Validating Consistency between Architecture and Design Descriptions

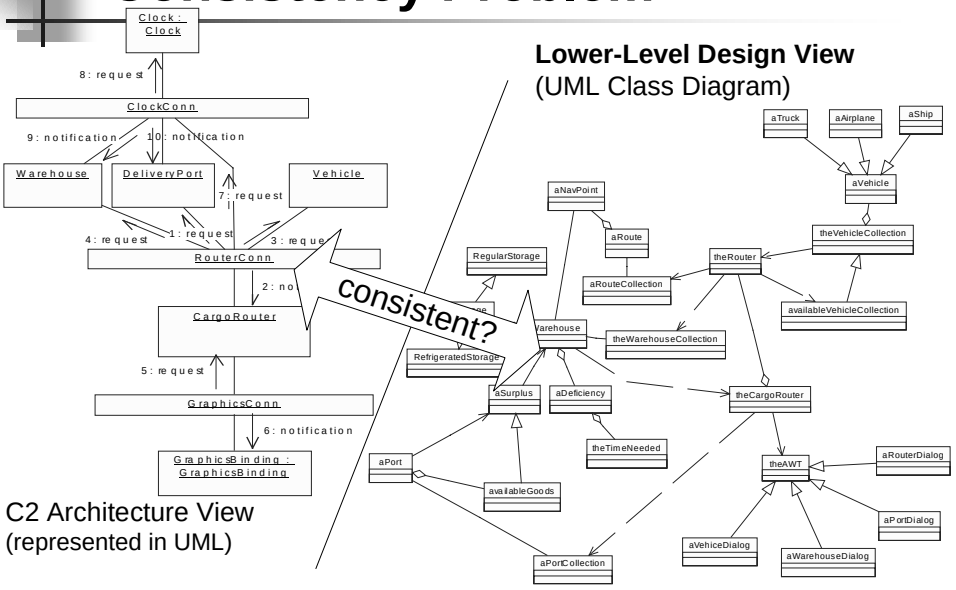
Alexander Egyed
University of Southern California (USC)

Sponsors: DARPA/EDCS, USAF/IFTD, and USC-CSE Affiliates

Models and Diagrams

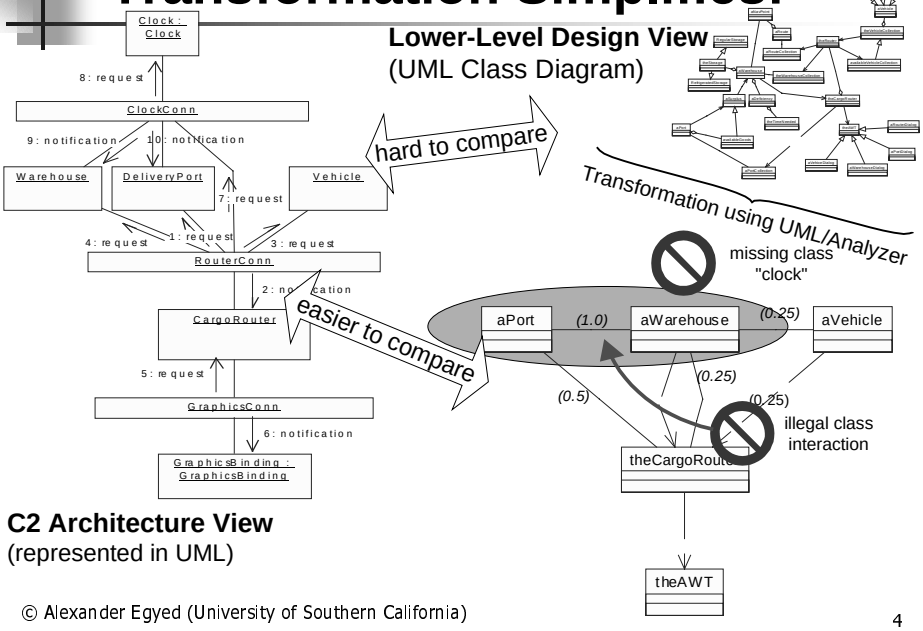
- **Views model (stakeholders) concerns**
- **Many software development models exist – many concerns can be modeled**
- **Different views for different audiences**
- **Diagrammatic and textual views**
- **Independent but connected**
- **Going from architecture to design and back**

Consistency Problem



C2 Architecture View
(represented in UML)

Transformation Simplifies!



C2 Architecture View
(represented in UML)

Comparison Rules

Concrete relation has no corresponding abstraction:

$\forall$ r $\in$ relations, is_abstraction(r) $\wedge$ is_interpretation(r) $\Rightarrow$ realization(r) $\neq$ NULL

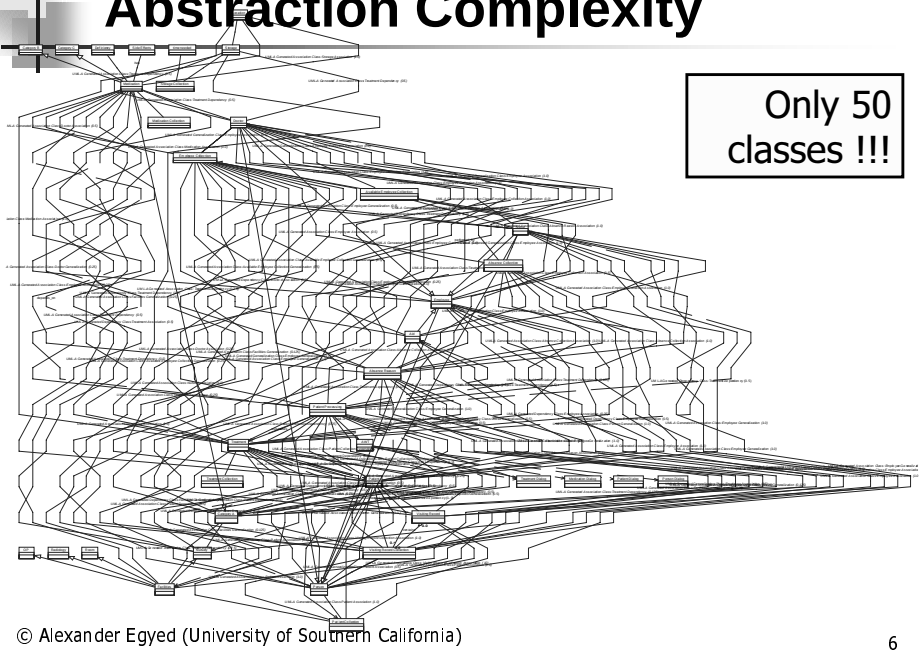
Cardinality of refinement does not match abstraction:

$\forall$ r $\in$ relations, has_realization(r) $\wedge$ is_abstraction(r) $\wedge$ (type(r) = association) $\wedge$ (type(realization(r)) = association) $\Rightarrow$ cardinality(r) = cardinality(realization(r))

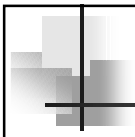
Abstract classifier has not been refined:

$\forall$ c $\in$ classifiers, is_realization(c) $\wedge$ is_refineable(r) $\Rightarrow$ $\exists$ ic $\in$ c->interpretations, is_abstraction(ic)

Abstraction Complexity



Only 50 classes !!!



Conclusion

- Model-based development improves complexities of large scale systems (separation of concerns)
- Models are independent but related – a major strength but also a major weakness
- We proposed two concepts on how to reduce error-prone, manual, and repetitive development going from architecture to design
- Our approaches are tool supported