TyDe Systems are Trustworthy Systems

Jan de Muijnck-Hughes (StrathCyber & MSP)

University of Strathclyde Glasgow

Problem: Separate Code & Specifications!

- Run-time testing
- Extensive auditing
- Wide test coverage

Solution: Intrinsic Code & Specifications!

- Compile-time errors
- Easier auditing
- Targeted testing

Developing Type-Driven approaches to make specifications first-class software engineering artefacts.

TyDe Approach

Functional Programming

Describe the structure of code & specifications.

Type Systems

Explore (new) meaning of code & specifications.

Dependent Types

Formal reasoning about, and realisation of, code & specifications.

Research Areas

Retrofitting Languages with New Type Systems

Add more expressive types onto existing code

- Reason about new properties
- Run new static analyses
- No change in language

Highly Assured Compilers for Highly Assured Code

Executable language specifications

- Reason about language design
- Reuse tests from production
- Explore safe new extensions

RFCs as Types; Types as RFCs

Incorporate RFCs within language design

- RFCs for static analysis
- RFCs for code generation
- RFCs for discovery

Engineering with Dependent Types

Codify common engineering idioms

- Investigate Human Factors
- Discover design (anti-) patterns
- Explore problems & solutions

jfdm@discuss.systems jfdm@twitter.com https://tyde.systems Jan.de-Muijnck-Hughes@strath.ac.uk