

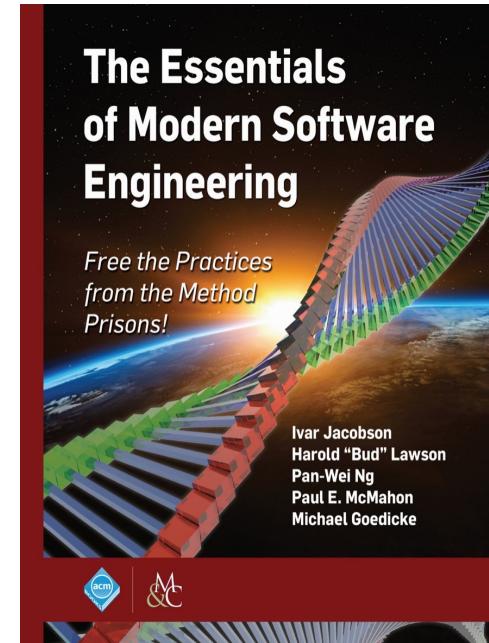
La retrospettiva in Scrum con Essence



*Corso di Ingegneria del Software
CdL Informatica Università di Bologna*

Agenda

- La retrospettiva dei team agili
- Essence: un linguaggio per processi e pratiche
- Uso di Essence per le retrospettive



Processo di sviluppo e retrospettiva

Un **processo di sviluppo** governa

- **Chi** deve fare **Cosa**
- **Quando** farlo
- **Come** raggiungere un determinato obiettivo

Una **retrospettiva** è una riunione periodica del team che analizza come sta andando il processo; è una delle più diffuse pratiche agili

Mettere in atto un processo di sviluppo del software

Quando più persone collaborano, è necessario seguire una disciplina di collaborazione, che chiamiamo «modello del processo di sviluppo» (alcuni chiamano tali discipline «metodi», altri «metodologie»)

I modelli del processo di sviluppo sono insiemi di ruoli, artefatti e «buone» pratiche

I modelli **agili** del processo di sviluppo sono stati inventati per piccoli team (3-7 persone), e includono un ruolo «Product Owner» (una persona) ed un ruolo «team» (l'insieme delle persone che collaborano, incluso il PO)

In Scrum c'è inoltre il ruolo di facilitatore (Scrum Master), che aiuta gli altri membri del team a seguire la disciplina di collaborazione

Nota bene: Quando c'è da costruire un grande sistema software i modelli agili occorre «scalarli», cioè adattarli, per poter coordinare team più grandi o molti più programmatore

Obiettivo: consolidare la competenza dei team

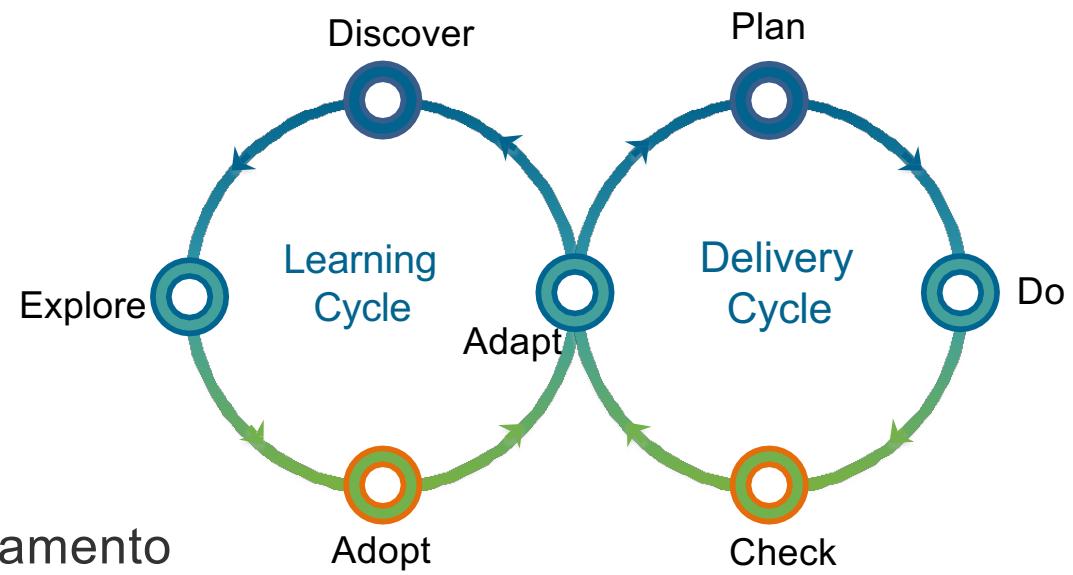
Motivo frequente di fallimento dei progetti agili:

Mancanza di competenza sufficiente nell'applicare tecniche base della visione agile.

Competenza

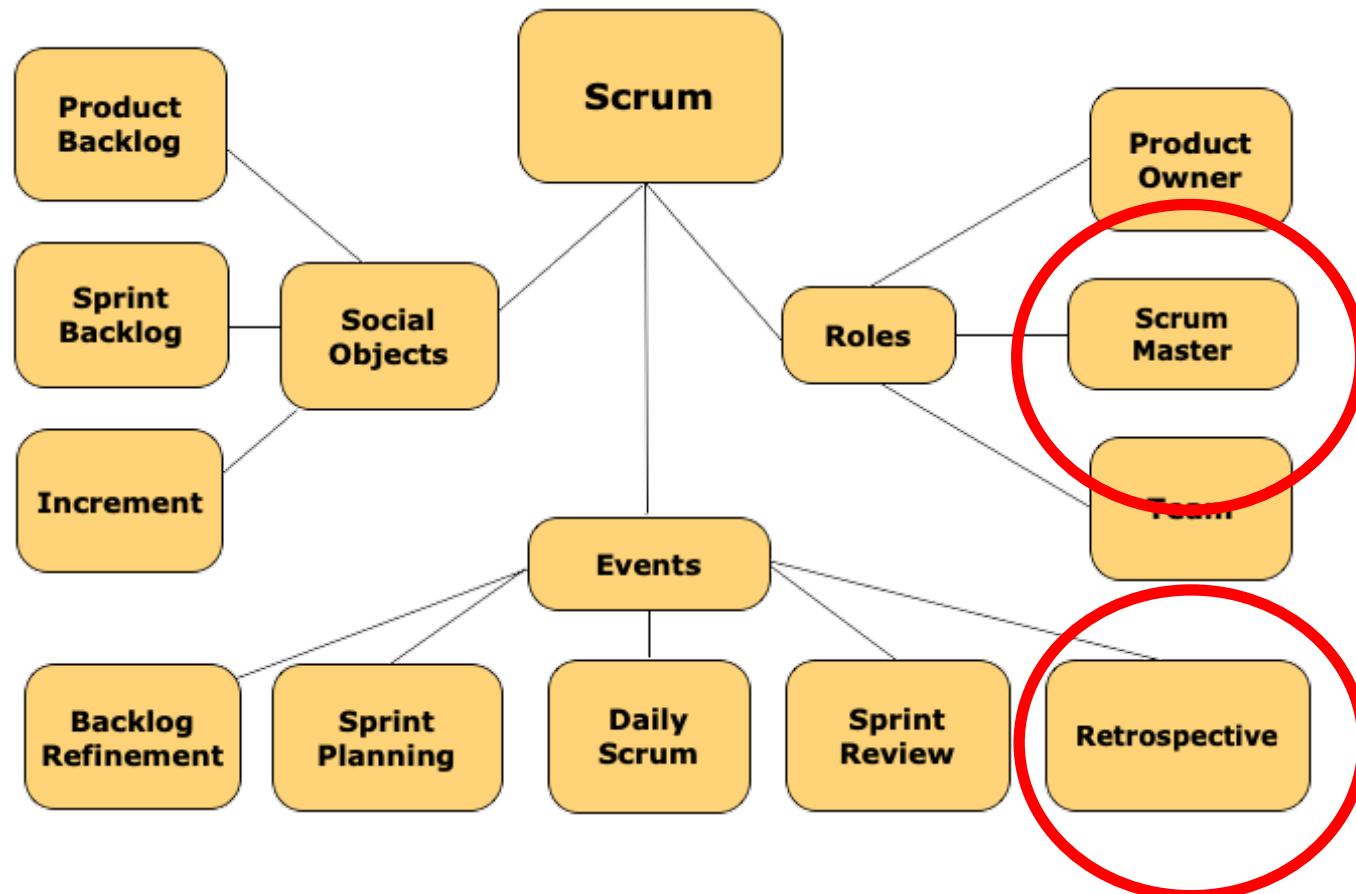
- Imparare una nuova pratica
- Guida nell'apprendimento della pratica
- Adozione della pratica
- Eventuale modifica della pratica

- Valori agili: trasparenza, adattamento
- Valori Scrum:
 - Autoorganizzazione del team
 - Team cross-funzionale
 - Miglioramento continuo del processo

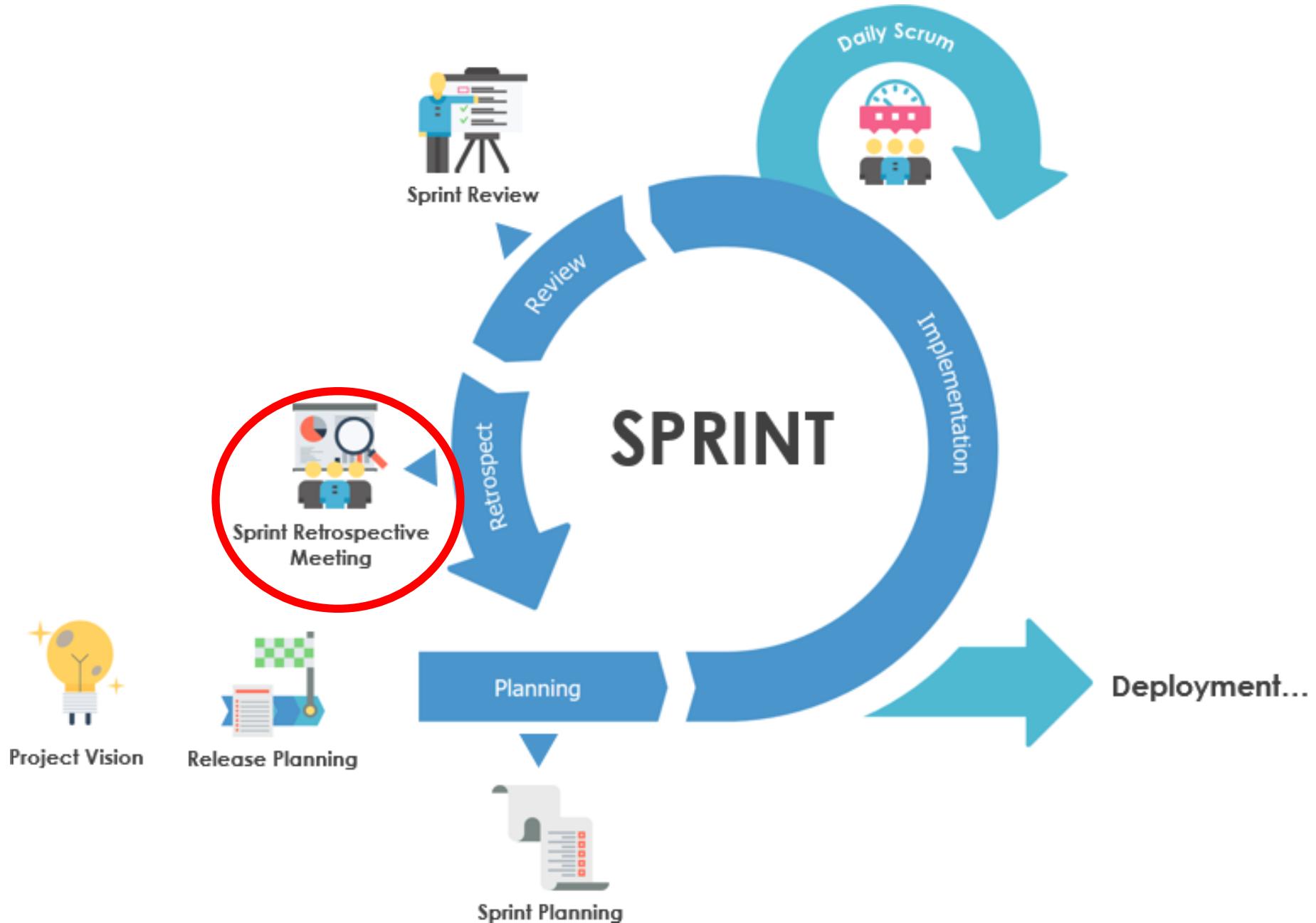


Il doppio ciclo del team agile:
apprendere / sviluppare

La retrospettiva è uno degli eventi del modello Scrum



Quando si fa la retrospettiva? Alla fine di ciascuno sprint



Cos'è una retrospettiva

12 Principio agile: *ad intervalli regolari, il team riflette su come diventare più efficiente, quindi rivede e modifica il proprio comportamento di conseguenza*

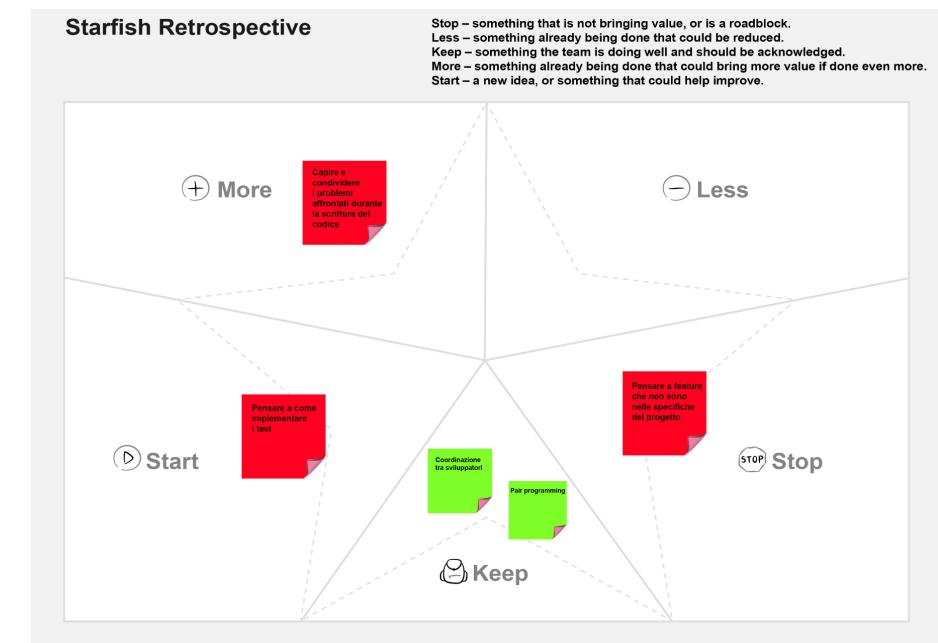
La retrospettiva è una riunione del team in cui si analizza il risultato dello sprint dal punto di vista del processo.

Si tiene come ultimo atto dello sprint

Lo Scrum Master *facilita* la retrospettiva, in quanto SM è il *process owner* e deve aiutare i colleghi del team a rivedere cosa è andato bene e cosa è andato male durante l'ultimo sprint

Domande a tutti:

Cosa è andato bene? Cosa ti è piaciuto?
Cosa è andato male? Cosa NON ti è piaciuto?
Cosa dobbiamo fare di diverso?



<https://metroretro.io/templates/the-starfish-retrospective>

Cos'è una retrospettiva

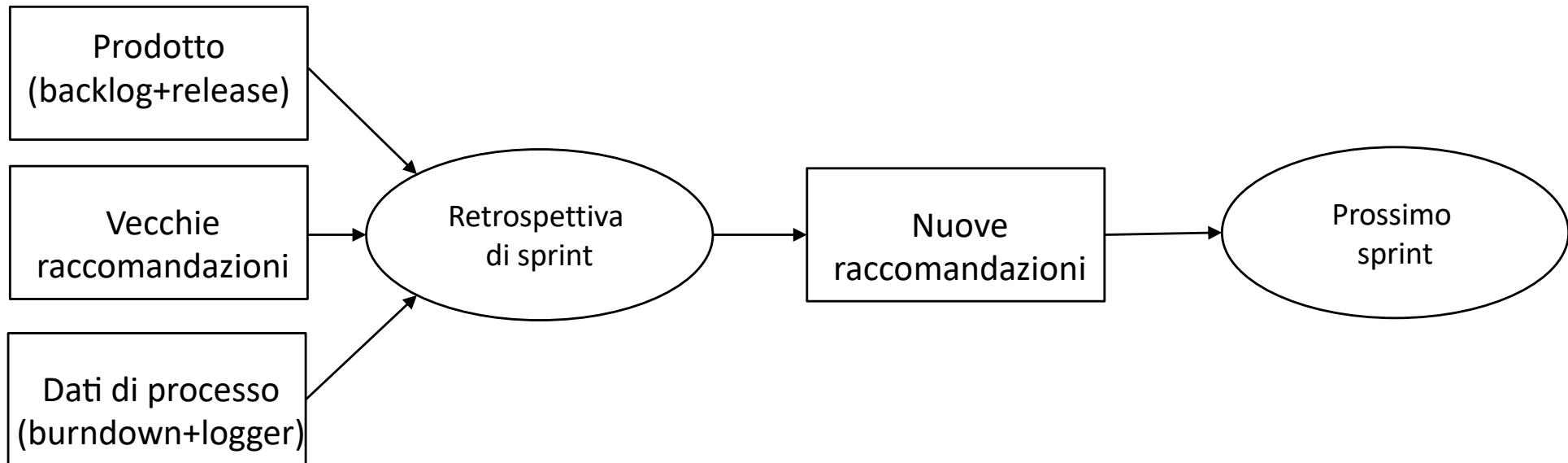
Starfish Retrospective

Stop – something that is not bringing value, or is a roadblock.
Less – something already being done that could be reduced.
Keep – something the team is doing well and should be acknowledged.
More – something already being done that could bring more value if done even more.
Start – a new idea, or something that could help improve.



Obiettivi della retrospettiva

1. Identificare vincoli e sprechi nel processo di sprint (analizzando i dati disponibili).
2. Identificare ciò che funziona o non funziona.
3. Favorire la conversazione tra i membri del team.
4. Ottenere idee per migliorare.
5. Redigere un piano d'azione per l'implementazione soluzioni raccolte.



Modello di una retrospettiva

Ruoli coinvolti

Il Product Owner:

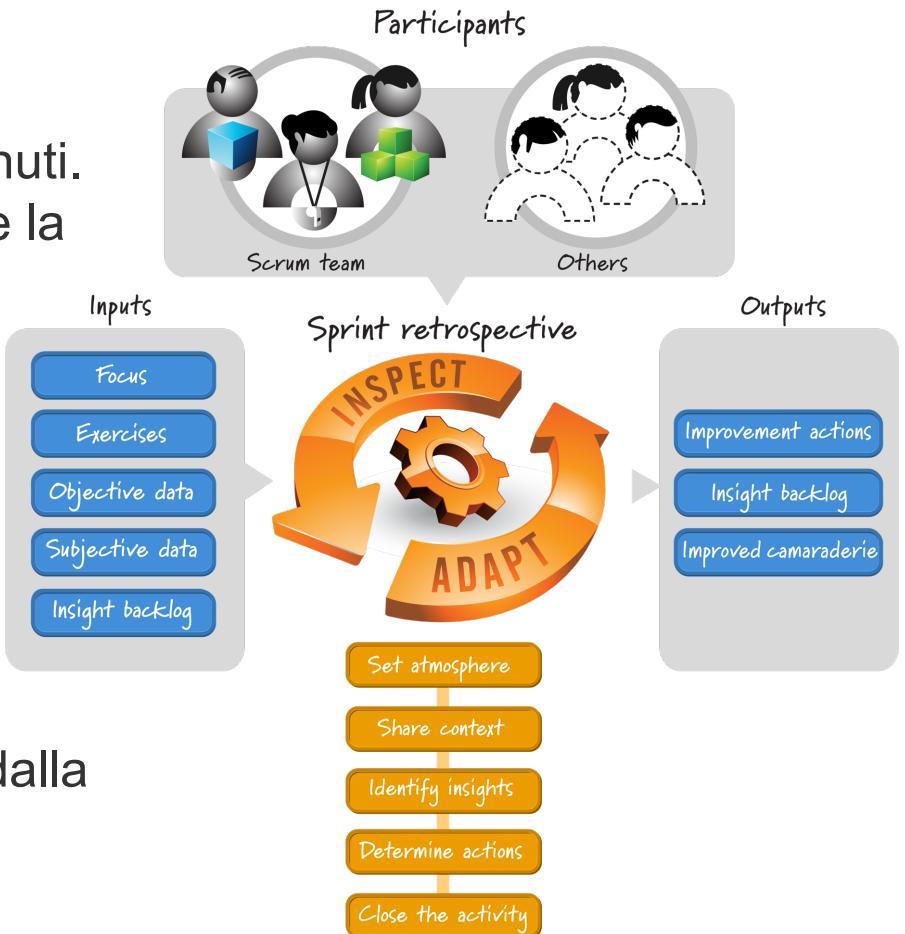
1. Riassume le sue valutazioni sui risultati ottenuti.
2. Richiede suggerimenti al team per migliorare la performance del team stesso
3. Offre supporto per risolvere i problemi organizzativi.

Lo Scrum Master (process owner):

1. Organizza e modera l'incontro.
2. Identifica i temi principali da trattare.
3. Ricorda i risultati di precedenti sprint.
4. Tiene traccia delle proposte che emergono dalla discussione.

Il Team di sviluppo:

1. Propone idee / lamentele / soluzioni
2. Definisce le priorità delle idee / soluzioni da attuare.
3. Concorda su come attuare le proposte più importanti.
4. Identifica i compiti per l'implementazione delle proposte selezionate.



Copyright © 2012, Kenneth S. Rubin and Innolution, LLC. All Rights Reserved.

Conversazioni strutturate

Nello sprint planning il team sceglie le pratiche da usare durante lo sprint

Le pratiche scelte possono essere modificate da uno sprint all'altro (ma meglio non modificarle *durante* lo sprint)

Nella retrospettiva ciascuna pratica viene analizzata e valutata dai membri del team (rosso=male, verde=bene)

Essence puo aiutare il team a farsi un'opinione (vedere slide successive)

CARDS	HI PO	ISM	DEV	EV	DEV	Motivation
Scrum Master	Red	Yellow	Yellow	Green	Yellow	poco aiuto nella gestione della distribuzione del lavoro per le API
Product Owner	Yellow	Green	Green	Green	Green	
Scrum Team	Yellow	Green				
Developers	Red	Yellow				poco impegno rispetto al lavoro svolto
Product Backlog	Yellow	Yellow	Yellow	Yellow	Yellow	
Sprint Planning	Red	Yellow	Yellow	Yellow	Yellow	Poca pianificazione
Daily Scrum	Red	Yellow	Red	Red	Red	Non è stato fatto con costanza
Sprint Goal	Yellow	Green	Yellow	Green	Green	Va definito all'inizio dello sprint
Self Management	Red	Red	Red	Red	Red	Poca comunicazione durante il lavoro

Prima retrospettiva

Poco tempo dedicato all'aiuto e alla gestione della distribuzione del lavoro da parte dello Scrum Master
I devs hanno dedicato troppo poco tempo allo sviluppo delle funzionalità scelte

Pianificazione superficiale del lavoro durante lo Sprint Planning

Daily Scrum fatto senza costanza

Sprint Goal definito ad inizio sprint poco chiaro
Scarsa comunicazione durante il lavoro

CARDS	HI PO	ISM	DEV	EV	DEV	DEV	Motivation/Ressourcen
Scrum Master							
Product Owner							
Developers	Yellow						
Scrum Team	Green	Yellow					
Product Backlog							
Sprint Planning					Yellow	Green	
Sprint Goal							Obiettivo sovraccarico rispetto alla reale disponibilità del team
Self Management				Red			

Ultima retrospettiva

Aumento del tempo dedicato a questa fase (da meno di un'ora a più di 3 ore)
Miglioramento generale nei voti assegnati.

Le discussioni delle retrospettive precedenti hanno comportato un miglioramento tangibile dello spirito collaborativo del team

Miglioramento nella gestione della fase di retrospettiva
Autocritica di un DEV

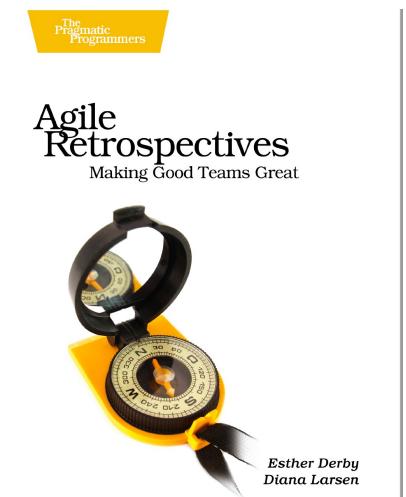
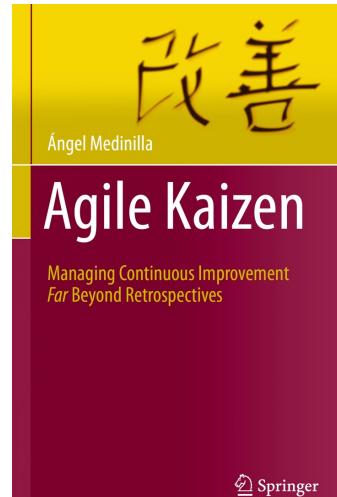
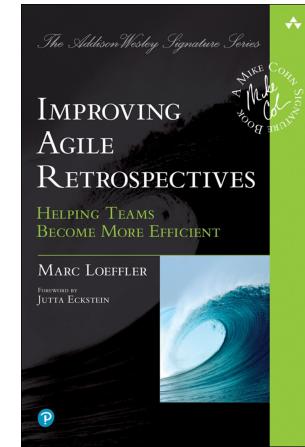
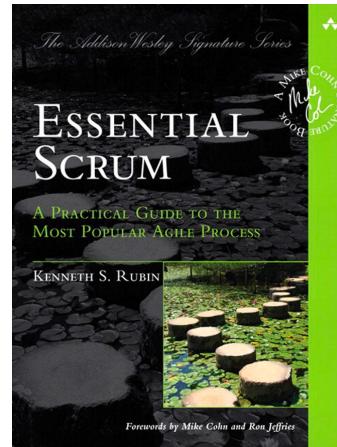
Riferimenti per retrospettive

Siti

<https://retromat.org/>

<http://retrospectivewiki.org>

<https://www.tastycupcakes.org>



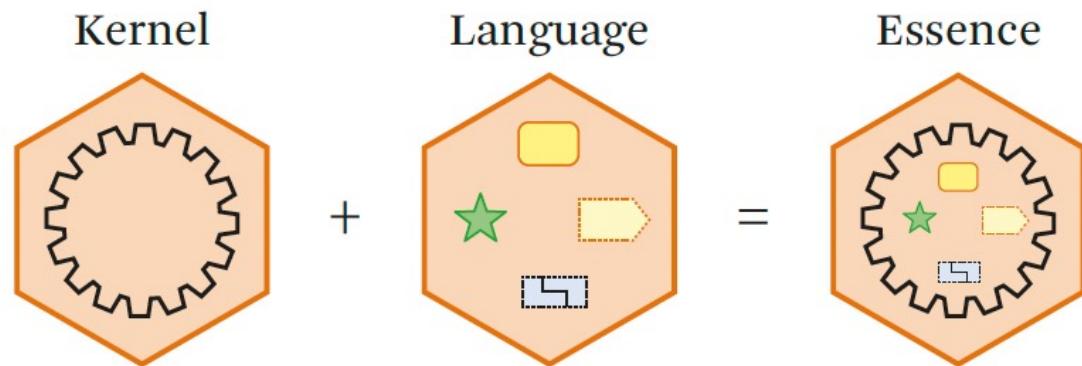
Retrospettive: uso di Essence

Per aiutare lo SM a gestire le retrospettive suggeriamo l'uso di Essence

Essence è un (meta)linguaggio di descrizione di metodi e pratiche di sviluppo

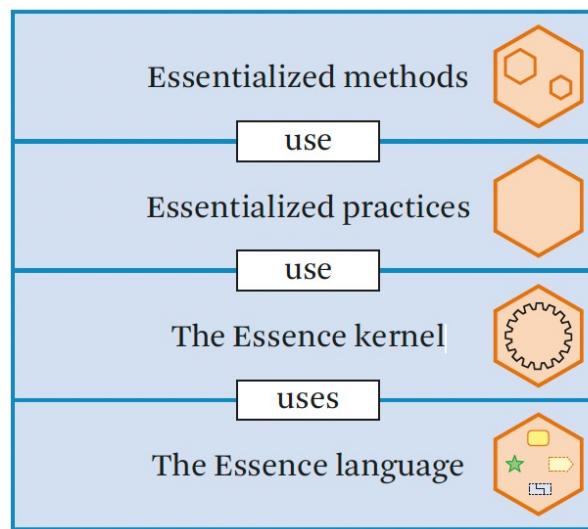
Descrizione tecnica

- Si concentra sull'essenziale
- componenti:
 - The Essence Language
 - The Essence Kernel
 - Pratiche essenzializzate (eg TDD)
 - Metodo (eg Scrum): combinazione di pratiche



Scopo

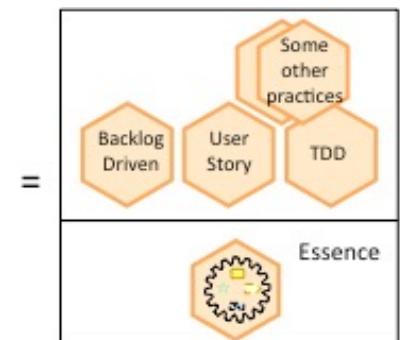
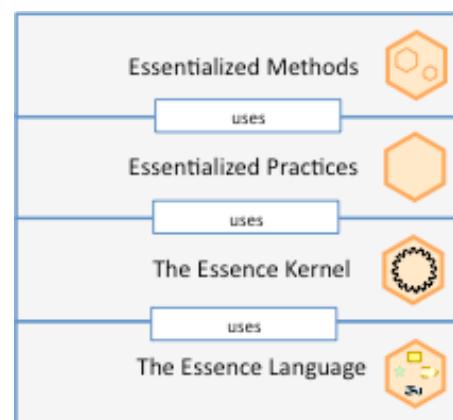
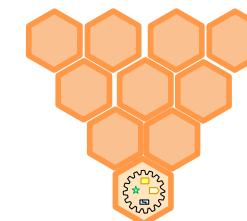
- Serve a riflettere sul processo
- Crea occasioni per conversare
- Insieme di giochi «seri» legati allo sviluppo del software



L'obiettivo di Essence

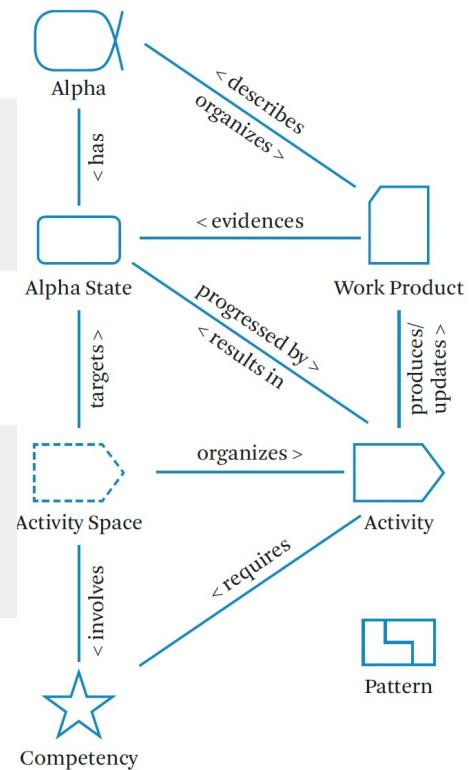
- Essence si concentra sugli aspetti essenziali del processo di sviluppo, cioè sulle buone pratiche e come vanno sviluppate
- Supporta l'auto-addestramento del team mediante carte checklist che permettono di «giocare» migliorando l'armonia del team
- **Le pratiche vengono rese indipendenti dal metodo in cui sono state definite**
- I team possono costruire il proprio metodo componendo le pratiche preferite

I metodi sono composizioni di pratiche



Gli elementi di base di Essence e le loro relazioni

Element Type	Syntax	Meaning of Element Type
Alpha		An essential element of the development endeavor that is relevant to an assessment of the progress and health of the endeavor.
Work Product		A tangible thing that practitioners produce when conducting software engineering activities.
Activity		A thing that practitioners do.
Competency		An ability, capability, attainment, knowledge, or skill necessary to do a certain kind of work.
Activity Space		A placeholder for something to do in the development endeavor. A placeholder may consist of zero to many activities.
Pattern		An arrangement of other elements represented in the language.



Gli alpha e i loro stati

☐ Opportunity

The set of circumstances that makes it appropriate to develop or change a software system.

- Identified
- Solution Needed
- Value Established
- Viable
- Addressed
- Benefit Accrued



1.1.1

☐ Stakeholders

The people, groups, or organizations who affect or are affected by a software system.

- Recognized
- Represented
- Involved
- In Agreement
- Satisfied for Deployment
- Satisfied in Use



1.1.1

☐ Requirements

What the software system must do to address the opportunity and satisfy the stakeholders.

- Conceived
- Bounded
- Coherent
- Acceptable
- Addressed
- Fulfilled



1.1.1

☐ Software System

A system made up of software, hardware, and data that provides its primary value by the execution of the software.

- Architecture Selected
- Demonstrable
- Usable
- Ready
- Operational
- Retired



1.1.1

☐ Work

Activity involving mental or physical effort done in order to achieve a result.

- Initiated
- Prepared
- Started
- Under Control
- Concluded
- Closed



1.1.1

☐ Team

A group of people actively engaged in the development, maintenance, delivery, or support of a specific software system.

- Seeded
- Formed
- Collaborating
- Performing
- Adjourned



1.1.1

☐ Way of Working

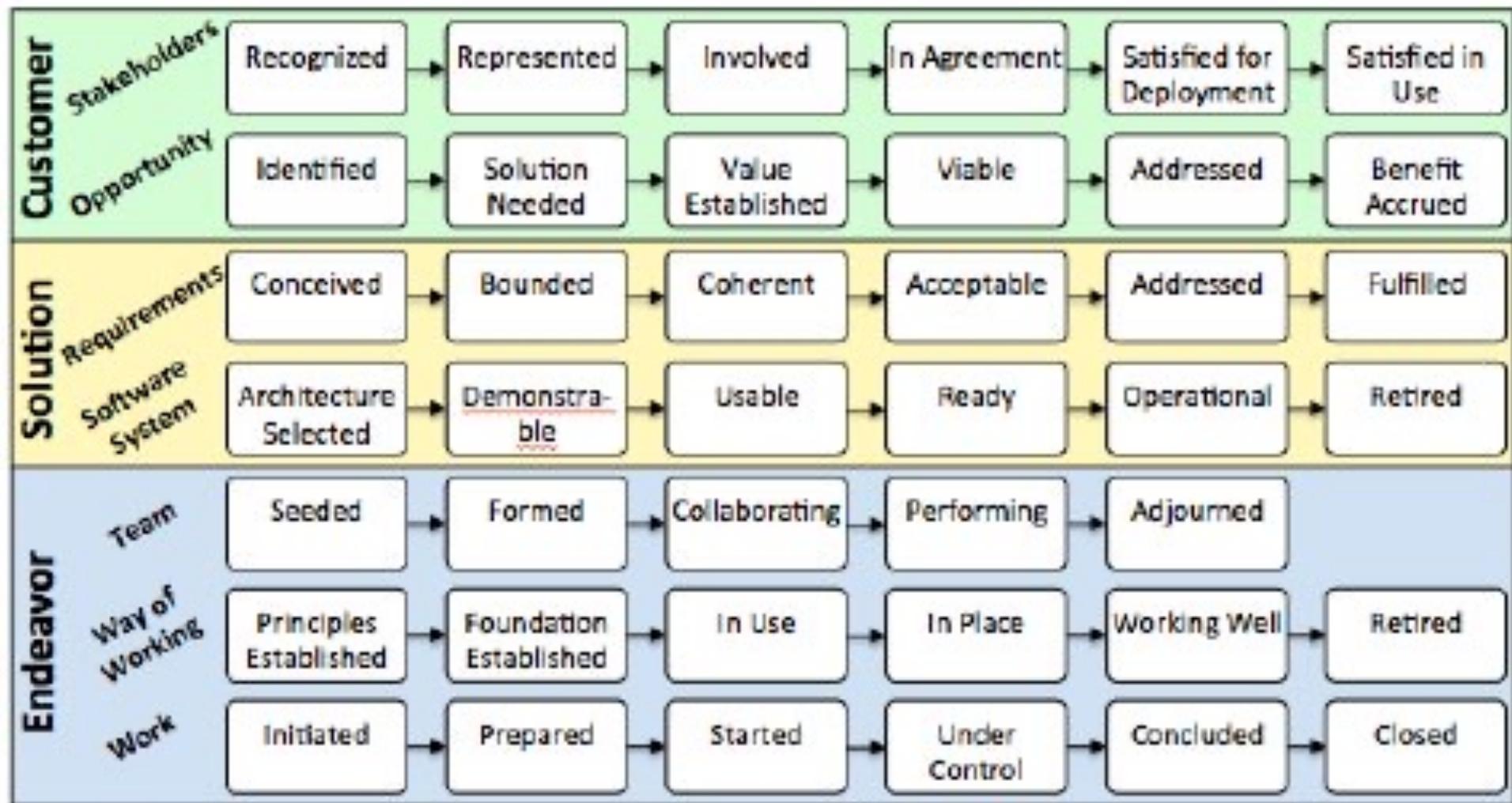
The tailored set of practices and tools used by a team to guide and support their work.

- Principles Established
- Foundation Established
- In Use
- In Place
- Working Well
- Retired



1.1.1

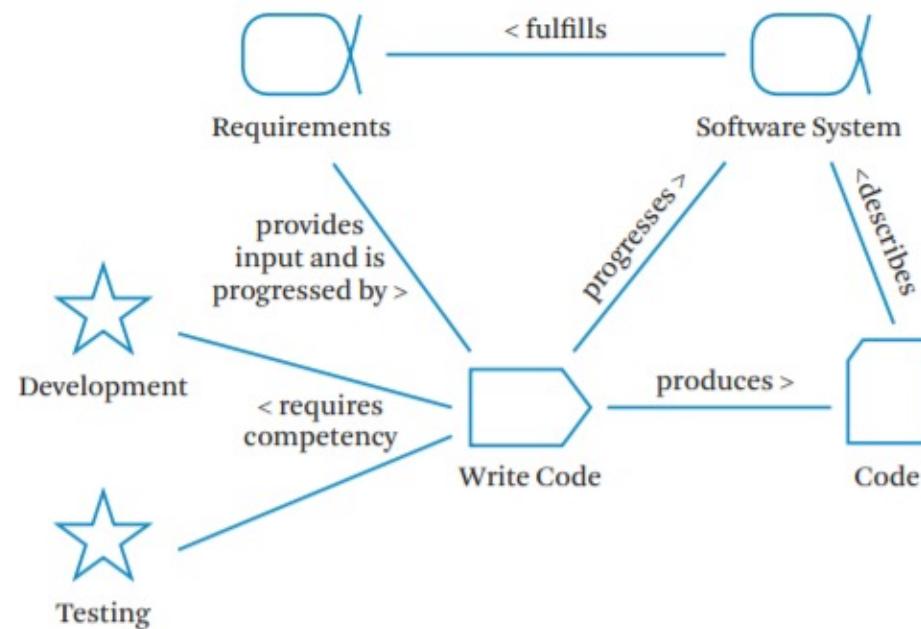
Gli alpha e i loro stati



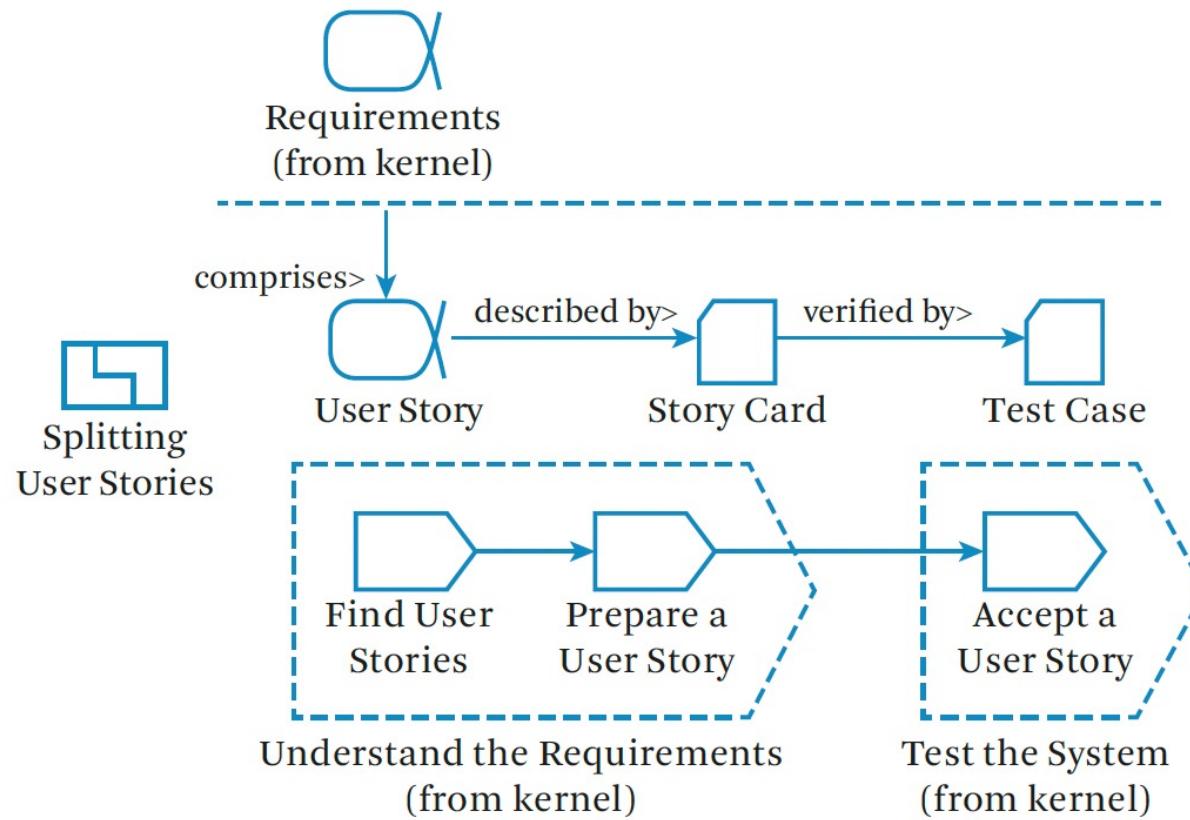
Esempio: essenzializzare un processo minimale di sviluppo

Questo diagramma mostra un processo “essenzializzato”.

La relazione tra i due alpha *Requisiti* e *Sistema software* è descritta dall’attività di scrittura di codice (*Write Code*), che produce l’artefatto Codice (*Code*), e che richiede due competenze: *Sviluppo* e *Testing*



Esempio: essenzializzare la scrittura di user story

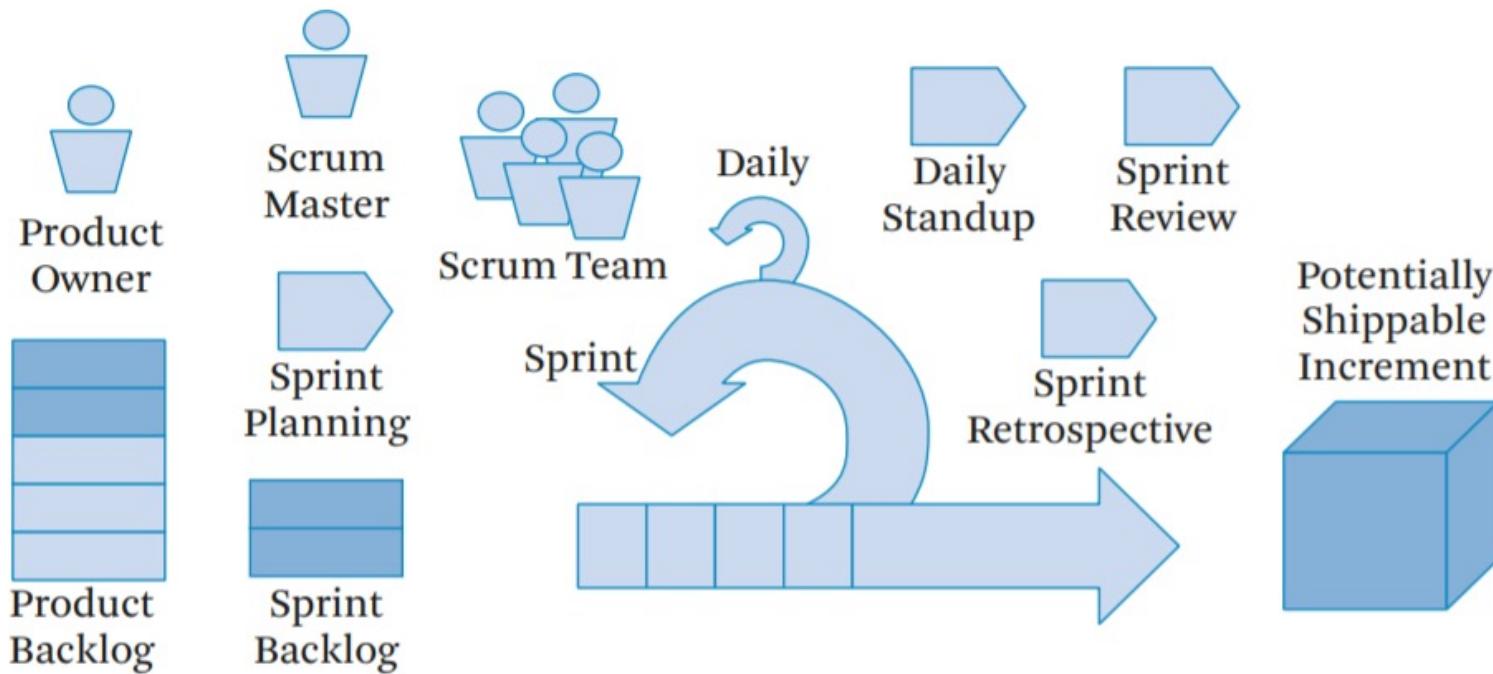


Un altro processo “essenzializzato”.

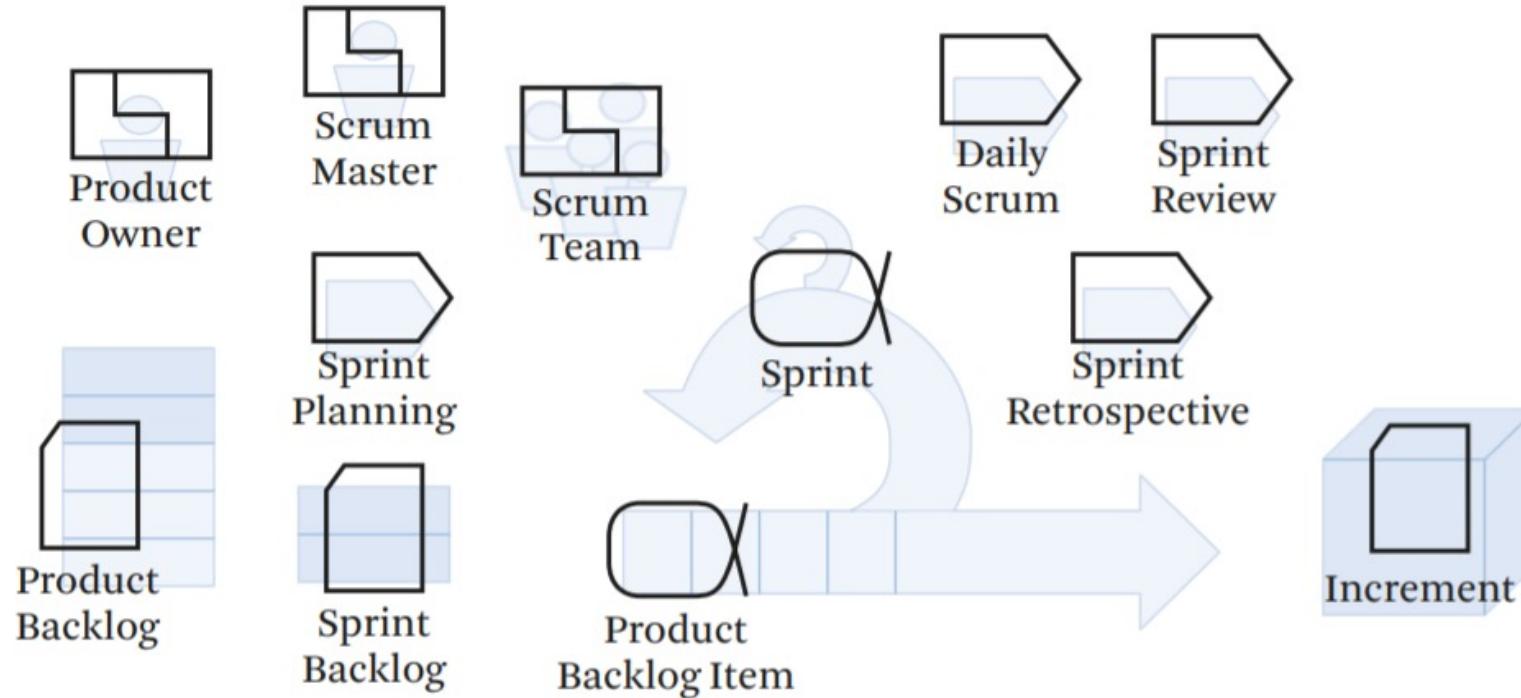
I requisiti diventano User Story, che vengono analizzate (attività *“Understand the requirements”*) e preparate per lo spazio di attività *Test the System* che include l’attività *Accept a User Story*.

Viene menzionato sulla sinistra anche il pattern “*Splitting User Stories*”

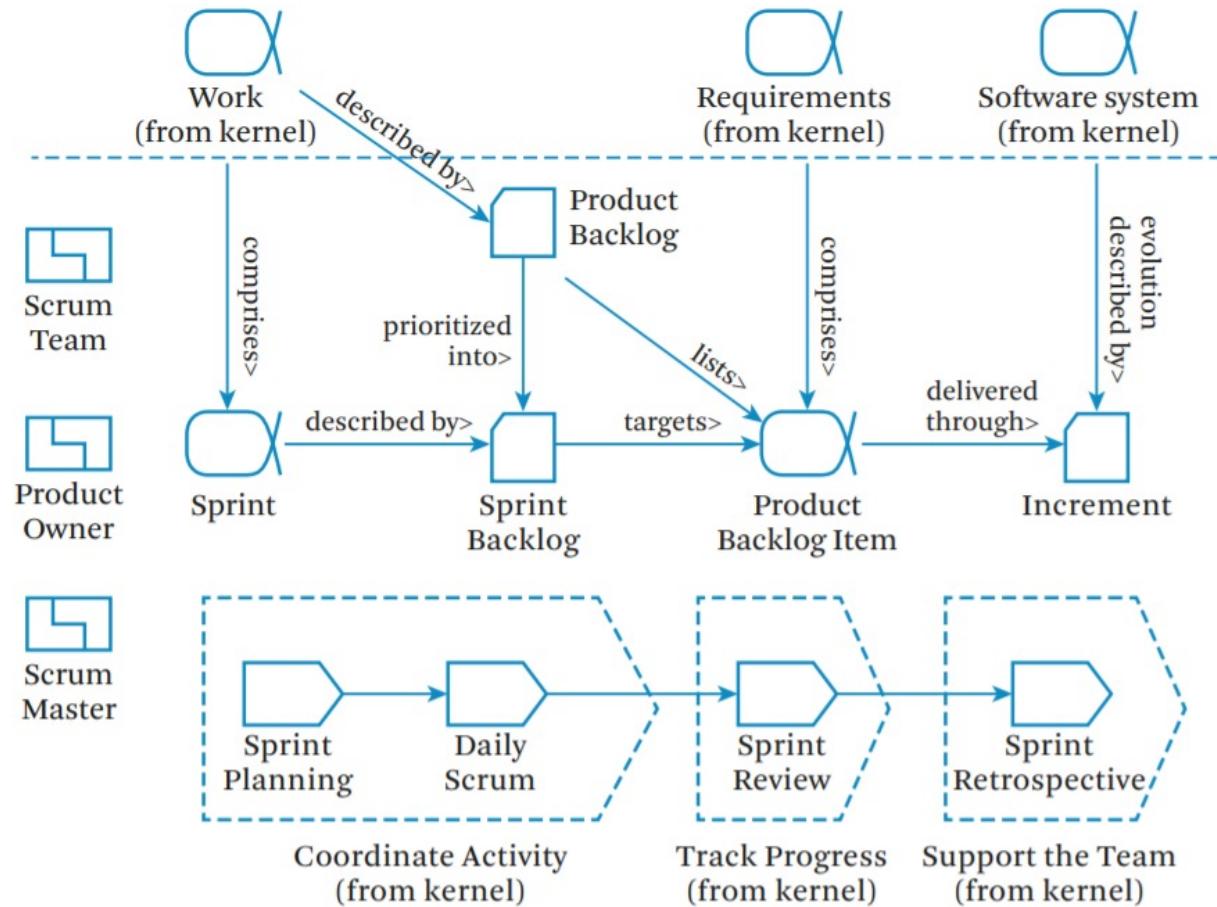
Gli elementi base di Scrum



Gli elementi di Scrum essenziali

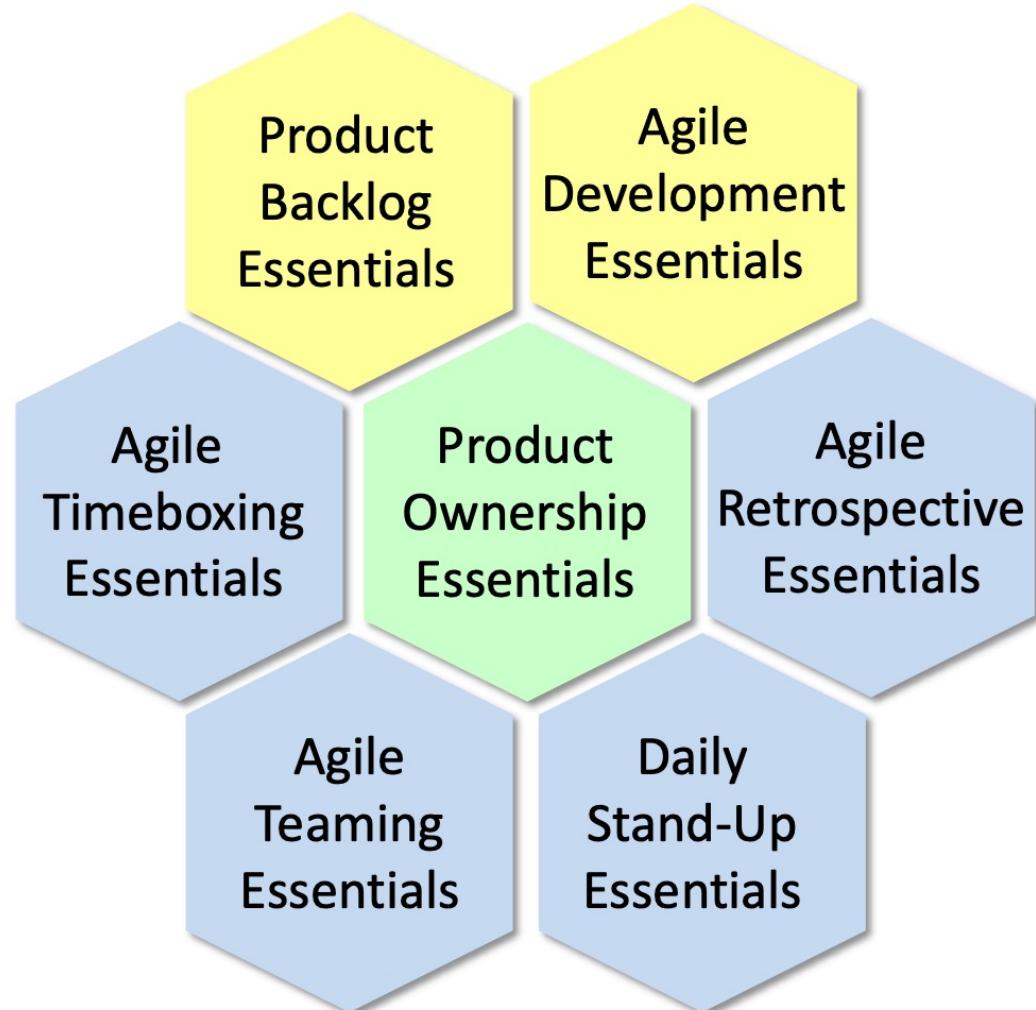


Modello di Scrum con Essence



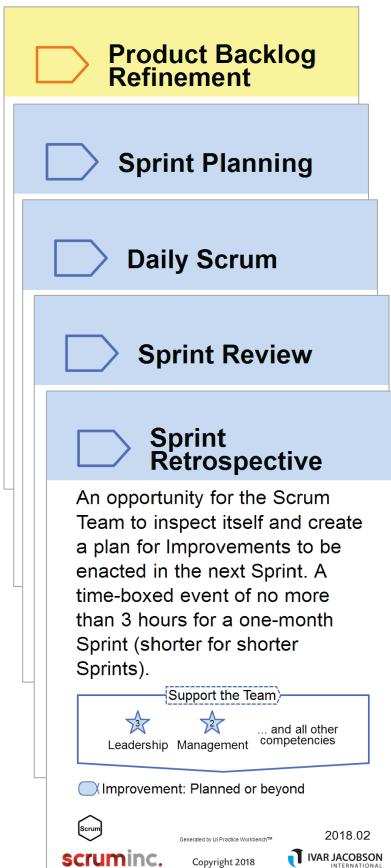
Le pratiche agili

Product backlog
Agile development
Agile timeboxing
Product Ownership
Agile retrospective
Agile teaming
Daily stand-up

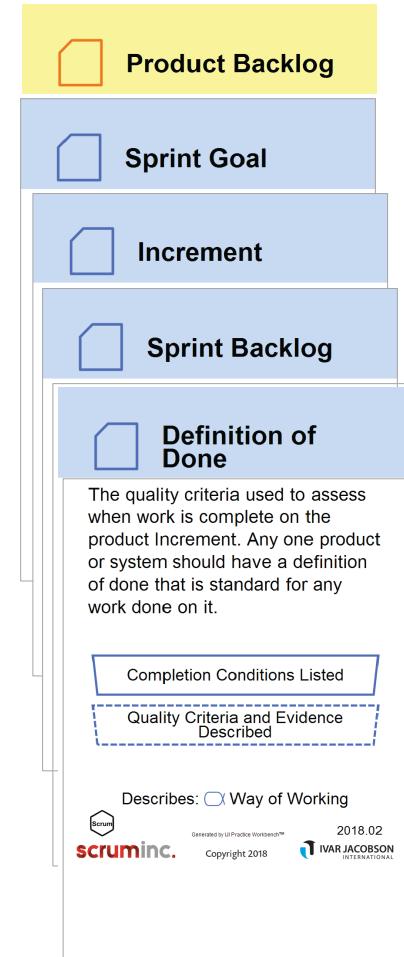
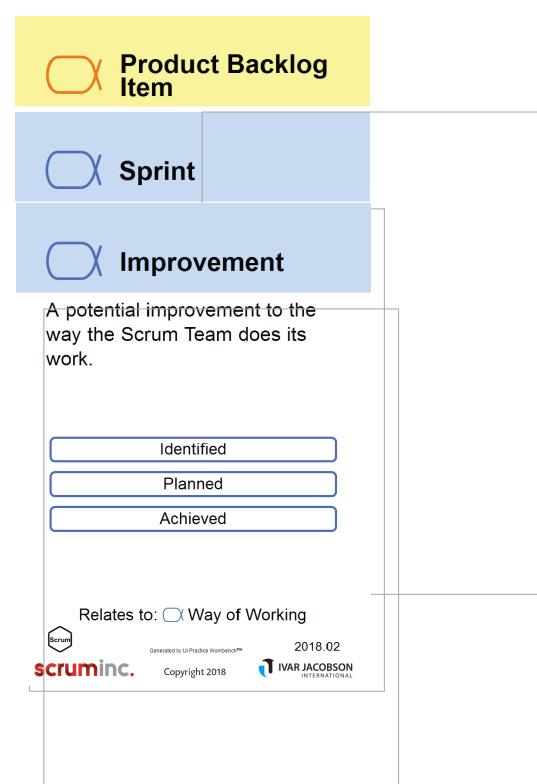


Carte delle pratiche di Scrum

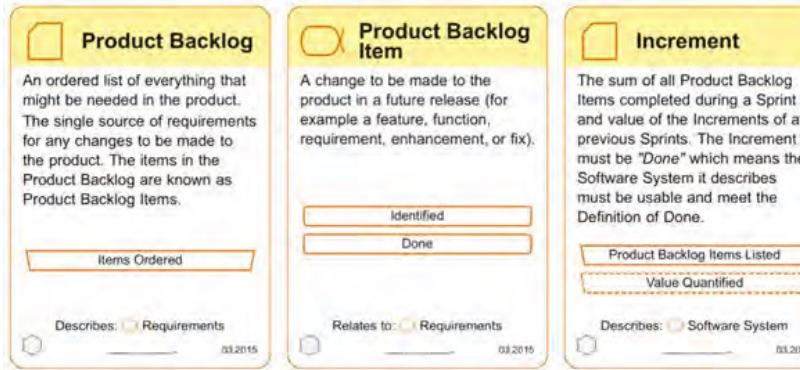
Pratiche (cose da fare)



Artefatti (cose con cui lavorare)



Esempio: Scrum Lite



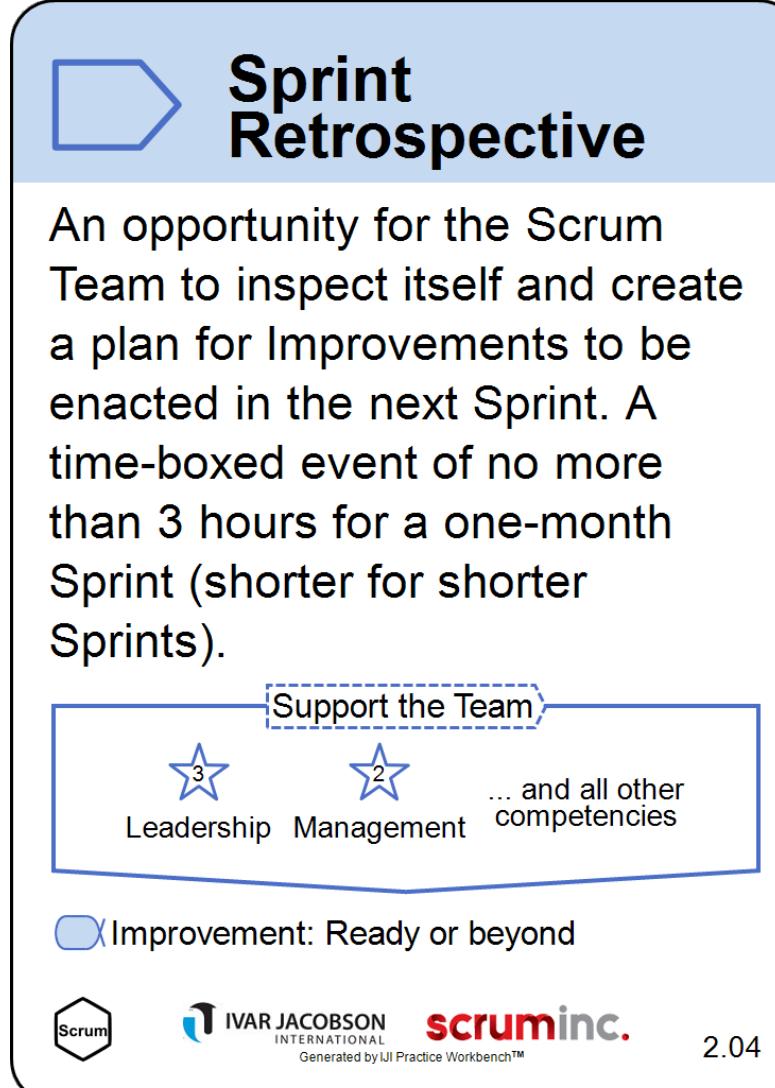
<https://iamhye.medium.com/guide-to-lite-agile-the-lightest-framework-for-individual-transformation-5b361afdebe2>

Carta Essence: retrospettiva

La retrospettiva è un incontro che costituisce un'opportunità per il team di ispezionare se stesso e di creare un piano di miglioramenti da eseguire nel prossimo sprint.

La retrospettiva è un incontro time-boxed di massimo 3 ore se lo sprint dura un mese

La retrospettiva è più corta se gli sprint sono più corti



The diagram is titled "Sprint Retrospective" with a blue arrow icon. It defines the retrospective as an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted in the next Sprint. It specifies a time-boxed event of no more than 3 hours for a one-month Sprint (shorter for shorter Sprints). A callout box labeled "Support the Team" lists competencies: Leadership (3 stars), Management (2 stars), and "... and all other competencies". Below the diagram is a blue circle icon with a checkmark, followed by the text "Improvement: Ready or beyond". At the bottom are logos for Scrum, Ivar Jacobson International, and scruminc., along with the version number 2.04.

Sprint Retrospective

An opportunity for the Scrum Team to inspect itself and create a plan for Improvements to be enacted in the next Sprint. A time-boxed event of no more than 3 hours for a one-month Sprint (shorter for shorter Sprints).

Support the Team

3 Leadership 2 Management ... and all other competencies

Improvement: Ready or beyond

Scrum IVAR JACOBSON INTERNATIONAL scruminc.
Generated by IJI Practice Workbench™

2.04

Carte Essence: attività di una retrospettiva

Agile Retrospective Essentials

Make incremental improvements to the way of working through regular, repeated retrospectives.

Mad, Sad, Glad Hold a Retrospective
 Improvement Resources

2018.09

Generated by UI Practice Workbench™

Hold a Retrospective

The whole team meets regularly to reflect on its way of working. Improvements are identified and prioritized, and actions agreed. At the next retrospective, the results are evaluated.

Improvement
 Support the Team
Leadership Management

Way of Working: Working Well (contributes to)
 Improvement: Action Agreed or beyond

2018.09

Generated by UI Practice Workbench™

Mad, Sad, Glad

A popular approach to team brainstorming to identify potential improvements. Team members write on sticky notes what has made them:

- *Mad* – frustrations
- *Sad* – disappointments
- *Glad* – things that went well

Part of its power is that it taps into people's emotions, and results in an unfettered flow of ideas that the team can then analyze, prioritize and action.

One Approach To: Hold a Retrospective
Ref: Mad, Sad, Glad

2018.09

Generated by UI Practice Workbench™

Improvement

A possible adaptation to improve a Team's Way of Working.

Identified
 Prioritized
 Action Agreed
 Trialed
 Results Evaluated
 In Use

Relates to: Way of Working

2018.09

Generated by UI Practice Workbench™

Gioco di retrospettiva: Progress Poker

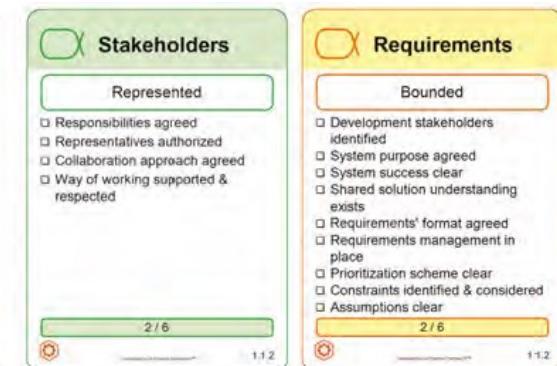
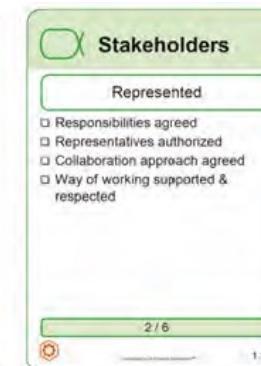
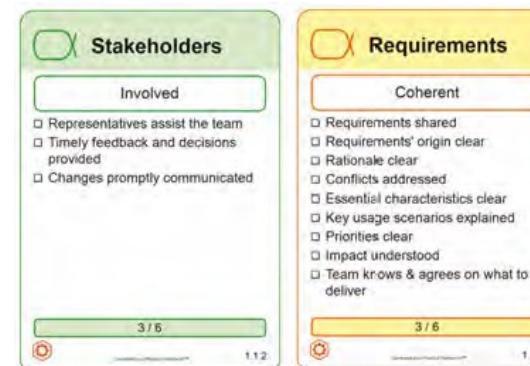
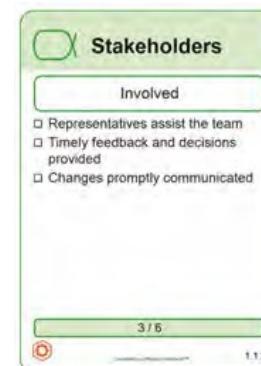
A seconda della fase, si scelgono alcune carte alpha e loro carte di stato e si cerca di decidere «cosa è stato fatto»

ogni membro del team dà una valutazione e spiega le proprie opinioni quando la valutazione differisce da quella dei compagni di team

Ogni membro del team deve riflettere e spiegare il motivo per cui ha valutato lo stato in quel modo



Carta dei requisiti con le sue carte di stato



Opinione di Paola

Opinione di Marcello

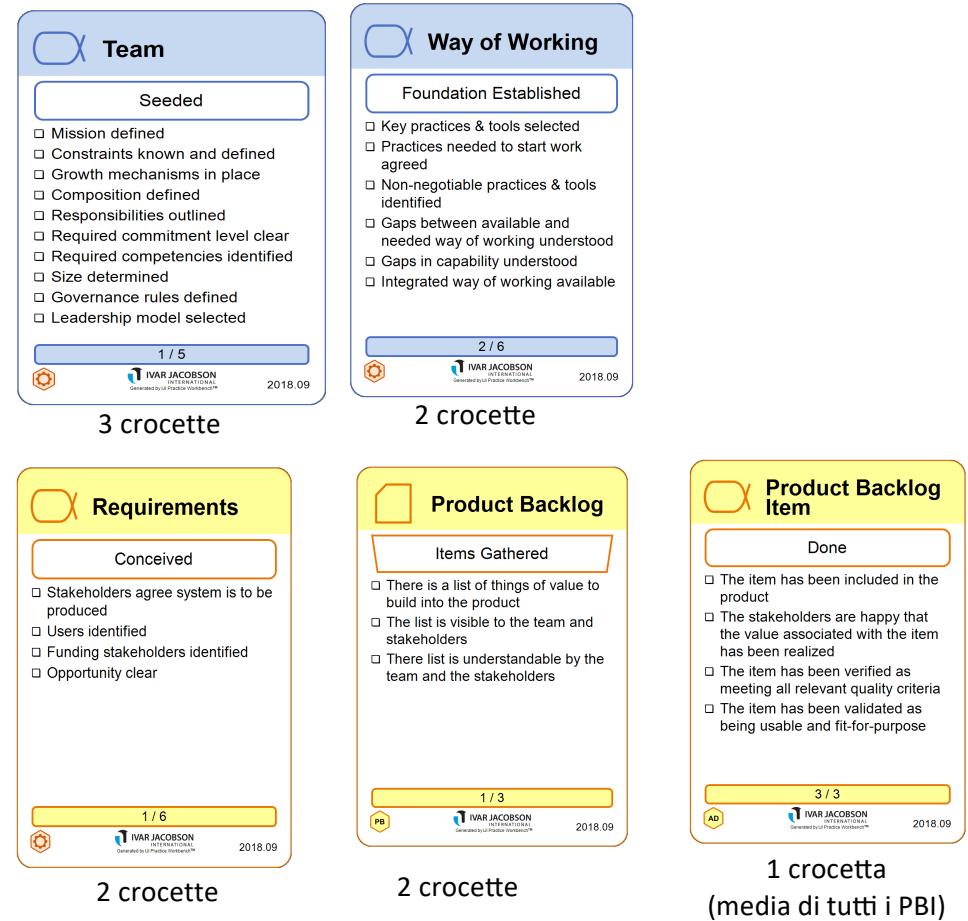
Esempi - Scenario 1 Bad Team

Bad team ha avuto una vita difficile. Le persone del gruppo si sono divise in due sottogruppi che si parlano raramente fra loro.

“Leader” ha assunto una posizione di potere, e ha svolto tutto il lavoro di setup del progetto da solo, mentre il secondo gruppo si è limitato a scrivere una decina di user story, senza chiedere nulla agli altri; senza stima di alcun tipo per cui non è dato avere stime di consegna o risultato.

La partita di Scrumble è stata frettolosa, è stata “persa” e il gruppo si è un po’ litigato.

Il sistema di sviluppo è stato abbozzato, nel senso che taiga è attiva, ma non ci sono documenti

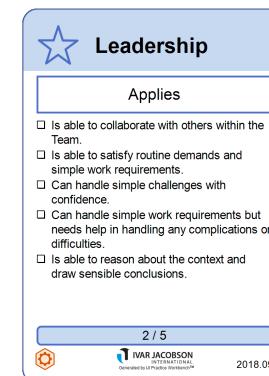


Average team

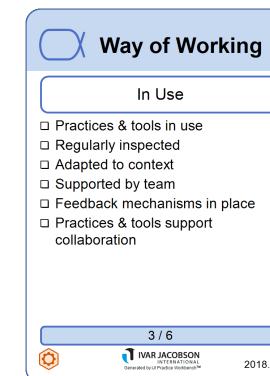
Average team ha essenzialmente fatto il suo dovere. “Leader” è stato eletto come coordinatore, e ha saputo riprendere alcuni membri del gruppo che tendevano a “sgarrare”; si sono creati due sottoteam, uno di competenze più web e uno di competenze più classiche (Java). È stato realizzato un progetto di prova a più mani usando git.

Il sistema di sviluppo è stato preparato, con Taiga online, Gitlab, e mattermost; le comunicazioni sono abbastanza frequenti. La partita di Scrumble è stata persa, ma è risultata utile, a detta dei partecipanti.

Il backlog è formato da una decina di item che seguono il pattern classico, e alcuni di essi, i più importanti, sono stati stimati usando un Planning Poker sotto la direttiva di “Analyst” che si è dimostrato il più attento a cogliere le problematiche del dominio



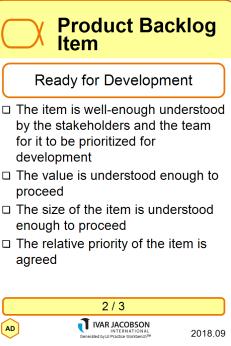
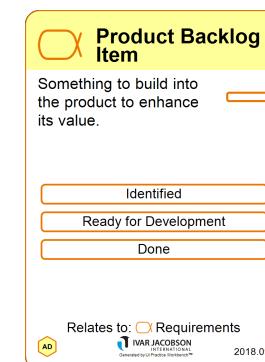
3 crocette



3 crocette



4 crocette

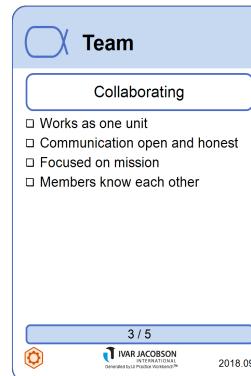


2 crocette
(media di tutti i PBI)

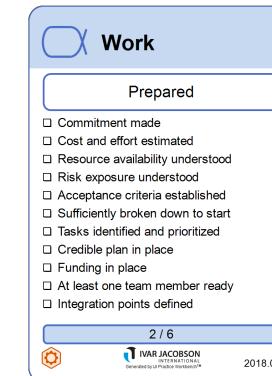
Dream team

Per Dream Team tutto va gonfie vele. Sotto la “dittatura illuminata” di Leader, il sistema di sviluppo è stato sviluppato nella sua interezza, identificando anche le tecnologia da usare. La partita di Scrumble è stata rivelatrice, e ha permesso di identificare le persone più adatte per i particolari ruoli.

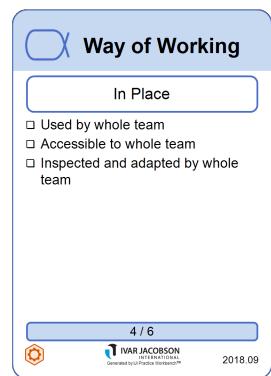
Il team ha compiuto un’analisi collegiale e ha realizzato due Epiche e una decina di user story, lavorando online tramite meet, mattermost e slack. Una prima versione di queste ultime sono state sottoposte agli stakeholder, che hanno fornito feedback interessanti, e provocato la modifica da un paio di storie. Di conseguenza, tutte le prime 8 storie sono stimate e messe in priorità. Su gitlab sono stati per ora realizzati due prodotti: un programma che raccoglie tutti i tweet relativi ad un #tag, e un programma Java per testare l’interfaccia utente di una sottosezione del programma



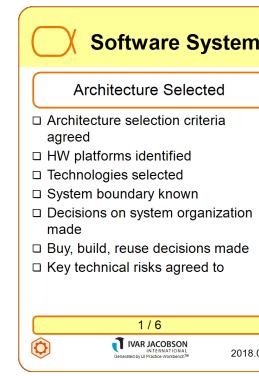
4 crocette



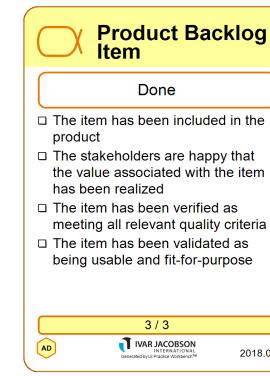
7 crocette



3 crocette



7 crocette



3 crocette
(media di tutti i PBI)

Gioco di retrospettiva: definiamo lo stato degli alpha



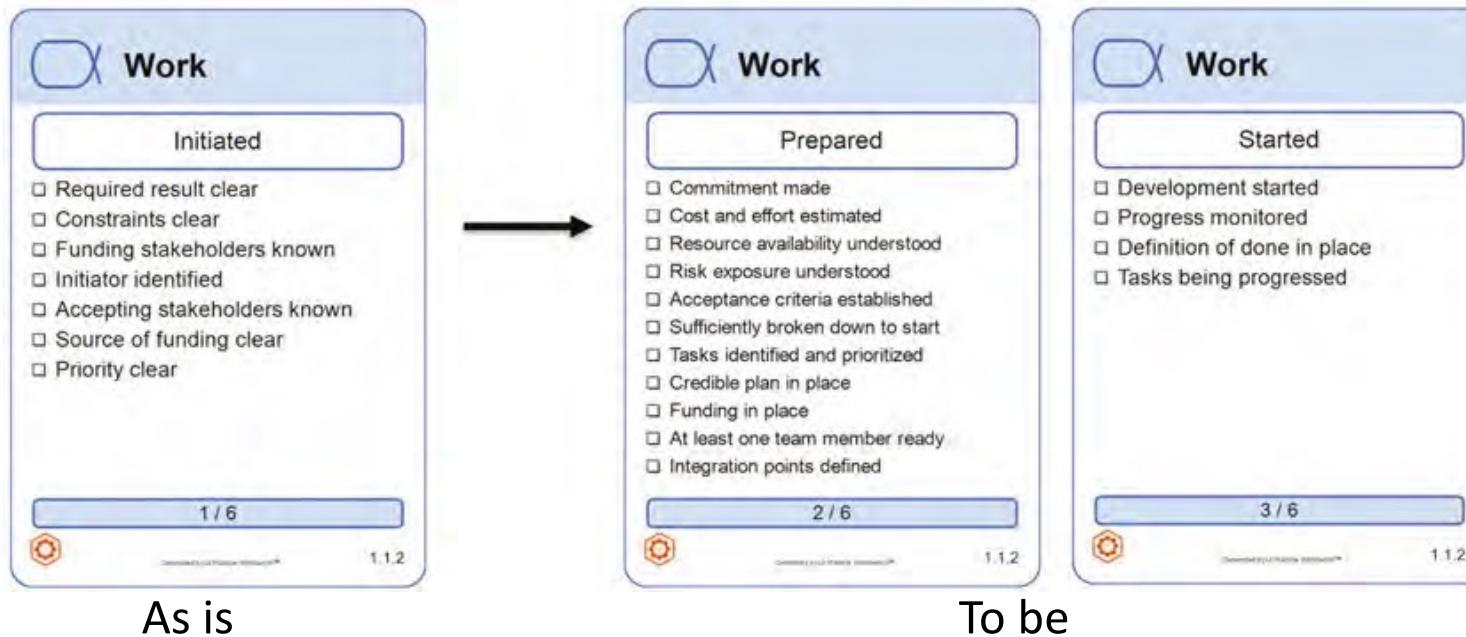
Prima e dopo la discussione
dello stato degli “stakeholder”

Gioco di retrospettiva: definiamo lo stato degli Alpha



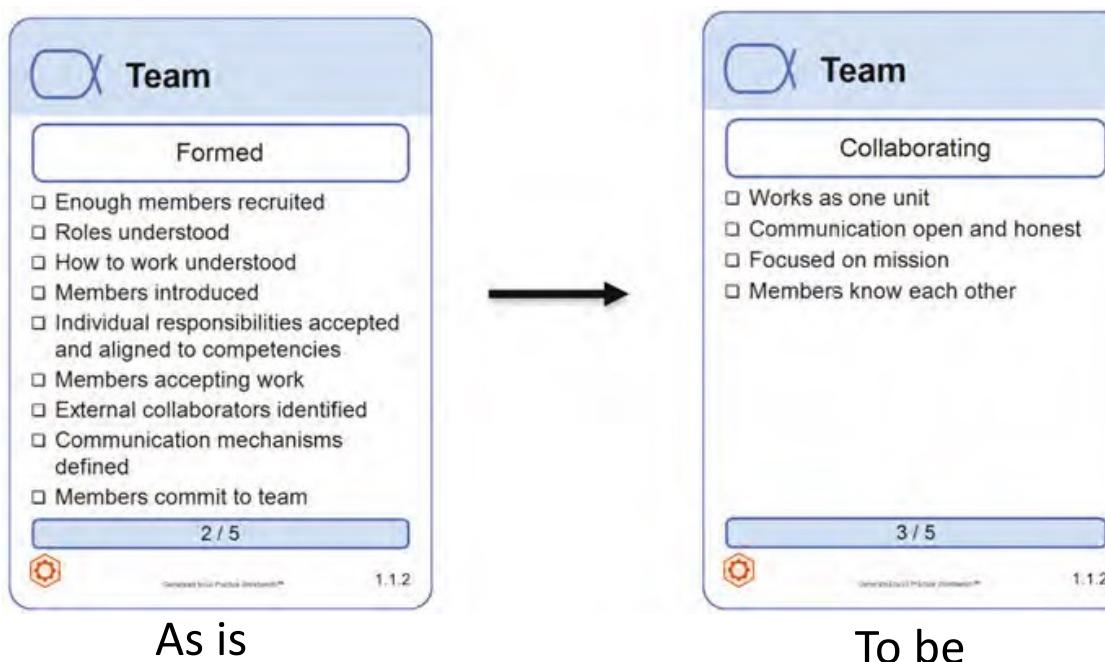
Posizioni definite di tutti gli alpha

Stati attuali ("as is") e stati desiderati ("to be")



As is

To be

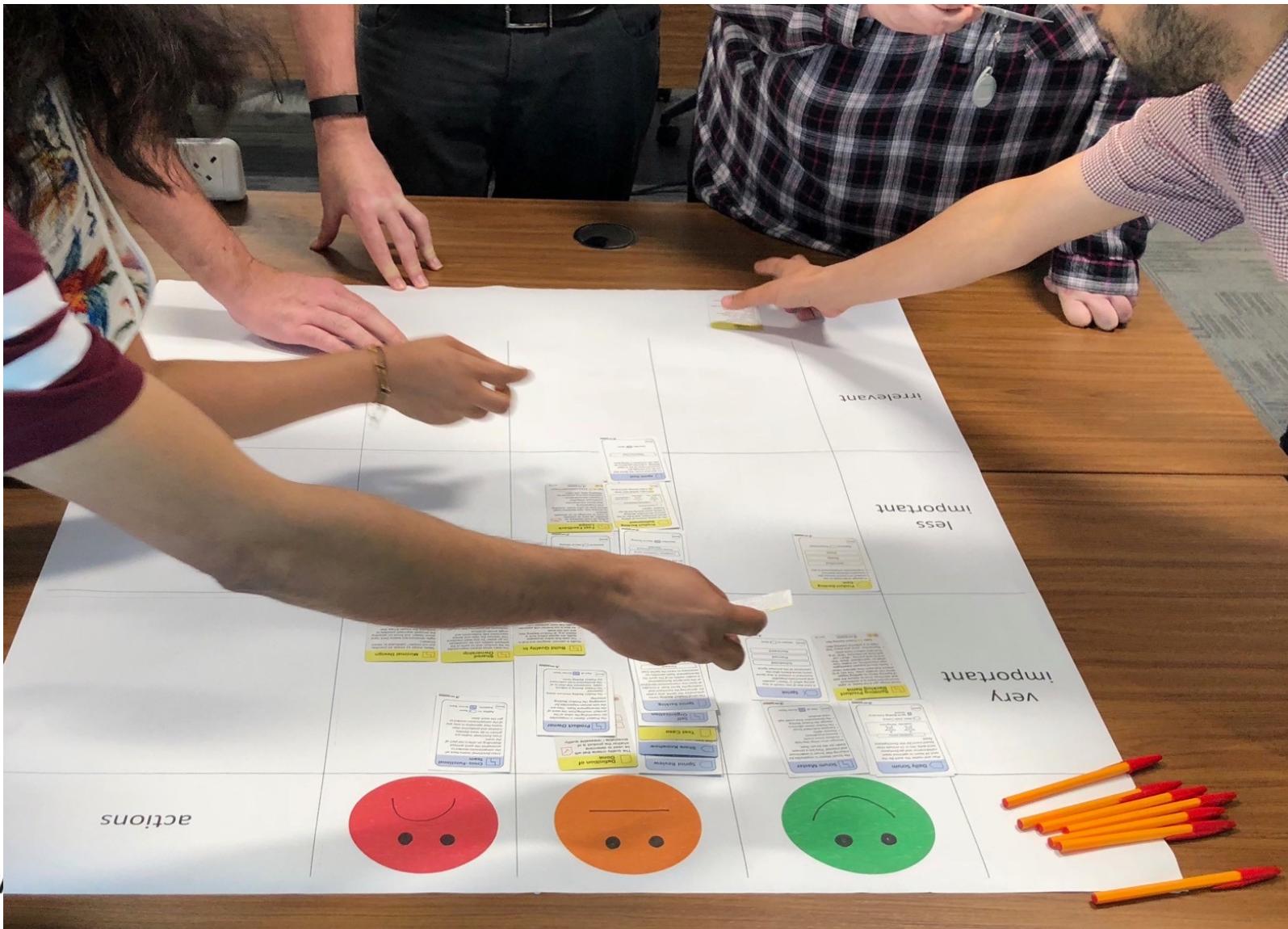


As is

To be

Gioco di retrospettiva: good mad sad

	<p>Sprint Backlog</p> <p>A highly visible, real-time picture of the work that the Developers plan to accomplish during the Sprint in order to achieve the Sprint Goal.</p> <ul style="list-style-type: none"> Primed with Improvements Sprint Goal Set Actionable Plan in Place Other Detail Captured <p>Describes: Sprint and Sprint Goal</p>    	<p>Product Goal</p> <p>A long-term objective for the Scrum Team that describes a desired future state of the product which can serve as a target for the Scrum Team to plan against. The team must fulfill (or abandon) one objective before taking on the next.</p> <ul style="list-style-type: none"> Identified Committed Fulfilled or Abandoned <p>Relates to: Requirements</p>    	<p>Sprint Goal</p> <p>The single objective set for a Sprint. It creates coherence and focus, encouraging the Scrum Team to commit to the goal.</p> <p>Idee non molto chiare su cosa volevamo implementare in questo sprint</p>    
	<p>Developers</p> <p>Developers are the people in the Scrum Team that are committed to creating any aspect of a usable Increment each Sprint. The Developers are accountable for:</p> <ul style="list-style-type: none"> Creating a plan for the Sprint, the Sprint Backlog Instilling quality by adhering to a Definition of Done Adapting their plan each day toward the Sprint Goal Holding each other accountable as professionals <p>Part of: Scrum Team</p>    	<p>Scrum Master</p> <p>The Scrum Master is accountable for ensuring that Scrum is understood and enacted. They are true leaders who serve the team in several ways:</p> <ul style="list-style-type: none"> Coaching self-management and cross-functionality Removing impediments Helping the Product Owner manage the Product Backlog effectively Helping the Team focus and create high-value products Helping the organization understand and enact Scrum <p>They are also accountable for the Scrum Team's effectiveness.</p> <p>Part of: Scrum Team</p>    	<p>Definition of Done</p> <p>A formal description of the state of the Increment when it meets the quality measures required for the product. It must be met by all Product Backlog Items. If there are multiple Scrum Teams working together on a product they must comply with the same definition of done.</p> <ul style="list-style-type: none"> Completion Conditions Listed Quality Criteria and Evidence Described <p>Describes: Way of Working and Product Backlog Item</p>     <p>divisione front/back impegnativa, ruoli che si "accavallano"</p>
	<p>Quantum Entanglement</p> <p>Dispersed teams are unable to work at the same level of productivity as co-located teams.</p> <p>Establish social, cultural and technical connections between the team members at the time the team is formed. Continually work to improve these connections.</p>    	<p>Increment</p> <p>A concrete stepping stone towards the Product Goal. A Sprint may produce multiple Increments. Each Increment is additive to all prior Increments. In order to provide value the Increment must be usable and meet the Definition of Done.</p> <ul style="list-style-type: none"> Product Backlog Items Listed Value Quantified <p>Describes: Sprint and Sprint</p>    	<p>Daily Scrum</p> <p>Plan and replan the work for the next 24 hours to optimize progress towards the Sprint Goal. A daily, 15-minute event for the Developers (including Scrum Masters and Product Owners) actively working on the Sprint Backlog.</p> <ul style="list-style-type: none"> Coordinate Activity Leadership Management <p>Work Under Control (contributes to)</p> <p> Sprint Backlog, Actionable Plan in Place or beyond</p>    



Giochi di retrospettiva con Essence

<https://essence.ivarjacobson.com/alphastatecards>

Progress Poker - Use this game to determine the state of any particular Alpha

Chase the State - Use this game to determine the state of your software development efforts.

Objective Go - Use this game to identify high-level goals and objectives for your team.

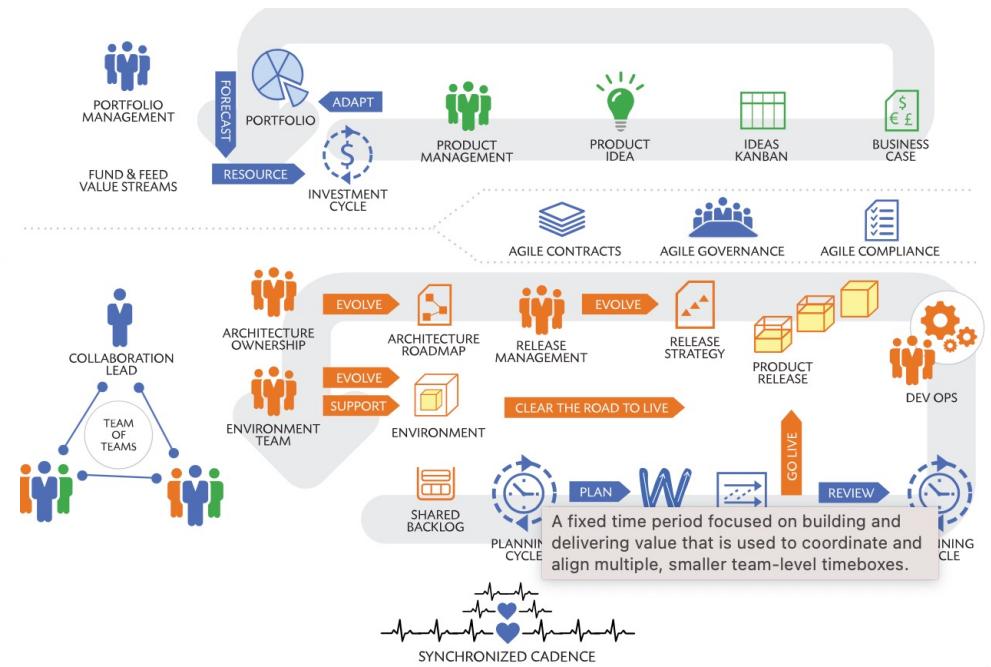
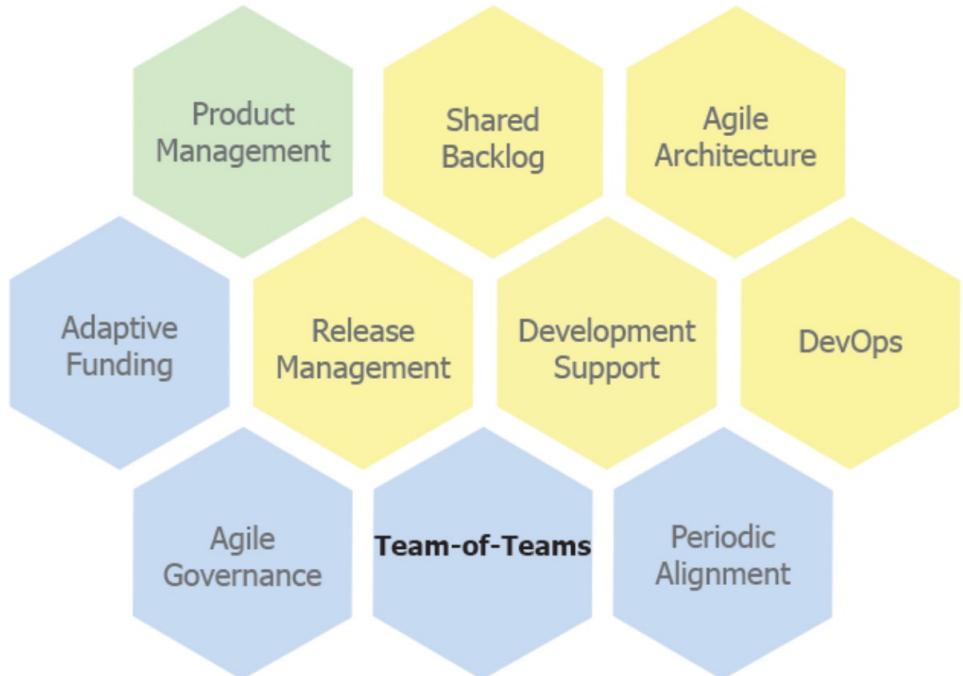
Checkpoint Construction - Use this game to define practice independent checkpoints with automatically generated practice independent checklists.

Lifecycle Layout - Use this game to visualize your software development lifecycle to form a starting point for team planning.

Milestone Mapping - Use this game to visualize your milestones and form a light-weight roadmap for your software development.

Health Monitoring - Use this game to visually track the health of your endeavor regardless of the practices or method being used.

Agile at a Scale in Essence: pratiche per team multipli



Essence, sviluppato dall'Object Management Group (OMG), è un metodo per descrivere e gestire i processi software in modo modulare e adattabile. Esso si basa su un "kernel" che include concetti fondamentali comuni a ogni processo di sviluppo software, come i competency, gli alpha (unità fondamentali che descrivono gli elementi chiave di un progetto, come requisiti e sistema), e i pattern.

Ecco i principali motivi per cui Essence rappresenta una buona idea per descrivere un processo:

1. **Flessibilità e Adattabilità**: Essence non impone una sequenza rigida di passi. Grazie ai suoi moduli, è facile personalizzare i processi per adattarli a team diversi o a contesti specifici, come progetti agili o DevOps.
2. **Orientamento ai Concetti Fondamentali**: Essendo costruito su concetti universali dello sviluppo software, Essence è utile per comprendere meglio gli aspetti fondamentali che ogni processo dovrebbe coprire, indipendentemente dalla metodologia specifica. Questo aiuta a evitare di perdersi nei dettagli o nelle varianti di implementazione, mantenendo il focus sugli obiettivi principali.
3. **Misurabilità del Progresso**: Essence supporta la misurazione dello stato di avanzamento grazie a checklist sugli stati degli alpha. Ogni alpha, come il team, il software system, o i requisiti, ha vari stati che indicano il livello di maturità raggiunto. Questo aiuta a monitorare con chiarezza i progressi del progetto.
4. **Comunicazione**: Essence fornisce un linguaggio comune e standardizzato che può essere compreso da sviluppatori, project manager e stakeholder, migliorando la comunicazione all'interno del team.
5. **Confronto tra Processi**: Essence consente di confrontare processi diversi, permettendo di capire come si differenziano e come si potrebbero migliorare o armonizzare.
6. **Supporto alla Crescita del Team**: Tramite i competency, Essence permette di definire le competenze necessarie e di monitorare la crescita professionale dei membri del team.

In breve, Essence è particolarmente utile quando si vuole descrivere un processo in modo modulare e flessibile, permettendo un'analisi chiara e dettagliata delle sue componenti chiave, facilitando anche il miglioramento continuo del team e del progetto stesso.

Riferimenti per Essence

- [Essify](#)
- [Sito del libro di testo](#)
- [Articolo di Ivar Jacobson](#)
- [Sito di Stefan Malich](#)

Tutte le carte di alcuni metodi agili e non (Scrum, Spotify, RUP, e altri) sono scaricabili (previa registrazione) da
[Essence Workbench](#)
in particolare guardare “Agile Essentials”

Sito [Essify](#), per essenzializzare il processo preferito dal team

Le carte di base (cioè gli alpha) e i loro stati sono in:
<https://essence.ivarjacobson.com/alphastatecards>
con una lista di giochi per analizzare lo stato del
processo

Ci sono molte altre carte per vari tipi di processi e attività.
Consultare:
<https://www.ivarjacobson.com/resources>

