Corso Ing del sw 2022-23

Il processo e l'ambiente di sviluppo

Il processo e l’ambiente di sviluppo

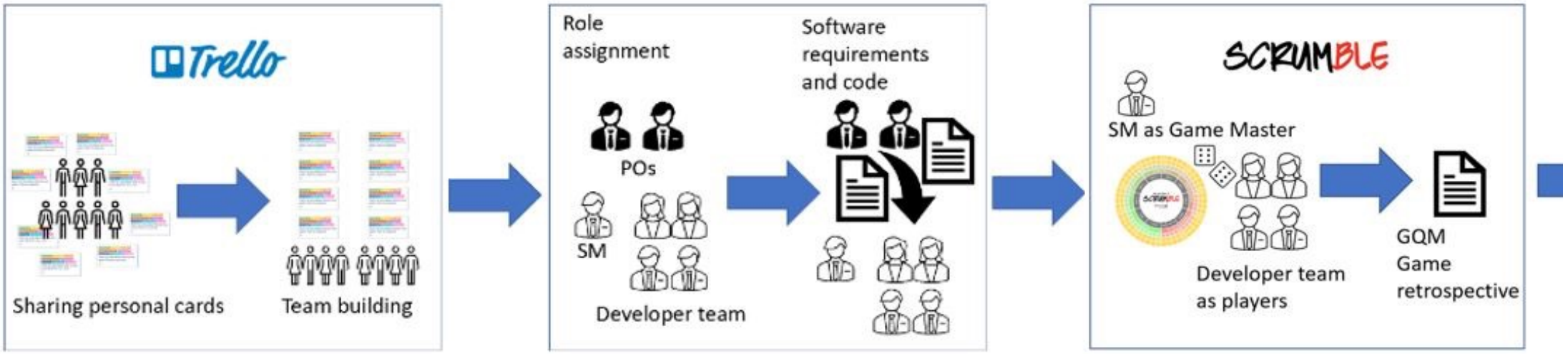
In questo progetto del corso di Ingegneria del software seguiremo i seguenti principi:

- Collaborazione agile, con coinvolgimento dei clienti-docenti
- Modello di processo basato su Scrum
- Uso di strumenti software allo stato dell’arte e open source
- Uso di pipeline di strumenti (es. GitLab+Jenkins+SonarQube)
- Uso delle carte Essence per le retrospettive

Iniziare il processo

- Creare il team con Trello
- Giocare almeno una partita a Scrumble (team building)
  - Assegnare i ruoli: PO SM developers
  - Riempire l'autovalutazione GQM
- Familiarizzare con l'ambiente CAS
  - Taiga
  - GitLab
  - Mattermost per le comunicazioni del team (le memorizza)
  - Scegliere un IDE (Eclipse, atom, visio) , arricchirlo con plugin e dashboard
- Per le retrospettive userete le carte Essence
  https://essence.ivarjacobson.com/services/agile-essentials-starter-pack-agile-practices

La fase iniziale (preparazione)



La partita a Scrumble

- Assegnare i ruoli: PO, SM, developers
- Leggere assieme le regole e usare il materiale di gioco
- Giocare
- Riempire lo schema GQM di autovalutazione (foglio excel)

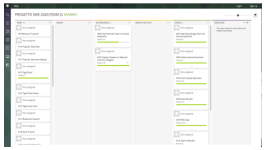
- Vedere le slide apposite su Scrumble
- Materiale disponibile anche su scrumble.pyxis-tech.com

Lo sviluppo

- Sprint zero
- Tre sprint di sviluppo, più altri (decide il team)
- Primo sprint: scrivete le user stories iniziali del backlog principale
- Ogni sprint: scegliete alcune user stories dal backlog principale
- Ogni sprint inizia con sprint planning che definisce sprint goal
- Ciclo di ciascuno sprint:
  - Stima delle user stories scelte per lo sprint, si ottiene la stima di sprint goal
  - Stesura del codice, versionamento (con Gitab), test (con Jenkins), analisi statica (con SonarQube)
  - Demo (sprint review)
  - Retrospettive con Essence, registrare il risultato sul wiki di Taiga

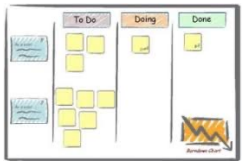
Calendario

- 6 ottobre inizio sprint zero
- 20 ottobre inizio sprint 1
- 31 ottobre stato avanzamento dei team (online) - user stories
- 3 novembre inizio sprint 2
  - 14 novembre Stato avanzamento
- 17 novembre inizio sprint 3
  - 1 dicembre stato avanzamento
- 2 dicembre inizio sprint 4



taiga (kanban)

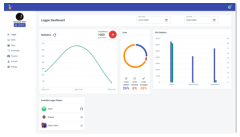
logger



Sprint Planning

Daily Stand Up

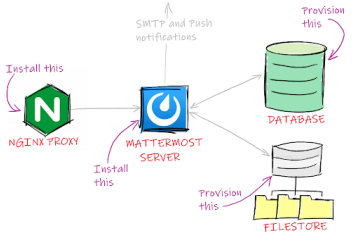
Backlog



dashboard

essence

Sprint Retrospective



mattermost

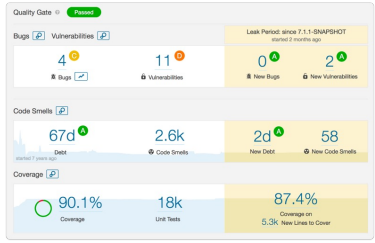
Gitlab (commit timeline)

taiga (burndown)

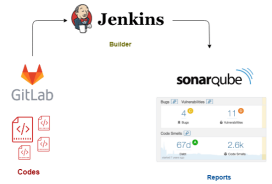
Jenkins

Sprint Review

Gitlab (versioning)



sonarqube



Le retrospettive: usare Essence

- Alla fine di ogni sprint il team effettua una retrospettiva
- Suggerisco di usare le carte Essence ed uno o più giochi di retrospettiva
- (vedere gruppo di slide sulle retrospettive con Essence)

Relazione finale ed esame

- La relazione racconta il processo (non il prodotto)
- L’esame orale (tutto il team è presente) verte sulla relazione finale
- L’esame si conclude con una demo del prodotto
- Voto unico al team

Meccanismo a punti

(assegnati dal docente alla fine del progetto, scala da 0 a max)

- Uso di Taiga, GitLab, Jenkins, SonarQube su CAS: 100 punti max ciascuno
- Mattermost su CAS: 200 punti max
- Toolchain gitlab con Jenkins e/o SonarQube e/o Mattermost: bonus 100 pt
- Uso del logger/dashboard: 50 punti per self-tracking, 100 punti per team tracking
- Team building con Scrumble: 50 punti max (secondo team building: 50 punti)
- Retrospettive con Essence: 25 punti per retrospettiva (max 100)
- Pair programming con IDE e plugin appositi (es. Saros): 10 punti/coppia
- Consegna dopo il 4 sprint entro natale: 300 punti. Consegna dopo il 5° sprint: 200 punti e via a scalare 50 punti per ogni sprint successivo.

Team target: 1000 punti

Come iniziare (sprint zero)

1. Creare il proprio team mediante Trello
2. Il team gioca una partita a Scrumble e registra l'autovalutazione sullo spreadsheet predisposto (da riportare nel report finale)
3. Il team deve configurarsi come tale sui vari servizi del server
4. Ciascun partecipante prenda confidenza con gli strumenti: Taiga, GitLab, Mattermost, Jenkins, Sonarqube
5. Ciascun partecipante configura il proprio IDE (con plugin lato client, vedere le varianti presenti in https://github.com/elPeroN)
6. Prendere confidenza con le retrospettive mediante carte Essence

L’ambiente CAS è open source

- CAS: un ambiente totalmente open source «modulare»
- Obbligo di usare gli strumenti open source proposti
- Proporre nuovi strumenti – in aggiunta o in alternativa - è possibile, ma solo open source

- Ambiente pubblicamente accessibile con controllo degli accessi
- Self tracking e team tracking (con logger e altri misuratori)

Come accedere agli strumenti sul server

- Taiga: aminsep.disi.unibo.it e poi fare signup
- GitLab: https://aminsep.disi.unibo.it/gitlab
- Mattermost: https://aminsep.disi.unibo.it/mattermost
- Jenkins: https://aminsep.disi.unibo.it/jenkins
- SonarQube: https://aminsep.disi.unibo.it/sonarqube
- Dashboard: https://aminsep.disi.unibo.it/dashboard (usare plugin)

NB: sono microservizi indipendenti, ciascuno con proprio account e passwd (NO single sign on). All'inizio occorre fare sign up

NB2: per SonarQube il procedimento è diverso, occorre intervento sistemisti

(NO sign up autonomo)

Se si vuole creare un proprio server

- Scaricare script docker da https://github.com/elPeroN

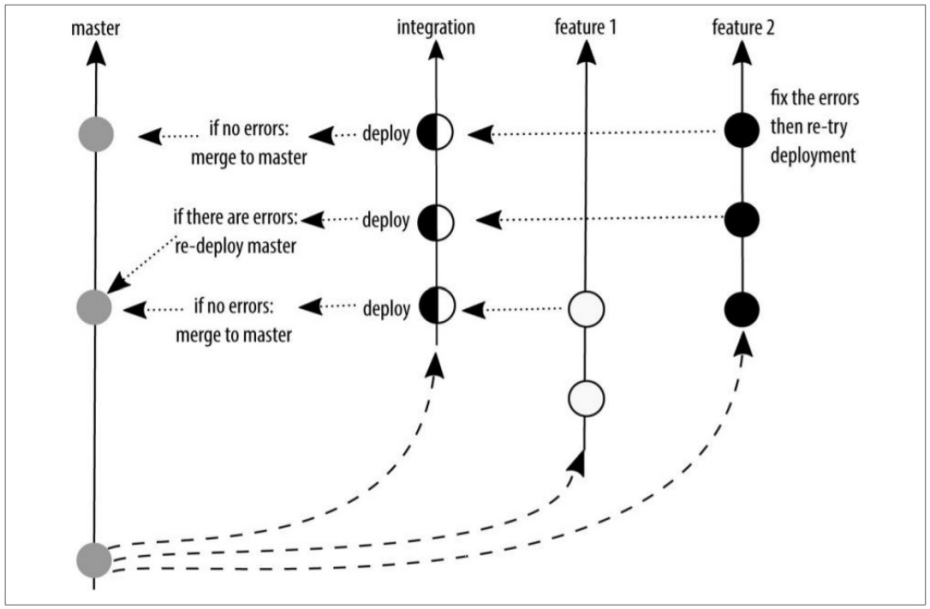
Creare i team negli strumenti

1. Su Mattermost
2. Su Taiga
3. Su GitLab

Taiga

- Inserire il team
- Inizializzare il backlog e aggiornarlo giorno per giorno
- Controllare che il burndown venga gestito automaticamente giorno per giorno
- Usare il wiki per documenti del team - es. diagrammi UML

GitLab



- GitLab è uno strumento per il versionamento del codice.
- Per un utilizzo in team rispetto ad un uso singolo sono necessarie almeno due convenzioni, quella di branching e quella di tagging

GitLab

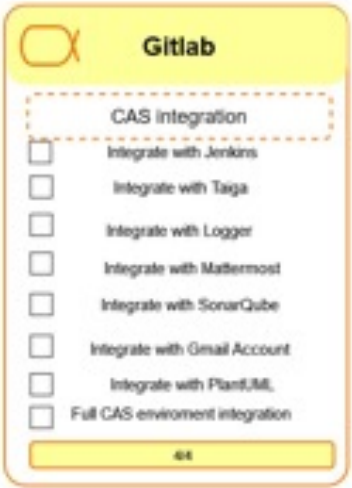

Git
Knowledge of commands and best practices. Knowing how to use git as a team
Contributors
Expert
Advanced
Intermediate
Beginner


Gitlab
How gitlab should be configured, and how it integrates with the existing environment and how it is used by team members.
Configured and integrated into the development environment
Instantiated
Integrated into the workflow
CAS integration


Gitlab
Configured and integrated into the development environment
GitLab installation was successful
Properly configured
The server can be reached from both inside and outside the network
1/4


Gitlab
Instantiated
Project repository created
License and visibility of the project decided
Branching conventions agreed
Release schedule planned
Conventions documentantion written and shared
Tag conventions agreed
2/4


Gitlab
Integrated into the workflow
All members understand the use of the branching convention
All members understand the use of the tagging convention
At least one sprint was completed without any problems regarding GitLab
3/4

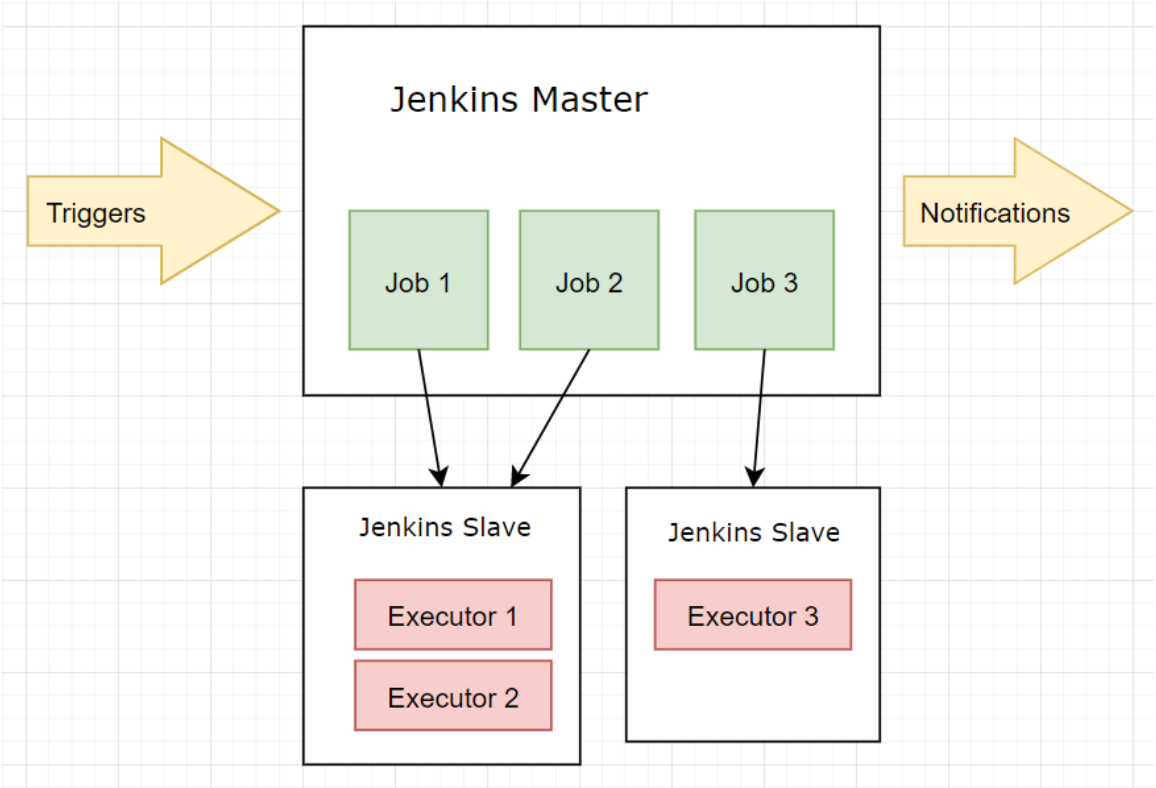

Gitlab
CAS integration
Integrate with Jenkins
Integrate with Taiga
Integrate with Logger
Integrate with Mattermost
Integrate with SonarQube
Integrate with Gmail Account
Integrate with PlantUML
Full CAS enviroment integration
4/4

Mattermost

- Scrum Master SM invita gli altri membri del team a fare gruppo
- GitLab e Mattermost possono parlarsi
  https://docs.gitlab.com/ee/integration/mattermost/

- Nota: esiste una versione sperimentale di un bot telegram che integra gli strumenti di CAS

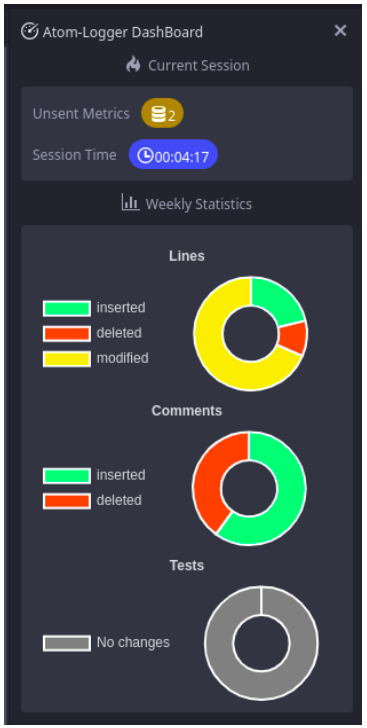
Jenkins https://it.wikipedia.org/wiki/Jenkins_(software)



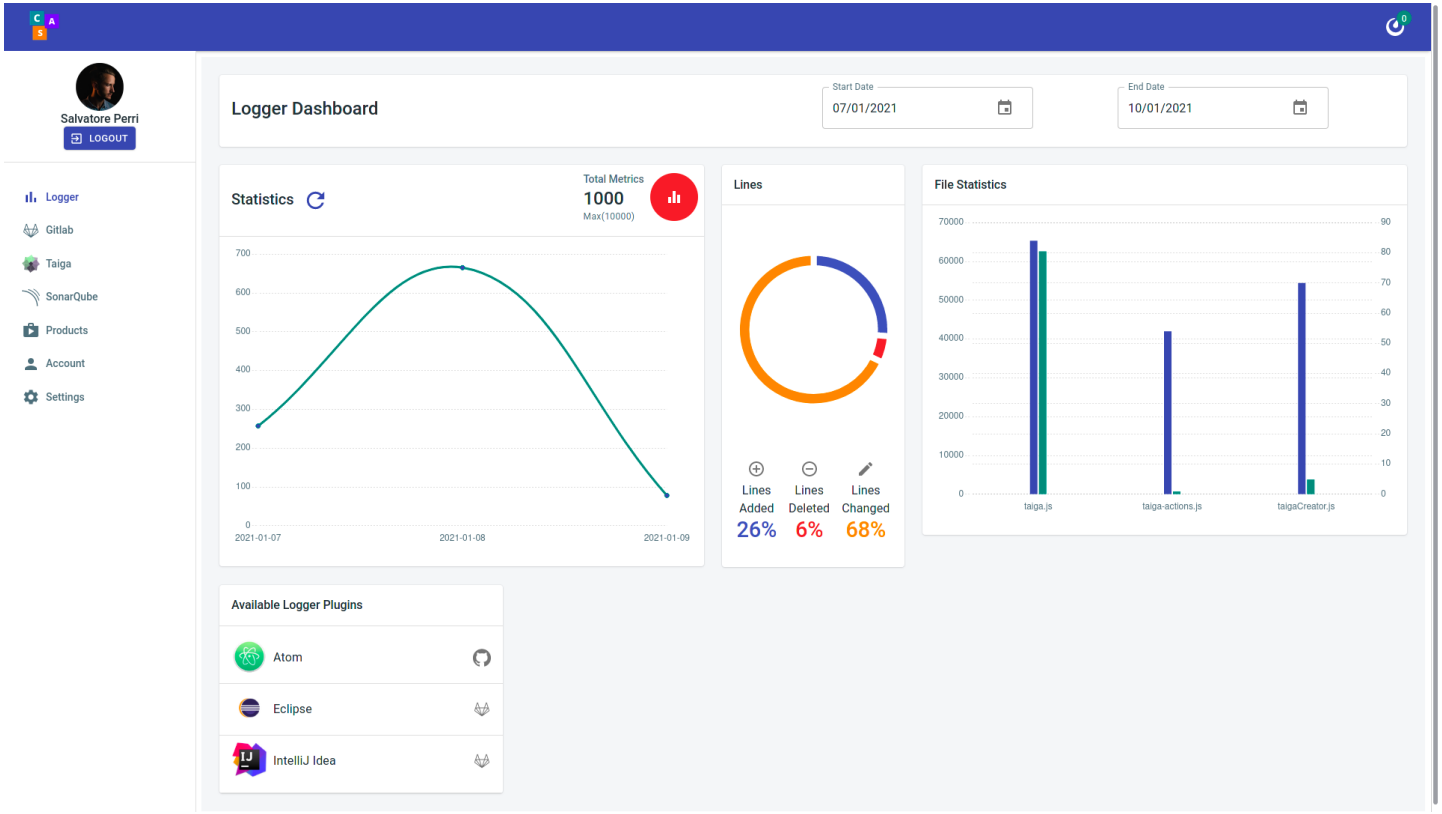
SonarQube

- Signup: a cura dei sistemisti - riempite il googledoc https://docs.google.com/spreadsheets/d/1VtICkfUEx8zvgrjhk9VDAB20lyYOz6AJJVYjT2cGp_A/edit?usp=sharing
- Documentazione: online https://docs.sonarqube.org/latest/ + libro + tesi

Logger



Interfaccia logger



Interfaccia dashboard

Documentazione

- Per gli strumenti CAS, vedere la documentazione online nei siti
  - taiga.pm e taiga.io
  - Mattermost.com
  - about.gitlab.com
  - Sonarqube.org
  - jenkins.io
  - Per il logger vedere la documentazione in https://github.com/elPeroN