Ping-Povo

DELIVERABLE 5 - G21

Marco Strada e Matej Del Coco

20 Dicembre, 2023

Contents

1 Approcci all'ingegneria del Software

In questa sezione vengono descritti sinteticamente i seminari presentati in aula durante l'anno. Ne riportiamo i punti di forza, spunti e criticità.

1.1 BlueTensor

La società BlueTensor Srl, e il suo CEO Jonni Malacarne, è specializzata nello sviluppo di progetti basati su AI, offrendo soluzioni **'taylor-made'** e piattaforme pronte all'uso. L'azienda è nota per la sua vasta esperienza e la partecipazione a **progetti multidisciplinari**, infatti alcuni dei settori in cui hanno collaborato variano dall'agricoltura al controllo qualità per prodotti.

Il **punto di forza** di BlueTensor risiede, oltre che nella grande coesione all'interno del team, nella **comprensione pratica** degli aspetti tecnici, dimostrata dalla capacità di sviluppare **soluzioni AI** altamente personalizzate. Questo aspetto risulta particolarmente ispirante anche per noi, evidenziando come l'ingegneria del software possa essere applicata in moltissimi ambiti a livello industriale (ciò che ora è chiamata **Enterprise 4.0**) e non solo.

Una **criticità** da noi riscontrata è la **specializzazione** necessaria per portare a termine ogni progetto. Come il CEO Jonni ha avuto modo di spiegare, per un progetto legato a uno studio di avvocati, serve una certa expertise sull'argomento, aspetto che potrebbe risultare problematico data la vasta gamma di progetti in ambiti molto diversi tra loro. Complessivamente, il seminario offre un'opportunità per esplorare l'intersezione tra l'ingegneria del software e l'intelligenza artificiale e il mondo delle industrie moderne.

1.2 Il Metodo Kanban

Il Kanban rappresenta una metodologia focalizzata sulla **visualizzazione del lavoro**, mantenendo un **flusso** costante e perseguendo il miglioramento continuo del processo. Ciò consente ai team di **gestire il lavoro** in modo più efficiente. Il processo utilizza con una **'lavagna' Kanban e sistemi Kanban**, ovvero una rappresentazione visiva del flusso di lavoro con lo scopo di controllare quest'ultimo in modo semplice ed efficace.

Kanban enfatizza il **limite di Work in Progress (WIP)**, prevenendo sovraccarichi al team e mantenendo il focus sul completamento dei compiti. Il lavoro segue un **sistema di pull**, dove i compiti vengono presi nel flusso di lavoro solo quando c'è la capacità di gestirli, evitando di spingere il lavoro indipendentemente dalla capacità disponibile. I principi chiave includono la **Gestione Visuale**, che aiuta i team a comprendere lo stato dei compiti in modo intuitivo; il **Miglioramento Continuo**, che incoraggia cambiamenti incrementali ed evolutivi per migliorare i processi nel tempo; e il **Flusso**, con l'obiettivo di ottenere un flusso di lavoro continuo e senza intoppi.

La **criticità** principale da noi riscontrata è che se il metodo non è applicato correttamente si corre il rischio di sovraccaricare il team e quindi di portare a un rallentamento generale.

1.3 IBM

IBM ci presenta la sua innovativa piattaforma IBM Cloud, un mezzo per esplorare in dettaglio i benefici e le sfide legate all'adozione del **Cloud Computing**. Questa piattaforma riveste un ruolo chiave nella **modernizzazione** dei sistemi software, nella trasformazione digitale (**Digital Twin**) e nell'apertura a nuove opportunità commerciali. Attraverso l'utilizzo del Cloud IBM, è possibile concentrarsi in modo più incisivo sullo sviluppo del codice, alleviando le preoccupazioni legate all'infrastruttura, e fruire di avanzate tecnologie come l'AI, Blockchain e Machine Learning.

Tuttavia, emergono delle sfide in termini di complessità delle architetture, **rischi di sicurezza** legati ad attacchi DDoS, malware e cybercrime, oltre alla necessità di ottimizzare le configurazioni per garantire sistemi più sicuri ed efficienti. Il seminario mira a stimolare una riflessione critica su tematiche cruciali come l'**ottimizzazione delle risorse**, la sicurezza informatica, il tutto con particolare attenzione all'ecosostenibilità nell'ambito dell'informatica industriale.

1.4 META

Il seminario META si focalizza sulle **responsabilità** nel software engineering, affrontando ambiti sia tecnici che non tecnici. Gli aspetti tecnici comprendono codifica, progettazione architetturale, gestione di progetti e mansioni correlate, mentre le responsabilità non tecniche includono il reclutamento (tramite interviste), la formazione del team e il mentoring.

Per quanto riguarda le **metodologie di sviluppo**, non è preso in considerazione un approccio specifico, con ciascun team che adotta le pratiche ritenute più efficaci. Gli strumenti utilizzati variano da Asana a Google Sheets per la gestione dei progetti. Riguardo ai linguaggi di programmazione, si fa uso di JS (React), SQL, C/C++/C, Java, Haskell, Python e Ocaml. La maggior parte degli strumenti e sistemi sono personalizzati e sviluppati internamente, con l'impiego di Mercurial per il controllo versione, VS Code e Buck come strumento di compilazione.

Tra i punti di forza vi è la **flessibilità** tipica di un approccio di tipo **AGILE** e l'utilizzo di una vasta gamma di linguaggi di programmazione, permettendo ai team di adattarsi alle esigenze specifiche. Tuttavia secondo noi tale diversità potrebbe generare sfide in termini di **coerenza e collaborazione** tra team che adottano strumenti e metodologie differenti.

1.5 U-Hopper

Il seminario U-Hopper fornisce un approfondimento sulle soluzioni di **intelligenza artificiale** e **data analysis**, evidenziando gli strumenti tecnologici e i framework utilizzati, sia proprietari che open source come Spark, MongoDB, MinIO etc. Particolare enfasi è posta sull'adozione dei migliori fornitori di servizi cloud per l'hosting e la distribuzione delle soluzioni.

Centrale è l'attenzione sul sistema di **big data**, comprendente le fasi di ingestione, elaborazione batch e in streaming, database in-memory, archiviazione e gestione dei compiti. Viene sottolineata l'importanza dei **log** per garantire stabilità e facilitare la risoluzione dei problemi, oltre alla revisione del codice mediante richieste di merge.

Con il mio compagno di sviluppo abbiamo trovato molto efficaci le best practices da loro chiamate **'sprint definition'** e la fase successiva **'sprint development'**, in quanto seguono uno approccio agile e allo stesso tempo dinamico, conforme alla nostra mentalità utilizzata nello sviluppo di Ping-Povo.

1.6 RedHat

Il seminario Red Hat, oltre a raccontare il 'viaggio' di Mario Fusco, mette in luce punti di forza, opportunità, sfide e livelli di coinvolgimento nel mondo dell'**open source**. Vengono enfatizzati i benefici derivanti dalla **collaborazione** nell'ambiente open source, quali condivisione della conoscenza, visibilità, revisione tra pari e supporto della comunità globale.

Si esamina il contesto aziendale legato all'open source, evidenziando come i clienti scelgano questo tipo di software per i **costi inferiori**, l'assenza di dipendenza da fornitori specifici e la migliore qualità generale. Tuttavia, si sottolinea che i clienti sono disposti a pagare per livelli di servizio adeguati.

Vengono esplorati i vari **livelli di partecipazione** a progetti open source, partendo dai principianti che segnalano bug fino agli elementi chiave del team. In particolare si mette in luce l'importanza delle relazioni pubbliche, dell'accettazione delle **contribuzioni esterne** e della capacità di mantenere un progetto open source sano nel tempo.

Questo ultimo punto in particolare punta a **aprire la nostra mente** a idee e spunti esterni e al saper includere questi all'interno del proprio progetto in modo efficace e senza scontri di interesse.

1.7 Microsoft

Il seminario Microsoft introduce il concetto di verifica automatizzata o manuale dei requisiti funzionali e non funzionali, fornendo un'analisi approfondita sul **testing**. Si focalizza sulla chiarezza nella definizione dei test, dei bug e delle terminologie associate, quali unit tests, functional tests, e altri tipi di test. Esplora inoltre le diverse motivazioni dietro l'esecuzione dei test, come la **prevenzione** della regressione comportamentale e prestazionale, l'identificazione di difetti noti e la verifica della conformità alle specifiche.

Il seminario presenta una serie di **strumenti di testing**, inclusi framework per test unitari, test di stress, di regressione e altri. Indica i contesti in cui il testing può avvenire, compresi l'ambiente di sviluppo, di build, di staging e quello di produzione. Vengono indicati inoltre i motivi per cui è importante condurre test, dalla scoperta di bug, all'esplorazione e all'adeguamento alle specifiche contrattuali. Tuttavia, emergono aspetti critici come la potenziale complessità nella gestione di un vasto corpus di informazioni e la mancanza di una chiara guida su quale strumento o approccio adottare in contesti specifici.

1.8 Molinari

Il seminario di Molinari riguardo la sistemazione delle **legacy systems** offre una visione dettagliata sui punti di forza e le criticità legate ai sistemi informativi obsoleti.

I sistemi legacy persistono perché ancora rispondono alle **esigenze** originali, garantendo potenza di calcolo, transazionalità e continuità aziendale. La loro conoscenza approfondita e affidabilità riducono i rischi operativi, contenendo una ricchezza di dati e **know-how** aziendale.

Alcuni punti critici sono: L'hardware e il software datati possono causare problemi di usabilità e incompatibilità; La **gestione e manutenzione** richiedono risorse e costi significativi; La presenza di sistemi legacy può limitare l'innovazione e aumentare i costi;

Nonostante queste criticità i sistemi legacy conservano un ruolo rilevante in un'azienda, richiedendo strategie ben progettate per la gestione, preservazione o, se necessario, migrazione verso soluzioni moderne.

1.9 Marsiglia

Il seminario sull'**Application Lifecycle Management** (ALM) introduce concetti fondamentali legati al ciclo di vita del software e alla **modernizzazione** delle applicazioni. Particolare attenzione è posta sulla gestione della complessità del software attraverso la chiara definizione dei requisiti e la decomposizione logica e fisica delle componenti software. Il seminario delinea l'evoluzione dei processi di sviluppo software, dalla tradizionale metodologia a cascata (waterfall) alle moderne pratiche come DevOps. Si sottolinea l'importanza dell'adozione di pratiche di automazione e di un **approccio lean e agile** nella gestione del ciclo di vita delle applicazioni.

Vengono inoltre esplorati concetti fondamentali legati all'architettura software, illustrando l'evoluzione dalle architetture monolitiche verso approcci moderni come i **microservizi** e le **applicazioni cloud-native**, con un focus sulla scalabilità, resilienza e l'impiego di container.

La presentazione termina sottolineando l'importanza della modernizzazione delle applicazioni tradizionali per massimizzare le opportunità offerte dalle nuove architetture, tecnologie e piattaforme.

1.10 APSS

Il seminario sull'Azienda Provinciale per i Servizi Sanitari (APSS) e il Dipartimento Tecnologie focalizza l'attenzione sulla **gestione coordinata delle attività sanitarie e sociosanitarie** nell'intero territorio provinciale. L'obiettivo principale è garantire la continuità e umanizzazione dei percorsi di cura, assicurando un'offerta equa e uniforme dei servizi sanitari su tutta la provincia. Ciò viene realizzato attraverso la **semplificazione e decentralizzazione** dei processi decisionali mediante l'adozione di nuove tecnologie e la **digitalizzazione** dei sistemi gestionali.

Il Dipartimento Tecnologie si occupa di diverse aree come l'ingegneria clinica, i sistemi informativi e l'integrazione dell'Intelligenza Artificiale (AI) nel settore sanitario. L'impiego crescente dell'AI nel campo sanitario comporta miglioramenti nella diagnosi, nell'efficienza operativa, nella ricerca medica e nell'assistenza personalizzata ai pazienti.

Tuttavia, l'introduzione dell'AI richiede una valutazione etica accurata e la gestione dei dati in conformità alle normative sulla **privacy**.

Il Fascicolo Sanitario Elettronico (FSE) assume un ruolo centrale, consentendo ai cittadini l'accesso alla propria storia clinica e la condivisione con i professionisti sanitari. Inoltre, agevola la gestione e l'**analisi dei dati** clinici per migliorare la fornitura dei servizi sanitari. Il Patient Summary, parte integrante del FSE, funge da punto centrale per l'accesso ai servizi del Servizio Sanitario Nazionale, supportando i professionisti sanitari nella diagnosi e nella **cura personalizzata** dei pazienti, oltre a fornire informazioni per ottimizzare i servizi sanitari.

L'approccio generale sottolinea l'**interconnessione dei dati** sanitari, la **facilità d'accesso** e l'utilizzo efficiente delle informazioni per migliorare l'assistenza sanitaria, mantenendo sempre un rigoroso rispetto per la **privacy** e la sicurezza dei dati dei pazienti.

2 Organizzazione del lavoro

Il lavoro è stato suddiviso quanto segue:

- Per ogni Deliverable abbiamo avuto un incontro iniziale per cominciare a definire l'obiettivo e una deadline personale. Successivamente abbiamo sempre cercato di individuare una strategia che sfruttasse i punti di forza di ciascuno di noi e da qui ci siamo ripartiti i vari compiti.
- Per la stesura dei documenti abbiamo deciso di usare il linguaggio di marcatura LaTeX, per garantire un'uniformità di stile e struttura tra tutti i documenti.
- Gli schemi sono stati creati tramite LucidChart, che mette a disposizione gratuitamente un'ampia selezione di diagrammi e schemi UML. Per quanto riguarda i mock-up abbiamo optato per Canva, servizio gratuito che fornisce numerosi supporti per la creazione di modelli mock-up.
- Per la parte di sviluppo software abbiamo utilizzato SvelteKit basato sul framework Svelte per la parte di front-end, TypeScript per la parte di back-end e SupaBase per quanto riguarda il Database.
- Durante il semestre, per confrontare le idee e progressi nello sviluppo del lavoro, ci siamo incontrati sia in ambito univeritario che personale in modo molto frequente, in modo da ripartirci il lavoro il più efficacemente e dinamicamente possibile.
- Da questo progetto sono emersi i punti nostri diversi punti di forza e differenze dovute anche dal diverso background di ciascuno di noi, con Matej proveniente da un Liceo Scientifico e Marco da un Liceo Classico.

3 Ruoli e Attività

COMPONENTE	RUOLO	PRINCIPALI ATTIVITÀ
Matej Del Coco	Sviluppatore Back-End / Grafico / Progettista	Ha dato il maggior contributo come progettista, ideando le varie funzioni e le varie componenti che fanno parte dell'applicativo. Ha contribuito anche alla stesura di diagrammi. Successivamente si è occupato dello sviluppo back end comprensivo di documentazione e testing.
Marco Strada	Project Manager / Sviluppatore Front-End / Analista	Il ruolo principale è stato quello della gestione del progetto e delle risorse umane. Si è occupato inoltre della scrittura della maggioranza dei documenti, della loro revisione generale e della realizzazione di diversi diagrammi. Successivamente ha sviluppato il front end dell'applicazione.

4 Carico e Distribuzione del Lavoro

	D1	D2	D3	D4	D5	TOT
Matej Del Coco	30	39	28	49	2	148
Marco Strada	32	35	39	40	7	153
TOT	62	74	67	89	9	301

Il carico di lavoro come si può vedere è stato bilanciato ed entrambi i componenti hanno contribuito pienamente a tutti i Deliverable, dividendosi equamente i compiti. Nel D1 il carico è stato quasi equo. Nel D2 Matej si è occupato maggiormente della creazione dei diagrammi, mentre Marco nella stesura di descrizioni testuali e dei requisiti non funzionali. Nel D3 Marco ha preso ha preso l'incarico di creare il diagramma delle classi con Matej che ha contribuito con codice in OCL. Nel D4 Matej si è occupato di back-end, mentre Marco di front-end. Il D5 è stato realizzato in maggioranza da Marco con la descrizione della maggior parte dei seminari, con Matej che si è occupato dei restanti seminari e della stesura del documento. La revisione finale dei documenti è stata effettuata in maniera congiunta.

5 Criticità

Essendo questo progetto pensato inizialmente per un gruppo da 3 persone, ed essendoci ritrovati a lavorare in 2, la mole di lavoro è stata molta. Un'altra criticità è stata data dal fatto che non sempre abbiamo mantenuto coerenza tra i documenti, sintomo dello sviluppo di idee e dell'applicazione con il passare del tempo. Spesso è capitato di dover ritornare a vecchi Deliverable per aggiustare qualche dettaglio cambiato nel corso del tempo. Inoltre è stato difficile inizialmente trovare la quadra sulle funzioni da implementare nell'applicativo, tuttavia tramite un dialogo continuo siamo riusciti a risolvere il tutto.

6 Autovalutazione

Nonostante la mancanza di un membro crediamo di aver svolto un lavoro notevole, sopperendo alla mancanza di un terzo compagno facendo spesso qualche sacrificio personale. Entrambi i membri del gruppo hanno contribuito quanto possibile a tutti i Deliverable, ognuno secondo le proprie possibilità e punti di forza. Alcuni squilibri nel monte ore sono dovuti semplicemente a una ripartizione del lavoro che andasse a benificiare le abilità di uno (D4 per Matej in particolare) o dell'altro (D3 per Marco). Possiamo dire che siamo soddisfatti del lavoro che abbiamo svolto e a fronte delle nostre considerazioni personali questa è la nostra autovalutazione:

Membro	Voto
Matej Del Coco	30
Marco Strada	30

DELIVERABLE 5, GRUPPO G21: Marco Strada e Matej Del Coco