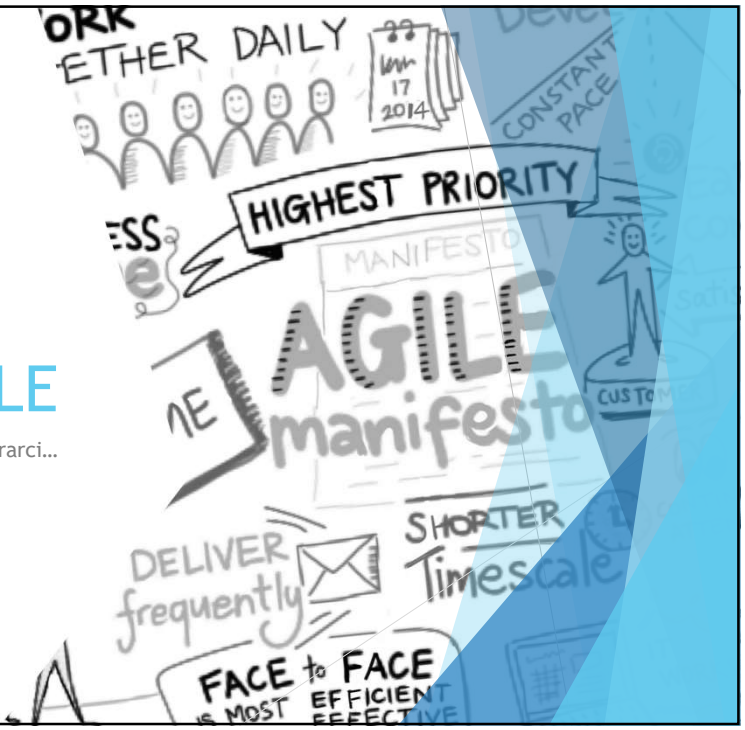
Il mondo AGILE

Metodologie per entrarci…



Cosa significa «Agile»?

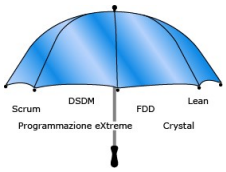
Definizioni enciclopedia *Treccani*

1. «Che si muove con facilità»
2. «Facile ad essere usato, maneggevole»

Nel Project Management
ci si riferisce all'**APM** (Agile Project Management)

Cosa è l'APM (Agile Project Management)

- E' un **approccio al Project Mgnt** (= metodologie + tool) che mirano al rilascio estremamente rapido del software
- Permette di avere subito **valore per il cliente**
- E' un insieme di **linee guida**
- Si utilizzano team di progetto **piccoli** e **autorganizzati** che mantengono contatti giornalieri con il cliente e con gli altri stakeholder
- Agile è l'ombrello, le metodologie sono le implementazioni

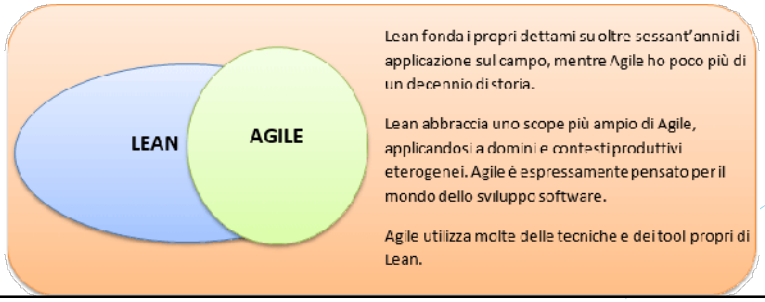


- Scrum
- XP
- Crystal
- DSDM (Dynamic Systems Development Model)
- FDD (Feature Driven Development)

erano le metodologie previste in origine (quelle incluse nell'Agile Manifesto (vedi dopo))

Agile vs Lean

- Spesso l'approccio **Agile** è confuso con le metodologie **Lean**
- Sono entrambe **filosofie / correnti di pensiero**
- Agile e Lean sono strettamente legati
  - Hanno il comune obiettivo di incrementare il Valore e la Qualità della soluzione realizzata, abbattendone il tempo ed i costi di realizzazione



Agile vs Lean

AGILE	LEAN
Finalizzato ad un'esecuzione rapida delle attività, adattandosi facilmente ai cambiamenti	Finalizzato ad un'esecuzione intelligente, si migliora praticamente tutto ciò che si fa eliminando tutto ciò che non porta valore al cliente
Rende flessibile il processo di sviluppo	Rende sostenibile il processo di sviluppo
Inizialmente utilizzato per lo sviluppo software, poi esteso al marketing, attualmente applicato in altri settori	Inizialmente utilizzato nella produzione tradizionale, ad oggi ampliato ed esteso a tutti i settori esistenti
Action loop: product backlog, sprint backlog, iterazioni (sprint), risultati potenzialmente rilasciabili al cliente	Action loop: costruire, misurare, migliorare
Metodo per dimostrare i progressi: dimostrazione di fatto, direttamente con i prodotti delle iterazioni	Metodo per dimostrare i progressi: convalida dell'apprendimento
Metodologie: SCRUM, XP, FDD, DSDM, Crystal Methods ecc.	Metodologie: Kanban, Kaizen, ecc.
Strumenti (toolkit): sprint, board, Scrum Master, acceptance test, user story mapping ecc.	Strumenti (toolkit): split (A/B) test, customer interview, funnel and cohort analysis, Customer Success Manager, ecc.

Ambiti delle metodologie Agile

Ci sono framework agili per

A. **Prodotti** (es: SCRUM)
B. **Progetti** (es: AgilePM)
C. **Programmi** (es: SAFe – Scaled Agile Framework)

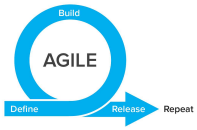
In ogni caso…

Nasce x snellire burocrazia

del Project Mgnt tradizionale

APM:
Caratteristiche fondamentali

1. Brevi iterazioni di sviluppo (*sprint*)



2. Lavoro a stretto contatto tra gli sviluppatori e stakeholder

3. Regolare ridefinizione delle priorità di lavoro

4. Approccio rapido e flessibile nell'indirizzare i cambiamenti di ambito

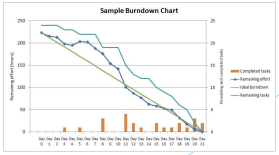
APM:
Tecniche di supporto

- Per la pianificazione: i **timebox**

- 

Per l'assegnazione di priorità: il metodo **MoSCoW** (vedi dopo)

- Per la reportizzazione del progresso: il **burndown charts**



- …

Metodo MoSCoW

- Tecnica impiegata per raggiungere con gli stakeholder un concetto condiviso circa l'importanza (priorità) che essi attribuiscono al raggiungimento di ciascun requisito
- Consiste nella categorizzazione dei requisiti secondo la tabella seguente:

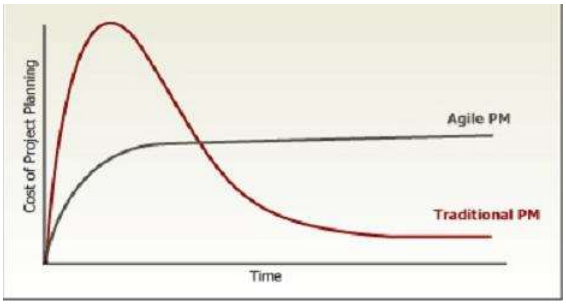
Lettera	Significato (inglese)	Descrizione
M	MUST ("Deve")	Descrive un requisito che **deve essere soddisfatto** nella soluzione finale, affinché essa sia considerata un successo.
S	SHOULD ("Dovrebbe")	Rappresenta un aspetto di **alta priorità** che – nei limiti del possibile – dovrebbe essere compreso nella soluzione. Spesso si tratta di un requisito critico che però può essere soddisfatto altrimenti, se strettamente necessario.
C	COULD ("Potrebbe")	Descrive un requisito che è considerato **auspicabile** ma non necessario. Sarà incluso se il tempo e le risorse lo permettono.
W	WON'T ("Non [avrà]")	Rappresenta un requisito che gli *stakeholders* hanno accettato di non veder implementato in una data release, ma che può essere considerato in futuro. Nota: a volte la parola *Would* ("sarebbe/avrebbe") sostituisce la locuzione *Won't* per dare un'idea più chiara di questa scelta.

Traditional PM vs. APM: triplo vincolo

Un modo molto semplice per capire la differenza tra Agile e forme di sviluppo più tradizionali è identificare il loro rapporto con il **triplo vincolo** (ambito, tempi, costi).

Traditional PM vs. APM: costi

Costo della pianificazione



Costo dei cambiamenti

APM: Un po' di storia

- I metodi agili (Agile **o Lightweight methodologies**) si affermano a metà degli anni '90 in ambito IT
- A valle del moltiplicarsi di modelli agili, nel 2001 17 degli ideatori di tali metodi hanno formulato un

Manifesto per lo sviluppo agile del software (Agile Manifesto)

- è il frutto di anni di sperimentazione di nuovi metodi e pratiche elaborati nel corso degli anni '90

Kent Beck	Jim Highsmith	Ken Schwaber
Mike Beedle	Andrew Hunt	Jeff Sutherland
Arie van Bennekum	Ron Jeffries	Dave Thomas
Alistair Cockburn	Jon Kern	
Ward Cunningham	Brian Marick	
Martin Fowler	Robert C. Martin	
James Grenning	Steve Mellor	

Urbino, 17-18/11/2017
iad
Italian Agile Day 2017
http://www.agileday.it

Torino, 03/02/2018
TORINO AGILE CONFERENCE
AGILE, COME RISPONDERE AL CAMBIAMENTO?
http://www.agileday.it/mini/2018/torino

APM: Un caso di successo

David Gram

- *Senior Innovation Director* del gruppo **Lego** fino al marzo 2017
- Esperto in **radical innovation**
- Riconosciuto come un **change-agent** particolarmente influente
- Ha aiutato l'azienda danese a superare una crisi finanziaria importante (nel 2003 era quasi fallita)
- Gram era a capo del Future Lab…
- … ha stabilito **cicli brevi di sviluppo** con forte interazione con il cliente finale…
- … quindi sosteneva fosse importante coinvolgere gli utenti (**user driven innovation**)

Fonte:
http://www.ilsole24ore.com/art/management/2017-06-29/sviluppo-agile-e-design-thinking-modello-lego-funziona-cosi-113232.shtml?uuid=AEqUJOoB

Agile Manifesto: i 4 **valori** fondamentali

1. **Gli individui e le interazioni** più che i processi e gli strumenti
2. **Il software funzionante** più che la documentazione esaustiva
3. **La collaborazione col cliente** più che la negoziazione dei contratti
4. **Rispondere al cambiamento** più che seguire un piano

Fermo restando il valore delle voci a destra, sono più importanti le voci a sinistra!

Agile Manifesto: i 12 princìpi

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. **Consegniamo frequentemente** software funzionante, con cadenza variabile da un paio di settimane a un paio di mesi, preferendo i periodi brevi.
4. Committenti e sviluppatori devono **lavorare insieme** quotidianamente per tutta la durata del progetto.
5. Fondiamo i progetti su **individui motivati**. Diamo loro l'ambiente e il supporto di cui hanno bisogno e confidiamo nella loro capacità di portare il lavoro a termine.
6. Una **conversazione faccia a faccia** è il modo più efficiente e più efficace per comunicare con il team ed all'interno del team.

Agile Manifesto: i 12 princìpi

7. Il **software funzionante** è il principale metro di misura di progresso.
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 alla **buona progettazione** esaltano l'agilità.
10. La **semplicità** - l'arte di massimizzare la quantità di lavoro non svolto - 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.

Punti di forza della metodologia Agile

1. l'**avvio** dell'implementazione è **rapido** e lo **sviluppo** è **incrementale**
2. i **requisiti possono evolvere** in corso d'opera
3. la **risposta** ad esigenze di cambiamento è **rapida**
4. **frequenti** momenti di **test** e di revisione dei requisiti
5. **collaborazione attiva** nello sviluppo tra fornitore e committente

Punti di debolezza della metodologia Agile

1. in assenza di pianificazione e documentazione del **lavoro** da svolgere, questo **può essere frainteso** o procedere in modo indisciplinato con conseguente rework
2. richiede **personale** del cliente molto **qualificato**
3. il **tempo richiesto al cliente** per il suo coinvolgimento è **elevato**
4. l'orizzonte è concentrato sul breve **termine**
   (c'è quindi il rischio che si perda la prospettiva di lungo periodo)
5. la **documentazione** prodotta è **poco dettagliata**
   (questo può creare problemi di utilizzo da parte dell'utente o di governo del progetto da parte del PM)

Aggile Giggi

Aggile Giggi
@agilegigi

So' programmatore con due cojoni cosí.

Rome, Lazio
Iscritto a settembre 2015
Nato nel 1969

Cercate su Twitter @agilegigi
e
diventate un suo follower

Aggile Giggi @agilegigi ·6 ott
Se ce guardi bbene controluce 'sta build nun è proprio rossa rossa, diciamo arancione scuro, deploya e annamosenaccasa. #poilunedíccepensamo

Aggile Giggi @agilegigi ·29 set
E vabbe', mo' nun è che proprio li deploy devono compila' tutti tutti pefforza. #poilunedíccepensamo

La base dell'APM: KISS principle

(Mantienilo semplice facendolo sembrare stupido)

Altre traduzioni:
- *Keep it sweet and simple*
- *Keep it short and simple*
- *Keep it simply smart*

- Aiuta a concepire le cose mantenendole a uno stadio di semplicità eliminando gli elementi non necessari e facilitando l'esposizione e la comprensione
- Aiuta a semplificare concetti complessi, integrando management e tecnica in un processo chiaro e lineare alla portata di tutti

KISS: origini

KEEP IT SIMPLE STUPID

Richiama in parte il Principio filosofico del **Rasoio di Occam**

A parità di fattori la spiegazione più semplice è da preferire

- *Novacula Occami* (in latino)
- "Principio metodologico espresso nel XIV secolo dal filosofo e frate francescano inglese **William of Ockham**, noto in italiano come Guglielmo di Occam. Tale principio, ritenuto alla base del pensiero scientifico moderno, nella sua forma più immediata suggerisce **l'inutilità di formulare più ipotesi di quelle che siano necessarie** per spiegare un dato fenomeno quando quelle iniziali siano sufficienti." (fonte: Wikipedia)
- La metafora del rasoio concretizza l'idea che sia opportuno, dal punto di vista metodologico, eliminare con tagli di lama e mediante approssimazioni successive le ipotesi più complicate.

KISS: how to

KEEP IT SIMPLE STUPID

Per applicare il principio KISS
è necessario introdurre nelle nostre attività
dei **momenti di pausa**
che ci permettano di:

- osservare con un certo distacco quanto stiamo realizzando
- verificare quanto i risultati siano in linea con le nostre aspettative

- In riferimento al codice sorgente di un programma significa non occuparsi delle ottimizzazioni fin dall'inizio, ma cercare di mantenere uno stile di programmazione semplice e lineare, demandando le ottimizzazioni al compilatore o a successive fasi dello sviluppo.