POLITECNICO DI TORINO

Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Sviluppo di un'applicazione web a supporto dell'empatia e delle competenze trasversali

Relatori
Prof. Mariagrazia Graziano

Candidato
Carlos Eduardo Torre

April 5, 2021

Compiled with LaTeX on April 5, 2021.

Abstract

Il lavoro proposto nasce da un'osservazione sulla base dell'esperienza condivisa dall'autore e da colleghi studenti e lavoratori: in genere, un neolaureato in ingegneria possiede un'estesa e profonda preparazione dal punto di vista tecnico ma, allo stesso tempo, presenta lacune nel sapersi relazionare efficacemente in un contesto lavorativo, lacune che possono essere più o meno evidenti a seconda dell'individuo. Pertanto, il lavoro svolto ha un duplice obiettivo: in primo luogo, cercare di trovare un riscontro effettivo sull'ipotesi di partenza e, poi, di avanzare proposte di soluzione a quello che può essere considerato uno svantaggio competitivo per i neolaureati ingegneri che si interfacciano al mondo lavorativo.
Innanzitutto, è stato proposto un sondaggio a studenti/dottorandi del Politecnico di Torino e a ingegneri già inseriti in contesti professionali. Sono state effettuate, in parallelo, svariate interviste: da ingegneri con ruoli, esperienze e background eterogenei a persone che hanno lavorato a lungo nelle risorse umane in settori differenti. Una volta constatata l'ipotesi formulata, si è delineata una base di partenza e comune ai risultati ottenuti tramite il sondaggio e le interviste. Questo ha permesso di indirizzare la ricerca verso l'approfondimento dell'impatto che può avere un approccio empatico e di ascolto attivo in un contesto lavorativo, sottolineando la necessità per gli studenti di intraprendere un percorso di sviluppo di competenze trasversali, normalmente trascurate durante la carriera universitaria.
Alla luce della base delineata e dell'analisi teorica effettuata, infine, sono state avanzate alcune proposte per migliorare il percorso universitario degli studenti. Proposte che hanno condotto anche allo sviluppo di una versione basilare di applicazione web per supportare il corso di Engineering Empathy e favorire la collaborazione tra studenti e docenti. L'analisi e l'applicazione sono volti a sensibilizzare le persone sull'importanza dello sviluppo dell'empatia e dell'ascolto attivo e, contemporaneamente, a permettere di creare uno spazio per trattare e potenziare le competenze trasversali, rendendo così professionalmente completa la figura del neolaureato in ingegneria.

Alla mia famiglia e a tutti i nostri sacrifici

Ringraziamenti

Come mi ha insegnato uno dei miei autori preferiti, quando si arriva a raggiungere un traguardo importante non resta che condividere un po' di serotonina con le persone a noi più care...

Innanzitutto, vorrei ringraziare la mia famiglia per tutto quello che ha fatto per me fino ad oggi, per tutto quello che ha costruito nel tempo e per essere riuscita a regalarmi una vita che poche altre persone hanno il privilegio di condurre. È solo grazie a voi se ho avuto la fortuna di intraprendere questo percorso e, allo stesso tempo, di potermi dedicare a passioni e molto altre attività. Non tanti altri ragazzi hanno la fortuna che ho avuto io di poter sempre contare sulla vostra enorme solidità sia dal punto di vista affettivo e protettivo sia economico, aspetti che mi hanno regalato il privilegio di poter continuare a giocare a calcio, coltivare molto tempo prezioso con la mia fidanzata e i pochi ma preziosi amici, leggere e imparare tantissimi argomenti al di fuori del campo di studi universitari, svolgere tante attività e vivere alcune esperienze memorabili. Grazie davvero per tutto. Voglio che sappiate che questo per me non è solo un piccolissimo traguardo, bensì è anche l'inizio di un nuovo percorso di vita da cui partire per mettere in pratica i valori che mi avete trasmesso e tutto quello che mi avete insegnato, perché soltanto in questo modo saprò di poter contraccambiare tutto quello che mi avete dato in questi 25 anni - e poco più - di vita.

Un ringraziamento speciale va a Federico che incredibilmente mi sopporta, stima e vuole bene da ormai quasi 10 anni; in aggiunta, voglio ringraziare anche tutta la tua famiglia per avermi dimostrato sempre grande affetto e vicinanza in tutti questi anni. Ci tengo a ringraziare tantissimo Alessandro, amico caro e speciale per me: purtroppo la lontananza geografica non ci permette di vederci spesso ma con il cuore e la testa siamo sempre vicini e connessi, stimandoci sempre l'un l'altro. Lo stesso vale per Massimiliano, per via del tuo ruolo/lavoro abbiamo avuto poche occasioni per vederci in questi ultimi anni, ma è sempre un piacere enorme poter condividere del tempo con te e, ogni tanto, ricordare qualche aneddoto divertente e/o romantico sul nostro passato.

Voglio ringraziare con tutto il cuore Carmen per aver reso speciali tutti questi anni, per essermi sempre stata accanto anche nei momenti più difficili tra noi e per aver riempito d'amore e passione incondizionati il nostro rapporto. Mi hai trasmesso tanto e, soprattutto, dimostrato che cosa voglia davvero dire dare incondizionatamente e regalare se stessi - e non solo - con tutto il cuore per amore. Hai dimostrato di avere valori molto importanti, valori che ho percepito anche nella tua famiglia che voglio ringraziare per avermi accolto, voluto bene e sostenuto a lungo.

Un ringraziamento particolare va anche alla Professoressa Graziano, senza la quale tutto questo lavoro non sarebbe stato possibile. Ammiro il suo coraggio per aver dato il via a questo nuovo percorso al Politecnico di Torino e per averci dato la possibilità di partecipare a un corso speciale, in cui abbiamo imparato molto, siamo cresciuti e maturati tutti insieme. Grazie anche ad Elisa per essere stata una compagna eccezionale di quest'avventura, ti ammiro molto per quello che stai facendo nella tua vita con enorme dedizione e spirito di sacrificio.

Voglio ringraziare le due squadre di calcio del Politecnico, con cui ho avuto la fortuna di condividere due esperienze incredibili e fantastiche in Cina. Alla prima squadra va un enorme ringraziamento per il feeling che si è creato immediatamente con tutti e per aver legato così velocemente da aver contribuito a rendere speciale la mia festa di laurea triennale; alla seconda, dire grazie è fin troppo riduttivo per l'esperienza meravigliosa che abbiamo vissuto e che ricorderemo per il resto della nostra vita. Grazie a tutti coloro che hanno organizzato il nostro meraviglioso viaggio, grazie ai Mister Camolese e Montanarelli, grazie a Pattavina che mi ha sempre sostenuto e grazie a tutti i ragazzi che ho avuto l'onore di guidare con la fascia da capitano in quella che probabilmente è stata una delle esperienze più entusiasmanti e favolose della mia vita.

Un grazie particolarmente strano va a tutta la fatica, il sudore che mi hanno fatto fare i miei professori del liceo, qualcuno in particolare. Questo mi ha permesso di "vivere di rendita", di poter affrontare il Politecnico e rendermi conto alla fine del percorso universitario che per me si è trattato di una semplice passeggiata rispetto a tutti i miei colleghi. Non avrei mai immaginato di potervi ringraziare, ma a quanto pare è stato un valore aggiunto nella mia vita.

Infine, ci tengo immensamente, ancora una volta, a ringraziare con tutto il mio cuore la mia famiglia, i miei amici più intimi e Carmen. Vi voglio tanto bene.

Carlos

Indice

Elenco delle figure

Capitolo 1

Introduzione

La presente tesi studia, analizza e approfondisce:

- l'impatto dell'empatia nel lavoro ingegneristico
- il percorso per sviluppare l'approccio empatico sia in contesto universitario sia lavorativo
- le fondamenta teoriche delle competenze trasversali derivanti dallo sviluppo dell'empatia (leadership e lavoro in gruppo, problem solving e creatività, public speaking) e che, al tempo stesso, hanno come fine ultimo appunto l'entrare in empatia e sentirsi connessi con le persone

Con l'obiettivo di dar vita, all'interno della percorso di studi universitari del Politecnico di Torino, a un nuovo percorso di sviluppo dell'empatia e delle competenze trasversali più richieste ad oggi per la figura del neolaureato ingegnere.

Il lavoro proposto nasce da un'osservazione sulla base dell'esperienza condivisa dall'autore e da colleghi e lavoratori: in genere, un neolaureato in ingegneria possiede un'estesa preparazione dal punto di vista tecnico ma, allo stesso tempo, presenta lacune nel sapersi relazionare efficacemente in un contesto lavorativo, lacune che possono essere più o meno presenti e di rilievo a seconda dell'individuo. Per quanto questa osservazione non sia ancora stata dimostrata scientificamente, trova riscontri in vari studi ma soprattutto nell'esperienza quotidiana di moltissime persone.
Le motivazioni principali per le quali è stato scelto questo argomento sono:

- la forte volontà da parte del sottoscritto di apportare un miglioramento all'interno del percorso di studi del Politecnico di Torino affinché tutti gli studenti abbiano la possibilità di sviluppare non solo competenze tecniche ma anche trasversali e relazionali attraverso un approccio empatico con le altre persone

- una riflessione del sottoscritto: il mondo si sta evolvendo in maniera molto veloce e, al giorno d'oggi, la tecnologia e la conoscenza sono (quasi) alla portata di tutti. Questo ha portato a un notevole miglioramento nella condivisione delle idee e soprattutto nel fatto che le competenze pratiche possano essere sviluppate (quasi) da chiunque in qualsiasi momento. Ci si ritrova a essere sovente iperattivi, in un processo di vorace sviluppo e apprendimento delle competenze tecniche. Pertanto, il sottoscritto ritiene che a lungo andare non sarà soltanto la quantità di nozioni teoriche e tecniche che si apprendono a rendere le persone in grado di dare valore aggiunto all'interno degli ambienti lavorativi: ciò che farà davvero la differenza non sarà più, appunto, la *quantità*, bensì la *qualità* che si mette nel proprio operato quotidiano. E la qualità sarà determinata dalla capacità di riuscire ad apportare, in contesto lavorativo, valore aggiunto da un punto di vista relazionale: chi è in grado di adattarsi maggiormente ai cambiamenti, a entrare in sintonia ed empatia con le persone, a creare un ambiente di lavoro sereno e armonioso, a lavorare efficacemente in svariati contesti di gruppo e a comunicare efficacemente le proprie idee... Possiede quella marcia in più che gli può permettere di differenziarsi dagli altri e apportare qualità, oltre alla quantità. Proprio come l'inglese è diventato, nel giro di pochi anni, dall'essere un valore aggiunto all'essere un elemento quasi obbligatorio, lo stesso in futuro probabilmente accadrà per le competenze relazionali poc'anzi elencate

Pertanto, in partenza sono stati stabiliti molteplici obiettivi:

- cercare di trovare un riscontro effettivo sull'ipotesi iniziale: non solo da un punto di vista scientifico, in quanto abbastanza complicato trovare al giorno d'oggi studi specifici e approfonditi su questi temi, ma soprattutto tentando di trovare persone che condividono l'ipotesi iniziale in base alla propria esperienza, sia nel contesto degli studi universitari sia in ambito lavorativo

- analizzare, da un punto di vista teorico, l'impatto che può avere un approccio empatico in un contesto lavorativo ingegneristico

- presentare, approfondire e analizzare tutte le competenze trasversali principalmente richieste alla figura dell'ingegnere (anche neolaureato) che derivano dall'emapatia e hanno come fine ultimo, appunto, l'empatia e la connessione con le persone: la leadership e la capacità di lavorare in gruppo, la creatività legata al problem solving e il public speaking

- avanzare proposte di soluzione a quello che può essere considerato uno svantaggio competitivo per i neolaureati ingegneri che si interfacciano al mondo lavorativo, proponendo nuove soluzioni e miglioramenti per il percorso universitario ingegneristico

- proporre un qualcosa che sia utilizzabile a livello pratico, nel breve termine, per apportare già un miglioramento all'interno del percorso di studi, sfruttando un modello di apprendimento efficace per lo sviluppo delle competenze trasversali

Per tentare di verificare l'ipotesi di partenza, si è proceduto parallelamente in due modi:

- È stato proposto un sondaggio a vari studenti e dottorandi del Politecnico di Torino e a lavoratori che ricoprono ruoli relativi all'ambito ingegneristico
- Sono state effettuate 7 interviste: cinque di queste ad ingegneri che ricoprono ruoli completamente differenti e hanno esperienze totalmente eterogenee in svariati contesti, mentre le restanti due interviste sono state svolte con persone che hanno lavorato a lungo nelle risorse umane in contesti diversi

I risultati ottenuti sono stati analizzati attentamente e a lungo e si è delineata una base di partenza per indirizzare il lavoro di ricerca verso l'ideazione di una o più soluzioni concrete ed applicabili per gli studenti del Politecnico di Torino.

L'elaborato è diviso concettualmente in quattro macro aree che rispecchiano la metodologia adottata nello svolgimento del lavoro:

- *Capitolo 2:* presenta la raccolta delle informazioni sulla situazione attuale riguardo alle relazioni in contesto universitario e lavorativo, a partire dal sondaggio fino alle interviste effettuate
- *Capitolo 3:* presenta l'analisi teorica dell'impatto che può avere un approccio empatico in un contesto lavorativo. In particolare, si mostra come l'empatia sia alla base di tutte le competenze trasversali che gli ingegneri sono invitati e tenuti a sviluppare al giorno d'oggi e, al tempo stesso, il fine ultimo
- *Capitolo 4:* mostra la necessità di un cambiamento sia in ambito lavorativo ma soprattutto anche nel corso degli studi universitari. Si avanzano alcune proposte di miglioramento e si presenta l'ideazione di un nuovo percorso, sfruttando il modello di apprendimento 70:20:10, per favorire lo sviluppo dell'empatia e delle competenze trasversali principalmente richieste ai neolaureati ingegneri
- *Capitolo 5:* presenta l'applicazione che è stata creata per supportare il nuovo percorso di apprendimento ideato
- *Capitolo 6:* presenta le conclusioni dell'intero lavoro

Capitolo 2

Analisi preliminare

2.1 Raccolta informazioni

L'ipotesi di partenza è rappresentata dall'osservazione, sulla base dell'esperienza condivisa dall'autore e da colleghi studenti e lavoratori, che un neolaureato in ingegneria possiede, in genere, un'estesa e profonda preparazione dal punto di vista tecnico ma, allo stesso tempo, presenta lacune nel sapersi relazionare efficacemente in un contesto lavorativo, lacune che possono essere più o meno evidenti a seconda dell'individuo.
Per tentare di trovare un riscontro sull'ipotesi iniziale, è stato proposto innanzitutto un sondaggio a studenti/dottorandi del Politecnico di Torino e a ingegneri già inseriti in contesti professionali. Inoltre, sono state effettuate in parallelo svariate interviste: da ingegneri con ruoli, esperienze e background eterogenei a persone che hanno lavorato a lungo nelle risorse umane in settori differenti.

2.1.1 Il sondaggio

Il sondaggio è stato creato tramite Google Forms [1] con l'obiettivo di cercare di comprendere quale sia la situazione attuale dal punto di vista delle persone che hanno vissuto in prima persona il percorso universitario ingegneristico e delle persone che, una volta terminati lo stesso tipo di studi, si sono interfacciate al mondo lavorativo.
Nel sondaggio sono stati trattati tre macro argomenti: competenze tecniche (hard skill), competenze trasversali (soft skill) e, infine, empatia ed ascolto attivo. La prima domanda ha l'obiettivo di identificare il percorso per l'utente, a seconda che si tratti di uno studente o di un lavoratore: in base alla risposta selezionata, ci sono due tipi di percorsi. Nel caso di uno studente, prosegue con le sezioni *Il lavoro di gruppo*, *Hard Skills vs Soft Skills*, *Il futuro è ora*, *L'empatia* e *Le tue idee contano*, per un totale di 30 domande. Nel caso di un lavoratore, invece, si prosegue con

le sezioni *Il lavoro di gruppo*, *Post studi*, *Hard Skills vs Soft Skills*, *Il futuro è ora*, *L'empatia* e *Le tue idee contano*, per un totale di 33 domande. Le sezioni da *Hard Skills vs Soft Skills* in avanti coincidono per entrambi i percorsi.
Quasi tutte le domande richiedono di inserire una valutazione da 1 a 7 per indicare quanto si è d'accordo con la frase corrispondente: in maniera crescente, il numero 1 indica *Per niente d'accordo* e il numero 7 *Totalmente d'accordo*. Si analizzano separatamente, nel seguito, le prime sezioni che non coincidono nei due percorsi, per poi analizzare in maniera congiunta le sezioni uguali per entrambi i percorsi.

Il lavoro di gruppo (studenti) In questa sezione, l'obiettivo è cercare di comprendere quale sia la situazione attuale per gli studenti: se si trovino a proprio agio o fatichino a lavorare in gruppo, se questo avviene in maniera efficace e così via. Le domande proposte sono le seguenti:

1. *Mi trovo sempre perfettamente a mio agio a lavorare in gruppo*

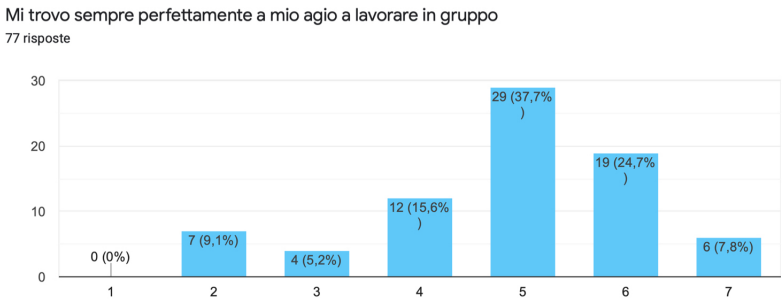

Figura 2.1. Il lavoro di gruppo (studenti), prima domanda

2. *Nei lavori di gruppo c'è sempre stata la giusta armonia e uno spirito positivo*

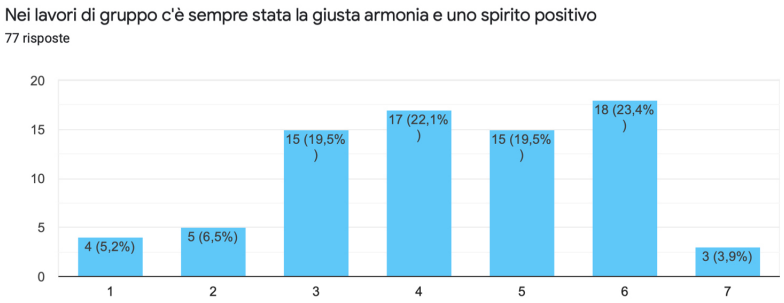

Figura 2.2. Il lavoro di gruppo (studenti), seconda domanda

3. *Preferisco/cerco di mettermi in gruppo con persone che conosco e so come lavorano*

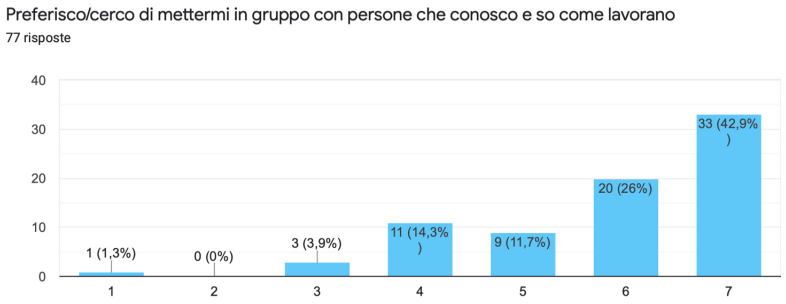

Figura 2.3. Il lavoro di gruppo (studenti), terza domanda

4. *Lavorare in gruppo mi ha spesso rallentato e/o portato via più tempo del previsto*

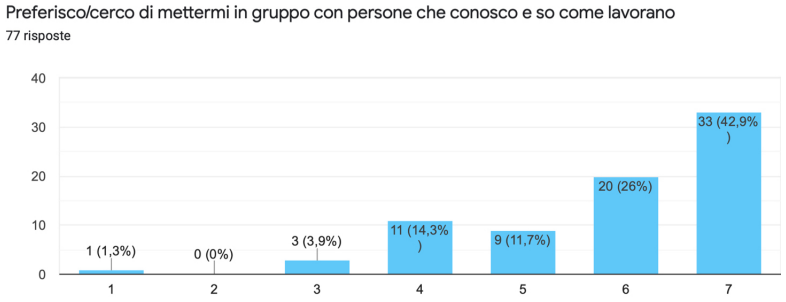

Figura 2.4. Il lavoro di gruppo (studenti), quarta domanda

5. *Capita spesso che non tutti i membri del gruppo apportino il proprio contributo*

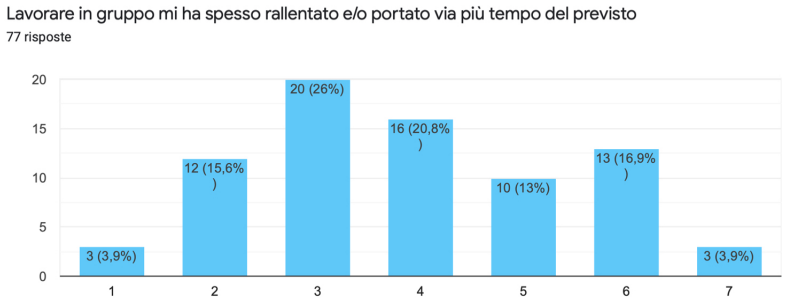

Figura 2.5. Il lavoro di gruppo (studenti), quinta domanda

6. *Le persone hanno obiettivi, tempi e interessi diversi: questo ritarda il lavoro di gruppo e/o penalizza il clima e l'armonia di gruppo*

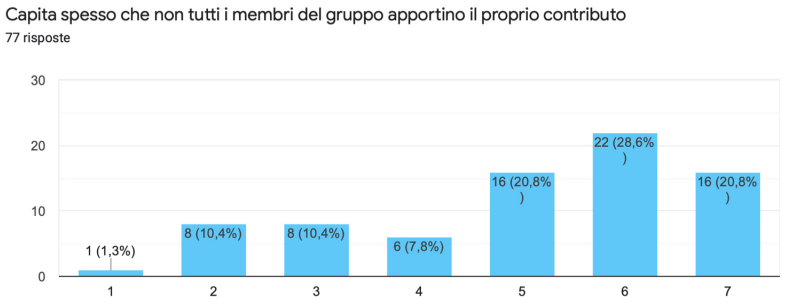

Figura 2.6. Il lavoro di gruppo (studenti), sesta domanda

7. *È più impegnativo lavorare con persone che provengono da altre università e/o hanno diversi background*

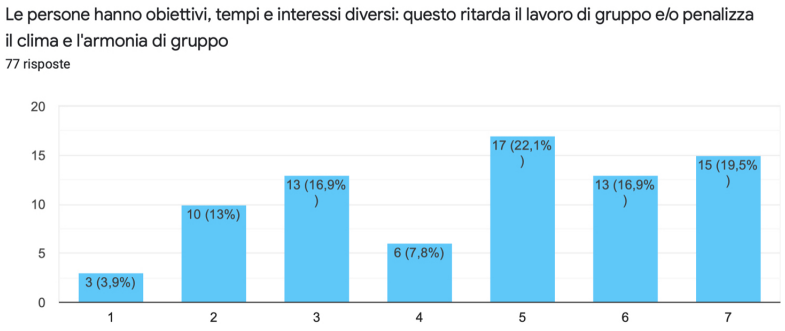

Figura 2.7. Il lavoro di gruppo (studenti), settima domanda

8. *Per lavorare bene in gruppo è fondamentale rispettare opinioni degli altri anche se non sono d'accordo*

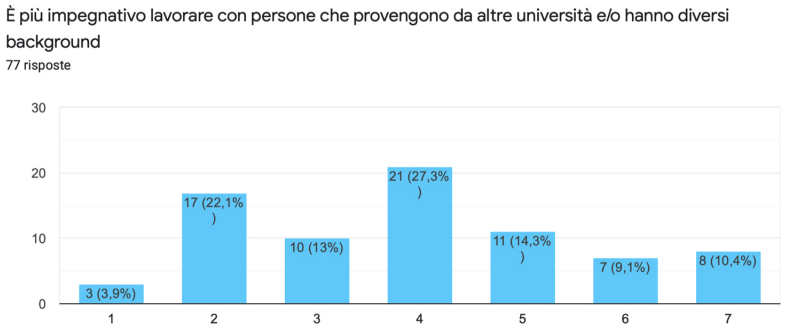

Figura 2.8. Il lavoro di gruppo (studenti), ottava domanda

9. *All'università ci hanno insegnato poco su come lavorare efficacemente in gruppo, come pianificare e realizzare un progetto*

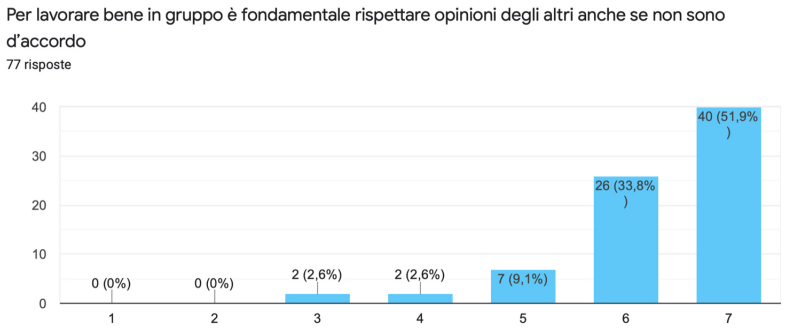

Figura 2.9. Il lavoro di gruppo (studenti), nona domanda

Il lavoro di gruppo (lavoratori) In questa sezione, l'obiettivo è quello di cercare di comprendere quale sia la situazione attuale per i lavoratori in ambito ingegneristico: se si trovino a proprio agio o fatichino a lavorare in gruppo, se questo avviene in maniera efficace e così via. Le domande proposte sono le seguenti:

1. *Mi trovo sempre perfettamente a mio agio a lavorare in gruppo*

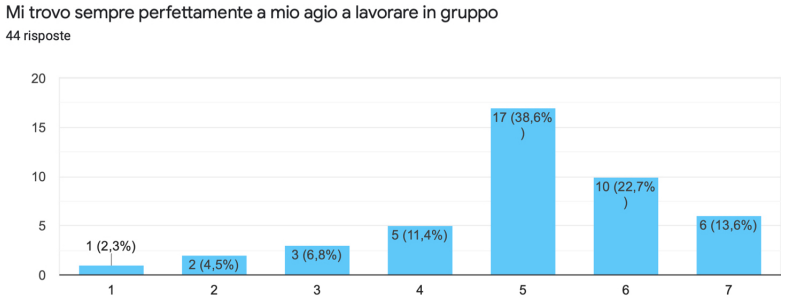

Figura 2.10. Il lavoro di gruppo (lavoratori), prima domanda

2. *Nei miei team c'è sempre stata la giusta armonia e uno spirito positivo/proattivo*

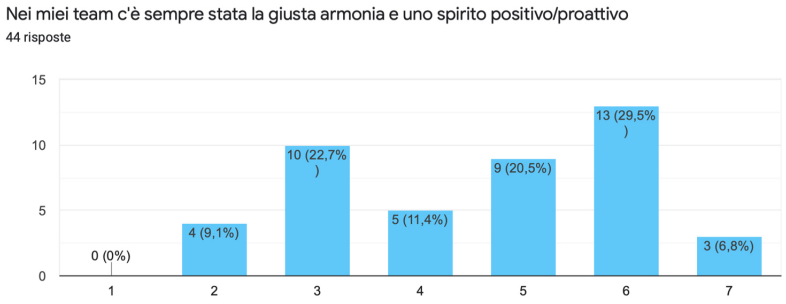

Figura 2.11. Il lavoro di gruppo (lavoratori), seconda domanda

3. *Preferisco essere in team con persone che conosco bene e so come lavorano*

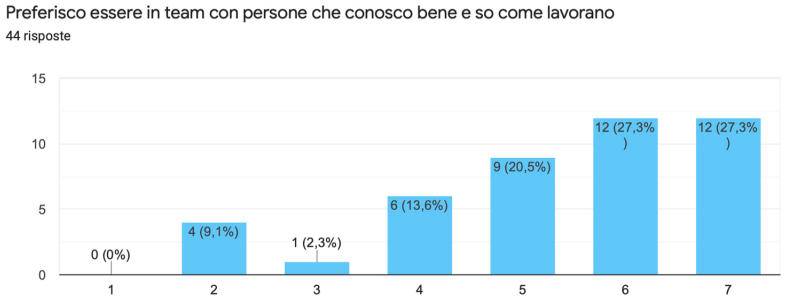

Figura 2.12. Il lavoro di gruppo (lavoratori), terza domanda

4. *Lavorare in team mi ha spesso rallentato e/o portato via più tempo del previsto*

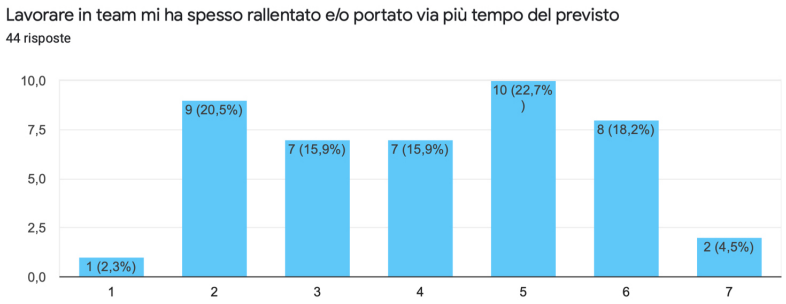

Figura 2.13. Il lavoro di gruppo (lavoratori), quarta domanda

5. *Le persone hanno tempi diversi, questo ritarda il lavoro di gruppo e/o penalizza il clima e l'armonia di gruppo*

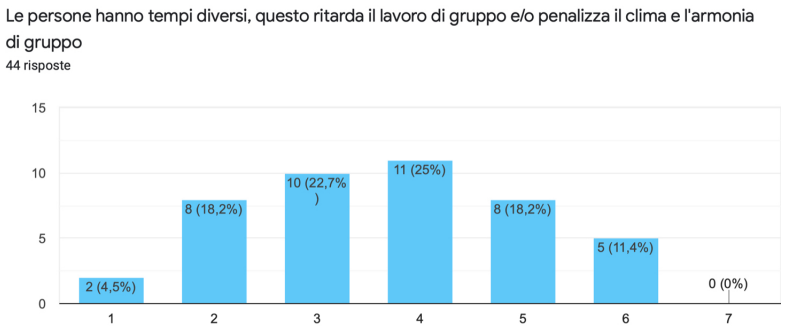

Figura 2.14. Il lavoro di gruppo (lavoratori), quinta domanda

Post studi (solo lavoratori) In questa sezione, l'obiettivo è cercare di comprendere quale sia l'opinione delle persone che hanno accumulato esperienza lavorativa e hanno la possibilità di riflettere sul proprio percorso universitario passato. Inoltre, si vuole capire se le aziende in cui lavorano investono o meno sulla formazione dei propri dipendente da un punto di vista relazionale. Le domande proposte sono le seguenti:

1. *All'università ci hanno insegnato poco su come interagire, lavorare efficacemente in gruppo, come pianificare e realizzare un progetto*

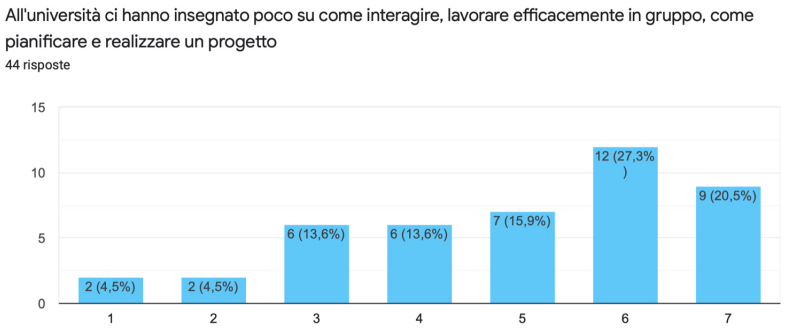

Figura 2.15. Post studi, prima domanda

2. *Il problem solving è l'unica Soft Skill "insegnata" e sviluppata all'università*

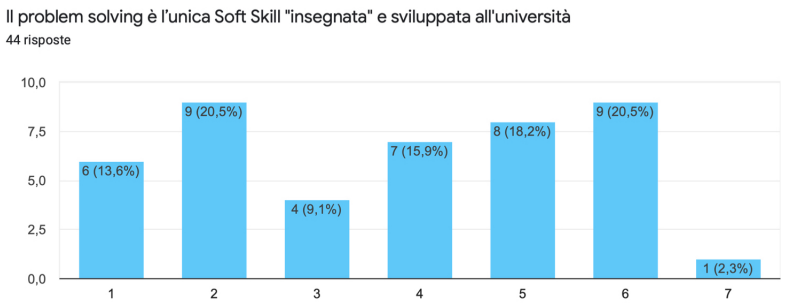

Figura 2.16. Post studi, seconda domanda

3. *Avrei voluto sviluppare all'università le Soft Skill necessarie oggi a lavoro, sia a livello teorico sia a livello pratico*

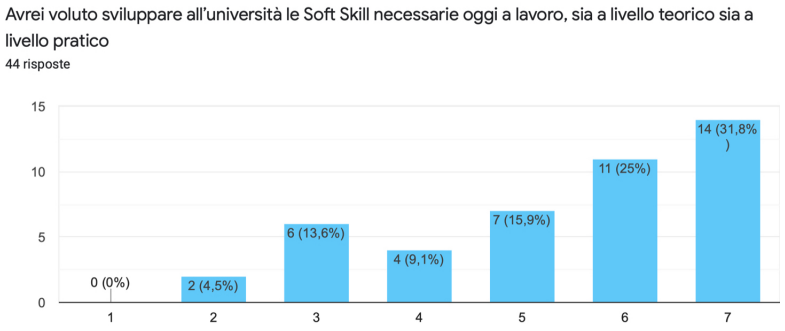

Figura 2.17. Post studi, terza domanda

4. *Quando un neolaureato in ingegneria si affaccia al mondo del lavoro, incontra difficoltà nel sapersi relazionare efficacemente*

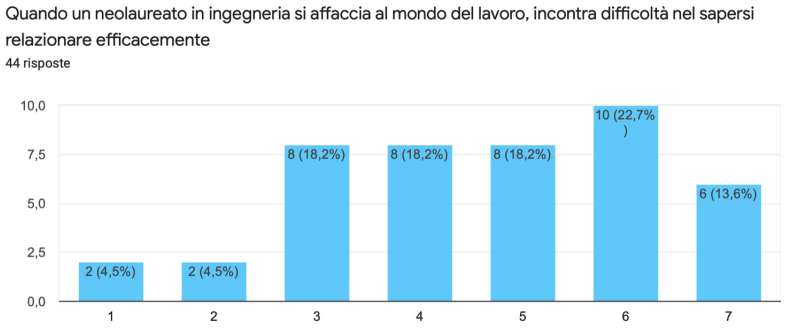

Figura 2.18. Post studi, quarta domanda

5. *Gli ingegneri sono molto competenti in ambito tecnico, carenti invece dal punto di vista delle Soft Skill*

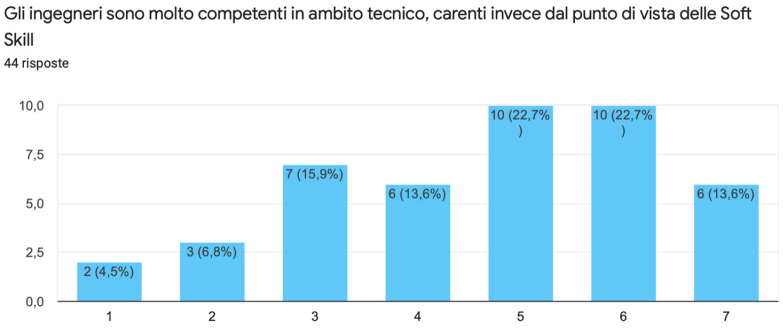

Figura 2.19. Post studi, quinta domanda

6. *Il percorso formativo e professionale è importante, ma puntare sulle proprie soft-skill e riuscire a far emergere i lati migliori del proprio carattere è fondamentale per una buona riuscita di un colloquio lavorativo*

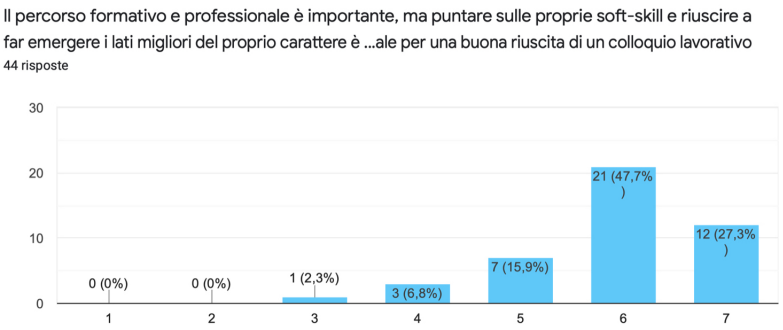

Figura 2.20. Post studi, sesta domanda

7. *La mia azienda investe molte risorse sulla formazione dei dipendenti dal punto di vista delle soft skills (ad esempio sulla comunicazione, leadership, team working, ecc.)*

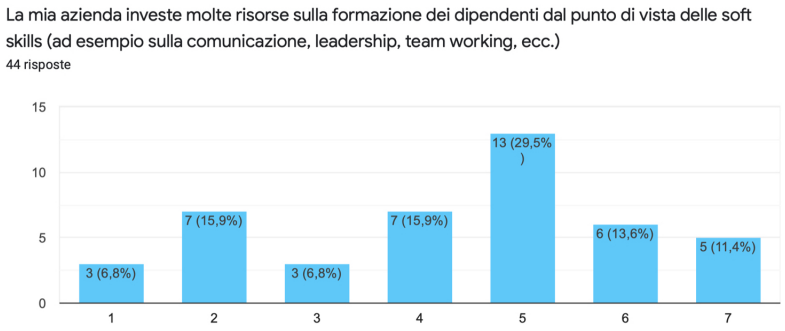

Figura 2.21. Post studi, settima domanda

Hard Skills vs Soft Skills In questa sezione, l'obiettivo è cercare di comprendere qual è l'opinione delle persone riguardo alle competenze tecniche e trasversali. Le domande proposte sono le seguenti:

1. *Quando devo fare presentazioni e/o discorsi tecnici in pubblico, provo disagio*

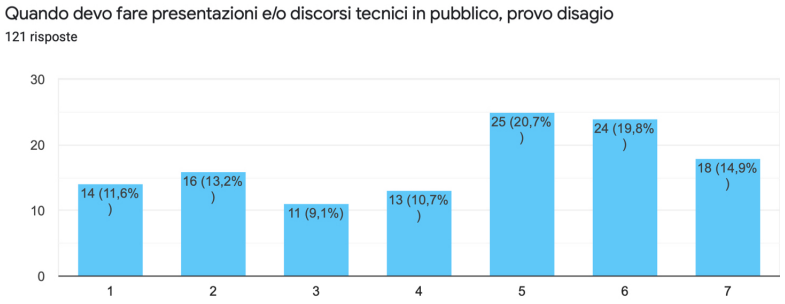

Figura 2.22. Hard Skills vs Soft Skills, prima domanda

2. *L'università è troppo orientata verso l'apprendimento delle hard skill, trascurando lo sviluppo delle soft skill necessarie nel mondo del lavoro*

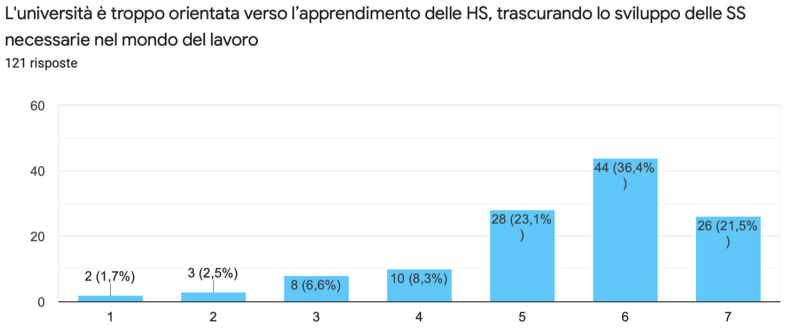

Figura 2.23. Hard Skills vs Soft Skills, seconda domanda

3. *Gli insegnanti sono molto competenti nel proprio ambito tecnico, ma spesso non entusiasmano e non fanno appassionare alla materia*

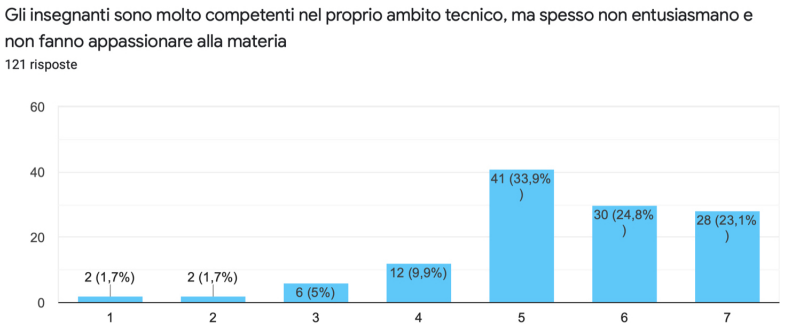

Figura 2.24. Hard Skills vs Soft Skills, terza domanda

4. *Al giorno d'oggi il lavoro degli ingegneri è particolarmente collaborativo ed è incentrato su progetti/team di lavoro, quindi diventa fondamentale sapersi relazionare efficacemente*

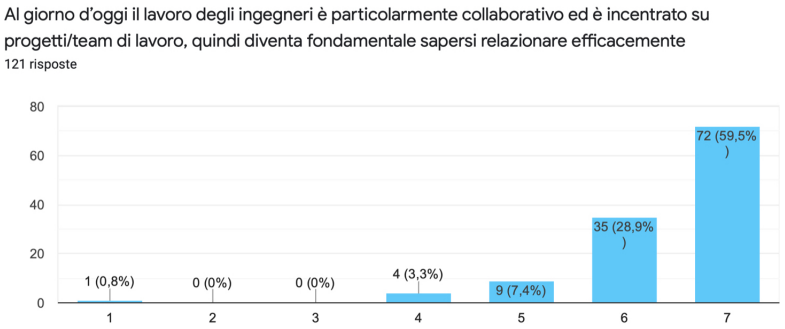

Figura 2.25. Hard Skills vs Soft Skills, quarta domanda

5. *Sono più importanti le Soft Skills o le Hard Skills?*

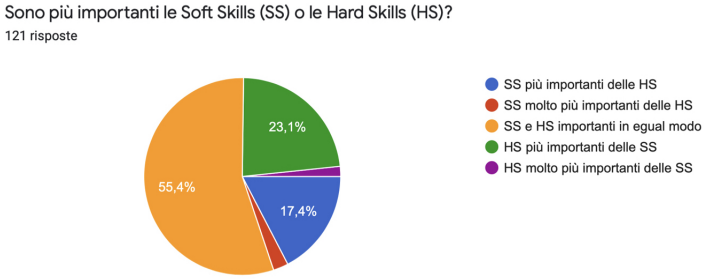

Figura 2.26. Hard Skills vs Soft Skills, quinta domanda

6. *Le soft skills non sono un talento con cui una persona nasce, bensì delle capacità importanti che possono essere imparate e sviluppate nel tempo*

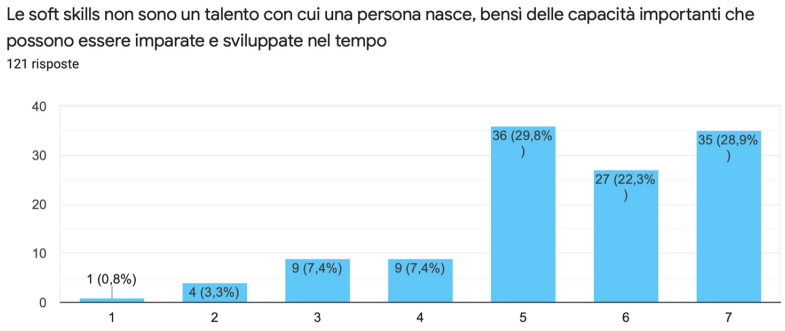

Figura 2.27. Hard Skills vs Soft Skills, sesta domanda

7. *Le hard skill ti permettono di qualificarti per un posto di lavoro, ma saranno le soft skill a determinare la crescita della tua carriera*

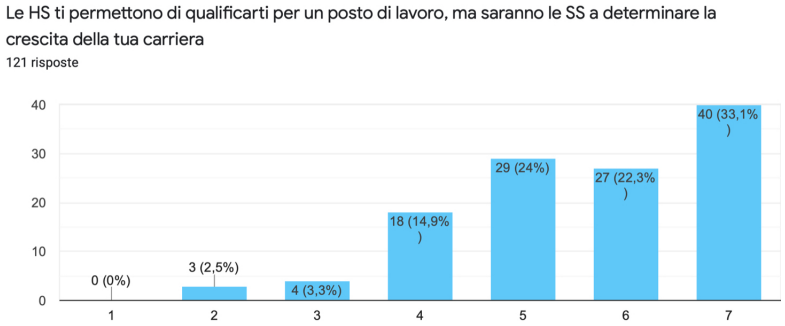

Figura 2.28. Hard Skills vs Soft Skills, settima domanda

Il futuro è ora In questa sezione, l'obiettivo è cercare di comprendere se le persone desiderino o meno un cambiamento del percorso universitario in termini di miglioramento e focalizzazione sullo sviluppo delle competenze relazionali. Le domande proposte sono le seguenti:

1. *Da quando sarebbe opportuno cominciare ad insegnare e far sviluppare le Soft Skill?*

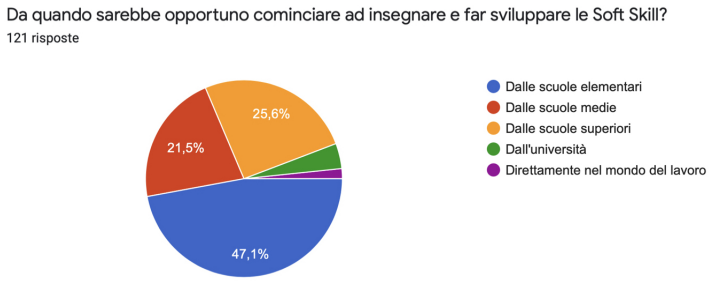

Figura 2.29. Il futuro è ora, prima domanda

2. *Il miglior modo per apprendere le soft skill è acquisirne PRIMA i concetti teorici di base e POI metterle in pratica nella vita quotidiana*

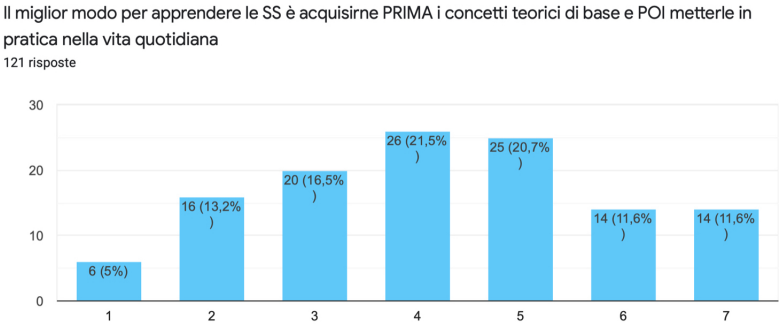

Figura 2.30. Il futuro è ora, seconda domanda

3. *Il miglior modo per apprendere le soft skill è metterle DIRETTAMENTE in pratica e acquisire esperienza man mano*

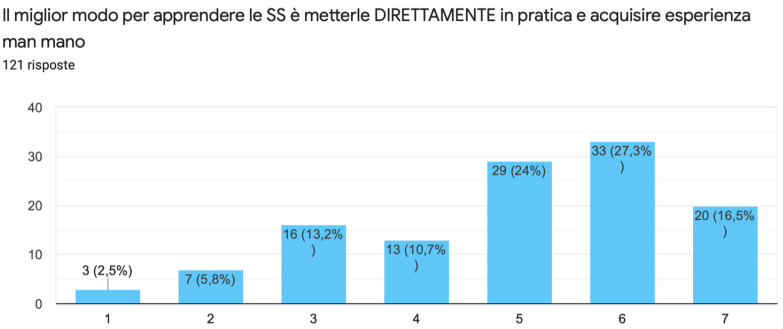

Figura 2.31. Il futuro è ora, terza domanda

4. *VORREI/AVREI VOLUTO che ci fossero corsi su misura sullo sviluppo delle Soft Skill*

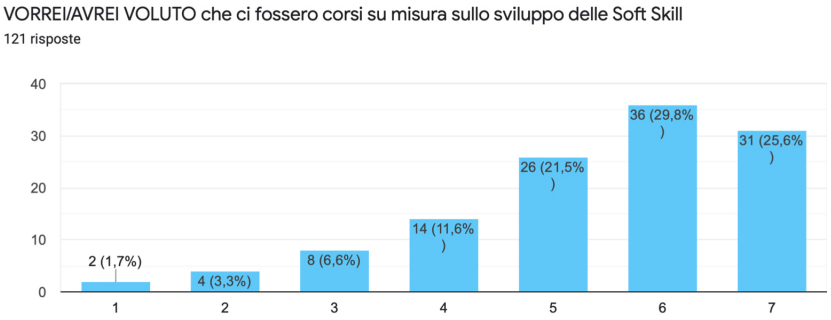

Figura 2.32. Il futuro è ora, quarta domanda

5. *È NECESSARIO che ci vengano creati il prima possibile corsi dedicati allo sviluppo delle Soft Skill*

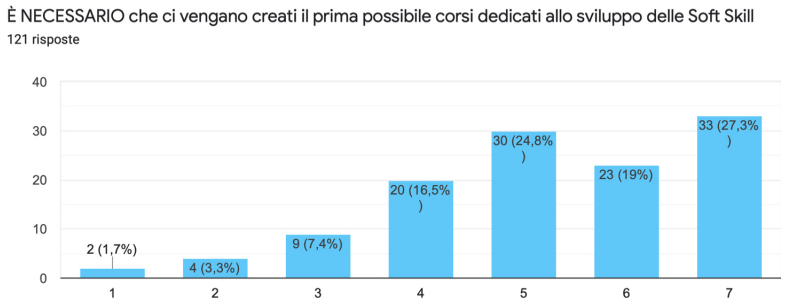

Figura 2.33. Il futuro è ora, quinta domanda

6. *Sarebbe meglio impiegare ore dedicate all'insegnamento delle SS oppure cambiare modo di insegnare le hard skill (sviluppando in contemporanea le soft skill)?*

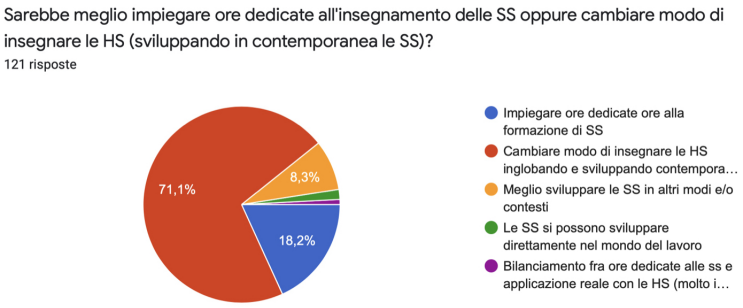

Figura 2.34. Il futuro è ora, sesta domanda

L'empatia In questa sezione, l'obiettivo è analizzare quanto le persone ritengano di riuscire ad entrare in empatia con se stesse e con il prossimo, notare se vi è una buona consapevolezza degli elementi caratterizzanti l'empatia. Le domande proposte sono le seguenti:

1. *Mi capita spesso di provare del disagio, di non riuscire a riconoscerlo bene e di non capire bene a cosa sia dovuto*

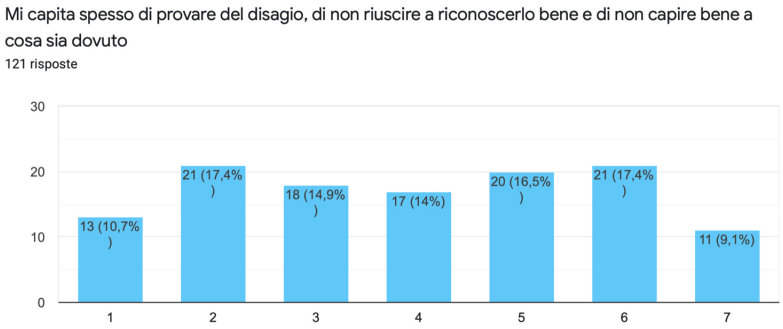

Figura 2.35. L'empatia, prima domanda

2. *Sono perfettamente in grado di riconoscere i miei bisogni e i miei sentimenti presenti nel qui e ora*

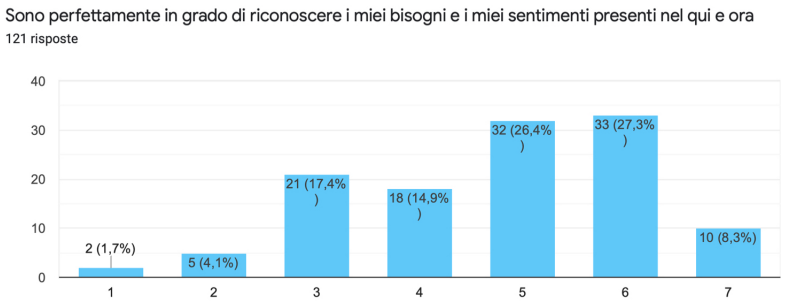

Figura 2.36. L'empatia, seconda domanda

3. *Riesco sempre a comunicare, alle persone con cui interagisco, e far valere in maniera non aggressiva i miei bisogni e i miei sentimenti*

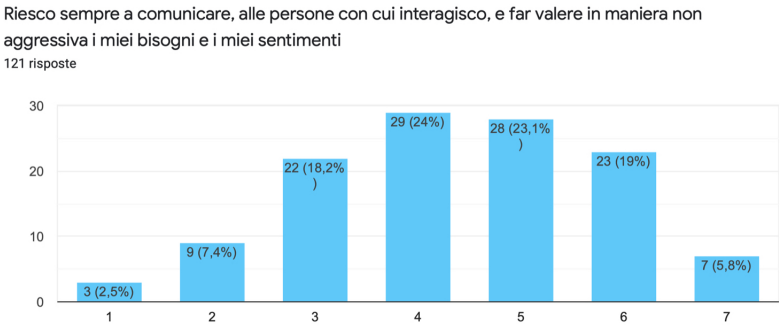

Figura 2.37. L'empatia, terza domanda

4. *Sono perfettamente in grado di riconoscere le emozioni, i bisogni e i sentimenti delle persone con le quali interagisco*

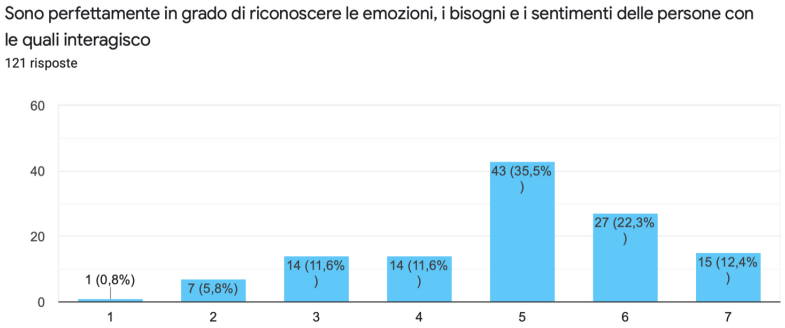

Figura 2.38. L'empatia, quarta domanda

5. *Riesco facilmente a concentrarmi su emozioni, bisogni e sentimenti presenti nel qui e ora (in me e nel prossimo) per agire in maniera tale da creare un ambiente non conflittuale sereno e armonico*

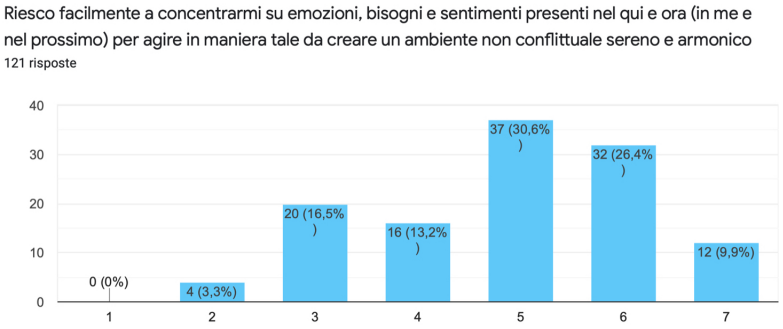

Figura 2.39. L'empatia, quinta domanda

6. *Riesco a dare il meglio di me quando c'è un clima sereno in un ambiente di lavoro*

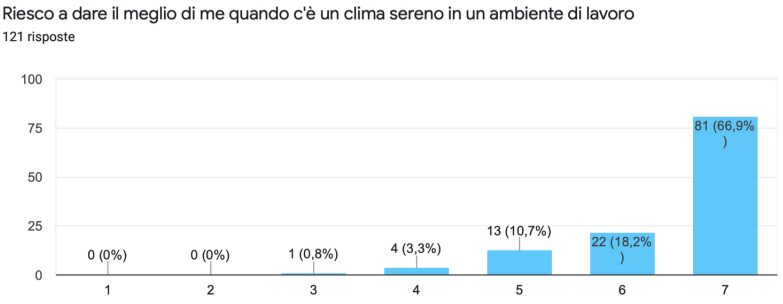

Figura 2.40. L'empatia, sesta domanda

7. *Entrare in empatia con gli altri membri del gruppo è indispensabile per creare un ambiente di lavoro sereno e in armonia*

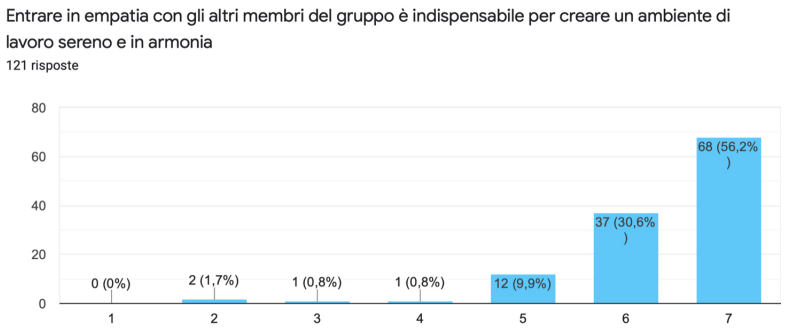

Figura 2.41. L'empatia, settima domanda

8. *Sviluppare la capacità di ascoltare è fondamentale per entrare in sintonia con i colleghi e raggiungere l'armonia di gruppo*

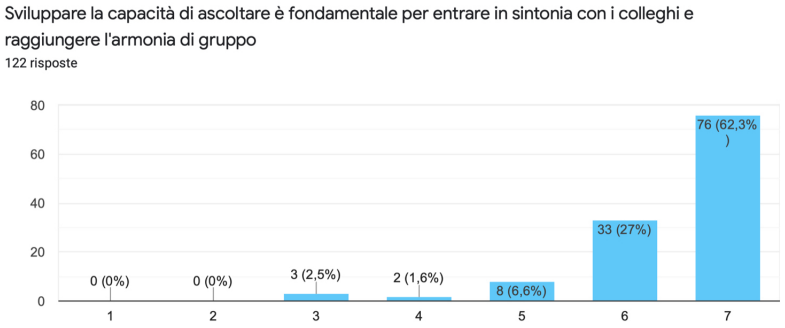

Figura 2.42. L'empatia, ottava domanda

Le tue idee contano In quest'ultima sezione, l'obiettivo è raccogliere il maggior numero di feedback aperti dalle persone per poi analizzarli e trarre delle conclusioni appropriate anche in base alle risposte fornite precedentemente nelle domande chiuse. Le domande proposte sono le seguenti:

1. *Come pensi che sia meglio sviluppare l'empatia e le soft skill? Vorresti ci fossero dei corsi su misura dedicati allo sviluppo di queste caratteristiche? Vorresti che venissero sviluppate durante l'apprendimento delle hard skills? Indica qui tutto quello che ti viene in mente!*
   In quest'ultima sezione è stata data la possibilità agli utenti di inserire la propria opinione e i propri feedback tramite un'area di testo

2.1.2 I risultati del sondaggio

Premessa È importante sottolineare che questo sondaggio è stato creato dal sottoscritto senza aver effettuato uno studio psicologico di come l'ordine delle domande

e la loro formulazione possa influire sui risultati, né del fatto che l'Italia si trova in un periodo particolare della storia e in una situazione anomala per via del COVID-19.

Sono state ottenute 121 risposte totali: 77 di queste da studenti e 44 da lavoratori, ovvero ex studenti.
Per facilitare l'analisi e la comprensione dei risultati, si è considerato un livello "alto" - corrispondente a un segnale di accordo - le risposte dal numero 5 al numero 7; viceversa si è considerato livello "basso" - corrispondente a un segnale di disaccordo - le risposte dal numero 1 al numero 3; le risposte con numero 4 sono state considerate, invece, come un valore neutro/medio.

Il lavoro in gruppo (studenti)

I risultati indicano che circa 3/4 delle persone si è trovata a proprio agio a lavorare in gruppo durante il percorso universitario: questo è un risultato che potrebbe essere un po' fuorviante se si considera che quasi tutte le persone preferiscono e hanno cercato di mettersi in gruppo con persone che già conoscono e con cui si sono trovate bene in precedenza. Circa 1/3 delle persone ritiene che il lavoro in gruppo lo abbia rallentato: anche in questo caso, se si tiene conto del precedente commento, si può ben comprendere che si tratta di una percentuale molto alta, che potrebbe aumentare notevolmente qualora i gruppi fossero sempre casualmente generati. Un risultato, che è da prendere assolutamente in considerazione ed è allarmante, è che il 70% delle persone ha indicato che sovente non tutti i membri apportano il proprio contributo: non di rado, infatti, su un gruppo di tre o quattro persone, succede che almeno una o due persone non portino avanti il progetto di gruppo, lasciando così l'intero lavoro alle restanti persone; allo stesso tempo, quelli che la pensano in maniera opposta sono solo il 22%. Più della metà delle persone, circa il 59%, ritiene che gli obiettivi e i modi di lavoro di altre persone completamente differenti dai propri penalizzi un gruppo, solamente il 34% circa ha espresso il parere opposto; anche in questo caso si tratta di un dato preoccupante perché il tipo di malessere generato da questa condizione non genera di certo un clima favorevole al lavoro armonioso di gruppo. Circa 1/3 delle persone ha indicato come più dispendioso il lavoro con persone che possiedono un background differente dal proprio: anche qui si tratta di un dato rilevante per lo stesso motivo precedentemente citato; in particolare, quelli che la pensano contrariamente sono poco più, ovvero il 39%. Quasi tutti sono d'accordo, infine, sul fatto che sia importante rispettare le opinioni degli altri anche se non si è d'accordo: facile da dirsi, ma sicuramente più difficile da fare quando magari ci sono decisioni importanti da prendere per scegliere strade da esplorare e seguire durante il cammino di un progetto e/o percorso lavorativo prolungato.

Il lavoro in gruppo (lavoratori)

È molto interessante notare come i dati, ottenuti da persone già impegnate nel mondo del lavoro, siano del tutto simili a quelli riscontrati dagli studenti. Anche in questo caso, 3/4 delle persone si trovano a proprio agio durante il lavoro in gruppo. Anche se il risultato ottenuto è molto simile al precedente, è importante sottolineare come i contesti siano completamente differenti: in questo caso, infatti, i lavoratori non possono scegliere (perlomeno in generale) le persone con cui andare a formare il proprio team di lavoro. Si tratta pertanto di un dato molto migliore e più attendibile, che rispecchia almeno due situazioni possibili: quando ci si integra in un contesto lavorativo, si migliora e si impara a collaborare efficacemente in gruppo, oppure gli obiettivi e le pressioni lavorativi fanno in modo che ognuno sia focalizzato maggiormente sul risultato, evitando così di creare malesseri nel gruppo (allo stesso tempo, le relazioni che si creano potrebbero però essere meno intime e solamente rivolte all'obiettivo del lavoro prestabilito).
Si susseguono invece alcuni dati più ragguardevoli. Quasi 1/3 delle persone ritiene che non ci sia sempre stata la giusta armonia e uno spirito positivo, mentre il 57% delle persone la pensa al contrario. Anche in questo caso, una percentuale molto alta delle persone (75%) preferisce stare in gruppo con persone di cui conoscono il modo di lavoare; forse questo dato non è del tutto positivo, perché lascia in dubbio la presupposizione che potrebbe essere più complicato trovarsi a collaborare con persone sconosciute, con background e modi di fare completamente differenti dai nostri: questo potrebbe indicare ancora che vi è un notevole margine di miglioramento nel riuscire veramente a relazionarsi efficacemente per trovare la giusta armonia durante il lavoro di gruppo. Questo dato è supportato anche dal 30% delle persone che ritengono che persone con tempi e obiettivi diversi possono penalizzare il gruppo (contro solamente il 45%).
Un dato invece molto più allarmante, soprattutto se si considera che concerne il contesto lavorativo, è rappresentato dal fatto che quasi la metà delle persone ritiene che il lavoro di gruppo abbia rallentato e/o portato più tempo del previsto, contro solamente il 39% circa.

Post studi

Circa il 64% dei lavoratori ritiene che durante il percorso di studi universitari sia stato insegnato poco su come interagire, collaborare efficacemente in gruppo, pianificare e realizzare un progetto; a supporto di questo dato, il 41% delle persone ritiene che venga "insegnata" e sviluppata soltanto la competenza del problem solving, lasciando in disaccordo solamente il 43%. Allo stesso tempo, quasi 3/4 delle persone avrebbe voluto imparare e sviluppare le competenze trasversali durante la carriera universitaria: questi sono dati che fanno riflettere e dirigono il focus verso l'ideazione di nuove soluzioni. Più di metà delle persone ritiene che i neolaureati ingegneri abbiano difficoltà nel sapersi relazionare efficacemente, solamente il 27%

circa pensa il contrario. In aggiunta, quasi il 60% delle persone ritiene che gli ingegneri siano particolarmente competenti nel proprio ambito ma abbiano carenza nello sviluppo di competenze trasversali.

HS vs SS

Più di metà delle persone ha provato disagio nel momento in cui si è ritrovata a dover effettuare una presentazione in pubblico: dato allarmante e non prende nemmeno in considerazione la percentuale di persone che, per via del forte disagio, potrebbe avere addirittura rinunciato a tenere proprio la presentazione.
Quasi tutte le persone ritengono che il percorso di studi universitari sia particolarmente sbilanciato verso l'apprendimento delle competenze tecniche e che, allo stesso tempo, i docenti siano le prime figure a esprimere grande competenza tecnica ma poca abilità di coinvolgimento e di trasmissione d'entusiasmo nel divulgare conoscenza e passione nei confronti della materia.
Quasi tutti concordano che al giorno d'oggi il lavoro degli ingegneri sia prevalentemente in gruppo e, alla luce di questo, più di metà delle persone ritiene che le competenze tecniche siano tanto importanti quanto quelle trasversali: pochissime persone ritengono che le competenze tecniche siano molto più importanti di quelle trasversali. Quasi tutti sono d'accordo che le competenze trasversali non siano solamente un qualcosa per la quale si possiede una naturale predisposizione, ma sono competenze che possono essere apprese e sviluppate nel corso del tempo. Circa l'80% delle persone ritiene che siano proprio le competenze trasversali a determinare la crescita della propria carriera.

Nasce spontaneo uno spunto di riflessione importante al quale difficilmente si può dare una risposta univoca:

è giusto pensare che l'università debba formare gli studenti solo per saper lavorare da soli di fronte al proprio computer, oppure potrebbe e dovrebbe completare professionalmente la figura dell'ingegnere affinché abbia tutti gli strumenti per scegliere che tipo di carriera percorrere?

Dal punto di vista dello studio proposto in questa tesi, si vuole incentivare l'idea che il percorso di studi universitari possa creare una figura di ingegnere neolaureato professionalmente preparata e completa per entrare nel mondo del lavoro.

Il futuro è ora

Circa la metà delle persone ritiene che bisognerebbe iniziare a sviluppare le competenze trasversali già dalle scuole elementari. Se ci si riflette bene, fino alla quinta superiore questo avviene, più o meno a seconda del contesto in cui ci si ritrova, ma accade attraverso il metodo della punizione, che in realtà spesso porta a un

risultato differente da quello desiderato: tant'è vero che molti blocchi delle persone si formano durante questi anni in cui provano forti disagi e non riescono a superarli. Più di 3/4 delle persone avrebbe voluto dei corsi per sviluppare le competenze trasversali all'università e il 71% ritiene sia necessario creare corsi nuovi (solamente circa il 12% non la pensa in questo modo), il 44% ritiene sia possibile analizzare prima gli aspetti teorici e il 68% pensa sia fondamentale mettere direttamente in pratica. Il 71% delle persone ritiene che il metodo più efficace sia inglobare le competenze trasversali in quelle tecniche.

Empatia

Quasi la metà delle persone ritiene di non riuscire a riconoscere pienamente il disagio che prova, quando compare, e di capire a cosa sia dovuto. Il 62% ritiene di essere in grado di riconoscere sentimenti e bisogni nel qui e ora ma solamente meno del 50% dichiara di essere in grado di riuscire sempre a farli valere in maniera non aggressiva: un dato allarmante se si considera anche che magari molte persone potrebbero non avere il coraggio di sbilanciarsi in una domanda posta in questo modo, considerando anche il fatto che il questionario è anonimo, ma non è stato specificato al momento della presentazione. Il 70% crede di essere in grado di riconoscere gli stessi nelle altre persone e il 67% di essere in grado di concentrarsi su di essi per risolvere le situazioni.
Quasi tutti, infine, concordano sul fatto che in un ambiente in armonia riescono a dare il meglio di sé e che ascoltare attivamente per entrare in empatia con il prossimo è fondamentale per instaurare un clima lavorativo sereno.

Le tue idee contano

In questa sezione, le persone hanno potuto lasciare i propri feedback in formato aperto. Probabilmente questa sezione è la più importante perché i commenti ricevuti riassumono e completano alla perfezione i pensieri delle persone. Si riporta di seguito un'estrema sintesi di tutti i feedback ricevuti.

Innanzitutto, c'è chi avverte un forte scetticismo rispetto a questi temi, soprattutto in un contesto come quello del Politecnico. Qualcuno ha incontrato tanti colleghi che preferiscono isolarsi e lavorare individualmente e sostiene che forse all'università nasca dell'invidia verso questi colleghi per la loro bravura nelle competenze tecniche, senza pensare al fatto che sono carenti nelle importanti competenze relazionali. Alcuni sottolineano come in ambito universitario, specialmente nei corsi di ingegneria, empatia e competenze trasversali non vengono sviluppate come dovrebbero. La loro importanza viene completamente ignorata, tanto che lavorare in gruppo spesso diventa stressante, soprattutto in questo ultimo periodo di videochiamate e messaggi, con una scarsa possibilità di interazione, rendendo

quasi impossibile comprendere i propri compagni e le loro emozioni soprattutto se provenienti da culture diverse. Inoltre le aziende (soprattutto quelle medio piccole) spesso non le giudicano importanti, ma guardano solamente alla capacità tecnica del candidato. Per questo si ritene opportuno che per prime vengano sensibilizzate le aziende sull'importanza delle dell'empatia e delle competenze trasversali, dovrebbero investire di più per migliorare da questo punto di vista ogni singolo loro dipendente, indistintamente dal ruolo ricoperto. In questo modo si arriverebbe anche a richiederle sempre di più pure in ambito universitario e, a quel punto, dei corsi mirati allo sviluppo di queste caratteristiche diventerebbero fondamentali.
Alcuni vorrebbero una sensibilizzazione maggiore negli istituti di istruzione sull'importanza delle competenze trasversali, in quanto siamo animali sociali e oggi più che mai il lavoro richiede grande interazione e cooperazione. A volte ci si ritrova persino a terminare l'iter di selezione per candidati che presentano lacune gravi nel sapersi relazionare piuttosto che nelle competenze tecniche poiché, per esperienza personale e professionale, si ritene che chi ha una buona base in termine di atteggiamento e capacità relazionale può sempre essere in grado di colmare il gap tecnico, mentre purtroppo, molto più spesso di quanto si possa pensare, non è vero il contrario.

È particolarmente sentita la mancanza di suggerimenti su come gestire umanamente i tanti progetti di gruppo, come gestire le attività e collaborare efficacemente. Tra le altre cose, avere a disposizione un'infarinatura su come presentare efficacemente un progetto sarebbe fondamentale.
Alcune persone ritengono necessario che vengano creati corsi su misura (per qualcuno magari anche brevi e con poche nozioni e spunti di riflessione, per qualcuno ne basterebbe anche solo uno completo per tutto) per lo sviluppo dell'empatia e delle competenze trasversali, come ad esempio lo è stato Engineering Empathy per lo sviluppo dell'empatia e dell'ascolto, e dare spazio in altre materie tecniche per poter mettere in pratica quanto appreso nei corsi appositamente dedicati. Molti ritengono che il miglior modo per insegnare e sviluppare le competenze trasversali sia metterle in pratica senza scinderle da quelle tecniche: creare quindi progetti di gruppo in cui si vanno effettivamente a sviluppare le competenze trasversali.
Nella maggior parte dei feedback, si arriva a una conclusione comune: riuscire a inglobare e rendere compatibili lo sviluppo delle competenze tecniche contemporaneamente a quelle trasversali, in quanto le une vanno a completare le altre. Alcuni sottolineano che però questo affiancamento non deve portare a un soffocamento delle competenze trasversali da parte di quelle tecniche.
Alcuni propongono di creare corsi su misura in cui è possibile creare ambienti protetti in cui andare a imparare e mettere in pratica le competenze, per poi spostarsi su corso più tecnici e meno protetti per mettere definitivamente alla prova quanto appreso, in maniera progressiva anche con persone di background completamente

differenti. Qualcuno precisa l'importanza di avere dei corsi in cui porre le fondamenta perché praticare senza la giusta strategia può essere inefficace e contro produttivo.
Alcuni studenti di Engineering Empathy ritengono che il lavoro svolto sia stato estremamente utili per la crescita di sé, per lo sviluppo dell'autoconsapevolezza, e per imparare a comprendere il punto di vista altrui, imparando ad ascoltare realmente. Sottolineano anche l'importanza di partecipare a un corso come Engineering Empathy all'inizio del percorso universitario e non al suo termine in maniera tale da avere anche l'opportunità di mettere in pratica quanto appreso negli altri lavori di gruppo. Potrebbe essere non un esame a scelta ma obbligatorio poiché sono una parte fondamentale del nostro lavoro sia personale che rivolto agli altri.
Si puntualizza l'importanza di prendersi e avere del tempo per elaborare e riflettere su questi argomenti e sulle situazioni vissute perché non sono cose che si imparano leggendo, è fondamentale metterle in pratica mettendosi in gioco, ricevere dei feedback e autoanalizzarsi. Qualcuno ha trovato per esempio molto interessante un approccio in Spagna dove a fine di un corso veniva dato da valutare il proprio lavoro in un gruppo e commentare anche quello dei compagni, da qui il richiamo all'importanza di ricevere feedback. È sorprendente cosa si può ottenere con un pizzico di empatia in più nelle nostre vite, sentirsi bene con se stessi e con gli altri non è da poco e dovrebbe essere la base per una buona formazione.
È essenziale sottolineare che, per arrivare a questi risultati, i docenti dovrebbero essere i primi a sviluppare una certa sensibilità su questi temi e sviluppare in prima persona alcune di queste competenze, come ad esempio il fatto di saper esporre in maniera efficace le lezioni.

Qualcuno propone un insegnamento non tradizionale: per esempio, giornate di team building, magari organizzate, a turno, dagli stessi che poi ci parteciperanno. Un altro strumento utile possono essere giochi collaborativi. È fondamentale riuscire a spostare il focus da "collabora perché ti serve" a "collabora perché ti piace". Alcuni consigli indirizzano verso crediti extra e presentazioni pubbliche, si potrebbero anche sostituire i laboratori da due punti con laboratori da 10 punti in cui bisogna fare progetti molto più complessi, che permettono quindi allo studente di sviluppare sia le competenze tecniche e sia quelle trasversali, in quanto finora i progetti di gruppo finora siano unicamente incentrati sulle competenze tecniche. Qualcuno nota come sarebbe anche solo più efficace dare più spazio e risalto alle presentazioni in pubblico. Inoltre, qualcuno ha partecipato a corsi su competenze trasversali in triennale in un'altra università, ma venivano svolti con persone in situazioni non reali. Vista la possibilità di lavorare in gruppi nei laboratori, sarebbe utile sapere come applicare queste conoscenze nel apprendimento delle competenze tecniche e non solo in teoria/simulazioni fittizie della pratica.

Qualcuno ha addirittura espresso timore e disagio nel solo pensieri di affrontare

un corso come Engineering Empathy nonostante lo desidererebbe, ma per paura e timidezza e preferisce evitare! A supporto di questo fatto, altri ritengono che sarebbe fondamentale un percorso di supporto costante con uno psicologo sempre presente.

Si riportano, infine, integralmente due feedback ricevuti, ritenuti di grande valore ed estrema rilevanza:

Prima di diventare uno studente d'ingegneria, per tanti anni ho fatto l'attore di teatro. Da estroverso, il mondo della recitazione mi ha sempre affascinato, e l'idea di poter regalare un sorriso collettivo a chi era in platea mi portava a dare sempre il massimo, con naturalezza. La compagnia teatrale per cui recitavo accettava tutti, e parte degli studenti non si presentava alle audizioni per passione, ma per cercare di superare la propria timidezza. E facevano bene, perché quell'ambiente caloroso e collaborativo in breve tempo li trasformava, permettendo passi da gigante che, con l'approccio tecnico-punitivo della scuola, non avrebbero mai potuto raggiungere. Ad ogni fine corso, esibirsi davanti ad un pubblico non era più un peso, ma voglia di mettersi in gioco e grinta di dimostrare quanto duramente ci si era impegnati durante un anno intero per costruire anche solo due ore di spettacolo. Racconto questo aneddoto perché posso confermare con assoluta certezza che l'aver sviluppato delle soft skills, in questo caso durante il percorso da attore, mi ha aiutato tanto durante la carriera universitaria. Ad esempio, mi ha permesso in più occasioni di rimanere lucido durante gli orali, di non sentirmi in imbarazzo a parlare con un professore, di presentare agevolmente un progetto senza preoccuparmi di come avrei parlato, di ascoltare attivamente valorizzando le opinioni dei miei colleghi e allo stesso tempo di trovare il coraggio di affrontare con educazione quel collega che non lavora e che se lasciato a briglie sciolte potrebbe trascinare a fondo tutto il gruppo, compreso me. Per questo motivo sono convinto che avere uno o più corsi al Politecnico che premano (anche) su questi punti sia decisamente fondamentale. Negli anni ho visto tanti studenti preparatissimi non riuscire agli orali a dimostrare le proprie competenze perché spaventati dalla platea di studenti che li osservava durante l'esposizione. Per non parlare del fatto che sempre più studenti, anche per colpa della quarantena, faticano a trovare compagni di studio e perseguono la strada accademica da soli pensando che sia la scelta giusta. Tuttavia, non si può pensare di affrontare un percorso di studi in modo individuale se poi il mondo esterno (e del lavoro) è collettivo, e questo concetto sfugge a molti, preoccupati ormai quasi unicamente del voto e non curanti del fatto che non riuscire a lavorare attivamente e propositivamente in gruppo vuol dire alimentare delle lacune che, una volta arrivati in azienda, sono inammissibili e spesso difficilissime da colmare. Quindi, forza Engineering Empathy! Da parte mia il più sincero in bocca al lupo per questo nuovo progetto a colori, che possa essere di ispirazione a tanti studenti e professori per capire l'importanza dell'integrare nei corsi anche dei laboratori per le soft skills

e che, prima di diventare ingegneri, dobbiamo imparare ad essere umani.

Inizio dicendo che trovo interessantissimo questo sondaggio e molto attuale. Credo fortemente che "l'ingegnere medio" abbia delle enormi lacune nel rapportarsi con le persone, e sottovaluta questo problema. Dal mio punto di vista per la nostra carriera lavorativa(e non solo) è fondamentale sviluppare delle soft skills. Nella mia vita ho avuto la fortuna di avere famigliari che sono top manager in aziende private, grazie alle loro testimonianze ho capito quanto sia fondamentale la personalità e la leadership. Inoltre gioco a calcio a livello agonistico da quando ho 6 anni, e stare in questi ambienti, mi ha formato molto. Ho capito l'importanza di un gruppo, la difficoltà di creare "un unico punto di vista" tra persone profondamente diverse. Ho cambiato tante squadre, imparando a legare con persone che hanno un'età compresa fra i 18 e i 40. Onestamente non sono il tipico ingegnere "geniale", sono sempre riuscito a rimanere in tempo senza dare esami a settembre, ma non ho mai avuto una media alta. Credo però profondamente che fare queste esperienze mi abbia dato qualcosa in più a livello sociale e ho potuto constatarlo durante il tirocinio che ho svolto durante il terzo anno. Spero di essere d'aiuto con la mia testimonianza, a mio parere è un tema FONDAMENTALE che molti sottovalutano, le aziende sono fatte di persone e non solo di numeri.

2.1.3 Le interviste

Per migliorare la raccolta dati e informazioni, sono state effettuate nove interviste in aggiunta al sondaggio. Per semplicità di trattazione e per questioni di privacy, le persone intervistate verranno indicate con *Mister X*, indipendentemente dal genere sessuale.

Prima intervista

Mister X ha effettuato per svariati anni, circa dieci, calcoli strutturali, ovvero si è occupato di verificare virtualmente i componenti per prevedere il comportamento che avranno sul fisico in ambito, automotive, aerospace, trasporto ferroviario e aeronautica. Ha effettuato per anni questo tipo di lavoro tecnico, trovandosi a lavorare sempre all'interno di qualche team.
Mister X è ora un consulente di una grande azienda ingegneristica che mette a disposizione di altre aziende risorse che diventano come degli interni, perché danno accesso a tutti i tool e ai materiali riservati. Mister X è da dieci anni capo progetto di un'importante azienda automobilistica italiana: in particolare si occupa di tutto ciò che concerne i componenti interni delle vetture. Lavora pertanto su progetti ben definiti da portare a termine nei tempi e nei costi previsti dall'azienda. All'inizio, quando si occupava di calcoli strutturali, non eseguiva i propri compiti da solo ma collaborava all'interno di vari team, ora invece si ritrova a gestirli in prima persona:

ritiene che questo dipenda sia dall'esperienza acquisita sia dalla bravura.

Il suo progetto può iniziare con 5 persone e terminare con 50, è molto flessibile. A sua completa disposizione ha un team che si occupa di effettuare le valutazioni. Ritiene indispensabile sapersi relazionare, più il team è affiatato, più si effettua un importante lavoro di gruppo. Da soli non si va da nessuna parte: è fondamentale conoscere i punti deboli e forti di ognuno e sfruttarli al meglio, cercando di superare le problematiche. Per indole personale, si interessa sempre del lavoro svolto dai colleghi, non per totale mancanza di fiducia, bensì per poter essere certo che tutto sia svolto nella maniera ottimale. Ritiene indispensabili elementi come l'empatia e la fiducia ma, allo stesso tempo, preferisce sempre verificare con i propri occhi, quando la responsabilità è sua. Questo perché, a fronte di eventuali errori, diventa complicato per lui presentarsi a riunioni importanti, attorno a "tavoli pesanti", e giustificarsi dicendo "va bene ma non è colpa mia, perché un componente del team ha sbagliato": se anche non è stato lui ad effettuare quell'errore, è comunque chiamato a risponderne ed è pertanto fondamentale controllare qualsiasi cosa prima di ritrovarsi in situazioni di quel genere.

L'azienda ha un insieme di funzioni. In base all'avanzamento del progetto, Mister X può attingere ai gruppi di queste funzioni e deve occuparsi anche di contattare i fornitori di materiali per ricevere i pezzi necessari nei tempi e budget prestabiliti. Si interfaccia con tutto ciò che ha a che fare con l'intero progetto su cui deve lavorare. In teoria, dovrebbe rispondere del proprio operato a una persona ma in pratica non è così. Fa parte della funzione ingegneria che dovrebbe avere piena autonomia, ma lui, come capo progetto, si ritrova a confrontarsi con altri settori: ad esempio deve interfacciarsi con il responsabile piattaforma (il team atto a verificare l'arrivo dei componenti in produzione) che mette a disposizione un budget e con il quale condivide i tempi.

Alle riunioni si presentano sovente vari scenari.
A volte, c'è chi non è competente dal punto di vista tecnico e ha l'umiltà di lasciar proporre soluzioni a chi è più competente.
Altre volte, capita che persone poco competenti dal punto di vista tecnico propongano soluzioni in 2 secondi con ipotesi sbagliate, sembrando geniali all'istante. In queste occasioni, chi è davvero competente e sa che l'idea proposta non può funzionare, non deve aggredire, non deve rispondere con l'intenzione di dire all'altro che non capisce niente, in particolar modo (ma non solo) quando si ha di fronte persone con responsabilità di livello mondiale. Ha assistito a riunioni dove i confronti sono a muso duro, dove la cordialità non esiste e a volte sono addirittura saltate "sedie e tavolini", a seconda della gravità del problema. In questi casi è bene rispondere in maniera assertiva, cercando di verificare le soluzioni proposte ma

facendo comunque capire che, probabilmente, potrebbe non essere possibile mettere in pratica le idee proposte, sempre in maniera molto pacata e lasciando aperta un'esplicita verifica sul campo. Un conto è dire un no secco davanti a tantissime altre persone, un conto è far capire che si è compresa l'idea proposta e che si è disposti a verificarne la fattibilità. È fondamentale far capire in modo carino e non brusco all'interlocutore che sta dicendo una cosa che non funzionerà: magari dopo un'attenta analisi e verifica, gli si possono fornire anche slide sulla spiegazione del perché la soluzione non è fattibile. Agendo in questo modo, il discorso si conclude con toni pacifici ed è possibile proseguire tutti verso l'obiettivo; allo stesso tempo, non si portano avanti attriti perché ci si mostra propositivi: inoltre, una volta tirati fuori i dati, nessuno può smentirli. Il primo approccio è da evitare con qualsiasi persona, sia che faccia parte di un team che si gestisce, sia che si tratti di un diretto superiore. Per indole personale, cerca sempre di utilizzare il secondo approccio, anche se in passato ha erroneamente tentato anche il primo, ottenendo pochi risultati.

A volte capita che non si riesca a risolvere il problema magari perché c'è un blocco inevitabile (bloccante): ad esempio, il suo superiore potrebbe voler avere come obiettivo l'eliminazione di parte della rumorosità della vettura però allo stesso tempo non può mettere a disposizione un budget sufficiente. In questi casi si prova a mediare, ma se non c'è subito la soluzione può essere necessario effettuare un'escalation, in quanto c'è un bloccante importante che non permette di risolvere la situazione. È fondamentale non lasciare i problemi sulla scrivania, nel dimenticatoio; ognuno di essi va risolto il prima possibile perché comunque si ripresenta e rischia di causare un ritardo. Va risolto con i giusti mezzi, magari ottenendo un compromesso tra il budget e una soglia di rumore accettabile. Non può essere tutto perfetto perché altrimenti il prodotto verrebbe a costare cifre spropositate. Facendo escalation si espone il problema a un altro superiore di più alto livello, il quale chiama il suo capo e la persona che sta creando il disagio per tentare tutti assieme di sistemare la questione.
In tali situazioni ci si innervosisce, a volte non poco; ma ci sono due tipi di insoddisfazione: per il progetto e/o per problemi personali, di solito casi rari per i quali si è arrivati addirittura ad avvocati, ma normalmente questo non accade. È bene sempre tenere a mente che il motivo di queste riunioni non è personale, se c'è un problema va risolto tutti insieme, ognuno facendo la sua parte.
La "rabbia" può esserci, ma è sempre da evitare perché non porta a niente, non aiuta a risolvere i problemi; se si alza la voce, non si trovano soluzioni. L'amarezza viene fuori quando si prova ripetutamente a spiegare delle cose e non si viene ascoltati, ma è una reazione sbagliata perché comunque il problema non diventa più semplice: Il nervosismo, quindi, non serve a niente, se non a creare attriti, meno aperture, meno collaborazioni. Meglio avere un colloquio produttivo e costruttivo. Ma come è possibile riuscire a mantenere la calma anche in certe situazioni? Sono comportamenti che si imparano col tempo, infatti ci sono mansioni e ruoli che non

possono essere ricoperti subito anche se si è estremamente competenti dal punto di vista tecnico perché è necessario imparare a interfacciarsi con le persone.

Ad esempio, riceve la segnalazione dei problemi e lui stesso si reca in linea di montaggio per capire dove nasca la difficoltà, il motivo della lamentela di un lavoratore: magari proprio il capo settore di quell'operaio non gli ha mai parlato. Può succedere che in catena di montaggio si stiano eseguendo manovre sbagliate, allora Mister X gliele spiega con calma e l'operaio, trattato in questo modo, si sente coinvolto. Mister X magari coinvolge anche il suo capo settore nella visita che potrebbe anche dire "ma io non sapevo che lui facesse così": in realtà dovrebbe saperlo perché è fondamentale essere a conoscenza dei comportamenti. Procedendo in questo modo, Mister X viene sovente ringraziato per aver risolto il problema in maniera soft senza creare conflitti inutili in comunicazioni via email.
Mister X ritiene che il modo più efficace di approcciarsi nelle varie situazioni possa essere insegnato solo in parte. L'esperienza non è un di più, è molto importante. Lui l'ha compreso fin dagli albori: aveva elaborato una tesi di laurea straordinaria che aveva ricevuto grandi riconoscimenti ed era stato assunto immediatamente dall'azienda con la quale aveva collaborato. Dopo essere stato assunto, si è imbattuto immediatamente nella realtà, ovvero che la sua conoscenza era ancora poca, aveva tanto da imparare e molta esperienza da acquisire. Mister X sottolinea che un corso può dare il 30/40%, per il resto è necessario macinare esperienza. Quindi i corsi per relazioni, se effettuati nel modo appropriato, possono essere estremamente efficaci ma non sono ancora sufficienti: alcune nozioni possono essere assolutamente imparate e applicate fin da subito, per altre è necessario ritrovarsi in tante situazioni in cui applicare quanto appreso e analizzare i risultati ottenuti.

Mister X ritiene che le competenze trasversali siano più o meno importanti a seconda del ruolo ricoperto. Se una persona ricopre un ruolo tecnico come ad esempio il progettista, non necessita per forza di grandissime doti relazionali. Viceversa, più si sale di livello, sovente capita che tecnicamente non si conoscano nemmeno molti concetti specifici perché le persone che ricoprono ruoli decisionali hanno delle tabelle con dei numeri da rispettare e devono essere brave a saper gestire i vari team affinché si raggiungano gli obiettivi previsti.
Mister X ritiene che le competenze trasversali possano essere insegnate solamente in parte, arrivando a una percentuale che non supera il 30-40%.
Il team può essere composto da persone completamente diverse e che cambiano continuamente, quindi l'empatia che entra in gioco non è standard. Ci sono situazioni in cui magari una persona fa tutto correttamente eppure non dimostra di avere quel qualcosa in più che il suo capo vorrebbe. Questo aspetto non può essere insegnato e non è detto neanche che si possa sviluppare con il tempo. Tante volte si "scommette" sulle persone che poi non performano come si sperava. Molto dipende anche dal carattere individuale. L'università fornisce una base tecnica molto importante, anche se poi nel mondo del lavoro probabilmente si usa soltanto il 10%

di ciò che si è imparato; ma la base che fornisce insegna e aiuta a ragionare, perché per molti anni si sbatte la testa. L'università prepara l'ingegnere ma ognuno deve metterci qualcosa in più: tutto ciò può essere un insieme di cose che non si possono insegnare e che dipendono anche dalla propensione. Mister X utilizza se stesso come esempio: ha delle mansioni ben precise, ma non si limita soltanto a svolgerle, si preoccupa anche di altro e per gli altri; perché, come spiegato precedentemente, non può andare poi alle riunioni con i superiori che agiscono nei mercati mondiali e dir loro che un suo progettista ha sbagliato, Mister X è il responsabile perciò è per lui fondamentale cercare di capire tutto quello che viene fatto, anche se non fa parte della sua mansione. Non tutti lo fanno, molti si limitano a mandare email e sperare che tutto vada bene, ma si tratta di centinaia di migliaia di soldi che l'azienda spende/investe e per questo si aspetta risultati importanti, arrivando addirittura poi a cambiare i capi progetto qualora i risultati tardino ad arrivare.
In un'azienda piccola c'è maggiore possibilità di fare di tutto, essere più multidisciplinari e imparare tanto, mentre nelle aziende grandi è tutto vettorializzato e diventa difficile allargare il proprio know-how.

La sua azienda non dà la possibilità di partecipare a tanti corsi per lo sviluppo di competenze trasversali, anzi sono pochi i corsi sostenuti su questi temi. Sostiene che tutti i corsi effettuati finora non gli hanno dato molti risultati concreti e non lo hanno migliorato, alcuni li fanno perché obbligatori e non mirati alla formazione personale. La maggior parte dei corsi effettuati erano costituiti da una serie di slide da studiare. Ritiene che, per migliorarli, dovrebbero essere tenuti da persone con molta esperienza sul campo e non abbiano soltanto studiato la teoria; persone con vera esperienza pratica di anni e magari che abbiano anche partecipato a riunioni poco collaborative imparando così lezioni importanti. In questo modo i corsi diventano veramente efficaci, se c'è confronto e se ognuno porta la propria esperienza, creando interazione; è importante portare le esperienze perché ci sono aspetti che nessuno ci può effettivamente insegnare dall'oggi al domani, come ad esempio relazionarsi con il proprio team: ogni persona ha qualcosa da dare, la si può spronare ma non si può arrivare a farle fare qualcosa che non può fare, è meglio insistere su ciò che sa davvero fare e tirare fuori il meglio dalle sue capacità. Purtroppo per le relazioni non esiste un flow chart, c'è un aspetto psicologico che va approfondito e coltivato nel tempo.
Gli piacerebbe che ci fossero corsi per migliorare da questo punto di vista, anche solo con un corso iniziale di questo genere lo avrebbero aiutato ad affrontare quel che gli è capitato negli anni. Perciò, se fatti bene, i corsi sarebbero importanti e potrebbero dare un notevole aiuto.

Ogni tanto ha l'occasione di effettuare dei colloqui per selezionare dei nuovi ragazzi. Quando effettua un colloquio tecnico per progettisti, ovviamente richiede loro risposte tecniche. Quando è alla ricerca di un team leader, deve avere le stesse

conoscenze tecniche ma, allo stesso tempo, anche competenze relazionali: è proprio quest'ultimo aspetto che lui guarda attentamente e di più. Se dovesse dare una valutazione da 1 a 7 per l'importanza delle competenze tecniche e quelle trasversali, direbbe:

- *Progettista:* 6 per le competenze tecniche e 4 per quelle trasversali
- *Team leader:* 6 per le competenze trasversali ed è disposto a scendere anche a 5 per le competenze tecniche

Il team leader deve rendere coeso il gruppo e deve sapere tirar fuori il meglio dai ragazzi. Queste caratteristiche possono anche essere acquisite da corsi, è un fattore apprezzato, però deve saperle mettere in pratica e dimostrarglielo nel colloquio. Il voto di laurea è rilevante fino a un certo punto perché guarda maggiormente le esperienze vissute dal candidato: magari qualcuno che esce con una valutazione di 80 ha dovuto lavorare e ha un altro vissuto, ne tiene conto e ne dà un peso; viceversa, c'è chi si laurea con 110 e lode però non ha raccolto esperienze che lo abbiano maturato dal punto di vista relazionale, magari in questo caso si guarda se è stato fatto un master all'estero e anche corsi sulle competenze trasversali possono essere un ottimo valore aggiunto. Se un candidato ha seguito un corso per imparare a relazionarsi, lo nota in fase di colloquio, ma se uno si vanta tanto vuol dire che forse non ha compreso molto e le lezioni non hanno avuto l'effetto desiderato: insomma, se sono stati seguiti corsi validi, si nota dall'approccio che diventa più professionale e disponibile, portando ad avere un modo di porsi migliore. Di certo non è sufficiente scriverlo sul curriculum, bisogna dimostrarlo.
Se un candidato ha soltanto mezzora di colloquio a disposizione, come fa a stupirlo e colpirlo? Non può che farlo attraverso tutto quello che è al di fuori delle cose standard, Mister X vuole vedere che tipo di persona ha di fronte.

Mister X ritiene, infine, che la formazione per le competenze trasversali possa essere un settore in forte espansione; negli ultimi tre anni non ne ha fatto nemmeno uno, probabilmente perché l'azienda non lo vede come una risorsa. A suo avviso, ci sono molti margini, se sono fatti bene, e magari le aziende possono anche cambiare idea se lo vedono e vivono davvero.

Seconda intervista

Mister X è un ingegnere che ha sempre lavorato in ambito giornalistico, nel settore delle automobili. Attualmente lavora nell'ufficio stampa, in particolare nell'ufficio comunicazione esterna centrale, direzione europea. Si occupa di tutto quello che è la comunicazione degli enti tecnici relativi alla progettazione di automobili, quindi coordina la comunicazione delle tecnologie. Gli enti tecnici hanno lui come punto di riferimento per quanto riguarda la stampa. Lavora sia direttamente con i giornalisti

quando ci sono interviste o richieste di approfondimento tecnico sia con i colleghi che si occupano della comunicazione del prodotto, dei vari brand o mercati in cui sono presenti. Il suo è prettamente un ruolo di coordinamento e di addetto stampa, lavora direttamente con i giornalisti in ambito europeo e coordina gli addetti stampa anche degli altri uffici.

Ha un team di due persone che lo aiutano anche nella gestione dei test e delle prove che fanno le riviste specializzate e poi svolge un lavoro di coordinamento con tutte le altre funzioni dell'azienda. C'è un rapporto continuo con i colleghi della comunicazione che si occupano dei brand oppure dei mercati. Allo stesso tempo, anche al di fuori del team, esteso e comprendente la comunicazione e l'ufficio stampa, è importante coordinarsi bene e lavorare con tutti i colleghi degli enti tecnici perché bisogna istruirli e dare indicazioni per comunicare all'esterno con i giornalisti; mentre gli altri hanno più conoscenza tecnica, lui fornisce indicazioni su come gestire le relazioni perché quando ci si interfaccia all'esterno, magari in occasione del lancio di un nuovo prodotto, è importante comunicare tutti gli aspetti tecnici in modo preciso e approfondito. Ovviamente loro lo sanno fare ma allo stesso tempo bisogna mettere un filtro e dei limiti, perché magari ci potrebbero essere anche altre soluzioni in fase di sviluppo che non devono essere comunicate all'esterno; quindi lui e il suo team danno le precise indicazioni necessarie, organizzando i contenuti che possono essere trasmessi e quelli che non si possono diffondere al momento.

Il suo superiore è il capo della comunicazione di tutta Europa e lo sente almeno una volta a settimana, anche di più se ci sono esigenze, ma in linea generale hanno una riunione di allineamento settimanale. A parte questo non hanno un rapporto quotidiano. Le persone che lavorano con lui, appartenenti al suo team, sono abbastanza autonome: a volte si sentono qualche volta al giorno, se no sono abbastanza organizzate per procedere e circa due o tre volte a settimana per aggiornarsi su come procedere. Con i colleghi degli altri brand e delle altre funzioni ci sono numerosi scambi; lo stesso discorso vale anche verso l'esterno perché il lavoro è intenso tra incontri, chiamate, email o altro. A volte delega alcune richieste alle persone che lo aiutano perché c'è molto da gestire.
A mister X piace avere rapporto umano soddisfacente con le persone con cui si relaziona, conosce il proprio capo da tanti anni e con lui ha un buon rapporto; il suo superiore gli lascia ampia delega, se ha bisogno si sentono, altrimenti si aggiornano settimanalmente come spiegato precedentemente. Se è tutto sotto controllo, capita che si sentano anche solo per due chiacchiere al di fuori del contesto lavorativo. Con le persone che lavorano con lui ha un rapporto che reputa molto buono, anche se sovente le comunicazioni sono strettamente operative. Con l'esterno, invece, il rapporto deve rimanere sempre buono e stretto; anche quando capita di non sentire un giornalista per tanto tempo, cerca di scambiare con lui due chiacchiere al di fuori del campo lavorativo. Anche per necessità del lavoro, a suo avviso è importante

avere una buona empatia, soprattutto con interlocutori all'esterno.

Quando si ritrova a dover reclutare nuove figure per il lavoro, cerca educazione, diligenza e precisione. Preferisce persone che magari non sono estremamente competenti dal punto di vista tecnico ma con un approccio umano apprezzabile, non che sappiano risolvere problemi molto complessi ma che sappiano relazionarsi efficacemente; ovviamente, questo è anche dovuto al tipo di lavoro perché devono gestire la comunicazione con l'esterno. Quando non riescono a risolvere problemi di natura tecnica, possono sempre appoggiarsi e farsi aiutare dai colleghi degli enti tecnici per tutti gli aspetti più specialistici; il suo team deve essere più competente dal punto di vista relazionale, paziente e disponibile a gestire tutte le situazioni, mantenendo sempre modi eleganti.
È un qualcosa che si può imparare? Secondo Mister X tutto questo dipende molto dall'educazione familiare e scolastica che si riceve. Questi aspetti maturano prima dell'arrivo all'università, proprio nello sviluppo delle persone; ognuno ha il proprio percorso e vive esperienze diverse che lo plasmano e, a seconda della maturazione della consapevolezza che uno può maturare di se stesso, può mettersi in discussione per migliorarsi costantemente, autoanalizzarsi, mettere in discussione il proprio approccio. C'è chi non riflette tanto su questi aspetti e non si fa simili pensieri, dipende anche dal carattere; forse chi è più aperto in tal senso ha più possibilità nel futuro, perché è attento ed è capace di notare aspetti che le altre persone non riescono a notare. In questo senso l'educazione familiare e scolastica nei primi anni di vita possono essere decisivi. Questo discorso vale in qualsiasi ambito della formazione, perché dipende sempre se una persona è in grado di recepire informazioni e gli insegnamenti che riceve. Poi certe tecniche di comunicazione vanno aldilà dell'aspetto educativo perché sono ben definite, anche proprio a livello pratico; perciò, se una persona ha voglia di imparare, può farlo però Mister X crede che tutto quanto non possa prescindere da una certa sensibilità personale. Conta la predisposizione perché non ci sono sempre regole specifiche e universali: a volte capita di vedere miglioramenti incredibili in certe persone dal punto di vista comunicativo e relazionale ma anche in questi casi, a suo avviso, contano molto il carattere, la personalità e la predisposizione.

A volte capitano conflitti con i colleghi, si cerca sempre di risolvere la questione. Mister X cerca di concentrarsi sull'obiettivo e di lasciar perdere gli aspetti secondari e le questioni personali, di dimenticare immediatamente il modo in cui si è arrivati al conflitto. Avendo in mente l'obiettivo, cerca di portare le persone in quella direzione, agendo in maniera tale da farle arrivare da sole alla soluzione: ci vuole tanta energia e tanta pazienza. A volte è molto più efficace costruire autorevolezza nel tempo, piuttosto di arrivare allo scontro e imporre le proprie idee dando ordini con poche spiegazioni: diventa fondamentale condurre le persone a comprendere le motivazioni delle scelte necessarie e riportarle a condividere le finalità, in maniera da

renderle, loro stesse, promotrice degli obiettivi necessari. Non capita praticamente mai che i conflitti si trasformino in questioni personali perché crede che alla base di tutto nel suo contesto ci sia una buona considerazione reciproca tra colleghi. Poi è chiaro che si ha sempre a che fare con essere umani, a volte si impara a conoscersi bene e ci si stima ben sapendo che non si è perfetti.

Mister X sarebbe molto interessato a partecipare a corsi per imparare a gestire persone e conflitti e sviluppare maggiormente, in generale, le competenze trasversali. La formazione dev'essere effettuata su tutti gli aspetti in tutti gli ambiti; alcuni argomenti sono stati affrontati nelle sessioni di formazione cui ha partecipato fino ad oggi. Nei corsi messi a disposizione c'è una base comune a tutti, poi ognuno può scegliere quali seguire a seconda dei propri bisogni. I corsi non mancano nella sua azienda, sarebbe bello averne di più. Alcuni sono stati efficaci e interessanti, soprattutto quelli effettuati con i dirigenti. A volte capita che certi corsi siano costituiti solamente da slide e in questi casi sono considerati più che altro un ripasso, ma in altri sono proposte interazioni e scambi di un certo profilo, che aggiungono valore e portano maggiore soddisfazione a chi partecipa.

Terza intervista

Mister X ha lavorato in tante azienda, da alcune piccolissime startup a giganti imprenditoriali come Google. Vive da nove anni in Gran Bretagna, perciò ha sviluppato una visione differente da quella italiana e sotto alcuni aspetti forse più aggiornata. Ha lavorato sempre in ambito IT, cominciando come sviluppatore, e adesso il suo lavoro è nel campo delle energie rinnovabili in una grande azienda.
Mister X è manager di 3 team: security engineering, site reliability engineering e data integration team: ogni team è composto in media da 6 persone. Ha un diretto superiore, denominato Headway Engineer, il suo superiore è il CTO (Chief Technical Officer): può contattarli quando desidera perché c'è molta apertura e disponibilità al confronto. Ha una buona relazione con il suo superiore e ci tiene a precisare che la gerarchia è meno pesante rispetto all'Italia.

Nella sua azienda, ma anche nella maggior parte di quelle per cui ha lavorato, vi sono dei cosiddetti *One-on-One*: sono dei veri e propri meeting individuali con il superiore; ogni manager, pertanto, partecipa con il superiore a questi meeting sovente allargati a tutti i membri del gruppo o in alternativa organizzati anche individualmente. Se si fanno i meeting con i superiori di almeno due livelli, si parla di *Step-One-on-One*. Questi incontri hanno una durata variabile tra la mezzora o l'ora e vengono effettuati ogni singola settimana con l'obiettivo di conoscersi meglio e di migliorare il morale del team. È difficile da spiegare ma è facile vedere gli effetti degli One-on-One perché un gruppo diventa particolarmente affiatato. Perciò in generale non si parla di lavoro, si parla di sé, come vanno le cose, quali problemi ci

sono, come migliorare certe situazioni, desideri, dubbi, vita privata e così via. Mister X, dunque, dedica ogni settimana 7 ore e mezza a svolgere i suoi One-on-One. Quest'abitudine è presente in tutte le aziende per cui ha lavorato fuori dall'Italia in ambiente anglosassone, persino nelle startup e in Google. La cadenza è importante, è fondamentale non spostare mai gli One-on-One se non c'è emergenza. Esistono libri, corsi, materiale formativo per imparare a condurre un One-on-One ma Mister X sottolinea come si impari molto di più ricevendolo. Utilizzano un documento Google in cui si inseriscono anche argomenti di cui vogliono parlare. In questi meeting spesso vengono fatte anche delle richieste o si stabiliscono delle azioni da compiere, pertanto è fondamentale tenerne traccia e mantenere la propria parola tra un meeting e l'altro.
Oltre a questo tipo di meeting, vengono svolti anche incontri *Team Fun*: si svolgono vere e proprie attività ricreative tutti insieme, come ad esempio giocare a pallone.

Esiste anche una *Retrospecting Board*, una lavagna anonima in cui ognuno ha la possibilità di appendere un post in 3 colonne: *Start*, *Stop* e *Continue*. Questa lavagna non riguarda solo il contesto lavorativo ma vale proprio in generale, è anonima. Mister X è favorevole al rilascio di feedback anonimi perché permettono alle persone di entrare in una condizione di *psychological safety*, ovvero di sicurezza mentale. A suo avviso è meglio che sia così, le persone in questo modo sono più invogliate a dire ciò che pensano realmente. Ovviamente poi ci sono anche persone che non hanno problemi a dire sempre apertamente ciò che pensano, ma altre no. Ciò che viene segnato sulla lavagna viene periodicamente discusso tutti insieme per poi prendere una serie di decisioni. Quando si raggiunge questo tipo di coesione e sicurezza mentale, succede anche che le persone dicano di aver scritto un particolare post ed esprimano a pieno il proprio pensiero. Mister X spiega come l'anonimato non sia una questione di nascondersi, ma una vera e propria sensazione di sicurezza psicologica che si instaura nelle persone consapevoli di poter scrivere ciò che pensano davvero e che verrà considerato in maniera neutra, senza un giudizio sulla persona: in seguito si comprende anche che non c'è alcun tipo di problema a parlare apertamente. Mister X ritiene che, per arrivare allo stesso livello di sicurezza mentale senza anonimato, si possa trasmettere questo senso di fiducia anche attraverso i One-on-One: se il dipendente nota che può dire quello che vuole, ovviamente con i giusti modi, confidenzialmente e senza ripercussioni, se nota la considerazione verso ciò che dice, allora è possibile ricreare lo stesso clima di sicurezza. Il punto di arrivo non è togliere l'anonimato, ma è far sì che tutti si sentano sicuri di poter esprimere la propria opinione; poi, se il tool è anonimo o no, è solo un mezzo ma non rappresenta l'obiettivo. Questo senso di sicurezza non è un qualcosa che si può imporre dall'alto, ma dipende dalle azioni di tutte le persone e dalla cultura dell'azienda: se un dipendente nota che il suo manager mette tutti nelle condizioni di sentirsi sicuri, questa cultura viene poi trasmessa anche alle persone che gestisce. Non è un'imposizione dall'alto ma è fondamentale *to lead by*

example, ovvero guidare le persone in maniera tale da dare il giusto esempio. Tutto questo dipende anche dalla grandezza aziendale: nelle startup è più difficile perché tutte le persone si conoscono ma è proprio quello il momento giusto per cominciare a creare un clima di serenità.

Mister X non nota differenze di comportamento tra le persone che ricoprono ruoli di dimensioni differenti: il rapporto rimane più o meno lo stesso con tutti. All'estero, i livelli dei ruoli sono più confidenziali; spesso non si sa esattamente i livelli degli altri colleghi (ad esempio se si tratta di un senior manager o di un junior manager), perciò si è sempre alla pari. Lo stesso discorso vale soprattutto all'interno dei team, dove nessuno conosce il livello del collega, a meno che non venga esplicitamente dichiarato o confidato. La mentalità è orientata al voler lavorare e collaborare tutti insieme; non conta il livello e ogni persona non deve preoccuparsi della gerarchia, bisogna considerare quello che si dice senza pensare al ruolo ricoperto da chi lo ha detto: è anche questo che contribuisce alla sensazione di sicurezza mentale analizzata precedentemente. C'è più attenzione alle relazioni tra colleghi e manager all'estero rispetto all'Italia,
Esistono inoltre delle *Performance Review*: ogni dipendente ha la possibilità di dare un resoconto e una valutazione al proprio manager. Ci sono dei processi, come quelli elencati sinora, che devono essere standard quando l'azienda diventa grande.

Le competenze relazionali diventano importanti salendo di livello, quando si svolgono mansioni prettamente tecniche potrebbero servire relativamente. La maggior parte delle persone che ha incontrato finora e che svolgono lavori tecnici, ad esempio gli sviluppatori web, è timida e introversa; in questi casi, per il manager è fondamentale capire come relazionarsi con loro e parlare anche in maniera soft, cercare di introdurre anche argomenti che vanno al di fuori del lavoro. Mister X ritiene che, per un manager, le competenze trasversali siano importanti tanto quanto quelle tecniche ed entrambe devono coesistere perché sono necessarie per l'avanzamento della carriera. Le capacità tecniche si possono sempre imparare; quelle relazionali possono anche essere imparate ma è sempre meglio avere un buon punto di partenza.
Ci sono corsi sullo sviluppo delle competenze trasversali per i manager; chi è interessato ha la possibilità di accedere anche a corsi aggiuntivi, o magari c'è anche la possibilità che il manager voglia far seguire un corso specifico a una persona specifica, per farla migliorare sotto certi aspetti. Di solito questi corsi sono su una piattaforma online e possono essere seguiti quando si vuole; sono organizzati da consulenti esterni tramite un sistema e-learning. Ci sono anche corsi interattivi a numero (c'è una soglia minima di iscritti al di sotto della quale il corso non parte): entrambi i tipi sono efficaci ma dipende dagli argomenti, a volte sono meglio quelli interattivi altre volte gli altri. Ritiene che la teoria e pratica siano importanti allo stesso modo: ad esempio, per gestire un gruppo, pensa sia importante avere una

base formativa e allo stesso tempo mettere in pratica quanto appreso, acquisendo così esperienza.
Per Mister X non è stato limitante non aver seguito e partecipato a corsi prima di entrare al lavoro; non saprebbe dire se gli avrebbero permesso di fare carriera più velocemente imparando certe cose in precedenza. Inizialmente, da giovani, non si ha nemmeno idea dell'esistenza e dell'importanza di certe competenze, perciò non ne sentiva la mancanza prima di entrare in un'azienda; poi, lavorando e migliorandosi anche attraverso i corsi, ha pensato svariate volte che sarebbe ritornato molto utile avere certe informazioni in passato. C'è poca riflessione all'inizio su questi temi soprattutto in Italia, perché non c'è la giusta sensibilizzazione sull'importanza delle competenze relazionali.
Mister X sottolinea che non è obbligatorio diventare dei manager man mano che si avanza con l'esperienza, si può benissimo anche decidere di avanzare nel proprio ambito tecnico, aggiungendo eventualmente attività di coaching e mentoring. Se si scegliere di ricoprire ruoli manageriali è perché si è interessati, ma non è corretto pensare che sia obbligatorio diventare manager.

Mister X ha avuto spesso la possibilità di effettuare dei colloqui per selezionare dei candidati. Generalmente, i colloqui sono divisi in code interview, design interview e behavioral interview. La parte da lui condotta, molto importante, è proprio l'ultima e ha come obiettivo di comprendere che tipo di persona abbia di fronte, le sue capacità relazionali. Mister X cerca di inquadrarne il comportamento, ad esempio chiedendo come si muoverebbe in certi scenari. Alle sue domande non esistono risposte giuste o sbagliate, semplicemente ne ricava un'indicazione sul tipo di carattere e sugli interessi (ad esempio la preferenza su lavori tecnici, sulla gestione dei team, sulla predisposizione ad attività di coaching e mentoring e così via) dei candidati. In questi contesti, Mister X prova anche a fidarsi del proprio istinto perché non vi è una scienza esatta per comprendere il carattere di una persona in un solo colloquio. Ovviamente, nelle prime fasi dei colloqui vi è una scrematura di natura tecnica: per arrivare al behavioral interview è fondamentale possedere i requisiti per le competenze tecniche, non è possibile essere assunti per simpatia nel behavioral interview ma, allo stesso tempo, a parità di competenze tecniche la spunta chi dimostra di essere più brillante proprio nell'ultima fase della selezione.

Quarta intervista

Mister X ha studiato psicologia con indirizzo marketing organizzazioni. In passato ha lavorato per due aziende, una di assicurazioni e una di ingegneria sempre in ambito risorse umane per la formazione e gestione dei talenti. Oggi lavora in un sede italiana di una multinazionale di consulenza che opera in varie Nazioni.
Si occupa dello sviluppo delle persone (talent management e development, ovvero sviluppo e gestione talenti), alcuni suoi colleghi valutano e monitorano performance

mentre Mister X coordina il team che si occupa dello sviluppo delle competenze, del coaching e della leadership. Vi è anche un dipartimento di formazione per quanto riguarda le competenze tecniche e tutto ciò che riguarda la formazione interna all'azienda passa dal suo team: vi sono corsi sia e-learning sia organizzati in aula. Mister X gestisce anche i neoassunti, si occupa del loro *onboarding* per le competenze che devono sviluppare una volta entrati in azienda. In particolare, lavora con i team leader di ogni livello, aiutandoli a sviluppare la leadership, e con gli internal coach, direttamente a contatto con il personale per un supporto completo, per aiutarli a sviluppare le competenze necessarie. Mister X si occupa dei dipendenti interni e non dei servizi, che rappresentano il core business, forniti all'esterno dall'azienda.

Quotidianamente, Mister X si confronta con la responsabile diretta di tutto il compartimento, sopra di quest'ultima vi è la direttrice delle risorse umane che si interfaccia al partner di riferimento a capo di ogni struttura, anche nelle funzioni di staff. Il rapporto con la sua superiore è stretto, basato su stima e rispetto reciproco della seniority che ciascuno porta. Ogni settimana, dedicano circa mezzora come momento di connessione per allinearsi. La direttrice delle risorse umane è una persona molto concreta, che riesce a mettere sul tavolo i diversi punti di vista e creare sinergia e ordine. I vari contatti sono un po' macchinosi, soprattutto nell'ultimo anno in cui c'è stata la pandemia, perché i meeting sono schedulati sempre in anticipo; nonostante questo, la sua responsabile è sempre presente, la direttrice ha diversi progetti aperti perciò è difficile avere dei momenti di confronto, ma è conciliante e si fida. Mister X solitamente gestisce un team di due persone cui si aggiungono dei dipendenti junior a seconda dei progetti, anche se non c'è volontà di imporre una gerarchia troppo definita. Nel suo dipartimento, infatti, cercano di creare delle *Bubble* per tentare di focalizzarsi sulle interazioni all'interno del team. Il concetto di Bubble è stato importato dalla sua responsabile, che segue studi sui trend di come si dovrebbe lavorare efficacemente in gruppo. Nella teoria è facile, nella pratica è difficile. I colleghi impegnati in consulenza lavorano seguendo il principio della Bubble: i team nascono e muoiono a seconda dei progetti, tutti lavorano su più progetti contemporaneamente e, in questo modo, i team sono veramente molto flessibili. Per Mister X è un po' più complicato riuscire ad implementare sempre questo tipo di meccanismo perché il team è più piccolo. Procedendo in questo modo le persone non sono incasellate in un ruolo sempre uguale e hanno la possibilità di osservare più leadership diverse, questo ha un grande valore formativo.
Le relazioni sono molto particolari e variano a seconda delle persone, ma cercando di costruire una relazione alla pari tenendo conto della seniority. C'è molta competenza ma allo stesso tempo disponibilità al confronto, ma Mister X rimane comunque il responsabile finale. Con uno dei due membri del team si sente sempre, anche al di fuori del contesto lavorativo, con l'altro in maniera più strutturata. Ogni settimana viene schedulato un meeting con il team allargato per allinearsi.

Mister X ritiene che l'empatia sia una competenza che si allena tramite la flessibilità: osservando e analizzando tante persone, è possibile riuscire a tirare fuori il meglio da tutte le relazioni. Adora l'idea di cambiare dinamicamente le persone nei gruppi di lavoro perché permette di analizzarsi meglio, di comprendere perché certe relazioni funzionino meglio rispetto ad altre; se, invece, si rimane fissi sempre con le stesse persone, che magari non piacciono nemmeno, non si ha molta possibilità di crescita e maturazione. C'è più varietà anche di stile e di modi di fare.
L'empatia è molto importante, facilita le relazioni sia all'interno del team sia nel business, perché permette di avere successo anche con i clienti. Olia e rende più semplici le relazioni. L'aspetto delicato è proprio le relazioni, più del lato tecnico. L'empatia è un'attitudine, che per certe persone è naturale, ma è anche possibile imparare. La sua azienda possiede anche uno strumento che serve a sviluppare l'empatia, Mister X sostiene, metaforicamente, che esistano diamanti naturali ma anche diamanti sintetici, che possono magari essere anche meno fragili di quelli naturali. Per imparare, è necessario osservare con differenti chiavi di lettura e sforzarsi di trovare sempre il valore che c'è dall'altra parte. La parte di teoria in questo senso conta, ma vi è bisogno anche di riflessione e sperimentazione per imparare davvero. È necessario comprendere le dinamiche, cosa accede dentro di noi quando ci si arrabbia e che cosa fa scaturire certe reazioni. È fondamentale riuscire a imparare a esercitare empatia anche con persone opposte a noi, che ci stanno antipatiche e magari parlano di cose che non apprezziamo per niente.
Se si vuole esercitare empatia per mestiere, ad esempio un venditore per chiudere contratto di lavoro, non è necessaria un'autoanalisi. Se, al contrario, si vuole rendere l'empatia un vero e proprio punto di forza personale, bisogna continuare a imparare e conoscersi andando sempre più nel dettaglio: si può decidere anche di smettere quando ci si reputa sufficientemente soddisfatti delle proprie relazioni.

A Mister X non è mai stato insegnato come gestire le persone. La teoria è diversa dalla pratica e lo sa bene perché insegna quotidianamente ai manager come gestire le persone: sulla carta è tutto funzionante, poi entrano in scena le persone che creano dinamiche e mondi inesplorati. È fondamentale concedersi del tempo per imparare.
A Mister X è capitato di tenere diversi corsi per chi non ricopriva un ruolo di manager e ha notato che si trattava di corsi vuoti, nel senso che è possibile imparare la teoria ma non si ha lo spazio di applicazione: Mister X concorda con il ciclo di Kolbe, ritiene che sia proprio la sperimentazione e la riflessione a permettere l'apprendimento per davvero. Solo la teoria, non funziona. Al contrario, i corsi per i manager che vivono quotidianamente situazioni in cui possono applicare i concetti imparati sono molto più proficui; si parte solitamente dalle esperienze che si vivono, si ragiona sulle difficoltà incontrate e si cerca di trovare soluzioni tramite il ragionamento basato anche sui concetti teorici, si fanno simulazioni e si riflette con feedback tra pari. In questo modo è possibile osservare le situazioni da lenti diversi. Teoria e pratica non si possono slegare: il 70% è l'esperienza sul campo, il

20% è il feedback che si osserva e, infine, il 10% è rappresentato dalla teoria, che rappresenta un seme che viene gettato per svilupparsi sul campo. Proprio alla luce di questo, è molto utile concedersi uno spazio per riflettere e analizzare le situazioni.

Nella sua azienda non si svolgono attività di mentoring, soltanto di coaching. Mister X considera il coaching come un acceleratore di apprendimento. Avere spesso dei capi diversi permette di osservare leadership diverse, ma questo può creare anche disorientamento nel sapere conciliare i punti di vista: per questo entra anche in gioco il percorso di coaching. Anche perché, nel breve termine, si può apprendere direttamente dai propri leader che forniscono importanti feedback ma, sul lungo termine, esiste appositamente la figura del coach, che non conosce al 100% la persona. Questo permette di analizzare le difficoltà che si hanno quotidianamente e, in aggiunta, di parlare anche su aspirazioni e obiettivi sul lungo termine.

A Mister X è capitato più volte di effettuare dei colloqui di selezione e lo trova difficile, soprattutto perché si tratta di un gioco delle parti in cui l'autenticità è difficile da scovare, tutti sembrano perfetti durante i colloqui di lavoro. È proprio per questo che, nella parte che concerne la behavioral interview, utilizza una tattica ben definita: fa partire il candidato da una situazione, che gli è successa in passato, in cui ha cambiato opinione o in cui ha fatto cambiare opinione a qualcun altro. Gli chiede di analizzare la situazione, che cosa è successo e come è avvenuto il processo anche dal punto di vista interiore, come si è sentito e cos'ha provato. Questo modo di procedere esce dal solito schema del "come farei" e permette di passare all'atto pratico, "come ho fatto". Inoltre, se il candidato prova a mentire questo salta subito all'occhio perché si presentano varie lacune nel discorso. In questo modo, è possibile farsi un'idea un po' più chiara del tipo di persona che si ha di fronte.
Per Mister X è fondamentale che il candidato possa inserirsi nel migliore dei modi all'interno del gruppo di lavoro, perché deve avere un buon livello di empatia per saper stare con persone diverse. Apprezza in particolare la flessibilità, la curiosità che porta a interrogarsi sul diverso e la capacità di riuscire a cambiare opinione quando può essere necessario. Che valore aggiunto può apportare nel team? Non a livello di conoscenza tecnica, ma di persona. L'aspetto relazionale viene ricercato ed è fondamentale. Non assumerebbe mai persona che non sappia lavorare in team e non sappia essere collaborativa, nonostante possa essere molto competente negli aspetti tecnici. A volte prevalgono le competenze tecniche nelle selezioni però Mister Xs preferirebbe l'aspetto umano, perché ritiene che le competenze tecniche si possano sempre imparare e/o limare, invece per gli aspetti relazionali il processo è molto più lento: guarderebbe per il 70% l'aspetto relazionale, per il 30% la tecnica.

Mister X non ha mai effettuato corsi di formazione prima di entrare nel mondo del lavoro e ritiene che sia un qualcosa che, se appassiona, si segue per la vita. La

sua azienda investe poco su certe competenze ma molto in altre, dà la priorità magari alle certificazioni che permettono di acquisire nuovi clienti. L'investimento sulle competenze relazionali dev'essere fatto su base continuativa perché è un processo lento e che dura a lungo nel tempo. Alcune persone escono dai corsi trasformate (ad esempio nei corsi sulla leadership) sostenendo che sia scattato qualcosa dentro di loro, una scintilla. Questo non succede in tutti i corsi e in tutte le persone ma ha un impatto concreto.
Ritiene, tuttavia, che si faccia troppo poco e l'impatto di questa mancanza deteriori sia il business sia le prestazioni personali. A volte, durante i suoi corsi, capita che ci siano persone che devono assolutamente andar via perché il loro capo continua a fare richieste da portare a termine a breve; si dà sempre, purtroppo, la priorità a questo. Ciò capita con persone di tutti i livelli, anche neoassunti che sono costretti a lavorare mentre partecipano ai corsi di formazione. Alcune persone sono scettiche, contestano molte affermazioni perché non si fidano e vogliono testare l'effettiva competenza del formatore. All'estero, invece, ci sono meno problemi da questo punto di vista. Anche solo un esempio banale: quando si inizia la formazione, si spengono i cellulari, qui in Italia sembra di fare una richiesta assurda. Anche in questi casi diventa fondamentale dimostrarsi assertivi ed empatici, magari le persone si distraggono il cellulare per il semplice fatto che il proprio capo continua a mandare richieste senza dare la possibilità di prendersi del tempo per dedicarsi a pieno alla formazione.
Mister X ritiene che, all'università, potrebbero essere svolti corsi su comunicazione e relazioni; diventa più complesso, invece, svolgere corsi su come gestire le persone, per la maggior parte dei ragazzi non ha senso perché manca l'esperienza necessaria da mettere sul tavolo e sulla quale discutere, non avebdhy nemmeno uno spazio in cui mettere in pratica quanto appreso. Per quanto riguarda la comunicazione e gli aspetti relazionali, invece, possono essere messi in pratica durante la carriera universitaria o anche semplicemente a casa; la leadership potrebbe essere sviluppata, invece, solamente se venissero creati gruppi di lavoro con delle gerarchie e con un lungo termine, altrimenti non si ha la possibilità di affrontare situazioni in cui effettivamente diventa necessario gestire persone e problemi che si creano.

Quinta intervista

Per questioni di privacy, quest'intervista non può essere riportata integralmente: vengono annotati alcuni concetti approfonditi durante l'intervista, ma in maniera molto generale.

Mister X lavora per Google e, in precedenza, ha lavorato per altre aziende grandi e molto importanti.
Durante l'intervista, sono stati trattati numerosi argomenti anche di natura molto

vasta, ha raccontato moltissimi aneddoti e i temi principali, tra quelli che possono essere riportati, sono i seguenti:

- Alla Google c'è un'attenzione estrema a tutto ciò che concerne la vita dei propri dipendenti. Si riporta un paio di aneddoti per comprendere quanto è elevato il livello di attenzione alle persone: si dedica persino del tempo per decidere tutti insieme che tipo di merendine inserire all'interno delle macchinette; vengono organizzate riunioni con i fondatori (Larry Page e Sergey Brin) che rispondono a tutto ciò che i dipendenti desiderano chiedere loro. I fondatori sono sempre disponibili ad ascoltare in queste occasioni e capita che i due leader facciano promesse davvero stravaganti, rigorosamente mantenute. Vi sono le performance review dei capi. Google paga due terzi delle attività svolte dai dipendenti per imparare un qualcosa di nuovo, anche se non strettamente legate alla mansione svolta; paga inoltre un terzo delle attività di svago, perché l'idea di base è che se una persona trascorre del tempo svolgendo un'attività che la interessa e appassiona, allora è più contenta e serena: il risultato finale è senza dubbio un livello di performance maggiore.

- È fondamentale porre estrema attenzione ai bias della mente: ad esempio, per non creare dei bias mentali per la distinzione uomo/donna in fase di colloquio di selezione, è permesso inviare il curriculum senza informazioni quali il sesso e l'età. Colui che svolge colloquio viene istruito e allenato al massimo delle possibilità per non farsi ingannare dai bias.

- Il concetto di *Googliness*: i candidati vengono portati a pranzo dopo i tre primi colloqui, per capire se sono persone socievoli con cui è piacevole trascorrere il tempo, delle vere persone da Google. Google cerca persone talentuose, proattive e alla ricerca dell'eccellenza ma, allo stesso tempo, umili, oneste, che sappiano relazionarsi e con cui è bello stare insieme. Ovviamente si tratta solamente di una vaga idea di che cosa si intenda con Googliness, per scoprirlo meglio è necessario vivere un'esperienza in prima persona.

- Come fare per portare questa attenzione alle persone e agli aspetti relazionali al di fuori delle più grandi aziende mondiali? Il primo passo è la consapevolezza dell'attenzione necessaria per i dettagli. Quindi, in primo luogo, è opportuno sensibilizzare le persone e le aziende

Sesta intervista

Mister X è da poco in pensione. Negli ultimi vent'anni, ha lavorato in un ente pubblico per lo sviluppo risorse umane all'interno della direzione del personale: in base agli obiettivi dell'ente e dell'organizzazione, doveva mettere a fuoco la struttura organizzativa, quindi le unità operative e i dipendenti, e collocare le persone sulla base del loro potenziale, oltre che delle competenze specialistiche. In aggiunta, si

occupava di allestire la formazione per le persone: sia per i neoassunti, che entrano in un'organizzazione e hanno bisogno di condividere una visione organizzativa, sia per tutti coloro che effettuavano passaggi e mobilità interne (ad esempio da un servizio a un altro oppure da un compito a un altro) sulla base di una valutazione del personale, dopo aver recepito le esigenze di far ricoprire a determinate persone i ruoli più consoni ad esse.

Il suo team era composto da tre psicologhe e altre due colleghe con compiti amministrativi. Mister X è stato responsabile per dieci anni del servizio sviluppo risorse umane. Il rapporto, da parte sua, era basato sulla condivisione di progetti e obiettivi, che ovviamente discendevano dagli obiettivi generali della direzione del personale, tradotti in termini di ricerca e selezione del personale: decideva come comportarsi soprattutto affidandosi all'esperienza delle psicologhe. Organizzavano riunioni a cui partecipavano tutti, la conoscenza e visione di quelle amministrative era la più importante perché aveva visione dall'esterno, era a contatto con altri. Era un team unito, condivideva tutto, poi si suddividevano gli impegni in base ai compiti
Mister X è specializzato in formazione all'interno delle aziende: ideava, grazie all'aiuto del suo team, un piano per la formazione delle persone. Condividevano un'impostazione iniziale e, a seguito di un brainstorming collettivo di idee, valutavano quali proposte potevano essere operativamente concrete in formazione; quelle scartate magari potevano essere riprese in altri momenti più o meno futuri.

Mister X si occupava maggiormente di tenere i corsi di formazione, che erano rivolti a categorie ben precise di dipendenti. Ad esempio, c'era un percorso specifico per i neoassunti. Delineavano gli obiettivi partendo dal neoassunto e dalle sue aspettative, condividendo le finalità con i valori aziendali.
In aggiunta, lavorava per un progetto di comunicazione sia interna a un gruppo di servizio sia verso il cittadino/utente. Erano progetti molto interattivi, venivano proposti casi, filmati, giochi di ruolo per provare a mettersi nei panni delle altre persone e vedere che tipo di messaggio ricevessero. L'obiettivo era arrivare a dedurre i concetti teorici attraverso delle esercitazioni mirate. Al termine dei corsi, vi era una rilevazione del gradimento ed efficacia. Solitamente i corsi erano molto utili ai partecipanti, ma a volte si venivano a creare veri e propri contrasti tra i partecipanti al corso e i loro diretti superiori: questi ultimi non avevano le stesse impostazioni e le stesse tecniche a disposizione, quindi scadeva un po' la possibilità di praticare quanto appreso nel corso di formazione. Per risolvere questo inconveniente, un passaggio importante è stato condividere il piano con tutti i dirigenti e farlo approvare; sarebbe stato ancora meglio ideare un percorso di formazione a partire dalla condivisione di un obiettivo con il diretto responsabile.
La formazione non esauriva mai perché c'erano i cosiddetti follow up: cos'è cambiato rispetto a prima? Ora qual è la parte critica? Magari si comprende che il

problema era un altro e non era stato ben identificato. Questo è il giusto modo di procedere e il tipo di lavoro che può portare risultati concreti.
Fornivano, inoltre, un servizio di counseling ai dipendenti: colloqui individuali su richiesta dei dipendenti in cui venivano recepiti problemi, bisogni e proposte dei dipendenti rispetto al proprio incarico. Durante i colloqui con dipendenti, si è partiti sempre dai fatti in maniera tale da avere una comunicazione con uno sviluppo, altrimenti si correva il rischio di rimanere astratti e nel mondo delle impressioni di ognuno.

La completa valutazione avveniva attraverso un team, composto da 3 psicologhe del lavoro, con le quali ha elaborato un percorso che, attraverso l'utilizzo anche di auto questionari, permetteva alla persona di esprimere effettivamente quello che era il proprio desiderio e sentiva essere il proprio percorso; a volte è un qualcosa che le persone si immaginano, ma su cui non riflettono sufficientemente. Questo avveniva in almeno due o tre incontri.
Inizialmente, veniva creata una scheda di profilo, ovvero un documento in cui è indicato il tipo di persona che si cerca, quali caratteristiche deve avere e, oltre alle competenze meramente tecniche, quali caratteristiche personali deve possedere: la scheda veniva pure allegata al bando di concorso. Un metodo per effettuare la selezione in base alle caratteristiche indicate. La valutazione avveniva sia a livello di competenze tecniche sia trasversali: le competenze tecniche facevano sbarramento anche se con un certo grado di flessibilità perché si possono sempre acquisire, invece le psicologhe del lavoro si soffermavano ad analizzare la mentalità della persona, la sua capacità di stare in team, la facilità nel relazionarsi e la curiosità innovativa. Alcune persone sono più produttive ed efficaci in compiti schematici e rigidi, altre invece in ruoli in cui prevalgono le competenze relazionali e la capacità di guardare le situazioni con occhi diversi, lavorando a stretto contatto con altri.

Mister X fatica a immaginare corsi sulle competenze trasversali che siano sganciati dall'organizzazione e da un contesto pratico. Crede che all'università possa essere insegnato qualcosa solo se prima vengono raccolte le domande dei partecipanti: quindi è opportuno effettuare una rilevazione preliminare e, partendo dai loro bisogni timori e convinzioni, si imposta il lavoro. Dopo il corso, è importante monitorare i cambiamenti, per vedere se quello che è stato fatto abbia portato effettivamente un valore aggiunto.
A suo avviso, la combinazione formativa perfetta è rappresentata da corsi interattivi, per suscitare interesse e permettere l'apprendimento di nozioni e situazioni, e da attività di tutoraggio per fare in modo che quanto appreso non venga dimenticato e messo da parte. Ritiene fondamentale avere momenti in cui si ripensa attivamente all'esperienza, la si analizza e si considerano i feedback ricevuti che sono estremamente importanti per sviluppare aspetti positivi da mettere a disposizione di tutti.
Mister X aggiunge che all'università sarebbe importante cominciare già a imparare

come funziona l'organizzazione aziendale, il lavoro per processi o per obiettivi, quali sono le varie forme organizzative che ci possono essere. Inoltre, sarebbe efficace incominciare già ad utilizzare uno strumento come il diario di bordo per rilevare quotidianamente le aspettative e avere un confronto con la realtà.

Quando ha effettuato colloqui di selezione per valutare le competenze relazionali dei candidati, era solito inventare casi e chiedere al candidato di rispondere anche in forma diretta. Le capacità comunicative dipendono dalla personalità, ma si possono certamente apprendere. Quello che conta è l'intenzione, se non è onesta l'altro lo percepisce. Non si può non comunicare (paradigma) perché anche il silenzio è una forma di comunicazione, quindi vi è una parte teorica che è possibile apprendere ma deve corrispondere effettivamente a un qualcosa che viene esperimentato e trovato valido nella modalità comunicativa (ad esempio anche in famiglia): in questo modo, l'intenzione vera si rivela. Se le tecniche vengono apprese per il ruolo di venditore, invece, è tutto un altro discorso: si imparano metodologie ma si tratta di un'altra disciplina, perché la comunicazione diventa uno strumento del mestiere. L'intenzione onesta può essere allenata attraverso la formazione o un'attività di tutoraggio, affinché ci sia coerenza tra ciò che si prova e ciò che il prossimo riceve.

Settima intervista

Mister X è un ingegnere elettronico. Mentre frequentava l'università ha lavorato con un'altra persona realizzando una startup. Il primo lavoro da dipendente è stato come stagista nel mondo dell'industria, occupandosi delle telecomunicazioni. Successivamente ha cambiato ambito entrando a far parte di una azienda con il ruolo di test engineering, occupandosi di realizzare test specifici in base alle norme del cliente. In seguito, ha nuovamente cambiato azienda, entrando a far parte di una agenzia di consulenza dove si occupa di test direttamente a bordo della vettura. È stato poi due anni in un'altra azienda, in un'altra città, dove ha cambiato modalità di lavoro: ha cominciato a gestire team di persone. Infine è tornato alle... origini: da circa 6 anni ricopre un ruolo più gestionale, ovvero di team manager in un'azienda di consulenza che vende servizi ingegneristici ad altre aziende. Si occupa, in particolare, di:

- gestire tutto quello che è il team in termini di risorse umane,
- delivery di progetti che lo porta a interfacciarsi sia con i clienti per capire se ci sono problematiche sia con i colleghi per capire dove possono dare supporto
- sviluppare idee di business con i colleghi, cercando nuovi progetti, tra i clienti che segue ma anche con i nuovi
- tutto ciò che concerne la parte amministrativa dei progetti in carico, quale la gestione degli ordini, la fatturazione e così via

L'intero team è composto da circa 180 persone divise in due aree geografiche diverse: Torino e Napoli. Il team segue i clienti che si occupano di tutto ciò che è legato al mondo automotive. In particolare, a Napoli, i clienti si occupano della parte tecnica dalla progettazione meccanica fino agli stampi; a Torino, invece, il team segue i clienti legati al mondo del power trend. Questa area divisa tra Napoli e Torino viene gestita da Mister X e altri due team manager. I 180 colleghi sono divisi in unità di team più piccoli, guidati dai team leader.

Nella parte manageriale, la gestione è piramidale, lui risponde direttamente a un superiore. Il contesto in cui lavora è abbastanza amichevole: si collabora e si parla abbastanza sovente. L'ambiente è composto da soli giovani. A suo avviso, il capo non è propriamente empatico ma Mister X ritiene che sia sempre aperto al confronto e al colloquio. Ciò che limita il suo rapporto con il capo è la forte pressione di business all'interno dell'azienda.
Per un team leader, che svolge un lavoro tecnico, sono più importanti gli aspetti pratici rispetto a quelli relazionali. Dal team manager in su, le competenze trasversali sono più importanti perché devono essere in grado di gestire efficacemente team molto grandi e di relazionarsi appropriatamente con i clienti. Nella sua azienda si dà importanza all'aspetto relazionale, non solo a quello tecnico.
Il suo team è composto da tre persone e due di queste erano già suoi colleghi prima che lui ne diventasse il loro diretto superiore: il rapporto è ottimo. Mister X si ritiene una persona molto collaborativa e cerca di rendersi sempre molto disponibile. Con ogni membro del team effettua una *riunione di perimetro* settimanale dove partecipano i tre team manager e i cinque team leader distribuiti su più aree geografiche (che hanno appunto i team manager come diretto superiore). In queste riunioni si discute dei vari problemi lavorativi di ognuno; in questi momenti ognuno può portare il proprio contributo.

A Mister X è capitato di svolgere dei colloqui di selezione: ha avuto a che fare con una grande varietà di persone, anche in termini di ruolo da ricoprire, dal neolaureato fino a ruoli senior. Normalmente si verifica subito la base tecnica che necessita il ruolo per cui si ricerca personale e si verifica che non manchi. Ciò che invece viene guardato con maggiore attenzione è l'approccio: Mister X cerca di capire se queste persone hanno un bagaglio di competenze trasversali in linea con quello che può essere utile al ruolo che andranno ad assumere in azienda. Predilige appunto le competenze trasversali rispetto a quelle tecniche, in proporzione del 70% e 30%.
Mister X non ha acquisito tramite corsi le competenze necessarie per diventare capo di un team: ciò che lo ha aiutato ad acquisire le competenze per sapersi relazionare è lo sport di squadra che ha praticato fin da piccolo. Grazie appunto all'esperienza passata, non ha sofferto una grande mancanza nell'approccio e nelle relazioni dirette.

La sua azienda investe molto sui corsi basati sullo sviluppo delle competenze trasversali. I corsi hanno una durata breve, di circa una settimana, e sono prevalentemente in aula, costituiti da una parte teorica e un insieme di esercitazioni che permettono di creare momenti interattivi. Mister X ritiene che un corso, effettuato prima di immettersi nel mondo del lavoro, sarebbe stato utile per acquisire le basi perlomeno della comunicazione. Ha partecipato in seguito al corso di Engineering Empathy nella versione serale rilasciata dalla medesima insegnante: in questo contesto, analisi transazionale e autoanalisi sono state per lui fondamentali e ritiene che lo sarebbero state anche prima di entrare nel mondo del lavoro. Allo stesso tempo, reputa che aver affrontato il corso di Engineering Empathy in questo momento, in cui possiede esperienze acquisite sul lavoro, abbia giovato molto, perché ha potuto confrontarsi portando in campo esempi concreti e realmente vissuti.

Capitolo 3

Analisi teorica

3.1 L'empatia

Il presente capitolo mira ad analizzare da un punto di vista teorico tutti gli aspetti che concernono le competenze trasversali in maniera approfondita. Tutte le abilità, tuttavia, derivano da quella che è la madre delle capacità, senza la quale nessuna di esse potrebbe esistere: **l'empatia**, che fonda le sue radici sull'ascolto attivo.
Se da una parte l'empatia rappresenta il presupposto dal quale nascono tutte le altre competenze, dall'altro rappresenta anche il fine, perché ogni persona intenzionata a sviluppare e migliorare le proprie competenze trasversali e relazionali ha l'obiettivo di imparare a conoscere meglio se stessa, sapersi esprimere in maniera autentica ed efficace e, infine, comprendere i punti di vista altrui col fine di poter sentirsi connesso con le persone e creare relazioni solide e durature con il prossimo. A supporto di questo, lo psichiatra robert waldinger ha riportato in una conferenza TED [2] il risultato di uno studio di Harvard sulla felicità durato 75 anni, monitorando la vita di 724 persone: ciò che rende davvero felici e sane le persone sono le relazioni buone, solide e durature. Oltre a questo risultato, ha esposto anche quelle che sono le tre conclusioni principali che si possono trarre da questo lunghissimo studio:

- se da un lato la solitudine è un'esperienza tossica per le persone e le uccide, dall'altro le connessioni sociali (famiglia, comunità, amici e così via) le rendono più felici e sane dal punto di vista fisico, portandole a vivere più a lungo. Sentirsi sconnessi dal punto di vista sociale ed essere isolati porta a un peggioramento della salute prima del previsto, il cervello deteriora prima del dovuto, sono meno felici e conducono vite più brevi

- non conta il numero delle amicizie e se si vive una relazione particolare stabile, bensì la qualità dei rapporti più stretti: ad esempio, un matrimonio pieno di litigi e privo di affetto risulta essere più nocivo rispetto a un divorzio. Vivere relazioni con calore fa sentire le persone protette

- le relazioni di qualità proteggono non solo il corpo ma anche il cervello. Ad esempio, avere una relazione stabile in età avanzata fa sentire le persone protette, se si hanno relazioni in cui si sa di poter contare sull'altro in caso di bisogno si mantiene una buona memoria più a lungo, anche se ci sono litigi sovente

Le persone che sono risultate essere le più felici in questo studio sono quelle che hanno messo più sforzo per rendere i colleghi amici e quelle che avevano investito più tempo nella loro vita per creare relazioni importanti.
Quindi, riassumendo, entrare in empatia e connetterci a pieno con chi sta intorno a noi dovrebbe rappresentare - e rappresenta - anche il fine ultimo.

L'empatia non è compassione, pietà o imitazione degli stati d'animo altrui.
L'empatia rappresenta la capacità innata nell'essere umano di *mettersi nei panni dell'altro*, di immergersi nella realtà altrui per comprenderne emozioni, pensieri, sentimenti e comportamenti conseguenti; permette di riconoscere e capire a pieno lo stato d'animo dei nostri interlocutori.
L'empatia si basa sul concetto di ascolto attivo: si donano il proprio tempo e le proprie energie con disponibilità e vero interesse nei confronti dell'interlocutore. Si mettono da parte il proprio ego e i propri pensieri, col fine di focalizzarsi sull'ascoltare a pieno ciò che l'altro ha da trasmetterci.

Per quanto sia una capacità innata, sovente le esperienze di vita tendono a limitare la volontà e l'abilità di entrare in empatia con le persone e di sentirsi veramente connessi con loro. Vi sono vari tipi di empatia:

- *comportamentale:* comprendere i comportamenti degli altri e le cause che li scatenano
- *emozionale:* comprendere le emozioni provate dagli altri
- *cognitiva:* comprendere valori, credenze, prototipi cognitivi e strutture mentali del prossimo
- *relazionale:* comprendere le mappe delle relazioni affettive

Il proseguimento del capitolo è mirato a:

1. identificare un percorso, dal punto di vista teorico, per riuscire a ritrovare e/o sviluppare l'abilità di entrare in empatia con il prossimo. Il procedimento, che ricorda e rispecchia il percorso analizzato da Daniel Goleman in [3], consiste in tre passi:
    - effettuare un percorso di consapevolezza su se stessi tramite l'analisi transazionale (AT)

- imparare a comunicare in maniera autentica i pensieri, le emozioni, i sentimenti e i bisogni presenti nel qui e ora all'interno della persona tramite la comunicazione nonviolenta (CNV)
- imparare a comprendere ciò che è vivo all'interno dei nostri interlocutori in maniera tale da riuscire a trovare la giusta strada da percorrere insieme per rendere la vita meravigliosa per entrambe le parti: questo avviene tramite l'ascolto attivo e la consapevolezza e le capacità apprese grazie all'AT e la CNV

2. presentare i concetti e le più rilevanti competenze trasversali, per la figura del neolaureato ingegnere, che si possono sviluppare a partire dall'empatia, le cosiddette competenze trasversali: leadership, public speaking e la creatività legata al problem solving

3. presentare l'ACT, che può in prima battuta essere un ottimo alleato per superare alcune insicurezze

3.2 Analisi transazionale (AT)

L'analisi transazionale (AT) [6] è una teoria della personalità e fornisce una rappresentazione della struttura del nostro punto di vista psicologico, mettendo l'accento sia sul singolo sia sul singolo in relazione con il prossimo e con il gruppo. L'AT fornisce, inoltre, una teoria della comunicazione, dello sviluppo infantile, della psicopatologia; può essere utilizzata come psicoterapia. sistematica per la crescita e lo sviluppo della persona e viene utilizzata molto anche a livello di counseling, soprattutto in contesti educativi. È opportuno sottolineare fin da subito quelle che sono le premesse dell'AT in termini di principi base:

- in termini di essenza, ogni individuo è OK ed è dotato di valore e dignità. Io accetto il prossimo per quello che è: posso non essere d'accordo su ciò che fa e sui suoi comportamenti, ma la sua essenza in quanto essere umano è per me OK. Vi è parità di livello, nessuno è superiore a qualcun altro, nonostante l'enorme differenza tra ogni individuo.

- ogni individuo, a meno di gravi danni, possiede le capacità necessarie per pensare, pertanto ha la piena responsabilità di ciò che vuole dalla vita; si porterà sempre dietro le conseguenze delle proprie decisioni

- ogni persona decide il proprio destino tramite le proprie decisioni, che può cambiare nel corso del tempo tramite un processo di consapevolezza che porta a una decisione attiva per cambiamenti reali e duraturi

3.2.1 Gli stati dell'io

Una colonna portante dell'analisi transazionale è il modello degli stati dell'io che ci permette di capire come funzioniamo e come esprimiamo la nostra personalità tramite comportamenti ben definiti: questo processo prende il nome di analisi strutturale.

Lo stato dell'Io è stato definito come una combinazione di emozioni e di esperienza direttamente collegata a uno schema uniforme di comportamento: secondo Berne, infatti, quando siamo in contatto con le emozioni e le esperienze che definiscono un certo stato dell'Io, assumeremo comportamenti collegato a quello stato.
Gli stati dell'io sono: Genitore (G), Adulto (A), Bambino (B). Un individuo si trova nello stato dell'Io Genitore quando i suoi comportamenti (C), i suoi pensieri (P) e le sue emozioni (E) sono "copiati" dai genitori o dalle figure genitoriali; nello stato dell'Io Adulto quando C-P-E sono una risposta diretta al qui e ora sfruttando le capacità di persona adulta; nello stato dell'Io Bambino quando C-P-E riproducono situazioni di quando si era bambini. Questo modello strutturale è di primo ordine e ogni individuo necessita di tutti e tre gli stati dell'Io per una personalità equilibrata: abbiamo bisogno dell'Adulto per trovare soluzioni ai problemi nel qui e ora, abbiamo bisogno del Genitore per essere in grado di adattarci alla società e rispettarne le regole e, infine, abbiamo bisogno del Bambino per sprigionare la spontaneità, la creatività e l'intuizione presenti in noi fin dall'età infantile.

Gli stati dell'Io possono essere analizzati anche da un punto di vista funzionale e non strutturale. Il modello funzionale classifica i comportamenti osservati, quindi il contenuto; il modello strutturale, invece, si occupa di classificare i ricordi e le strategie presenti nella memoria, quindi il processo. Pertanto, se si vogliono analizzare le interazioni tra le persone si utilizza il modello funzionale, si utilizza invece il modello strutturale per analizzare che cosa avviene all'interno di una persona. Il modello funzionale, quindi, risponde alla domanda: in che modo utilizziamo questi stati dell'Io?
Lo stato Adulto rimane tale, ma lo stato del Bambino e del Genitore possono essere divisi in due sottocategorie. Lo stato del Bambino può essere suddiviso in:

- *Bambino Adattato (BA):* si ripropongono comportamenti che avevamo adottato per adeguarci o ribellarci a ciò che i genitori si aspettavano da noi
- *Bambino libero (BL):* comportamenti non censurati che hanno derivazione infantile (termine utilizzato non in senso dispregiativo)

Lo stato del Genitore può essere suddiviso in:

- *Genitore Normativo (GN):* comportamenti copiati dai genitori quando tentavano di controllarci e criticarci

- *Genitore Affettivo (GA):* si ripropongono comportamenti di quando i genitori si prendevano cura di noi.

Entrambe le suddivisioni per lo stato Bambino e Genitore possono essere ulteriormente suddivise in positivo e negativo.

Per comprendere in quale stato si trova una persona, è possibile analizzarne i comportamenti nel qui e ora. Per analizzare l'importanza che detiene ciascuna delle parti funzionali, è possibile utilizzare uno strumento chiamato egogramma.

Sovente, la distinzione dello stato in cui si trova una persona non è di facile lettura perché potrebbe comportarsi in un modo pur vivendo all'interno qualcos'altro. È possibile, perciò, effettuare una distinzione tra il Sé esecutivo e il vero Sé: nel primo caso si tratta dello stato che detiene il potere esecutivo in un dato momento, nel secondo caso lo stato in cui una persona sta vivendo se stessa. Si ha un'incongruenza quando il comportamento non coincide con quelli che attuerebbe il vero Sé: ad esempio, ci stiamo annoiando ad una lezione e diamo spazio al nostro interno al Bambino Libero, mentre all'esterno assumiamo atteggiamenti da persone adulte che mostrano attenzione e interesse alla lezione.

È possibile, inoltre, che in un dato momento non sia possibile distinguere lo stato dell'Io di una persona poiché vi è un miscuglio: si ha contaminazione quando vi è una sovrapposizione degli stati G-A o A-B, doppia contaminazione in presenza di entrambi.
Si ha, infine, esclusione quando viene escluso un particolare stato dell'Io. Se si ha una doppia esclusione, ci si ritrova in uno stato costante. L'esclusione non è mai totale ma si riferisce a momenti e situazioni precisi.

3.2.2 Le transazioni

Ogni volta che si presenta una qualche forma di comunicazione tra 2, o più, persone si hanno delle transazioni: ad ogni stimolo transazionale (S) corrisponde una risposta (R) transazionale. Le conversazioni in generale possono essere viste come una catena di transazioni: un individuo per primo fornisce uno S, da parte dell'altro individuo vi è una R che rappresenta il nuovo S per il primo individuo.

Le transazioni possono essere:

- *complementari:* i vettori transazionali sono paralleli

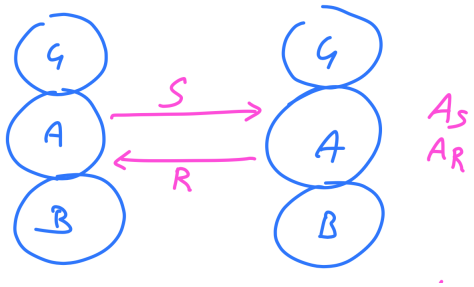

Figura 3.1. Transazione parallela

- *incrociate:* i vettori transazionali non sono paralleli

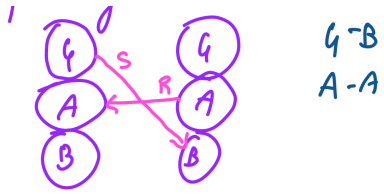

Figura 3.2. Transazione incrociata

- *ulteriori:* vengono inviati due messaggi contemporaneamente, uno a livello sociale (di solito in maniera verbale) e uno a livello psicologico (di solito in maniera non verbale). Il messaggio a livello sociale è rappresentato dalle parole utilizzate e vi è un'incoerenza con il messaggio trasmesso a livello non verbale. Le transazioni ulteriori possono essere duplici o angolari:

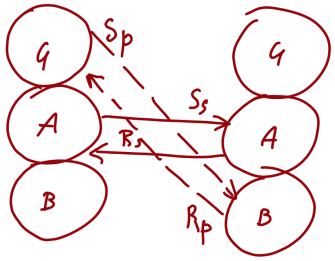

Figura 3.3. Transazione ulteriore duplice

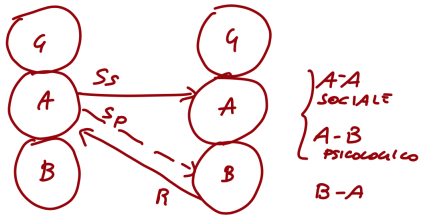

Figura 3.4. Transazione ulteriore angolare

Per analizzare una comunicazione, pertanto, è fondamentale osservare sempre i segnali non verbali, che coincidono con il messaggio verbale tranne nelle transazioni ulteriori. Da questi tre tipi discendono tre fondamentali regole di comunicazione, in corrispondenza ordinata con i tipi elencati precedentemente:

1. Con le transazioni complementari, la comunicazione può continuare indefinitamente
2. Con le transazioni incrociate, si ha un'interruzione della comunicazione (a meno che uno dei due individui cambi stato dell'Io per ristabilirla)
3. I comportamenti che ne derivano dipendono dal livello psicologico e non da quello sociale

In particolare, si ha un'incongruenza quando i messaggi trasmessi dalle parole sono contraddittori rispetto all'aspetto non verbale della conversazione.

3.2.3 Le carezze

Ogni individuo presenta un bisogno di stimolazione fisica o mentale, indicato da Berne come un bisogno di stimoli. Una carezza può essere definita come l'unità fondamentale del riconoscimento, ovvero rappresenta tutto ciò che in uno scambio comunicativo tra due persone comporta una certa dose di emozione in una o entrambe le persone, a prescindere dalla piacevolezza. Le carezze sono proprio il mezzo tramite il quale gli individui possono dare e ricevere questi stimoli. Le carezze possono essere:

- *verbali o non verbali:* a seconda che si tratti di una carezza trasmessa tramite le parole oppure tramite altri mezzi, come ad esempio un abbraccio oppure un cenno col capo. In genere le carezze comportano una certa dose di entrambi i tipi
- *positive o negative:* chi riceve la carezza la vive in maniera piacevole o spiacevole?
- *condizionate o non condizionate:* sono condizionate se si riferiscono a ciò che una persona fa/ha fatto, viceversa se si riferiscono a ciò che una persona è
- *interna o esterna:* la carezza proviene da me stesso o da terzi?

Un concetto fondamentale da sottolineare è che ogni individuo preferisce andare alla ricerca di carezze negative piuttosto che non ricevere stimoli del tutto. Le carezze false sembrano positive inizialmente ma alla fine danno una connotazione negativa.

L'economia delle carezze, concetto elaborato da Steiner, consiste di cinque regole che sono state innescate in tutti noi dai nostri genitori:

1. non dare carezze
2. non chiedere carezze
3. non accettare carezze
4. non rifiutare carezze
5. non donare carezze a te stesso

Esse ci sono state impresse dai genitori affinché potessero avere un alto grado di controllo nei nostri confronti: essendo le carezze in questo modo un bene scarso, il prezzo può essere definito di conseguenza dal genitore, che può così controllarci e a farci comportare come desidera.

3.2.4 Strutturazione del tempo

Quando gli individui si relazionano in coppia o in gruppo, soddisfano il proprio bisogno di strutturazione del tempo in sei particolari modi elencati sotto (l'intensità e l'imprevedibilità delle carezze e il grado di rischio psicologico aumentano man mano):

- *isolamento*: una persona può essere fisicamente presente nel gruppo ma si isola, rivolgendo l'attenzione magari al suo interno oppure immaginando altro. Le uniche carezze che può dare e ricevere vanno/arrivano a/da se stessa. Mentre si isola può accedere a qualsiasi stato dell'Io e per gli altri può essere difficile capire quale per via della possibile mancanza di segnali esterni. Evita così il rischio psicologico del Bambino di essere rifiutato.
- *rituali*: tipiche interazioni sociali di rito, come ad esempio la ben nota domanda "Come stai?". I riti vengono imparati fin da bambini, dalla propria famiglia, in maniera conforme alla cultura di riferimento. Per svolgere i rituali ci si trova nello stato del Bambino Adattato, nonostante siano programmati dall'Io Genitore. I riti forniscono carezze positive molto familiari.
- *passatempi*: il contenuto del passatempo non è invece rigido e programmato, si comincia a parlare di un qualcosa senza intraprendere azioni relative ad esso. Sovente si tratta di argomenti del passato recente o remoto invece che del qui e ora. Ad esempio gli uomini potrebbero parlare della partita del giorno prima o dello sport in generale. In questo modo si può esplorare l'altro e capire che tipo di "partner" potrà essere per un eventuale scambio di carezze più intense nei giochi o nell'intimità. Ci si può trovare sia nello stato G sia nello stato B.

- *attività*: la comunicazione si rivolge al raggiungimento di uno scopo comune e non al semplice chiacchierare riguardo ad esso. Le forze e l'energia vengono impiegate nello svolgere l'attività di riferimento nello stato predominante dell'Io Adulto. Le carezze possono essere sia positive condizionate sia negative condizionate, sovente a seconda del fatto che l'attività sia svolta bene o male.

- *giochi*: riproposizione di strategie che si utilizzavano durante l'infanzia e che non sono più utili alla persona adulta; comportano una serie di svalutazioni. I giochi tipicamente sono uno scambio di transazioni che terminato con una sensazione di malessere e cambiamento di ruolo, in cui la persona finisce stupita per chiedersi cosa sia effettivamente accaduto; inizialmente vi è uno scambio di carezze sia positive sia negative, ma alla fine solamente negative. Gli stati predominanti sono il GN-negativo, il GA-negativo e il BA-negativo.

- *intimità*: qua è dove le persone esprimono se stesse in maniera autentica: vengono comunicate le vere emozioni provate e i desideri senza censurare nulla, non vi sono messaggi nascosti. Le intenzioni sono chiare e il fine è sempre costruttivo. Sovente l'intimità viene sostituita dai giochi perché le persone percepiscono - paradossalmente, perché è ben chiaro che non sia così - un minor grado di rischio psicologico. Si possono scambiare carezze sia positive, vissute in maniera davvero piacevole e gratificante, sia negative ma non vi è mai svalutazione.

3.2.5 I copioni

Il copione rappresenta la storia che ognuno di noi ha scritto della propria vita in maniera completamente univoca, una sorta di vero e proprio piano specifico di vita. Si comincia a scriverlo subito dopo la nascita e già all'età di circa 7 anni abbiamo scritto i dettagli più rilevanti, mentre negli anni successivi affiliamo qualche ritocco: la maggior parte delle persone non è consapevole di aver scritto questione copione né di che cosa tratti.
Il copione viene scritto perché il bambino, posto in una posizione di inferiorità per via della sua piccolezza e vulnerabilità fisica, deve/vuole scegliere e adottare una serie di strategie per sopravvivere in un mondo fatto di minacce, in maniera tale soddisfare i propri bisogni: desidera ottenere amore e attenzione incondizionati. Il copione è rinforzato dai genitori, tramite i cosiddetti messaggi di copione, e da eventi esterni ma si basa su decisioni prese già dall'infanzia. Il finale, ovvero la scelta decisiva che rappresenta la scena finale, viene chiamato tornaconto; tutte le scene dall'inizio fino alla fine non sono altro che azioni programmate per raggiungere il finale prescelto. Le decisioni si basano sulle emozioni provate e sullo specifico tipo di esame di realtà del bambino. Sovente, da adulti, non si fa altro che interpretare la realtà in maniera che appaia secondo quando descritto dal copione, in modo tale

da non sentir minacciati i nostri bisogni o la sopravvivenza stessa.

I copioni nella realtà sono un miscuglio ma possono essere approssimativamente classificati in:

- *vincitore*: l'obiettivo dichiarato, qualunque esso sia, viene raggiunto in maniera felice, agevole e fluida.
- *perdente*: l'obiettivo dichiarato, qualunque esso sia, non viene raggiunto oppure viene raggiunto ma si è insoddisfatti. Si può distinguere, a seconda della gravità, un copione di primo, secondo o terzo grado.
- *non vincitore*: la persona non ottiene né una grande vittoria né una grande sconfitta, punta a sopravvivere, ad esempio quasi come se a scuola si puntasse sull'avere la sufficienza in tutto ma nulla di più e nulla di meno.

In particolare, il copione - o semplici aree di esso - può essere cambiato consapevolmente. Si dice che si è nel copione quando si attuano strategie che scegliemmo da bambini e questo spiega come, sovente, le persone adottino comportamenti che sono sofferenti, autodistruttivi e autolesionisti: queste decisioni, quando eravamo piccoli, parevano essere le migliori strategie per sopravvivere e vedere soddisfatti i propri bisogni ma non sono proprie dell'adulto e non risultano pertanto essere appropriate per risolvere i problemi nel qui e ora.

Un argomento strettamente correlato ai copioni sono le posizioni di vita, ovvero delle convinzioni che si costruisce il bambino riguardo se stesso e le persone che lo circondano:

- io non sono OK, tu non sei OK
- io sono OK, tu non sei OK
- io non sono OK, tu sei OK
- io sono OK, tu sei OK

3.3 Comunicazione nonviolenta (CNV)

La Comunicazione NonViolenta (CNV) [7] è un particolare tipo di comunicazione, ideata da Marshall Bertram Rosenberg nel 1960 circa e chiamata alternativamente comunicazione empatica; ha come obiettivo quello di rendere trasparente e autentica la comunicazione tra due o più persone e di creare una connessione molto più profonda tra le parti, in maniera tale che non si creino conflitti oppure che vengano

risolti efficacemente con autenticità.

Per comprendere a pieno cosa sia la CNV, è necessario prima analizzare anche le ragioni per cui la strada verso l'empatia viene bloccata, secondo Rosenberg: giudicare e analizzare, classificare e dire cosa sia assolutamente giusto o sbagliato, incolpare sempre gli altri (o anche noi stessi) e non assumersi le proprie responsabilità. Rosenberg sottolinea come la comunicazione che blocca l'empatia, ovvero la comunicazione violenta che aliena la vita, possa essere vista secondo le 4 "A" che egli ha identificato:

- *Analizzare:* effettuare giudizi implica idee su cosa sia giusto o sbagliato, analisi che classificano la realtà e creano dicotomie, portando a credere che il prossimo sia dalla parte del torto o agisca con cattiveria se i suoi comportamenti non seguono i nostri valori. Tutto questo è alla base della violenza, che sia verbale o fisica, in quanto impedisce di pensare a se stessi e al prossimo in termini di vulnerabilità

- *Assenza di responsabilità:* vi è una completa negazione delle proprie responsabilità, in termini dei sentimenti che si provano, dei propri bisogni, pensieri e comportamenti. Questo avviene solitamente attribuendo la responsabilità a fattori esterni, come ad esempio forze vaghe, la propria condizione o le pressioni esterne, i pensieri e i comportamenti degli altri e così via

- *Avanzare pretese:* pretendere che gli altri facciano un qualcosa implica una minaccia, sia essa implicita o esplicita; se il prossimo non agisce secondo le nostre pretese verrà punito in qualche modo, spesso anche attraverso il senso di colpa, rendendolo non contento di fare ciò che noi vogliamo che lui faccia

- *Assegnare meriti o colpe:* credere che ci sia sempre qualcosa di giusto e qualcosa di sbagliato porta anche ad assegnare meriti, magari proprio a noi stessi che seguiamo i nostri valori, e colpe, magari proprio al prossimo che agisce secondo i propri valori

La CNV pone le sue fondamenta su alcuni concetti e assunzioni molto semplici da comprendere ma allo stesso tempo molto profondi. Innanzitutto la CNV premette che l'empatia sia una caratteristica innata nell'essere umano, mentre i comportamenti, gli atteggiamenti e le strategie violente sia un qualcosa che viene appreso, nel tempo, dalla cultura che ci circonda. La CNV sposta il focus su che cosa è presente nel qui è ora dentro di me e dentro del prossimo: alla luce di ciò, il focus finisce su che cosa potrebbe rendere la vita meravigliosa. Per poter rendere la vita meravigliosa, bisogna imparare ad esprimersi con autenticità e onestà. Ciò che ci trattiene dal farlo, di solito, è un misto di pensieri: crediamo di essere responsabili di come si sente l'altro, temiamo la sua reazione e di non essere in grado di gestirla, crediamo che comunicare onestamente possa essere considerato un giudizio

nei confronti del prossimo.
La CNV assume, inoltre, che tutte le persone provino piacere e soddisfazione quando contribuiscono liberamente alla propria vita ma, soprattutto, anche a quella degli altri; la connessione tra noi e il prossimo è una risorsa di fondamentale importanza per il nostro benessere.
Tutte le persone hanno dei bisogni, i quali sono completamente universali, ovvero che potenzialmente vivono dentro tutti noi, e in ogni momento cercano di soddisfarli nel modo migliore che conoscono e che è presente nel proprio repertorio; la comunicazione violenta, di cui si è trattato precedentemente, ha origine proprio dal mancato soddisfacimento di uno o più bisogni e non è altro che una tragica espressione dell'insoddisfazione degli stessi. In particolare, Rosenberg sottolinea come sia più facile trovare la strada per rendere la vita meravigliosa quando si parla di bisogni, riuscendo al contempo a soddisfare i bisogni di tutte le parti coinvolte.

I quattro passi della CNV verso l'onestà, che si basano sull'ascolto di se stessi, sull'ascolto del prossimo e sull'espressione autentica di ciò che si sente e si ha bisogno, sono:

- *Osservazioni:* il primo passo consiste proprio nell'osservare ciò che ci circonda, ciò che è vivo in noi e nel prossimo, perché è proprio ciò che osserviamo ad influenzare il nostro benessere, ovvero come stiamo nel qui e ora. L'osservazione è ciò che fornisce uno spazio per prendersi cura dei bisogni e per poter veramente comunicare con il prossimo; non è altro che l'esatto opposto del giudizio, dal quale non può che nascere una reazione nel prossimo che si sente attaccato ed è pronto a contrattaccare a sua volta. Ecco alcuni esempi di giudizio:

  – *Oggi Paolo si è arrabbiato senza alcun motivo*
  – *Dovresti fare una pausa perché lavori sempre troppo*
  – *Sei un vero egoista*

  Che dovrebbero essere espressi sotto forma di osservazioni:

  – *Oggi ho visto Paolo arrabbiarsi con il collega perché non ha portato a termine il compito che gli aveva affidato*
  – *Ho notato che questa settimana hai lavorato più di otto ore tutti i giorni, forse potresti prenderti una pausa per riposarti*
  – *Tra tutte le riunioni che abbiamo fatto, ricordo che tu mi abbia dato la possibilità di esporre la mia opinione*

  Le osservazioni partono dal fenomenologico, ovvero da circostanze ben precise e definite nello spazio e nel tempo e non forniscono giudizi o analisi personali

sulle situazioni osservate. Ciò che ci accade sarà lo stimolo per i sentimenti che proviamo ma non ne sono assolutamente la causa

- *Sentimenti:* il secondo passo consiste nel saper riconoscere i sentimenti che proviamo nel qui e ora. Siamo abituati ad avere poco contatto con noi stessi. Si pensi alla "banalissima" domanda che probabilmente riceviamo più spesso tutti i giorni: come stai? Rispondiamo sempre "bene", di getto, senza neanche pensarci, per abitudine, senza neanche provare ad ascoltare ciò che abbiamo dentro di noi. Molto spesso non siamo neanche in grado di riconoscere che cosa siano veramente i sentimenti, perché tendiamo a confonderli con i pensieri. Un classico esempio di ciò è "mi sento incapace", in realtà questo non esprime un vero sentimento, bensì il pensiero di essere incapace: è importante sapere distinguere le due cose e, per riprendere l'esempio, saper comprendere qual è il sentimento che sta alla base del pensiero di pensare di essere incapaci. Capita tantissime volte di utilizzare la parola "sento" e poi esprimere in realtà un pensiero; sento di essere un fallimento, sento di essere stata manipolata, sento di essere sempre in ritardo e di fretta, in realtà, non sono altro che espressioni che indicano un pensiero su noi stessi e non il sentimento che stiamo realmente provando.
  Per riuscire a riconoscere i sentimenti che proviamo nel qui e ora, è fondamentale anche avere un vocabolario sufficientemente ricco per saper esprimere ciò che sentiamo. Ecco una lista di parole che indicano realmente i sentimenti: Ecco altri esempi di frasi che non esprimono sentimenti, bensì dei pensieri:

  – *Mi sento di non essere mai compreso*
  – *Mi sento che hai fatto bene a liberarti per venire al mio matrimonio*
  – *Sento che ormai non mi ami più*

  Per concludere, si riportano degli esempi in cui vengono veramente espressi dei sentimenti:

  – *Mi sento frustrato perché noto che ciò che dico non è mai considerato da voi un'idea su cui lavorare tutti insieme*
  – *Sono davvero felice che tu sia riuscito a trovare il modo di essere presente al mio matrimonio*
  – *Mi sento triste da quando hai deciso di dedicare meno tempo al nostro rapporto*

  Una volta riconosciuti i sentimenti che si provano, bisogna accettarli e, soprattutto, assumersi la responsabilità, cercando di capire perché li si provino, guardando ai bisogni.

- *Bisogni:* i bisogni sono l'energia che ci guida, in maniera consapevole o inconsapevole, all'azione: sono la radice dei sentimenti in quanto il sentimento, che si prova dopo aver ricevuto uno stimolo, dipende solo esclusivamente dal bisogno presente in quel momento, sia che il bisogno sia soddisfatto sia che non lo sia. Il sentimento che proviamo a seguito di uno stimolo, pertanto, deriva solo ed esclusivamente dal bisogno che abbiamo in quel momento: in caso di bisogno soddisfatto, proveremo un sentimento "positivo", "negativo" in caso non sia soddisfatto. Anche in questo caso diventa fondamentale riconoscere e, soprattutto, accettare il bisogno che abbiamo. Qui è dove solitamente nasce la violenza, perché non riconosciamo, neghiamo o attribuiamo a forze esterne il nostro bisogno interiore, finendo per giudicare. Come già citato in precedenza, i bisogni sono universali per tutti e di seguito ne troviamo una lista sufficientemente esauriente Ecco alcuni esempi di quando, dato uno stimolo, si riconoscono, accettano e comunicano efficacemente i propri sentimenti derivati dai bisogni vivi nel qui e ora:

  - *Mi sento davvero triste quando dici che mi vuoi abbandonare da un giorno all'altro perché io ho bisogno di sicurezza e sapere che posso contare su di te*
  - *Sono davvero grato che tu mi abbia aiutato con il mio progetto perché avevo bisogno di un po' di pace e intimità e, grazie al tuo aiuto, avrò a disposizione qualche giorno per rilassarmi un po' con la mia famiglia*
  - *Quando dici che le mie idee fanno schifo mi sento frustrato, perché lavoro sempre duramente per cercare di dare il mio meglio e ho bisogno di riconoscimento per i miei sforzi*

- *Richieste:* L'ultimo step della CNV è rappresentato dalle richieste. Arrivati sin qui, è necessario avere ben chiaro che cosa desideriamo chiedere all'altro per rendere la nostra vita più ricca, procedendo in maniera tale che anche gli altri siano disposti a rispondere con empatia, a partecipare e contribuire a soddisfare i nostri bisogni e rendere la vita meravigliosa per tutti.
  Per capire come effettuare una richiesta efficace, è importante comprendere che cosa non è una richiesta efficace: chiedere qualcosa senza sapere che cosa realmente desideriamo interiormente, chiedere all'altro qualcosa in maniera vaga e ambigua (cosa che spesso nasconde giochi interpersonali, come analizzato nell'AT), dire e pretendere che l'altro non faccia un qualcosa, non effettuare affatto la richiesta e fermarsi al terzo step (all'altro può sembrare una pretesa).
  Chiarito questo, è possibile comprendere come una richiesta in CNV sia espressa in un linguaggio chiaro, positivo, concreto e onesto, perché solo in questo modo è possibile essere autentici e rivelare al prossimo ciò che davvero desideriamo. Ecco alcuni esempi di richieste CNV:

- *Voglio avere le idee più chiare riguardo al nostro accordo, sei disposto a lasciarmi ancora un giorno per riflettere sulla situazione?*
- *Vorrei che tu mi dicessi onestamente che cosa non hai gradito di questa cena insieme*
- *Desidero che quando ci incontriamo in strada continuiamo a salutarci in maniera pacifica, sei disposto?*

Questi quattro passi sono anche l'esatta antitesi dei quattro che caratterizzano la comunicazione violenta: i giudizi, i pensieri, le strategie e, infine, le pretese. Rimettendo insieme tutti i prezzi, ecco alcuni esempi di espressioni oneste e autentiche secondo il modello della CNV:

- *Mi sento irritata quando arrivo a casa la sera tardi e vedo che non hai pulito i piatti del pranzo, perché ho bisogno di ordine, soprattutto quando arrivo molto tardi la sera e sono stremata. Per favore, puoi lavare i piatti quando finisci di mangiare?*

- *Sono davvero felice quando riusciamo a chiamarci e raccontarci quello che facciamo durante le nostre giornate perché ho davvero bisogno di condividerlo con il mio migliore amico. Ti andrebbe di sentirci più spesso, almeno una volta a settimana, magari proprio la domenica sera come abbiamo fatto oggi?*

- *Sono esitante ad andare a chiedere un aumento al capo, perché tutte le altre volte non ha accettato, mi ha anche urlato contro, e io ho davvero bisogno di riconoscimento per tutti i risultati che porto alla nostra azienda e questo mi rende triste. Ti andrebbe di ascoltare il mio discorso un paio di volte e dirmi se c'è qualcosa che miglioreresti?*

Infine, si potrebbe pensare che questo modo di interagire con il prossimo possa far reprimere anche la trasmissione di ciò che proviamo davvero, come ad esempio la rabbia. Rosenberg sostiene che in CNV è possibile trasmettere anche la propria rabbia in CNV, ma per farlo in maniera autentica, onesta ed empatica come analizzato finora, è necessario sollevare il prossimo da qualsiasi responsabilità circa la nostra rabbia e/o il nostro malessere; questo perché non è una frase o un'azione dell'altro a farci arrabbiare, non ne è affatto la causa, bensì è soltanto lo stimolo che richiama un nostro bisogno insoddisfatto. Così facendo è possibile non giudicare, assumersi le responsabilità dei propri bisogni e sentimenti scaturiti, trasmettendo al prossimo autenticamente ciò che abbiamo dentro. Per rendere tutto questo possibile, Rosenberg consiglia sempre di prendere un bel respiro in maniera tale di renderci conto quali giudizi potremmo stare per fare, per poi riconnetterci con i sentimenti provati derivanti dai nostri bisogni insoddisfatti nel qui e ora.

3.4 AT e CNV: guarigione verso l'autonomia e l'empatia

3.4.1 AT

Per autonomia si intende un comportamento, pensiero o emozione che è una risposta nel qui e ora e non a convenzioni del copione. Come se l'io adulto avesse incorporato caratteristiche di bambino libero e genitore affettivo.
Uscire dal copione consiste nel cominciare a sfruttare a pieno le potenzialità dell'adulto, invece di ricorrere a strategie infantili, per soddisfare le proprie esigenze.
Per *guarigione* dal copione si intende prendere consapevolezza delle scelte che si prendono ad oggi, che sono dettate dal copione, e imparare progressivamente a esercitare nuove scelte: è un processo, un lungo percorso a step che prevede anche passi all'indietro perché il copione rappresenta la cosiddetta *zona di comfort*. Allo stesso tempo, dal punto di vista della strutturazione del tempo, questo predispone all'apertura e alla condivisione di emozioni autentiche: permette pertanto di smettere di vivere le situazioni come dei giochi e di cominciare ad entrare in intimità e connessione con il prossimo.
Per poter uscire dal copione, è necessario decidere di intraprendere un percorso interiore che, tramite una presa di consapevolezza, permetta di decidere se continuare a seguire o uscire da questi schemi. Questo permette alle persone di decidere cosa volere dalla propria vita, senza che tutte le azioni quotidiane siano reazioni predeterminate da pagine scritte in un passato che ormai appartiene solo alla memoria.
Il percorso non è facile, non è semplice assolutamente, è una strada tortuosa, lunga e di cui forse è difficile addirittura vederne il tracciato e dunque la fine: ma perché è così complicato? È così complesso perché non si è consapevoli dell'esistenza del copione, né esattamente di cosa sia, finché non si intraprende un percorso che porta alla consapevolezza degli schemi scritti nello stesso.

La parte più difficile di tutte è rendersi conto dell'esistenza del copione e soprattutto anche realizzare che passiamo molto tempo della nostra vita a vivere secondo il copione. Una volta che si prende consapevolezza di questo, non resta che scegliere come vivere il presente: continuare ad agire come è stato scritto nel mio copione, o sfruttare a pieno tutte le risorse dell'Adulto che è in noi affrontando quindi ogni situazione, che la vita propone, con estrema consapevolezza? Questo è fondamentale, perché la decisione non può esser presa da terzi, ma solo e semplicemente dentro di noi.
Si può scegliere: continuare a vivere le relazioni come dei giochi, oppure si desidera entrare veramente intimità con il prossimo, creando un forte sentimento di connessione? Si preferisce continuare a reagire alla vita, oppure essere proattivi, entrare in contatto con il mondo interiore e condividere i pensieri e le emozioni autentiche? Questo è ciò che ci vuole per essere veramente in grado di entrare in empatia con

il prossimo. Sta solo ed unicamente a noi sceglierlo.

Infine: una volta che prendiamo consapevolezza del nostro copione, come ci si può comportare?
Ci si potrebbe arrabbiare, inveire contro se stessi o, più facilmente ancora, contro i genitori o gli eventi che sono capitati finora e capitano tuttora. O, alternativamente, ci si potrebbe addirittura assumere tutte le colpe e pensare che tutto ciò che ci è capitato fino ad oggi è colpa nostra. Prendersi le colpe è sicuramente un atteggiamento umile e che probabilmente rende le persone forti per un momento, ma non sempre è la scelta giusta da fare perché non sempre è colpa nostra, o solamente tutta nostra. Possiamo scegliere di comportarci in maniera completamente differente: invece che inveire e arrabbiarci, invece che addossarci colpe che non abbiamo, si può scegliere di fare la cosa migliore della vita: ci si può assumere la *responsabilità*. Assumersi la responsabilità vuol dire tirare fuori il *coraggio* e diventare consapevoli che si è veramente consapevoli di ciò che ci succede. Tirare fuori il coraggio non è da poco, ma se si è in grado di farlo, diventa possibile sfruttare l'Adulto che è in noi per affrontare la vita ed entrare veramente in intimità con le persone, creando un'empatia positiva che contagerà sempre più persone.
Non resta che *scegliere*: scegliere chi vogliamo essere e come vogliamo vivere, pienamente consapevoli di tutte le risorse che risiedono nel nostro Io Adulto per affrontare ogni situazione che la vita regalerà.

3.4.2 CNV

Siamo esseri speciali e dobbiamo imparare a sostituire la violenza contro noi stessi con l'empatia. Spesso durante la vita si imparano lezioni con energia di violenza, sensi di colpa e vergogna. In questo modo, si sta permettendo che l'apprendimento e la crescita siano motivati dall'odio perché si impara appunto per "senso" di dovere, senza focalizzarsi su quella che dovrebbe essere una scelta: insomma, si impara nel modo non corretto.
Le persone che continuano a pensare o a dirsi che cosa avrebbero dovuto fare, continueranno a non farlo perché è nella indole umana ribellarsi alla costrizione. I giudizi verso se stessi sono tragiche espressioni di bisogni insoddisfatti, quindi quando si sta facendo qualcosa che non arricchisce la propria vita, diventa fondamentale ridare valore a se stessi in modo tale da prendere la direzione che si desidera, motivati dal rispetto e dall'empatia verso di sé e non dall'odio, dal senso di colpa o dalla vergogna.
Se siamo stati imperfetti, relazioniamoci con i sentimenti e i bisogni insoddisfatti che si generano e che ci hanno portati lì. Il rincrescimento ci aiuta ad imparare ma senza odiarci. Dobbiamo imparare a perdonare noi stessi, ascoltandoci con empatia: Il fallimento è stato una scelta verso un tentativo che non è andato a buon

fine e fare scelte motivate dal desiderio di rendere meravigliosa la vita, anziché dettate dalla paura, dalla vergogna, dal senso di colpa, dal senso di dovere, dall'obbligo.

Diventa quindi fondamentale tradurre i "Devo" in "Scelgo" e riconoscere in se stessi che si fanno determinate azioni perché si sceglie di farle. Ci vuole tempo e sforzo per imparare a essere sinceri con se stessi, ma si scoprirà l'energia dietro le azioni, perché ci sono valori importanti dietro le azioni.
Infine, se si riesce a liberarsi dai conflitti interiori, se ne si diventa consapevoli entrando in empatia con se stessi, diventa possibile liberarsi dal dovere e dal condizionamento. È possibile concentrarsi su ciò che si desidera fare e non su ciò che è andato storto, sostituendo, al linguaggio che uccide i sogni, un linguaggio che rende onesti, genuini e autentici, aiutando in questo modo se stessi e gli altri.

3.5 Empatia, lavoro in team e leadership

Leadership è un termine inglese che deriva da *to lead*, che significa *condurre*, *guidare*. Pertanto, con il termine leadership ci si riferisce a una competenza trasversale che identifica la capacità di guidare un gruppo di persone verso il raggiungimento di determinati obiettivi: questo può avvenire in qualsiasi contesto, che sia lavorativo all'interno di un'azienda o sportivo all'interno di una squadra, o altro ancora.

Quando si parla di leadership, vi è un aspetto che è anche emerso dai risultati ottenuti dal sondaggio e dalle interviste: le persone hanno una forte predisposizione a credere che si tratti di una competenza che diviene necessaria man mano che si scalano le gerarchie all'interno di un'azienda/organizzazione, ovvero quando si arriva a ricoprire ruolo più gestionali, detti anche manageriali.
Durante il corso di Engineering Empathy, sono state fatte tante esercitazioni di gruppo: in ognuna di esse, veniva scelto preliminarmente un leader, ovvero un membro del gruppo che fosse incaricato di condurre l'esercitazione. Questa metodologia di lavoro è anche alla base del metodo di Toastmasters, la più grande organizzazione no profit mondiale che ha come obiettivo di rendere dei veri leader le persone e, allo stesso tempo, di migliorare le loro abilità comunicative e relazionali.

Simon Oliver Sinek, un insegnante della Columbia University, è diventato particolarmente celebre per aver ideato il concetto di *cerchio d'oro* esponendolo in [8] e in [9]. La seguente immagine [10] presenta graficamente quello che è concettualmente il cerchio d'oro:

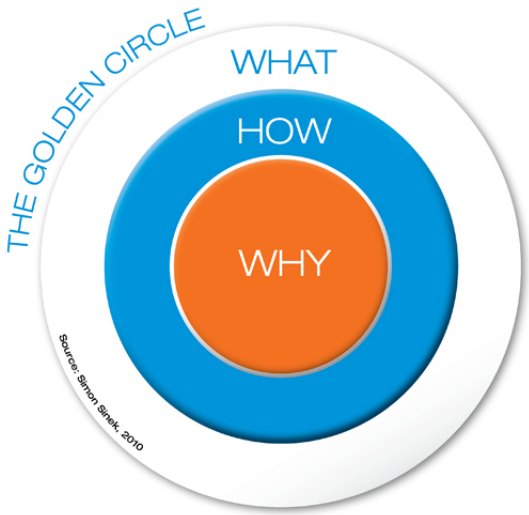

Figura 3.5. Il cerchio d'oro di Simon Sinek

Simon Sinek sostiene che i più grandi leader siano in grado di ispirare la gente perché comunicano nella sequenza esattamente opposta rispetto a tutte le altre persone: solitamente, le persone partono spiegando il che cosa, poi il come e poi il perché (sempre che arrivino anche ad esternare quest'ultimo); al contrario, i grandi leader procedono in maniera inversa, partono dal centro del cerchio comunicando fin da subito il perché, per poi procedere con il come e, solamente alla fine, spiegare il cosa. Un esempio riportato da Simon Sinek è quello relativo all'azienda Apple. Un ipotetico competitor esporrebbe un messaggio di marketing di questo genere:

"Facciamo computer fantastici. Sono ben progettati, semplici da usare e intuitivi. Ne volete comprare uno?" [9]

Mentre la Apple procederebbe nell'esatto verso opposto:

In tutto ciò che facciamo, crediamo nelle sfide allo status quo. Crediamo nel pensiero alternativo. Sfidiamo lo status quo facendo prodotti ben progettati, semplici da usare e intuitivi. E quindi facciamo computer fantastici. Ne volete comprare uno? [9]

È chiaro che nel primo caso la comunicazione parte dall'esterno del cerchio per terminare al centro (anzi, non ci arriva nemmeno), mentre nel secondo caso si parte dal centro per arrivare all'esterno. Il risultato è che le persone si sentono profondamente ispirate. E, risultati alla mano, questo tipo di comunicazione ispira davvero le persone, perché è ben noto come tecnicamente i prodotti della Apple possano essere meno performanti e più costosi rispetto a quelli dei competitor ma, nonostante ciò, le persone fanno la fila fuori dai negozi Apple ogni volta in cui viene

effettuato il lancio di un nuovo iPhone.
Si sottolinea in particolare come questa struttura rispecchi in pieno la sequenza dei tre cervelli [11] proposta negli anni '60 dal neurologo Paul MacLean ([12]:

- *il cervello rettile:* corrisponde alla parte più antica che reagisce agli stimoli con una risposta del tipo scapa o combatti, e viene ingaggiato dal *perché*. È la parte di cervello che ha bisogno di fidarsi fin da subito: comunicando il perché, si va ad entrare in empatia con le persone

- *il cervello limbico:* è la parte del cervello legata alle emozioni e all'entusiasmo e viene sedotto dal *come*, che permette di entrare in uno stato di empatia ancora più grande

- *la neocorteccia:* è la parte più recentemente sviluppatasi nell'essere umano, è la parte più intelligente ed è legata alla logica; viene catturata dal *cosa*

Spesso la comunicazione, da un mero punto di vista linguistico, procede partendo subito utilizzando la logica. È anche per questo motivo che la maggior parte delle persone non riesce ad entrare in empatia con le persone. È impossibile riuscire a entrare in empatia con le persone se si vuole imporre la propria logica: è fondamentale prima guadagnarsi la fiducia del cervello rettile e rinforzare l'entrata in empatia legandosi emozionalmente al cervello limbico: solo a questo punto, è possibile esporre anche la propria logica, tramite il *cosa*. Si vuole far notare, infine, che questa struttura ideata nell'epoca moderna rispecchia anche il pensiero del lungimirante filosofo greco Aristotele sulla tripartizione per una comunicazione persuasiva: ethos (infondere credibilità e dimostrare somiglianza con l'interlocutore), pathos (evocare emozioni e sentimenti) e logos (ragionamento e argomentazioni logiche).

Simon Sinek propone anche la sua visione di leadership in [13], che rispecchia totalmente l'introduzione di questo capitolo: per essere davvero dei leader all'interno di un gruppo è fondamentale riuscire a entrare in empatia con le persone che lo compongono. Questo perché le aziende, le organizzazioni, i gruppi sono composti da persone e non da tabelle e numeri da rispettare. Questo approccio corrisponde anche ad uno dei feedback ricevuti dal sondaggio, in particolare l'ultimo riportato in corsivo nella sottosezione 2.1.1. Nella prefazione, Simon Sinek sottolinea come nella storia non sia mai stato il management a tirar fuori un'organizzazione da una crisi, a farlo, invece, sono sempre stati i leader. Nonostante questo, ad oggi si cerca di costruire dei grandi manager, invece che dei grandi leader. Non è sufficiente che i leader forniscano il perché, i leader sanno di avere a che fare con delle persone e non con dei numeri. La più grande differenza tra gestire (management) e condurre (leadership) è: per la gestione, si ha sempre un qualcosa di misurabile, una serie di obiettivi spesso proprio numerici da raggiungere e le persone diventano mezzi per raggiungere gli obiettivi preimpostati; per condurre, non si hanno metriche

di riferimento e tutto ciò che conta sono le persone, le relazioni che si instaurano con esse e tra di esse e il senso di sicurezza che si è in grado di infondere in loro. Ad oggi, spesso il comportamento che prevale nelle gerarchie è quello di cercare sempre di farsi notare per ottenere riconoscimento e approvazione dai superiori; in realtà, dovrebbe funzionare tutto nella maniera opposta, ovvero che ogni persona sia felice di mettersi al servizio e sacrificarsi per proteggere i membri, a qualsiasi livello della gerarchia. Questo spirito non si ottiene tramite educazione, corsi di formazione o altro, bensì attraverso un forte senso di connessione appartenenza ed empatia nei confronti degli altri membri. È proprio per questo motivo che i leader danno priorità alle persone e al loro benessere, perché solo così ogni singola persona potrà sentirsi protetta e fornire lo stesso tipo di protezione nei confronti degli altri membri; i leader, dando l'esempio di protezione dall'alto, sono in grado di ispirare tutte le persone a fare lo stesso con gli altri membri dell'organizzazione. L'unico modo che si ha a disposizione per fare tutto ciò è proprio l'empatia che si crea tra le persone. La vera leadership consiste nel mettere l'interesse di chi ci sta vicino dinnanzi ai propri e, allo stesso tempo, proteggerli.

Per fare un parallelismo con la sequenza dei tre cervelli, la leadership consiste nel fornire un forte senso di sicurezza (cervello rettile) e saper ispirare affinché si crei un gran senso di appartenenza (cervello limbico) a quella che è la visione del gruppo: a questo punto, qualsiasi implementazione, percorso, scelta logica saranno apprezzate dalla neocorteccia perché il *cosa* sarà supportato da forti *perché* e *come* in cui tutte le persone del gruppo si identificano a pieno.

Infine, si ritiene che il miglior esercizio per sviluppare la leadership sia quello di creare gruppi dinamici in cui si viene a scegliere ogni volta un leader incaricato di condurre il gruppo verso gli obiettivi dell'esercitazione: il metodo è molto simile al concetto di bubble, di cui si è parlato in un'intervista specifica, utilizzato in ambito risorse umane e, inoltre, analogo a quello che viene anche sfruttato nei club di Toastmasters: ad ogni incontro, infatti, ogni persona svolge un ruolo specifico e i ruoli vengono quasi sempre scambiati in maniera tale che le persone possano sia imparare a condurre sia imparare a lasciarsi guidare dal leader. Si possono trarre ulteriori benefici se si mantiene un diario di bordo per ogni esercitazione, sia a livello personale sia a livello di gruppo: oltre a quanto spiegato e descritto nella sezione apposita (3.7), è bene mantenere anche traccia dei sentimenti che si provano in base ai bisogni che vengono soddisfatti o meno a seconda delle situazioni (come spiegato nella sezione 3.3).

3.6 Ascolto attivo

Ascoltare veramente i nostri interlocutori non significa esprimere simpatia per ciò che ci viene comunicato o per la persona con la quale stiamo comunicando, né tantomeno fingere di essere interessati all'argomento o alla persona stessa. Per donare un vero e proprio ascolto attivo, è necessario essere autentici in ogni momento, assumendo un atteggiamento di vero interesse genuino e di apertura verso ciò che ci viene comunicato.

3.6.1 Scala dei livelli di ascolto

A seconda del livello di qualità dell'ascolto che si dà o riceve dall'interlocutore, è possibile distinguere una serie di livelli che vengono analizzati tramite La scala dei livelli di ascolto [14], che è composta da 8 stadi, disposti in maniera progressiva seguendo un ordine crescente rispetto all'ascolto che si pone quando l'interlocutore parla con noi. I primi quattro, decisamente di basso livello, sono:

- *Ascolto schermato/distorsivo/impreciso:* non si presta attenzione e non si comprende realmente ciò che viene comunicato. La realtà viene distorta o fraintesa da chi sta ascoltando, sia i fatti sia le emozioni provate, oscurando così i dati reali; non si capisce realmente ciò che viene comunicato in quanto non si presta sufficiente attenzione. Ci possono essere ostacoli di ogni tipo a schermare l'ascolto, come ad esempio pregiudizi culturali, stati emotivi alterati, antipatie già presenti in precedenza e così via. Gran parte dei segnali non verbali sono bloccati e si mantiene il focus su sé stessi e sui propri stati d'animo

- *Ascolto giudicante/aggressivo:* chi ascolta non si sforza di comprendere, interrompe molto spesso chi sta comunicando. Nonostante una scarsa attenzione, sente parte di ciò che viene comunicato e lo sfrutta per effettuare sentenze e giudizi, anche criticando quanto viene detto, non necessariamente in maniera verbale (ad esempio con smorfie di disappunto). Questo tipo di ascolto innesca aggressività e una serie di scambi comunicativi poco piacevoli

- *Ascolto apatico/passivo:* vi è totale disinteresse nei confronti di chi comunica. Chi ascolta è assente mentalmente, magari anche assorbito dai propri pensieri, non riuscendo pertanto a comprendere ciò che sente e non entrando in connessione con l'interlocutore; non si danno segnali né verbali né non verbali e si dimostra di essere spenti, svogliati e senza energie.

- *Ascolto a tratti:* l'ascolto procede a momenti alternati, si dona un'attenzione discontinua e una scarsa concentrazione, magari anche dovuta ad altri pensieri che si hanno per la testa, perdendosi parte di ciò che accade nel qui e ora. Questo è il tipo più frequente di ascolto che si dà e riceve quotidianamente:

ascoltiamo maggiormente le parti che ci interessano e in altri momenti ci si fa assorbire da pensieri esterni o altre azioni (ad esempio ascoltare mentre usiamo il cellulare). Per chi comunica è molto dispendioso, dal punto di vista di energie mentali ed emotive, avere a che fare con un interlocutore distratto

I successivi quattro invece sono di una qualità nettamente superiore:

- *Ascolto selettivo:* non si hanno particolare intenzioni empatiche, si dirige l'attenzione in particolare verso un unico canale sensoriale oppure un unico elemento, che sia un argomento della conversazione o una parte della comunicazione non verbale, per ottenere informazioni più precise, dimostrando perciò interesse verso quell'unico elemento e non sul resto, o comunque molto meno. In questo stadio, esistono alcune tecniche per favorire questo tipo di ascolto:
  - *Reflecting:* fare da specchio, magari riformulando ciò che si è appena compreso; questo serve principalmente per vedere se si è capito realmente, ma anche a favorire l'apertura verso il raccoglimento di informazioni più precise
  - *Deflecting:* tentare di riconoscere se vengono inserite informazioni superflue e che non hanno a che fare con ciò che si vuole trasmettere, in maniera tale da escluderle dalla conversazione e concentrarsi su ciò che è più importante
  - *Probing:* si tratta di testare le informazioni recapitate ponendo domande collegate, serve anche a raccogliere informazioni più precise e approfondite
  - *Recap:* ricapitolare e riassumere ciò che è stato detto per poi continuare a ricevere nuove informazioni
  - *Contatto:* fornire contatti visivi e cenni del capo/corpo che dimostrano vicinanza e comprensione

  Si dimostra una forte presenza mentale anche se non si fa tutto questo senza un fine vero e proprio, soprattutto non quello di ricevere e comprendere le emozioni altrui.

- *Ascolto attivo/supportivo:* si dona il proprio tempo e le proprie energie, si supporta, incoraggia e invita l'interlocutore a continuare a comunicare, cercando di aiutarlo anche nell'esplorare e approfondire più attentamente i concetti espressi, tramite riformulazioni di ciò che riceviamo e l'utilizzo di domande che possono essere sia aperte sia chiuse. Vi è dimostrazione di apertura e interesse verso ciò che viene comunicato grazie al linguaggio del corpo, ovvero attraverso segnali del corpo, si frenano i propri pensieri e non si sta in attesa di prendere il proprio turno nella conversazione ma si dà spazio all'altro di

aprirsi. Questo può essere fatto come atto strategico, come ad esempio nel coaching o nella leadership, oppure come atto d'amore.

- *Ascolto empatico:* si mostra vero interesse nei confronti della conversazione anche tramite segnali non verbali, si ascolta attentamente mettendo da parte i propri giudizi, si fanno domande che favoriscano la comprensione, l'approfondimento e la connessione con il prossimo a un livello personale più profondo. L'ascolto empatico è caratterizzato da un interesse genuino nel comprendere a pieno ciò che l'interlocutore ci vuole trasmettere, sia dal punto di vista di pensiero sia dal punto di vista di emozioni, sentimenti e bisogni. Per fare ciò, è necessario trovarsi in uno stato di silenzio interiore, ovvero lasciar andare pensieri e giudizi, e di quiete emozionale; questo ci permette di predisporci a cogliere tutto ciò che l'interlocutore trasmette, rimanendo sempre centrati.
  Il decalogo dell'ascolto empatico [15] è una serie di 10 regole necessarie per poter mettere in pratica l'ascolto empatico:
  - *"non interrompere l'altro"* [15]
  - *"non giudicarlo prematuramente; non esprimere giudizi che possano bloccare il flusso espressivo altrui"* [15]
  - *"ricapitolare di tanto in tanto quanto si è capito (quindi se ho capito bene, è successo che...), riformulare i punti critici (ok, non ti risponde subito al telefono, e tu ci rimani molto male, capito), fare parafrasi (quindi, se capisco bene è come se....?)"* [15]
  - *"non distrarsi, non pensare ad altro, non fare altre attività mentre si ascolta (tranne prendere eventuali appunti), usare il pensiero per ascoltare, non vagare"* [15]
  - *"non correggere l'altro mentre afferma, anche quando non si è d'accordo, rimanere in ascolto"* [15]
  - *"non cercare di sopraffarlo"* [15]
  - *"non cercare di dominarlo"* [15]
  - *"non cercare di insegnargli o impartire verità, trattenere la tentazione di immettersi nel flusso espressivo per correggere qualcosa che non si ritiene corretto"* [15]
  - *"non parlare di sé"* [15]
  - *"testimoniare interesse e partecipazione attraverso i segnali verbali e il linguaggio del corpo"* [15]
- *Ascolto simpatetico:* è il massimo livello di ascolto possibile ed anche il più raro. Si dimostra pieno interesse genuino, volontà di comprendere in profondità, provando desiderio e piacere nel nel comprendere ciò che il prossimo ci

vuole comunicare, apprezzando e gradendo il vero contatto umano. Si entra in empatia con il prossimo, ovvero ci si mette nei suoi panni per comprendere a pieno i pensieri e ciò che si prova; si crea una forte connessione emotiva, che richiede appunto grandi capacità empatiche, soprattutto grazie alla completa sospensione del giudizio interiore, creando uno stato interno di silenzio interiore sia mentale sia emozionale, mantenendo la propria centratura, un perfetto equilibrio interno

3.6.2 Tecniche di ascolto attivo

Quando si ascolta attivamente il proprio interlocutore, si dimostra curiosità e interesse nei suoi confronti e nei confronti di ciò che ci trasmette: si dirige verso di egli il proprio ascolto, si forniscono feedback e incoraggiamenti, si colgono e trasmettono le emozioni, il tutto senza mai giudicare o dirgli come deve sentirsi, bensì osservando e cogliendo più segnali possibili. Per fare tutto questo, esistono varie tecniche che possono risultare utili quando si vuole davvero ascoltare il nostro interlocutore.

Tecniche verbali

- *Domande aperte:* favorisce la continuazione del discorso, introducendo anche approfondimenti su argomenti già citati precedentemente. "Perché hai deciso di dire quelle cose a Giovanni?"
- *Domande chiuse:* "È avvenuto prima o dopo?"
- *Tecnica dello specchio:* consiste nel ripetere ciò che il nostro interlocutore ci ha detto, senza alterare. Ciò aiuta l'interlocutore a entrare ancora più nel vivo del concetto che ha espresso e permette di approfondire ulteriormente
- *Parafrasi:* parafrasare ciò che ci è stato detto per constatare se si è realmente compreso ciò che l'altro ci ha detto
- *Riassunto:* "Allora se ho capito bene..."
- *Incoraggiamenti verbali:* "Ok", "Perfetto", "Sì, sì"

Tecniche paraverbali Le tecniche paraverbali consistono nel dare segnali fatici di contatto, in maniera tale che l'interlocutore percepisca la nostra vicinanza emotiva e il nostro interesse.

Tecniche non verbali Utilizzare il corpo per esprimere il proprio interesse nei confronti del nostro interlocutore: postura aperta e in avanti, corpo rilassato, avvicinamento e allontanamento , espressione del volto partecipativa e attenta (no segnali di dubbio, ironia, giudizio o aggressione), sguardo attento e diretto agli occhi dell'interlocutore, movimenti delle sopracciglia associati ai punti salienti del discorso, cenni di assenso col capo, gesti morbidi, lenti e rotatori per incoraggiare a proseguire e così via.

3.6.3 Ascolto olistico

L'ascolto olistico vuol dire ascoltare tutto in maniera più ampia, quindi in maniera molto profonda e non limitata alle sole parole. Quando l'interlocutore parla, la sua voce può trasmettere tanti segnali, che a parole non vengono detti e che rendono possibile comprendere anche le emozioni che chi parla sta provando. Spesso il non verbale dà molte più informazioni rispetto a ciò che viene veramente detto

- *Volume della voce:* può essere alto, medio, basso
- *Tono della voce:* può essere alto, basso, roco, tremolante, entusiastico o pragmatico, rispettoso o irriverente, formale o informale, serio o rispettoso https://alessandramartelli.com/risorse-scaricabili/guida-essenziale-al-tono-di-voce/
- *Silenzio:* le pause utilizzate hanno profondi significati
- *Aspetto esteriore:* l'aspetto di una persona può trasmettere molto della propria personalità, come ad esempio i tatuaggi, i vestiti utilizzati, la cura, l'utilizzo di accessori, i capelli e così via
- *Postura:* ci possono essere segnali di chiusura o apertura, avvicinamento o allontanamento
- *Espressione del viso:* dalle espressioni e microespressioni del viso è possibile riconoscere le emozioni provate dall'interlocutore
- *Camminata:* agitata, rilassata, aperta

3.6.4 Ascolto empatico in CNV

Nell'ascolto empatico in CNV si vuole entrare in uno stato di quiete interiore per togliere le orecchie giudicanti dal prossimo, riuscendo così a ricevere ciò che l'altro vuole trasmetterci con empatia.

Le 4 orecchie Supponendo di ricevere un messaggio verbale, che sia verbale oppure no, come ad esempio "Sei la persona più egocentrica che abbia mai incontrato", abbiamo 4 modi per riceverlo:

- *Incolpare l'altro:* orecchie da sciacallo verso l'altro. Risponderemo ad esempio: Sei tu quello egoista
- *Incolpare noi stessi:* orecchie da sciacallo verso noi stessi. Risponderemo ad esempio: Scusami, hai ragione, avrei dovuto essere più sensibile
- *Percepire i nostri sentimenti e bisogni:* orecchie da giraffa verso noi stessi. Risponderemo ad esempio: quando sento dire che sono la persona più egocentrica che tu abbia mai incontrato mi sento triste, perché ho davvero bisogno che i miei sforzi nel comportarmi spesso come desideri tu vengano riconosciuti
- *Percepire i sentimenti e i bisogni dell'altro:* orecchie da giraffa verso l'altro. Risponderemo ad esempio: forse ti senti arrabbiato perché questa volta ho deciso di agire secondo i miei pensieri e tu avevi bisogno di partecipazione alla tua idea anche da parte mia?

Presenza e non presenza L'empatia richiede uno stato di silenzio interiore e quiete emozionale, in maniera tale da poter mettersi in ascolto del prossimo con tutto il nostro essere; ciò implica il fatto di liberarsi dai pregiudizi che possediamo nei confronti di chi ci comunica, ascoltandolo anche se non siamo d'accordo con ciò che ci dice. Soltanto in questo modo è possibile dare all'altro lo spazio necessario per esprimersi completamente e, allo stesso tempo, di sentirsi veramente compresi. Spesso tendiamo a dare risposte affrettate e non presenti: diamo consigli, cerchiamo di tirare su il morale, educhiamo, consoliamo, raccontiamo una storia, zittiamo, commiseriamo o interroghiamo.

Ascoltare in CNV vuol dire mettersi nei panni degli altri, dirigendo il nostro focus su ciò che gli altri osservano, sui sentimenti e bisogni altrui, accantonando i propri, per comprende poi cosa davvero richiedono; questo ci risulta spesso difficile perché siamo abituati a prenderci la responsabilità dei sentimenti e bisogni degli altri e a riceverli in maniera personale.
Una tecnica molto utile, per riuscire a fare ciò, consiste nel parafrasare sotto forma di domanda ciò che abbiamo compreso sotto forma di CNV, facendo sentire l'altro davvero ascoltato e inducendolo eventualmente a fare qualche correzione, riuscendo in questo modo davvero a capire ciò che ci viene trasmesso. Può essere utile fare una domanda unica che corrisponde all'intera parafrasi, ma può rivelarsi più semplice effettuare due o più domande separate su ciò che l'altro osserva, sente, di cui ha bisogno e, infine, su ciò che richiede; bisogna stare attenti perché, facendo tante domande separate relative ai 4 passi CNV, le persone potrebbero pensare di essere

sotto esame e irritarsi.
È fondamentale avere un tono di voce adeguato, che non sia critico, sarcastico né giudicante; è più opportuno avere un tono sensibile con il quale effettuare domande per non toccare negativamente la sensibilità di chi si sente parafrasare ciò che sta trasmettendo. Allo stesso tempo, dobbiamo essere coscienti del fatto che la nostra parafrasi potrebbe essere fraintesa ("non cercare di abbindolarmi con queste tecniche psicologiche") perché spesso chi parla potrebbe essere spaventato e non fidarsi, aver bisogno di capire se le intenzioni dell'altro sono davvero genuine.

Ecco un esempio di ascolto in CNV:
A: Il mio collega mi dice sempre che cosa devo fare, è troppo seccante.
B: Sei seccato perché vorresti che si fidasse di te e del fatto che sei in grado di prendere le giuste decisioni da solo?

Ricevere un no Entrare in empatia con l'altro ci aiuta a comprendere quali sentimenti e bisogni si celano dietro quel no e che sono vivi in lui nel qui e ora; in questo modo è possibile individuare ciò che desidera davvero e comprendere il motivo per cui non risponde come vorremmo.

3.7 Diario personale e di gruppo

Durante il corso di Engineering Empathy, ogni studente ha mantenuto un diario di bordo: un elemento molto importante perché può apportare grandi benefici alle persone. Può far emergere pensieri, emozioni e sentimenti, scoprire lati che non si conoscevano prima e anche essere utilizzato come elemento terapeutico. Alla fine del corso, è stato sorprendente scoprire gli enormi benefici ricavati sia individualmente sia a livello di gruppo.

Il lavoro è stato strutturato affinché si presentassero quattro livelli, dell'intero diario, ben distinti:

1. *Diario personale (livello 1):* è sempre stato mantenuto segreto e intimo, perciò è stato possibile scrivere davvero ogni singolo pensiero, e non solo, intimo. È stato particolarmente utile rileggerlo nei mesi successivi, dopo aver maturato consapevolezze ben maggiori rispetto al passato

2. *Diario interno (livello 2):* è stato condiviso con l'insegnante. Perciò, partendo dal diario personale, sono state filtrati tutti gli aspetti più intimi e personali che non si volessero condividere (aspetti e nomi di familiari, nomi e aspetti intimi con amici e partner e così via)

3. *Diario esterno (livello 3):* è stato condiviso con tutti gli amici del corso, quindi partendo dal diario interno, anche qui sono stati filtrati alcuni elementi che non si volevano rendere noti all'esterno

4. *Diario di gruppo (livello 4):* è stato creato separatamente per ogni gruppo di lavoro creatosi durante le esercitazioni svolte nei tre mesi del corso; i diari di gruppo sono stati creati in maniera totalmente collaborativa tra tutti i membri di ogni gruppo e sono sempre stati resi pubblici anche con gli altri gruppi

Per semplicità, ogni diario è stato mantenuto fisicamente separato dagli altri, in maniera tale da non creare troppa confusione all'interno di uno solo. Inoltre, ogni diario è stato strutturato in quattro parti:

1. *Stream of consciousness:* in questa prima parte, è fondamentale scrivere di getto tutto ciò che passa per la testa, senza filtrare nulla, senza porsi domande o pregiudizi, senza pensare a cosa si scrive e come lo si scrivere, perché l'obiettivo è semplicemente mettere per iscritto tutto quanto. Inizialmente non è stato semplice, soprattutto perché il cervello viaggia a una velocità nettamente superiore rispetto a quello della scrittura ma, una volta svolto un po' di allenamento, è diventato anche più semplice avere i pensieri più organizzati, fluidi e chiari, rendendo appunto più semplice il processo di stesura in contemporanea

2. *Sketch:* una volta messi per iscritti i pensieri, le idee e i concetti, è stato creato un disegno, una bozza, uno schizzo, una forma, un paio di linee: insomma, un qualsiasi cosa che venisse in mente in un istante e che rispecchiasse le sensazioni di ciò che si provava dopo aver messo per iscritto tutti i pensieri nella prima fase. Non ci si deve preoccupare che i disegni siano belli o brutti, troppo complicati o semplicemente una linea/curva, bensì bisogna buttarsi di getto come fatto durante lo stream of consciousness

3. *Situazioni:* a questo punto, ci si prende qualche momento per riflettere e richiamare alla mente uno o più episodi passati che rispecchiassero i concetti descritti nello stream of consciousness e le sensazioni trasmesse durante la fase di sketch. Per ogni episodio, è stato descritto nei minimi dettagli, analizzato per identificare che cosa ogni singolo aspetto dell'episodio avesse provocato in noi in termini di pensiero, emozione, sentimento e comportamento; in questo modo è stato possibile sia scoprire cose che precedentemente non ci erano chiare sia poter tornare nei giorni/mesi seguenti a rianalizzare la situazioni con occhi più consapevoli e maturi per rendersi conto di aspetti che mai erano stati colti precedentemente

4. *Consapevolezza:* infine, l'ultimo passo consiste nell'ipotizzare a una soluzione, a un tipo di comportamento preciso da adottare da quel momento in futuro

quando si ripresentano situazioni simili. In questo modo, quando ricapitano quelle situazioni, è possibile scegliere come comportarsi e non reagire in base a ciò che il nostro subconscio indica, dopo aver vissuto magari tante situazioni spiacevoli e aver appreso modi di comportarsi poco appropriati per la situazione

Ogni argomento importante trattato all'interno del corso, ogni situazione significativa vissuta, ogni lavoro di gruppo è stato sviluppato seguendo queste quattro modalità per il diario di livello 1. Dopodiché, ogni sezione del diario di livello 1 è stata scremata per poter passare al livello successivo, il 2, e lo stesso procedimento è stato ripetuto per passare dal livello 3 al 4 e, infine, dal livello 3 al 4.
Alla fine del corso, è stato svolto un progetto di gruppo in cui è stato mantenuto l'ultimo diario. È stato sorprendente notare la differenza con i primi diari del primo mese di corso, è stata maturata una consapevolezza molto più grande che ha portato a benefici immensi e difficilmente descrivibili. La più grande differenza, tuttavia, si è notata nei diari personali e, in aggiunta, il vantaggio più prezioso è stato poter rileggere i diari personali dei mesi precedenti con occhi e maturità differenti.

Inoltre, si vuole sottolineare anche come la scrittura espressiva sia particolarmente utilizzata anche come elemento terapeutico.

Infine, si precisa che durante tutta la collaborazione con la collega Elisa Stroscio è stato mantenuto un diario personale e un piccolo diario di bordo di gruppo.

3.8 Problem solving e creatività

Una delle critiche mosse nei confronti del percorso di studi è quella di essere totalmente concentrata solamente sullo sviluppo delle competenze tecniche e, al più, della competenza trasversale del *problem solving*, ovvero l'abilità di scegliere ed utilizzare una serie di tecniche e metodologie per trovare la soluzione a problemi, solitamente legati alla natura delle competenze tecniche. Tuttavia, una critica aggiuntiva che è emersa dal sondaggio è rappresentata dal fatto che in realtà non sempre l'abilità del problem solving viene sviluppata in maniera opportuna, perché spesso può succedere che sia sufficiente imparare a svolgere meccanicamente una serie di esercizi, magari seguendo un algoritmo ben noto a priori: questo ridurrebbe lo sforzo necessario per analizzare con precisione e scegliere la miglior strategia da adottare per risolvere il problema. In aggiunta, mancherebbe anche quello che è un ingrediente fondamentale che tante volte nella storia ha reso possibile la soluzione di problemi che sembravano essere insormontabili per l'essere umano: la creatività. Pertanto, si è deciso di effettuare una ricerca per comprendere se esiste uno strumento facilmente applicabile fin da subito per migliorare la figura del neolaureato ingegnere dal punto di vista dello sviluppo del problem solving in maniera abbinata

alla creatività. La ricerca ha condotto verso uno strumento molto semplice quanto potente: *I sei cappelli per pensare* [16], ideato da Edward De Bono, considerato un maestro assoluto per quanto riguarda il pensiero creativo.

3.8.1 I sei cappelli per pensare

Edward De Bono ritiene che la difficoltà principale nel pensare consista nella confusione che si viene a creare in quanto si cerca di fare troppe cose contemporaneamente: quando si ragiona su un problema, lo si guarda da molti punti di vista in maniera caotica. Lo scopo del metodo dei sei cappelli, invece, è quello di fornire uno strumento per pensare da un punto di vista per volta: l'approccio consiste nel far finta di indossare un cappello, ognuno di un colore differenze, per volta in maniera tale da analizzare l'oggetto del ragionamento da sei punti di vista differenti in maniera organizzata: in questo modo, è possibile dirigere il pensiero proprio come farebbe un direttore d'orchestra. Allo stesso tempo, è possibile arrivare a soluzioni creative. Più ci si allena e più diventerà semplice e normale riuscire ad arrivare a soluzioni creative perché lo scopo del metodo è proprio quello di indossare i cappelli tramite un processo intenzionale che, tramite una strategia ben precisa per dirigere l'attenzione in maniera specifica su aspetti completamente differenti del problema, conduce alla creazione di una mappa chiara, precisa, oggettiva e neutrale del problema analizzato, allargando sempre di più la propria visuale e prospettiva da cui si analizza il problema in questione. Una volta ottenuta la mappa, è possibile scegliere poi il percorso migliore da seguire. Un altro vantaggio di questo metodo è rappresentato dal fatto di poter scegliere di cambiare approccio (e chiedere che venga fatta la stessa cosa) all'interno di una conversazione che potrebbe prendere una brutta piega se non venisse cambiato registro.

Si presentano di seguito i sei cappelli che rappresentano i diversi punti di vista da cui analizzare il problema:

- *Il cappello bianco:* il colore bianco rappresenta l'oggettività e la neutralità, proprio per questo il cappello bianco focalizza l'attenzione sui fatti e, pertanto, sull'oggettività dei dati. Per pensare in questo modo, è necessario far finta di essere un computer e trattare tutte le informazioni con imparziale obiettività

- *Il cappello rosso:* il colore rosso rappresenta le emozioni, pertanto il cappello rosso serve a concentrarsi sull'aspetto emotivo del problema. Quindi in questa fase vengono tirate in gioco tutte le sensazioni, i presentimenti, il sospetto e tutti gli aspetti emotivi che possono intercorrere al pensiero del problema

- *Il cappello nero:* il colore nero rappresenta la negatività, quindi il cappello nero viene utilizzato per riflettere sugli aspetti negativi, ovvero sui motivi per

i quali una determinata cosa non potrebbe o dovrebbe essere fatta. Questa è l'occasione giusta per tirar fuori le passate esperienze, i motivi per i quali le cose sono scorrette o false, i motivi per cui una cosa non può funzionare. Non si tratta di essere controversi ma di cercare di riportare con obiettività i motivi per cui certi elementi della mappa sono negativi

- *Il cappello giallo:* il colore giallo rappresenta la positività e la solarità, il cappello giallo sottolinea gli aspetti positivi e le speranze ottimistiche delle soluzioni al problema. Si tratta dell'esatto opposto del cappello nero, perciò si tirano fuori tutti gli elementi positivi da poter inserire nella mappa; in particolare, quando si ha a che fare con idee innovative, è sempre meglio "indossare" il cappello giallo rispetto a quello nero

- *Il cappello verde:* il colore verde rappresenta l'abbondanza e la fertilità, in quanto richiama all'erba, perciò il cappello verde vuole favorire la creatività e la produzione di idee nuove ed innovative. Qui è dove viene fuori la maggior parte delle idee creative perché si cerca continuamente di trovare alternative andando oltre ciò che pare ovvio e noto agli occhi di tutti. Esistono varie tecniche per fare ciò, per esempio quello della provocazione: si parte da una parola completamente casuale (ad esempio estrapolata casualmente dal dizionario) e si lascia lavorare il cervello tramite associazioni libere, favorendo così il pensiero laterale, con l'obiettivo di fuoriuscire dagli schermi di pensiero abituali.

- *Il cappello blu:* il colore blu rappresenta il cielo e il freddo, per questo motivo il cappello blu è utilizzato come strumento per astrarsi e organizzare il processo del pensiero, perciò è relativo anche a tutti gli altri cappelli. Il cappello blu serve come elemento di controllo per organizzare tutto il processo di pensiero e dar vita alla metafora del direttore d'orchestra.

Infine, si vuole sottolineare l'importanza di esprimersi riguardo ai cappelli tramite i colori e non le loro funzionalità: chiedere a una persona di fornire un punto di vista emotivo potrebbe scaturire in lei una reazione di opposizione nell'esprimersi, chiederle invece di *indossare il cappello rosso* rende il processo neutro e privo di pregiudizi.

3.9 Public speaking

Il presente capitolo vuole fornire una panoramica di tecniche di public speaking, apprese durante la partecipazione di diversi incontri di Toastmasters [17] e durante l'analisi di centinaia di Ted Talks [18] ascoltati con estrema attenzione dal punto di vista delle tecniche di esposizione utilizzate.

3.9.1 Gestione della paura

La paura di parlare in pubblico è chiamata glossofobia e un sondaggio svolto dall'azienda YouGov [19] ha mostrato come sia una delle paure più diffuse nell'essere umano. Premettendo che non è stata effettuata un'analisi approfondita in ambito psicologico, l'autore ha identificato quelle che sono le cause più comuni di questa paura, in condizioni normotipiche, e ha analizzato e provato su se stesso alcune delle tecniche più diffuse per imparare a gestire l'ansia da discorso in pubblico: in particolare, si è notato che la maggior parte di queste è comune ad altre situazioni nella vita. I sintomi più comuni sono: battito cardiaco accelerato, nervosismo, tremore, sudore eccessivo, sensazione di annebbiamento al cervello, gola ristretta, voce tremolante e così via.
Le cause e le soluzioni più comuni più comuni sono:

- La prime due cause che si vogliono riportare sono quelle più naturali e antiche: si tratta dell'insita incapacità dell'essere umano di riconoscere la differenza tra gli animali feroci (situazione derivante dall'antichità in cui l'uomo doveva difendersi dai pericoli) e le persone che compongono il pubblico e hanno gli occhi tutti rivolti verso l'oratore; allo stesso tempo, essendo animali sociali, non vogliamo rimanere esclusi dal gruppo, dalla comunità e, pertanto, temiamo che il fatto di non essere all'altezza ci porti a perdere status ed essere esclusi

- Avere pensieri limitanti che incidono negativamente sulla prestazione. Solitamente questo avviene tramite domande che frullano nella mente, come ad esempio: *e se le persone si annoiano? E se mi dimentico il discorso? E se non riesco a rispondere alle domande che mi vengono effettuate? E così via.* In questi casi, il metodo più efficace è quello di individuare il pensiero e reinterpretare la paura come uno stimolo a migliorare da quel punto di vista. Se si teme di annoiare le persone, allora ci si può chiedere come fare in modo che ciò non avvenga e, con debito anticipo, preparare un discorso che sia non noioso. Se si teme di dimenticarsi il discorso, allora ci si chiede come fare in modo di ricordarsi i tratti salienti e il filo conduttore: si possono quindi imparare e utilizzare semplici tecniche mnemoniche, come ad esempio la tecnica dei loci, per far sì che ciò non avvenga. Se si teme di non saper rispondere le domande allora nei giorni precedenti si possono immaginare e scrivere tutte le domande che potrebbero essere effettuate, preparandosi una bozza di risposta. E così via.

- La mancanza di una preparazione adeguata. Questo si riconduce al punto precedente e, allo stesso tempo, denota che sarebbe opportuno creare dei contesti in cui le persone possono esercitarsi in maniera efficace

- Ansia sociale, traumi o situazioni negative vissute in passato. In questi casi, la soluzione migliore è quella di fare affidamento a uno specialista, uno psicologo/psicoterapeuta o un medico.
  Se si tratta di una lieve ansia sociale, può essere tranquillamente superata con un percorso autonomo che prevede prima una presa di consapevolezza della situazione e poi la delineazione di piccoli obiettivi graduali da raggiungere attraverso una strategia precisa: le tecniche in questi casi possono essere la terapia cognitivo comportamentale (cognitive behavioral therapy), l'ACT (Acceptance and Commitment Therapy, sezione 3.10) o la "semplice" capacità di rimanere connessi nel qui e ora (un esercizio molto utile a questo scopo è quello della meditazione o semplicemente di attenzione focalizzata al respiro). Le tecniche palliative più utilizzate, nel momento in cui si vive l'ansia, sono l'attenzione al respiro o le tecniche del radicamento (grounding techniques) che servono a focalizzarsi e connettere il proprio corpo con la terra (esistono vari esercizi facilmente reperibili online).

Nel Giugno 2012, Amy Cuddy ha tenuto un celbere discorso ad una conferenza TED [20] sull'importanza del linguaggio del corpo. Oltre a raccontare aneddoti molto interessanti, Amy Cuddy sottolinea come negli studi scientifici vi sia stato un importante cambio di prospettiva: se prima si pensava che i pensieri potessero influire sulla postura del corpo, ora invece è noto che vale anche il viceversa, ovvero che la postura del nostro corpo (più facile da controllare rispetto ai pensieri) influenza le emozioni che proviamo e i pensieri che facciamo. In particolare, Amy Cuddy mostra come sia possibile assumere delle posizioni fisiologiche di espansione per influenzare positivamente le nostre emozioni e i nostri pensieri, in maniera tale che anche se non siamo in grado di affrontare la paura di tenere ad esempio un discorso in pubblico possiamo utilizzare questa tecnica per fingerci tranquilli e rilassati e riuscire a raggiungere l'obiettivo desiderato. Piccole modifiche alle azioni comuni possono portare a straordinari cambiamenti: due minuti in cui si praticano posture di espansione portano a un cambiamento ormonale importante: si ha un innalzamento del testosterone addirittura del 20% e, allo stesso tempo, si ha una riduzione del 25% circa dell'ormone dello stress, il cortisolo ([21]). Si tratta di un cambio di paradigma enorme perché permette a chiunque di essere in grado di influenzare il proprio stato emotivo in maniera positiva e affrontare situazioni che, senza l'aiuto di questo piccolo trucco, difficilmente sarebbero state fronteggiate ([22]).

Analisi del pubblico, selezione dell'argomento e del titolo

Il primo passo per preparare un discorso da presentare per il pubblico è studiare il pubblico e categorizzarlo, perché se si ha chiaro ciò che esso cerca, allora gli si può offrire quello che desidera. Conoscere il pubblico può portare l'oratore ad essere ascoltato, a una vendita, a motivare, a persuadere o, più in generale, ad ottenere

l'obiettivo desiderato per tale discorso.

Per analizzare il pubblico, è necessario porsi delle domande preliminari, come ad esempio: *perché sono proprio io a fare la presentazione (sono stato scelto)? Qual è l'obiettivo del discorso? Come si vuole essere percepiti? Da che tipo di persone è composto il pubblico nello specifico? Quali preoccupazioni o problemi presenta il pubblico che si possono risolvere tramite il discorso? Quali benefici vogliamo apportare alle persone con il nostro discorso? Cosa si desidera ottenere dal pubblico con il nostro discorso? Come si può far comprendere alle persone che ciò che sto dicendo ha un grande valore?*
Queste non sono altro che domande di esempio che ci si può porre prima di dover preparare un discorso, più si entra nel dettaglio, più si conosce alla perfezione il pubblico al quale presentare il discorso e più ci sarà la probabilità di effettuare un discorso che apporti davvero valore in chi ascolta.

Una volta effettuata questa prima analisi, si può passare a selezionare l'argomento di cui parlare. È fondamentale scegliere un argomento che interessi sia l'oratore sia il pubblico e, allo stesso tempo, è fondamentale che l'argomento si concentri su un tema specifico, senza rimanere troppo vago. In questa fase, può essere utile anche aver chiaro quanto tempo si ha a disposizione per presentare il discorso in maniera tale da capire quante e quali informazioni selezionare in maniera tale da trasmettere gli aspetti che potrebbero davvero interessare il pubblico.
Dopodiché, si può passare alla creazione del discorso oppure scegliere già in prima battuta un titolo. Che la scelta venga effettuata subito o più avanti, è importante utilizzare un titolo che catturi l'attenzione e che racchiuda l'essenza del discorso. Quindi è fondamentale scegliere un titolo che faccia comprendere a chi lo legge se vale la pena ascoltare il discorso e se lo stesso può apportare valore in chi decide di ascoltarlo. Ad esempio, se si deve tenere un discorso per spiegare a dei giovani neolaureati come scrivere un Curriculum Vitae (CV) efficace, si potrebbe scegliere un titolo del tipo:

Neolaureati e CV

Che risulta sicuramente meno efficace di un titolo come il seguente:

Neolaureato? I 3 passi per creare un CV efficace

3.9.2 La struttura del discorso

La capacità più importante di un buon oratore è probabilmente quella di esprimere concetti complessi in maniera chiara e lineare. Come si può raggiungere una buona chiarezza e un'efficace linearità nel presentare un discorso? Procedere attraverso una sequenza ben precisa: anticipare, presentare e ripetere al pubblico il messaggio

che si vuole trasmettere; tutto questo si traduce nella capacità di di strutturare il proprio discorso in tre parti: introduzione (per anticipare il messaggio), corpo (per consegnare il messaggio) e conclusione (per ricalcare il messaggio). È altresì importante bilanciare la durata delle tre fasi in maniera proporzionale ed equilibrata: ad esempio, per un discorso di 30 minuti, si potrebbe trattare l'introduzione e il finale per 5 minuti ciascuno e dedicare 20 minuti al corpo. Questa struttura ovviamente non è solo limitata ai discorsi in pubblico, ma anche ai libri, ai film, ai racconti, nelle favole e così via. I discorsi strutturati in questa maniera sono più semplici da capire e da ricordare, più credibili e gradevoli per le persone che ascoltando; inoltre, sono più semplici da ricordare per gli oratori.

Introduzione

L'introduzione anticipa il messaggio e stabilisce il tema e gli argomenti/punti a supporto. In questa fase, è opportuno evitare le solite introduzioni banali e prevedibili, come ad esempio *Buongiorno a tutti, sono Mister X e sono contento di essere qui*. Esistono tantissime tecniche per creare un'introduzione efficace, se ne riportano tre particolarmente utilizzate:

- *Citazione/aneddoto:* si può utilizzare una citazione particolarmente famosa, oppure una testimonianza (che è anche una potente arma di persuasione, soprattutto se relativa ad una figura autorevole), per identificare quello che è il tema principale del discorso oppure raccontare un aneddoto, preferibilmente vissuto in prima persona, in linea con ciò di cui si parlerà.

- *Domanda/problema-soluzione:* si può introdurre il proprio discorso tramite una domanda aperta, in maniera tale che le persone siano curiose di ascoltare per comprenderne la risposta. È anche possibile presentare direttamente quello che è il problema che si vuole trattare (magari proprio attraverso una domanda) e invitare le persone a darsi una risposta (soluzione) implicita.

- *Silenzio/sorriso:* Può sembrare scontato ma spesso è sufficiente un sorriso per entrare in sintonia con il pubblico, in quanto esso genera empatia e trasmette la sensazione che il discorso o l'oratore che si sta per ascoltare sarà piacevole. Allo stesso modo, il silenzio iniziale trasmette una sensazione di calma e pace da parte dell'oratore e fa in modo che l'attenzione del pubblico sia completamente incanalata sulle sue prime parole

Corpo

Il corpo rappresenta la fase del discorso in cui si va a trasmettere il messaggio vero e proprio. Può particolarmente utile suddividerlo in tre punti o comunque in argomentazioni separate e ciò può essere ottenuto attraverso svariate tecniche; se ne presentano tre di seguito:

- *Cronologico:* presentare i tre argomenti del discorso in maniera cronologica
- *Causa/effetto:* presentare ogni punto del corpo mettendo in relazione la causa di un evento con l'effetto apportato
- *Problema/soluzione:* analizzare prima nel dettaglio il problema per poi fornire una soluzione ben precisa

Conclusione

La conclusione è la fase del discorso in cui si vuole riepilogare quanto trattato precedentemente, in particolar modo i punti trattati nel corpo, per lasciar sedimentare nel pubblico; allo stesso tempo è fondamentale richiamare all'azione, ovvero essere chiari nell'esprimere ciò che si desidera che il pubblico faccia. In questa parte del discorso è fondamentale essere rapidi, chiari e diretti, ricordandosi sempre di focalizzare l'attenzione sui benefici di quanto proposto tramite il discorso.
Infine, è preferibile non concludere con il solito cordiale *Grazie dell'attenzione*, bensì creare un finale con climax ascendente per conquistare il pubblico per rafforzare il messaggio trasmesso. Le tecniche per questa fase sono molto simili a quelle trattate per l'introduzione.

3.10 Acceptance and Commitment Therapy (ACT)

In una delle risposte del sondaggio, è stato mostrato il chiaro disagio che molte persone possono provare nel dover affrontare un corso come Engineering Empathy e, più in generale, nel dover magari esporsi per effettuare presentazioni in pubblico o semplici lavori di gruppo tra colleghi. Per questo motivo, è stata avanzata la proposta dell'affiancamento di uno psicologo nel nuovo percorso proposto in questa tesi. Allo stesso tempo, è comprensibile come questo cambiamento possa essere difficilmente operato, pertanto si è deciso di effettuare una ricerca, indirizzata dai suggerimenti di una decina di psicologi intervistati, per trovare e analizzare nel dettaglio un qualche strumento che fosse applicabile fin da subito e in autonomia per ovviare al problema di fondo.
La ricerca è ricaduta sulla Terapia di accettazione e di impegno nell'azione (Acceptance and Commitment Therapy - ACT), che è una nuova branca della psicoterapia basata sull'esperienza diretta, ma con valido sostegno scientifico. L'obiettivo potrebbe essere quello di includere un breve percorso sull'ACT all'interno del corso Engineering Empathy e sulla nuova piattaforma analizzata nel capitolo 5. Nel seguito, si vuole presentare la tecnica dell'ACT e si rimanda ad ulteriori approfondimenti nel futuro.

L'ACT [23] pone le sue radici smentendo quelli che sono i principali miti riguardo la felicità:

- La natura degli esseri umani è quella di essere felici. Non è affatto così, le statistiche mostrano come la probabilità di soffrire di disturbi psichiatrici sia quasi del 30%, senza considerare tutti i problemi legati allo stress, le relazioni, i pregiudizi e così via
- Se una persona prova sofferenza mentale o non è felice, vuol dire che ha qualcosa che non funziona. Anzi, è vero proprio il contrario: i meccanismi attuati da una mente sana portano alla sofferenza psicologica
- Bisogna scacciare i sentimenti negativi per migliorare la propria vita
- Bisogna essere capaci di controllare i propri pensieri e i propri sentimenti/emozioni. L'idea dell'illusione di controllo ci è stata instillata durante la giovinezza ricevendo frasi del tipo: *"Non piangere"*, *"Non c'è niente di cui aver paura"*, *"Non essere così triste"*, *"Cose che capitano a tutti"*, *"Fattela passare"*, *"Tirati su"*, *"Reagisci"* [23] e così via.

Quando si ha un problema come ad esempio il prurito, ci si gratta e il problema passa. Immaginando invece di avere un'allergia cutanea o un problema più grave, grattarsi in quel caso peggiorerebbe la situazione. Pertanto, grattarsi è una valida soluzione per un prurito leggero e relativamente raro su una pelle sana, non lo è invece in caso di problemi più persistenti. Questo porta a unn'altra affermazione che rappresenta una pietra miliare dell'ACT:

La soluzione è il problema

Questo perché le persone tendono ad adottare strategie che ingigantiscono i propri problemi. Ad esempio, se una persona soffre di ansia sociale e teme il confronto ad una festa con un gruppo di amici, solitamente tende ad evitare quella situazione e prova un rapido sollievo. Questo però ingigantisce il problema, che si ripresenterò più grande alla volta successiva. Siccome queste situazioni non succedono una volta ma si ripetono poi continuamente, diventa un circolo vizioso e ci si ritrova a non essere più in grado di risolvere il problema. Le soluzioni che operano le persone senza rendersi conto che però ingigantiscono i propri problemi sono chiamate strategie di controllo (ad esempio la repressione, la discussione interiore con i propri pensieri, il voler dominare i propri pensieri, l'autocostrizione e così via). Le strategie di controllo, come esaltato dall'esempio precedente, funzionano e vanno bene se utilizzate raramente, se utilizzate in situazioni in cui possono funzionare e se non impediscono di fare cose che sono importanti per i valori della persona.
È per questo che nella società del benessere di oggi la maggior parte delle persone non è comunque felice e vive costantemente sotto stress: ci si ritrova nella cosiddetta trappola della felicità. L'ACT serve proprio a fornire una strategia pratica

per uscire da questa trappola e permettere di riconnettersi con quelli che sono i valori importanti per ogni persona. Il primo passo, ovviamente, consiste nel fatto di accorgersi di essere all'interno della trappola.

I principi fondamentali dell'ACT [23] sono 6:

1. *Defusione:* è fondamentale rapportarsi con i propri pensieri in una maniera nuova in maniera tale da diminuire l'influenza che possiedono sulla persona. Praticare la defusione (il contrario di fusione che significa unire o mescolare) dai propri pensieri consente di scegliere i comportamenti da adottare senza farsi influenzare negativamente appunto da pensieri limitanti.

2. *Accettazione/Espansione:* una volta riconosciuti i pensieri limitanti e le emozioni negative provate, è necessario far spazio loro. Solitamente, le persone a questo punto cercano di praticare strategie di controllo, che è l'esatto opposto di ciò che è più opportuno fare, ovvero aprirsi, accettarli, accoglierli concedendo loro dello spazio interiore per vivere, in questo modo ci si accorge che fluiscono via ben prima del previsto

3. *Connessione:* a questo punto, è fondamentale connettersi con il qui e ora per rimanere concentrati su ciò che sta accadendo nel presente per non farsi trasportare da pensieri negativi sul passato o da preoccupazioni per il futuro. In questa fase, entra in scena anche la mindfulness che è un potentissimo alleato: "uno stato mentale di consapevolezza, apertura e concentrazione capace di dare enormi benefici sul piano sia fisico sia psicologico" [23]

4. *Il sé osservante:* utilizzare il sé osservante per "astrarsi" dai pensieri limitanti che generano emozioni e sentimenti negativi

5. *Valori:* conoscere i propri valori è fondamentale per avere sempre ben chiara la direzione verso cui puntare, in maniera tale da riuscire a non farsi guidare dal fiume di pensieri limitanti ed emozioni negative

6. *Azione impegnata:* alla fine dei conti, l'azione che si compie è quella che conta davvero; per effettuare l'azione più corretta per ognuno di noi, è fondamentale avere chiara la destinazione, che dipende appunto dai valori della persona

Questi sei principi cardine possono essere racchiusi in tre passi fondamentali che racchiudono l'essenza dell'ACT:

1. *Acceptance:* si parte dalla capacità di rendersi conto che i pensieri non sono altro che storie che racconta la mente umana, che è appunto una grande narratrice di storie (ad esempio: *"fallirò", "non sono capace", "non ci riesco proprio", "non sono portato"* e così via). Dopodiché avvengono la fase di accettazione dei pensieri limitanti, in cui si spegne l'interruttore della lotta

contro di essi, e la fase di accoglimento delle emozioni e dei sentimenti negativi: per fare questo, è necessario osservare le emozioni negativi e non pensare ad ese. I pensieri negativi e limitanti con i conseguenti sentimenti ed emozioni generati vengono spesso indicati come demoni (ovvero esseri inesistenti che fanno paura finché viene dato loro questo potere) attraverso la metafora di una barca: si immagini di essere in una barca di notte al buio e di avere un patto con dei demoni che si trovano nascosti sotto una coperta: finché la barca non avrà una direzione ben precisa, come l'arrivo a riva, i demoni staranno lì tranquilli, in caso contrario "salgono immediatamente sul ponte, agitano le ali, sguainano le zanne e minacciano di fare a brandelli la persona" [23]. I demoni, in quanto pensieri/storie, possono anche non finire mai se ci si fonde con essi, pertanto diventa fondamentale rendersi conto che sono appunto soltanto dei pensieri e non fatti reali (il racconto di un qualcosa non è un/il fatto). Esistono varie tecniche per praticare la defusione dai propri demoni, quali ad esempio:

- Riconoscere che si tratta di un pensiero: *Sto avendo il pensiero di...*
- Ridicolizzare un'immagine che ha un significato molto negativo per la persona e che rappresenta un demone che appare insormontabile. È possibile effettuare la stessa operazione magari con un pregiudizio, canticchiandolo a ritmo di una canzoncina divertente. Magari si può prendere l'immagine spaventosa e ridurla in miniatura, racchiuderla in un piccolo schermo televisivo, impostarne i sottolitoli, cambiare l'audio o la colonna sonora e così via.
- dare un nome alle storielle che si ripetono più spesso dentro la testa e poi: *Ah, ecco la storiella X di me che non piaccio a nessuno...*
- ringraziare la menta, che nonostante tutto richiama certe immagini o storie al fine di proteggere la persona da eventi spiacevoli (si pensi all'esempio precedente dell'ansia alla festa)

Per riuscire ad effettuare tutto questo, è importante conoscere la differenza tra il sé pensante e il sé osservante. Il primo è colui che pensa, immagina, visualizza, giudica, sogna e così via. Le strategie di controllo agiscono proprio nel vano tentativo di controllare appunto il sé pensante, che non è affatto semplice da gestire in realtà. Il secondo, invece, è colui che è consapevole di ciò che accade senza però pensare; "è la parte della persona responsabile della concentrazione, dell'attenzione e della consapevolezza" [23]: quindi, il sé osservante può incanalare l'attenzione verso i pensieri senza produrne. Quando si è completamente concentrati su un qualcosa, non la si sta pensando. L'obiettivo, pertanto, è proprio quelli di utilizzare questa parte della persona, il sé osservante, per riuscire a praticare l'accettazione dei demoni, delle emozioni e dei sentimenti negativi. Questa abilità è fondamentale perché la

mente non smetterà mai di raccontare storielle negative, quindi di presentare demoni che appaiono insormontabili. Bisogna riconnettersi con il qui e ora, procedimento che accade grazie al sé osservante che non lotta contro la realtà né si oppone, bensì la osserva e l'accoglie per quello che è.

2. *Commitment:* a questo punto, non resta che riconnettersi con quelli che sono i propri valori in maniera tale da avere la meta chiara e scegliere le azioni da compiere affinché siano in linea con i propri obiettivi, dettati appunto dai propri valori che non sono altro che la direzione da seguire nella propria vita.

L'ACT si tratta perciò di un differente approccio da adottare nei confronti di tutte le situazioni che capitano nella vita ed è estremamente utile per riuscire ad affrontare situazioni che magari ci possono apparire insormontabili, come ad esempio quelle di riuscire a tenere una presentazione in pubblico, come accennato all'inizio della sezione. È chiaro che a tutto questo, deve essere abbinato un percorso graduale di miglioramento della competenza attraverso una serie di obiettivi graduali.

3.11 Il modello di apprendimento 70:20:10

Il modello 70:20:10 è un modello di apprendimento per lo sviluppo di capacità; è stato creato negli anni 80 da tre ricercatori che lavoravano per il Centro di Leadership Creativa [24] e, ad oggi, è molto utilizzato dai professionisti dello sviluppo delle risorse umane in tutto il mondo, anche nelle aziende italiane come testimoniato dalle interviste. Si tratta di una formula che rappresenta una linea guida per indicare quali dovrebbero essere idealmente le fonti di apprendimento per persone che ricoprono ruoli manageriali. In particolare, il 70% deriva dalle sfide che si devono affrontare quotidianamente sul campo di lavoro; il 20% deriva dalle interazioni con persone dello stesso livello (quindi con i propri pari), persone che ricoprono ruoli simili oppure mentori/coach, ma anche e soprattutto dai feedback scambiati; infine, il 10% deriva dalla formazione formale. Nonostante non trovi molto riscontro dalla ricerca scientifica, questo modello è ampiamente utilizzato in tutto il mondo e la sua forza risiede anche nel proporre una combinazione di diverse forme di apprendimento.

Nel 2018, è stato effettuato uno studio qualitativo [25] che ha provato ad analizzare gli effetti dell'applicazione di questo modello su manager di aziende nel settore pubblico australiano. Seguendo il lavoro svolto in questa ricerca, il modello non ha portato ai risultati sperati e gli autori hanno delineato quattro possibili cause derivanti da idee sbagliate: pensare che un apprendimento esperienziale, senza una struttura, possa condurre allo sviluppo e all'acquisizione di competenze in maniera automatica (che tra l'altro è anche il modo più complicato per sviluppare determinate competenze [5]); non avere le idee chiare sul ruolo dell'apprendimento sociale e

considerarlo da un punto di vista troppo limitante; tralasciare il ruolo del supporto attivo a seguito della formazione; non avere le idee chiare su come integrare i tre modelli di apprendimento (esperienziale, sociale e formale) in maniera efficace.

Questa ricerca non ha dato conferma sulla validità scientifica del modello, bensì ha fornito delle idee molto più chiare su come raddrizzare la mira per applicare il modello 70:20:10 in maniera molto più efficace, riuscendo a integrare le tre fasi in maniera tale che conducano verso il risultato desiderato. L'elemento che lega alla perfezione i tre ingredienti è proprio l'aspetto sociale del modello.
In primo luogo, non bisogna confondere l'apprendimento basato sull'esperienza con l'apprendimento strada facendo a lavoro: è essenziale fornire una struttura all'apprendimento di questo tipo e non lasciarlo al caso affinché sia più valorizzato; quindi è più efficace fornire alle persone delle tecniche per affrontare le sfide che il contesto lavorativo pone loro. Spesso le persone vengono incaricate di ricoprire un certo ruolo senza avere la preparazione necessaria e si pensa che si acquisiranno le competenze necessarie lungo il percorso intrapreso. È proprio la confusione che c'è trai due concetti a rendere meno profittevole il modello 70:20:10. Pertanto non è necessario fornire inizialmente grandi quantità di nozioni, ma anche solo avere una piccola ma precisa strategia può notevolmente migliorare il risultato.
Inoltre, non si può pensare che basti l'apprendimento formale e la pratica, è opportuno fornire un supporto attivo e costante durante il processo di sviluppo di una determinata competenza; in particolare, sarebbe davvero molto efficace avere un mentore, una figura che indichi la strada da seguire. È fondamentale non avere una visione troppo restrittiva di quello che è l'apprendimento sociale. Considerare quest'ultimo derivante solamente dalle attività di coaching, mentoring e così via è limitante perché l'apprendimento sociale avviene anche tramite osservazioni e analisi derivanti dalle interazioni che si stabiliscono tra colleghi; questo potrebbe portare a un'incongruenza tra quanto imparato formalmente e quanto appreso osservando il comportamento dei colleghi e delle persone circostanti.
Infine, per quanto riguarda invece l'apprendimento legato alla formazione formale, è fondamentale che si tratti di un percorso di valore e che permetta anche l'interazione e metta in campo l'esperienza pratica di chi forma in maniera tale che non si sleghino le nozioni teoriche dalle situazioni in cui possono essere applicate. È assolutamente importante che ci sia uno spazio in cui poter applicare direttamente quanto appreso durante i programmi di formazione formale.

Capitolo 4

Proposte di innovazione

4.1 Necessità di miglioramento

Dopo aver analizzato attentamente i risultati del sondaggio e delle interviste e aver effettuato una lunga ricerca sulle fondamenta teoriche dell'empatia e delle competenze trasversali derivanti, è stato possibile identificare un filo conduttore che lega tutti i dati raccolti e che si presenta di seguito.

La figura del neolaureato ingegnere presenta lacune, che possono essere più o meno rilevanti a seconda dell'individuo in questione, nel sapersi relazionare efficacemente con le persone una volta entrato nel mondo del lavoro. Le competenze relazionali derivano certamente dalle caratteristiche della personalità e dell'ambiente familiare in cui si è vissuto ma, allo stesso tempo, sono delle abilità che possono essere sviluppate [26] e allenate nel tempo. Il loro apprendimento e l'acquisizione dell'esperienza necessaria sono un processo molto più lento rispetto a quello delle capacità tecniche. È imprescindibile non separare l'insegnamento delle competenze tecniche da quelle trasversali, bensì è doveroso riuscire a trovare un approccio che permetta di svilupparle contemporaneamente. Inoltre, è fondamentale in ambiente lavorativo saper comunicare e presentare le proprie idee efficacemente e sapersi approcciare correttamente in contesti costituiti da insiemi di persone che collaborano. Per sviluppare l'empatia e le competenze trasversali derivate è necessario avere delle fondamenta e una strategia da seguire per evitare di ricadere nella situazione descritta nella sezione 3.11, in cui l'apprendimento avviene in maniera non strutturata e, pertanto, risulta decisamente meno efficace.
È importante effettuare un percorso di consapevolezza di se stessi e di sé in relazione al gruppo e, allo stesso tempo, non solo apprendere questi concetti teoricamente, bensì metterli in pratica quotidianamente. Diventa necessario concedersi del tempo per imparare e acquisire l'esperienza necessaria, darsi del tempo per analizzare quanto accaduto, comprendere che cosa è stato fatto in maniera efficace e poisitiva e cosa invece può essere migliorato e questo avviene principalmente anche grazie

alla ricezione di feedback di altre persone.

Pertanto, i risultati del sondaggio, delle interviste e della ricerca teorica hanno supportato l'ipotesi inizialmente formulata; anche dal punto di vista della letteratura, si è trovato un riscontro positivo in quest'ottica (tra le tante ricerche, ad esempio: [27], [28] e [29]). Quindi si è cercato di ideare un approccio pratico in ambito universitario per migliorare la figura del neolaureato ingegnere dal punto di vista delle competenze relazionali.
Potrebbe essere più efficace mantenere due percorsi differenti per quanto riguarda:

1. Lo sviluppo dell'empatia, dell'ascolto attivo, della creatività, della capacità di lavorare in gruppo, dell'apertura nei confronti dei feedback e la loro gestione e del la capacità di adattarsi alle situazioni, attraverso un percorso di consapevolezza di sé e di se stessi all'interno del gruppo

2. Lo sviluppo delle altre competenze trasversali. In particolare, quelle che sembrano essere più rilevanti e soprattutto che possono essere allenate già a partire dal percorso universitario sono: la leadership e la capacità di lavorare in gruppo, la creatività legata al problem solving e l'abilità di comunicare e presentare efficacemente in pubblico (public speaking)

Le competenze analizzate nel primo punto sono trattate più che efficacemente nel corso Engineering Empathy. Per quanto riguarda invece quelle esposte del secondo punto, non esistono ancora dei corsi in cui vengono trattati questi temi; in alcuni, in realtà, è possibile presentare dei progetti apertamente, ma questo avviene in maniera non strutturata e con una preparazione che non è assolutamente adeguata. Pertanto, si segnalano alcune proposte di miglioramento che sono derivate da una lunga, profonda, attenta e lungimirante analisi di tutti i dati raccolti.

4.1.1 Le proposte

Spostare il corso di *Engineering Empathy* (EE) al primo anno del percorso triennale come credito libero. Inoltre, alla fine del terzo anno accademico, oppure all'inizio del primo anno della magistrale, sarebbe opportuno mettere a disposizione una seconda versione di questo corso, magari *Engineering Empathy II* (EE II).
In Engineering Empathy sarebbe essenziale sviluppare i temi dell'empatia, dell'ascolto attivo, della capacità di lavorare in gruppo, dell'apertura nei confronti dei feedback e la loro gestione e della capacità di adattarsi alle situazioni. Questo avviene attraverso un percorso di consapevolezza di sé e di se stessi all'interno del gruppo tramite l'utilizzo di strumenti quali l'analisi transazionale, la comunicazione nonviolenta e l'ascolto attivo: è già stato ideato e identificato ed è stato portato a termine con estrema efficacia durante l'anno accademico 2019/2020, il primo in cui è stata tenuta la prima versione ufficiale di questo corso.

Dopo aver posto le basi di tutti questi temi, dopo aver cambiato radicalmente prospettiva e dopo aver preso consapevolezza di quanto sia differente vivere anche il contesto studentesco/lavorativo riuscendo ad entrare in profonda connessione con le altre persone, lo studente dovrebbe essere tenuto a mantenere un diario di bordo per analizzare l'andamento di tutti i progetti di gruppo che svolge da quando ha partecipato a EE in avanti.
Una volta giunti a EE II, è possibile attuare il vero processo di *follow up* di cui si è trattato nelle interviste. Gli obiettivi sono due: in primo luogo, continuare il percorso iniziato al primo anno, però questa volta con occhi completamente differenti; in secondo luogo, è possibile analizzare tutto ciò che ha funzionato e cosa non ha funzionato nei lavori di gruppo degli anni precedenti. In questo modo, lo studente può sia avere uno spazio di riflessione e analisi sia ricevere una serie di importanti feedback su tutto l'operato degli anni precedenti.

In parallelo, sarebbe opportuno creare (almeno) un nuovo corso da zero per lo sviluppo di competenze trasversali che andrebbero a completare la figura del neolaureato ingegnere: la leadership e il public speaking. Questo corso potrebbe essere inserito nel carico didattico del secondo periodo del primo anno o, al più, del primo periodo del secondo anno; avrebbe quattro obiettivi:

- presentare tutti i concetti teorici necessari per quanto riguarda le competenze trasversali derivanti dall'empatia e più richieste al giorno d'oggi per la figura dell'ingegnere (che costituisce l'apprendimento formale, ovvero il 10% del modello 70:20:10)
- fornire uno spazio per mettere in pratica quanto assimilato, creando quindi scenari ed esercitazioni su misura (che costituisce l'apprendimento tramite l'acquisizione di esperienza, ovvero il 70% del modello 70:20:10)
- lavorare in gruppo con gli altri studenti e ricevere continuamente feedback (che costituisce l'apprendimento sociale, ovvero il 20% del modello 70:20:10)
- porre le basi per tutti i prossimi corsi contenenti progetti tecnici da svolgere in lavori di gruppo

In aggiunta, ogni singolo corso del percorso universitario dovrebbe avere un progetto finale di gruppo. Se proprio non è possibile, si potrebbero sostituire le classiche esercitazioni di laboratorio o le relazioni di laboratorio con presentazioni orali finali in pubblico.

Infine, come sottolineato in qualche risposta del sondaggio, potrebbe essere particolarmente utile avere una serie di psicologi a disposizione settimanale per affrontare questo nuovo percorso. Ci si rende conto che per molte persone potrebbe essere uno sforzo molto grande tanto che il disagio provato porta addirittura a rinunciare

a sfruttare certe occasioni per mettersi in gioco. Al Politecnico di Torino è già stata messa a disposizione degli studenti la possibilità di usufruire di una piccola serie di incontri gratuiti, è opportuno proseguire in questa direzione.

Lo sport I giochi collaborativi sono sicuramente un elemento importante per lo sviluppo di alcune qualità relazionali e della personalità. Lo sport, pertanto, può essere visto come una naturale palestra per lo sviluppo delle competenze trasversali senza nemmeno che una persona se ne accorga. Il Politecnico di Torino ha già mosso enormi passi avanti con la creazione della squadra di calcio e di quella di pallavolo ma ancora non basta, perché tutto questo è accessibile a poche persone e, in aggiunta, per un periodo di tempo troppo breve. La squadra di calcio ha avuto l'opportunità di partecipare due volte al WEUFT (World Elite University Football Tournament), il torneo mondiale di calcio per le università. Il primo anno, da quando la squadra del Politecnico di Torino ha avuto l'occasione di partecipare, è stato svolto a Wuhan, il secondo a Guangzhou.
Oltre alla quantità innumerevoli di benefici non riguardanti la trattazione in questione, è essenziale anche dal punto di vista dello sviluppo di competenze relazionali che il Politecnico di Torino continui a promuovere le attività sportive tra gli studenti e che, inoltre, prosegua sia con la creazione di nuove squadre sia con l'organizzazione di molteplici attività sportive durante l'anno accademico.

Queste proposte e idee di miglioramento del percorso universitario di ingegneria potrebbero richiedere tempistiche medio lunghe per essere implementate, anche perché prima di tutto bisognerebbe formare e sensibilizzare i docenti in primis. Perciò, la volontà di creare una soluzione pratica e utilizzabile nel breve termine ha condotto allo sviluppo di una versione basilare di applicazione web, presentata nel capitolo 5, per supportare il corso di Engineering Empathy e favorire la collaborazione tra studenti e docenti in maniera multidisciplinare ([4]).
Inoltre, lo strumento dell'ACT (sottosezione **??**) è stato presentato e analizzato per andare a fornire un supporto immediato a tutti quegli studenti che magari provano del disagio, in attesa e nella speranza che in futuro vengano messi ulteriori psicologi a disposizione.

Si sottolinea, infine, come tutta l'analisi effettuata e l'applicazione siano volti a sensibilizzare le persone sull'importanza dello sviluppo dell'empatia e dell'ascolto attivo e, contemporaneamente, a permettere di creare uno spazio per trattare e potenziare le competenze trasversali correlate, rendendo così professionalmente completa la figura del neolaureato in ingegneria.

Capitolo 5

La piattaforma "The Engineering Empathy Way"

5.1 Obiettivi, organizzazione e funzionalità offerte

Dopo aver analizzato i risultati ottenuti dal sondaggio e dalle interviste, si è compresa la necessità di operare un miglioramento nel modo di insegnare e di mettere in pratica le competenze; alla luce di ciò, sono state formulate varie proposte per migliorare il percorso universitario ingegneristico e, in aggiunta, si è deciso di creare un nuovo progetto, chiamato The Engineering Empathy Way.
Si tratta di un'applicazione web volta a:

- supportare ed estendere il corso di Engineering Empathy
- creare una piccola community protetta e sicura per favorire la collaborazione tra studenti. L'obiettivo ultimo è che questa community sia costituita da tutti gli studenti, e non solo, del Politecnico di Torino
- supportare l'acquisizione, lo sviluppo e la messa in campo dell'empatia e delle altre competenze trasversali analizzate nelle sezioni precedenti, normalmente non favorite durante il percorso di studi ma che ad oggi sono fondamentali per uno sviluppo completo del neolaureato in ingegneria. Tutto questo supportando e sfruttando il modello 70:20:10, descritto nella sezione 3.11:
  - il 70% dell'apprendimento avverrà tramite la messa in pratica di esercizi mirati per le varie competenze. Questa costituirà l'esperienza necessaria affinché l'apprendimento non rimanga solo teorico. È fondamentale che l'apprendimento esperienziale non avvenga in maniera non strutturata, proprio per questo verranno creati esercizi su misura volti alla

realizzazione di contesti pratici in cui applicare quanto appreso nella formazione teorica. Inoltre, la maggior parte delle esercitazione prevederà la creazione di gruppi di lavoro per favorire la collaborazione

- il 20% dell'apprendimento è costituito dall'aspetto sociale. La creazione della community di pari serve proprio a questo, a fare in modo che si abbiano continui scambi e feedback da persone che condividono il medesimo percorso di crescita. La creazione di gruppi è volta sia all'apprendimento esperienziale sia all'apprendimento sociale, per avere sempre feedback immediati
- il 10% è costituito dalla formazione teorica, offerta tramite il materiale di studio proposto e le video pillole

La prima versione di questo progetto è stata sviluppata dal sottoscritto e dalla collega Elisa Stroscio. In particolare, il sottoscritto:

- ha scelto le tecnologie e gli strumenti da utilizzare e, in aggiunta, ha ideato tutte le principali funzionalità offerte dall'applicazione
- ha fornito alla collega una versione base e minimale di partenza in termini di codice, per quanto riguarda la struttura del frontend
- ha sviluppato l'intera infrastruttura del backend
- ha sviluppato l'intero meccanismo di messaggistica istantanea, sia per quanto riguarda il lato frontend sia il lato backend

Il resto del lavoro è momentaneamente in fase di svolgimento per conto della collega. In particolare, prima di procedere con lo sviluppo, la collega ha fornito un insieme di slide molto dettagliate su quello che sarebbe dovuto essere il risultato finale in termini visivi, tenendo in considerazione la struttura base prescelta dal sottoscritto.

Prima di partire con lo sviluppo del codice, il sottoscritto ha definito gli obiettivi, i risultati attesi, la suddivisione dei compiti e i requisiti dell'applicazione tramite un documento Google condiviso con la docente e la collega.
Il codice è sempre stato condiviso tra il sottoscritto e la collega tramite un repository privato su GitHub [30]: questo ha permesso di lavorare contemporaneamente sul codice, di averlo sempre aggiornato e di mantenere le varie versioni nel tempo. Una volta terminata la prima versione, l'intero codice è stato reso pubblico in un nuovo repository [31] creato dalla docente di riferimento. L'obiettivo e la speranza sono che tanti nuovi studenti contribuiscano a migliorare e rendere sempre più performante l'applicazione.

L'iscrizione alla piattaforma avviene tramite la mail istituzionale e la creazione di una nuova password. Per completare la registrazione, è opportuno che il nuovo utente clicchi su un link generato appositamente per lui: al momento attuale, questo link viene solamente stampato nella console ma sono state create tutte le funzioni necessarie per inviare questo link alla mail inserita, in futuro si utilizzerà un servizio mail apposito per effettuare questa operazione.

Al momento attuale, l'intera applicazione prevede tre sezioni in globale:

- *Gestione dei corsi:* in questa sezione, l'utente può accedere a tutto ciò che riguarda ogni corso/percorso di apprendimento: informazioni generali (quali obiettivi, risultati attesi e numero di persone per formare i gruppi di lavoro), materiale di studio, esercizi, video pillole, possibilità di caricare nuovo materiale e soluzioni di esercizi. In particolare, i docenti creatori del percorso, possono operare anche delle modifiche. Gli studenti possono richiedere di eseguire l'upload di un certo materiale di studio, per metterlo a disposizione di tutti, ma per diventare ufficialmente pubblico dev'essere prima verificato dal docente di riferimento. Infine, gli studenti appartenenti al corso di Engineering Empathy possono anche effettuare l'upload delle video pillole per metterle a disposizione di tutta la comunità

- *Gestione delle chat:* l'utente può creare delle nuove chat con la possibilità di messaggiare istantaneamente con persone che sono iscritte agli stessi corsi e può eliminare chat già presenti

- *Gestione del proprio profilo:* in questa sezione, l'utente può modificare le informazioni che riguardano il proprio profilo, in particolare nome e cognome, password e foto profilo

Infine, si è deciso di utilizzare come logo dell'applicazione la raffigurazione di una giraffa, simbolo della CNV e dell'empatia:

Figura 5.1. Logo dell'applicazione

5.2 Architettura

5.2.1 L'architettura a livelli

Negli anni '70, l'International Organization for Standardization (ISO) propose un modello concettuale, chiamato Open Systems Interconnection (OSI), con l'obiettivo di rendere standard diverse comunicazioni, organizzando in una serie di livelli (layer) tutti i protocolli che stavano nascendo in quel periodo e che sarebbero divenuti i protocolli di internet [32]. Si tratta di una vera e propria architettura a livelli, che consente di scomporre in maniera modulare un sistema complesso, rendendo molto più semplice l'implementazione dei servizi forniti ad ogni livello; ciascun protocollo appartiene a un livello, che fornisce un certo servizio, perciò è possibile parlare anche di modello di servizio. In particolare, ogni livello fornisce un servizio che effettua certe operazioni utilizzando i servizi del livello inferiore; pertanto, finché un certo livello fornisce il medesimo servizio al livello superiore, utilizzando i medesimi servizi dello strato inferiore, il sistema intero rimarrà invariato se si cambia il protocollo utilizzato ad un certo livello.

Lo stack internet, invece, è molto simile al modello ISO/OSI, ma presenta cinque livelli:

1. *fisico:* ha il compito di inviare sulla rete fisica tutti i bit che devono essere scambiati nella comunicazione; i protocolli di questo livello sono dipendenti dal mezzo fisico di comunicazione, come ad esempio la fibra ottica, il cavo coassiale, il doppino intrecciato e così via. Si occupa perciò del collegamento diretto tra i due host

2. *collegamento:* vede le sequenze di bit come una serie di pacchetti, chiamati frame.

3. *rete:* instradamento dei pacchetti nella rete. Il protocollo più utilizzato è IP e i pacchetti scambiati a questo livello si chiamano datagrammi.

4. *trasporto:* gestione del canale logico di comunicazione tra i due host. I protocolli più usati sono TCP e UDP. I pacchetti scambiati a questo livello si chiamano segmenti.

5. *applicazione:* sede di realizzazione dei servizi applicativi. Il protocolli più usati sono: HTTP, che favorisce lo scambio di documenti web tra due host, SMTP, per la posta elettronica, FTP, per lo scambio di file tra host, e DNS, per tradurre i nomi degli host. I pacchetti scambiati a questo livello si chiamano messaggi.

Questi cinque livelli corrispondono a cinque dei sette livelli dello stack ISO/OSI, che sono: fisico, collegamento, rete, trasporto, sessione, presentazione e applicazione. I due mancanti nello stack internet sono dunque i livelli:

- *sessione:* delimitazione e sincronizzazione dello scambio dati, fornendo un meccanismo per regolare la creazione e la chiusura di una connessione logica tra le due macchine in comunicazione

- *presentazione:* insieme di servizi (come ad esempio la crittografia, la compressione e la descrizione dei dati) volti all'interpretazione del significato dei dati scambiati, quindi analizzando la loro sintassi e semantica e non vedendoli più solamente come dei bit

Questi due livelli mancano completamente nello stack internet? La risposta a questa domanda è che i due livelli sono inglobati nel livello applicazione: sarà pertanto compito degli sviluppatori capire se i servizi offerti da quei due livelli sono necessari e, in caso affermativo, implementarli direttamente a livello applicativo.

5.2.2 Il protocollo HTTP

HTTP (Hyper-Text Transfer Protocol) è un protocollo che opera a livello applicazione, livello 7 dello stack; permette il trasporto di contenuti generici sul web e delle applicazioni basate sul web.

HTTP si appoggia tipicamente su TCP come protocollo di trasporto, a livello 4, ma in certi casi può essere conveniente utilizzare UDP. È considerato un protocollo semplice, intuitivo e leggero da utilizzare perché gestisce una serie di scambi elementari, ovvero un susseguirsi di richieste da parte del client e di risposte da parte del server; i pacchetti HTTP, che siano di richiesta o di risposta, sono formati da due parti ben distinte:

- *intestazione:* comprende alcune righe di testo con un preciso significato ed è terminata dalla sequenza di carriage return e line feed (CR/LF = 0x0d, 0x0a), il che la rende separata con una riga vuota dalla prossima sezione.

- *corpo:* il contenuto del corpo è specificato nell'intestazione, tramite il campo Content-Type (ad esempio MIME: text/html, image/gif); la lunghezza è specificata nel campo Content-Length dell'intestazione; il corpo può essere inoltre codificato, ciò viene specificato nel campo Content-Encoding dell'intestazione e, in tal caso, Content-Length indica la lunghezza del messaggio codificato, non di quello decompresso

Inoltre, HTTP non fornisce una gestione degli stati, nel senso che ogni richiesta è indipendente dalle altre e il server tantomeno è tenuto a memorizzare la storia delle richieste precedenti: è necessario, pertanto, utilizzare altri meccanismi per implementare una gestione degli stati. Un'altra caratteristica base di HTTP è il fatto che ogni pacchetto può contenere dati qualsiasi ed è proprio questo ciò che rende il protocollo indipendente dal contenuto.

Ogni transazione HTTP fa riferimento a una qualche operazione su una o più risorse informative, che vengono indicate tramite una URL (Uniform Resource Locator), una stringa del tipo *schema://hostname[:port]/path* che può essere facilmente scomposta in parti:

- *schema:* rappresenta il protocollo da utilizzare, che può essere HTTP o HTTPS

- *hostname:* l'indirizzo tramite cui è possibile raggiungere il server di riferimento,

- *port:* la porta è facoltativa e, insieme all'hostname, completa l'indirizzo a cui raggiungere il server; indicandone una in particolare, il client specifica con

quale porta vuole interagire, altrimenti le porte di default per http e https sono, rispettivamente, 80 e 443

- *path:* specifica la posizione della risorsa che si vuole ottenere all'interno del server, questo percorso può essere reale o meno in riferimento al file system del server

Con il termine risorsa si vuole indicare un qualsiasi contenuto informativo coinvolto nella transazione HTTP.

Si entra nel seguito maggiormente nel dettaglio riguardo alla composizione dei pacchetti di richieste e risposte.

Intestazioni Sia pacchetti di richiesta sia quelli di risposta possono contenere intestazioni generali (relative a proprietà generali riguardanti il messaggio trasmesso), intestazioni relative a un'entità (meta-informazioni sul corpo del messaggio nel caso in cui sia presente, oppure sulla URL); ci sono poi intestazioni relative alla richiesta e intestazioni relative alla risposta. Non si entra maggiormente nel dettaglio dei vari tipi di intestazione perché esula dagli scopi di analisi della piattaforma The Engineering Empathy Way.

Le richieste Quando il client effettua una richiesta HTTP, significa che vuole effettuare un'azione sull'oggetto che viene specificato nel path della URL. L'intestazione della richiesta contiene l'azione da svolgere, l'URL relativa dell'oggetto di riferimento su cui svolgere l'azione e la versione del protocollo utilizzata. Ci possono essere ulteriori sotto-intestazioni e, dopodiché, come specificato precedentemente, ci sarà una riga bianca a cui segue il corpo, se presente, della richiesta.
Esistono vari tipi di azione che si possono effettuare sulle risorse:

- *GET:* il client desidera semplicemente recuperare la risorsa specificata nella URL, perciò il corpo della richiesta sarà vuoto
- *POST:* in questo caso, il client solitamente desidera che il server prenda il contenuto del corpo della richiesta per creare una nuova risorsa e, molto probabilmente, memorizzarla. Ad esempio, il client potrebbe voler aggiungere un nuovo oggetto ad una certa collezione, oppure effettuare l'upload di un file, o altro ancora.
- *PUT:* il client vuole aggiornare una determinata risorsa, specificandone il nuovo contenuto nel corpo della richiesta
- *DELETE:* il client vuole eliminare una data risorsa, indicata sempre tramite la URL

- *HEAD:* è molto simile alla GET però prevede solo lo scambio delle intestazioni, questo viene fatto solitamente per validare la URL di riferimento
- *TRACE, OPTIONS, CONNECT, PATCH:* altre possibili azioni, poco rilevanti per The Engineering Empathy Way

Esistono un paio di proprietà che, a seconda dell'azione in questione, possono essere soddisfatte o meno:

- *Sicurezza:* l'azione è considerata sicura quando non determina dei cambiamenti nello stato del server
- *Idempotenza:* l'azione è idempotente se l'effetto provocato a seguito dell'invio di tante richieste identiche, è uguale all'invio di una sola richiesta di tale genere

Pertanto, le richieste di tipo POST non sono né idempotenti né sicure, le richieste di tipo HEAD e GET soddisfano entrambe le proprietà e PUT e DELETE sono idempotenti ma non sicure.

Le risposte Le risposte sono il risultato ottenuto a seguito di una certa richiesta. Ogni risposta contiene una linea di stato, comprendente la versione del protocollo usato dal server, il codice di stato, composto da 3 caratteri numerici e una descrizione in forma testuale del codice di stato; possono esserci, in aggiunta, altre sotto-intestazioni e, come spiegato precedentemente, una riga bianca che separa l'intestazione dall'eventuale corpo del pacchetto.

I codici di stato possono essere catalogati a seconda del valore della prima delle 3 cifre, che si riportano di seguito insieme ad esempi di codici più frequentemente utilizzati all'interno di The Engineering Empathy Way:

- *1xx:* messaggio informativo.
  - *100:* continua. Informa il client che può proseguire con l'invio del corpo della richiesta
- *2xx:* esito positivo.
  - *200:* la richiesta ha avuto un esito positivo e l'azione è stata portata a termine con successo.
- *3xx:* reindirizzamento.
- *4xx:* errore del client.
  - *400:* richiesta non valida. La richiesta presenta uno o più errori sintattici

  - *401:* non autorizzato. Il client non dispone dei privilegi sufficienti e necessari per effettuare l'operazione sulla risorsa
  - *403:* operazione non consentita. La richiesta non è autorizzabile da parte del server
  - *404:* non trovato. La risorsa specificata nella URL non è stata trovata

- *5xx:* errore del server.
  - *500:* errore interno al server. Si è verificato un errore all'interno del server e ciò non permette di portare a termine l'azione richiesta del client.
  - *501:* non implementato. Il metodo richiesto non è stato implementato sul server

Si riporta, a titolo di esempio, un'immagine della console durante l'esecuzione di The Engineering Empathy Way dove viene mostrata la risposta con esito positivo a seguito di una richiesta POST di autenticazione effettuata dal client al server durante un'interazione:

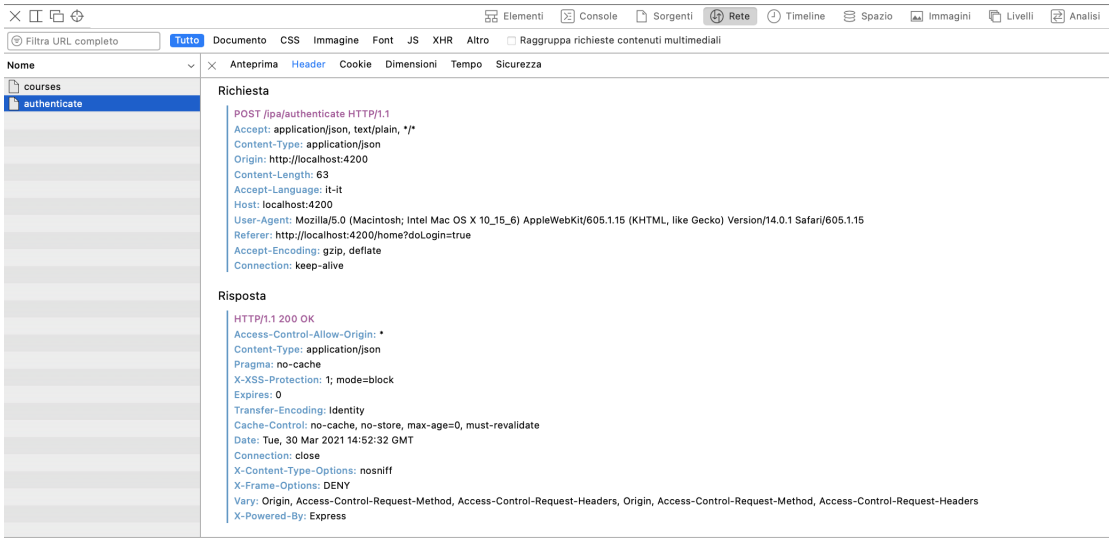

Figura 5.2. Richiesta di autenticazione con codice 200 di successo

Autenticazione L'autenticazione è un meccanismo molto importante da implementare. Come riportato su Wikipedia, "L'autenticazione è l'atto di confermare la verità di un attributo di una singola parte di dato o di una informazione sostenuto vero da un'entità. In contrasto con l'identificazione, che si riferisce all'atto di confermare l'identità di una persona o qualcosa, l'autenticazione è il processo di

confermare davvero l'identità" [33].

Il protocollo HTTP, in dettaglio, offre due meccanismi per l'autenticazione: basic authentication e digest access authentication. Per quanto riguarda The Engineering Empathy Way, si è scelto di utilizzare un meccanismo basato sui JWT, come descritto nella sezione appositamente dedicata alla sicurezza.

5.2.3 L'architettura dell'applicazione

The Engineering Empathy Way è costituita da due parti ben definite: il frontend che corrisponde al lato client, e il backend, che corrisponde al lato server.

Il frontend, analizzato nella sezione 5.3, è stato sviluppato utilizzando:

- il linguaggio di programmazione Typescript, un'estensione di Javascript. In particolare, è stato scelto il framework Angular
- Angular Material, ovvero l'implementazione in Angular dello standard di Google chiamato Material Design
- il paradigma a pagine singole, come spiegato nella sottosezione 5.3.3

Il backend, analizzato nella sezione 5.4, è stato sviluppato utilizzando:

- il linguaggio di programmazione Java e, in particolare, è stato scelto il framework Spring Boot, descritto nella sottosezione 5.4.2
- un database relazionale, MariaDB, analizzato nella sottosezione 5.4.7
- un servizio di gestione ed archiviazione file, MinIO, presentato nella sottosezione 5.4.8
- Docker sia per utilizzare i servizi di MariaDB e MinIO, sia per rendere interoperabile l'intera applicazione

L'applicazione, globalmente, può essere vista come un insieme di interazioni tra client e server, tramite l'utilizzo del protocollo HTTP e di API REST, e tra server e database (o il servizio di archiviazione file); il codice software, lato client, viene scaricato direttamente sulla macchina del client ed eseguito per mezzo del browser. Vi sono alcuni svantaggi nell'adozione di questo approccio: i browser non sono nativi e limitano l'accesso ad alcune funzionalità del sistema operativo (creando la cosiddetta *sandbox*), è opportuno rendere compatibile il codice per i browser più comunemente usati, dipende dall'utilizzo della rete (se l'utente è sconnesso, non può effettuare operazioni) e dipende dal server, che se non è funzionante rende impossibile l'utilizzo dell'applicazione.
I benefici di adottare questo approccio sono molteplici:

- Siccome i browser sono ampiamente disponibili e utilizzati in tutti i principali sistemi operativi, si ha la compatibilità su ogni sistema, che sia desktop, mobile o altro.

- Al giorno d'oggi l'esperienza degli sviluppatori in HTML, CSS e JS è molto avanzata, perciò è più semplice sfruttare tutta la conoscenza acquisita per costruire le proprie viste

- Ad ogni accesso all'applicazione, il codice viene scaricato nuovamente, perciò è molto semplice fare aggiornamenti e renderli direttamente disponibili a tutti

- Le richieste AJAX permettono di rendere dinamica l'interazione tra l'utente e l'applicazione, proprio come se stesse utilizzando una vera e propria applicazione desktop

5.3 Panoramica sul frontend

5.3.1 Introduzione

Il frontend è stato sviluppato utilizzando il linguaggio di programmazione Typescript, un'estensione di Javascript. In particolare, è stato scelto il framework Angular [34], con l'ausilio di Angular Material [35], ovvero l'implementazione in Angular dello standard di Google chiamato Material Design [36]. Questo capitolo vuole fornire un'introduzione all'architettura generale del frontend senza la pretesa di risultare esaustivo, una trattazione più completa verrà effettuata dalla collega di riferimento.

JavaScript e Typescript

JavaScript è un linguaggio di scripting progettato per operare su un sistema esistente e interpretato in fase di esecuzione direttamente dal browser lato client; è conforme alle specifiche di ECMAScript, uno standard dell'organizzazione ECMA per creare linguaggi general purpose di scripting, e viene utilizzato principalmente nella programmazione web, perché è in grado di dare dinamicità all'interazione dell'utente con le pagine web.

TypeScript è sempre un linguaggio di scripting, creato da Microsoft, che rappresenta un'estensione di JavaScript, in quanto permette di utilizzare tutte le funzionalità di JavaScript ma anche delle funzionalità aggiuntive; perciò qualsiasi programma scritto in JavaScript può essere considerato anche un programma in TypeScript perfettamente funzionante. TypeScript introduce alcune caratteristiche molto importanti, tra cui il fatto di poter avere più controllo sul codice tramite l'assegnazione

di tipi statici, l'utilizzo di interfacce, classi e annotazioni, cose che non possono essere fatte in JavaScript.

Transpilazione

La transpilazione è un processo che prevede una trasformazione e la compilazione, infatti sovente i *transpiler* sono chiamati anche *source-to-source compiler*. Uno strumento molto usato che viene in nostro aiuto proprio per effettuare questa operazione è Babel, che prende in input codice scritto in TypeScript, lo traduce in codice equivalente in JavaScript (source-to-source) e permette di compilarlo (compiler) per essere poi utilizzato nei browser più comunemente utilizzati al giorno d'oggi, come ad esempio - in ordine rigorosamente alfabetico - Firefox, Google Chrome, Microsoft Edge, Opera, Safari e così via.

Node.JS e npm

Node.JS [37] è un ambiente di esecuzione a run-time basato su JavaScript; è stato costruito sulla base del motore Google V8 JavaScript, è open source ed è compatibile con tutte le principali piattaforme, come MacOS X, Microsoft, Linux, Unix, e così via. Il più grande vantaggio che offre è quello di eliminare la sandbox tipica delle applicazioni eseguite su browser, permettendo di accedere anche alle funzionalità più avanzate del sistema operativo; viene pertanto utilizzato per il lato server delle applicazioni web basate su HTTP e per l'utilizzo di comandi da terminale e di molti strumenti di sviluppo. Si basa sulla programmazione asincrona, infatti elimina l'attesa sincrona nel contesto del singolo thread e prosegue l'esecuzione con le prossime istruzioni.

Npm, Node package manager [38], è una vastissima libreria software open source che è nata per gestire i moduli software di Node.JS: contiene più di ottocento mila moduli di codice, ovvero dei blocchi di codice riutilizzabile, che permettono di risolvere molti dei più generici problemi che si riscontrano durante lo sviluppo. Npm viene installato automaticamente quando si installa Node.JS nel proprio sistema e può essere tranquillamente utilizzato da linea di comando.
Per installare un nuovo pacchetto a livello globale nel sistema, è sufficiente eseguire il comando *npm install -g <package_name>*; se invece ci si trova all'interno della cartella di un progetto e si vuole installare lì dentro una nuova dipendenza, è sufficiente rimuovere il flag -g dal comando precedente. Ad esempio, per installare il transpilatore Babel, si utilizza il comando *npm install @babel/cli*.
Nel contesto di un progetto di un'applicazione, tutti i pacchetti necessari per il corretto funzionamento di un'applicazione sono indicati in un file chiamato *package.json*, il cui contenuto è scritto in JSON (JavaScript Object Notation); questo

file deve contenere almeno due campi, il nome e la versione. A gestire tutte le dipendenze dell'applicazione ci pensa appunto npm, che attraverso il comando *npm install* provvede a installare tutte le librerie necessarie. Si riporta di seguito il file *package.json* di The Engineering Empathy Way:

```
{
  "name": "teew",
  "version": "0.0.0",
  "scripts": {
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
    "test": "ng test",
    "lint": "ng lint",
    "e2e": "ng e2e"
  },
  "private": true,
  "dependencies": {
    "@angular/animations": "^11.0.5",
    "@angular/cdk": "^11.0.3",
    "@angular/common": "^11.0.5",
    "@angular/compiler": "^11.0.5",
    "@angular/core": "^11.0.5",
    "@angular/forms": "^11.0.5",
    "@angular/material": "^11.0.3",
    "@angular/platform-browser": "^11.0.5",
    "@angular/platform-browser-dynamic": "^11.0.5",
    "@angular/router": "^11.0.5",
    "@types/file-saver": "^2.0.1",
    "dateformat": "^4.4.1",
    "file-saver": "^2.0.5",
    "jquery": "^3.5.1",
    "net": "^1.0.2",
    "rxjs": "~6.5.4",
    "sockjs-client": "^1.3.0",
    "stompjs": "^2.3.3",
    "tslib": "^2.0.0",
    "zone.js": "~0.10.2"
  },
  "devDependencies": {
    "@angular-devkit/build-angular": "^0.1100.5",
    "@angular/cli": "^11.0.5",
    "@angular/compiler-cli": "^11.0.5",
    "@types/jasmine": "~3.6.0",
    "@types/jasminewd2": "~2.0.3",
    "@types/node": "^12.12.54",
    "@types/sockjs-client": "^1.5.0",
    "@types/stompjs": "^2.3.4",
    "codelyzer": "^6.0.0",
    "jasmine-core": "~3.6.0",
```

```
46      "jasmine-spec-reporter": "~5.0.0",
47      "karma": "~5.0.0",
48      "karma-chrome-launcher": "~3.1.0",
49      "karma-coverage-istanbul-reporter": "~3.0.2",
50      "karma-jasmine": "~4.0.0",
51      "karma-jasmine-html-reporter": "^1.5.0",
52      "moment": "^2.27.0",
53      "protractor": "~7.0.0",
54      "ts-node": "~8.3.0",
55      "tslint": "~6.1.0",
56      "typescript": "~4.0.5"
57    }
58  }
```

In particolare, l'oggetto *scripts* è utilizzato per automatizzare alcune operazioni che vengono effettuate tipicamente: ad esempio *npm start* viene tradotto in *ng serve* per avviare il frontend.

Grazie a npm è anche possibile installare la Angular CLI (Command Line Interface), che permette di eseguire una serie di comandi molto utili, tramite il comando *npm install -g @angular/cli*. Dopodiché, è possibile creare un nuovo progetto Angular tramite, ad esempio, il comando *ng new Teew_client*. Altri comandi molto utili sono:

- *ng generate component component_name –flat –module:* comando per creare un nuovo componente chiamato component_name
- *ng serve:* permette di far partire l'applicazione
- *ng build [–prod]:* serve ad eseguire il build del progetto e, se specificato, il flag –prod permette di creare la versione di produzione i cui file vengono salvati nella cartella /dist. In particolare, il codice scritto in Typescript viene transpilato e minimizzato (ovvero vengono rimossi tutti i caratteri non necessari dal codice sorgente senza alterare il corretto funzionamento) in alcuni file JavaScript
- *ng add <library-name>:* permette di aggiungere una libreria all'interno del progetto, come ad esempio ng add @angular/material. La differenza con il corrispondente comando npm è che quest'ultimo installa solamente la libreria senza configurare tutto il necessario al corretto utilizzo all'interno del progetto

5.3.2 Angular Material

Material Design [36] è uno standard che è stato inventato da Google per permettere ai team di sviluppatori, design e quant'altro di creare applicazioni (web, iOS, Android ecc.) che possano fornire un'ottima User Experience (UE) tramite un design di alta qualità progettato in maniera estremamente efficace.

Alcuni framework forniscono la propria implementazione di questo standard e, in particolare, anche Angular, che fornisce a chi sviluppa con Angular JS (Angular JavaScript) una libreria di componenti per la User Interface (UI, in italiano interfaccia utente): *Angular Material* [35]. I componenti di Angular Material sono proprio i mattoncini che, posti tutti insieme, vanno a costruire l'interfaccia utente: esistono componenti per la navigazione, per compiere azioni, per gli input (interazioni dell'utente), per la comunicazione o, più semplicemente, per effettuarne il display organizzando il contenuto.
Utilizzare Angular Material offre principalmente due vantaggi: sapere che la UI e la UX saranno efficaci e, non meno importante, non dover reinventare tutto da zero, perché si possono utilizzare componenti già preimpostati e pertanto ci si può concentrare direttamente sulla loro personalizzazione a proprio piacimento.

Per installare Angular Material nel proprio progetto, è sufficiente eseguire da terminale, dopo essersi spostati nella cartella del progetto (cd project) il seguente comando: *ng add @angular/material*, che installa anche il cdk e le animazioni. Per utilizzare i componenti di Angular Material, invece, è sufficiente importare i moduli necessari in app.module.ts

import MatButtonModule from '@angular/material/button'

e il componente corrispondente in app.component.ts

import MatButton from '@angular/material/button'

una volta fatto ciò, si possono utilizzare i componenti così importati a proprio piacimento.

Gli elementi di base di Angular Material sono i *componenti*: hanno il proprio tag (*<tag-name>*), hanno il proprio modello come classe TS, hanno un proprio template ed espongono un'interfaccia che mette a disposizione proprietà, metodi, eventi generati e dati che possono generare come output.

5.3.3 Angular

Angular [34] è un framework che è stato creato da Google, si basa su HTML, CSS e JS e serve allo sviluppo del frontend di un'applicazione web. Si basa sui componenti, introduce i concetti di routing, dependency injection, data binding e interagisce efficacemente con un server REST; in Angular è possibile creare template HTML, ovvero delle classi che corrispondono a componenti per gestire i template, effettuare il binding dei dati tra i componenti e il template, introdurre la logica di business e di modello attraverso l'utilizzo dei servizi e, infine, impacchettare i componenti e i servizi dentro a moduli facilmente riutilizzabili.

Single Page Application (SPA)

Esistono principalmente due tipi di applicazioni: *round trip*, quelle che effettuano il rendering lato client, e *single page* (a pagina singola), ovvero quelle moderne in cui il rendering è effettuato lato client.

In The Engineering Empathy Way si è scelto di adottare un'architettura a pagina singola: il server fornisce uno scheletro della pagina da visualizzare e ogni cambiamento necessario sarà applicato dal client, il quale richiede dinamicamente al server le risorse da mostrare, in maniera tale che non si debba ricaricare ogni volta la pagina intera da mostrare. Nessuna pagina viene ricaricata e non vi è nessun trasferimento di controllo ad altre pagine, in quanto ve n'è solamente una. Ciò che avviene è che, invece di cambiare pagina, viene cambiata una porzione della vista all'interno della medesima pagina; infatti, quando è necessario mostrare una nuova vista, essa viene generata direttamente dal client, che richiede le informazioni necessarie da mostrare sotto forma di JSON tramite richieste XHR a un server REST, e non più caricata dal server, cosa che invece avveniva nelle applicazioni tradizionali. Una SPA potrebbe essere vista semplicemente come un'insieme di componenti che rappresentano delle entità logiche: ognuna di queste possiede la propria rappresentazione grafica e i propri dati e implementa la propria parte di logica di funzionamento.

Il codice lato client diventa più complesso, in quanto gestisce una parte della logica di business dell'applicazione, come ad esempio il routing e più in generale il flusso di esecuzione; capita spesso infatti che prima di arrivare a contattare il server, si operi con lo strato di logica di business, ovvero il controller, direttamente del lato client. Le informazioni che vengono scambiate con il server sono riguardanti solamente il modello, in quanto la struttura visiva (HTML) delle informazioni è gestita completamente dal client.

MVC

Un'applicazione può essere vista come una *black box* (scatola nera): uno strumento informatico che consente all'utente di fornire degli input, tipicamente tramite un'interfaccia grafica (GUI = graphical user interface), per poi ricevere in risposta degli output a seconda delle richieste effettuate; tutte le informazioni e i vincoli necessari devono essere contenuti in essa. Al crescere della complessità dell'applicazione, risulta estremamente conveniente suddividere l'applicazione in strati e utilizzare un approccio modulare, in maniera tale che ciascuno strato abbia un compito ben preciso e che possa comunicare con gli strati adiacenti mediante interfacce, rispettando il principio di separazione delle responsabilità.

Il paradigma MVC è un pattern architetturale che serve proprio a stratificare l'applicazione, suddividendo logicamente l'applicazione in:

- *modello:* questo strato contiene i dati dell'applicazione e determina le regole di evoluzione degli stessi; esso rappresenta il punto di partenza nello sviluppo del progetto

- *vista:* rappresentazione visuale, solitamente sotto forma di interfacce grafiche, del modello - solitamente di una sua parte - per l'utente; la vista non modifica il modello, bensì ne riceve solamente il contenuto da presentare

- *controllore:* intercetta gli input dell'utente, o più in generale le richieste in input, convertendole in comandi per il modello e/o la vista; in pratica, estrae i parametri dalle richieste, sceglie l'interfaccia di servizio da richiamare per effettuare eventuali manipolazioni del modello e sceglie la vista atta a presentare all'utente il modello risultante dopo le operazioni

Il client recupera, tramite l'utilizzo dei servizi, il modello dei dati dal server, il quale espone tutti i dati sotto forma di JSON tramite delle *API* (Application Programming Interface) di tipo *REST* (come analizzato nella sottosezione 5.4.5). Una volta ricevuto il modello da visualizzare, il controllore di riferimento effettua il binding della porzione di modello necessaria alla vista (il template in Angular), che si occupa di effettuarne il display per l'utente. In Angular, in particolare, il controllore è sostituito dal ViewModel, quindi il modello MVC può essere considerato MVVM. Il ViewModel rappresenta il binder (ovvero un elemento che lega) che serve a collegare i dati tra la vista e il modello: ad esempio, il ViewModel espone alla vista i dati contenuti nel modello mentre la vista, a sua volta, associa i dati alle proprietà del ViewModel.

In particolare, lo stato dell'applicazione viene mostrato direttamente nella URL, in maniera tale che l'utente possa salvare l'URL di riferimento per poi ritornarci direttamente quando ne ha bisogno; pertanto, è necessario trovare un meccanismo che permetta di riflettere lo stato nella URL e questo viene effettuato tramite il *routing*, analizzato in seguito.

Struttura base

Il punto di inizio per l'interpretazione dell'intera applicazione è rappresentato dal file *index.html*, il quale contiene il componente <app-root> che viene personalizzato a seconda dell'applicazione. La struttura base, per effettuare il bootstrap dell'applicazione, prevede: *main.ts, app.module.ts (AppModule), app.component.ts (AppComponent), app-root (app.component.html)* e *index.html*; è presente anche il file *angular.json* che contiene le specifiche di configurazione del progetto, ad

esempio specifica il main file per effettuare il bootstrap dell'applicazione attraverso l'*AppModule*, che è annotato come il modulo root tramite il decoratore @NgModule ed è costituito da:

- *Declarations:* indica i componenti definiti all'interno del modulo; per utilizzare un componente nei template, dev'essere prima dichiarato qui, altrimenti Angular mostra un messaggio di errore
- *Imports:* indica le dipendenze del modulo
- *Providers:* indica i servizi che devono essere iniettati tramite il meccanismo di dependency injection (dipendenza in forma dichiarativa)
- *Bootstrap:* indica il componente radice che serve ad effettuare il bootstrap dell'applicazione

Dopodiché, il processo di bootstrap procede con la creazione di tutti i componenti che sono inseriti nella lista corrispondente e li inserisce all'interno dell'albero di componenti che vengono visualizzati poi all'interno del browser.
Tutto il codice è organizzato in moduli che rappresentano dei pacchetti (contenitori), ovvero un blocco con un insieme di funzionalità ben definite composto da codice sorgente, componenti, direttive, servizi e così via; ogni modulo presente all'interno del progetto dev'essere importato all'interno del modulo radice (AppModule). I moduli direttamente forniti da Angular, come ad esempio *HttpClientModule*, sono i cosiddetti *NgModules* ed è possibile importarli per utilizzarli. L'organizzazione per moduli favorisce la riusabilità.

Angular introduce il concetto di templating (un misto di JS e HTML) per poter integrare correttamente l'utilizzo delle variabili e delle stringhe tra HTML e JS. Per gestire le callback relative ad AJAX, vengono utilizzate funzionalità di HTML 5 combinate ai dati ricevuti dal server che implementa le API REST. Tutta la logica di presentazione è affidata al Template, che serve semplicemente a mostrare opportunamente i dati all'utente; esso non si occupa assolutamente di modificare il modello.

La logica di business viene implementata tramite i componenti (Component), che gestiscono il passaggio delle trasformazioni del modello ed effettuano il binding tra il Template e il modello di dominio; neanche i Component si occupano di modificare il modello. La logica del modello di dominio, infine, è affidata ai servizi che si occupano di gestire la comunicazione con il server per effettuare le opportune operazioni sul modello.

Componenti

I componenti sono degli elementi software in grado di contenere un insieme di funzionalità e dati correlati, con il compito di di contenere la logica di business e

mediare tra la vista e il modello, in maniera tale da permettere all'utente di interagire con l'applicazione; possono essere visti come le unità fondamentali che vanno a costruire l'interfaccia grafica. I componenti possono comunicare tra loro tramite delle opportune interfacce, ovvero i servizi; in particolare, si occupano di mostrare i dati, ricevere input dall'utente e compiere azioni opportune a seconda degli input ricevuti. Questo approccio favorisce l'indipendenza tra i vari componenti e ha il grande vantaggio di renderli sostituibili, fintanto che portano a termine le stesse identiche funzionalità, e anche riutilizzabili in più occasioni. Pertanto, le SPA possono essere considerate come un albero, la cui radice è rappresentata dall'applicazione stessa, di componenti organizzati secondo una gerarchia padre/figlio: ogni componente è responsabile di mostrare in maniera ricorsiva i propri figli e se stesso. Esistono due tipi di componenti: presentational e smart/container. I primi si occupano solamente di mostrare i dati, che vengono scambiati tramite interfacce, in maniera corretta all'utente, mentre i secondi mantengono lo stato dell'applicazione, che si riflette nella navigazione.

I componenti non sono altro che classi con il decoratore @Component. Il componente di default è AppComponent, che è mappato dal selettore *app-root* (utilizzato nel template in questo modo: <app-root></app-root>). I componenti corrispondono alle classi TypeScript, possono essere visti come controllori che gestiscono la logica e i dati e ad ognuno di essi corrispondono un file HTML di template e un file CSS di stile. In Angular è possibile definire dei tag personalizzati che vengono rimpiazzati in maniera dinamica con il proprio contenuto.
La classe TS del componente può interagire con il template in due modi: proprietà, che rappresentano i veri e propri dati che devono essere mostrati, e metodi, che corrispondono alle azioni che il template può attivare nel componente, che poi è responsabile di eseguirle. Il template ha il compito di mostrare i dati all'utente e può anche contenere un minimo di logica.

Ogni componente ha il proprio ciclo di vita: una nuova istanza della classe di un componente viene creata quando l'utente accede alla vista che concerne quel componente e, similmente, distrutta quando l'utente abbandona quella porzione di vista. È possibile intervenire nei vari momenti del ciclo di vita grazie a particolari funzioni messe a disposizione del programmatore, come ad esempio ngOnInit, ngOnDestroy, e così via. Ogni template può essere ulteriormente personalizzato tramite l'utilizzo di CSS.

Data binding

Il data binding permette ai dati di fluire dalla classe TS del componente al corrispondente template e questo può essere effettuato in quattro metodi, ma quelli utilizzati sono soltanto tre:

- *Interpolation:* riflette lo stato del modello dal componente al template (DOM). La sintassi è *{{ <expression> }}* in cui l'espressione viene prima valutata nel contesto del componente e in seguito convertita in stringa e sostituita direttamente all'interno del template, viene utilizzata in particolare per le proprietà del componente o per i valori ritornati da metodi, ma anche per gli input nei template o per le variabili referenziate direttamente nei DOM.

- *Property binding:* permette di impostare una proprietà di un certo elemento del template, quindi anche in questo caso i dati fluiscono dal componente al template. La sintassi è *[<property>]="<expression>"*

- *Event binding:* permette di aggiornare il modello perché permette di far fluire i dati dal DOM al componente. La sintassi è *(<eventName>)="<statement>"*, dove *statement* può essere la chiamata a un metodo passando certi argomenti; viene utilizzata soprattutto per gestire gli eventi scatenati da azioni compiute dall'utente

Servizi

I servizi sono utilizzati dai componenti e rappresentano lo strato che implementa la logica di business, gestisce lo stato dell'applicazione e permette di recuperare e sincronizzare il modello con il server. È importante che i servizi siano separati dai componenti, perché questi ultimi non devono recuperare o salvare dati, quindi più in generale non devono interagire con il/i server remoto/i; inoltre, non si devono preoccupare da dove provenga il modello, bensì devono soltanto occuparsi di presentarlo correttamente all'utente, delegando il resto delle responsabilità ai servizi. Per creare un servizio, si può utilizzare il comando *ng generate service <service-name>)*. Essi presentano la decorazione @Injectable in quanto i servizi non vengono mai istanziati direttamente dai componenti bensì iniettati grazie al meccanismo della Dependency Injection: il servizio dev'essere registrato tra i Provider e i componenti devono solamente dichiararne la dipendenza come proprietà privata attraverso il costruttore. In questo modo è possibile per Angular non dover creare una nuova istanza ogni volta dei medesimi servizi, bensì riutilizzare sempre la stessa.

In particolare, in The Engineering Empathy Way si è fatto ampio utilizzo di servizi HTTP, ovvero servizi che permettono di recuperare e scambiare dati con un server remoto in maniera asincrona tramite l'utilizzo del protocollo HTTP: è importante sottolineare come questa operazione sia effettuata in maniera asincrona, perché ci potrebbero essere problemi nella rete o nel server stesso e tutto ciò non deve precludere il corretto funzionamento del frontend, oltre che rallentarne eccessivamente l'esecuzione. Quando poi si riceve una risposta, viene richiamata un'apposita funzione di callback in grado di elaborare la risposta opportunamente.

Nella sezione relativa alla messaggistica istantanea (5.5), viengono presentati i servizi relativi al corretto funzionamento delle chat.
È possibile utilizzare un HttpClient importandolo dalla libreria @angular/common/http (import HttpClientModule from '@angular/common/http'), importandolo tramite @NgModules(imports: [HttpClientModule]) e, infine, iniettandolo nel costruttore del servizio in cui lo si vuole utilizzare.

Le API di HttpClient pongono le proprie basi sull'utilizzo delle API fornite dalla libreria *RxJS Observable*. Ad esempio, il metodo *HttpClient.get* ritorna un Observable con il corpo della risposta HTTP in formato JSON: questo significa che la funzione ritorna immediatamente senza avere ancora pronto il risultato, ovvero la risposta, che arriva in seguito. È possibile anche indicare il tipo del corpo che ci si aspetta, in questo modo informiamo il compilatore che può controllare se ne facciamo un utilizzo corretto. Per accedere al risultato dell'Observable, è possibile utilizzare il metodo *subscribe*, che accetta come argomento una funzione da utilizzare come callback quando il risultato sarà pronto.È opportuno effettuare la *unsubscribe* per evitare di creare perdite di memoria (memory leak), in quanto con la subscribe vengono mantenuti i riferimenti agli Observable; l'operazione di unsubscribe solitamente è effettuata quando il componente viene distrutto, evento che può essere intercettato tramite la funzione *ngOnDestroy*. Per gestire gli errori che ci possono essere a seguito dell'invio della richiesta, si può inserire una seconda callback nel metodo subscribe per gestire l'errore riscontrato.
In The Engineering Empathy Way tutto questo è stato usato per creare servizi che possano effettuare richieste HTTP, e di conseguenza ricevere risposte HTTP, per contattare il server che espone API di tipo REST. Nei servizi è sufficiente avere l'URL di base dell'insieme di API di riferimento e poi costruire le richieste alle URL specifiche per ogni caso.

Intercettatori HTTP

Gli intercettatori HTTP sono delle classi che implementano l'interfaccia *HttpInterceptor* (e il metodo *intercept()*) e che possono essere implementate; grazie ad esse è possibile intercettare le richieste e le risposte HTTP in maniera tale da ispezionarle per poi eventualmente applicarne opportune trasformazioni, che possono essere necessarie ad esempio per questioni di autenticazione e autorizzazione. Per poter utilizzare gli intercettatori, è necessario importarli in AppModule () e indicarli nei providers. È anche possibile creare una catena di intercettatori che elaborano le richieste e le risposte uno dopo l'altro; in questo caso, è fondamentale prestare attenzione all'ordine in cui vengono indicati nell'array providers, in quanto sarà proprio ciò a determinare l'ordine della sequenza in cui verranno eseguiti. Siccome le richieste e le risposte HTTP (*HttpRequest* e *HttpResponse*) sono oggetti immutabili, prima di effettuare le trasformazioni desiderate è necessario crearne una copia

che potrà essere modificata a proprio piacimento: questo viene effettuato tramite la funzione *clone*.

Routing

Angular si basa sul concetto di Single Page Application (SPA), che prevede che il server invii al client la struttura di base di una sola pagina per poi lasciare al client il compito di apporre modifiche, tramite l'utilizzo di codice JS, in base alle interazioni dell'utente. Nonostante ciò, è possibile navigare tra varie viste e rifletterne lo stato nella URL e ciò è permesso dal concetto di Routing: ogni vista avrà quindi la propria URL di riferimento, tramite la quale sarà appunto possibile recuperare la vista specifica desiderata. Ma perché riflettere lo stato dell'applicazione, ovvero la vista corrente, nella URL? Da un lato, per poter ricaricare la pagina mantenendo la stessa vista e, dall'altro, per salvare la URL in maniera tale da poter tornare in seguito sulla vista salvata desiderata e condividerla con altre persone, affinché possano accedere alla medesima vista.
In Angular, il Routing viene implementato attraverso l'utilizzo di tre componenti:

- *Routes:* serve a indicare tutte le route che l'applicazione supporta e implementa
- *RouterOutlet:* serve a indicare ad Angular dove mostrare il contenuto della route corrente, infatti il Router mostrerà il componente subito dopo il tag <router-outlet> utilizzato nel template HTML
- *RouterLink:* è una direttiva che serve a collegare e creare le route. Tramite essa, è possibile chiedere ad Angular di navigare a una specifica Route senza dover ricaricare la pagina

Inoltre, Angular mette a disposizione il servizio *Router* che, data una URL, mostra il componente opportuno. Per utilizzarlo, bisogna importarlo e iniettarlo nei componenti. In particolare, quando la URL nel browser cambia, il Router si occupa di cercare la *Route* corrispondente in maniera tale da determinare il componente che dev'essere mostrato. La Route radice dell'applicazione è rappresentata dall'AppComponent che serve da base per poi navigare nelle altre viste.
È possibile in ogni momento recuperare la URL dello stato in cui ci si trova attraverso un servizio router chiamato *ActivatedRoute*: il Router, in particolare, mantiene lo stato delle Route (*RouterState*) visitate tramite la costruzione di un albero delle ActivatedRoute, perciò è sempre possibile poter risalire a tutti i percorsi effettuati.
In aggiunta, Angular offre le *Route Guards*, delle classi che servono a verificare se la Route e il componente in questione possono essere visitati dall'utente corrente, verificando in particolare l'autenticazione e l'autorizzazione dell'utente. Queste classi ritornano vero se la navigazione può procedere, falso altrimenti; possono anche decidere di reindirizzare l'utente su un'altra Route, eliminando pertanto la richiesta

effettuata per una certa Route e, di conseguenza, l'intera navigazione desiderata dall'utente. Questo meccanismo è implementato tramite due interfacce in particolare: *CanActivate*, che permette di mediare la navigazione a una certa Route, e *CanDeactivate*, che permette invece di mediare la navigazione quando si sta uscendo dalla Route corrente, passando il componente che sta per essere disattivato come parametro della funzione.

Si riporta di seguito il file *app-routing.module.ts*, escluse le importazioni necessarie, utilizzato per configurare tutte le *route* necessarie al corretto funzionamento della navigazione nelle varie pagine dell'applicazione:

```
const routes: Routes = [
  { path: 'home', component: HomeComponent, canActivate: [
   AlreadyAuthGuard]},
  { path: '',redirectTo: '/home', pathMatch: 'full'},

  { path: 'chats', canDeactivate: [ChatGuard], component:
   ChatViewComponent,
    children: [
      { path: '', component: ChatViewHomeComponent},
      { path: ':chatId', resolve: {chat: ChatGuard}, component:
   MessageListComponent},
      { path: '**', component: NotFoundComponent}
    ]
  },

  { path: 'notification/:action/:token', component:
   NotificationComponent},

  { path: 'teacher', canActivate: [AuthGuard], canDeactivate: [
   CourseGuard], component: TeacherComponent,
    children: [
      { path: 'home', component: TeacherHomeComponent },
      { path: '', redirectTo: 'home', pathMatch: 'full'},
      { path: 'courses/:courseName', resolve: {course: CourseGuard
   },
        children: [
          { path: 'home', component: CourseHomeContComponent},
          { path: 'students', component: StudentsContComponent,
   },
          { path: 'material', component:
   TeacherStudyMaterialComponent},
          { path: 'assignments',component: SelfTestComponent },
          { path: 'pills', component: TeacherVideoPillsComponent
   },
          { path: 'teams', component: TeacherTeamsContComponent},
          { path: 'uploads', component:
   TeacherUploadRequestsComponent}
        ]
```

```
29        },
30        { path: '**', component: NotFoundComponent}
31      ]
32    },
33
34    { path: 'student', canActivate: [AuthGuard], canDeactivate: [
      CourseGuard], component: StudentComponent,
35      children: [
36        { path: 'home', component: StudentHomeComponent, },
37        { path: '',redirectTo: 'home', pathMatch: 'full'},
38        { path: 'courses/:courseName', resolve: {course: CourseGuard
      }, canDeactivate: [CourseGuard],
39          children: [
40            { path: 'home', component: StudentCourseHomeComponent},
41            { path: 'material', component:
      StudentStudyMaterialComponent},
42            { path: 'tests',component: StudentSelfTestComponent},
43            { path: 'pills', component: StudentVideoPillsComponent},
44            { path: 'teams', component: GroupsContComponent},
45            { path: 'uploads', component:
      StudentUploadRequestsComponent}
46          ]
47        },
48        { path: '**', component: NotFoundComponent},
49      ]
50    },
51
52    { path: '**', canActivate:[AlreadyAuthGuard], component:
      PageNotFoundComponent}
53  ];
54
55
56  @NgModule({
57    imports: [
58      RouterModule.forRoot(routes, { enableTracing: false,
      relativeLinkResolution: 'legacy' })
59    ],
60    exports: [RouterModule]
61  })
62  export class AppRoutingModule { }
```

Risulta evidente come ci siano 5 route principali: */home, /chats, /teacher, /student*, più quella relative alle notifiche (per il processo di registrazione alla piattaforma). Tutte le richieste relative a queste route vengono intercettate da *AlreadyAuthGuard* che verifica se l'utente è già autenticato oppure no. Le route */chats, /teacher e /student* presentano delle route figlie, che vengono gestite dai corrispondenti componenti e servono a fornire le varie funzionalità che vengono offerte agli utenti.

5.4 Il backend

5.4.1 Introduzione

Il backend è stato sviluppato utilizzando il linguaggio di programmazione Java e, in particolare, è stato scelto il framework Spring Boot [43]. Inoltre, esso utilizza un database *MariaDB* relazionale, per salvare tutti i dati necessari al corretto funzionamento dell'applicazione, e un servizio di gestione ed archiviazione file, *MinIO* (Sottosezione 5.4.8).

Si trattano di seguito alcuni strumenti avanzati e concetti rilevanti per lo sviluppo del codice e le analisi successive: le funzioni lambda, gli stream e le annotazioni in Java; si vuole, inoltre, fornire una panoramica sull'architettura generale delle applicazioni basate su JavaEE.

Funzioni lambda

Le funzioni lambda, anche chiamate funzioni anonime, non sono altro che delle funzioni che vengono definite senza un identificatore, ovvero un nome tramite il quale richiamarle; quello che fanno in maniera "celata" è creare un'interfaccia funzionale (dichiarata tramite l'annotazione *@FunctionalInterface*), ovvero composta da un unico metodo astratto. Gli esempi più classici di interfacce funzionali "pre-built" in Java sono: *java.lang.Runnable, java.util.Comparable e java.util.concurrent.Callable.*

Grazie ad esse, è possibile introdurre funzioni con una sintassi molto compatta e di facile interpretazione; dopo aver introdotto gli stream, si riporterà un esempio completo che mostra anche l'utilizzo della sintassi più comune per le funzioni lambda.

Stream

Uno stream è un flusso di dati utilizzato per processare una collezione di elementi, creando una pipeline offrendo la possibilità di applicare dei metodi per ottenere un risultato desiderato ed elaborando gli elementi in maniera funzionale. Per creare un nuovo stream a partire da una collezione, si utilizza il metodo *stream()* applicato direttamente sulla collezione. Vi sono due tipi di operazioni all'interno degli stream: le operazioni intermedie e le operazioni terminali.
Lo stream non cambia la struttura dati originale, semplicemente permette di ottenere il risultato desiderato tramite una serie di operazioni in pipeline; opera in maniera *lazy* (pigra), ovvero che se non c'è un'operazione terminale non viene eseguita nemmeno nessuna operazione intermedia.
Le operazioni intermedie servono a modificare il flusso di dati in input, producendo un nuovo stream in output che sarà input della seguente operazione. Le operazioni

terminali segnano la fine dello stream e servono a ritornare il risultato finale ottenuto dopo aver applicato una serie di trasformazioni al flusso iniziale. Di seguito un esempio di operazioni intermedie più utilizzate in The Engineering Empathy Way:

- *map:* applica una certa funzione ad ogni elemento del flusso
- *flatmap:* effettua una map sul flusso in input e il flusso in output è l'insieme dei risultati ottenuti applicando la funzione della map ad ogni elemento del flusso.
- *filter:* applica un filtro ad ogni elemento del flusso, chi soddisfa la/e condizione/i continua il suo percorso all'interno del flusso
- *sorted:* applica un ordinamento al flusso in input

Di seguito un esempio operazioni terminali più utilizzate in The Engineering Empathy Way:

- *collect:* inserisce e raggruppa tutti gli elementi risultanti dalla pipeline di operazioni intermedie applicate precedentemente in una nuova collezione (grazie ai Collectors)
- *forEach:* itera su ogni elemento del flusso applicando una certa funzione data

Si riporta, di seguito, un esempio che mostra l'utilizzo degli stream combinato alle funzioni lambda (all'interno del metodo *filter*)

```
1 List<Team> teams = teamRepository.findAll();
2 Optional<Team> teamOptional = teams.stream()
3         .filter(t -> t.getCourse() == course)
4         .filter(t -> t.getMembers().contains(student) && t.
    getStatus().equals(TEAM_ACTIVE))
5         .findFirst();
```

Annotazioni

Le annotazioni sono un potente strumento che serve ad inserire e aggiungere dei metadati e descrivono caratteristiche di package/classi/metodi/campi (e, solo in fase di compilazione, anche parametri e variabili locali) e che sono disponibili durante l'esecuzione del programma, all'interno del codice. Esse permettono di informare il compilatore (ad esempio per ignorare warning o rilevare errori utili), condizionare l'esecuzione di un programma o ancora per generare informazioni durante la compilazione di un intero package.

Java permette al programmatore di definire delle nuove annotazioni, ma questo esula dagli scopi di analisi dell'applicazione The Engineering Empathy Way. Ciò

che è più rilevante prendere in considerazione è come utilizzare annotazioni già definite e messe a disposizione del programmatore tramite librerie/framework. In particolare, le annotazioni, devono essere poste subito prima della dichiarazione degli elementi ai quali si riferiscono: la sintassi prevede una chiocciola, seguita dal nome dell'annotazione e, tra parentesi tonde, è possibile inserire una lista di coppie nome=valore separate da virgola. Un esempio di annotazione, il cui significato sarà più chiaro nel seguito della trattazione della sottosezione 5.4.7, è:

```
1 @OneToMany(mappedBy = "assignment") private List<Feedback>
    feedbacks;
```

Le annotazioni, maggiormente utilizzate durante la fase di sviluppo di The Engineering Empathy Way, risalgono al progetto Lombok [47], che permette di compattare e risparmiare di molto il codice da scrivere cosiddetto boilerplate (insiemi di righe di codice che si ripetono sempre in maniera identica o del tutto simile, cosa che succede spesso utilizzando un linguaggio verboso come Java). Il progetto Lombok viene ripresi nella sottosezione services.

Si vuole sottolineare come le annotazioni siano un ottimo strumento per risparmiare tempo, energie e possibilità di commettere errore in grandi quantità di codice che svolgono funzionalità minime.

Applicazioni JavaEE

Java Enterprise Edition (JavaEE) [39] è una tecnologia che racchiude una serie di specifiche atte a supportare la messa in campo di applicazioni web; sebbene appaia obsoleta perché risalente a circa vent'anni fa, è importante mostrarne le caratteristiche fondamentali poiché è la base da cui partono gli sviluppi odierni.

Esistono almeno due tipi di navigazione tra cui è possibile scegliere:

- *navigazione pagina per pagina:* ogni volta che si riceve una richiesta, viene generata una risposta contenente l'intera pagina HTML da mostrare all'utente, pagina che va a sostituire completamente quella precedentemente visualizzata. Questa forma di navigazione è molto semplice da implementare, ma allo stesso tempo particolarmente inefficiente.

- *navigazione a pagina singola:* in questo caso, invece, la struttura HTML della pagina viene inviata soltanto a fronte della prima richiesta ricevuta e, a partire dalla seconda, le richieste vengono effettuate attraverso codice JavaScript secondo il paradigma AJAX, consentendo il trasferimento solamente della/e parte/i che necessitano di essere aggiornate, o addirittura solo di una descrizione del cambiamento necessario. Il framework Angular, adottato per lo sviluppo client, procede proprio in questo modo. Seppur sia vero che in

questo modo si ottiene un'efficienza più elevata, bisogna ricordare che questo modo di procedere rende più complicata la gestione della navigazione lato client, la gestione del ciclo di vita delle pagine (si può dover ricorrere a tecniche di download progressivo per non mostrare ritardi durante la fase di renderizzazione) e, infine, dell'indicizzazione nei motori di ricerca: sappiamo benissimo che, al giorno d'oggi, per un'attività commerciale, il fatto di non apparire nei risultati di ricerca (o comunque troppo in fondo) equivale quasi a non esistere sul mercato.

Pertanto, le applicazioni JavaEE possono essere considerate come un insieme di classi Java all'interno di un programma contenitore, che ne gestisce il ciclo di vita e l'esecuzione; in particolare, queste applicazioni sono indipendenti sia dal contenitore di riferimento sia dal sistema operativo del server sui cui vengono eseguite. I *contenitori (container)* sono quindi a loro volta delle applicazioni che vengono eseguite all'interno del server con il preciso scopo di gestire le applicazioni JavaEE, fornendo alcune funzionalità fondamentali:

- monitoraggio delle connessioni in entrata provenienti dalla rete
- possibilità di applicare funzioni alle richieste entranti (ad esempio analisi della richiesta, gestione delle sessioni, controlli di sicurezza...)
- scelta dell'applicazione a cui la richiesta è destinata, nonché addirittura della classe, con eventuale istanziazione, specifica che ha il compito di elaborarla
- inoltro della risposta al client, eventualmente applicando delle codifiche o trasformazioni necessarie
- offerta di servizi standardizzati per i componenti, cosicché risparmino complessità di implementazione

I container usufruiscono di un template standard per descrivere le applicazioni web: ogni applicazione deve trovarsi in una specifica e propria sottocartella, dev'essere distribuita in formato *.war* (Web ARchive, ovvero un insieme di file *.jar*, che può essere facilmente ottenuto grazie ad uno specifico comando Maven) e dev'essere pubblicata a partire da una URL di base; un file XML, in aggiunta fornisce al container dettagli di configurazione, come ad esempio dei parametri, corrispondenza fra URL e classi, politiche di gestione della sessione e vincoli di sicurezza. Nell'archivio war, devono essere presenti due sottocartelle con due file fondamentali: *META-INF/MANIFEST.MF* per descrivere le proprietà dei file (come ad esempio autore, versione ecc.) e *WEB-INF/web.xml* che descrive la configurazione dell'applicazione. Per installare un'applicazione JavaEE è sufficiente copiare e incollare l'archivio nella cartella corrispondente all'interno del container.

All'interno del container, vi sono inoltre i componenti di base di un'applicazione JavaEE:

- *Servlet:* classe Java, alla quale è possibile accedere tramite delle Servlet API (interfacce e classi che determinano l'interazione tra il container e il codice applicativo) che gestisce le richieste in entrata al container. In particolare si distinguono le due interfacce *javax.servlet.http.HttpServletRequest* e *javax.servlet.http.HttpServletResponse* (che derivano da HttpServlet, che è una specializzazione di Servlet per il protocollo HTTP, estendendo *ServletRequest* e *ServletResponse*, a loro volta appartenenti al package javax.servlet e indipendenti dal protocollo utilizzato).

- *Filtri:* componente java che permette di applicare delle trasformazioni alle richieste/risposte ricevute dal servlet. Ogni filtro può modificare richieste/-risposte relative ad un insieme di URL e, in base alla logica di implementazione, può semplicemente inoltrare, modificare dei parametri o, addirittura, reindirizzare ad un indirizzo completamente diverso. I filtri sono tipicamente utilizzati per operazioni come autenticazione, compressione dati, conversione di formato, crittografia, log delle richieste/risposte e così via.

- *Pagine JSP:* documenti testuali che spiegano come elaborare le richieste, per creare una risposta in base ai parametri ricevuti

- *Listener:* elemento che governa il ciclo di vita, le politiche di attivazione e controlla le risorse (e la concorrenza) di tutto l'insieme dei componenti presenti

In The Engineering Empathy Way si è scelto di utilizzare *Apache Tomcat* perché semplice, intuitivo, gratuito e interoperabile. L'applicazione si trova in una directory, il cui nome è per default la radice dell'URL dell'applicazione, nella cartella *webapps*: qui vi sono le JSP e la sottocartella WEB-INF, contenente il file web.xml e le classi, in una sottocartella.

Maven

Maven [40], un progetto open source di Apache Foundation nato nel 2003, è uno strumento di supporto per l'intero ciclo di vita del software, dalla creazione alla gestione/manutenzione. Maven supporta l'intero processo di compilazione del codice sorgente: compilazione del codice sorgente, collegamento con le risorse, compilazione ed esecuzione dei test, preparazione del pacchetto finale, messa in campo e rimozione dei file intermedi.

Per portare a termine le proprie funzionalità, Maven usufruisce di un file *POM* (Project Object Model), scritto nel linguaggio di markup XML (Extensible Markup

Language). In questo file vengono indicati il nome e la versione dell'applicazione, il tipo di artefatto, le cartelle contenenti il codice sorgente, l'elenco delle dipendenze, i profili e i plugin per l'adattamento del processo. Il progetto e gli artefatti sono identificati in maniera univoca tramite la sintassi *GAV* (groupId:artifactId:version): il *groupId* rappresenta l'identificatore del progetto, che di solito deriva direttamente dal pacchetto Java, l'*artifactId* è un identificatore Java per il nome del progetto e *version* è la versione del progetto (qualora fosse presente il suffisso -SNAPSHOT, si vuole indicare che vi è uno sviluppo in corso). L'elemento *<packaging>* identifica il tipo di artefatto da costruire: quando viene omesso, viene utilizzato *jar* come valore di default.
In particolare, per generare le dipendenze necessarie al corretto funzionamento dell'applicazione, si possono utilizzare due strumenti: Spring Initializr [41] e Maven Repository [42]. Tramite questi strumenti è molto semplice ricercare i pacchetti e le librerie di cui il progetto si serve, per poi inserire le dipendenze direttamente all'interno del proprio file POM.
Per far riferimento alle dipendenze, si utilizza sempre la sintassi GAV precedentemente descritta, eventualmente aggiungendo l'elemento <scope> per indicare in quale fase del ciclo di vita del progetto dev'essere presa in considerazione (compile, per l'intero ciclo di vita, provided, in fase di compilazione, runtime, in fase di esecuzione, o test, in fase di test).
Vanno specificati, infine, i plugin che vengono utilizzati per la gestione del progetto, sempre tramite la notazione GAV; i plugin servono a rendere possibile lo svolgimento di tutte le operazioni effettuate da Maven.

Nella sottosezione seguente, 5.4.2, viene riportato il file *pom.xml* del progetto The Engineering Empathy Way.

5.4.2 Spring Boot

Spring Boot [43] è un'estensione di Spring, ovvero un framework per il linguaggio di programmazione Java che si basa su una serie di oggetti che interagiscono tra di loro per offrire e consumare certi tipi di servizi in linea con la logica di business. Perciò, il codice scritto dai programmatori solitamente prevede una parte di logica, che concerne le classi di ogni singolo oggetto che viene istanziato, e una parte di configurazione, che regola come gli oggetti vengono istanziati e collegati tra loro; ovviamente, la somma delle due parti può rendere particolarmente complessa la scrittura e la manutenzione del codice.
Alla luce di ciò, il framework Spring rende dichiarativa la responsabilità di individuare e connettere tutti gli oggetti che vivono all'interno dell'applicazione, cosa che normalmente avviene in forma procedurale; in questo modo, il programmatore deve soltanto definire le classi e dichiarare, tramite delle opportune annotazioni che semplificano il meccanismo, il modo in cui sono interconnessi gli oggetti, ovvero

la configurazione: così il framework si assume la responsabilità di generare tutto il codice necessario per fare in modo che il sistema funzioni correttamente rispettando le dipendenze desiderate dal programmatore. Quindi il collegamento degli oggetti avviene in maniera automatica proprio grazie a Spring, che si prende l'onere di analizzare le annotazioni usate dal programmatore e produrre il codice necessario. Ciò che realmente rende possibile tutto ciò è la presenza di un contenitore che mantiene riferimenti a tutti gli oggetti presenti all'interno dell'applicazione, gestendone interamente il ciclo di vita.

Spring boot offre, in aggiunta alle funzionalità offerte da Spring, la possibilità di eliminare molto codice boilerplate necessario a configurare le applicazioni. Per creare un progetto Spring Boot, solitamente si ricorre a *Spring Initializr*, il quale permette al programmatore di selezionare le dipendenze desiderate e di includerle direttamente nel file POM generato al momento della creazione del progetto. In particolare, il framework fornisce più di 50 starter-pack comprendenti librerie e configurazioni in grado di risolvere i tipici problemi iniziali nella creazione di un progetto di questo tipo. Si riporta di seguito il file POM generato per l'applicazione The Engineering Empathy Way:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.2.6.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>

    <groupId>it.polito.ai</groupId>
    <artifactId>teew_server</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>TeewServer_backend</name>
    <description>Backend for The Engineering Empathy Way application. Exposes a REST API.</description>

    <properties>
        <java.version>1.8</java.version>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
```

```
26              <artifactId>spring-boot-starter-security</artifactId>
27          </dependency>
28          <dependency>
29              <groupId>io.jsonwebtoken</groupId>
30              <artifactId>jjwt</artifactId>
31              <version>0.9.1</version>
32          </dependency>
33          <dependency>
34              <groupId>org.springframework.boot</groupId>
35              <artifactId>spring-boot-starter-mail</artifactId>
36              <version>2.2.6.RELEASE</version>
37          </dependency>
38          <dependency>
39              <groupId>org.springframework.hateoas</groupId>
40              <artifactId>spring-hateoas</artifactId>
41              <version>1.0.4.RELEASE</version>
42          </dependency>
43          <dependency>
44              <groupId>com.opencsv</groupId>
45              <artifactId>opencsv</artifactId>
46              <version>5.0</version>
47          </dependency>
48          <dependency>
49              <groupId>org.modelmapper</groupId>
50              <artifactId>modelmapper</artifactId>
51              <version>2.3.7</version>
52          </dependency>
53          <dependency>
54              <groupId>org.springframework.boot</groupId>
55              <artifactId>spring-boot-starter-data-jpa</artifactId>
56          </dependency>
57          <dependency>
58              <groupId>org.springframework.boot</groupId>
59              <artifactId>spring-boot-starter-thymeleaf</artifactId>
60          </dependency>
61          <dependency>
62              <groupId>org.springframework.boot</groupId>
63              <artifactId>spring-boot-starter-web</artifactId>
64          </dependency>
65          <dependency>
66              <groupId>mysql</groupId>
67              <artifactId>mysql-connector-java</artifactId>
68              <scope>runtime</scope>
69          </dependency>
70          <dependency>
71              <groupId>org.projectlombok</groupId>
72              <artifactId>lombok</artifactId>
73              <optional>true</optional>
74          </dependency>
75          <dependency>
```

```
76              <groupId>org.springframework.boot</groupId>
77              <artifactId>spring-boot-starter-test</artifactId>
78              <scope>test</scope>
79              <exclusions>
80                  <exclusion>
81                      <groupId>org.junit.vintage</groupId>
82                      <artifactId>junit-vintage-engine</artifactId>
83                  </exclusion>
84              </exclusions>
85          </dependency>
86          <dependency>
87              <groupId>org.springframework</groupId>
88              <artifactId>spring-test</artifactId>
89          </dependency>
90          <dependency>
91              <groupId>com.google.api-client</groupId>
92              <artifactId>google-api-client</artifactId>
93              <version>1.31.1</version>
94          </dependency>
95          <dependency>
96              <groupId>com.google.api-client</groupId>
97              <artifactId>google-api-client-java6</artifactId>
98              <version>1.31.1</version>
99          </dependency>
100         <dependency>
101             <groupId>com.google.oauth-client</groupId>
102             <artifactId>google-oauth-client-jetty</artifactId>
103             <version>1.31.2</version>
104         </dependency>
105         <dependency>
106             <groupId>com.google.oauth-client</groupId>
107             <artifactId>google-oauth-client</artifactId>
108             <version>1.31.2</version>
109         </dependency>
110         <dependency>
111             <groupId>com.google.apis</groupId>
112             <artifactId>google-api-services-drive</artifactId>
113             <version>v3-rev197-1.25.0</version>
114         </dependency>
115         <dependency>
116             <groupId>org.springframework.boot</groupId>
117             <artifactId>spring-boot-starter-websocket</artifactId>
118             <version>2.4.1</version>
119         </dependency>
120         <dependency>
121             <groupId>io.minio</groupId>
122             <artifactId>minio</artifactId>
123             <version>8.0.3</version>
124         </dependency>
125     </dependencies>
```

```
126
127     <build>
128         <plugins>
129             <plugin>
130                 <groupId>org.springframework.boot</groupId>
131                 <artifactId>spring-boot-maven-plugin</artifactId>
132             </plugin>
133             <plugin>
134                 <groupId>org.apache.maven.plugins</groupId>
135                 <artifactId>maven-compiler-plugin</artifactId>
136                 <configuration>
137                     <source>9</source>
138                     <target>9</target>
139                 </configuration>
140             </plugin>
141         </plugins>
142     </build>
143 </project>
```

In particolare, il plugin Boot-Maven permette di fondere automaticamente tutti i file jar del progetto in un unico file jar finale, il che semplifica notevolmente l'importazione dello stesso all'interno di *Docker*, strumento analizzato nella sottosezione 5.6.1. Si vuole far notare che in questo momento ci sono alcune dipendenze che non risultano essere utilizzate: tutte quelle relative alle API necessarie all'utilizzo di Google Drive (righe 93-118), all'utilizzo di Spring Hateoas (righe 41-44, 51-54) e al servizio mail (righe 36-39). Queste dipendenze sono incluse perché durante lo sviluppo dell'applicazione è stata fornita una base di partenza per implementare tali funzionalità.

L'applicazione basata su Spring ha, come punto di ingresso, la classe TeewServerApplication:

```
1  @SpringBootApplication
2  @Log
3  public class TeewServerApplication {
4
5      @Value("${spring.mail.username:}")
6      private String springMailUsername;
7
8      @Value("${spring.mail.password:}")
9      private String springMailPassword;
10
11     @Value("${FRONTEND_HOST:}")
12     private String frontendHost;
13
14     @Value("${FRONTEND_PORT:4200}")
15     private String frontendPort;
16
17     @Value("${USE_REAL_EMAIL_ADDRESS:false}")
```

```
18      private boolean useRealMailAddress;
19
20      @Value("${DEFAULT_MAIL_ADDRESS:carlitosstorre@gmail.com}")
21      private String defaultEmailAddress;
22
23      @Bean
24      ModelMapper modelMapper() {
25          return new ModelMapper();
26      }
27
28      @Bean
29      @Qualifier("userDetailsService")
30      JwtUserDetailsService userDetailsService() {
31          return new JwtUserDetailsService();
32      }
33
34      @Bean
35      @Qualifier("jwtUserDetailsService")
36      JwtUserDetailsService jwtUserDetailsService() {
37          return new JwtUserDetailsService();
38      }
39
40
41      @Bean
42      public CommandLineRunner runner(AdminService adminService) {
43          return args -> {
44
45              /* Controllo se sia stato configurato un host per il
    frontend */
46              if(frontendHost.isEmpty()) {
47                  log.warning("Frontend host not configured.
    Confirmation link will have local validity.");
48                  GlobalVariables.FRONTEND_URL = "http://localhost:"+
    frontendPort;
49              } else {
50                  log.info("Frontend hostname configured. URL: https
    ://"+frontendHost+":"+frontendPort);
51                  GlobalVariables.FRONTEND_URL = "http://"+
    frontendHost+":"+frontendPort;
52              }
53
54              /* Controllo se sia stato configurato Spring Mail per
    le notifiche */
55              if(springMailPassword.isEmpty() || springMailUsername.
    isEmpty()) {
56                  log.warning("Mail server not configured.
    Confirmation emails will not be sent.");
57                  GlobalVariables.IS_MAIL_SERVER_CONFIGURED = false;
58              } else {
```

```
59                log.info("Mail server configured. Username: " +
    springMailUsername);
60                GlobalVariables.IS_MAIL_SERVER_CONFIGURED = true;
61
62                /* Controllo se devo usare le email fornite per l'
    invio del link della registrazione */
63                if(useRealMailAddress) {
64                    log.info("Email provided during signup will be
    used for registration notification.");
65                    GlobalVariables.USE_REAL_EMAIL_ADDRESS = true;
66                } else {
67                    log.warning("Notification email will be sent to
     the default address: "+defaultEmailAddress);
68                    GlobalVariables.USE_REAL_EMAIL_ADDRESS = false;
69                    GlobalVariables.DEFAULT_MAIL_ADDRESS =
    defaultEmailAddress;
70                }
71            }
72
73            /* Inserisco ruoli e utenti di default */
74            adminService.insertInitialRoles();
75            adminService.insertAdmin();
76            adminService.insertSomeTeachers();
77            adminService.insertSomeStudents();
78            adminService.insertSomeCourses();
79            adminService.insertSomeTeams();
80            adminService.insertSomeReports();
81            adminService.insertSomeStudyMaterials();
82            adminService.insertSomeVideoPills();
83            adminService.insertSomeChatsWithMessages();
84        };
85    }
86
87    public static void main(String[] args) {
88        SpringApplication.run(TeewServerApplication.class, args);
89    }
90
91 }
```

Il *main* è la prima funzione che viene eseguita, anche se in questo caso è stata inserita la funzione *runner* volta sia a configurare il servizio mail, se presente, sia a inizializzare alcuni elementi nel database.

Spring è un framework POJO-oriented (Plain Old Java Object) in quanto non obbliga il programmatore ad utilizzare una precisa gerarchia di ereditarietà per le classi, né tantomeno ad implementare delle interfacce specifiche. Spring si basa su tre pietre miliari:

- *Inversion of Control (IoC):* normalmente, nelle applicazioni basate sulla programmazione orientata agli oggetti, ogni oggetto istanziato è responsabile di

mantenere dei riferimenti agli oggetti da cui dipende. In particolare, il ciclo di vita degli oggetti è molto semplice: un oggetto viene istanziato tramite la keyword *new*, rendendolo pronto per l'utilizzo, e viene eliminato definitivamente dal *Garbage Collector* quando non viene più utilizzato. Questo approccio rende il codice molto più verboso del necessario, complica la fase di test dei singoli strati dell'applicazione e rende più difficile leggere e comprendere il codice in questione, nonché il suo riutilizzo. In Spring, invece, si vuole eliminare l'obbligo per gli oggetti di mantenere quei riferimenti sul piano del codice sorgente, ma ovviamente si desidera mantenere tutti i legami e le dipendenze in fase di esecuzione. Proprio per questo motivo è stato introdotto il concetto di Inversion of Control, ovvero l'eliminazione della dipendenza diretta in favore della creazione di un'interfaccia di supporto, la quale è responsabile di che cosa dev'essere fatto e quando. Ciò favorisce due pattern: il pattern ad *eventi*, ovvero che chi produce un evento è slegato da chi lo consuma, e il pattern a *plug-in*, ovvero che chi ospita e gestisce il ciclo di vita del plug-in è slegato da chi lo fornisce. Inoltre, questo meccanismo rende il ciclo di vita degli oggetti più complesso rispetto al caso tradizionale (in cui vi è l'istanziazione e l'eliminazione automatica tramite il Garbage Collector): in questo caso, infatti, gli oggetti potrebbero ricevere delle notifiche per gestirne il ciclo di vita, se le classi di riferimento sono denotate con annotazioni oppure implementano interfacce particolari.

- *Dependency Injection (DI):* è il responsabile dell'accoppiamento tra oggetti in fase d'esecuzione, è proprio grazie a questo concetto che si riesce a ottenere l'indipendenza tra oggetti in fase di compilazione e, quindi, a livello di codice sorgente. Sarà un componente esterno, il framework Spring, a prendersi cura di avere un grafo delle dipendenze, in base agli elementi dichiarativi del programmatore, e generarle in maniera opportuna; in fase d'esecuzione, le dipendenze vengono iniettate come proprietà degli oggetti. In particolare, i vari oggetti/componenti dell'applicazione possono essere legati da Spring in tre modi, che possono anche essere combinati a piacere: tramite una configurazione esplicita con un file XML, una configurazione esplicita con classi Java o una scoperta implicita, con autowiring automatico.

- *Aspect Oriented Programming (AOP):* rappresenta un paradigma di programmazione che rende possibile la descrizione dei comportamenti trasversali dell'applicazione: le funzionalità trasversali vengono chiamate *cross-cutting concern*, che possono essere modularizzati in *aspect*, ovvero delle classi apposite, il cui lavoro compiuto viene chiamato *advice*.

Come analizzato precedentemente, nelle applicazioni Spring vi è un container che si occupa di creare, configurare, collegare e gestire il ciclo di vita degli oggetti. In Spring, i *Bean* sono le classi identificate dall'annotazione *@Component*, oppure

una delle annotazioni che sono annotate con la stessa, come ad esempio *@Component, @Service, @Controller, @Repository, @Configuration* e così via.

L'*autowiring* è la funzionalità che permette di delegare a Spring il compito di risolvere automaticamente le dipendenze tra bean, cercandoli nel contesto dell'applicazione (*ApplicationContext*). Esistono tre tipologie, a seconda di dove si inserisce l'annotazione di autowiring e possono essere utilizzate promiscuamente:

- *Per costruttore:* l'annotazione *@Autowiring* precede la dichiarazione del costruttore di classe, in questo caso per ogni parametro viene ricercato se è disponibile un bean per il tipo indicato e, se presente, viene iniettato, se no viene lanciata un'eccezione. Ad esempio, nella classe *ChatRoomServiceImpl* è stato fatto l'autowiring di un oggetto di tipo *TeacherRepository* attraverso l'istruzione *@Autowired private TeacherRepository teacherRepository;*
- *Per tipo:* l'annotazione *@Autowiring* viene applicata ai singoli campi dell'oggetto, oppure ai metodi setter.
- *Per nome:* l'annotazione *@Resource(name="value")* indica che si desidera che venga iniettato un bean con il nome "value" noto al contenitore.

La classe principale di Spring Boot è annotata con *@SpringBootApplication*, che a sua volta equivale alle annotazioni *@Configuration*, per indicare che i suoi metodi possono definire dei bean, *@ComponentScan*, che attiva la ricerca di componenti all'interno del package corrente, e *@EnableAutoConfiguration*, che configura l'ambiente di esecuzione sulla base dei bean istanziati, delle classi e del file di configurazione.

In ogni progetto Spring Boot è presente un file di configurazione: *src/main/-resources/application.properties*, nel quale è possibile inserire indicazioni su come configurare il progetto, attraverso righe con coppie di *chiave = valore*, che possono entrambi dipendere dalle dipendenze aggiunte all'interno del progetto. Di seguito si riporta il file application.properties del progetto The Engineering Empathy Way:

```
spring.datasource.url=jdbc:mysql://localhost:3308/teew
spring.datasource.username=root
spring.datasource.password=pwd
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

# Minio Host
spring.minio.url=http://localhost:9000
# Minio Bucket name for your application
spring.minio.bucket=teewBucket
# Minio access key (login)
spring.minio.access-key=root
# Minio secret key (password)
spring.minio.secret-key=password
```

```
14
15
16 spring.jpa.generate-ddl=true
17 spring.jpa.show-sql=true
18
19 spring.mail.host=smtp.gmail.com
20 spring.mail.port=587
21 spring.mail.username=carlitosstorre@gmail.com
22 spring.mail.password=
23 spring.mail.properties.mail.smtp.auth=true
24 spring.mail.properties.mail.smtp.starttls.enable=true
25
26 jwt.secret=javainuse
27
28 spring.servlet.multipart.max-file-size = 500MB
29 spring.servlet.multipart.max-request-size = 500MB
30
31 spring.servlet.multipart.enabled=true
32 upload.file.extensions=jpg, jpeg, gif, png, video/mp4;charset=UTF-8
```

Tomcat viene automaticamente attivato quando il progetto dipende da *spring-boot-starter-web* (a meno che non sia presente in maniera esplicita un riferimento ad un altro container), mettendo l'applicazione in ascolto sulla porta 8080 per default. Ogni volta che viene ricevuta una richiesta in entrata, controlla il verbo HTTP e la URL associati e invoca un metodo del controllore responsabile di gestire la richiesta: solitamente si può ritornare un valore che permette di scegliere, in resources, la vista da far visualizzare all'utente, ma nel caso di The Engineering Empathy Way vengono ritornati oggetti JSON, tramite le *REST API* (sottosezione 5.4.5), che sarà poi il client a gestire per rappresentarli correttamente all'utente.

5.4.3 Architettura MVC

Un'applicazione può essere vista come una *black box* (scatola nera): uno strumento informatico che consente all'utente di fornire degli input, tipicamente tramite un'interfaccia grafica (GUI = graphical user interface), per poi ricevere in risposta degli output a seconda delle richieste effettuate; tutte le informazioni e i vincoli necessari devono essere contenuti in essa. Al crescere della complessità dell'applicazione, risulta estremamente conveniente suddividere l'applicazione in strati e utilizzare un approccio modulare, in maniera tale che ciascuno strato abbia un compito ben preciso e che possa comunicare con gli strati adiacenti mediante interfacce, rispettando il principio di separazione delle responsabilità.

Il paradigma MVC è un pattern architetturale che serve proprio a stratificare l'applicazione, suddividendo logicamente l'applicazione in:

- *modello:* questo strato contiene i dati dell'applicazione e determina le regole di evoluzione degli stessi; esso rappresenta il punto di partenza nello sviluppo del progetto

- *vista:* rappresentazione visuale, solitamente sotto forma di interfacce grafiche, del modello - solitamente di una sua parte - per l'utente; la vista non modifica il modello, bensì ne riceve solamente il contenuto da presentare

- *controllore:* intercetta gli input dell'utente, o più in generale le richieste in input, convertendole in comandi per il modello e/o la vista; in pratica, estrae i parametri dalle richieste, sceglie l'interfaccia di servizio da richiamare per effettuare eventuali manipolazioni del modello e sceglie la vista atta a presentare all'utente il modello risultante dopo le operazioni

In particolare, per ogni precisa richiesta ricevuta dal client, viene definita una particolare URL a cui viene mappato un controllore responsabile di gestire la richiesta: la analizzerà, utilizzerà un'interfaccia di servizio per richiedere una certa manipolazione del modello in maniera consona alla richiesta e, infine, sceglierà la vista corretta a rappresentare la porzione di modello necessaria per soddisfare la richiesta. In realtà, in The Engineering Empathy Way, il controllore non restituisce mai una vista, bensì dei dati sotto forma di JSON: è di fatto il client a scegliere opportunamente come rappresentare le informazioni in maniera grafica.

Il modello adottato in The Engineering Empathy Way viene analizzato nella sottosezione 5.4.7; i controllori e i dati (sotto forma di JSON) ritornati nelle risposte HTTP da essi al client sono analizzati nella sottosezione 5.4.5.

5.4.4 Spring Security e JWT

È buona norma implementare i vincoli di sicurezza di un'applicazione in maniera dichiarativa, invece di inserirli a livello applicativo, affinché sia l'ambiente di esecuzione a implementare opportunamente i meccanismi necessari. Per poter entrare maggiormente in dettaglio, bisogna chiarire il significato di alcuni termini:

- *Principal:* utente che desidera interagire con l'applicazione web

- *Autenticazione:* processo di identificazione degli utenti, validando il fatto che il principal sia veramente chi "dice" di essere

- *Credenziali:* informazioni fornite dal principal per dimostrare la propria identità

- *SecurityContext:* interfaccia Spring contenente i dettagli di autenticazione del principal

- *Autorizzazione:* dopo che un utente è stato autenticato, bisogna verificare a quali risorse è autorizzato ad accedere
- *Granted Authority:* interfaccia Spring contenente i dettagli di autorizzazione del principal
- *Risorsa sicurizzata:* risorsa a cui il principal può accedere se è stato prima autenticato e se detiene i requisiti necessari di autorizzazione

Sia il processo di autenticazione sia quello di autorizzazione necessitano il mantenimento di una base dati aggiornata con le informazioni relative ad ogni utente del sistema. È necessario fornire dei meccanismi per permettere agli utenti di registrarsi alla piattaforma, di effettuare il login, di aggiornare le proprie informazioni (anche relative all'autenticazione, come ad esempio la password) e, infine, meccanismi che controllino le autorizzazioni sulle risorse da parte di ciascun utente.
Per quanto riguarda la registrazione degli utenti, si potrebbero utilizzare informazioni quali: userID, identificativo dell'utente univoco all'interno della piattaforma, email, password, domande di sicurezza con relative risposte e captcha, ovvero un meccanismo che eviti l'attacco massivo da parte di robot. In The Engineering Empathy Way, si è scelto di far registrare gli utenti attraverso email e password: in particolare, le email devono appartenere al dominio del Politecnico (*@polito.it*, per i docenti, e *@studenti.polito.it* per gli studenti) e devono avere nella prima parte una matricola valida, ovvero nella forma *dxxxxx*, per i docenti, oppure *sxxxxx*, per gli studenti, dove x rappresenta una qualsiasi cifra compresa tra 0 e 9. In particolare, dopo che l'utente ha inserito i dati, viene inviata una mail di conferma contenente due link: uno per confermare la registrazione e uno per rifiutarla. Per permettere la registrazione, è stato creato un apposito controllore che espone delle API, che sono le uniche a non essere soggette a restrizioni e vincoli di sicurezza, anche se ovviamente al proprio interno controllano la validità dei dati ricevuti.

Per poter usufruire dei meccanismi di sicurezza, come prima cosa sono state importate le due dipendenze necessarie nel file *pom.xml*, come mostrato nella sottosezione 5.4.1: la dipendenza a Spring Security [44] e quella relativa al JWT, riportate di seguito per semplicità:

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-security</artifactId>
</dependency>
<dependency>
   <groupId>io.jsonwebtoken</groupId>
   <artifactId>jjwt</artifactId>
   <version>0.9.1</version>
</dependency>
```

Per implementare i meccanismi di autenticazione ed autorizzazione, si è scelto di utilizzare il sistema basato su *JWT (JSON Web Token)* [45] in The Engineering Empathy Way.
Esso prevede l'invio, da parte del client, di richieste con un'intestazione, codificata su *Base64* formata nel seguente modo: *<header>.<payload>.<signature>*, dove header indica i metadati sul tipo di token, payload contiene informazioni relative all'utente e le relative autorizzazioni e, infine, signature rappresenta una firma delle parti precedenti.

L'immagine [45] seguente rappresenta in maniera dettagliata l'architettura generale utilizzata per implementare il meccanismo di autenticazione:

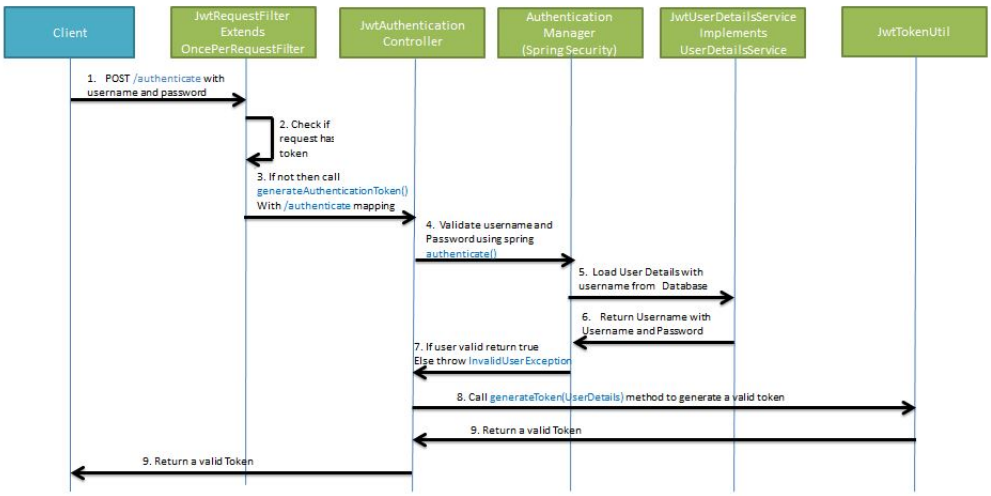

Figura 5.3. Architettura gestione sicurezza

Per identificare gli utenti, si utilizza un oggetto di tipo *AuthenticationManager*, che indica se le credenziali fornite dall'utente sono valide o no.

Si possono dichiarare le restrizioni di accesso in due modi: a livello di URL e a livello di singoli metodi di servizio:

- Per quanto riguarda le URL, sono state protette tramite dichiarazioni di configurazione estendendo l'interfaccia *WebSecurityConfigurerAdapter*. In particolare, in uno dei tre metodi da implementare nella classe *WebSecurityConfig* per poter estendere correttamente quell'interfaccia, sono state inserite le seguenti righe di codice:

```
1 httpSecurity
2                 .csrf().disable()
```

```
3                 .authorizeRequests()
4                 .antMatchers("/authenticate")
5                 .permitAll()
6                 .antMatchers("/notification/**")
7                 .permitAll()
8                 .antMatchers("/API/students/registration")
9                 .permitAll()
10                .antMatchers("/API/teachers/registration")
11                .permitAll()
12                .antMatchers("/registration")
13                .permitAll()
14                .antMatchers("/ws/**")
15                .permitAll();
16
17        httpSecurity.authorizeRequests()
18                .antMatchers("/**")
19                .authenticated()
20                .and()
21                .exceptionHandling()
22                .authenticationEntryPoint(
    jwtAuthenticationEntryPoint)
23                .and()
24                .sessionManagement()
25                .sessionCreationPolicy(SessionCreationPolicy.
    STATELESS);
26
27        httpSecurity.addFilterBefore(jwtRequestFilter,
    UsernamePasswordAuthenticationFilter.class);
```

Tutte le richieste per le URL che non corrispondono a quelle indicate alle righe 4, 6, 8, 10, 12 o 14 (che corrispondono alle richieste di login, di verifica della registrazione, di registrazione di studente, di registrazione di docente, di registrazione o di connessione ad una chat) devono essere autenticate. Negli spunti di miglioramento, analizzati nella sezione 5.7, sarà indicata anche la possibilità di migliorare la gestione della sicurezza a livello di connessione ad una chat.
Inoltre, tramite la riga 27, è stato aggiunto il filtro *jwtRequestFilter*, che ha il compito di controllare che le richieste effettuate dispongano di un JWT corretto.

- Per quanto riguarda i metodi di servizio, invece, si possono utilizzare alcune annotazioni: *@PreAuthorize*, *@PostAuthorize*, *@PreFilter*, *@PostFilter*. In The Engineering Empathy Way, tutti i controlli relativi all'autorizzazione sono stati fatti tramite la prima annotazione, di cui si riporta qualche esempio:

```
@Override
@PreAuthorize("hasRole('ROLE_STUDENT')")
public TeamDTO proposeTeamForCourse(String courseName, String teamName,
                                    Integer timeout, List<String> memberIds) {...}

    @Override
    @PreAuthorize("hasRole('ROLE_TEACHER')")
    public CourseDTO deleteCourse(String name) {...}

@Override
@PreAuthorize("hasRole('ROLE_STUDENT') || hasRole('ROLE_TEACHER')")
public CourseDTO getCourse(String courseName) {...}
```

Figura 5.4. Esempi di controllo autorizzazione

Nel dettaglio dell'esempio, nel metodo *proposeTeamForCourse* (richiamato per effettuare una proposta di team) viene effettuato un controllo che l'utente che effettua la richiesta sia uno studente, nel metodo *deleteCourse* (utilizzato per cancellare dal sistema un intero corso e tutto il materiale relativo) che sia un docente e, nel metodo *getCourse* (usato per recuperare le informazioni relative a un particolare corso), che abbia ruolo di studente o di docente (o entrambi).

Infine, è indispensabile implementare una delle funzionalità avanzate: *CORS*, ossia Cross-Origin Resource Sharing, che rappresenta un meccanismo basato su HTTP grazie al quale il browser può effettuare richieste a host differenti da quelli su cui è stato scaricato il codice applicativo. Nel caso di The Engineering Empathy Way, il client presente su localhost:4200 può effettuare richieste a localhost:8080 dove è presente il server in ascolto.

5.4.5 REST API e controllori

REST API

REST è un acronimo che sta per REpresentational State Transfer e rappresenta un concetto inventato da Roy Fielding nella sua tesi di dottorato nel 2000. Esso si basa su una serie di principi:

- l'architettura client-server è priva di stati
- un server può ospitare uno o più servizi e gestire un insieme di informazioni: URI (Uniform Resource Identifier), formato di rappresentazione (JSON, XML, ecc.) e supporto ad operazioni CRUD.

- per ogni URL, ci sono una serie di informazioni associate, come ad esempio riferimenti a singole entità, collezioni di entità e risultati di diversi calcoli funzionali

Siccome ogni entità ha un nome univoco, è possibile associare URL ad esse e in particolare utilizzare gli identificativi delle entità stesse. Per supportare le operazioni CRUD, si utilizza il verbo del protocollo HTTP:

- *GET:* chiede una rappresentazione di una collezione o di un entità precisa. Ad esempio:

  GET http://localhost:8080/API/courses/cnv

- *POST:* aggiunta di una nuova entità. Ad esempio

  POST http://localhost:8080/API/courses/cnv/teachers

  per aggiungere un insegnante (passato nel corpo della richiesta) al corso relativo alla CNV

- *PUT:* aggiornamento di una risorsa. Ad esempio:

  PUT http://localhost:8080/API/courses/cnv

  per aggiornare le informazioni (passate nel corpo della richiesta) relative al corso sulla CNV

- *DELETE:* chiede di eliminare una certa entità. Ad esempio:

  DELETE http://localhost:8080/API/courses/cnv

  per eliminare il corso relativo alla CNV

In particolare, per negoziare il formato (JSON, XML, test, ecc.) si utilizza il meccanismo nativo di HTTP, basato sull'uso dei campi *Content-type:* e *Accept:* nell'intestazione, in quanto il server può fornire diverse rappresentazioni e il client può richiederne una di suo gradimento.

È stato ideato un modello, chiamato *Il modello di maturità di Richardson*, per indicare il livello di adesione ai principi dell'architettura REST:

- *Livello 0:* si utilizza HTTP come protocollo di trasporto, verbo POST (il reale comando è indicato come parametro) e una sola URL

- *Livello 1:* ad ogni entità corrisponde una URL, si usa il verbo POST (il reale comando è indicato come parametro)
- *Livello 2:* come il livello 1, ma i comandi sono mappati sui verbi delle richieste HTTP (GET, POST, PUT, DELETE) e i risultati delle varie operazioni sono indicati tramite i codici di stato (ad esempio 200 - Ok, 404 - Not found, ecc.)
- *Livello 3:* come il livello 2, ma si aggiungono collegamenti ipermediali per scoprire dinamicamente i servizi offerti dal server, spianando la strada alla realizzazione dei microservizi

In The Engineering Empathy Way si è scelto di implementare un grado di adesione pari al livello 2.

Controllori

Per sfruttare le API di tipo di REST, sono stati creati vari controllori, ovvero delle classi annotate tramite *@RestController*. Il client invia una serie di richieste al server, il quale le analizza e può ricevere dei dati associati. Ogni richiesta viene mappata a un controllore, che si occupa di gestire richieste appartenenti a un particolare insieme di URL; il controllore ottiene tutte le informazioni, necessarie alla corretta gestione della richiesta, in tre modi:

- *formato della stringa dell'URL:* ogni URL fa esplicito riferimento a una particolare risorsa all'interno del server e, per ogni URL e per ogni verbo HTTP, c'è un metodo incaricato di gestirne la richiesta. In particolare, è possibile inserire delle variabili all'interno dell'URL, che il controllore sa di attendere tramite l'utilizzo dell'annotazione *@PathVariable("variableName")*, che precede la dichiarazione della variabile stessa all'interno dei parametri del metodo di riferimento. Si riporta un esempio tratto dal controllore, il quale estrae il token della registrazione come variabile nella URL ed è responsabile della verifica dell'avvenuta registrazione in maniera corretta:

```
@GetMapping("/notification/confirm/{tokenId}")
@ResponseStatus(value=HttpStatus.OK)
public void confirm(@PathVariable String tokenId) {
    try {
        notificationService.confirm(tokenId);
    } catch (NotificationServiceException nse) {
        throw new ResponseStatusException(HttpStatus.BAD_REQUEST, nse.getMessage());
    }
}
```

Figura 5.5. Primo esempio di controllore per URL con variabile

- *parametri:* ogni richiesta può avere uno o più parametri associati, che il controllore può intercettare grazie all'utilizzo dell'annotazione *@ReuqestParam("paramName")* la quale, come prima, precede la dichiarazione della variabile che ne ospiterà il valore. In particolare, un parametro può essere opzionale e questo può essere indicato *required=false*. L'immagine seguente mostra un metodo del *CourseController* che si attende come parametro il path da cui recuperare il file dell'esercizio corrispondente, inserito nella URL:

```
@GetMapping("/{courseName}/assignments")
ResponseEntity<List<AssignmentDTO>> getAssignmentsForCourse(@PathVariable String courseName,
                                                            @RequestParam String path) {
    try {
        List<AssignmentDTO> assignmentsDTOS = courseService.getAssignmentsForCourse(courseName, path);
        return ResponseEntity.status(HttpStatus.OK).body(assignmentsDTOS);
    } catch (ServiceException tse) {
        throw new ResponseStatusException(HttpStatus.NOT_FOUND, tse.getMessage());
    }
}
```

Figura 5.6. Secondo esempio di controllore per URL con parametro

- *corpo:* le richieste possono opzionalmente avere un corpo (ad esempio, quelle POST e PUT) e il controllore può intercettarne il valore attraverso l'annotazione *@RequestBody()*. La figura sottostante rappresenta un metodo del *CourseController* che riceve nel corpo il file del materiale di studio corrispondente a una richiesta di upload da parte di uno studente:

```
@PostMapping("{courseName}/uploadRequests")
ResponseEntity<UploadRequestDTO> requestUploadStudyMaterial(@PathVariable String courseName,
                                                            @RequestParam String name,
                                                            @RequestParam String path,
                                                            @RequestBody MultipartFile file) {

    try {
        courseService.requestUploadStudyMaterial(courseName, name, path, file);
        return new ResponseEntity(HttpStatus.OK);
    } catch (ServiceException tse) {
        throw new ResponseStatusException(HttpStatus.BAD_REQUEST);
    }
}
```

Figura 5.7. Secondo esempio di controllore per URL con corpo

The Engineering Empathy presenta 14 controllori, mostrati nella figura sottostante:

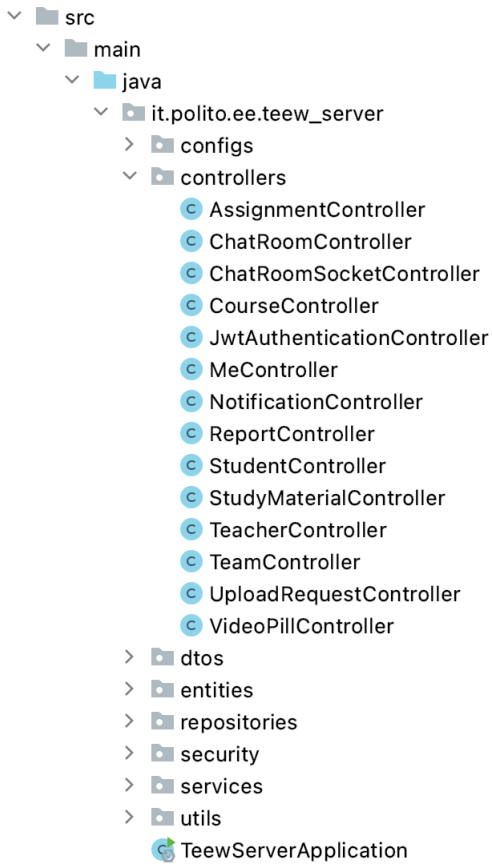

Figura 5.8. I controllori

Più precisamente, vi è un controllore per la verifica della registrazione (*NotificationController*; il *JwtAuthenticationController* è stato utilizzato per gestire i processi di registrazione e autenticazione; il *MeController* per gestire i profili degli utenti; infine, gli altri controllori sono facilmente riconducibili alle operazioni necessarie da effettuare sulle varie collezioni di entità presenti nel modello di dati resi persistenti nel database.

5.4.6 Servizi e DTO

Un'applicazione offre un insieme di funzionalità all'utente e questo avviene tipicamente sotto forma di API, i cui metodi corrispondenti costituiscono la logica del servizio che si vuole offrire. Lo strato di servizio è proprio ciò che permette di erogare le funzionalità ed è composto da una o più classi denotate con *@Service*. Si riportano di seguito i servizi presenti in The Engineering Empathy Way, che sono

organizzati in svariate directory che includono l'interfaccia di servizio e la classe corrispondente che implementa l'interfaccia:

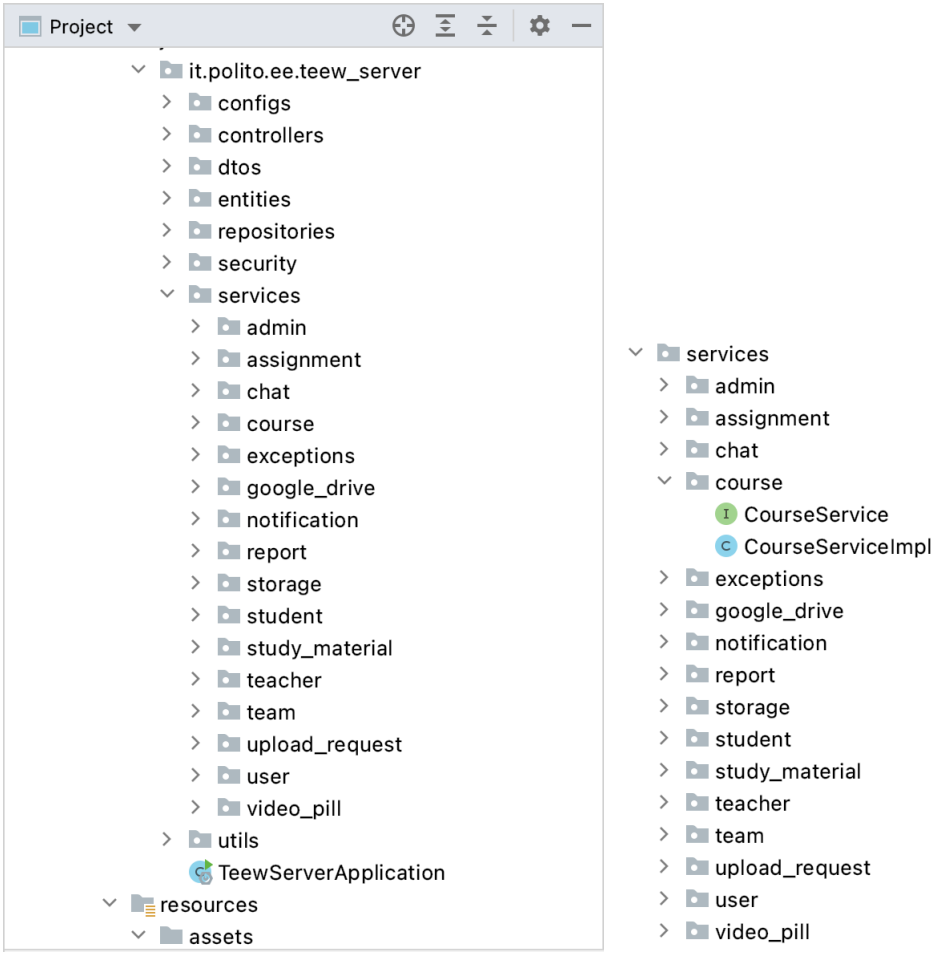

Figura 5.9. Elenco dei servizi ed esempio per il servizio sui corsi

I servizi permettono di accedere e modificare il modello dei dati presenti nel database, analizzato nella sottosezione 5.4.7. In particolare, il servizio opera con due tipi di oggetti: i DTO e le entità.
Solitamente, i servizi ritornano degli oggetti che devono poi essere inoltrati alla vista; nel caso di The Engineering Empathy Way vengono invece serializzati in JSON e inviati tramite una risposta HTTP al client, visto che sono state utilizzate le API di tipo REST. Questi oggetti prendono il nome di *DTO*, ovvero *Data Transfer Object*, e rappresentano i dati che l'interfaccia di servizio riceve dal e/o restituisce al controllore. È bene sottolineare che questi oggetti si differenziano da quelli che vengono salvati direttamente all'interno del database, che vengono chiamati *Entity*, perché rappresentano le entità così come vengono memorizzate nella

base dati. Proprio per questo è necessario che i servizi dispongano di funzioni di utilità in grado di mappare gli oggetti entità in oggetti DTO, in maniera tale da poter restituire questi ultimi al controllore di riferimento.

Il motivo più importante per cui si separano questi due tipi di oggetto è che si vuole mantenere separata l'informazione che viene trasferita da client a server, e viceversa, con quella che viene salvata nel database: infatti, non sempre è necessario restituire tutto ciò che è presente nel database, oppure magari si possono ritornare informazioni mappate in maniera differente da come sono salvate perché il client ne ha bisogno in maniera differente. I dati possono anche non essere per forza memorizzati interamente in una sola base dati o magari alcune informazioni ricevute dal client non necessitano di essere salvate e così via. La separazione tra entità e DTO, anche se non è sempre necessaria perché i due oggetti spesso contengono le medesime proprietà, avviene per alcune ragioni molto importanti [46]:

- le entità rappresentano gli oggetti mappati nel database, mentre i DTO gli oggetti che vengono inviati (e ricevuti dalle) alle viste, anche se nel nostro caso vengono inviati sotto forma di JSON al client invece che direttamente alle viste

- è importante separare responsabilità differenti, in maniera tale che non condividano le stesse risorse

- facilità di manutenzione del codice, soprattutto in vista di cambiamenti futuri nelle relazioni tra oggetti e nelle loro proprietà

- se cambiano i requisiti, si può modificare solamente la parte necessaria (quella relativa alla persistenza o a quella relativa alle informazioni da scambiare con il client)

- risparmio di spazio nel database: se magari il DTO necessita della data di nascita e dell'età di una persona, nel database non è necessario salvare entrambe le informazioni, infatti avendo anche solo a disposizione la data di nascita salvata sarà molto facile recuperare l'età della persone con una semplice sottrazione con la data odierna

- se si cambia ad esempio una proprietà nell'entità e le viste non vengono aggiornate continuando a cercare di mostrare il contenuto precedente, ritornare la nuova entità creerebbe errori in quando ci sarebbe una discordanza tra le proprietà nuove e quelle vecchie che si tenta di mostrare nelle viste.

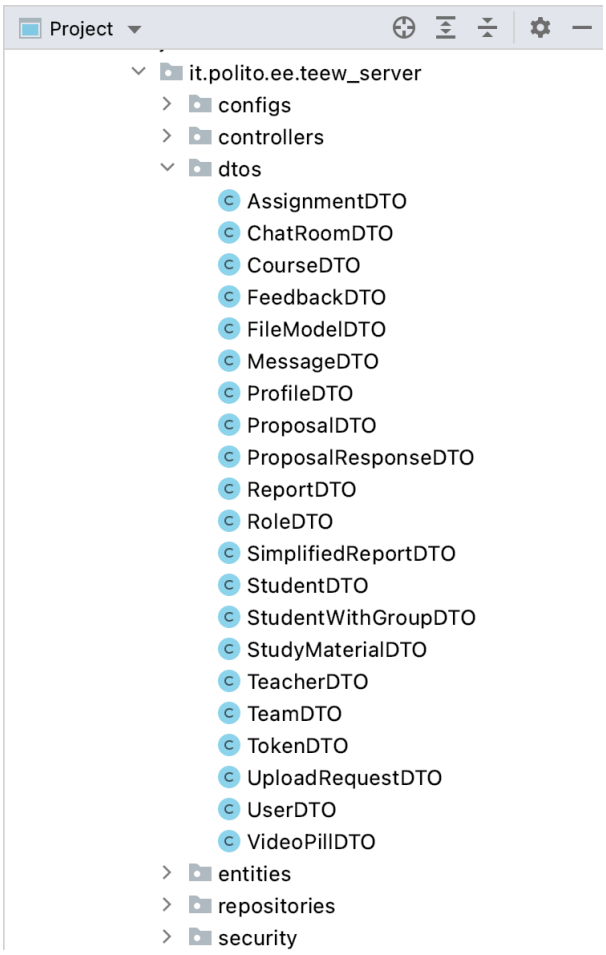

Figura 5.10. Gli oggetti DTO

Per poter gestire agilmente gli oggetti DTO (e anche le entità, descritte nel seguito), come prima cosa è stata importata la dipendenza al progetto Lombok [47] nel file *pom.xml*, come mostrato nella sottosezione 5.4.1, riportata di seguito per semplicità:

```
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
</dependency>
```

Il progetto Lombok è una libreria Java che mette a disposizione del programmatore una serie di annotazioni molto utili quando si devono gestire classi che contengono dati, in maniera tale che il programmatore non debba scrivere ripetutamente tante righe di codice macchinose, concentrandosi quindi su altre parti più importanti. Ad esempio, quelle più utilizzate in The Engineering Empathy Way sono:

- *@Data:* serve a far creare automaticamente i metodi *toString*, *getAttribute*, *setAttribute*, *EqualsAndHashCode* e *@RequiredArgsConstructor* prendendo in considerazione tutti gli attributi dichiarati nella classe

- *@AllArgsConstructor:* crea un costruttore, per quella classe, inizializzando tutti gli attributi

- *@Log:* permette di stampare nella console delle informazioni di log, per favorire il debug dell'applicazione

- *@NoArgsConstructor:* crea un costruttore vuoto

- *@Builder:* viene applicato a una classe e serve a produrre automaticamente il codice necessario a renderla istanziabile con codice più comprensibile, si ad esempio il comando utilizzato per creare un oggetto di tipo MinioClient in maniera agevole:

```
1 MinioClient minioClient = MinioClient.builder()
2             .endpoint("http://localhost:9000")
3             .credentials("root", "password")
4             .build();
```

5.4.7 Modello dei dati e database

Prima di partire con la realizzazione via codice delle entità da persistere nel database, è stato realizzato un diagramma che rappresenta le relazioni tra le varie entità, tramite la versione gratuita del programma StarUML [48]:

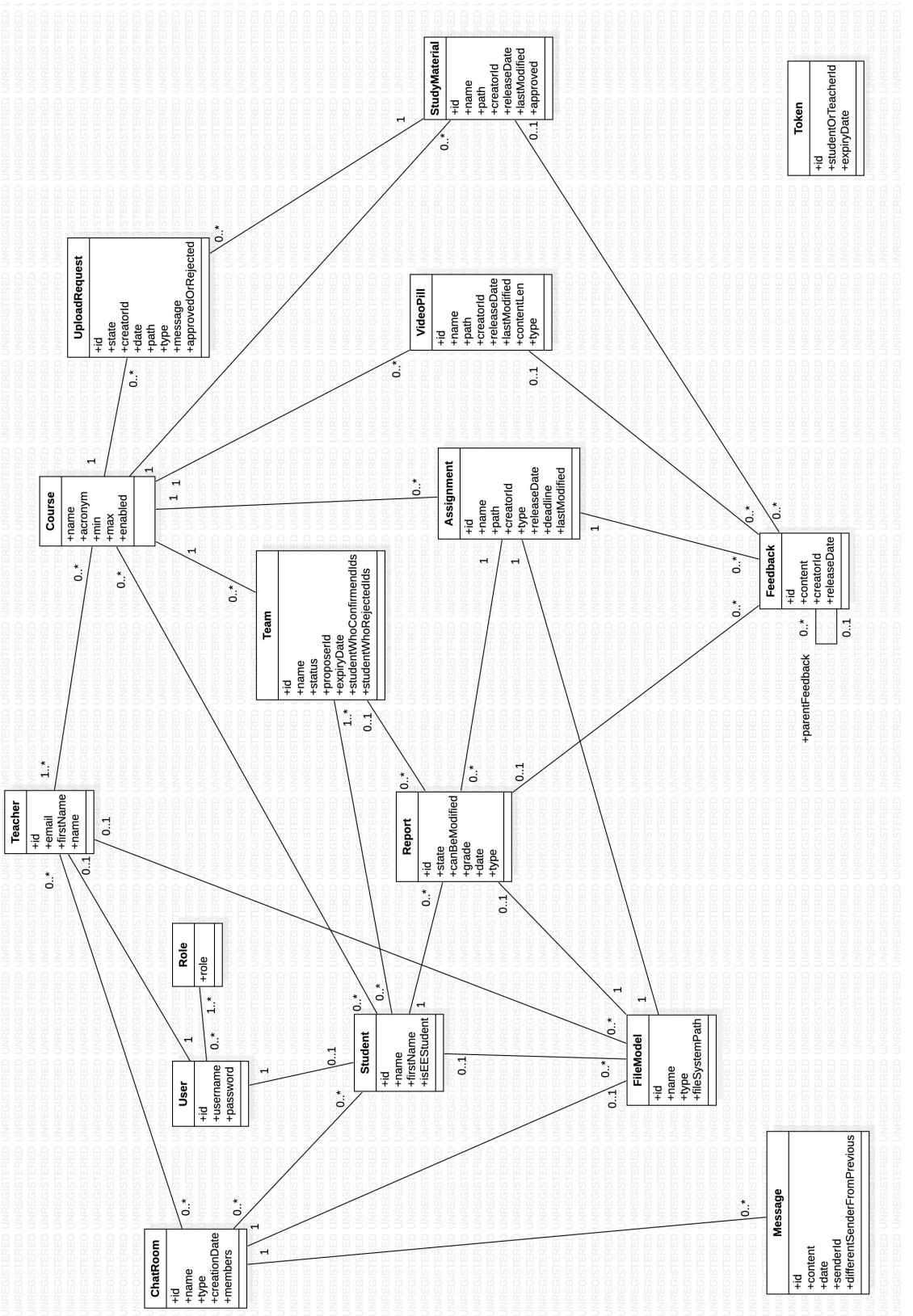

Figura 5.11. Modello dei dati

Per poter persistere correttamente i dati in maniera semplice, come prima cosa è stata importata la dipendenza a Spring Data nel file *pom.xml*, come mostrato nella sottosezione 5.4.1, riportata di seguito per semplicità:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

Ad ognuna delle entità raffigurate corrisponde un'entità a livello di codice. Come database, si è scelto di utilizzare MariaDB [49]: si tratta di un famoso database relazionale molto veloce e robusto, volto alla scalabilità, che è stato originariamente creato dagli sviluppatori di MySQL con l'obiettivo di renderlo sempre open source con la licenza GNU General Public License.
Nel dettaglio, ne è stata installata un'immagine tramite l'utilizzo di Docker, con il seguente comando:

docker run -v /Users/Carlitos/Desktop/db_teew:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=pwd -p 3306:3306 -d mariadb:latest

Ovviamente, /Users/Carlitos/Desktop/ dev'essere sostituito con il path, della propria macchina, in cui lo si vuole installare. Dopodiché, tramite l'IDE IntelliJ Idea utilizzato per lo sviluppo, è stato creato il database *teew* necessario ad ospitare tutte le tabelle con i dati da persistere. Quindi il database si trova alla porta 3306 del localhost (ovvero la macchina corrente in cui viene avviata l'applicazione); ha come username "root" e come password "pwd"; utilizza, come driver la classe "com.mysql.cj.jdbc.Driver"; i dettagli della connessione al database sono specificati nel file application.properties con l'elenco delle proprietà del progetto:

```
spring.datasource.url=jdbc:mysql://localhost:3306/teew
spring.datasource.username=root
spring.datasource.password=pwd
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
```

Per accedere ai dati persistenti, il modulo Spring Data permette di semplificare notevolmente l'accesso al db tramite due metodi possibili:*JDBC* oppure *JPA*, che permettono anche di accedere a basi dati non relazionali (come ad esempio MongoDB) in altri modi che dipendono dal DBMS (DataBase Management System). In The Engineering Empathy Way si è deciso di realizzare appunto un database relazionale, *MariaDB*, al quale si accede tramite JPA.

JPA (Java Persistance API) rappresenta una specifica di JavaEE che, tramite implementazioni come *Hibernate*, permette di implementare il concetto di *ORM*, Object Relational Mapping, ovvero uno strato software che dà la possibilità di mappare, tramite delle opportune annotazioni, le classi di Java in tabelle del database e gli oggetti istanziati (relativi alle classi) in record delle tabelle corrispondenti.

Tutto questo è gestito da classi *DAO (Data Access Object)*, chiamate Repository, che Spring implementa automaticamente offrendo metodi di default per accedere ai record della tabella desiderati; se si vogliono metodi aggiuntivi, è necessario aggiungere metodi alle interfacce Repository e, se necessario, implementare le query desiderate. Le interfacce Repository<T, Id> devono estendere *org.springsource.data.repository*: in questo caso, T rappresenta la classe denotata da @Entity che si vuole persistere Id il tipo della chiave primaria. Le classi Repository presenti in The Engineering Empathy Way sono:

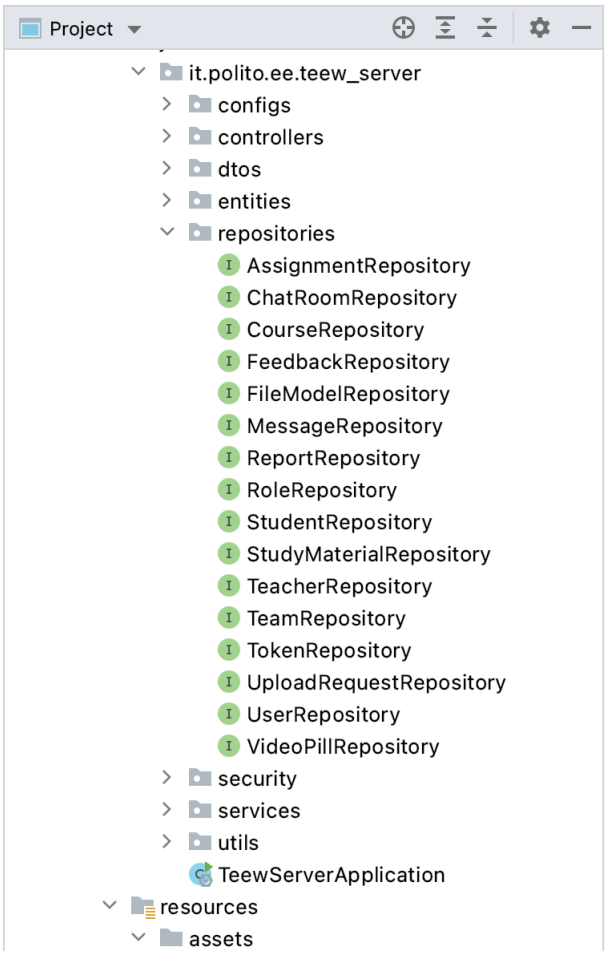

Figura 5.12. Le classi Repository

Lo strato di persistenza gestisce l'accesso ai dati tramite operazioni come *findAll(), save(), saveAll(), delete(), deleteAll()* e così via. Ad esempio, attraverso il comando

```
teacherRepository.findAll();
```

è possibile accedere a tutte le istanza di tipo Teacher presenti nel database.

In particolare, ogni istanza ha un identificatore univoco ed è dotata di metodi getter e setter per recuperarne e impostarne i valori di ogni attributo; ogni riferimento diretto o indiretto (ad esempio liste) ad altre entità viene considerato una relazione: ci possono essere relazioni 1-1, N-1 e M-N. La differenza principale tra le relazioni in Java e nel database è data dal fatto che nel database le relazioni sono intrinsecamente bidirezionali, ovvero è sempre possibile risalire da un'entità all'altra una volta che si conosce lo schema; in Java, invece, le relazioni sono mantenute tramite dei riferimenti: l'oggetto A avrà un riferimento all'oggetto B ma non è detto che sia presente anche un riferimento all'oggetto A nell'oggetto B, cosa che diventa obbligatoria se si vuole percorrere la relazione in entrambe le direzioni. In quest'ultimo caso, è altrettanto importante fare in modo che le modifiche siano propagate in entrambe le direzioni.
Le annotazioni usufruibili dal programmatore sono:

- *@Entity:* applicata a una classe, definisce un'entità che viene mappata come tabella. Ogni oggetto che verrà istanziato per quella classe, rappresenterà un record della tabella. Il nome della tabella può essere specificato tramite l'annotazione *@Table(name="tableName").* Le entità presenti in The Engineering Empathy Way sono:

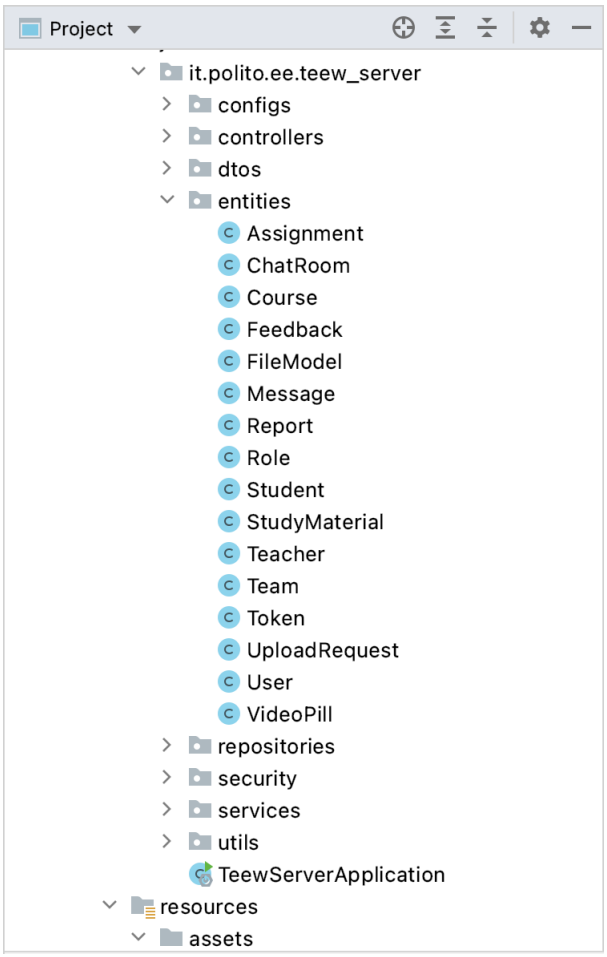

Figura 5.13. Le entità

- *@Id:* applicato a un attributo, rappresenta la chiave primaria per quell'entità. Se si vuole lasciare al framework l'onere di creare identificativi univoci e progressivi, si può utilizzare l'annotazione @GeneratedValue, il cui valore di default è *GenerationType.AUTO*, che utilizza la strategia più adatta a seconda del dialetto SQL che viene adottato dal DBMS
- *@OneToOne:* crea una relazione uno a uno: ad ogni istanza della classe in cui è utilizzata ne corrisponde soltanto una dell'altra indicata
- *@OneToMany:* crea una relazione uno a molti: ad ogni istanza della classe in cui è utilizzatae, ne corrispondono molte dell'altra indicata
- *@ManyToOne:* crea una relazione molti a uno: ad ogni istanza della classe in cui è utilizzata ne corrisponde soltanto una dell'altra indicata e viene salvato soltanto l'identificatore dell'entità di destinazione

- *@ManyToMany:* crea una relazione molti a molti: ad ogni istanza della classe in cui è utilizzata ne corrispondono molte dell'altra indicata e viene creata una tabella aggiuntiva per mappare la relazione

- *@Cascade:* permette di aggiornare e propagare le modifiche alle varie tabelle secondo certi vincoli. I valori possibili sono *CascadeType.ALL, CascadeType.DETACH, CascadeType.MERGE, CascadeType.PERSIST, CascadeType.REFRESH, CascadeType.REMOVE.*

Si riporta di seguito un esempio di utilizzo di queste annotazioni:

```
@ManyToMany
@JoinTable(name = "student_course",
           joinColumns = @JoinColumn(name="student_id"),
           inverseJoinColumns = @JoinColumn(name="course_name")) private List<Course> courses = new ArrayList<>();
@ManyToMany(mappedBy = "members") private List<Team> teams = new ArrayList<>();
@OneToOne private User user;
@OneToOne private FileModel profileImageFileModel;
@OneToMany @LazyCollection(LazyCollectionOption.FALSE) private List<Report> reports = new ArrayList<>();
@ManyToMany
@JoinTable(name = "student_chat_room",
           joinColumns = @JoinColumn(name = "student_id"),
           inverseJoinColumns = @JoinColumn(name="chat_room_id"))
//private Set<ChatRoom> chatRooms = new TreeSet<>(Comparator.comparing(ChatRoom::getCreationDate));
private List<ChatRoom> chatRooms = new ArrayList<>();
```

Figura 5.14. Annotazioni all'interno di entità

Inoltre, è possibile impostare dei vincoli su esistenza e unicità tramite le annotazioni *@Nullable* e *@Unique* oppure settando *nullable="true"* e *unique="true"* all'interno di altre annotazioni.

5.4.8 MinIO

MinIO [50] è un cloud native storage service, ovvero un servizio minimalista volto alla scalabilità sul web, basato su cloud e che fornisce delle API per rendere persistenti dei file; è stato creato completamente da zero negli ultimi quattro anni, si basa sugli standard di Amazon S3 e Kubernetes, ma ovviamente è interoperabile anche con gli altri sistemi. Si tratta quindi di un'ottima soluzione come sistema di archiviazione, molto simile ad Amazon S3 ma che può essere ospitato localmente. È possibile archiviare file di dimensione varia, dai pochi KB fino a un massimo di 5 TB; i file vengono raggruppati nei cosiddetti *bucket*, che sono dei separatori logici (come se fossero delle cartelle del file system tramite cui raggruppare i file a piacimento) per i file resi persistenti. Per accedere ai vari bucket, è necessario fornire l'indirizzo HTTP del servizio MinIO e, in aggiunta, la chiave e il segreto di accesso.

MinIO è molto semplice da configurare ed è composto da tre principali componenti:

- *MinIO server:* rappresenta il vero e proprio servizio di archiviazione di oggetti
- *MinIO client:* è il componente che fornisce una serie di comandi, e alternativi ai classici UNIX, supportando sempre sia il file system sia l'architettura cloud di Amazon S3
- *MinIO client SDK:* fornisce le API necessarie per interagire con il sistema di archiviazione, quindi per effettuare operazioni sui bucket e sugli oggetti salvati, tramite linguaggi comunemente utilizzati come Java, Haskell, Go, Python, JavaScript

MinIO si è rivelato molto utile per poter memorizzare in maniera semplice ed efficace tutti i file caricati dai vari utenti. L'installazione locale è avvenuta tramite il comando: *brew install minio/stable/minio*; a questo punto, per far partire il servizio sarebbe stato sufficiente eseguire *minio server path/to/store/data*, ma è stato opportuno inizializzarlo tramite l'utilizzo di Docker. Il comando base per tirar su il servizio sulla porta 9000 è il seguente:

docker run -p 9000:9000 -e "MINIO ROOT USER=root" -e "MINIO ROOT PASSWORD=pwd" minio/minio server /data

Ma questo non prevede il salvataggio dei dati in maniera permanente sul file system, una volta arrestato il container così creato. Per archiviare in maniera permanente è stato sufficiente aggiungere il parametro *-v /real/path/to/store/data:/data* che permette di mappare il path */data* al path reale */real/path/to/store/data* del file system della macchina. Ecco il comando completo:

docker run -p 9000:9000 -v /real/path/to/store/data:/data -e "MINIO ROOT USER=root" -e "MINIO ROOT PASSWORD=pwd" minio/minio server /data

Per poter usufruire del servizio di archiviazione MinIO, come prima cosa è stata importata la dipendenza necessaria nel file *pom.xml*, come mostrato nella sottosezione 5.4.1, riportata di seguito per semplicità:

```
<dependency>
   <groupId>io.minio</groupId>
   <artifactId>minio</artifactId>
   <version>8.0.3</version>
</dependency>
```

È necessario aggiornare il file *application.properties* aggiungendo le informazioni necessarie al corretto funzionamento di MinIO:

```
spring.minio.url=http://localhost:9000
spring.minio.bucket=teewBucket
spring.minio.access-key=root
spring.minio.secret-key=password
```

Le quali indicano che MinIO è disponibile alla porta 9000 della macchina localhost (quella corrente in cui è stata avviata l'applicazione); il nome del bucket principale è "teewBucket"; la chiave d'accesso è "root" e la chiave segreta è "password".

Per verificare che il servizio sia attivo in maniera corretta, è sufficiente aprire il browser e connettersi a http://localhost:9000:

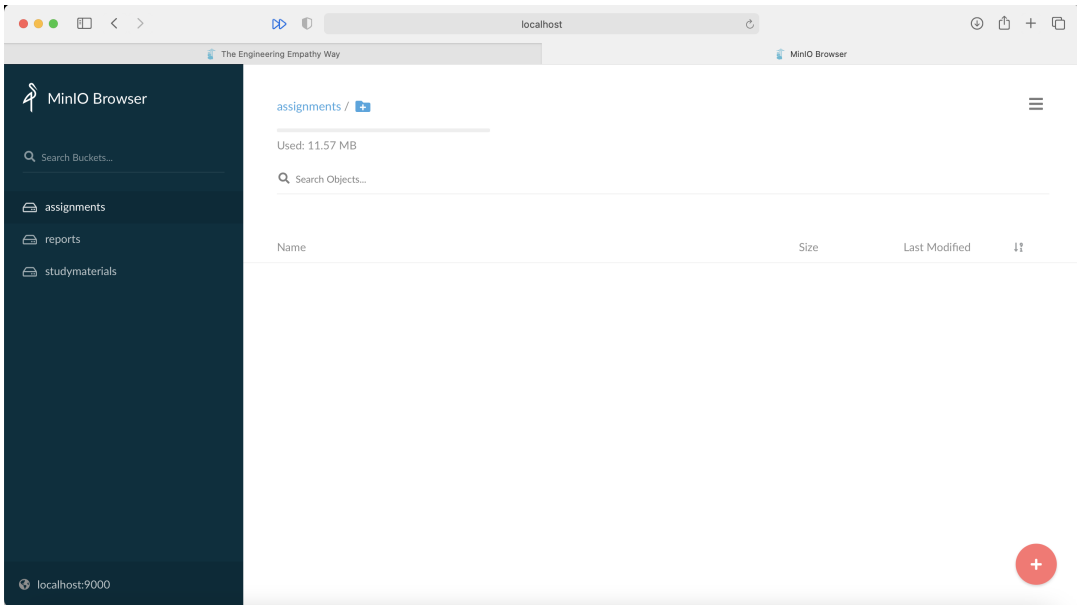

Figura 5.15. Corretto funzionamento di MinIO

In The Engineering Empathy Way, l'archiviazione dei file è stata organizzata in 6 bucket:

- *Assignments:* qui vengono salvati tutti i file che riguardano gli esercizi assegnati dai docenti dei vari corsi. Questo bucket può contenere anche sottodirettori e ogni file viene salvato con il nome dell'identificativo utilizzato nel database

- *Chats:* qui vengono salvate tutte le immagini per le chat di gruppo. Ogni immagine viene salvata con il nome dell'identificativo della chat utilizzato nel database

- *ProfileImages:* qui vengono archiviate tutte le immagini di tutti i profili degli utenti. Ogni immagine viene salvata con nome corrispondente alla matricola dell'utente

- *Reports:* qui vengono salvati tutti i file che riguardano gli esercizi assegnati dai docenti dei vari corsi. Ogni immagine viene salvato con il nome dell'identificativo utilizzato nel database

- *StudyMaterials:* qui vengono archiviati tutti i file che riguardano il materiale ufficiale di apprendimento, rilasciato dai docenti dei vari corsi oppure dagli studenti che ne fanno richiesta. Questo bucket può contenere anche sottodirettori e ogni file viene salvato con il nome dell'identificativo utilizzato nel database

- *VideoPills:* qui vengono salvati tutti i file che riguardano le video pillole rilasciate dagli studenti del corso Engineering Empathy e messe a disposizione della community. Ogni video viene salvato con il nome dell'identificativo utilizzato nel database

5.5 La chat

5.5.1 Introduzione

Si vuole analizzare l'intero meccanismo di messaggistica istantanea, implementato in The Engineering Empathy Way, in questa sezione completamente dedicata. In questo caso, sia il frontend sia il backend sono stati interamente sviluppati dal sottoscritto.
In The Engineering Empathy Way, si possono creare delle chat tramite cui messaggiare istantaneamente con altri membri della comunità. Per accedere alla sezione delle chat, è sufficiente cliccare sul tasto corrispondente nella toolbar in alto; in questo modo, l'utente atterra nella home della sezione chat: alla sinistra c'è la barra di navigazione, in cui l'utente può gestire le varie chat e aprire quella che desidera, mentre sulla destra vi è una schermata di benvenuto, come mostrato nella seguente figura:

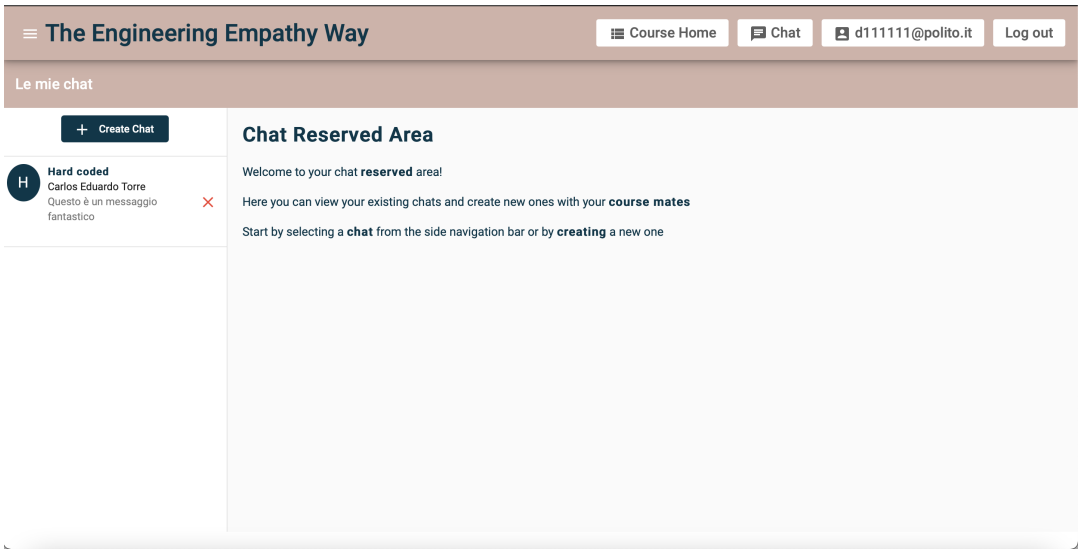

Figura 5.16. Sezione chat - schermata di benvenuto

Nella barra di navigazione sul lato sinistro, l'utente può creare nuove chat cliccando sul bottone *Create chat*. Vi sono due tipi di chat: possono creare: singole, qualora sia tra due persone, e di gruppo, se composta da tre o più persone:

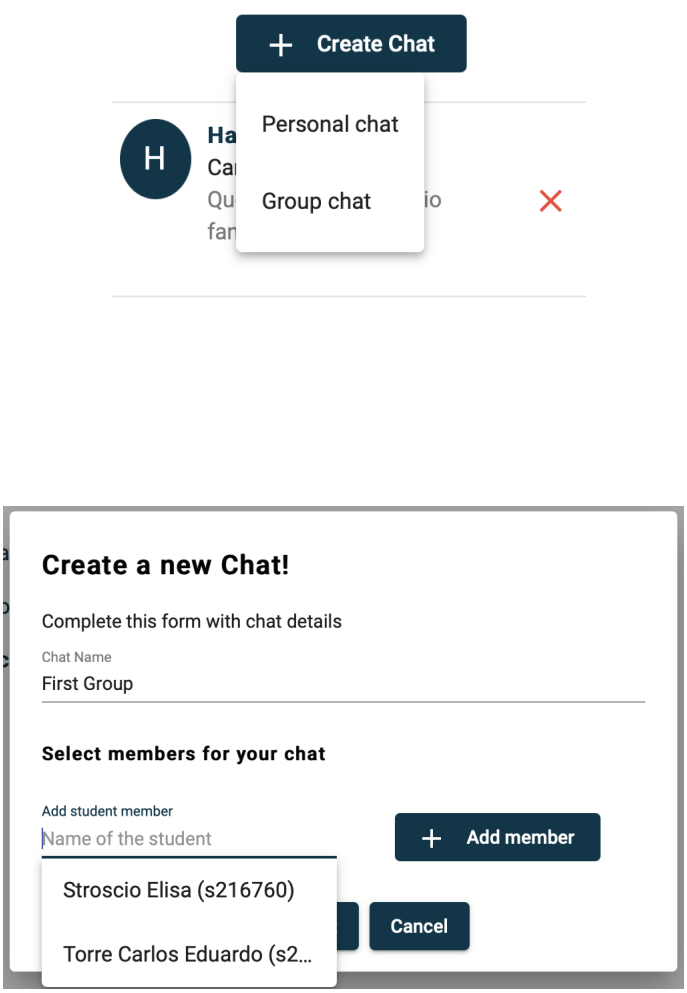

Figura 5.17. Creazione di una nuova chat

In seguito, è possibile inviare messaggi, agli altri membri, all'interno della chat; la figura sottostante riporta un esempio di messaggi scambiati in una chat di gruppo:

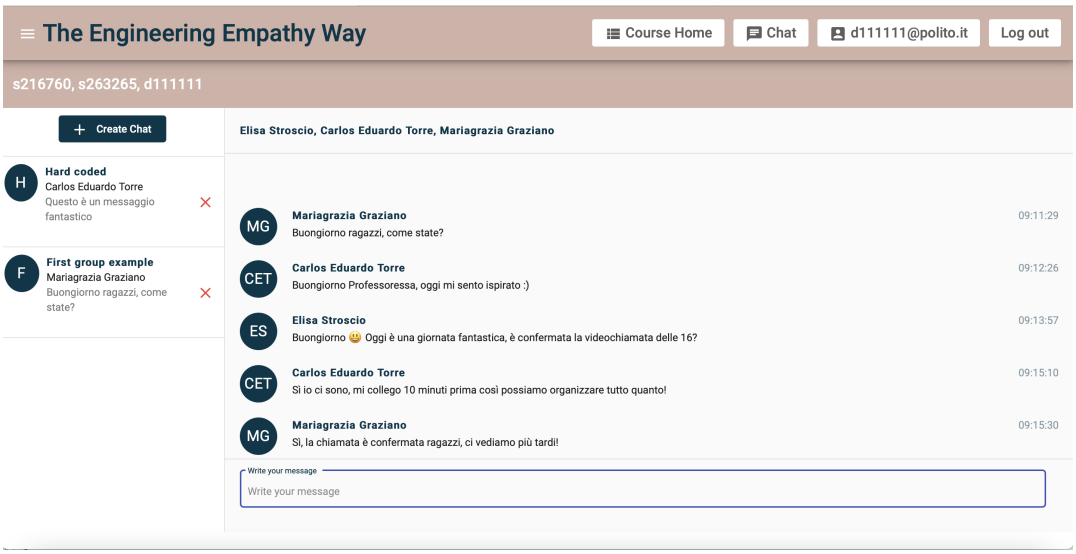

Figura 5.18. Esempio di messaggi scambiati

Per eliminare una chat presente, è sufficiente cliccare sulla x rossa che compare nella barra di navigazione di fianco alla chat corrispondente, facendo comparire così una finestra che richiede conferma dell'operazione:

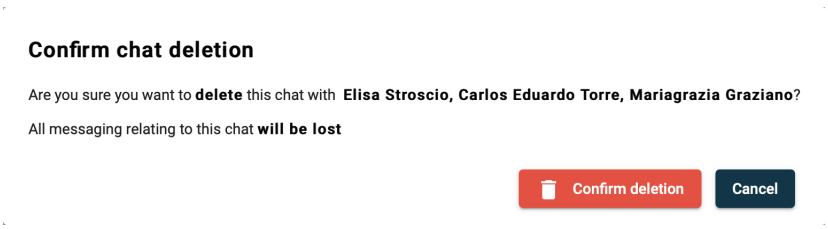

Figura 5.19. Eliminazione chat

L'intero processo di messaggistica istantanea è stato implementato attraverso l'utilizzo dei *Web Scoket*: pertanto, è stato necessario implementarli sia all'interno del client (frontend) sia all'interno del server (backend).

5.5.2 Frontend

Per poter utilizzare i web socket, è stato necessario installare le classi SockJS [51] (in particolare, SockJS-client [52]) e StompJS [53] tramite i comandi:

npm install sockjs-client; *npm install stompjs*

Una volta installati i moduli necessari, è stato creato un servizio apposito, *WebSocketService*, che viene utilizzato dal servizio *ChatService* per permettere all'utente di connettersi alla chat selezionata:

```
import { Injectable } from '@angular/core';
import * as Stomp from 'stompjs';
import * as SockJS from 'sockjs-client';
import { Message } from '../models/message.model';


@Injectable({
  providedIn: 'root'
})
export class WebSocketService {

  webSocketEndPoint: string = 'http://localhost:8080/ws';
  topic: string = "/topic/messages/";
  stompClient: any;
  /*options = {
    key: require("sslKey"),
    cert: require("sslCert")
  };*/

  constructor() {
  }

  _connect(chatId: number, callback: Function) {
      console.log("Initialize WebSocket Connection");
      let ws = new SockJS(this.webSocketEndPoint); //,this.options);
      this.stompClient = Stomp.over(ws);
      const _this = this;
      _this.stompClient.connect({}, function () {
        _this.stompClient.subscribe(_this.topic+chatId, function (response) {
          callback(response);
        });

          _this.stompClient.reconnect_delay = 2000;
      }, this.errorCallBack);
  };

  _disconnect(chatId: number) {
      if (this.stompClient !== null) {
          this.stompClient.disconnect();
      }
      console.log("Disconnected");
  }

  // on error, schedule a reconnection attempt
  errorCallBack(error: any) {
```

```
46        console.log("errorCallBack -> " + error)
47        /*setTimeout(() => {
48            this._connect();
49        }, 5000);*/
50    }
51
52    _send(chatId: number, message: Message) {
53      console.log("calling logout api via web socket");
54      this.stompClient.send(`/API/${chatId}/messaggi`, {}, JSON.
      stringify(message));
55    }
56
57
58    onMessageReceived(response) {
59        console.log("Message Recieved from Server :: " + response.
      body);
60    }
61 }
```

Quando l'utente seleziona la chat *chatId* desiderata, il componente *MessageListComponent* utilizza il servizio *ChatService* per connettersi alla chat: questo servizio richiama la funzione *_connect* del *WebSocketService*, che instaura una connessione con il server tramite il socket e sottoscrive l'utente all'URL

http://localhost:8080/ws/API/topic/messages/chatId

in maniera tale che possa ricevere i messaggi inoltrati dal server. Allo stesso tempo, il *ChatService* passa come parametro la funzione *callback*, che viene salvata per essere richiamata quando il server inoltra un messaggio ai membri della chat; la callback appartiene al *MessageListComponent* e si occupa appunto di gestire il messaggio ricevuto e di mostrarlo a video.

Una volta in cui l'utente esce da quella chat (per entrare in un'altra chat oppure spostarsi in un'altra sezione dell'applicazione), il componente *MessageListComponent* utilizza il servizio *ChatService* per disconnettersi dalla chat: richiama la funzione *_disconnect* del *WebSocketService* che provvede a disconnettere l'utente dalla chat in questione.

5.5.3 Backend

Per poter usufruire dei web socket, come prima cosa è stata importata la dipendenza necessaria nel file *pom.xml*, come mostrato nella sottosezione 5.4.1, riportata di seguito per semplicità:

```
1 <dependency>
2     <groupId>org.springframework.boot</groupId>
3     <artifactId>spring-boot-starter-websocket</artifactId>
4     <version>2.4.1</version>
```

```
</dependency>
```

Dopodiché, è stato creato un direttorio per ospitare la classe *WebSocketConfig* in cui è stata inserita la configurazione necessaria al corretto funzionamento del meccanismo di messaggistica istantanea:

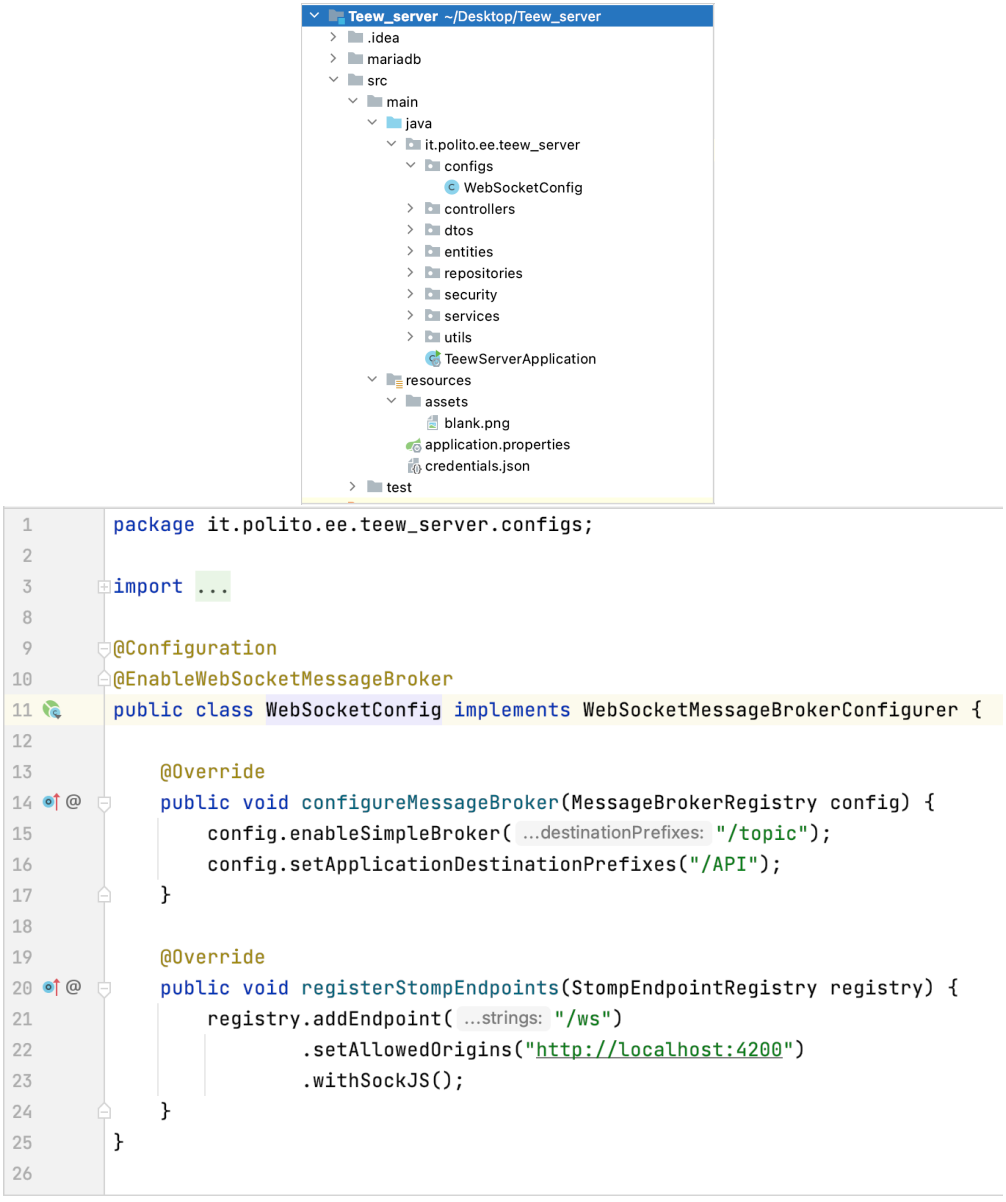

Figura 5.20. Configurazione chat

È stato registrato un endpoit */ws* a cui possono giungere richieste provenienti dall'origine *http://localhost:4200*; con esso, è stato configurato un *Broker* al prefisso

/API/topic.
Il server espone *ChatRoomSocketController*, un controllore apposito che, quando riceve i messaggi per una determinata chat *chatId* all'URL

http://localhost:8080/ws/API/chatId/messaggi

si occupa di inoltrarli agli altri membri, che si sono opportunamente sottoscritti a *http://localhost:8080/ws/API/topic/messages/chatId* per ricevere tutti i messaggi che vengono inviati istantaneamente in tale chat da qualsiasi membro. La figura sottostante mostra il controllore che si occupa di ricevere il messaggio inviato da un membro della chat a *http://localhost:8080/ws/API/chatId/messaggi*, di renderlo persistente tramite lo strato di servizio e, infine, di inoltrarlo ai membri della chat sottoscritti a *http://localhost:8080/ws/API/topic/messages/chatId*:

```
1  package it.polito.ee.teew_server.controllers;
2
3  import ...
11
12 @Controller
13 public class ChatRoomSocketController {
14
15     @Autowired
16     private ChatRoomService chatRoomService;
17
18
19
20
21     @MessageMapping("/{chatRoomId}/messaggi")
22     @SendTo("/topic/messages/{chatRoomId}")
23     public MessageDTO sendMessage(@DestinationVariable Long chatRoomId, MessageDTO messageDTO) {
24         MessageDTO m = chatRoomService.addMessage(chatRoomId, messageDTO);
25         return m;
26     }
27 }
28
```

Figura 5.21. Controllore esposto dal server per la gestione dei messaggi

5.6 Il deployment

5.6.1 Docker

Una delle parti più difficili da affrontare quando si sviluppa un'applicazione è il suo deployment, ossia la sua distribuzione al resto del mondo. Per effettuare il deployment, esistono principalmente due strade: virtualizzare gli host e i relativi sistemi operativi oppure creare dei container che ospitino le applicazioni nel contesto di un singolo sistema operativo, che può essere replicato su più host. Nel primo caso, l'obiettivo è quello di separare in maniera logica le risorse, disponibili su un certo

host, tra le varie applicazioni, consentendo l'ottimizzazione dell'utilizzo dell'hardware sottostante che viene suddiviso tra le varie applicazioni software, tramite la creazione delle virtual machine; i container, invece, pongono le radici sul concetto di virtualizzazione del software: permettono anch'essi la separazione delle risorse per le applicazioni, ma condividono il kernel con l'host, rendendo così possibile richiedere meno risorse hardware, visto che non devono emularlo per eseguire l'intero sistema operativo. Nel container viene inserito e impacchettato tutto ciò di cui necessita l'applicazione, come ad esempio le librerie da cui dipende e i servizi su cui si basa; in questo modo, è possibile eseguire l'applicazione su tutte le macchine. Il sistema operativo vede il container come un insieme di *namespace* diversi e isolati tra loro: i namespace sono un insieme di processi, creati tramite la system call *clone*, che possono accedere alle risorse kernel in maniera virtualizzata, come ad esempio:

- *PID:* viene definito un insieme indipendente di identificatori di processi
- *Mount:* trasformazione di path di file in riferimenti al file system sottostante
- *UserID:* insieme di identificativi per utenti e gruppi, con privilegi isolati
- *UTS:* definizione di strutture uname alternative
- *IPC:* politiche di comunicazione tra processi differenti
- *Network:* permette il controllo dello stack di rete

Il fatto che il filesystem di ogni container sia indipendente dagli altri permette di utilizzare versioni differenti di librerie condivise tra diverse applicazioni. Ogni virtual machine necessita della propria immagine del disco, il che porta ad utilizzare un file system stratificato, in maniera tale da non sovrautilizzare lo spazio di memorizzazione e supportare la presenza di più container nello stesso host; il file system perciò risulta essere uno stack di strati che possono essere solamente letti, con l'aggiunta di uno strato sia in lettura sia in scrittura per permettere il salvataggio di dati, e sarà il driver a fornire una vista unificata del file system.
I principali vantaggi dei container sono:

- *Scalabilità:* ogni host può ospitare tantissimi container diversi
- *Portabilità:* senza dover modificare qualcosa, è possibile portare il container da un ambiente di sviluppo a quello di test o di produzione.
- *Riutilizzo:* parti del container, o volendo anche tutto, possono essere riutilizzate a piacere

- *Incapsulamento:* nel container, viene impacchettato tutto il necessario per far funzionare l'applicazione, come il codice, le dipendenze, la configurazione di rete ecc.

- *Versionamento:* è possibile mantenere le varie versioni

- *Leggerezza:* i container richiedono meno risorse hardware rispetto alle virtual machine

Docker [54] è uno strumento open source che è stato realizzato per facilitare la distribuzione dei servizi, rendendo molto semplice riuscire a creare, eseguire il deployment e avviare le applicazioni attraverso un'implementazione standardizzata dei *container*. Lo slogan rende chiaro gli obiettivi di questo strumento potentissimo: *package once, run everywhere*. È formato da quattro componenti fondamentali:

- *Docker Engine:* si tratta di un processo demone che viene eseguito in background nell'host che ospita il container, permettendo ad esso di accedere a tutte le funzionalità offerte da Docker. Esso serve a standardizzare la gestione dei container, permettendo di spostare ed eseguire facilmente l'applicazione da un host a un altro, purché supportino Docker Enginer. Vi è una versione gratuita (CE, community edition) e una versione a pagamento (EE, enterprise edition).

- *Docker Client:* rappresenta un'interfaccia per le API esposte da Docker Engine, fornisce diversi comandi tramite cui è possibile accedere a tutte le funzionalità.

- *Docker Image:* un'immagine senza stata ed immbutabile che è composta da un insieme di file e parametri, atti a configurare l'applicazione da utilizzare a runtime

- *Docker Container:* un container è un'istanza in esecuzione di una precisa immagine; dispone di un file system stratificato che è costituito da uno strato di lettura e scrittura sovrapposto a tutti gli altri strati in sola lettura, comuni a tutti i container che si basano sulla medesima immagine

Per avviare un container, è sufficiente eseguire il comando *docker run <imageName>*: l'engine cerca nell'host l'immagine necessaria, se non la trova la scarica da un repository remoto per salvarla in locale, in maniera tale da utilizzare come template di base per far partire il container che avvierà le applicazioni al proprio interno. Una volta avviato il container, viene identificato tramite un ID, creato uno strato di file system in lettura e scrittura per i dati, creato un file system sottostante basato su UnionFS, allocata un'interfaccia di rete con un certo indirizzo IP e, infine, eseguito il processo indicato nel Dockerfile.

Tutto questo procedimento si basa sull'utilizzo di uno o più file di testo *Dockerfile* opportunamente definito dal programmatore. Il Dockerfile serve a descrive l'immagine che si vuole costruire tramite una serie di istruzioni, che non sono case sensitive. Una volta scritto il Dockerfile, è possibile costruire l'immagine desiderata tramite il comando

docker build .

che dev'essere eseguito nella medesima directory in cui è presente il Dockerfile, con questo preciso nome, perché questa directory rappresenta il *context build*, ovvero il contesto da cui i file sono inviati al Docker Engine per essere elaborati. Per quanto riguarda The Engineering Empathy Way, sono stati creati tutti i Dockerfile necessari e si trovano nel branch *docker* all'interno del repository del progetto.

Docker offre uno strumento aggiuntivo fondamentale per il deployment di applicazioni più complesse basate sempre su Docker: *Docker Compose*. Esso si basa su un singolo file *YAML* che permette di attivare l'insieme di eseguibili tramite un unico comando. In particolare, uno stack applicativo definito grazie a Docker Compose può essere esseguito su un singolo host oppure all'interno di un cluster, ma per questo intervengono *Docker Swarm* oppure *Kubernetes*, altri due strumenti fondamentali. Docker compose permette di gestire l'intero ciclo di vita dell'applicazione e dei singoli servizi presenti al suo interno.
Il processo prevede la creazione di un Dockerfile per ogni servizio che fa parte dell'applicazione (oppure bisogna disporre delle immagini relative), definire i servizi nel file *docker-compose.yml*, eseguire *docker-compose build* e, infine, eseguire *docker-compose up*; per arrestare il processo in esecuzione, sarà sufficiente eseguire il comando *docker-compose down*.

Per quanto riguarda The Engineering Empathy Way, è stato creato il file *docker-compose.yml* necessario ad effettuare il deployment dell'intera applicazione. Si trova nel branch *docker* all'interno del repository del progetto e, al momento attuale, presenta un errore relativo al meccanismo di messaggistica istantanea, pertanto non è stato possibile effettuare il deployment finale dell'applicazione: si considera questo passo come spunto di miglioramento in futuro.

5.7 Aggiunte e miglioramenti futuri

5.7.1 Servizio mail

Al momento attuale, la conferma della registrazione all'applicazione avviene tramite un link che viene stampato sulla console. L'applicazione è stata creata con tutto il codice necessario affinché questo link possa essere inviato alle mail ufficiali del

Politecnico di Torino, sia dei docenti sia degli studenti, tramite un servizio email. È necessario scegliere quale adottare e configurarne i parametri necessari nelle righe di codice corrispondenti.

5.7.2 Documentazione

È necessario creare una documentazione online che sia usufruibile da parte di tutta la comunità, in maniera tale che chiunque possa avere la possibilità di comprenderne il funzionamento e apportarne i miglioramenti desiderati.

5.7.3 Sicurezza e ottimizzazione del codice

È necessario implementare ulteriori meccanismi di sicurezza, a partire dall'utilizzo del protocollo HTTPS: a questo proposito, si deve fare attenzione a riuscire a integrare l'utilizzo del protocollo HTTPS con quello dei web socket, necessari per il corretto funzionamento del meccanismo di messaggistica istantanea.
Inoltre, è fondamentale effettuare molti più controlli sugli input ricevuti, sia per quanto riguarda il frontend sia, soprattutto, per quanto riguarda tutto il backend.
Infine, sarebbe opportuno ottimizzare il codice, sia in termini di quantità di righe di codice sia di prestazioni finali per garantire una User Experience migliore.

5.7.4 User experience

L'applicazione è stata sviluppata seguendo lo standard di Angular Material ma, nonostante questo, presenta varie lacune per quanto riguarda l'usabilità da parte dell'utente: dev'essere necessariamente migliorata affinché l'applicazione diventi molto più gradevole e fluida da utilizzare.
In aggiunta, al momento attuale, tutto il frontend è ottimizzato per un utilizzo via desktop. È necessario ottimizzare l'applicazione affinché sia facilmente utilizzabile anche attraverso i dispositivi mobili.

5.7.5 Nuove funzionalità

Ad oggi, The Engineering Empathy Way fornisce soltanto le funzionalità base per le quali è stata progettata. Sarebbe meraviglioso se in futuro venissero aggiunte molte più funzionalità. Dopo aver perfezionato tutte le funzionalità base, l'obiettivo sarebbe di arrivare a creare nuove funzionalità legate all'AI (intelligenza artificiale, in particolare al Machine Learning). Un esempio, tra le infinite possibilità di funzionalità molto avanzate che si potrebbero implementare in futuro, è rappresentato dal riconoscimento delle microespressioni facciali: questo apporterebbe un notevole miglioramento sul percorso di apprendimento e dello sviluppo dell'empatia. Un altro esempio potrebbe essere un meccanismo di riconoscimento automatico della

qualità delle presentazioni effettuate dai membri della comunità: questo permetterebbe di poter ricevere dei feedback anche molto oggettivi sullo svolgimento degli esercizi proposti e apporterebbe un salto enorme nel processo di miglioramento per un determinato percorso. Un ulteriore passo in avanti consisterebbe nel progettare soluzioni di realtà virtuale per esporre le persone nelle condizioni di effettuare virtualmente discorsi in pubblico, in maniera tale da ottenere enormi benefici potendo praticare anche in maniera indipendente: i benefici di questa pratica sono molti e duraturi [55], [56], [57].

5.7.6 Deployment

È essenziale portare a termine il deployment dell'intera applicazione seguendo quanto descritto nella sezione 5.6. Soltanto in questo modo sarà possibile rendere disponibile l'applicazione a tutta la comunità di studenti.

5.7.7 Hateoas e microservizi

Hateoas

Come analizzato nel capitolo dedicato, vi sono quattri livelli di adesione al modello di maturità di Richardson. Uno dei miglioramenti futuri per The Engineering Empathy Way è sicuramente quello di rendere l'applicazione completamente aderente al livello 3 del modello di Richardson. In questo modo, sarà possibile scoprire dinamicamente i servizi offerti dal server REST rendendolo più documentato in quanto, per ogni oggetto trasferito, il server aggiungerà una serie di collegamenti URI che mostreranno quali altre azioni possono essere compiute sull'oggetto stesso; questo spianerà anche la strada verso l'introduzione di un'architettura a microservizi.

Tutto questo sarà possibile grazie all'introduzione di un particolare componente dell'architettura REST: HATEOAS, che sta per Hypermedia As The Engine Of the Application State. In Spring vi è direttamente un'implementazione, che si chiama appunto Spring HATOEAS [58], una libreria che offre una serie di API per un pieno supporto alla realizzazione di servizi che aderiscono al livello 3 del modello di Richardson. Per importare Spring HATOEAS come dipendenza del progetto, si deve inserire la seguente dipendenza nel file pom.xml:

```
<dependency>
    <groupId>org.springframework.hateoas</groupId>
    <artifactId>spring-hateoas</artifactId>
    <version>1.0.4.RELEASE</version>
</dependency>
```

Dopodiché si utilizzerà la classe *org.springframework.hateoas.RepresentationModel* come classe base da estendere per far sì che le classi che derivate possano inserire dei link dinamici all'interno delle loro istanze.

Microservizi

I microservizi [59] rappresentano un'architettura distribuita che prevede che il server sia costituito da un insieme di servizi indipendenti tra loro. Ognuno di questi componenti, che risultano per tanto essere unità di software che può essere rimpiazzato e aggiornato in maniera indipendente, viene eseguito nel proprio processo, o addirittura contenitore o macchina; possiede inoltre tutti gli strati (database ecc.) di cui ha necessità per funzionare correttamente e comunica con gli altri attraverso il protocollo HTTP. Per mantenere tutto il sistema funzionante e coeso, vi è una piccola parte di logica centralizzata. Questo tipo di architettura presenta molti vantaggi ma anche qualche svantaggio:

- *Vantaggi:* permette di distribuire velocemente e in maniera affidabile delle applicazioni che nell'insieme possono risultare pesanti e complesse; favorisce la scalabilità, la manutenibilità e la testabilità dell'intero sistema; favorisce la modularità in quanto ogni modulo vive nel proprio processo/contenitore/macchina e i vari servizi risultano debolmente accoppiati tra loro; ogni modulo ha il proprio ciclo di vita e può essere distribuito (deployment) indipendentemente dagli altri moduli che possono continuare a restare in piedi; fornisce supporto all'utilizzo di diversi linguaggi di programmazione e anche di tecnologie differenti nei diversi moduli; è possibile organizzare l'intera logica intorno alle funzionalità che si vogliono fornire

- *Svantaggi:* l'ambiente distribuito può creare problemi in quanto le interazioni sono asincrone e potrebbero fallire, ciò comporta una complessità di programmazione e debug maggiore; la manutenzione diventa più complessa in quanto i sistemisti devono mantenere tanti processi separati; la consistenza si indebolisce perché basta anche un solo fallimento tra le varie interazioni per aggiungere complessità

Capitolo 6

Conclusioni ed aspettative future

Nel capitolo 2 sono state presentate le informazioni ottenute tramite il sondaggio e le interviste effettuate; nel capitolo 3 sono stati approfonditi tutti gli aspetti teorici sull'empatia e le competenze trasversali, derivate dall'empatia, maggiormente richieste al giorno d'oggi agli ingegneri e ai neolaureati ingegneri; nel capitolo 4 si è mostrata la necessità di un cambiamento nel percorso di studi ingegneristici e si è analizzato il modello di apprendimento 70:20:10 da adottare per lo sviluppo dell'empatia e delle competenze trasversali; infine, nel capitolo 5 si è presentata la nuova applicazione The Engineering Empathy Way come prima soluzione applicabile nel breve termine per migliorare dal punto di vista relazionale la figura del neolaureato ingegnere.

Non è stato semplice trovare una letteratura scientifica che supportasse sia la validità dell'ipotesi iniziale sia gli aspetti teorici trattati riguardanti l'empatia e le competenze trasversali: questo ha presentato un grande limite allo studio proposto. Nonostante questo, ci sono tutti i presupposti affinché tutto il lavoro presentato trovi il debito sostegno scientifico in futuro e si spera che questi temi vengano approfonditi maggiormente.

La fase di raccolta informazioni e analisi teorica di tutti gli strumenti hanno impiegato mesi di lavoro e ricerca, tempo che è stato sottratto dalla possibilità di terminare l'applicazione in tempo, di renderla completa e disponibile fin da subito alla comunità del Politecnico di Torino. Ci si augura che il lavoro venga proseguito da altri studenti desiderosi di portare a termine l'operato e di continuare a credere che il percorso universitario possa essere migliorato nella direzione proposta dal presente lavoro.

Bibliografia

[1] https://www.google.com/forms/about/

[2] Waldinger R., *What makes a good life? Lessons from the longest study on happiness* [Video], Conferenze TED, https://www.ted.com/talks/robert_waldinger_what_makes_a_good_life_lessons_from_the_longest_study_on_happiness?referrer=playlist-the_most_popular_talks_of_all, (2015, Novembre)

[3] Goleman D., *Intelligenza emotiva. Che cos'è e perché può renderci felici*, Italia, Bur Rizzoli (Best BUR), 2019

[4] Hoople, G. D., Choi-Fitzpatrick A., *Engineering Empathy: A Multidisciplinary Approach Combining Engineering, Peace Studies, and Drones*, Columbus (Ohio), 2017, 10.18260/1-2–28251

[5] Kumar S., Hsiao J. K., *Engineers Learn "Soft Skills the Hard Way": Planting a Seed of Leadership in Engineering Classes*, 2007, https://doi.org/10.1061/(ASCE)1532-6748(2007)7:1(18)

[6] Stewart I., Joines V., *L'analisi transazionale. Guida alla psicologia dei rapporti umani*, Italia, Garzanti, 2019

[7] Rosenberg M. B., *Nonviolent communication. A Language of Life*, 3 ed., Encinitas (California), PuddleDancer, 2015

[8] Sinek S., *Start with why. How great leaders inspire everyone to take action*, Harlow (England), Penguin Books, 2011

[9] Sinek S., *How great leaders inspire action* [Video], Conferenze TED, https://www.ted.com/talks/simon_sinek_how_great_leaders_inspire_action, (2009, Settembre)

[10] https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.flickr.com%2Fphotos%2Fgavinjllewellyn%2F6353289537&psig=AOvVaw2r8S08U06yclCrIZ95Cwsl&ust=1617272673363000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCNCRiK-o2u8CFQAAAAAdAAAAABAJ

[11] MacLean, Paul D., *The Triune Brain in Evolution: Role in Paleocerebral Functions*, New York, Plenum Press, 1990

[12] MacLean P. D., *Triune Brain. In: Comparative Neuroscience and Neurobiology. Readings from the Encyclopedia of Neuroscience*, Boston (USA), Birkhäuser, 1988, https://doi.org/10.1007/978-1-4899-6776-3_51

[13] Sinek S., *Leaders eat last. Why some teams pull together and others don't*, London (England), Portfolio Penguin, 2017

[14] https://www.studiotrevisani.it/la-scala-dei-livelli-di-ascolto/

[15] https://www.studiotrevisani.it/corso-sullempatia-e-ascolto-attivo/

[16] De Bono E., *Sei cappelli per pensare. Manuale pratico per ragionare con creatività ed efficacia*, Italia, Bur Rizzoli (Best BUR), 2019

[17] https://www.toastmasters.org

[18] https://www.ted.com

[19] https://today.yougov.com/topics/lifestyle/articles-reports/2014/03/27/argh-snakes

[20] Cuddy, A.J.C., *Your body language may shape who you are* [Video], Conferenze TED, https://www.ted.com/talks/amy_cuddy_your_body_language_may_shape_who_you_are, (2012, Giugno)

[21] Cuddy A.J.C., Wilmuth CA, Yap AJ, Carney DR., *Preparatory power posing affects nonverbal presence and job interview performance*, J Appl Psychol. 2015 Jul;100(4):1286-1295. doi: 10.1037/a0038543. Epub 2015 Feb 9. PMID: 25664473.

[22] Cuddy A.J.C., Wilmuth C. A., Carney D.R., *The Benefit of Power Posing Before a High-Stakes Social Evaluation*, Harvard Business School Working Paper, No. 13-027, September 2012.

[23] Harris R., *The happiness trap. Stop struggling, start living*, London (England), Robinson Publishing, 2008

[24] https://www.ccl.org

[25] Johnson SJ, Blackman DA, Buick F., *The 70:20:10 framework and the transfer of learning. Human Resource Development Quarterly*, 2018;29:383–402, https://doi.org/10.1002/hrdq.21330

[26] , Walther J., Miller S. E., Sochacka N., *A Model of Empathy in Engineering as a Core Skill, Practice Orientation, and Professional Way of Being: A Model of Empathy in Engineering*, Journal of Engineering Education, 2017, 106. 123-148. 10.1002/jee.20159.

[27] , Schulz B., *The importance of soft skills: Education beyond academic knowledge*, 2008, Journal of Language and Communication. 2. 10.1016/0006-3207(93)90452-7.

[28] , Colman B., Willmot P., *How Soft Are 'Soft Skills' in the Engineering Profession?*, 44th SEFI Conference, 12-15 September 2016, Tampere, Finland

[29] , Hess J. L., Fila N. D., *The Development and Growth of Empathy Among Engineering Students*, 2016 ASEE Annual Conference Exposition, New Orleans, Louisiana, 2016, June, ASEE Conferences, 2016. https://peer.asee.org/26120 Internet. 05 Apr, 2021
[30] https://github.com
[31] https://git.vlsilab.polito.it/poli-empathy/The-Engineering-Empathy-Way.git
[32] KuroseJ.F., Ross K.W., *Reti di calcolatori e internet. Un approccio top-down*, trad.it (*Computer networking. A top-down approach*, 6 ed., Pearson) 6 ed., USA, Pearson Education, 2013
[33] https://it.wikipedia.org/wiki/Autenticazione
[34] https://angular.io
[35] https://material.angular.io
[36] https://material.io/design
[37] https://nodejs.org/en/
[38] https://www.npmjs.com
[39] https://www.oracle.com/java/technologies/java-ee-glance.html
[40] https://maven.apache.org
[41] https://start.spring.io
[42] https://mvnrepository.com
[43] https://spring.io/projects/spring-boot
[44] https://spring.io/projects/spring-security
[45] https://dzone.com/articles/spring-boot-security-json-web-tokenjwt-hello-world
[46] https://softwareengineering.stackexchange.com/questions/373284/what-is-the-use-of-dto-instead-of-entity
[47] https://projectlombok.org
[48] https://staruml.io
[49] https://mariadb.org
[50] https://min.io
[51] https://github.com/sockjs
[52] https://github.com/sockjs/sockjs-client
[53] https://github.com/stomp-js/stompjs
[54] https://www.docker.com
[55] Vanni F, Conversano C, Del Debbio A, Landi P, Carlini M, Fanciullacci C, Bergamasco M, Di Fiorino A, Dell'Osso L., *A survey on virtual environment applications to fear of public speaking*, Eur Rev Med Pharmacol Sci. 2013 Jun;17(12):1561-8. PMID: 23832719.
[56] Nazligul M.D., Yilmaz M., Gulec U., Gozcu M.A., O'Connor R. V., Clarke P., *Overcoming Public Speaking Anxiety of Software Engineers Using Virtual Reality Exposure Therapy*, (2017), 191-202. 10.1007/978-3-319-64218-5_15

[57] Hinojo-Lucena, Francisco-Javier, *Virtual Reality Treatment for Public Speaking Anxiety in Students. Advancements and Results in Personalized Medicine*, Journal of personalized medicine vol. 10,1 14. 1 Mar. 2020, doi:10.3390/jpm10010014
[58] https://spring.io/projects/spring-hateoas
[59] https://microservices.io