Test e collaudo del software

Continuous Integration and Testing

Relatore

Felice Del Mauro

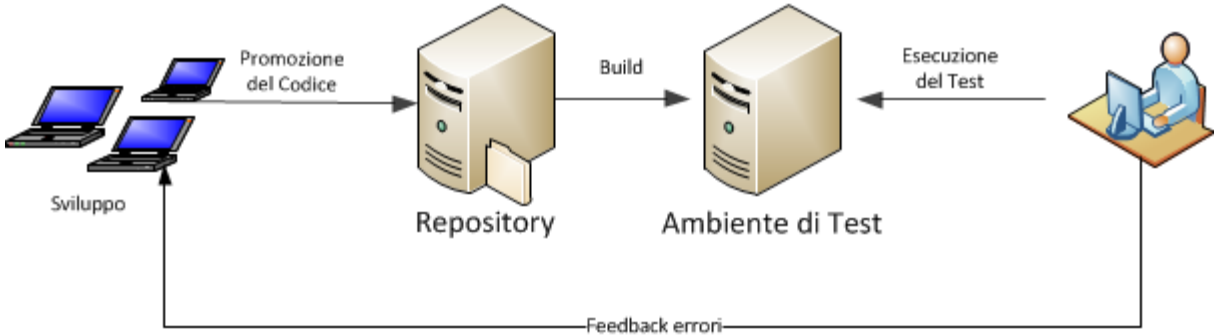
Roma, 29 novembre 2010

aicq
Associazione Italiana Cultura Qualità

Cosa è la Continuous Integration

"A software development practice where members of a team integrate their work frequently, usually each person integrates at least daily, leading to multiple integrations per day.
Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible"
(Martin Fowler)

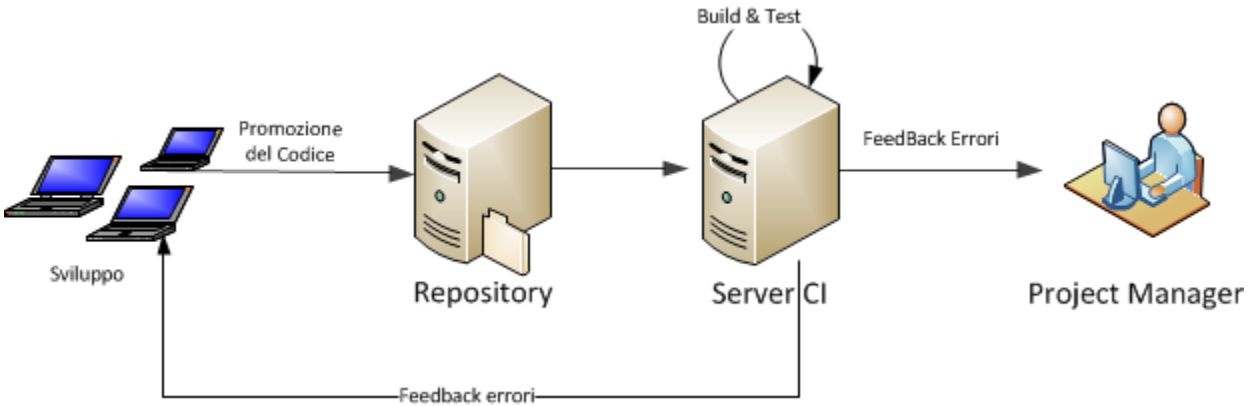
In un processo di test tradizionale ...



Il test

- è eseguito alla fine dello sviluppo
- è soprattutto di sistema
- è essenzialmente manuale

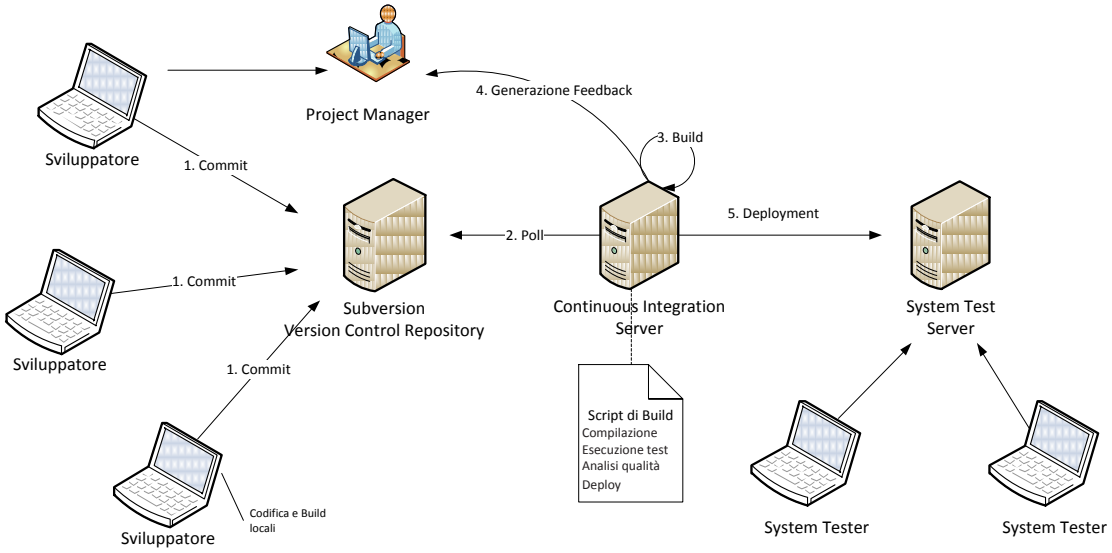
In un processo di CI ...



Il test

- viene anticipato alla fase di sviluppo
- è soprattutto automatico
- è ripetuto frequentemente ed inserito in un processo di build

I passi del ciclo ...



1. Uno sviluppatore promuove il suo codice nel Repository di Controllo delle Versioni
2. Il server CI, in polling sul repository, scopre un cambiamento
3. Il server CI estrae l'ultima copia del codice ed esegue lo script di build che integra il software;
4. Il server CI genera una mail di feedback sui risultati e la invia ai responsabili;
5. Se ci sono le condizioni, il server CI effettua il deploy nell'ambiente di test di sistema e continua il polling.

Esempi di tools a supporto della CI

Gestione della Configurazione	Subversion	http://subversion.tigris.org/
	Concurrent Versions System (CVS)	http://www.cvshome.org
Server di Integrazione	CruiseControl	http://cruisecontrol.sourceforge.net/
	Hudson	http://hudson-ci.org/
Tools di Build	ANT	http://ant.apache.org/
	Maven	http://maven.apache.org/
Test automatico (Unitario, Integrazione e Funzionale)	JUnit (test unitario classi Java)	www.junit.org
	DBUnit (test database)	www.dbunit.org
	FEST (test GUI Java Swing)	http://code.google.com/p/fest/
	Selenium (test Web Applications)	http://seleniumhq.org/

- Sono Open Source
- I più diffusi sono Java Oriented
- Non sono tutti specifici della CI (come la Gestione della Configurazione o gli strumenti di Build)

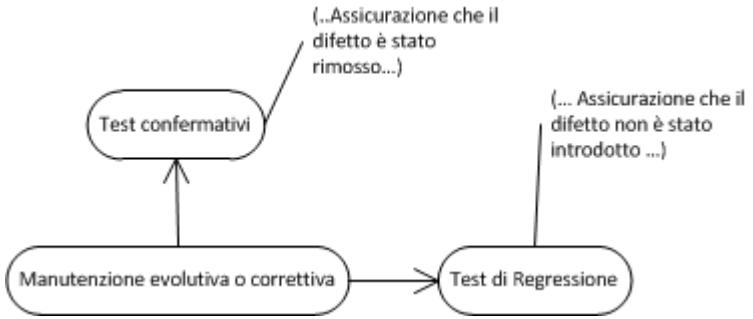
Ambiti di applicazione ideali

Applicazioni in Sviluppo

- Codifica e test unitario
- Test di Integrazione
- Test di Sistema

Applicazioni in Manutenzione

- Test Confermativi
- Test di regressione



Benefici derivanti dall'adozione della CI

- Riduzione dei rischi

  Le anomalie sono individuate e corrette fin dalle prima fasi della realizzazione in quanto i test sono eseguiti numerose volte e la probabilità di scoperta degli errori è elevata

- Riduzione dei processi manuali ripetitivi;

  La riduzione dei processi ripetitivi induce una riduzione dell'impegno effettivo (effort) facendo risparmiare tempo e costi; il tutto con un effetto positivo anche sul morale degli sviluppatori che pos-sono dedicarsi a compiti più professionali e di valore aggiunto

- Miglioramento della visibilità dello stato del progetto.

  La CI fornisce una maggiore visibilità dello stato del progetto attraverso una informativa oggettiva, dettagliata ed in tempo reale. Un sistema di CI può fornire metriche di qualità che riducono l'incertezza nelle valutazioni ed aiutano nel prendere le decisioni durante la realizzazione del sistema.

Costi derivanti dall'adozione della CI

- Sovraccarico del lavoro di automazione
  La CI si basa sull'automazione dei test di integrazione. L'attività più onerosa per il progetto è la codifica ed il mantenimento dei programmi di test; superato questo ostacolo, l'introduzione del server di CI ha un impatto trascurabile

- Strumenti per il test automatico non diffusi e non multipiattaforma
  Gli ambienti tecnologici di realizzazione delle applicazioni software sono molteplici ed estremamente eterogenei e soprattutto per le suite integrate, ad esempio applicativi di Office Automation o le suite di Business Intelligence, non non sono disponibili strumenti per il test automatico a basso costo o addirittura open source come per le tecnologie java

- Cambiamento organizzativo e metodologico
  Le pratiche implicate dalla CI, come ad esempio il refactoring o il Test Driven Development, hanno un grande impatto sul modo tradizionale di lavorare e possono disorientare in quanto si scontrano con abitudini ormai consolidate, per cui è necessaria della formazione

Strategie di introduzione della CI

- Introduzione graduale ed incrementale

  Il primo passo può essere il build automatico periodico (ad esempio settimanale o giornaliero). I passi successivi possono essere l'automazione di alcuni test con un aumento della frequenza di build, fino a raggiungere l'integrazione "continua"

- Facilitate le Aziende di Prodotto rispetto ai System Integrator

  I System Integrator, a differenza delle aziende di prodotto, operano su più progetti che adottano tecnologie diverse per le quali è più difficile trovare un tool multipiattaforma

- Ciclo di Maturazione

  Analogia con il modello TMMi (Foundation, 2009) e l'obiettivo è il livello 3:

  "testing is no longer a phase that follows coding. It is fully integrated into the development lifecycle and the associated milestones"

Grazie per l'attenzione

Relatore

Felice Del Mauro, Responsabile tecnico iStream

f.delmauro@istreamzone.it

www.istreamzone.it