Contents & Goals

Last Lecture:
• Basic Object System Signature $S$ and Structure $D$, System State $\sigma \in \Sigma$
  (Seems like they're related to class/object diagrams, officially we don't know yet...)

This Lecture:
• Educational Objectives:
  Capabilities for these tasks/questions:
  • Please explain this OCL constraint.
  • Please formalise this constraint in OCL.
  • Does this OCL constraint hold in this system state?
  • Can you think of a system state satisfying this constraint?
  • Please un-abbreviate all abbreviations in this OCL expression.
  • In what sense is OCL a three-valued logic? For what purpose?
  • How are $D(C)$ and $\tau C$ related?

What is OCL? And What is It Good For?

• OCL: Object Constraint Logic.

OCL/Beispiel

TeamMember
  name : String
  age : Integer

Location
  participants : Integer
  meetings * : String

Meeting
  title : String
  numParticipants : Integer
  start : Date
  duration : Time

context TeamMember
inv:
  age => 18

context Meeting
inv:
  duration > 0

((C) Prof. Dr. P. Thiemann, http://proglang.informatik.uni-freiburg.de/teaching/swt/2008/)

What's It Good For?

• Most prominent:
  write down requirements supposed to be satisfied by all system states. Often targeting all alive objects of a certain class.

• Not unknown:
  write down pre/post-conditions of methods (Behavioural Features). Then evaluated over two system states.

• Common with State Machines:
  guards in transitions.

• Lesser known:
  provide operation bodies.

• Metamodeling: the UML standard is a MOF-Model of UML. OCL expressions define well-formedness of UML models (cf. Lecture ∼ 21).
Generalised notation:

\[ \text{isEmpty}, \quad \in \{ 1 \rightarrow \text{expr} \} \]

For example:

\[ \tau \in \text{Int} \]

\[ \text{Set}(\text{expr} 1) \]

\[ \tau \cup \tau \]

\[ \tau \notin \text{Int} \]

\[ \text{expr} \]

\[ \text{set-of-} \]

\[ \tau \rightarrow \text{Int} \]

\[ \text{Diagram} \]

\[ \text{ModelInstances} \]

\[ \text{Deployment} \]

\[ \text{Use Case} \]

\[ \text{State M} \]

\[ \text{Obj-type} \]

\[ \text{denotes the} \]

\[ \text{logical variables} \]

\[ \in \tau \]

\[ \text{Variable} \]

\[ \in \text{set-of-} \]

\[ \text{Equality} \]

\[ \text{Set} \]

\[ \text{OCL} \]

\[ \text{Planning} \]

Next time:

\[ \text{OCL} \]

\[ \text{Semantic of OCL} \]

\[ \text{Language Constructs} \]

\[ \text{Temporal Logic} \]

\[ \text{OCL} \]

\[ \text{Expression Examples} \]

\[ \text{Diagram Examples} \]

\[ \text{State Machines} \]

\[ \text{Diagram} \]

\[ \text{UML} \]
we may alternatively write ("abbreviate as")\[OMG, 2006\].

\[w \mapsto v\n\]

\[\tau \mapsto w_1, \ldots, w_n: \text{Set}\]

\[\tau\mapsto w\in \text{Set} \mid \text{true} = \text{expr} \quad \tau\mapsto w\text{; context}\]

\[\tau\mapsto w\text{; context}\]

\[\text{iterate} \quad \text{expr} = \text{self} \quad \tau\mapsto \text{result} \quad \tau\mapsto \text{iter}():\]

\[\text{iterate}\]

\[\text{OCL Syntax 3/4: Iterate}\]

\[\text{OCL Syntax 4/4: Context}\]

Abbreviations on top of Iterate
Examples (from lecture "Softwaretechnik 2008")

TeamMember

name : String
age : Integer

Location

participants 2..* meetings

Meeting
title : String
numParticipants : Integer
start : Date
duration: Time

move(newStart : Date)

1 *

context TeamMember
inv: age => 18

context Meeting
inv: duration > 0

((C) Prof. Dr. P. Thiemann,
http://proglang.informatik.uni-freiburg.de/
Examples (from lecture "Softwaretechnik 2008")

– 03 – 2014-10-28 – Soclsyn –

18/23

OCL/Mehr Navigation/Beispiele

TeamMember
name : String
age : Integer

Location

participants 2..* meetings

Meeting
title : String
numParticipants : Integer
start : Date
duration: Time

move(newStart : Date)

1 *

context Meeting
inv:
participants->size() = numParticipants

context Location
inv:
name="Lobby"
implies
meeting->isEmpty()

((C) Prof. Dr. P. Thiemann,
http://proglang.informatik.uni-freiburg.de/teaching/swt/2008/

Among others:
• Enumeration types
• Type hierarchy
• Complete list of arithmetical operators
• The two other collection types Bag and Sequence
• Casting
• Runtime type information
• Pre/post conditions (maybe later, when we officially know what an operation is)
• ...

OCL Semantics: The Task

expr ::= w : τ (w)
| expr1 = τ expr2 : τ × τ → Bool
| oclIsUndefined τ (expr1) : τ → Bool
| { expr1, . . . , exprn } : τ × · · · × τ → Set (τ)
| isEmpty (expr1) : Set (τ) → Bool
| size (expr1) : Set (τ) → Int
| allInstances C : Set (τ C) → Bool
| v (expr1) : τ C → τ (v)
| r1 (expr1) : τ C → τ D
| r2 (expr1) : τ C → Set (τ D)

• Given an OCL expression expr, a system state σ ∈ ΣDS, and a valuation of logical
variables β, define
I⁄llbracket · , · /rrbracket (·, ·) : OCLExpressions (S) × ΣDS × (W → I(T ∪ TB ∪ TC)) → I(Bool)
i.e.
I⁄llbracket expr /rrbracket (σ, β) ∈ {true, false, ⊥ Bool}.

References

