Alma Mater Studiorum Universita di Bologna
Dipartimento di Informatica - Scienza e Ingegneria (DISI)
Laurea in Ingegneria Informatica

Esecuzione Efficiente di Web App Rust su Piattaforma Web Assembly

Relatore: Paolo Bellavista

Candidato: Davide Crociati

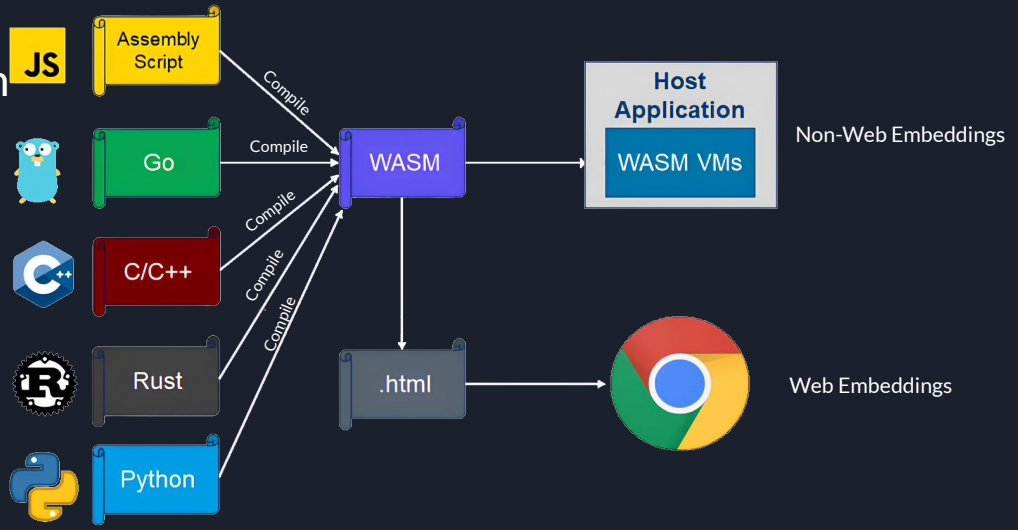
21 Ottobre 2023

Obiettivi della tesi

- Studio di WebAssembly e WebAssembly System Interface nel contesto di Web Application CPU-Intensive
- Integrazione di WebAssembly server-side in applicazione Rust
- Proof of concept mediante lo sviluppo di un prototipo che sfrutta tecniche di Digital Image Processing
- Analisi comparativa con la medesima app in Node.js

WebAssembly

- Formato binario nato come compilation target per il Web
- Efficiente
- Portabile
- Sicuro: esecuzione in sandbox
- Inizialmente pensato per migliorare le prestazioni di Javascript all'interno del Web

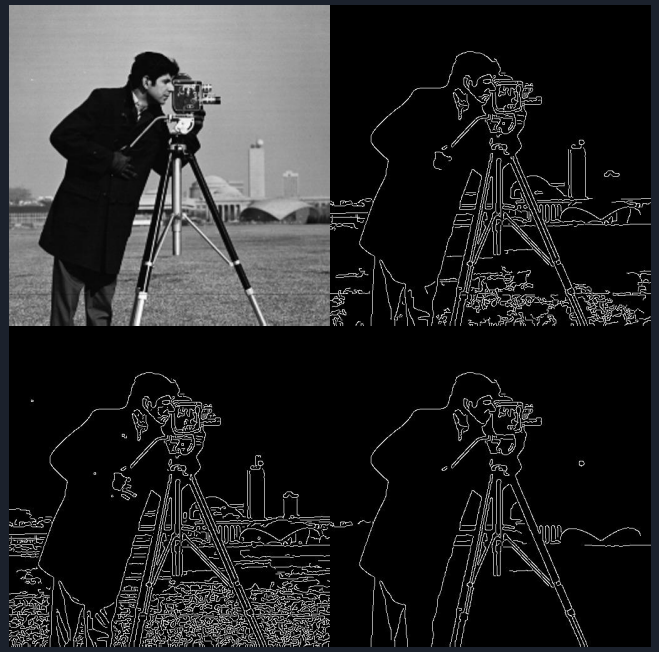


WebAssembly System Interface

- Accesso al file system e a risorse di sistema (syscall)
- Sistema di sicurezza basato su capabilities
- Portabilità
- Esecuzione di WebAssembly al di fuori del browser
- Nuova piattaforma per la programmazione distribuita

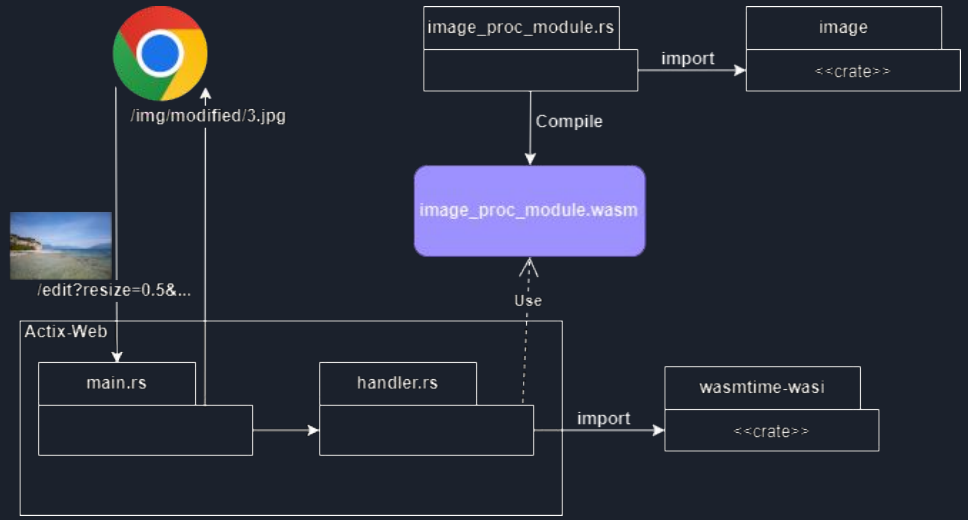
Proof of concept

- Applicazione Web con architettura Client Server
- Funzionalità di:
  - Ridimensionamento
  - Rotazione di 90°
  - Ribaltamento in orizzontale
  - Conversione in bianco e nero
  - Regolazione del contrasto
  - Modifica della luminosità
- Secondo prototipo in Javascript



Proof of concept: Rust + WebAssembly

- Back-end sviluppato in Rust tramite il framework Actix-Web
- Utilizzo di modulo WebAssembly per operazioni CPU-Intensive (elaborazione di immagini)
- Integrazione di WASI tramite il runtime environment Wasmtime



Analisi prestazioni: Latenza

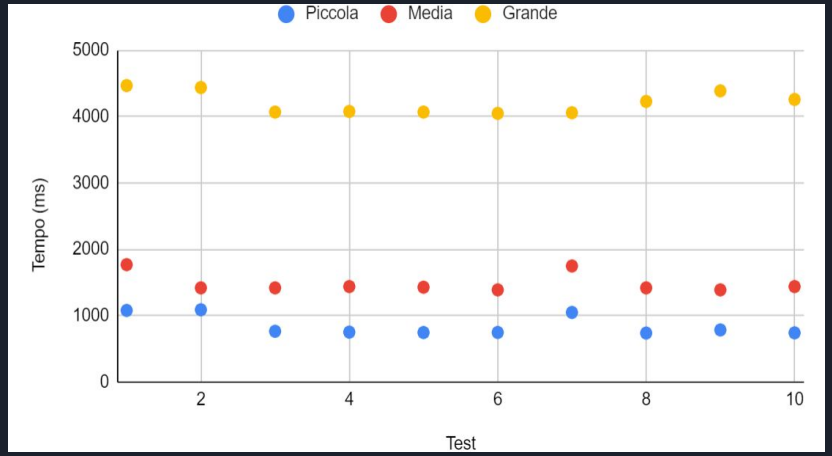
Valore medio latenza su 10 test:

	Immagine 0.82 Mpx	Immagine 5.48 Mpx	Immagine 20.7 Mpx
Rust+WASI	850 ms	1487 ms	4212 ms
Node.js	704 ms	3081 ms	11335 ms

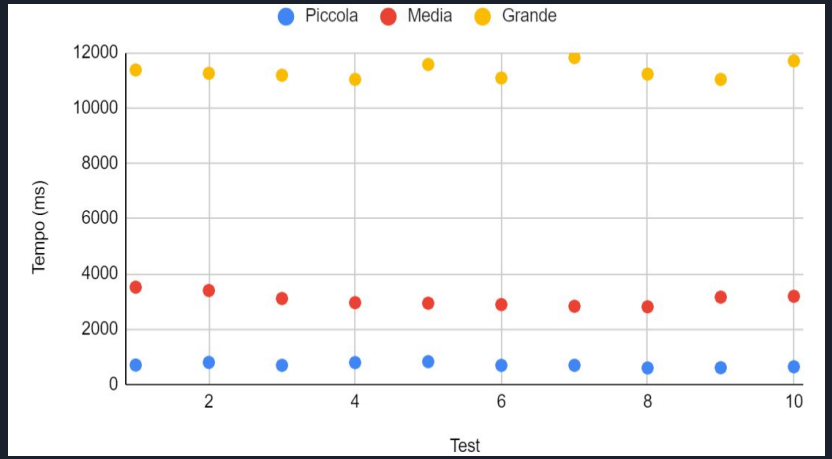
Deviazione standard test Rust: ≈ 150 ms

Deviazione standard test Node.js: ≈ 200 ms

Rust+WASI:



Node.js



Analisi prestazioni: Utilizzo CPU

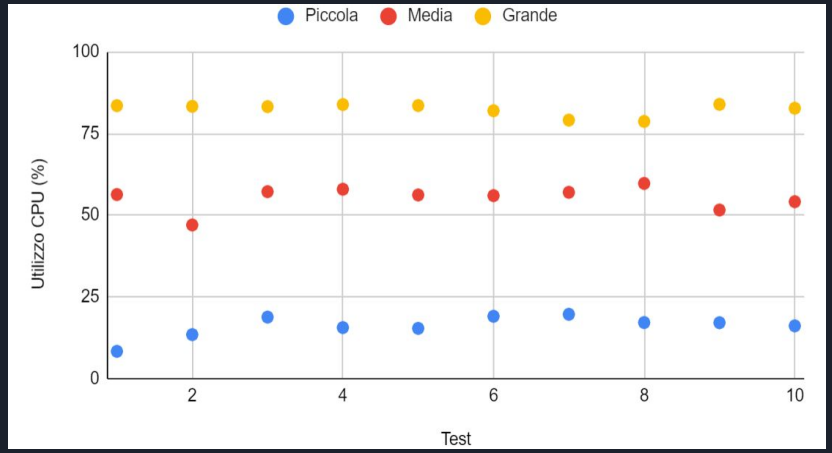
Valore medio utilizzo CPU su 10 test:

	Immagine 0.82 Mpx	Immagine 5.48 Mpx	Immagine 20.7 Mpx
Rust+WASI	16.09 %	55.35 %	82.44 %
Node.js	20.87 %	16.33 %	18.96 %

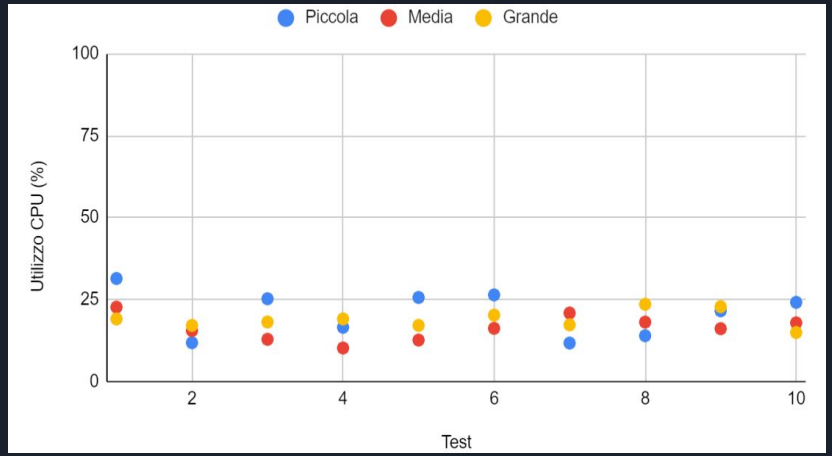
Deviazione standard test Rust: ≈ 2.5 %

Deviazione standard test Node.js: ≈ 4 %

Rust+WASI:



Node.js



Analisi prestazioni: Consumo di memoria

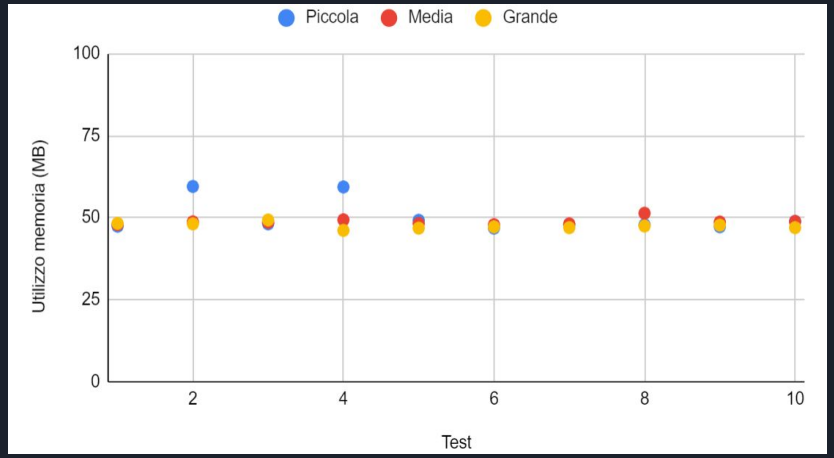
Valore medio consumo di memoria su 10 test:

	Immagine 0.82 Mpx	Immagine 5.48 Mpx	Immagine 20.7 Mpx
Rust+WASI	50.24 MB	48.79 MB	47.51 MB
Node.js	80.51 MB	79.80 MB	79.92 MB

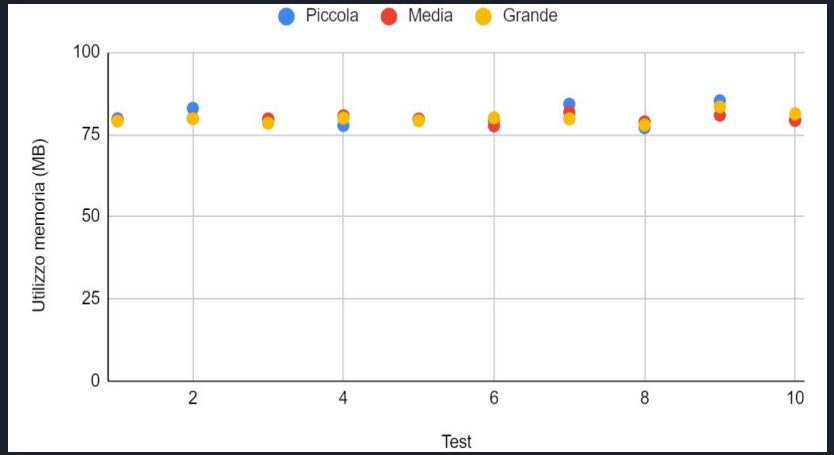
Deviazione standard test Rust: ≈ 3 MB

Deviazione standard test Node.js: ≈ 2 MB

Rust+WASI:



Node.js



Conclusioni

- WebAssembly offre ottime prestazioni per attività CPU-Intensive
- Portabilità di moduli Wasm
- Sicurezza
- WASI in costante evoluzione

Grazie per l’attenzione!