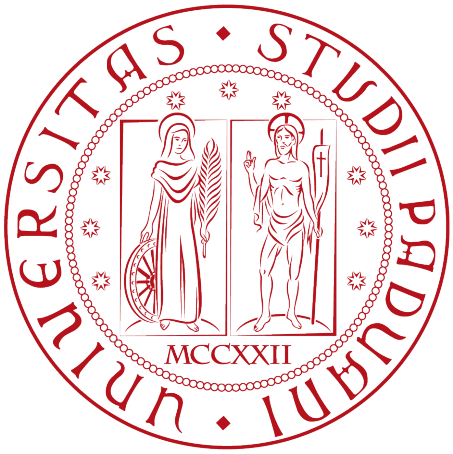
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Stato applicativo multi-finestra in JavaScript

Tesi di laurea triennale

Relatore

Prof.Gilberto Filè

Laureando

Giovanni Jiayi Hu

Anno Accademico 2017-2018

A rock pile ceases to be a rock pile the moment a single man contemplates it, bearing within him the image of a cathedral.

— Antoine de Saint-Exupéry, The Little Prince

Sommario

Il presente documento è la relazione finale del lavoro svolto durante il periodo di stage del laureando Giovanni Jiayi Hu presso l'azienda WorkWave Italy Srl, della durata di 312 ore.

Lo scopo dello stage è stato l'esplorazione e l'apprendimento delle più recenti tecnologie web per la realizzazione di applicazioni web multi-finestra. A tale scopo è stato necessario eseguire un'attività di Ricerca & Sviluppo (R&D), testarne la loro maturità e realizzare un Proof of Concept che sfrutti tali tecnologie per poter estrarre porzioni di interfaccia grafica dall'applicazione principale in una nuova pagina autonoma, ma sincronizzata a livello di stato applicativo.

La prima fase delle attività ha portato dunque alla nascita della prima versione di una libreria battezzata col nome *Stargate*, ispirato dall'ononima serie tv di fantascienza.

In secondo luogo è stato richiesta l'evoluzione e l'integrazione di Stargate nell'applicazione web in via di sviluppo denominata *WorkWave Route Manager*. Quest'ultima è la nuova versione di uno dei prodotti principali che l'azienda offre ai propri clienti e permette di pianificare, dirigere, tracciare e analizzare le rotte dei propri veicoli in tempo reale.
L'integrazione ha avuto difatti l'obiettivo di permettere la visualizzazione della mappa Google Maps, ricca di rotte ed veicoli, su un monitor separato full-screen.

Sia *Stargate* che *Route Manager* sono basati su TypeScript, un linguaggio tipizzato che compila in JavaScript, ed utilizzano le librerie React 16 e Redux 4. In particolare *Stargate* è usufruibile su qualsiasi applicazione web JavaScript, ma fornisce già le integrazioni per agevolarne l'uso con React e Redux.

Organizzazione del testo

Il secondo capitolo descrive ...

Il terzo capitolo approfondisce ...

Il quarto capitolo approfondisce ...

Il quinto capitolo approfondisce ...

Il sesto capitolo approfondisce ...

Nel settimo capitolo descrive ...

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

* gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;
* per la prima occorrenza dei termini riportati nel glossario viene utilizzata la seguente nomenclatura: *parola*[g];
* i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

Ringraziamenti

Innanzitutto, vorrei esprimere la mia gratitudine al Prof. Gilberto Filè, relatore della mia tesi, per la disponibilità che mi ha offerto durante il periodo di stage e per i preziosi insegnamenti durante questi tre anni.

Un ringraziamento speciale a Cesare d'Amico e a tutti i colleghi di WorkWave per la calorosa accoglienza fin dai primi giorni di stage. E naturalmente un sentito grazie a Matteo Ronchi, che ha saputo sempre guidarmi saggiamente durante questa esperienza e condividere pazientemente le sue preziose conoscenze.

Vorrei inoltre dare un caloroso abbraccio ad Elisa per l'affetto e per avermi insegnato ad amare i dettagli della vita. Un sentito ringraziamento invece per tutti gli amici, sia di università che di vita, i quali mi hanno sopportato e tenuto compagnia in questo percorso.

Desidero infine ringraziare con affetto i miei genitori per il costante sostegno ed incitamento nei miei progetti e nelle mie decisioni.

Padova, Settembre 2018 Giovanni Jiayi Hu

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

L'azienda

1.1 Descrizione

WorkWave, una divisione di IFS, è una società americana fondata nel 1984 con anche sede in Italia che fornisce soluzioni di Field Service Management e che connette ogni aspetto di un business attraverso le sue piattaforme unificate e di facile uso. L'insieme delle soluzioni della compagnia permettono ai professionisti di servizi ultimo-miglio di facilmente assegnare ed automatizzare attività di vendita e marketing, migliorando l'efficienza ed incrementando la visibilità delle operazioni sul campo attraverso le soluzioni mobile.

Le piattaforme di WorkWave forniscono ad oltre 8 mila clienti un livello senza precedenti di analisi del business, permettendolo loro di aumentare l'efficienza, il guadagno e garantendo un'eccezionale customer experience.

Figura 1.1: Logo dell'azienda WorkWave

1.2 Servizi offerti

WorkWave aiuta aziende nel campo Field Service Management ed industrie di trasporti e logistica mitigare gli aspetti dolorosi che incontrano ogni giorno, consentendo loro di salvare tempo, spese e migliorando il servizio agli utenti. Per Field Service Management si intendono risorse impiegate per intradare verso i domicili dei clienti, quali localizzazione dei veicoli, gestione delle attività degli operatori, pianificazione ed impiego delle attività, garanzia della sicurezza dei conducenti ed integrazione di tali servizi con depositi, fatturazione ed altri servizi back-office.

WorkWave fornisce sia soluzioni per l'installazione e manutenzione hardware che servizi software per l'aiuto della gestione di tali attività.

La suite dei servizi software cloud, mobile e marketing permette a compagnie di ogni dimensione di facilmente stimare attività, pianificare ed dirigere operatori mobili con facilità. Di seguito si elencano i principali correlati all'attività di stage:

* *WorkWave Service*: è un servizio software che consente ti velocemente schedulare rotte efficienti, visionare la produttività in tempo reale degli operatori, visualizzare stime e gestire pagamenti;

* **WorkWave Route Manager**: è un set di servizi per la gestione dei veicoli al fine di migliorare l'efficienza e la scalabilità attraverso pianificazione dinamica e miglioramenti intelligenti delle rotte. L'algoritmo proprietario del Route Manager garantisce che le migliori rotte per gli operatori siano usate per salvare tempo, costi e migliorare la soddisfazione dei clienti;

* **WorkWave GPS**: fornisce un'intuitiva panoramica dei propri veicoli e delle propie risorse con un potente servizio GPS che cattura le azioni dell'operatore e le posizioni in tempo reale dei veicoli. Permette inoltre di migliorare la sicurezza dei guidatori, segnalare comportamenti errati e riportare incidenti per velocità, frenata improvvisa, curve o altro.

1.3 Route Manager

Figura 1.2: Logo WorkWave Route Manager

WorkWave Italy è la sede italiana di WorkWave dove sono concentrati gli sviluppi dell'algoritmo di routing e del Route Manager nella nuova versione denominata Unified UI, contesto di sviluppo delle attività dello stage descritte in questo documento. WorkWave Route Manager automatizza la pianificazione delle rotte per migliorarne l'efficienza e la comunicazione tra l'amministrazione e i guidatori dei veicoli, completamente customizzabile tramite le sue API.

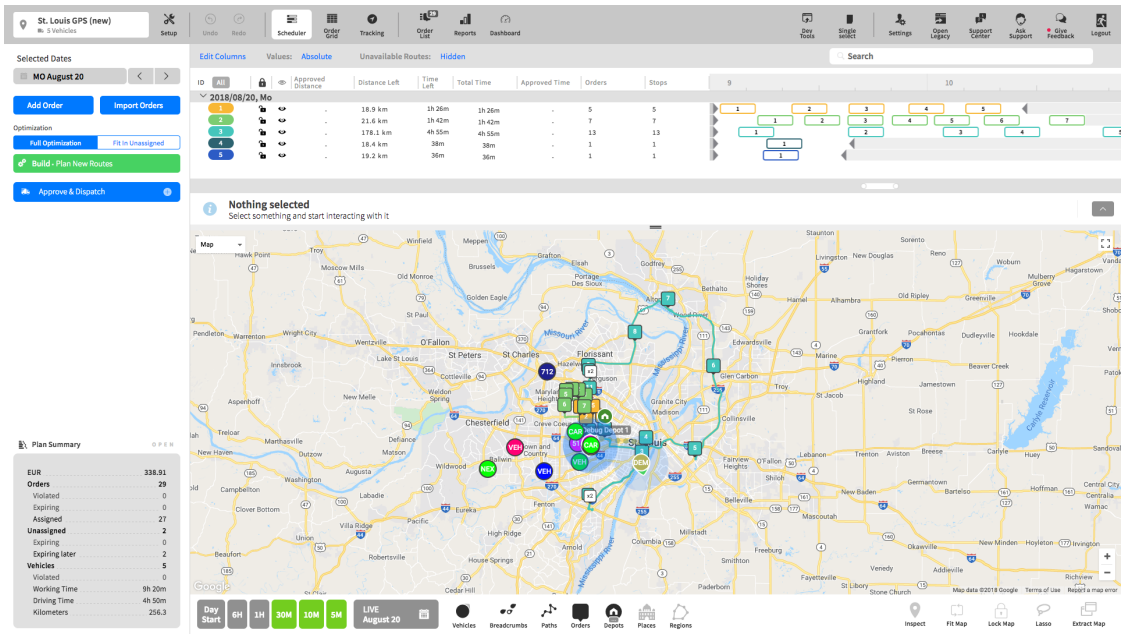


Figura 1.3: WorkWave Route Manager - Unified UI

In particolare, per quanto riguarda Routing & Scheduling, rende possibile navigare attraverso le richieste dei clienti per gli orari di ricezione, schedulare le attività dei guidatori giornalmente ed eseguire report sulle performance. Attraverso le impostazioni, il software fornisce rotte ottimali in base ai propri vincoli stradali.

È inoltre possibile fare aggiustamenti manuali alle rotte via drag&drop, approvare i piani e mandarli in esecuzione agli operatori sui veicoli. Oltre a ciò consente di visualizzare istantaneamente gli effetti sul numero di ordini possibili per i veicoli disponibili, il tempo stimato di completamento delle attività e comparare il costo per miglio.

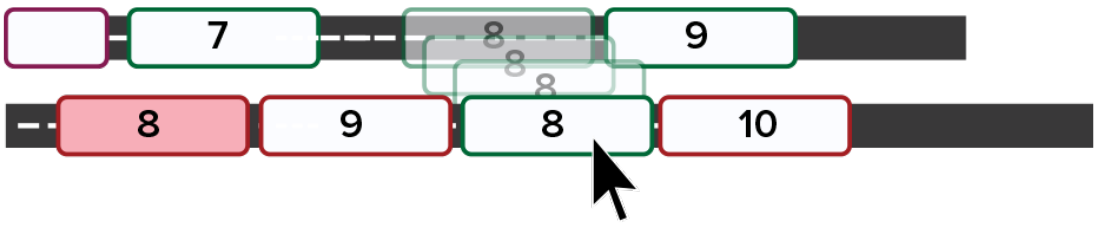


Figura 1.4: WorkWave Route Manager - Scheduler degli ordini

Capitolo 2

Lo stage

2.1 Presentazione del progetto Stargate

Il progetto dello stage consiste nella realizzazione di una libreria TypeScript per applicazioni web, che permetta di aprire qualsiasi componente di User Interface (UI) in una nuova pagina mantenendo lo stato sincronizzato con l'applicazione padre. Tale libreria è denominata **Stargate**, ispirandosi all'omonima serie televisiva di fantascienza incentrata su portali spaziali nelle diverse galassie. Idealmente la libreria dovrebbe funzionare per qualsiasi applicazione web scritta in JavaScript, tuttavia il requisito obbligatorio minimale è che si integri con librerie React e Redux col minor sforzo possibile.

La prima fase dello stage richiede dunque la realizzazione di un Proof of Concept che verifichi in vitreo la possibilità di spostare un componente UI React dalla finestra principale (*padre*) verso una nuova *figlia*, mantendo lo stato consistente. Tali componenti UI sono identificati col nome di *widget*, qualora siano mostrati su una pagina separata.
Un widget deve continuare a visualizzare i dati provenienti dal padre e, se questi si aggiornano, anche il componente nella finestra figlia deve aggiornarsi consistemente. Viceversa le interazioni utente nel componente devono essere propagate alla pagina principale. In sostanza, il componente deve esibire lo stesso comportamento di quando si trovi nell'applicazione principale, sebbene fisicamente si trovi su una diversa pagina del browser.
Infine non vi devono essere vincoli sul numero di componenti aperte in contemporanea su pagine diverse, sia che siano componenti di tipologia diversa sia che siano lo stesso componente istanziato molteplici volte ma indipendenti tra loro.

In secondo luogo, è necessario integrare *Stargate* all'interno del prodotto *Route Manager* implementando la possibilità di estrarre la mappa Google Maps su una nuova pagina. L'obiettivo è permettere all'utente di visualizzare le informazioni sulla mappa secondo molteplici prospettive a sua discrezione, ad esempio mostrandolo tutti i veicoli su una pagina, solo una singola rotta real-time su un'altra. Inoltre permette di usufuire della mappa su schermi multipli, funzionalità fortemente desiderata in quanto la mole delle informazioni a schermo è elevata e l'applicazione

principale mostra difficoltà nel visualizzarle tutte in una sola pagina web.

Lo sviluppo di *Stargate* dovrà dunque avere le seguenti caratteristiche:

* Supporto multi-finestra di componenti React;
* Supporto per grandi moli di dati, ad esempio geospaziali, in continuo aggiornamento;
* Possibilità di eseguire il componente in una nuova pagina che risieda su un processo separato del sistema operativo, affinché alleggerisca il carico computazione dell'applicazione principale. In particolare questa caratteristica è utile per evitare che i calcoli geospaziali vengano eseguiti dal processo padre;
* Supporto a molteplici finestre in contemporanea, tutte sincronizzate rispetto all'applicazione padre;
* Supporto obbligatorio unicamente per i browser moderni Google Chrome e Firefox. Gli utenti che utilizzino browser non compatibili con *Stargate*, potranno usuguire della normale esperienza utente ma senza la possibilità di aprire nuove finestre;
* Supporto a multi-sessione. Qualora l'utente apra due istanze dell'applicazione padre ed in ognuna crei una nuova finestra per un componente, ciascuno di questi deve essere sincronizzato col rispettivo padre e non deve creare conflitti con l'altra applicazione principale.

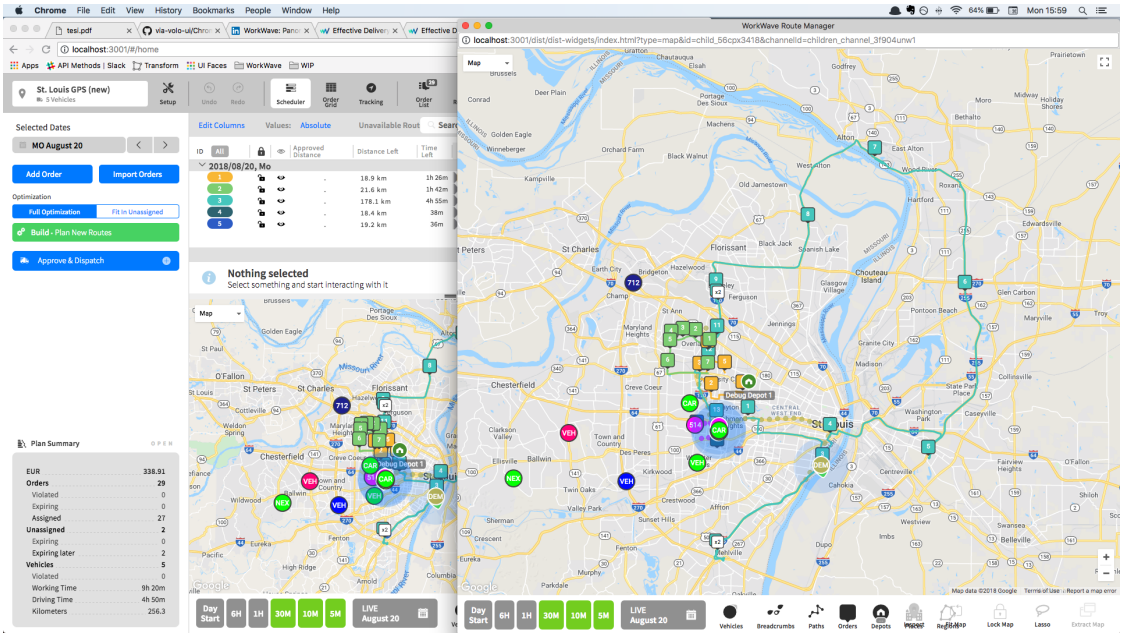


Figura 2.1: WorkWave Route Manager con Google Maps in una nuova finestra attraverso Stargate

2.2 Obiettivi

Si farà riferimento agli obiettivi secondo le seguenti notazioni:

* **O** per i requisiti obbligatori, vincolanti in quanto obiettivo primario richiesto dal committente;

* **D** per i requisiti desiderabili, non vincolanti o strettamente necessari, ma dal riconoscibile valore aggiunto;

* **F** per i requisiti facoltativi, rappresentanti valore aggiunto non strettamente competitivo.

Le sigle precedentemente indicate saranno seguite da una coppia sequenziale di numeri, identificativo del requisito. Si quindi prevede lo svolgimento dei seguenti obiettivi:

ID	Descrizione
Obbligatori	
O01	Supporto multi-finestra di componenti React
O02	Supporto per grandi moli di dati in continuo aggiornamento, dell'ordine di alcuni MByte
O03	Supporto a molteplici finestre in contemporanea, tutte sincronizzate rispetto all'applicazione padre
O04	Supporto per i browser moderni Google Chrome v56 e Firefox v38
O05	Supporto a multi-sessione
O06	Gestione della configurazione di *Route Manager* per supportare i widget
O07	Produzione della documentazione d'uso di *Stargate*
O08	Utilizzo del linguaggio TypeScript v3
Desiderabili	
D01	Possibilità di eseguire il componente in una nuova pagina che risieda su un processo separato del sistema operativo
D02	Supporto prestazionale fino ad almeno 5 tabs simultanee
D03	Supporto componenti JavaScript non React, ad esempio Angular 2
Facoltativi	
O04	Supporto per i browser Internet Explorer v11 e Edge v12

Tabella 2.1: Tabella degli obiettivi

2.3 Pianificazione

In accordo col tutor Matteo Ronchi, l'attività dello stage è stata suddivisa nelle seguenti fasi:

* **Fase 1**: Analisi dello stato dell'arte per la comunicazione cross-page in JavaScript
* **Fase 2**: Realizzazione Proof of Concept in React e Redux
* **Fase 3**: Progettazione per integrazione in *Route Manager* del widget Google Map. Implementazione dell'integrazione ed evoluzione della libreria.
* **Fase 4**: Validazione e stesura documentazione

Non vi è stata necessità di un periodo iniziale di formazione sulle tecnologie TypeScript, React e Redux io quanto già in mio possesso. Mi sono anche trovato subito a mio agio con i processi di sviluppo aziendali, già incontrati da me in altre occasioni.
Essendo inoltre un'attività di Ricerca e Sviluppo, vi è stata una continua interazione col tutor aziendale per la definizione dei successivi step, ma a monte è stata pianificata un'ipotetica suddivisione delle ore nel seguente modo:

Ore	**Descrizione dell'attività**
40	Analisi dello stato dell'arte tecnologico
80	Realizzazione Proof of Concept in React e Redux
32	Progettazione architteturale
104	Integrazione in *Route Manager* del widget Google Map
24	Gestione configurazione per supportare widget
16	Validazione e Collaudo
8	Refactoring prima del rilascio in produzione
8	Stesura documentazione
Totale: 312 ore	

Tabella 2.2: Tabella della suddivisione delle ore

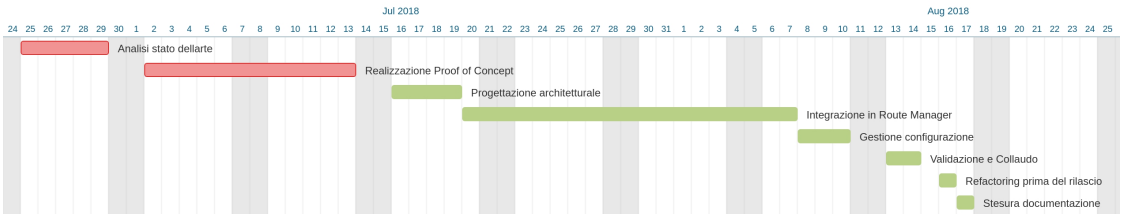


Figura 2.2: Diagramma Gantt della pianificazione

2.4 Analisi preventiva dei rischi

Durante la fase di analisi iniziale sono stati individuati alcuni possibili rischi a cui si potrà andare incontro. Si è quindi proceduto a elaborare delle possibili soluzioni per far fronte a tali rischi.

Descrizione	**Piano di emergenza**	**Rischio**
Difficoltà tecnologica: vi è il rischio che lo stato dell'arte dello sviluppo web non consenta di aprire finestra come processi separati o che non sia possibile effettuare la comunicazione tra pagine diverse.	È importante effettuare un'attenta attività di ricerca iniziale, al fine di comprendere lo stato dell'arte per quanto riguarda la comunicazione cross-page in applicazioni web. In caso di difficoltà nella ricerca della soluzione tecnica ideale, si adotterà quella con miglior compromesso di affidabilità-performance tra quelle disponibili.	Occorrenza: Alta Pericolosità: Alta
Difficoltà di integrazione: vi è il rischio che non sia possibile integrare le tecnologie adottate nel contesto dell'applicazione *Route Manager*, in quanto è già in sviluppo da oltre un anno e non progettata in partenza per essere multi-finestra.	In caso di verifica del rischio, si analizzerà il problema col tutor Matteo Ronchi per trovare la miglior soluzione da adottare in *Stargate* oppure in *Route Manager*	Occorrenza: Alta Pericolosità: Alta

Tabella 2.3: Tabella dell'analisi dei rischi

2.5 Aspettative aziendali

L'azienda WorkWave spera grazie allo stage di poter implementare una funzionalità fortemente desiderata nei loro prodotti software, in particolare in *Route Manager*, con la possibilità di mostrare su pagine diverse qualsiasi componente dell'interfaccia. Ciò apre difatti le porte per una serie di possibili nuove interazioni da parte

dell'utente.

La prima immediata conseguenza è il poter mostrare la mappa Google Maps su un monitor esterno ed in full-screen, estremamente utile sia durante i meeting che negli uffici del clienti del *Route Manager* per tenere sotto costante monitoraggio le rotte ed i veicoli. Il tutto mantendo contemporeaneamente aperta l'applicazione principale su un'altra pagina, ove poter fare le modifiche alle pianificazioni ed altre attività.

Un ulteriore caso d'uso è la possibilità di aprire qualsiasi componente in una nuova pagina, permettendo agli utenti di customizzare i propri flussi di lavoro in modo da tenere sempre aperte alcuni widget durante la navigazione all'interno dell'applicazione.

Infine per l'azienda è anche un'opportunità di conoscere il tirocinante e valutarlo per una possibile futura assunzione.

2.6 Aspettative personali

Ho intrapreso questo stage con l'obiettivo primario di venire in contatto con un ambiente di lavoro focalizzato sul lavoro in team e sulla qualità dei loro prodotti. Ritengo difatti fondamentale apprendere non solo conoscenze tecniche, ma anche quelle sociali e sul modo di lavorare. Grazie a WorkWave ho avuto quindi l'opportunità di conoscere cosa significa lavorare in squadra, fare Pair Programming per pensare insieme alla risoluzione di un problema e condividere le esperienze nella realizzazione ed evoluzione di un prodotto proprio dell'azienda.

È altresì importante professionalmente il contatto con i processi lavorativi di un'azienda che opera a livello world-wide, ma cercando al coltempo di mantenersi agile ed efficiente. È stata difatti concordata la possibilità di partecipare agli stand-up e sono stati spiegati gli strumenti di pianificazione e comunicazione interni.

Ritengo inoltre indispensabile confrontarmi con colleghi molto più esperta e con maggiore esperienza, in particolare il tutor Matteo Ronchi, che ha condiviso pazientemente le ragioni dietro decisioni architteturale apprese grazie alla sua esperienza in progetti passati. Infine è molto istruttivo realizzare il livello delle conoscenze richieste per realizzare ed evolvere un prodotto complesso come *Route Manager*, sia a livello di Design/User Experience che di sviluppo tecnico.

Capitolo 3

Processi e metodologie

Durante il periodo di stage, ho avuto l'opportunità di entrare in contatto con i processi aziendali e diversi strumenti a supporto del mio lavoro, di seguito illustrati.

3.1 Accertamento di Qualità

Il processo di Accertamento di qualità provvede a garantire che il prodotto software sia conforme alle aspettative di qualità desiderate. Nello specifico, durante il mio periodo di stage, sono venuto a contatto con le seguenti pratiche di sviluppo.

3.1.1 Pull Request

Una Pull Request è una proposta di modifica al repository effettuata su Github. Essa è obbligatoria per qualsiasi modifica e deve essere sempre realizzata tramite un git branch dedicato, avente un nome univoco e semantico rispetto alle modifiche proposte.

Lo scopo della Pull Request in WorkWave è favorire la discussione delle modifiche da parte del team e permetterne un'attenta ispezione prima ritenerla valida. Tuttavia essa è anche un'opportunità di apprendimento sia per chi esegue la review che per chi la riceve, in quanto entrambi hanno modo di apprendere diversi approcci allo stesso problema.

È stato inoltre spiegato che essa permette anche, nel lungo termine, di venire a conoscenza di problematiche nel processo di sviluppo e poterle migliorare. Ad esempio la continua segnalazione di norme di sintassi è un indice della necessità di introdurre uno strumento automatico per la formattazione del codice.

3.2 Gestione della configurazione

3.2.1 Versionamento

L'azienda WorkWave organizza il proprio codice sorgente all'interno di diverse repository Git raggruppate sotto l'organizzazione GitHub dell'azienda. In particolare

è stato creato un repository dedicato al versionamento della prima versione della libreria *Stargate* assieme al Proof of Concept. Successivamente il codice sorgente della libreria è stato direttamente integrato nel repository dell'applicazione *Route Manager*.

3.2.2 Ambiente di verifica

Figura 3.1: Jenkins

Il processo di verifica è il più automatizzato possibile tramite tools eseguiti automaticamente con Jenkins https://jenkins.io/. Lo stesso procedimento avviene ad ogni Pull Request proibendone l'accettazione se le verifiche non sono superate.
Inoltre sono presenti script automatici che permettono di rilasciare in ambiente di sviluppo, demo, testing e produzione attraverso l'interfaccia grafica dashboard di Jenkins.

3.2.3 Ambiente di rilascio

Figura 3.2: Amazon Cloudfront

Il rilascio automatico eseguito da Jenkins porta al caricamento dell'applicazione *Route Manager* su Amazon Cloudfront https://aws.amazon.com/it/cloudfront/, un servizio di Content Delivery Network (CDN) che permette di distribuire l'applicazione con latenza minima nelle diversi Paesi del mondo.

3.3 Gestione di Progetto

3.3.1 Stand-up

Il team si incontra quotidianamente per lo stand-up, un incontro informale senza durata prefissata, che permette ai vari membri di allinearsi reciprocamente sullo stato di avanzamento ed eventuali problematiche. In particolare, nel caso di WorkWave tale attività è indispensabile in quanto vi sono alcuni del team che lavorano in remoto o negli Stati Uniti.
Tutti i membri, non solo i programmatori, sono invitati a partecipare e ad esporre su cosa stiano lavorando ed potenziali criticità, permettendo anche di trasmettere maggiore consapevolezza e conoscenza del progetto ai diversi partecipanti.

3.3.2 Ticketing

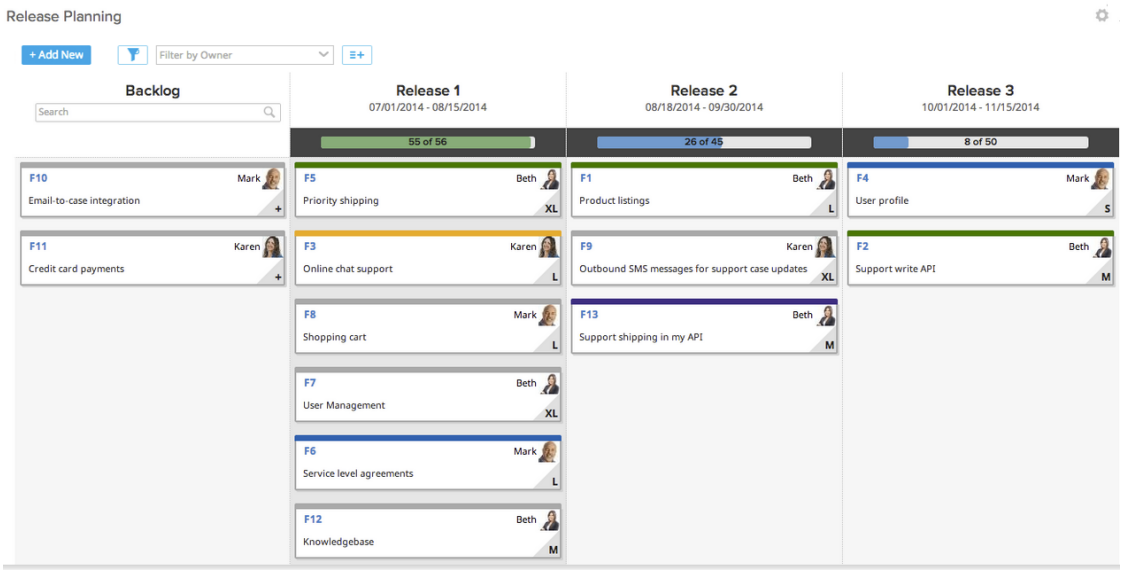


Figura 3.3: CA Agile Central

L'azienda utilizza lo strumento CA Agile Central `https://www.ca.com/it/products/ca-agile-central.html` per la gestione delle attività.

Capitolo 4

Architettura per la comunicazione cross-page

In questo capitolo viene presentata prima l'architettura multi-process dei browser, in particolare di Chrome, per la gestione delle pagine web. Uno degli obiettivi è difatti la possibilità di eseguire i componenti nelle finestre figlie come processi separati, in modo da migliorare e non inficiare il processo dell'applicazione principale.
In seguito si illustra invece lo stato dell'arte delle diverse soluzioni per la comunicazione cross-page in JavaScript tra pagine su processi diversi e l'architettura finale utilizzata dalla libreria *Stargate*.

4.1 Architettura multi-processi

Quando la maggior parte dei browser moderni fu progettata inizialmente, le pagine web erano semplici e avevano poco o nessun codice attivo. Per tale motivo, i browser renderizzano tutte le pagine usando lo stesso processo, al fine di mantenere basso l'utilizzo delle risorse.

Tuttavia, le pagine web odierne sono decisamente più attive a partire da siti statici ma con tanto uso di JavaScript fino a vere e proprie applicazioni web come Gmail. Grosse parti di queste applicazioni girano all'interno del browser, così come le normali applicazioni eseguono in un sistema operativo e, proprio come questi, il browser deve dunque tenere le applicazioni separate tra di loro.

Oltre a ciò, le parti del browser che renderizzano HTML, JavaScript e CSS sono diventate straordinariamente complesse nel corso del tempo. Diventa perciò palese che browser i quali pongono tutto il lavoro in un processo affrontano seri problemi di rubustezza, responsitività e sicurezza.

Se un'applicazione web causasse un crash nel rendering engine, porterebbe la terminazione anche delle altre pagine web aperte. Le applicazioni web inoltre competono reciprocamente per l'uso della CPU ed ognuna di esse è single-thread per design di JavaScript, per cui rischierebbero di diventare non responsive alle interazioni utente. Infine anche la sicurezza è un fattore in rischio poiché una pagina

web potrebbe sfruttare vulnerabilità del browser per accedere a dati delle altre pagine nello stesso processo.

4.1.1 Cosa fa ogni processo?

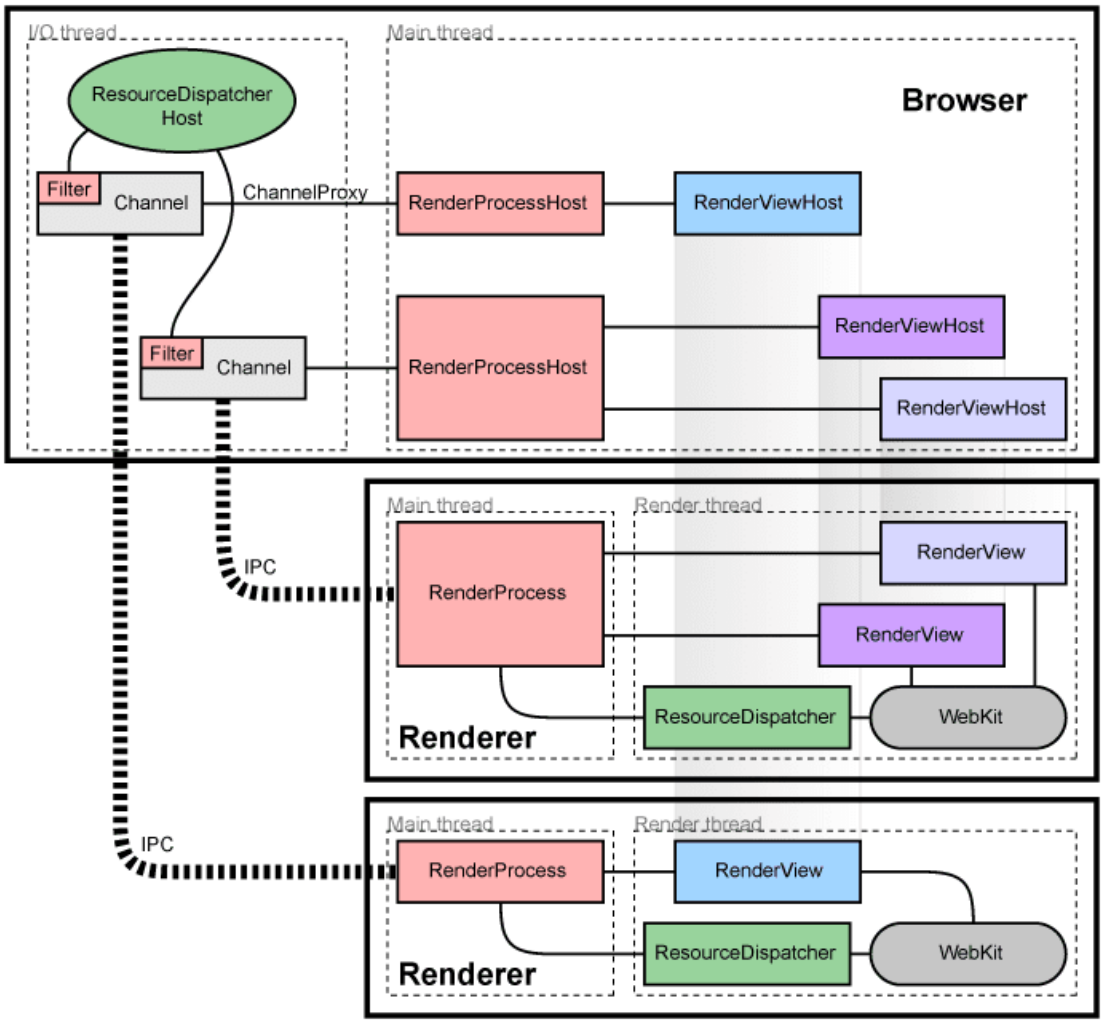


Figura 4.1: Architettura multi-process in Chrome

Il browser crea tre differenti tipi di processi: browser, renderer ed estensioni.

* **Browser**: esiste un unico processo browser, il quale gestisce i tab, le finestre e il browser stesso. Gestisce inoltre tutti i collegamenti delle pagine con il file system, la rete, input utente etc. ma non esegue alcun contenuto delle pagine;
* **Renderer**: il processo browser crea molteplici processi renderer, ognuno responsabile per la visualizzazione di una pagina web. I processi renderer contengono la complessa logica per la gestione di HTML, CSS, JavaScript, immagini e così via. Google Chrome, Safari ed altri utilizzano un rendering engine basato sul progetto open-source WebKit, mentre Firefox ed Edge hanno il proprio;

* **Estensioni**: il processo browser crea anche un processo per ogni estensione

4.1.2 Strategie multi-process

Una volta che il browser ha creato il processo omonimo, crea un processo renderer per ogni istanza di pagina web visitata dall'utente. Può essere pensato come un processo separato per ogni tab del browser, ma con l'eccezione di consentire a due tab di convidere lo stesso processo qualora siano collegati tra di loro e mostrino lo stesso sito.

Per esempio, se un tab ne apre un altro usando JavaScript o se viene aperto un link verso lo stesso sito in un nuovo tab, questi condivideranno lo stesso processo renderer. Si permette così ai tab correlati di comunicare via JavaScript e condividere la cache. Al contrario, se viene aperta una pagina di un sito diverso verrò riservato un nuovo processo.

Per essere precisi, si definisce un "sito" come un dominio registrato (ad esempio google.come o bbc.co.uk) e racchiude anche i sotto-domini (mail.google.com) e le porte (google.com:8080). Un' "istanza di sito" è invece un'insieme di pagine collegate provenienti dallo stesso sito. Due pagine sono considerate connesse se vi sono riferimenti reciproci in codice JavaScript, ad esempio una apre la seconda programmaticamente. Mentre se l'utente digita manualmente lo stesso indirizzo in due tab diverse, vengono considerate due istanze diverse con processi distinti.

Di seguito si illustrano nel dettaglio le diverse strategie multi-processi adottati dai browser moderni.

Process-per-site-instance

Normalmente i browser usano una strategia "Process-per-site-instance", ovvero lo stesso sito aperto in tab diversi con riferimenti reciproci saranno renderizzati dallo stesso processo. A volte è difatti necessario o desiderabile condividere il processo, quando per esempio un'applicazione web apre una nuova finestra con cui si aspetta di comunicare in maniera sincrona.

In generale invece, ogni nuova finestra o tab che non siano lo stesso sito possiede un nuovo processo.

Process-per-site

Raggruppa tutte le pagine dello stesso sito nello stesso processo, indipendentemente dalla presenza di riferimenti reciproci. Questa strategia è basata esclusivamente sul dominio del contenuto e non sulle relazioni tra le tab. Di conseguenza può risultare in processi molto onerosi.

Process-per-tab

Esiste anche una strategie più semplice che dedica un processo renderer per ogni gruppo di tab. Per ovvi motivi è estremamente inefficiente.

Capitolo 5

Progettazione e codifica

Breve introduzione al capitolo

5.1 Tecnologie e strumenti

Di seguito viene data una panoramica delle tecnologie e strumenti utilizzati.

Tecnologia 1

Descrizione Tecnologia 1.

Tecnologia 2

Descrizione Tecnologia 2

5.2 Ciclo di vita del software

5.3 Progettazione

Namespace 1

Descrizione namespace 1.

Classe 1: Descrizione classe 1

Classe 2: Descrizione classe 2

5.4 Design Pattern utilizzati

5.5 Codifica

Capitolo 6

Verifica e validazione

Capitolo 7

Conclusioni

7.1 Consuntivo finale

7.2 Raggiungimento degli obiettivi

7.3 Conoscenze acquisite

7.4 Valutazione personale

Appendice A

Appendice A

Citazione

Autore della citazione

Bibliografia

* Multi-process architecture https://blog.chromium.org/2008/09/multi-process-architect
html