Ten Years of OCL@Dresden from a Personal Viewpoint

Heinrich Hußmann
LMU München

Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

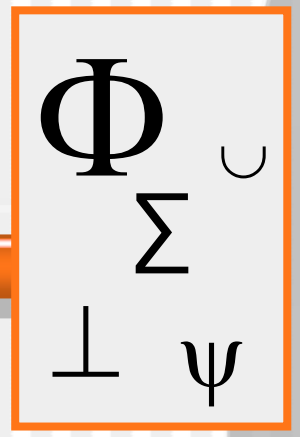
Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

Personal Background 1999

- Industrial Experience
- Academic Background:
  Formal Specification Languages
- Specifically:
  - Algebraic Specifications (SPECTRUM)
  - Experimental Tool Development (RAP)

Formal Methods



- Only few standard languages
- High diversity and incompatibility of approaches
- Practicians complain about missing tool support
- Research-oriented tools (e.g. theorem provers)

Experimental Tool Development

- Clear orientation for research work ("spine" to which attach different work)
- Provides success experiences
- Feedback, case studies possible
- Good basis for establishing community
  - Research
  - Application

Wanted !

- Standardized language
- In need for tool support
- Relevant for practice

Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

History of OCL

- OCL was developed at IBM in 1995
  - Original purpose:
    “business engineering language”,
  - Inspired by ”Syntropy” (Cook/Daniels)
- 1997 IBM and ObjecTime (Selic)
  - Response to OMG request for proposals on OOA&D (UML)
  - OCL was first IBM contribution to UML

Intentions of OCL

"OCL can be called a "formal" language, but unlike other currently available formal languages such as Objective-Z or VDM++, OCL is not designed for people who have a strong mathematical background. The users of OCL are the same people as the users of UML: software developers with an interest in object technology. OCL is designed for usability, although it is underpinned by mathematical set theory and logic."

Jos Warmer and Anneke Kleppe 2003

OCL Design Decisions

- Textual language
  - Integration with UML diagrams?
- Object-oriented flavour
  - Single class context
- Functional programming style
  - Different from all popular languages
  - Three-valued logic (true, false, undefined)
- Declarative: Executable constraints

Why Work on OCL?

- OCL was new
- OCL was targeting at a bridge between theory and practice
- OCL was in need of tool support
  - In particular:
    Tools to support programming
    - Specification vs. implementation

Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

Maturation of Technologies

"It is commonly thought that it takes 10 years for technology to mature from its initial conception to its wide-spread use."

William E. Riddle, 1984

What to Expect in Ten Years

- Refined standard
  - Precise semantics
  - Extensions
- Various tools
  - Interoperable and semantically consistent
- Extensive practical application
  - Program validation
  - Meta-level: Language definition

Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

Revisiting the Expectations 1

- Refined standard
  - Precise semantics, extensions
- Revisions of OCL standard
  - Towards OCL/UML 2.0 (around 2002)
    - Two-stage process
      - Dresden team involved in first stage
      - Then “Finalization Task Force” took over
    - Formal (set-theoretic) semantics added – and removed
    - Various inconsistent additions
  - No big changes anymore in current version

Semantical Problem in OCL 2.1: Undefined Values

- OCL distinguishes between two “undefined” situations:
  - Invalid / null
  - Non-computable vs. absent values
    - Optional attribute/relation, when absent: undefined only if evaluated further
- Two-stage evaluation semantics
  - Formally defined by Brucker/Krieger/Wolff **2009**
  - Double completion of value domains with “bottom”
- What happened?
  - Informal, semantically unclear additions

Semantical Problem in OCL 2.1: OrderedSet

- What is an “OrderedSet”?
  - A set where the elements have a natural order? (mathematics)
  - A sequence without duplications? (Java)
- Gogolla **2009**:
  - Uncertainty about meaning of type
  - Incomplete definitions in standard

Revisiting the Expectations 2

- Various tools
  - Interoperable and semantically consistent
- Tool comparison paper (Gogolla/Kuhlmann/Büttner 2008):
  - Defines a first benchmark for OCL tools
  - OCL tools differ drastically in implemented semantics

Revisiting the Expectations 3

- Extensive practical application
  - Program validation
  - Meta-level: Language definition
- Apparently, OCL is used in practice
  - However, to a very limited extent
    - Sometimes, proprietary variants are used
  - “Non-standard” usages (e.g. queries)
- Successful usage on meta-level

A Discouraging Quote (2009)

"Case studies have shown that writing OCL specifications is difficult, error-prone and time-consuming."

Joanna Chimiak-Opaka 2009

Summary: We Are Sinking!

- No precise semantics
- Difficult to use
- Limited acceptance in practice
- Tools are rare and inconsistent
- *Worse situation than with traditional formal specification languages!*

Outline

- Starting Point 1999
- Why Work on OCL?
- Expectations
- Ten Years Later...
- And the Future?

Experience and Hopes

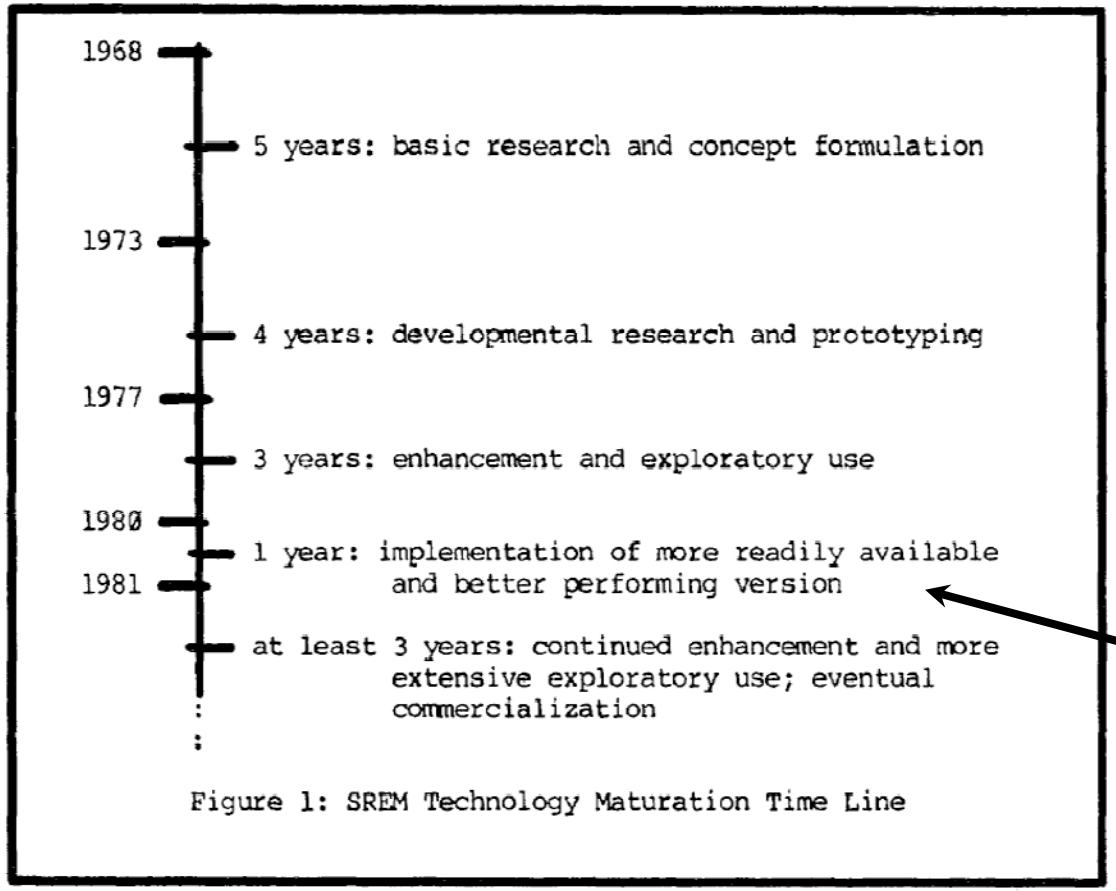


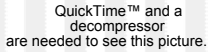
Figure 1: SREM Technology Maturation Time Line

William E. Riddle:
The magic number eighteen plus or minus three: a study of software technology maturation,
ACM SIGSOFT SEN, April **1984**

For OCL (started 1995): Work on better version of language & tools is required!

Hype Cycle

Where is OCL?



spiegel.de