

Modelli di processo di sviluppo di software “moderni”

Alberto Gianoli

Dove?

- ❖ Pressmann, cap. 3, parte cap. 4 (xp)
- ❖ Sommerville, cap. 4, parte cap. 17 (xp)
- ❖ Se vi interessa scrum: Ken Schwaber, Jeff Sutherland, La Guida a Scrum.
La Guida Definitiva a Scrum: Le Regole del Gioco (PDF), scrumguides.org

Sviluppo Agile: perché?

- ✧ problemi complessi richiedono contributo di molte persone
- ✧ soluzione inizialmente non chiara
- ✧ requisiti che molto probabilmente cambieranno
- ✧ posso pensare di sviluppare il programma in “incrementi”
- ✧ è necessaria una collaborazione stretta e un feedback rapido con gli stakeholders

Quasi tutti i problemi difficili nello sviluppo sw riguardano le interazioni di persone, non i computer: sviluppare sw è una attività sociale. Chi lo sa fare bene ha una certa abilità nelle relazioni interpersonali.

Metodologie di sviluppo Agile

I principi su cui si basa una metodologia leggera che segue i punti indicati dall'Agile Manifesto sono quattro:

- ❖ le persone e le interazioni sono più importanti dei processi e degli strumenti (ossia le relazioni e la comunicazione tra gli attori di un progetto software sono la migliore risorsa del progetto)
- ❖ è più importante avere software funzionante che documentazione (bisogna rilasciare nuove versioni del software ad intervalli frequenti, e bisogna mantenere il codice semplice e avanzato tecnicamente, riducendo la documentazione al minimo indispensabile)
- ❖ bisogna collaborare con i clienti al di là del contratto (la collaborazione diretta offre risultati migliori dei rapporti contrattuali)
- ❖ bisogna essere pronti a rispondere ai cambiamenti più che aderire al progetto (gli sviluppatori dovrebbero essere autorizzati a suggerire modifiche in ogni momento)

Metodologie di sviluppo Agile

Tutti i metodi agili hanno in comune:

- ❖ **rilasci frequenti del prodotto sviluppato**
- ❖ **collaborazione continua del team di sviluppo con il cliente**
- ❖ **documentazione di sviluppo ridotta**
- ❖ **continua e sistematica valutazione dei valori e rischi dei cambiamenti**

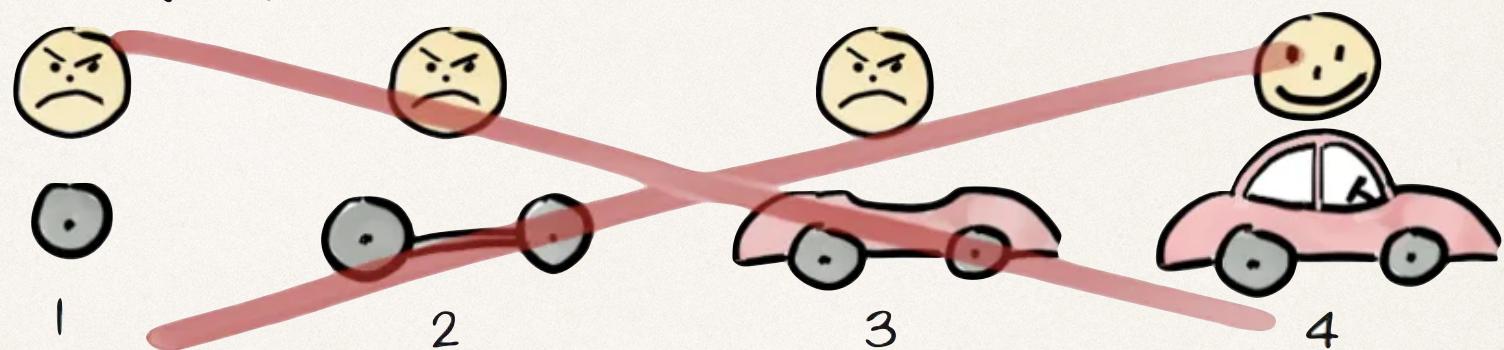
- ❖ Metodi agili: Extreme Programming (XP), Scrum, Feature-Driven Development (FDD), Adaptive Software Process, Crystal Light Methodologies, Dynamic System Development Method (DSDM), Lean Development, ...

Minimal Viable Product (MVP)

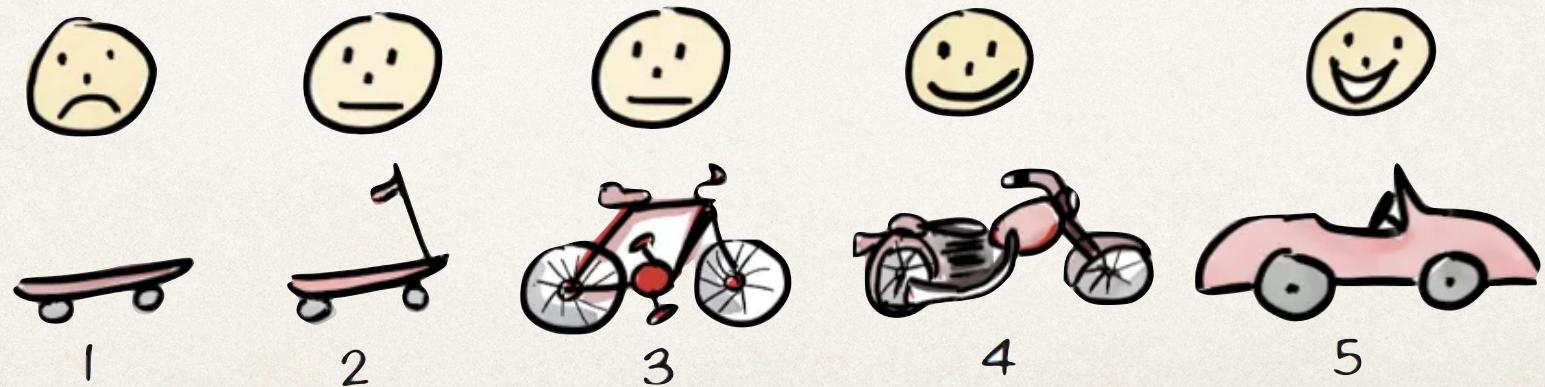
- ❖ Prodotto Minimo Funzionante (minimal viable product): è il prodotto con i più alto ritorno sugli investimenti rispetto al rischio
- ❖ È una strategia che mira a evitare di costruire prodotti che i clienti non vogliono: cerco di massimizzare le informazioni apprese dal cliente per ogni euro speso
- ❖ MVP quindi non è un prodotto minimo, ma un processo iterativo di generazione di idee, prototipazione, presentazione, raccolta dati, analisi e apprendimento
- ❖ MVP è alla base del movimento “Lean”, che è alla base di Agile

Minimal Viable Product (MVP)

Not like this....



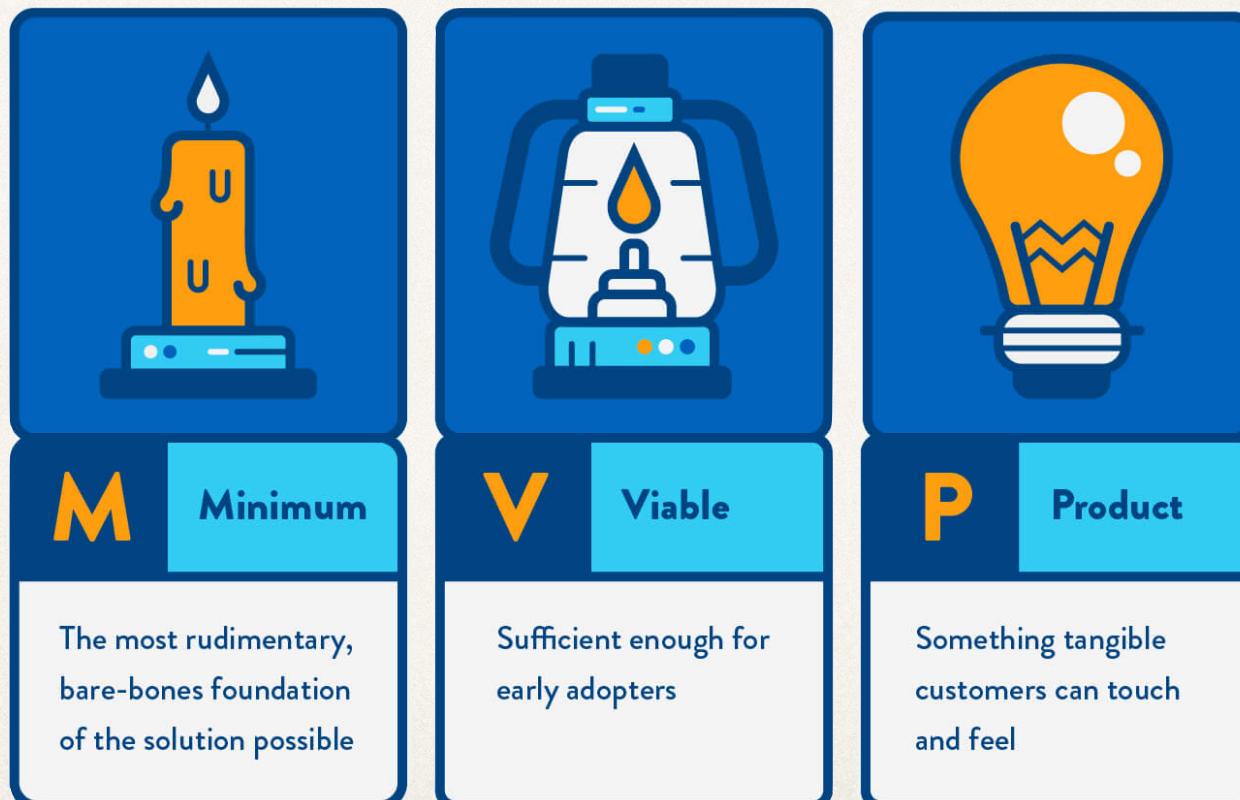
Like this!



by Henrik Kniberg

Minimal Viable Product (MVP)

WHAT IS A MINIMUM VIABLE PRODUCT



Programmazione estrema (XP): move light and move fast

- ✿ Approccio recente (1999)
 - ✿ il codice è il prodotto
 - ✿ il codice è la documentazione
 - ✿ codifica con un amico
 - ✿ scrivi prima i casi prova e prosegui con la codifica finché questi non sono stati superati
 - ✿ punta sempre al progetto più semplice per superare i test
 - ✿ inizia con il minimo numero di funzioni desiderate e migliora il risultato mano a mano che procede il lavoro

Team di sviluppo XP

- ❖ Di solito meno di 10 persone, riunite nello stesso locale
- ❖ Un rappresentante del cliente è sempre presente per rispondere a domande sui requisiti (e deve essere capace di / avere autorità per rispondere...)
- ❖ Il team usa comportamenti di sviluppo semplici e allo stesso tempo capaci di informare tutti sullo stato del progetto
- ❖ Il team è pronto ad adattare i comportamenti alla situazione specifica

XP core practice

- ❖ XP è definito dalle pratiche usate
 - ❖ Le pratiche variano nel tempo e a seconda del progetto in cui vengono utilizzate
1. Planning the game
 2. Simple design
 3. Pair programming
 4. Testing
 5. Refactor
 6. Short releases
 7. Coding standard

I requisiti sono “user stories”

- Sono usate al posto di documenti dettagliati di specifica dei requisiti
- Sono scritte dai clienti: cosa si aspettano dal programma
- Una storia è descritta da max un paio di frasi, scritte in linguaggio naturale, con la terminologia del cliente
- Utili per stimare i tempi e i costi di un rilascio

XP: planning the game

- ❖ Sviluppo dell'applicazione accompagnato dalla stesura di un piano di lavoro
- ❖ Piano definito e aggiornato a intervalli brevi e regolari dai responsabili del progetto, secondo le priorità aziendali e le stime dei programmati
- ❖ I programmati partecipano attivamente alla pianificazione
- ❖ La pianificazione coinvolge sia utenti responsabili del progetto sia sviluppatori per stabilire un equilibrio dinamico tra le esigenze di tutti
- ❖ Gli utenti finali presentano gli obiettivi descrivendo scenari (storie)
- ❖ Gli sviluppatori stimano il tempo necessario a realizzare di ogni storia
- ❖ Le storie vengono ordinate da utenti e responsabili secondo la loro priorità di realizzazione, dopo che ne è stata stimata la rispettiva difficoltà
- ❖ Dalla sintesi delle valutazioni i responsabili del progetto pianificano l'attività: l'insieme delle storie da realizzare per il prossimo rilascio e le date previste

Il planning Game si rigioca dopo ogni incremento

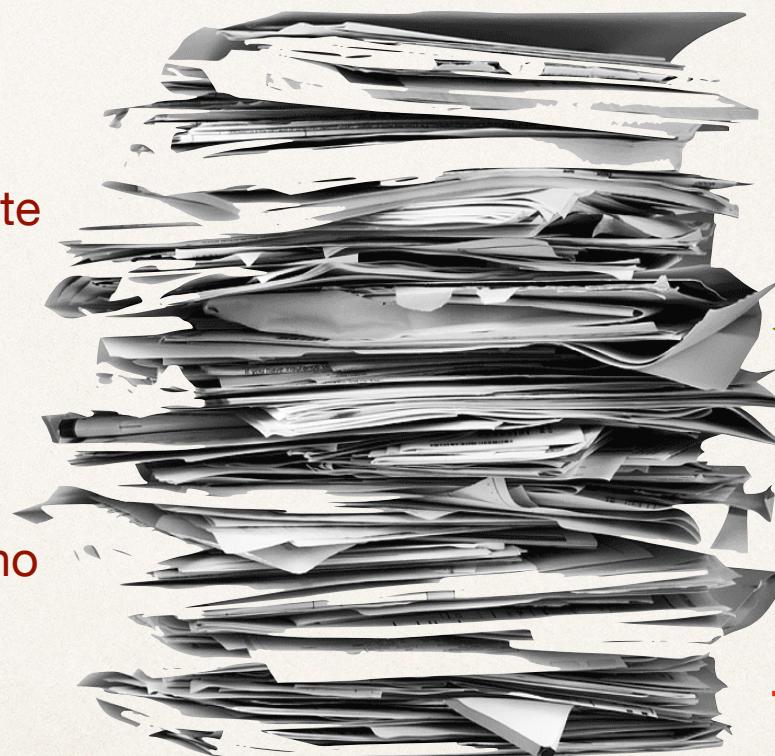
Storie ed interazioni

Alta priorità

Le storie ad alta prio
dovrebbero essere ben definite

Le storie a bassa prio possono
essere meno definite

Bassa priorità



Lista delle storie

} Ogni iterazione implementa
prima le storie a più alta prio

In ogni momento posso
aggiungere nuove storie

In ogni momento posso
ridefinire le prio

In ogni momento posso
eliminare storie

XP: simple design e pair programming

No Big Design Up Front
~~(BDUF)~~

- ❖ La struttura dell'applicazione deve essere la più semplice possibile
- ❖ L'architettura del sistema deve essere comprensibile da tutte le persone coinvolte nel progetto
- ❖ Non devono esserci parti superflue o duplicazioni
- ❖ Le parti che compongono il sistema devono essere solo quelle strettamente necessarie alle esigenze correnti
- ❖ Solo se richiesto da nuove circostanze si progettano nuovi componenti o si riprogettano quelli già esistenti
- ❖ La scrittura del codice è fatta da coppie di programmatore che lavorano al medesimo terminale
- ❖ Le coppie non sono fisse ma sono formate associando le migliori competenze per risolvere il problema specifico
- ❖ Il lavoro in coppia permette, scambiandosi periodicamente i ruoli, di mantenere mediamente un alto livello di attenzione
- ❖ Il locale di lavoro deve permettere senza difficoltà di lavorare a coppie

XP: testing

Scegli una user story e definisci i test prima del codice: Test Driven Development (TDD)

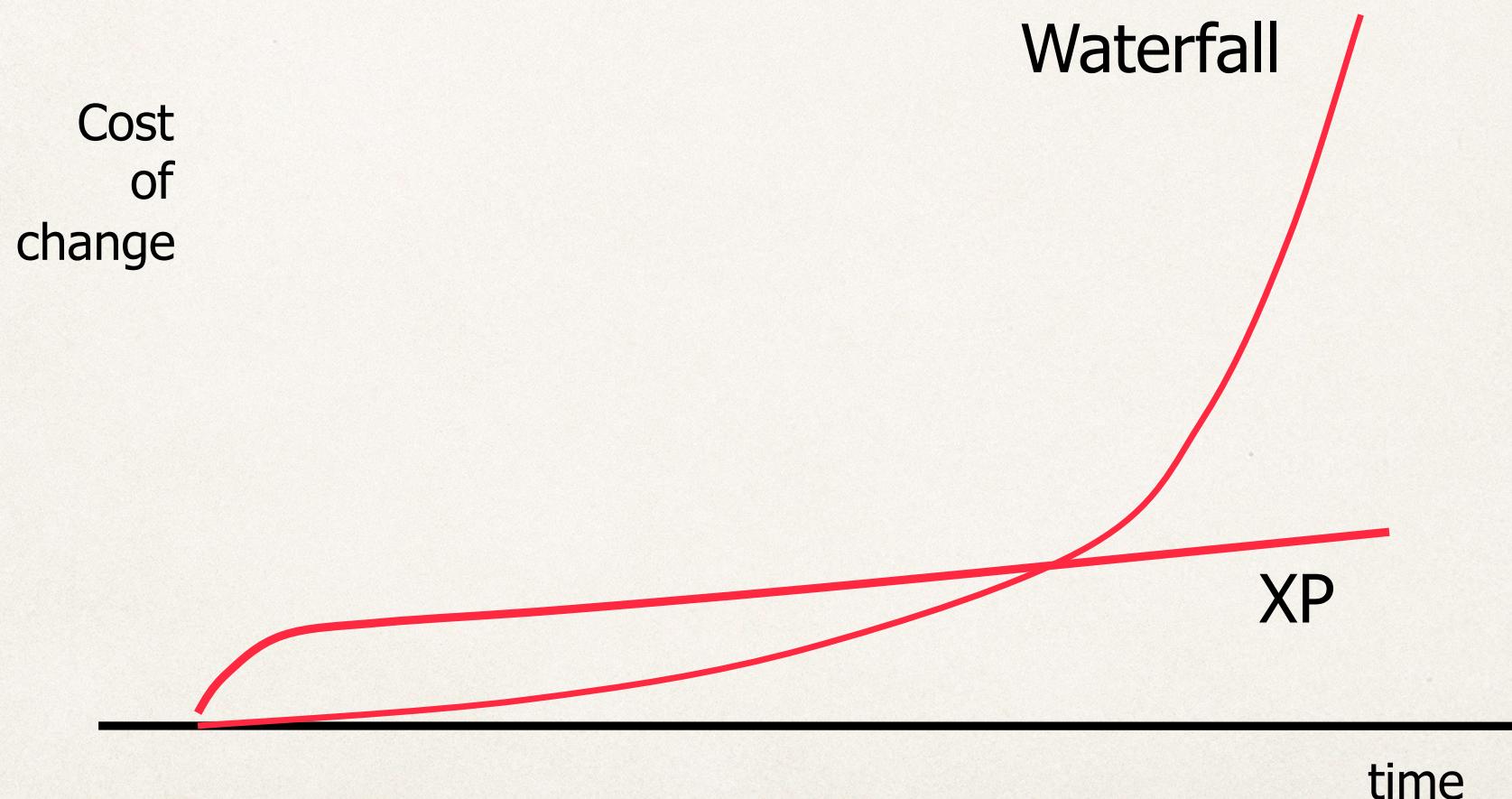
- ✿ Ogni funzionalità va sottoposta a verifica, in modo che si possa acquisire una ragionevole certezza della sua correttezza
- ✿ Test di sistema costruiti sulla base delle storie concordate con il committente
- ✿ Test di unità che devono poter essere rieseguiti automaticamente, con tempi dell'ordine dei minuti
- ✿ Ogni ristrutturazione o modifica del codice deve mantenere inalterato il risultato dei test già considerati
- ✿ I test vengono, solitamente, scritti prima della codifica della funzionalità

XP: refactor

Refactoring: migliorare il codice esistente senza cambiarne le funzionalità

- ❖ specie dopo molti cambiamenti nel tempo il codice diventa poco maneggevole
- ❖ i programmatore spesso continuano a utilizzare codice non più mantenibile perché continua a funzionare
- ❖ quando stiamo rimuovendo ridondanza, eliminiamo funzionalità non utilizzate e rinnoviamo un design obsoleto, stiamo rifattorizzando
- ❖ il refactoring mantiene il design semplice, evita inutili complessità, mantiene il codice pulito e conciso in modo che sia facilmente comprensibile, modificabile e estensibile

XP: costo delle modifiche



XP: la riunione in piedi

- ❖ Tutti i giorni: riunione di 15 min
 - ❖ in piedi (così dura meno) in cerchio
 - ❖ a turno ognuno risponde alle stesse 3 domande:
 - cosa hai fatto ieri?
 - cosa farai oggi?
 - quali ostacoli stai incontrando?
 - ❖ può essere il momento in cui si formano le coppie

XP: problemi tipici

- ❖ Il codice non viene testato completamente
- ❖ Il team produce pochi test utili
- ❖ Il cliente non partecipa a testare il sistema
- ❖ I test non funzionano prima dell'integrazione
- ❖ I test sono troppo lenti
- ❖ Le storie sono troppo complicate
- ❖ Il manager vuole la documentazione di sistema, o la specifica dei requisiti
- ❖ Il team è sovraccarico di compiti
- ❖ Cowboy coders: alcuni membri scelgono di ignorare la metodologia del team

SCRUM

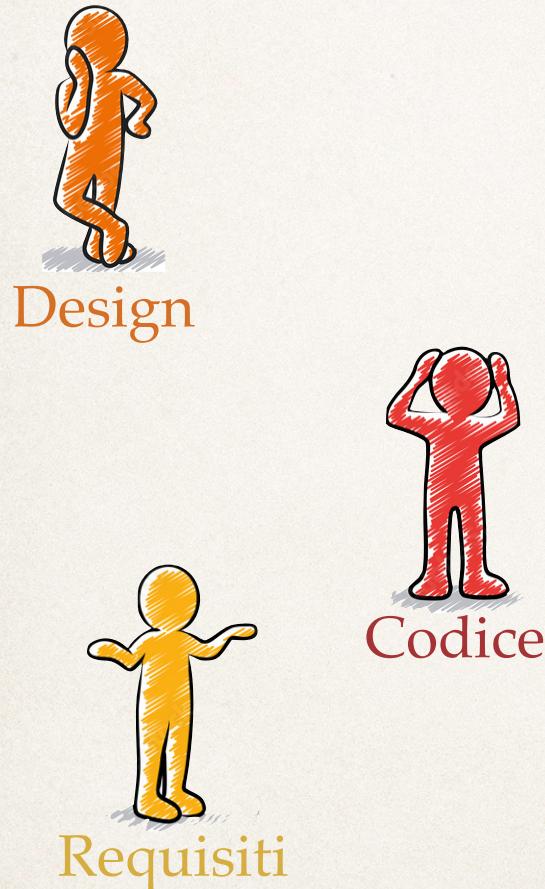
- ❖ Scrum: nel gioco del rugby è la mischia
- ❖ Scrum: nello sviluppo del sw è un processo in cui un insieme di persone si muove all'unisono per raggiungere un obiettivo predeterminato; l'obiettivo garantisce la soddisfazione delle ambizioni di squadra e personali
- ❖ E' un processo:
 - ❖ per gestire e controllare lo sviluppo del sw
 - ❖ iterativo, incrementale, si può applicare allo sviluppo e gestione di ogni tipologia di prodotto
 - ❖ fornisce alla fine di ogni iterazione un set di funzionalità potenzialmente rilasciabili

SCRUM: ma perché il rugby?

“The... ‘relay race’ approach to product development... may conflict with the goals of maximum speed and flexibility. Instead the holistic or ‘rugby’ approach - where a team tries to go the distance as a unit, passing the ball back and forth - may better serve today’s competitive requirements.”

H. Takeuchi and I. Nonaka,
“The New New Product Development Game”, *Harvard Business Review*, 1986

SCRUM: avanzare come squadra?

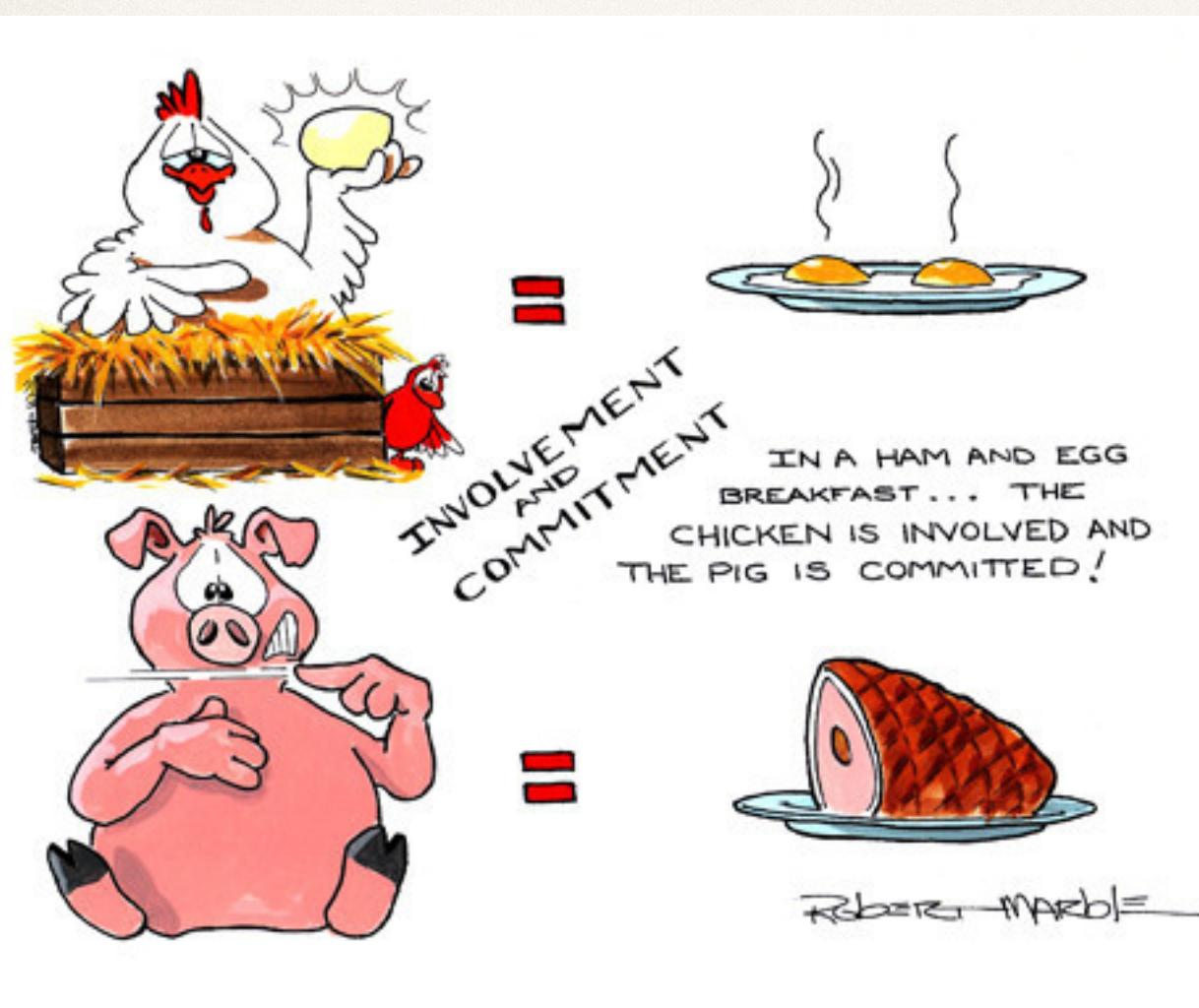


Non “una fase alla volta”:
il team scrum fa un po’ di tutto
durante uno sprint



H. Takeuchi and I. Nonaka, “The New New Product Development Game”, *Harvard Business Review*, 1986

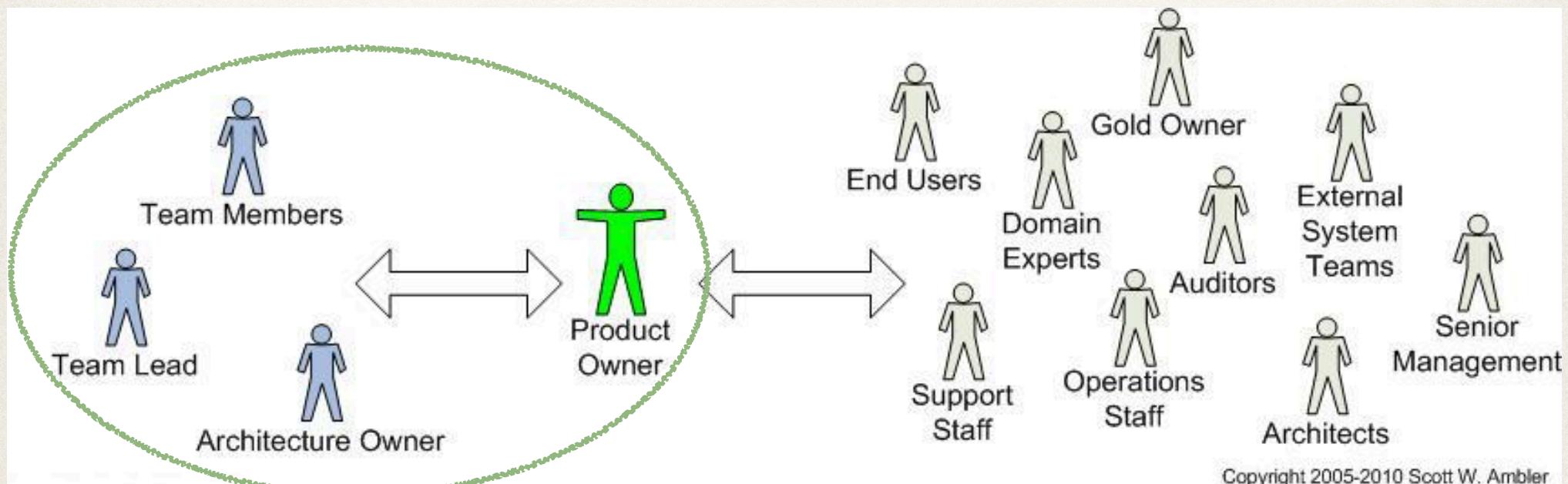
SCRUM: commitment...? Galline e maiali



Involved: to engage as a participant

Committed: bound or obligated, as under a pledge to a particular cause, action or attitude.

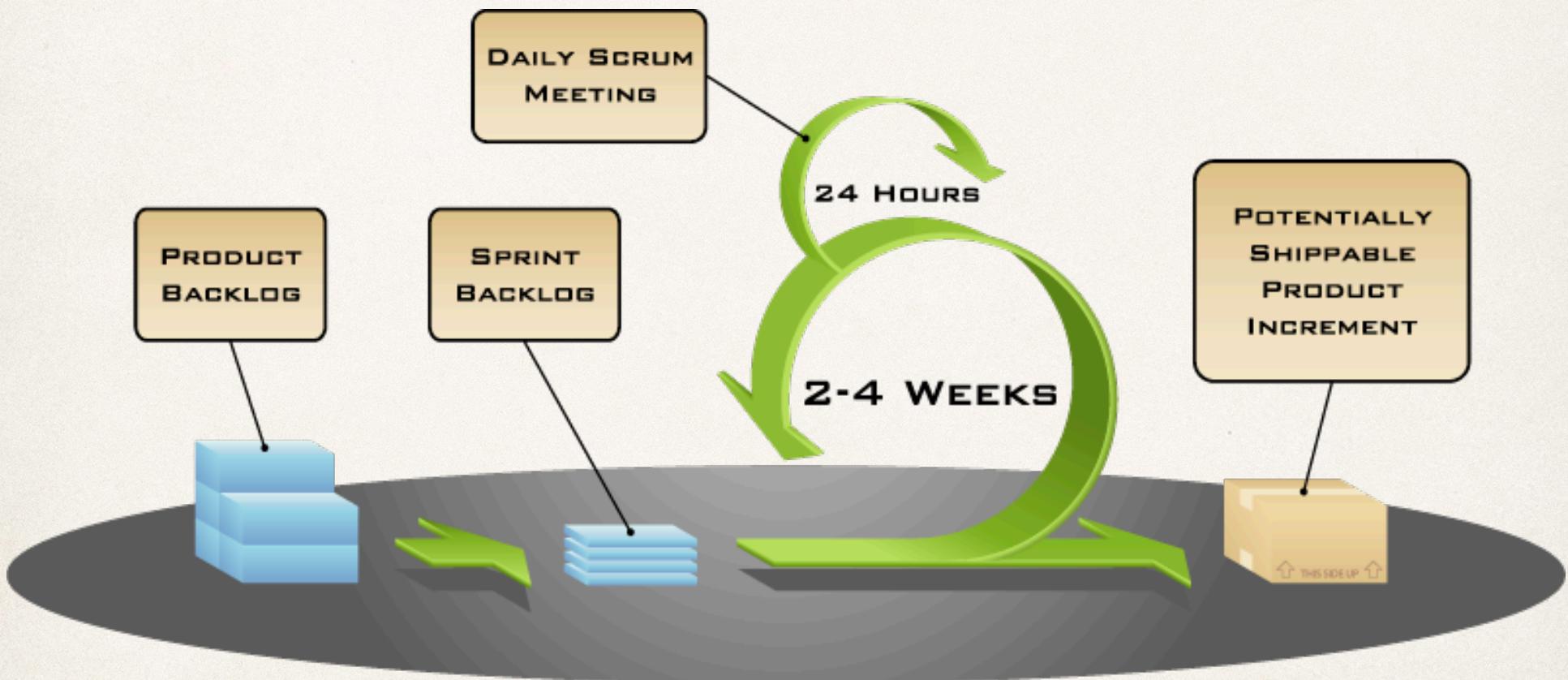
SCRUM: commitment...? Ruoli “pig” e ruoli “chicken”



Committed

Involved

SCRUM: il modello

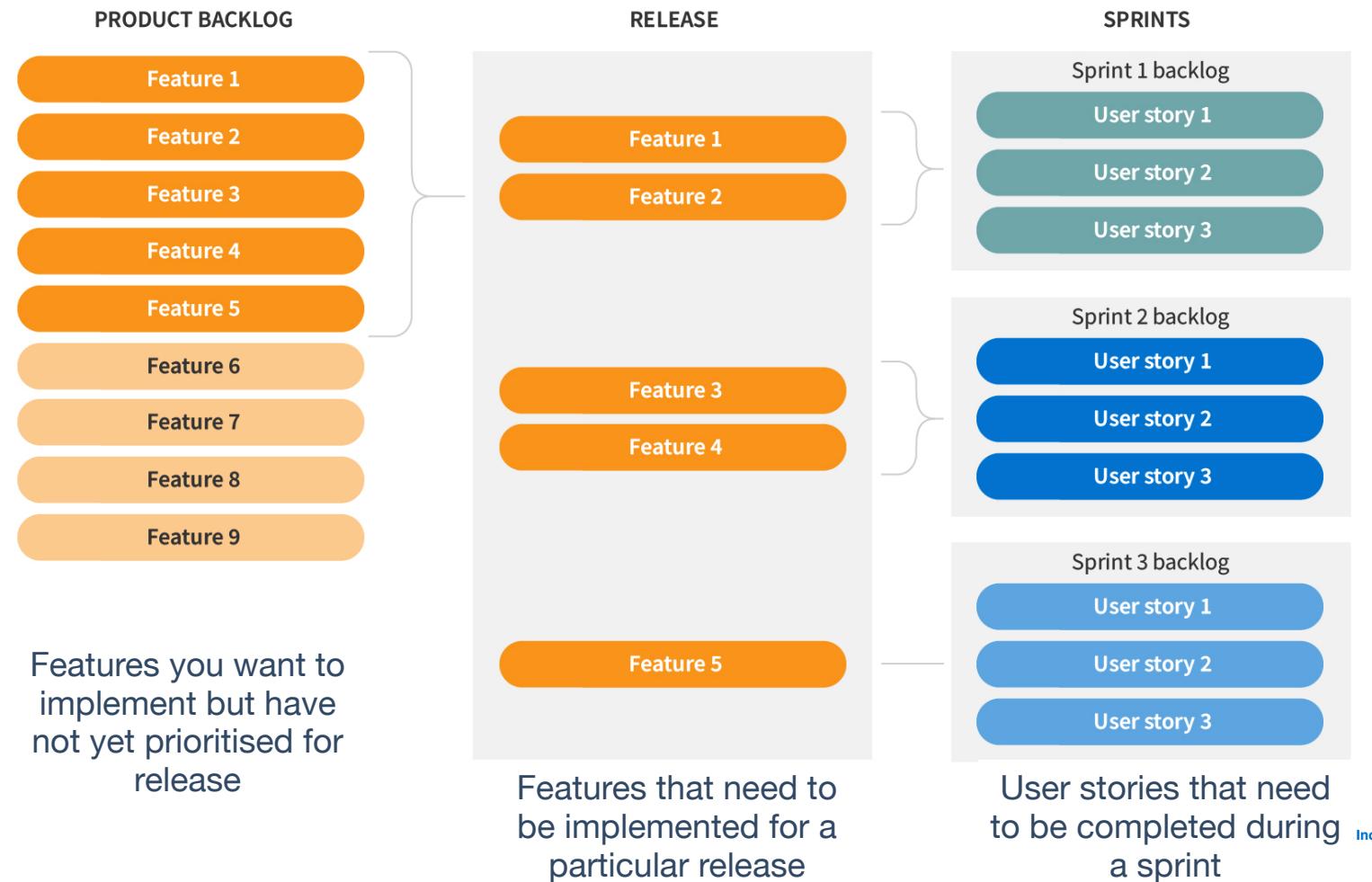


COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE

SCRUM: product, release, sprint...

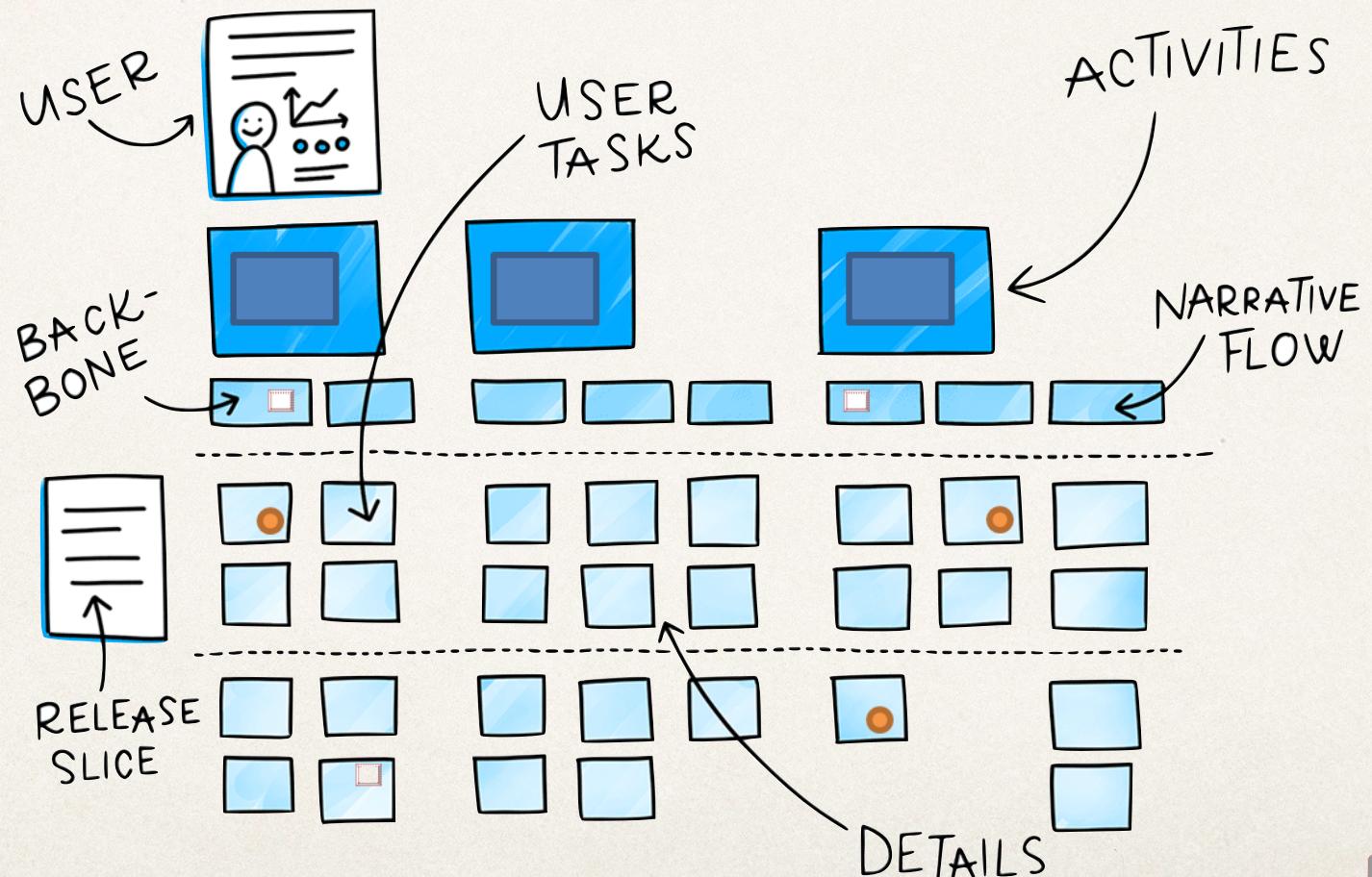
	Product backlog	Release backlog	Sprint backlog
Scopo	Definisce le features e le priorità	Features della release	User stories da completare nello sprint
Owner	Product Owner	PO e team	Team decide quanti sprint e quante storie per sprint
Work items	Contiene product features (e associate user stories)	Product features prese da PB, ognuna completabile in una release	Contiene user stories
Priorità	Compara importanza strategica delle features	Ordina features su loro business value e effort necessario	Ordina user stories su importanza funzionalità e tempo stimato
Refining	Continuous refinement: si aggiungono features, priorità cambiano	PO definisce acceptance criteria, controlla capacità team di fare la release	Team pianifica ogni sprint, decide quali stories affrontare

SCRUM: product, release, sprint...

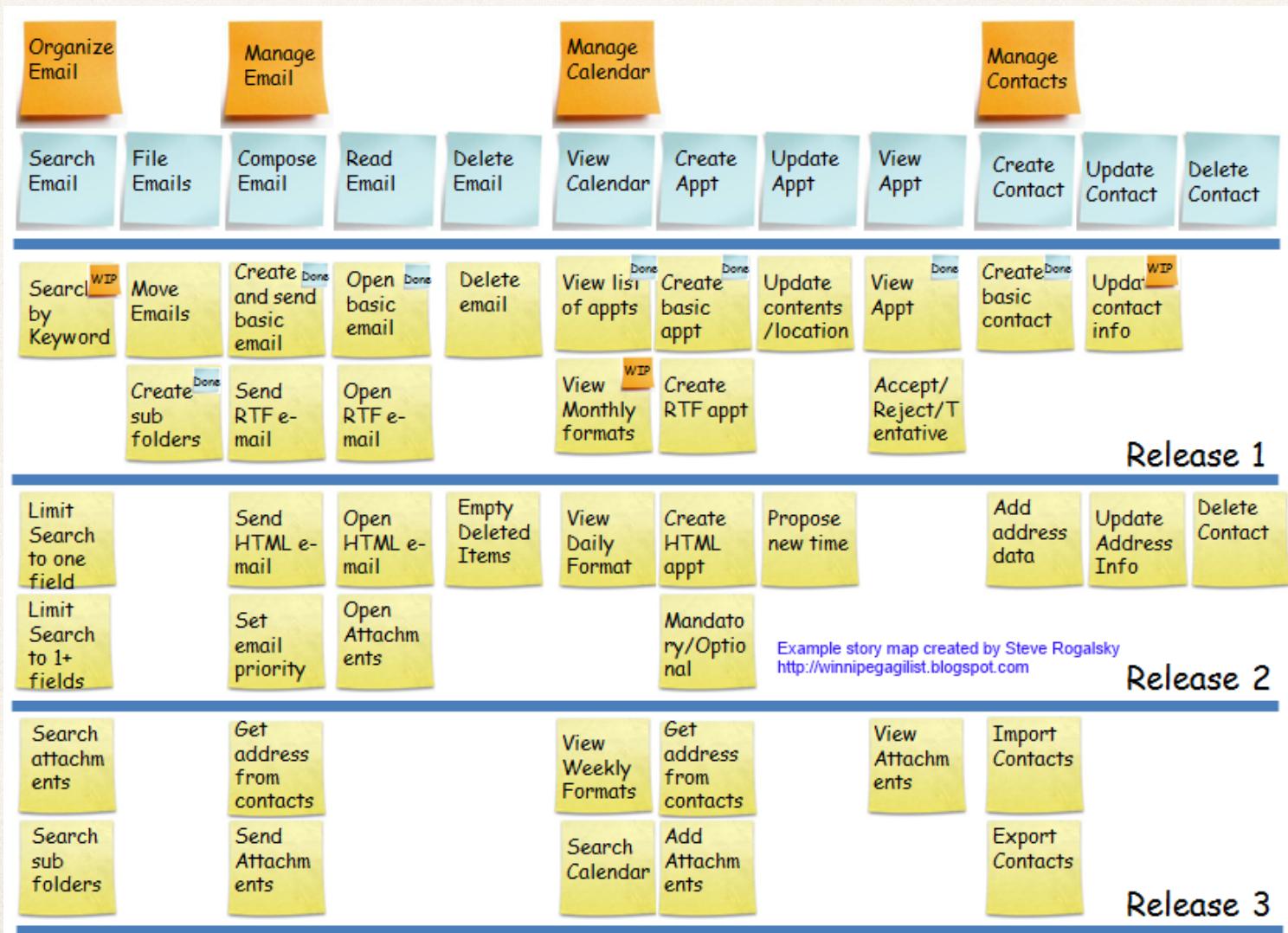


SCRUM: user story mapping

USER STORY MAPPING



SCRUM: user story mapping



SCRUM, tre fasi: fase 1

- ❖ Pre-game phase
 - ❖ Planning sub-phase
 - ❖ Include la definizione del sistema che deve essere sviluppato. Viene creata una Product Backlog List, che contiene tutti i requisiti attualmente conosciuti
 - ❖ Architecture sub-phase
 - ❖ Viene pianificato un design di alto livello del sistema, inclusa l'architettura, in base agli elementi contenuti nel Product Backlog

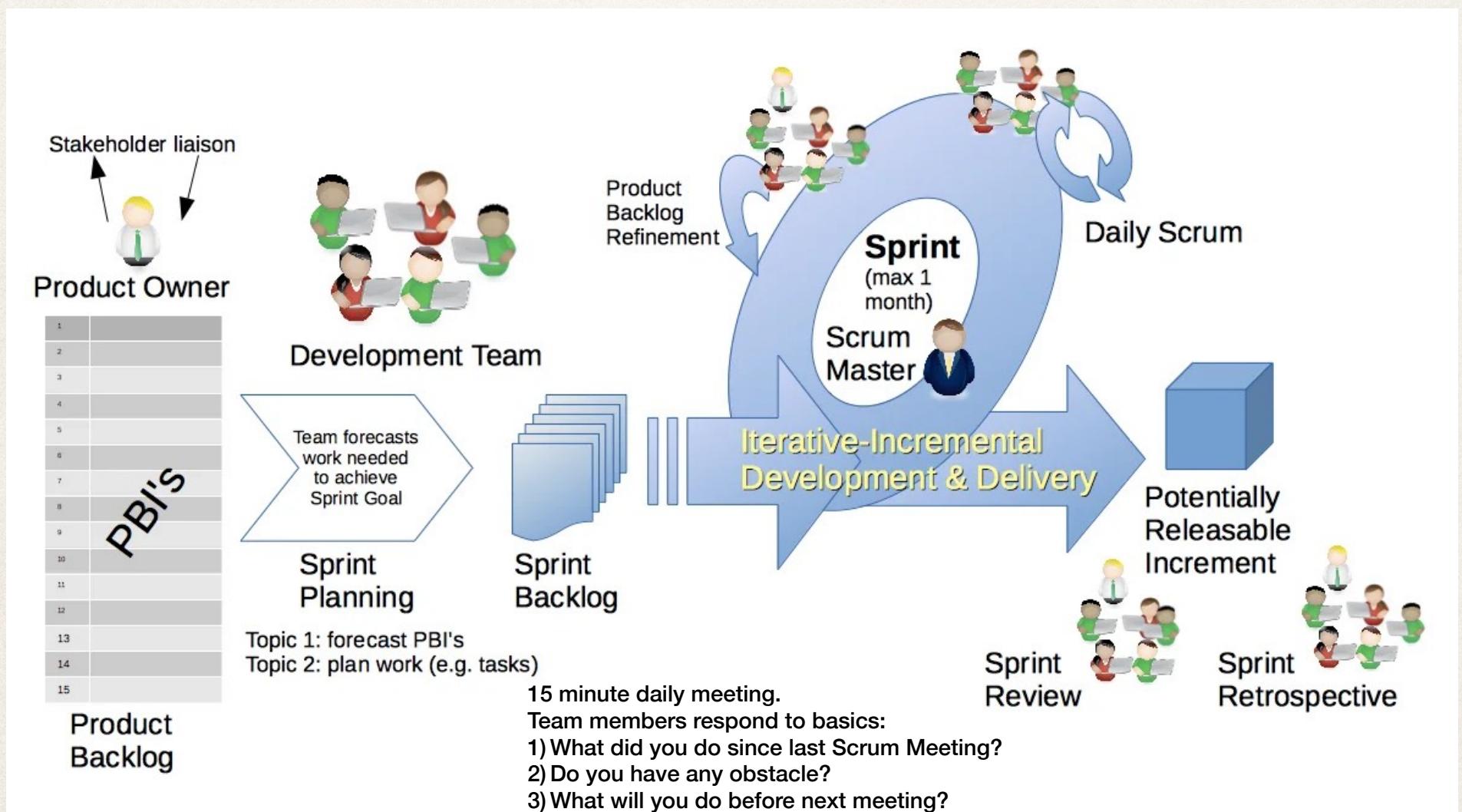
SCRUM, tre fasi: fase 2

- ❖ Development (game) phase
 - ❖ si sviluppa il sistema attraverso una serie di “sprint”
 - ❖ cicli iterativi nei quali vengono sviluppate o migliorate una serie di funzionalità
 - ❖ ogni sprint include le tradizionali fasi di sviluppo del sw
 - ❖ l’architettura del sistema evolve durante lo sviluppo negli sprint
 - ❖ uno sprint tipicamente dura da una settimana a un mese.

SCRUM, tre fasi: fase 3

- ❖ Post-game phase
 - ❖ chiusura definitiva della release

SCRUM: sprint e scheduling



SCRUM: i ruoli

Product Owner

- ❖ La persona a cui fanno riferimento tutti i soggetti interessati al progetto, incluso il cliente finale: rappresenta gli stakeholders.
- ❖ Scrive il product backlog in forma di user stories, è responsabile della definizione di “Fatto”.
- ❖ Figura di raccordo: effettua stime, aggiusta i processi che presentano problemi, gestisce l'intero procedimento secondo la pianificazione inizialmente fatta
- ❖ Poteri:
 - ❖ accettare o rifiutare i risultati di un lavoro
 - ❖ terminare uno sprint se necessario

SCRUM: i ruoli

Membro del team

- ❖ Responsabilità
 - ❖ costruiscono il prodotto
 - ❖ decidono cosa fare in ciascuno sprint
- ❖ Caratteristiche
 - ❖ cross-functional (l'eccessiva specializzazione rischia di avere alcune persone cariche di lavoro e altre che aspettano)
 - ❖ team organizzati indipendentemente
 - ❖ senza project o team manager
 - ❖ ognuno fa una cosa alla volta (no multitasking), di solito full-time
- ❖ Team
 - ❖ idealmente $7(\pm 2)$ persone (close relation, collaborate effectively)
 - ❖ se è possibile nella stessa sede / ufficio
 - ❖ stabile (non cambia) durante lo sprint

Si scala su grossi progetti
facendo “scrum di scrum”

SCRUM: i ruoli

Membro del team

- ❖ Il team si autogestisce, è
 - ❖ auto-organizzato: nessuno può dire al Team come trasformare il Product Backlog in incrementi di funzionalità potenzialmente rilasciabili, nemmeno Scrum Master
 - ❖ cross-funzionale: include tutte le abilità necessarie a creare un incremento di prodotto alla fine dello sprint
 - ❖ alla pari: i membri sono tutti Sviluppatori allo stesso livello, nessuna eccezione
 - ❖ unitario: non ci sono sottogruppi, senza eccezioni per attività o domini particolari
 - ❖ i singoli membri potranno avere specializzazioni personali, ma l'intero Team è responsabile dell sviluppo: “quello non l’ho fatto io” non è accettato

SCRUM: i ruoli

Scrum Master

- ❖ Si occupa di supportare il team (ne fa parte), garantendo le condizioni ambientali e le motivazioni necessarie ad eseguire al meglio il lavoro commissionato
- ❖ Non ha autorità sul team: serve il team, non lo dirige (non può licenziare nessuno)
- ❖ Soprassiede ai rituali dello scrum (ne è il responsabile)
 - ❖ non “cosa” bisogna fare (quello è il product owner) ma il come lo stiamo facendo
 - ❖ tiene aggiornato lo stato di progresso di lavoro, in modo che tutto il team sappia come sta andando il progetto
- ❖ **Meglio se non partecipa come programmatore**

SCRUM: i rituali

Sprint Planning

- ❖ Pianificazione iniziale di gruppo (PO, SM, Team): circa 1 ora per ogni settimana di sprint
- ❖ Team decide cosa entra a far parte dello Sprint Backlog: sceglie le user stories insieme al PO
- ❖ Team e PO concordano la “Definition of Done” per ogni elemento nello Sprint Backlog
- ❖ Il Team fa una stima di quanto lavoro riuscirà a fare nello sprint basandosi sulla sua “velocità” di sviluppo

SCRUM: i rituali

Daily scrum (o stand-up)

- ❖ Max 15 min, in piedi, ogni membro del Team dice cosa ha fatto ieri, cosa pianifica oggi, che impedimenti ha trovato

Sprint Review

- ❖ Max 4 h, riguarda i prodotto: cosa è stato o non è stato completato in questo sprint.

Sprint Retrospective

- ❖ Max 3 h, riguarda il processo: cosa è andato bene e quali impedimenti si sono incontrati

SCRUM: i rituali

Quanti meeting?

Sprint duration	Sprint planning	Daily scrum	Sprint review	Sprint retro
1 week	< 2 h	15 min	1 h	45 min
2 weeks	< 4 h	15 min	2 h	1.5 h
4 weeks	< 8 h	15 min	4 h	3 h

SCRUM: i rituali

Quanti meeting?

A TYPICAL 2
WEEKS
CALENDAR

Week	Monday	Tuesday	Wednesday	Thursday	Friday
1	Daily Scrum SPRINT PLANNING	DS	DS MEETING XYZ	DS REFINEMENT	DS MEETING XYZ
2	DS MEETING XYZ	DS REFINEMENT	DS MEETING XYZ	DS REFINEMENT	DS MEETING XYZ SPRINT REVIEW RETROSPECTIVE

SCRUM: i rituali

Chi comanda il meeting?

	owner	facilitator
Backlog grooming	PO	SM *
Sprint planning 1	PO	SM *
Sprint planning 2	Team	SM *
Daily scrum	Team	SM *
Sprint review	PO	SM *
Sprint retrospective	Team	SM *

* OR ANYONE WITH
FACILITATION
EXPERTISE

SCRUM: fail fast technique

- ❖ Scrum vuole che tu fallisca: è conosciuto per lo slogan “fail fast”
- ❖ Sembra una presa in giro, ma c’è un’ottima ragione, e nello slogan il punto chiave è “fast”...
- ❖ I progetti falliscono: l’abbiamo detto, impossibile non incappare in problemi.
- ❖ Scrum vuole che le magagne saltino fuori in fretta, perché prima vengono trovate, prima impari, e i costi per il progetto sono minori.

Fail fast, learn, iterate

SCRUM: quali sono i patti

Il Team promette agli Stakeholders:

- ❖ Il PO del TEAM difenderà gli interessi degli stakeholders
- ❖ Il tempo degli stakeholders sarà usato saggiamente, saranno poste solo questioni utili allo sviluppo
- ❖ Il lavoro sarà della massima qualità, compatibilmente coi vincoli imposti
- ❖ Alla fine di ogni sprint saranno consegnate nuove funzionalità che potranno essere validate dagli stakeholders

Gli Stakeholders promettono al Team:

- ❖ Gli stakeholders saranno disponibili per aiutare il team quando necessario
- ❖ Lo ScrumMaster sarà coadiuvato a rimuovere gli ostacoli
- ❖ I vincoli e le priorità non cambieranno durante uno sprint senza il consenso del Team
- ❖ Partecipare a uno sviluppo Scrum non danneggerà la carriera dei membri del Team

SCRUM: problemi tipici

- ❖ Ignoranza dei valori agili e di SCRUM
- ❖ Prodotto non testato alla fine dello sprint
(cattiva definizione di “fatto”)
- ❖ Backlog non pronto all’inizio dello sprint
(cattiva definizione di “ready”)
- ❖ Mancanza di/Cattiva facilitazione
- ❖ Mancanza di supporto da parte dei manager
- ❖ Mancanza di supporto da parte degli stakeholders
- ❖ Gestione caotica di “scrum di scrum”

XP o SCRUM: confronto

XP	SCRUM
Orientato alla qualità	Orientato al project management
Iterazioni: 1-2 settimane	Sprint: 2-4 settimane
Requisiti sempre modificabili	Requisiti modificabili alla fine dello sprint
Il cliente ordina le storie	Il cliente ordina le storie
Coaching informale	Scrum master certificato
Buone pratiche tipiche: TDD, Pair programming, planning the game, refactoring	Buone pratiche tipiche: retrospettiva post-mortem, uso di strumenti di PM, planning poker