ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Triennale

Titolo

Relatore
prof. Paolo Bellavista

Laureando
Davide Crociati

Ottobre 2023

A voi

Sommario

Piacere, sommario.

Indice

Elenco delle figure

Elenco delle tabelle

Elenco dei listati

Capitolo 1

Introduzione

1.1 Contestualizzazione

1.1.1 Evoluzione delle applicazioni web

Inizialmente le applicazioni web erano costituite da semplici pagine statiche contenenti testo e immagini. Col passare degli anni, grazie all'adozione di JavaScript e di librerie e framework correlati, hanno progressivamente acquisito un carattere più dinamico, con l'introduzione di livelli crescenti di interattività. Un cambiamento significativo in tal senso, è avvenuto con l'avvento delle **Single Page Application (SPA)** e di **AJAX**. Tale combinazione, ha infatti introdotto un nuovo paradigma di sviluppo, in cui l'intera appplicazione viene caricata una sola volta e le successive interazioni con l'utente avvengono grazie al caricamento dinamico di contenuti e dati provenienti da un web server, eliminando così l'attesa nel caricamento di una nuova pagina e favorendo l'esperienza d'uso.
Parallelamente, la complessità delle funzionalità offerte è cresciuta in modo esponenziale, spaziando da applicazioni di grafica 3D, a simulatori, ad applicazioni di modifica di documenti, immagini e video. Oggi, è sempre più comune incontrare siti web in grado di gestire complesse operazioni in tempi rapidi, assicurando così quell'interattività alla quale ormai siamo abituati. Tuttavia questo progresso è stato accompagnato da un aumento del numero di richieste effettuate in rete e dall'utilizzo intensivo di risorse computazionali, sia lato cliente, che lato servitore.

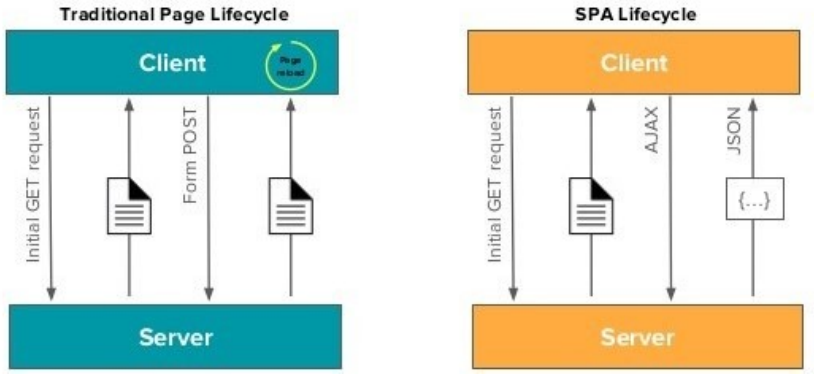

Figura 1.1: Differenza nel tipo di richieste tra SPA e MPA.

1.1.2 Importanza dell'ottimizzazione

Ad oggi, l'ottimizzazione delle prestazioni è quindi diventata un aspetto cruciale nello sviluppo di applicazioni web. Gli utenti si aspettano interazioni con bassa latenza, caricamenti rapidi e risposte immediate. Questa esigenza mette in risalto l'importanza di bilanciare l'aggiunta di funzionalità sofisticate, con l'offerta di una *User Experience* ottimale.

I tempi di caricamento prolungati possono portare a un alto tasso di abbandono delle pagine, riducendo l'opportunità di coinvolgere nuovi utenti. Inoltre, con l'aumentare dell'utilizzo di dispositivi mobili e di conseguenza, di connessioni instabili, l'ottimizzazione diventa ancor più critica per assicurare un'esperienza coerente su diverse piattaforme e condizioni di rete. Tutto ciò non riguarda solo il lato client, ma coinvolge anche il lato server. Un carico eccessivo sui server può influire negativamente sulla scalabilità, causando ritardi nelle risposte e possibili interruzioni del servizio. L'ottimizzazione deve quindi coinvolgere tutti gli aspetti dell'architettura delle applicazioni web.

Nell'implementare ottimizzazioni, sono nati varie soluzioni interessanti. Ad esempio, per gestire task che svolgono molte operazioni di Input/Output si è distinto **Node.js**, mentre per quanto riguardo l'esecuzione di attività che sfruttano molto la CPU è emerso **WebAssembly(Wasm)**, in sinergia con l'interfaccia di sistema **WebAssembly System Interface (WASI)**.

1.2 Motivazioni e Obiettivi

La crescente complessità delle applicazioni web e l'esigenza di offrire agli utenti esperienze interattive sempre più coinvolgenti hanno portato l'ambito dello sviluppo web a una svolta significativa. Le aspettative degli utenti si sono evolute verso applicazioni che offrano prestazioni reattive, interattività immediata e funzionalità avanzate.
È proprio questo insieme di aspettative a essere alla base delle motivazioni che hanno guidato la scelta del tema di questa tesi di laurea. In particolare, la presente ricerca si propone di esplorare in profondità il complesso equilibrio tra l'implementazione di funzionalità sofisticate (principalmente **CPU-Intensive**) e l'ottimizzazione delle prestazioni all'interno delle applicazioni web. Il fulcro di questa indagine sarà una comparazione dettagliata tra due approcci di sviluppo distinti per un'applicazione che consenta **l'elaborazione server-side di immagini** caricate da un utente. Non verrà esplorato, in modo approfondito il campo dell'elaborazione digitale di immagini, ma ci si limiterà all'implementazione di funzionalità usate spesso da utenti comuni, come ad esempio ridimensionamento, rotazione, aumento/diminuzione di luminosità e contrasto e altre che verranno specificate nel capitolo 3.
Tale tipologia di applicazione, si sposa bene per lo scopo finale della tesi: valutare come due aprrocci (e due linguaggi) piuttosto differenti, ma sempre più diffusi al giorno d'oggi, risolvano il problema di un'applicazione web che svolga operazioni dall'alto costo computazionale. Le due modalità di sviluppo in questione riguarderanno l'utilizzo delle tecnologie JavaScript e WebAssembly server-side e quindi rispettivamente del runtime environment Node.js e del linguaggio di programmazione Rust in combinazione con WebAssembly System Interface.
La scelta di Node.js deriva dalla sua crescente popolarità dovuta all'utilizzo del linguaggio JavaScript, dall'approccio asincrono nella gestione delle richieste e dalla sua ottima scalabilità per applicazioni fortemente File-System-Intensive. Per quanto riguarda la seconda tecnologia principale si è optato per Rust in quanto consente, sia la scrittura di moduli che poi potranno successivamente essere compilati in WebAssembly, sia il loro utilizzo efficiente all'interno del codice, mantenendo in questo modo un'ottima coerenza e risultando, teoricamente, una buona scelta per lo sviluppo di applicazioni computazionalmente complesse.
Si intende esplorare le opportunità offerte dalle precedenti tecnologie nell'ottica di

un'ottimizzazione delle prestazioni. In questo contesto l'obiettivo sarà quello di comprendere i benefici specifici di ciascun approccio, individuando le circostanze in cui uno dei due possa risultare vantaggioso in termini di efficienza computazionale e reattività. Inoltre si vorrà sottolineare l'importanza, sin dalla fase di progettazione, di un'attenta analisi per determinare se un'applicazione sia maggiormente orientata al calcolo intensivo o all'uso massiccio del File System. Per raggiungere in maniera efficace e misurabile gli obiettivi, si sfrutterà un'analisi approfondita delle performance e delle prestazioni delle due applicazioni sviluppate.

Figura 1.2: Il campo dell'elaborazione digitale di immagini è molto ampio e sarebbe possibile integrare anche operazioni avanzate (edge detection, pattern recognition etc.), che richiederebbero risorse computazionali molto elevate. Per lo scopo di questa tesi ci si limiterà a elaborazioni più semplici, ma in ogni caso rilevanti sufficienti per mettere alla prova la CPU.

1.2.1 Analisi Comparativa delle Tecnologie

Un elemento iniziale di questa ricerca sarà un analisi dettagliata delle tecnologie prese in esame. Inizialmente, nel capitolo 2 verrà svolta un'analisi comparativa delle due tecnologie, presentando i vantaggi e gli svantaggi teorici di entrambe. Sarà infatti fondamentale comprendere come ciascuna affronti la complessità legata ad operazioni I/O-intensive e CPU-intensive, per valutare nel modo più opportuno i risultati ottenuti in fase di test e benchmark dell'applicazione sviluppata.
Si procederà poi ad illustrare in modo approfondito il funzionamento delle API sfruttate, soprattutto per quanto riguardo WebAssembly e WASI.

Figura 1.3: Rust e Wasm

Figura 1.4: Node.js

1.2.2 Valutazione dell'Impatto di Wasm

1.3 Trend di evoluzione del web

1.3.1 Paradigmi di Sviluppo Moderni

1.3.2 Esperienze Utente Migliorate ?

1.4 Struttura della tesi

This is a reference to a chapter 1. This is a reference to a figure 1.5. This is a reference to some code 1.1. This is a citation [2].

```
1 import suca
2
3 print "hello␣world!"
```

Listato 1.1: I directly included a portion of a file

```
1 public void prepare(AClass foo) {
2         AnotherClass bar = new AnotherClass(foo)
3 }
```

Listato 1.2: Some code in another language than the default one

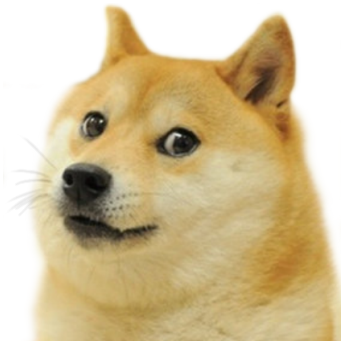

Figura 1.5: This is not a figure. It's a caption.

Capitolo 2

Capitolo 2

2.1 Tecnologie

Capitolo 3

Capitolo 3

3.1 Prototipo

Ringraziamenti

Grazie a Grazia, Graziella e la sorella.

Bibliografia

[1] Carlino Cane. *No one cited me*. A cura di Anche Leggoo. first edition. O'Reilly, 2009. URL: http://bar.com/.

[2] Darth Vader e Neo Cortex. «Hell yeah! I'm a super important paper!» In: *Stampo Solo* 51 (2008), pp. 107–113. DOI: 10.1145/1327452.1327492. URL: http://www.foo.com.