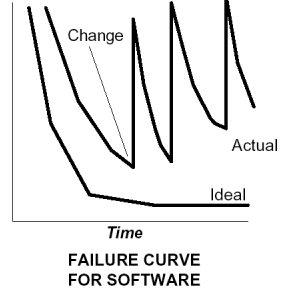
**Progettazione agile**

Il software diventa fragile quando deve essere modificato



Il compito del ing. del software è quello di gestire i cambiamenti in modo costanti che si avvicinino alla curva ideale, il cambiamento sono i picchi inferiori, bisogna mantenerli vicini alla curva ideale.

**Agile Design**

Le motivazioni verso queste modalità sono:

* Alto numero di progetti interrotti
* Alto rischio per progetti grossi
* Non rispetto delle consegne
* Costi di pianificazioni elevati
* Difficoltà e rischi in fase di manutenzione
* Problemi di qualità
* Difficoltà nel coinvolgimento dei clienti durante la fase di pianificazione
* Cambi di contesto
* Natura del software

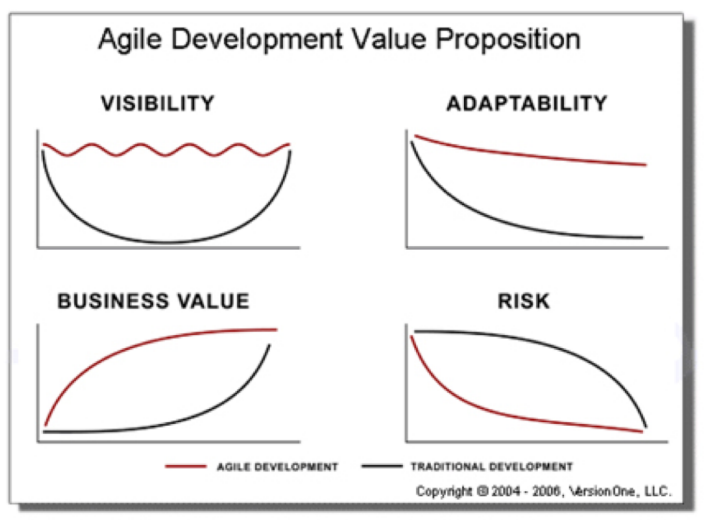
Le fasi di pianificazione ed esecuzione sono integrate insieme, non ci sono fasi a compartimenti stagni

Il coinvolgimento del committente è indispensabile per il buon andamento del progetto, per poter dare delle priorità.

Sviluppo in piccole iterazioni, che aggiungono poca funzionalità alla volta consentendo una crescita progressiva del software.

Il testing automatico DEVE essere presente in ogni software.

La progettazione agile non è infallibile, ma è la migliore rispetto ad altri metodi.



La semplicità nelle soluzioni comporta una più facile modifica

La comunicazione è fondamentale per affrontare i problemi, tramite l’integrazione continua si può comunicare si ci sono delle problematiche.

Feedback su tutto il sistema (unità di test e integrazione continua)

Feedback sull’utilizzabilità del sistema

Feedback sull’andamento del progetto

Tramite il test automatico si possono fare modifiche senza rischiare di rompere tutto.

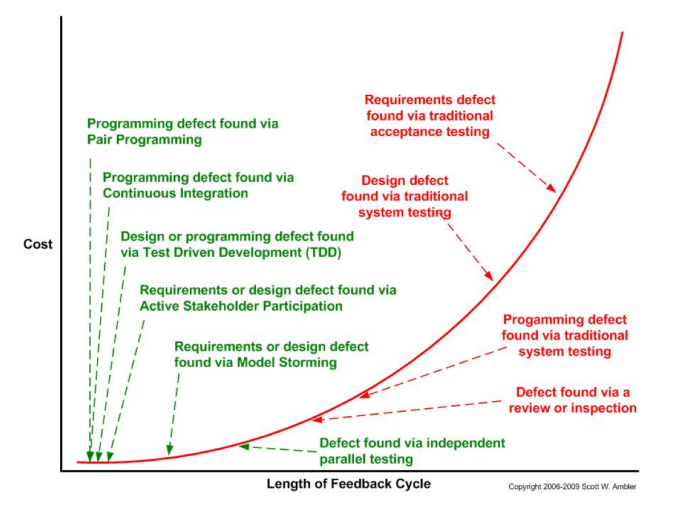


Immagine che contiene testo

Descrizione generata automaticamente

Manifesto per lo sviluppo agile del SW

1. La nostra massima priorità è soddisfare il cliente rilasciando software di valore, fin da subito e in maniera continua.
2. Accogliamo i cambiamenti nei requisiti, anche a stadi avanzati dello sviluppo. I processi agili sfruttano il cambiamento a favore del vantaggio competitivo del cliente.
3. Rilasciamo software funzionante in modo frequente, con iterazioni variabili da un paio di settimane a un paio di mesi, preferendo i periodi brevi.
4. Committenti e sviluppatori devono lavorare insieme per tutta la durata del progetto.
5. Vogliamo basare i nostri progetti su individui motivati. Diamo loro l'ambiente e il supporto di cui hanno bisogno e confidiamo nella loro capacità di portare a termine con successo il lavoro
6. La conversazione diretta è il modo più efficiente e più efficace per comunicare con il team e all'interno del team
7. Il software funzionante è il principale metro di misura dello sviluppo
8. I processi agili promuovono uno sviluppo sostenibile. Gli sponsor, gli sviluppatori e gli utenti dovrebbero essere in grado di mantenere indefinitamente un ritmo costante
9. La continua attenzione all'eccellenza tecnica e al buon design esaltano l'agilità
10. La semplicità è essenziale
11. Le architetture, i requisiti e la progettazione migliori emergono da team che si auto-organizzano
12. A intervalli regolari il team riflette su come diventare più efficace, dopodiché regola e adatta il proprio comportamento di conseguenza

**Debito tecnico**

La fretta nello sviluppo di nuove funzionalità porta a dimenticarsi di inserire nelle stime dei costi quello della manutenzione “ordinaria”, che andrebbe fatta dopo ogni iterazione.

Se non si è abituati a ripulire continuamente il codice si incorre presto nel problema chiamato “debito tecnico”

