Software Design, Modelling and Analysis in UML

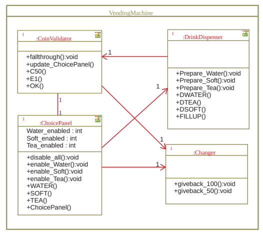
Lecture 2: Semantical Model

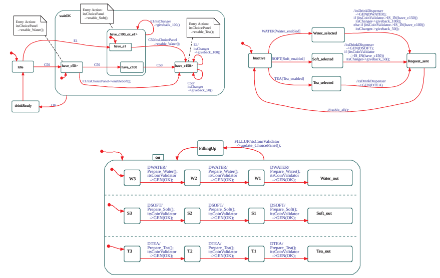
2015-10-22

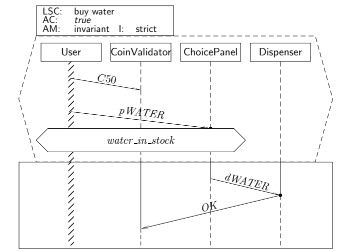
Prof. Dr. Andreas Podelski, **Dr. Bernd Westphal**

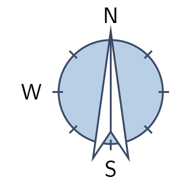
Albert-Ludwigs-Universität Freiburg, Germany

Course Map









UML

$\mathcal{CD}$, $\mathcal{SM}$ $\quad$ $\varphi \in$ OCL $\quad$ $\mathcal{CD}$, $\mathcal{SD}$

Model

$\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$, SM $\quad$ $expr$ $\quad$ $\mathscr{S}, SD$

$M = (\Sigma_{\mathscr{S}}^{\mathscr{D}}, A_{\mathscr{S}}, \to_{SM})$ $\quad$ $B = (Q_{SD}, q_0, A_{\mathscr{S}}, \to_{SD}, F_{SD})$

Instances

$\pi = (\sigma_0, \varepsilon_0) \xrightarrow[u_0]{(cons_0, Snd_0)} (\sigma_1, \varepsilon_1) \cdots$ $\quad$ $w_\pi = ((\sigma_i, cons_i, Snd_i))_{i \in \mathbb{N}}$

$G = (N, E, f)$ $\quad$ Mathematics

$\mathcal{OD}$ $\quad$ UML

Contents & Goals

Last Lecture:

- Introduction: Motivation, Content, Formalia

This Lecture:

- **Educational Objectives:** Capabilities for following tasks/questions.
  - What is a signature, an object, a system state, etc.?
  - What is the purpose of signature, object, etc. in the course?
  - How do Basic Object System Signatures relate to UML class diagrams?
- **Content:**
  - Basic Object System Signatures
  - Structures
  - System States

Semantical Foundation

Basic Object System Signature

Definition. A (Basic) Object System Signature is a quadruple

$$\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$$

could have chosen alt.
C_1 C_Δ
$C_?$ $C_\square$

where

- $\mathscr{T}$ is a set of (basic) types,
- $\mathscr{C}$ is a finite set of classes,
- V is a finite set of typed attributes, i.e., each $v \in V$ has a type
  - $\tau \in \mathscr{T}$, or
  - $C_{0,1}$ or C_*, where $C \in \mathscr{C}$

  (written $v : \tau$ or $v : C_{0,1}$ or $v : C_*$),
- $atr : \mathscr{C} \to 2^V$ maps each class to its set of attributes.

total function powerset of V

Note: Inspired by OCL 2.0 standard OMG (2006), Annex A.

Basic Object System Signature Example

$\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$ where

- (basic) types $\mathscr{T}$ and classes $\mathscr{C}$ (both finite),
- typed attributes V, τ from $\mathscr{T}$, or $C_{0,1}$ or C_*, for some $C \in \mathscr{C}$,
- $atr : \mathscr{C} \to 2^V$ mapping classes to attributes.

Example: set of basic types $\mathscr{T}$ set of classes $\mathscr{C}$ attributes V attributes mapping atr

$$\mathscr{S}_0 = (\{Int\}, \{C, D\}, \{\underbrace{x : Int}_{\text{attribute } x \text{ has (basic) type } Int}, p : C_{0,1}, \underbrace{n : C_*}_{\text{attribute } n \text{ has (desired) type } C_*}\}, \{\underbrace{C \mapsto \{p, n\}}_{C \text{ has attributes } p \text{ and } n}, D \mapsto \{x\}\})$$

"maps to"

$atr(C) = \{p, n\}$

$atr(D) = \{x\}$

Basic Object System Signature Another Example

$\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$ where

- (basic) types $\mathscr{T}$ and classes $\mathscr{C}$ (both finite),
- typed attributes V, τ from $\mathscr{T}$, or $C_{0,1}$ or C_*, for some $C \in \mathscr{C}$,
- $atr : \mathscr{C} \to 2^V$ mapping classes to attributes.

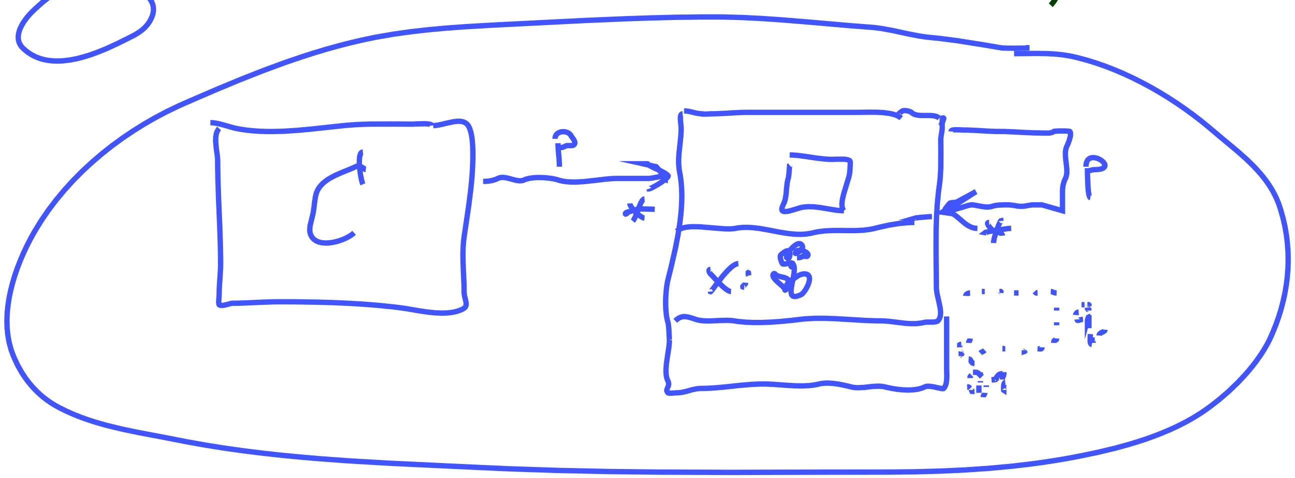
Example:

no, $Int \notin \mathcal{T}$ of $\mathcal{S}_1$

$$\mathcal{S}_1 = (\{\$, MyType\}, \{C, \square\}, \{\cancel{x : Int}\ x : \$, p : \square_*, q : \square_{0,1}\},$$

$$\{C \mapsto \{p\}, \square \mapsto \{x, p\}\})$$

need not be used (if not used, may be omitted)



looks like a class diagram...

Basic Object System Structure

Definition. A Basic Object System Structure of $\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$ is a domain function $\mathscr{D}$ which assigns to each type a domain, i.e.

- $\tau \in \mathscr{T}$ is mapped to $\mathscr{D}(\tau)$,
- $C \in \mathscr{C}$ is mapped to an infinite set $\mathscr{D}(C)$ of (object) identities.

  Note: Object identities only have the "$=$" operation.
- Sets of object identities for different classes are disjoint, i.e.

$$\forall\, C, D \in \mathscr{C} : C \neq D \to \mathscr{D}(C) \cap \mathscr{D}(D) = \emptyset.$$

- C_* **and** $C_{0,1}$ for $C \in \mathscr{C}$ are mapped to $2^{\mathscr{D}(C)}$.

We use $\mathscr{D}(\mathscr{C})$ to denote $\bigcup_{C \in \mathscr{C}} \mathscr{D}(C)$; analogously $\mathscr{D}(\mathscr{C}_*)$.

Note: We identify objects and object identities,
because both uniquely determine each other (cf. OCL 2.0 standard).

Basic Object System Structure Example

Wanted: a structure for signature

$$\mathscr{S}_0 = (\{Int\}, \{C, D\}, \{x : Int, p : C_{0,1}, n : C_*\}, \{C \mapsto \{p, n\}, D \mapsto \{x\}\})$$

$\mathscr{D}$ needs to map:

- $\tau \in \mathscr{T}$ to **some** $\mathscr{D}(\tau)$,
- $C \in \mathscr{C}$ to **some** set of identities $\mathscr{D}(C)$ (infinite, disjoint for different classes),
- C_* and $C_{0,1}$ for $C \in \mathscr{C}$: always mapped to $\mathscr{D}(C_*) = \mathscr{D}(C_{0,1}) = 2^{\mathscr{D}(C)}$.

$$\begin{aligned}
\mathscr{D}(Int) &= \mathbb{Z} \\
\mathscr{D}(C) &= \mathbb{N}^+ \times \{C\} \cong \{1_C, 2_C, 3_C, \ldots\} \\
\mathscr{D}(D) &= \mathbb{N}^+ \times \{D\} \cong \{1_D, 2_D, 3_D, \ldots\} \\
\mathscr{D}(C_{0,1}) = \mathscr{D}(C_*) &= 2^{\mathscr{D}(C)} \\
\mathscr{D}(D_{0,1}) = \mathscr{D}(D_*) &= 2^{\mathscr{D}(D)}
\end{aligned}$$

$$\begin{aligned}
\mathscr{D}_1(Int) &= \{-2, -1, 0, 1, 2\} \\
\mathscr{D}_1(C) &= \{a, aa, aaa, \ldots\} \\
\mathscr{D}_1(D) &= \{b, bb, bbb, \ldots\} \\
&= 2^{\mathscr{D}_1(C)} \\
&= 2^{\mathscr{D}_1(D)}
\end{aligned}$$

System State

set of all object identities in $\mathscr{D}$

partial function mapping attributes (for V) to ~~type~~ values (from $\mathscr{D}(\mathscr{T})$ and $\mathscr{D}(\mathscr{C}_*)$)

Definition. Let $\mathscr{D}$ be a structure of $\mathscr{S} = (\mathscr{T}, \mathscr{C}, V, atr)$.

A system state of $\mathscr{S}$ wrt. $\mathscr{D}$ is a **type-consistent** mapping

partial function

$$\sigma : \mathscr{D}(\mathscr{C}) \nrightarrow (V \nrightarrow (\mathscr{D}(\mathscr{T}) \cup \mathscr{D}(\mathscr{C}_*))).$$

set of attributes

set of all values in $\mathscr{D}$

That is, for each $u \in \mathscr{D}(C)$, $C \in \mathscr{C}$, if $u \in \mathrm{dom}(\sigma)$

- $\mathrm{dom}(\sigma(u)) = atr(C)$
- $(\sigma(u))(v) \in \mathscr{D}(\tau)$ if $v : \tau, \tau \in \mathscr{T}$
- $(\sigma(u))(v) \in \mathscr{D}(D_*)$ if $v : D_{0,1}$ or $v : D_*$ with $D \in \mathscr{C}$

$: V \nrightarrow \mathscr{D}(\mathscr{T}) \cup \mathscr{D}(\mathscr{C}_*)$

We call $u \in \mathscr{D}(\mathscr{C})$ alive in σ if and only if $u \in \mathrm{dom}(\sigma)$.

We use $\Sigma_{\mathscr{S}}^{\mathscr{D}}$ to denote the set of all system states of $\mathscr{S}$ wrt. $\mathscr{D}$.

System State Example

$$\mathscr{S}_0 = (\{Int\}, \{C, D\}, \{x : Int, p : C_{0,1}, n : C_*\}, \{C \mapsto \{p, n\}, D \mapsto \{x\}\})$$

$$\mathscr{D}(Int) = \mathbb{Z}, \quad \mathscr{D}(C) = \{1_C, 2_C, 3_C, ...\}, \quad \mathscr{D}(D) = \{1_D, 2_D, 3_D, ...\}$$

Wanted: $\sigma : \mathscr{D}(\mathscr{C}) \nrightarrow (V \nrightarrow (\mathscr{D}(\mathscr{T}) \cup \mathscr{D}(\mathscr{C}_*)))$ such that (i) $\mathrm{dom}(\sigma(u)) = atr(C)$, and

(ii) $\sigma(u)(v) \in \mathscr{D}(\tau)$ if $v : \tau, \tau \in \mathscr{T}$, (iii) $\sigma(u)(v) \in \mathscr{D}(C_*)$ if $v : D_*$ with $D \in \mathscr{C}$.

$\sigma_1 = \emptyset$ ("empty function")

alive in σ_1: none

$$\sigma_2 = \{ \underbrace{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{1_C, 5_C\}\}}, \underbrace{5_C \mapsto \{p \mapsto \{1_C\}, n \mapsto \emptyset\}}, \underbrace{3_D \mapsto \{x \mapsto 3\}} \}$$

alive in σ_2: $1_C, 5_C, 3_D$

not alive: everybody else

$\sigma(17_D) = \{x \mapsto 0\}$

$\sigma(17_D)(x) = 0$

$$\sigma_3 = \{ \underbrace{1_D \mapsto \{x = 27\}}, \underbrace{2_D \mapsto \{x = 27\}}, \underbrace{17_D \mapsto \{x \mapsto 0\}} \}$$

alive objects in σ_3: $1_D, 2_D, 17_D$

System State Example

$$\mathscr{S}_0 = (\{Int\}, \{C, D\}, \{x : Int, p : C_{0,1}, n : C_*\}, \{C \mapsto \{p, n\}, D \mapsto \{x\}\})$$

$$\mathscr{D}(Int) = \mathbb{Z}, \quad \mathscr{D}(C) = \{1_C, 2_C, 3_C, ...\}, \quad \mathscr{D}(D) = \{1_D, 2_D, 3_D, ...\}$$

Wanted: $\sigma : \mathscr{D}(\mathscr{C}) \nrightarrow (V \nrightarrow (\mathscr{D}(\mathscr{T}) \cup \mathscr{D}(\mathscr{C}_*)))$ such that (i) $\mathrm{dom}(\sigma(u)) = atr(C)$, and

(ii) $\sigma(u)(v) \in \mathscr{D}(\tau)$ if $v : \tau, \tau \in \mathscr{T}$, (iii) $\sigma(u)(v) \in \mathscr{D}(C_*)$ if $v : D_*$ with $D \in \mathscr{C}$.

Two options:

- **Concrete, explicit** identities:

$$\sigma_c = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}, 5_C \mapsto \{p \mapsto \emptyset, n \mapsto \emptyset\}, 1_D \mapsto \{x \mapsto 23\}\}.$$

- **Alternative**: **symbolic** system state.

$$\sigma_s = \{c_1 \mapsto \{p \mapsto \emptyset, n \mapsto \{c_2\}\}, c_2 \mapsto \{p \mapsto \emptyset, n \mapsto \emptyset\}, d \mapsto \{x \mapsto 23\}\}$$

assuming $c_1, c_2 \in \mathscr{D}(C)$, $d \in \mathscr{D}(D)$, $c_1 \neq c_2$.

System State: Spot the 10 (?) Mistakes

$$\mathscr{S}_0 = (\{Int\}, \{C, D\}, \{x : Int, p : C_{0,1}, n : C_*\}, \{C \mapsto \{p, n\}, D \mapsto \{x\}\})$$

$$\mathscr{D}(Int) = \mathbb{Z}, \quad \mathscr{D}(C) = \{1_C, 2_C, 3_C, ...\}, \quad \mathscr{D}(D) = \{1_D, 2_D, 3_D, ...\}$$

Wanted: $\sigma : \mathscr{D}(\mathscr{C}) \nrightarrow (V \nrightarrow (\mathscr{D}(\mathscr{T}) \cup \mathscr{D}(\mathscr{C}_*)))$ such that (i) $\mathrm{dom}(\sigma(u)) = atr(C)$, and (ii) $\sigma(u)(v) \in \mathscr{D}(\tau)$ if $v : \tau, \tau \in \mathscr{T}$, (iii) $\sigma(u)(v) \in \mathscr{D}(C_*)$ if $v : D_*$ with $D \in \mathscr{C}$.

empty set $\{\}$

$↯$ (iii) $1_C \notin 2^{\mathscr{D}(C)}$

$↯$ (ii), $2.3 \notin \mathscr{D}(Int)$

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}, \quad 5_C \mapsto \{p \mapsto \emptyset, n \mapsto 1_C\}, \quad 1_D \mapsto \{x \mapsto 2.3\}\}.$

$↯$ (iii)

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}, \quad 5_C \mapsto \{p \mapsto 1_C, n \mapsto \emptyset\}, \quad 1_D \mapsto \{x \mapsto 23\}\}.$

$↯$ (iii) $\notin 2^{\mathscr{D}(C)}$ $↯$

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{1_D\}\}, \quad 5_C \mapsto \{p \mapsto \emptyset, n \mapsto \emptyset\}, \quad 1_D \mapsto \{x \mapsto 22\}\}.$

$↯$ (i) $p \in atr(C)$!

$↯$ (i), $p \notin atr(D)$

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}, \quad 5_C \mapsto \{n \mapsto \emptyset\}, \quad 1_D \mapsto \{x \mapsto 1, p \mapsto \{1_C\}\}\}.$

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}, \quad 5_C \mapsto \{p \mapsto \emptyset, n \mapsto \{9_C\}\}\}$

Dangling References

Definition. Let $\sigma \in \Sigma_{\mathscr{S}}^{\mathscr{D}}$ be a system state.

We say attribute $v \in V_{0,1,*}$, i.e. $v : C_{0,1}$ or $v : C_*$, in object $u \in \mathrm{dom}(\sigma)$ has a dangling reference if and only if the attribute's value comprises an object which is not alive in σ, i.e. if

$$(\sigma(u))(v) \not\subseteq \mathrm{dom}(\sigma).$$

alive objects

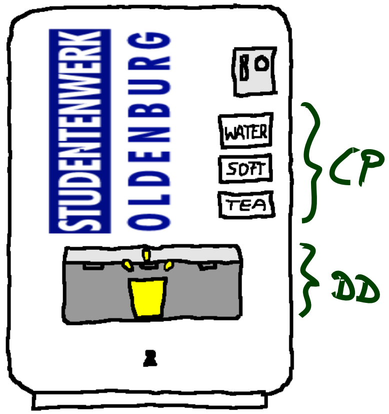
We call σ closed if and only if no attribute has a dangling reference in any object alive in σ.

Example:

- $\sigma = \{1_C \mapsto \{p \mapsto \emptyset, n \mapsto \{5_C\}\}\}$

$$(\sigma(1_C))(n) = \{5_C\} \not\subseteq \{1_C\} = \mathrm{dom}(\sigma)$$

A Complete Example: Vending Machine



$$\mathscr{S} = \Big(\{Bool, Nat\},$$
$$\{VM, CP, DD\},$$
$$\{cp : CP_*,\ dd : DD_{0,1},\ wen : Bool,\ win : Nat\},$$
$$\{VM \mapsto \{cp, dd\},\ CP \mapsto \{wen\},\ DD \mapsto \{win, wen\}\} \Big)$$

$$\mathscr{D}(Bool) = \{true, false\}$$
$$\mathscr{D}(Nat) = \mathbb{N}$$
$$\mathscr{D}(VM) = \{1_{VM}, 2_{VM}, \dots\}$$
$$\mathscr{D}(DD) = \{1_{DD}, \dots\}$$
$$\mathscr{D}(CP) = \{1_{CP}, \dots\}$$

$$\mathscr{D}(DD_{0,1}) = 2^{\mathscr{D}(DD)} = 2^{\{1_{DD}, 2_{DD}, \dots\}}$$

context DD inv:
wen implies win > 0

$$\sigma = \{ 7_{VM} \mapsto \{ dd \mapsto \{1_{DD}\},\ cp \mapsto \{3_{CP}, 5_{CP}\}\},$$
$$1_{DD} \mapsto \{ win \mapsto 13,\ wen \mapsto true\}$$
$$3_{CP} \mapsto \{ wen \mapsto true\},$$
$$5_{CP} \mapsto \{ wen \mapsto false\} \}$$

References

OMG (2006). Object Constraint Language, version 2.0. Technical Report formal/06-05-01.

OMG (2011a). Unified modeling language: Infrastructure, version 2.4.1. Technical Report formal/2011-08-05.

OMG (2011b). Unified modeling language: Superstructure, version 2.4.1. Technical Report formal/2011-08-06.