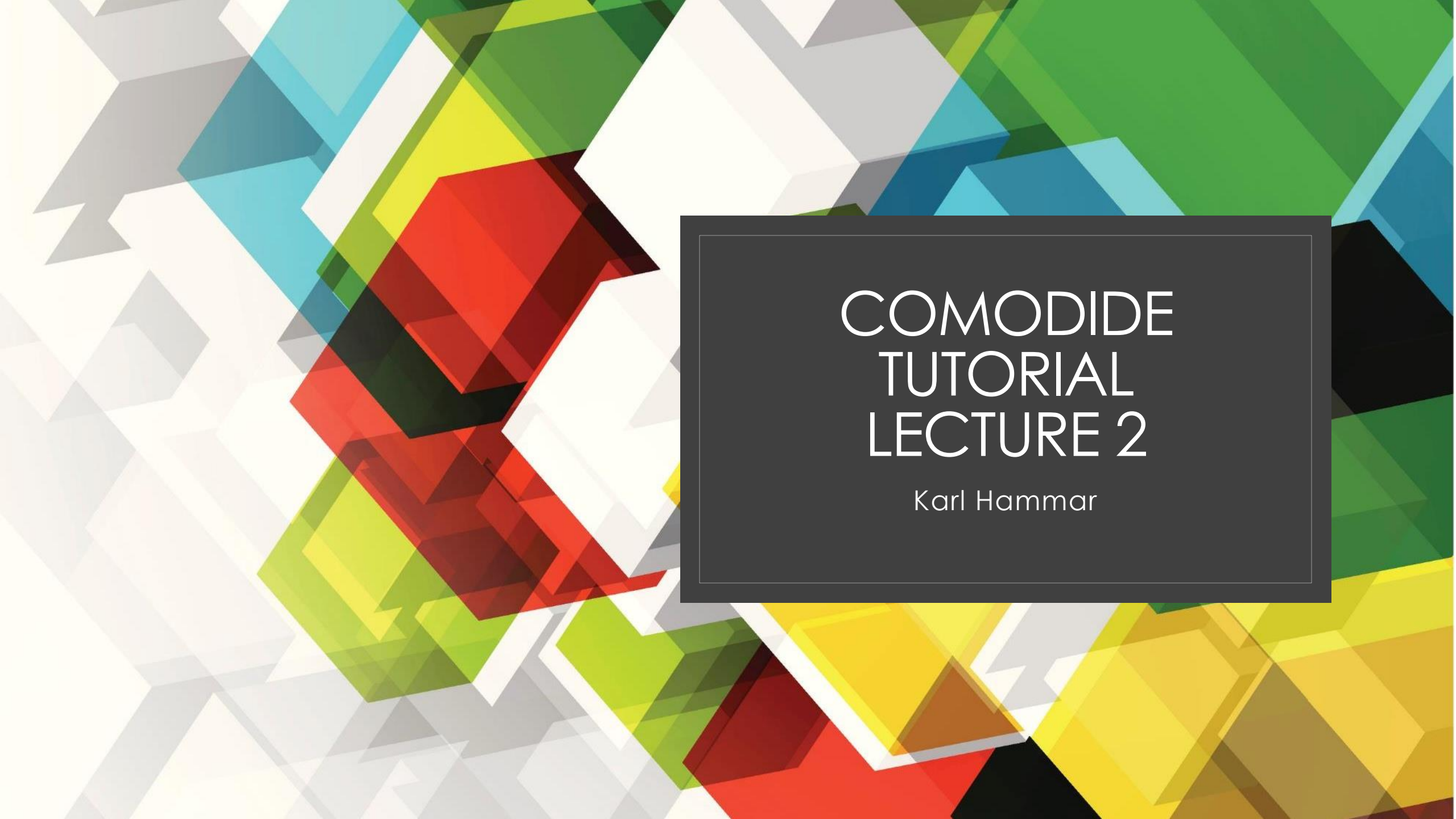
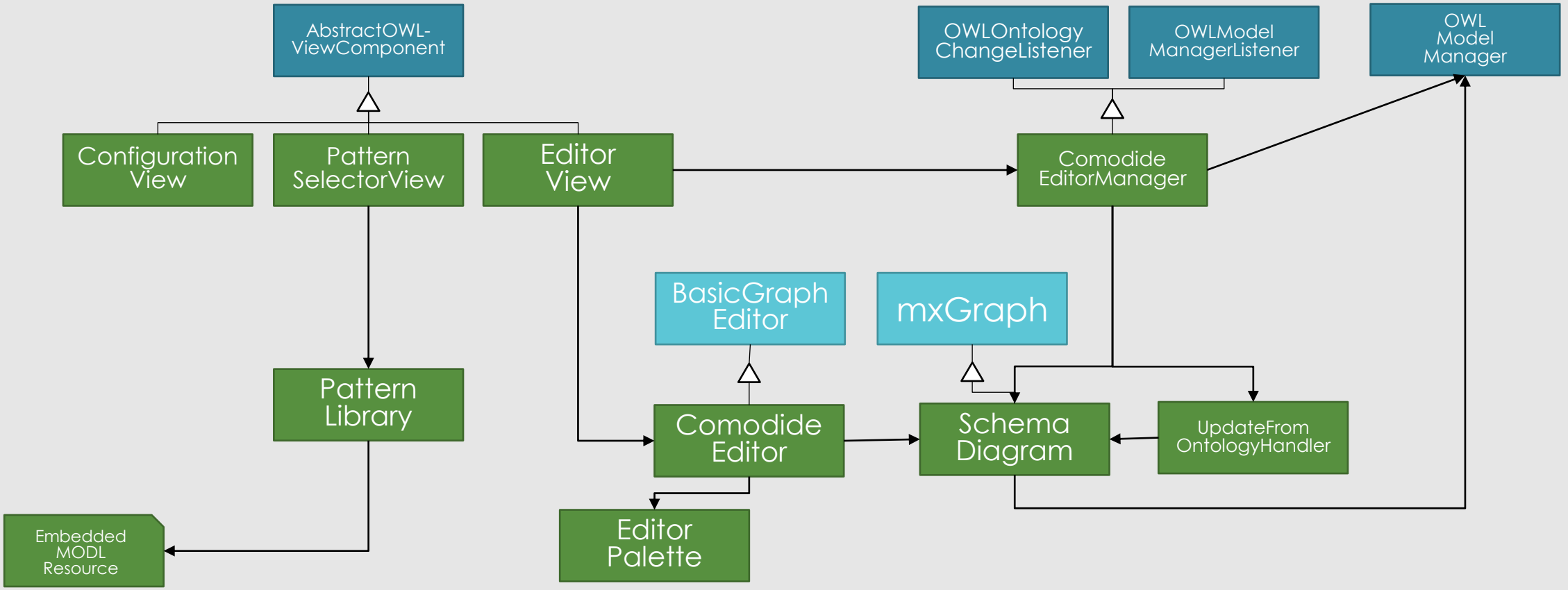


COMODIDE TUTORIAL LECTURE 2

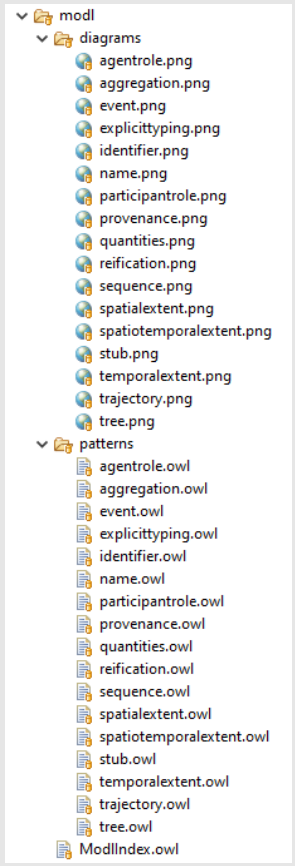
Karl Hammar

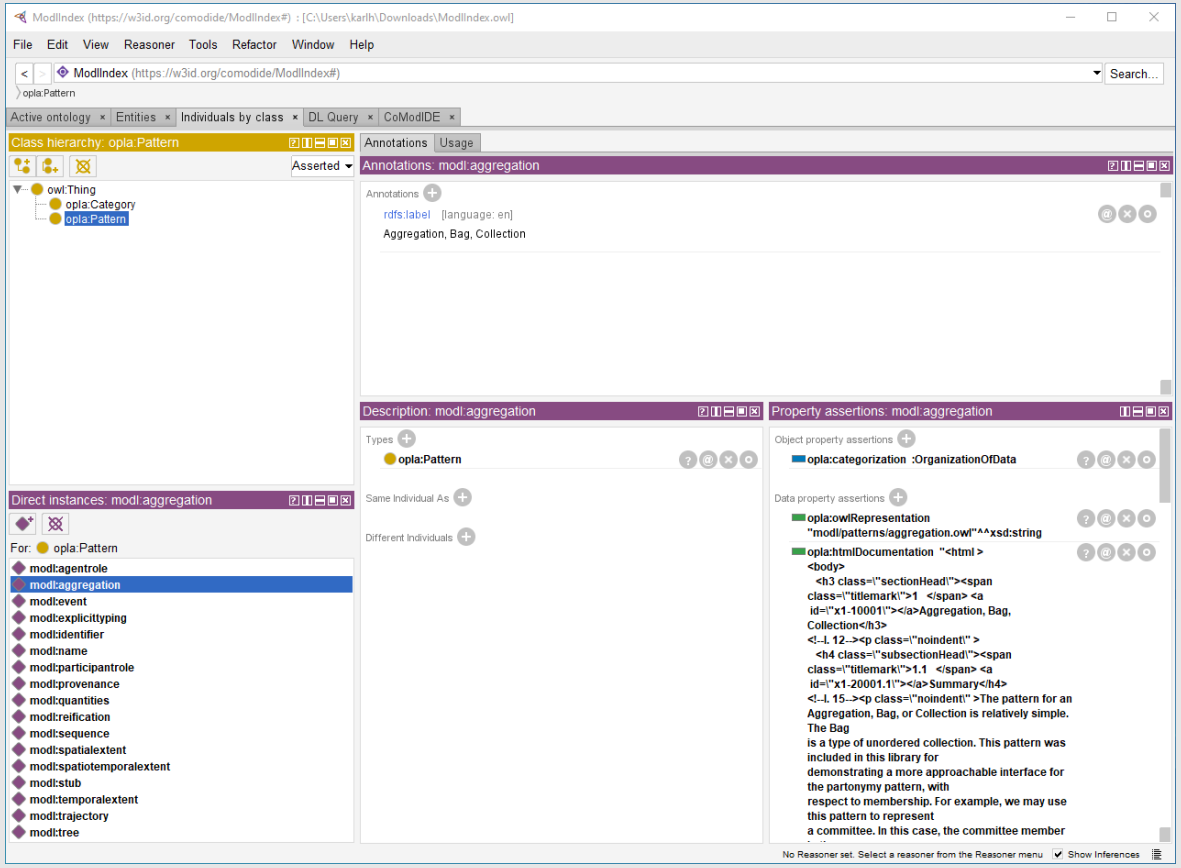
CoModIDE Architecture Overview



2020-05-27

Embedded MODL Library



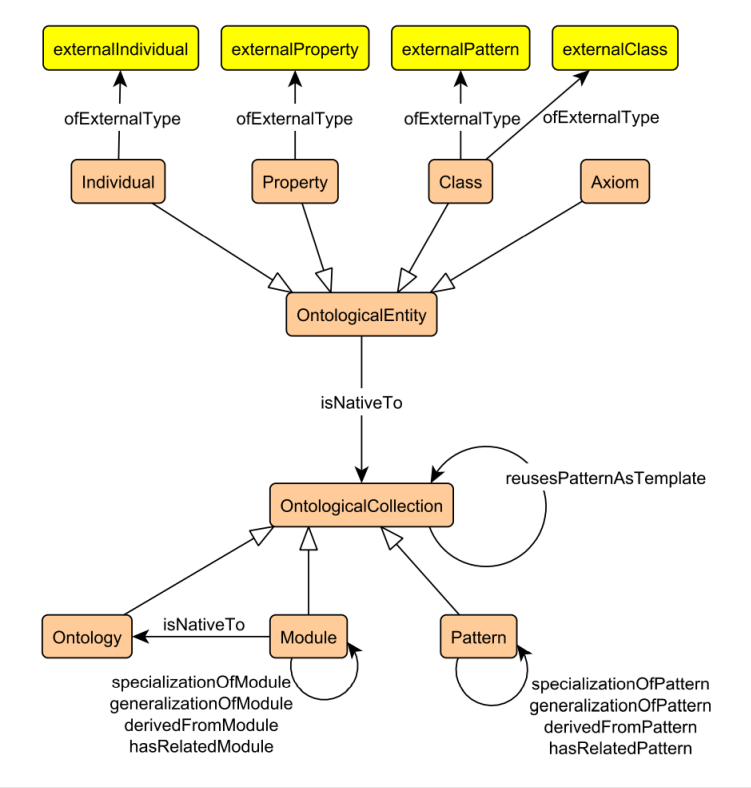


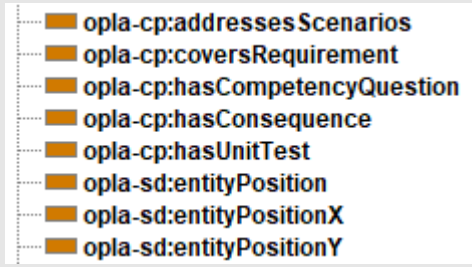
2020-05-27

Constructing ODPs

- Top-down vs bottom-up design
  - Construct reasonable model based on understanding of shared reality
    - Makes for nicely coherent catalogues
    - Tends toward overcomplication
  - Extract patterns from existing ontologies or other models
    - Guarantees real-world relevance and applicability
    - Tends toward duplication of effort
- Modelling simplicity
  - Goal is to communicate the proposed design solution. Simplicity is paramount.
- Choose good names
  - Classes: Not too abstract, not too concrete
  - Properties: Not too short, not too long
- Documentation
  - Graphical illustration*
  - Example use*
  - Description
  - Competency questions
- Annotate the module
- Reuse the module*

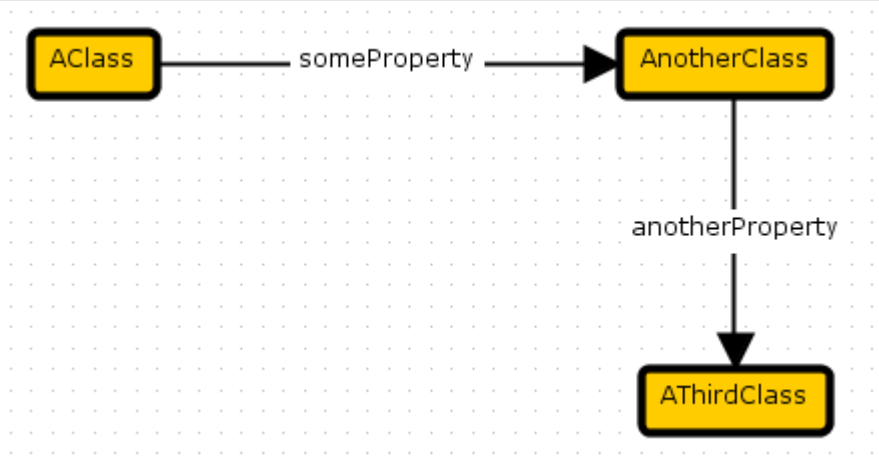
OPLa Annotations





https://github.com/cogan-shimizu-wsu/Extended-OPLa/

Edge Inspector



Edge creation axioms:

- [ ] RDFS Domain/Range
- [x] AllValuesFrom constraint
- [x] SomeValuesFrom constraint

Equivalent To

anotherProperty some AThirdClass

SubClass Of

anotherProperty only AThirdClass

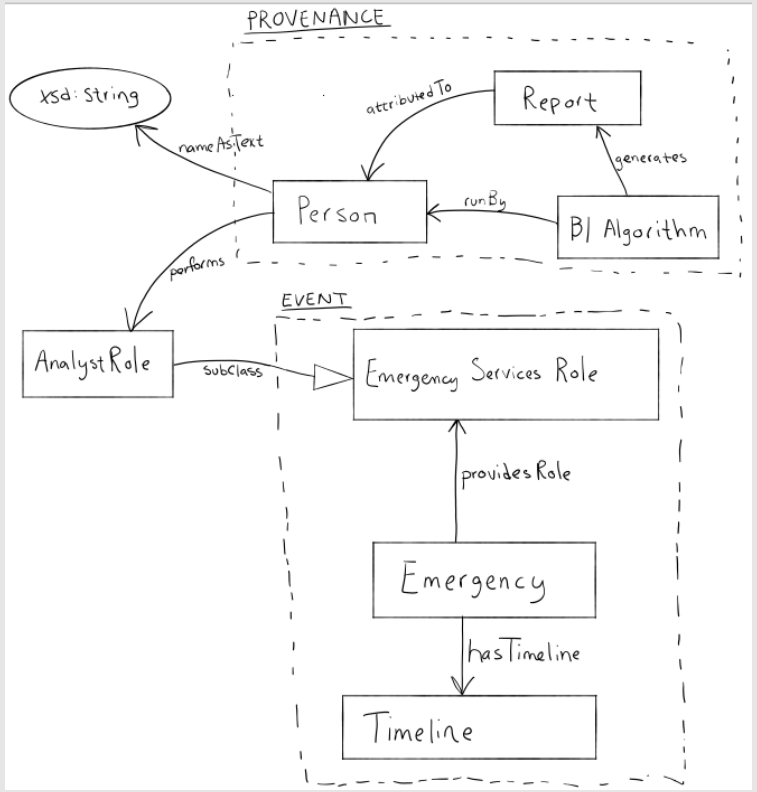
General class axioms

anotherProperty some AThirdClass SubClassOf AnotherClass

Custom pattern library

- No-brainer
- But which libraries?
  - They are all rather poor..
- Annotation support tooling may be needed
- However: potentially very useful for reuse within projects, where a project-specific pattern repository can be configured

Grouping and folding of modules



- Folding into self-contained units of functionality
- Necessary for scaling up graphical modelling
- We already have the OPLa annotations and basic support machinery – just need to render it

Automatic Module Composition

- "Snapping" together of modules as they are dropped.
- Requires some notion of slots and fillers, or interfaces and implementations.
- Could be expressed using OPLa or using basic OWL constructs.