Course Map
Contents & Goals

Last Lecture:
- Object Diagrams
  - partial vs. complete; for analysis; for documentation...

This Lecture:
- Educational Objectives: Capabilities for following tasks/questions.
  - What is a class diagram?
  - For what purposes are class diagrams useful?
  - Could you please map this class diagram to a signature?
  - Could you please map this signature to a class diagram?

- Content:
  - Study UML syntax.
  - Prepare (extend) definition of signature.
  - Map class diagram to (extended) signature.
  - Stereotypes.

UML Class Diagrams: Stocktaking
Recall: Signature vs. Class Diagram

Basic Object System Signature

Another Example

\[ \mathcal{S} = (\mathcal{T}, \mathcal{C}, V, \text{atr}) \]  

- (basic) types \( \mathcal{T} \) and classes \( \mathcal{C} \) (both finite),
- typed attributes \( V, \tau \) from \( \mathcal{T} \) or \( \mathcal{C}_0,1, \) or \( \mathcal{C}^* \), for some \( \mathcal{C} \in \mathcal{C} \),
- \( \text{atr} : \mathcal{C} \rightarrow 2^V \) mapping classes to attributes.

Example:

\[ \mathcal{S} = \{ \{ \mathcal{C}_1, \ldots, \mathcal{C}_n \} \} \]

\[ \text{Package::} \mathcal{C} \]

\[ + r : \mathcal{C}_0,1 = \text{expr} \]

\[ s : \mathcal{D}_x \{ \text{ordered} \} \]

\[ - v : \text{Int} = 27 \]

\[ w : \text{Float} \{ \text{readOnly} \} \]

That’d Be Too Simple
What Do We Want / Have to Cover?

A class
- has a set of stereotypes,
- has a name,
- belongs to a package,
- can be abstract,
- can be active,
- has a set of attributes,
- has a set of operations.

Each attribute has
- a visibility,
- a name, a type,
- a multiplicity, an order,
- an initial value,
- a set of properties, such as readOnly, ordered, etc.

Wanted: places in the signature to represent the information from the picture.

Extended Signature
Extended Signature

Definition. An (Extended) Object System Signature is a quadruple $\mathcal{S} = (\mathcal{F}, \mathcal{C}, V, \text{atr})$ where

- $\mathcal{F}$ is a set of (basic) types,
- $\mathcal{C}$ is a finite set of classes $\langle C, S_C, a, t \rangle$ where
  - $S_C$ is a finite (possibly empty) set of stereotypes,
  - $a \in \mathbb{B}$ is a boolean flag indicating whether $C$ is abstract,
  - $t \in \mathbb{B}$ is a boolean flag indicating whether $C$ is active,
- $V$ is a finite set of attributes $\langle v : T, \xi, \text{expr}_0, P_v \rangle$ where
  - $T$ is a type from $\mathcal{F}$, or $C_0, C_1, C^*$ for some $C \in \mathcal{C}$,
  - $\xi \in \{\text{public}, \text{private}, \text{protected}, \text{package}\}$ is the visibility,
  - an initial value expression $\text{expr}_0$ given as a word from a language for initial value expressions, e.g., OCL, or C++ in the Rhapsody tool,
  - a finite (possibly empty) set of properties $P_v$.
- $\text{atr} : \mathcal{C} \rightarrow 2^V$ maps each class to its set of attributes.

We use $S_{\mathcal{C}}$ to denote the set $\bigcup_{C \in \mathcal{C}} S_C$ of stereotypes in $\mathcal{S}$.

Conventions

- We write $\langle C, S_C, a, t \rangle$ if we want to refer to all aspects of $C$.
- If the new aspects are irrelevant (for a given context), we simply write $C$, i.e., old definitions are still valid.
- We write $\langle v : T, \xi, \text{expr}_0, P_v \rangle$ if we want to refer to all aspects of $v$.
- Write only $v : T$ or $v$ if details are irrelevant.

Note:
All definitions we have up to now principally still apply as they are stated in terms of, e.g., $C \in \mathcal{C}$ — which still has a meaning with the extended view.

For instance, system states and object diagrams will remain mostly unchanged.

The other way round: most of the newly added aspects do not contribute to the constitution of system states or object diagrams.
From Class Boxes to Extended Signatures

A class box $n$ induces an (extended) signature class as follows:

$$\langle S_1, \ldots, S_k \rangle$$

$$\xi_1 \ v_1 : T_1 = expr_{1} \{ P_{1,1}, \ldots, P_{1,m_1} \}$$

$$\vdots$$

$$\xi_\ell \ v_\ell : T_\ell = expr_{\ell} \{ P_{\ell,1}, \ldots, P_{\ell,m_\ell} \}$$

$$C(n) := \langle C, \{ S_1, \ldots, S_k \}, a(n), t(n) \rangle$$

$$V(n) := \{ \langle v_1 : T_1, \xi_1, expr_{1} \{ P_{1,1}, \ldots, P_{1,m_1} \} \rangle, \ldots, \langle v_\ell : T_\ell, \xi_\ell, expr_{\ell} \{ P_{\ell,1}, \ldots, P_{\ell,m_\ell} \} \rangle \}$$

$$attr(n) := \{ C \mapsto \{ v_1, \ldots, v_\ell \} \}$$

where

- “abstract” is determined by the font:
  $$a(n) = \begin{cases} 
  \text{true} & \text{if } n = \text{C or } n = \text{C}\{A\} \\
  \text{false} & \text{otherwise}
  \end{cases}$$

- “active” is determined by the frame:
  $$t(n) = \begin{cases} 
  \text{true} & \text{if } n = \text{C} \\
  \text{false} & \text{otherwise}
  \end{cases}$$
Example

\[
\begin{align*}
\langle \{S_1, \ldots, S_k\} \rangle \\
C, \\
\xi_1, v_1 : T_1 = \text{expr}_1 \{P_{1,1}, \ldots, P_{1,m_1}\} \\
\vdots \\
\xi_k, v_k : T_k = \text{expr}_k \{P_{k,1}, \ldots, P_{k,m_k}\}
\end{align*}
\]

\[\downarrow\]

\[C(n) := (C, \{S_1, \ldots, S_k\}, \alpha(n), \iota(n))\]

\[V(n) := \{\langle v_1 : T_1, \xi_1, \text{expr}_1, \{P_{1,1}, \ldots, P_{1,m_1}\} \rangle, \ldots, \langle v_l : T_l, \xi_l, \text{expr}_l, \{P_{l,1}, \ldots, P_{l,m_l}\} \rangle\}\]

\[\text{atr}(n) := \{C \mapsto \{v_1, \ldots, v_l\}\}\]

\[\text{atr}(n) := \{C \mapsto \{v_1, \ldots, v_l\}\}\]

What If Things Are Missing?

It depends.

- What does the standard say? (OMG, 2011a, 121)
  “Presentation Options.
  The type, visibility, default, multiplicity, property string may be suppressed from being displayed, even if there are values in the model.”

- Visibility: There is no “no visibility” — an attribute has a visibility in the (extended) signature.
  Some (and we) assume public as default, but conventions may vary.

- Initial value: some assume it given by domain (such as “leftmost value”, but what is “leftmost” of Z?).
  Some (and we) understand non-deterministic initialisation if not given.

- Properties: probably safe to assume \{\} if not given at all.
Example Cont’d

\[
\begin{aligned}
\langle S_1, \ldots, S_k \rangle \\
C \\
\xi_1 : T_1 = \text{expr}_1 \{ P_{1,1}, \ldots, P_{1,m_1} \} \\
\xi_k : T_k = \text{expr}_k \{ P_{2,1}, \ldots, P_{2,m} \}
\end{aligned}
\]

\[
C(n) := \langle C, \{ S_1, \ldots, S_k \}, a(n), t(n) \rangle
\]
\[
V(n) := \{ \langle n_1 : T_1, \xi_1, \text{expr}_1, \{ P_{1,1}, \ldots, P_{1,m_1} \} \rangle, \ldots, \langle n_k : T_k, \xi_k, \text{expr}_k, \{ P_{2,1}, \ldots, P_{2,m} \} \rangle \}
\]
\[
\text{atr}(n) := \{ C \mapsto \{ v_1, \ldots, v_l \} \}
\]

\[
\begin{aligned}
\langle S_1, +, \{ \text{s1}, \text{ordered} \} \rangle \\
\langle S_1, +, \{ \text{ordered} \} \rangle
\end{aligned}
\]

\[
\text{From Class Diagrams to Extended Signatures}
\]

- We view a class diagram \( CD \) as a graph with nodes \( \{ n_1, \ldots, n_N \} \) (each “class rectangle” is a node).
- \( E(CD) := \{ C(n_i) \mid 1 \leq i \leq N \} \)
- \( V(CD) := \bigcup_{i=1}^{N} V(n_i) \)
- \( \text{atr}(CD) := \bigcup_{i=1}^{N} \text{atr}(n_i) \)

- In a UML model, we can have finitely many class diagrams, \( \mathcal{C} = \{ CD_1, \ldots, CD_k \} \), which induce the following signature:

\[
\mathcal{I}(\mathcal{C}) = \left( \mathcal{I}, \bigcup_{i=1}^{k} E(CD_i), \bigcup_{i=1}^{k} V(CD_i), \bigcup_{i=1}^{k} \text{atr}(CD_i) \right).
\]

(Assuming \( \mathcal{I} \) given. In “reality” (i.e. in full UML), we can introduce types in class diagrams, the class diagram then contributes to \( \mathcal{I} \). Example: enumeration types.)
Is the Mapping a Function?

**Question:** Is \( \mathcal{I}(\mathcal{C} \mathcal{D}) \) well-defined?

**References**
References


