo stato reale di un progetto o possono mostrare un trend eccessivamente positivo o negativo. Prima di qualsiasi presentazione dei dati, il Test Manager deve eseguire una review sia dell'accuratezza sia del messaggio che desidera trasmettere.

Esistono cinque dimensioni principali per cui è monitorato l'avanzamento del test:

- Rischi (di qualità) del prodotto
- Difetti
- Test
- Copertura del test
- Confidenza

I rischi di prodotto, i difetti, i test e la copertura del test possono essere (e spesso lo sono) misurati e documentati con appropriate modalità nel corso del progetto o della manutenzione. Se queste misure sono relative ai criteri di uscita definiti nel test plan, possono fornire uno standard oggettivo per giudicare il completamento dell'effort del test. La confidenza è misurabile attraverso indagini o utilizzando la copertura come metrica surrogata; tuttavia, la confidenza è spesso documentata in modo soggettivo.

Le metriche relative ai rischi di qualità includono:

- Percentuale dei rischi completamente coperti dai test superati
- Percentuale dei rischi per i quali alcuni o tutti i test sono falliti
- Percentuale dei rischi non ancora completamente testati
- Percentuale dei rischi coperti, suddivisi per categoria di rischio
- Percentuale dei rischi identificati dopo l'analisi dei rischi di qualità iniziale

Le metriche relative ai difetti includono:

- Numero cumulativo di difetti rilevati (documentati) rispetto al numero cumulativo di difetti risolti
- Mean Time Between Failure o tasso di failure rilevati
- Ripartizione del numero o percentuale di difetti classificati come:
  - Elementi o componenti di test particolari

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

  - Root cause
  - Sorgente dei difetti (ad es. specifiche dei requisiti, nuova funzionalità, regressione, ecc.)
  - Rilasci del test
  - Fase di introduzione, rilevazione e correzione del difetto
  - Priorità/severità
  - Report rifiutati o duplicati
- Trend nel tempo trascorso dal reporting alla risoluzione del difetto
- Numero di bug fix che hanno introdotto nuovi difetti (chiamati anche bug figli)

Le metriche relative ai test includono:

- Numero totale di test case pianificati, specificati (Implementati), eseguiti, superati, falliti, bloccati e non eseguiti
- Stato dei regression test e dei test confermativi
- Ore di test giornaliere pianificate rispetto alle ore effettivamente svolte
- Disponibilità dell'ambiente di test (percentuale di ore di test pianificate quando l'ambiente di test è utilizzabile dal team di test)

Le metriche relative alla copertura del test includono:

- Copertura dei requisiti e degli elementi di progettazione
- Copertura dei rischi
- Copertura dell'ambiente e della configurazione
- Copertura del codice

È importante che il Test Manager comprenda come interpretare e utilizzare le metriche di copertura al fine di capire e documentare lo stato del test. Per i livelli di test più alti, come il testing di sistema, il testing di integrazione di sistemi e il testing di accettazione, le basi di test principali sono di solito prodotti di lavoro come le specifiche dei requisiti, le specifiche di progettazione, gli use case, le user story, i rischi di prodotto, gli ambienti e le configurazioni supportate. Metriche strutturali di copertura del codice si applicano di più ai livelli più bassi di test, come il testing di componente (p.e., copertura delle istruzioni e delle decisioni) e il testing di integrazione dei componenti (p.e., copertura delle interfacce). Mentre i Test Manager possono utilizzare le metriche di copertura del codice per misurare l'estensione che i test esercitano sulla struttura del sistema sotto test, il reporting dei risultati dei test di livello più alto non dovrebbero in genere riportare metriche di copertura del codice. Inoltre, i Test Manager dovrebbero comprendere che, anche se il testing di componente e il testing di integrazione dei componenti raggiunge il 100% di copertura strutturale, rimangono difetti e rischi di qualità da indirizzare nei livelli più alti di test.

Le metriche possono essere correlate anche alle attività del processo fondamentale di test (descritto sia nel Syllabus Foundation Level sia in questo Syllabus). In questo modo, le metriche possono essere utilizzate in tutto il processo di test per monitorarne il processo di test stesso e l'avanzamento verso gli obiettivi di progetto.

Le metriche per monitorare le attività di pianificazione e controllo dei test possono includere:

- Copertura dei rischi, dei requisiti, e degli altri elementi della base di test
- Rilevazione dei difetti
- Ore effettive rispetto alle ore pianificate per sviluppare il testware ed eseguire i test case

Le metriche per monitorare le attività di analisi dei test possono includere:

- Numero di condizioni di test identificate
- Numero di difetti rilevati durante l'analisi dei test (ad es., Identificando i rischi o altre condizioni di test utilizzando la base di test)

Le metriche per monitorare le attività di progettazione dei test possono includere:

- Percentuale delle condizioni di test coperte dai test case
- Numero di difetti rilevati durante la progettazione dei test (ad es., nello sviluppo dei test rispetto alla base di test)

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Le metriche per monitorare le attività di implementazione dei test possono includere:

- Percentuale degli ambienti di test configurati
- Percentuale di record di dati di test caricati
- Percentuale di test case automatizzati

Le metriche per monitorare le attività di esecuzione dei test possono includere:

- Percentuale di test case pianificati, eseguiti, superati e falliti
- Percentuale delle condizioni di test coperte dai test case eseguiti (e/o superati)
- Percentuale dei difetti tracciati/risolti pianificati rispetto agli effettivi
- Percentuale della copertura pianificata rispetto a quella effettivamente raggiunta

Le metriche per monitorare le attività di avanzamento e completamento dei test includeranno un mapping rispetto alle milestone, ai criteri di ingresso e uscita (criteri definiti e approvati nella pianificazione dei test) e possono includere:

- Numero di condizioni di test, test case o specifiche di test pianificati rispetto a quelli eseguiti, suddivisi per superati o falliti
- Numero totale di difetti, spesso suddivisi per severità, priorità, stato attuale, sottosistema impattato, o altre classificazioni (si veda capitolo 4)
- Numero di modifiche richieste, accettate, realizzate e testate
- Costo pianificato rispetto al costo effettivo
- Durata pianificata rispetto alla durata effettiva
- Date pianificate delle milestone di test rispetto alle date effettive
- Date pianificate delle milestone di progetto relative al testing (p.e., freeze del codice) rispetto alle date effettive
- Stato dei rischi (di qualità) di prodotto, spesso suddivisi in mitigati e non mitigati, aree di maggior rischio, nuovi rischi rilevati dopo l'analisi dei test, ecc.
- Percentuale dell'effort, dei costi o del tempo che sono stati persi a causa di eventi bloccanti o di modifiche pianificate
- Stato dei regression test e dei test confermativi (re-test)

Le metriche per monitorare le attività di chiusura dei test possono includere:

- Percentuale dei test case eseguiti, superati, falliti, bloccati e non eseguiti durante l'esecuzione dei test
- Percentuale dei test case verificati nel repository come test case riutilizzabili
- Percentuale dei test case automatizzati sul totale, o percentuale dei test case automatizzati pianificati rispetto agli effettivi
- Percentuale dei test case inseriti nei regression test
- Percentuale di defect report risolti/non risolti
- Percentuale dei prodotti del lavoro del test identificati e archiviati

Inoltre, tecniche standard di project management, quali work breakdown structure, sono spesso utilizzate per monitorare il processo di test. Nei team Agile, il testing è parte dell'avanzamento di una user story nel burn-down chart. Nei casi in cui vengono utilizzate tecniche di Lean management, l'avanzamento del testing a livello di user story è spesso controllato dallo spostamento della user story tra le colonne della Kanban board.

Dato un insieme definito di metriche, le misure possono essere documentate nel linguaggio naturale in forma narrativa, numericamente in tabelle o attraverso grafici. Le misure possono essere utilizzate per vari scopi, che includono:

- Analisi, per scoprire quali trend e cause possono essere percettibili attraverso i risultati del test
- Reporting, per comunicare i risultati dei test ai partecipanti di progetto e agli stakeholder
- Controllo, per modificare la sequenza del testing o del progetto nel suo insieme, e permonitorare i risultati di tale modifica.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Le modalità appropriate per raccogliere queste misure, analizzarle e preparare report dipendono dagli obiettivi, dalla necessità di specifiche informazioni e dalle capacità delle persone che utilizzeranno tali misure. Inoltre, il contenuto specifico dei test report dovrebbe variare in base all'audience.

Ai fini del controllo dei test, è essenziale che le metriche di tutto il processo di test (una volta completata la pianificazione dei test) forniscano al Test Manager le informazioni necessarie per guidare l'effort del test verso il raggiungimento della mission, delle strategie e degli obiettivi di test. Pertanto, la pianificazione deve prendere in considerazione queste necessità di informazioni, e il monitoraggio deve includere la raccolta di tutte le metriche necessarie. Il volume di informazioni necessarie e l'effort speso per collezionarle dipende da diversi fattori di progetto, tra cui le dimensioni, la complessità e i rischi.

Il controllo dei test deve rispondere alle informazioni generate dal testing e alle modifiche delle condizioni progettuali o ambientali. Ad esempio, se il testing dinamico rileva cluster di difetti in aree che precedentemente si riteneva poco probabile contenessero molti difetti, o se il periodo di esecuzione dei test viene ridotto a causa di un ritardo nella partenza del testing, devono essere riviste l'analisi dei rischi e la pianificazione. Questo potrebbe causare la riassegnazione delle priorità ai test e la riallocazione dell'effort di esecuzione dei test rimanenti.

Se il test progress report evidenziasse delle discrepanze rispetto al test plan, dovrebbe essere eseguito il controllo dei test. Il controllo dei test ha lo scopo di ridefinire il progetto e/o il testing in una direzione dimaggior successo. Quando si utilizzano i risultati dei test per influenzare o misurare l'effort per controllare il progetto, dovrebbero essere considerate le seguenti opzioni:

- Revisione dell'analisi dei rischi di qualità, delle priorità dei test e/o dei test plan
- Inserimento di risorse o l'aumento dell'effort di progetto o del test
- Posticipo della data di rilascio
- Rilassamento o rafforzamento dei criteri di uscita
- Modifica dell'ambito (funzionale e/ non-funzionale) del progetto

L'implementazione di tali opzioni richiede in genere un consenso tra gli stakeholder di progetto o di operation, e l'approvazione da parte dei project manager o degli operation manager.

Le informazioni rilasciate in un test report dovrebbero dipendere in gran parte dalla necessità di informazioni da parte dell'audience target, ad esempio il project management o il business management. Un project manager è probabilmente interessato ad avere informazioni dettagliate sui difetti; un business manager potrebbe considerare lo stato dei rischi del prodotto come un'informazione chiave.

2.7 Valore di Business del Testing

Il Test Manager deve lavorare per ottimizzare il testing in modo da rilasciare un buon valore di business. Un testing eccessivo non rilascia un buon valore di business, poiché il testing causerà ritardi non giustificabili e costerà più dei benefici che porta. Un testing troppo ridotto non rilascia un buon valore di business, poiché troppi difetti saranno rilasciati agli utenti. L'ottimale è compreso tra questi due estremi. Il Test Manager deve aiutare gli stakeholder del testing a comprendere questo ottimale e il valore rilasciato dal testing.

Mentre la maggior parte delle organizzazioni considerano il testing valutabile genericamente, pochi manager, compresi i Test Manager, sono in grado di quantificare, descrivere, o articolare quel valore. Inoltre, molti Test Manager, test lead e tester si focalizzano su dettagli tattici del testing (aspetti specifici dell'attività o del livello di test), ignorando gli aspetti strategici più ampi (di livello più alto) relativi al testing, che altri partecipanti del progetto, in particolare i manager, tengono conto.

Il testing rilascia valore all'organizzazione, al progetto e/o a operation sia in modo quantitativo sia qualitativa:

- I valori quantitativi comprendono la rilevazione di difetti che vengono corretti prima del rilascio, la rilevazione di difetti noti prima del rilascio (non corretti ma documentati, eventualmente con

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

workaround), la riduzione del rischio attraverso l'esecuzione dei test, e il rilascio di informazioni sullo stato del progetto, del processo e del prodotto
- I valori qualitativi includono i valori qualitativi includono una percezione migliorata della qualità, rilasci più regolari e pianificati, maggiore confidenza, protezione da responsabilità legali e riduzione del rischio di fallimento totale o addirittura di vite.

I Test Manager dovrebbero comprende quali di questi valori si applicano alla loro organizzazione, al loro progetto e/o manutenzione, ed essere in grado di comunicare sul testing nei termini di questi valori.

Un metodo consolidato per misurare il valore e l'efficienza del testing in modo quantitativo è chiamato costo della qualità (o talvolta, costo della non-qualità). Il costo della qualità comporta la classificazione dei costi di progetto e di esercizio in quattro categorie:

- Costi di prevenzione. Ad esempio, la formazione degli sviluppatori su come scrivere codice più manutenibile o più sicuro.
- Costi di rilevazione. Ad esempio, la scrittura dei test case, la configurazione degli ambienti di test, e la review dei requisiti.
- Costi di failure interno. Ad esempio, la correzione dei difetti rilevati durante il testing o le review, prima del rilascio in produzione.
- Costi di failure esterno. Ad esempio, i costi di supporto associati al software difettoso rilasciato ai clienti.

Una parte del budget del testing è un costo di rilevazione (cioè il denaro che sarebbe speso anche se nessun difetto fosse stato rilevato dai tester, come il denaro speso per sviluppare i test), mentre il rimanente è un costo di failure interno (cioè il costo effettivo associato ai difetti rilevati). I costi totali di rilevazione e di failure interno sono in genere ben al di sotto dei costi di failure esterno, rendendo il testing un eccellente valore di business. Una volta determinati i costi di queste quattro categorie, i Test Manager sono in grado di creare un convincente business case per il testing.

Maggiori informazioni sul valore di business del testing, che include il costo della qualità, è presente in [Black2003].

2.8 Testing Distribuito, Outsourced o Insourced

In molti casi, parte o forse anche tutto l'effort del test è svolto da persone in luoghi diversi, appartenenti a diverse aziende, o indipendenti dal gruppo di progetto. Se l'effort del test è svolto in diverse sedi, si parla di testing distribuito. Se l'effort del test è svolto da persone che non sono dipendenti della società (non sono colleghi) e che non sono co-locate con il team di progetto, si parla di testing in outsourcing. Se l'effort del test è svolto da persone che sono co-locate con il team di progetto ma sono colleghi, si parla di testing in insourcing.

Comune a tutti questi tipi di effort del test è la necessità di canali di comunicazione chiari, e di aspettative sulla mission, sulle attività e sui rilasci ben definite e condivise. Il team di progetto deve fare minor affidamento su canali di comunicazione informali, come le conversazioni di corridoio o eventi sociali con i colleghi. È importante aver definito le modalità con cui la comunicazione dovrebbe svolgersi, inclusi anche argomenti da indirizzare come l'escalation dei problemi, i tipi di informazione da comunicare e i metodi di comunicazione da utilizzare. Ogni persona coinvolta nelle relazioni del team deve avere una chiara comprensione del proprio ruolo e responsabilità, nonché dei ruoli e responsabilità delle altre persone coinvolte, per evitare incomprensioni e aspettative irrealistiche. Differenze di località, fuso orario, cultura e lingua rendono ancora più probabili la presenza di problemi di comunicazione e di incomprensione.

Comune a tutti questi tipi di effort del test è anche la necessità di allineamento delle metodologie. Benché un disallineamento delle metodologie si possa verificare in qualsiasi progetto, è più probabile che emerga in situazioni dove il lavoro è distribuito e/o realizzato da entità esterne. Se due gruppi di test utilizzano metodologie diverse o se il gruppo di test utilizza una metodologia diversa da quella del gruppo di sviluppo o del project management, possono manifestarsi problemi importanti, in particolare

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

durante l'esecuzione dei test. Ad esempio, se un gruppo di progetto utilizza lo sviluppo Agile, mentre il fornitore di servizi del testing applica una metodologia di test predefinita che presuppone un ciclo di vita sequenziale, la tempistica e la natura dei rilasci degli elementi di test verso il fornitore di servizi saranno probabilmente oggetto di conflitto.

Nel testing distribuito, la divisione del lavoro del test su diverse sedi deve essere esplicita e decisa in modo intelligente. Senza tale guida, il team più competente può non svolgere il lavoro del test per cui è qualificato. Inoltre, se ogni team non sa di cosa sia responsabile, non può fare ciò che si supponga debba fare. Le aspettative per ognuno dei team devono essere chiaramente comunicate. Senza una gestione accurata, il lavoro del test nel suo complesso può essere impattato negativamente da lacune (che aumentano il rischio di qualità residuo al momento del rilascio) e da sovrapposizioni (che riducono l'efficienza).

Infine, per tutti questi tipi di effort del test, è fondamentale che tutto il team di progetto sviluppi e mantenga la fiducia che ogni team di test svolga in modo appropriato il proprio ruolo indipendentemente da limiti organizzativi, culturali, di lingua, e geografici. La mancanza di fiducia porta a inefficienze e ritardi nelle attività di verifica, creando inoltre accuse di colpe per i problemi, e conflitti organizzativi interni.

2.9 Gestire l'Applicazione degli Standard Industriali

Nei Syllabi Foundation Level e Advanced Level, si fa riferimento a una serie di standard, che coprono i cicli di vita dello sviluppo software, il testing del software, le caratteristiche di qualità del software, le review e il defect management. I Test Manager dovrebbero essere consapevoli degli standard, della politica della propria organizzazione sull'uso degli standard, e se questi standard sono richiesti, necessari o utili all'utilizzo.

Gli standard possono essere:

- Internazionali o con obiettivi internazionali
- Nazionali, come le applicazioni nazionali degli standard internazionali
- Dominio specifico, come quando gli standard internazionali o nazionali sono sviluppati adattati per specifici

Gli organismi standard internazionali comprendono ISO e IEEE. ISO è International Standards Organization, chiamato anche IOS, International Organization for Standardization. E' composto da membri che rappresentano, per il loro paese, l'organismo nazionale più rilevante per l'area di standardizzazione. Questo organismo internazionale ha emesso una serie di standard utili per i tester del software, come ISO 9126 (in corso di sostituzione con ISO 25000), ISO 12207 [ISO12207], e ISO 15504 [ISO15504].

IEEE è Institute of Electrical and Electronics Engineers, un'organizzazione di professionisti con sede negli Stati Uniti (U.S.), ma con rappresentanti nazionali in oltre cento nazioni. Questa organizzazione ha emesso una serie di standard che sono utili per i tester del software, come ad esempio IEEE 829 [IEEE829] e IEEE 1028 [IEEE1028].

Molte nazioni hanno i propri standard nazionali, alcuni dei quali hanno implicazioni con il testing del software. Un esempio è lo standard inglese BS-7925-2 [BS7925-2], che fornisce informazioni relative a molte delle tecniche di progettazione dei test descritte nei Syllabi Advanced Level Test Analyst e Technical Test Analyst.

Alcuni standard sono specifici per particolari domini, e hanno implicazioni con il testing del software, la qualità del software e lo sviluppo software. Ad esempio, nel campo avionico, lo standard U.S. Federal Aviation Administration DO-178B (e il suo europeo UE equivalente /DE 12B) si applica al software utilizzato in aeromobili civili. Questo standard prescrive determinati livelli di criteri di copertura strutturale sulla base del livello di criticità del software sotto test.

Un altro esempio di standard specifico di un dominio specifico è relativo ai sistemi medicali, U.S. Food and Drug Administration Title 21 CFR Part 820 [FDA21]. Lo standard raccomanda alcune tecniche di

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

test strutturali e funzionali. Lo standard raccomanda inoltre strategie e principi del testing che sono consistenti con i Syllabi ISTQB®.

In alcuni casi, il testing è influenzato da standard o metodologie che non sono principalmente indirizzate al testing, ma che influenzano pesantemente il contesto del processo software in cui il testing si svolge. Un esempio è il framework CMMI® di process improvement. Comprende due aree di processo chiave, la verifica e la validazione, che sono considerate dei riferimenti ai livelli di test (come il testing di sistema e il testing di accettazione). Ha anche implicazioni sulla strategia di test, essendo spesso interpretata in modo che il testing analitico basato sui requisiti debba essere incluso come parte della strategia di test.

Tre altri esempi importanti sono il PMI®, Prince 2® e ITIL®. PMI e Prince 2 sono comunemente utilizzati come framework di riferimento per il project management nel Nord America e in Europa, rispettivamente. ITIL è un framework per garantire che uno gruppo IT rilasci servizi di qualità alla propria organizzazione. Le attività e la terminologia di questi framework differiscono significativamente dai Sillabi e dal glossario ISTQB®. Quando lavora in un'organizzazione che utilizza PMI, Prince 2, e / o ITIL, il Test Manager deve comprendere i framework selezionati, le loro implementazioni e la loro terminologia in modo sufficientemente bene per poter operare efficacemente in tale contesto.

Qualunque siano gli standard o le metodologie adottati, è importante ricordare che sono stati creati da gruppi di professionisti. Gli standard riflettono l'esperienza e la conoscenza collettiva del gruppo di origine, ma anche le debolezze. I Test Manager dovrebbero essere a conoscenza degli standard che si applicano al loro contesto e ambiente, sia standard formali (internazionali, nazionali o specifici del dominio) o normative in-house o pratiche raccomandate.

Quando si considera l'utilizzo di diversi standard, occorre tenere presente che alcuni standard possono essere inconsistenti con altri standard, o addirittura fornire definizioni in conflitto tra loro. Il Test Manager dovrebbe determinare l'utilità dei diversi standard per il contesto specifico del testing. Le informazioni riportate in uno standard possono essere utili in un progetto, o possono ostacolarlo. Tuttavia, gli standard possono fornire un riferimento a comprovate best-practice, e fornire una base per organizzare il processo di test.

In alcuni casi, la conformità agli standard è obbligatoria e ha implicazioni per il testing. Il Test Manager deve essere a conoscenza di tali requisiti di adesione agli standard, e garantire che sia mantenuta una conformità adeguata.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

3. Review (review) – 180 minuti

Parole Chiave:

Audit, ispezione, management review, moderatore, piano di review, review (review), review informali, review tecnica, reviewer, walkthrough.

Obiettivi di Apprendimento per le Review

3.2 Management Review e Audit
TM-3.2.1 (K2) Comprendere le caratteristiche chiave della management review e dell'audit

3.3 Gestire le Review
TM-3.3.1 (K3) Analizzare un progetto per selezionare il tipo di review appropriata e per definire un piano per condurre le review, per poter garantire l'esecuzione, il follow-up e la responsabilità appropriata
TM-3.3.2 (K2) Comprendere i fattori, le competenze e il tempo richiesto per la partecipazione alle review

3.4 Metriche per le Review
TM-3.4.1 (K3) Definire le metriche di processo e di prodotto da utilizzare nelle review

3.5 Gestire le Review Formali
TM-3.5.1 (K3) Spiegare, utilizzando esempi, le caratteristiche di una review formale

3.1 Introduzione

Le review, come attività di testing statico per i prodotti, sono state introdotte nel Syllabus ISTQB Foundation Level. Gli audit e le management review si focalizzano maggiormente sul processo di test, invece che sui prodotti di lavoro del software.

Poiché le review sono una forma di testing statico, i Test Manager possono essere responsabili del loro successo global, in particolare per i prodotti del testware. In un contesto più ampio dei progetti software, questa responsabilità dovrebbe essere parte della politica organizzativa. Data la possibile applicazione delle review formali in molte discipline, sia prima che durante lo svolgimento dei progetti software, la parte responsabile può essere un Test Manager, o un Quality Assurance Manager, o un Review Coordinator formato. In questo Syllabus, la parte responsabile (chiunque essa sia) è indicata come il review leader.

Il review leader dovrebbe assicurare che esiste un ambiente favorevole all'implementazione dei fattori di successo definiti nel Syllabus ISTQB Foundation Level. Inoltre, il review leader dovrebbe elaborare un piano di misure per garantire che le review forniscano un valore effettivo.

Dato che i tester hanno una forte conoscenza del comportamento operativo e delle caratteristiche richieste del sistema software, il coinvolgimento del tester nel processo di review è importante.

I partecipanti alle review dovrebbero aver ricevuto una formazione sulla review per meglio comprendere i rispettivi ruoli in qualsiasi processo di review. Tutti i partecipanti alla review devono impegnarsi ad ottenere i vantaggi di una review ben condotta.

Se svolte in modo appropriate, le review sono il più efficace, e più conveniente, metodo per contribuire alla qualità complessiva rilasciata. È quindi di fondamentale importanza che i review leader siano in grado di implementare review efficienti nei loro progetti e dimostrarne i benefici.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Le review che possono essere svolte durante un progetto sono:

- Review contrattuali, svolte in fase di inception (inizio) del progetto e durante le milestone di progetto principali
- Review dei requisiti, svolte quando i requisiti siano disponibili per la review, che idealmente coprono sia i requisiti funzionali sia i requisiti non-funzionali
- Review di progettazione di alto livello, svolte quando la progettazione architetturale generale è disponibile per la review
- Review di progettazione dettagliata, svolte quando la progettazione dettagliata è disponibile per la review
- Code review, svolte quando sono creati i singoli moduli software, e possono includere i test di componente (unit test), i loro risultati e il codice stesso
- Review dei prodotti di lavoro del test, che possono coprire i test plan, le condizioni di test, i risultati dell'analisi dei rischi di qualità, i test, i dati di test, gli ambienti di test e i risultati dei test
- Review di inizio test (readiness dei test) e review di fine test, che verificano rispettivamente i criteri di ingresso del test prima di iniziare l'esecuzione dei test, e i criteri di uscita del test prima di concludere il testing
- Review di accettazione, usate per ottenere l'approvazione del cliente o dello stakeholder per il sistema.

Oltre ad applicare diversi tipi di review a un prodotto, è importante per un review leader ricordare che le review possono trovare difetti in un documento statico, ma dovrebbero essere potenziate con altre forme di testing statico (come l'analisi statica) e con il testing dinamico del codice. L'utilizzo di una combinazione di queste tecniche migliora la copertura del test e localizzerà più difetti.

Tecniche diverse hanno diversi focus. Ad esempio, una review può eliminare un problema a livello di requisiti, prima che il problema venga implementato nel codice. L'analisi statica può aiutare a far rispettare gli standard di codifica e a verificare problemi che potrebbe essere troppo laborioso per il team rilevare con il semplice esame del prodotto di lavoro. Le ispezioni possono non solo guidare la scoperta e rimozione dei difetti, ma anche formare gli autori su come evitare di creare difetti nei loro prodotti di lavoro.

Il Syllabus ISTQB Foundation Level ha introdotto i seguenti tipi di review:

- Review informale
- Walkthrough
- Review tecnica
- Ispezione

In aggiunta, i Test Manager possono anche essere coinvolti in:

- Management review
- Audit

3.2 Management Review e Audit

Le management review sono utilizzate per valutare lo stato, monitorare l'avanzamento e prendere decisioni sulle azioni future. Queste review supportano decisioni sul futuro del progetto, come adattare il livello delle risorse, implementare azioni correttive o modifiche all'ambito del progetto.

Di seguito sono riportate le caratteristiche principali delle management review:

- Condotte da/per manager che hanno responsabilità diretta sul progetto o sul sistema
- Condotte da/per uno stakeholder o un decision maker, ad esempio un manager di livello superiore o un direttore
- Verifica la consistenza e le deviazioni dai piani
- Verifica l'adeguatezza delle procedure di gestione
- Valuta i rischi di progetto

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Valuta l'impatto delle azioni e delle modalità per misurare questi impatti
- Produce di liste di action item, problemi da risolvere e decisioni prese

Le management review dei processi, come le retrospettive del progetto (cioè le lessons learned), sono una parte integrante delle attività di miglioramento dei processi.

I Test Manager dovrebbero parteciparvi e possono iniziare le management review relative all'avanzamento del testing.

Gli audit sono di solito eseguiti per dimostrare la conformità a un insieme definito di criteri, in genere uno standard applicabile, un vincolo normativo, o un obbligo contrattuale. Come tale, gli audit hanno lo scopo di fornire una valutazione indipendente della conformità a processi, normative, standard, ecc. Le seguenti sono caratteristiche chiave degli audit:

- Condotte e moderate da un lead auditor
- Evidenza delle conformità raccolte attraverso interviste, testimonianze e l'esame di documenti
- Risultati documentati, che includono osservazioni, raccomandazioni, azioni correttive e una valutazione superato/fallito.

3.3 Gestione delle Review

Le review dovrebbero essere pianificate in modo da svolgersi in checkpoint o milestone pianificate durante il progetto software. Tipicamente le review dovrebbero essere svolte dopo la definizione e la progettazione dei requisiti, eseguendo le review partendo dagli obiettivi di business e proseguendo fino al più basso livello di progettazione. Le management review dovrebbero svolgersi al raggiungimentodelle principali milestone di progetto, spesso come parte di un'attività di verifica prima, durante o dopo l'esecuzione dei test e altre fasi di progetto significative, La strategia della review deve essere coordinata con la politica di test e la strategia generale di test.

Prima di formulare un piano generale di review a livello di progetto, il review leader (che può essere un Test Manager) dovrebbe tener conto di:

- Cosa dovrebbe essere sottoposto a review (prodotto e processi)
- Chi dovrebbe essere coinvolto in review specifiche
- Fattori di rischio rilevanti da coprire

Nelle fasi iniziali di pianificazione del progetto, il review leader dovrebbe identificare gli elementi da sottoporre a review, e selezionare il tipo di review appropriata (informale, walkthrough, review tecnica o ispezione, o un misto di due o tre tipi) e il livello di formalità. Questo è il punto in cui potrebbe essere consigliata della formazione aggiuntiva sulla review. Per questo può essere allocato un budget (tempo e risorse) per il processo di review. La definizione del budget dovrebbe includere una valutazione del rischio e il calcolo del ritorno nell'investimento (return on investment, ROI).

Il ritorno nell'investimento per le review è la differenza tra i costi di condurre la review e il costo di rilevare gli stessi difetti in una fase successiva (o addirittura di non rilevarli) perché la review non è stata svolta. Il calcolo del "Costo della Qualità" può essere usato per determinare tale valore (si veda paragrafo 2.7).

L'identificazione del momento ottimale in cui svolgere una review dipende dai seguenti:

- Disponibilità degli elementi da sottoporre a review in un formato sufficientemente consolidato
- Disponibilità del personale adeguato per la review
- Momento in cui la versione finale dell'elemento dovrebbe essere disponibile
- Tempo richiesto per il processo di review di quell'elemento specifico

Adeguate metriche per la valutazione della review dovrebbero essere definite dal review leader durante la pianificazione dei test. Se vengono utilizzate le ispezioni, dovrebbero allora essere condotte brevi ispezioni a richiesta dell'autore, non appena parti del documento vengono completate (p.e., singoli requisiti o paragrafi).

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Gli obiettivi del processo di review devono essere definiti durante la pianificazione dei test. Questi includono come condurre review efficaci ed efficienti, e come raggiungere il consenso su decisioni relative ai feedback della review.

Review di progetto vengono frequentemente organizzate per il sistema completo, ma possono essere richieste anche per sottosistemi o singoli elementi software. Il numero di review, il tipo di review, l'organizzazione delle review e le persone coinvolte dipendono tutte dalle dimensioni e dalla complessità del progetto, e dai rischi del prodotto.

Per essere efficienti, i partecipanti alle review devono avere un livello appropriato di conoscenza, sia tecnica che procedurale. Il rigore e l'attenzione al dettaglio sono alcune delle competenze richieste ai reviewer per svolgere review efficaci. La chiarezza e la corretta prioritizzazione sono attributi da considerare per buoni commenti di review. La necessità di una conoscenza procedurale può significare che sia necessaria della formazione, per garantire che i reviewer comprendano il proprio ruolo e responsabilità nel processo di review.

La pianificazione della review dovrebbe indirizzare i rischi associati ai fattori tecnici, ai fattori organizzativi e ai fattori legati alle persone nell'esecuzione delle review. La disponibilità di reviewer con una sufficiente conoscenza tecnica è fondamentale per una review di successo. Tutti i team nel progetto dovrebbero essere coinvolti nella pianificazione delle review, il che dovrebbe assicurare che ogni team si impegni per il successo del processo di review. La pianificazione deve assicurare che ogni organizzazione allochi il tempo sufficiente ai reviewer per prepararsi e per partecipare alle review nei momenti appropriati della schedulazione di progetto. Il tempo dovrebbe essere pianificato per qualsiasi formazione tecnica o di processo necessaria ai reviewer. Dovrebbero identificati reviewer di backup nel caso in cui i reviewer originali non siano più disponibili a causa di cambiamenti nelle pianificazioni del personale o di business.

Durante l'esecuzione effettiva delle review formali, il review leader deve assicurare che:

- Siano fornite dai partecipanti le misure adeguate a consentire una valutazione di efficienza della review
- Siano create e manutenute checklist per migliorare future review
  Siano definite la valutazione della severità e della priorità dei difetti per utilizzarli nella defect management rilevati durante le review (si veda il Capitolo 4).

Dopo ogni review, il review leader dovrebbe:

- Raccogliere le metriche di review e assicurare che i problemi rilevati siano sufficientemente risolti per raggiungere gli obiettivi di test specifici per la review
- Utilizzare le metriche di review come input del calcolo nell'investimento (ROI) delle review
- Fornire informazioni sui feedback agli stakeholder rilevanti
- Fornire feedback ai partecipanti della review

Per valutare l'efficacia delle review, i Test Manager possono confrontare i risultati effettivi rilevati nel testing successivi (cioè dopo le review) con i risultati dei review report. Nel caso in cui un prodotto di lavoro sia stato sottoposto a review, approvato in base alla review, ma successivamente sia stato trovato difettoso, il review leader dovrebbe identificare come il processo di review potrebbe aver permesso che alcuni difetti siano sfuggiti. Probabilmente le cause includono problemi con il processo di review (p.e., criteri di ingresso/uscita deboli), composizione del team di review non appropriata, strumenti di review non adeguati (checklist, ecc.) insufficiente formazione ed esperienza dei reviewer, ridotta preparazione e troppo poco tempo per il review meeting.

Un pattern di difetti sfuggiti (specialmente difetti major, critici), ripetuti nei diversi progetti, indica che esistono problemi importanti nel condurre le review. In questa situazione il review leader deve eseguire la review del processo o prendere opportune azioni. È anche possibile che, per varie ragioni, le review possano perdere la loro efficacia nel tempo. Tale effetto si rileverà durante le retrospettive di progetto, per l'efficacia ridotta della rilevazione dei difetti durante le review. Anche in questo caso il review leader deve investigare e correggere le cause. Comunque, le metriche di review non dovrebbero essere

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

utilizzate per premiare o punire i singoli reviewer o autori, ma dovrebbero focalizzarsi sul processo di review stesso.

3.4 Metriche per la Review

I review leader (che, come riportato nei precedenti paragrafi, possono essere Test Manager) devono garantire che siano disponibili metriche per:

- Valutare la qualità dell'oggetto sottoposto a review
- Valutare il costo di condurre la review
- Valutare il vantaggio che si è ottenuto nell'avere condotto la review

I review leader possono usare le misure per determinare il ritorno sull'investimento e l'efficienza delle review. Queste metriche possono essere utilizzate anche per la reportistica e per le attività di miglioramento dei processi (process improvement).

Per ogni prodotto di lavoro sottoposto a review, le seguenti metriche possono essere misurate e utilizzate per la valutazione del prodotto:

- Dimensione del prodotto (pagine, linee di codice, ecc.)
- Tempo di preparazione (prima della review)
- Tempo per condurre la review
- Tempo di rework per correggere i difetti
- Durata del processo di review
- Numero di difetti rilevati e relativa severità
- Identificazione dei cluster dei difetti nel prodotto di lavoro (cioè aree con una densità di difetti più alta)
- Tipo di review (review informale, walkthrough, review tecnica o ispezione)
- Densità media dei difetti (p.e., difetti per pagina o per migliaia di linee di codice)
- Difetti residui stimati (o densità residua dei difetti)

Per ogni review le seguenti metriche possono essere misurate e documentate per la valutazione del processo:

- Efficacia di rilevazione dei difetti (defect detection effectiveness o defect detection percentage DDP), tenendo in considerazione i difetti rilevati successivamente nel ciclo di vita
- Miglioramento dell'effort e della tempistica del processo di review
- Percentuale di copertura dei prodotti del lavoro pianificati
- Tipi di difetti rilevati e relativa severità
- Sondaggi ai partecipanti sull'efficacia ed efficienza del processo di review
- Metriche del costo della qualità dei difetti della review rispetto ai difetti del test dinamico e ai difetti in produzione
- Correlazione dell'efficacia della review (tipo di review rispetto a defect detection effectiveness)
- Numero di reviewer
- Difetti rilevati rispetto alle ore lavorative spese
- Stima del tempo di progetto salvato
- Effort medio dei difetti (cioè tempo totale di rilevazione e correzione diviso per il numero totale dei difetti)

Inoltre, le metriche menzionate per la valutazione del prodotto sono anche utili per la valutazione del processo.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

3.5 Gestire le Review Formali

Il Syllabus ISTQB Foundation Level descrive le diverse fasi di una review formale: pianificazione, kick-off, preparazione individuale, review meeting, rework e follow-up. Per implementare correttamente le review formali, i review leader devono assicurarsi che tutte i passi del processo di review siano eseguiti.

Le review formali hanno una serie di caratteristiche quali:

- Criteri di entrata e di uscita definiti
- Checklist che devono essere utilizzate dai reviewer
- Rilasci come report, evaluation sheet o altri review summary sheet
- Metriche per il reporting sull'efficacia, l'efficienza e l'avanzamento della review

Prima di iniziare una review formale, il review leader dovrebbe confermare il raggiungimento dei prerequisiti della review (come definiti nella procedura o nell'elenco dei criteri di ingresso).

Se i prerequisiti per la review formale non sono soddisfatti, il review leader può proporre al review authority una delle seguenti azioni per la decisione finale:

- Ridefinizione della review con obiettivi rivisti
- Azioni correttive necessarie affinché la review proceda
- Posticipo della review

Nell'ambito del controllo di una review formale, queste review sono monitorate nel contesto del programma globale, e sono associati alle attività di quality assurance del progetto. Il controllo delle review formali include informazioni sui feedback, utilizzando metriche di prodotto e di processo.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

4. Defect Management – 150 minuti

Parole Chiave:

anomalia, comitato di defect management, contenimento di fase, difetto, failure, priorità, risultato di falso-negativo, risultato di falso-positivo, root cause, severità.

Obiettivi di Apprendimento per il Defect Management

4.2 Il Ciclo di Vita del Difetto e il Ciclo di Vita dello Sviluppo Software

TM-4.2.1 (K3) Sviluppare un processo di defect management per una organizzazione del testing, che include il workflow del defect report, che può essere utilizzato per monitorare e controllare i difetti di un progetto durante tutto il ciclo di vita del testing

TM-4.2.2 (K2) Spiegare il processo e i partecipanti necessari per un defect management efficace

4.3 Informazioni del Defect Report

TM-4.3.1 (K3) Definire i dati e le informazioni di classificazione che dovrebbero essere raccolte durante il processo di defect management

4.4 Valutare la Capacità del Processo in Base alle Informazioni del Defect Report

TM-4.4.1 (K2) Spiegare come le statistiche dei defect report possono essere utilizzate per valutare la capacità del testing e del processo di sviluppo software

4.1 Introduzione

Il processo di defect management di un'organizzazione e lo strumento utilizzato per gestire questa attività sono di fondamentale importanza, non solo per il team di test ma per tutti i team coinvolti nello sviluppo software. Le informazioni raccolte da un efficace defect management permettono al Test Manager e agli altri stakeholder di progetto di conoscere meglio lo stato di un progetto durante il ciclo di vita dello sviluppo, e la raccolta e l'analisi dei dati nel tempo può aiutare a rilevare aree di potenziale miglioramento per i processi di test e di sviluppo.

Oltre a comprendere il ciclo di vita complessivo del difetto e come venga utilizzato per monitorare e controllare sia il processo di test che il processo di sviluppo software, il Test Manager deve anche avere familiarità dei dati che sono critici da catturare e deve essere un sostenitore dell'utilizzo appropriato sia del processo sia dello strumento di defect management.

4.2 Il Ciclo di Vita del Difetto e il Ciclo di Vita dello Sviluppo Software

Come spiegato nel Syllabus Foundation Level, i difetti vengono introdotti quando una persona commette un errore durante la creazione di un prodotto del lavoro. Questo prodotto di lavoro può essere una specifica dei requisiti, una user story, un documento tecnico, un test case, il codice del programma, o qualsiasi altro prodotto di lavoro realizzato nel corso del processo di sviluppo software e di manutenzione.

I difetti possono essere introdotti in qualsiasi momento del ciclo di vita dello sviluppo software e in qualsiasi prodotto software correlato. Pertanto, ogni fase del ciclo di vita dello sviluppo software dovrebbe includere attività per rilevare e correggere potenziali difetti. Ad esempio, le tecniche di test statico (cioè, review e analisi statica) possono essere utilizzate sulle specifiche di requisiti, sulle specifiche di progettazione e sul codice prima di rilasciare questi prodotti di lavoro alle attività successive. Quanto prima un difetto viene rilevato e corretto, tanto minore è il costo della qualità

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

complessivo per il sistema; il costo di qualità per un dato livello di difetti viene minimizzato quando ogni difetto viene corretto nella stessa fase in cui è stata introdotto (cioè, quando il processo software raggiunge il contenimento di fase perfetto). Inoltre, come spiegato nel Syllabus Foundation Level, il testing statico trova direttamente i difetti, piuttosto che rilevare i failure, e quindi il costo di rimozione del difetto è inferiore poiché non sono necessarie le attività di debugging per isolare il difetto.

Durante le attività di test dinamico come il testing di componente, il testing di integrazione e il testing di sistema, la presenza di un difetto si rivela quando causa un failure, che rileva una discrepanza tra i risultati effettivi e i risultati attesi di un test (cioè un'anomalia). In alcuni casi, un risultato falso-negativo si verifica quando il tester non osserva l'anomalia. Se il tester rileva l'anomalia, si è verificata una situazione che richiede ulteriori indagini. L'indagine inizia con la compilazione di un defect report.

Nello Test-Driven Development, le attività di test come gli unit test automatizzati vengono utilizzati come specifiche di progettazione eseguibili. Non appena il codice viene sviluppato, viene immediatamente eseguito utilizzando questi test. Fino a quando lo sviluppo dell'unità non sarà completato, alcuni o tutti i test falliranno. Pertanto, il failure di tale test non costituisce un difetto e non è normalmente tracciato.

4.2.1 Workflow e Stati del Difetto

La maggior parte delle organizzazioni del testing utilizzano uno strumento per gestire i defect report attraverso il ciclo di vita del difetto. Un defect report viene in genere aggiornato attraverso un workflow e procede attraverso una sequenza di stati, che costituiscono il ciclo di vita del difetto. Nella maggior parte di questi stati, uno stato ha un solo proprietario (owner) del defect report che è responsabile di eseguire l'attività che, una volta completata, causerà lo spostamento del defect report allo stato successivo (e verrà assegnato al prossimo proprietario). Negli stati finali, come ad esempio quando il defect report viene chiuso (che generalmente significa che il difetto è stato corretto e la correzione è stata verificata attraverso un test confermativo), cancellato (che generalmente significa che il defect report non è valido), non riproducibile (che generalmente significa che l'anomalia non può più essere osservata), o differito (che generalmente significa che l'anomalia si riferisce a un difetto reale, ma che il difetto non verrà corretto durante il progetto), il report non ha più un proprietario, perché non sono necessarie ulteriori azioni.

Per i difetti rilevati dai tester durante il testing, esistono in particolare tre stati dove le azioni sono di responsabilità del team di test:

- Stato iniziale
  - In questo stato, uno o più tester raccolgono le informazioni necessarie per la persona responsabile della correzione del difetto per riprodurre l'anomalia (per le informazioni da includere nel defect report si veda il paragrafo 4.3).
  - Questo può anche essere indicato come stato "aperto" o "nuovo".
- Stato restituito
  - In questo stato, chi ha ricevuto il report lo ha rifiutato o sta chiedendo al tester di fornire ulteriori informazioni. Questo stato può indicare una carenza nel processo di raccolta iniziale delle informazioni o nel testing stesso, e i Test Manager dovrebbero monitorare un numero eccessivo di restituzioni. Il tester deve fornire le informazioni aggiuntive, o confermare che il report dovrebbe essere rifiutato.
  - Questo può anche essere indicato come stato "rifiutato" o "informazioni mancanti".
- Stato di test confermativo
  - In questo stato, il tester eseguirà un test confermativo (spesso seguendo i passi per riprodurre il failure come riportato nel defect report) per determinare se la correzione (fix) ha effettivamente risolto il problema. Se il test confermativo indica che il difetto è stato corretto, il tester dovrebbe chiudere il report. Se il test confermativo indica che il difetto non è stato corretto, il tester dovrebbe riaprire il report, riassegnandolo al precedente proprietario, che può quindi completare le attività necessarie per correggere il difetto.
  - Questo può anche essere indicato come stato "risolto" o "in verifica"

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

4.2.2 Gestire Defect Report Invalidi e Duplicati

In alcuni casi, un'anomalia può manifestarsi non come sintomo di un difetto, ma a causa di un problema con l'ambiente di test, con i dati di test, con un altro elemento del testware, o a causa di incomprensione del tester. Se il tester apre un defect report che in seguito si rileva non essere relativo a un difetto nel prodotto di lavoro sotto test, questo è un risultato falso-positivo. Tali report sono normalmente cancellati o chiusi come defect report invalido. Inoltre, in alcuni casi un difetto può presentare sintomi diversi che possono apparire al tester del tutto non correlati tra loro. Se due o più defect report vengono compilati ma successivamente si rileva che sono correlati alla stessa root cause, uno dei defect report viene generalmente mantenuto, mentre gli altri vengono chiusi come defect report duplicati.

Mentre i defect report invalidi e duplicati rappresentano un certo livello di inefficienza, una certa quantità di tali report è inevitabile e deve essere accettata come tale dal Test Manager. Quando i Test Manager tentano di eliminare tutti i defect report invalidi e duplicati, il numero dei risultati falsi-negativi in genere aumenta, poiché i tester vengono scoraggiati dal segnalare i defect report. Questo diminuisce l'efficacia della rilevazione dei difetti dell'organizzazione del testing, che nella maggioranza dei casi è un obiettivo chiave dell'organizzazione del testing.

4.2.3 Defect Management Cross-Funzionale

Anche se l'organizzazione del testing e il Test Manager sono in genere i proprietari del processo completo di defect management e dello strumento di defect management, un team cross-funzionale è generalmente responsabile di gestire i difetti segnalati per un dato progetto. Oltre al Test Manager, i partecipanti del comitato di defect management (o comitato di triage dei difetti) includono normalmente lo sviluppo, il project management, il product management e altri stakeholder che hanno interesse al software in sviluppo.

Non appena le anomalie vengono rilevate e inserite nello strumento di defect management, il comitato di defect management dovrebbe incontrarsi per determinare se ogni defect report rappresenta un difetto valido, e se dovrebbe essere corretto o differito. Questa decisione richiede che il comitato di defect management consideri i benefici, i rischi e i costi associati alla correzione (fixing) o non correzione del difetto. Se il difetto è da correggere, il team dovrebbe stabilire la priorità di correzione del difetto rispetto alle altre attività di progetto. Il Test Manager e il team di test possono essere consultati sull'importanza di un difetto e dovrebbero fornire in modo oggettivo le informazioni disponibili.

Uno strumento di defect management non dovrebbe essere usato come sostituto di una buona comunicazione e neanche i meeting del comitato di defect management dovrebbe essere usato come sostituto all'utilizzo efficace di un buon strumento di defect management. La comunicazione, il supporto di uno strumento adeguato, un ciclo di vita dei difetti ben definito, e un comitato di defect management responsabilizzato sono tutti necessari per un defect management efficace ed efficiente.

4.3 Informazioni del Defect Report

Quando viene rilevato un difetto (come parte del testing statici) o si osserva un failure (come parte del testing dinamico), le persone coinvolte dovrebbero raccogliere i dati e includerli nel defect report. Queste informazioni dovrebbero essere sufficienti per tre scopi:

- Gestire il report durante il ciclo di vita del difetto
- Valutare lo stato del progetto, soprattutto in termini di qualità del prodotto, e l'avanzamento dei test
- Valutare la capacità del processo (come discusso nel paragrafo 4.4)

I dati necessari per la gestione dei defect report e per lo stato del progetto possono variare in base a quando viene rilevato il difetto nel ciclo di vita, con in genere meno informazioni nelle fasi iniziali (p.e., review dei requisiti e testing di componente). Tuttavia, le informazioni di base raccolte dovrebbero

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

essere consistenti durante tutto il ciclo di vita e idealmente tra tutti i progetti, per consentire un confronto significativo dei dati dei difetti durante il progetto e tra tutti i progetti.

La raccolta dei dati dei difetti può aiutare nel monitoraggio e controllo dell'avanzamento dei test, e nella valutazione dei criteri di uscita. Ad esempio, le informazioni dei difetti dovrebbero supportare l'analisi della densità dei difetti, la trend analysis dei difetti rilevati e risolti, il tempo medio dalla rilevazione alla risoluzione del difetto, e l'intensità del failure (p.e., analisi MTBF).

I dati dei difetti da raccogliere possono includere i seguenti:

- Il nome della persona che ha rilevato il difetto
- Il ruolo della persona (p.e., utente finale, business analyst, sviluppatore, persona del supporto tecnico)
- Il tipo di test in esecuzione (p.e., testing di usabilità, performance testing, regression testing)
- Una sintesi del problema
- Una descrizione dettagliata del problema
- I passi per riprodurre il failure (per un difetto), con i risultati effettivi e attesi (che evidenziano l'anomalia), includendo screen shot, dump di database e log, se applicabili
- La fase del ciclo di vita di introduzione, rilevamento e correzione del difetto, includendo il livello di test se applicabile
- Il prodotto di lavoro in cui è stato introdotto il difetto
- La severità dell'impatto sul sistema e/o sugli stakeholder di prodotto (generalmente determinata dal comportamento tecnico del sistema)
- La priorità di correzione (fix) del problema. (di solito determinata dall'impatto di business del failure)
- Il sottosistema o componente in cui il difetto è presente (per l'analisi dei cluster dei difetti)
- L'attività di progetto in corso al momento della rilevazione del problema
- Il metodo di identificazione che ha rilevato il problema (p.e., review, analisi statica, testing dinamico, utilizzo in produzione)
- Il tipo di difetto (di solito corrispondente a una tassonomia dei difetti, se utilizzata)
- La caratteristica di qualità impattata dal difetto
- L'ambiente di test in cui è stato osservato il difetto (per il testing dinamico)
- Il progetto e il prodotto in cui il problema è presente
- L'attuale proprietario, cioè, la persona attualmente assegnata a lavorare sul problema, assumendo che il report non sia nello stato finale
- Lo stato attuale del report (di solito gestito dallo strumento di defect management come parte del ciclo di vita)
- Il prodotto di lavoro specifico (p.e., elementi di test e loro versione), in cui è stato osservato il problema, e i prodotti di lavoro specifici in cui il problema è stato alla fine risolto
- L'impatto sugli interessi degli stakeholder di progetto e di prodotto
- Conclusioni, raccomandazioni e approvazioni per le azioni intraprese o non intraprese per risolvere il problema
- Rischi, costi, opportunità e benefici associati alla correzione o non correzione del difetto
- Le date in cui le varie transizioni del ciclo di vita del difetto si sono verificate, i proprietari del report in ogni stato, e le azioni intraprese dai membri del team di progetto per isolare, correggere e verificare il fix del difetto
- Una descrizione di come il difetto sia stato alla fine risolto e raccomandazioni per il testing del bug fix (se il difetto è stato risolto da una modifica del software)
- Altri riferimenti, come il test che ha rivelato il difetto e il rischio, il requisito o un altro elemento della base di test relativo al difetto (per il testing dinamico).

Esistono diversi standard e documenti, come ISO 9126 [ISO9126] (in corso di sostituzione con ISO25000), IEEE 829 [IEEE829], IEEE 1044 [IEEE1044], e Orthogonal Defect Classification che aiutano il Test Manager a determinare quali informazioni raccogliere per il defect reporting.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Qualunque siano le specifiche informazioni che sono state determinate essere necessarie per i defect report, è fondamentale che i tester inseriscano informazioni complete, concise, accurate, oggettive, rilevanti etempestive. Anche quando l'intervento manuale e la comunicazione face-to-face superano i problemi coni i dati dei defect report per la risoluzione dei singoli difetti, i problemi con i dati dei defect report possono diventare ostacoli insormontabili per una valutazione appropriata dello stato del progetto, dell'avanzamento del test e della capacità del processo.

4.4 Valutare la Capacità del Processo in Base alle Informazioni del Defect Report

Come discusso nel Capitolo 2, i defect report possono essere utili per il monitoraggio e il reporting dello stato del progetto. Mentre le implicazioni delle metriche sul processo sono principalmente indirizzate nel Syllabus Expert Test Management [ISTQB ETM SYL], a livello Advanced, i Test Manager dovrebbero essere consapevoli di cosa significhino i defect report in termini di valutazione delle capacità del testing e dei processi di sviluppo software.

Oltre alle informazioni di monitoraggio dell'avanzamento del test menzionate nel capitolo 2 e nel paragrafo 4.3, le informazioni sui difetti richiedono il supporto di iniziative di miglioramento del processo. Esempi includono:

- Utilizzare le informazioni sulle fasi di introduzione, rilevazione e correzione, per singola fase, per valutare il contenimento di fase e per suggerire come migliorare l'efficacia di rilevamento dei difetti in ogni fase
- Utilizzare le informazioni sulla fase di introduzione per l'analisi di Pareto delle fasi in cui vengono introdotti il maggior numero di difetti, per permettere miglioramenti mirati che riducano il numero totale di difetti
- Utilizzare le informazioni sulla root cause dei difetti per determinare le ragioni sottostanti all'introduzione dei difetti, in modo da permettere process improvement che riducano il numero totale di difetti
- Utilizzare le informazioni sulle fasi di introduzione, rilevazione e correzione per eseguire l'analisi del costo della qualità, in modo da minimizzare il costo associato ai difetti
- Utilizzare le informazioni dei componenti dei difetti per eseguire un'analisi dei cluster dei difetti, in modo da comprendere meglio i rischi tecnici (per il testing basato sul rischio) e avviare una reingegnerizzazione dei componenti più problematici

L'uso di metriche per valutare l'efficacia e l'efficienza del processo di test è discusso nel Syllabus Expert Test Management [ISTQB ETM SYL].

In alcuni casi, i team possono scegliere di non tenere traccia dei difetti rilevato durante tutto o parte del ciclo di vita dello sviluppo software. Mentre questo viene spesso fatto in nome dell'efficienza e per ridurre il sovraccarico di processo, in realtà riduce notevolmente la visibilità sulle capacità del processo di test e dello sviluppo software. Questo rende i miglioramenti suggeriti di difficile applicazione a causa della mancanza di dati affidabili.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

5. Miglioramento del Processo di Test (Test Process Improvement) - 135 minuti

Parole Chiave

Capability Maturity Model Integration (CMMI), Critical Testing Processes (CTP), Systematic Test and Evaluation Process (STEP), Test Maturity Model integration (TMMi), TPI Next (TPI)

Obiettivi di Apprendimento per Miglioramento del Processo di Test

5.2 Processo di Miglioramento del Test (Test Process Improvement)
TM-5.2.1 (K2) Spiegare, utilizzando esempi, perché è importante migliorare il processo di test

5.3 Migliorare il Processo di Test
TM-5.3.1 (K3) Definire un piano di test process improvement utilizzando il modello IDEAL

5.4 Migliorare il Processo di Test con TMMi
TM-5.4.1 (K2) Riassumere il background, gli obiettivi e l'ambito del modello di test process improvement TMMi

5.5 Migliorare il Processo di Test con TPI Next
TM-5.5.1 (K2) Riassumere il background, gli obiettivi e l'ambito del modello di test process improvement TPI-Next

5.6 Migliorare il Processo di Test con CTP
TM-5.6.1 (K2) Riassumere il background, gli obiettivi e l'ambito del modello di test process improvement CTP

5.7 Migliorare il Processo di Test con STEP
TM-5.7.1 (K2) Riassumere il background, gli obiettivi e l'ambito del modello di test process improvement STEP

5.1 Introduzione

Una volta stabilito, il processo di test globale di un'organizzazione dovrebbe essere sottoposto ad un continuous improvement (continuo miglioramento). In questo Capitolo, vengono prima affrontate problematiche di miglioramento generiche, seguite da un'introduzione di alcuni modelli specifici, che possono essere utilizzati per il test process improvement (miglioramento del processo di test). I Test Manager dovrebbe assumere che saranno loro la forza trainante delle modifiche e dei miglioramenti del processo di test, e quindi dovrebbero avere familiarità con le tecniche più diffuse nel settore discusse in questo capitolo. Ulteriori informazioni sui processi di miglioramento del test sono discussi nel Syllabus Expert Level Improving the Test Process.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

5.2 Processo di Miglioramento del Test (Test Process Improvement)

Non appena le organizzazioni utilizzano il testing per migliorare il software, le tecniche di process improvement possono essere selezionate e utilizzate per migliorare il processo dello sviluppo software e i rilasci software risultanti. Il process improvement può essere applicato anche ai processi di test. Sono disponibili diversi metodi e modalità per migliorare il testing del software e dei sistemi contenenti software. Questi metodi mirano a migliorare il processo, e quindi i rilasci, fornendo linee guida e aree di miglioramento.

Il testing costituisce spesso una parte importante dei costi totali di progetto. Tuttavia, viene data poca attenzione al processo di test nei diversi modelli di process improvement, come ad esempio CMMI® (si veda di seguito per i dettagli).

I modelli di test process improvement, come il Test Maturity Model integration (TMMi®), Systematic Test and Evaluation Process (STEP), Critical Testing Processes (CTP) e TPI Next®, sono stati sviluppati per indirizzare la mancanza di attenzione al testing in molti modelli di process improvement., Questi modelli, usati in modo appropriato, possono fornire un grado di metriche cross-organizzative che possono essere utilizzate per un confronto di benchmarking.

I modelli presentati in questo Syllabus non sono intesi essere una raccomandazione per il loro utilizzo, ma forniscono una vista rappresentativa di come i modelli funzionino e cosa includano.

5.2.1 Introduzione al Process Improvement

I process improvement sono rilevanti per il processo di sviluppo software e per il processo di test. Imparare dai propri errori consente di migliorare il processo che l'organizzazione sta utilizzando per sviluppare e testare il software. Il ciclo di miglioramento di Deming: Plan, Do, Check, Act (PDCA), è stato usato per molti decenni, ed è ancora rilevante quando i tester devono migliorare il processo attualmente in uso.

Una premessa per il process improvement è la convinzione che la qualità di un sistema è fortemente influenzata dalla qualità del processo utilizzato per sviluppare il software. La qualità migliorata nell'industria del software riduce la necessità di risorse per manutenere il software e fornisce quindi più tempo per creare più soluzioni migliori per il futuro. I modelli di processo forniscono un momento per iniziare a migliorare, misurando le capacità dei processi dell'organizzazione rispetto al modello. I modelli forniscono anche un framework per migliorare i processi dell'organizzazione sulla base del risultato di un assessment (valutazione).

Un assessment di processo porta a determinare la capacità del processo, il che incoraggia un process improvement. Questo inoltre può incentivare un successivo assessement di processo per misurare l'effetto del miglioramento.

5.2.2 Tipi di Process Improvement

L'uso di modelli di assessment è un metodo comune, che garantisce un approccio standardizzato al miglioramento dei processi di test utilizzando pratiche consolidate e affidabili.

I modelli di process improvement sono classificati in due tipologie:

1. Il modello di riferimento dei processi, che fornisce una misura della maturità come parte dell'assessment, per valutare la capacità di un'organizzazione comparata con il modello, di valutare l'organizzazione all'interno del framework, e di fornire una roadmap per migliorare il processo.
2. Il modello di riferimento dei contenuti, che fornisce valutazioni business-driven (orientate al business) sulle opportunità di miglioramento di un'organizzazione includendo, in alcuni casi, il benchmarking rispetto ai valori medi del settore attraverso l'uso di misure oggettive. Questa valutazione può essere utilizzata per creare una roadmap per migliorare il processo.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Il test process improvement può essere ottenuto anche senza modelli utilizzando, ad esempio, approcci analitici e meeting retrospettive.

5.3 Migliorare il Processo di Test

L'industria IT può adottare i modelli di test process improvement per raggiungere un livello più alto di maturità e professionalità. I modelli standard di settore stanno contribuendo a sviluppare metriche e misure cross-organizzative che possono essere utilizzate per un confronto. A causa della necessità di process improvement nel settore del testing, si sono affermate diversi insiemi di processi raccomandati. Questi includono: STEP, TMMi, TPI Next e CTP. I modelli a stage, come TMMi e CMMi, forniscono standard per il confronto tra diverse aziende e organizzazioni. I modelli continui, come CTP, STEP e TPI Next, permettono ad una organizzazione di indirizzare i propri problemi di maggior priorità con maggiore libertà nella priorità di implementazione. Ognuno di questi modelli viene discusso in questo paragrafo.

Tutti questi modelli permettono a un'organizzazione di determinare dove è posizionata rispetto al suo attuale processo di test. Una volta eseguito l'assessment, TMMi e TPI Next suggeriscono una roadmap per migliorare il processo di test. In alternativa, STEP e CTP forniscono all'organizzazione i mezzi per determinare dove ottenere il più alto ritorno sull'investimento nel process improvement, permettendo all'organizzazione di selezionare la roadmap appropriata.

Una volta concordato che i processi di test dovrebbero essere sottoposti a review e migliorati, i passi di implementazione del process improvement potrebbero essere definiti nel modello IDEAL℠ [IDEAL 96]:

- Initiating
- Diagnosis
- Establishing
- Acting
- Learning

Initiating - Avvio del processo di miglioramento

Prima di iniziare le attività di process improvement, gli obiettivi, l'ambito e la copertura dei process improvement vengono concordati dagli stakeholder. Anche la scelta del modello di process improvement deve essere fatta in questo momento. Il modello può essere selezionato sia tra le opzioni pubbliche disponibili (come CTP, STEP, TMMi e TPI Next) o essere sviluppato internamente. Inoltre, dovrebbero essere definiti i criteri di successo e un metodo con cui saranno misurati durante l'attività di miglioramento.

Diagnosis - Diagnosi della situazione attuale

Viene effettuato l'approccio di assessment concordato e viene creato un report dell'assessment del test che contiene una valutazione dell'attuale pratica di test e una lista dei possibili process improvement.

Establishing - Creazione di un piano di test process improvement

Viene prioritizzata la lista dei possibili process improvement. La prioritizzazione potrebbe essere definita in base al ritorno sull'investimento, ai rischi, all'allineamento con la strategia dell'organizzazione, e/o a benefici misurabili quantitativi o qualitativi. Dopo aver stabilito l'ordine di priorità, viene sviluppato un piano per l'implementazione dei process improvement.

Acting - Implementazione del miglioramento

Viene implementato il piano di test process improvement per il rilascio dei miglioramenti. Potrebbe includere training o mentoring, il piloting dei processi e infine, il loro rilascio completo.

Learning - Apprendimento dal programma di miglioramento

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Dopo aver completamente rilasciato i process improvement, è essenziale verificare quali benefici (tra quelli stabiliti in precedenza, oltre ai benefici potenzialmente non attesi) sono stati ottenuti. È anche importante verificare quali tra i criteri di successo per l'attività di process improvement siano stati soddisfatti.

A seconda del modello di processo utilizzato, durante questo passo del processo inizia il monitoraggio del livello successivo di maturità e viene presa la decisione di avviare di nuovo il processo di miglioramento oppure interrompere l'attività a questo punto.

5.4 Migliorare il Processo di Test con TMMi

Il Testing Maturity Model integration (TMMi) è composto da cinque livelli di maturità ed è stato ideato per essere complementare al CMMI®. Ognuno dei livelli di maturità contiene delle aree di processo definite che devono raggiungere l'85% degli obiettivi specifici e generici prima che l'organizzazione possa avanzare al livello successivo.

I livelli di maturità TMMI sono:

- Livello 1: Iniziale
  Il livello iniziale rappresenta uno stato in cui non esiste un processo di test formalmente documentato o strutturato. I test sono in genere sviluppati ad hoc dopo la codifica, e il testing è visto come debugging. Lo scopo del testing è aver compreso che stiamo dimostrando che il software funziona.
- Livello 2: Gestito
  Il secondo livello viene raggiunto quando i processi di test sono chiaramente separati dal debugging. Può essere raggiunto attraverso la definizione degli obiettivi e della politica di test, introducendo i passi presenti nel processo di test fondamentale (p.e., la pianificazione dei test), e implementando i metodi e le tecniche di base del test.
- Livello 3: Definito
  Il terzo livello viene raggiunto quando un processo di test viene integrato nel ciclo di vita dello sviluppo software, e viene documentato attraverso standard, procedure e metodi formali. Vengono svolte review e dovrebbe esistere una funzionalità distinta di testing del software che può essere controllata e monitorata.
- Livello 4: Misurato
  Il livello quattro viene raggiunto quando il processo di test è in grado di essere effettivamente misurato e gestito a livello organizzativo a vantaggio di progetti specifici.
- Livello 5: Ottimizzato
  Il livello finale rappresenta uno stato di maturità del processo di test in cui i dati del processo di test possono essere usati per aiutare a prevenire i difetti, e il focus è quello di ottimizzare il processo stabilito.

Per ulteriori informazioni su TMMI, si veda [vanVeenendaal11] e [www.tmmi.org].

5.5 Migliorare il Processo di Test con TPI Next

Il modello TPI Next definisce 16 aree chiave, ognuna delle quali copre un aspetto specifico del processo di test, come la strategia di test, le metriche, gli strumenti di test e l'ambiente di test.

Il modello definisce quattro livelli di maturità:

- Iniziale
- Controllato
- Efficiente
- Ottimizzato

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

Sono definiti specifici checkpoint per valutare ogni area chiave in ognuno dei livelli di maturità. I risultati sono riassunti e visualizzati attraverso una matrice di maturità, che copre tutte le aree chiave. La definizione degli obiettivi di process improvement e la loro implementazione possono essere adattati in base alle esigenze e alla capacità dell'organizzazione di test.

L'approccio generico rende TPI Next indipendente da qualsiasi modello di process improvement. Copre sia gli aspetti di test engineering sia il supporto prendere decisioni manageriali [deVries09].

Per ulteriori informazioni su TPI Next, si veda [www.tpinext.com]

5.6 Migliorare il Processo di Test con CTP

La premessa di base del modello di process assessment Critical Testing Process (CTP) è che alcuni processi di test sono critici. Questi processi critici, se bene eseguiti, supporteranno i team di test con successo. Al contrario, se tali attività non vengono svolte in modo adeguato, anche i singoli tester talentuosi e i Test Manager avranno poca probabilità di avere successo. Il modello identifica dodici processi di test critici. CTP è principalmente un modello di riferimento dei contenuti.

Il modello CTP è un approccio context-sensitive che permette di adattare il modello, includendo:

- Identificazione di sfide specifiche
- Identificazione di attributi dei processi ottimali
- Selezione dell'ordine e dell'importanza di implementazione dei process improvement.

Il modello CTP è adattabile al contesto di tutti i modelli di ciclo di vita dello sviluppo software.

Oltre alle interviste dei partecipanti, il modello CTP include l'utilizzo di metriche per eseguire il benchmark delle organizzazioni rispetto alle medie di settore e alle best practice.

Per ulteriori informazioni su CTP, si veda [Black03].

5.7 Migliorare il Processo di Test con STEP

Il modello Systematic Test and Evaluation Process (STEP), come CTP e a differenza di TMMi e TPI Next, non richiede che i process improvement si realizzino in un ordine specifico.

STEP è soprattutto un modello di riferimento dei contenuti, che si basa sull'idea che il testing è un'attività del ciclo di vita che inizia durante la formulazione dei requisiti e continua fino al ritiro del sistema. La metodologia STEP enfatizza "test then code" utilizzando una strategia di test basata sui requisiti, per garantire che la creazione precoce dei test case validi la specifica dei requisiti prima della progettazione e della codifica.

Le premesse di base della metodologia includono:

- Una strategia di test basata sui requisiti
- Il testing comincia all'inizio del ciclo di vita
- I test sono utilizzati come requisiti e come modelli di utilizzo
- La progettazione del testware guida la progettazione software
- I difetti vengono rilevati in anticipo oppure evitati del tutto (attività preventiva)
- I difetti sono sistematicamente analizzati
- I tester e gli sviluppatori lavorano insieme

In alcuni casi il modello di assessment STEP viene unito al modello di maturità TPI Next.

Per ulteriori informazioni su STEP, si veda [Craig02].

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

6. Strumenti di Test e Test Automation – 135 minuti

Parole Chiave:

Strumento open source, strumento custom

Obiettivi di Apprendimento per Strumenti di Test e Test Automation

6.2 Selezione dello Strumento
TM-6.2.1 (K2) Descrivere i problemi di gestione quando si seleziona uno strumento open source
TM-6.2.2 (K2) Descrivere i problemi di gestione quando si prendono decisioni su uno strumento custom
TM-6.2.3 (K4) Valutare una determinata situazione per definire un piano per la selezione dello strumento, che includa i rischi, i costi e i benefici

6.3 Ciclo di Vita dello Strumento
TM-6.3.1 (K2) Spiegare le diverse fasi del ciclo di vita di uno strumento

6.4 Metriche dello Strumento
TM-6.4.1 (K2) Descrivere come la raccolta e la valutazione di metriche possano essere migliorate utilizzando gli strumenti

6.1 Introduzione

Questo capitolo espande il Syllabus Foundation Level coprendo una serie di concetti generali sugli strumenti e l'automazione, che il Test Manager deve considerare.

6.2 Selezione dello Strumento

Esistono una varietà di problematiche differenti che un Test Manager deve considerare nella scelta degli strumenti di test.

La scelta più comune è stata storicamente di acquistare uno strumento commerciale realizzato da un fornitore. In alcuni casi può essere l'unica scelta praticabile. Tuttavia, esistono anche altre possibilità, come ad esempio strumenti open source e strumenti custom, che sono opzioni fattibili.

Indipendentemente dal tipo di strumento, un Test Manager deve essere attento a investigare il costo totale di ownership per tutta la durata prevista dello strumento, eseguendo una analisi costi-benefici. Questo argomento è trattato nel paragrafo sul ritorno sull'investimento (ROI, Return on Investment).

6.2.1 Strumenti Open Source

Sono disponibili strumenti open source per quasi ogni aspetto del processo di test, dalla gestione dei test case al tracciamento dei difetti, al test automation, solo per citarne alcuni. Una differenza importante degli strumenti open source è che, mentre lo strumento stesso non ha generalmente un costo elevato di acquisto iniziale, può non essere disponibile un supporto formale per lo strumento. Molti strumenti open source, tuttavia, hanno persone disposte a fornire un supporto non tradizionale o informale agli utenti.

Inoltre, molti strumenti open source sono stati originariamente creati per risolvere un problema specifico o indirizzare una singola tematica; per questo lo strumento può non eseguire tutte le funzionalità di uno

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

strumento similare realizzato da un fornitore. Per questo motivo, dovrebbe essere eseguita un'attenta analisi delle reali esigenze del gruppo di test prima di selezionare uno strumento open source.

Un beneficio nell'utilizzo di strumenti open source è che in genere possono essere modificati o estesi dai loro utenti. Se l'organizzazione del testing ha le competenze di base, lo strumento può essere modificato per lavorare con altri strumenti oppure essere modificato per adattarsi alle esigenze del team di test. Più strumenti possono essere combinati per risolvere i problemi che gli strumenti realizzati da fornitori non possono indirizzare. Ovviamente, se vengono utilizzati più strumenti vengono fatte più modifiche, e dovranno essere considerate più spese aggiuntive e maggiore complessità. Un Test Manager deve garantire che il team non inizi a utilizzare strumenti open source solo per il piacere di utilizzarli; come con altri strumenti, l'effort deve avere sempre il target di derivare un ROI positivo.

Un Test Manager deve comprendere lo schema delle licenze dello strumento selezionato. Molti strumenti open source sono una variante della General Public License GNU, che specifica che la distribuzione del software debba rispettare gli stessi termini dell'originale. Se il team di test dovesse apportare modifiche allo strumento per supportare meglio il testing, queste modifiche dovrebbero essere rese disponibili a tutti gli utenti esterni dello strumento sotto licenza. Un Test Manager dovrebbe verificare le implicazioni legali di ridistribuire il software per la propria organizzazione.

Le organizzazioni che sviluppano software safety-critical o mission-critical, o che sono soggette alla conformità rispetto a normative, possono avere problemi con l'utilizzo di strumenti open source. Mentre molti strumenti open source sono di qualità molto alta, l'accuratezza di uno specifico strumento open source non è stata probabilmente certificata. Gli strumenti realizzati da fornitori sono spesso certificati per la loro accuratezza e idoneità ad un particolare compito (p.e., DO-178B). Anche se la qualità di uno strumento open source può essere già buona, la certificazione può essere di responsabilità del gruppo che lo sta utilizzando, e può creare spese aggiuntive extra.

6.2.2 Strumenti Custom

L'organizzazione del testing può constatare di avere una specifica esigenza per la quale nessun strumento open source o nessun strumento realizzato da fornitori è disponibile. Le ragioni possono essere una piattaforma hardware proprietaria, un ambiente custom o un processo che è stato modificato in modo non usuale. In questi casi, se nel team esiste la competenza di base, il Test Manager può voler considerare lo sviluppo di uno strumento custom.

I benefici di sviluppare uno strumento custom sono che lo strumento può soddisfare le esigenze del team in modo accurato e può operare in modo efficiente nel contesto richiesto dal team. Lo strumento può essere realizzato per interfacciarsi con altri strumenti in uso e per generare i dati nel formato esatto richiesto dal team. Inoltre, lo strumento può essere utilizzato nell'organizzazione al di là del progetto considerato. Tuttavia, prima di pianificare il rilascio dello strumento per altri progetti, è importante rivedere prima lo scopo, gli obiettivi, i benefici e i svantaggi possibili.

Un Test Manager che prende in considerazione di creare strumenti custom deve anche considerare le possibili problematiche negative. Gli strumenti custom dipendono spesso dalla persona che crea lo strumento. Quindi, gli strumenti custom devono essere adeguatamente documentati in modo che possano essere manutenuti da altri. Se questo non viene fatto, gli strumenti possono diventare orfani e cadere in disuso quando il creatore dello strumento abbandona il progetto. Nel corso del tempo, gli strumenti custom possono estendere il loro ambito oltre l'intento iniziale, e questo può causare problemi di qualità nello strumento, che portano a defect report di risultati falsi-positivi o alla creazione di dati inaccurati. Il Test Manager deve tenere a mente che uno strumento custom è soltanto un altro prodotto software, e come tale, è soggetto allo stesso tipo di problematiche di sviluppo di qualsiasi altro prodotto software. Gli strumenti custom dovrebbero essere progettati e testati per assicurarsi che funzionino come previsto.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

6.2.3 Ritorno sull'Investimento (ROI, Return on Investment)

Il Test Manager deve garantire che tutti gli strumenti introdotti nell'organizzazione del testing aggiungano valore al lavoro del team, potendo mostrare un ROI positiva per l'organizzazione. Per garantire che uno strumento raggiungerà benefici reali e duraturi, dovrebbe essere eseguita un'analisi costi-benefici prima di acquisire o sviluppare uno strumento. In quest'analisi, il ROI dovrebbe prendere in considerazione i rischi che possono ridurre il valore dello strumento e i costi sia ricorrenti sia non-ricorrenti, che possono essere costi economici, costi di risorse o costi di tempo.

I costi non-ricorrenti includono:

- Definizione dei requisiti dello strumento per raggiungere le finalità e gli obiettivi
- Valutazione e selezione dello strumento e del fornitore dello strumento corretti
- Acquisto, personalizzazione o sviluppo dello strumento
- Esecuzione di formazione (training) iniziale sullo strumento
- Integrazione dello strumento con altri strumenti
- Acquisizione di hardware/software necessari per supportare lo strumento

I costi ricorrenti includono:

- Proprietà dello strumento
  - Licenze ed eventuali tasse
  - Manutenzione per lo strumento stesso
  - Manutenzione degli artefatti creati dallo strumento
  - Formazione continua e mentoring
- Portabilità dello strumento in ambienti differenti
- Adattamento dello strumento ad esigenze future
- Miglioramento della qualità e dei processi per garantire un utilizzo ottimale degli strumenti selezionati

Il Test Manager dovrebbe anche considerare i costi di opportunità inerenti a qualsiasi strumento. Il tempo speso per l'acquisizione, l'amministrazione, la formazione e l'utilizzo dello strumento avrebbe potuto essere impiegato per le attività di test effettive; quindi, maggiori risorse di test possono essere necessarie sino a quando l'utilizzo dello strumento non vada a regime.

Esistono molti rischi per l'utilizzo dello strumento; non tutti gli strumenti forniscono in realtà dei benefici che superino i rischi. I rischi relativi a uno strumento sono stati discussi nel Syllabus Foundation Level. Il Test Manager dovrebbe anche considerare i seguenti rischi nel calcolo del ROI:

- Immaturità dell'organizzazione (non è pronta per l'uso dello strumento)
- Artefatti creati dallo strumento difficili da manutenere, che richiedono diverse review quando il software sotto test viene modificato
- Minor coinvolgimento dei Test Analyst nelle attività di test, che può ridurre il valore del testing (p.e., il defect detection effectiveness può essere ridotto quando vengono eseguiti solo script automatizzati).

Infine, un Test Manager deve vedere i benefici che possono derivare dall'utilizzo dello strumento. I benefici per l'introduzione e l'utilizzo dello strumento possono includere:

- Riduzione del lavoro ripetitivo
- Riduzione della durata del ciclo di test (p.e., utilizzando regression test automatizzati)
- Riduzione dei costi di esecuzione dei test
- Incremento di alcuni tipi di test (p.e., regression testing)
- Riduzione dell'errore umano nelle diverse fasi del testing. Esempi includono:
  - I dati dei test possono essere più validi utilizzando strumenti di generazione dei dati
  - Il confronto dei risultati dei test può essere più accurato con strumenti di comparazione
  - L'inserimento dei dati di test può essere più corretto inserendoli con uno strumento di scripting
- Riduzione dell'effort richiesto per accedere alle informazioni sui test. Esempi includono:

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

  - Report e metriche generati dallo strumento
  - Riutilizzo di prodotti del test, come test case, test script e dati di test
- Incremento del testing, che non era possibile senza strumenti (p.e., performance test, load test)
- Miglioramento dello stato dei tester, che creano i test automatizzati, e dell'organizzazione nel suo insieme, dimostrando di comprendere e utilizzare strumenti sofisticati.

In generale, raramente un team di test utilizza un unico strumento; il ROI totale che sarà ottenuto dal team di test è di solito una funzione basata su tutti gli strumenti che vengono utilizzati. Gli strumenti richiedono di condividere informazioni e di collaborare insieme. È consigliabile una strategia globale e a lungo termine degli strumenti di test.

6.2.4 Processo di Selezione

Gli strumenti di test sono un investimento a lungo termine, che si estende probabilmente su molte iterazioni di un singolo progetto e/o su molti progetti. Un Test Manager deve considerare uno strumento da acquisire da diversi punti di vista:

- Per il business, è necessario un ROI positivo. Al fine di ottenere un valore alto nell'investimento, l'organizzazione dovrebbe garantire che questi strumenti che devono interoperare, che possono includere sia strumenti di test sia strumenti non di test, possano effettivamente lavorare insieme. In alcuni casi per realizzare questa interoperabilità devono essere migliorati i processi e la connettività di utilizzo dello strumento, e per raggiungere questo può essere richiesto un certo periodo di tempo.
- Per il progetto, lo strumento deve essere efficace (p.e., evitare errori durante l'esecuzione del testing manuale, come errori di battitura durante l'inserimento di dati). Lo strumento può richiedere una quantità di tempo apprezzabile per iniziare a ottenere un ROI positivo. In molti casi, il ROI può verificarsi nel secondo rilascio o durante la manutenzione, piuttosto che nel progetto iniziale dove era stata implementata l'automazione. Il Test Manager dovrebbe prendere in considerazione il ciclo di vita totale dell'applicazione.
- Per la persona che utilizza lo strumento, lo strumento deve supportare i membri di progetto nello svolgere i propri compiti in modo più efficiente ed efficace. Deve essere considerata la curva di apprendimento per assicurarsi che gli utenti saranno in grado di imparare lo strumento velocemente e con minimo stress. Gli strumenti di test, quando vengono introdotti la prima volta, richiedono formazione e mentoring per gli utenti.

Per poter garantire che tutti i punti di vista siano presi in considerazione, è importante creare una roadmap per l'introduzione dello strumento di test.

Il processo di selezione di uno strumento di test, come già discusso nel Syllabus Foundation Level, segue i seguenti passi:

- Valutare la maturità dell'organizzazione
- Identificare i requisiti per lo strumento
- Valutare gli strumenti
- Valutare il supporto del fornitore o del servizio (strumenti open source, strumenti custom)
- Identificare i requisiti interni per il coaching e mentoring nell'uso dello strumento
- Valutare le esigenze di formazione, considerando le competenze attuali di test automation del team di test
- Stimare i costi-benefici (come discusso nel paragrafo 6.2.3)

Per ogni tipo di strumento, indipendentemente dalla fase di test in cui deve essere utilizzato, un Test Manager dovrebbe considerare le capacità elencate di seguito:

- Analisi
  - Lo strumento sarà in grado di "comprendere" l'input che è stato fornito?
  - Lo strumento è adatto allo scopo?
- Progettazione:

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

- Lo strumento aiuterà a progettare il testware sulla base delle informazioni esistenti (p.e., strumenti di progettazione dei test che creano i test case dai requisiti)?
- La progettazione può essere generata automaticamente?
- Il codice effettivo del testware può essere totalmente o parzialmente generato in un formato utilizzabile e gestibile?
- I dati di test necessari possono essere generati automaticamente (p.e., i dati generati dall'analisi del codice)?

- Selezione dei test e dei dati
  - Come fa lo strumento a seleziona i dati di cui ha bisogno (p.e., quali test case eseguire con quale insieme di dati)?
  - Lo strumento può accettare criteri di selezione inseriti manualmente o automaticamente?
  - Lo strumento può determinare come eseguire l'anonimizzazione ("scrub") dei dati di produzione in base agli input selezionati?
  - Lo strumento può determinare quali test sono necessari in base ai criteri di copertura (p.e., dato un insieme di requisiti lo strumento può attraversare la tracciabilità per determinare quali test case siano necessari per l'esecuzione)?
- Esecuzione
  - Lo strumento verrà eseguito automaticamente o sarà necessario un intervento manuale?
  - Come verrà fermato e riavviato lo strumento?
  - Lo strumento dovrebbe essere in grado di "ascoltare" gli eventi pertinenti (p.e., uno strumento di test management dovrebbe aggiornare automaticamente lo stato del test case se un defect report relativo al test case viene chiuso)?
- Valutazione
  - Come farà lo strumento a determinare se ha ricevuto un risultato appropriato (p.e., lo strumento utilizzerà un oracolo del test per determinare la correttezza di una risposta)?
  - Quale tipo di capacità di recupero dagli errori possiede lo strumento?
  - Lo strumento fornisce adeguato logging e reporting?

6.3 Ciclo di Vita dello Strumento

Esistono quattro diversi stadi nel ciclo di vita utile di uno strumento, che un Test Manager deve gestire:

- Acquisizione. Lo strumento deve essere acquisito come discusso sopra (si veda paragrafo 6.2). Dopo che la decisione di adottare lo strumento è stata presa, il Test Manager dovrebbe assegnare a qualcuno (spesso un Test Analyst o un Technical Test Analyst) il ruolo di amministratore dello strumento. Questa persona dovrebbe prendere decisioni su come e quando lo strumento sarà utilizzato, dove verranno memorizzati gli artefatti creati, la naming convention, ecc. Prendere queste decisioni in anticipo, piuttosto che lasciare che accadano in un modo ad hoc può fare una differenza significativa nell'eventuale ROI dello strumento. Sarà probabilmente necessario fornire formazione agli utenti dello strumento.
- Supporto e manutenzione. Saranno richiesti processi per il supporto e la manutenzione dello strumento. La responsabilità di manutenere lo strumento può essere assegnata all'amministratore dello strumento, o a un team dedicato agli strumenti. Se lo strumento deve interfacciarsi con altri strumenti, dovrebbero poi essere presi in considerazione lo scambio dei dati e i processi per la cooperazione. Devono essere considerate le decisioni sul backup e restore dello strumento e dei suoi output.
- Evoluzione. Deve essere considerata la conversione. Con il passare del tempo, l'ambiente, le esigenze di business o i problemi col fornitore possono richiedere importanti modifiche allo strumento o al suo utilizzo. Ad esempio, il fornitore dello strumento può richiedere un aggiornamento dello strumento che causa problemi con gli strumenti con cui si interfaccia. Una modifica necessaria all'ambiente per motivi di business può causare problemi con lo strumento. Più complesso è l'ambiente operativo dello strumento, maggiore è la probabilità che una sua modifica evolutiva possa interferire con il suo utilizzo. A questo punto, in base al ruolo che lo

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

strumento gioca nel testing, può essere necessario che un Test Manager assicuri che l'organizzazione garantisca la continuità del servizio.

- Ritiro. Arriverà il momento che lo strumento raggiungerà il termine del suo ciclo di vita utile. A questo punto lo strumento dovrà essere ritirato in modo adeguato. La funzionalità fornita dallo strumento dovrà essere sostituita e i dati dovranno essere preservati e archiviati. Questo può accadere perché lo strumento è al termine del suo ciclo di vita, o semplicemente perché ha raggiunto un punto in cui i benefici e le opportunità di conversione ad un nuovo strumento superano i relativi costi e rischi.

Durante il ciclo di vita dello strumento, è responsabilità del Test Manager garantire un regolare funzionamento e una adeguata continuità del servizio che lo strumento fornisce al team di test.

6.4 Metriche dello Strumento

Un Test Manager può progettare e raccogliere delle metriche oggettive dagli strumenti utilizzati dai Test Analyst e dai Technical Test Analyst. Strumenti differenti possono catturare dati preziosi in tempo reale e possono ridurre l'effort della raccolta dei dati. Questi dati possono essere usati dal Test Manager per gestire l'effort globale del test.

Strumenti differenti sono focalizzati a collezionare differenti tipi di dati. Esempi includono:

- Strumenti di test management possono fornire una varietà di metriche diverse. La tracciabilità dai requisiti ai test case e agli script automatizzati permette di ottenere metriche di copertura. Una fotografia istantanea dei test attualmente disponibili, dei test pianificati e dell'attuale stato di esecuzione (superato, fallito, saltato, bloccato, in coda) può essere fornita in qualsiasi momento.
- Strumenti di defect management possono fornire una ricchezza di informazioni sui difetti, includendo lo stato attuale, la severità e priorità, la distribuzione nel sistema, ecc. Altri dati illuminanti come la fase in cui i difetti sono stati introdotti e rilevati, o la percentuale di difetti sfuggiti, possono aiutare il Test Manager a guidare il process improvement.
- Strumenti di analisi statica possono rilevare e fornire report sui problemi di manutenibilità.
- Strumenti per performance testing possono fornire informazioni preziose sulla scalabilità del sistema.
- Strumenti di copertura possono aiutare il Test Manager a comprendere quanto il sistema sia stato attualmente esercitato dal testing.

I requisiti di reporting degli strumenti dovrebbero essere definiti durante il processo di selezione dello strumento. Questi requisiti devono essere implementati in modo appropriato durante la configurazione dello strumento per garantire che il reporting delle informazioni tracciate dallo strumento sia comprensibile agli stakeholder.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

7. Competenze delle Persone – Composizione del Team - 210 minuti

Parole Chiave:

Indipendenza del Testing

Obiettivi di Apprendimento per Competenze delle Persone – Composizione del Team

7.2 Competenze Individuali

TM-7.2.1 (K4) Utilizzare un foglio di valutazione delle competenze, analizzare i punti di forza e di debolezza dei membri del team relativamente all'utilizzo di sistemi software, alla conoscenza del dominio di business, alle aree di sviluppo dei sistemi, al test del software e alle competenze interpersonali

TM-7.2.2 (K3) Analizzare una specifica valutazione delle competenze di un team per definire un piano di formazione e di sviluppo delle competenze

7.3 Dinamiche del Team di Test

TM-7.3.1 (K2) Per una specifica situazione, discutere i soft skill e hard skill richiesti per guidare un team di test

7.4 Adattamento del Testing all'Interno di un'Organizzazione

TM-7.4.1 (K2) Spiegare le opzioni per il testing indipendente

7.5 Motivazione

TM-7.5.1 (K2) Fornire esempi di fattori motivanti e demotivanti per i tester

7.6 Comunicazione

TM-7.6.1 (K3) Spiegare i fattori che influenzano l'efficacia della comunicazione all'interno di un team di test, e tra un team di test e i suoi stakeholder

7.1 Introduzione

I Test Manager di successo selezionano, assumono e mantengono i team con il giusto mix di competenze. Le competenze necessarie possono cambiare nel tempo, quindi oltre ad assumere in primo luogo le persone giuste, è importante fornire una formazione adeguata e delle opportunità di crescita per trattenere il team di test e mantenerlo a un elevato di prestazioni elevato. Oltre alle competenze del team, il Test Manager, deve anche mantenere un insieme di competenze che permetteranno di funzionare in modo efficace in un ambiente molto dinamico e soggetto a forti pressioni.

Questo capitolo descrive come valutare le competenze, come colmare le lacune per creare un team sinergico che sia coeso internamente ed efficace in una organizzazione, e anche come motivare quel team e come comunicare in modo efficace.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

7.2 Competenze Individuali

L'abilità di una persona di testare il software può essere ottenuta attraverso l'esperienza o l'educazione e la formazione. Ognuno dei seguenti elementi può contribuire alla base di conoscenza del tester:

- Utilizzo di sistemi software
- Conoscenza del business o del dominio
- Partecipazione alle varie fasi delle attività del processo di sviluppo software, inclusa l'analisi, lo sviluppo e il supporto tecnico
- Partecipazione alle attività del testing software

Gli utenti finali dei sistemi software hanno una buona conoscenza di come funziona il sistema, di dove i failure avrebbero l'impatto maggiore, e di come il sistema dovrebbe reagire in varie situazioni. Gli utenti con competenze di dominio conoscono quali sono le aree di maggior importanza per il business e come queste aree influiscano sulla capacità dell'azienda di soddisfare i propri requisiti. Questa conoscenza può essere utilizzata per aiutare a prioritizzare le attività di test, creare i test case e i dati di test realistici, e creare o verificare gli use case.

La conoscenza del processo di sviluppo software (analisi dei requisiti, architettura, progettazione e codifica) fornisce una chiara visione sul come gli errori portano all'introduzione dei difetti, dove possono essere rilevati i difetti e in primo luogo come prevenire l'introduzione dei difetti. L'esperienza nel supporto tecnico fornisce la conoscenza della user experience, delle aspettative e dei requisiti di usabilità. L'esperienza nello sviluppo software è importante per l'uso degli strumenti di test che richiedono competenze di programmazione e progettazione, e per partecipare all'analisi statica del codice, alle code review, al testing di componente, e al testing di integrazione con focalizzazione tecnica.

Specifiche competenze nel testing del software includono le capacità discusse nei Sillabi Foundation Level, Advanced Level Test Analyst e Advanced Level Technical Test Analyst, come l'abilità di analizzare una specifica, partecipare all'analisi del rischio e progettare test case, e la maestria nell'eseguire i test e registrare i risultati.

In particolare, per i Test Manager, avere la conoscenza, competenza ed esperienza nel project management è importante perché il test management include molte attività di project management, p.e., preparare un piano, tracciare l'avanzamento e fare reporting agli stakeholder. In assenza di un project manager, il Test Manager può ricoprire il ruolo di entrambi, Test Manager e project manager, in particolare durante le ultime fasi di un progetto. Queste competenze sono in aggiunta alle capacità discusse nel Syllabus Foundation Level e in questo Syllabus.

In aggiunta alle competenze tecniche, le competenze interpersonali, come fornire e ricevere critiche costruttive, influenzare e negoziare, sono tutte importanti nel ruolo del tester. Tester tecnicamente competenti hanno buona probabilità di fallire se non possiedono anche i necessari soft skill. Oltre a lavorare in modo efficace con gli altri, il professionista di test di successo deve essere anche ben organizzato, attento ai dettagli e possedere forti capacità comunicative scritte e verbali.

Il team di test ideale possiede un mix di competenze e livelli di esperienza, e i membri del team dovrebbero avere la volontà e la capacità di insegnare e imparare dai loro colleghi. In alcuni ambienti, alcune competenze saranno più importanti o più considerati di altre. Ad esempio, in un ambiente di test tecnico dove sono richieste competenze nel testing delle API e nella programmazione, le competenze tecniche possono essere considerate più importanti della conoscenza del dominio. In un ambiente di test black-box, la competenza di dominio può assumere maggior valore. È importante ricordare che gli ambienti e i progetti sono soggetti a cambiamenti.

Quando crea un foglio (spreadsheet o matrice) di valutazione delle competenze, il Test Manager dovrebbe elencare tutte le competenze che sono importanti per l'attività lavorativa e che sono appropriate per la posizione. Una volta che tali competenze sono state definite, ogni singolo individuo nel team può essere valutato rispetto a un sistema di punteggio (p.e., una valutazione da 1 a 5, dove 5 è il livello di competenza più alto previsto per quell'area). I singoli individui possono essere valutati per determinare le aree forti e deboli e, sulla base di queste informazioni, possono essere creati dei piani di

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

formazione individuali o di gruppo. Il Test Manager può fissare per i singoli individui un obiettivo di prestazione che migliori le loro competenze in particolari aree, e dovrebbe definire criteri specifici da utilizzare per valutare le competenze individuali.

Le persone dovrebbero essere assunte a lungo termine, non solo per il loro contributo in un singolo progetto. Quando il Test Manager investe nei tester e crea un ambiente di apprendimento continuo, i membri del team saranno motivati ad accrescere le proprie competenze e conoscenze in modo da essere pronti per la prossima opportunità.

Un Test Manager raramente sarà in grado di assumere dei membri del team perfetti. E, anche se i membri del team fossero perfetti per il progetto attuale, potrebbero non essere il mix perfetto per il prossimo progetto. Il Test Manager dovrebbe assumere persone che siano intelligenti, curiose, adattabili, desiderose di lavorare, in grado di lavorare in modo efficace come parte di un team, e con la volontà e capacità di imparare. Anche se un insieme di individui perfetti probabilmente non è disponibile, una team forte può essere costruito bilanciando i punti di forza e di debolezza dei singoli individui.

Utilizzando un foglio di valutazione delle competenze, il Test Manager può identificare dove il team è forte e dove è debole. Queste informazioni costituiscono la base di un piano di sviluppo di formazione e di sviluppo delle competenze. Il Test Manager, partendo dalle debolezze che hanno il maggiore impatto sull'efficacia e sull'efficienza del team, dovrebbe decidere come indirizzare queste aree. Un approccio è la formazione, p.e., mandare le persone a un corso di training, organizzare sessioni di training in-house, sviluppare training custom, utilizzare corsi e-learning. Un altro approccio è il self-study, p.e., libri, webinar, risorse Internet. Un ulteriore approccio è il cross-training, p.e., identificare una persona che ha interesse ad apprendere una competenza in modo che possa lavorare con qualcuno che già possiede quella competenza perché lavora su un'attività che la richiede, oppure avere esperti locali che fanno brevi presentazioni sulla loro area di competenza (il mentoring è un approccio similare, dove una persona con un nuovo ruolo viene affiancata con una persona senior che ha già svolto quel ruolo, e la persona senior fornisce assistenza e consigli.). Oltre ad indirizzare le debolezze, il Test Manager dovrebbe ricordarsi di sfruttare i punti di forza identificati nella valutazione delle competenze, come parte del piano di formazione e sviluppo delle competenze. Per ulteriori informazioni sui piani di sviluppo del team di test, si veda [McKay07].

7.3 Dinamiche del Team di Test

Per costruire il migliore team possibile, la selezione del personale è una delle funzioni più importanti del management di un'organizzazione. Esistono molti elementi da considerare, in aggiunta alle competenze specifiche individuali richieste per l'attività lavorativa. Quando si seleziona un singolo individuo per inserirlo nel team, devono essere considerate le dinamiche del team. Questa persona sarà complementare ai tipi di competenze e personalità che già esistono all'interno del team di test? È importante considerare i vantaggi di avere una varietà di personalità nel team di test, nonché un mix di competenze tecniche. Un team di test forte è in grado di gestire più progetti di diversa complessità, gestendo con successo anche le interazioni interpersonali con altri membri del team di progetto.

Il testing è spesso un'attività con forti pressioni. Le schedulazioni dello sviluppo software sono spesso compresse, anche irrealistici. Gli stakeholder hanno alte aspettative dal team di test, a volte in modo irragionevole. Il Test Manager deve assumere persone che gestiranno bene situazioni con fortipressioni, che possono evitare la frustrazione e che possano concentrarsi sul lavoro anche quando le scadenze sembrano impossibili. È compito del Test Manager affrontare le problematiche relative alla schedulazione e alle aspettative, ma il Test Manager deve anche comprendere che queste pressioni sono sentire dai membri del team. Quando il Test Manager sta acquisendo persone per il team, è importante considerare l'ambiente di lavoro e come i tipi di personalità corrispondano a tale ambiente. In un ambiente in cui esiste più lavoro del tempo necessario per svolgerlo, il Test Manager dovrebbe essere alla ricerca di persone che, terminati i propri compiti, chiedano cosa possano fare dopo.

I singoli individui lavoreranno di più e avranno più cura in quello che fanno, se si sentono valorizzati e necessari. Devono capire che ognuno è un membro importante del team e il loro contributo è vitale per

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

il successo di tutto il team. Quando viene creata questa dinamica, si verificherà un cross-training informale, verrà fatto un bilanciamento del carico di lavoro dagli stessi membri del team, e il Test Manager avrà più tempo per affrontare i problemi esterni.

I nuovi membri devono essere rapidamente integrati nel team e assistiti con adeguata supervisione e supporto. Ogni persona dovrebbe avere un ruolo definito nel team. Questo può essere basato su un processo di valutazione individuale. L'obiettivo è di rendere ogni persona un individuo di successo, contribuendo però al successo globale del team. Questo viene ottenuto, in gran parte, facendo corrispondere i tipi di personalità con i ruoli del team, investendo sulle competenze innate del singolo individuo e aumentando il loro portfolio delle competenze.

Quando si decide chi assumere o aggiungere al team di test, può essere utile una valutazione oggettiva delle competenze. Questa può essere svolta attraverso interviste, verifica del candidato, review a campione del lavoro svolto e verifica delle referenze. Dovrebbero essere valutate le competenze tecniche e interpersonali.

Le competenze tecniche (hard skill) possono essere dimostrate da:

- Derivare test case da un documento dei requisiti
- Eseguire la review della documentazione tecnica (requisiti, codice, ecc.)
- Scrivere i commenti della review in modo chiaro, comprensibile e oggettivo
- Applicare diverse tecniche di test in modo appropriato a specifici scenari (p.e., utilizzando le tabelle di decisione per il testing di un insieme di regole di business)
- Valutare un failure e documentarlo accuratamente
- Dimostrare di aver ben compreso le informazioni di classificazione dei difetti
- Dimostrare di aver ben compreso le root cause dei difetti
- Utilizzare uno strumento per testare una determinata API, includendo test positivi e negativi
- Utilizzare SQL per trovare e modificare le informazioni del database per testare un determinato scenario
- Progettare un harness di test automation, che eseguirà test multipli e raccoglierà i risultati dei test
- Eseguire test automatizzati e fare troubleshhoting dei failure
- Scrivere test plan e specifiche di test
- Scrivere test summary report che includano valutazioni sui risultati dei test

Le capacità interpersonali (soft skill) possono essere dimostrate da:

- Presentare informazioni relative a un progetto di test che è in ritardo
- Spiegare un defect report ad uno sviluppatore che pensa non ci sia nessun difetto
- Formare un collaboratore
- Presentare al management un problema relativo ad un processo che non è efficace
- Eseguire la review di un test case creato da un collaboratore e presentare le osservazioni a quella persona
- Intervistare un collaboratore
- Complimentarsi con uno sviluppatore

Benché questo non sia un elenco completo e le specifiche competenze richieste variano tra i diversi ambienti e organizzazioni, è un elenco di competenze che sono comunemente necessarie. Creando questionari/interviste efficaci e eseguendo demo sulle competenze, il Test Manager può valutare le competenze del candidato e determinare le aree di forza e di debolezza. Una volta creato un buon set di valutazioni, dovrebbe essere applicato a tutto il personale esistente per determinare le aree di crescita e la formazione.

Il Test Manager deve anche possedere, oltre alle stesse competenze dei propri collaboratori, eccellenti competenze di comunicazione e capacità diplomatiche. Il Test Manager deve essere in grado di disinnescare le situazioni controverse, deve conoscere i mezzi di comunicazione corretti da utilizzare, e deve essere in grado di focalizzarsi sulla creazione e sul mantenimento di relazioni all'interno dell'organizzazione.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

7.4 Adattamento del Testing all'Interno di un'Organizzazione

Le organizzazioni hanno molti modi per adattare il testing alla propria struttura organizzativa. Benché la qualità sia di responsabilità di ognuno durante tutto il ciclo di vita dello sviluppo software, un team di test indipendente può contribuire significativamente a un prodotto di qualità. Il livello di indipendenza del testing varia considerevolmente nella pratica, come si vede dal seguente elenco, ordinato per livello crescente di indipendenza:

- Nessun tester indipendente
  - In questo caso non esiste indipendenza e lo sviluppatore esegue il testing del proprio codice
  - Lo sviluppatore, se ha il tempo sufficiente per eseguire il testing, determinerà se il codice funziona come previsto, il che può o meno corrispondere ai requisiti effettivi
  - Lo sviluppatore è in grado di correggere velocemente eventuali difetti rilevati
- Il testing è svolto da uno sviluppatore differente da colui che ha scritto il codice
  - Esiste poca indipendenza tra lo sviluppatore e il tester
  - Uno sviluppatore che sta eseguendo il testing del codice di un altro sviluppatore può essere riluttante a tracciare i difetti
  - La mentalità di uno sviluppatore verso il testing è di solito focalizzata sui test case positivi
- Il testing è svolto da un tester (o un team di test) che fa parte del team di sviluppo
  - Il tester (o team di test) può riportare al project management al responsabile dello sviluppo
  - La mentalità del tester è focalizzata maggiormente sulla verifica dell'aderenza ai requisiti
  - Poiché il tester è un membro del team di sviluppo, può avere responsabilità di sviluppo in aggiunta al testing
  - Il responsabile del tester può essere più focalizzato sul mantenere la schedulazione piuttosto che sul raggiungimento degli obiettivi di qualità
  - Nel team, il testing può avere un livello di importanza inferiore rispetto allo sviluppo
  - Il tester può non avere l'autorità per influenzare gli obiettivi di qualità o per aderire a questi obiettivi
- Il testing viene svolto da specialisti del test dell'organizzazione di business, della comunità di utenti, o di altre organizzazioni tecniche non di sviluppo
  - Le informazioni sui risultati dei test sono riportati oggettivamente agli stakeholder
  - La qualità è l'obiettivo principale di questo team
  - Lo sviluppo e la formazione delle competenze sono focalizzati sul testing
  - Il testing è visto come un percorso di carriera
  - Esiste un management specifico dedicato al gruppo di test, focalizzato sulla qualità
- Il testing viene svolto su specifici tipi di test da specialisti di test esterni
  - L'expertise viene applicata a specifiche aree dove il testing generalizzato è probabilmente insufficiente
  - I tipi di test potrebbero essere l'usabilità, la sicurezza, le prestazioni o altre aree dove la specializzazione è necessaria
  - La qualità dovrebbe essere il focus di questi specialisti, ma la loro vista è limitata ad altre aree di specializzazione. Un prodotto che funziona bene può non soddisfare i suoi requisiti funzionali, ma può non essere rilevato dallo specialista delle prestazioni
- Il testing viene svolto da un'organizzazione esterna all'azienda
  - Questo modello definisce la massima indipendenza tra il tester e lo sviluppatore
  - Il trasferimento delle conoscenze può non essere sufficiente per il tester per eseguire un testing in modo competente
  - Saranno necessari requisiti chiari e una struttura di comunicazione ben definita
  - La qualità dell'organizzazione esterna deve essere verificata regolarmente attraverso audit

Questo elenco si basa sul tipico focus dei singoli individui, ma questo può essere non vero in una particolare organizzazione. Una persona e un titolo non determinano necessariamente il focus della persona. I responsabili dello sviluppo possono essere focalizzati sulla qualità e così possono essere bravi Test Manager. Test Manager indipendenti possono riportare a una catena manageriale focalizzata sulla schedulazione e quindi possono essere più focalizzati sulla schedulazione piuttosto che sulla

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

qualità. Per determinare la posizione migliore di un team di test all'interno di un'organizzazione, il Test Manager deve comprendere gli obiettivi dell'organizzazione.

Esistono diversi gradi di indipendenza tra le organizzazioni di sviluppo e di test. È importante comprendere che può esistere un compromesso dove una maggiore indipendenza comporta un maggior isolamento e un minor trasferimento delle conoscenze. Un livello inferiore di indipendenza può aumentare la conoscenza, ma può anche introdurre obiettivi in conflitto tra loro. Il livello di indipendenza può anche essere determinato dal modello di sviluppo software in uso, p.e., all'interno dello sviluppo Agile i tester sono più spesso parte del team di sviluppo.

Una qualsiasi delle opzioni sopra può essere presente in un'organizzazione. Il testing può essere svolto all'interno dell'organizzazione di sviluppo oppure da un'organizzazione di test indipendente, e può essere svolta una certificazione finale da parte di un'organizzazione esterna. È importante comprendere le responsabilità e le aspettative di ogni fase del testing, e definire quei requisiti che massimizzano la qualità del prodotto finale, pur rimanendo all'interno dei vincoli di schedulazione e budget.

7.5 Motivazione

Esistono molti modi per motivare un singolo individuo con una posizione nel testing. Questi includono:

- Riconoscimento per il lavoro svolto
- Approvazione dal management
- Rispetto all'interno del team di progetto e tra colleghi
- Incremento della responsabilità e dell'autonomia
- Ricompense adeguate al lavoro svolto (che includono aumenti di stipendio, aumento di responsabilità, riconoscimenti)

Esistono influenze di progetto che possono rendere questi strumenti motivazionali di difficile applicazione. Ad esempio, un tester può lavorare molto duramente in un progetto che ha una scadenza impossibile. Il tester può fare tutto ciò che è nelle sue possibilità per guidare il focus del team sulla qualità, può svolgere ore extra o effort aggiuntivo, e comunque il prodotto potrebbe essere rilasciato prima del dovuto a causa di influenze esterne. Il risultato può essere un prodotto di scarsa qualità, nonostante i migliori sforzi del tester. Questo può essere facilmente un fattore demotivante se il contributo del tester non è stato compreso e misurato, indipendentemente dal successo o meno del prodotto finale.

Il team di test deve assicurarsi di tracciare le metriche appropriate per dimostrare la bontà del lavoro svolto per realizzare il testing, mitigare i rischi e memorizzare accuratamente i risultati. Se questi dati non vengono raccolti e pubblicati, è facile per un team demotivarsi quando non riceve il riconoscimento che si aspetta per un lavoro ben fatto.

Il riconoscimento non è manifestato solo in modo non tangibile con il rispetto e l'approvazione, ma anche in modo concreto con opportunità di promozione, maggiore responsabilità e aumenti di merito. Se il team di test non è rispettato, queste opportunità possono non essere disponibili.

Il riconoscimento e il rispetto vengono acquisiti quando è chiaro che il tester contribuisce ad incrementare il valore del progetto. In un singolo progetto questo si ottiene coinvolgendo il tester all'inizio del progetto e con un continuo coinvolgimento durante il ciclo di vita. Nel corso del tempo i tester acquisiranno rispetto dagli altri stakeholder di progetto per il loro contributo positivo al progetto. Questo contributo dovrebbe anche essere quantificato in termini di riduzione del costo della qualità e di mitigazione del rischio.

Il Test Manager gioca un importante ruolo nel motivare i singoli individui di un team di progetto e nel vedere il team di progetto come un vincitore all'interno di un'organizzazione più ampia. Il Test manager dovrebbe riconoscere i risultati ottenuti dai singoli tester, e deve anche dare valutazioni corrette e oneste degli errori commessi. Un Test Manager equo e coerente motiverà i membri del team dando l'esempio.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

7.6 Comunicazione

La comunicazione del team di test avviene principalmente con i seguenti mezzi:

- Documentazione dei prodotti di test, strategia di test, test plan, test case, test summary report, defect report, ecc.
- Feedback forniti sui documenti sottoposti a review - requisiti, specifiche funzionali, use case, documentazione dei test di componente, ecc.
- Raccolta e diffusione di informazioni - interazione con gli sviluppatori, gli altri membri del team di test, il management, ecc.

La comunicazione tra i tester e gli altri stakeholder deve essere professionale, oggettiva ed efficace, per poter creare e mantenere il rispetto per il team di test. Diplomazia e oggettività sono richieste quando si forniscono feedback, in particolare feedback costruttivi, sui prodotti del lavoro degli altri. Inoltre, la comunicazione dovrebbe essere focalizzata al raggiungimento degli obiettivi di test e al miglioramento della qualità sia dei prodotti sia dei processi utilizzati per produrre i sistemi software.

I Test Manager comunicano con un vasto pubblico, includendo utenti, membri del team di progetto, management, gruppi di test esterni e clienti. La comunicazione deve essere efficace per il pubblico target. Ad esempio, un report realizzato per il team di sviluppo che mostra il trend e la tendenza sul numero e sulla severità dei difetti rilevati, potrebbe essere troppo dettagliato per essere appropriato durante un briefing dell'executive management. In qualsiasi comunicazione, ma in particolare durante le presentazioni, è importante comprendere il messaggio che viene inviato, il modo con cui il messaggio può essere ricevuto e la spiegazione necessaria per creare le condizioni giuste affinché il messaggio venga accettato. Dal momento che il Test Manager è spesso nella posizione di dover presentare informazioni sullo stato del progetto, è importante che queste informazioni siano a un livello appropriato di dettaglio (p.e., i manager di solito vogliono vedere il trend dei difetti piuttosto che i singoli difetti),e che siano descritte in modo chiaro e facilmente comprensibile (p.e., grafici semplici e colorati).Comunicare in modo efficace aiuta a mantenere l'attenzione del pubblico, trasmettendo comunque il messaggio corretto. Ogni presentazione dovrebbe essere considerata dal Test Manager come un'opportunità per promuovere la qualità e i processi di qualità.

Un Test Manager non comunica solo con persone esterne all'organizzazione (comunicazione esterna). Una parte importante del lavoro del Test Manager è comunicare in modo efficace all'interno del team di test (comunicazione interna) per trasmettere notizie, istruzioni, modifiche nella priorità e altre informazioni standard che viene fornita nel normale processo di test. Un Test Manager può anche comunicare con singoli individui, di livello superiore (comunicazione verso l'alto) e inferiore (comunicazione verso il basso) nella catena manageriale dell'organizzazione. Indipendentemente dalla direzione della comunicazione, vengono applicate le stesse regole; la comunicazione dovrebbe essere appropriata all'audience, il messaggio dovrebbe essere trasmesso in modo efficace e la sua comprensione dovrebbe essere confermata.

I Test Manager devono padroneggiare con i diversi mezzi di comunicazione. Molte informazioni vengono comunicate via e-mail, attraverso interazioni verbali, meeting formali o informali, report formali o informali, e anche attraverso l'uso di strumenti di test management, come gli strumenti di defect management. Tutte le comunicazioni devono essere professionali e oggettive. Una verifica preventiva, sia di contenuto che di qualità, è un passo necessario anche nella comunicazione più urgente. La comunicazione scritta vive spesso a lungo, oltre la durata del progetto attuale. E 'importante per il Test Manager produrre documentazione professionale di qualità, che è rappresentativa di un'organizzazione che promuove la qualità.

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

8. Riferimenti

8.1 Standard

- [BS7925-2] BS 7925-2, Software Component Testing Standard.
- [FDA21] FDA Title 21 CFR Part 820, Food and Drug Administration, USA
- [IEEE829] IEEE Standard for Software and System Test Documentation
- [IEEE1028] IEEE Standard for Software Reviews and Audits
- [IEEE1044] IEEE Standard Classification for Software Anomalies
- [ISO9126] ISO/IEC 9126-1:2001, Software Engineering - Software Product Quality
- [ISO12207] ISO 12207, Systems and Software Engineering, Software Life Cycle Processes
- [ISO15504] ISO/IEC 15504, Information Technology - Process Assessment
- [ISO25000] ISO/IEC 25000:2005, Software Engineering - Software Product Quality Requirements and Evaluation (SQuaRE)
- [RTCA DO-178B/ED-12B]: Software Considerations in Airborne Systems and Equipment Certification, RTCA/EUROCAE ED12B.1992.

8.2 Documenti ISTQB

- [ISTQB_AL_OVIEW] ISTQB Advanced Level Overview, Version 1.0
- [ISTQB_ALTM_SYL] Syllabus ISTQB Advanced Level Test Manager, Version 1.0
- [ISTQB_ALTTA_SYL] Syllabus ISTQB Advanced Level Technical Test Analyst Syllabus, Version 1.0
- [ISTQB ETM SYL] Syllabus ISTQB Expert Test Management, Version 1.0
- [ISTQB_FL_SYL] Syllabus ISTQB Foundation Level Syllabus, Version 2011
- [ISTQB_GLOSSARY] Standard glossary of terms used in Software Testing
- [ISTQB ITP SYL] Syllabus ISTQB Expert Improving the Test Process, Version 1.0

8.3 Marchi registrati

I seguenti marchi registrati sono riportati nel presente documento:

- CMMI®, IDEAL℠, ISTQB®, ITIL®, PRINCE2®, TMMi®, TPI Next®
- CMMI is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.
- IDEAL is a service mark of Software Engineering Institute (SEI), Carnegie Mellon University
- ISTQB is a registered trademark of the International Software Testing Qualifications Board
- ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office
- PRINCE2 is a registered trademark of the Cabinet Office
- TMMi is a registered trademark of TMMi Foundation
- TPI Next is a registered trademark of Sogeti Nederland B.V.

8.4 Libri

[Black03]: Rex Black, “Critical Testing Processes,” Addison-Wesley, 2003, ISBN 0-201-74868-1
[Black09]: Rex Black, “Managing the Testing Process, third edition,” John Wiley & Sons: New York, 2009, ISBN 0-471-22398-0
[Black11]: Rex Black, Erik van Veenendaal, Dorothy Graham, “Foundations of Software Testing,” Thomson Learning, 2011, ISBN 978-1-84480-355-2
[Craig02]: Rick Craig, Stefan Jaskiel, “Systematic Software Testing,” Artech House, 2002, ISBN 1-580-53508-9

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board

[de Vries09]: Gerrit de Vries, et al., “TPI Next”, UTN Publishers, 2009, ISBN 90-72194-97-7
[Goucher09]: Adam Goucher, Tim Riley (editors), “Beautiful Testing,” O’Reilly, 2009, ISBN 978-0596159818
[Jones07]: Capers Jones, “Estimating Software Costs, second edition,” McGraw-Hill, 2007, ISBN 978-0071483001
[Jones11]: Capers Jones and Olivier Bonsignour, “Economics of Software Quality,” Pearson, 2011, ISBN 978-0132582209
[McKay07]: Judy McKay, “Managing the Test People,” Rocky Nook, 2007, ISBN 978-1933952123
[Musa04]: John Musa, “Software Reliability Engineering, second edition,” Author House, 2004, ISBN 978-1418493882
[Stamatis03]: D.H. Stamatis, “Failure Mode and Effect Analysis,” ASQC Quality Press, 2003, ISBN 0-873-89300
[vanVeenendaal11] Erik van Veenendaal, “The Little TMMi,” UTN Publishers, 2011, ISBN 9-490-986038
[vanVeenendaal12] Erik van Veenendaal, “Practical Risk-based Testing,” UTN Publishers, 2012, ISBN 978-9490986070
[Whittaker09]: James Whittaker, “Exploratory Testing,” Addison-Wesley, 2009, ISBN 978-0321636416
[Weigers03]: Karl Weigers, “Software Requirements 2,” Microsoft Press, 2003, ISBN 978-0735618794

8.5 Altri Riferimenti

I seguenti riferimenti riportano informazioni disponibili su Internet. Questi riferimenti sono stati verificati al momento della pubblicazione del Syllabus Advanced Level.

http://www.istqb.org
http://www.sei.cmu.edu/cmmi/
http://www.tmmi.org/
http://www.tpinext.com/

© International Software Testing Qualifications Board / ITAlian – Software Testing Qualification Board