Metodi e Tecnologie per lo Sviluppo Software - Summary

Leonardo Baldo

Contents

1 Issue Tracking System **3**
- 1.1 Utilizzo . . . 3
- 1.2 Work Item . . . 3
  - 1.2.1 Caratteristiche . . . 3
- 1.3 Workflow . . . 4
- 1.4 Funzionalità . . . 4
  - 1.4.1 Filtri . . . 4
  - 1.4.2 Board . . . 5
  - 1.4.3 Report . . . 5
- 1.5 Configurazione . . . 5
  - 1.5.1 Obiettivi . . . 5
  - 1.5.2 Configurazine . . . 5
  - 1.5.3 Utilizzo . . . 5

2 Version Control System **7**
- 2.1 Caratteristiche . . . 7
- 2.2 Tipologie di VCS . . . 7
  - 2.2.1 Local VCS . . . 7
  - 2.2.2 Centralized VCS - CVCS . . . 7
  - 2.2.3 Distributed VCS - DVCS . . . 7
  - 2.2.4 Cloud-Based DVCS . . . 7
- 2.3 Nozioni . . . 8
- 2.4 Tipologie di Workflow . . . 8
  - 2.4.1 Centralized WF . . . 8
  - 2.4.2 Feature Branch WF . . . 9
  - 2.4.3 Gitflow Model WF . . . 9
  - 2.4.4 GitHub Flow . . . 10
  - 2.4.5 GitLab Flow . . . 10
  - 2.4.6 Forking WF . . . 11
- 2.5 CVCS vs DVCS . . . 11

3 Framework Scrum **12**
- 3.1 Caratteristiche . . . 12
  - 3.1.1 3 Pilastri . . . 12
  - 3.1.2 Proprietà . . . 12
- 3.2 Sprint . . . 12
- 3.3 Ruoli . . . 13
  - 3.3.1 Product Owner . . . 13
  - 3.3.2 Scrum Master . . . 13
  - 3.3.3 Development Team . . . 13
- 3.4 Eventi . . . 14
  - 3.4.1 Sprint Planning . . . 14
  - 3.4.2 Daily Scrum Meeting . . . 14
  - 3.4.3 Sprint Review . . . 15
  - 3.4.4 Sprint Retrospective . . . 15
- 3.5 Artefatti . . . 15
  - 3.5.1 Product Backlog . . . 15
  - 3.5.2 Sprint Backlog . . . 16
  - 3.5.3 Definition of Done . . . 16
  - 3.5.4 Acceptance Criteria . . . 16

1 Issue Tracking System

Issue Tracking System: computer software package that manages and maintains lists of issues, as needed by an organization.

- **Issue:** criticità, attività/evento da gestire
- **Tracking:** registrare, lasciare delle tracce

1.1 Utilizzo

- **Condividere** le informazioni
  - unica repository dove trovare le informazioni
  - sistema di notifica
  - dashboard
- Implementare un processo per misurarne la **qualità**
- Avere un'istantanea dello **stato del progetto**
  - attività da fare
  - in corso d'opera
  - completate
- Decidere quando e cosa **rilasciare**
- Assegnare e dare **priorità alle attività**
- Consultare il **tempo impiegato**
- Avere una chiare **assegnazione delle attività**
- **Memoria storica** di tutti i cambiamenti del progetto

1.2 Work Item

Work item: singola attività minima del progetto, gestita mediante un workflow e mantenuta all'interno di un'unica piattaforma e di un'unica repository.

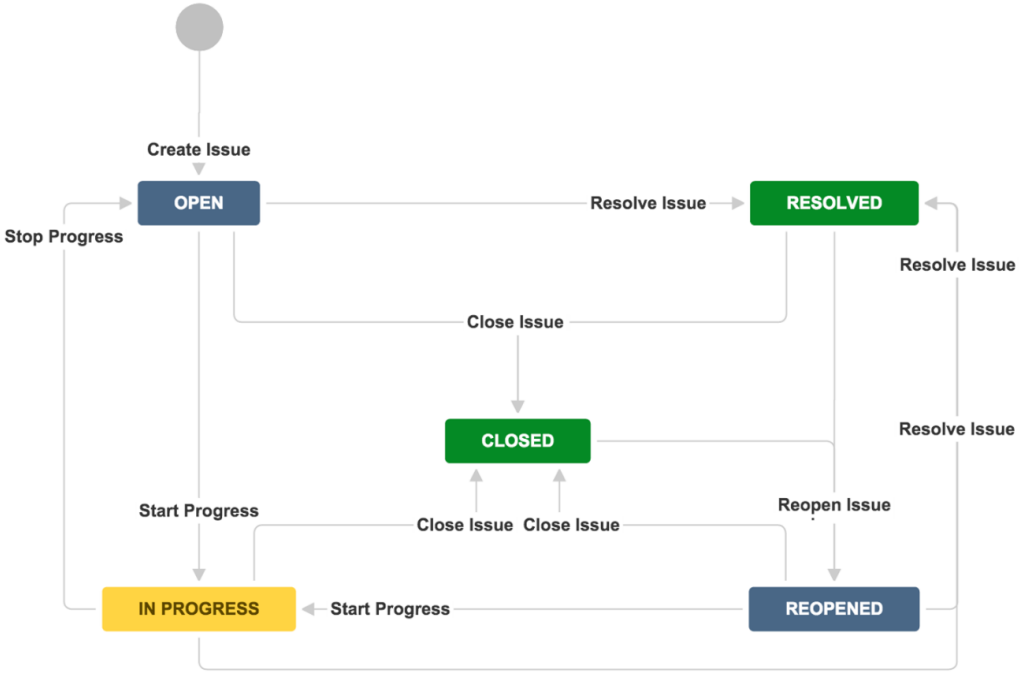
1.2.1 Caratteristiche

- **Progetto:** progetto a cui si riferisce
- **ID:** identificativo univoco
- **Descrizione:** descrizione dell'attività
- **Tipo:** categoria del work item
- **Stato:** stato all'interno del workflow in cui si trova il work item
- **Priorità:** importanza del work item in relazione con gli altri work item del progetto
- **Tag:** permettono di classificare i work item, anche di diversi tipi
- **Collegamenti:** permettono di collegare tra loro i work item
- **Assegnatario:** identifica chi è il responsabile per svolgere l'attività
- **Segnalante:** identifica chi ha segnalato l'attività
- **Data:** data di creazione, di ultimo aggiornamento, di risoluzione
- **Allegati:** file allegati

1.3 Workflow

Workflow: insieme di stati e transizioni che un Work Item attraversa durante il suo ciclo di vita.

- Permette di implementare il processo da seguire per completare l'attività
- Viene associato ad un progetto e può essere associato a uno o più tipi
- Permette di registrare tutte le transizioni e cambi si stato



1.4 Funzionalità

Gestione:

- Ricerca avanzata dei work item
- Salvataggio di ricerche
- Esportazione
- Reporting

Integrazione:

- Integrazione con il Source Code Management
- Integrazione con l'ambiente di sviluppo

Condivisione:

- Notifiche
- Bacheche o Board
- Dashboard
- Definizione di Road Map e Release Notes

1.4.1 Filtri

- Ricercare i work item in base ai campi
- Salvati per facilitare le ricerche più frequenti
- Risultati possono essere esportati
- Base per creare report, board e dashboard

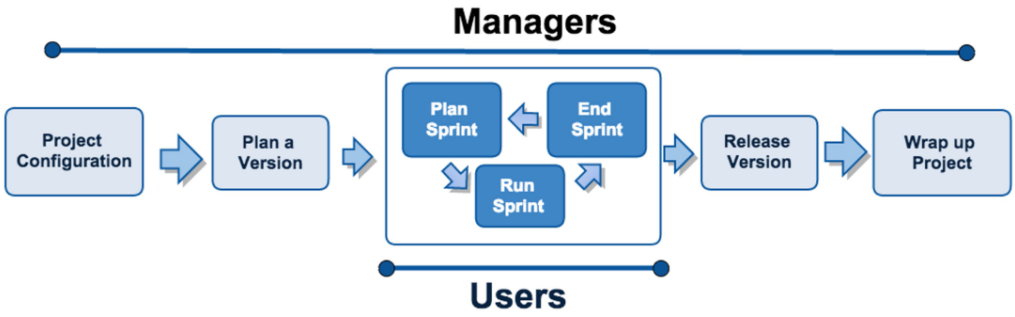
1.4.2 Board

- Visualizzare i work item di uno o più progetti, offrendo in modo flessibile e iterativo di visualizzazione, gestione e visualizzare dati di sintesi sulle attività in corso
- Configurare e visualizzare i work item ricercati con un filtro
- Interagire velocemente con i work item

1.4.3 Report

- Monitorare e avere una visione d'insieme del progetto

1.5 Configurazione



1.5.1 Obiettivi

- Identificare i processi richiesti per la gestione del progetto:
  - Procedure e best practices definiti dai framework di qualità presenti in azienda o richiesti dal cliente
  - Vincoli imposti dal cliente
  - Modalità di gestione del progetto del team
- Identificare e configurare gli strumenti che permettono di implementare i processi:
  - Identificazione e definizione dei tipi, campi custom, workflow e collegamenti che ci permettono di tracciare le informazioni richieste dal processo

1.5.2 Configurazine

Admin:

- Crea un nuovo progetto
- Definisce il processo da seguire:
  - tipi di work item, campi custom, workflow, collegamenti
  - seleziona il modello di stima
  - board e report per processo
- Aggiunge gli utenti e assegna ruoli/permessi

Capo progetto:

- Definisce le versioni [release]
- Definisce le componenti del progetto
- Definisce il lavoro da svolgere [backlog]:
  - priorità
  - assegnatario
  - stima
- Definisce la prima iterazione

1.5.3 Utilizzo

Team di Sviluppo:

- Riceve le notifiche dei work item assegnati
- Selezionano i work item in base alle priorità
- Avviano e completano la lavorazione:
  - avanzano gli stati del workflow
  - aggiornano la stima a finire
  - registrano il tempo impiegato
- Documentano lo stato dell'attività (commenti) e compilano e campi nel work item
- Completano tutte le attività presenti nell'iterazione
- Effettuano il rilascio

Capo progetto:

- Monitora l'avanzamento e il completamento delle attività (filtri, board. dashboard, report)
- Definisce le nuove versioni
- Definisce le nuove iterazioni
- Definisce, aggiorna e monitora le attività (priorità, verifica stima)
- Produce i report richiesti dal cliente

2 Version Control System

VCS: a component of software configuration management, is the management of changes to documents, computer programs, large web sites, and other collections of information.

2.1 Caratteristiche

- Sono sistemi software
- Registrano modifiche avvenute ad un insieme di file
- Condivisione di file e modifiche
- Funzionalità: merging, tracciamento modifiche

2.2 Tipologie di VCS

2.2.1 Local VCS

- Tool più vecchi
- Registrano solo storia cambiamenti
- Non gestiscono la condivisione
- Esempi: SCSS, IDE(Eclipse, IntelliJ)

2.2.2 Centralized VCS - CVCS

- Meno vecchi e molto diffusi
- Gestiscono sia la condivisione, che il tracciamento della storia
- Ogni sviluppatore è un client che ha nel suo spazio di lavoro solo una versione del codice
- Facili da apprendere
- Esempi: CVS, Subversion(SVN), Perforce, TFS

2.2.3 Distributed VCS - DVCS

- Version Database distribuito per duplicazione in ogni nodo
  - quando il nodo principale non è disponibile, è possibile continuare a lavorare e registrare i cambiamenti
  - migliore risoluzione dei conflitti
  - diversi tipi di flussi di lavoro
- Difficili da apprendere
- Esempi: Git, Mercurial

2.2.4 Cloud-Based DVCS

- VCS as a Service
- Version Database gestito in un servizio Cloud
- Esempi: GitHub, GitLab

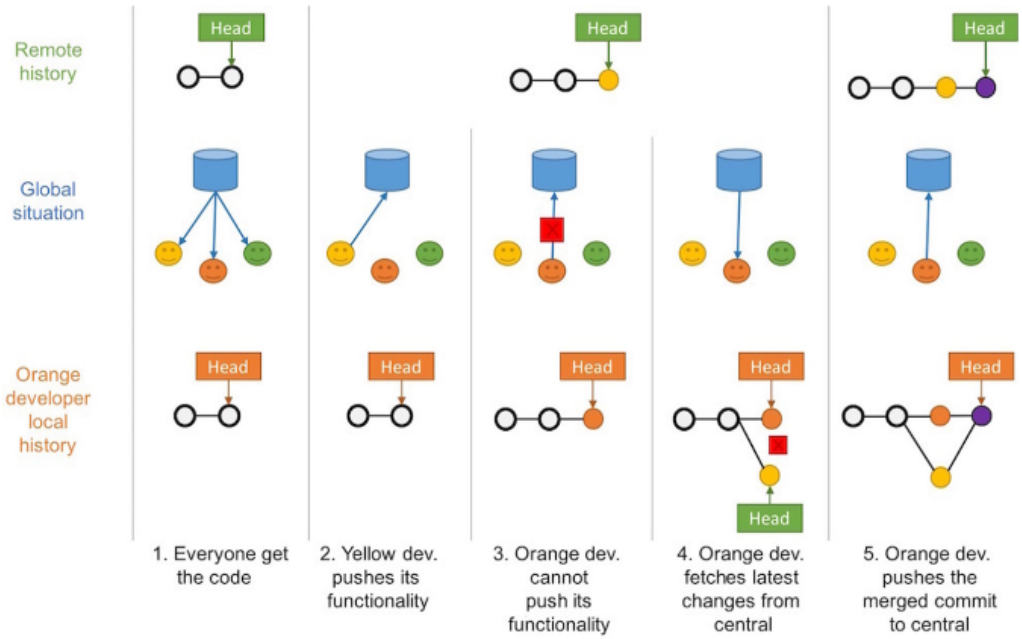
2.3 Nozioni

- **DIFF:** each set of changed lines
- **COMMIT:** set of DIFF
- **HEAD:** last commit
- **BRANCH:** pointer to a single commit
  - HEAD is the latest branch, know as **main** branch
  - to integrate a branch, you have to merge it
- **PULL REQUEST:** way of handling branch merges to main
  1. branch pushed to the central server
  2. ask to be merged on a central repo
  3. review change before merging
  4. pull request closed or merged to destination branch

2.4 Tipologie di Workflow

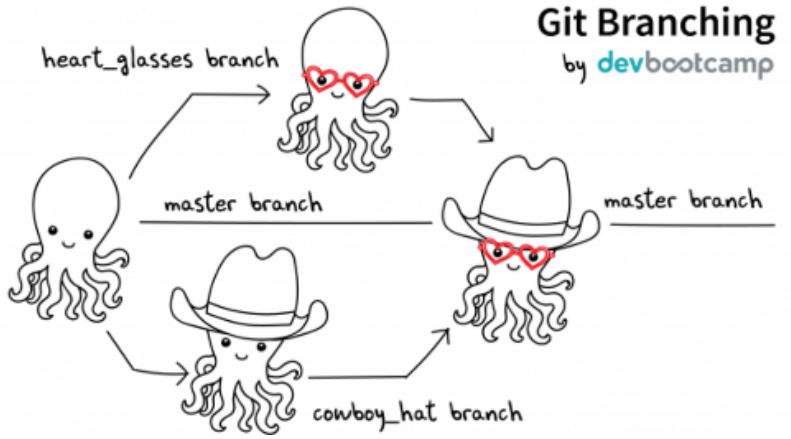
2.4.1 Centralized WF

- Utilizzo naturale di un CVCS come SVN o CVS
- Facile da capire e da usare
- Collaborazione bloccata quando il server centralizzato è fuori uso o la cronologia è interrotta



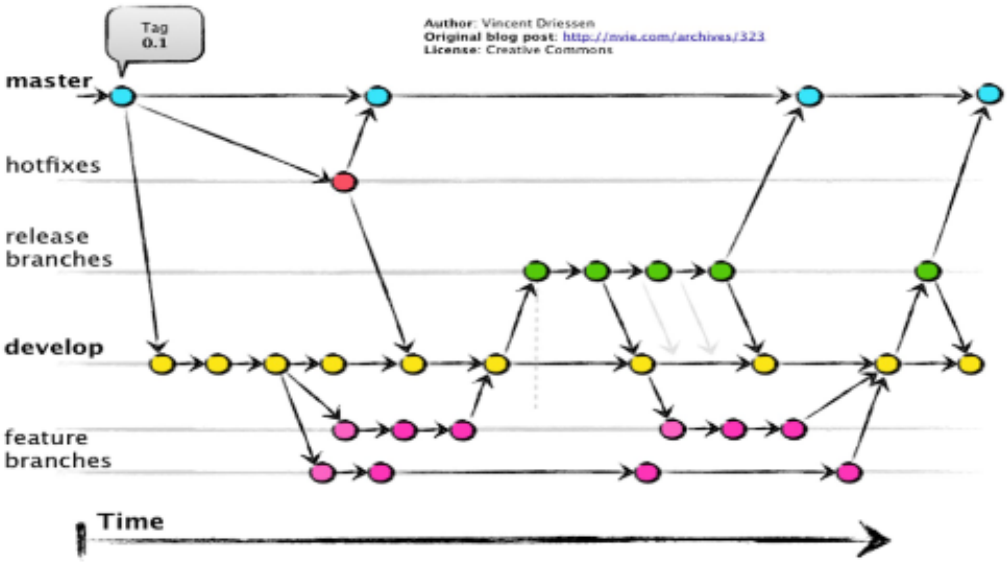
2.4.2 Feature Branch WF

- L'obiettivo è quello di utilizzare un solo ramo per caratteristica (DVCS)
- L'incapsulamento consente di lavorare senza distribuire la base di codice principale
- Collaborazione più facile
- Più facile da tracciare



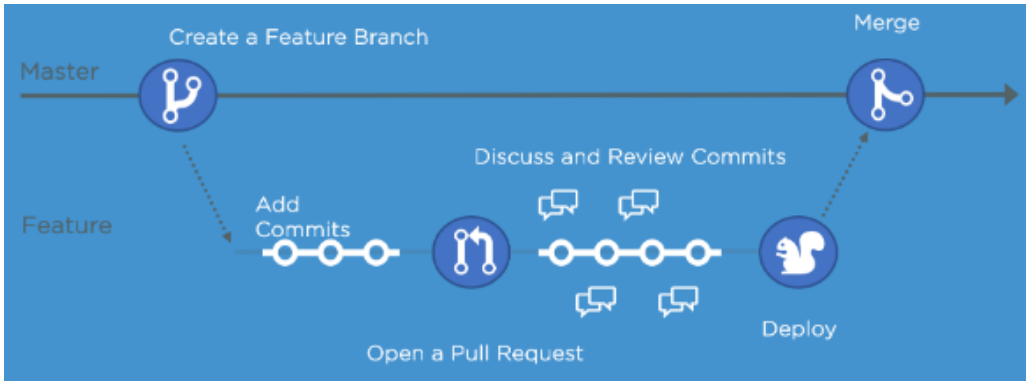
2.4.3 Gitflow Model WF

- main branch: codice rilasciato
- developer branch: snapshot per la prossima release
- feature branch: nuova funzionalità



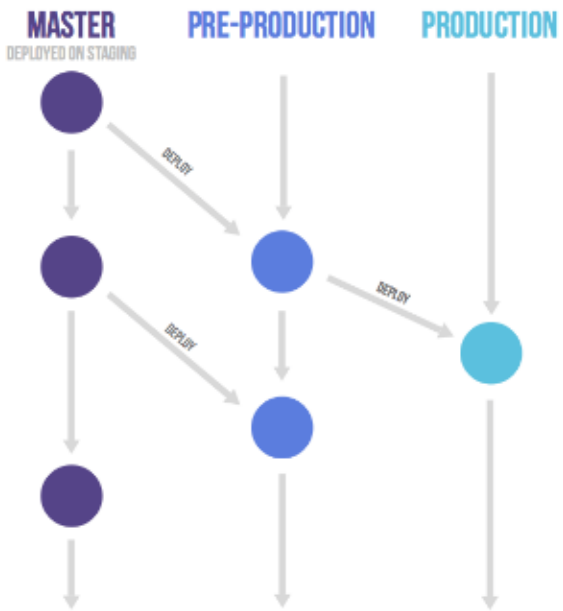
2.4.4 GitHub Flow

- Approccio più veloce di sviluppo
- Focalizzato sulle caratteristiche per unire i nuovi rami con il ramo master
- Flusso di lavoro perfetto per piccoli team e progetti



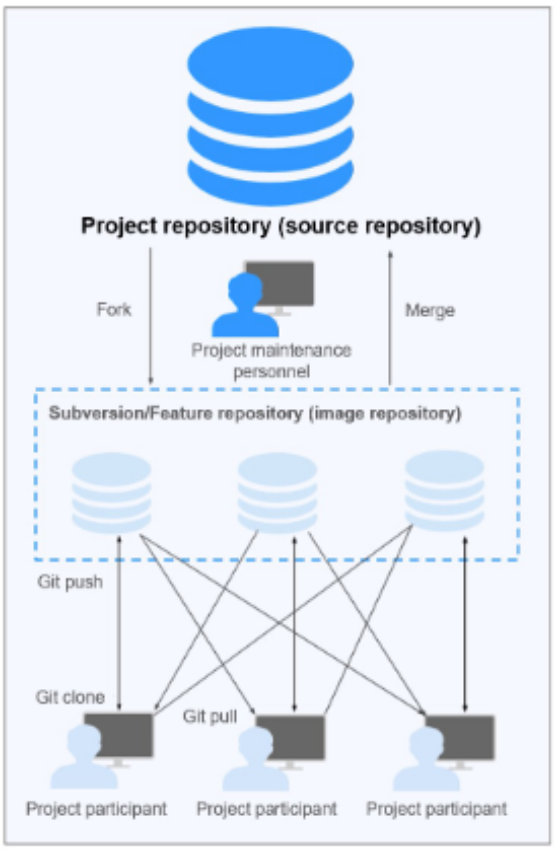
2.4.5 GitLab Flow

- Approccio di sviluppo più attento all'affidabilità
- Processo di test a più fasi



2.4.6 Forking WF

- Concetti di push forward del file system distribuito
- Ogni utente fa il fork del repo principale e può proporre richieste di pull tra i repo
- Gestione delle autorizzazioni migliorata
- Autonomia per un migliore processo di collaborazione
- Decentrato per nuovi modelli



2.5 CVCS vs DVCS

CVCS:

+ apprendimento più semplice
+ lock file
- meno recenti
- centralized workflow
- commit più lenti
- single point of failure

DVCS:

+ più recenti
+ distribuiti
+ migliori workflow
+ commit veloci
- no lock file
- apprendimento difficile

3 Framework Scrum

Scrum: processo agile che nasce per lo sviluppo di progetti complessi, che ci permette di concentrarci sulla consegna del maggior valore business nel più breve tempo.

3.1 Caratteristiche

- Leggero
- Facile da capire
- Difficile da padroneggiare

3.1.1 3 Pilastri

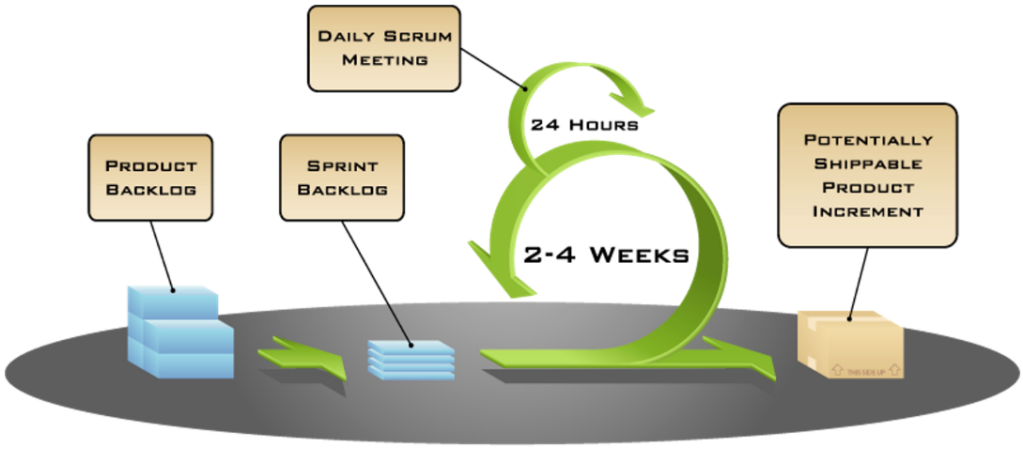
- **Trasparenza:** linguaggio comune per una conoscenza condivisa
- **Controllo:** ispezioni pianificate per prevenire variazioni non desiderate
- **Adattamento:** aggiustamenti per minimizzare ulteriori deviazioni tramite feedback continuo

3.1.2 Proprietà

- Gruppi che si **auto-organizzano**
- Il prodotto evolve attraverso **sprint** di durata fissa
- I requisiti sono trattati come elementi di una lista detta "product backlog"
- Non vengono prescritte particolari pratiche ingegneristiche
- Si basa sull'attività empirica cioè la conoscenza si basa sull'esperienza e le decisioni si basano su ciò che è conosciuto
- **Processo iterativo e incrementale** per ottimizzare il controllo dello sviluppo e il controllo del rischio

3.2 Sprint

- I progetti Scrum progrediscono attraverso una serie di sprint
- Durata tipica di 2-4 settimane: una durata costante favorisce un ritmo migliore
- Il prodotto è progettato, realizzato e testato durante lo sprint



3.3 Ruoli

3.3.1 Product Owner

- Definisce le caratteristiche del prodotto
- Rappresenta il desiderio del committente
- Decide date e contenuto del rilascio
- È responsabile della redditività del prodotto (ROI)
- Definisce le priorità delle caratteristiche del prodotto in base al valore che il mercato gli attribuisce
- Adegua le caratteristiche e la priorità ad ogni iterazione, secondo quanto necessario
- Responsabile che il Product Backlog sia chiaro e ordinato
- Accetta o rifiuta i risultati del lavoro

3.3.2 Scrum Master

- Rappresenta la conduzione del progetto
- Responsabile dell'adozione dei valori e delle pratiche Scrum
- Rimuove gli ostacoli
- Si assicura che il gruppo di lavoro sia pienamente operativo e produttivo
- Favorisce una stretta cooperazione tra tutti i ruoli e le funzioni
- Protegge il gruppo di lavoro da interferenze esterne
- Servant leader: aiuta Product Owner e Team di sviluppo condividendo la gestione e le decisioni con il team

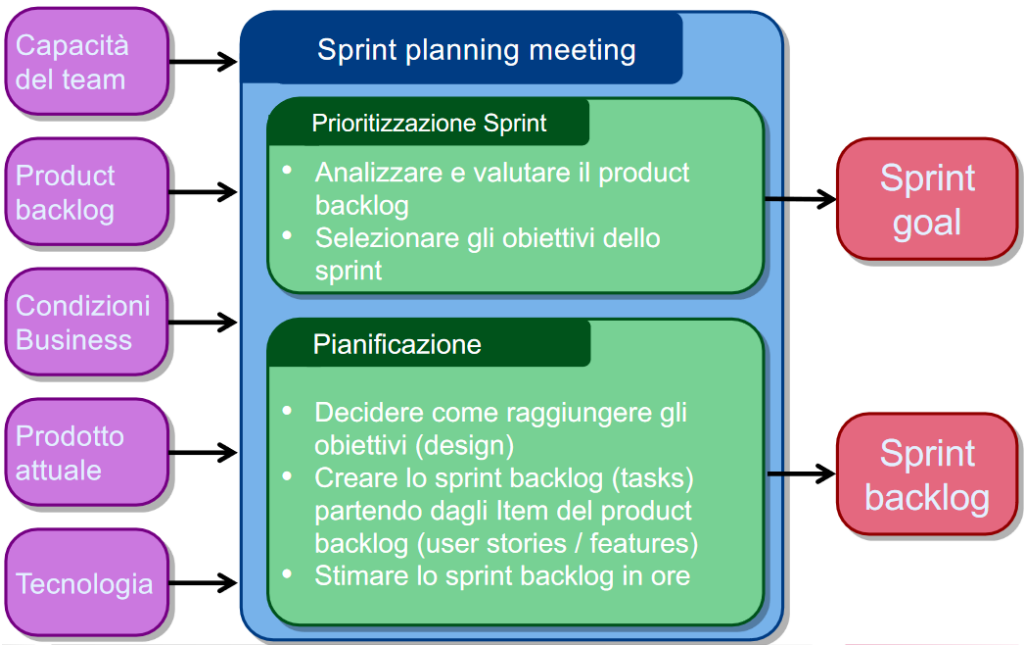
3.3.3 Development Team

- Tipicamente 5-9 persone
- Responsabili di realizzare l'incremento in conformità alla Definition of Done
- Competenze trasversali (cross functional): programmatori, tester, progettisti di user experience, ...
- Membri di progetto dovrebbero lavorare full-time
- Possono esserci eccezioni (e. amministratori di database)
- Il gruppo di lavoro si auto-organizza: idealmente senza titoli, ma in rari casi può essere una possibilità

3.4 Eventi

3.4.1 Sprint Planning

- È un evento "Time boxed" di 8h per Sprint di 1 mese
- Cosa può essere realizzato durante lo Sprint? Il Team seleziona dal product backlog gli item che può impegnarsi a completare
- Viene creato lo Sprint backlog collaborativamente da tutto il team
- Vengono identificate le Task, e ciascuno di questi viene stimato (1-16 ore)
- Come completare il backlog?
- Decomposizione delle User Story



3.4.2 Daily Scrum Meeting

- Incontro giornaliero di 15 minuti, fatto in piedi
- Non per la soluzione di problemi, ma per sincronizzarsi su quanto fatto e pianificare la giornata per il raggiungimento dello Sprint Goal
- Si aggiorna la scrumboard
- Aiuta ad evitare altre riunioni non necessarie
- In caso può partecipare anche il Product Owner
- Domande:
  - Cosa hai fatto ieri?
  - Cosa farai oggi?
  - C'è qualcosa che ti impedisce di farlo?

3.4.3 Sprint Review

- Time boxed: 4h per Sprint di 1 Mese
- Il gruppo di lavoro presenta ciò che ha realizzato durante lo sprint
- Viene validato e accettato quanto realizzato
- Tipicamente in forma di demo delle nuove caratteristiche o dell'architettura sottostante
- Informale:
  - Regola delle 2 ore per la preparazione
  - Niente slide
- Partecipa tutto il gruppo
- Tutti sono invitati (anche gli esterni)

3.4.4 Sprint Retrospective

- Si celebra dopo la Sprint Review e prima del prossimo Sprint Planning
- Time boxed: 3 ore per Sprint di 1 mese
- Si valuta ciò che sta funzionando e ciò che non sta funzionando
  - Come migliorare la qualità del prodotto?
  - La Definition of Done è appropriata?
  - Che miglioramenti possiamo apportare al prossimo Sprint?
- Partecipa tutto il gruppo di lavoro:
  - Scrum Master
  - Product Owner
  - Development Team

3.5 Artefatti

3.5.1 Product Backlog

- I requisiti, funzionalità, miglioramenti, fix da realizzare nei prossimi rilasci
- Una lista di tutti i "desiderata"
- Idealmente espressa in modo che ciascun elemento ha valore per gli utenti o I clienti del prodotto
- Priorità assegnate dal Product Owner mentre il Dev. Team stima ogni item
- Priorità rivalutate all'inizio di ogni sprint con il Development Team
- Raffinamento continuo, è una lista dinamica che evolve con il prodotto

User Stories: Item che compongono il Product Backlog e andranno scomposte in Task

3.5.2 Sprint Backlog

- Ogni componente del Development Team si sceglie qualcosa da fare
- La stima del lavoro rimanente è aggiornata ogni giorno
- Ogni membro del gruppo di lavoro può aggiungere, cancellare o modificare parti dello sprint backlog
- Il lavoro da svolgere durante lo sprint "emerge"
- Se il lavoro non è chiaro, definire un elemento dello sprint backlog con una stima temporale più ampia, e decomporlo successivamente
- Aggiornare il lavoro rimanente man mano che diventa più chiaro
- Massima visibilità della scrumboard

3.5.3 Definition of Done

- Il minimo set di attività per definire che un'attività è completata
- Può variare per gruppo di lavoro
- Deve essere bene chiaro per tutti i membri del gruppo di lavoro
- È utilizzato per verificare se un'attività è da ritenersi completata

3.5.4 Acceptance Criteria

- Permette di confermare se la storia è completa e funziona come voluto
- Frasi semplici condivise da Product Owner e Development Team
- Possono essere incluse con la User Story
- Rimuovono l'ambiguità dei requisiti