WHITE PAPER | SETTEMBRE 2016

Testing, analisi e processo decisionale?

In questa serie di articoli, Paul Gerrard, Principal di Gerrard Consulting, si occupa di alcuni temi chiave dell'ambito del testing. Nello specifico in questo articolo, Paul Gerrard esamina il ruolo di analisi e reporting di test a supporto del processo decisionale. Come fare per quantificare il testing? Quantificare i costi è abbastanza semplice, ma quanta attività di test è sufficiente? E comunque, qual è il valore del testing? Si tratta di quesiti non facili, cui i tester si trovano a dover rispondere.

L'analisi dell'attività di test è una disciplina che riguarda l'intero ciclo di vita, dall'idea allo sviluppo alla produzione, fino alla dismissione finale, e va al di là del semplice utilizzo dell'output degli strumenti di gestione test.

Paul Gerrard
Gerrard Consulting

Sponsorizzato da

ca technologies

Introduzione

Al livello più essenziale, lo scopo dei test consiste nel raccogliere informazioni su alcuni aspetti di un sistema e, potenzialmente, prendere una decisione sulla base dei risultati di uno o più test.[1] I test forniscono le informazioni più preziose necessarie agli sviluppatori (per correggere i difetti), ai project manager (per comprendere lo stato di avanzamento e gestirlo) e agli stakeholder (per rimanere aggiornati e informati). In questo senso, il testing è (quasi) onnipotente: rappresenta l'unica fonte di conoscenza per il corretto completamento dei progetti sistemici.

Nello spazio digitale o mobile, i business rilasciano app che acquisiscono informazioni sui loro utenti e sugli schemi di utilizzo delle app stesse. Le app moderne raccolgono dati in tempo reale, e li ritrasmettono al fornitore sempre in tempo reale, o quasi. Questi dati vengono analizzati per individuare trend, schemi di comportamento, preferenze degli utenti e opportunità di miglioramento, oppure nuove iniziative di mercato. Le app sono pensate anche come strumenti per raccogliere informazioni che modellano il processo decisionale.

Introduzione all'analisi di testing

Nel mio articolo, "Thinking Big: Introducing Test Analytics",[2] sostengo l'opportunità di pensare ai test di pre-release e al monitoraggio e analisi in produzione post-release come a due attività senza soluzione di continuità. Utilizzo l'espressione "analisi di testing" per indicare una disciplina che riguarda l'intero ciclo di vita, dall'idea allo sviluppo alla produzione, fino alla dismissione finale. Una disamina completa relativa a testing e analisi mobile è reperibile in "The Mobile Analytics Playbook".[3]

La mia definizione di analisi di testing:

> *L'acquisizione, l'integrazione e l'analisi dei dati di monitoraggio di produzione e test per dare forma al processo decisionale di business e collegato allo sviluppo software*

Solitamente, i tester gestiscono l'output degli strumenti di gestione di test come principale fonte di informazioni per il reporting: ma si tratta di punto di vista troppo limitato. Numero di casi di test, requisiti gestiti (qualunque cosa significhi), report di incidenti o bug segnalati, diagnosticati, corretti o ignorati, serie di sintesi e serie in-time forniscono tutti informazioni interessanti, ma anche limitanti.

Le pratiche moderne: opportunità per il test

I precedenti articoli parte di questa serie fanno luce su come migliorare le attività di reporting e di analisi.
Il ragionamento alla base del cosiddetto approccio "shift-left "(esecuzione di test in parallelo con le attività di sviluppo), dei modelli di test, di DevOps e dell'automazione, e le discipline di test emergenti che questi favoriscono, ci costringono a ripensare quello che intendiamo per reporting di test. L'obiettivo fondamentale dell'attività di test, ovvero raccogliere e analizzare i dati sui quali basare il processo decisionale, resta invariato, ma le opportunità di miglioramento della modalità di reporting dei dati che ne derivano sono numerose:

1. La disciplina "shift-left" mira a ridurre, se non a eliminare, i fraintendimenti sui requisiti nelle prime fasi del ciclo di vita. In questo modo eviteremo che, nelle fasi successive, requisiti, difetti e crisi generino costi e confondano le acque del reporting.
2. Il passaggio all'automazione pervasiva nei regimi DevOps genera gran parte dei dati necessari in modo automatico. L'acquisizione e le analisi dei risultati non sono più manuali; il reporting avviene quasi istantaneamente.

3. "A New Model for Testing"[4] mette la modellazione al centro della progettazione dei test. Quando i tester utilizzano modelli significativi, anche la misurazione della copertura che diventa possibile risulta più rilevante.
4. L'analisi di testing può essere acquisita anche durante sessioni esplorative. Modelli di utilizzo e di sistema possono essere creati prima delle sessioni di test, come riferimenti di base, mentre la copertura dei test durante le sessioni può essere acquisita durante l'attività.
5. Il testing in produzione è sempre più comune. A cosa è dovuto questo? Nessuno può testare tutti i tipi di device mobile (l'ecosistema Android include più di 24.000 device).[5] Il valore dell'analisi di monitoraggio in produzione è evidente.
6. I prodotti di monitoraggio, registrazione, generazione di avvisi e analisi sono disponibili a un livello più ampio. Le applicazioni, così come l'infrastruttura, sono già equipaggiate in questo senso. Scenari che i tester non potevano riprodurre in laboratorio oggi possono essere monitorati (e, quindi, testati) sul campo.
7. Alcune aziende stanno abbandonando procedure burocratiche di gestione degli incidenti. I difetti vengono corretti alla rilevazione.
8. Gli incidenti sono un retaggio del passato. Lo strumento di controllo del codice sorgente, i test unitari automatizzati che indirizzano e proteggono le modifiche, forniscono evidenza e nuove visualizzazioni del cambiamento e dell'impatto. Vedere la visualizzazione del codice sorgente "History of Python" per un esempio di ciò che è possibile ottenere.[6]

Grazie a queste nuove pratiche, tecniche, strumenti e visualizzazioni, la natura dell'attività di test sta cambiando. Esiste un'opportunità reale per migliorare il nostro prodotto chiave: le informazioni che forniamo agli stakeholder. Ma in che modo l'attività di test supporta il processo decisionale?

Attività di test e processo decisionale

Nel corso degli anni, ho partecipato a molti progetti di test, di varie dimensioni. In ciascuno di essi, il momento inevitabile della decisione arriva al termine di una fase di test: qual è il prossimo passo? Eseguiamo la release nella prossima fase? Eseguiamo la release in produzione o la mettiamo a disposizione dei clienti? Rimandiamo il progetto e lo ampliamo? Ci fermiamo per ripensare i nostri obiettivi? Abbandoniamo ogni speranza e rinunciamo al progetto?

Tutte domande apparentemente importanti: ma l'adozione di una decisione avviene a molti livelli. Ad esempio, quando un tester rileva un problema e ne discute con un utente e con lo sviluppatore. Le decisioni arrivano rapidamente: Si tratta di un bug? Può essere corretto? Può essere corretto rapidamente? Qual è l'impatto della modifica? Esiste una soluzione alternativa praticabile? Vale la pena procedere alla correzione? Il problema ha qualche importanza per l'utente?

Il testing supporta il processo decisionale a tutti i livelli. I ruoli degli stakeholder di test (dirigenti, clienti, utenti, project manager, Operations e sviluppatori) possono essere molteplici. Di conseguenza, le informazioni che l'attività di test deve fornire variano in base alla prospettiva e alla necessità di prendere decisioni dei loro destinatari. Non esiste una formula magica per stabilire quali informazioni sono necessarie: dobbiamo chiedere ai nostri stakeholder quali dati sarebbero per loro più utili, preziosi ed economici.

Ma c'è un problema. Come fare per quantificare il testing? Quantificare i costi è abbastanza semplice, ma quanta attività di test è sufficiente? E comunque, qual è il valore del testing? Si tratta di quesiti non facili, cui i tester si trovano a dover rispondere.

Sono un grande appassionato di fisica e mi sono divertito ad applicare alcune etichette "scientifiche" a un principio e a due teorie.

Test funzionali su larga scala

Quando testiamo la funzionalità dei componenti a un livello superiore dell'architettura, soprattutto a livello di integratori e di applicazioni, possiamo trovarci a dover simulare migliaia o milioni di device sul campo. Il numero di combinazioni e permutazioni potrebbe essere superiore a qualsiasi capacità di calcolo o previsione. Dalle nostre simulazioni deriveranno continuamente scenari da testare, risultati da registrare e, potenzialmente, simulazioni da ripetere per studiarle in un secondo momento.

I componenti di livello superiore devono essere testabili. Avremo bisogno di elementi come gestori di eccezioni, utilità che inseriscono dati, acquisiscono e replicano o riproducono scenari. Cem Kaner ha scritto parecchio di quello che definisce "high volume automated testing" (testing automatizzato a volumi elevati)[7], che costituisce un buon punto di partenza.

Queste tecniche potrebbero anche essere definite testing di Big Data. Avremo bisogno di quanto segue:

- Trovare dati adatti al nostro scopo
- Generare, contrassegnare, modificare ed eseguire il seed dei dati, in modo da poterne tracciare l'utilizzo
- Strumenti per monitorare l'utilizzo dei dati contrassegnati e della capacità di riconciliare i dati derivanti da raccolta, storage, utilizzo e smaltimento
- Nuovi strumenti di visualizzazione di test per supportare diagnostica e debugging

Tutto gira intorno ai volumi. I singoli test possono essere importanti o meno, ma la gestione di risultati, visualizzazioni e processi decisionali su larga scala richiederà molto tempo.

Il principio di indeterminazione dei test

Se hai mai pianificato di eseguire "abbastanza" test e hai mai previsto una data di completamento per l'attività di test, sai bene quanto sia difficile ottenere una stima attendibile. L'elemento che prevedibilmente creerà problemi, nonché la principale variabile nell'esecuzione dei test pianificati, è relativo al volume di test che verrà bloccato e all'attività di retesting che sarà necessaria. Ma questo fattore dipende principalmente dal numero di bug rilevati, e da quanto è complesso correggerli e testarli. Informazioni che non sono note fino a dopo il completamento della maggioranza delle attività di test, correzione e re-test. È uno dei tanti paradossi del testing.

La sfida insita nella previsione e nella pianificazione dei test è agilmente espressa dal principio di indeterminazione dei test:

- Possiamo prevedere lo stato del test, ma non quando verrà raggiunto.
- Possiamo prevedere quando un test terminerà, ma non il suo stato.

Puoi definire un criterio d'uscita o di completamento (ad esempio, tutti i test eseguiti e superati), ma saprai con certezza quando raggiungerai quel punto solo quando l'avrai raggiunto. Con quale frequenza impostiamo i criteri di uscita e non riusciamo a soddisfarli? Dobbiamo considerare i criteri di uscita come ipotesi della pianificazione. Se non riusciamo a soddisfare i criteri, le nostre ipotesi sono sbagliate, la pianificazione è errata ed è necessario ripeterla, o rideterminare la portata del progetto.

Possiamo stabilire a priori le tempistiche di test e fissare una data di completamento. Ma qual è il volume dell'attività di test che riusciremo a eseguire durante quel periodo, e sarà sufficiente?

La teoria della relatività dei test

Il valore di un test è influenzato da molti fattori. Un singolo caso di test potrebbe riguardare cinque righe di codice nel test unitario di un componente; un altro potrebbe essere relativo a cinque milioni di righe di codice in un sistema di grandi dimensioni. Chiaramente, il valore del secondo test è maggiore. Ma chi può dire quale sia questo valore?

È necessario porci una domanda meno ambiziosa. A prescindere dall'opinione del tester, per un giudizio di valore dobbiamo affidarci ai nostri stakeholder. Questi soggetti non potranno attribuire un valore a qualsiasi test, ma di solito sono in grado di dire quale tra due o più test è più importante. Da questo punto di vista, non possiamo assegnare un valore assoluto, ma possiamo definire un valore relativo, ovvero stabilire che un test è più o meno importante di un altro. Questo può sembrare un grosso svantaggio, ma in realtà non è così.

La complessità nella pianificazione dei test è in gran parte riconducibile alla loro portata. Sappiamo che non possiamo testare tutto, quindi dobbiamo in qualche modo definire delle priorità. Conoscere il valore relativo dei test implica la capacità di selezionare quelli più importanti e di mettere gli altri da parte, almeno temporaneamente.

Solo lo o gli stakeholder per conto dei quali eseguiamo i test possono giudicarne il valore.

E per quanto riguarda i rischi di insuccesso? Il valore di un test non è in qualche modo collegato ai rischi della modalità di insuccesso che il test è pensato per evidenziare? Può essere: una valutazione del rischio (quantificato o meno) è uno dei fattori da considerare nella definizione del valore dei test. Ma rimane un problema.

Dato un test con un determinato valore, un secondo test con una funzione simile è valido quanto il primo? Solitamente no. Per spiegare questo concetto, dobbiamo fare riferimento alla teoria quantistica.

La teoria quantistica dei test

Quando eseguiamo un test, otteniamo un risultato e lo confrontiamo con una determinata aspettativa.

- Se il risultato corrisponde alle aspettative, risponde, in modo incrementale, all'esigenza o al requisito di un utente. Il test aumenta in modo incrementale la nostra conoscenza e la nostra fiducia.
- Se il risultato non corrisponde alle aspettative, non risponde, in modo incrementale, all'esigenza o al requisito di un utente. Il test aumenta in modo incrementale la nostra conoscenza, ma riduce la nostra fiducia (fino a quando il sistema non viene corretto e ri-testato con successo).

Ogni test genera un singolo "quanto" di evidenza. Un dato sì/no, vero/falso, 1 o 0. Quando tutti i test sono stati eseguiti, aggreghiamo questi "quanti" di test e otteniamo una visione più completa della situazione (incidentalmente, è quello che avviene nel ciclo "applica test, interpreta risultato" in "A New Model for Testing"[8]).

La cosa importante è che, quando eseguiamo un test, esso ha valore solo se ne deriviamo una conoscenza di cui prima non disponevamo. Se ne derivano conoscenze scarse o nulle, il test ha poco o nessun valore. Un test valido è significativo in quanto aumenta la nostra conoscenza del sistema da testare. Quando dobbiamo giudicare se due test sono meglio di uno, dobbiamo guardare al valore potenziale del secondo test e alla sua rilevanza.

Un secondo test può essere utile in sé ma, se viene abbinato a un quasi duplicato, il suo significato, e il suo valore effettivo, saranno scarsi. La rilevanza di un test è direttamente associata all'aumento incrementale della copertura che fornisce.

Il tester, o quantomeno la persona che ha progettato il modello di test, è il miglior giudice della rilevanza di un test. Per essere rilevante, qualsiasi misurazione della copertura deve sempre essere valutata rispetto a un modello di test.

Riepilogo

Fatto questo discorso, a quali conclusioni dobbiamo arrivare?

Gli approcci emergenti costituiti da DevOps, continuous delivery e shift-left offrono una grande opportunità per ridurre il problema dello scarso valore dei requisiti, per anticipare i test all'interno del ciclo e individuare e risolvere i problemi in modo meno caotico. Ma il maggiore ricorso all'automazione implica anche che sarà più facile che mai raccogliere i dati sui risultati dei test. Come approfittare di questa opportunità?

Dobbiamo creare modelli di test che siano rilevanti per i nostri stakeholder. In questo modo, lo stakeholder può riconoscere il valore dei test che proponiamo. In qualità di tester e autori di modelli, possiamo fungere da consulenti sul significato di questi test: sulla loro copertura, se vuoi, rispetto ai modelli stessi. In questo modo, possiamo ottenere test di valore e ridurre al minimo i duplicati.

La discussione relativa a valore e significato dei test aiuta anche a misurare meglio il valore dell'automazione dei nostri test. Nella mia esperienza, in molti progetti di automazione dei test una delle difficoltà si presenta nel momento in cui di un test di sistema completo viene definita la portata per l'automazione. Quei test, quando eseguiti la prima volta come test funzionali, hanno un valore. Ma quando vengono eseguiti più volte come test di regressione, il loro significato (e quindi il loro valore) si riduce notevolmente.

L'obiettivo di quello che potrebbe essere meglio definito test anti-regressione[9] consiste nell'individuare un comportamento indesiderato dopo l'esecuzione di modifiche. Forse allora solo il 10% dei tuoi test di sistema è necessario per "dare l'allarme", per così dire. Il restante 90% avrà la stessa copertura, e quindi non sarà significativo a questo riguardo.

Questo articolo dovrebbe aiutarti a tenere discussioni più informate sul processo decisionale, e a determinare quale volume di attività di test sia sufficiente e quale sia il valore del tuo lavoro come tester. Dopo tutto, teoria della relatività e teoria quantistica possono essere utili anche in questo contesto!

L'autore

Paul Gerrard è un consulente, formatore, autore, webmaster, sviluppatore, tester, relatore, allenatore di canottaggio ed editore. Ha svolto incarichi di consulenza su tutti gli aspetti del test e del QA software, specializzandosi nella test assurance. Partecipa con presentazioni e tutorial a conferenze sul testing in Europa, USA, Australia, Sud Africa e, occasionalmente, ha anche vinto dei premi grazie a questi interventi.

Ha studiato presso l'università di Oxford e l'Imperial College di Londra; nel 2010, ha vinto l'Eurostar European Testing Excellence Award e, nel 2013, l'European Software Testing Awards (TESTA) Lifetime Achievement Award.

Nel 2002, Paul Gerrard ha scritto "Risk-Based E-Business Testing", insieme a Neil Thompson, e nel 2009 "The Tester's Pocketbook". È stato co-autore di "The Business Story Pocketbook" con Susan Windsor nel 2011 e ha pubblicato "Lean Python" nel 2014.

Sempre nel 2014 Paul Gerrard è stato Programme Chair della Conferenza EuroSTAR di Dublino.

È Principal di Gerrard Consulting Limited, direttore di TestOpera Limited e host del Test Management Forum.

E-mail: paul@gerrardconsulting.com
Twitter: @paul_gerrard
Sito web: gerrardconsulting.com

Per ulteriori informazioni, visita **Sviluppo e test** con CA Technologies.

Entra in contatto con CA Technologies all'indirizzo ca.com/it

CA Technologies (NASDAQ: CA) crea software che promuove l'innovazione all'interno delle aziende, consentendo loro di cogliere le opportunità offerte dall'application economy. Il software rappresenta il cuore di qualsiasi business, in ogni settore. Dalla pianificazione allo sviluppo, fino alla gestione e alla sicurezza, CA Technologies collabora con le aziende di tutto il mondo per cambiare il nostro modo di vivere, interagire e comunicare, in ambienti mobile, cloud pubblici e privati, distribuiti e mainframe. Per ulteriori informazioni, visita il sito **ca.com/it**.

Riferimenti

1 Paul Gerrard, "The Tester's Pocketbook", http://testers-pocketbook.com
2 Pau Gerrard, "Thinking Big: Introducing Test Analytics", ottobre 2013, http://blog.gerrardconsulting.com/?q=node/630
3 Julian Harty, Antoine Aymer, "The Mobile Analytics Playbook", http://www.themobileanalyticsplaybook.com/
4 Cem Kaner, "The Insapience of Anti-Automationism", febbraio 2013, http://context-driven-testing.com/?p=69
5 Paul Gerrard, "A New Model for Testing", 2012, http://gerrardconsulting.com/?q=taxonomy/term/281
6 Cory Goldberg, "History of Python", Gource development visualization, giugno 2012, https://www.youtube.com/watch?v=cNBtDstOTmA
7 Paul Gerrard, "Regression Testing—What to Automate and How", gennaio 2010, http://gerrardconsulting.com/?q=node/547
8 Paul Gerrard, "A New Model for Testing", 2012, http://gerrardconsulting.com/?q=taxonomy/term/281
9 Cory Goldberg, "History of Python", Gource development visualization, giugno 2012, https://www.youtube.com/watch?v=cNBtDstOTmA

CS200-218519_0916