UNIVERSITÀ
DI TRENTO

Ingegneria del Software - DISI - Anno Accademico 2023-2024

Daniele Ye - Alessandro Dalfovo. - Giovanni Panighel

Nome progetto:

Tripply

Gruppo G42 - Deliverables 4

Indice:

Approcci all'Ingegneria del Software: Analisi delle metodologie descritte nei seminari

1. **Blue Tensor:**

   Durante il seminario la medesima azienda ha esposto il metodo di ingegneria da loro applicato, ovvero il metodo Agile applicato allo sviluppo di intelligenze artificiali: analizzano il contesto di riferimento, le funzionalità richieste e definiscono obiettivi e limiti. Successivamente usando le User Stories ricavate dallo Story Mapping definiscono requisiti funzionali e non funzionali, come anche l'architettura hardware e software che utilizzeranno. Dopo aver preparato i dati utili alla loro applicazione, sviluppano le singole componenti del software, attraverso più fasi di sviluppo, training, test e deploy, per poi assemblarle nel prodotto finale. Questa metodologia di sviluppo permette di avere un contatto stretto con il cliente.

   Nel nostro progetto abbiamo fatto uso della metodologia Agile, soprattutto nella fase di produzione del codice.

2. **Kanban Simulation:**

   Durante il seminario il Dr. York Rüssler ha esposto il metodo Kanban: esso permette di mantenere alta la produttività di un'azienda evitando la sovrapproduzione. Il Dr. York ha poi organizzato una simulazione di applicazione del metodo Kanban nel contesto generale di lavoro di una azienda di sviluppo software. Questa simulazione ha mostrato che, mettendo un limite ai progetti in sviluppo e suddividendo il lavoro in modo che i lavoratori senza un proprio compito aiutino chi ne ha già, abbia aumentato la produttività dell'azienda rispetto alla solita suddivisione di lavoro, in cui ognuno pensa al proprio lavoro senza aiutare i propri colleghi.

   Durante il nostro progetto abbiamo fatto uso delle pratiche del metodo Kanban per velocizzare il lavoro svolto.

3. **IBM:**

IBM non ha presentato vere e proprie metodologie di Ingegneria del Software nel corso del seminario, invece ha voluto parlare dei mezzi da loro messi a disposizione per poter sviluppare progetti, ciò includono servizi per la sicurezza, come il Gate Appliance Juniper, per il lavoro in Cloud, come Code Engine, CloundantDB, uno starter kit Node-RED, Toolchain.

Per il nostro progetto abbiamo deciso di usare un servizio di cloud hosting per il deploy di nome Render. Non abbiamo usato quindi un servizio IBM, ma il seminario è servito comunque per avvicinarci a questo tipo di tecnologie.

4. META

Durante il seminario META ha esposto la loro struttura dei team di lavoro, e spiegando che non adottano un tipo specifico di Ingegneria del Software, e che quindi i team di sviluppo hanno pieno controllo su come pianificare il lavoro e il proprio progetto. Hanno specificato inoltre che danno molto peso a come e quanto una persona si possa adattare a nuovi modi di lavorare e linguaggi di programmazione. Prestano inoltre particolare attenzione a sviluppare prodotti che riescano a soddisfare la clientela, che siano solidi e che riescano a portare un successo dal punto di vista economico.

Sebbene non abbiano parlato di vere e proprie metodologie di ingegneria, abbiamo prestato attenzione ad essere flessibili per quanto riguarda i potenziali imprevisti e la loro risoluzione, come anche ad imparare e ad adottare i nuovi linguaggi di programmazione usati per la realizzazione del progetto.

5. U-Hopper

Durante il seminario il relatore Daniele Morandi, oltre a presentare ciò di cui si occupa l'azienda, cioè elaborare grandi quantità di dati per lo sviluppo di intelligenze artificiali, hanno esposto la loro metodologia di lavoro, ovvero Agile: dividono il progetto in piccole parti, sviluppando queste ultime attraverso delle milestones, che permettono di definire le issue in maniera dettagliata. Sulla base di queste issue definiscono le scadenze e il tempo di

lavoro del progetto. Una volta finita la parte di sviluppo passano alla fase di testing, e sulla base dei risultati di questi test fanno la revisione del codice. Una volta passata la revisione, le parti del progetto passano alla fase di produzione. Questa metodologia di lavoro permette a U-Hopper di lavorare e di individuare eventuali problemi legati all'implementazione e alla progettazione più efficientemente e più velocemente.

Durante lo sviluppo del nostro progetto abbiamo usato il più possibile la metodologia Agile.

6. RedHat

Il relatore Mario Fusco durante il seminario ha presentato la metodologia di lavoro Open Source: il codice sorgente è reso disponibile al pubblico e agli altri programmatori, che possono studiarlo e modificarlo. I maggiori vantaggi di fare progetti open source sono i seguenti: la conoscenza è resa disponibile a chiunque, anche a chi sta imparando a scrivere codice, altri programmatori possono contribuire a migliorare il codice già scritto, in ambito aziendale è ciò che hai scritto a contare (quindi si viene valutati tramite meritocrazia), non sono più necessari colloqui di lavoro, in quanto l'azienda che vuole assumere sa già in anticipo la qualità del programmatore e attorno al progetto si può formare una community formata da professionisti che può aiutare a migliorare il codice del progetto. Il problema più rilevante è che si devono trovare altri modi di monetizzare il prodotto, in quanto non è necessario pagare per usufruire di quest'ultimo.

Il seminario ci ha ispirato per usare nel nostro progetto Express, un framework per applicazioni web Node.js completamente open source.

7. Microsoft

Diego Colombo, relatore del seminario di Microsoft, ha esposto la tematica del software testing, spiegando il TDD, ovvero il Test Driven Development. Quest'ultimo ha l'importante funzione di individuare i bug e altri errori nel codice, tramite test case multipli per correggerli nel più breve tempo possibile.

Perciò il TDD permette di individuare errori in un tempo molto breve, ridurre il tempo di correzione del codice, aumentare la produttività del team di sviluppo e produrre codice più malleabile e flessibile. Questo tipo di test però non vengono fatti dal punto di vista del cliente, ma solo da quello funzionale della parte che viene testata.

Nel nostro progetto abbiamo deciso di adottare il TDD, utilizzando Mocha per effettuare i test.

8. Molinari

Il relatore Andrea Molinari ha portato la sua esperienza del campo dei Sistemi Legacy e dell'ingegneria del software. I sistemi Legacy sono sistemi informativi obsoleti in una parte o in tutte le sue componenti, ma che sono ancora utilizzati in molti ambiti (per esempio nella gestione di banche e nei software per studi commercialisti) per il semplice motivo che sono ancora affidabili, robusti, e sicuri dal punto di vista fisico, e che possono interagire con o essere inclusi nei oppure essere convertiti in sistemi di nuova generazione. Il mondo di questi sistemi sono pesantemente influenzati dall'ingegneria del software, in quanto sono richieste molte competenze inerenti proprio alla progettazione e sviluppo di sistemi hardware e software, tra cui la conoscenza di UML e altri linguaggi di alto livello. La gestione di progetti nel mondo legacy non sfrutta la metodologia Agile, se non in rarissime occasioni data la sua poca scalabilità per progetti legacy di grandi dimensioni, quanto quelle Predictive, e necessita di moltissime aree di competenze, costi e manodopera esperta. L'ambito legacy, per quanto tratti hardware e software datati, non è superfluo o inutile, ma fornisce ancora ambienti stimolanti, mentre non esiste più l'approccio ad ambienti isolati.

Nonostante il tema estremamente interessante, non abbiamo trovato nulla di utile da applicare al nostro progetto.

9. Marsiglia

In questo seminario il relatore Gerardo Marsiglia ha parlato del ciclo vitale delle applicazioni e la loro modernizzazione. Fondamentale ruolo ha l'ingegneria del software e la progettazione del software, in particolare la sua architettura (che comprende la tecnologia su cui gira il progetto, l'applicazione e i dati), dato che dopo il lancio del software si deve provvedere alla sua manutenzione. Nelle grandi aziende affinché essa venga eseguita correttamente è necessario coordinare efficientemente i team di sviluppo e rispettare gli standard architetturali stabiliti dall'architetto del software, sfruttando delle automazioni dove possibile (come per l'effettuazione dei test appena il codice è pronto), pratiche a cui il DevOps sta indirizzando. Tra le pratiche del DevOps usate durante i processi di delivery è il cosiddetto Continuous Delivery: le modifiche apportate al codice vengono regolarmente compilate e testate automaticamente e unite in un repository unico, in attesa poi di essere rilasciate al cliente.

Essendo la nostra un'applicazione ancora in fase prematura rispetto a quelle già esistenti in commercio e rispetto anche al suo ciclo di vita, non abbiamo potuto sfruttare al meglio le tematiche di questo seminario.

10. APSS & Trentino.ai

In questo seminario i relatori Alessandro Bazziga e Alessandro Zorer hanno discusso della complessità di gestione di un'azienda grande quanto l'Azienda Provinciale per i Servizi Sanitari, parlando anche degli aspetti più tecnici e vicini all'ingegneria del software.
In particolare Tizio ha parlato del Dipartimento Tecnologico dell'azienda, che si occupa di introdurre, mantenere e massimizzare i benefici delle nuove tecnologie; inoltre fornisce: le principali infrastrutture informatiche (cloud, data center etc...), le applicazioni con cui dottori e pazienti interagiscono, i servizi per l'analisi di dati e tutte le apparecchiature mediche. Tra i problemi che il dipartimento deve affrontare vi è la gestione delle macchine e dei software legacy e i costi (sia di denaro che di tempo) per aggiornarli con nuovi macchinari e/o software (che è uno dei motivi per cui raramente vengono affrontati in un'ottica Agile, e si preferiscono altre metodologie), la messa in

sicurezza dei dati dei pazienti, l'analisi dei dati dal punto di vista statistico e di ricerca e l'uso delle AI per migliorare la qualità operativa.

Nonostante le tematiche molto interessanti, non abbiamo trovato il modo per sfruttarle per il nostro progetto.

Organizzazione del lavoro

Inizialmente abbiamo lavorato assieme al D1 e al D2, suddividendo i vari paragrafi dei due documenti e assegnandoli in egual misura ai nostri componenti.
Successivamente per la stesura degli ultimi tre deliverable ognuno di noi ha lavorato a un documento specifico, sottoponendolo alla revisione da parte degli altri due.
In particolare, il D3 è stato scritto da Alessandro Dalfovo, il D4 da Daniele Ye e il D5 da Giovanni Panighel, mentre per l'implementazione del sito hanno lavorato assieme Daniele e Alessandro.
Abbiamo lavorato sia in modalità sincrona, organizzando il lavoro di ognuno e suddividendoci i compiti, sia asincrona, tenendo aggiornati gli altri compagni sugli sviluppi inerenti alla propria porzione di lavoro assegnata e prestando aiuto a chi avesse dei dubbi riguardanti i vari deliverable.

Ruoli e attività

Componente del Team	Ruolo	Principale Attività
Daniele Ye	Project Leader, Sviluppatore, Progettista, Redattore	Ha redatto la documentazione del D4, collaborato alla stesura del D1 e del D2 e ha sviluppato una parte del front-end e del back-end (in particolare la sezione del Login) e parte delle API. Ha inoltre

		revisionato il D1, il D2, il D3 e il D5.
Alessandro Dalfovo	Sviluppatore, Progettista, Redattore	Ha redatto il D3, collaborato alla stesura del D1 e del D2 e ha sviluppato la maggior parte sia del back-end che del front-end, prodotto la maggior parte delle API e svolto il testing dell'applicazione. Ha inoltre revisionato il D1, il D2, il D4 e il D5.
Giovanni Panighel	Analista, Progettista, Redattore	Ha redatto il D5, collaborato alla stesura del D1 e del D2 e ha revisionato tutti i documenti prodotti.

Carico e distribuzione del lavoro

Dai file di log abbiamo ricavato questa tabella, raffigurante la quantità di tempo di lavoro per ogni deliverable per ogni componente del gruppo. Daniele Ye e Alessandro Dalfovo hanno impiegato molte più ore nel D4 rispetto agli altri deliverable dato che includeva anche lo sviluppo del codice sorgente (del quale se ne è occupato maggiormente Alessandro Dalfovo) e non solo la documentazione (della quale se ne è occupato maggiormente Daniele Ye). Inoltre abbiamo speso più tempo per il D2 che per il D1, in quanto ci siamo ritrovati a cambiare più volte i diagrammi e le loro descrizioni contenuti nel documento. Il D5 infine ha richiesto molte ore per essere redatto in quanto non sempre siamo riusciti a seguire i seminari in presenza o online, e perciò Giovanni Panighel ha dovuto visionarli nuovamente.

	D1	D2	D3	D4	D5	TOT

Daniele Ye	18	28	1	83	1	131
Alessandro Dalfovo	8	10	23	71	1	113
Giovanni Panighel	11	19	1	1	25	57
TOT	37	57	25	155	27	301

Criticità

Inizialmente abbiamo riscontrato delle difficoltà nel definire i requisiti funzionali del progetto, rallentando la stesura sia del D1 che del D2, in quanto con l'avanzare del progetto non sempre i requisiti erano coerenti tra loro, soprattutto tra quelli del D1 e del D2. Abbiamo avuto problemi anche nella progettazione degli UML, dato che siamo sempre stati abituati a sviluppare il codice delle applicazioni prima di definire gli UML delle varie componenti. Inoltre durante la scrittura del codice di Tripply abbiamo dovuto modificare gli UML fatti precedentemente, in quanto ci siamo accorti che alcune parti del progetto non potevano essere realizzate come le avevamo ideate, mentre altre volte ci siamo accorti che non avevamo pensato ad alcuni componenti necessari che non erano incluso nei nostri UML iniziali.

Autovalutazione

Complessivamente siamo riusciti a coordinarci ottimamente e a lavorare con costanza. Indubbiamente Daniele Ye e Alessandro Dalfovo sono stati le persone che più hanno lavorato al progetto, in quanto sono stati i responsabili dello sviluppo del codice della applicazione, mentre Giovanni Panighel è stato il componente che ha contribuito di meno allo sviluppo del progetto, sebbene sia stato il responsabile della revisione dei vari documenti. In base alle considerazioni appena svolte e alla quantità di ore di lavoro attuate, la nostra autovalutazione è:

	Voto
Daniele Ye	29
Alessandro Dalfovo	27
Giovanni Panighel	26