

Come gestivamo i progetti qualche anno fa...

- MSF 3, RUP, etc...
- C'era la fase di Envisioning che oltre al documento di Vision (molto utile ancora oggi...) produceva quello che era chiamato "Iron Triangle" ovvero il triangolo di:
 - Risorse
 - Feature
 - Tempi
- C'era inoltre una quarta dimensione, ovvero la Qualità
- Non si poteva (a qualità costante, ovvero senza intervenire sulla quarta dimensione) modificare una delle tre senza modificare anche le altre...



Microsoft Visual Studio Team System

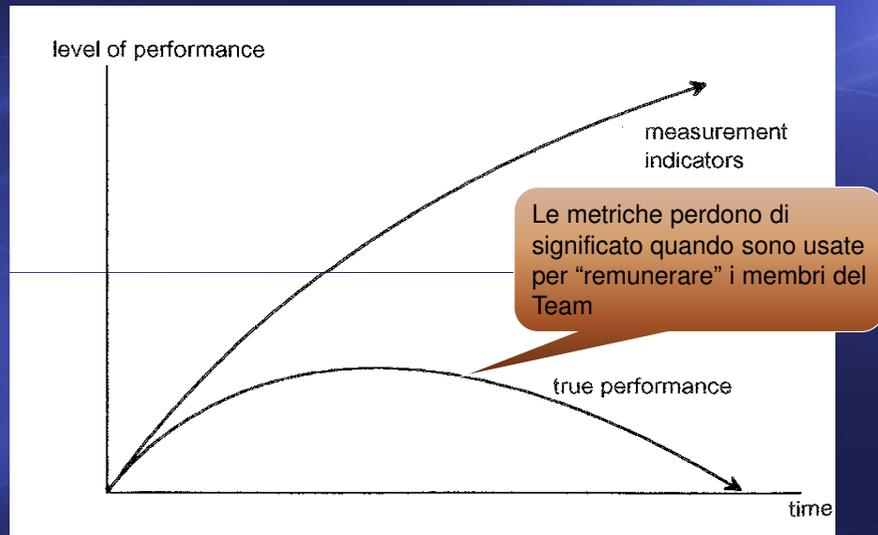
Ciclo di vita del software: strumenti, opportunità e vantaggi dell'adozione di Team Foundation Server e di Visual Studio Team System.

Ma poi il mondo è cambiato...

- Sono intervenuti una serie di cambiamenti sia di mentalità, sia nelle richieste dei clienti:
 - Sviluppo Agile
 - Agile Project Management (Declaration of Interdependence, Theory of Constraints, etc...)
 - Do more with less!
- Questi cambiamenti però sono difficili da comprendere, realizzare e portare a compimento se le uniche variabili su cui possiamo giocare sono:
 - Risorse
 - Feature
 - Tempi
 - ...e qualità!

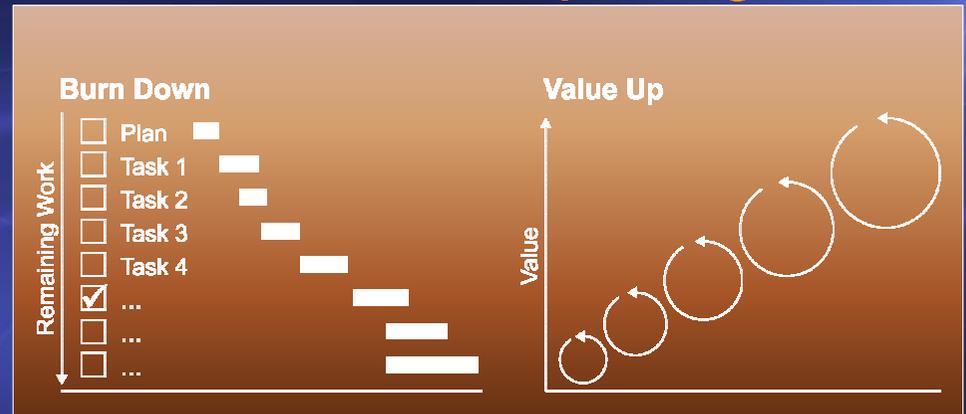
Cos'è cambiato nella gestione dei progetti

Metriche Prescrittive vs. Descrittive



Robert D. Austin, *Measuring and Managing Performance in Organizations*

Serve un cambio di paradigma...



Tradizionalmente i progetti sono stati sempre gestiti decomponendo le attività in Task e misurando l'avanzamento in base al numero di task risolti.

Un nuovo paradigma derivato dal Lean Manufacturing e dal modello della "Teoria dei Vincoli" basa gli avanzamenti sul "valore" consegnato al cliente, e non su una lista di Task completati.

Alcuni esempi:

- Misurare l'efficacia dei Tester misurando il numero di Bug trovati
 - Vengono trovati TANTI Bug stupidi e POCHI Bug che richiedono più lavoro di investigazione
- Misurare l'efficacia dei Programmatori misurando il numero di Bug che producono
 - I Programmatori sono amici dei Tester, e questi non loggano i Bug nel sistema di Bug Tracking per non arrecare danni ai loro amici...
 - I Bug vengono segnalati direttamente ai Programmatori che li correggono senza loggarli, senza farne il Triaging, etc...
 - Si perde traccia dei Bug, di quando sono stati prodotti e risolti... Il progetto è fuori controllo...
- Pensate ai Manager delle società della New Economy:
 - Erano pagati in premi e Stock Option
 - Hanno fatto gonfiare il valore delle società a dismisura...
 - I più furbi se ne sono andati prima dello scoppio della bolla!!!

Serve un cambio di paradigma...

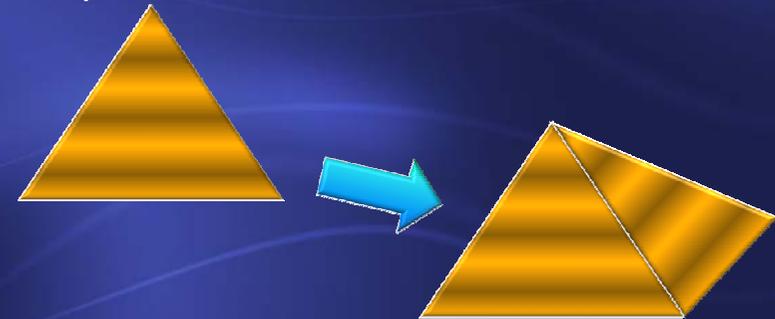
	Work-down	Value-up
Pianificazione e gestione dei cambiamenti	Pianificazione e design "up-front"	Change happens, embrace it
Indicatore principale di avanzamento progetto	Completamento dei task	Conta solo il completamento delle attività che danno "valore"
Qualità ricercata	Conformità alle specifiche	"Valore" per il customer
Accettazione della varianza	I task vanno identificati e stimati "deterministicamente"	La varianza è parte "intrinseca" del processo
Prodotti "intermedi"	Documentazione, modelli, e tutti gli artifacti intermedi richiesti dal processo	Il minimo indispensabile per minimizzare l'incertezza
Approccio alla risoluzione dei problemi	Intervenire su Risorse, Feature, Schedule e Qualità	Identificare e rimuovere i colli di bottiglia
Approccio alla fiducia	Controllo e misurazione Incentivazione legata alla conformità con la pianificazione	Pride of workmanship and teamwork

E perchè non le usavamo in precedenza?

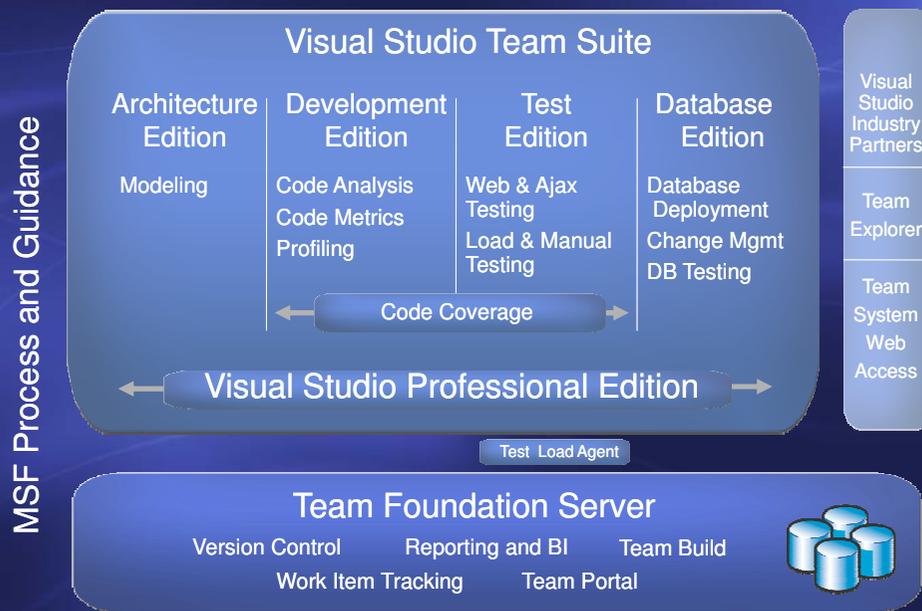
- Generalmente è molto pesante raccogliere, catalogare, interpretare i dati per ottenere delle metriche.
- Di solito le metriche che si hanno facilmente sono la conformità al piano (ritardi!) e l'andamento dei Bug
 - “Gestire un progetto utilizzando l'andamento dei Bug come indicatore è come guidare alla cieca sterzando solo quando si va a sbattere contro un ostacolo.”
- Avete provato i tool per la raccolta di metriche di progetto?
 - Non sono integrati nell'ambiente di sviluppo
 - Richiedono lavoro aggiuntivo da parte degli sviluppatori
 - Molto spesso sono web-based - gli sviluppatori devono ricordarsi URL, Login, etc...
- Morale della favola... Non si usano!!!

Metriche Monodimensionali Vs. Multidimensionali

- E' possibile analizzare le stesse informazioni in modo diverso avendo a disposizione più dati.
- Molto spesso la realtà è fatta di tante facce diverse, e vederne solo una non da una visione completa:



Visual Studio Team System



Un esempio:

- Due tester:
 - Mario – ha loggato 100 Bug in un mese nel progetto precedente
 - Luigi – ha loggato 80 Bug in un mese nel progetto precedente
- Quale scegliereste nel vostro prossimo progetto?
 - Mario? Sicuramente si guardando solo questa metrica...
- Analizziamo altri dati:
 - 40 Bug segnalati da Mario ritenuti critici e corretti prima della release – gli altri 60 rimandati ad una release successiva
 - 60 Bug segnalati da Luigi ritenuti critici e corretti prima della release – gli altri 20 rimandati ad una release successiva
- Chi scegliereste ora?
- E se vi dicessi che tra le funzionalità testate da Mario sono “scappati” 10 Bug critici e in quelle testate da Luigi solo 5?

Supported Platforms

Visual Studio Team System, Professional e Standard	• Team Explorer integrates in VS2008/VS2005
Visual Studio.NET 2003, VS 6.0, Delphi, Sql Server Management Studio, TOAD for SQL Server, Power Builder, Cobol, Access, etc...	• Standard MSSCCI Source Control Provider based on Team Explorer
Web	• Visual Studio Team System Web Access
Any Windows based Development or Design Tool	• Team Explorer GUI or Command Line • Expression v3 will have TFS integration
Eclipse or other Development Tools on any Platform	• TeamPrise (3rd party tool) cross platform TFS plug-in for Eclipse, GUI and Command Line

Team Foundation Server



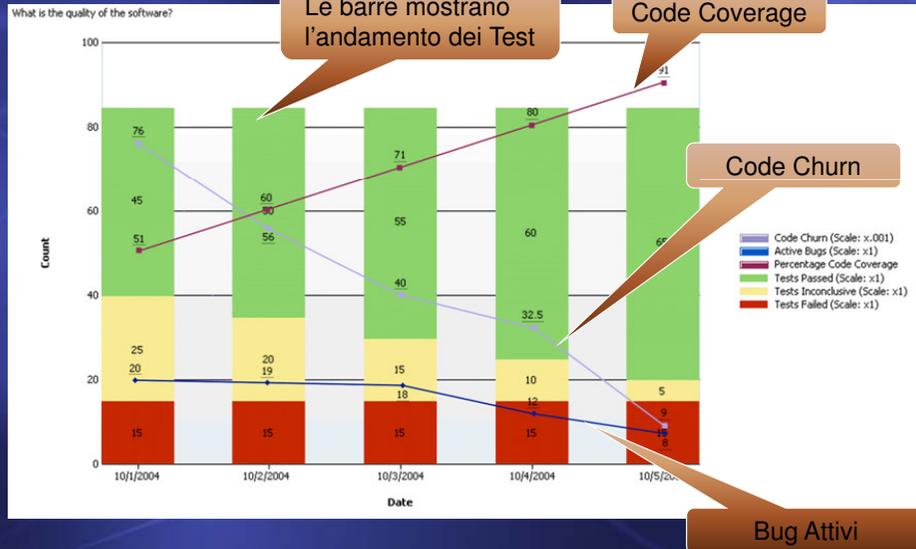
Tracking e Report con Team Foundation Server

Demo

Supported Methodologies

- Built in - Microsoft Solution Framework
 - Agile
 - CMMI
- Third parties:
 - Scrum - Various Scrum implementations
 - eXtreme Programming
 - Unified Process
 - Lean
- Custom
 - Many customers have customized an existing methodology or implemented a new one from scratch

Qual'è la qualità del lavoro prodotto?

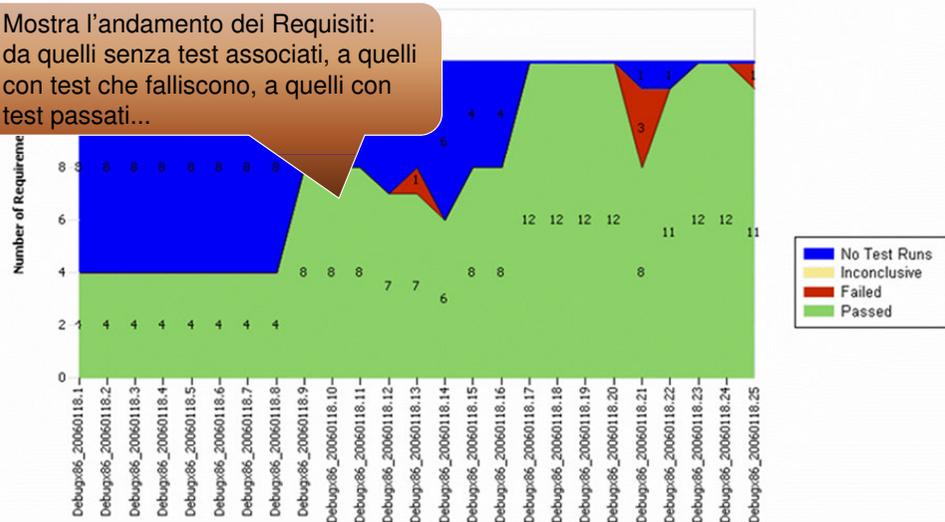


Report e WorkItem... Che informazioni ci danno?

Come va il test dei requisiti?

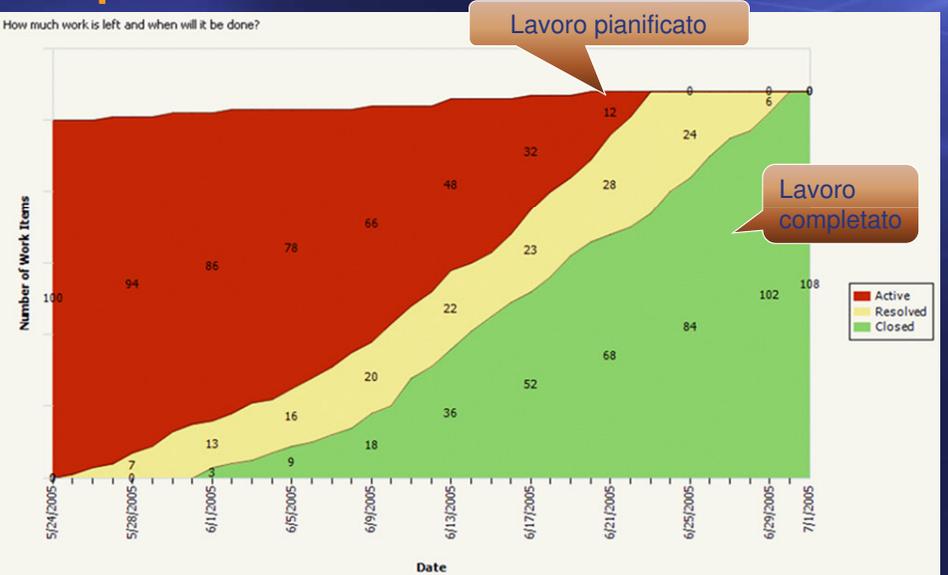
What is the test status for our requirements?

Mostra l'andamento dei Requisiti: da quelli senza test associati, a quelli con test che falliscono, a quelli con test passati...

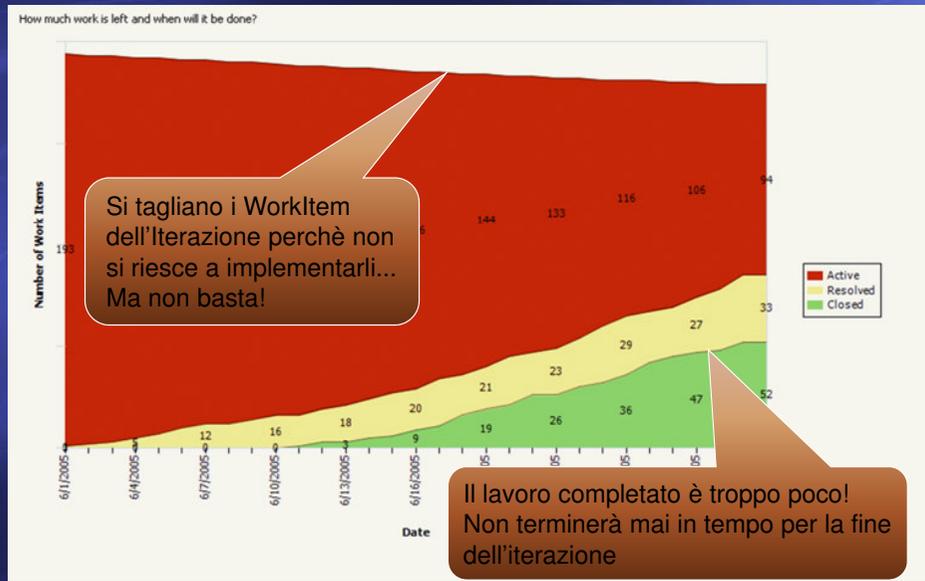


Cosa possiamo fare nel tempo disponibile?

How much work is left and when will it be done?

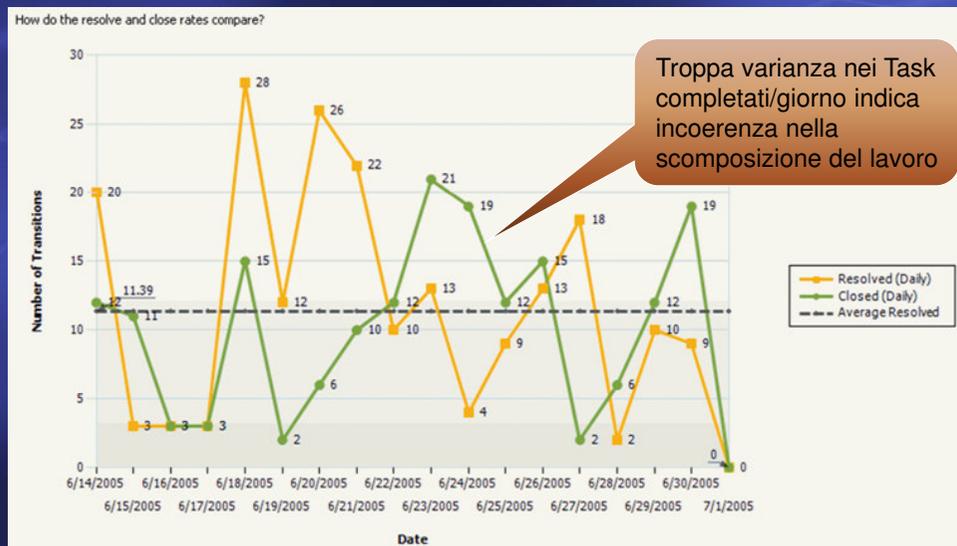


Sottostime



Usare i report per capire i problemi del progetto

Decomposizione dei task incoerente

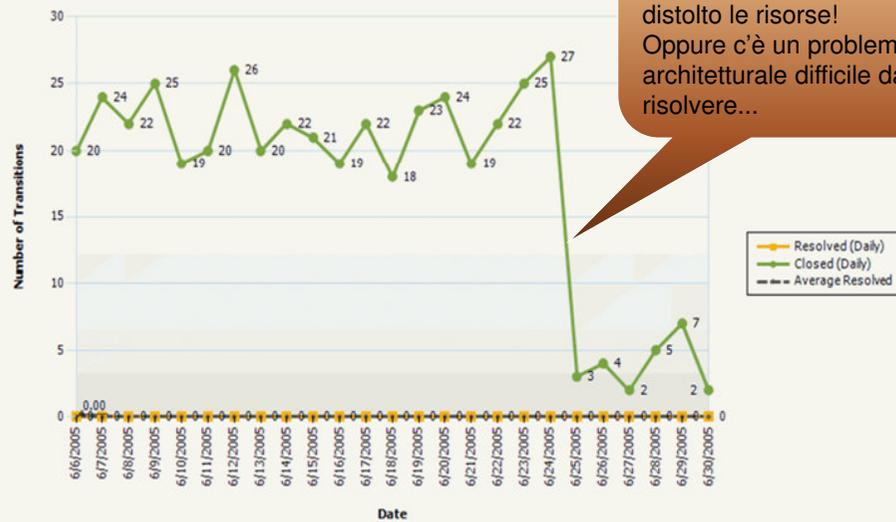


Problemi nei progetti IT

- Sottostime
- Decomposizione dei task incoerente
- Sindrome delle “macerie nascoste”
- Gestione inadeguata dei bug
- Che fine hanno fatto le risorse?
- Pratiche di sviluppo troppo lasche
- Andamento “inadeguato” delle Build
- Unit Test inadeguati
- I test passano... Ma la soluzione non funziona
- La soluzione è bloccata nella fase di test
- Troppo poco test

Che fine hanno fatto le risorse?

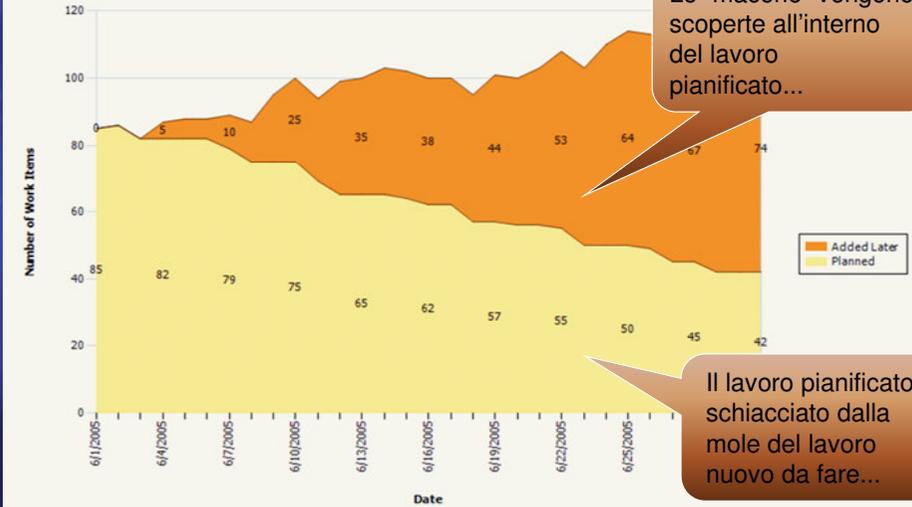
How do the resolve and close rates compare?



Se la velocità del progetto crolla... O è Agosto... O ci hanno distolto le risorse! Oppure c'è un problema architetturale difficile da risolvere...

Sindrome delle "macerie nascoste"

How much unplanned work do we have?

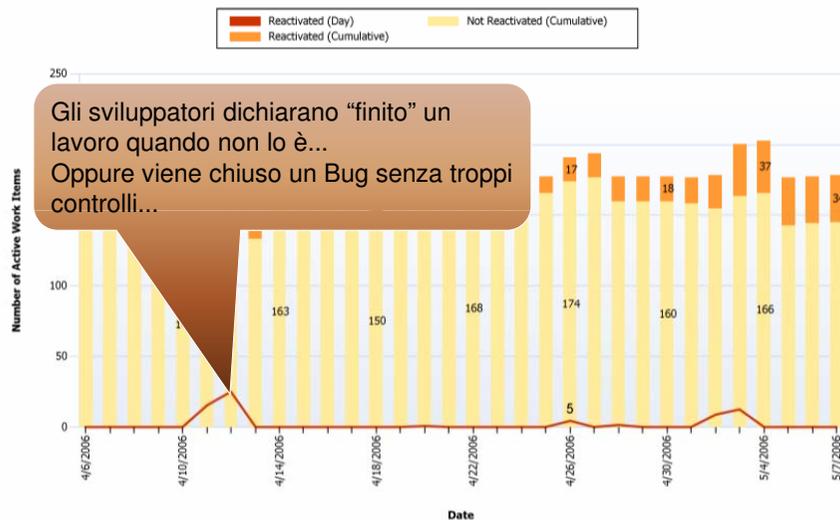


Le "macerie" vengono scoperte all'interno del lavoro pianificato...

Il lavoro pianificato è schiacciato dalla mole del lavoro nuovo da fare...

Pratiche di sviluppo troppo lasche

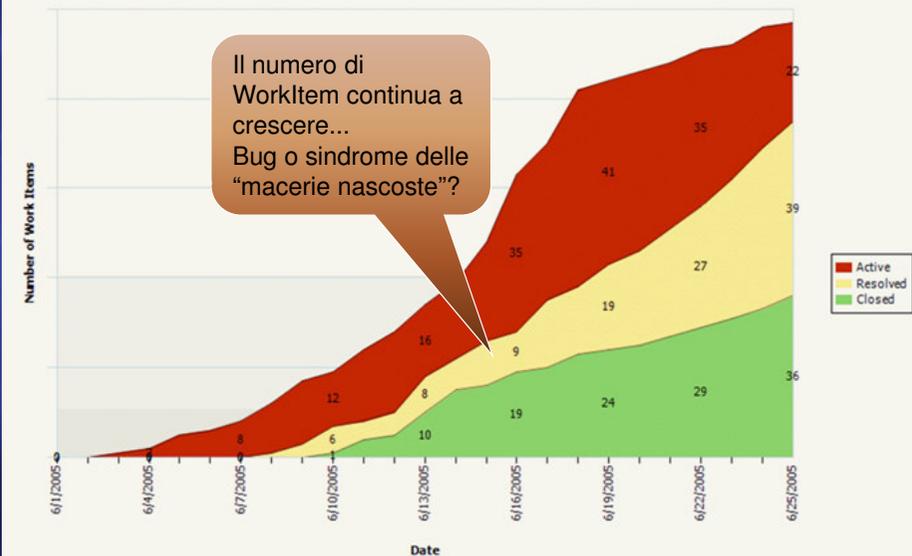
How much rework is being generated through poor initial quality?



Gli sviluppatori dichiarano "finito" un lavoro quando non lo è... Oppure viene chiuso un Bug senza troppi controlli...

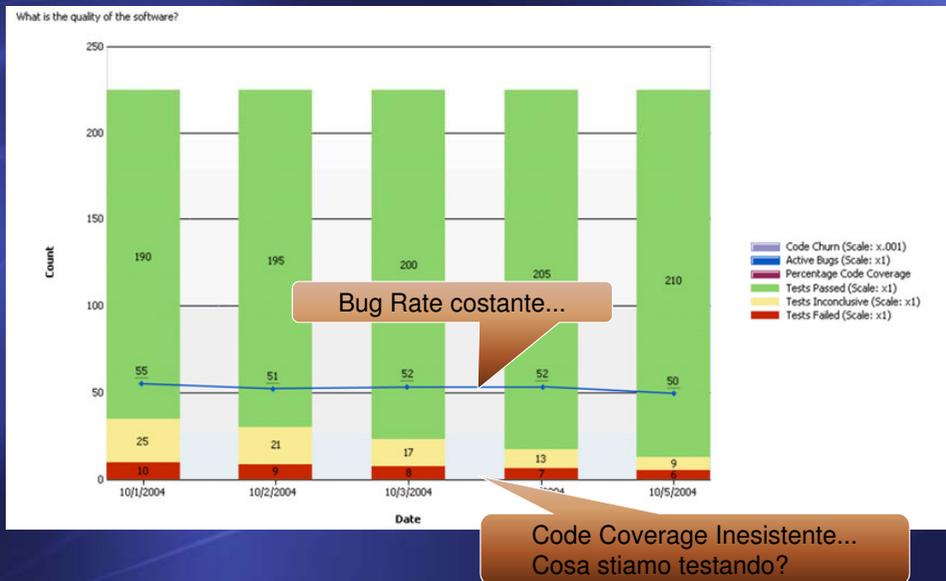
Gestione inadeguata dei bug

How much work is left and when will it be done?



Il numero di WorkItem continua a crescere... Bug o sindrome delle "macerie nascoste"?

I test passano... Ma la soluzione non funziona



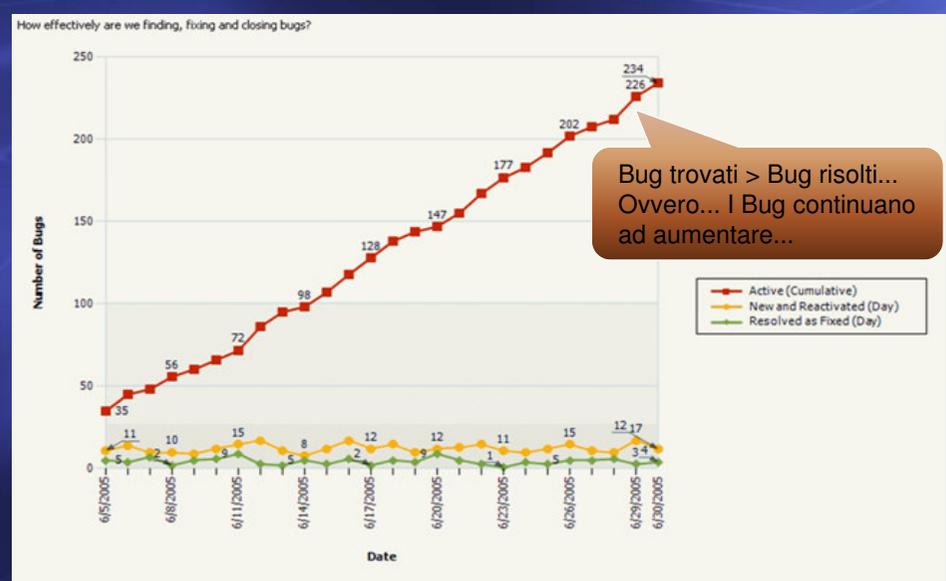
Andamento "inadeguato" delle Build

What are the details for the builds?

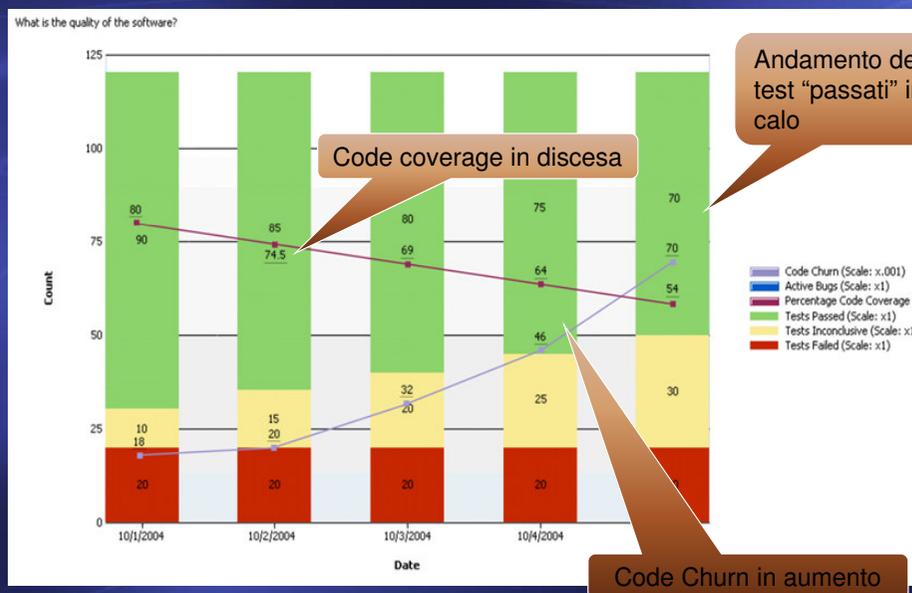
Build Name	Flavor	Platform	Build Quality	% Tests Passed	% Code Coverage	Code Churn
Debugx86_20060222.5	Debug	x86	Unexamined	88.889 %	0.699 %	573
Debugx86_20060222.4	Debug	x86	Unexamined	94.444 %	0.5 %	No Data
Debugx86_20060222.3	Debug	x86	Unexamined	94.737 %	0.599 %	148
Debugx86_20060222.2	Debug	x86	Unexamined	100 %	0.3 %	260
Debugx86_20060222.1	Debug	x86	Unexamined	100 %	0.1 %	398
Debugx86_20060221.21	Debug	x86	Unexamined	100 %	0.2 %	123
Debugx86_20060221.20	Debug	x86	Unexamined	100 %	No Data	120
Debugx86_20060221.19	Debug	x86	Unexamined	98.667 %	6.893 %	318
Debugx86_20060221.18	Debug	x86	Unexamined	100 %	4.196 %	401
Debugx86_20060221.17	Debug	x86	Unexamined	98.182 %	4.396 %	1245
Debugx86_20060221.16	Debug	x86	Unexamined	96.364 %	3.896 %	3484
Debugx86_20060221.15	Debug	x86	Unexamined	100 %	2.597 %	1264
Debugx86_20060221.14	Debug	x86	Unexamined	97.619 %	2.198 %	2194

Il 100% dei test passano...
Ma quanto stanno testando?

I test passano... Ma la soluzione non funziona



Unit Test inadeguati



Cosa fare se i membri del Team non collaborano... Usiamo le Policy!

I test passano... Ma la soluzione non funziona

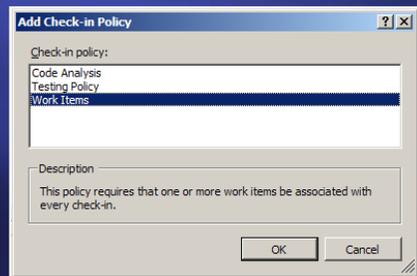
Were there bugs found without corresponding tests?

ID	State	Priority	Title
56	Active	3	Payment web service doesn't return on Visual Basic.NET clients
80	Active	3	All DateTime fields should be ReadOnly by default
81	Active	3	Close timeout when checking in add of 4947 and 164 dirs
82	Active	3	Update unshelve to use pend methods to resolve issues correctly
83	Active	3	MS Project... data/cannot sync
84	Proposed	3	Build failure in build: Nightly

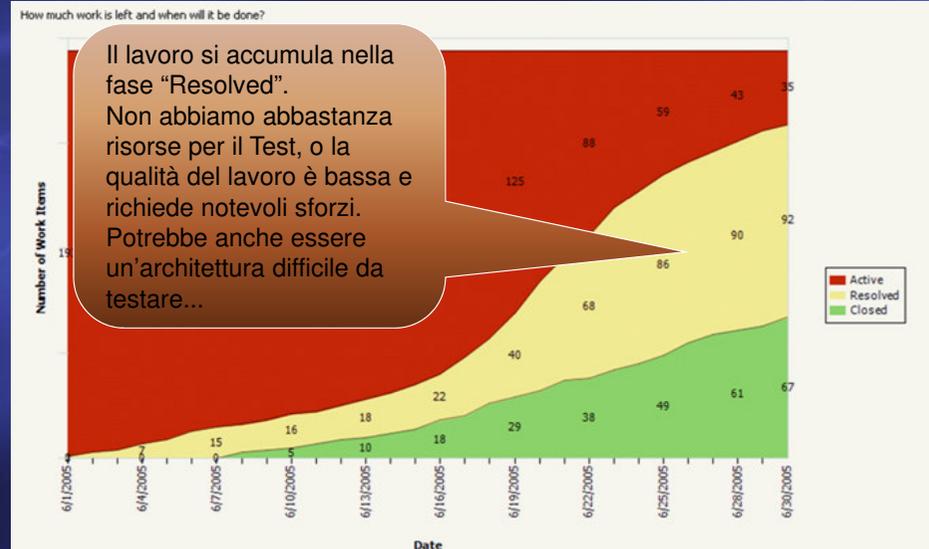
I Test automatizzati non rilevano questi bug, che non hanno test associati... Risultato... Bisogna testarli a mano!

Politiche di CheckIn

- E' possibile definire politiche di CheckIn che richiedano ad esempio di aver passato tutti gli Unit Test e che il Code Coverage sia superiore ad una certa percentuale.
- Oppure che la Code Analysis non abbia dato errore.
- Chi esegue il CheckIn può forzare la policy, ma deve motivare la scelta, motivazione che viene registrata.
- Il Project Manager può verificare la motivazione, per controllare che non ci siano leggerezze.



La soluzione è bloccata nella fase di test...



Come otteniamo i dati delle Build?

Build centralizzate e Continuous Integration

CheckIn Notes

- E' possibile creare dei campi per delle note.
- Le note sono simili ai commenti, ma sono più specifiche.
- Se una nota è obbligatoria, non può essere "saltata", a differenza delle CheckIn Policy

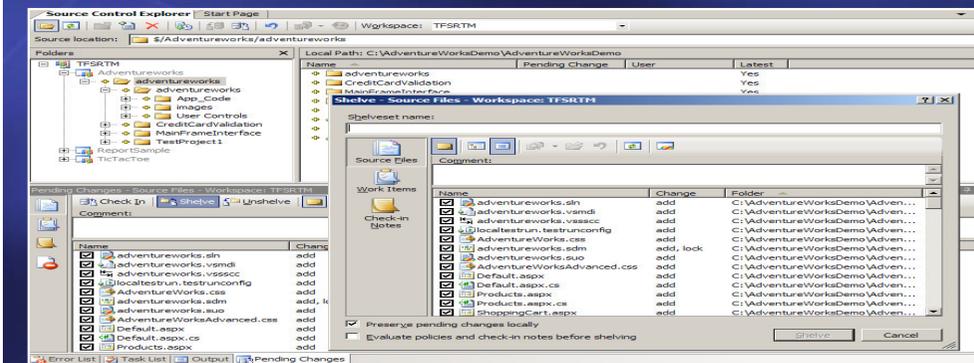


Build centralizzata

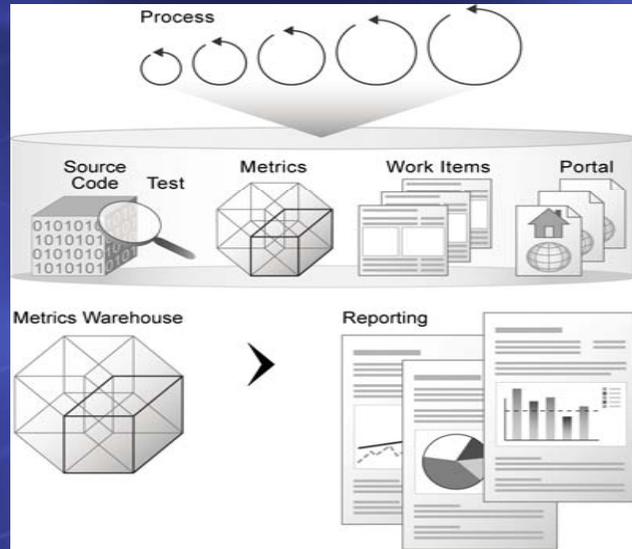
- Avere una build centralizzata è un'ottimo modo per sapere sempre in tempo reale come sta andando il progetto.
- Team Foundation Server dispone di un servizio di Team Build, che permette di gestire Build centralizzate su richiesta, su una o più macchine.
- Le Build possono includere anche l'esecuzione di Test e del Code Coverage per determinare la qualità della Build stessa.
- Convienè schedare una Build centralizzata ogni giorno (Daily Build) per verificare che le nuove modifiche si integrino con quelle esistenti senza problemi di compilazione.

Shelving – salvare i file sul server temporaneamente

- Lo "shelveset" è un'area sul server in cui mettere temporaneamente i file
- I file nello "shelveset" possono essere visibili solo all'utente che li ha salvati, o possono essere visibili da tutti
 - E' molto utile ad esempio in caso di code-review prima di approvare i check-in



La trasparenza è il valore fondamentale di VSTS



Continuous Integration

- La pratica della Continuous Integration è l'estremizzazione del concetto della Daily Build o della Build schedulata.
- La Continuous Integration richiede un servizio che rimanga in ascolto sull'evento di Check In e che faccia partire una nuova Build.
- In questo modo eventuali errori di integrazione verranno subito identificati.
- Sia in caso di Daily Build, sia in caso di Continuous Integration è utile inviare mail ai membri del team in caso di errore.

Basta poco per avere...
Trasparenza Totale