Contents & Goals

Last Lectures:
- associations syntax and semantics

This Lecture:
- **Educational Objectives:** Capabilities for following tasks/questions.
  - Please explain this class diagram with associations.
  - Which annotations of an association arrow are semantically relevant?
  - What’s a role name? What’s it good for?
  - What is “multiplicity”? How did we treat them semantically?
  - What is “reading direction”, “navigability”, “ownership”, . . . ?
  - What’s the difference between “aggregation” and "composition"?

- **Content:**
  - Associations and OCL
  - Btw.: where do we put OCL constraints?
Recall: We consider associations of the following form:

\[ \langle r : \langle \text{role}_1 : C_1, \mu_1, P_1, \xi_1, \nu_1, o_1 \rangle, \ldots, \langle \text{role}_n : C_n, \mu_n, P_n, \xi_n, \nu_n, o_n \rangle \rangle \]

Only these parts are relevant for extended system states:

\[ \langle r : \langle \text{role}_1 : C_1, \ldots, P_1, \ldots \rangle, \ldots, \langle \text{role}_n : C_n, \ldots, P_n, \ldots \rangle \rangle \]

(recall: we assume \( P_1 = P_n = \{\text{unique}\} \)).

The UML standard thinks of associations as \textbf{n-ary relations} which \textit{live on their own} in a system state.

That is, \textbf{links} (= association instances)

- \textbf{do not} belong (in general) to certain objects (in contrast to pointers, e.g.)
- are \textit{“first-class citizens” next to objects},
- are (in general) \textbf{not} directed (in contrast to pointers).
Links in System States

Definition. Let $D$ be a structure of the (extended) signature $\mathcal{S} = (\mathcal{T}, \mathcal{C}, V, atr)$. A system state of $\mathcal{S}$ wrt. $D$ is a pair $(\sigma, \lambda)$ consisting of

- a type-consistent mapping $\sigma : D(\mathcal{C}) \rightarrow (atr(\mathcal{C}) \rightarrow D(\mathcal{T}))$,
- a mapping $\lambda$ which assigns each association $\langle r : \langle role_1 : C_1, \ldots, role_n : C_n \rangle \rangle \in V$ a relation $\lambda(r) \subseteq D(C_1) \times \cdots \times D(C_n)$ (i.e. a set of type-consistent n-tuples of identities).

Association/Link Example

Signature:

$\mathcal{S} = (\{Int\}, \{C, D\}, \{x : Int, \langle A_{C, D} : (c : C, 0..*, +, \{\text{unique}\}, \times, 1), \langle n : D, 0..*, +, \{\text{unique}\}, >, 0\rangle \rangle, \{C \mapsto \emptyset, D \mapsto \{x\}\})$

A system state of $\mathcal{S}$ (some reasonable $D$) is $(\sigma, \lambda)$ with:

$\sigma = \{1_C \mapsto \emptyset, 3_D \mapsto \{x \mapsto 1\}, 7_D \mapsto \{x \mapsto 2\}\}$

$\lambda = \{A_{C, D} \mapsto \{(1_C, 3_D), (1_C, 7_D)\}\}$
**Legitimate question**: how do we represent system states such as

\[
\sigma = \{1_C \mapsto \emptyset, 3_D \mapsto \{x \mapsto 1\}, 7_D \mapsto \{x \mapsto 2\}\} \\
\lambda = \{A\_C\_D \mapsto \{(1_C, 3_D), (1_C, 7_D)\}\}
\]

as **object diagram**?

---

**Associations and OCL**
**OCL and Associations: Syntax**

**Recall**: OCL syntax as introduced in Lecture 03, interesting part:

\[
expr ::= \ldots \mid r_1(expr_1) : \tau_C \rightarrow \tau_D \quad r_1 : D_{0,1} \in \text{atr}(C) \\
| \quad r_2(expr_1) : \tau_C \rightarrow \text{Set}(\tau_D) \quad r_2 : D_* \in \text{atr}(C)
\]

Now becomes

\[
expr ::= \ldots \mid \text{role}(expr_1) : \tau_C \rightarrow \tau_D \quad \mu = 0..1 \text{ or } \mu = 1 \\
| \quad \text{role}(expr_1) : \tau_C \rightarrow \text{Set}(\tau_D) \quad \text{otherwise}
\]

if that is

\[
\begin{cases}
\langle r : \ldots, \langle \text{role} : D, \mu, \ldots \rangle, \ldots, \langle \text{role} : C, \ldots \rangle, \ldots \rangle \in V \text{ or} \\
\langle r : \ldots, \langle \text{role} : C, \ldots \rangle, \ldots, \langle \text{role} : D, \mu, \ldots \rangle, \ldots \rangle \in V, \text{role} \neq \text{role}'
\end{cases}
\]

Note:

- Association name as such doesn’t occur in OCL syntax, role names do.
- \( expr_1 \) has to denote an object of a class which “participates” in the association.
OCL and Associations Syntax: Example

\[
\begin{align*}
\text{expr} &::= \ldots | \text{role(expr)} : \tau_C \rightarrow \tau_D & \mu = 0..1 \text{ or } \mu = 1 \\
| \text{role(expr)} : \tau_C \rightarrow \text{Set}(\tau_D) & \text{ otherwise} \\
\end{align*}
\]

if

\[
\langle r : \ldots, \text{role} : D, \mu, \ldots, \text{role} : C, \ldots \rangle \in V \text{ or } \langle r : \ldots, \text{role} : C, \ldots \rangle, \ldots \rangle \in V, \text{ role } \neq \text{ role}'.
\]

Figure 7.21 - Binary and ternary associations.[OMG, 2007b, 44].

OCL and Associations: Semantics

Recall: (Lecture 03)

Assume \(\text{expr}_1 : \tau_C\) for some \(C \in \mathcal{E}\). Set \(u_1 := I[\text{expr}_1](\sigma, \beta) \in \mathcal{D}(\tau_C)\).

- \(I[r_1(\text{expr}_1)](\sigma, \beta) := \begin{cases} u & \text{if } u_1 \in \text{dom}(\sigma) \text{ and } \sigma(u_1)(r_1) = \{u\} \\
\bot & \text{otherwise} \end{cases} \)

- \(I[r_2(\text{expr}_1)](\sigma, \beta) := \begin{cases} \sigma(u_1)(r_2) & \text{if } u_1 \in \text{dom}(\sigma) \\
\bot & \text{otherwise} \end{cases} \)

Now needed: \(I[r(\text{expr}_1)]((\sigma, \lambda), \beta)\)

- We cannot simply write \(\sigma(u)(\text{role})\).
  Recall: \(\text{role}\) is (for the moment) not an attribute of object \(u\) (not in \(\text{atr}(C)\)).

- What we have is \(\lambda(r)\) (with \(r\), not with \(\text{role}\)) — but it yields a set of \(n\)-tuples, of which some relate \(u\) and other some instances of \(D\).

- \(\text{role}\) denotes the position of the \(D\)'s in the tuples constituting the value of \(r\).
Assume \( \text{expr}_1: \tau_C \) for some \( C \in \mathcal{C} \). Set \( u_1 := I[\text{expr}_1]((\sigma, \lambda), \beta) \in \mathcal{D}(\tau_C) \).

- \( I[\text{role}(\text{expr}_1)]((\sigma, \lambda), \beta) := \begin{cases} u, & \text{if } u_1 \in \text{dom}(\sigma) \text{ and } L(\text{role})(u_1, \lambda) = \{ u \} \\ \bot, & \text{otherwise} \end{cases} \)

- \( I[\text{role}(\text{expr}_1)]((\sigma, \lambda), \beta) := \begin{cases} L(\text{role})(u_1, \lambda), & \text{if } u_1 \in \text{dom}(\sigma) \\ \bot, & \text{otherwise} \end{cases} \)

where

\[ L(\text{role})(u, \lambda) = \{(u_1, \ldots, u_n) \in \lambda(\text{r}) \mid u \in \{u_1, \ldots, u_n\}\} \downarrow i \]

if

\[ \langle r : \ldots \langle \text{role}_1 : \ldots \rangle, \ldots \langle \text{role}_n : \ldots \rangle, \ldots \rangle, \text{role} = \text{role}_i \rangle \in \mathcal{V} \]

Given a set of \( n \)-tuples \( A \), \( A \downarrow i \) denotes the element-wise projection onto the \( i \)-th component.

---

**OCL and Associations Example**

\[ I[\text{role}(\text{expr}_1)]((\sigma, \lambda), \beta) := \begin{cases} L(\text{role})(u_1, \lambda), & \text{if } u_1 \in \text{dom}(\sigma) \\ \bot, & \text{otherwise} \end{cases} \]

\[ L(\text{role})(u, \lambda) = \{(u_1, \ldots, u_n) \in \lambda(\text{r}) \mid u \in \{u_1, \ldots, u_n\}\} \downarrow i \]

\[ \mathcal{V} = \{\ldots, \{A \subset C !, C : D \rightarrow \}, C : D \rightarrow \}, \ldots \} \]

\[ \sigma = \{1_C \mapsto \emptyset, 3_D \mapsto \{x \mapsto 1\}, 7_D \mapsto \{x \mapsto 2\}\} \]

\[ \lambda = \{A \subset C \rightarrow \{(1_C, 3_D), (1_C, 7_D)\}\} \]

\[ I[\text{self} . \text{n}]((\sigma, \lambda), \{\text{self} \mapsto 1_C\}) = I[I[\text{n}](\{A\subset C \rightarrow \}) \downarrow (\sigma, \lambda), \{\text{self} \mapsto 1_C\}) = L(\text{n})(1_C, 3_D) \]

\[ = \{\langle u, x \rangle \in \lambda(\text{A : C} !, \text{A : C} !) \mid \langle x \rangle \in \{x : 3_D\}\} \downarrow 2 \]

\[ = \{3_a, 3_b\} \]
Recapitulation: Consider the following association:

\[ \langle r : \langle \text{role}_1 : C_1, \mu_1, P_1, \xi_1, \nu_1, o_1 \rangle, \ldots, \langle \text{role}_n : C_n, \mu_n, P_n, \xi_n, \nu_n, o_n \rangle \rangle \]

- **Association name** \( r \) and **role names/types** \( \text{role}_i/C_i \) induce extended system states \( \lambda \).
- **Multiplicity** \( \mu \) is considered in OCL syntax.
- **Visibility** \( \xi \)/**Navigability** \( \nu \): well-typedness.

Now the rest:

- **Multiplicity** \( \mu \): we propose to view them as constraints.
- **Properties** \( P_i \): even more typing.
- **Ownership** \( o \): getting closer to pointers/references.
- **Diamonds**: exercise.
Rhapsody Demo

References