

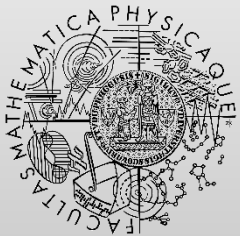
Model-Based Specification in VDM

<http://d3s.mff.cuni.cz>

Department of
Distributed and
Dependable
Systems



Pavel Parízek



FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

Vienna Development Method (VDM)

- Formal specification languages
 - VDM-SL
 - VDM++
- Combination: model-based + algebraic
 - Abstract modeling (data + contracts)
 - Executable subset (prototyping implementation)
- Tools
 - validation, analysis, testing
 - code generation (Java, C++)

- Syntax
 - ASCII text, graphical
- Features
 - Basic types: numeric, character, token, quote
 - Collections: set, sequence, map
 - Type constructors: union, cartesian product, record (composite)
 - Functions (pure, no side effects)
 - Operations (modify global state)

Example

- Management system for public transport
- Key concepts
 - Modules (import, export)
 - Implicit definition of functions/operations
 - Contracts (precondition, postcondition)
 - Explicit definition of functions/operations
 - Prototype implementation (algorithm)
 - Control-flow structures
 - imperative, functional

Proving correctness

Implicit definition

$f(p:T_p)r:T_r$
pre pre- $f(p)$
post post- $f(p,r)$

Explicit definition

$f:T_p \rightarrow T_r$
 $f(p) == \dots$

Proof obligation

forall $p:T_p$ • pre- $f(p) \Rightarrow f(p):T_r$ and post- $f(p, f(p))$

Refinement – another perspective

- Abstract data representation AR
- New concrete data representation CR
- Abstraction function $\alpha: CR \rightarrow AR$
- Proof obligations
 - $\forall a:AR \bullet \exists c:CR \wedge a = \alpha(c)$
 - $\forall c:CR \bullet \text{pre-OpA}(\alpha(c)) \Rightarrow \text{pre-OpC}(c)$
 - $\forall c^{\sim}, c:CR \bullet \text{pre-OpA}(\alpha(c^{\sim})) \wedge \text{post-OpC}(c^{\sim}, c) \Rightarrow \text{post-OpA}(\alpha(c^{\sim}), \alpha(c))$

Case studies

- International conference on Rigorous State Based Methods: ABZ
 - <https://abz2021.uni-ulm.de/>
 - <https://www.southampton.ac.uk/abz2018/participants/programme.page>

- VDMTools

- <http://fmvdm.org/vdmtools/>
- Checks syntax, types, integrity
- Interpreter (debugger)
- Code generation (Java, C++)

- Overture

- <http://overturetool.org/>

Literature

- C.B. Jones. Systematic software development using VDM. Prentice-Hall, 1990
 - <https://www.amazon.com/Systematic-Software-Development-Prentice-hall-International/dp/0138807337>
 - <https://dl.acm.org/citation.cfm?id=94062>