SOFTWARE DESIGN PATTERNS FOR ONTOLOGY-ENABLED TRACEABILITY

Parastoo Delgoshaei and Mark Austin

Department of Civil and Environmental Engineering and ISR,
University of Maryland, College Park, MD 20742.

CSER 2011 Presentation, Redondo Beach, CA, April 15, 2011

OUTLINE

Part 1: Background and Motivation (Mark Austin)

1. A quick look at Research and Education at ISR,
2. Project Motivation / Observations from the Master of Science in Systems Engineering Program
3. State-of-the-Art Traceability,
4. Ontology-Enabled Traceability Mechanisms,
5. Prototype Implementation: Washington DC Metro System.

Part 2: Current Work (Parastoo Delgoshaei)

1. Objectives and Scope for the Current Project,
2. Design Patterns,
3. Model-View-Controller Design Pattern,
4. System Architecture,
5. Results of Software Experiments,
6. Future Work and Potential Benefits.

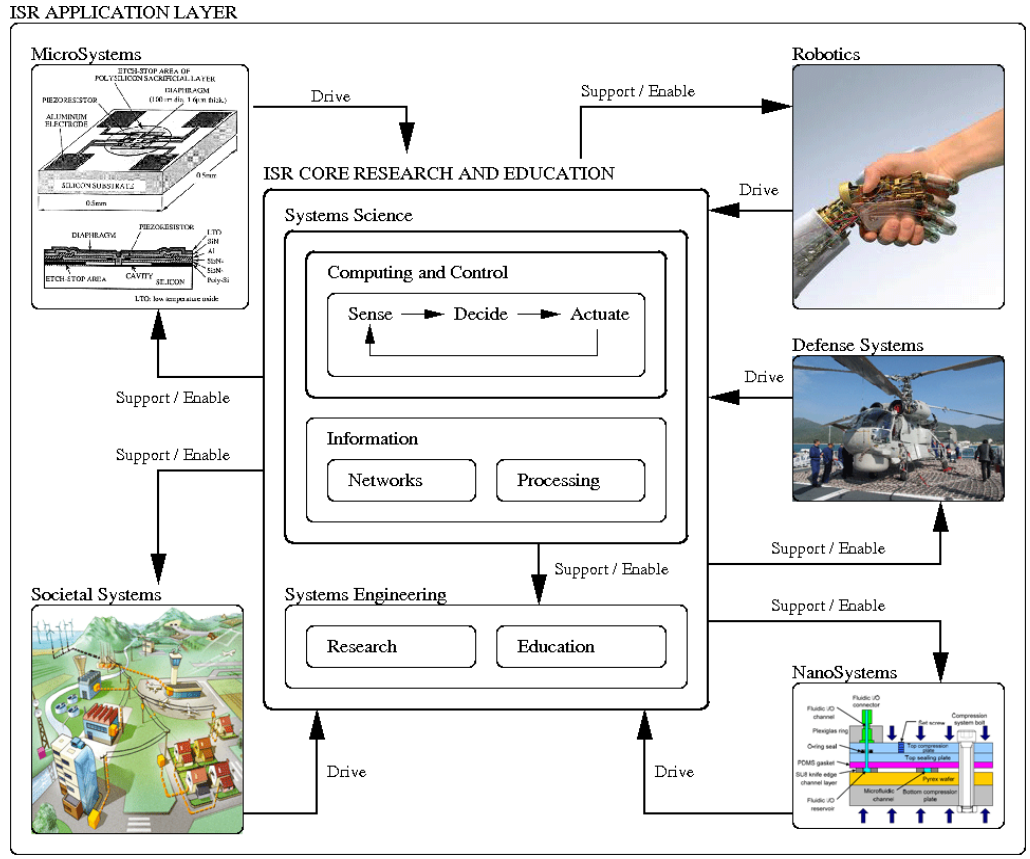
RESEARCH AND EDUCATION AT ISR

Faculty: 40 faculty working on research and education in the system sciences and systems engineering.

Application areas: Microsystems, Societal Systems, Robotics, Defense Systems, Nanosystems.

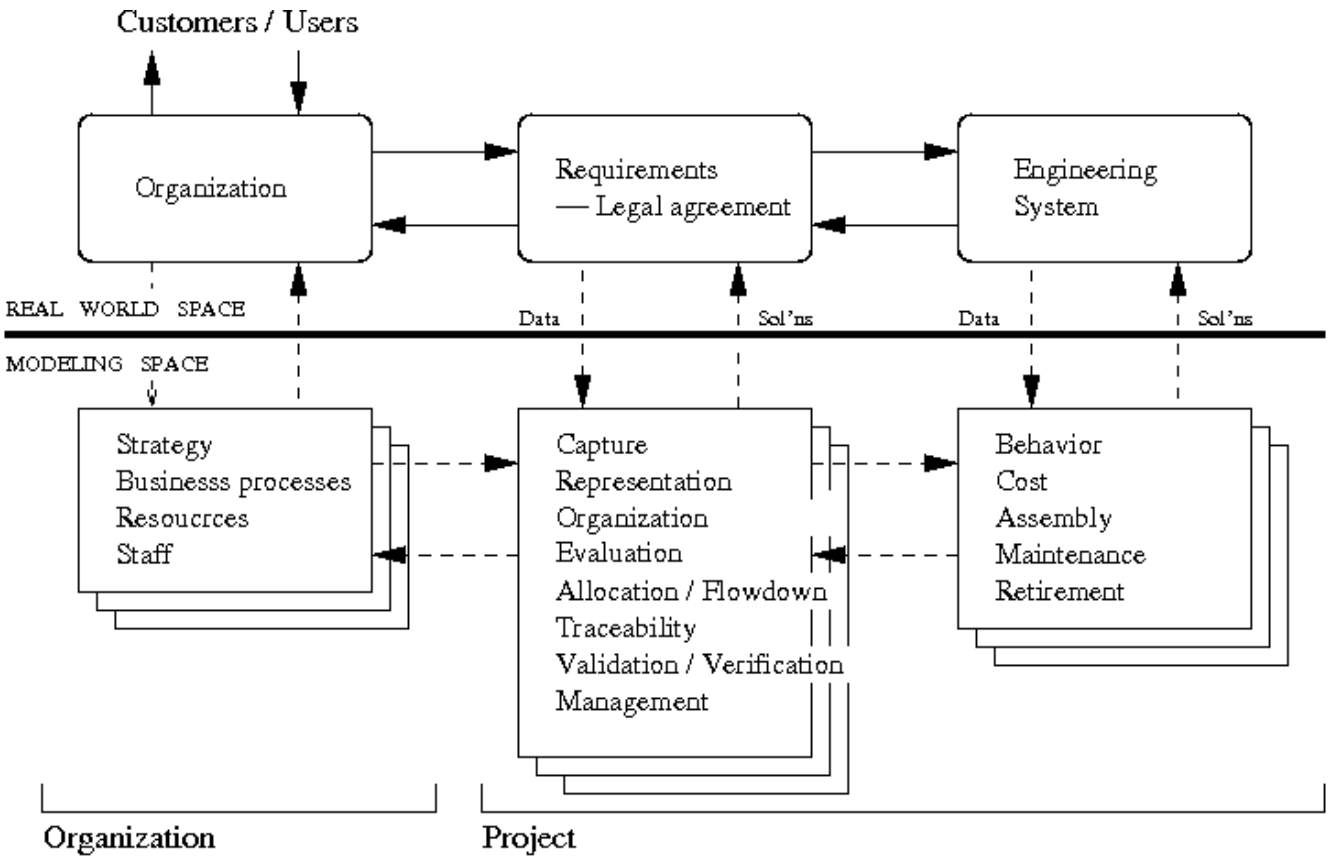
Education: Master of Science in Systems Engineering.

Moving forward: Research and education needs to be integrated together.



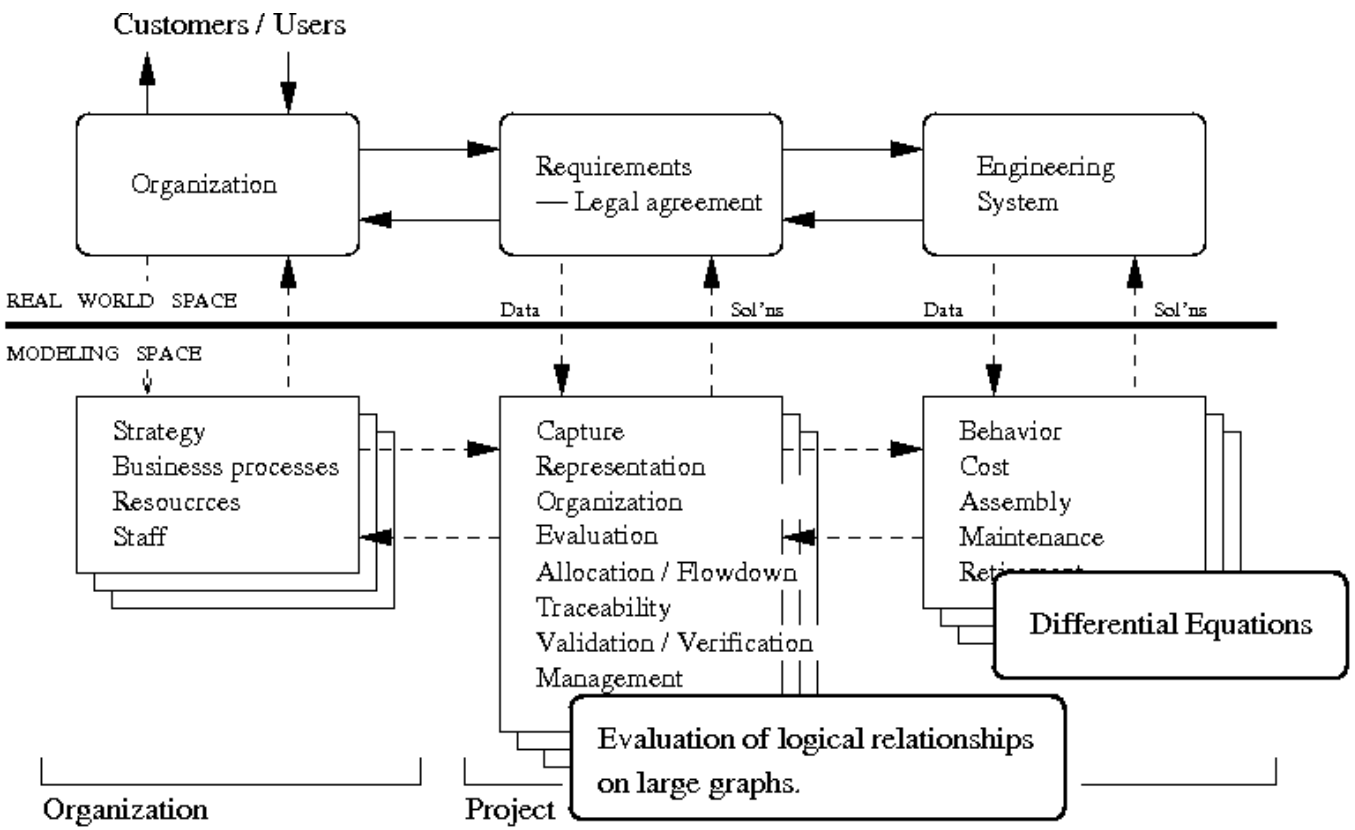
PROJECT MOTIVATION / MSSE EDUCATION

Create Big-Picture View: Students need to understand how things fit together.



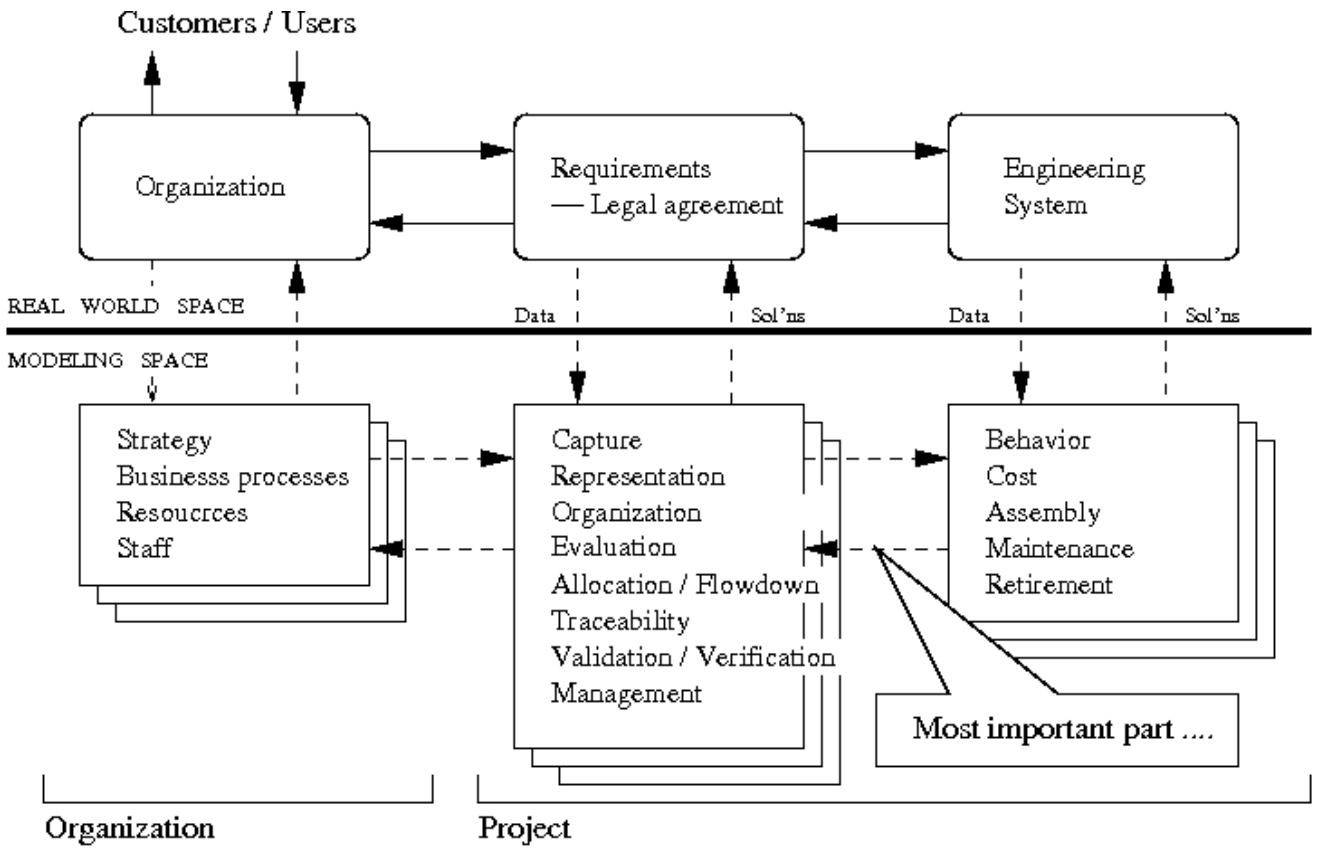
PROJECT MOTIVATION / MSSE EDUCATION

Focus on Model-Based Systems Engineering: The mathematics needed for formal approaches to systems engineering is foreign to many engineers.



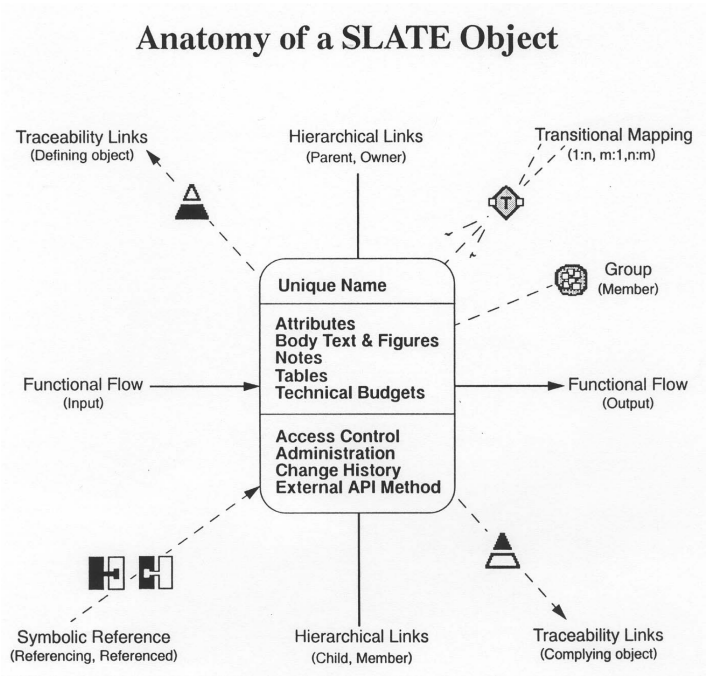
PROJECT MOTIVATION / MSSE EDUCATION

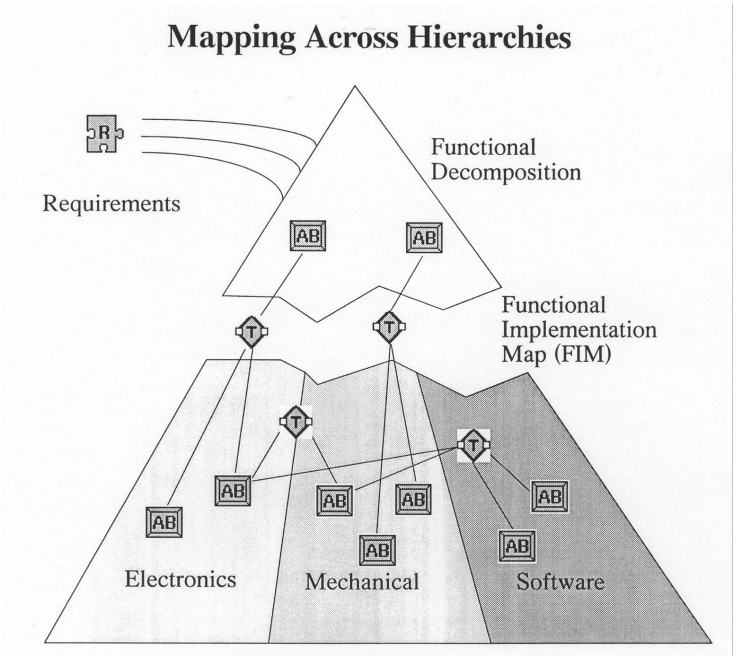
Observation: Traceability connects things together …. and without it, you cannot manage the consequences of change. Also, your boss won't understand how you add value to the organization.



STATE-OF-THE-ART TRACEABILITY

State-of-the-Art Traceability with SLATE..

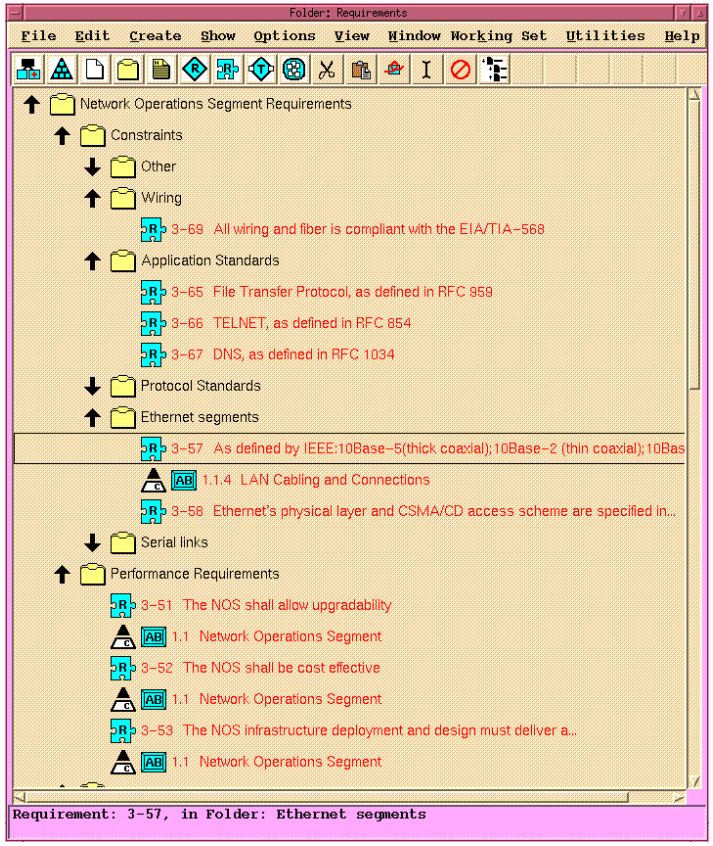


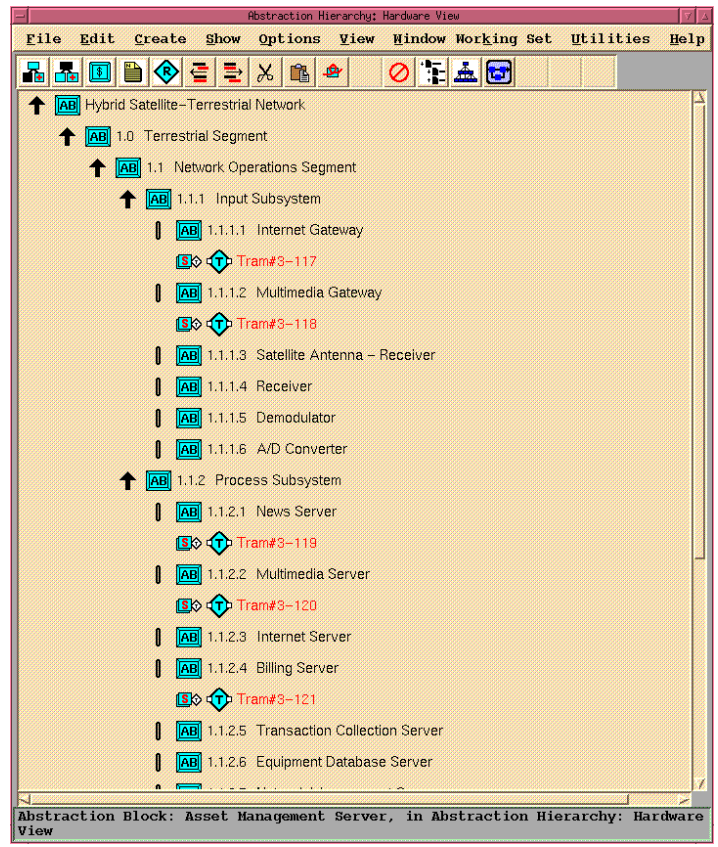


Note: Use of abstraction blocks only makes sense at the earliest stages of development, and where a system doesn't already exist. Doesn't apply for SoS.

STATE-OF-THE-ART TRACEABILITY

Visualization of traceability relationships is far from intuitive.





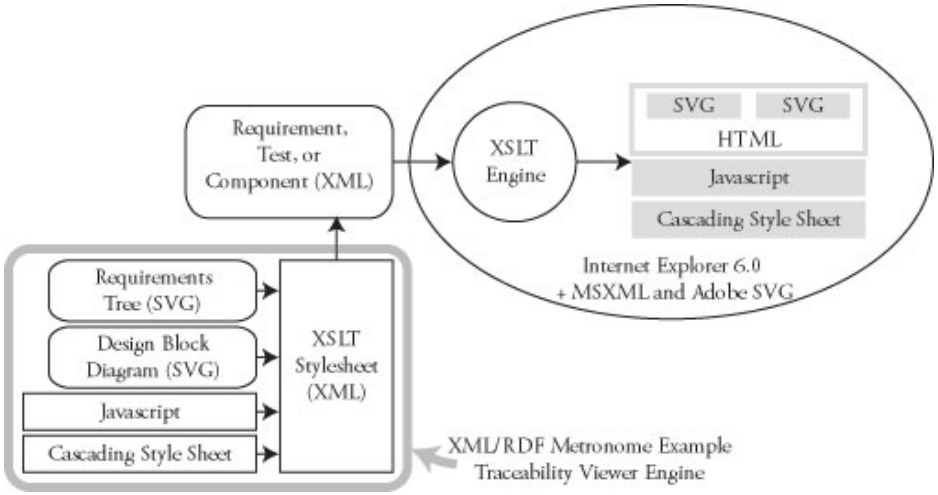
Most engineers want to visualize system developments using notations they are familiar with.

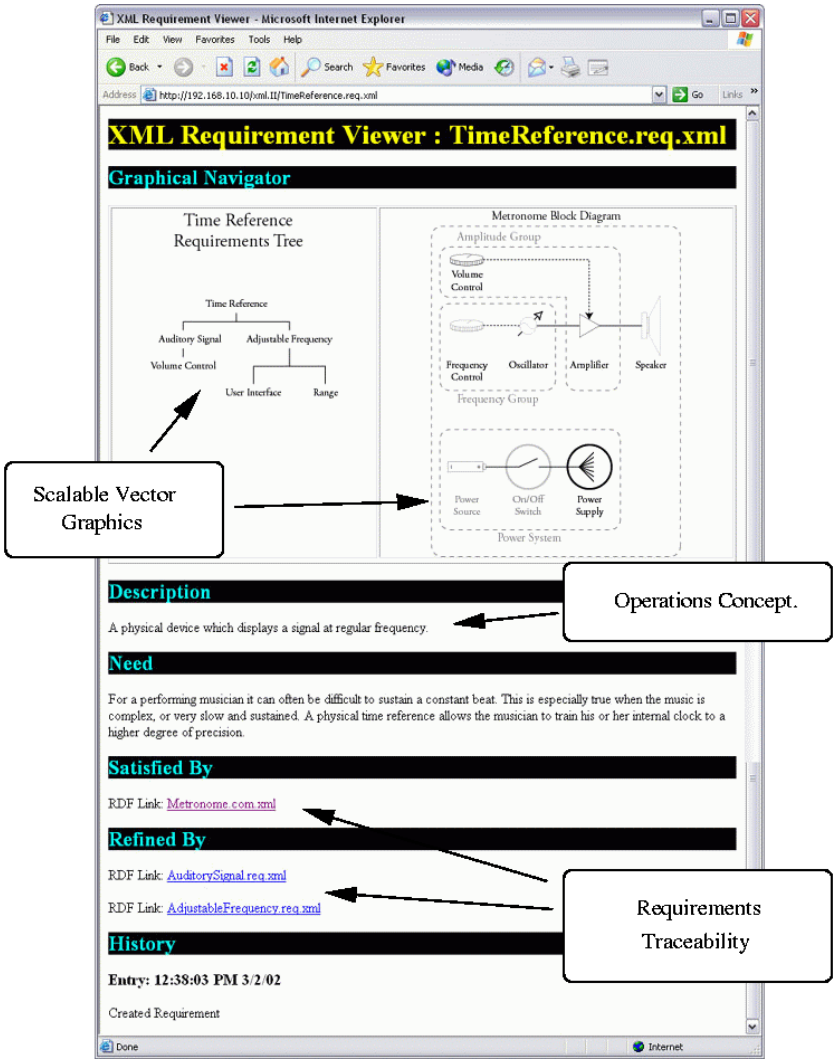
IMPROVING UPON STATE-OF-THE-ART TRACEABILITY

Surely we can do better!!!

Our first step: Explore use of XML and RDF technologies to improve visualization of requirements traceability.

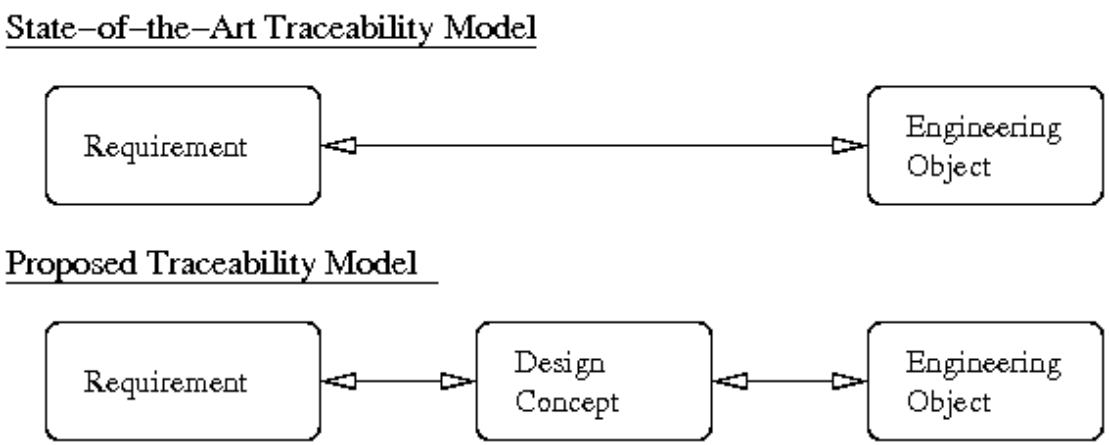
Credit: Web prototype developed and implemented by Scott Selberg in 2003.





HERE'S WHAT'S NEW …..

New idea: Ontology-enabled Traceability Mechanisms.

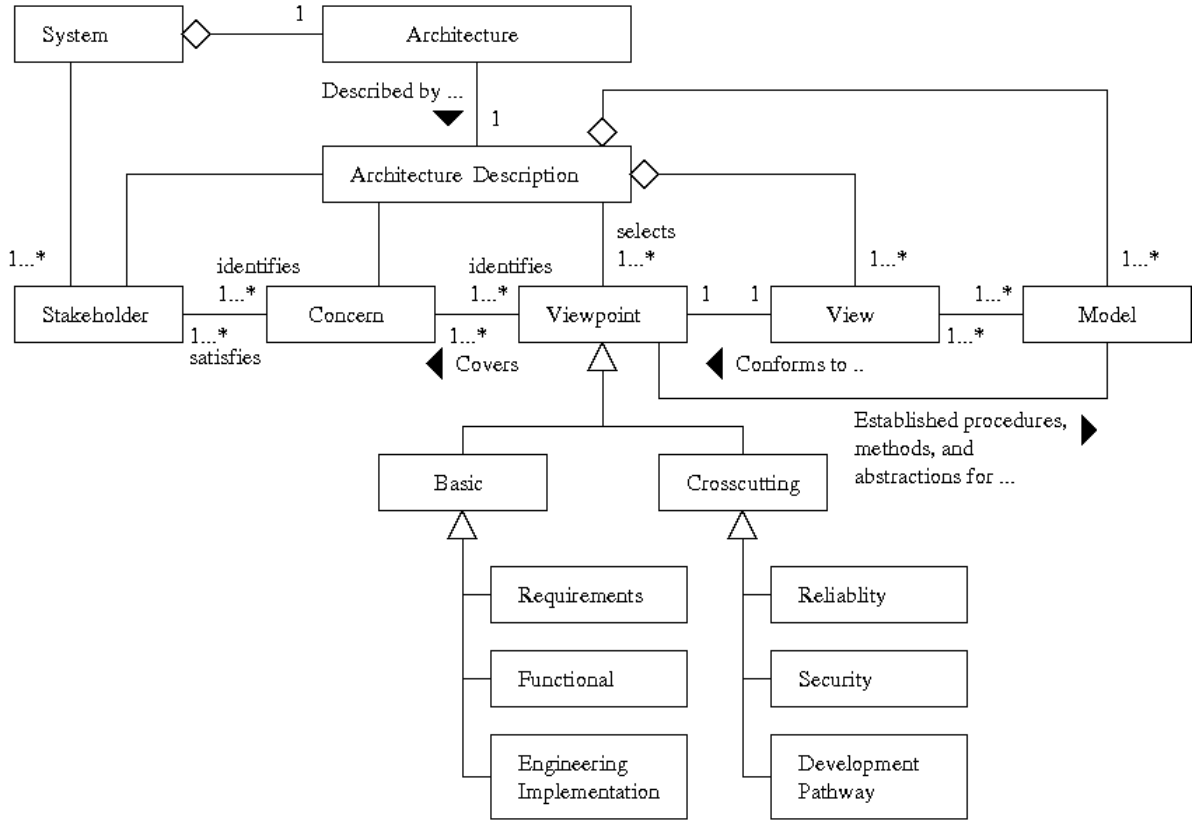


Approach: **Requirements are satisfied through implementation of design concepts**. Now traceability pathways are threaded through design concepts.

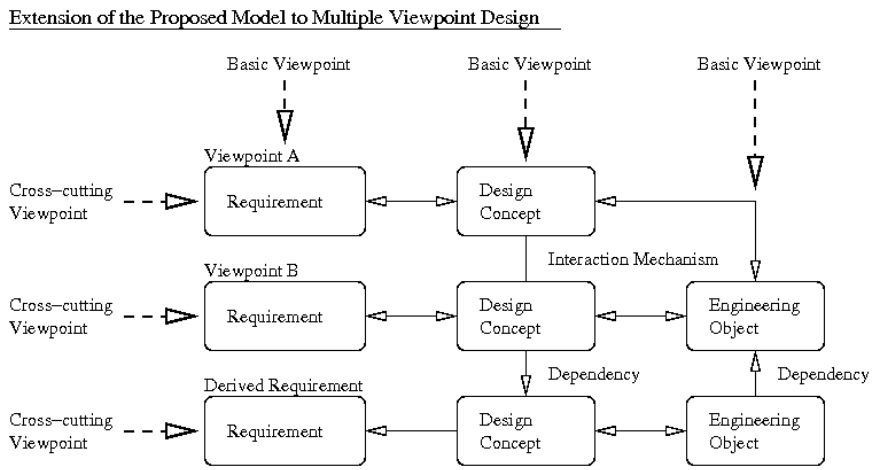
Key Benefit: Rule checking can be attached to "design concepts" – therefore, we have a pathway for **early validation**.

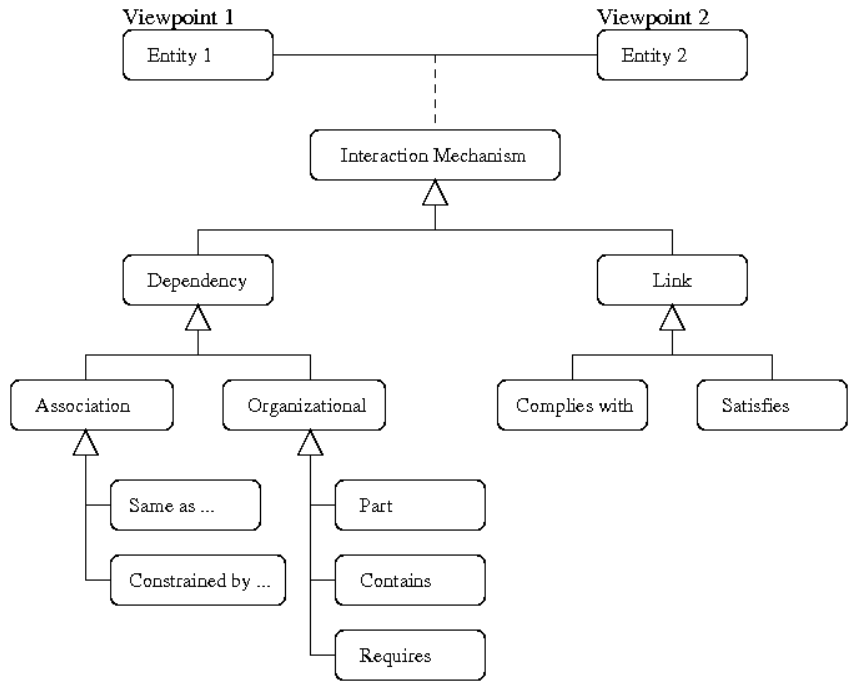
SUPPORT FOR MULTIPLE-VIEWPOINT DESIGN

Team-based design is a multi-disciplinary activity. We need a model for multiple-viewpoint design and mechanisms for capturing interactions between design concerns.



SO HOW MIGHT ONTOLOGY-ENABLED TRACEABILITY FOR MULTIPLE-VIEWPOINT DESIGN WORK?





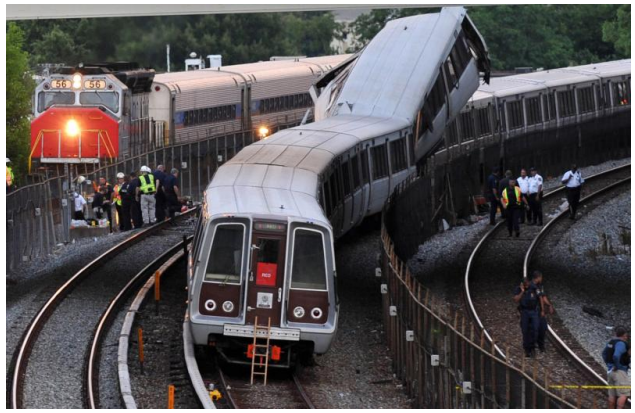
Multiple-viewpoint ontology-enabled traceability will correspond to graph of design entities: requirements, ontologies, and engineering objects.

We need models to capture the various mechanisms of interaction between viewpoints.

PROTOTYPE IMPLEMENTATION: WASHINGTON D.C. METRO SYSTEM

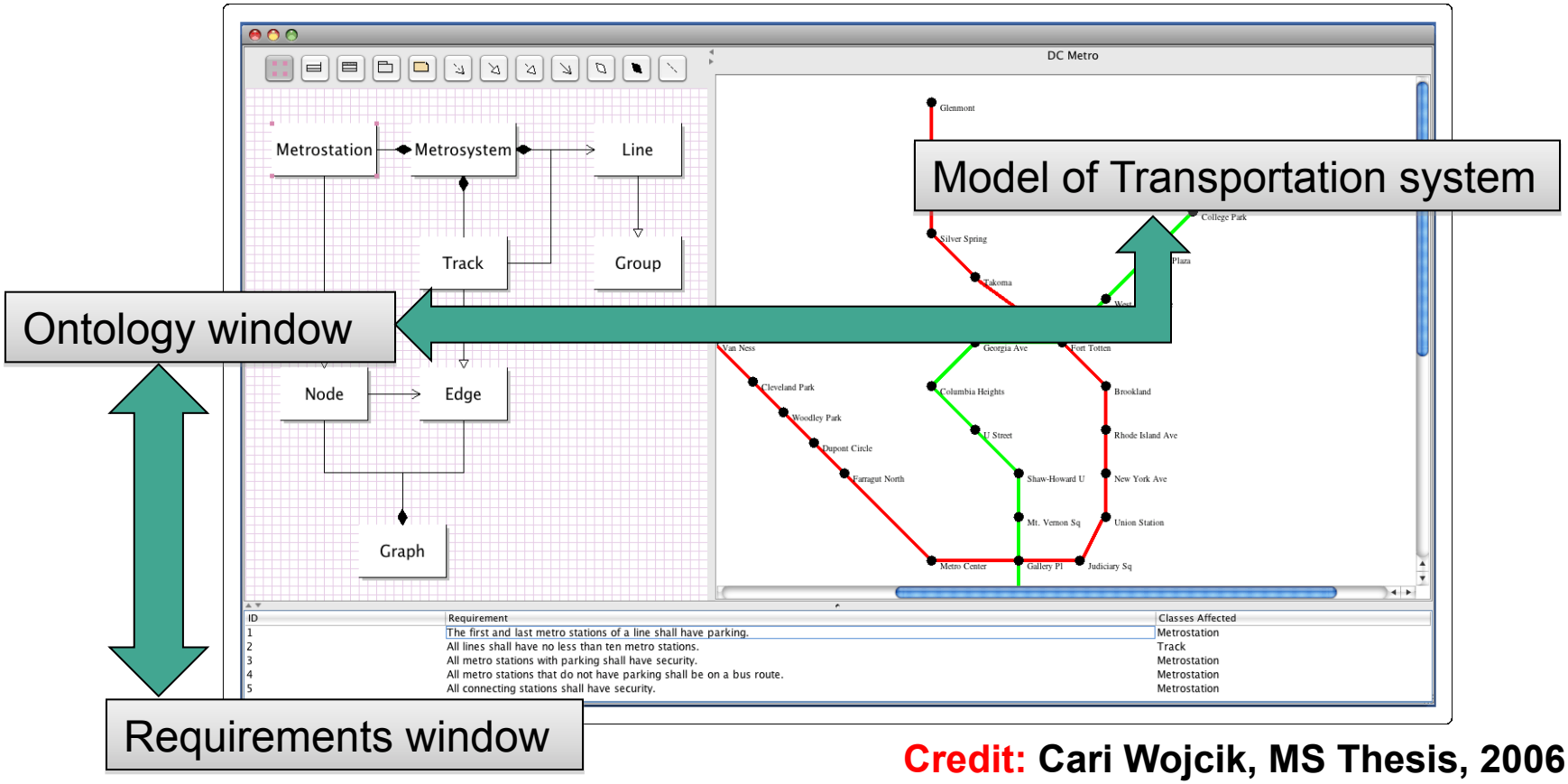
Statistics: Second largest rail transit system in the US; serves a population of 3.5 million; 200+ million passenger riders per year.

Safety-critical system: Major accident in June, 2009.



PROTOTYPE IMPLEMENTATION: ONTOLOGY-ENABLED TRACEABILITY FOR WASHINGTON D.C. METRO SYSTEM.

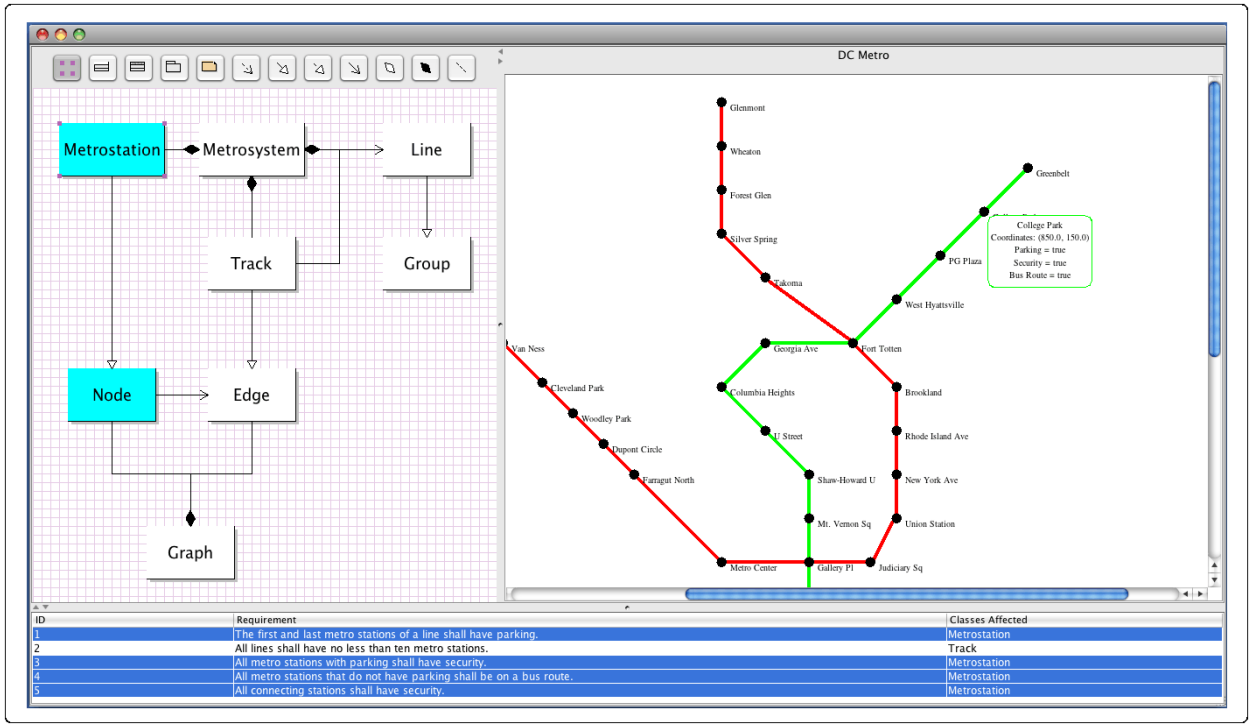
Very simple. UML representation for one ontology. All traceability relationships are hard-coded. Visualization cuts across stages of system development.



Credit: **Cari Wojcik, MS Thesis, 2006.**

PROTOTYPE IMPLEMENTATION: ONTOLOGY-ENABLED TRACEABILITY FOR WASHINGTON DC METRO SYSTEM.

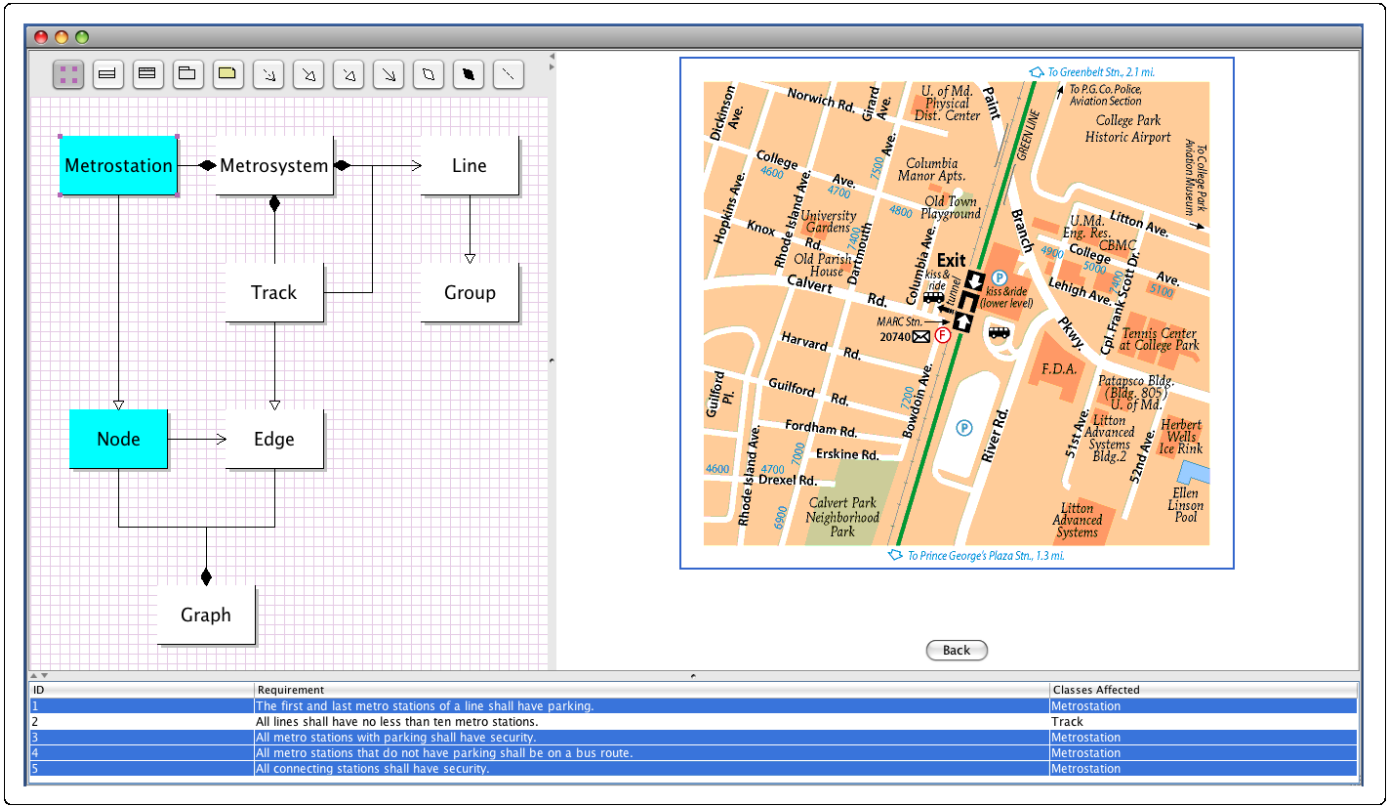
Designers are provided with mechanisms to interact with the system in multiple ways.



Traceability relationship from the College Park Metro Station back to defining design concepts (MetroStation and Node) and defining requirements.

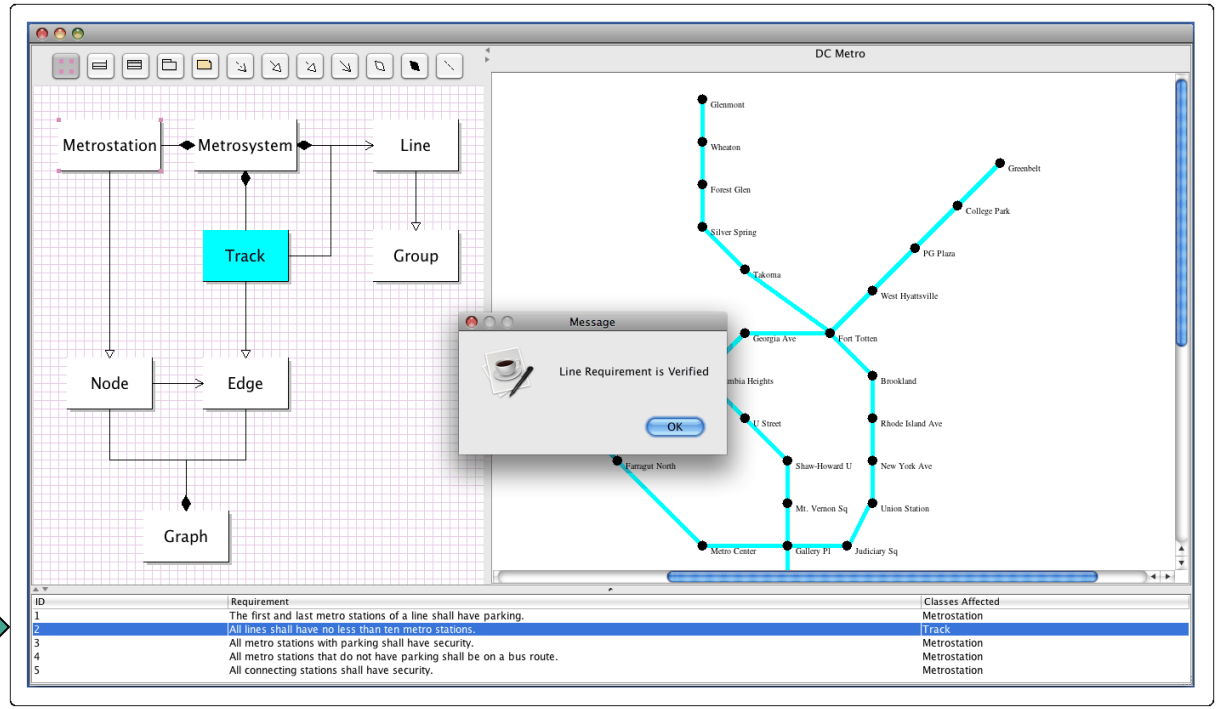
PROTOTYPE IMPLEMENTATION

Detailed Map View of the College Park Metro Station



PROTOTYPE IMPLEMENTATION: ONTOLOGY-ENABLED TRACEABILITY (WITH VERY BASIC RULE CHECKING).

Key Advantage: **Design rules and procedures for design rule checking can be attached to ontologies**.



Design rule checking is triggered by double clicking on a requirement. Visualization shows the extent of ontologies and engineering entities involved in the rule checking.

FAST FORWARD TO 2010-2011

. **New student:** Parastoo Delgoshaei, MS Thesis.

ASSESSMENT OF VERSION I

Software prototype works great, but support for scalability is missing.

Focus on system architecture (stations and tracks). Where are the trains?

Weak support for multiple viewpoints, integration of functionality and performance, and tradeoff analysis.

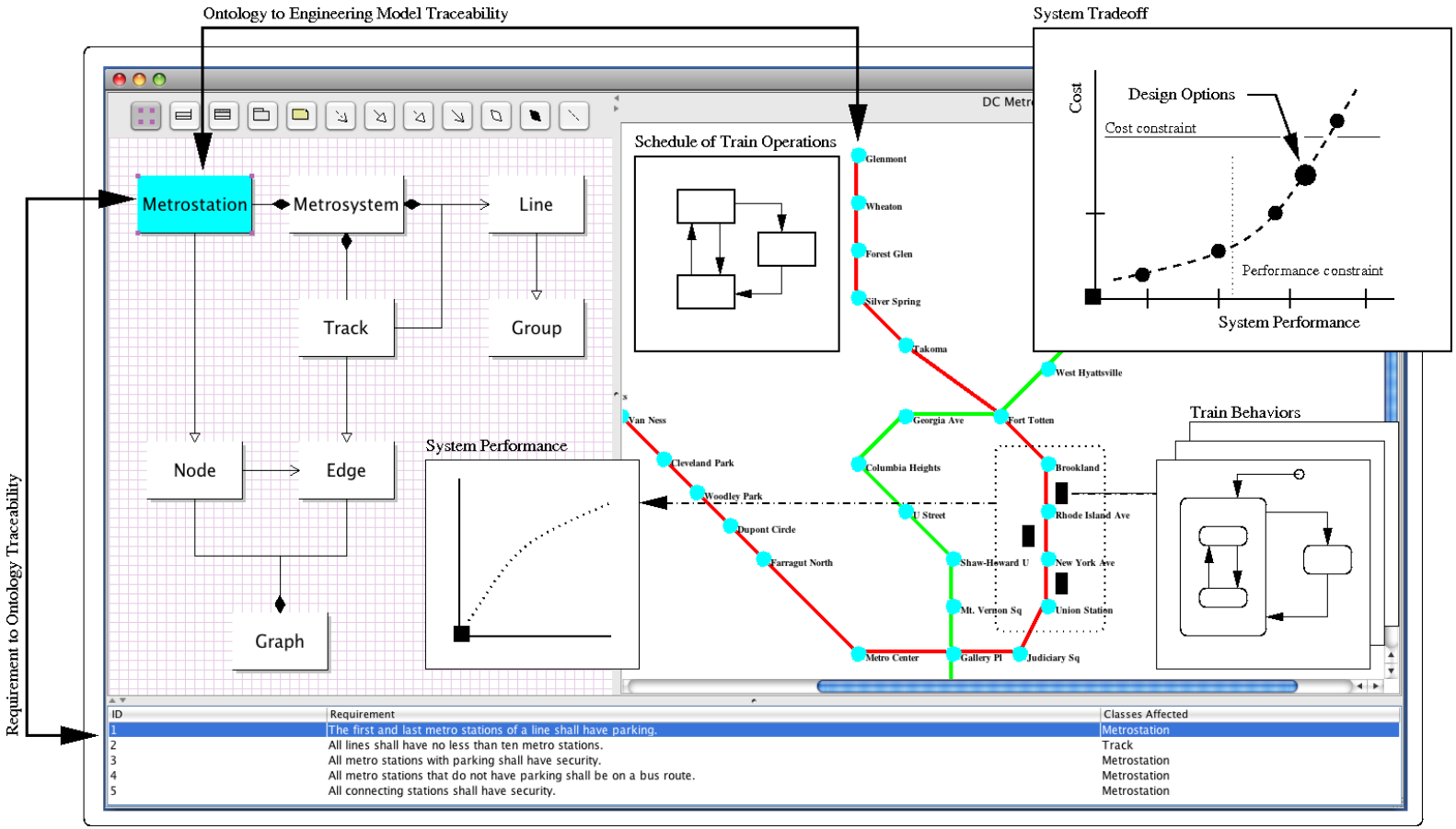
PLAN OF WORK FOR VERSION II

Re-design implementation to maximize use of software design patterns.
Model system schedules and train behavior with finite state machines.

System-level behavior will correspond to a network of communicating finite state machines.

Project status: We are building software prototypes -- work in progress!

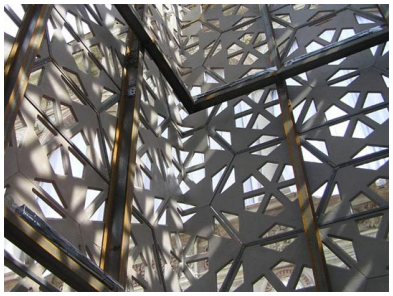
SCHEMATIC FOR VERSION II



DESIGN PATTERNS

Motivation: Experienced designers know that instead of returning to first principles, routine design problems can be best solved by adapting solutions to designs that that have worked well for them in the past.

Definition: A design pattern is simply….



A description of a problem of a recurring problem, and,
A description of a core solution to that problem stated in such a way that it can be reused many times.

Software Design Patterns: A few examples …

Behavioral	Structural	System
Command	Adapter	Model-View-Controller
Interpreter	Bridge	Session
Mediator	Composite	Router
Observer …	Decorator …	Transaction …

MODEL-VIEW-CONTROLLER DESIGN PATTERN

Approach and Benefits

Divide a component or subsystem into three logical parts – model, view, controller – making it easier to modify or customize each part.

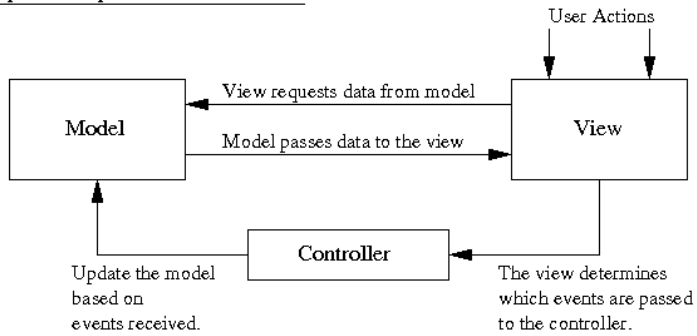
Purpose of Logical Components:

Model: Store the element's state and provide a means for changing the state.

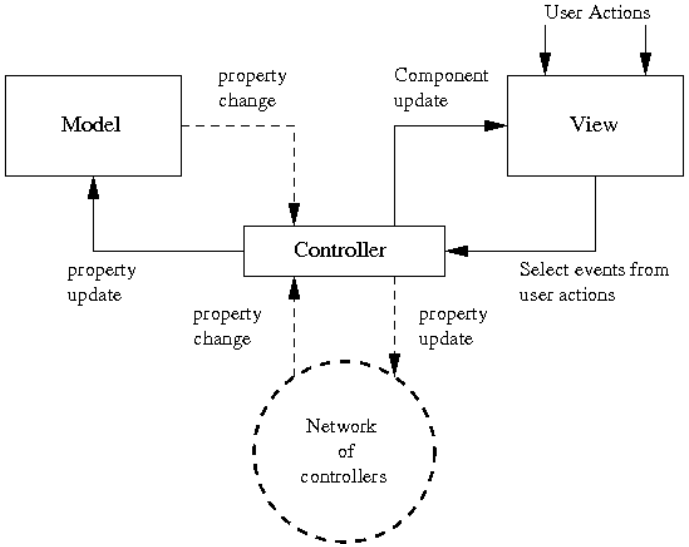
View: Representation of the component or subsystem.

Controller: Map incoming actions to their impact on the model.

Simplified Implementation of MVC

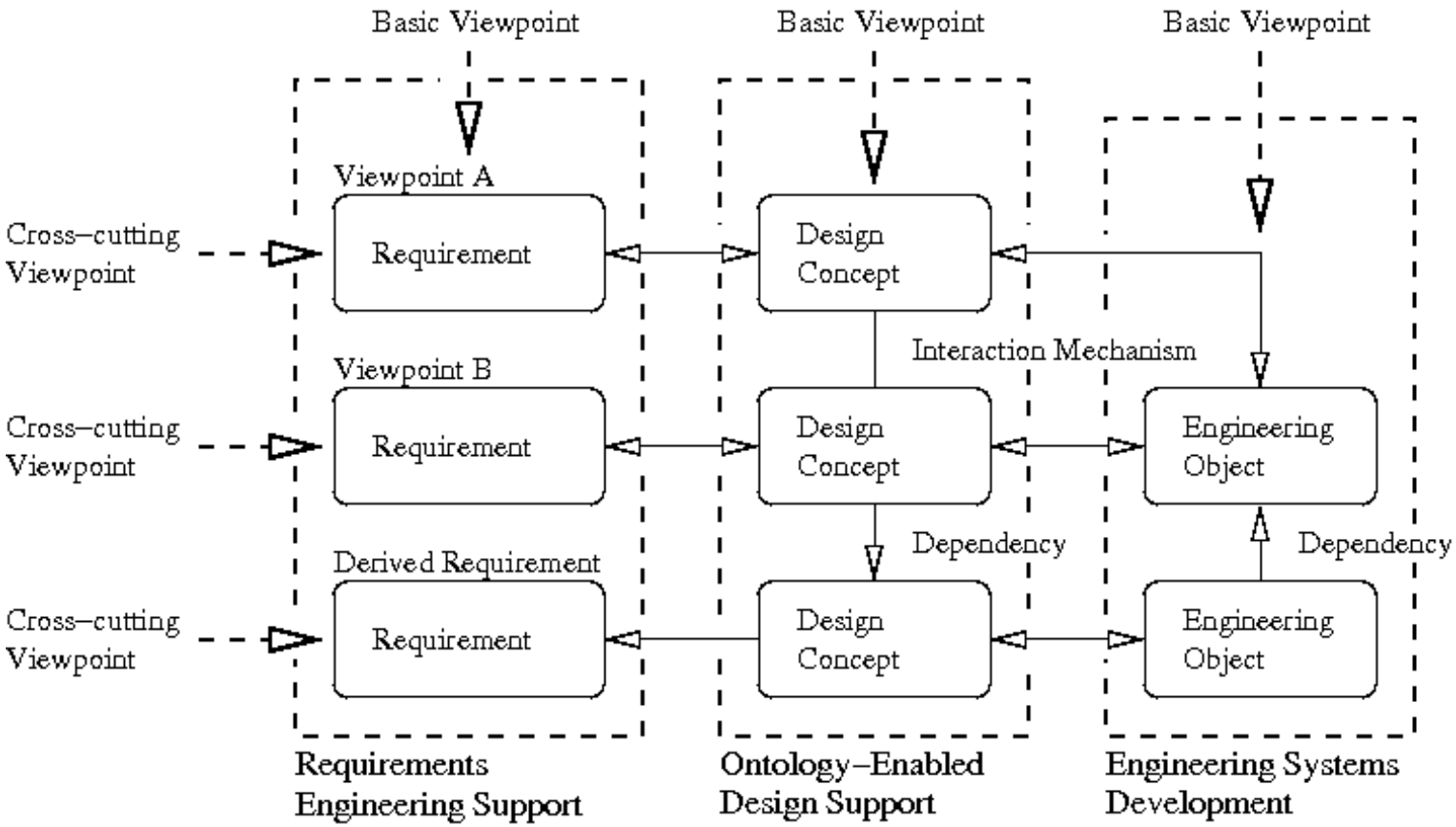


Implementation of MVC with the Controller acting as a Mediator



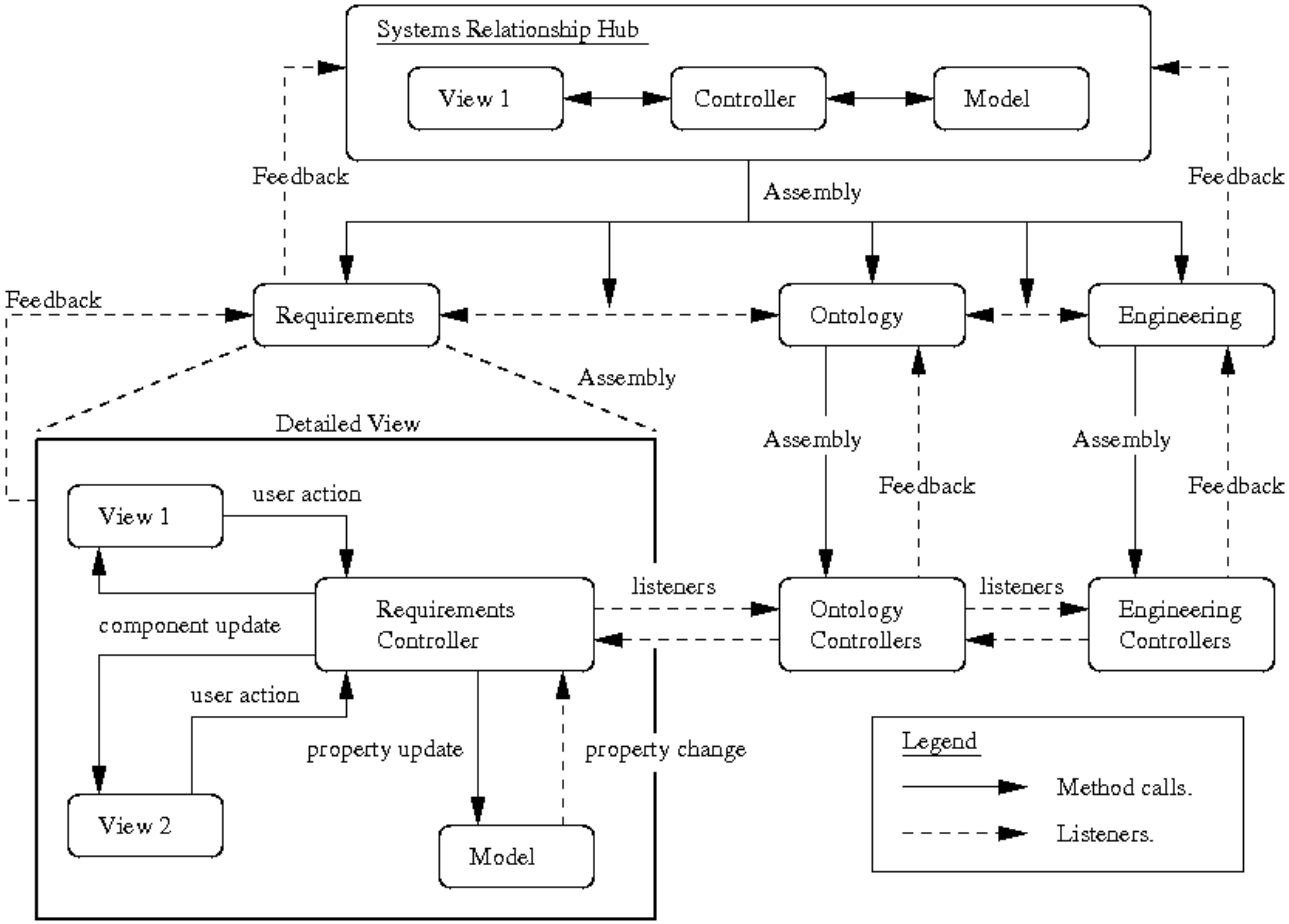
SYSTEM ARCHITECTURE

Architecture for Software Support



SYSTEM ARCHITECTURE

New pattern: Pyramid of model-view-controllers.

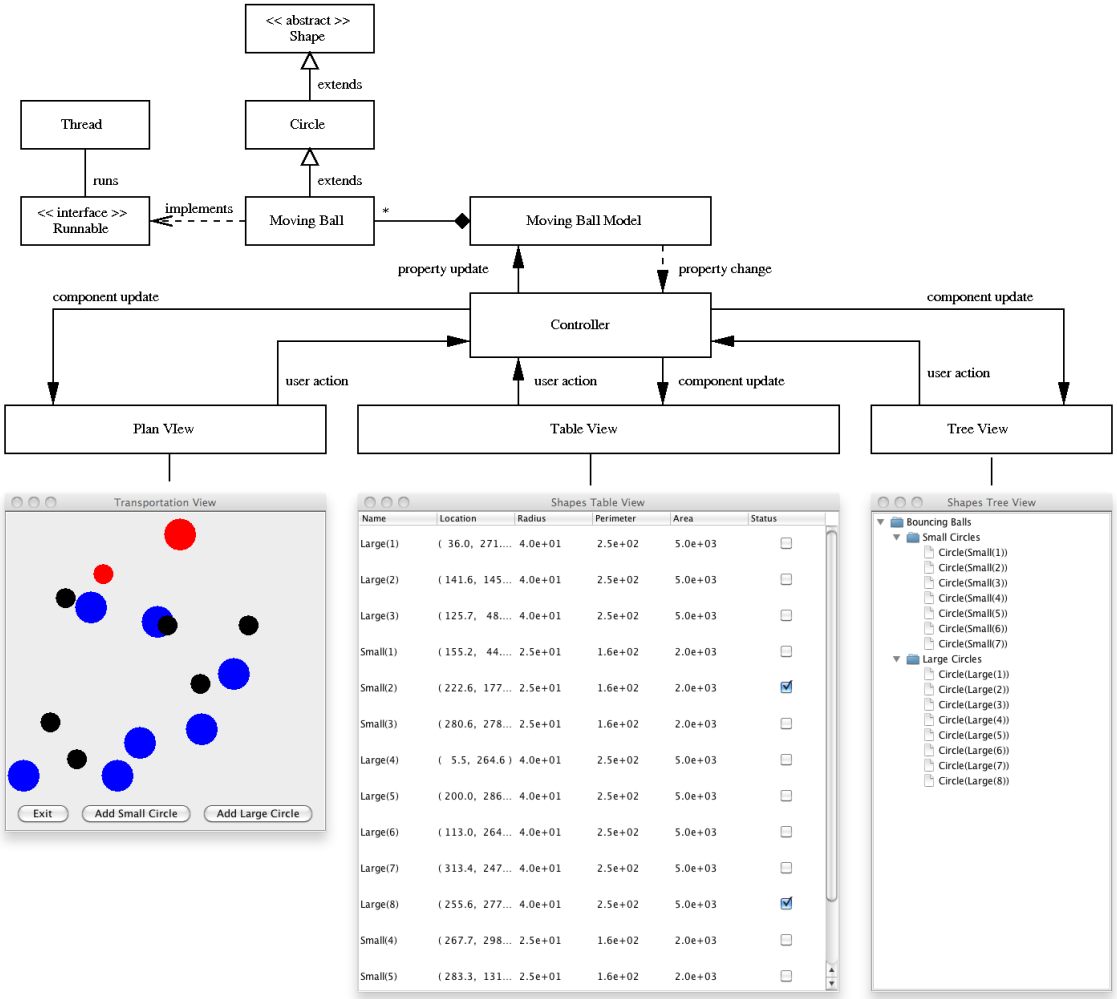


SOFTWARE EXPERIMENTS

Challenge: Traceability and visualization support in systems having many concurrent behaviors.

Prototype: Bouncing balls interface implemented as threads of behavior.

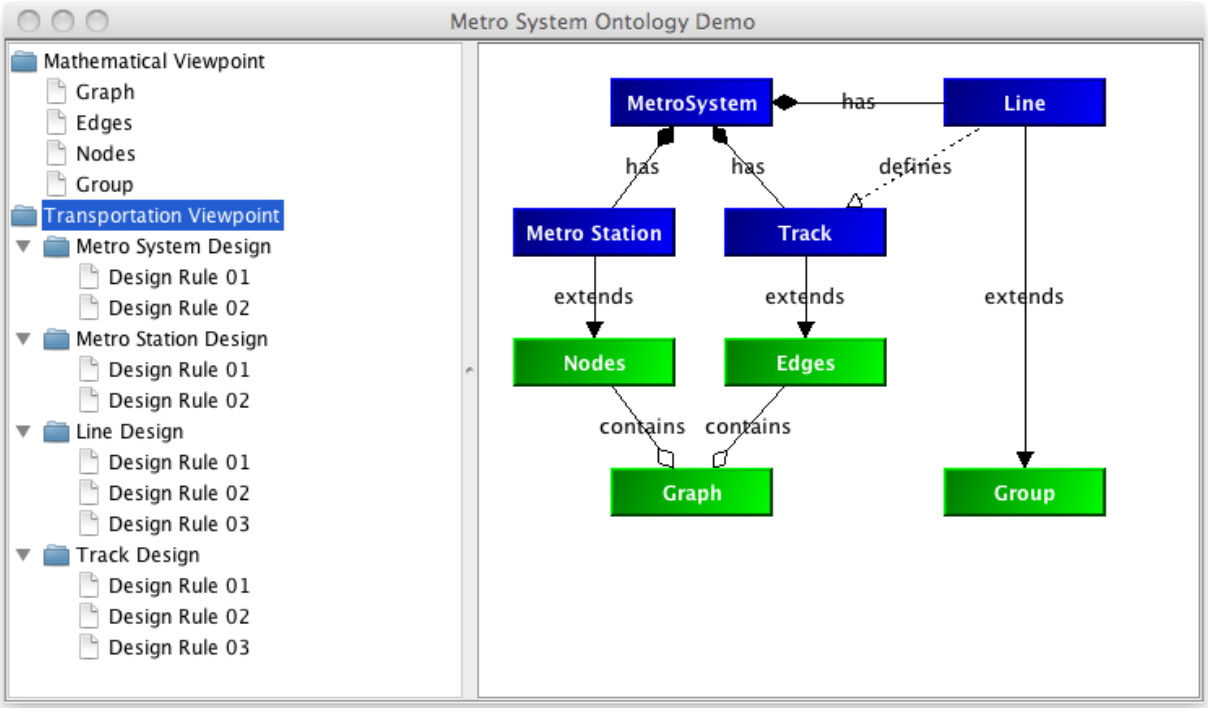
Use adapter, observer, and model-view-controller design patterns to facilitate traceability and support for table and tree viewpoints.



SOFTWARE EXPERIMENTS

Challenge: We need support for modeling and visualization of graphs from multiple perspectives.

Prototype: Use adapter, observer and model-view-controller to synchronize visualizations in response to user inputs/actions.



FUTURE WORK AND POTENTIAL BENEFITS

Proposed Work (2012 and beyond):

1. Explore feasibility of extending ontology-enabled traceability mechanisms to multiple-viewpoint design,

2. Explore use of Semantic Web Technologies (e.g., OWL = Web Ontology Language and SWRL = Semantic Web Rule Language) for representation of ontologies and rule-checking,

3. Design software infrastructure to conduct system trade studies.

4. Design and implement a scalable, networked, system implementation.

Potential benefits/payoffs?

Fewer design/management errors due to superior representation of traceability relationships; built-in support for design rule checking at the earliest possible moment; improved economics of system/SoS development and management.

THE END!

Questions?