

Ingegneria del Software

Agile

Prof. Paolo Giorgini

A.A. 2024/2025

From the Kanban Simulation

Insight 3

**“Agility” doesn’t just mean working faster.
Agility is a strategy for surviving in an
uncertain, rapidly changing environment.
The essence of agility is inspection and
adaptation (empiricism).**

From the Kanban Simulation

Insight 4

In a fast-changing environment short lead times are quite important:

- **Time-to-market, MVP, incremental deployment**
- **Customer expectations, performance of competitors**
- **Feedback!!!**

From the Kanban Simulation

Insight 6

- **Fully loaded systems are slow, unstable, unpredictable, and have long response times (Eg. Hard Drive).**
- **Counterintuitive: If all are a 100% „busy“, we don't become faster but rather slower!**
- **We should not manage resource utilization but rather the flow of work!**

Metodologie Agile

- In IS, per **metodologia agile** o **metodo agile** si intende un particolare metodo per lo sviluppo del software che coinvolge quanto più possibile il committente, ottenendo in tal modo una elevata reattività alle sue richieste
- Esistono un certo numero metodologie Agile
 - Agile alliance, organizzazione no-profit creata allo scopo di diffonderle
 - <http://www.agilealliance.org>



Manifesto

Vengono privilegiati i seguenti elementi:

- Gli individui e le loro interazioni rispetto ai processi ed agli strumenti
- Software funzionante rispetto ad un'ampia documentazione
- La collaborazione col cliente rispetto alla negoziazione dei contratti
- La pronta risposta ai cambiamenti rispetto all'esecuzione di un piano

Ache se viene attribuito un valore agli elementi riportati a destra, sono più importanti quelli a sinistra

Cos'è l'Agile?

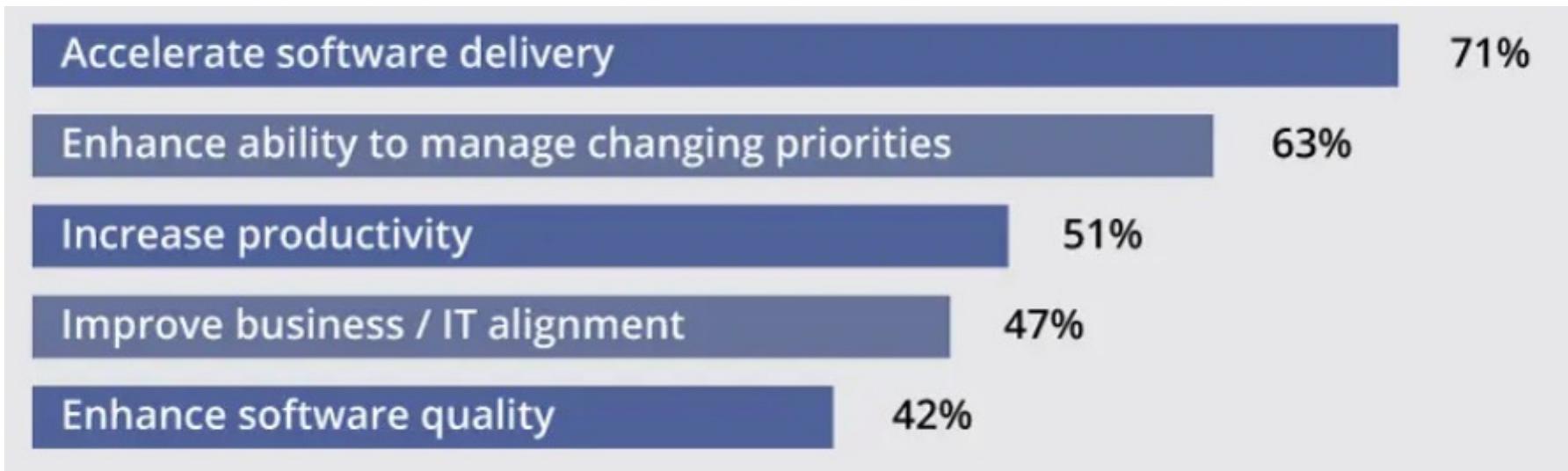
- Reazione efficace (rapida e adattiva) ai cambiamenti
- Comunicazione efficace fra tutti gli stakeholder
- Assorbimento del cliente nel team di sviluppo
- Organizzazione del team che lo ponga in diretto controllo del proprio lavoro

Producendo ...

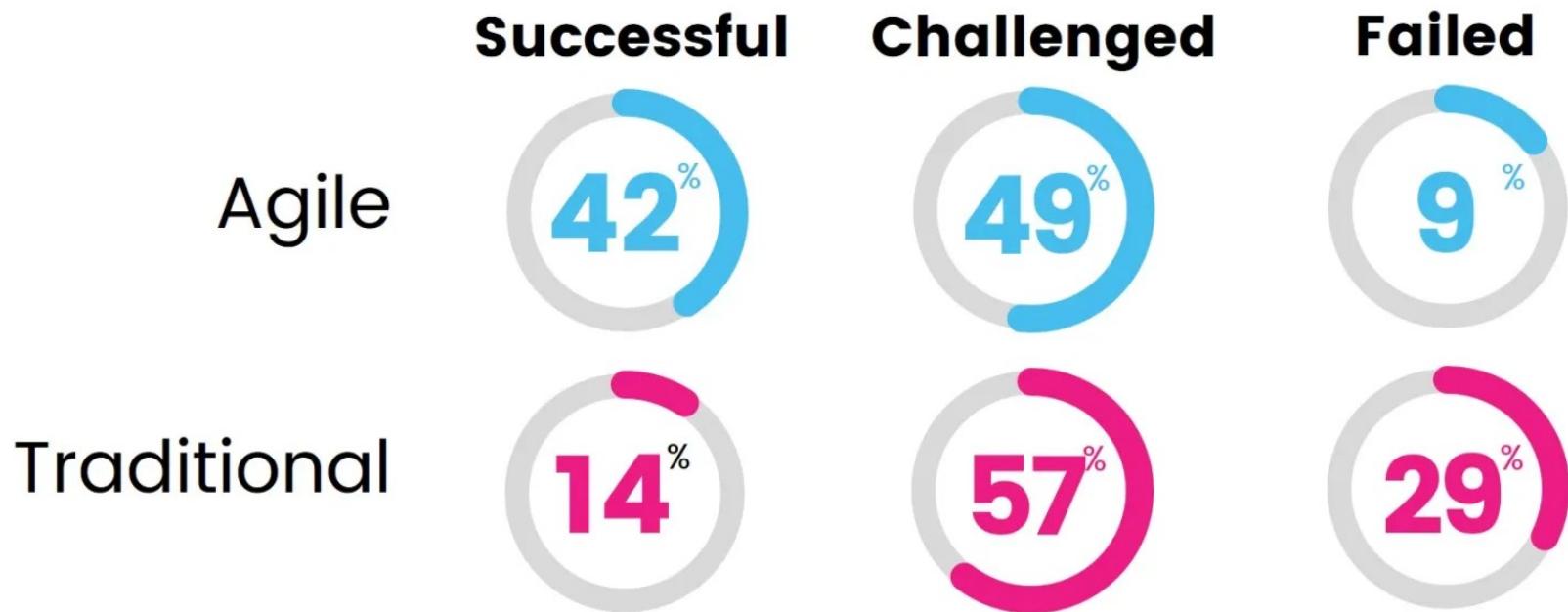
- Consegne incrementali e frequenti di software

Top five reasons for adopting agile

Respondents were asked why their teams adopted Agile methodologies and techniques. These were the most responded benefits:



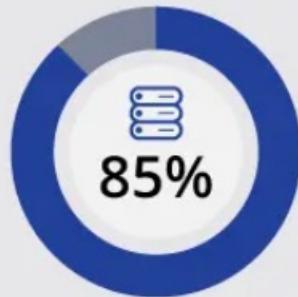
Agile vs Waterfall



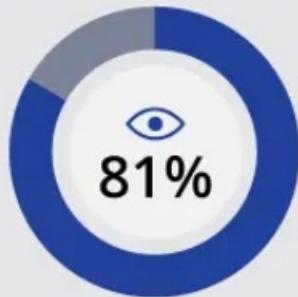
Source: Standish (2020) CHAOS Report

Top Five Agile Techniques Employed

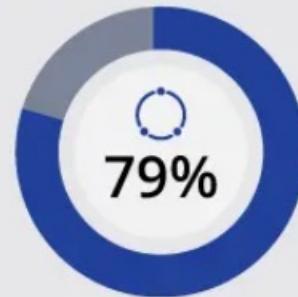
These are the five most used tactics that help teams adhere to the twelve principles of Agile.



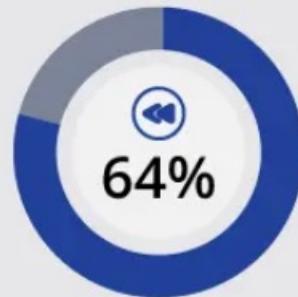
Daily Standup



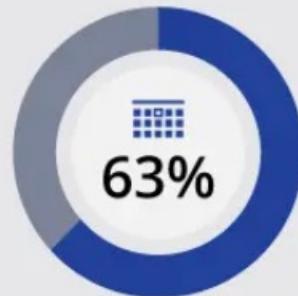
Retrospectives



Sprint / Iteration Planning

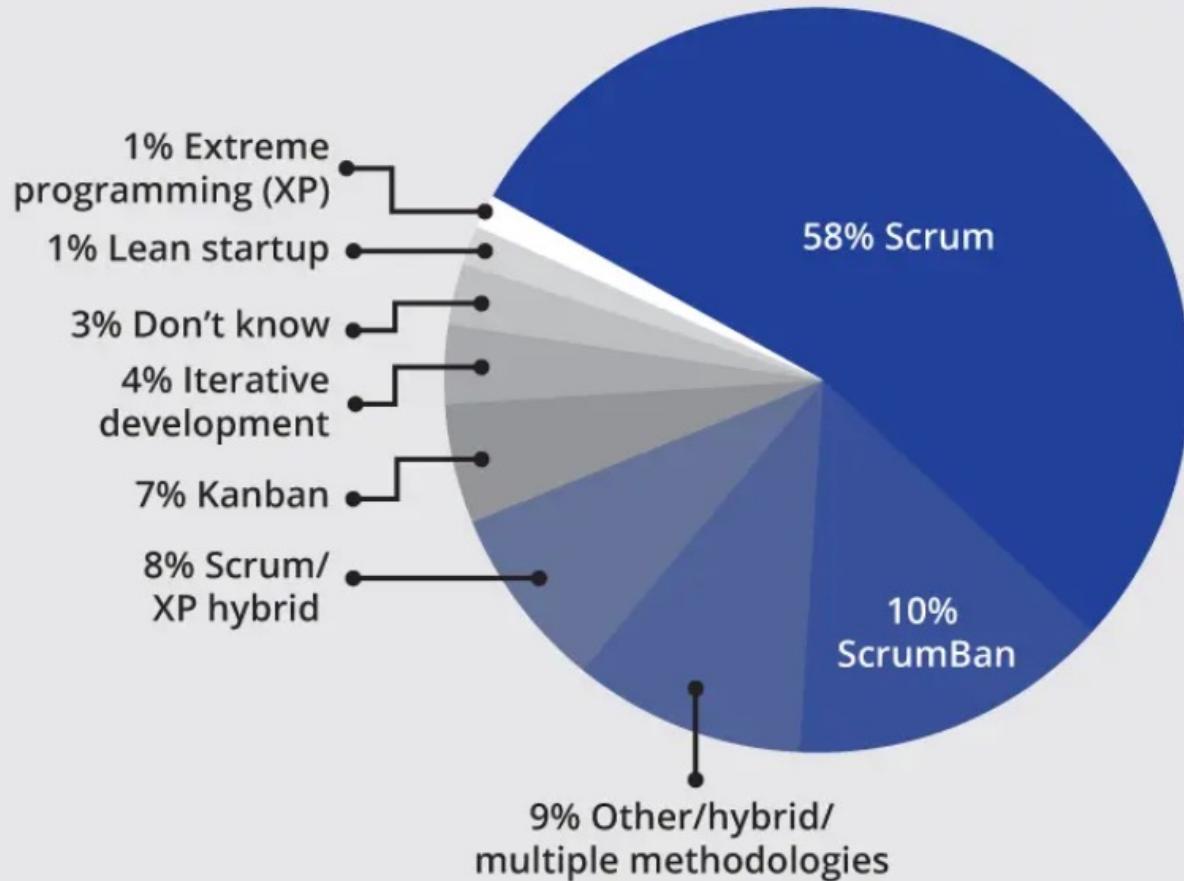


Short Iterations



Kanban

Top Five Agile Methodologies



Processi Agili

- Guidati dalla descrizione del cliente di che cosa gli serve: scenario
 - una descrizione di possibili interazione tra utenti e sistema (descrizione del funzionamento)
- Sviluppano software in maniera iterativa con forte enfasi sulle attività di costruzione
 - molteplici *incrementi software*
 - Adattamento ai cambiamenti

Agile Manifesto: 12 principles



Caratteristiche delle metodologie Agile

Tutte le metodologie agili hanno una serie di **punti in comune**:

1. Reazione efficace (rapida e adattiva) ai cambiamenti;
2. Comunicazione efficace fra tutti gli stakeholder;
3. Assorbimento del cliente nel team di realizzazione;
4. La stretta collaborazione tra gli stakeholder di business e i team;
5. La comunicazione diretta, personale, anziché scritta;
6. Il frequente rilascio di nuovi prototipi funzionanti;
7. I gruppi di risorse sono auto organizzati;
8. Obiettivo di realizzare prototipi semplici ed efficienti.

Agile vs IS

Non si deve pensare erroneamente che l'agilità dia la licenza di improvvisare le soluzioni. E' necessario un processo ed è fondamentale la disciplina

Non si tratta di scegliere fra agilità e ingegneria del software ma di definire un approccio agile all'ingegneria del software

Molti concetti della metodologia agile non sono altro che adattamenti dei migliori concetti di ingegneria del software

Fattori umani

- Qualità chiave per il team agile
 - Competenza
 - Obiettivi Comuni e Collaborazione
 - reciproca fiducia e rispetto
 - Capacità di prendere decisioni
 - Capacità di soluzione creativa dei problemi
 - Auto-organizzazione

Critiche alle Metodologie Agili

- I passaggi alle metodologie agili richiedono un **fortissimo cambiamento culturale all'interno dell'azienda**; il cambiamento deve essere condiviso dal team: è molto difficile imporlo dall'alto.
- Possono complicare le negoziazioni contrattuali con il cliente.
- **Funzionano solo per quelle aziende o per quei team che dispongono di risorse senior, con molta esperienza e capacità di autonomia.**
- Le aziende si espongono ai rischi legati al turn over delle risorse umane.
- Le attività di progettazione iniziale del software sono poste in secondo piano e quindi poco curate; il lavoro di progettazione deve però essere continuo e distribuito durante tutto lo svolgimento del processo.
- La documentazione prodotta può essere poca e anche di scarsa qualità;
- La dinamicità non deve essere confusa con il “Cowboy Programming”: i **metodi agili richiedono una disciplina rigorosa e un processo ben definito anche se di tipo adattivo.**

Processi agile (1)

- **Iterativo e incrementale**
 - L'intera applicazione viene realizzata in **unità incrementali** dette **iterazioni**. Il tempo di sviluppo di ciascuna iterazione è piccolo (due settimane in media), fisso e rigorosamente rispettato. Ogni iterazione rappresenta quindi un **mini incremento** delle funzionalità del sistema che viene rilasciato sopra al risultato dell'iterazione precedente
- **Coinvolgimento attivo del cliente**
 - Si ha un forte **coinvolgimento del committente** mediante collaborazione faccia-a-faccia. Il risultato di ogni iterazione viene testato e approvato dal cliente stesso. Il riscontro ottenuto è implementato in iterazioni successive, riducendo così i rischi e garantendo una maggiore soddisfazione del cliente
- **Guidato dalle funzionalità**
 - Viene posta una maggiore enfasi nel fornire le **funzionalità richieste** nell'applicazione. Il principio Pareto dell'80/20 viene applicato per selezionare il **20% di funzionalità** che verranno effettivamente utilizzate per **l'80% del tempo**

Processi agile (2)

- **Tempo fisso**
 - Ogni **iterazione** ha una **durata limitata di tempo**, in cui un nuovo incremento delle funzionalità viene realizzato e consegnato
- **Consegna basata su priorità**
 - Alle **funzionalità** viene assegnata una **priorità** in base a esigenze del cliente, rischi di sviluppo, opportunità di business. Dapprima vengono sviluppate le funzionalità con **priorità maggiore**, quindi al termine di ogni iterazione, le priorità del progetto vengono rivalutate
- **Adattiva**
 - La metodologia in generale è molto adattabile e flessibile, in modo tale che l'applicazione realizzata possa soddisfare l'afflusso di nuovi requisiti durante lo sviluppo. L'obiettivo non è quello di rimuovere l'incertezza all'inizio, ma piuttosto quello di **adattarsi a necessità mutevoli**
- **Responsabilizzare il team**
 - I **team di progetto** sono in genere **piccoli** e caratterizzati da un alto grado di **interazione e comunicazione**. Dal momento che l'intero team è attivamente coinvolto, esso stesso ha il potere di **prendere decisioni**

Processi agile (3)

- **Centrato sulle persone**
 - Viene posta una maggiore enfasi su come le persone più qualificate ad effettuare lo sviluppo **lavorino assieme**, che non sul seguire i processi. La documentazione e le altre attività non propriamente di sviluppo e test sono ridotte al minimo
- **Sviluppo rapido**
 - Lo sviluppo viene effettuato velocemente, utilizzando moderne tecnologie di sviluppo leggere. Questo ha il vantaggio di consentire **cicli di feedback ridotti**
- **Rilasci frequenti**
 - Procedimento composto da **piccoli passi**, il team di sviluppo è in grado di produrre versioni del software in tempi più ridotti; **rilasci più frequenti**
- **Testing**
 - **Testing** o la verifica automatica del **corretto funzionamento** del sistema. Si applica sia al codice, che ai dati, e agli altri prodotti a cui le diverse discipline sono orientate (come i modelli o la documentazione). Dove non è possibile un test automatico, viene proposta una **verifica manuale**

Processi agile (4)

- **Più disciplina**
 - Volendo essere veloci, tutto deve essere consegnato correttamente già la prima volta. Il processo implica molto **gioco di squadra e autodisciplina**. Quindi, ai membri del team si richiede di essere altamente qualificati e organizzati
- **Semplicità**
 - L'accento è posto sul mantenere **ogni cosa il più semplice possibile** ed essere aperti al cambiamento. È importante sottolineare che quest'ultimo aspetto è sempre presente nell'insieme dei valori di tutte le metodologie agili

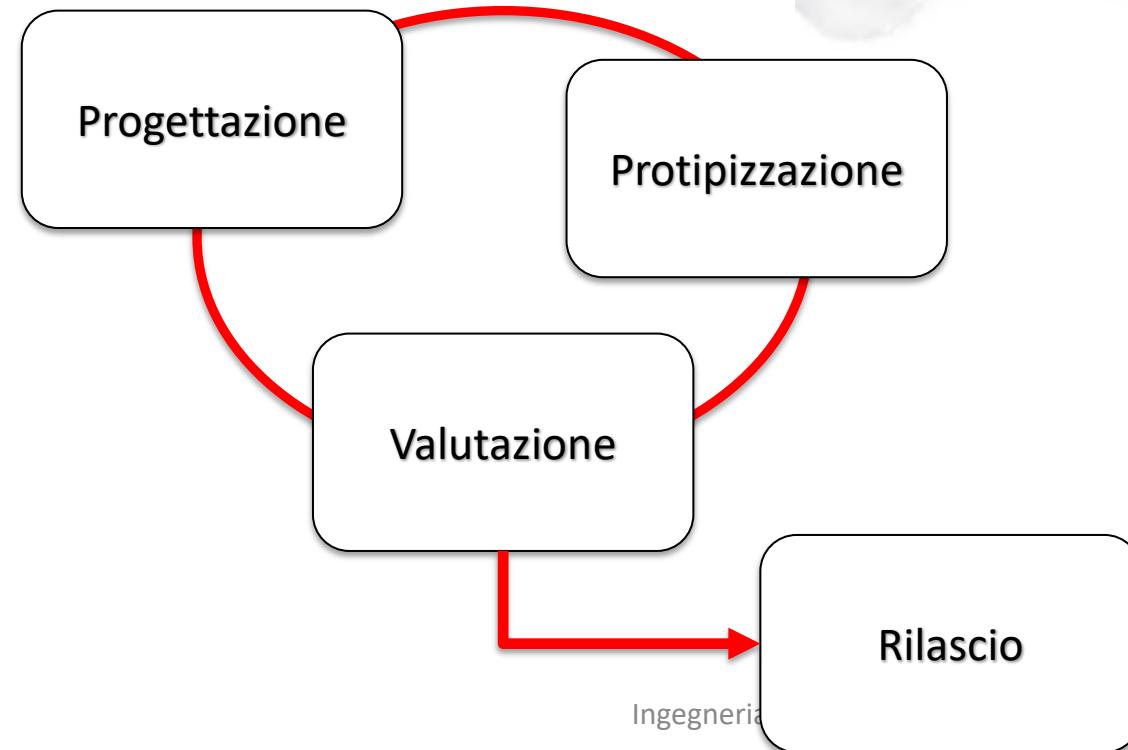
Modello iterativo (prototipale)

Sviluppo del prodotto/servizio attraverso successive versioni, incomplete , del prodotto,

Disposable Prototype: contiene solo alcune delle funzionalità, tipicamente le interfacce (poi è eliminato).

Time Box Prototype: è una replica abbastanza fedele, ma in piccolo, del prodotto finale, (poi è eliminato).

Evolutionary Prototype: evolve step by step nel prodotto finale.



Versioni successive del prodotto

- **Disposable Prototype**: solo alcune delle funzionalità (poi è eliminato)
- **Time Box Prototype**: replica abbastanza fedele, ma in piccolo, del prodotto finale (poi è eliminato).
- **Evolutionary Prototype**: evolve step by step nel prodotto finale

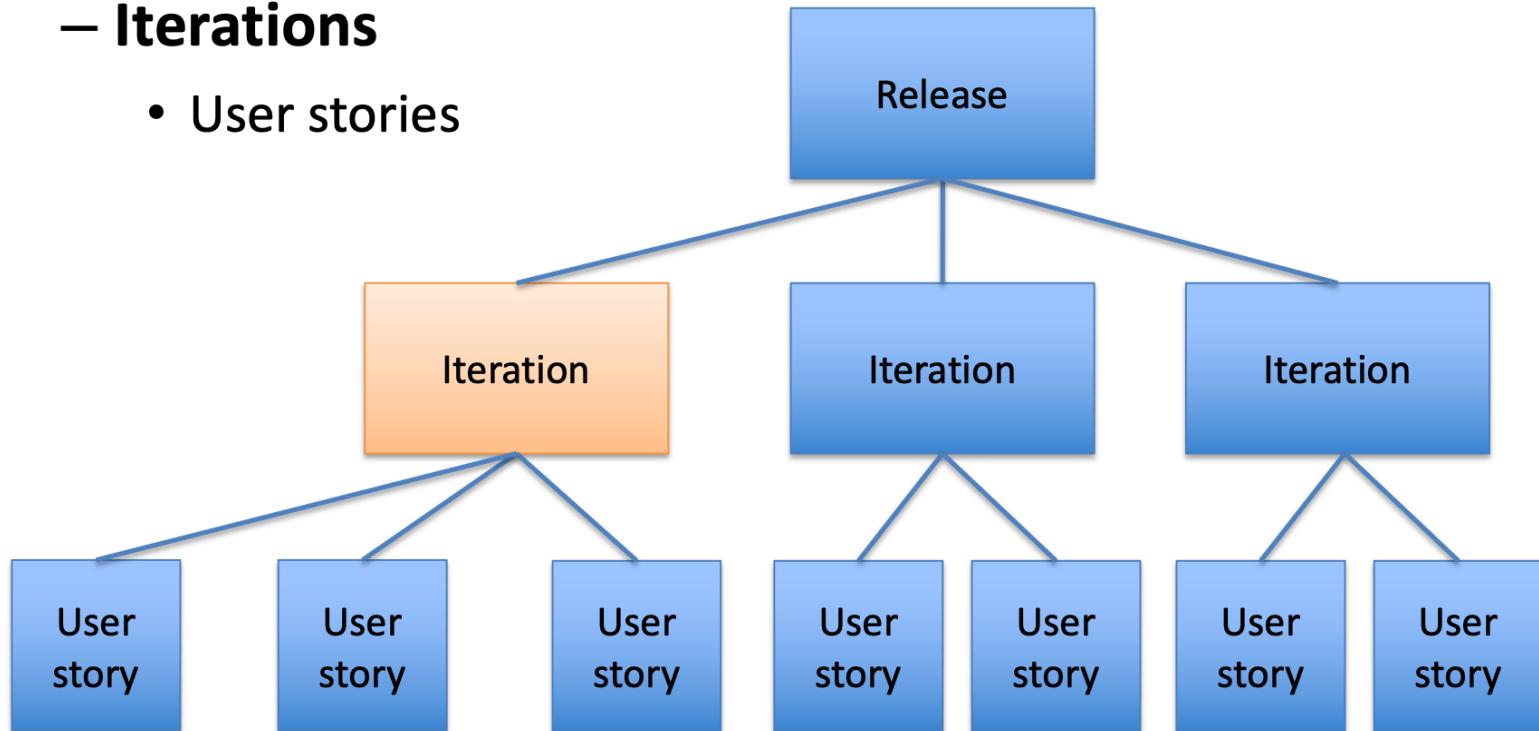


Ciclo di vita dello sviluppo agile



Release, iterations e user stories

- Release
 - Iterations
 - User stories



Iterazione 0

- L'iterazione iniziale convenzionalmente denominata **Iteration 0** (o **Cycle 0**), ha lo scopo di lanciare il progetto, focalizzandosi sui seguenti obiettivi:
 - garantire una partecipazione attiva degli stakeholder;
 - ottenere fondi e supporto necessari;
 - iniziare a costituire il team;
 - creare un modello iniziale dei requisiti;
 - realizzare un modello iniziale dell'architettura;
 - effettuare il setup degli ambienti.

Iterazioni di sviluppo

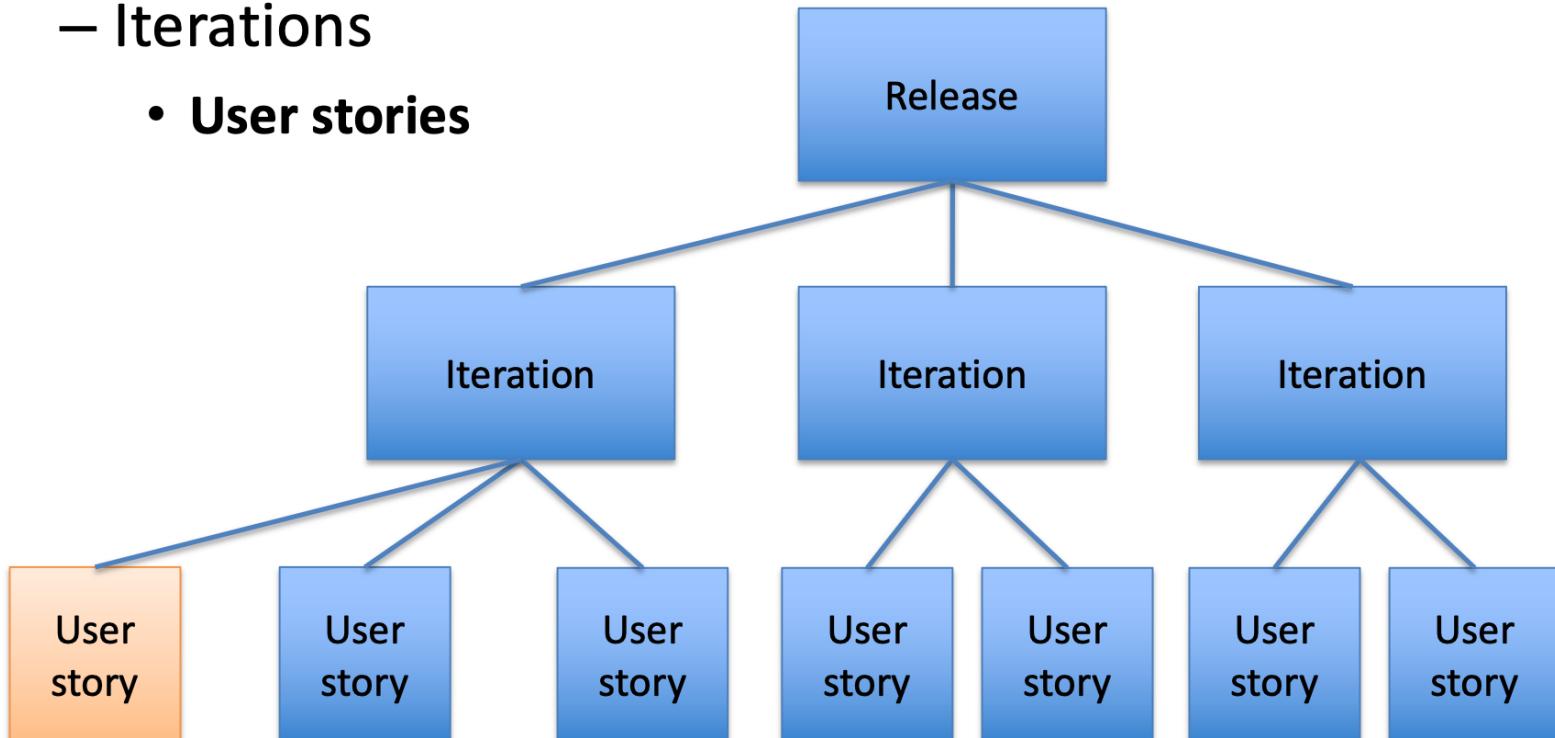
- **Collaborazione** stretta con gli stakeholder e gli altri membri del team
- **Analisi e progettazione emergente**
 - I requisiti vengono analizzati con la tecnica del model storming - criterio di "just-in-time"- per alcuni minuti, quindi realizzati con attività della durata di alcune ore o qualche giorno
 - La progettazione e l'architettura del sistema è "emergente"
 - Guidati dai modelli di architettura (diagrammi abbozzati a mano su lavagne) si procede con il "Test-Driven Design" (TDD) in cui iterativamente si scrive un test e poi solo il codice di produzione sufficiente a soddisfare le esigenze di tale test
- **Assicurare continuamente una qualità senza compromessi**
 - L'utilizzo continuo delle tecniche di refactoring sul codice applicativo e/o sugli schemi del database è necessario per assicurare che si abbia sempre il miglior design possibile
- **Rilascio continuo e regolare di software funzionante**
 - al termine di ogni ciclo di sviluppo/iterazione, si dovrebbe sempre avere un sistema parziale funzionante che sia possibile mostrare a cliente e stakeholder
- **Test, test e... test**
 - l'intero processo di sviluppo agile è caratterizzato da una notevole quantità di test realizzati durante tutto lo sviluppo del sistema

Rilascio

- **Rilascio (fine partita)** si procede a effettuare la transizione del sistema in produzione
 - **Test finale del sistema:** i test finali di sistema e di accettazione devono essere effettuati a questo punto, anche se la maggior parte dei test dovrebbe essere stati compiuti durante le iterazioni di sviluppo
 - **Rework:** ovviamente non ha senso effettuare delle verifiche finali se non si alloca una certa quantità di tempo a correggere le anomalie riscontrate
 - **Training:** deve essere fatta formazione e corsi a utenti, sistemisti e operatori di supporto, per consentirgli il miglior utilizzo del sistema
 - **Deploy del sistema:** questo aspetto riguarda l'insieme di tutte le attività di messa in produzione del sistema stesso
- L'intero ciclo di sviluppo può ripetersi per i successivi rilasci previsti dal progetto, passando dalla release N alla N+1

Release, iterations e user stories

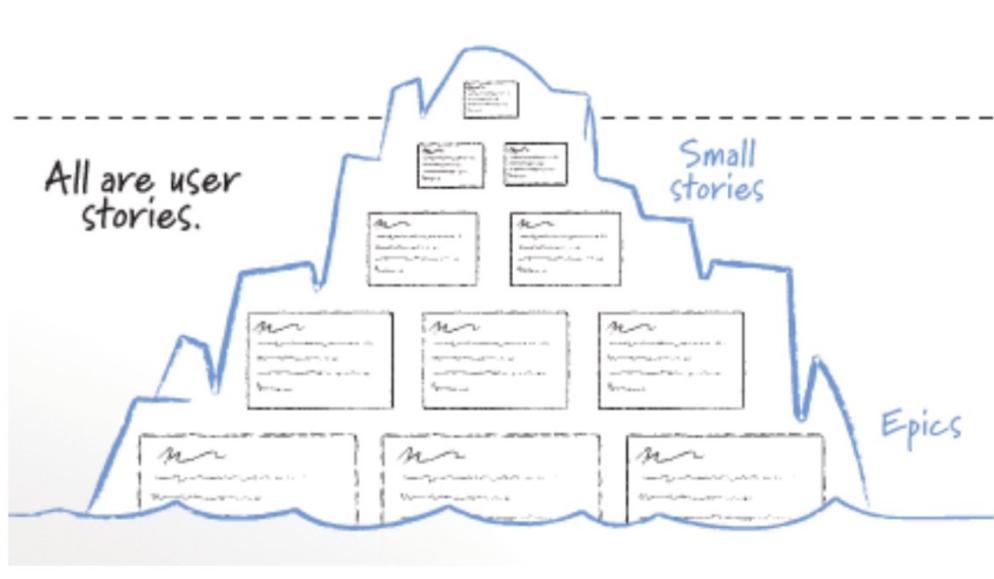
- Release
 - Iterations
 - User stories



User Story

A user story is an end goal, not a feature, expressed from the software user's perspective.

The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.



User story:
As a [user], I [want to], [so that]

Il modello Scrum

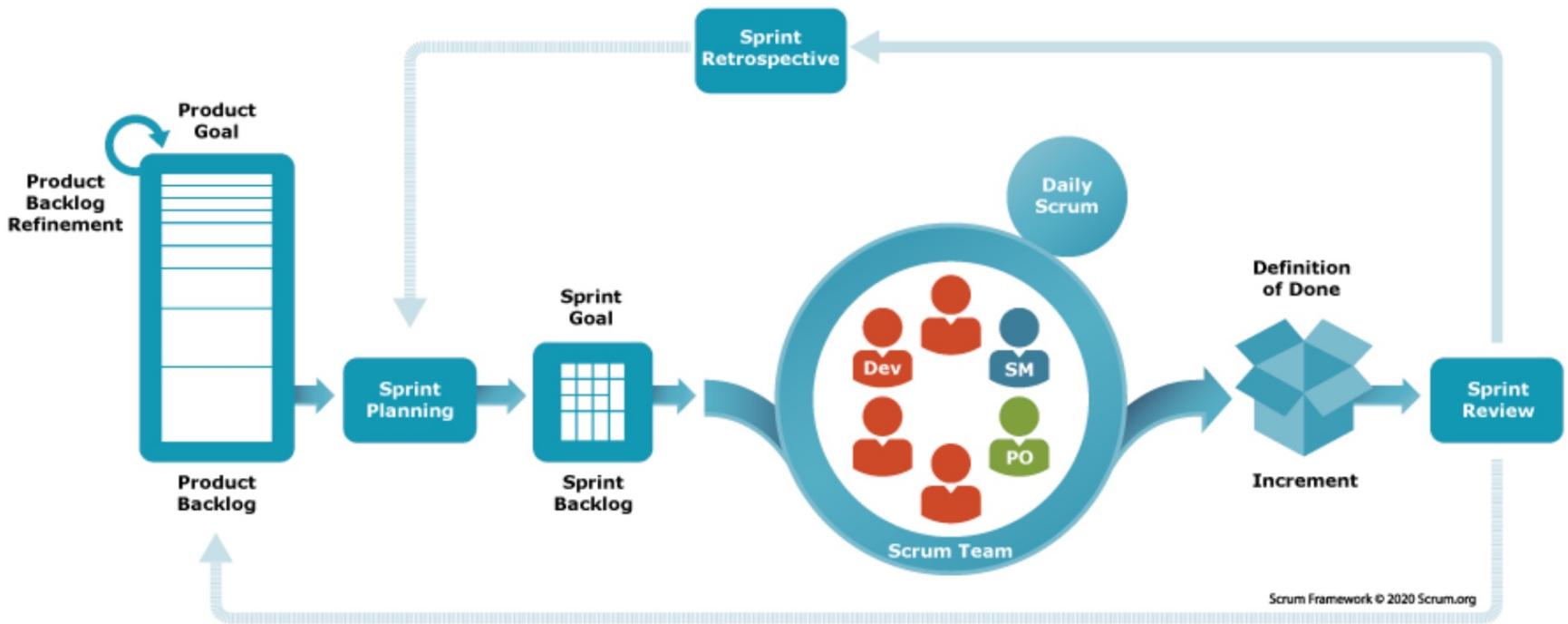
- Insieme di **pratiche e regole** interrelate che
 - ottimizzano l’ambiente di sviluppo, riducono l’**overhead** organizzativo, sincronizzando i requisiti emergenti dal mercato con prototipi iterativi
- Un insieme di funzionalità del prodotto realmente utilizzabili vengono rilasciate e messe in produzione **ogni quindici o trenta giorni**
- Scrum è un insieme di valori, pratiche e regole all’interno di un unico framework di sviluppo che può essere rapidamente implementato e ripetuto

Il modello Scrum: principi

- Piccoli team di lavoro organizzati
- Adattabilità a cambiamenti tecnici o richiesti dal mercato
- Produzione di frequenti incrementi software
- Suddivisione del lavoro e delle persone che lo svolgono
- Costante attività di testing e documentazione
- Capacità di dichiarare un prodotto concluso

Uno dei primi **errori che si fanno usando Scrum** è pensare che non serva pianificare.

In realtà **usare Scrum significa pianificare continuamente dandosi sia degli obiettivi a medio-lungo termine con Vision e Roadmap di progetto**; sia degli obiettivi a breve e brevissimo termine con Pianificazione dello Sprint e della giornata.



Source: <https://www.scrum.org/resources/what-scrum-module>

I punti fondamentali di Scrum

- Basa su tre semplici punti: **Sprint**, **Backlog** e **Scrum Meeting**
 - prevede di dividere il progetto in blocchi rapidi di lavoro (**sprint**) alla fine dei quali consegnare una versione del sistema al cliente, indica come definire i dettagli del lavoro da fare nell'immediato futuro (**backlog**) per averne una definizione estesa, organizza riunioni giornaliere del team di sviluppo (**daily scrum**) per verificare cosa si è fatto e cosa si farà

Ruoli

Ruoli (maiali)

ScrumMaster	Gestisce e garantisce il processo (tipicamente al posto del Project Manager standard), ma non gestisce il progetto o il team
Product Owner	Rappresenta i committenti e il business
Team	Un gruppo inter-funzionale di circa 7 persone che effettua l'analisi, la progettazione l'implementazione, i test, auto-organizzandosi e modificando il proprio processo mediante gli incontri di retrospective

- **Product Owner**

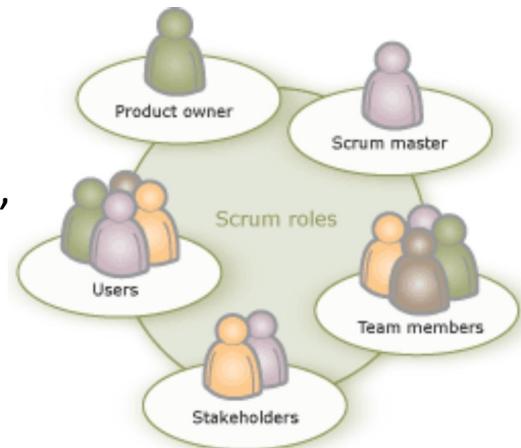
- Single person
- Product responsible

- **Team**

- ~7 self-organized members”

- **Scrum master**

- protect team
- manage process, call meetings
- Help product owner, the team and the organization



The Product backlog

Id	Name	User story	How to Demo	Importance
1	Log in	As a customer I want to log in to the internet bank so that I can do my banking stuff.	Given that I am at the start page for the internet bank, when I fill in my user name, password and press the Log in button, then I should be logged in to my account at the bank.	1
2	Log out	As a customer I want to be able to log out from my banking session, so that I can be sure that my kids won't be able to steal my money.	Given that I am logged in to the internet bank, when I press the Log out button, then I should be logged out from the session and redirected to the start page.	2

Requirements docs vs the product backlog

Not just user stories

Priority	Item	Details (wiki URL)	Initial Size Estimate
1	As a buyer, I want to place a book in a shopping cart (see UI sketches on wiki page)	...	5
2	As a buyer, I want to remove a book in a shopping cart	...	2
3	Improve transaction processing performance (see target performance metrics on wiki)	...	13
4	Investigate solutions for speeding up credit card validation (see target performance metrics on wiki)	...	20
5	Upgrade all servers to Apache 2.2.3	...	13
6	Diagnose and fix the order processing script errors (bugzilla ID 14823)	...	3
7	As a shopper, I want to create and save a wish list	...	40
8	As a shopper, I want to add or delete items on my wish list	...	20

"The Product Backlog includes a variety of items, primarily new customer features ("enable all users to place book in shopping cart"), but also major engineering improvement goals (e.g., "rewrite the system from C++ to Java"), improvement goals (e.g. "speed up our tests"), research work ("investigate solutions for speeding up credit card validation"), and, possibly, known defects ("diagnose and fix the order processing script errors") if there are only a few problems. (A system with many defects usually has a separate defect tracking system.)" (The Scum Primer)

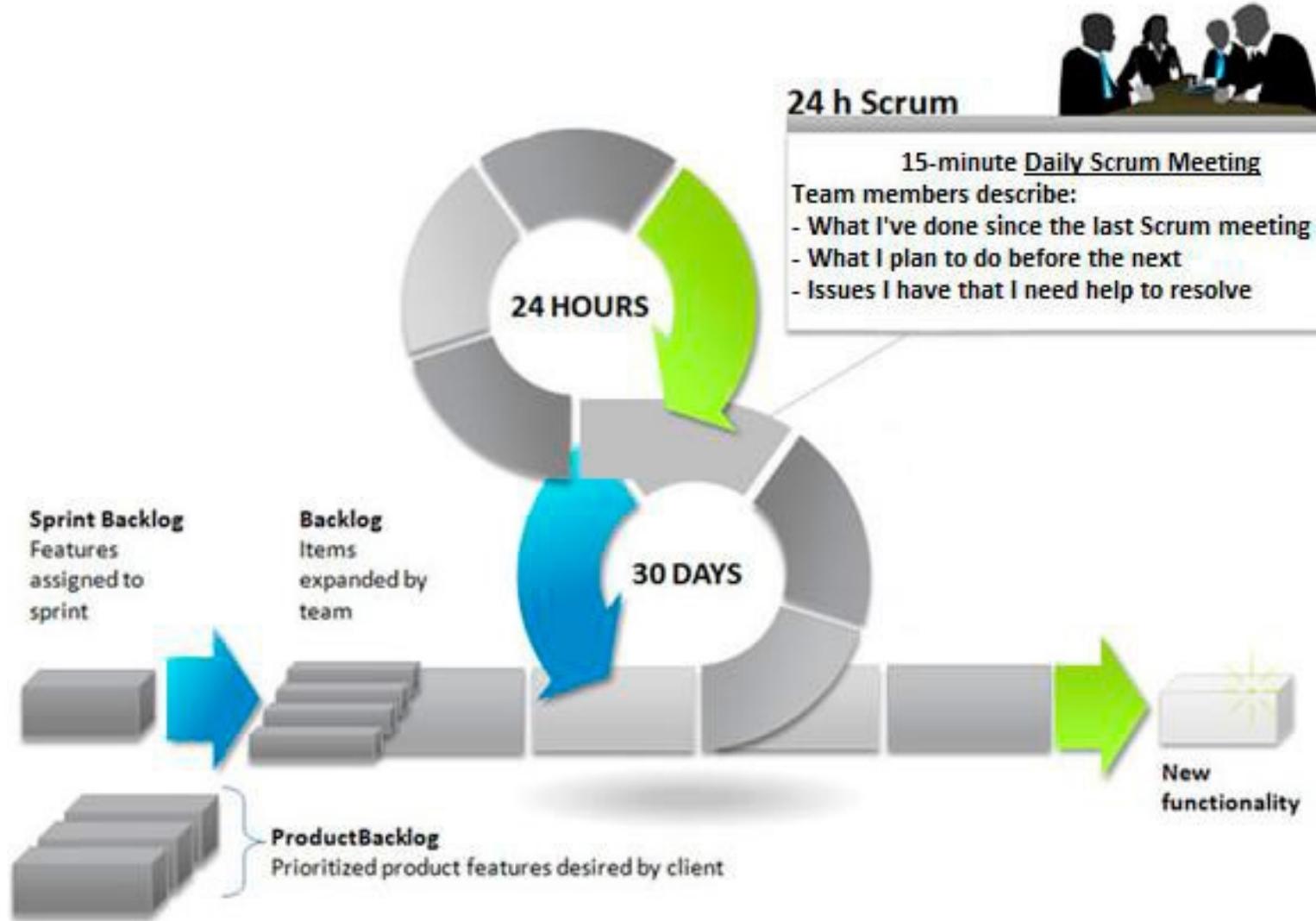
Sprint

- **A time-limited period of work with defined output.** It is the scrum way to define an iteration.
- Each sprint delivers “**potentially shippable**” increments – working software
- Lasts from **2 weeks to 2 months**, typically

“Potentially shippable”

- Non significa “shipped”
- Significa **testato** nella funzionalità, non necessariamente per le prestazioni o l'usabilità.
- Significa "codice di qualità"
- Non è un **prototipo sperimentale!**
 - Lo sviluppo iterativo non è prototipazione
 - È una parte del prodotto finale, con una buona qualità del software

Sprint

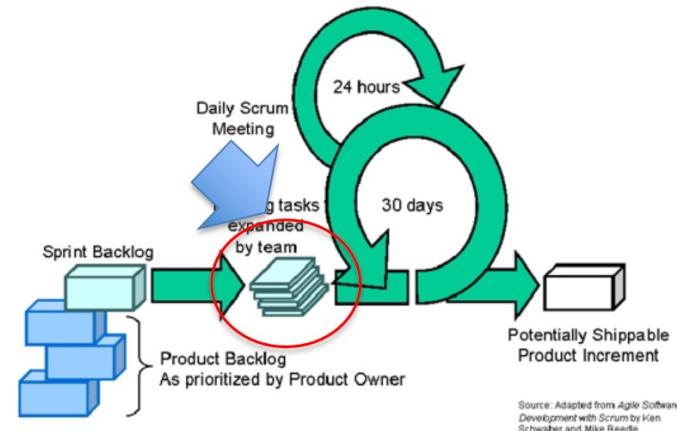


Scrum: attività di sviluppo

- Backlog (elenco delle cose da fare)
 - Un elenco a priorità dei requisiti del progetto o delle caratteristiche che rappresentano un valore per il cliente
- Sprint
 - Focalizzato su una porzione del backlog – 30 giorni max
- Riunione Scrum
 - Riunioni brevi (15 min) quotidiane - Scrum Master
 - Che cosa hai fatto dall'ultima riunione scrum?
 - Che ostacoli stai affrontando?
 - Che cosa pensi di ottenere prima della prossima riunione?
- Demo
 - Dimostrazione e consegna dell'incremento software al cliente
 - Non necessariamente contiene tutte le funzionalità pianificate, ma quelle che possono essere fornite nella finestra temporale

Sprint planning meeting

- It generates:
 - A sprint **GOAL**
 - List of **team members**
 - The **sprint backlog**
 - **Demo date**
 - **Time and place** for the daily meetings
- The product owner **HAS** to attend



Source: Adapted from Agile Software Development with Scrum by Ken Schwaber and Mike Beedle.

Artefatti

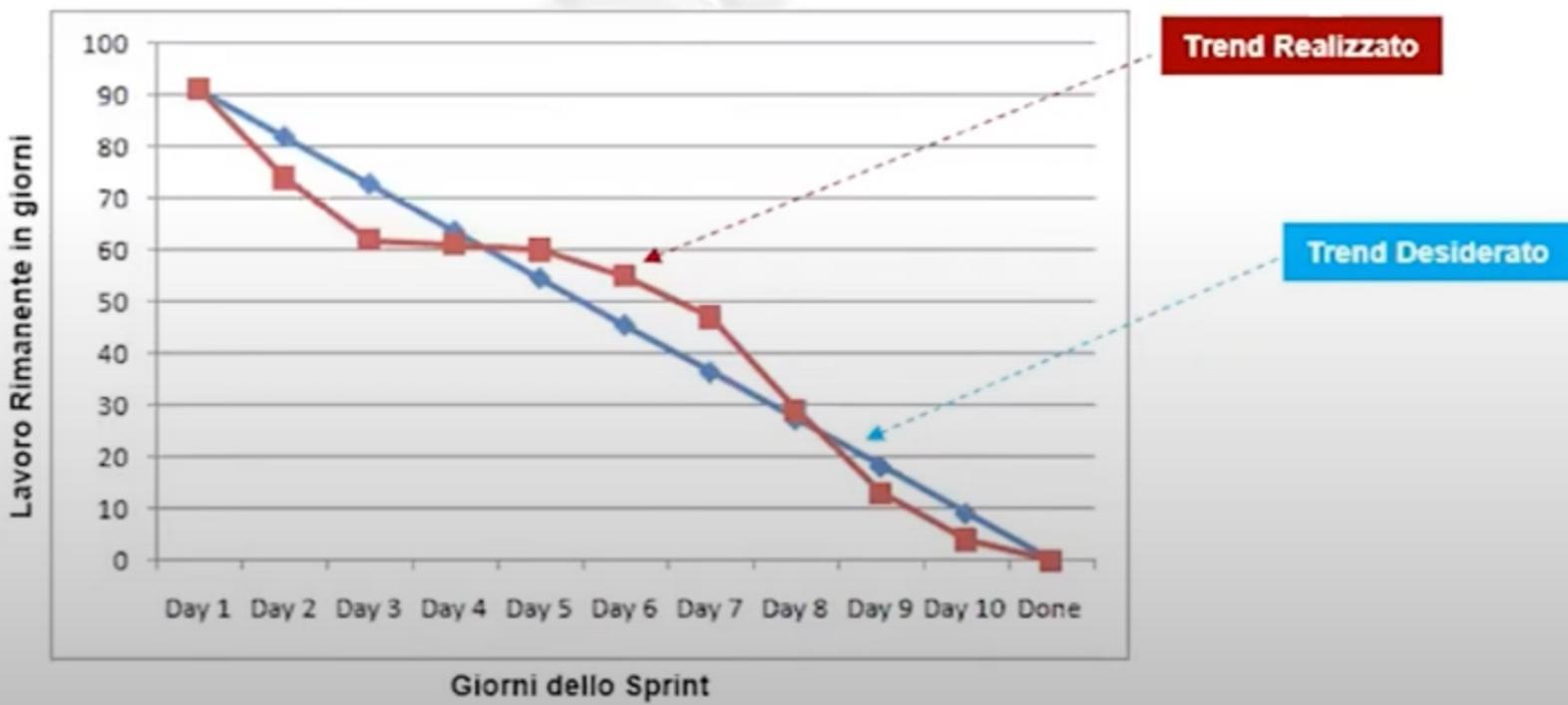
Product Backlog	Documento ad alto livello per l'intero progetto. Contiene i backlog item : descrizioni di tutte le caratteristiche richieste, lista dei desiderata e così via, con priorità assegnate in base al valore di business
Sprint Backlog	Documento che contiene informazioni su come il team sta procedendo nell'implementare le funzioni per lo sprint a venire
Burn-down Chart	Grafico mostrato in pubblico che rappresenta il lavoro che deve essere ancora completato nello sprint backlog. Aggiornato ogni giorno, dà una visione semplice e immediata del modo in cui procede lo sprint

Pratiche

Daily Scrum	ogni giorno durante lo Sprint, viene tenuta una riunione di verifica sullo stato del progetto
Post-scrum	consente a gruppi di team di discutere del loro lavoro, concentrandosi su aree di sovrapposizione e d'integrazione
Sprint Planning meeting	è frequentato da Product Owner, Scrum Master, intero Scrum Team, e da tutti i manager del caso interessati o della clientela
Sprint Review	presenta agli stakeholder il lavoro completato ("la demo")
Sprint Retrospective	consente un continuo miglioramento del processo in base a un principio di "inspect & adapt"

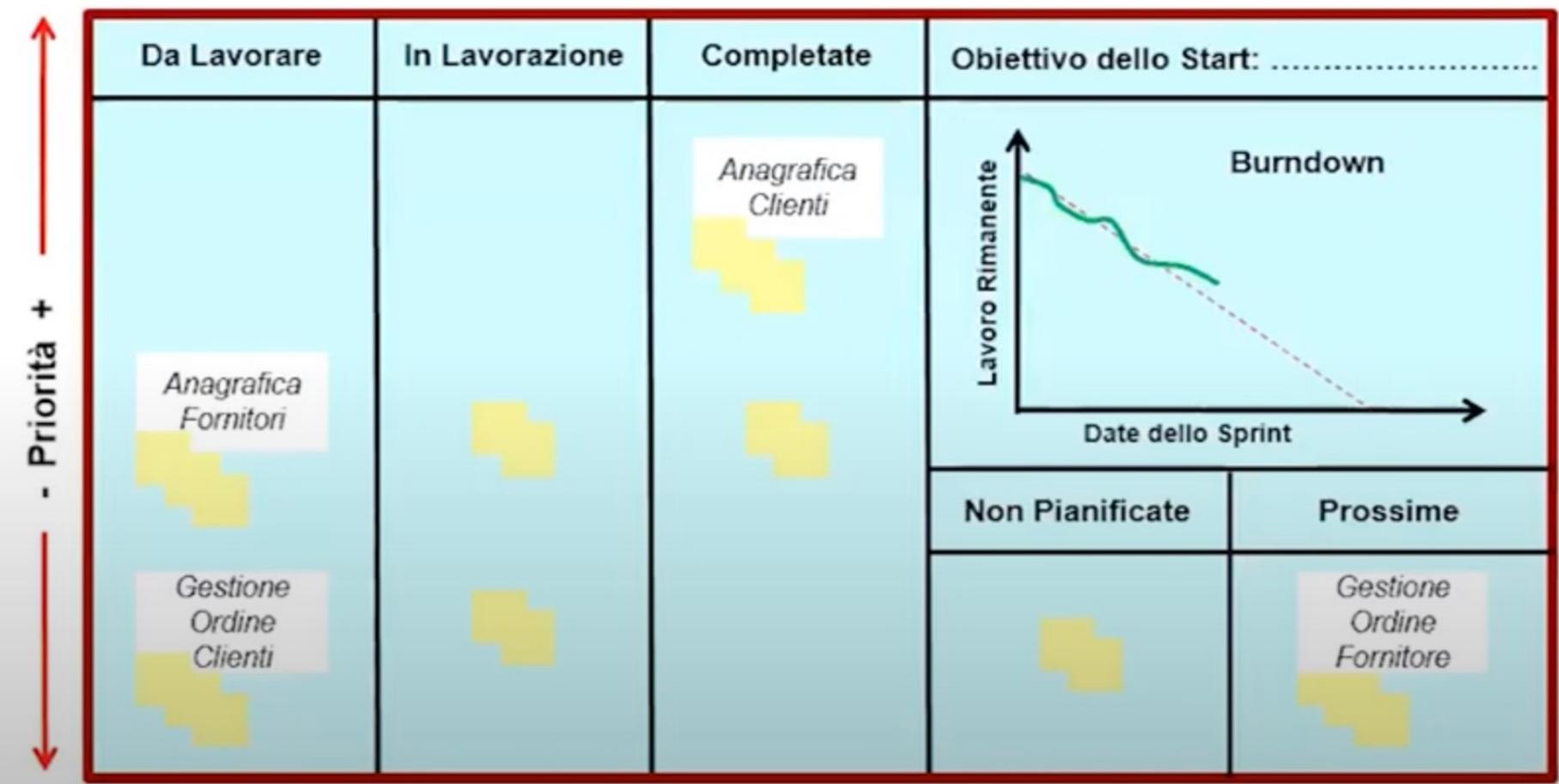
Il Burndown Chart

Il **Burndown Chart** è un diagramma cartesiano il cui scopo è quello di monitorare il lavoro ultimato rispetto a quello da completare: sull'asse X sono indicati i giorni dello Sprint (date oppure giorni), mentre sull'asse Y sono indicati i giorni di Effort (lavoro rimanente).



Lo Sprint Backlog

Esempio di Taskboard:

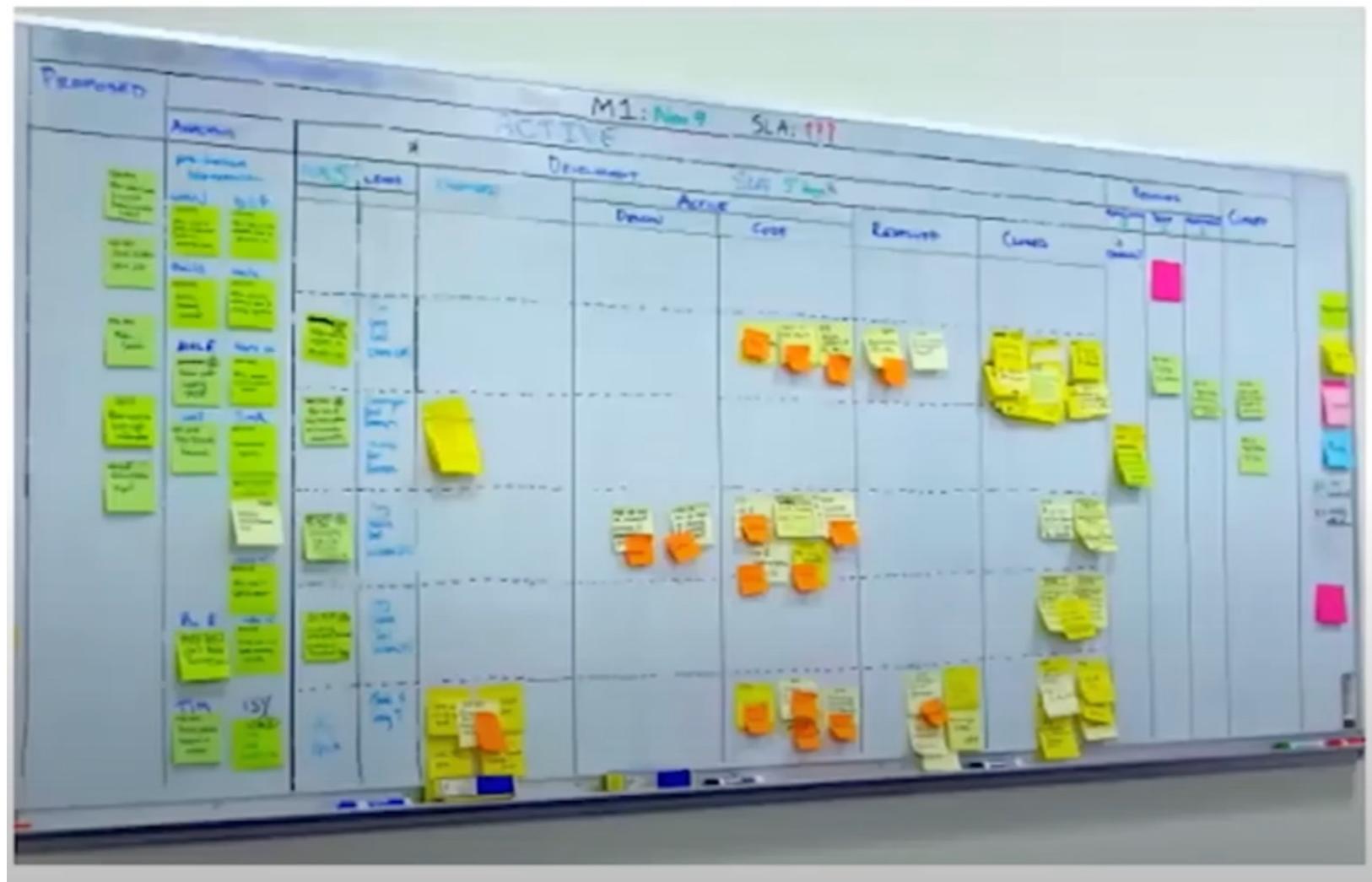


Sprint Retrospective

Esempio:

Cose Fatte Bene	Cose da Migliorare	Miglioramenti
		

Kanban



Differenza SCRUM - KANBAN

- SCRUM metodologia **Push**
 - Spinge all'interno in determinato periodo time box (sprint) una determinata quantita' di task da svolgere
- KANBAN metodologia **Pull**
 - Estrae dal proprio backlog in base al Working in progress cioe' in relazione a quello che siamo in grado di fare / realizzare