



FACULTY OF SCIENCE, MATHEMATICS AND COMPUTING SCIENCE

# On the use of Object-Role Modeling to Model Active Domains

**P. (Patrick) van Bommel**  
**S.J.B.A. (Stijn) Hoppenbrouwers**  
**H.A. (Erik) Proper**  
**Th.P. (Theo) van der Weide**

[www.cs.ru.nl/iris/am](http://www.cs.ru.nl/iris/am)

**Radboud University Nijmegen**





# On the use of Object-Role Modeling to Model Active Domains

## Menu

- The basic idea
- Key elements:
  - Active domains from a fact oriented perspective
  - ORC for temporal rules
  - Graphical representation
- Next steps
- Discussion



# On the use of Object-Role Modeling to Model Active Domains

## Menu

- The basic idea
- Key elements:
  - Active domains from a fact oriented perspective
  - ORC for temporal rules
  - Graphical representation
- Next steps
- Discussion



# On the use of Object-Role Modeling to Model Active Domains

## Driving question for this paper

- Understanding the foundations of modeling
- What goes on when people produce models?
  - In a system development context
  - Modeling for different purposes
  - Focusing on different aspects



# On the use of Object-Role Modeling to Model Active Domains

## Observation

- Many modeling techniques in use:
  - Information, activity, business, architecture models, ...
- Each of these techniques:
  - Focuses on a specific aspects of a domain
  - Is specifically suitable for studying/representing this aspect
- But al:
  - Are concerned with concepts and their relations
  - Deal with facts about a domain



# On the use of Object-Role Modeling to Model Active Domains

## Hypothesis

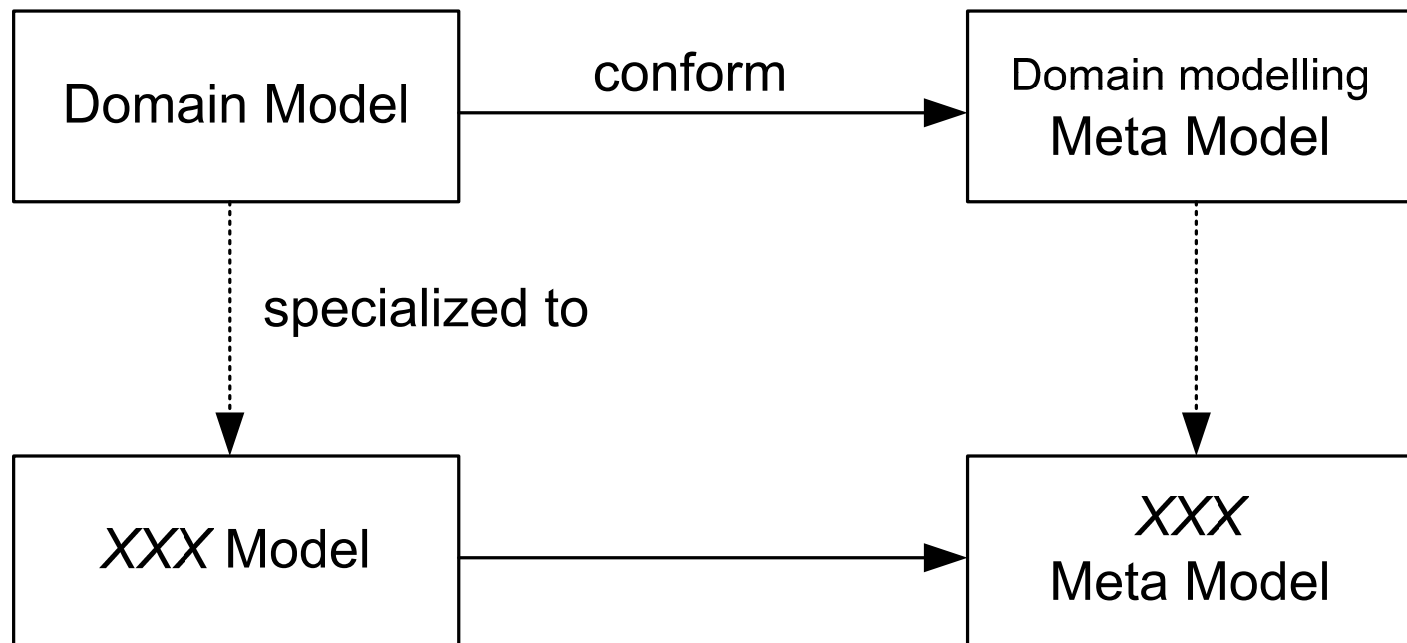
- When modeling different aspects of a domain, there is always an underlying domain model/ontology
- More operational:
  - Any:
    - activity model, conceptual database model, requirements model, sequence diagram, set of architecture principles, etc
    - has
    - an accompanying (underlying) domain model
    - of
    - the underlying concepts and relations



# On the use of Object-Role Modeling to Model Active Domains

## Consequence

- Modeling an aspect *XXX* of a domain can be regarded as:
  - Creating a domain model of the core concepts dealing with *XXX*
  - Specializing this to a model in a suitable *XXX* modeling technique





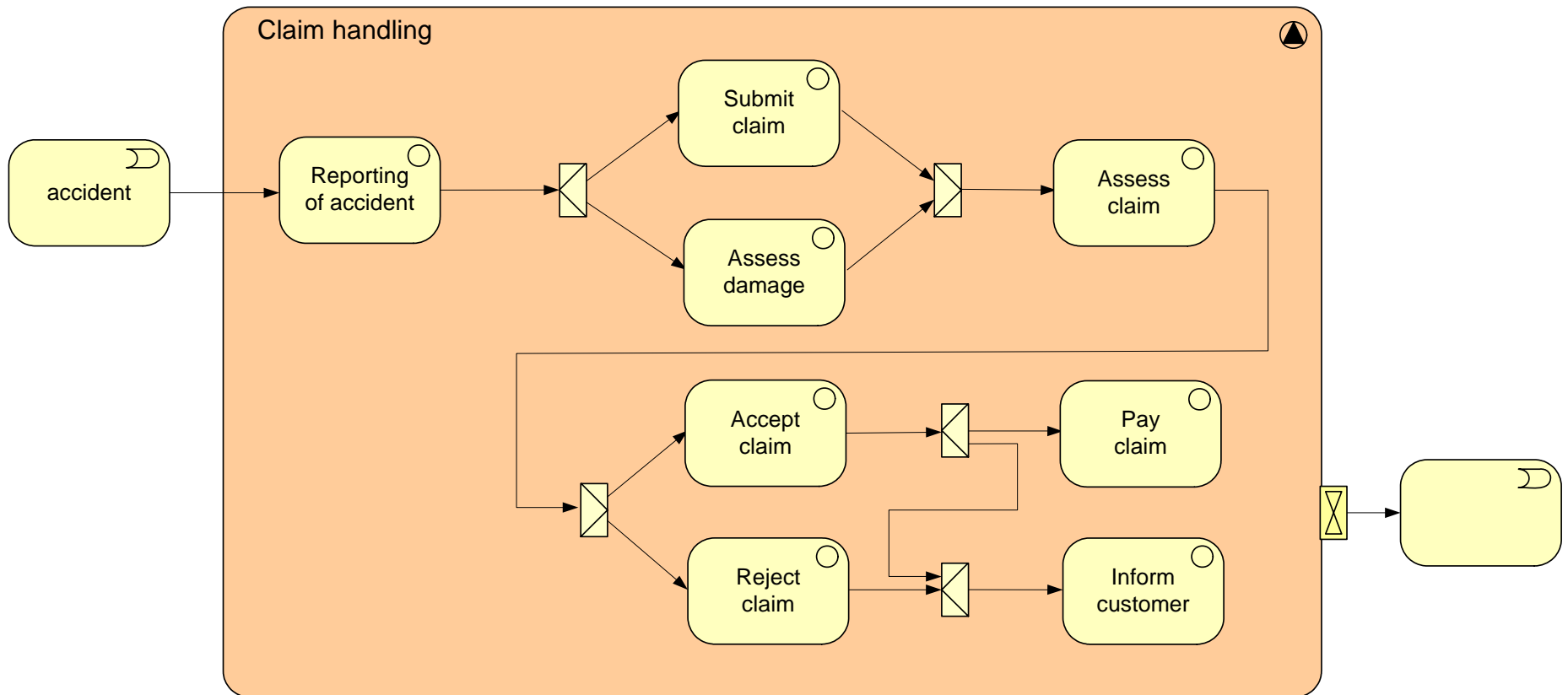
# On the use of Object-Role Modeling to Model Active Domains

## Example

- ORM domain model of a work domain
- Use this as a base for specialization towards four *viewpoints*:
  - Work-flow model
  - Work-role model
  - Resource model
  - Actor model
- Used in first year's course on organizational modeling
- Educational consideration:
  - Integration of ArchiMate, TestBed, YAWL, DEMO and AGR
  - Unified notation / look&feel: WORM; WOrk Role Modeling
  - Founded on ORM's elaborate *way of working*

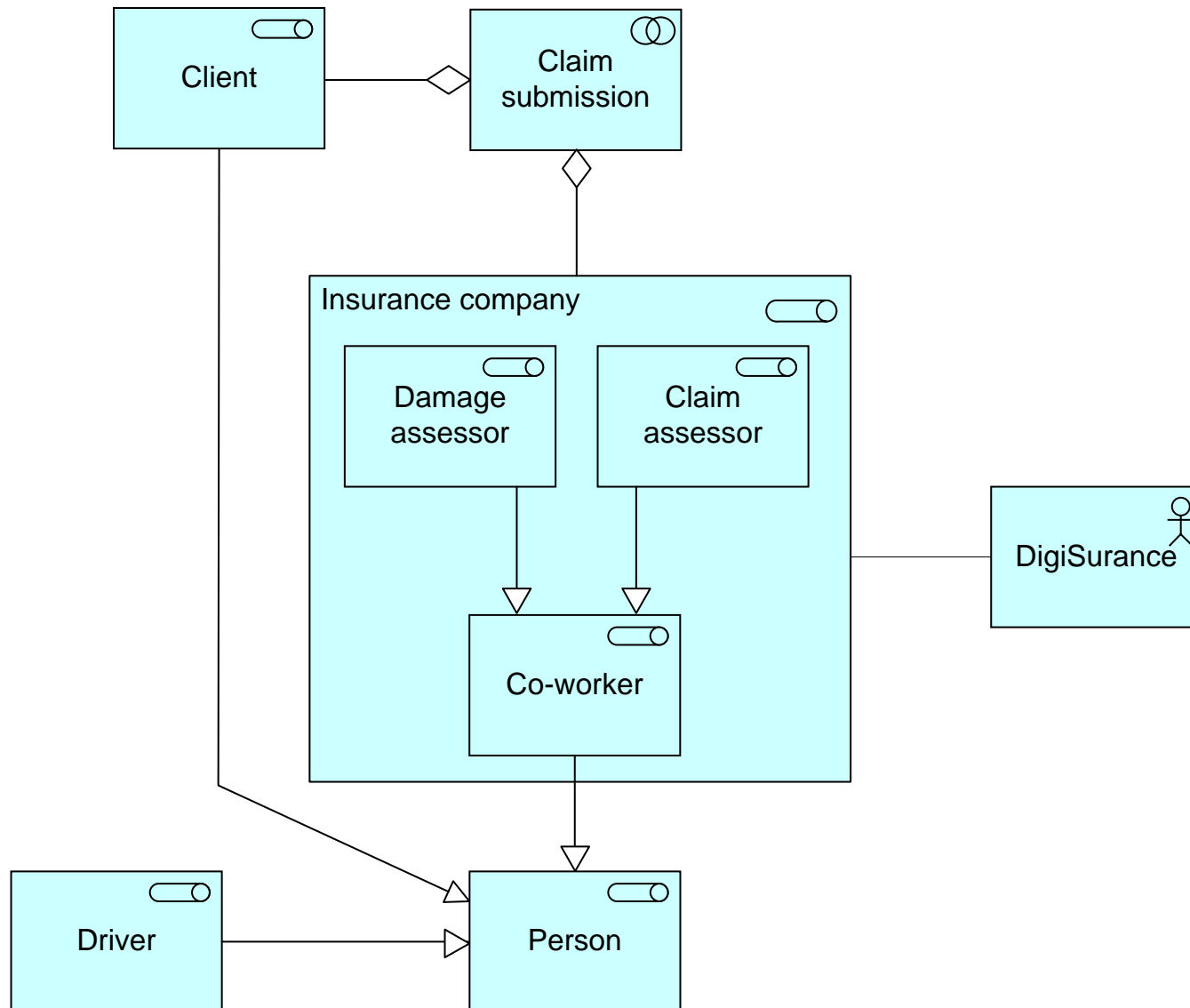


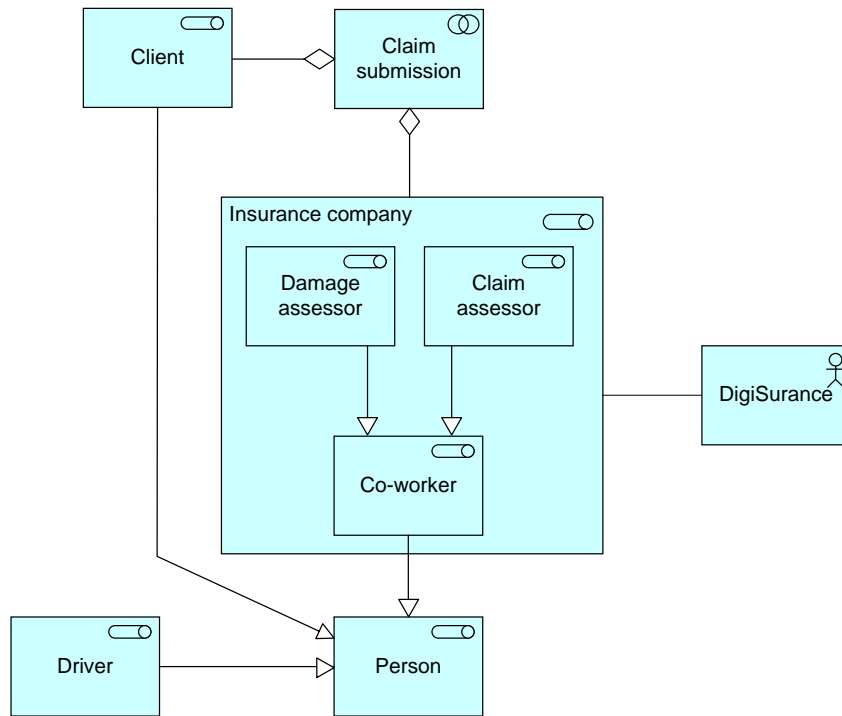
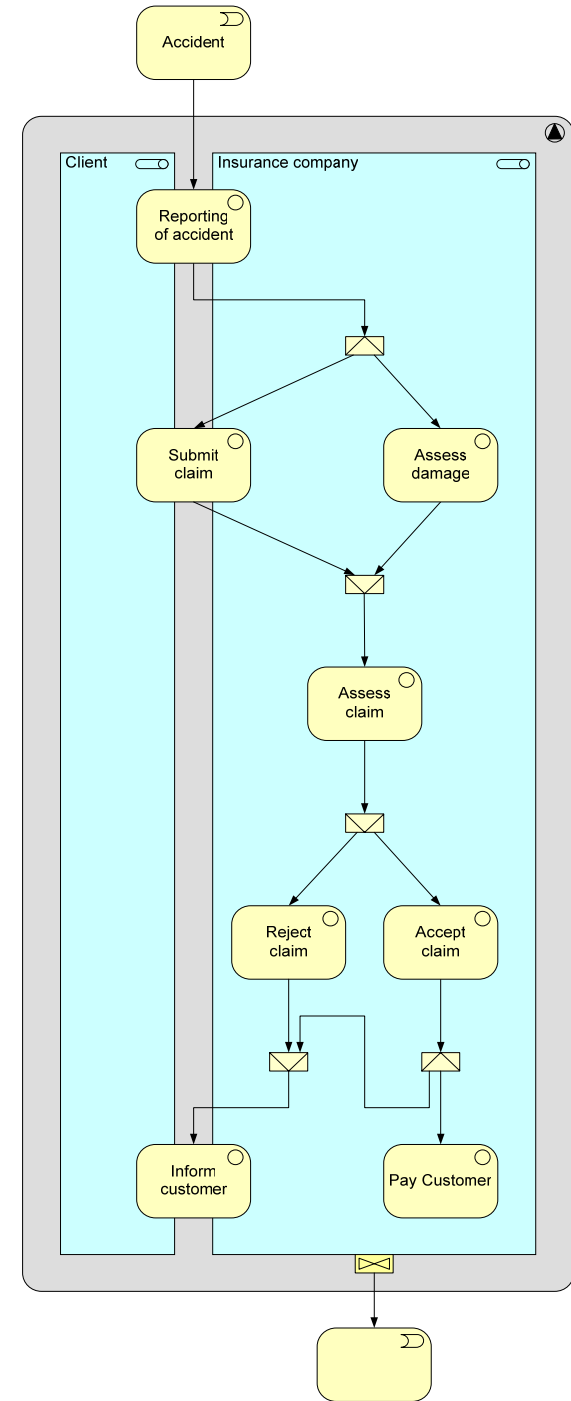
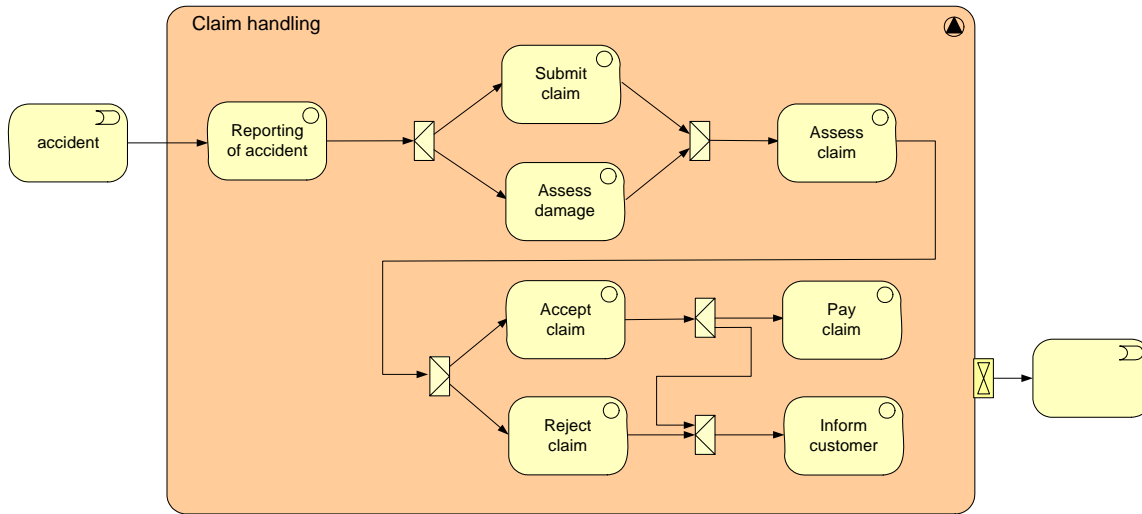
# On the use of Object-Role Modeling to Model Active Domains

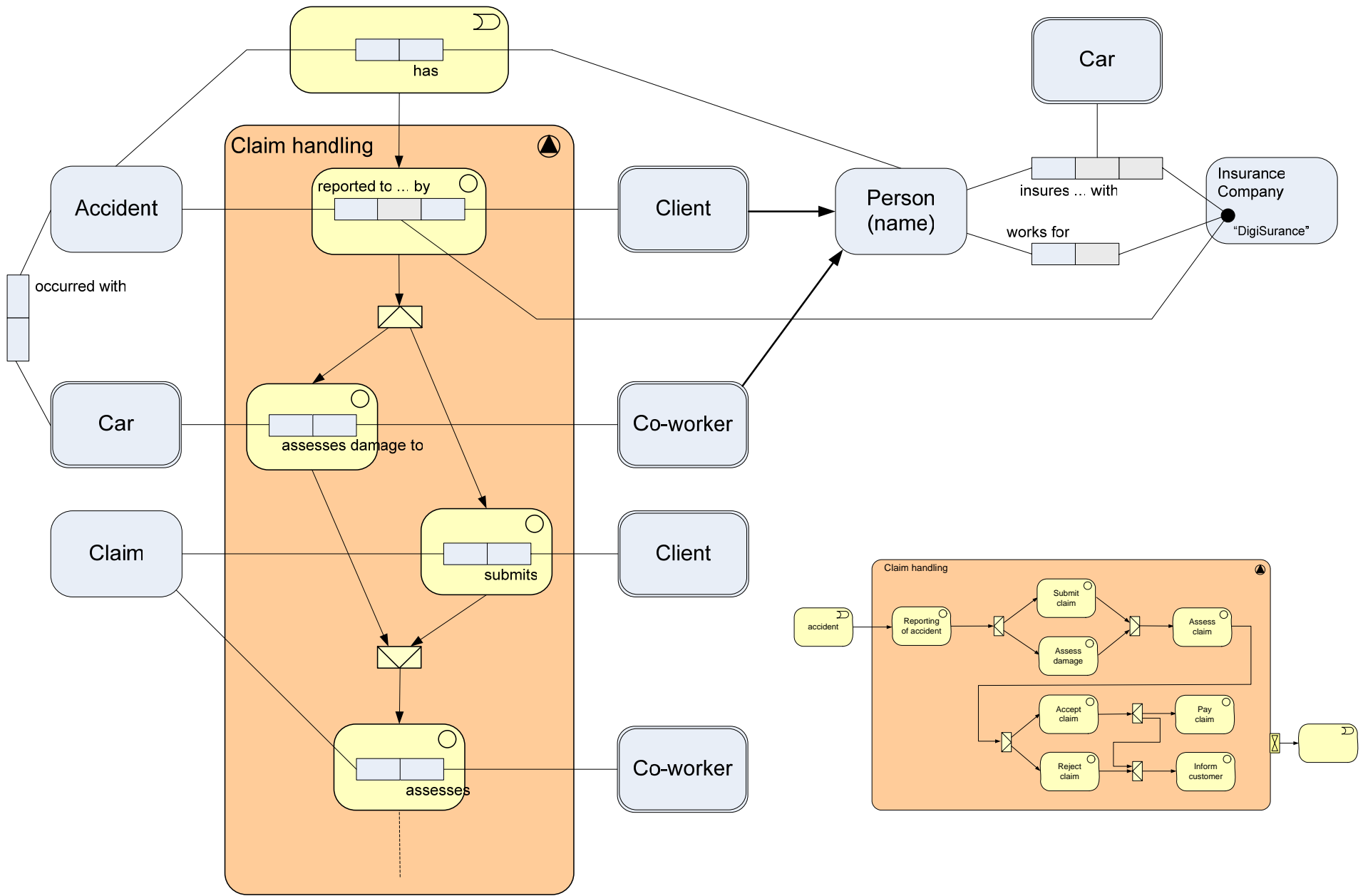


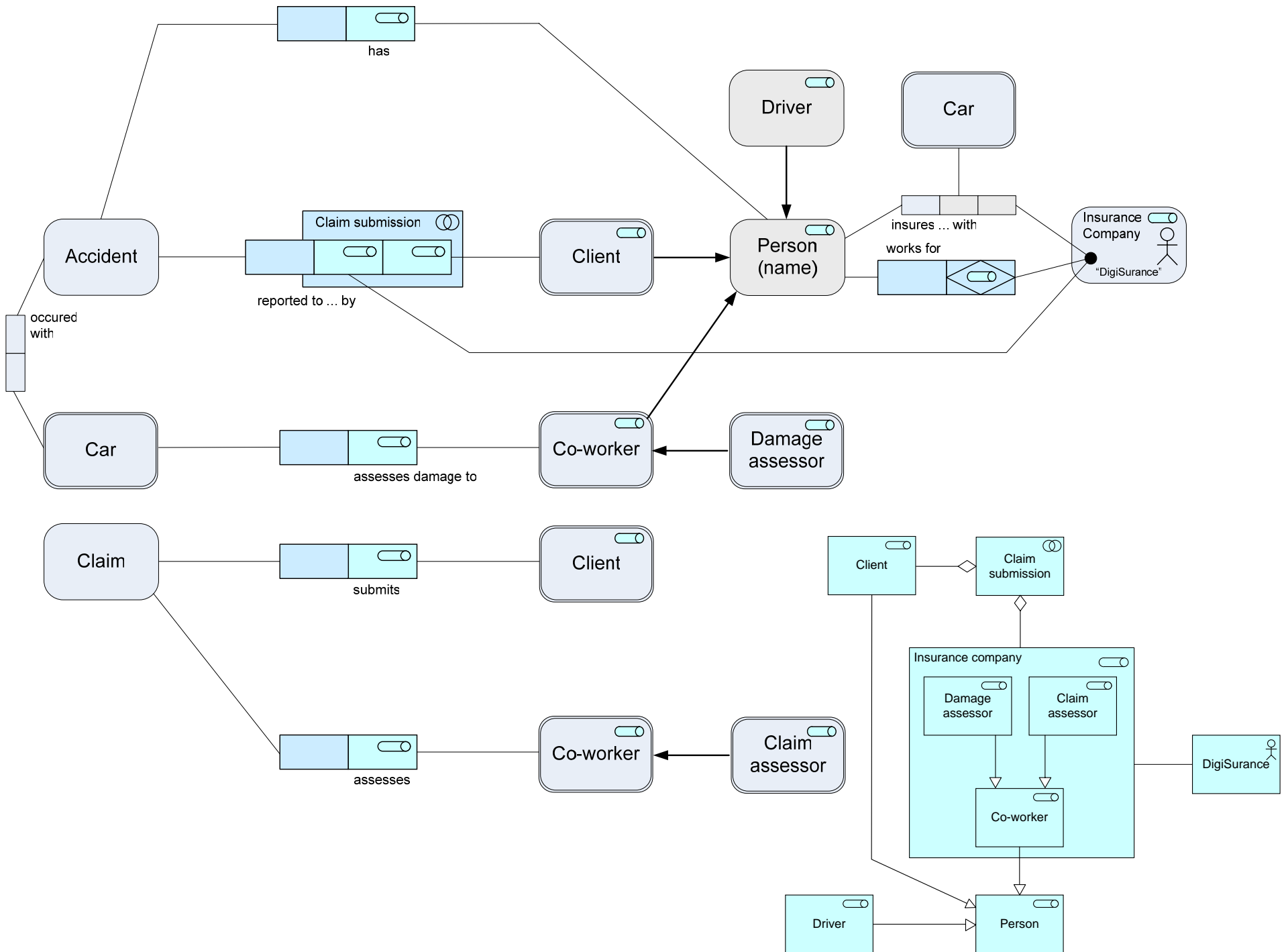


# On the use of Object-Role Modeling to Model Active Domains











## On the use of Object-Role Modeling to Model Active Domains

### Suggested way of working to students

- First produce ORM model of domain
- Then re-interpret object/role types in terms of the specific viewpoint one is aiming for
- Leads to adorned ORM model
- Re-draw relevant part of ORM diagram in dedicated notation
- But .... in reality not linear ... of course



# On the use of Object-Role Modeling to Model Active Domains

## Consequence

- Modeling process can be done linearly:
  - First produce a domain model
  - Then specialize this to an *XXX* model
- But also “inversely”
  - Create *XXX* model
  - Implies underlying domain model
- And iteratively
  - Switch between elaboration of *XXX* and domain model



# On the use of Object-Role Modeling to Model Active Domains

## Hypothesis

- Viewing a model in terms of a specialization hierarchy of meta-models aids in being more explicit about modeling decisions
- Using a domain modeling approach with a well-defined conceptualization procedure aids even more to this
  - Our “usual” candidate: ORM/NIAM



# On the use of Object-Role Modeling to Model Active Domains

## Menu

- The basic idea
- Key elements:
  - Active domains from a fact oriented perspective
  - ORC for temporal rules
  - Graphical representation
- Next steps
- Discussion



# On the use of Object-Role Modeling to Model Active Domains

## Logbook

- Observe an active domain
- I.e. events occur in the domain
- Reported as facts:
  - *Traffic light 20 is green*  
ceased being true at 11:03:20 on 22-05-2006
  - *Employee John works on the completion of order 50*  
started being true at 09:30 on 19-05-2006
- Two kinds of facts:
  - Acts
  - Effects
  - *Effects could be “big bang” effects*



# On the use of Object-Role Modeling to Model Active Domains

## From logbook to domain model

- Use ORM CSDP
  - **Warning:**
    - ORM was designed as a conceptual database design method
  - Relaxation required:
    - Identification rules
    - Relaxation of sub-typing rules
    - Allow for instances to appear in domain model
- ORM with temporal extension
  - At least ordering of events
  - Either as part of meta-model or as standardized patterns in model



# On the use of Object-Role Modeling to Model Active Domains

## Object-Role Calculus

- Proposed as a unification of:
  - RIDL, Lisa-D, Elisa-D, FORML, ConQuer
- Rule language & calculus which exploits ORM's rich verbalizations
  - Formalized natural language
  - Controlled language
- Temporal dimension



# On the use of Object-Role Modeling to Model Active Domains

## Defining Object-Role Calculus

- Counting layer:
  - Sets, bags, uncertainty, ..
- Calculus layer:
  - Description logic, Predicate logic, Deontic logic, Model logic, ..
  - Tradeoff between expressiveness and computability
- Paths layer
  - Paths through the conceptual model
  - Additional connectives
- Presentation layer
  - Verbalization or graphical depiction of paths in a controlled language



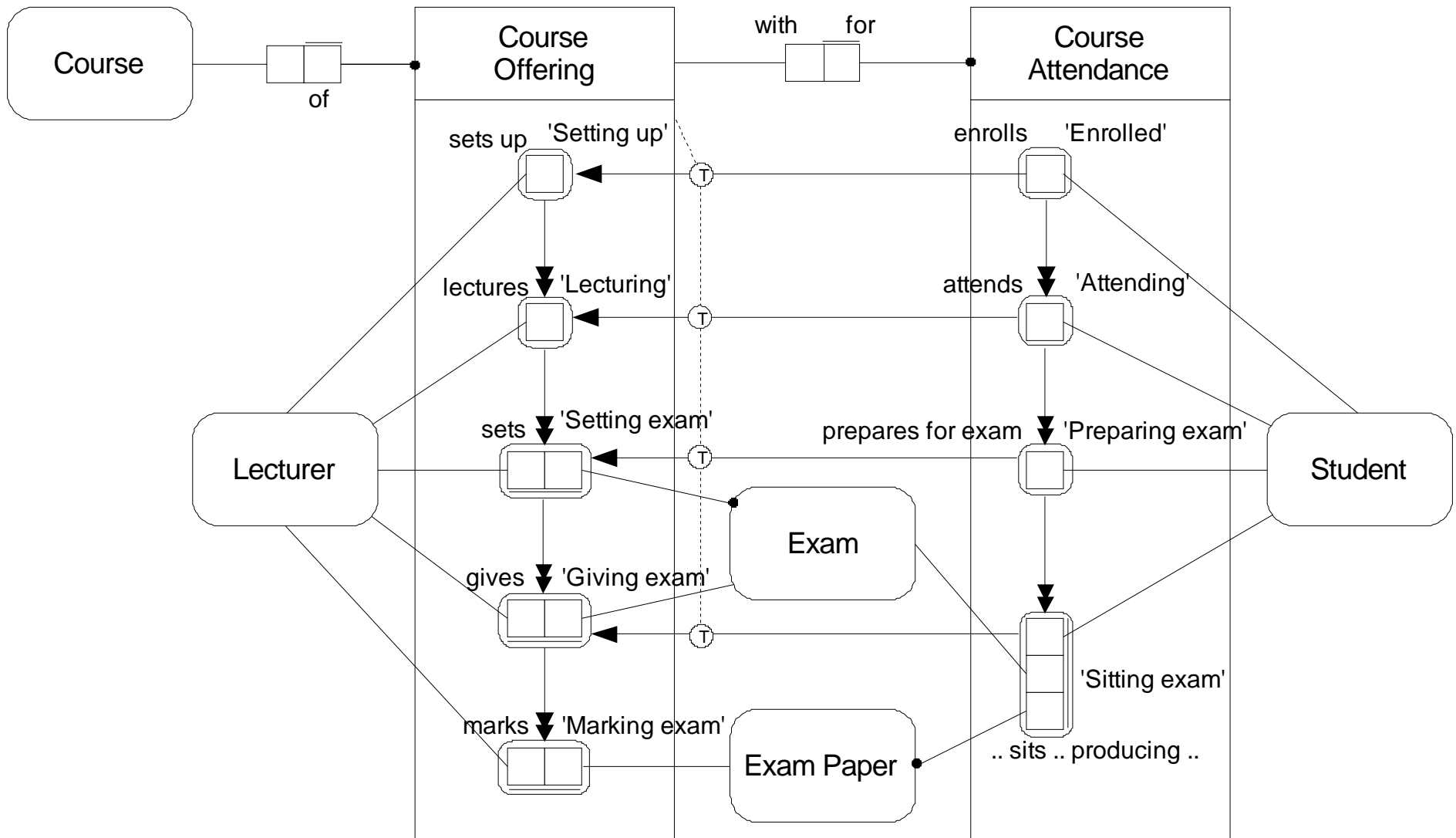
# On the use of Object-Role Modeling to Model Active Domains

## In this paper

- Sets
- Temporal (Kripke) logic
- Path expressions as rigid verbalizations
- Accompanied by rich verbalizations

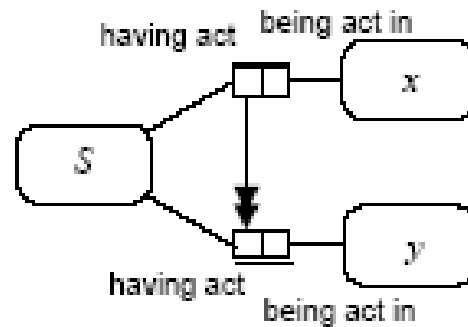
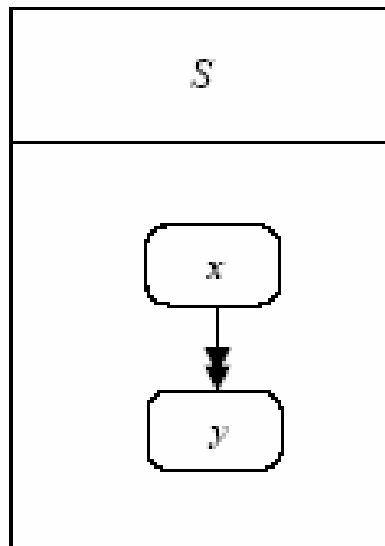


# On the use of Object-Role Modeling to Model Active Domains





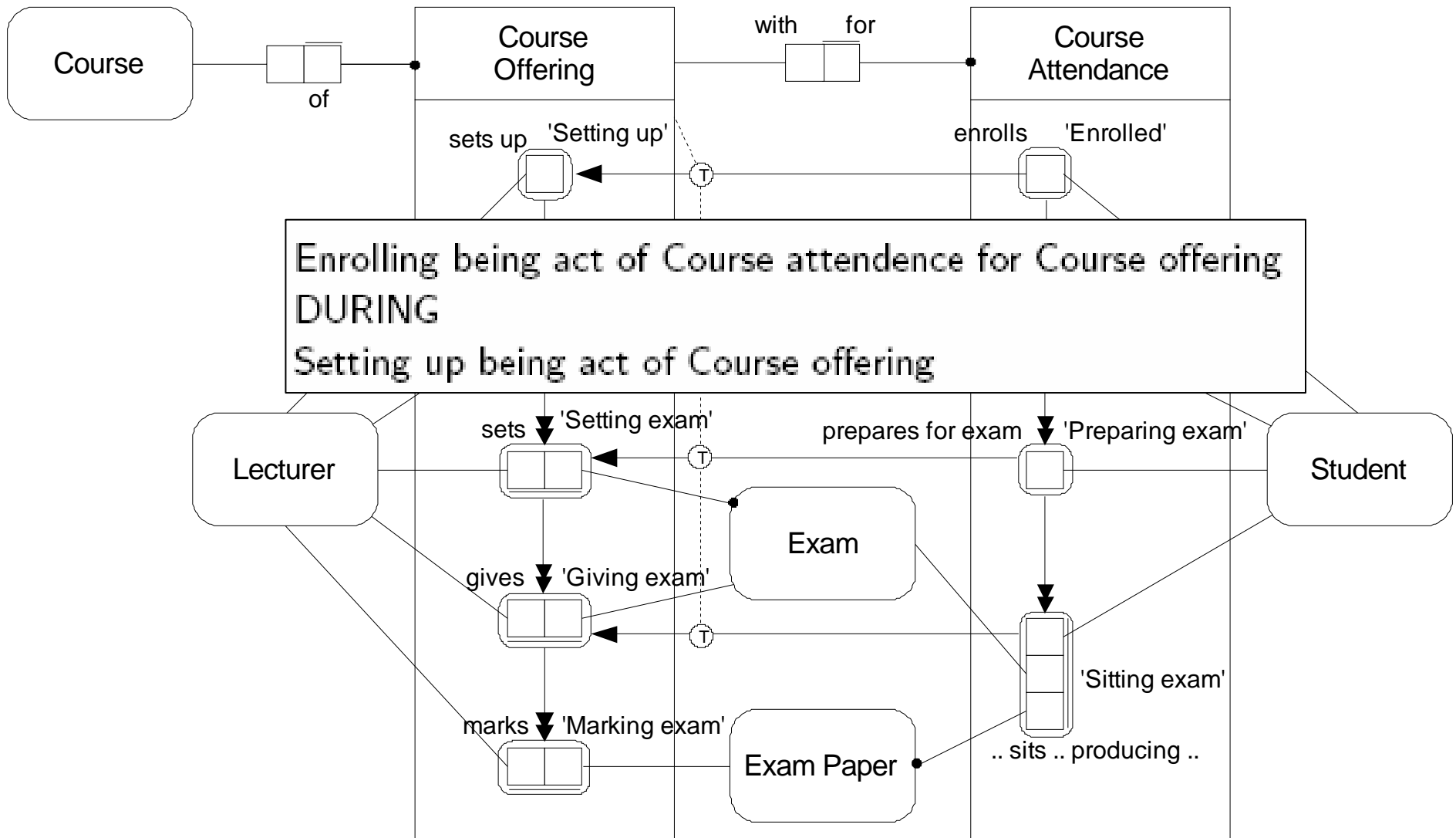
# On the use of Object-Role Modeling to Model Active Domains



$$x \twoheadrightarrow_S y \triangleq x \text{ being act of } S \text{ PRECEDES } y \text{ being act } S$$



# On the use of Object-Role Modeling to Model Active Domains





# On the use of Object-Role Modeling to Model Active Domains

## Menu

- The basic idea
- Key elements:
  - Active domains from a fact oriented perspective
  - ORC for temporal rules
  - Graphical representation
- **Next steps**
- Discussion



# On the use of Object-Role Modeling to Model Active Domains

## Hypothesis

- When modeling different aspects of a domain, there is always an underlying domain model/ontology
- More operational:
  - Any:
    - activity model, conceptual database model, requirements model, sequence diagram, set of architecture principles, etc
    - has
    - an accompanying (underlying) domain model
    - of
    - the underlying concepts and relations



# On the use of Object-Role Modeling to Model Active Domains

## Next steps – Theoretical validation

- Elaborate “The WORM example”
- Further evolution of lecture notes
- ORM as a general domain modeling language
- Four integrated viewpoints for organizational modeling as specialization targets:
  - Work-flow model
  - Work-role model
  - Resource model
  - Actor model

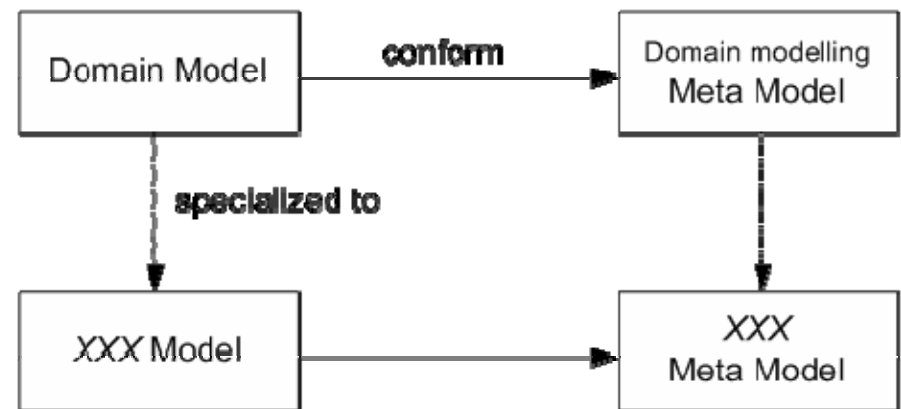


# On the use of Object-Role Modeling to Model Active Domains

## Next steps – Theoretical validation

- Show how it fits with other modeling techniques
  - Software: UML
  - Agent-based systems: AGR, ...
  - Organizations: YAWL, DEMO, e3-Value, ...
  - Enterprise Architecture: ArchiMate, ...
- Is ORM (+ Time) a suitable “root”?
- Is it a suitable general purpose domain modeling language?

- **Refine WORM?**

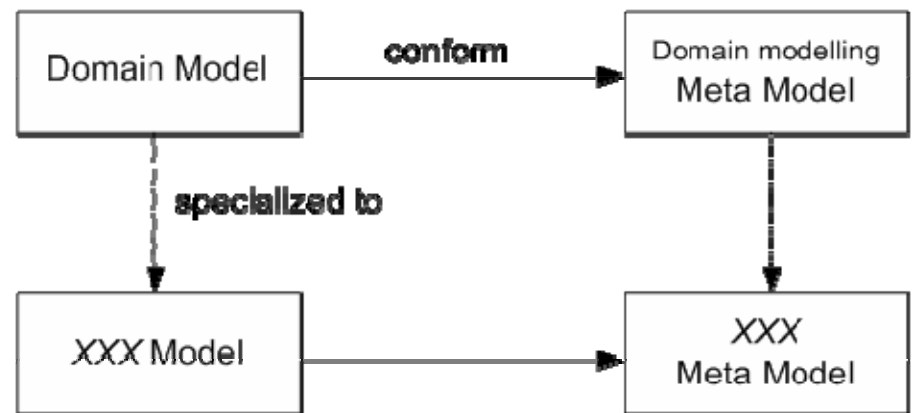




# On the use of Object-Role Modeling to Model Active Domains

## Next steps – Theoretical validation

- Can we built a modeling tool that supports this?
- Can we support modelers by automated reasoning wrt the inter-model relationships?

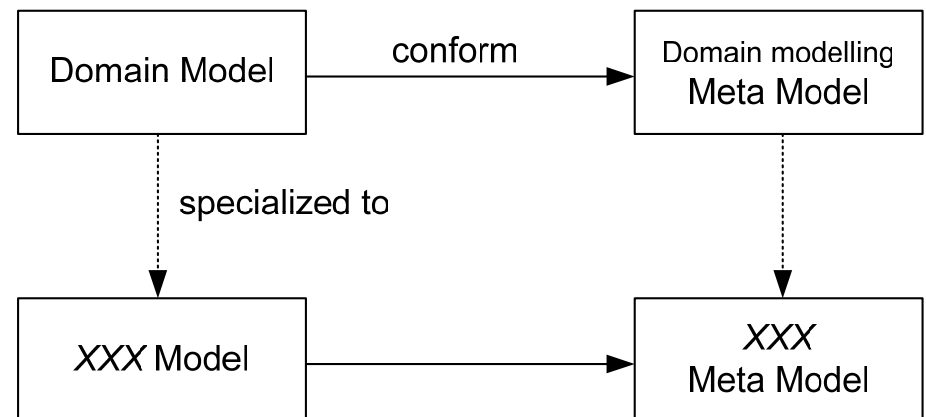




# On the use of Object-Role Modeling to Model Active Domains

## Next steps – Empirical validation

- Does modeling work this way?
- Does it help modelers?
  - “Force”/Invite modelers to be more explicit about their domain understanding?





# On the use of Object-Role Modeling to Model Active Domains

## Menu

- The basic idea
- Key elements:
  - Active domains from a fact oriented perspective
  - ORC for temporal rules
  - Graphical representation
- Next steps
- **Discussion**



# On the use of Object-Role Modeling to Model Active Domains

## Discussion

- Why not just use UML?
  - We're not into defining/engineering standards
  - We want to understand and teach the science of modeling
- Why bother?
  - From the art of modeling to a science of modeling!?
- How to validate?