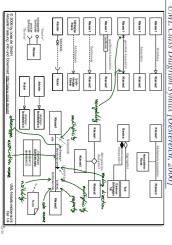
## Software Design, Modelling and Analysis in UML Prof. Dr. Andreas Podelski, Dr. Bernd Westphal Albert-Ludwigs-Universität Freiburg, Germany Lecture 08: Class Diagrams II 2013-11-20

not in UML Class Diagram Syntax [Oestereich, 2006] THE PROPERTY OF THE PROPERTY O Open Control Vi Non-set Acceptation (Acceptation) the State of Specific St. laster is

# UML Class Diagram Syntax [Oestereich, 2006]



## Contents & Goals

## Last Lectures:

class diagram — except for associations; visibility within OCL type system

## This Lecture:

- Educational Objectives: Capabilities for following tasks/questions.
  Please explain this class diagram with associations.
  Which amountaions of an association arous are semantically relevant?
  What's a role nam? What's it good for?
  What's "andisplicity"? How did we treat them semantically?
  What is "nading direction," "awayallary," "ownership"...?
  What's the difference between "aggregation" and "composition"?

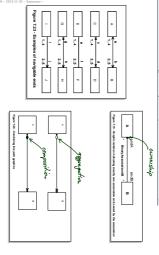
- Study concrete syntax for "associations".
   Rudgos Ly Dewo
   (Temporarily) extend signature, define mapping from diagram to signature.
   Study effect on OCL.
- Where do we put OCL constraints?

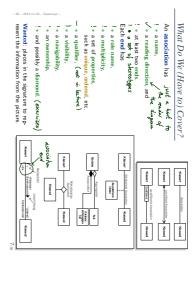
2/90

3/50

Associations: Syntax

UML Class Diagram Syntax [OMG, 2007b, 61;43]





# (Temporarily) Extend Signature: Basic Type Attributes

## Also only for the course of this lecture

- $\bullet$  we only consider basic type attributes to "belong" to a class (to appear in atr(C)).
- associations are not "owned" by a particular class (do not appear in atr(C)), but live on their own.

## Formally: we only call

# a signature (extended for associations) if

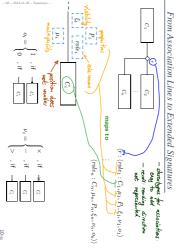
 $atr: \mathcal{C} \to 2^{\{v \in V \mid v: \tau, \tau \in \mathcal{F}\}}$ .  $(\mathcal{T},\mathcal{C},V,atr)$ 

# (Temporarily) Extend Signature: Associations

Only for the course of Lectures 08/09 we assume that each attribute in V• either is  $\langle v:\tau,\xi,expr_0,P_v\rangle$  with  $\tau\in\mathcal{F}$  (as before), or is an association of the form •  $r,\ note_i$  are just names,  $C_i\in \mathscr{C},\ 1\leq i\leq n,$ • the multiplicity  $\mu_i$  is an expression of the form n ≥ 2 (at least two ends),  $\mu ::= \bigstar | \ N \ | \ N_{\bullet}M \ | \ N_{\bullet} \ast \ | \ \mu_{\bullet}\mu$  $(r:(role_1:C_1,\mu_1,P_1,\xi_1,\nu_1,o_1),$  $\langle role_n : C_n, \mu_n, P_n, \xi_n, \nu_n, o_n \rangle \rangle$ the class where this association aud is located  $(N, M \in \mathbb{N})$ 

•  $P_i$  is a set of properties (as before), •  $\xi \in \{+, -, \#, \sim\}$  (as before), •  $\nu_i \in \{\times, -, >\}$  is the navigability, •  $o_i \in \mathbb{B}$  is the ownership.

0/90



## Association Example

 $\mathcal{S} = (\{\text{lut}\}, \{\mathcal{C}, \mathcal{D}\}, \{x: \text{lut},$ {CHB, only basic type DH {x}}) absiliables have <r: <c: C, O. \*, Ø, -, X, 12, (n.D, 0.\*, 0,+,>, 0>>},

o<sub>i</sub> ∈ B is the ownership.

(Temporarily) Extend Signature: Associations

Only for the course of Lectures 08/09 we assume that each attribute in V

• either is  $\langle v: \tau, \xi, expr_0, P_v \rangle$  with  $\tau \in \mathcal{F}$  (as before),

or is an association of the form

Alternative syntax for multiplicities  $\mu ::= N..M \mid N..* \mid \mu, \mu$  $(r : (role_1 : C_1, \mu_1, P_1, \xi_1, \nu_1, o_1),$  $(N, M \in \mathbb{N}, \mathbb{L}(X))$ 

• r,  $rove_t$  are just names,  $\psi_t \in v$ ,  $t \leq t \leq n$ , • the multiplicity  $\mu_t$  is an expression of the form Note: N could abbreviate 0..N, 1..N, or N..N. We use last one. and define st and N as abbreviations.

 $\mu ::= \ast \mid N \mid N..M \mid N..\ast \mid \mu,\mu$ 

P<sub>i</sub> is a set of properties (as before).
ξ ∈ {+, -, #, ~} (as before).
ν<sub>i</sub> ∈ {×, -, >} is the navigability.  $(N, M \in \mathbb{N})$ 

8/50

11/50

## What If Things Are Missing?

Most components of associations or association end may be omitted. For instance [OMG, 2007b, 17]. Section 6.4.2, proposes the following rules:

Name: Use

if the name is missing.  $A_{-}\langle C_1 \rangle_{-} \cdots _{-} \langle C_n \rangle$ 

 Reading Direction: no default. C ACD D

Role Name: use the class name at that end in lower-case letters

Other convention: (used e.g. by modelling tool Rhapsody) profes colds were with C  $\frac{itsC - itsD}{D}$  D for for 0 D

12/50

## What If Things Are Missing?

Properties: ∅

Navigability and Ownership: not so easy. [OMG, 2007b, 43]

"Various options may be chosen for showing navigation arrows on a diagram.

Suppress arous for associations with nonigability in both directions, and show a rows only for associations with one-way marigability. In this case, the two-way navigability cannot be distringuished from situations where there is no navigation at all however, the latter case occurs rarely in practice."

Visibility: public

13/90

• Multiplicity: 1  $nm = 0... \text{ or *} \frac{it}{i!} \text{ there are no fixed, written,} \\ nm y opinion, it's safer to assume 0... \text{ or *} \frac{it}{i!} \text{ there are no fixed, written,} \\ agreed conventions ("expect the worst").$ 

In practice, it is often convenient to suppress some of the arrows and crosses and just show exceptional situations:

Show all arrows and x's. Navigation and its absence are made completely explicit.
 Suppress all arrows and x's. No inference can be drawn about navigation.
 This is similar to any situation in which information is suppressed from a view.

## Wait, If Omitting Things...

...is causing so much trouble (e.g. leading to misunderstanding),
 why does the standard say "In practice, it is often convenient..."?

Is it a good idea to trade convenience for precision/unambiguity?

It depends.

Convenience as such is a legitimate goal.

In UML-As-Sketch mode, precision "doesn't matter", so convenience (for writer) can even be a primary goal.

In UML-As-Blueprint mode, precision is the primary goal.
 And misunderstandings are in most cases annoying.

But: (even in UML-As-Blueprint mode)
If all associations in your model have multiplicity \*,
then it's probably a good idea not to write all these \*s.
So: tell the reader about it and leave out the \*'s.

14,50

Association Semantics

Rhapsody Demo

What's left? Named association with at least two typed ends, each having

Overview

 a multiplicity, a role name, a visibility, a set of properties,
 a navigability, and an ownership.

Extend system states, introduce so-called links as instances of associations — depends on name and on type and number of ends.

Integrate role name and multiplicity into OCL syntax/semantics.

Extend typing rules to care for visibility and navigability

Consider multiplicity also as part of the constraints set Inv(CD).

Properties: for now assume P<sub>v</sub> = {unique}.

Properties (in general) and ownership: later.

15/90

16/50

Association Semantics: The System State Aspect

5-{15-> fauto), 25 > fautos, 35 + sauso, 22 + fautos}

OBJECT DIAGRAMS:

Association/Link Example



 $\mathscr{S}=(\{Int\},\{C,D\},\{x:Int,$  $\langle A\_C\_D: \langle c:C,0..*,+, \{\mathtt{unique}\}, \times, 1 \rangle,$  $\{C \mapsto \emptyset, D \mapsto \{x\}\})$  $\langle n:D,0..*,+,\{\mathtt{unique}\},>,0\rangle\rangle\},$ 

A system state of  ${\mathscr S}$  (some reasonable  ${\mathscr D}$ ) is  $(\sigma,\lambda)$  with:  $\sigma = \{1_C \mapsto \emptyset, 3_D \mapsto \{x \mapsto 1\}, 7_D \mapsto \{x \mapsto 2\}\}$  $\lambda = \{A\_C\_D \mapsto \{(1_C, 3_D), (1_C, 7_D)\}\}$ 

20/50

ES M COS

355 m to the head hyperdyn

WE WILL NOT BOWALLY DEPINE THY

Associations in General

Recall: We consider associations of the following form:

 $\langle r: \langle role_1: C_1, \mu_1, P_1, \xi_1, \nu_1, o_1 \rangle, \dots, \langle role_n: C_n, \mu_n, P_n, \xi_n, \nu_n, o_n \rangle \rangle$ 

(recall: we assume  $P_1=P_n=\{\mathtt{unique}\}$ ).  $\langle r : \langle role_1 : C_1, \neg P_1, \neg \neg \neg \rangle, \dots, \langle role_n : C_n, \neg P_n, \neg \neg \neg \neg \rangle \rangle$  Only these parts are relevant for extended system states:

The UML standard thinks of associations as **n-ary relations** which "live on their own" in a system state.

That is, links (= association instances)

do not belong (in general) to certain objects (in contrast to pointers, e.g.)

are "first-class citizens" next to objects,

are (in general) not directed (in contrast to pointers).

17/50

18/90

Links in System States

 $\langle r:\langle role_1:C_1,...,P_1,...,...\rangle,...,\langle role_n:C_n,...,P_n,...,...\rangle$ 

Only for the course of lectures 08/08 we change the definition of system states:

A system state of  $\mathscr S$  wrt.  $\mathscr D$  is a <u>pair</u>  $(\sigma,\lambda)$  consisting of  $\sigma$  a type-consistent mapping  $\sigma: \mathscr D(\mathscr S) \to (atr(\mathscr S) \to \mathscr D(\mathscr S)),$ Definition. Let  $\mathscr D$  be a structure of the (extended) signature  $\mathscr S=(\mathscr S,\mathscr E,V,atr).$ for associations

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

19,50

Extended System States and Object Diagrams

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?

21,50

References

[Oestereich, 2006] Oestereich, B. (2006). Analyse und Design mit UML 2.1, 8. Auflage. Oldenbourg, 8. edition.

[OMG, 2007a] OMG (2007a), Unified modeling language: Infrastructure, version 2.1.2. Technical Report formal/07-11-04.

[OMG, 2007b] OMG (2007b). Unified modeling language: Superstructure version 2.1.2. Technical Report formal/07-11-02.

50/50