Prefazione

Rispetto ai miei amici, ho scoperto un po' in ritardo il mondo della programmazione. È capitato per caso al liceo, perché avevo bisogno di aggiungere un corso al curriculum e il consulente di orientamento della scuola me ne ha suggerito uno: "Computer Ed". Mi aspettavo di scoprire come funzionano i computer, ma, con mia grande sorpresa, il corso riguardava la programmazione. Non ci è voluto molto prima che l'argomento mi conquistasse e così ho cambiato il mio indirizzo professionale: dall'architettura di edifici all'architettura software.

Nel 2001 sono stato assunto in Dovico Software e mi sono occupato della manutenzione e dell'estensione della loro applicazione client/server, scritta in C++. Il vento del cambiamento stava soffiando forte: nel 2004 Dovico decise di adottare un modello *software-as-a-service* e così sono passato a lavorare all'applicativo web. Ho comunque continuato a svolgere la manutenzione delle applicazioni C++, ma il mio obiettivo principale era ormai diventato lo sviluppo web in C# e JavaScript. Al giorno d'oggi, mi occupo ancora di sviluppo web, ma la mia attività si è spostata più sul lato dell'architettura software: creazione di API, utilizzo di database ed esplorazione di nuove tecnologie.

Mi piace poter restituire alla community degli sviluppatori quello che ho ricevuto, attraverso blog e conferenze. A settembre 2017 mi è stato chiesto se fossi interessato a tenere una presentazione per un gruppo locale di utenti. Mentre cercavo idee sugli argomenti da trattare, mi sono imbattuto in un articolo di PSPDFKit che parlava di una tecnologia chiamata WebAssembly (`https://pspdfkit.com/blog/2017/webassembly-a-new-hope`).

Avevo letto della tecnologia Native Client (PNaCI) di Google: codice compilato C o C++ che poteva essere eseguito nel browser web Chrome a velocità quasi native. Ho anche letto della tecnologia asm.js di Mozilla, che permetteva di compilare codice C o C++ in un sottoinsieme del linguaggio JavaScript e farlo funzionare molto velocemente nei browser che ne offrivano il supporto. Nei browser che non offrivano il supporto di asm.js, il codice funzionerebbe comunque, ma a velocità standard, perché comunque è solo codice JavaScript. Sia quel che sia, questa è stata la prima volta che ho sentito parlare di WebAssembly.

WebAssembly adotta i miglioramenti offerti da asm.js e cerca di affrontarne le carenze. Non solo permette di scrivere il codice in una certa varietà di linguaggi e compilarlo in qualcosa che funzioni in modo sicuro in un browser, ma è già disponibile per tutti i principali browser, sia desktop sia mobili! Di più: è disponibile anche al di fuori di

un browser, grazie a Node.js. Sono rimasto sbalordito dal suo potenziale e da allora ho dedicato ogni momento libero a cercare di approfondire questa tecnologia e a divulgarla. Verso la fine del 2017, i miei post sul blog sono stati notati da Manning Publications, e sono stato contattato per vedere se fossi interessato a scrivere un libro su WebAssembly. All'inizio, il libro copriva più linguaggi e mostrava come impiegare la tecnologia dal punto di vista dello sviluppatore sia backend sia frontend. Fin dalla prima revisione, tuttavia, era risultato evidente che il libro era troppo dispersivo, quindi abbiamo deciso che sarebbe stato meglio restringere l'ambito al solo linguaggio di programmazione C++ e concentrarci maggiormente sugli sviluppatori backend.

La community e i gruppi di lavoro WebAssembly non sono rimasti fermi mentre io scrivevo questo libro. In effetti, attualmente sono in corso di sviluppo diversi progressi. Di recente, è nata anche la possibilità di utilizzare moduli WebAssembly multithread anche nella versione desktop del browser Google Chrome, e questo senza nemmeno la necessità di attivare un flag. WebAssembly ha tutte le potenzialità per aiutare a portare lo sviluppo web a livelli completamente nuovi e io sono entusiasta di poter partecipare a questo sviluppo.