BON

Case Study
Conference Management

Largely based on slides
by Prof. Paige

Conference Management

- System to be used for managing a technical conference
- Should help organizers follow up a series of events taking place during preparation of the programme, and to handle contributions and registrations
- Apply BON method and come up with a BON specification

Basic Tasks

- Monitor scheduled events and check that actions required are carried out
  - paper deadlines
  - registration deadlines
- automate process of tasks to avoid duplication of effort and to provide warnings
- serve as an information repository for all committees
  - technical sessions, tutorials, exhibition of commercial products

Delineate System Boundary

- Conferences usually have three committees
  - » **program committee**
    - > **responsible for the technical sessions**
  - » **organizing committee**
    - > **responsible for logistics**
  - » **finance committee**
    - > **responsible for the money**
- Committees are usually distributed
  - » **Unrelated software may be used by different groups**

Delineate System Boundary – 2

- What information is entering the system?
  - » **registration forms, payments (wire cheque), submitted paper/tutorial, reviewer report**
- What information is leaving the system?
  - » **programme, calls for papers, invitations, participation**
  - » **letters of acceptance, rejection, confirmation**
  - » **proceedings, tutorial notes, posters, session signs**
  - » **list of attendees, financial status, badges**
- Accept fuzziness in requirements for now
  - » **Type of system needed will dictate refinements**

Candidate Classes

- No ideal way to do it
  - » **Concentrate on identifying the real world domain abstractions**
  - » **Ignore design / implementation issues**
- Attendee
  - » **registered person at the conference**
- Committee
  - » **peers organizing / evaluating the conference**
- Conference
  - » **system managing events consisting of tutorials, technical sessions, exhibitions**

Candidate Classes – 2

- Conference_Room
  - » **location for events**
- Contributor
  - » **person listed in the final programme**
- Paper
  - » **authored technical paper submitted**
- Programme
  - » **conference events description**
- Referee
  - » **reviewer of submitted technical papers**

Candidate Classes – 3

- Registration
  - » **record of attendee participation**
- Send_Out
  - » **message sent from committee to people involved in event**
- Task
  - » **elementary action**
- Tutorial
  - » **short training course**

Class Selection & Clustering

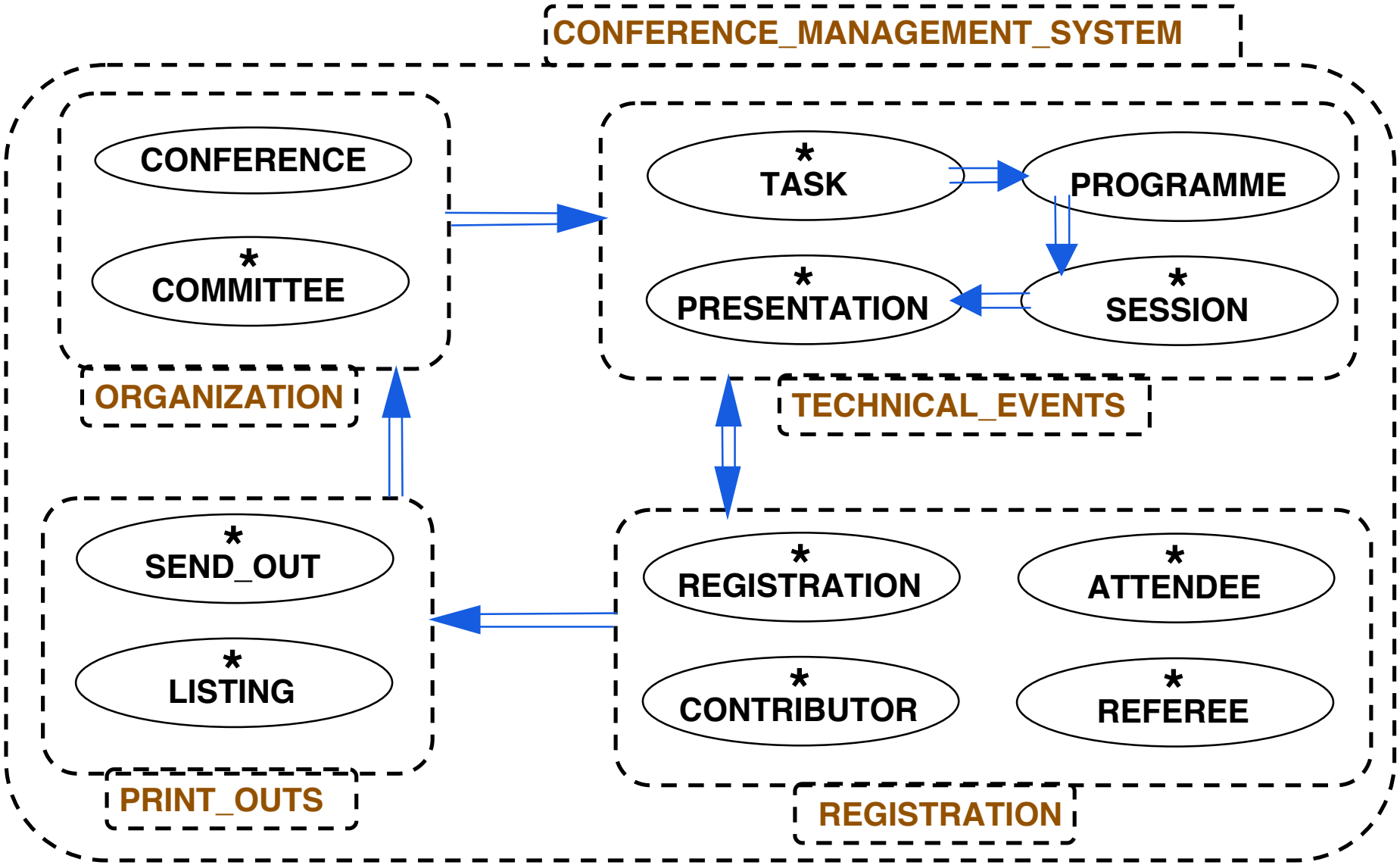
- These are candidates
  - » **may be insufficient for needs, may have redundant classes**
- Tentative
  - » **systematic description of class properties and constraints may confirm our choices**

Class Selection & Clustering – 2

SYSTEM	Conference Management System	Part #
Purpose General conference administration support		Indexing

Cluster	Description
ORGANIZATION	Handle major events occurring during the conference from initial decisions through to conclusion
TECHNICAL_EVENTS	Putting together the programme, record status of contributions, check in reviews and follow a precise timetable of what is to be done
REGISTRATION	Collect registration data, produce lists, print badges, send form letters. Store data relevant to whatever may change the cost/benefit of the conference
PRINT_OUTS	Record all possible formats used to print out information: preformatted letters, badges, final programme, session signs, etc.

Cluster Chart



Classification

- System contains some very general classes
  - » **Presentation, Registration, Contributors**
    - > **Introduced because there will be variations**
  - » **Registration – advanced, discount, complementary**
  - » **Contributors – speaker, author, keynote, tutorial, moderator**
  - » **Send_Outs – contributor, attendee, supplier**
  - » **Paper – rejected, selected**
- Use of subclasses to type instances instead of using data flags in an object

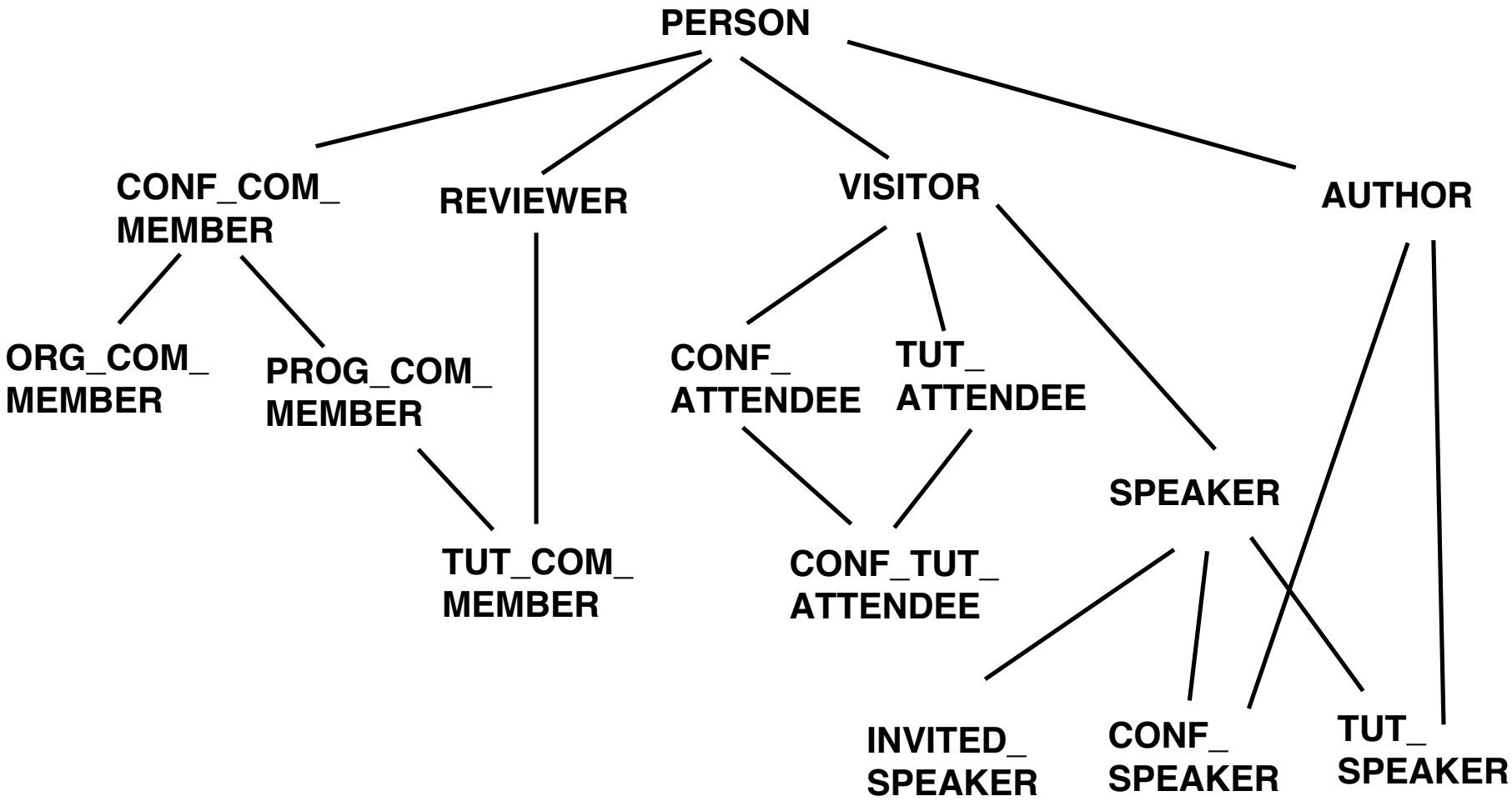
Classification – 2

- Cluster Print_Outs may be reusable in a system with little to do with conferences
  - » **Implement in this way**
- Our design should handle the variations in the previous slide
- Choice is to classify using **inheritance** or **client relations**
- Goal
  - » **Get rid of case discriminations that make maintenance difficult**

Classification – 3

- Inheritance may be preferable
  - » **similar objects have many similar features**
  - » **but some require different treatment**
- Client may be preferable
  - » **Roles played by an object can change dynamically**
  - » **When variants can be combined**
    - > **both advanced and discount registration**

Inheritance Classification



Inheritance Classification – 2

- Classify roles of people attending conference with inheritance
- Observe a combinatorial explosion
  - » **Even multiple inheritance is not enough**
- Replace with one class, PERSON, with properties to model arbitrary combinations of attendee roles
- Similar with registration
  - » **Have a single class REGISTRATION**

Informal Class Definitions

- Clarify interfaces of each class informally **before** we formally define them using BON's class interface diagrams
- Use class chart format for this
- Some of the constraints associated with CONFERENCE will not be translated into formal BON specifications but are nonetheless interesting

Informal PERSON Class

CLASS	PERSON	Part #
Type of Object	Person whose address is kept track of	Indexing
Queries	Name, Title, Affiliation, Address, Registration	
Commands	Register, Cancel, Substitute	
Constraints	Conference visitors are all registered	

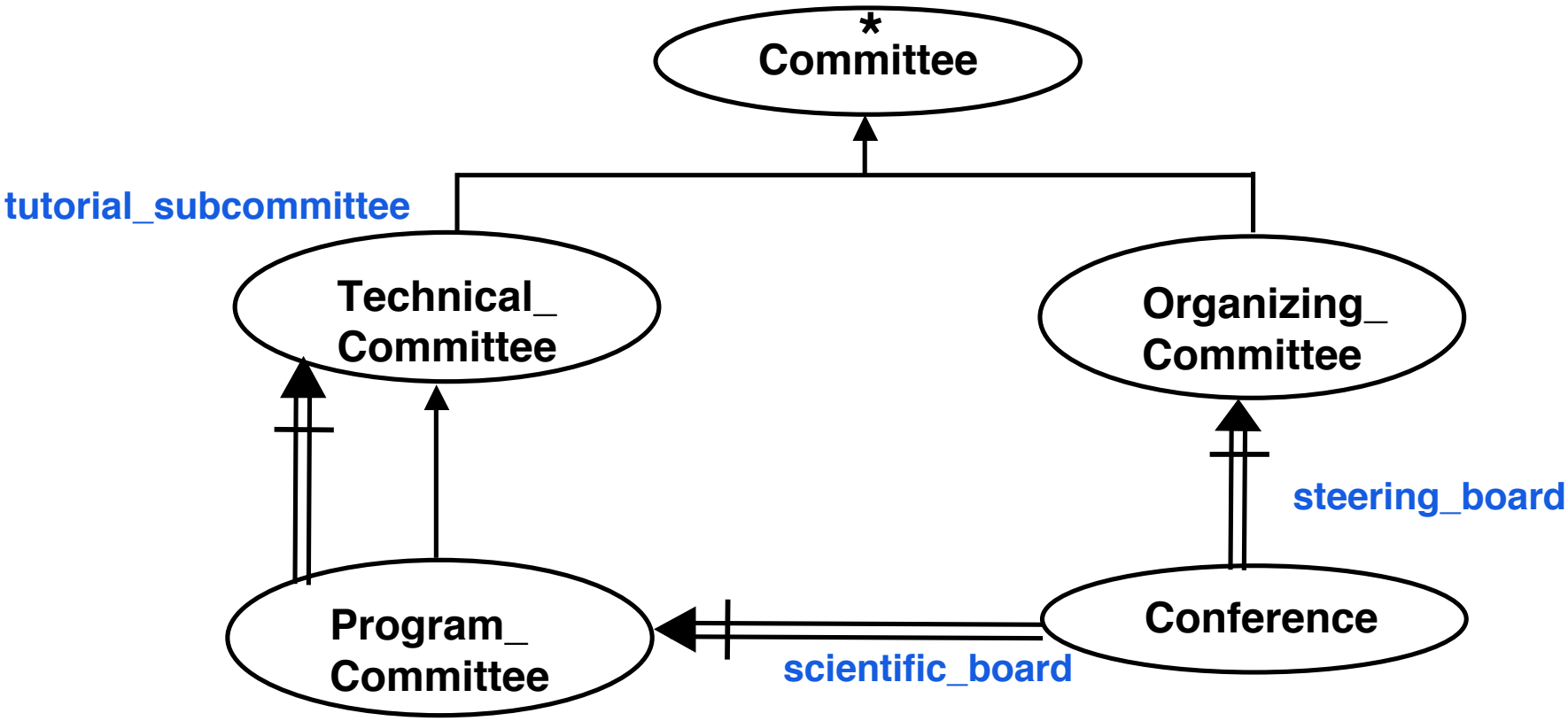
Informal CONFERENCE Class

CLASS	CONFERENCE	Part #
Type of Object	Generic conference	Indexing
Queries	Name, Location, Capacity, Programme, Budget, Attendees, Organizing committee, program committee	
Commands	Prepare, Shut down	
Constraints	Run only once a year Total registrations ▯ capacity Location serviced by international airport Accommodation capacity ≥ capacity	

Informal COMMITTEE Class

CLASS	COMMITTEE	Part #
Type of Object	Conference committee	Indexing
Queries	Chairperson, Members	
Commands	Set-up, Select chair	
Constraints	Chair is committee member	

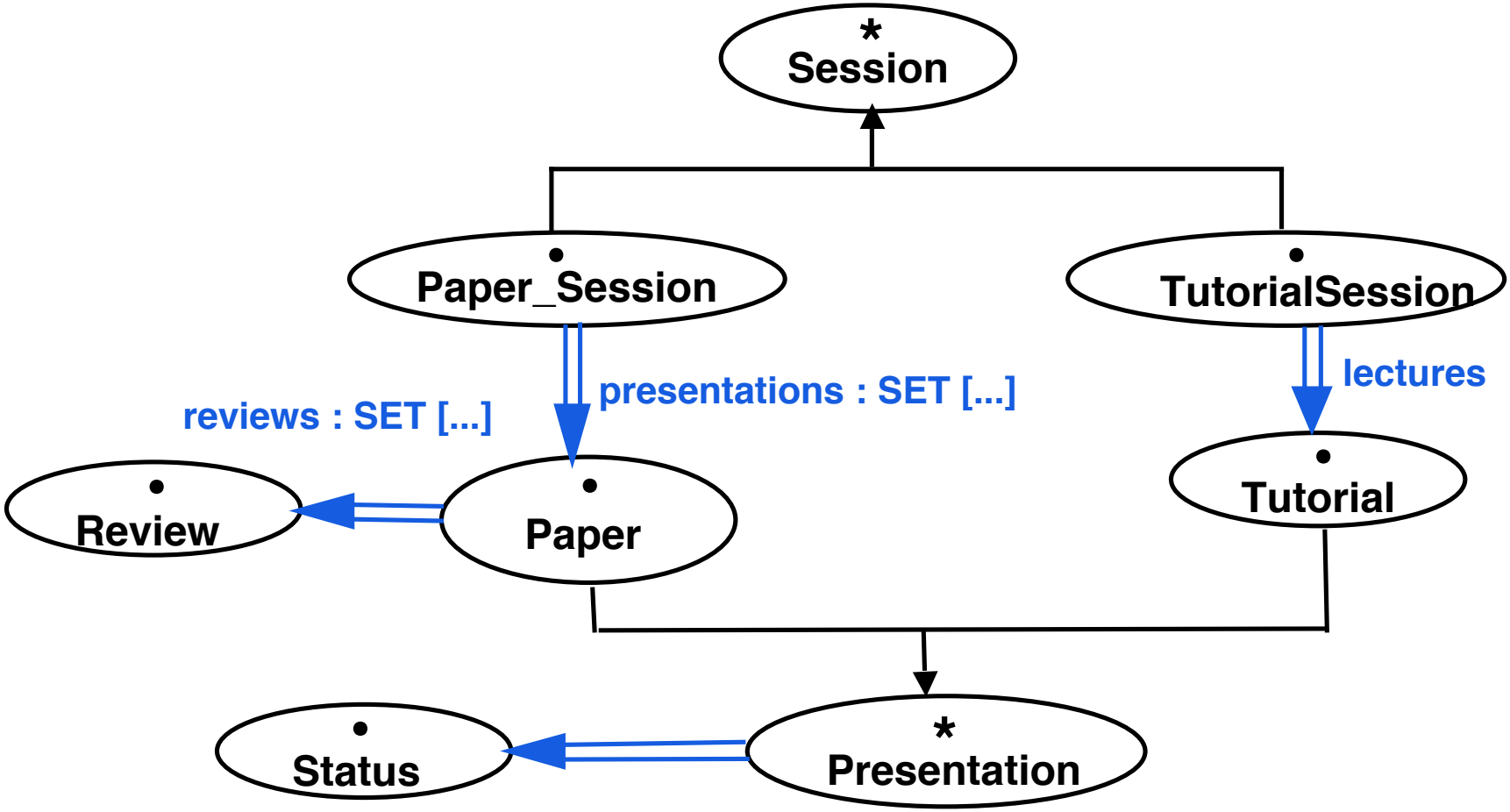
Committee Classification



Informal PROGRAMME_COMMITTEE Class

CLASS	PROGRAMME_COMMITTEE	Part #
Type of Object	Conference programme committee	Indexing
Inherits from	TECHNICAL_COMMITTEE	
Queries	Tutorial subcommittee, Acceptance standards, Review meeting, Referees	
Commands	Prepare call for papers, Dispatch submission to reviewers, Enter review results, Select session chairs, Prepare programme	
Constraints	Members ≤ 40 Industrial - Academic ≥ 4 Final selected papers ≥ 30	

Technical Events Classification



Technical Events – notes

- Details for technical events cluster
- Tutorial session has a single lecture
  - » **other attributes: attendees, chair**
- A paper session has several presentations
- Note persistence

Informal TUTORIAL Class

CLASS	TUTORIAL	Part #
Type of Object	Submitted tutorial	Indexing
Queries	Capacity, Number of attendees, Technical prerequisite, Track, Duration	
Commands	------	
Constraints	Number of attendees ≤ capacity	

Informal REGISTRATION Class

CLASS	REGISTRATION	Part #
Type of Object	Record of attendee participation, fees keeping track of affiliation, sessions	Indexing
Queries	Attendee, Conference days, Selected tutorials, Date, Amount paid, Invoiced, Confirmed	
Commands	Confirm participation, Invoice, Send documents	
Constraints	Invoice has no affect for free access attendees	

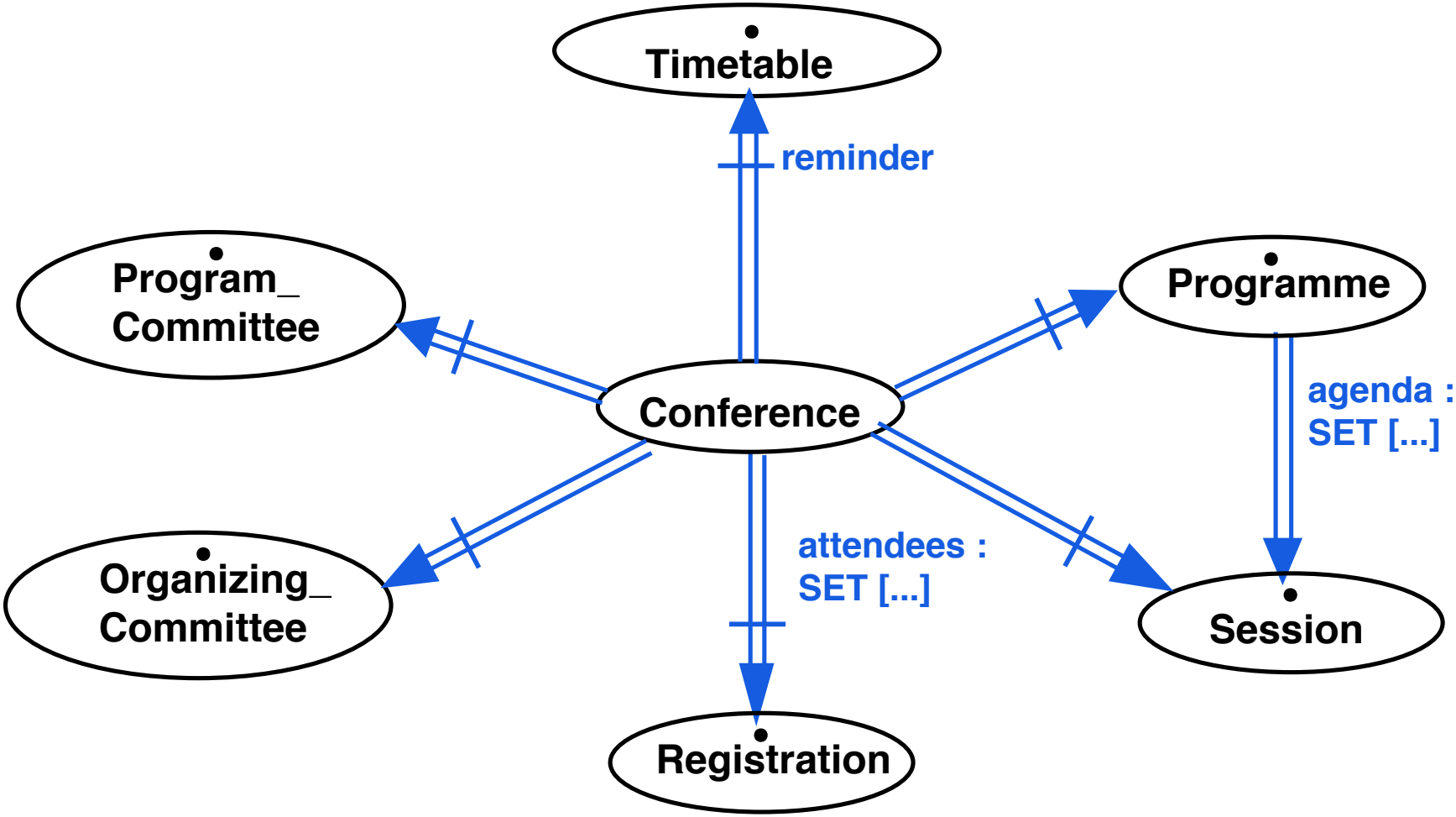
Registrations

- Every registration comes in an order
  - » **there may be several attendees per order**
- Need to connect registration to attendee
- One option
  - » **how about one registration per received order**
  - » **group registration?**
  - » **Not good, need to be able to cancel individual registration**

Registrations –2

- Need a two-way client relationship between PERSON and REGISTRATION
  - » **What if the relation was unidirectional?**
- A PERSON can visit any part of the conference
  - » **the REGISTRATION to which it is attached dictates which parts**
    - > **proper badge will be issued**

Conference Management



Informal PROGRAMME Class

CLASS	PROGRAMME	Part #
Type of Object	Information pertaining to final programme and preparation	Indexing
Queries	Paper and tutorial deadline, Final contribution deadline, Preliminary programme, Contributions, Agenda	
Commands	Update, Print	
Constraints	Final deadline = submission deadline + 4 months Contributions accepted by programme committee	

Formal Class Description

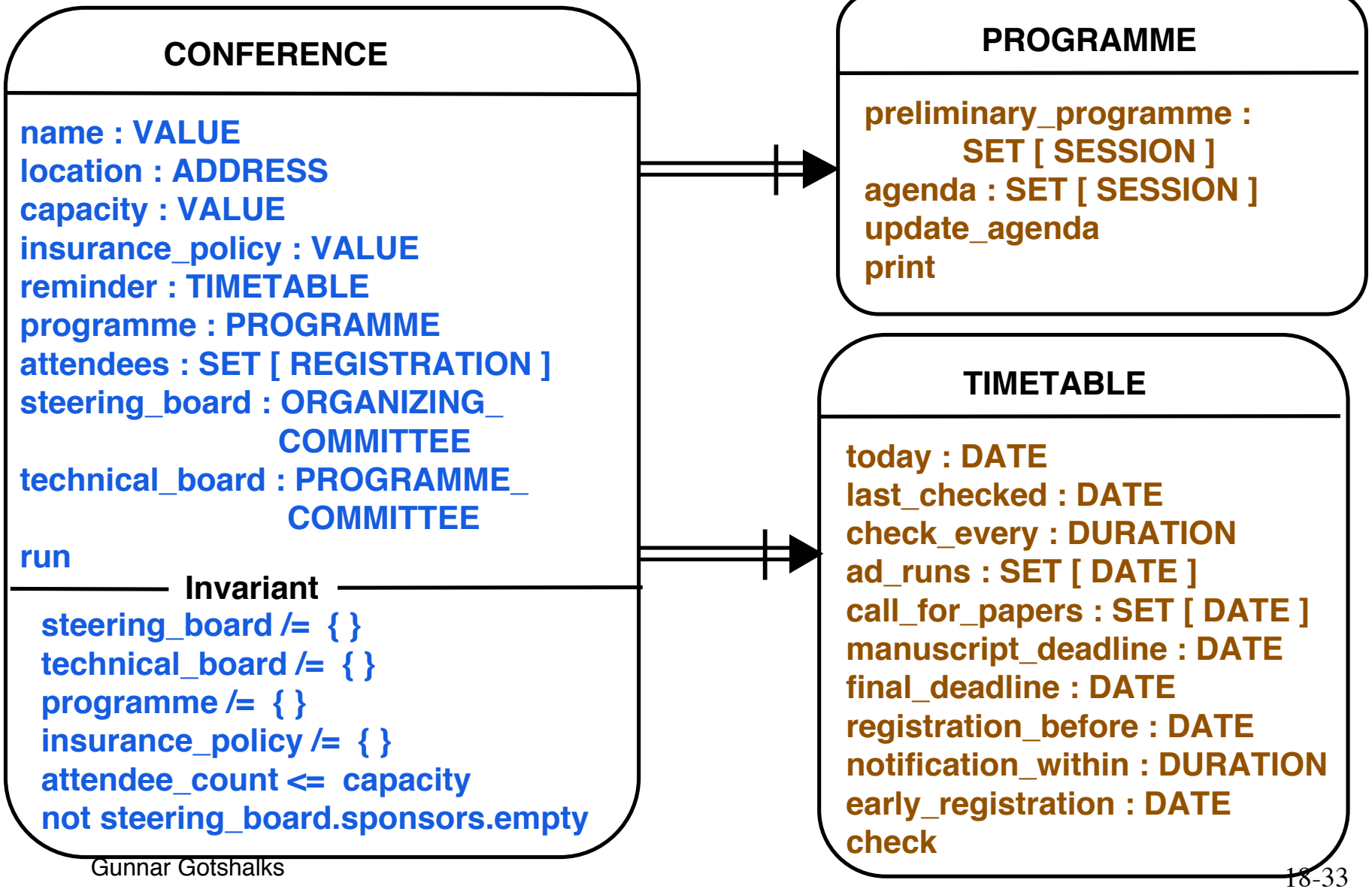
- Specify each class in a formal manner
  - » **giving signature of each routine**
    - > **parameters, result**
  - » **starting from class charts**
- Formal specifications used to specify behaviour of routines and class
  - » **pre- post- conditions, invariants**
- Design decisions as to what types to use for features

Formal Class Description – 2

- Can use general types – e.g. VALUE – rather than specific types – e.g. STRING, INTEGER
- Differentiate between types that should be different
- Show similarities among types that should be similar

Don't over constrain the implementation

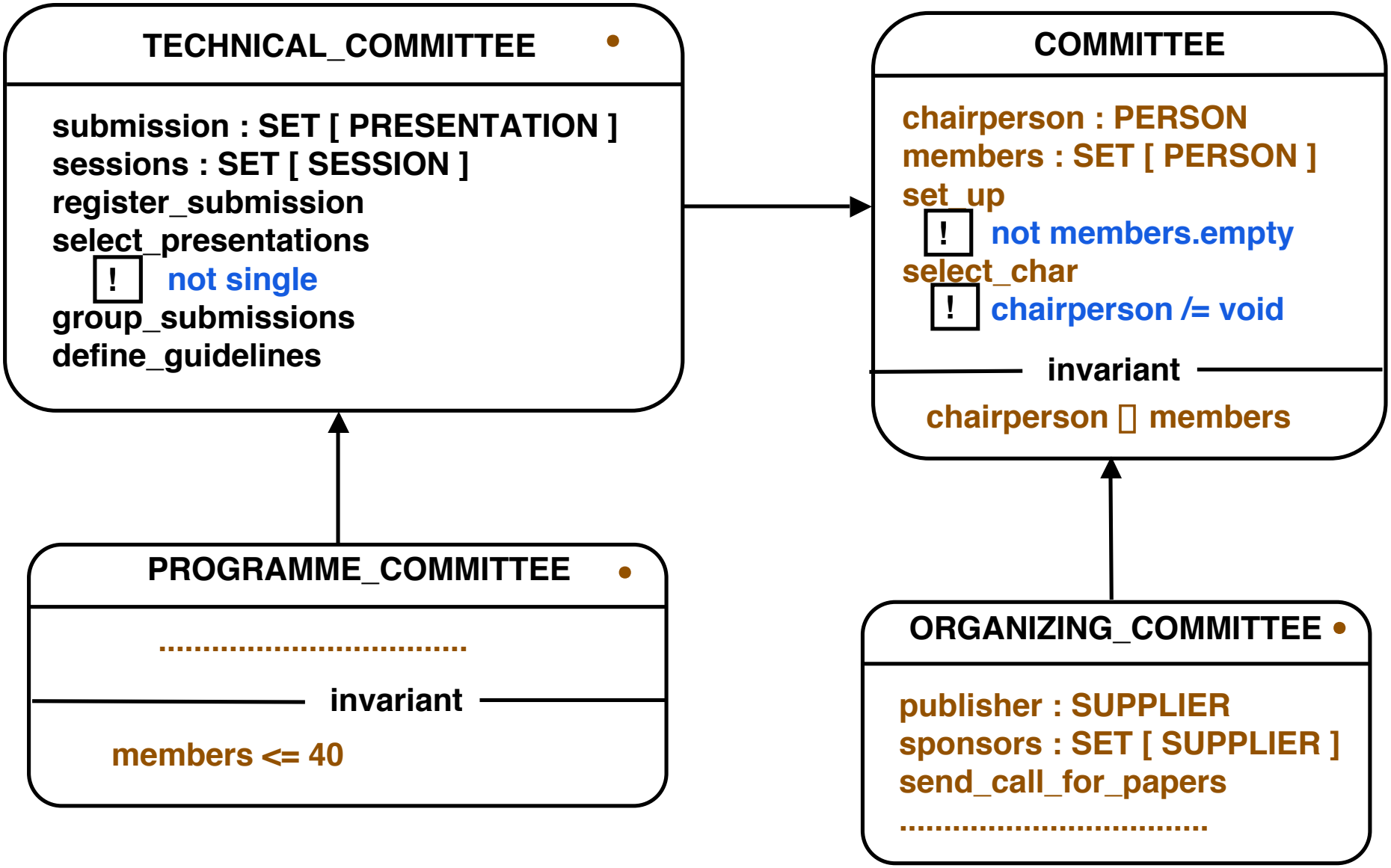
Organization Cluster



Notes – Organization Cluster

- Class invariant
  - » **rules imposed because client-supplier assertions belong in the client classes**
- Merging of commands
  - » **only keep run in the class diagram**
    - > **The attributes were considered more important for this diagram.**

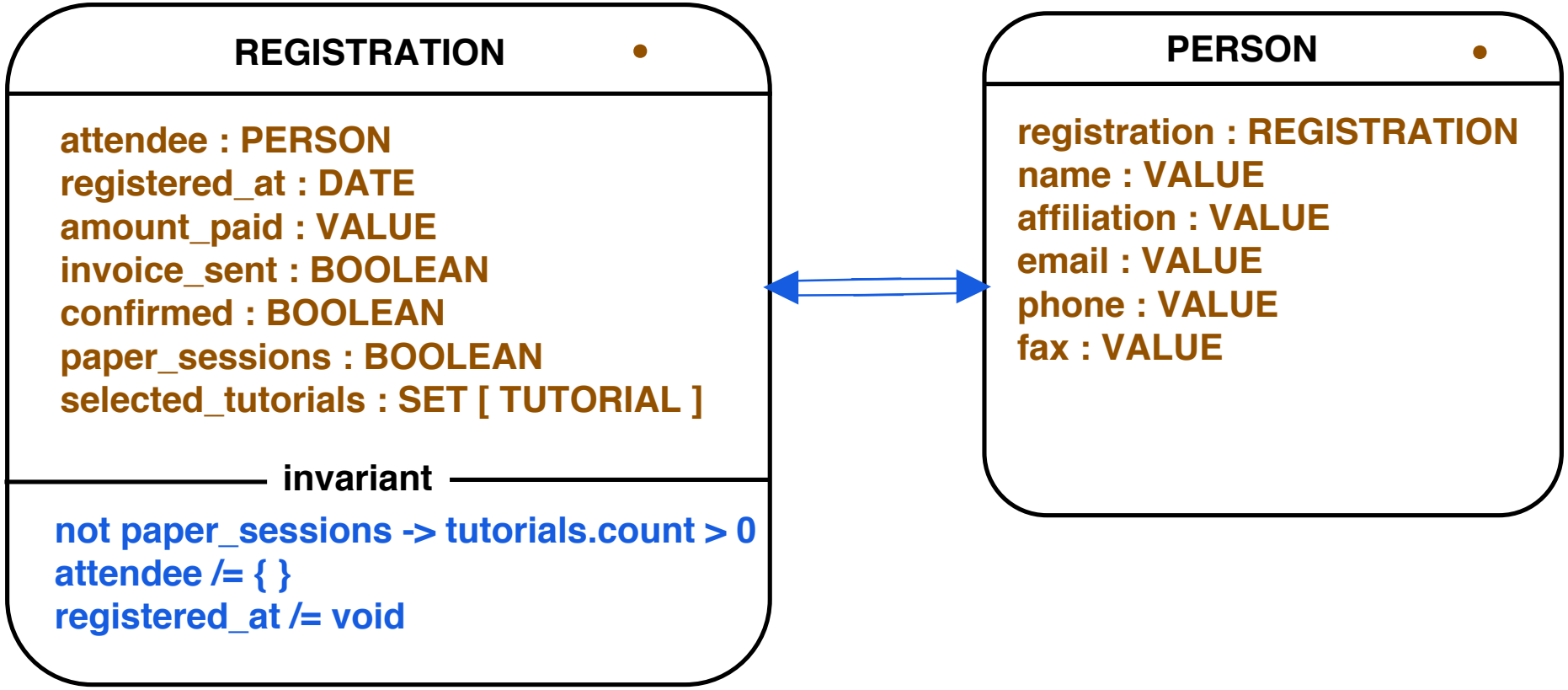
Committees


TECHNICAL_COMMITTEE
submission : SET [PRESENTATION]
sessions : SET [SESSION]
register_submission
select_presentations
!
not single
group_submissions
define_guidelines
COMMITTEE
chairperson : PERSON
members : SET [PERSON]
set_up
!
not members.empty
select_char
!
chairperson /= void
invariant
chairperson ∈ members
PROGRAMME_COMMITTEE
.....
invariant
members <= 40
ORGANIZING_COMMITTEE
publisher : SUPPLIER
sponsors : SET [SUPPLIER]
send_call_for_papers
.....

Notes – Committees

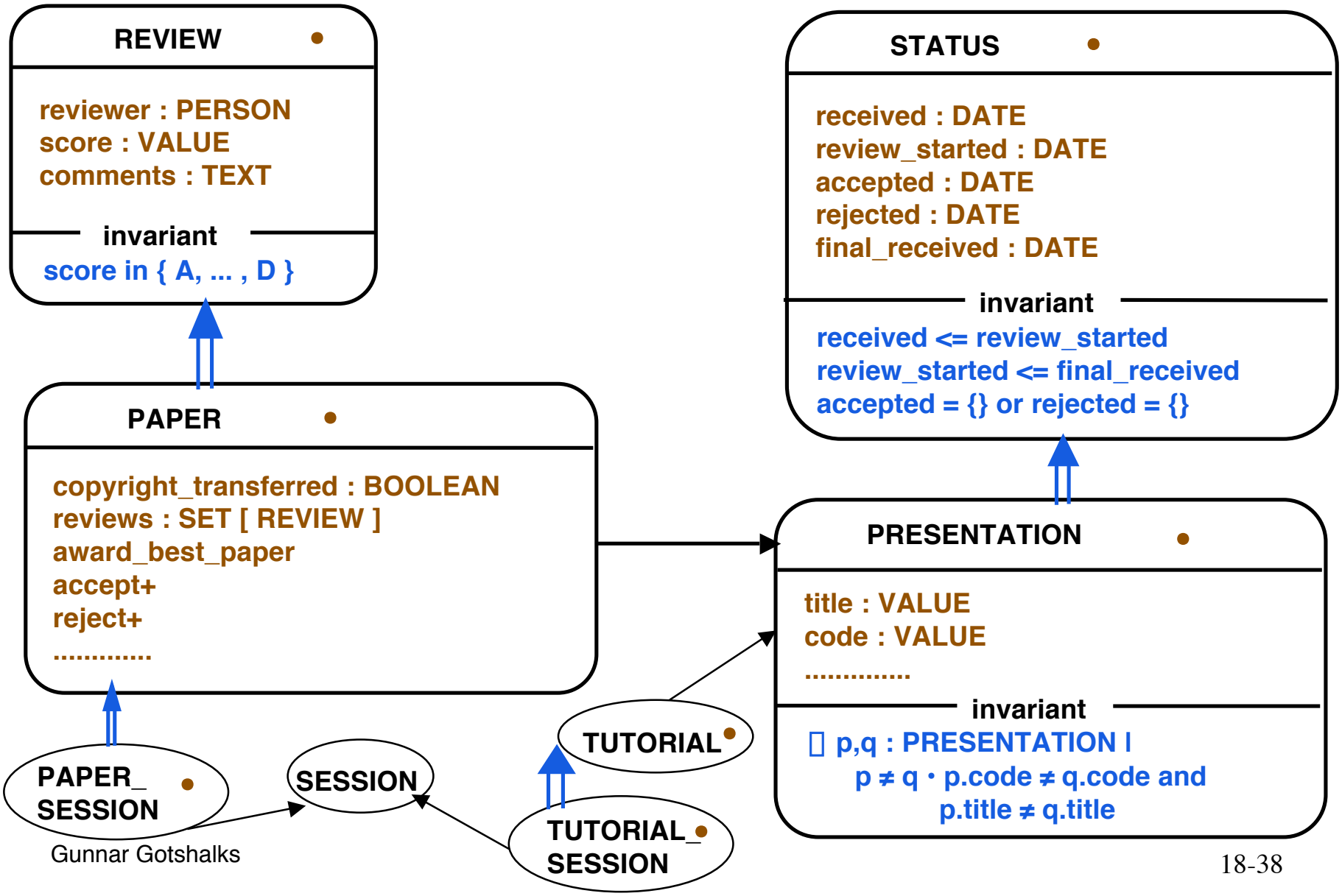
- There are no arguments to features
  - » **too high level for such detail**
  - » **Done in detail design**
- Could use VALUE instead of SUPPLIER
  - » **but it does not capture similarities**

Registration Cluster



Why not use STRING instead of VALUE?

Technical Events Cluster



Gunnar Gotshalks