

1-Pagina Samenvatting — Interoperabiliteitswhitepaper

1-Pagina Samenvatting - Interoperabiliteitswhitepaper

Doel

Deze samenvatting beschrijft de kern van een uniforme interoperabiliteitsarchitectuur waarin alle talen - COBOL, Fortran, C, ML64, C#, Python en toekomstige talen - communiceren via één stabiele C-ABI DLL-laag. Dit model elimineert taal-specifieke afhankelijkheden en creëert een toekomstbestendige, onderhoudsarme en foutbestendige multi-language omgeving.

Kernprobleem

Heterogene systemen kampen met:

- incompatibele calling conventions
- compiler-specifieke symbol-mangling
- verschillende memory-modellen
- inconsistente struct-layouts
- onvoorspelbare error-afhandeling
- hoge onderhoudskosten bij taal-naar-taal koppelingen

Zonder centrale architectuur ontstaat een fragiel systeem dat moeilijk te testen, debuggen en uitbreiden is.

Oplossing: C-ABI als uniforme taalgrens

Alle talen communiceren uitsluitend via een C-ABI-compatibele DLL-laag.

Dit elimineert:

- Fortran-module-afhankelijkheden
- COBOL-runtime-exposure
- taal-specifieke memory-modellen
- ABI-breuken bij compiler-updates

De C-ABI wordt het enige contract tussen alle talen.

Architectuurmodel

```
COBOL →  
→ C-ABI DLL ← Fortran 95  
ML64 →  
→ C# / Python / Rust
```

Eigenschappen:

- expliciete symbol-exports
- stabiele ABI
- geen directe taal-naar-taal koppelingen
- volledige scheiding tussen interface en implementatie

Uniforme technische fundamenten

Calling-conventions

Win64 x64-ABI voor alle talen (RCX/RDX/R8/R9, shadow space, RAX return).

Type-mapping

Gestandaardiseerd voor integers, floats, pointers, structs en booleans.

Memory-ownership

Drie patronen: caller-allocates, callee-allocates, buffer-passing.

Error-model

Uniforme foutcodes, optionele `get_last_error()`, geen exceptions over de ABI.

Exportbeheer

.def-file verplicht, symbolen nooit wijzigen, ABI-versie via `get_abi_version()`.

Voordelen

- drastisch lagere onderhoudskosten
- voorspelbare en stabiele ABI
- veilige interoperabiliteit tussen alle talen
- eenvoudiger testen en debuggen
- toekomstbestendig voor nieuwe talen
- geen compiler-lock-in

Conclusie

De C-ABI-gebaseerde interoperabiliteitsarchitectuur vormt een duurzame ruggengraat voor organisaties die legacy-systemen en moderne technologie willen combineren zonder technische schuld. Het model is eenvoudig, robuust en bewezen effectief in complexe multi-language omgevingen.