Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica

LA MISURAZIONE DEL TEAMWORK NEI PROGETTI AGILI

Relatore:
Chiar.mo Prof.
PAOLO CIANCARINI

Presentata da:
ANDREA SCHINOPPI

I Sessione
Anno Accademico 2021/2022

5 parole chiave per caratterizzare il contenuto della dissertazione:

Agile

Teamwork

Scrum

Ingegneria del Software

Modello di maturità

Alla mia famiglia che ha
sempre rappresentato per me
un solido punto di riferimento

Abstract

Lo scopo di questa tesi è quello di aggregare ed elaborare tutti i dati che sono stati generati durante lo svolgimento del progetto del corso di Ingegneria del Software presso la facoltà di Informatica dell'Università di Bologna nell'Anno Accademico 2021/2022.

In particolare saranno elaborati i dati provenienti da un questionario rivolto a tutti i partecipanti e dall'ambiente CAS (Compositional Agile System).

Al progetto hanno partecipato 79 studenti, suddivisi in 16 team.

La dissertazione consisterà nell'applicazione di un modello di qualità del teamwork e uno di maturità, seguendo il metodo adottato da Sofia Zani nella sua tesi lo scorso anno.

Alla fine saranno analizzati tutti gli aspetti presi in considerazione e sarà ricercata una correlazione tra i due modelli.

Indice

Elenco delle tabelle

Elenco delle figure

Capitolo 0

INTRODUZIONE

L'obiettivo di questa tesi è quello di analizzare i dati ottenuti a seguito dello sviluppo del progetto per il corso di Ingegneria del software dell'anno accademico 2021/2022.

L'analisi sarà condotta prevalentemente basandosi su un modello di qualità del teamwork e uno di maturità.

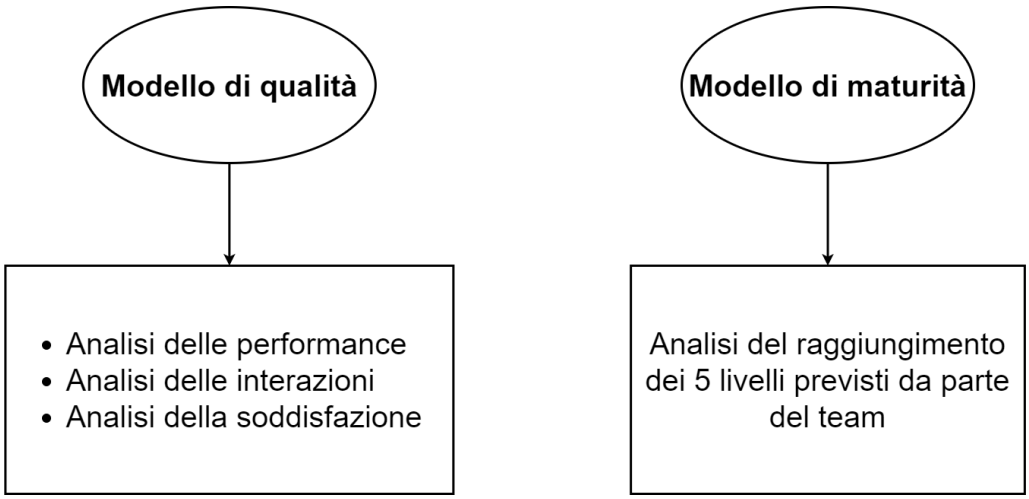

Figura 1: Riassunto dei due modelli

Nel corso della dissertazione saranno effettuate anche altre analisi utili a comprendere meglio i risultati e a fornire ulteriore contesto allo studio.

0.1 Lavori precedenti

Questa tesi segue la stessa metodologia utilizzata da Sofia Zani lo scorso anno nella sua tesi [Zan20] e nel relativo articolo [CMZ21].

In particolare il modello di qualità è descritto nell'articolo di Hoegl e Gemuenden [HG01] e ripreso da Lindsjørn et al. [LSD+16].

Il modello di maturità per i team agili invece è stato proposto da Yin et al. [YFdS11], basandosi su quello creato da Chetankumar et al. [CR09].
Anche Gren et al. hanno discusso del concetto di maturità del teamwork nei team agile [GGJ20].

0.2 Struttura della tesi

Questa dissertazione è divisa in 7 capitoli, oltre al presente capitolo introduttivo:

- Nel capitolo 1 saranno descritti il progetto del corso e la raccolta dei dati
- Nel capitolo 2 sarà analizzato il processo seguito dagli studenti
- Nel capitolo 3 sarà descritto ed applicato il modello di qualità del teamwork
- Nel capitolo 4 sarà descritto ed applicato il modello di maturità del teamwork
- Nel capitolo 5 sarà effettuato un confronto tra i risultati ottenuti per l'anno 2021/2022 e quelli dell'anno precedente
- Nel capitolo 6 saranno brevemente presentate ulteriori analisi
- Infine, nel capitolo 7 saranno riportate le conclusioni

Capitolo 1

DESCRIZIONE DEL PROGETTO E RACCOLTA DATI

Il progetto per l'anno accademico 2021/2022 consisteva nella realizzazione di un'applicazione in grado di interagire con le API del noto social network Twitter per poter raccogliere ed organizzare tweet sia in maniera asincrona, sia in tempo reale. L'elenco completo dei requisiti si trova in Appendice C, ma, in breve, possono essere così riassunti:

- Raccogliere i tweet
- Classificare i tweet geolocalizzati
- Classificare i tweet pubblicati in una data area geografica
- Classificare i tweet contenenti un certo punto di interesse
- Classificare i tweet con una certa parola chiave
- Classificare i tweet geolocalizzati di una persona specifica, per seguirne gli spostamenti
- Analizzare il sentimento (sentiment analysis) di una serie di tweet

Al termine della raccolta era necessario che l'applicazione fosse in grado di visualizzare i dati in forma aggregata tramite una dashboard che presentasse una mappa, dei grafici e una term cloud con le parole più utilizzate nei tweet elaborati.

Tutto il progetto doveva essere svolto seguendo rigorosamente le pratiche dettate dal manifesto Agile[FH+01] dando maggior rilevanza al processo di produzione del software piuttosto che al prodotto finale.

1.1 Descrizione del processo

In questa sezione viene presentato il processo di produzione del software adottato.

È possibile dividere il processo in tre fasi: fase preliminare, fase di esecuzione e fase conclusiva, di seguito illustrate.

Fase preliminare

Durante questa prima fase è avvenuta la formazione dei team di sviluppo tramite l'applicativo Trello: ogni studente ha creato la propria card sulla piattaforma, inserendo un nickname, una breve descrizione delle proprie conoscenze e le proprie preferenze in termini di programmazione (ad esempio sviluppo front-end, sviluppo back-end o testing).
In questo modo si è cercato di formare team che, pur ricoprendo tutti i ruoli necessari durante lo sviluppo, rispecchiassero il più possibile le abilità e le preferenze dei singoli componenti.

Una volta formati, i team hanno assegnato i due ruoli di Product Owner operativo (POo) e Scrum Master (SM) a due componenti scelti a seguito della compilazione di un questionario (presentato in appendice B) legato ad una partita al serious game Scrumble[1], in cui si simula il processo di sviluppo di un software. La parte restante dei membri ha costituito il team degli sviluppatori.

La figura del POo si è rivelata necessaria per rappresentare i team davanti ai veri Product Owner, ovvero i professori.
L'aggiunta della specifica "operativo" indica infatti che non si tratta di un membro esterno ai gruppi, bensì di uno studente con un doppio ruolo: supporto al team e sviluppatore.
Un discorso analogo può essere fatto per gli Scrum Master che, nella maggior parte dei casi, non si sono limitati soltanto a fornire supporto al proprio team ma hanno assunto in prima persona il ruolo di developer.

[1] `http://scrumble.pyxis-tech.com/`

Fase di esecuzione

Una volta formati i team e assegnati i ruoli è iniziata la fase di sviluppo: ogni team ha lavorato per almeno tre sprint della durata di tre settimane ciascuno alla realizzazione del prodotto richiesto.
L'ambiente di lavoro è stato il cosiddetto Compositional Agile System (in breve, CAS) [CMPR20], formato da diverse applicazioni open source presenti sui server dipartimentali. In particolare sono stati utilizzati:

- Taiga per il project management
- Gitlab per il controllo di versione
- Jenkins per testing e Continuous Integration (CI)
- Mattermost come chat tra i membri dei singoli team
- SonarQube per l'analisi statica del codice
- Un logger a scelta degli studenti per monitorare il ritmo di scrittura del codice

Visti i requisiti, ogni team ha stilato, utilizzando Taiga, un product backlog in cui i macro-obiettivi sono stati suddivisi in User Stories (US) nella forma

"come [tipo di utente] voglio [richiesta] per [finalità]"

Ad ogni US sono stati assegnati dei criteri di accettazione (inclusi dei test specifici) e un grado di difficoltà stimato, espresso in story points.
In base alla priorità percepita dal POo le user stories sono state distribuite tra gli sprint.
Ogni US è stata assegnata ad un membro del team che era responsabile di segnalarne il completamento su Taiga, al fine di aggiornare il grafico di burndown.
Durante questa fase, i team si sono riuniti in meeting, più o meno frequenti, per discutere dell'andamento del progetto e di come avrebbero agito i singoli membri.
All'inizio del terzo sprint, per simulare il cambiamento dinamico dei requisiti all'interno di un vero progetto, i PO hanno creato due US aggiuntive obbligatorie.

Al termine di ogni sprint è stata organizzata una riunione tra i team e i product owner in cui era richiesto di presentare una demo funzionante del prodotto all'attuale stato di avanzamento e una retrospettiva con carte Essence[2] riguardante il processo appena terminato.

[2]Le carte Essence sono reperibili sul sito `https://essence.ivarjacobson.com/`

Fase conclusiva

Una volta ultimato il prodotto, agli studenti è stato richiesto di stilare un report conclusivo in cui ricostruire quanto più fedelmente il processo seguito e di creare un video demo della propria creazione.
Infine, prima della consegna del progetto è stato richiesto ad ogni studente di compilare un questionario finale personale con domande relative alla percezione di alcuni aspetti del processo. La valutazione, assegnata a seguito di una discussione con i PO, è stata identica per tutti i membri dei singoli team.

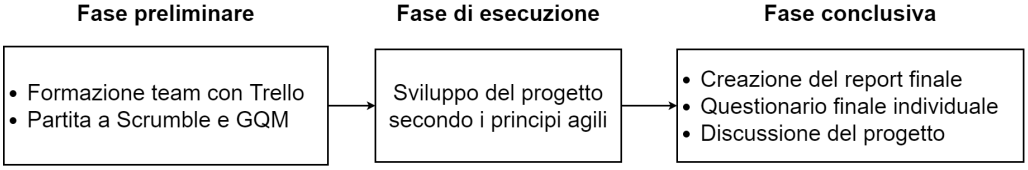

Figura 1.1: Riassunto del processo seguito

1.2 Raccolta ed elaborazione dei dati

In questa sezione viene descritto il processo di raccolta dei dati e la loro prima elaborazione al fine di renderli comparabili e utilizzabili per questa analisi.
I dati analizzati provengono dagli strumenti presenti nell'ambiente CAS, dai report finali e dai questionari individuali.

Dati ottenuti dall'ambiente CAS

Le principali fonti di dati all'interno dell'ambiente CAS sono stati Gitlab, Taiga e SonarQube. Per quanto riguarda Gitlab, è stato utilizzato prevalentemente per analizzare la struttura delle repository dei team e ottenere dati riguardanti la frequenza dei commit e alla composizione del codice per linguaggi di programmazione utilizzati.

Taiga è stato utilizzato per visualizzare il numero di sprint effettuati da ogni team, come le user stories sono state divise in task, i relativi punteggi assegnati in base alla difficoltà percepita (espressi in story points) e i grafici di burndown, che rappresentano l'andamento degli story points rimanenti al completamento dello sprint in relazione al tempo.

Infine, da SonarQube sono stati estratti i dati relativi al numero di righe di codice (LoC) e di commento scritte e alla qualità del codice, in termini di affidabi-

lità, sicurezza, manutenibilità, numero di test svolti ed eventuali blocchi di codice duplicati.

Dati ottenuti dai report finali

I report finali sono stati una fonte di dati fondamentale in quanto, oltre a contenere tutti le informazioni obbligatorie, la maggior parte dei team ha inserito elementi aggiuntivi riguardanti il proprio processo.

Tali elementi sono serviti per comprendere meglio alcune risposte contenute nel questionario individuale o a interpretare meglio certi dati.
Entrando più nel dettaglio, le relazioni finali dovevano obbligatoriamente contenere:

1. Descrizione del prodotto da costruire:
   (a) Scope e backlog di prodotto – ovvero le user stories complessive,
   (b) Diagramma dei casi d'uso,
   (c) Diagramma delle classi o degli oggetti
2. Per ogni sprint:
   (a) sprint goal,
   (b) sprint backlog,
   (c) definition of done,
   (d) per ogni user story, almeno un esempio di test fatto,
   (e) burndown dello sprint,
   (f) schermata finale di SonarQube sul codice prodotto,
   (g) risultato della retrospettiva con Essence
3. Descrizione del processo seguito (numero e durata degli sprint) e inoltre:
   (a) Autodescrizione del team e utilizzo di Trello,
   (b) Risultato dello Scrumble iniziale,
   (c) sintesi dei dati del logger/dashboard e di gitinspector,
   (d) retrospettiva finale con Essence
   (e) Diagramma di deployment del prodotto
4. Lista e descrizione degli artefatti con eventuali credenziali di accesso

Dati ottenuti dai questionari individuali

I questionari individuali sono stati la fonte principale dei dati analizzati.
Il "questionario finale individuale", riportato in Appendice A è stato sottoposto a tutti gli studenti pochi giorni prima della discussione finale con i PO tramite Google Forms.

I 37 quesiti richiedevano tipi diversi di risposte, qui riportati nella tabella seguente:

Numero domanda	Tipologia di risposta
1	Numero del team, da 1 a 17
2	Numero di matricola
3a, 3b	Risposta booleana (si / no)
4, 6, 7, 8	Linguaggio naturale
5a, 5b	Stima numerica di linee di codice
9a, 9b, 9c, 9d, 9e, 9f	Valore percentuale con somma 100%
Da 10 a 25 e da 31 a 37	Valutazione da 1 a 5
26 , 27	Valutazione da 0 a 10
28, 29, 30	Stima numerica di ore

Tabella 1.1: Tabella relativa alle tipologie di risposte alle domande del questionario finale

Come è possibile notare, le tipologie di risposte richieste sono eterogenee.

I dati sono stati aggregati e analizzati tramite un foglio di calcolo elettronico. Sono stati necessari alcuni interventi volti ad approfondire o correggere alcune risposte che, a seconda dei casi, erano state omesse o inserite in modo errato e sarebbero state inutilizzabili per l'analisi.

1.2.1 Correzione e aggregazione dei dati

In primo luogo sono state effettuate le dovute correzioni ai dati, in particolare, sono state modificate alcune risposte relative alle domande 9, in quanto, probabilmente a causa di errori di calcolo, parte delle somme percentuali non raggiungeva o superava il 100% richiesto. Tutte queste risposte sono state riportate alla somma corretta tramite proporzioni.

Inoltre, diversi studenti non avevano risposto alle domande 5a e 5b in quanto al momento della compilazione non sarebbero stati in grado di formulare le stime richieste. Gli studenti in questione sono stati contattati in un secondo momento per fornire le risposte mancanti.

In seguito alle correzioni è iniziata la fase di aggregazione dei dati.
La prima operazione di aggregazione dei dati è consistita nel suddividere tutte le risposte aperte scritte in linguaggio naturale in macro categorie, in modo da poterle utilizzare a fini statistici.

In seguito sono state calcolate media e deviazione standard (indicata con σ) di tutte le risposte che richiedessero una valutazione numerica, sia da 1 a 5 che da 0 a 10 e delle risposte con oggetto una stima numerica di righe di codice o di ore, divise sia secondo i team di appartenenza, sia esaminando l'intero campione.

Infine sono stati aggregati tutti i dati provenienti dall'ambiente CAS, in particolare i report di SonarQube, le statistiche sui product backlog di Taiga e i dati statistici delle repository di Gitlab.

1.3 Informazioni preliminari

L'analisi è stata condotta su 16 team di sviluppo, per un totale di 79 studenti. Ogni team è identificato univocamente dal proprio numero progressivo scelto in fase di formazione.

Originariamente i gruppi erano 17 ma il team 15 alla data di fine raccolta dei dati (3 maggio 2022) non ha discusso il proprio progetto ed è quindi stato escluso dallo studio.

Ogni team era formato da cinque componenti, ad esclusione dei team 8 e 17 (quattro membri) e 13 (sei membri).

Capitolo 2

ANALISI DEL PROCESSO

In questo capitolo sarà analizzato più nel dettaglio il processo descritto nel capitolo precedente.

2.1 Utilizzo di Trello

La piattaforma Trello ha permesso di creare i team tramite una dashboard in cui gli studenti hanno inserito la propria scheda con nickname e competenze in ambito di programmazione.
Per quanto riguarda la selezione delle competenze, alcuni studenti hanno inserito tutto ciò che rientrasse nelle loro abilità effettive, mentre altri solo delle preferenze per il ruolo che avrebbero voluto avere in questo progetto.

L'analisi inerente Trello si propone di capire se ad ogni studente è stato assegnato un ruolo da lui scelto sulla piattaforma oppure no, tuttavia, proprio per la doppia interpretazione che è stata data alla scelta dei ruoli, non è possibile ottenere risposta utilizzando solo Trello (infatti uno studente che è in grado di, ad esempio, testare il codice non è detto che voglia assumere tale ruolo).

Per completare l'analisi è necessario sfruttare la domanda 24 del questionario finale individuale: "La suddivisione delle task seguiva sempre le tue richieste sulla base della tua competenza in quel campo?" che ha ottenuto valutazioni positive dalla maggior parte degli studenti (media: 4,09; σ: 0,83).
Solo tre partecipanti hanno dato una valutazione inferiore a 3 su 5, in particolare nei team 7, 9 e 16.
Altri dodici hanno dato una valutazione intermedia di 3 su 5 e i restanti una valutazione superiore.

È quindi possibile concludere che nonostante Trello non sia stato utilizzato al meglio da tutti gli studenti, alla fine è stato quasi sempre possibile trovare un accordo che soddisfacesse tutti sull'assegnazione dei ruoli.

2.2 GQM di Scrumble

Dopo la formazione, ogni team ha giocato una partita al serious game Scrumble per conoscere meglio Scrum e per assegnare i ruoli di Scrum Master e Product Owner operativo.
Dopo la partita ad ogni membro dei team è stato richiesto di compilare un questionario di tipo goal-question-metric redatto da Roberto Barbone [Bar20] e riportato di seguito insieme alle tabelle con le valutazioni medie:

GOAL	QUESTION	METRIC	QUESTIONS	EVALUATION
Learn	Do team members understand the Scrum roles?	Knowledge of Scrum roles by questions	Q1	1 = *no idea of the Scrum roles* 5 = *perfect knowledge of the roles and their jobs*
	Do team members feel they learned the process?	Opinions from the participants	Q2	1 = *couldn't repeat the game* 5 = *could play the game as a Scrum Master by himself*
	Does everyone keep up with the other players?	Check during every sprint retrospective if every one is on point	Q3	1 = *totally lost* 5 = *leads the game driving the other players*
Practice	Are the game mechanics linear and repeatable?	Opinions from the participants	Q4	1 = *feels the game is unrepeatable* 5 = *feels the game could be played in any situation*
	Do team success in completing the game?	Number of User Stories completed	Q5	1 = *0 to 3 stories* 2 = *4 to 6* 3 = *7 to 9* 4 = *10 to 12* 5 = *13 to 15*
	Do team members efficiently estimate during sprint planning?	Uniformity in evaluating the size and the priority of user stories	Q6 ONLY DEV TEAM	1 = *abnormal difference from the other players* 5 = *coherent and uniform with the group most of the time*
Cooperation	Do team members know each other better?	Level of players' serenity throughout the game	Q7	1 = *never speaks with the other players* 5 = *talks friendly to anyone in every situation*
	Does the game let all players cooperate?	Contribution of every player during the game	Q8	1 = *never puts effort in doing something* 5 = *every time is willing to understand what is going on*
	Do team member consult each other about a topic?	Sharing of ideas	Q9	1 = *never asks for an opinion* 5 = *wants to discuss about every topic*
Motivation	Do team members encourage collegues in need?	Players explain something other players don't understand	Q10	1 = *not involved by the game* 5 = *always makes sure everyone is on point*
	Does PO help the team?	Quality of PO's advices to get better in the next sprints	Q11 ONLY FOR PO	1 = *poor/absent advices* 5 = *wise and helpful suggestions when is required*
	Does the team come up with good ideas?	Effectiveness of sprint retrospective	Q12	1 = *doesn't express opinions during retrospective* 5 = *feels the retrospective fundamental to express opinions*
Problem Solving	Do team members behave well when facing a problem?	Level of the technical debt at the end of the game	Q13	*On the game board, if the debt pawn is on the lowest stage, the evaluation is 5, for every higher stage it decreases by 1*
	Does team organize their tasks properly?	Average of tasks left at the end of each sprint	Q14 ONLY DEV TEAM	*Calculate the average of tasks left for each sprint:* 1 = *21+* 2 = *16-20* 3 = *11-15* 4 = *6-10* 5 = *0-5*
	Does PO plan efficiently the Sprint Backlog?	Average of tasks left at the end of each sprint	Q15 ONLY FOR PO	*Same evaluation as* **Q14** *for the PO*

Figura 2.1: Domande GQM poste in seguito alla partita a Scrumble

	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	T16	T17
Q1	3,8	3,4	3,2	3,0	2,9	4,0	3,2	4,0	4,0	3,6	4,0	4,8	4,7	3,8	4,0	2,6
Q2	3,6	5,0	4,4	4,0	2,8	5,0	4,8	3,8	4,0	4,4	4,8	4,6	4,0	3,8	5,0	2,8
Q3	5,0	4,4	3,0	4,0	3,1	5,0	5,0	4,2	4,0	3,4	4,6	4,6	4,8	4,4	4,0	2,8
Q4	5,0	2,4	3,8	2,0	5,0	4,0	3,6	5,0	5,0	4,8	4,8	3,4	3,0	2,8	4,0	3,8
Q5	4,0	5,0	5,0	5,0	1,1	5,0	5,0	3,0	5,0	5,0	2,0	3,0	2,0	5,0	5,0	4,0
Q6	5,0	4,3	4,0	5,0	3,7	3,0	5,0	4,0	4,0	4,7	4,7	5,0	4,0	4,0	4,0	3,0
Q7	5,0	4,2	4,0	4,0	3,9	3,8	5,0	4,2	4,0	4,2	4,4	3,6	5,0	4,8	4,0	5,0
Q8	4,0	3,8	3,8	5,0	3,5	4,8	4,0	4,0	5,0	3,8	4,6	4,8	5,0	2,8	5,0	4,8
Q9	5,0	4,4	3,4	4,0	4,2	5,0	5,0	4,0	4,0	4,6	4,2	4,6	5,0	5,0	5,0	4,6
Q10	3,4	2,8	3,4	3,0	3,5	5,0	5,0	4,0	3,4	4,2	5,0	5,0	4,3	4,6	5,0	4,4
Q11	5,0	2,0	2,0	4,0	2,5	4,0	4,0	5,0	4,0	4,0	4,0	4,0	5,0	5,0	5,0	4,0
Q12	5,0	1,8	3,0	4,0	3,3	4,0	3,8	3,4	3,0	3,6	4,4	4,2	4,0	4,2	4,0	3,4
Q13	5,0	5,0	5,0	5,0	5,0	5,0	4,6	3,0	5,0	5,0	4,2	5,0	4,0	4,8	5,0	4,0
Q14	4,0	4,0	5,0	5,0	2,0	3,0	5,0	4,0	5,0	5,0	5,0	3,0	3,0	4,3	5,0	4,0
Q15	4,0	4,0	5,0	5,0	2,0	4,0	5,0	4,0	5,0	5,0	5,0	3,0	3,0	3,0	5,0	4,0

Tabella 2.1: Media delle risposte del questionario GQM

Domanda	Valutazione media	Deviazione standard
Q1	3,7	0,6
Q2	4,2	0,7
Q3	4,1	0,7
Q4	3,9	1,0
Q5	4,0	1,4
Q6	4,2	0,7
Q7	4,3	0,5
Q8	4,3	0,7
Q9	4,5	0,5
Q10	4,1	0,8
Q11	4,0	1,0
Q12	3,7	0,7
Q13	4,7	0,6
Q14	4,1	1,0
Q15	4,1	1,0

Tabella 2.2: Valutazione media e deviazione std per ogni risposta del questionario GQM

Come è possibile notare dalle tabelle precedenti, nel complesso la partita svolta a Scrumble ha avuto esito positivo nella maggior parte dei casi.
Solamente il team 5 ha riferito di non essere riuscito a vincere la partita, infatti è anche quello con le valutazioni complessivamente più basse, in particolare alla domanda 5, dove la valutazione media di 1,1 indica che sono state completate poche (tra zero e tre) user stories nel corso della partita.
Altri team, in particolare 11 e 13, hanno assegnato punteggi bassi alla stessa domanda, tuttavia non hanno riportato nessuna particolare difficoltà o sconfitta nel report finale.

Il team 6 ha dichiarato di avere avuto numerose difficoltà durante lo svolgimento del gioco, tuttavia ha ottenuto punteggi più che nella media, anche se i membri hanno assegnato punteggi più bassi alle domande inerenti l'organizzazione delle task.

Le domande che nel complesso hanno ricevuto i punteggi più bassi sono state la 1 ("I membri del team hanno capito i ruoli di Scrum?") e la 12 ("Il team ha buone idee?" riferita all'efficacia delle retrospettive). Da questo è possibile dedurre che il primo impatto con i ruoli di Scrum e con le pratiche agili, in particolare le retrospettive, sia stato generalmente percepito come complesso, nonostante gli altri aspetti siano risultati chiari.

È interessante osservare le risposte alle domande Q14 e Q15 che utilizzano come metric la media delle task non completate in ogni sprint.
La differenza è che a Q14 dovevano rispondere solo gli sviluppatori mentre a Q15 solo il POo.

Si può notare come in tutti i team ad eccezione del 14 la valutazione media degli sviluppatori coincida con quella del POo, indicando che tutto il team ha lavorato in maniera sincrona.

L'utilità percepita della partita a Scrumble, secondo le risposte alla domanda 36 del questionario finale individuale è stata mediamente di 3,65 su 5 ($\sigma : 1,01$).

Nessun team ha effettuato più di una partita a Scrumble.

2.3 Taiga

Dopo la formazione, ogni team ha creato il proprio progetto sulla piattaforma Taiga, dove sono disponibili strumenti per monitorare e organizzare lo sviluppo del proprio progetto.
In particolare è possibile creare il Product backlog, ovvero l'insieme di tutte le user story che compongono il progetto, e lo sprint backlog, ovvero le user story opportunamente suddivise in task assegnate ai singoli componenti che dovranno essere svolte nel corso di un determinato sprint.
Ad ogni user story vengono preventivamente assegnati dei punteggi in quattro ambiti (UX, Design, Back e Front) secondo la difficoltà percepita dal team. La priorità delle US è invece stabilita dal POo.

La gestione delle User story avviene tramite una dashboard interattiva (o Kanban) sulla quale è possibile spostare le singole task in base allo stato di completamento, da "Nuovo" a "Concluso" passando per "In fase di esecuzione" e "Pronto per il test".
Quando tutte le task appartenenti ad una US sono segnate come completate, dal punteggio complessivo per quello sprint backlog viene sottratto il punteggio assegnato alla user story.
Grazie a questo sistema Taiga crea i cosiddetti burndown chart, ovvero dei grafici che rappresentano i punti mancanti a completare lo sprint in relazione al tempo. è anche possibile vedere una linea di completamento "ideale" che indica a che punto dovrebbe trovarsi il grafico in una determinata data.
Un esempio di burndown chart è l'immagine seguente:

Figura 2.2: Esempio di burndown chart (sprint 2 del team 3)

Taiga mette a disposizione anche una Wiki per i team dove sono stati caricati i documenti richiesti durante lo svolgimento del progetto come ad esempio le retrospettive degli sprint.

Dai dati che è possibile ottenere nella sezione "Team" si può concludere che, in generale, la gestione di Taiga non è stata affidata ad un solo membro del team, bensì ognuno ha contribuito all'aggiornamento sia della Wiki, sia del Kanban.

In alcuni team sono stati creati automatismi in grado di cambiare lo stato di una task tramite push su Gitlab, in altri lo spostamento delle task sul corretto stato di completamento era responsabilità dello studente a cui la task era assegnata.

L'utilità percepita di Taiga, secondo le risposte alla domanda 31 del questionario finale individuale è stata mediamente di 3,94 su 5 ($\sigma : 0,87$).

2.4 IDE e Logger

Ogni studente poteva scegliere liberamente l'editor di testo o IDE da utilizzare per lo sviluppo che poteva anche essere liberamente cambiato o affiancato da altri.
In particolare, dalle risposte alla domanda 4 del questionario finale si può dedurre che sono stati utilizzati i seguenti software per la scrittura del codice:

Software	**Frequenza di utilizzo (studenti)**
Atom	54
Visual Studio Code	21
IntelliJ Idea	10
Vim	2
Visual Studio Codium	1
Eclipse	1
Emacs	1

Tabella 2.3: Tabella relativa all'utilizzo dei software per scrivere codice

Sommando le frequenze di utilizzo e sapendo che nessuno studente ha utilizzato più di due software differenti, da questa tabella è possibile desumere che la scelta di utilizzare più di un editor di testo è stata effettuata da 11 sviluppatori, ovvero dal 13,92% del campione.

Analogamente, è stato caldamente suggerito di monitorare il ritmo dello sviluppo individuale utilizzando un logger che registrasse le modifiche del codice di ogni studente.
I logger suggeriti erano quelli per Atom[1] e per IntelliJ Idea[2] ma gli sviluppatori potevano sceglierne altri o cambiare liberamente durante lo sviluppo.

I logger utilizzati sono stati i seguenti:

[1] `https://github.com/elPeroN/Atom-logger`
[2] `https://gitlab.com/fulvio1993/logger-intellij`

Logger	Frequenza di utilizzo (studenti)
Logger per Atom	53
Logger per IntelliJ Idea	3
Logger per Eclipse	1
WakaTime	1
Nessun logger	22

Tabella 2.4: Tabella relativa alla frequenza di utilizzo dei Logger

L'ultima riga della tabella comprende sia coloro che non hanno mai utilizzato un logger, sia coloro i quali eventualmente non hanno riportato di averlo usato, né rispondendo al questionario, né nel report finale del proprio team.
Alcuni studenti hanno riportato di aver utilizzato il logger per Atom in maniera passiva, ovvero scrivendo il codice su un altro IDE mantenendo Atom aperto in background in modo che il logger rilevasse le modifiche effettuate.

Come si può notare da questa tabella sommando le frequenze di utilizzo, la scelta di utilizzare più di un logger è stata effettuata da un solo sviluppatore (1,27% del campione).

2.5 Gitlab

Prima di iniziare lo sviluppo del progetto, ogni team ha creato la propria repository su Gitlab.

I team 3 e 6 hanno creato due repository, una per gestire il front-end e una per il back-end.
La repository è stata utilizzata per conservare il codice scritto ed avere la possibilità di effettuare version control.

La piattaforma offre la possibilità di visualizzare interessanti statistiche, come ad esempio la distribuzione dei commit per giorni della settimana o i linguaggi di programmazione da cui è composto il progetto.

Questi dati sono ora presentati in forma aggregata con l'ausilio di grafici:

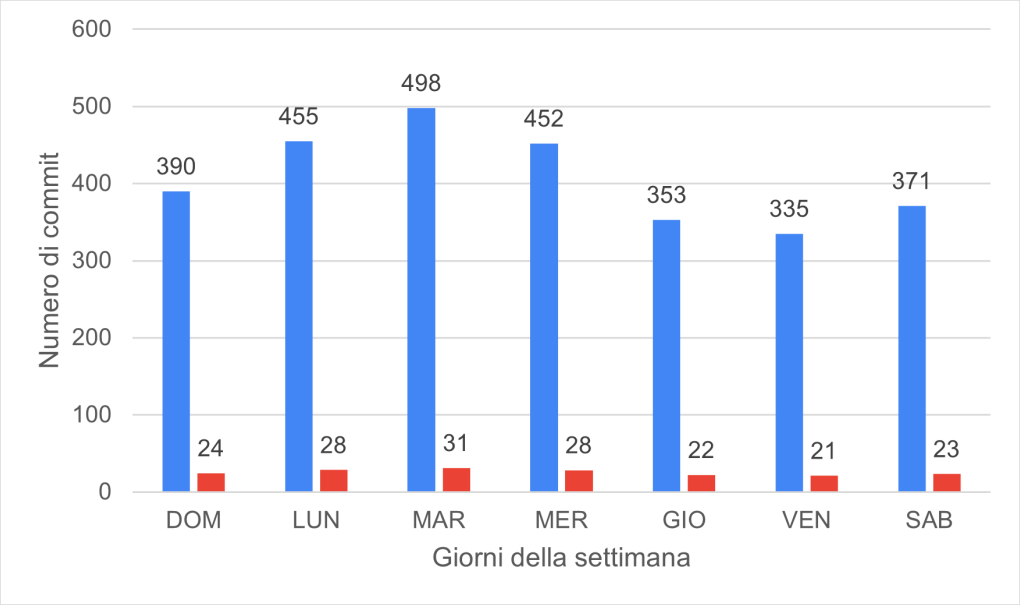

Figura 2.3: Distribuzione dei commit con media per giorni della settimana

Da questo grafico è possibile notare come siano stati i primi tre giorni della settimana quelli con più commit in assoluto, seguiti da un calo nella seconda metà.

Ogni team ha goduto della massima autonomia organizzativa da questo punto di vista quindi non è possibile notare una distribuzione uniforme nei commit dei singoli team. Ad esempio il team 16 ha effettuato globalmente solo due commit di martedì ma ben 51 di mercoledì.

Inoltre alcuni team hanno effettuato molti più commit di altri quindi bisognerebbe prendere in considerazione il valore di un singolo commit rispetto al totale.

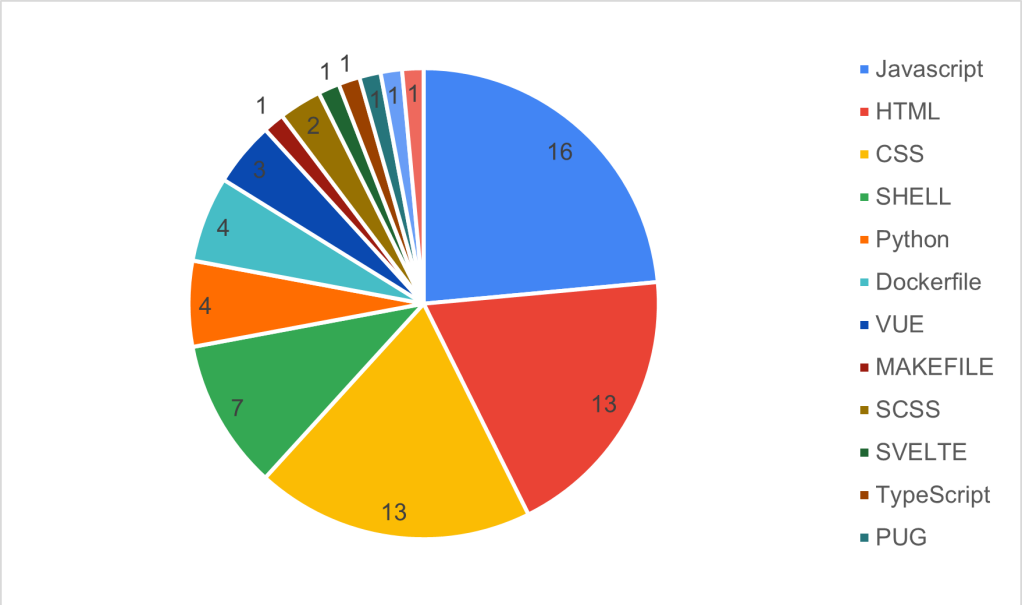

Figura 2.4: Numero di team che hanno utilizzato un determinato linguaggio

Da questo grafico,invece, è possibile notare come i classici strumenti di sviluppo web (Javascript, HTML e CSS) siano stati utilizzati da quasi la totalità dei team.

Infatti la maggior parte dei progetti è stata consegnata sotto forma di web app con hosting su container Docker di dipartimento. Un'altra scelta è stata quella di creare un'applicazione sfruttando Python come back-end.

Alcuni progetti sono stati corredati di makefile o dockerfile a seconda delle esigenze.

L'utilità percepita di Gitlab, secondo le risposte alla domanda 32 del questionario finale individuale è stata mediamente di 4,61 su 5 ($\sigma : 0,71$).

2.5.1 Jenkins

La suite CAS mette a disposizione anche Jenkins, tool per impostare la Continuous Integration (CI) e il Continuous Deployment (CD).

In particolare, Jenkins permette di eseguire alcune azioni a seguito di un push su Gitlab, come ad esempio il trasferimento del nuovo codice sul server host della propria web app, anche a condizione che i test di SonarQube diano esito positivo.

Nonostante fosse consigliato, non tutti i team ne hanno fatto uso, anche a causa delle difficoltà incontrate in fase di impostazione per l'accesso ai server dipartimentali.
Alcuni team hanno impostato Jenkins in locale su una propria macchina in modo da ovviare ad alcuni di questi problemi.

L'utilità percepita di Jenkins, secondo le risposte alla domanda 35 del questionario finale individuale è stata mediamente di 2,94 su 5 ($\sigma : 1,42$).
L'alta deviazione standard è data dalla disomogeneità delle risposte, infatti i team che ne hanno fatto uso hanno indicato un'utilità molto più alta rispetto ai team che non lo hanno usato.

2.6 SonarQube

A partire dal secondo sprint è stato reso disponibile SonarQube.
Si tratta della piattaforma dedicata all'analisi automatizzata del codice: esso fornisce una valutazione letterale per affidabilità. sicurezza e manutenibilità del codice. Inoltre calcola il la percentuale di test effettuati rispetto alle funzioni scritte e, se presenti, individua blocchi di codice duplicati.

Se il progetto non risulta "product ready", ovvero risulta privo di sufficienti test o una delle valutazioni letterali risulta più bassa di A, SonarQube lo rende evidente comunicando il fallimento del "Quality Gate".

Nel caso di progetti con CI/CD, i test di qualità potevano essere fatti partire automaticamente a seguito di un push su Gitlab e il deployment automatico poteva essere impedito in caso di fallimento.

È da evidenziare come tutti i team fossero obbligati ad eseguire almeno un test per user story ma in alcuni casi questi non compaiano su SonarQube, probabilmente a causa di un errore o dei criteri di riconoscimento dei test all'interno dei progetti.

Grazie ai dati sulla struttura del codice è stato anche possibile concludere che un progetto è composto mediamente da 2345 linee di codice e da 276 commenti (10,53% del totale),

L'utilità percepita di SonarQube, secondo le risposte alla domanda 33 del questionario finale individuale è stata mediamente di 3,29 su 5 ($\sigma : 0,96$).

2.7 Mattermost

Mattermost è la piattaforma di messaggistica istantanea consigliata per questo progetto. Permette ai team di creare gruppi e di utilizzare una chat per comunicare.

Leggendo le risposte date al questionario finale individuale è possibile accorgersi del fatto che Mattermost è stato il programma CAS meno utilizzato e più criticato. Questo probabilmente è dovuto al fatto che gli studenti fossero già abituati all'uso di altre piattaforme (come ad esempio Discord o Telegram) e che quindi abbiano visto Mattermost quasi come un'imposizione, nonostante di fatto non fosse così.

Grazie all'integrazione con Gitlab e Jenkins era possibile ricevere messaggi automatici a seguito di ogni push e al termine del deployment automatico.

L'utilità percepita di Mattermost, secondo le risposte alla domanda 34 del questionario finale individuale è stata mediamente di 2,33 su 5 ($\sigma : 1,31$). Anche in questo caso l'alta deviazione standard indica disaccordo tra i team.

2.8 Pratiche agili utilizzate

La domanda numero 6 del questionario finale individuale richiedeva di indicare un elenco di pratiche agili utilizzate sia dai singoli studenti che dal team durante lo sviluppo.
La tabella che segue indica secondo quanti studenti è stata utilizzata una determinata pratica agile:

Pratiche agili	**Studenti**
Sprint planning	73
Pair programming	73
Retrospettiva	48
Daily scrum	33
Sprint review	20
Weekly scrum	17
Refactoring	13
Mob programming	9
Planning poker	8
Testing	8
Continuous integration	6
Prioritization	5
Face-to-face communication	4
Versioning	4
Product backlog	4
Review	3
Chase the state (essence)	2
Sprint goal	2
Comunicazione stretta	2
Sprint backlog	2
Cross functional team	2
Timeboxing	1
Self tracking	1
Sviluppo iterativo	1
Feedback	1

Tabella 2.5: Tabella inerente le pratiche agili utilizzate durante lo sviluppo

Come si può notare, alcune delle pratiche fondamentali dello sviluppo agile, come il product backlog, sono state menzionate da pochi studenti nonostante fossero obbligatorie durante lo svolgimento del progetto.

Questo probabilmente è dovuto al fatto che nel momento della compilazione del questionario gli studenti hanno inserito le pratiche che più hanno utilizzato in prima persona o che hanno percepito maggiormente come utili.

Dai report finali è possibile tuttavia notare una tendenza quasi contraria a quella espressa da questa tabella: in quel contesto infatti sono state messe in evidenza pratiche che nel questionario finale sono state poco menzionate, in particolare il versioning e le retrospettive.

Segue un grafico riassuntivo delle pratiche agili menzionate in relazione al numero di citazioni:

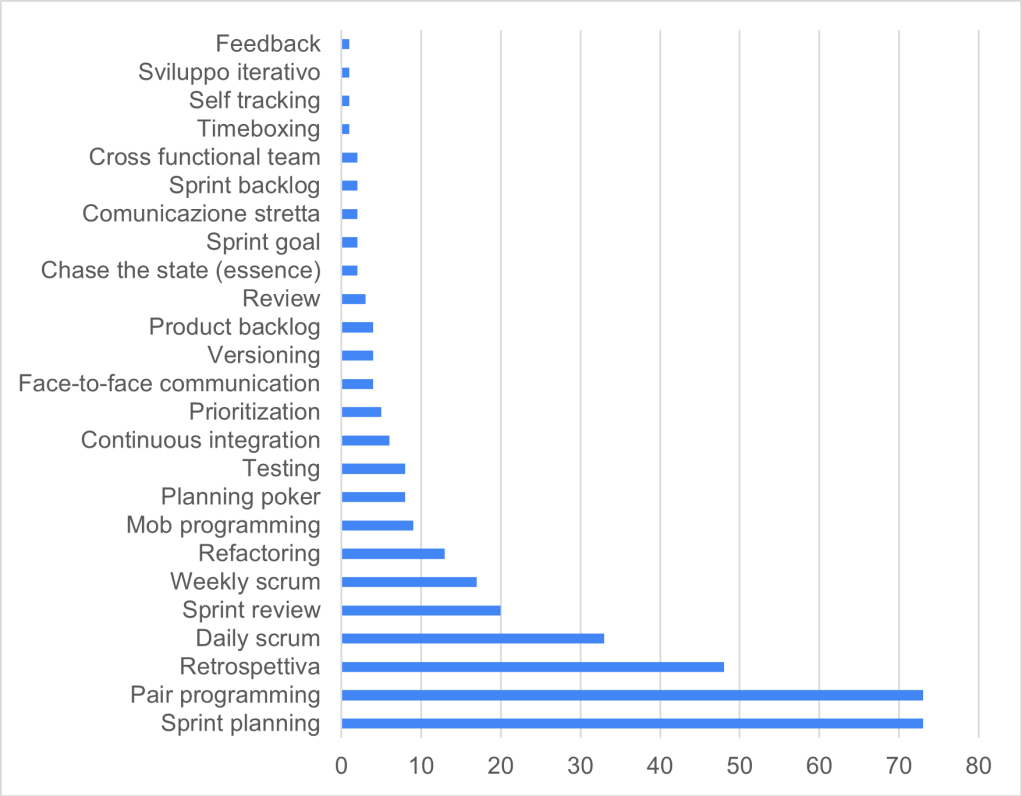

Figura 2.5: Numero di citazioni per ciascuna pratica agile

2.8.1 Retrospettive

Una delle pratiche agili obbligatorie durante lo svolgimento del progetto erano le retrospettive con carte Essence.

In breve, si tratta di carte con la descrizione di un'attività tipica dello sviluppo agile che vanno collocate all'interno di una matrice sulla base dell'autovalutazione che il team si assegna per lo svolgimento dell'attività stessa (bene, normale, male) e sulla base della priorità che ha quell'attività nel contesto dello sviluppo (alta, media, bassa).

Inoltre è presente una quarta colonna in cui il team prende appunti sugli aspetti che può migliorare nel prossimo sprint, sempre sulla base della priorità dei miglioramenti.

	Bene	Normale	Male	Miglioramenti
Priorità alta				
Priorità media				
Priorità bassa				

L'utilità percepita delle retrospettive, secondo le risposte alla domanda 37 del questionario finale individuale è stata mediamente di 3,75 su 5 ($\sigma : 0,98$)

Capitolo 3

MODELLO DI QUALITÀ DEL TEAMWORK

Una volta raccolti i dati ed effettuate le dovute modifiche, è stato possibile applicare il modello di qualità del teamwork.

Si tratta di un modello inizialmente sviluppato e descritto da Martin Hoegl e Hans Georg Gemuenden [HG01], basato su analisi delle interazioni interne ai team, analisi delle performance e analisi della soddisfazione dei team in relazione al prodotto finale.

Successivamente questo modello è stato applicato ai team di sviluppo Agile da Dingsør et al. [LSD+16]

Questo modello mette in relazione le performance del team, la qualità del teamwork, misurata secondo alcune categorie, e il successo personale dei membri del team per restituire una stima della qualità totale.
Nelle sezioni che seguono saranno presentate ed effettuate tali analisi.

3.1 Analisi delle performance

3.1.1 Analisi della produttività

In questa sezione sarà presentata l'analisi della produttività dei team effettuata basandosi su quanto riportato nei quesiti 5a e 5b (Stima del numero di LoC scritte in totale e quelle rimaste) e 28, 29 e 30 (stima delle ore di coding, incontri di team e per lo SM di supporto al team) del questionario finale individuale.

Questa analisi si divide nelle tre parti di seguito riportate:

- Analisi della produttività del team di sviluppatori, incluso il POo;
- Analisi della produttività dello Scrum Master
- Analisi dell'andamento della produzione in relazione agli sprint e al numero di ore di lavoro stimate

Per ognuna di queste sarà presentata una tabella con i dati in forma aggregata ottenuti dalle risposte ad alcune domande del questionario finale individuale, dai report finali e da Taiga.
Seguirà una breve spiegazione dei dati e una sintetica analisi.

Analisi della produttività degli sviluppatori

Team	LoC verificate	LoC sviluppatori	Ore sviluppatori	LoC/h
1	1148 Diff: -1852	m: 888 σ: 371	m: 40 σ: 27	22
2	1858 Diff: -448	m: 1404 σ: 1222	m: 65 σ: 78	22
3	1640 Diff: -2810	m: 1400 σ: 1417	m: 96 σ: 41	15
4	4609 Diff: 1209	m: 1050 σ: 208	m: 58 σ: 19	18
5	2043 Diff: 343	m: 1250 σ: 2105	m: 49 σ: 51	26
6	3312 Diff: -3588	m: 1113 σ: 686	m: 103 σ: 44	11
7	1704 Diff: 674	m: 263 σ: 149	m: 76 σ: 51	3
8	4380 Diff: -470	m: 2433 σ: 2290	m: 110 σ: 78	22
9	3043 Diff: 768	m: 3000 σ: 4021	m: 69 σ: 9	44
10	2433 Diff: -487	m: 1025 σ: 465	m: 90 σ: 43	11
11	4472 Diff: 422	m: 1450 σ: 1702	m: 71 σ: 65	20
12	1706 Diff: -544	m: 4025 σ: 6324	m: 113 σ: 15	36
13	1949 Diff: -1301	m: 2840 σ: 4046	m: 85 σ: 39	33
14	2530 Diff: -470	m: 1225 σ: 602	m: 99 σ: 51	12
16	3951 Diff: 651	m: 1400 σ: 868	m: 86 σ: 17	16
17	1172 Diff: -6201	m: 3760 σ: 3366	m: 113 σ: 21	33

Tabella 3.1: Analisi della produttività degli sviluppatori (POo incluso)

Nella seconda colonna sono riportate per ciascun team le linee di codice che compongono il progetto secondo quanto riportato dall'analisi di SonarQube.

"Diff" indica la differenza tra le LoC verificate e la media di LoC stimate come rimaste nel codice finale dai team.
Una differenza negativa significa quindi che gli studenti hanno sovrastimato le righe di codice rimaste.
Viceversa una differenza positiva indica una sottostima.

Nella terza colonna sono riportate media e deviazione standard delle righe di codice dichiarate in totale dagli sviluppatori, comprese quelle eliminate.
Si nota come in alcuni team i valori di σ siano particolarmente elevati, questo perché alcuni membri hanno inserito una stima molto più elevata degli altri.
Un esempio di ciò sono i team 12 e 13, dove due studenti hanno dichiarato di aver scritto un totale rispettivamente di 13500 e 10000 LoC.
Mediamente uno sviluppatore dichiara di aver scritto 1778 righe di codice.

Si può notare come generalmente tutti i team abbiano avuto difficoltà nello stimare il numero di LoC, sia quelle scritte che quelle rimaste, tendendo perlopiù ad una sovrastima (mediamente di 882 LoC).

Questo porta chiaramente ad una modifica dei risultati relativi anche alla produttività in termini di LoC/h. Diversi team infatti riportano delle produzioni orarie molto elevate proprio a causa della sovrastima di linee di codice scritte.

Mediamente uno sviluppatore dichiara di aver programmato per 83 ore, anche se sono presenti picchi fino a 200 ore in alcuni team.

Analisi della produttività degli Scrum Master e riunioni di team

Team	LoC SM	Ore coding SM	Supporto (h)	Riunioni di team (h)
1	1500	35	30	m: 46, σ: 11
2	600	72	54	m: 35, σ: 13
3	1500	200	20	m: 89, σ: 43
4	1500	80	10	m: 11, σ: 2
5	600	10	20	m: 50, σ: 32
6	7000	190	15	m: 72, σ: 28
7	600	60	35	m: 57, σ: 20
8	4000	200	42	m: 79, σ: 44
9	500	60	10	m: 26, σ: 11
10	400	40	25	m: 43, σ: 9
11	450	5	20	m: 30, σ: 6
12	0	0	32	m: 35, σ: 10
13	450	100	60	m: 58, σ: 27
14	1000	150	35	m: 68, σ: 34
16	2000	120	40	m: 50, σ: 17
17	1646	130	60	m: 56, σ: 21

Tabella 3.2: Analisi della produttività degli Scrum Master e riunioni di team

Questa tabella prende in considerazione i dati dichiarati dallo Scrum Master di ciascun team e quelli relativi alla stima di ore passate in riunioni di team.

Mediamente uno SM dichiara di aver scritto 1484 linee di codice. Si tratta di un dato interessante considerando che è piuttosto vicino alla produzione media di uno sviluppatore e che lo SM dovrebbe più essere una figura di supporto al team.

In diversi casi lo SM ha contribuito più di tutti alla produzione codice scritte, come si può notare ad esempio nei team 6 e 8.
Solo nel caso del team 12 lo SM non ha prodotto codice.

Passando alle ore trascorse a programmare, si può notare come alcuni Scrum Master abbiano lavorato un ammontare di tempo almeno pari ai loro compagni developer.

In altri casi, come nei team 5 e 11, il basso numero di ore di coding e l'alto numero di LoC portano a tassi di produzione oraria esagerati.

Mediamente uno Scrum Master dichiara di aver programmato per 91 ore, 8 in più di uno sviluppatore. Questo dato nella maggior parte dei casi è poco veritiero ed è probabilmente da ricondurre alla difficoltà segnalata precedentemente di automisurare il proprio lavoro sia in termini di LoC sia in termini di ore, tuttavia è indice del fatto che gli SM hanno assunto anche il ruolo di developer nella maggior parte dei gruppi.

Per quanto riguarda le ore di supporto fornite al team, queste si attestano mediamente a 32. I valori più alti si registrano comunque in corrispondenza di SM particolarmente efficienti anche nella produzione di codice.

Emblematico il caso del team 12, dove lo SM non ha prodotto codice e ha fornito relativamente poche ore di supporto al team. Infatti nonostante si tratti esattamente della quantità media di ore, bisogna considerare che tutti gli altri SM nel frattempo supportavano il team svolgendo anche il ruolo di sviluppatore. Nel caso dello Scrum Master, in particolare nel contesto di questo progetto, ci si dovrebbe aspettare quantomeno che ad una diminuzione delle ore di programmazione corrisponda un aumento delle ore di supporto al team.

L'ultima colonna indica il tempo medio stimato dai team passato in riunioni.
Mediamente un team ha speso 50 ore ad effettuare riunioni, tuttavia alcuni riportano un valore di deviazione standard eccessivamente elevati.
Questo potrebbe essere dovuto a due fattori: il disaccordo del team sul numero di ore di riunione o, più probabilmente, le assenze di alcuni membri alle riunioni. Infatti, trattandosi di una misura personale, gli studenti che per qualsiasi ragione non hanno partecipato ad alcune riunioni, hanno indicato meno ore, aumentando il valore di σ.

Un esempio è dato dal team 3 che ha riportato sulla Wiki di Taiga tutte le presenze alle riunioni, evidenziando che alcuni membri sono stati sempre presenti, mentre altri hanno saltato degli incontri.

Analisi dell'andamento della produzione

Team	S0	S1	S2	S3	S4	S5	S6	Stima ore
1	/	14,29%	44,22%	41,50%	/	/	/	195
2	/	9,66%	34,03%	46,22%	10,08%	/	/	332
3	/	16,84%	15,24%	67,91%	/	/	/	585
4	/	1,22%	31,14%	67,64%	/	/	/	310
5	18,71%	12,93%	35,71%	13,95%	18,71%	/	/	255
6	/	15,95%	16,56%	19,17%	11,96%	22,55%	13,80%	600
7	14,94%	21,84%	0,00%	42,53%	20,69%	/	/	365
8	/	2,77%	9,00%	21,19%	22,30%	44,74%	/	530
9	2,00%	22,72%	24,72%	38,31%	12,25%	/	/	335
10	/	16,77%	22,54%	60,69%	/	/	/	400
11	/	7,16%	25,58%	42,20%	25,06%	/	/	290
12	/	17,53%	20,77%	25,56%	36,15%	/	/	450
13	/	6,67%	42,67%	36,00%	14,67%	/	/	525
14	5,01%	15,83%	46,04%	33,12%	/	/	/	545
16	10,65%	21,01%	7,40%	20,41%	16,27%	24,26%	/	465
17	/	17,03%	5,30%	26,65%	51,02%	/	/	470

Tabella 3.3: Analisi dell'andamento della produzione

In questa tabella è riportata la ripartizione percentuale di completamento degli story points previsti da ogni team e la stima delle ore totali che ci sono volute per completare il progetto.

È stata scelta questa tipologia di misurazione in quanto gli story points non sono omogenei nell'indicare la difficoltà di una US. Infatti i gruppi hanno utilizzato misurazioni differenti: per un team la difficoltà massima poteva essere indicata da 100 story points mentre per un altro da 5. Non essendo questo massimo dichiarato, l'unico modo per monitorare l'andamento della produzione è sfruttare la percentuale di SP completati rispetto al totale per ogni sprint.

Prima di iniziare l'analisi occorre specificare che lo "Sprint 0" (indicato con S0) è uno sprint preparatorio avvenuto nello stesso periodo dello Sprint 1 e solitamente utilizzato come contenitore di task da svolgere prima di iniziare a programmare, come terminare la partita a Scrumble o creare un account per gli strumenti CAS.

Non tutti lo hanno considerato uno Sprint a sé stante: molti hanno persino omesso le task preparatorie o le hanno accorpate a quelle dello Sprint 1 sotto un'unica user story fittizia.

Dai dati riportati nella tabella precedente è possibile ricavare che solo 5 team hanno scelto di indicare uno Sprint 0.

Inoltre si possono ottenere informazioni su quanti sprint siano stati necessari per terminare il progetto, in particolare:

- cinque team hanno impiegato 3 sprint
- otto team hanno impiegato 4 sprint
- due team hanno impiegato 5 sprint
- un solo team ha impiegato 6 sprint

Passando all'andamento della produzione è possibile accorgersi di come lo Sprint 3 sia stato in assoluto quello più produttivo, dove alcuni team hanno completato più del 60% degli story points previsti.
Questo potrebbe essere dovuto al fatto che diversi team hanno terminato il progetto, o comunque erano vicini alla fine, durante il terzo sprint e che quindi si siano impegnati maggiormente, aumentando di conseguenza il ritmo di lavoro.

Da notare anche come la suddivisione del lavoro non sia in nessun caso perfetta ma piuttosto altalenante.

Un caso particolare è rappresentato dal team 7 il cui Sprint 2 è indica lo 0% di completamento: dal report finale è possibile scoprire che il team ha lavorato ma ha incontrato difficoltà nel completamento delle US previste, trasferendole così allo sprint successivo.

3.1.2 Analisi della qualità del prodotto finale

La tabella che segue rappresenta la conversione in percentuale sia delle valutazioni date da SonarQube ad affidabilità, sicurezza e manutenibilità, sia delle valutazioni finali date dai PO a seguito della discussione del progetto.

In particolare, per le valutazioni letterali, considerando A come il 100%, è stato sottratto un 20% per ogni lettera successiva, fino ad F, corrispondente allo 0%. Invece per le valutazioni finali il voto espresso in trentesimi è stato semplicemente convertito in percentuale.

Team	Affidabilità	Sicurezza	Manutenibilità	Valutazione	Media
1	100%	100%	100%	83,3%	95,8%
2	40%	60%	100%	83,3%	70,8%
3	100%	100%	100%	93,3%	98,3%
4	100%	100%	100%	100,0%	100,0%
5	60%	60%	100%	86,7%	76,7%
6	100%	100%	100%	93,3%	98,3%
7	100%	100%	100%	76,7%	94,2%
8	100%	100%	100%	70,0%	92,5%
9	100%	100%	100%	100,0%	100,0%
10	100%	100%	100%	100,0%	100,0%
11	60%	100%	100%	80,0%	85,0%
12	60%	60%	100%	93,3%	78,3%
13	100%	100%	100%	86,7%	96,7%
14	100%	100%	100%	90,0%	97,5%
16	20%	60%	100%	70,0%	62,5%
17	100%	100%	100%	80,0%	95,0%

Tabella 3.4: Analisi della qualità del prodotto finale secondo SonarQube e i PO

Per prima cosa, è possibile notare che tutti i team hanno ottenuto il punteggio massimo in manutenibilità.
In secondo luogo come il focus della valutazione finale del progetto non fosse il prodotto finale, bensì il processo seguito, infatti a parità di valutazioni da parte di SonarQube, i PO hanno dato voti differenti.

La valutazione media è stata di circa 26 trentesimi corrispondente all'86,7%, mentre la qualità media dei prodotti finali si attesta al 90,1%.

3.2 Analisi delle interazioni interne al team

In questa sezione sarà presentata l'analisi delle interazioni interne al team sfruttando le domande del questionario finale basate su:

- comunicazione
- coordinazione
- prioritizzazione dello sforzo
- mutuo supporto
- coesione
- bilanciamento nella contribuzione dei membri

Per ogni categoria saranno presentate le risposte alle relative domande sotto forma di percentuale calcolata secondo la formula:

$$\frac{(Media\ aritmetica\ delle\ risposte\ alla\ domanda) - 1}{4} \times 100$$

in questo modo l'intervallo dei possibili valori di risposta viene convertito da $[1-5]$ a $[0-4]$, permettendo di far cominciare da zero l'intervallo.

Di seguito sono riportati i risultati suddivisi per categoria.

3.2.1 Comunicazione

Per questa categoria sono state considerate le risposte alle domande 10, 11, 12, 13, 14, 28 e 29 del questionario finale:

1. Tempo speso in incontri di team? e Tempo speso in lavoro individuale?
2. C'è stata una comunicazione spontanea frequente nel team durante il lavoro al progetto?
3. La comunicazione di informazioni importanti è sempre stata condivisa con tutti i membri del team?
4. La comunicazione di informazioni nel team è sempre avvenuta in modo tempestivo?
5. Il team è stato in grado di fornirti (o ti ha aiutato a reperire) delle informazioni utili e precise quando hai avuto dei dubbi?
6. Il team ha mai chiesto aiuto all'esterno?

Team	Riunioni	Spontaneità	Condivisione	Tempestività	Reperire informazioni	Aiuto esterno	Media
1	54,12%	90% σ:0,55	100% σ: 0	80% σ: 0,84	70% σ: 1,1	65% σ: 0,89	76,52%
2	34,65%	60% σ:1,14	90% σ: 0,89	65% σ: 1,14	75% σ: 0,71	70% σ: 0,84	65,77%
3	43,20%	80% σ:0,84	60% σ: 0,55	65% σ: 0,89	75% σ: 1,41	85% σ: 0,89	68,03%
4	15,07%	95% σ:0,45	100% σ: 0	95% σ: 0,45	95% σ: 0,45	95% σ: 0,45	82,51%
5	49,50%	80% σ:0,45	95% σ: 0,45	90% σ: 0,55	85% σ: 0,89	85% σ: 0,55	80,75%
6	37,50%	80% σ:0,45	100% σ: 0	95% σ: 0,45	95% σ: 0,45	90% σ: 0,55	82,92%
7	43,85%	75% σ:0,71	95% σ: 0,45	85% σ: 0,55	80% σ: 0,45	70% σ: 0,84	74,81%
8	37,28%	68,75% σ:0,96	87,5% σ: 0,58	81,25% σ: 0,96	81,25% σ: 0,96	81,25% σ: 0,5	72,88%
9	27,96%	90% σ:0,55	90% σ: 0,55	80% σ: 0,45	85% σ: 0,55	50% σ: 0	70,49%
10	35,17%	100% σ:0	90% σ: 0,55	85% σ: 0,89	100% σ: 0	55% σ: 0,84	77,53%
11	34,09%	80% σ:1,3	95% σ: 0,45	95% σ: 0,45	90% σ: 0,55	65% σ: 0,55	76,52%
12	28,23%	95% σ:0,45	95% σ: 0,45	85% σ: 0,55	80% σ: 1,3	70% σ: 1,1	75,54%
13	40%	66,67% σ:0,82	91,67% σ: 0,52	83,33% σ: 0,52	91,67% σ: 0,52	70,83% σ: 0,75	74,03%
14	38,42%	100% σ:0	100% σ: 0	85% σ: 0,55	80% σ: 0,84	80% σ: 0,84	80,57%
16	34,97%	65% σ:1,52	95% σ: 0,45	75% σ: 0,71	55% σ: 0,84	55% σ: 0,84	63,33%
17	32,37%	68,75% σ:1,26	75% σ: 0,82	62,5% σ: 0,58	81,25% σ: 0,96	100% σ: 0	69,98%

Tabella 3.5: Analisi della comunicazione nei team

Per quanto concerne la media delle riunioni, essa indica quante ore di riunioni sono state svolte rispetto alle ore totali di lavoro per ciascun team. Mediamente un team ha dedicato il 36,7% del proprio tempo alle riunioni.
Il team 1 è quello che in assoluto ha dedicato la maggior parte del tempo alle riunioni con un valore del 54,1%.

La domanda 2 sulla spontaneità della comunicazione ha ottenuto un risultato medio di 80,9%. Si evidenziano alti valori della deviazione standard in corrispondenza dei team 2, 11, 16 e 17, dove alcuni membri hanno dato una valutazione molto più bassa rispetto agli altri.

Si segnala anche un valore di deviazione standard pari a 0 in corrispondenza del team 10.

La terza domanda ha ottenuto un punteggio medio del 91,2% con valori di σ relativamente bassi, indice di accordo tra i membri dei team.

La domanda numero quattro riguardo la tempestività della comunicazione ha un punteggio medio di 81,7%. Un alto valore della deviazione standard è presente solo in corrispondenza del team 2 dove uno studente si è discostato dagli altri con una valutazione di 2 su 5.

Per quanto riguarda l'aiuto nel reperire informazioni, la domanda ha ottenuto un punteggio medio di 82,5%. In questo caso la deviazione standard ha minore influenza dato che uno studente che non ha mai necessitato di aiuto avrà fornito una valutazione più bassa.

Infine l'ultima domanda per questa categoria ha ottenuto un punteggio medio di 74,2%. Per il calcolo del punteggio è stato considerato il risultato complementare rispetto a quello ottenuto (100% - risultato ottenuto) dato che una risposta con valore massimo indica che il team ha sempre chiesto aiuto all'esterno, il che, dal punto di vista di questa analisi, è negativo.

In conclusione il risultato medio per la categoria di comunicazione è 74,51%.

3.2.2 Coordinazione

Per questa categoria sono state considerate le risposte alle domande 15 e 16 del questionario finale:

1. Quando hai dovuto svolgere task che erano strettamente correlati con quelli di un compagno avete avuto dei problemi nel coordinarvi?
2. C'è sempre stata un'accettazione condivisa nell'assegnazione dei task nel team?

Team	Coordinazione task correlate	Assegnazione task	Media
1	m: 70%, σ: 0,84	m: 95%, σ: 0,45	82,5%
2	m: 75%, σ: 0,71	m: 95%, σ: 0,45	85%
3	m: 80%, σ: 1,1	m: 90%, σ: 0,55	85%
4	m: 90%, σ: 0,55	m: 80%, σ: 0,45	85%
5	m: 80%, σ: 0,84	m: 100%, σ: 0	90%
6	m: 95%, σ: 0,45	m: 100%, σ: 0	97,5%
7	m: 65%, σ: 0,55	m: 95%, σ: 0,45	80%
8	m: 81,25%, σ: 0,5	m: 100%, σ: 0	90,63%
9	m: 80%, σ: 0,45	m: 95%, σ: 0,45	87,5%
10	m: 65%, σ: 1,52	m: 80%, σ: 0,45	72,5%
11	m: 60%, σ: 1,14	m: 80%, σ: 0,84	70%
12	m: 80%, σ: 0,84	m: 95%, σ: 0,45	87,5%
13	m: 79,17%, σ: 1,33	m: 100%, σ: 0	89,6%
14	m: 65%, σ: 1,52	m: 85%, σ: 0,55	75%
16	m: 65%, σ: 1,14	m: 95%, σ: 0,45	80%
17	m: 75%, σ: 0,82	m: 87,5%, σ: 0,58	81,25%

Tabella 3.6: Analisi della coordinazione nei team

Per quanto riguarda la prima domanda, il punteggio medio è di 75,3%.
Si evidenzia una deviazione standard generalmente piuttosto alta, indice del fatto che in quasi nessun team tutti si siano adeguatamente coordinati.

La seconda domanda ha ottenuto un punteggio medio del 92% con valori di σ bassi, in alcuni casi pari a 0. Questo, insieme ai dati ottenibili dai report finali, permette di concludere che nei team la suddivisone delle task sia stata fatta sulla base delle capacità e preferenze di ognuno.

Il punteggio medio per questa categoria è 83,68%.

3.2.3 Prioritizzazione dello sforzo

Per questa categoria sono state considerate le risposte alla domanda 20 del questionario finale: "Secondo te ogni membro del team ha dato la sua contribuzione al meglio delle sue capacità e disponibilità di tempo?"

Team	Contribuzione al meglio delle capacità
1	m: 70%, σ: 0,45
2	m: 55%, σ: 0,45
3	m: 70%, σ: 0,45
4	m: 85%, σ: 0,89
5	m: 80%, σ: 1,1
6	m: 85%, σ: 0,89
7	m: 65%, σ: 1,14
8	m: 56,25%, σ: 1,26
9	m: 80%, σ: 0,45
10	m: 85%, σ: 0,55
11	m: 70%, σ: 1,1
12	m: 80%, σ: 0,84
13	m: 62,5%, σ: 1,05
14	m: 80%, σ: 0,84
16	m: 65%, σ: 1,14
17	m: 68,75%, σ: 0,96

Tabella 3.7: Analisi della prioritizzazione dello sforzo

Questa domanda ha ricevuto un punteggio medio di 72,34% e presenta alcune differenze tra i team.

Infatti se in alcuni sono presenti valutazioni alte con bassi valori della deviazione standard, in altri sono presenti punteggi piuttosto bassi, mantenendo però valori di σ bassi.
In questi team, come ad esempio il 2, si è ritenuto di fatto che non tutti abbiano contribuito al massimo delle proprie capacità.

Un terzo caso è rappresentato dai team con deviazione standard elevata: in questi team alcuni membri hanno assegnato punteggi alti e altri punteggi inferiori.
Ad esempio nel team 5 uno studente ha assegnato un punteggio di 2 su 5 a questa domanda.

3.2.4 Mutuo supporto

Per questa categoria sono state considerate le risposte alle domande 17, 18 e 19 del questionario finale:

1. Se hai avuto difficoltà con lo svolgimento dei tuoi task hai avuto aiuto dal team?
2. Sul darsi obiettivi e risolvere questioni importanti il team ha sempre risolto in fretta?
3. Ritieni che la cooperazione sia sempre andata bene nel team?

Team	Aiuto	Obiettivi e questioni importanti	Cooperazione	Media
1	m: 80%, σ: 1,3	m: 75%, σ: 0,71	m: 90%, σ: 0,55	81,67%
2	m: 75%, σ: 1,22	m: 35%, σ: 0,55	m: 60%, σ: 0,89	56,67%
3	m: 80%, σ: 1,3	m: 95%, σ: 0,45	m: 70%, σ: 0,45	81,67%
4	m: 90%, σ: 0,89	m: 90%, σ: 0,55	m: 100%, σ: 0	93,33%
5	m: 85%, σ: 1,34	m: 55%, σ: 1,1	m: 65%, σ: 0,89	68,33%
6	m: 85%, σ: 0,89	m: 80%, σ: 0,45	m: 90%, σ: 0,89	85%
7	m: 70%, σ: 1,3	m: 60%, σ: 1,14	m: 75%, σ: 1	68,33%
8	m: 87,5%, σ: 1	m: 62,5%, σ: 1,29	m: 62,5%, σ: 1	70,83%
9	m: 85%, σ: 0,55	m: 80%, σ: 0,45	m: 85%, σ: 0,55	83,33%
10	m: 95%, σ: 0,45	m: 80%, σ: 0,45	m: 90%, σ: 0,89	88,33%
11	m: 60%, σ: 1,52	m: 90%, σ: 0,55	m: 75%, σ: 1,22	75%
12	m: 75%, σ: 1,22	m: 70%, σ: 0,45	m: 90%, σ: 0,55	78,33%
13	m: 87,5%, σ: 1,22	m: 58,33%, σ: 0,82	m: 62,5%, σ: 0,55	69,44%
14	m: 90%, σ: 0,55	m: 75%, σ: 0	m: 90%, σ: 0,55	85%
16	m: 75%, σ: 1,22	m: 45%, σ: 0,45	m: 60%, σ: 1,34	60%
17	m: 50%, σ: 1,15	m: 68,75%, σ: 0,96	m: 75%, σ: 0,82	64,58%

Tabella 3.8: Analisi del mutuo supporto nei team

La prima domanda ha ottenuto un punteggio medio di 79,38%, tuttavia è possibile notare in quasi tutti i team un alto valore della deviazione standard.
Questo potrebbe essere indice del fatto che non tutti i membri dei team si siano sentiti adeguatamente aiutati in caso di necessità.

La domanda su obiettivi e questioni importanti ha ottenuto un punteggio medio di 69,97%. Si tratta di un punteggio piuttosto basso, dovuto ad alcune criticità riscontrate in certi team, quali il 2 e il 16.

Infine l'ultima domanda, che riassume in generale l'andamento della cooperazione nel team, ha ottenuto un punteggio medio di 77,50%, evidenziando criticità negli stessi team che hanno assegnato un punteggio basso anche alla domanda precedente.

Per concludere, il punteggio medio di cooperazione è stato 75,62%.

3.2.5 Coesione

Per questa categoria sono state considerate le risposte alle domande 22 e 23 del questionario finale:

1. Secondo te il tuo lavoro è stato più o meno importante rispetto a quello del resto del team?
2. Secondo te, tutti i membri si sono sempre impegnati per continuare a farsi sentire del team e tenere unito il gruppo?

Team	Importanza proprio lavoro	Impegno per unità gruppo	Media
1	m: 50%, σ: 0,71	m: 90%, σ: 0,55	70%
2	m: 50%, σ: 1,22	m: 45%, σ: 0,45	47,5%
3	m: 50%, σ: 0,71	m: 65%, σ: 0,55	57,5%
4	m: 60%, σ: 0,55	m: 95%, σ: 0,45	77,5%
5	m: 50%, σ: 0,71	m: 75%, σ: 1	62,5%
6	m: 60%, σ: 0,55	m: 95%, σ: 0,45	77,5%
7	m: 50%, σ: 0	m: 65%, σ: 0,55	57,5%
8	m: 68,75%, σ: 0,96	m: 56,25%, σ: 0,5	62,5%
9	m: 55%, σ: 0,45	m: 90%, σ: 0,55	72,5%
10	m: 45%, σ: 0,84	m: 90%, σ: 0,55	67,5%
11	m: 60%, σ: 1,14	m: 70%, σ: 1,1	65%
12	m: 55%, σ: 0,45	m: 90%, σ: 0,55	72,5%
13	m: 58,33%, σ: 1,03	m: 70,83%, σ: 0,75	64,58%
14	m: 70%, σ: 0,84	m: 85%, σ: 0,55	77,5%
16	m: 60%, σ: 0,55	m: 65%, σ: 1,14	62,5%
17	m: 62,5%, σ: 0,58	m: 68,75%, σ: 0,96	65,63%

Tabella 3.9: Analisi della coesione nei team

È interessante osservare come alla prima domanda la risposta media sia del 56,54%. Ciò indica infatti che mediamente uno studente ritiene che il suo lavoro sia ugualmente importante rispetto a quello degli altri.

Per quanto riguarda la seconda domanda, il risultato medio è di 75,99%: si tratta di un punteggio medio-alto, nonostante in alcuni team, come il numero 5 e il numero 11, il valore della deviazione standard sia piuttosto alto, indice del fatto che non tutti hanno contribuito allo stesso modo per mantenere unito il gruppo.

In conclusione, per questa categoria il punteggio medio è 66,26%.

3.2.6 Bilanciamento dello sforzo

Per questa categoria sono state considerate le risposte alle domande 9, 21 e 24 del questionario finale:

1. La suddivisione dei task seguiva le tue richieste sulla base della tua competenza in quel campo?
2. È stato un argomento discusso lo sforzo di ogni membro del team perché si riteneva che qualcuno non desse il suo contributo?
3. Indicare a seguito una propria stima del contributo percentuale al progetto fornito da ogni componente del team in forma anonima, incluso voi stessi.

Team	Suddivisione task	Discussioni contributo	Contributo singoli	Media
1	m: 85%, σ: 0,55	m: 80%, σ: 0,84	93,24%	86,08%
2	m: 70%, σ: 0,84	m: 50%, σ: 1,22	86,43%	68,81%
3	m: 80%, σ: 0,45	m: 85%, σ: 0,55	94,66%	86,55%
4	m: 85%, σ: 0,55	m: 95%, σ: 0,45	95,43%	91,81%
5	m: 80%, σ: 0,45	m: 100%, σ: 0	92,51%	90,84%
6	m: 90%, σ: 0,55	m: 85%, σ: 0,89	92,9%	89,3%
7	m: 55%, σ: 1,3	m: 100%, σ: 0	94,62%	83,21%
8	m: 87,5%, σ: 0,58	m: 100%, σ: 0	85,07%	90,86%
9	m: 80%, σ: 1,3	m: 95%, σ: 0,45	93,55%	89,52%
10	m: 95%, σ: 0,45	m: 90%, σ: 0,55	94,05%	93,02%
11	m: 65%, σ: 0,89	m: 90%, σ: 0,55	87,22%	80,74%
12	m: 80%, σ: 0,84	m: 95%, σ: 0,45	94,3%	89,77%
13	m: 75%, σ: 0,89	m: 54,17%, σ: 1,17	85,76%	71,64%
14	m: 70%, σ: 0,45	m: 90%, σ: 0,55	92,84%	84,28%
16	m: 65%, σ: 1,14	m: 70%, σ: 0,84	92,56%	75,85%
17	m: 75%, σ: 0,82	m: 93,75%, σ: 0,5	95,27%	88,01%

Tabella 3.10: Analisi del bilanciamento dello sforzo dei team

Il punteggio medio assegnato alla prima domanda è di 77,34% e, salvo poche eccezioni, il valore di σ è abbastanza contenuto.

La seconda domanda ha ricevuto un punteggio medio di 85,81%.
Si notano subito i bassi valori attribuiti dal team 2 e dal team 13, dovuti al fatto che all'interno di questi team il lavoro non sia stato distribuito in modo uniforme tra i componenti.

Il punteggio dell'ultima domanda è ottenuto sottraendo a 100 la media della deviazione standard della stima del contributo di ogni membro.
Infatti se σ fosse 0 significherebbe che per un certo studente tutti i membri del team hanno dato uguale contributo in percentuale.
Se la media delle deviazioni standard di tutti i membri fosse quindi 0, ai fini di questa analisi, rappresenterebbe il punteggio massimo ottenibile.
Questa domanda ha ottenuto un punteggio medio di 91,90%.
I picchi più bassi si possono trovare nel team 2 e nel team 13.

In conclusione il punteggio medio inerente il bilanciamento dello sforzo è 85,02%.

3.3 Analisi della soddisfazione del team

In questa sezione sarà effettuata l'analisi della soddisfazione dei team tramite le domande 25, 26 e 27 del questionario finale:

1. Con questo progetto hai imparato una nuova metodologia e mezzi di lavoro per te utili?
2. Che autovalutazione daresti complessivamente al prodotto del tuo team?
3. Che autovalutazione daresti complessivamente al processo del tuo team?

Team	**Imparato qualcosa di nuovo**	**Autoval. prodotto**	**Autoval. processo**
1	m: 85%, σ: 0,89	m: 86%, σ: 0,55	m: 86%, σ: 1,52
2	m: 85%, σ: 0,89	m: 70%, σ: 0	m: 64%, σ: 1,82
3	m: 85%, σ: 0,55	m: 80%, σ: 0	m: 80%, σ: 1
4	m: 90%, σ: 0,55	m: 88%, σ: 0,84	m: 82%, σ: 1,1
5	m: 80%, σ: 0,84	m: 80%, σ: 0,71	m: 68%, σ: 1,64
6	m: 100%, σ: 0	m: 86%, σ: 0,89	m: 92%, σ: 0,84
7	m: 80%, σ: 0,84	m: 76%, σ: 0,89	m: 68%, σ: 0,84
8	m: 87,5%, σ: 0,58	m: 75%, σ: 1,29	m: 77,5%, σ: 0,5
9	m: 90%, σ: 0,55	m: 90%, σ: 0	m: 82%, σ: 0,84
10	m: 90%, σ: 0,89	m: 84%, σ: 0,55	m: 92%, σ: 0,84
11	m: 90%, σ: 0,55	m: 92%, σ: 1,1	m: 80% σ: 0,71
12	m: 95%, σ: 0,45	m: 74%, σ: 0,55	m: 78%, σ: 0,45
13	m: 95,83%, σ: 0,41	m: 86,7%, σ: 1,03	m: 85%, σ: 0,55
14	m: 90%, σ: 0,55	m: 80%, σ: 1	m: 82%, σ: 1,3
16	m: 65%, σ: 0,55	m: 70%, σ: 1,22	m: 66%, σ: 1,67
17	m: 93,75%, σ: 0,5	m: 82,5%, σ: 0,96	m: 82,5%, σ: 1,5

Tabella 3.11: Analisi della soddisfazione dei team

Dalle risposte alla prima domanda è possibile notare che la maggior parte degli studenti ritiene di aver acquisito nuove conoscenze e imparato ad utilizzare nuovi strumenti di sviluppo. Infatti il punteggio medio è 87,63%.

Per quanto riguarda l'autovalutazione del prodotto, il punteggio medio è 81,26%, indicando una soddisfazione generale per il prodotto creato. Fanno eccezione i team 2 e 16 che hanno dato una valutazione molto più bassa, seppur sufficiente.

Infine, il punteggio medio attribuito all'autovalutazione del processo seguito si attesta al 79%.

In quasi tutti i casi i punteggi sono simili a quelli della domanda precedente, ad eccezione dei team 5 e 7 che hanno dato una valutazione molto più bassa.

In conclusione la soddisfazione media dei team è pari all'82,65%.

3.4 Analisi complessiva qualità del teamwork

La tabella che segue riassume i risultati ottenuti nel corso dell'analisi della qualità del teamwork, divisi nelle categorie definite nel modello descritto da Martin Hoegl e Hans Georg Gemuenden [HG01] e successivamente applicato ai team di sviluppo Agile da Dingsør et al. [LSD+16].

Team	Qualità prodotto	Interazioni	Soddisfazione	Qualità Teamwork
1	95,83%	77,79%	85,67%	86,43%
2	70,83%	63,13%	73,00%	68,99%
3	98,33%	74,79%	81,67%	84,93%
4	100,00%	85,86%	86,67%	90,84%
5	76,67%	78,74%	76,00%	77,13%
6	98,33%	86,20%	92,67%	92,40%
7	94,17%	71,47%	74,67%	80,10%
8	92,50%	73,99%	80,00%	82,16%
9	100,00%	80,56%	87,33%	89,30%
10	100,00%	80,65%	88,67%	89,77%
11	85,00%	72,88%	87,33%	81,74%
12	78,33%	80,61%	82,33%	80,42%
13	96,67%	71,96%	89,18%	85,94%
14	97,50%	80,39%	84,00%	87,30%
16	62,50%	67,78%	67,00%	65,76%
17	95,00%	73,03%	86,25%	84,76%
Qualità media	90,10%	76,24%	82,65%	83,00%

Tabella 3.12: Analisi complessiva della qualità del teamwork

Cominciando dalla qualità del prodotto finale, si può notare come quasi tutti i team abbiano ottenuto punteggi medio-alti, nella maggior parte dei casi superiori all'80%, con una media che si attesta al 90,1%.

Proseguendo con l'analisi delle interazioni si riscontra un calo nelle valutazioni che restano però quasi tutte superiori al 70%, con un valore medio del 76,24%.

Per quanto riguarda la soddisfazione dei membri dei team, si può notare generalmente un valore più basso in corrispondenza dei team con un punteggio più basso nell'analisi delle interazioni.
Il punteggio medio di soddisfazione è dell'82,65%.

Quello che segue ora è invece un grafico riassuntivo dei risultati medi raggiunti per ognuno dei parametri del modello.

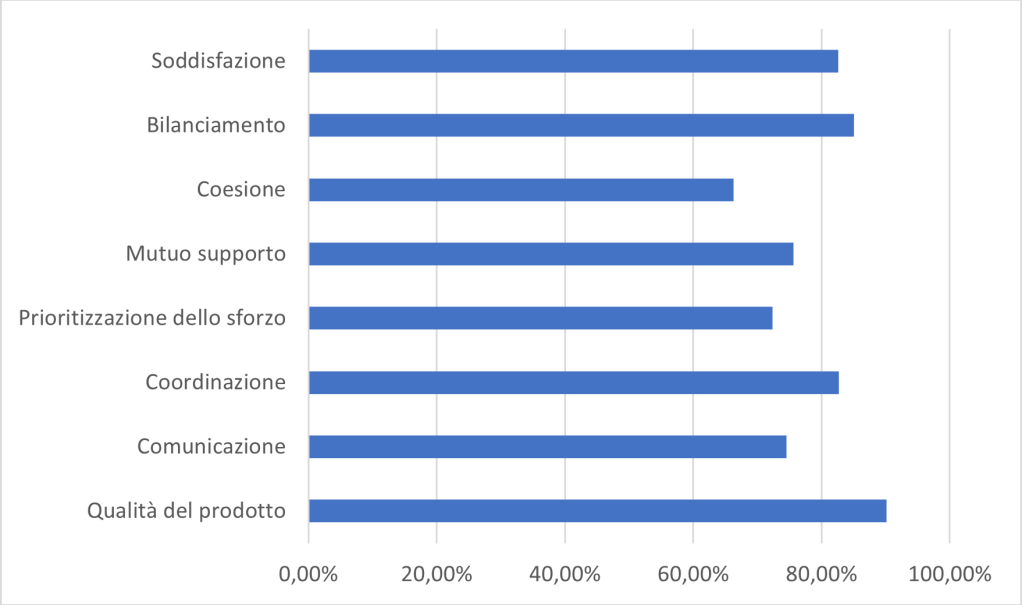

Figura 3.1: Risultati medi raggiunti per parametro del modello di qualità

Infine, uno sguardo al punteggio totale dei singoli team: solo i team 2, 5 e 16 hanno ottenuto una valutazione inferiore all'80%, mentre tutti gli altri team hanno valutazioni superiori. I team 4 e 6 hanno persino superato il 90%.

In conclusione dai dati appena presentati si può notare come il livello medio di qualità del teamwork per il progetto del corso di Ingegneria del Software dell'Anno Accademico 2021/2022 sia 83%.

Capitolo 4

MODELLO DI MATURITÀ DEL TEAMWORK

In questo capitolo sarà presentato il modello di maturità del teamwork proposto da Alexandre Yin[YFdS11] basato su quello di Patel Chetankumar[CR09].
L'applicazione di questo modello si basa sull'assegnazione di un punteggio percentuale per ognuno dei cinque livelli previsti.
Di seguito sono presentati e descritti tali livelli.

- Livello 1: rappresenta il livello di maturità minimo e indica l'incapacità di adempiere alla gestione basilare di Scrum.
- Livello 2: indica il raggiungimento di una gestione basilare di Scrum e dei requisiti minimi del progetto.
- Livello 3: aggiunge alle abilità del livello precedente la capacità di gestire le relazioni con i clienti e di gestire il progetto nel corso degli sprint.
- Livello 4: oltre agli obiettivi del livello 3 prevede la presenza di sistemi di versioning e di monitoraggio delle performance durante lo sviluppo.
- Livello 5: rappresenta il livello massimo raggiungibile e prevede la presenza di continuo miglioramento da parte del team tramite riunioni e retrospettive.

Ad ogni livello sono associate una o più obiettivi e per ognuno di essi è prevista l'assegnazione di un punteggio.
La media dei punteggi di un livello è chiamata "Key Process Area" (KPA).

Come spiegato da Sofia Zani[Zan20], il calcolo del KPA si basa su una serie di domande a cui rispondere con "Si", "No", "Parzialmente" o "Non applicabile": il punteggio finale è calcolato secondo la formula

$$\frac{\sum N_S + \frac{1}{2}\sum N_P}{t - \sum N_{NA}} \times 100$$

dove N_S rappresenta il numero di risposte "Si", N_P il numero di risposte "Parzialmente", t il numero totale di domande e N_{NA} il numero di risposte "Non applicabile"

Se il punteggio KPA è compreso tra 86% e 100% il livello di maturità è considerato completamente superato.

Un punteggio KPA compreso tra il 51% e l'85% indica il raggiungimento del livello di maturità, mentre un punteggio inferiore indica un raggiungimento parziale (fino al 16%) o mancato (al di sotto del 15%).
Non è possibile raggiungere un livello di maturità se non si è raggiunto il precedente.

4.1 Analisi maturità del teamwork

Di seguito saranno riportate le domande per ciascun livello e le relative risposte con calcolo del KPA e indicazione sul raggiungimento del livello in questione.
Si considera raggiunto un livello soltanto se $KPA \geq 86\%$.

Il raggiungimento del livello sarà contrassegnato dalla lettera "R" nell'apposita colonna di ogni tabella, viceversa "NR" indicherà il mancato raggiungimento.

La risposta a ciascuna domanda è basata sui dati ottenuti dai report finali, dai questionari finali e dagli strumenti CAS.

Livello 2

Le domande per stabilire il raggiungimento del livello 2 sono le seguenti, suddivise per obiettivo:

- Gestione basilare di Scrum
    1. Il ruolo di Scrum Master è stato adempito?
    2. Il ruolo del team di sviluppo è stato adempito?
    3. Gli artefatti Scrum (product backlog, sprint backlog, incremento) sono stati rilasciati?

4. Gli eventi Scrum sono stati svolti e c'era partecipazione ad essi?

- Gestione dei requisiti di progettazione

5. Il product backlog è stato raffinato durante gli sprint?
6. Si è praticato con successo l'evento di sprint planning?

Team	**Q1**	**Q2**	**Q3**	**Q4**	**KPA**	**Q5**	**Q6**	**KPA**	**Liv. 2**
1	P	S	S	S	87,50%	S	S	100,00%	R
2	S	S	S	S	100,00%	S	S	100,00%	R
3	S	S	S	S	100,00%	S	S	100,00%	R
4	S	S	S	S	100,00%	P	S	75,00%	R
5	S	S	S	S	100,00%	S	S	100,00%	R
6	S	S	S	S	100,00%	S	S	100,00%	R
7	S	S	S	S	100,00%	S	S	100,00%	R
8	S	S	S	P	87,50%	S	S	100,00%	R
9	P	S	S	S	87,50%	S	S	100,00%	R
10	S	S	S	S	100,00%	S	S	100,00%	R
11	S	S	S	P	87,50%	S	S	100,00%	R
12	S	S	S	S	100,00%	S	P	75,00%	R
13	P	S	S	P	75,00%	S	S	100,00%	R
14	S	S	S	S	100,00%	S	S	100,00%	R
16	S	S	S	S	100,00%	S	S	100,00%	R
17	S	S	S	S	100,00%	S	S	100,00%	R

Tabella 4.1: Raggiungimento del livello 2

Come si può notare tutti i team hanno raggiunto il secondo livello di maturità, nonostante non tutti abbiano ottenuto punteggio pieno.

Le principali criticità segnalate riguardano il ruolo dello Scrum Master e la partecipazione agli eventi Scrum, specialmente all'interno del team 13 che ha avuto entrambi i problemi.

Tutti i team hanno ritenuto adeguato il contributo degli sviluppatori e hanno rilasciato tutti gli artefatti Scrum necessari.

Livello 3

Le domande per stabilire il raggiungimento del livello 3 sono le seguenti:

- Gestione delle relazioni col cliente
  7. La Definition of Done è stata realizzata?
  8. C'è stato almeno un incontro prima della consegna tra PO e team?
  9. Si è praticato con successo l'evento di sprint review?
- Gestione del prodotto durante lo sprint
  10. Lo sprint backlog è presente ad ogni sprint ed è gestito sul kanban di Taiga?
  11. Vi sono state consegne di versioni incrementali del prodotto alla fine di ogni sprint?
  12. Il team ha progressivamente curato la qualità del prodotto ad ogni sprint?
  13. Il team ha utilizzato pair o mob programming durante la maggior parte degli sprint?

Team	**Q7**	**Q8**	**Q9**	**KPA**	**Q10**	**Q11**	**Q12**	**Q13**	**KPA**	**Liv. 3**
1	S	S	S	100,00%	S	S	S	P	87,50%	R
2	S	S	S	100,00%	S	P	P	P	62,50%	NR
3	S	S	S	100,00%	S	S	S	S	100,00%	R
4	S	S	S	100,00%	S	S	S	P	87,50%	R
5	S	S	S	100,00%	S	P	S	P	75,00%	R
6	S	S	S	100,00%	S	P	P	S	75,00%	R
7	S	S	P	83,33%	S	P	P	S	75,00%	NR
8	P	S	S	83,33%	S	P	P	S	75,00%	NR
9	S	S	S	100,00%	S	P	S	S	87,50%	R
10	S	S	S	100,00%	S	S	S	S	100,00%	R
11	P	S	S	83,33%	S	P	P	P	62,50%	NR
12	P	S	S	83,33%	S	P	P	P	62,50%	NR
13	S	S	S	100,00%	S	P	P	P	62,50%	NR
14	S	S	P	83,33%	S	S	P	S	87,50%	R
16	S	S	S	100,00%	S	P	P	S	75,00%	R
17	S	S	P	83,33%	S	P	S	P	75,00%	NR

Tabella 4.2: Raggiungimento del livello 3

Per quanto riguarda queste domande si segnala che la Q8 e la Q10 rappresentavano requisiti obbligatori durante il progetto che sono stati rispettati da tutti i team.

La domanda con il minor numero di risposte "Si" è la Q11, questo perché le consegne di versioni incrementali, ovvero le riunioni con i PO in cui si mostrava il prodotto, sono terminate al termine del terzo sprint, quindi i team che hanno utilizzato più sprint non possono aver effettuato ulteriori consegne di versioni incrementali.

Anche la domanda 13 ha ricevuto poche risposte positive dato che molti team hanno usato il pair o il mob programming solo in uno sprint, di solito uno dei primi.

In conclusione, guardando i risultati presentati nella tabella precedente, è possibile notare che ben sette team non hanno raggiunto il terzo livello di maturità. I risultati di questi team sono stati calcolati anche per i livelli successivi ma non saranno più presi in considerazione nel corso dell'analisi.

Livello 4

Le domande per stabilire il raggiungimento del livello 4 sono le seguenti:

- Gestione standardizzata del software
  14. Si è utilizzato Gitlab per gestire il codice del programma nel team?
- Gestione delle performance durante il processo
  15. Si è utilizzato un logger per il controllo della produttività del team?
  16. I grafici di burn down sono stati aggiornati ad ogni sprint?

Team	**Q14**	**KPA**	**Q15**	**Q16**	**KPA**	**Liv. 4**
1	S	100,00%	S	P	75,00%	R
~~2~~	S	100,00%	N	P	25,00%	NR
3	S	100,00%	S	S	100,00%	R
4	S	100,00%	P	P	50,00%	NR
5	S	100,00%	S	P	75,00%	R
6	S	100,00%	P	S	75,00%	R
~~7~~	S	100,00%	S	P	75,00%	NR
~~8~~	S	100,00%	P	P	50,00%	NR
9	S	100,00%	S	P	75,00%	R
10	S	100,00%	S	S	100,00%	R
~~11~~	S	100,00%	P	P	50,00%	NR
~~12~~	S	100,00%	P	S	75,00%	NR
~~13~~	S	100,00%	P	S	75,00%	NR
14	S	100,00%	P	P	50,00%	NR
16	S	100,00%	P	P	50,00%	NR
~~17~~	S	100,00%	N	P	25,00%	NR

Tabella 4.3: Raggiungimento del livello 4

Anche in questo caso la Q14 sull'utilizzo di Gitlab rappresentava un requisito obbligatorio per il progetto ed ha quindi ricevuto solo risposte positive.

La domanda successiva è stata valutata con "Si" se tutti i membri del team hanno indicato di aver utilizzato il logger. La valutazione è stata invece "Parzialmente" se anche un solo componente non ha utilizzato il logger.

La Q16 invece ha ricevuto risposta "Si" soltanto se tutti i grafici di burndown

risultavano aggiornati con regolarità nel corso di ogni sprint. La valutazione "Parzialmente" è stata data se solo alcuni grafici risultavano aggiornati con costanza.

A seguito dell'analisi per il raggiungimento del livello 4, altri tre team non hanno raggiunto gli obiettivi minimi. Come prima, questi non saranno più considerati nella discussione dei dati successivi.

Livello 5

Le domande per stabilire il raggiungimento del livello 5 sono le seguenti:

- Gestione del miglioramento delle performance

  17. Si è praticato con successo l'evento di daily Scrum o un weekly Scrum con regolarità?
  18. Si è praticato con successo l'evento di sprint retrospective?
  19. Se ci sono stati dei problemi si sono riuscite ad individuare le cause e ad applicare miglioramenti?

Team	**Q17**	**Q18**	**Q19**	**KPA**	**Liv. 5**
1	S	S	P	83,33%	NR
~~2~~	P	S	S	83,33%	NR
3	S	S	S	100,00%	R
~~4~~	P	S	S	83,33%	NR
5	P	S	P	66,67%	NR
6	S	S	S	100,00%	R
~~7~~	S	S	P	83,33%	NR
~~8~~	S	S	P	83,33%	NR
9	S	S	S	100,00%	R
10	S	S	S	100,00%	R
~~11~~	S	S	P	83,33%	NR
~~12~~	S	S	P	83,33%	NR
~~13~~	S	S	P	83,33%	NR
~~14~~	S	S	S	100,00%	NR
~~16~~	S	S	P	83,33%	NR
~~17~~	S	S	S	100,00%	NR

Tabella 4.4: Raggiungimento del livello 5

Anche lo svolgimento delle retrospettive (Q18) rappresentava un requisito obbligatorio quindi la relativa domanda ha ricevuto solo risposte "Si".

In questo caso la valutazione "P" assegnata al team 5 alla domanda 17 è dovuta al fatto che in nessuna retrospettiva l'evento di weekly scrum era posizionato nella zona indicante gli elementi positivi dello sprint, ma sempre tra quella negativa e quella intermedia.

Infine, per quanto riguarda l'ultima domanda, in generale i team sono riusciti ad applicare i miglioramenti ai problemi identificati durante lo sviluppo. Le valutazioni "Parzialmente" sono state date ai team che hanno inserito la carta Essence "improvement" in una zona della retrospettiva che non fosse quella indicante la massima riuscita o ai team che hanno descritto esplicitamente le difficoltà nell'applicare i miglioramenti.

In conclusione sono solo quattro i team che hanno dimostrato il massimo grado di maturità: i team 3, 6, 9 e 10.

4.1.1 Analisi complessiva maturità del teamwork

Team	Livello raggiunto	Media KPA
1	4	90,48%
2	2	81,55%
3	5	100,00%
4	3	85,12%
5	4	88,10%
6	5	92,86%
7	2	88,10%
8	2	82,74%
9	5	92,86%
10	5	100,00%
11	2	80,95%
12	2	82,74%
13	2	85,12%
14	3	88,69%
16	3	86,90%
17	2	83,33%

Tabella 4.5: Livello raggiunto dai team e relativa media KPA

Questa tabella riassume i risultati dell'applicazione del modello di maturità del teamwork: il livello medio raggiunto è il terzo (gestione basilare di Scrum, gestione dei requisiti del progetto e gestione delle interazioni con il cliente), mentre il KPA medio è dell'88,1%.

Alcuni team hanno raggiunto un livello inferiore ai propri risultati complessivi. Infatti, non essendo possibile raggiungere il livello successivo senza aver superato il precedente, qualche team non ha ottenuto un punteggio KPA sufficiente per un determinato livello nonostante nel successivo abbia ottenuto un risultato più che accettabile.

Per questo motivo i risultati sono stati presentati in forma completa nel corso della discussione anche se sono stati considerati per il commento e l'analisi solo i team che hanno effettivamente raggiunto il livello in questione.

Capitolo 5

CONFRONTO CON ANALISI PRECEDENTE

In questo capitolo sarà presentato un confronto con l'analisi effettuata lo scorso anno da Sofia Zani[Zan20].

Prima di presentare i dati è doverosa una premessa: per il progetto del 2020/2021 sono stati presentati due ulteriori questionari oltre al questionario finale individuale che hanno permesso di effettuare un'ulteriore analisi di efficienza dei team basata sul numero di task e User story previste e quelle effettivamente completate.

Per l'anno 2021/2022 questi questionari non sono stati somministrati e, anche utilizzando Taiga, non è più possibile ottenere questi dati garantendone la correttezza, quindi l'analisi di efficienza è stata omessa.

Oltre a ciò bisogna ricordare che il corso è passato da 6 a 9 CFU, con un conseguente carico di lavoro teoricamente maggiore per l'anno 2021/2022.

I dati saranno di seguito presentati sotto forma di tabella: la prima colonna indicherà il parametro preso in considerazione, la seconda il dato per l'Anno Accademico 2021/2022 e la terza il dato per l'Anno Accademico 2020/2021.

	2021/2022	**2020/2021**
Numero di team	16	11
Studenti coinvolti	79	57
Analisi delle performance		
Errore medio stima LoC	-882	32016 (619)
Media LoC sviluppatori	1688	6993 (721)
Media ore sviluppatori	83	59 (55)
Media LoC/h	20	119 (13)
Media LoC SM	1484	806
Media ore supporto SM	32	52
Media ore programmazione SM	91	42
Media ore riunioni	50	45
Media ore totali di lavoro	416	540
Qualità media prodotto	90,1%	92,0%
Analisi delle interazioni		
Qualità media comunicazione	74,51%	66,32%
Qualità media coordinazione	83,69%	85,61%
Qualità media mutuo supporto	75,62%	82,60%
Qualità media prioritizzazione sforzo	79%	71,80%
Qualità media coesione	66%	73,20%
Qualità media bilanciamento sforzo	85,02%	80,20%
Analisi della soddisfazione		
Media soddisfazione	82,65%	82,20%
Qualità media	83%	85,16%
Analisi della maturità del teamwork		
Livello medio maturità	3,18	2,36
Media KPA	88,10%	76,67%

Tabella 5.1: Confronto tra i dati 2021/2022 e quelli 2020/2021

Cominciando dalle prime due righe è subito possibile notare che per l'anno 2021/2022 il campione, sia come numero di team che come numero di studenti era più esteso.

Per quanto riguarda l'analisi delle performance, nelle prime quattro righe per l'anno 2020/2021 sono stati riportati due valori, quello reale e, tra parentesi, il valore ottenuto escludendo dal campione il team "B" che a questi quesiti ha risposto con valori esageratamente grandi con valori di deviazione standard troppo elevati.
Dai risultati si ricava che gli sviluppatori del 2021/2022 hanno mediamente scritto più linee di codice e con un ritmo maggiore.

Il ruolo di sviluppatore dello Scrum Master è stato mantenuto: in entrambi i casi la media delle che gli SM dichiarano di aver scritto si avvicina molto alla media delle linee dichiarate dagli sviluppatori.

Riguardo le ore di riunione i valori sono simili. Ciò che varia sono le ore di lavoro totali: nel caso del corso da 6 CFU le ore dichiarate sono il 29,8% in più rispetto al corso da 9 CFU.
Questo risultato potrebbe nuovamente essere dovuto alla difficoltà che gli studenti hanno nel quantificare la propria quantità di lavoro.

Passando all'analisi della qualità del teamwork, è possibile notare differenze minime solo per quanto riguarda la qualità del prodotto, la qualità della coordinazione e la soddisfazione media.
Tutti gli altri parametri hanno subito un incremento (comunicazione, prioritizzazione e bilanciamento dello sforzo) o un decremento (mutuo supporto e coesione).

Interessante notare come i criteri migliorati siano quelli strettamente legati allo svolgimento del progetto, ovvero organizzazione dello sforzo e comunicazioni inerenti il lavoro, mentre quelli che hanno subito un peggioramento sono più strettamente legati ai rapporti sociali tra gli sviluppatori.

Anche la qualità media complessiva, calcolata a partire dai valori medi ottenuti da ogni team (come in tabella 3.12) non ha subito grandi cambiamenti.

Passando al modello di maturità si ha che nell'anno 2021/2022 il risultato è migliore.
Tuttavia, come indicato nel capitolo precedente, alcuni requisiti sono passati da facoltativi a obbligatori, aumentando così KPA e livello medio raggiunto.

Capitolo 6

ALTRE ANALISI

In questo capitolo saranno presentate tre ulteriori analisi:

- L'analisi di un secondo modello di qualità del teamwork
- L'analisi dei contributi personali dei singoli studenti secondo le risposte alla domanda 9 del questionario finale
- L'analisi delle retrospettive personali degli studenti, basate sulle domande 7 e 8 del questionario finale

6.1 Confronto con altro modello di qualità

Durante lo svolgimento della presente analisi ho analizzato anche la teoria sviluppata da Christiaan Verwijs e Daniel Russo[VR21] riguardo la qualità e l'efficacia dei team agili.
Il modello presentato si basa sull'osservazione di sei elementi:

- Efficienza del team ("Team Effectiveness")
- Reattività ("Responsiveness")
- Interessamento degli Stakeholder ("Stakeholder Concern")
- Autonomia del team ("Team Autonomy")
- Miglioramento continuo ("Continuous Improvement")
- Supporto da parte del management ("Management Support")

A sua volta, ognuno dei sei elementi viene misurato secondo alcuni parametri, indicati nell'immagine seguente, tratta dall'articolo di presentazione della teoria.

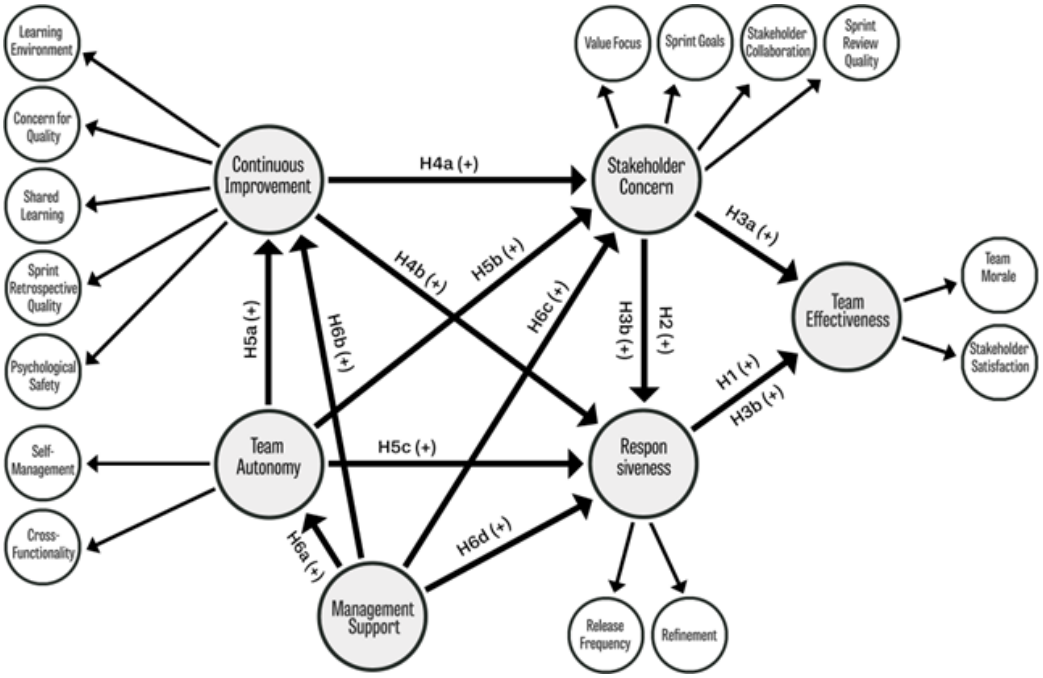

Figura 6.1: Immagine riassuntiva del modello di Verwijs e Russo

La conclusione a cui giungono gli autori è che i team più efficaci sono quelli che riescono a coniugare rilasci frequenti del prodotto durante lo sviluppo con l'abilità di adattarsi alle richieste degli stakeholder.
Per giungere a questo risultato ai team serve un buon grado di autonomia, miglioramento continuo e il supporto da parte del management.

6.1.1 Applicabilità al progetto e applicazione limitata

Questa teoria è sicuramente completa e applicabile in generale alla maggior parte degli sviluppi agili, tuttavia nel caso specifico dell'analisi presentata in questa dissertazione ci sono diversi limiti legati al regolamento dato agli studenti per completare il progetto.

In seguito sarà comunque effettuato un tentativo di applicazione parziale e limitata: per ognuno dei sei elementi sopra indicati saranno selezionate le sole caratteristiche misurabili con i dati a disposizione.

Efficienza del team ("Team Effectiveness") e supporto da parte del management ("Management Support") non sono stati analizzati in quanto assenti sia dati sulla soddisfazione dei PO e sul morale dei team, sia i manager.

Responsiveness

Per analizzare la frequenza dei rilasci è stato considerato il numero di sprint di cui ciascun team ha avuto bisogno per consegnare il progetto. Questo perché fino al terzo sprint le riunioni con i PO sono avvenute in date fissate ed erano obbligatorie, mentre successivamente non sono più avvenute.

Per mettere in relazione tempo di sviluppo e rilasci di software è stato dunque assegnato un punteggio del 100% ai team che hanno consegnato dopo tre sprint, con una diminuzione del 25% per ogni sprint aggiuntivo fino ad un minimo del 25%.

Per misurare il "refinement" dei requisiti, sono state considerate le valutazioni date alla domanda Q5 del modello di maturità "Il product backlog è stato raffinato durante gli sprint?", assegnando un puntaggio del 100% in caso di risposta "Si" e 0% altrimenti.

Team	Release Frequency	Refinement
1	100,00%	100,00%
2	75,00%	100,00%
3	100,00%	100,00%
4	100,00%	0,00%
5	75,00%	100,00%
6	25,00%	100,00%
7	75,00%	100,00%
8	50,00%	100,00%
9	75,00%	100,00%
10	100,00%	100,00%
11	75,00%	100,00%
12	75,00%	100,00%
13	75,00%	100,00%
14	100,00%	100,00%
16	50,00%	100,00%
17	75,00%	100,00%

Tabella 6.1: Analisi limitata - responsiveness

Stakeholder concern

In questo caso, per il "Value focus", ovvero quanto i membri del team concentrassero i loro sforzi per fornire ai PO del valore, è stato assegnato un punteggio di 100% a tutti i team in quanto si suppone che in un contesto di studio universitario tutti si sforzino di dare il meglio per ottenere una valutazione più alta possibile,ottenibile consegnando il maggior "value" possibile.

Gli altri due parametri sono stati misurati dalle retrospettive, assegnando il 100% nel caso in cui la carta fosse presente in posizione positiva, 50% in caso di carta assente o in posizione neutra e 0% altrimenti.

Team	Value focus	Stakeholder collaboration	Sprint Review Quality
1	100,00%	50,00%	100,00%
2	100,00%	100,00%	50,00%
3	100,00%	50,00%	50,00%
4	100,00%	50,00%	50,00%
5	100,00%	100,00%	50,00%
6	100,00%	100,00%	100,00%
7	100,00%	50,00%	50,00%
8	100,00%	100,00%	100,00%
9	100,00%	100,00%	50,00%
10	100,00%	100,00%	50,00%
11	100,00%	100,00%	50,00%
12	100,00%	100,00%	50,00%
13	100,00%	100,00%	100,00%
14	100,00%	100,00%	100,00%
16	100,00%	50,00%	50,00%
17	100,00%	100,00%	50,00%

Tabella 6.2: Analisi limitata - stakeholder concern

Team autonomy

Per questa caratteristica tutti i parametri sono stati misurati consultando le retrospettive e assegnando il 100% nel caso in cui la carta fosse presente in posizione positiva, 75% in caso di posizionamento tra positivo e neutro, 50% in caso di carta assente o in posizione neutra e 0% altrimenti.

Team	Self-Management	Cross-functionality
1	50,00%	50,00%
2	50,00%	50,00%
3	50,00%	100,00%
4	50,00%	50,00%
5	0,00%	100,00%
6	50,00%	100,00%
7	100,00%	50,00%
8	100,00%	75,00%
9	100,00%	75,00%
10	50,00%	100,00%
11	50,00%	100,00%
12	100,00%	100,00%
13	100,00%	100,00%
14	50,00%	50,00%
16	50,00%	100,00%
17	50,00%	50,00%

Tabella 6.3: Analisi limitata - team autonomy

Continuous improvement

Per misurare la qualità delle retrospettive si è fatto affidamento sulla carta corrispondente presente nelle retrospettive stesse, assegnando il 100% nel caso in cui la carta fosse presente in posizione positiva, 75% in caso di posizione tra il positivo e il neutro, 50% in caso di carta assente o in posizione neutra e 0% altrimenti.

Per misurare il "Quality concern", basato sulla realizzazione di una Definition of Done, sono state utilizzate le valutazioni della domanda Q7 del modello di maturità ("La Definition of Done è stata realizzata?"), assegnando il 100% solo in caso di risposta positiva.

Per quanto riguarda "Shared learning", sono state utilizzate le valutazioni medie della domanda 14 del questionario finale individuale ("Il team ha mai chiesto aiuto all'esterno? Esempio: ad una persona di un altro team?"), opportunamente convertite in percentuale.

Infine, sempre considerando il contesto di studio, l'ultimo parametro è stato valutato 100% per tutti i team.

Team	Retrospective	Quality Concern	Shared Learning	Learning Env.
1	100,00%	100,00%	35,00%	100,00%
2	50,00%	100,00%	30,00%	100,00%
3	50,00%	100,00%	15,00%	100,00%
4	50,00%	100,00%	5,00%	100,00%
5	100,00%	100,00%	15,00%	100,00%
6	100,00%	100,00%	10,00%	100,00%
7	50,00%	100,00%	30,00%	100,00%
8	50,00%	0,00%	18,75%	100,00%
9	100,00%	100,00%	50,00%	100,00%
10	100,00%	100,00%	45,00%	100,00%
11	100,00%	0,00%	35,00%	100,00%
12	50,00%	0,00%	30,00%	100,00%
13	75,00%	100,00%	29,17%	100,00%
14	100,00%	100,00%	20,00%	100,00%
16	50,00%	100,00%	45,00%	100,00%
17	50,00%	100,00%	0,00%	100,00%

Tabella 6.4: Analisi limitata - continuous improvement

Risultati

Vengono di seguito riportati i risultati medi per ogni team e confrontati con quelli ottenuti dal modello di Hoegl - Gemuenden[HG01].

Team	Analisi limitata	Hoegl - Gemuenden
1	80,45%	86,43%
2	73,18%	68,99%
3	74,09%	84,93%
4	59,55%	90,84%
5	76,36%	77,13%
6	80,45%	92,40%
7	73,18%	80,10%
8	72,16%	82,16%
9	86,36%	89,30%
10	85,91%	89,77%
11	73,64%	81,74%
12	73,18%	80,42%
13	89,02%	85,94%
14	83,64%	87,30%
16	72,27%	65,76%
17	70,45%	84,76%
Media	76,49%	83,00%

Tabella 6.5: Risultati dell'analisi limitata, confrontati con il modello Hoegl - Gemuenden

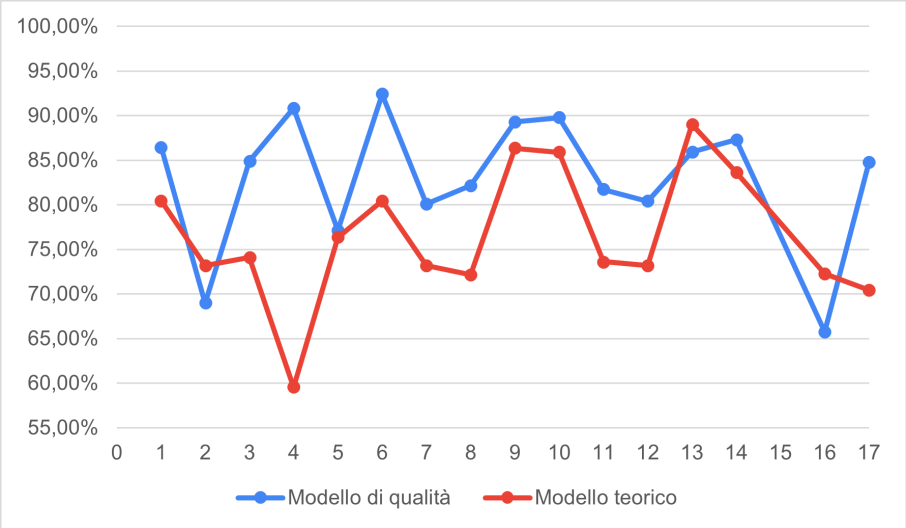

Figura 6.2: Confronto tra il modello di qualità e quello teorico proposto

È possibile notare una somiglianza dei dati nei team che hanno maggiormente curato le retrospettive con le carte Essence, utilizzando più carte e creando così tabelle più complete.
Infatti, per come è stata concepita, l'analisi limitata basa gran parte dei punteggi sulla presenza e sul posizionamento di carte nelle retrospettive, quindi anche solamente utilizzare più carte aumenta la possibilità di ottenere un punteggio maggiore.

In ogni caso, nonostante l'analisi appena effettuata restituisca risultati mediamente più bassi, è possibile in diversi casi notare un andamento similare dei due grafici, indice del fatto che le qualità dei membri dei team e la qualità del teamwork si riflettono in tutti gli aspetti dello sviluppo agile.

Solo i team 2, 13 e 16 hanno ottenuto risultati migliori con l'applicazione limitata del modello teorico rispetto all'altro.

Tuttavia, l'indice di Pearson ρ, indicante la possibilità di una correlazione lineare tra i due set di dati è di 0,29 su un massimo di 1 che significa che una correlazione è poco probabile. Si tratta di un risultato atteso dato che i due modelli si basano per la maggior parte su parametri differenti e valutati in maniera diversa.

6.2 Analisi dei contributi personali

In questa sezione saranno analizzate le risposte alla domanda 9 del questionario finale individuale: "Indicare a seguito una propria stima del contributo percentuale al progetto fornito da ogni componente del team in forma anonima, incluso voi stessi".

A causa della forma anonima è possibile essere certi soltanto del fatto che, per ogni risposta e per ogni team, le stime che ogni componente si attribuisce siano riconducibili a studenti diversi. Infatti l'ordine con cui sono stati valutati i compagni non è noto e non è determinabile.

Per questo motivo, nella seconda colonna della tabella che segue sono riportate le somme dei contributi che ogni studente ha associato a sé stesso.

La terza colonna invece riporta la media delle deviazioni standard di ogni serie di risposte.
Questa serve a capire se la stima del carico di lavoro risulta condivisa tra tutti i membri del team.

Team	Somma contributi personali	Deviazione standard media
1	107,20%	6,76%
2	122,00%	13,57%
3	104,00%	5,34%
4	100,00%	4,57%
5	102,20%	7,49%
6	108,00%	7,10%
7	94,20%	5,38%
8	122,50%	14,93%
9	108,20%	6,45%
10	99,00%	5,95%
11	120,00%	12,78%
12	112,50%	5,70%
13	168,00%	14,24%
14	115,00%	7,16%
16	119,10%	7,44%
17	100,00%	4,73%

Tabella 6.6: Analisi dei contributi personali

Cominciando l'analisi dalla seconda colonna è possibile notare la tendenza ad attribuire maggior valore al proprio lavoro, infatti nei team composti da 5 persone la percentuale media che ogni studente ha assegnato a sé stesso è 21,71%, ovvero quasi il 2% in più della contribuzione teoricamente perfetta.
Nei team con 4 componenti questo valore sale a 27,81%, quasi il 3% oltre la contribuzione paritetica.

Infatti, se ogni studente stimasse correttamente il proprio lavoro la contribuzione media dovrebbe corrispondere esattamente a quella perfetta, come nel caso dei team 4 e 17.

Sono numerosi i gruppi in cui la somma dei contributi individuali supera abbondantemente il 100%: in particolare i team 2, 8, 11, 13, 14 e 16.
In questi team sono presenti studenti che hanno valutato il proprio contributo il 50% del totale o due membri che ritengono di aver contribuito per più di un terzo ciascuno, lasciando agli altri tre la parte restante.

Addirittura nel team 13 uno studente si è attribuito il 90% dello sforzo complessivo e nel team 16 è presente l'unica valutazione dello 0% all'indirizzo di un compagno.

Non è improbabile che durante lo sviluppo di un progetto non banale come questo ci siano componenti che svolgono una parte più corposa del lavoro totale, tuttavia tali sbilanciamenti fanno pensare ad una sovrastima dovuta magari alla mancanza di supporto da parte del team o di comunicazione interna.

Per quanto riguarda la media della deviazione standard non sorprende che sia più alta in corrispondenza dei team sopra elencati. Chiaramente ogni studente ritiene di aver dato un contributo più o meno grande, quindi se altri inseriscono valori sovrastimati la deviazione standard aumenta.

Il caso più eclatante si ha col team 16 dove, come detto in precedenza, uno studente si è attribuito il 90% dello sforzo, facendo sì che anche il minimo contributo da parte degli altri cinque membri portasse ad uno squilibrio della valutazione.

6.3 Analisi delle retrospettive personali

Vengono ora presentate le risposte alle domande 7 e 8 del questionario finale individuale:

7. Retrospettiva personale: cosa hai imparato? Cosa è andato bene? Cosa cambieresti?
8. Cosa ha funzionato nel team? Cosa non ha funzionato? Osservazioni sul gruppo (se ha aiutato, se è stato d'intralcio, o inutile).

Per ogni risposta sono stati contati gli studenti che hanno citato, in modo positivo o negativo a seconda del contesto, un determinato aspetto.

Gli studenti erano liberi di rispondere come preferivano, citando quanti elementi desiderassero, per questo motivo la somma delle risposte non corrisponde al numero di studenti totali.

Nuove conoscenze

Nuove conoscenze	Studenti
Lavorare in gruppo	33
Nuovi software o strumenti	26
Lavorare agile	21
Pratiche agile	20
Rispettare deadline	14
Autovalutazione (retrospettive)	8
Software come prodotto sociale	1

Tabella 6.7: Nuove conoscenze acquisite dagli studenti

Diversi studenti hanno acquisito nuove conoscenze grazie a questo progetto, in particolare, per molti era il primo progetto con un gruppo così grande e con il metodo agile.

Inoltre quasi nessuno aveva mai sviluppato con gli strumenti utilizzati durante il progetto.

14 studenti hanno affermato anche che il progetto sia stato utile per imparare a rispettare le deadline per le consegne.

Elementi positivi e negativi dell'esperienza

Elementi positivi	Studenti
Teamwork	13
Comunicazione interna	11
Proprio codice / prodotto finale	4
Team building	3
Divisione dei compiti	2
Sprint planning	1
SM e PO	1
Utilizzo strumenti CAS	1

Tabella 6.8: Elementi ritenuti positivi dagli studenti

Per quel che riguarda gli elementi ritenuti positivi dell'esperienza ("Cosa è andato bene?"), la maggior parte degli studenti che ha dato una risposta ha indicato il teamwork e la comunicazione interna al team.

Elementi da cambiare	Studenti
Approccio ad uno o più sprint	9
Propria gestione del tempo	7
Problemi con IDE o logger	7
Mattermost	7
Scarsa comunicazione	5
Essere più seguiti durante il percorso	4
Stimare meglio le US	4
Scarsa collaborazione	3
Programmare meglio le attività	3
SonarQube e Jenkins arrivati tardi	3
Comunicare di più con PO	2
Difficoltà con strumenti CAS	2
API Twitter troppo limitate	2
Scrittura dei test	1
Assenze alle riunioni Scrum	1
Trovarsi in presenza alle riunioni Scrum	1
Scadenze troppo ravvicinate	1
Quantità di lavoro eccessiva	1

Tabella 6.9: Elementi ritenuti da cambiare dagli studenti

Per quanto riguarda gli aspetti critici ("Cosa cambieresti?"), le prime due risposte in ordine di voti riguardano aspetti interni al team, ovvero l'approccio ad uno o più sprint e la propria gestione del tempo.

Seguono critiche agli IDE, in particolare la versione di IntelliJ Idea che supporta il Logger, e ai Logger che ad alcuni studenti, specialmente con sistemi MacOS, hanno dato diversi problemi.

Una critica piuttosto diffusa è anche quella a Mattermost, ritenuto fondamentalmente poco utile.

Un aspetto da evidenziare è anche la maggior richiesta di supporto durante il percorso, specialmente nelle fasi iniziali e con le pratiche agili.

Infine, solo due studenti ritengono che le scadenze fossero troppo ravvicinate e che il carico di lavoro fosse eccessivo.

Valutazione del team

Cosa ha funzionato	**Studenti**
Team bonding	26
Lavoro in team	21
Impegno di tutti	15
Suddivisione compiti	14
Colleghi disponibili	11
Comunicazione	10
Applicazione di Agile	8
Aiuto reciproco	7
SM e PO	5
Sprint review	2
Divisione in sprint	1

Tabella 6.10: Tabella riassuntiva di cosa ha funzionato nei team

Passando alle risposte alla domanda 8, sono stati valutati positivamente tutti gli aspetti inerenti il teamwork, dal legame tra i membri alla corretta suddivisione dei compiti durante lo sviluppo.

Meno studenti hanno segnalato di aver apprezzato come il proprio team ha applicato i princìpi di Agile mentre solo uno ha indicato come positiva la divisione in sprint adottata dal proprio team.

Cosa non ha funzionato	Studenti
Lavorare in modo discontinuo	14
Incontrarsi poco	11
Comunicazione	8
Suddivisione compiti	6
Scelta priorità del team	6
Agile nelle fasi iniziali	6
Coordinazione iniziale	6
Lacune o problemi con tecnologie impiegate	4
Stima tempo da impiegare per task	4
Retrospettive	1
SM	1

Tabella 6.11: Tabella riassuntiva di cosa non ha funzionato nei team

La maggiore difficoltà incontrata da alcuni team è stata pianificare il ritmo di lavoro, infatti molti hanno criticato la discontinuità del proprio ritmo di sviluppo.

Altre critiche sono state in generale mosse alla frequenza degli incontri in presenza, limitati anche a causa delle restrizioni sanitarie.

Infine, alcuni hanno indicato le fasi iniziali del progetto come le più difficoltose e hanno nuovamente indicato che avrebbero preferito più supporto da parte dei PO proprio in quelle fasi.

Capitolo 7

CONCLUSIONI

In questo capitolo finale saranno riassunti i principali risultati in apposite sezioni.

7.1 Produttività

Mediamente il prodotto finale consegnato al termine di questo progetto contiene 2622 linee di codice di cui 257 di commenti (9,8%).

Uno studente dichiara mediamente di aver programmato e testato codice per 84 ore, arrivando ad ottenere così una produzione oraria media di oltre 31 LoC/h. Si tratta di un ritmo di produzione troppo elevato, dato probabilmente dalla sottostima delle ore di lavoro, dato che il conteggio delle righe di codice è stato effettuato da SonarQube.

Si tratta del problema inverso a quello identificato nel paragrafo 3.1.1, dove è stato concluso che in media gli sviluppatori sovrastimano la propria produzione in termini di LoC di 882 unità.

Per quanto riguarda il carico di lavoro per un corso da 9 CFU, escludendo le 90 ore di lezione, uno studente dovrebbe aver lavorato per circa 135 ore, mentre, includendo riunioni e ore di supporto al team, il tempo di lavoro medio dichiarato è stato di 144 ore. Tenendo conto dell'elevata deviazione standard di 75,6 nel calcolo della media si può però concludere che il carico di lavoro sia stato probabilmente proporzionato al numero di crediti.

In conclusione è quindi possibile evidenziare l'inaffidabilità dei membri dei team nel misurare senza specifici strumenti la propria produttività, tendendo a sovrastimare la quantità di codice scritta o a sottostimare il numero di ore di lavoro.

7.2 Attinenza ai princìpi Agili

In questa sezione sarà brevemente analizzata la fedeltà ai 12 princìpi[1] del Manifesto Agile[FH+01].

L'unico criterio che non è stato pienamente rispettato è il quarto:

"Committenti e sviluppatori devono lavorare insieme quotidianamente per tutta la durata del progetto"

Questo perché nel contesto universitario non è stato possibile avere una comunicazione quotidiana con i professori, anche se ci sono state diverse riunioni.

Per quanto riguarda il sesto principio ("Una conversazione faccia a faccia è il modo più efficiente e più efficace per comunicare con il team ed all'interno del team"), l'unico appunto doveroso è che le comunicazioni sono avvenute per la maggior parte online, anche a causa delle restrizioni dovute alla pandemia.

Si può quindi affermare che per il metodo con cui sono stati svolti tutti i progetti siano considerabili Agili.

[1] L'elenco completo in italiano è reperibile all'indirizzo `https://agilemanifesto.org/iso/it/principles.html`

7.3 Confronto tra i due modelli

Per concludere, in questa sezione sarà presentato un confronto grafico tra i risultati ottenuti dai due modelli.

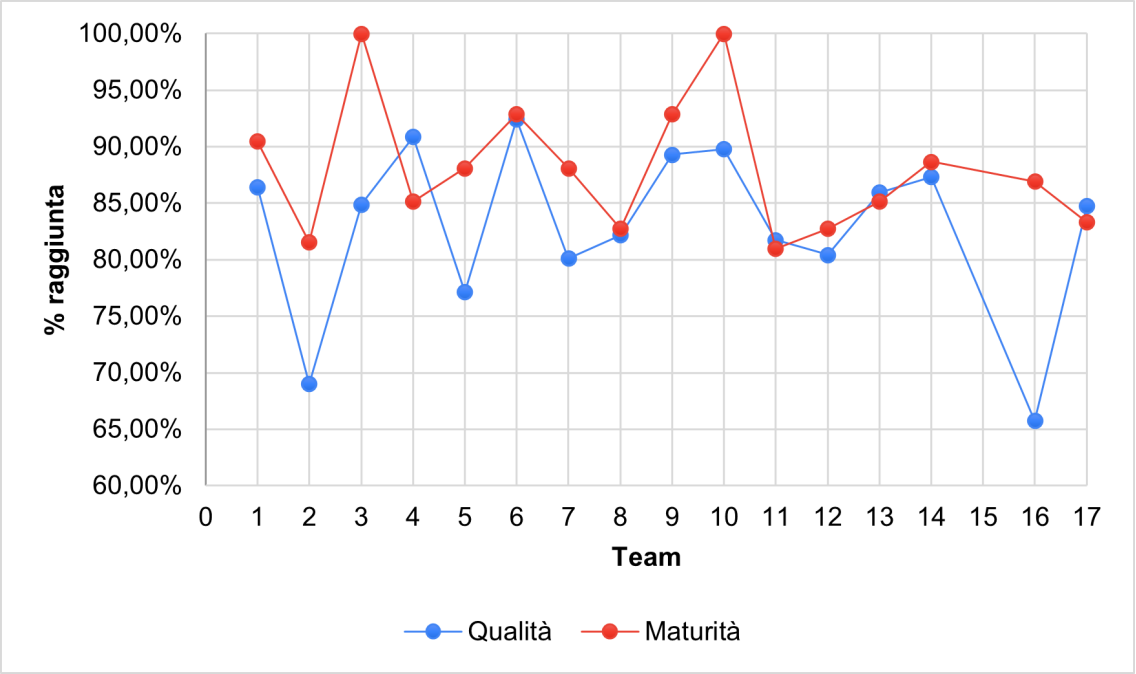

Figura 7.1: Confronto dei risultati ottenuti dal modello di qualità e da quello di maturità

Come si può notare, tutti i team tranne i numero 4, 11, 13 e 17 hanno ottenuto migliori risultati nel modello di maturità.

I grafici hanno un andamento abbastanza similare per quasi tutta l'estensione, tuttavia l'indice di Pearson ρ, che rivela se è possibile una correlazione lineare tra i set di dati, è uguale a 0,436 su un massimo di 1.
Questo significa che non è possibile stabilire una correlazione lineare tra i due modelli, come è avvenuto lo scorso anno con $\rho = 0,8$.

Probabilmente questa differenza è data dal fatto che nel progetto per l'anno 2021/2022 ci fossero molti più requisiti obbligatori che hanno influito sulle valutazioni nei due modelli, specialmente quello di maturità.

È quindi ipotizzabile una correlazione lineare solo se i team sono effettivamente liberi di sviluppare senza l'obbligo di dover seguire determinate regole, ma si tratta di un'ipotesi da verificare.

Riferimenti bibliografici

[Bar20] Roberto Barbone. Virtual team building through serious games: the scrumble case study. Tesi di laurea - Alma Mater Studiorum - Università di Bologna, 2020.

[CMPR20] Paolo Ciancarini, Marcello Missiroli, Francesco Poggi, and Daniel Russo. An Open Source Environment for an Agile Development Model. In *Proc. Int. Conf. on Open Source Systems (OSS)*, volume 582 of *IFIP Advances in Information and Communication Technology*, pages 148–162, Innopolis, Russia, 2020. Springer.

[CMZ21] Paolo Ciancarini, Marcello Missiroli, and Sofia Zani. Empirical evaluation of agile teamwork. In A. Paiva, A. Cavalli, P. Ventura Martins, and R. Pérez-Castillo, editors, *Proc. 14th Int. Conf. on Quality of Information and Communications Technology QUATIC*, volume 1439 of *Communications in Computer and Information Science*, pages 141–155. Springer, 2021.

[CR09] Patel Chetankumar and Muthu Ramachandran. Agile maturity model (amm): A software process improvement framework for agile software development practices. *International Journal of Software Engineering*, 2, 01 2009.

[FH+01] Martin Fowler, Jim Highsmith, et al. The agile manifesto. *Software development*, 9(8):28–35, 2001.

[GGJ20] Lucas Gren, Alfredo Goldman, and Christian Jacobsson. Agile ways of working: a team maturity perspective. *Journal of Software: Evolution and Process*, 32(6):e2244, 2020.

[HG01] Martin Hoegl and Hans Georg Gemuenden. Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization science*, 12(4):435–449, 2001.

[LSD+16] Yngve Lindsjørn, Dag IK Sjøberg, Torgeir Dingsøyr, Gunnar R Bergersen, and Tore Dybå. Teamwork quality and project success in software development: A survey of agile development teams. *Journal of Systems and Software*, 122:274–286, 2016.

[VR21] Christiaan Verwijs and Daniel Russo. A theory of scrum team effectiveness. *arXiv preprint arXiv:2105.12439*, 2021.

[YFdS11] Alexandre Yin, Soraia Figueiredo, and Miguel Mira da Silva. Scrum maturity model. *Proceedings of the ICSEA*, pages 20–29, 2011.

[Zan20] Sofia Zani. Comparazione empirica di sviluppi agili mediante un modello di qualità e un modello di maturità. Tesi di laurea - Alma Mater Studiorum - Università di Bologna, 2020.

Appendici

Appendice A

Questionario finale individuale

1. Numero del team in cui hai lavorato
2. Tua matricola personale (ultime sei cifre)
3.
   (a) Sei SM del tuo team?
   (b) Sei PO del tuo team?
4. Descrizione del tuo tool personale: quale IDE e se hai utilizzato il logger
5.
   (a) Indicare il numero di linee di codice complessivamente prodotte da te durante tutto il progetto (quindi considerando anche le cancellate):
   (b) Indicare la stima del numero di linee di codice di tua produzione individuale rimaste nel codice definitivo del prodotto finale:
6. Elenco delle pratiche agili che hai usato. (Esempi: daily scrum, sprint planning, pair programming...)
7. Retrospettiva personale: cosa hai imparato, cosa è andato bene, cosa cambieresti.
8. Cosa ha funzionato nel team, cosa non ha funzionato, con osservazioni sul gruppo (se ha aiutato, se è stato d'intralcio, o inutile).
9. Indicare a seguito una propria stima del contributo percentuale al progetto fornito da ogni componente del team in forma anonima, incluso voi stessi.
   (a) IO

(b) Componente 1

(c) Componente 2

(d) Componente 3

(e) Componente 4

(f) Componente 5

10. C'è stata una comunicazione spontanea nel team durante il progetto (per esempio: frequenti conversazioni in chat, incontri non programmati perché mentre lavoravate in solitaria comunque vi collegavate a canali di comunicazione comune come Discord, riunioni in presenza)?
11. La comunicazione di informazioni rilevanti è sempre stata condivisa con tutti i membri del team?
12. La comunicazione di informazioni nel team è sempre avvenuta in modo tempestivo?
13. Il team è stato in grado di fornirti (o ti ha aiutato a reperire) delle informazioni utili e precise quando hai avuto dei dubbi?
14. Il team ha mai chiesto aiuto all'esterno? Esempio: ad una persona di un altro team?
15. Quando hai dovuto svolgere delle task che erano strettamente correlate con quelle di un compagno avete avuto dei problemi nel coordinarvi?
16. C'è sempre stata un'accettazione condivisa nell'assegnazione delle task nel gruppo?
17. Se hai avuto difficoltà con lo svolgimento delle tue task hai avuto aiuto dal team?
18. Sul darsi obiettivi e risolvere questioni importanti il team ha sempre risolto in fretta?
19. Ritieni che la cooperazione sia sempre andata bene nel team?
20. Secondo te ogni membro del team ha dato la sua contribuzione al meglio delle sue capacità e disponibilità di tempo?
21. Ci sono state discussioni perché si riteneva che qualcuno non desse il suo contributo?
22. Secondo te il tuo lavoro è stato più o meno importante rispetto a quello del resto del team?

23. Secondo te, tutti i membri si sono sempre impegnati per continuare a farsi sentire del team e tenere unito il gruppo?
24. La suddivisione delle task seguiva sempre le tue richieste sulla base della tua competenza in quel campo?
25. Con questo progetto hai imparato un metodo e degli strumenti secondo te utili in futuro?
26. Che autovalutazione daresti complessivamente al prodotto del tuo team?
27. Che autovalutazione daresti complessivamente al processo del tuo team?
28. Quante ore totali stimi di aver passato a programmare/testare codice?
29. Quante ore pensi di aver passato in incontri col team o membri del team? (incontri di pair programming, riunioni di team..)
30. (Solo per SM o PO) Quante ore di supporto al team pensi di aver fatto? (se non sei SM o PO scrivi /)
31. Quanto è stato utile Taiga?
32. Quanto è stato utile GitLab?
33. Quanto è stato utile SonarQube?
34. Quanto è stato utile Mattermost?
35. Quanto è stato utile Jenkins?
36. Quanta è stata utile la partita a Scrumble?
37. Quanto è stato utile fare le retrospettive con Essence?

Appendice B

Questionario GQM Scrumble

- Goal: learn
  - Do team members understand the Scrum roles?
  - Do team members feel they learned the process?
  - Does everyone keep up with the other players?
- Goal: practice
  - Are the game mechanics linear and repeatable?
  - Do team success in completing the game?
  - Do team members efficiently estimate during sprint planning?
- Goal: cooperation
  - Do team members know each other better?
  - Does the game let all players cooperate?
  - Do team member consult each other about a topic?
- Goal: motivation
  - Do team members encourage collegues in need?
  - Does PO help the team?
  - Does the team come up with good ideas?
- Goal: problem solving
  - Do team members behave well when facing a problem?

– Does team organize their tasks properly?
– Does PO plan efficiently the Sprint Backlog?

Appendice C

Elenco completo dei requisiti

Nelle pagine seguenti è possibile trovare il documento contenente tutti i requisiti del progetto.

Progetto di Ingegneria del software 2021

Paolo Ciancarini

11 ottobre 2021

1 Introduzione

Vogliamo costruire uno strumento di raccolta e analisi dei tweet di Twitter, specie quelli con foto e geolocalizzabili. I tweet possono riferirsi ad una persona, ad un luogo, ad un evento. In alcuni casi i tweet riguardano situazioni di emergenza. In altri possono essere usati per diffondere informazioni o per farsi pubblicità.

Le persone che vivono un'emergenza e la comunicano attraverso un social media quale Twitter possono essere considerate dei sensori umani sul territorio e i messaggi che si scambiano generano, se aggregati in tempo reale, una serie di informazioni utili proprio per gestire l'emergenza stessa.

In una situazione di protezione civile, come subito dopo un terremoto distruttivo, i tweet costituiscono una delle poche fonti immediate ed economiche di informazione dal campo, utilizzabile per ricostruire una situazione operativa[1].

Ci sono tante altre situazioni interessanti e non emergenziali, ad esempio: la condivisione di un evento come un concerto o una partita di calcio, con le foto. Oppure i visitatori di un museo che in uno stesso periodo di tempo pubblicano tweet sul museo stesso. In questo caso i tweet possono essere "ricomposti" per offrire un punto di vista speciale sull'evento o sul luogo visitato.

Un'altra situazione tipo è quella di un gruppo di turisti che si sposta in una città, visitandola e condividendo commenti e foto via tweet: si può cercare di ricostruire i movimenti delle persone del gruppo a partire dai tweet che producono, piazzando inoltre le foto su una mappa.

2 Descrizione del problema

L'obiettivo del progetto è quello di creare un'applicazione in grado di raccogliere i tweet e organizzarli. Lo scopo è rilevare eventi locali o basati su particolari parole chiave esaminando i tweet stessi. La raccolta può essere storica (es. ultima settimana) o in stream in tempo reale.

L'applicazione dovrà permettere di visualizzare, consultare i tweet con certi hashtag e - sotto certe condizioni- attivare una procedura specifiche. Ad esempio, il sistema dovrà essere in grado di:

- raccogliere i tweet (scoprirete che certe raccolte di tweet sono a pagamento: ovviamente vogliamo usare solo tweet gratis, quindi imparate a raccoglierli voi)
- classificare i tweet geolocalizzati
- classificare i tweet pubblicati in una data area geografica
- classificare i tweet contenenti un certo punto di interesse, ad esempio #SanSiro
- classificare i tweet con una certa parola chiave, ad esempio #viaggiodiclasse5B
- classificare i tweet geolocalizzati di una persona specifica, per seguirne gli spostamenti
- analizzare il sentimento (*sentiment analysis*) di una serie di tweet, ad esempio un gruppo di persone che visita un museo gradisce la visita?

[1] blog.twitter.com/en_sea/topics/insights/2018/5-Tips-for-using-Twitter-during-emergencies-and-natural-disaster.html

I tweet così raccolti e classificati potranno essere aggregati in vari modi, principalmente in modalità grafica.

In particolare si richiede di aggregare i tweet in una *dashboard* (visualizzatore) interattiva, possibilmente mostrando viste coordinate che presentino diversi dettagli dei dati (es. le posizioni su una mappa, una *term cloud* (nuvola di parole dei termini usati nei tweet), un diagramma a barre con il numero di tweet nell'unità di tempo, ecc.), un grafico a torta con sentiment positivi e negativi. È un requisito fondamentale mostrare la posizione dei dati raccolti su una mappa (es. su Google Maps).

Suggerimento: iscrivetevi a Twitter (se non lo siete già) e trasmettete vostri tweet personali usando l'hashtag `#IngSw2021`. Pubblicate anche tweet su luoghi o eventi o persone a vostra scelta. Attivate sulla app Twitter del vostro telefono la geolocalizzazione dei tweet (questa è una funzione volontaria di Twitter[2]).

IMPORTANTE: ogni team può aggiungere specifiche funzioni di propria scelta al prodotto, previa approvazione del PO. La proposta di funzionalità è a cura del PO Operativo, che contatta il docente. Siate creativi, la cosa verrà apprezata adeguatamente in sede di esame.

Esempio: una proposta riguarda la possibilità di fare una mappatura dei bagni pubblici di una città, per esempio Bologna. Quando usate un bagno pubblico - in un bar, un ristorante, all'università,ecc.-, dopo mandate un tweet usando l'hashtag #WC-OK e un commento geolocalizzato. Si propone di costruire una mappa aggiornata dei bagni pubblici "di qualità" di Bologna.

3 Un processo agile "estremo"

I gruppi di progetto dovrebbero avere cinque membri. Tollerati gruppi con un membro in più o in meno solo se residuali.

Uno dei membri deve assumere il ruolo di Product Owner Operativo (in raccordo coi PO che sono i docenti); un altro membro sarà Scrum Master. Gli altri membri sono developer (sviluppatori).

Il PO Operativo è il responsabile del prodotto entro il team, e partecipa allo sviluppo. Lo Scrum Master è il coordinatore di processo entro il team (e quindi dei documenti di processo, in particolare del rapporto finale, che comunque verrà firmato da tutti i membri del team). Anche lo Scrum Master può partecipare allo sviluppo.

Il processo si svolge su almeno tre sprint di tre settimane ciascuno, a partire dall'11 ottobre. Ci sono inoltre una fase preliminare (*sprint zero*) e una fase conclusiva (*release sprint*), che conclude lo sviluppo.

Fase preliminare a) Team forming tramite Trello b) Addestramento a Scrum del gruppo mediante uso del gioco Scrumble (vedere link alla fine di questo documento), con raccolta degli indicatori di valutazione del team da parte dello Scrum Master. Una partita a Scrumble si può giocare sia in presenza sia con Teams e richiede circa un paio di ore.

Fase di esecuzione Il team esegue il progetto con almeno tre sprint della durata di tre settimane. Al termine di ciascuno sprint il team dovrà preparare una demo dello stato di avanzamento del prodotto (*Sprint review*), una riflessione sul processo (*retrospettiva*) usando le carte Essence, e una sessione di *backlog grooming* cioè una revisione del backlog dopo una riunione coi PO docenti.

IMPORTANTE: all'inizio di sprint 2 e di sprint 3 il PO (Docente) fornirà nuove user story, che dovranno essere aggiunte alle altre, eventualmente modificandole.

Fase conclusiva Preparazione del rapporto finale a cura del team, inclusa autovalutazione (verranno date indicazioni sul rapporto finale e sull'autovalutazione entro la fine del corso). Consegna del software su gitlab@CAS e del rapporto finale entro 25 gennaio. Si richiede la preparazione di un breve video con audio (max 3 min.) di presentazione del prodotto sviluppato, che mostri anche un'analisi "interessante" eseguita con il prodotto stesso.

Voti Voto unico al team basato su demo e qualità del rapporto finale.

Allo scopo di mostrare la flessibilità dell'approccio agile, ciascun team decide e inizia lo sviluppo con un insieme di user story che potrà essere integrato in corso d'opera con altre, richieste dal docente.

[2] https://help.twitter.com/en/safety-and-security/tweet-location-settings

Esempi (lista non esaustiva): aggiungere un elemento di filtro/selezione dei tweet, come la possibilità di selezionare una finestra temporale, in modo che l'utente del sistema possa selezionare i tweet solo di un determinato periodo; aggiungere la possibilità di cliccare su una delle parole della tag cloud in modo da filtrare i dati sulla base di tale parola chiave e di vedere quindi la mappa aggiornata di conseguenza; aggiungere altre sintesi grafiche di data analytics, quale un istogramma che mostri la frequenza con cui è comparsa una parola chiave nella finestra temporale prescelta.

Tecnologie: per lo sviluppo potete usare Java, Javascript, Python o altre tecnologie, a vostra scelta. Si richiede che il deployment sia comunque basato su docker. Per quanto possibile TUTTE le tecnologie utilizzate dovrebbero essere open source.

3.1 I servizi di CAS

L'uso dell'ambiente di sviluppo CAS [1] è obbligatorio; il nostro scopo è di fare una esperienza di sviluppo "estremo", cioè totalmente basato su tecnologie open source; inoltre vogliamo poter raccogliere dati di processo grazie alle funzionalità presenti nell'ambiente CAS [2]. L'ambiente CAS contiene i seguenti elementi, tutti open source:

1. Taiga (simile a Trello) per il project management e per raccogliere alcuni documenti nel suo wiki
2. Mattermost (simile a Slack) per le comunicazioni tra gli sviluppatori
3. SonarQube per l'analisi statica del codice
4. gitlab per il controllo delle versioni e della configurazione
5. jenkins per il testing e la *Continuous Integration*
6. logger server, per raccogliere dati di sviluppo direttamente dai client IDE arricchiti con plugin (disponibili plugin per Eclipse, Atom, IntelliJ).

L'ambiente CAS è disponibile in più versioni:

- server AMINSEP: all'indirizzo `https://aminsep.disi.unibo.it` si accede Taiga. Gli altri servizi si accedono aggiungendo il nome del servizio: ad esempio gitlab si accede da `https://aminsep.disi.unibo.it/gitlab`
- CAS si può istallare su un server privato del team, ma alla fine del progetto occorre duplicare il contenuto del gitlab privato su gitlab@CAS
- è accettabile deployare CAS su cloud, ma alla fine del progetto occorre duplicare il contenuto del gitlab privato su gitlab@CAS

Le carte Essence [3] per le retrospettive si trovano al link di cui sotto.
Altre informazioni verrano date a lezione.

4 Link

Twitter https://developer.twitter.com/en/docs

Tweet-Tracker https://github.com/Zacomo/Tweet-Tracker

Scrumble http://scrumble.pyxis-tech.com

Essence Introduzione http://semat.org/essence-user-guide

Carte Essence https://practicelibrary.ivarjacobson.com/start

Riferimenti bibliografici

[1] P. Ciancarini, M. Missiroli, F. Poggi, and D. Russo. An open source environment for an agile development model. In *IFIP International Conference on Open Source Systems*, volume 582 of *Advances in Information and Communication Technology*, pages 148–162. Springer, 2020.

[2] P. Ciancarini, M. Missiroli, and S. Zani. Empirical evaluation of agile teamwork. In *International Conference on the Quality of Information and Communications Technology*, pages 141–155. Springer, 2021.

[3] I. Jacobson, H. Lawson, P. Ng, P. McMahon, and M. Goedicke. *The Essentials of Modern Software Engineering*. ACM Books. Morgan & Claypool Publishers, 2019.

Ringraziamenti

Dedico questa ultima pagina ai ringraziamenti a coloro che in questi ultimi tre anni mi sono stati vicini.

Ringrazio in primo luogo il mio relatore, prof. Paolo Ciancarini per il supporto e i preziosi consigli durante lo svolgimento della tesi.

Ringrazio anche il prof. Andrea Cosso per i consigli sull'elaborazione statistica utili per la stesura della tesi.

Un ringraziamento va anche a tutti i miei amici, sia quelli conosciuti durante il percorso universitario, sia quelli di sempre.

Un ringraziamento particolare agli amici del gruppo "UAJAA", ai "Futuri informatici" e a Mario Miccichè per essere oltre che degli ottimi amici, dei validi compagni di studio e di progetti.

Grazie anche agli amici del "The", ai "Dadoni" e a Davide, Andrii, Enrico e Leo per non aver fatto mancare i momenti di spensieratezza prima e durante questi anni.

Infine, certamente non in ordine di importanza, ringrazio tutta la mia famiglia che mi ha sostenuto e supportato durante questi tre anni.