28/02/2022

Il corso tratta principalmente la condivisione e creazione di software gestito in maniera condivisa. In questo senso, il programma deve essere controllato nel suo codice sorgente e poi integrato attraverso la fase di build. Lo sviluppo avviene attraverso una modalità di integrazione continua, al fine di verificare se sia effettivamente funzionante in vari ambienti di sviluppo.

Come suggerisce il nome*, una continuous delivery pipeline* è un'implementazione del paradigma continuo, in cui build, test e distribuzioni automatizzati sono orchestrati come un flusso di lavoro di rilascio. Tutto ciò fa parte della creazione Agile dei progetti, al fine di gestire vari processi tradizionali e fornire molteplici e nuove funzionalità. Tutto ciò viene garantito attraverso vari principi, prendendo ad esempio l’esplorazione continua dei bisogna del mercato/clienti attraverso feedback, integrazione continua di caratteristiche, creazione e consegna di fasi di produzione e rilascio a richiesta, rendendo il prodotto disponibile una volta correttamente pronto.

Una piccola citazione all’Agile Project Management, approccio pragmatico che cerca di creare cose semplici, efficaci, collaborando tra i gruppi e mantenendo le cose separate, utili e disciplinate. Si dà priorità all’efficacia cercando di rimuovere gli ostacoli.

Gli esempi di build tracking sono Ubuntu, con diverse migliaia di bug. Segue Jira, dove si riportano le segnalazioni, menù e a destra la versione del software in cui un certo bug è stato sistemato. Anche GitHub presenta una serie di tag ed è un buon issue tracking system (trovare quindi le criticità, mettendo nero su bianco quante più cose possibili, utile per tutti gli attori di un progetto software). Questi sistemi mantengono una lista di problemi (issues), permettendo di trovare o localizzare una serie di problemi e permettendo di risolvere anche a catena i problemi (per esempio tramite servizio clienti). Un esempio di questo sono i bugtrackers, vedendo se c’è già un bug di quel tipo.

Abbiamo per esempio l’ITS per la gestione di Progetto, che facilita il management, lo sviluppo e il ciclo di vita di un certo progetto, per esempio anche in base alle richieste del cliente, segnate come requisiti. Quando nasce una certa attività, dopo una fase di analisi, si usa una stima per capire quanto tempi si impiega per certe attività. Si devono usare il più possibile, perché magari lo sviluppatore non lo fa e lo stesso project manager dovrebbe impiegarci di più; soprattutto in un contesto reale risulta quììuna perdita di tempo effettiva.

Tutto questo serve nelle riunioni col cliente, tra team di sviluppo e Project Manager, cosiddetta SAL (Stato Avanzamento Lavori). Sicuramente tutto questo è una misura della qualità, capendo dove effettivamente si è verificata una certa problematica/bug e avendo quindi istantanee. È importante dare una giusta priorità alle singole attività, quantificando il tempo impiegato singolarmente e capendo l’effettiva efficacia. I Work Item (quindi oggetti di lavoro in un progetto) devono avere una chiara assegnazione, riportando assegnatario e segnalante.

Ottimi prodotti in questo senso Jira, Github, ecc. I Work Item sono degli strumenti utili per capire come sviluppare e lavorare su un progetto, magari dal punto di vista del cliente facendo capire le sue esigenze e come usarlo. Una cosa utile in questi strumenti è capire la situazione di un bug, riproducendolo e capendo i passi che hanno portato alla sua scoperta/segnalazione/risoluzione. Sempre in merito a questi, esistono vari campi identificativi, come il tipo (campi, stati, schermate, workflow), stato, priorità, stato di risoluzione, versione di riferimento dove è stato trovato il bug, ecc.

Altre cose utili sono le etichette, collegamenti, assegnatario, segnalante, date di ultimo aggiornamento/risoluzione/stima delle attività, tempo speso ed eventuali allegati.

Per portare a compimento le attività, l’insieme di stati e transizioni dei Work Item e del loro tempo di vita è il Workflow, capendo lo storico delle segnalazioni e tracciando soluzione/risoluzione. Determinate esigenze del cliente sono documentate negli SLA (Service Level Agreement), per esempio la risoluzione di un bug; se per esempio non potessi risolvere una problematica perché non mi è possibile usarla, la metto in stato sospeso. Ciascun requisito può essere categorizzato e classificato in macrorequisito, requisito e sottorequisito, suddividendo e specificando l’individuazione e l’utilizzo delle sottoattività.

Una volta trovato un requisito da risolvere, si associa tutto ad una specifica Change Request, capendo quanto mi costa.

Altro strumento utile sono le notifiche, che segnalano la risoluzione e fungono da tracciamento per specifiche richieste. Citiamo anche i filtri, i quali possono essere salvati ed esportati; in generale sono quindi una base per creare report, board e dashboard. I report sono suddivise in tipologie e caratterizzate graficamente da vari tipi di grafico (torta, istogramma, etc.).

Per creare un buon workflow, al di là del tempo impiegato ed evitando di buttarlo, si seguono le best practices usate dall’azienda o un contratto, che vincola la progettazione, creazione e sviluppo delle singole attività. Esso normalmente inizia, viene pianificata una versione, vengono risolte attività e poi rilasciata una versione. Nel team, in base all’ITS il manager (amministratore) definisce, una volta creato il progetto, il processo da seguire, con relativi work item, eseguendo e censendo un modello di stima e aggiungendo gli specifici utenti, assegnando utenti. Nel team di sviluppo gli utenti selezionano i work item, avviano e completano la lavorazione documentando le attività e programmando il rilascio.

Il manager monitora l’avanzamento, definendo le nuove versioni ed iterazioni.