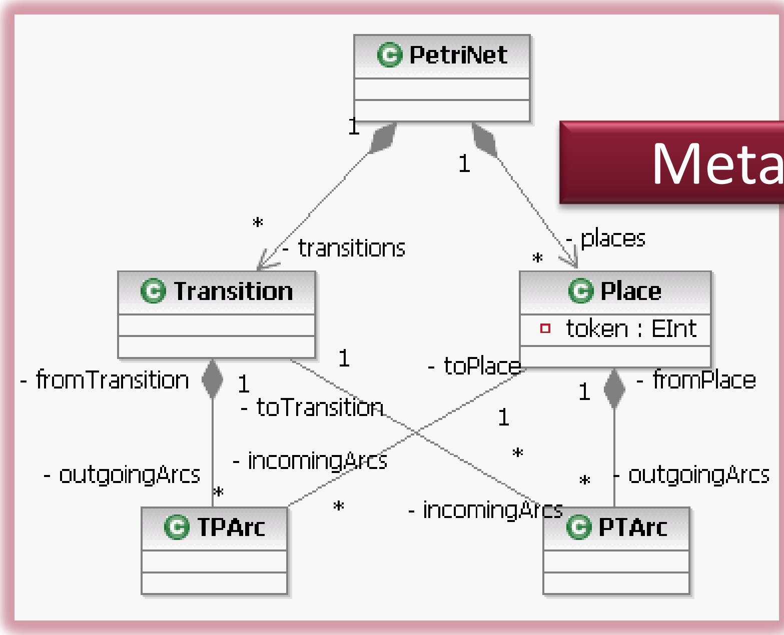
Domain Modeling

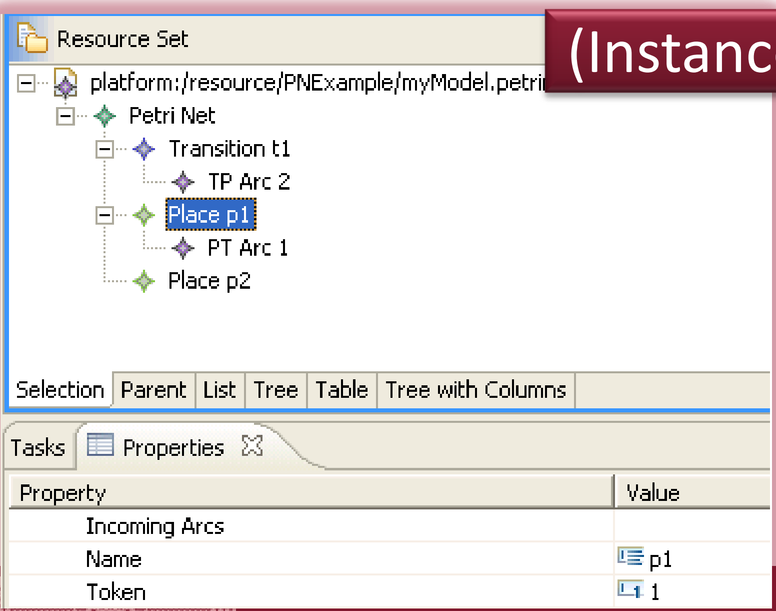
Model Driven Systems Engineering

Lecture 2

Metamodel: Specify Concepts an Appl. Domain



Metamodel



Resource Set

- platform:/resource/PNExample/myModel.petri
  - Petri Net
    - Transition t1
      - TP Arc 2
    - Place p1
      - PT Arc 1
    - Place p2

Property	Value
Incoming Arcs	
Name	p1
Token	1

(Instance) Model

- Metamodel:
  - Precise specification of domain concepts
  - A language for defining the abstract syntax of a DSM
- Goal: to define...
  - Basic concepts
  - Relations between concepts
  - Attributes of concepts
  - Abstraction / refinement (Taxonomy, Ontology) between model elements
  - Aggregation
  - Multiplicity restrictions
  - Derived features

Metamodels and instance models

Reference / Association

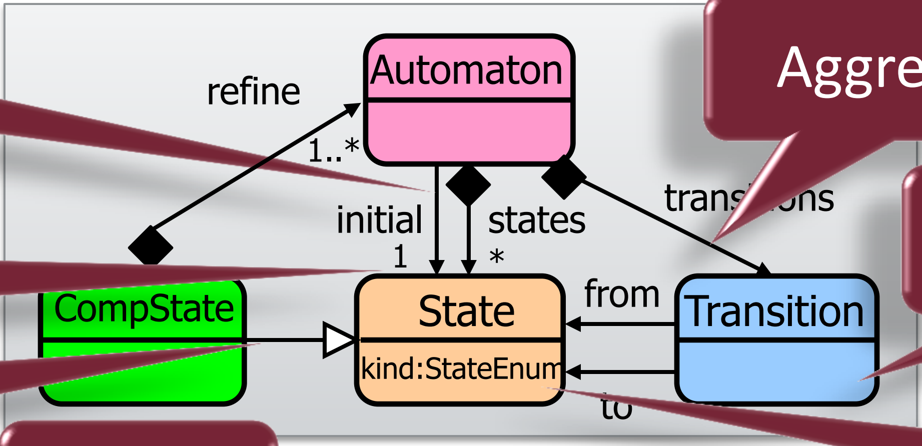
Multiplicity

Generalization

Aggregation

Class

Attribute



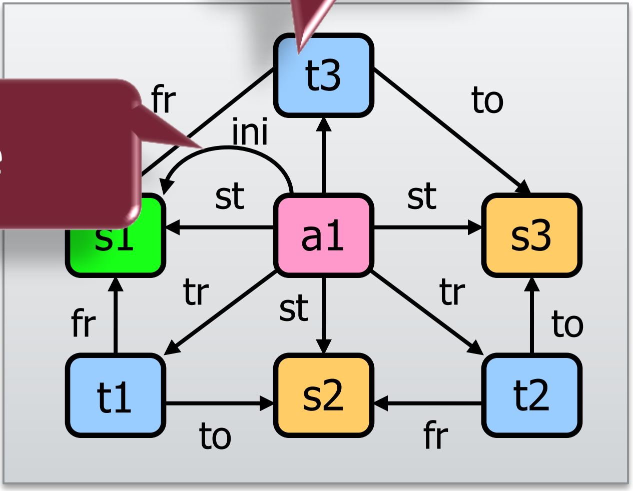
Metamodel

Metamodel level

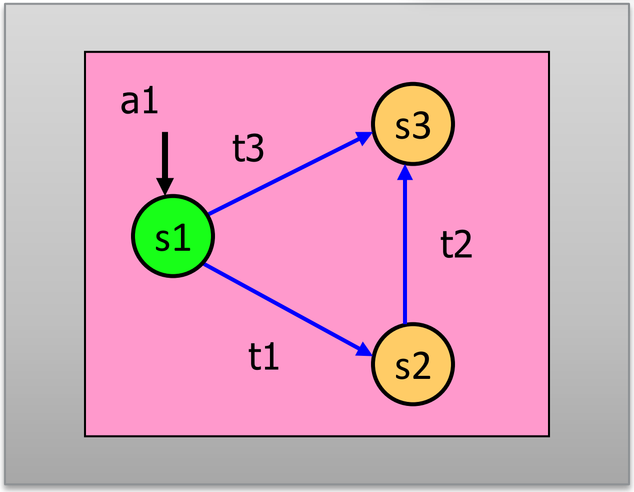
Model level

Object

Role



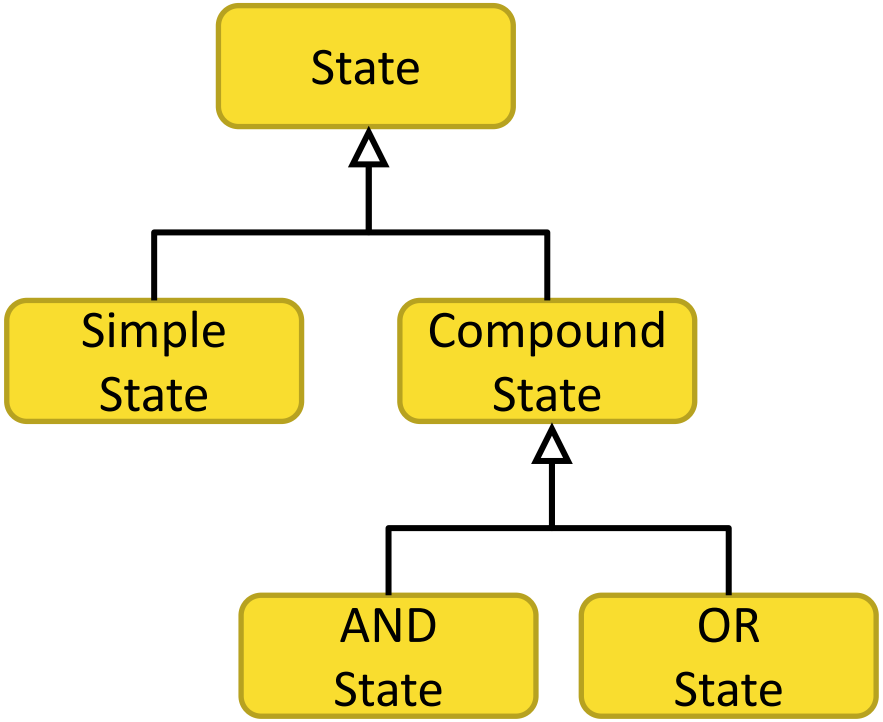
Abstract syntax



Concrete syntax

Classes and Objects

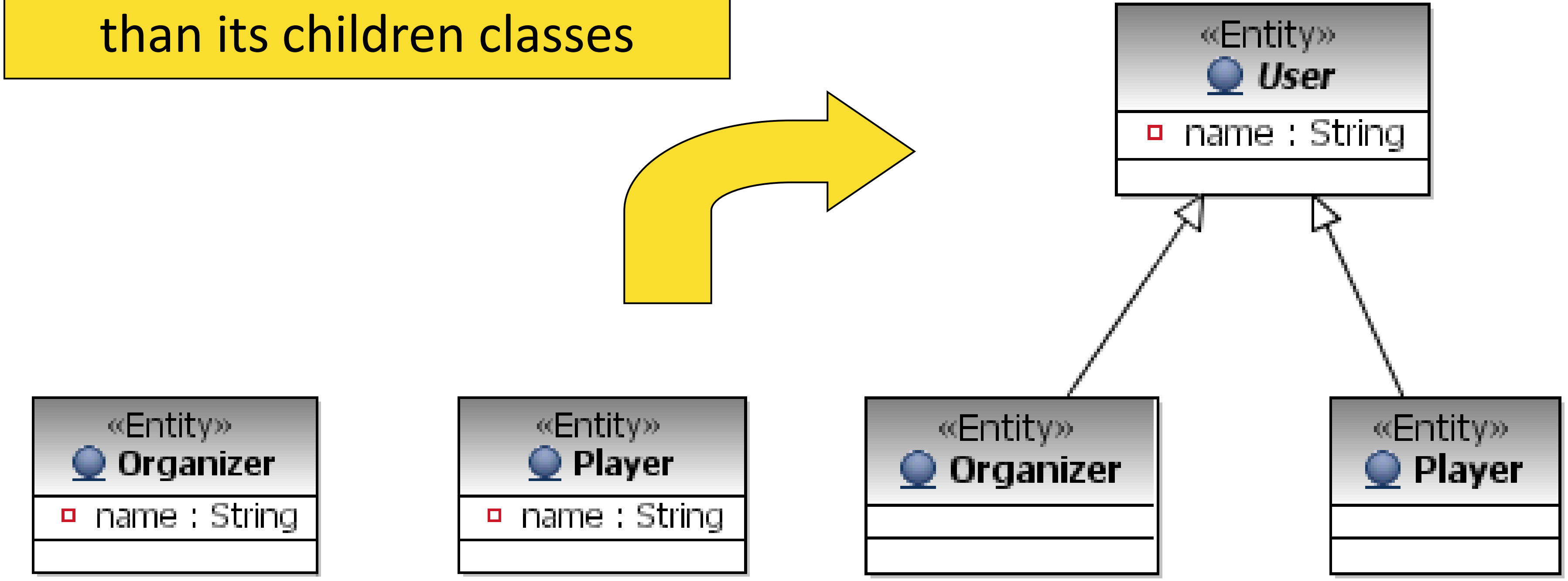
Type hierarchy



- Generalization
  - ≅Inheritance
  - Transitive
  - Reflexive? / Irreflexive?
- How to read?
  - ***SimpleState*** is a subclass of ***State***
  - ***State*** is a superclass of ***SimpleState***
- Substitutability
  - ✓ Subclass instead of Superclass
  - × ~~Superclass instead of Subclass~~

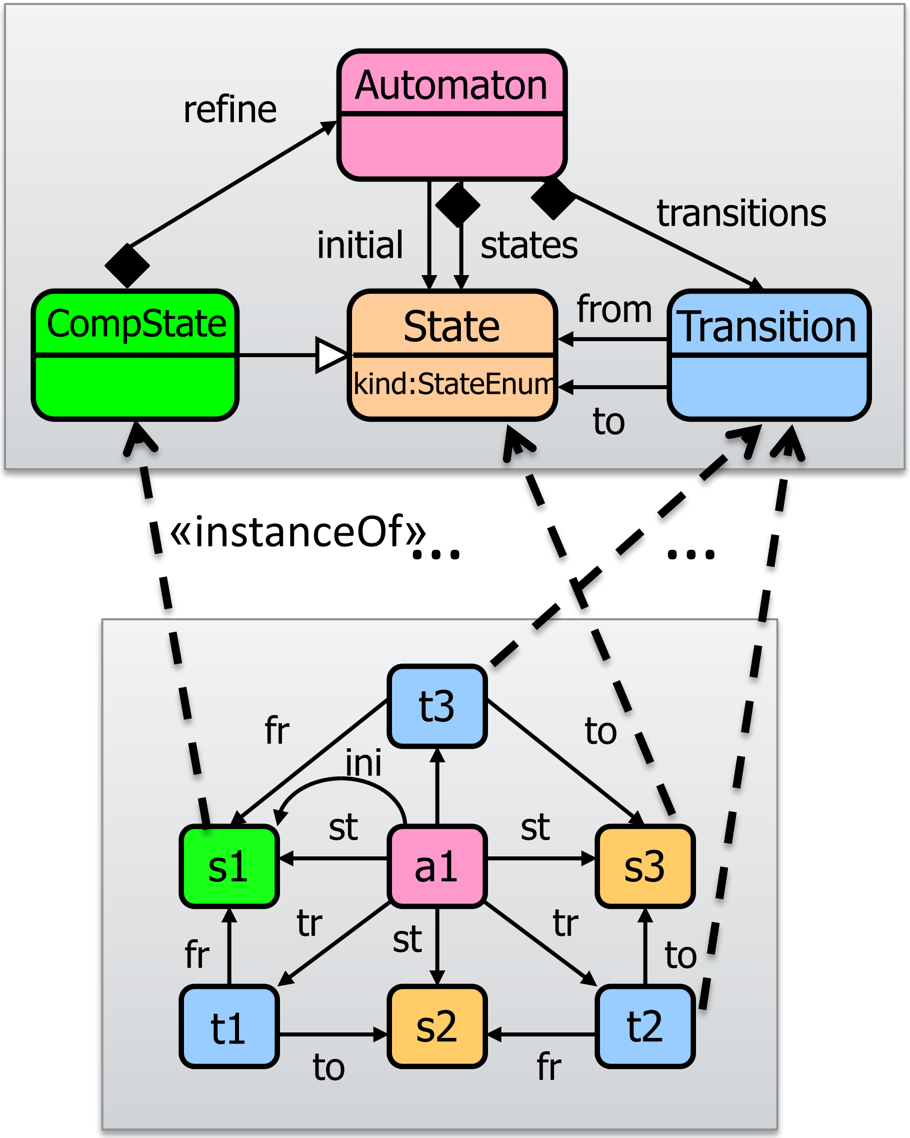
Typical Use of Generalization

Parent class is more general than its children classes



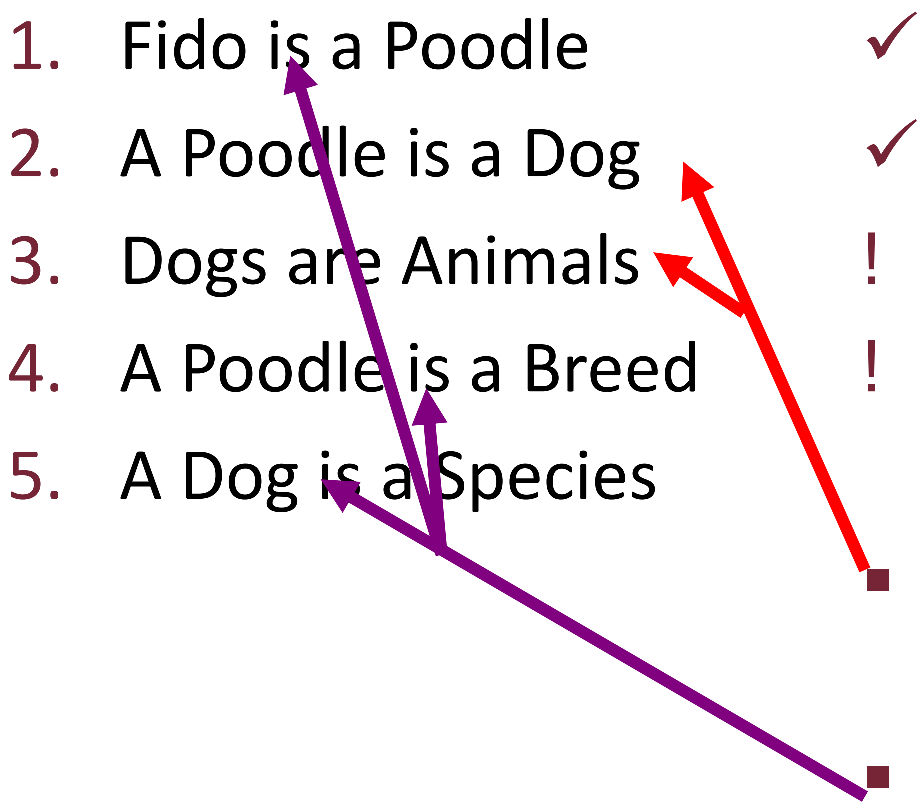
Aim: Lift up common features to the superclass

Type conformance /Instantiation /Classification



- Each model element is ***an instance of*** *(conforms to)* a metamodel element
- **Direct type**:
  - No other type exists lower in the type hierarchy
  - s1 → CompState
- **Indirect type**:
  - Superclass of the direct type
  - s1 → State

Classification vs. Generalization



1. Fido is a Poodle
2. A Poodle is a Dog
3. Dogs are Animals
4. A Poodle is a Breed
5. A Dog is a Species

✓ 1+2 = Fido is a Dog

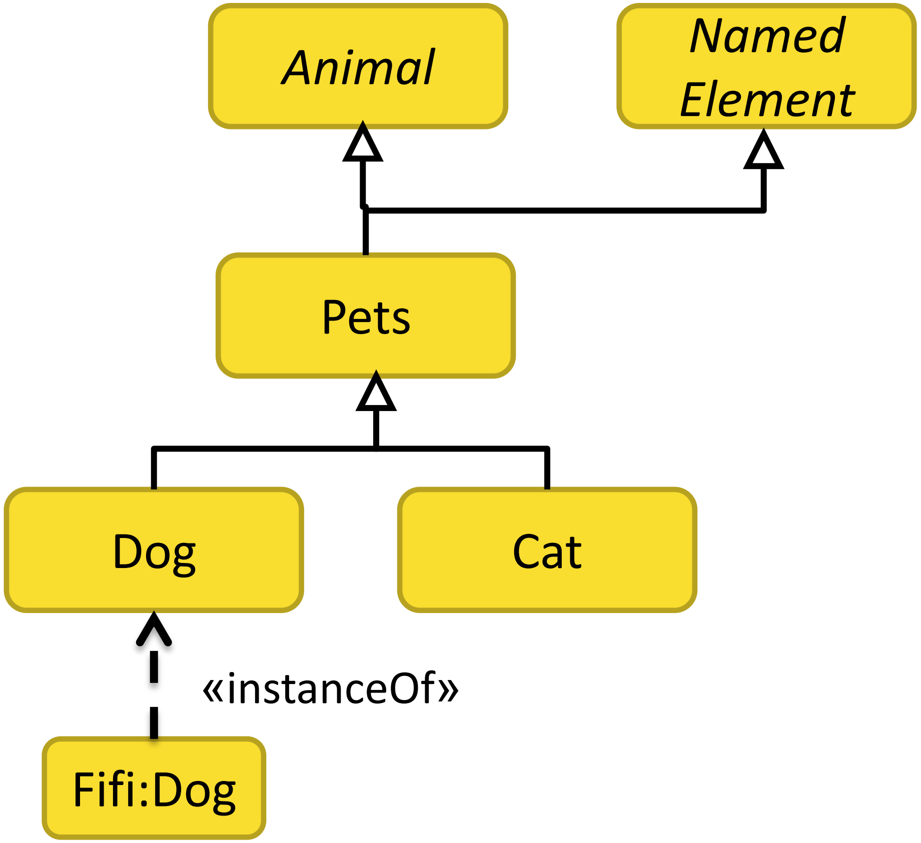
✓ 1+2+3 = Fido is an Animal

! 1+4 = Fido is a Breed

! 2+5 = A Poodle is a Species

- Generalization (SupertypeOf) is transitive
- Classification (InstanceOf) is NOT transitive

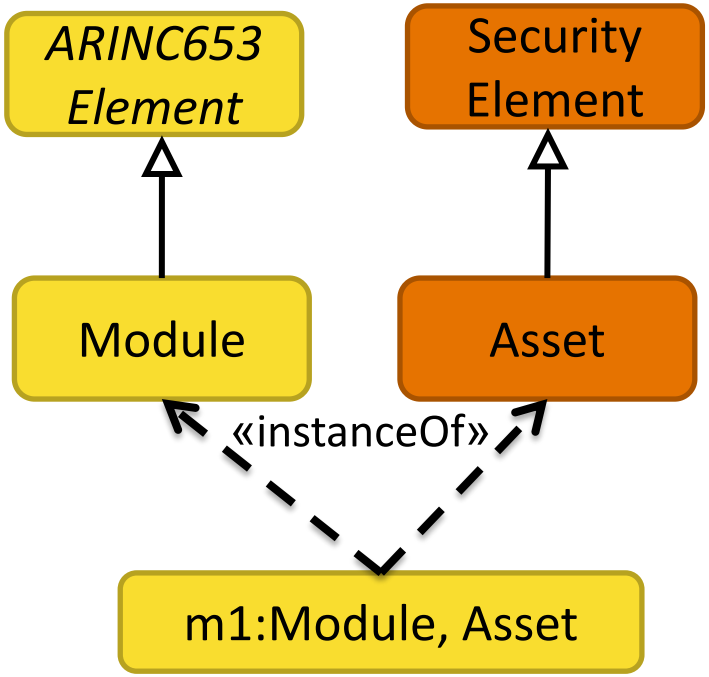
Multiple inheritance



How many types does Fifi have?

- Multiple inheritance:
  - A class in the metamodel has more than 1 supertype
  - Typical use: merge features from different classes
    - One is generic, thus reused in different domains (cf. NamedElement)
    - Other is a general but domain-specific superclass (cf. Animal)
- Restriction:
  For each model element: a single type

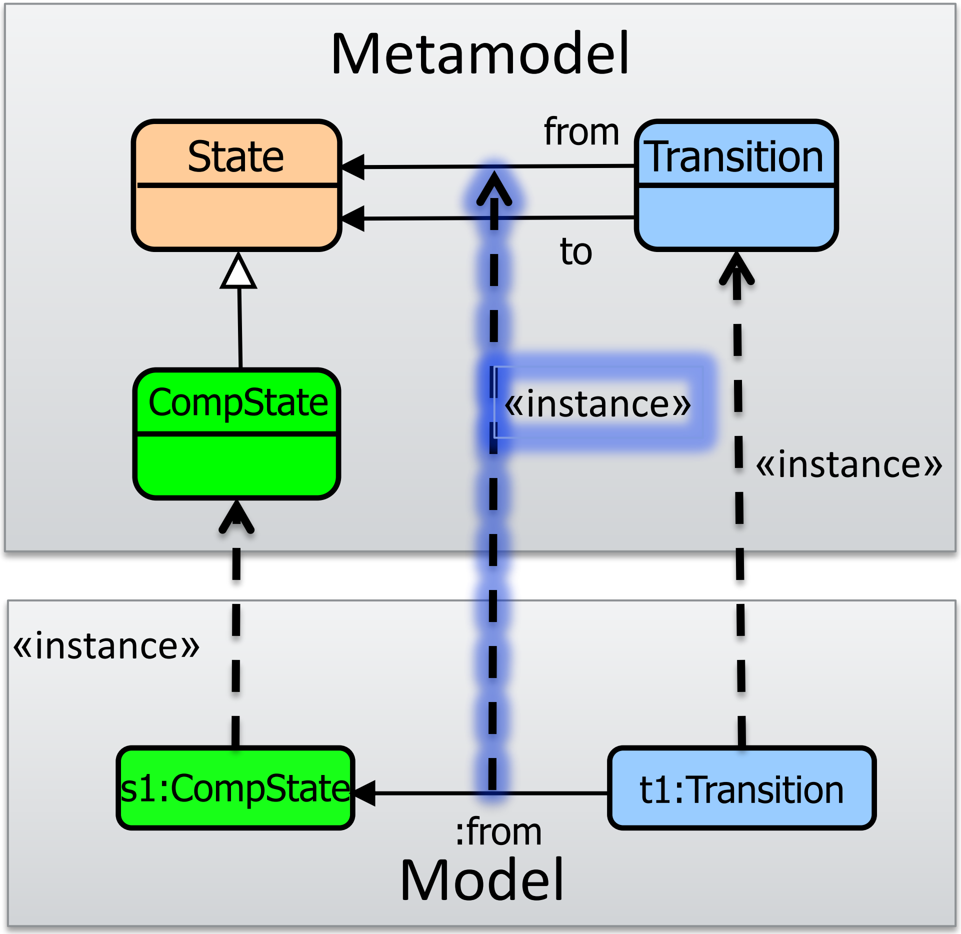
Multiple classification



- Multiple typing / classification:
    - One model element typed against multiple metamodels
    - Rationale: Multi-paradigm / view modeling
    - UML Stereotypes
- Restriction: For each model element: a single type in a domain

References and Links

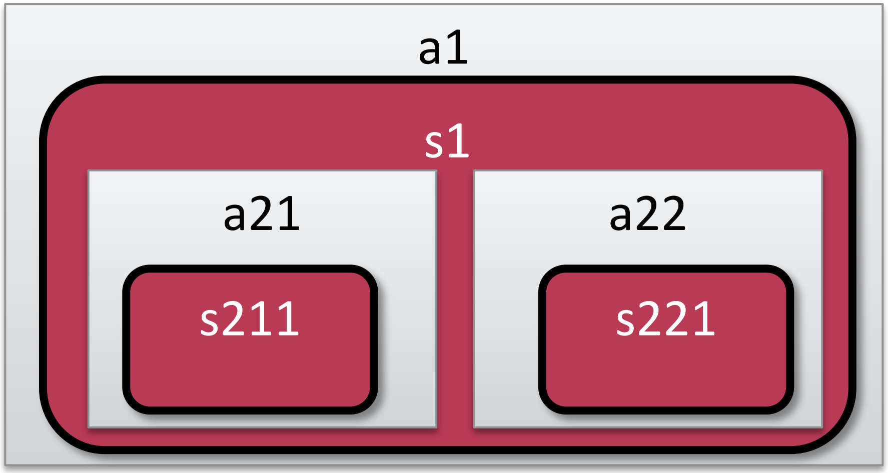
Type conformance of references

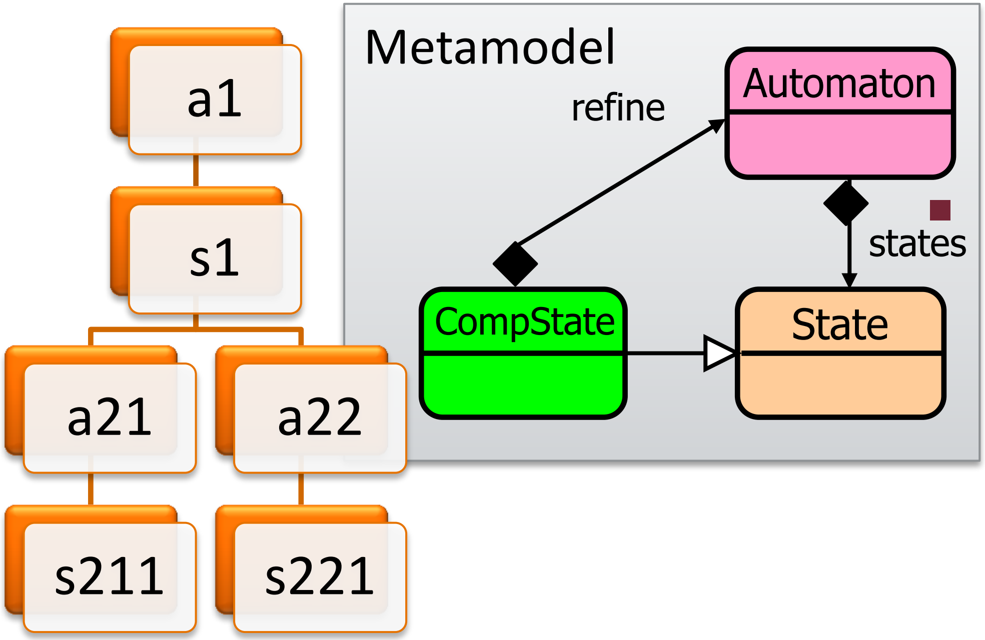


Can you define generalization for references?

- A link in a model is **type conformant** if
  - ***type(src(link))*** is subtype of ***src(type(link))***
  - ***type(trg(link))*** is subtype of ***trg(type(link))***
  - Informally:
    - The type of the source object is a subtype of the source class of the link's type.
    - The type of the target object is a subtype of the target class of the link's type.

Containment hierarchy

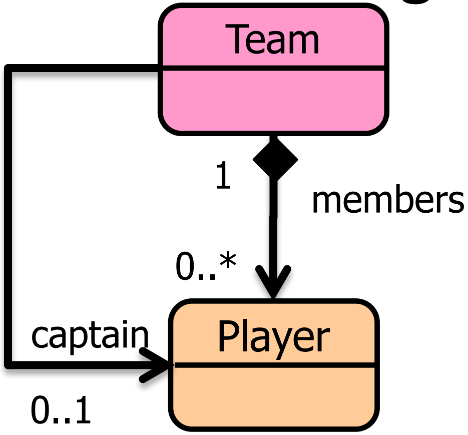




- Each model element has a unique parent
  - N children → 1 parent
  - Single root element
- Aggregation as relationship:
  - Defined in the metamodel along reference edges
  - Provides restriction for instance models
- Circularity
  - No circular containment (in the model)
  - Aggregation relations in the metamodel may be circular (hierarchy)

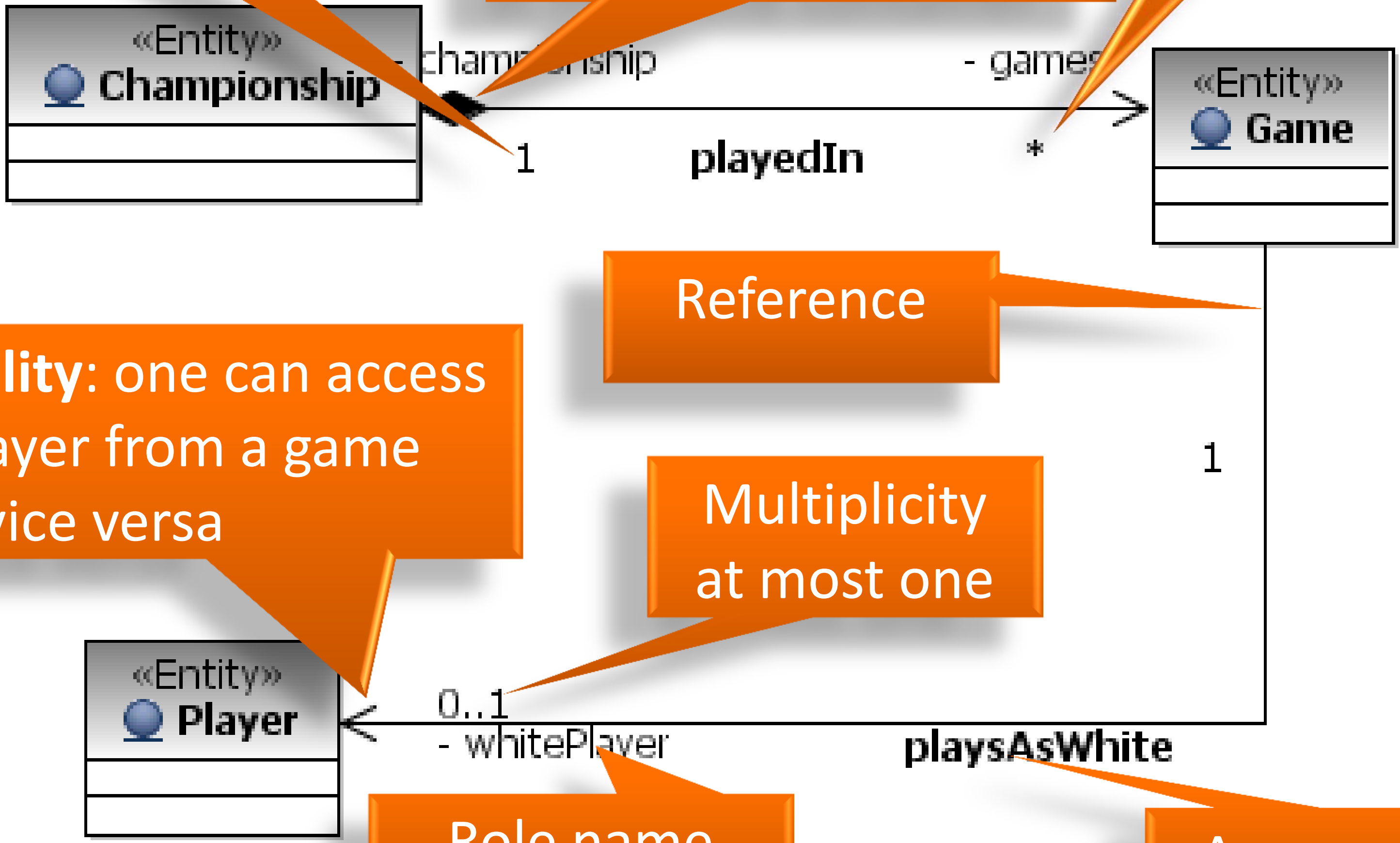
Multiplicity restrictions

- Definition: Lower bound .. Upper bound
  - Lower bound: 0, 1, (non-negative integer)
  - Upper bound: 1, 2, … * (positive integer + any)
- Scope:
  - References: allowed number of links between objects of specific types
  - Attributes: e.g. arrays of strings (built-in values)



Which are the most common multiplicity definitions in practice?

Notation Guide



Multiplicity should be 1 for aggregation

Composition: at most one container

Multiplicity many

Reference

Navigability: one can access white player from a game but not vice versa

Multiplicity at most one

Role name

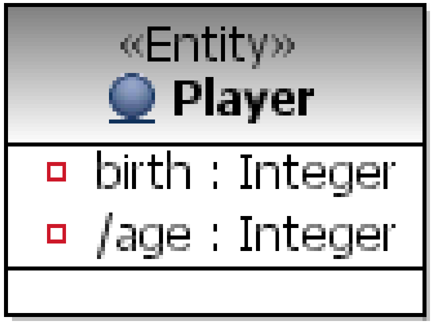
Assoc. name

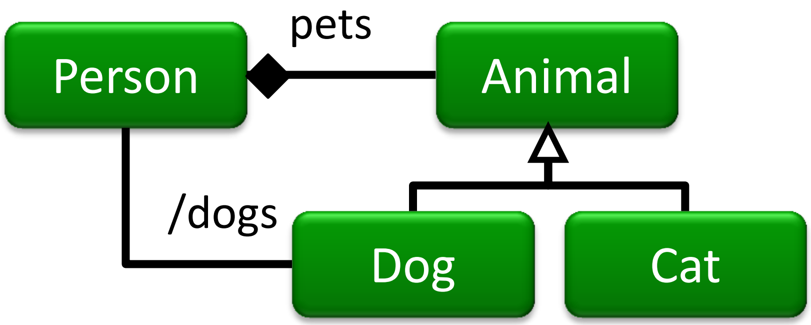
Advanced Concepts and Best Practices

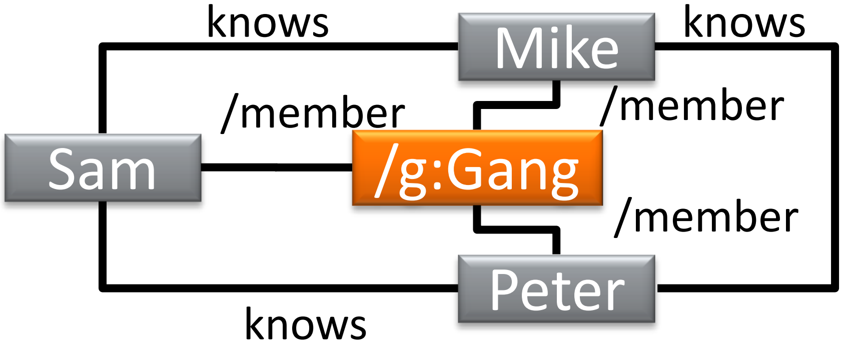
In Domain Modeling

Derived Features

- A derived feature can be calculated from others
    - Usage: helpers for designers / tools
    - It need not be persisted
    - Automatic updates
- Derived attributes:
  `age = currYear – birth`
- Derived references:
  `dogs =  -- pets --> Dog`
- Derived objects:
    - „Gang”:
      everyone knows everyone

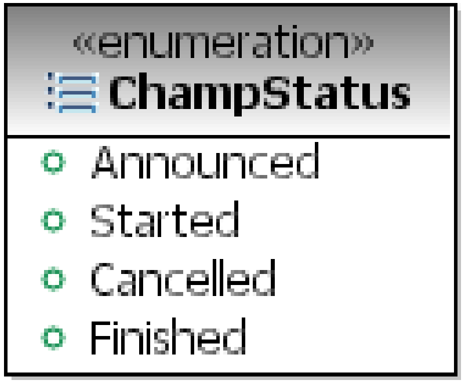






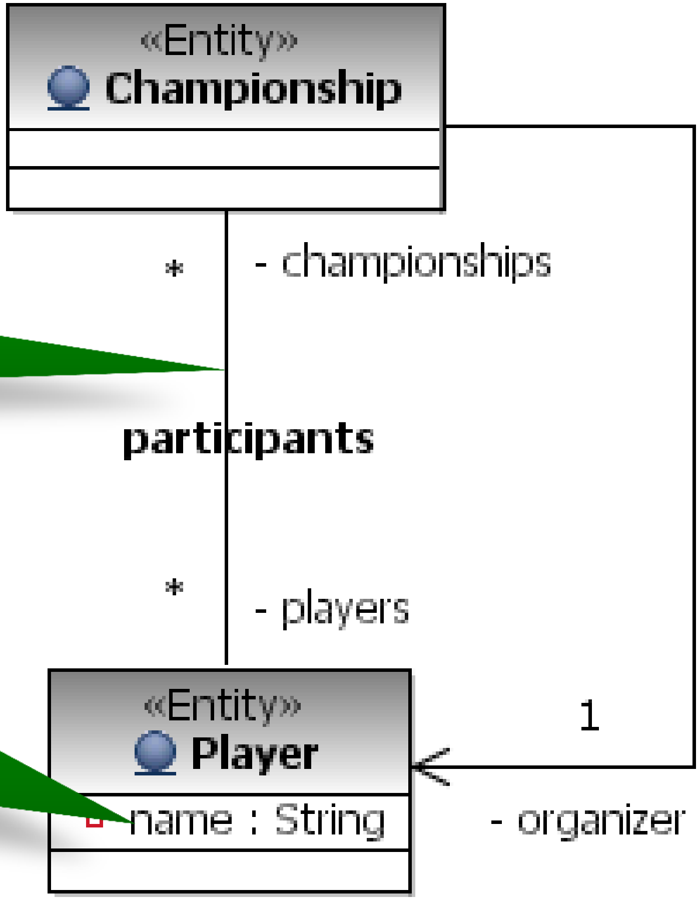
Enumerations

- Enumeration:
  - a fixed set of symbolic values
  - represented as a class with values as attributes



- Usage:
  - Frequently define possible states
  - Use enumerations instead of hard-wired String literals whenever possible
  - Can be better than Boolean – extensibility
  - Use as attribute types

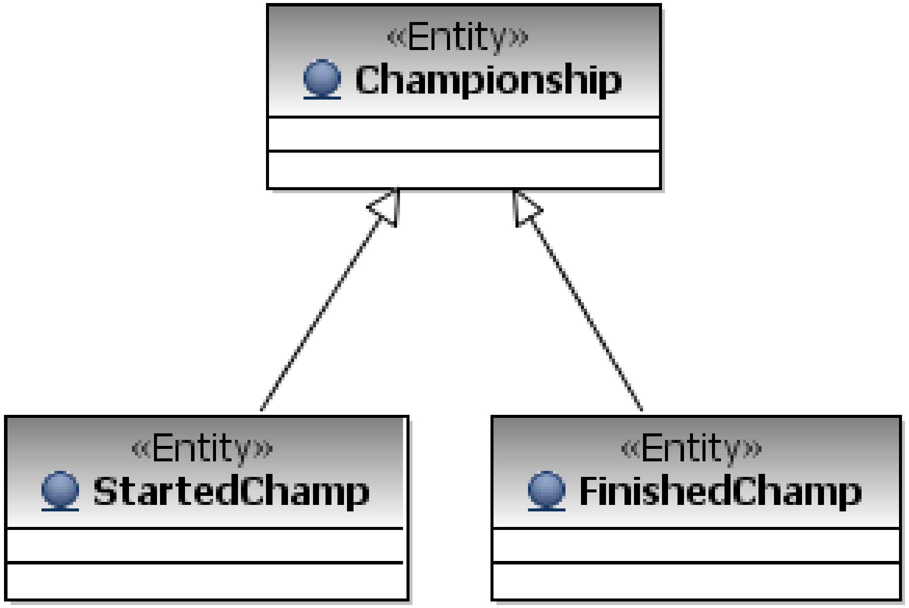
Built-in classes vs. User defined classes

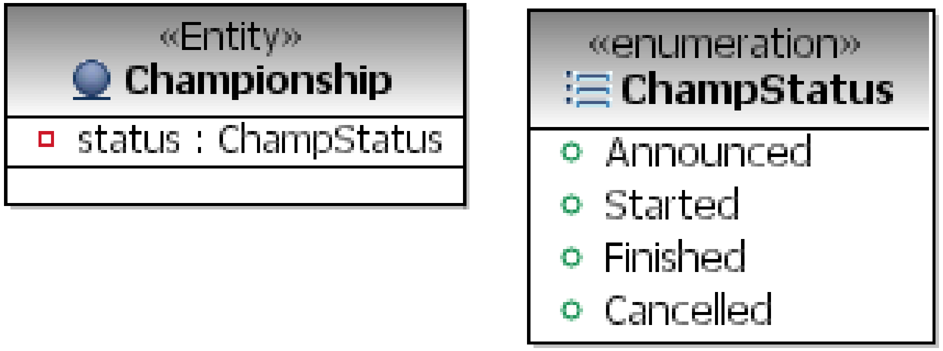


User classes:
Associations

Built-in classes +
Enumeration Type:
Attributes

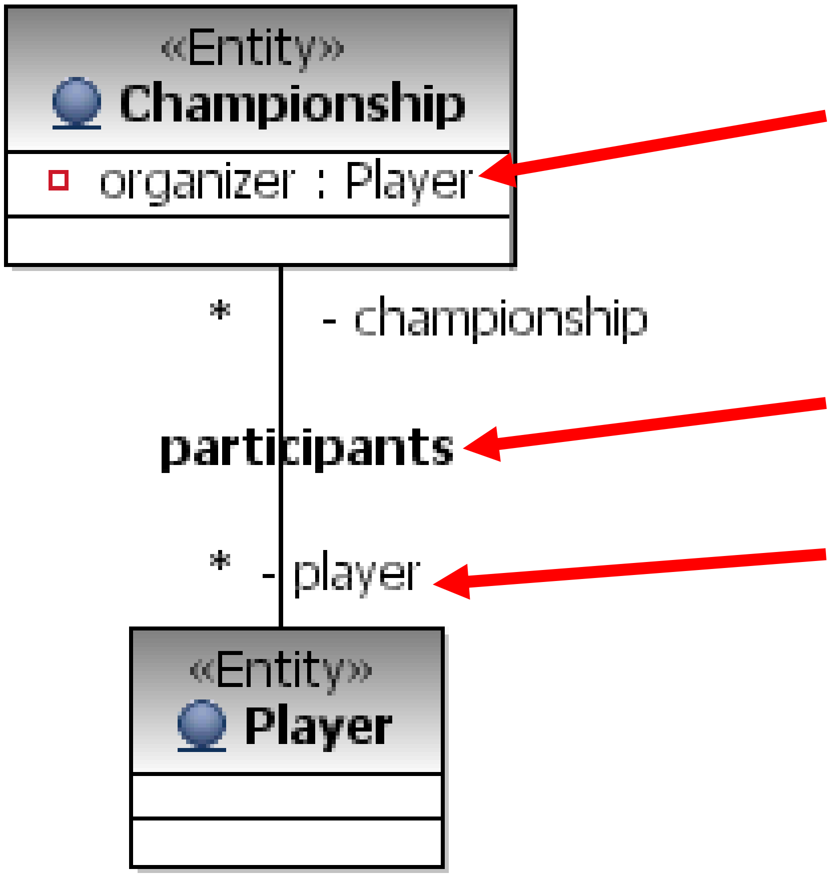
When to avoid generalization?





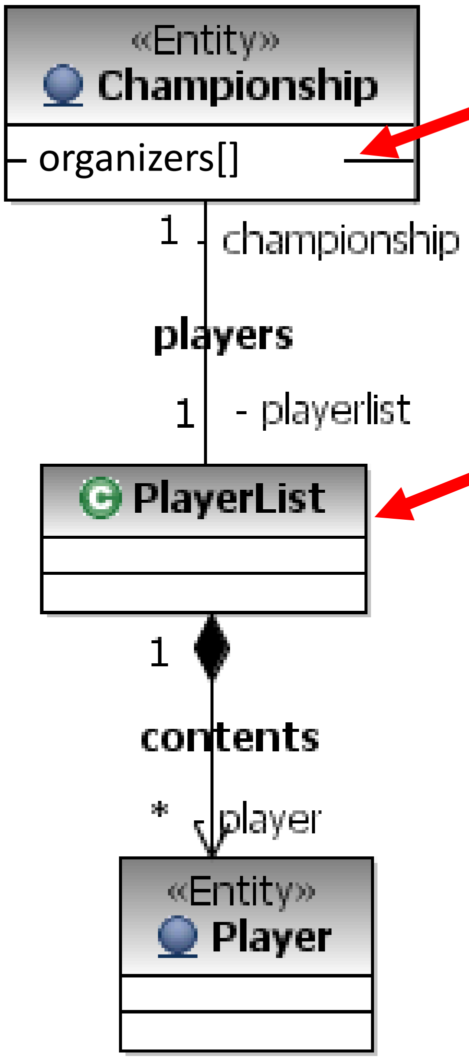
- What happens if a started championship is finished?
- Problem: Retyping of an object is required
- NOTE:
  Use status attribute with enumeration values to store the state of an object that can change

What is Bad Design/Smell here?



- Properties of a user defined type (class) should rather be denoted explicitly
  - OK, if multiplicity is 1
- Naming of associations:
  - prefer verbs to nouns
  - OK: *participatesIn*, *participantsOf*
- Naming of roles:
  - 1: singular
  - *: plural
  - OK: *players*, *championships*

What is Bad Design/Smell here?



- Arrays in attributes
  - Solution:
    an *organizes* association
- Explicit lists
  - Solution:
    a single *playsIn* association
- NOTE:
  Lists and arrays are programming constructs and not domain elements!

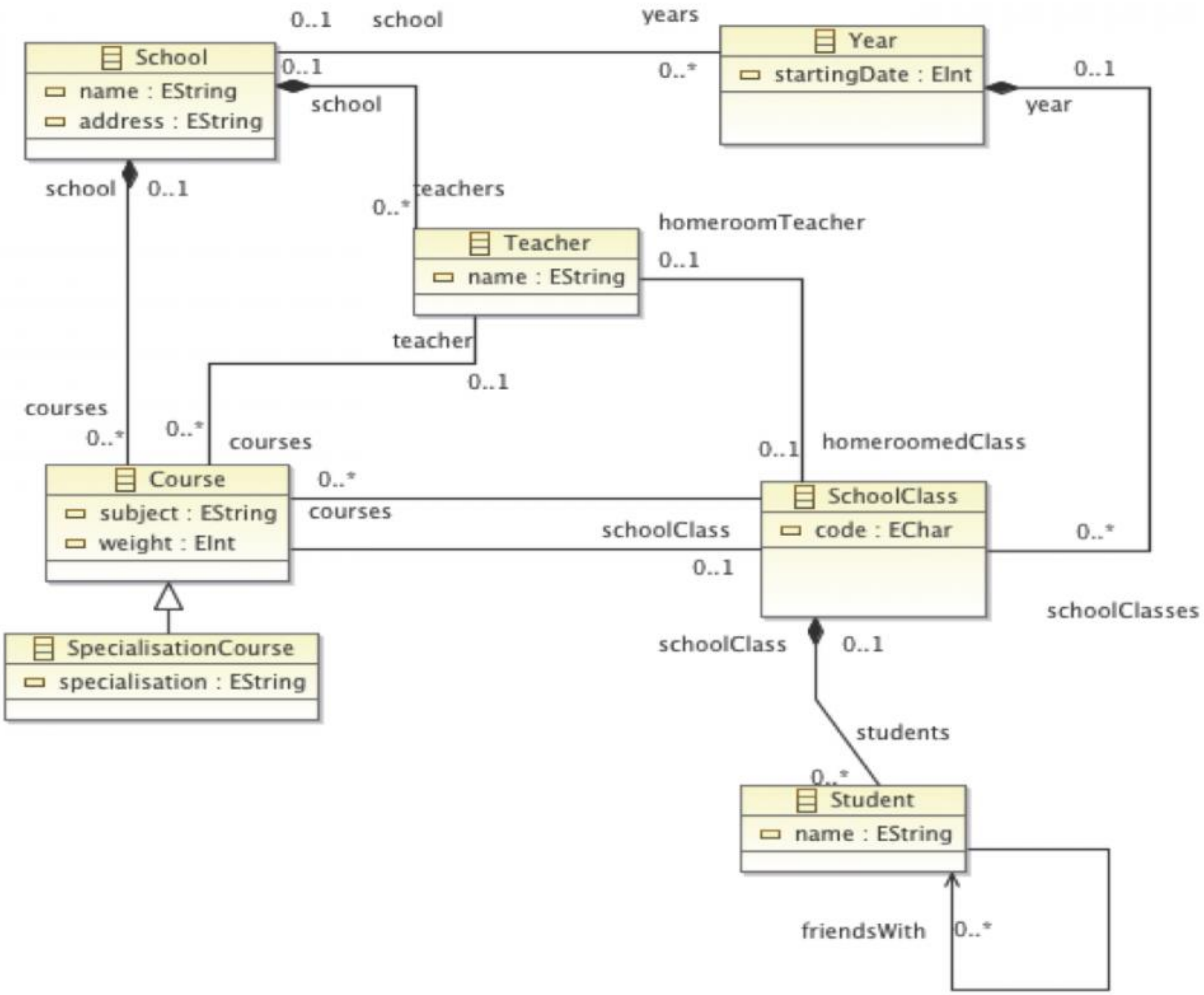
Domain Modeling Examples

Practical exercises

The School Domain

- A school (identified by its name and address) has teachers as employees who teach courses (identified by their subject) in different years.
- Each class in a specific school year has a headmaster (homeroom) teacher
- Students of a specific year attend their own classes, and they may be friends with each other
- Teachers and students are identified by their names.
- Specialization courses can be taken by 11th and 12th grade students

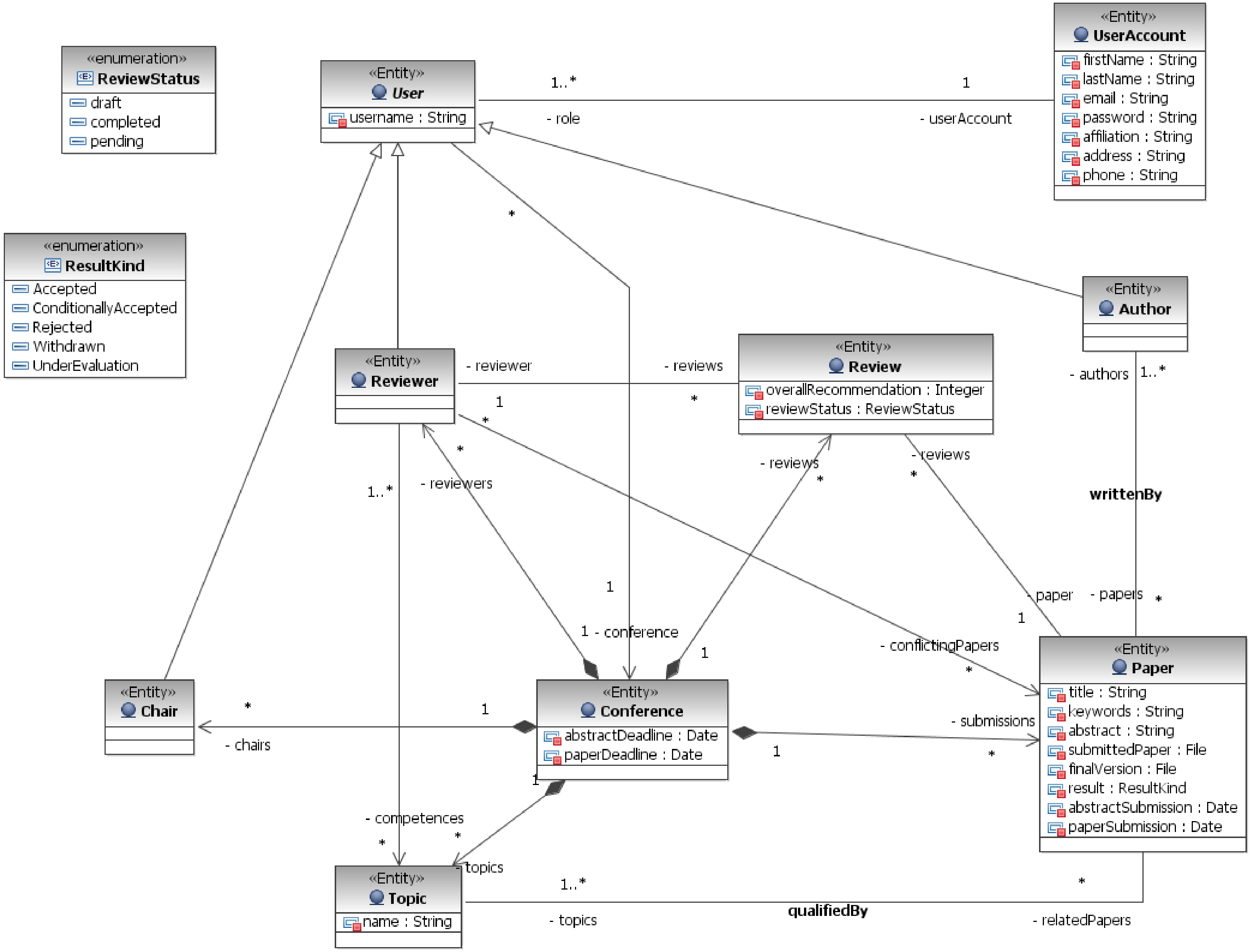
The School Domain



Paper Review System: The Story

- The paper review system is used by authors who log in electronically for the conference and then fill in a form including their name, the most important attributes of the paper to be submitted (such as title, abstract), and mark the conference topics related to the paper. The paper itself is usually submitted by a later deadline using the paper ID received when registering the paper. Later the authors may observe the reviews received for their paper. If their paper gets accepted by the program committee, the final version of the paper needs to be uploaded to the system
- The paper review system is also used by the reviewers, who receive their login parameters in email. They need to fill in their contact details for the conference chair when logging in to the system for the first time.
- After skimming through the titles and abstracts of submitted papers, each reviewer indicates their conflicts (i.e. those paper where the authors are close colleagues or former co-author). He or she also indicates those topics where he or she is an expert.
- The conference chair assigns the papers to at least three reviewers using semi-automated assistance from the system. The basis of assignment is the relevant topics indicated by the reviewers.
- The reviewers fill a review form to evaluate the paper from different aspects including a three-line summary, originality, strong and weak points, reviewer’s confidence, author comments, confidential comments. The most important part is the overall recommendation, which can be a score and a textual assessment ranging from strong reject to strong accept.
- Finally, the conference chairs decide on the acceptance or rejection of each paper and send a notification mail to the authors together with the reviews of the paper.

The Paper Review System


«enumeration»
ReviewStatus
draft
completed
pending
«enumeration»
ResultKind
Accepted
ConditionallyAccepted
Rejected
Withdrawn
UnderEvaluation
«Entity»
User
username : String
«Entity»
UserAccount
firstName : String
lastName : String
email : String
password : String
affiliation : String
address : String
phone : String
1..*
1
- role
- userAccount
*
«Entity»
Author
- authors
1..*
writtenBy
«Entity»
Reviewer
- reviewer
- reviews
1
*
«Entity»
Review
overallRecommendation : Integer
reviewStatus : ReviewStatus
*
*
- reviewers
1..*
- reviews
*
- reviews
*
1
1 - conference
1
- paper
- papers
*
1
- conflictingPapers
*
«Entity»
Paper
title : String
keywords : String
abstract : String
submittedPaper : File
finalVersion : File
result : ResultKind
abstractSubmission : Date
paperSubmission : Date
«Entity»
Chair
*
1
- chairs
«Entity»
Conference
abstractDeadline : Date
paperDeadline : Date
- submissions
1
*
1
- competences
*
*
- topics
«Entity»
Topic
name : String
1..*
*
qualifiedBy
- topics
- relatedPapers