Migrating Custom DSL Implementations to a Language Workbench (Tool Demo)

Custom DSL Implementations

An error occurred during the generation of this view:

TYPE ERROR: Expression b has type integer instead of boolean

Context: Océ develops and uses domain-specific languages (DSLs) for model-based development.

Problem: Custom implementations using conventional technologies (XML for syntax, Python for static analysis and code generation) are flexible, but:

- No IDE support
- Hard to implement advanced language features
- Concise syntax is missing
- No traceable error reporting

Solution: Migrate to a language workbench.

Extra requirement: Automate forward and backward migration.

DSL Migration Architecture

Syntactx

Parsers and pretty printers derived from syntax definition, for both the old (XML) syntax and the new (DSL) syntax.

Normalization

The key element of this architecture is the introduction of a normalized AST format with bi-directional transformation to and from each syntax AST.

Static Semantics & Code Generation

Define the rest of the architecture once on the normalized AST.

Forward Migration

For porting existing programs written in the old XML syntax to the new DSL syntax.

Backward Migration

For programs written in the new DSL to be backward compatible with existing tooling still based on the XML syntax.

Modular Language Definition

Inter-language: XML and DSL variants of the languages share syntax and static semantics definition.

Intra-language: DSL re-uses syntax, static semantics, and transformations from IDL.