

DSL Evolution

Dr. Laurence Tratt

<http://tratt.net/laurie/>

King's College London

2007/05/20

Overview

- This introduction.
- 'Experts views'.
- Discussion.

Overview

- We'll assume a broad definition of DSL (textual or modelling).
- Issue: Evolve a DSL (change syntax and / or semantics).
- How do we update the DSL implementation?
- How do we migrate user code?
- What happens if we evolve multiple times?

Programming languages

- What happens when programming languages evolve? The same problem?
- Generally not. Why?
- Is it the first commandment? Thy shalt not break backwards compatibility.
- Or the rich history of designing programming languages?

DSLs vs programming languages

- Programming languages don't vary much, syntactically or semantically.
- DSLs syntax and semantics widely.
- Programming language implementation widely understood.
- DSLs, DSEs, DSMs, DSxx.
- Fundamentally: each DSL wipes the precedent slate clean.
- Over-arching, incompatible programming language changes rare.
- Complete redesign and / or mammoth extension of DSLs common.

DSL implementation

- A key differentiator (excl. DSM?)? Standalone vs integrated.
- Standalone: big, expensive things. Like turning an oil tanker.
- Integrated: small, cheap, hackish. Like sculpting molasses.

Migrating user code

- Traditional programming language: do it yourself.
- Textual DSLs: the same?
- DSMLs: ...?

The experts views

Open discussion