

# Static and Dynamic Modeling and Analysis of Architectures

Roshanak Roshandel

Computer Science Department  
*University of Southern California*  
roshande@usc.edu

# Motivation

- Software architecture's promise
  - Architecture Description Languages (ADLs)
  - Formal Modeling Notations
- Static vs. Dynamic
- Narrow focus to provide sophisticated analysis capabilities
  - Wright, SADL

# Static Modeling

- Represent system structure in terms of components, connectors and their interconnection
- Provide **declarative** behavior, interaction, and distribution models
  - Specify properties that must hold true at *discrete* points during the system's life span
  - Allow analysis of functional and extra-functional (mis)match
  - Do not specify *how* the properties are to be achieved
- *C2SADEL, UniCon, ACME*

# Dynamic Modeling

- Describes the details of interaction among components and connectors
- Provides **imperative** behavior, interaction, and distribution models
  - A *continuous* view of how to arrive at a given property or desired state
  - Allow analysis and simulation of dynamic behavior
  - Suffer from large state space and (often) implicit model of architectural structure
- *StateCharts, PetriNets, Wright, Rapide*

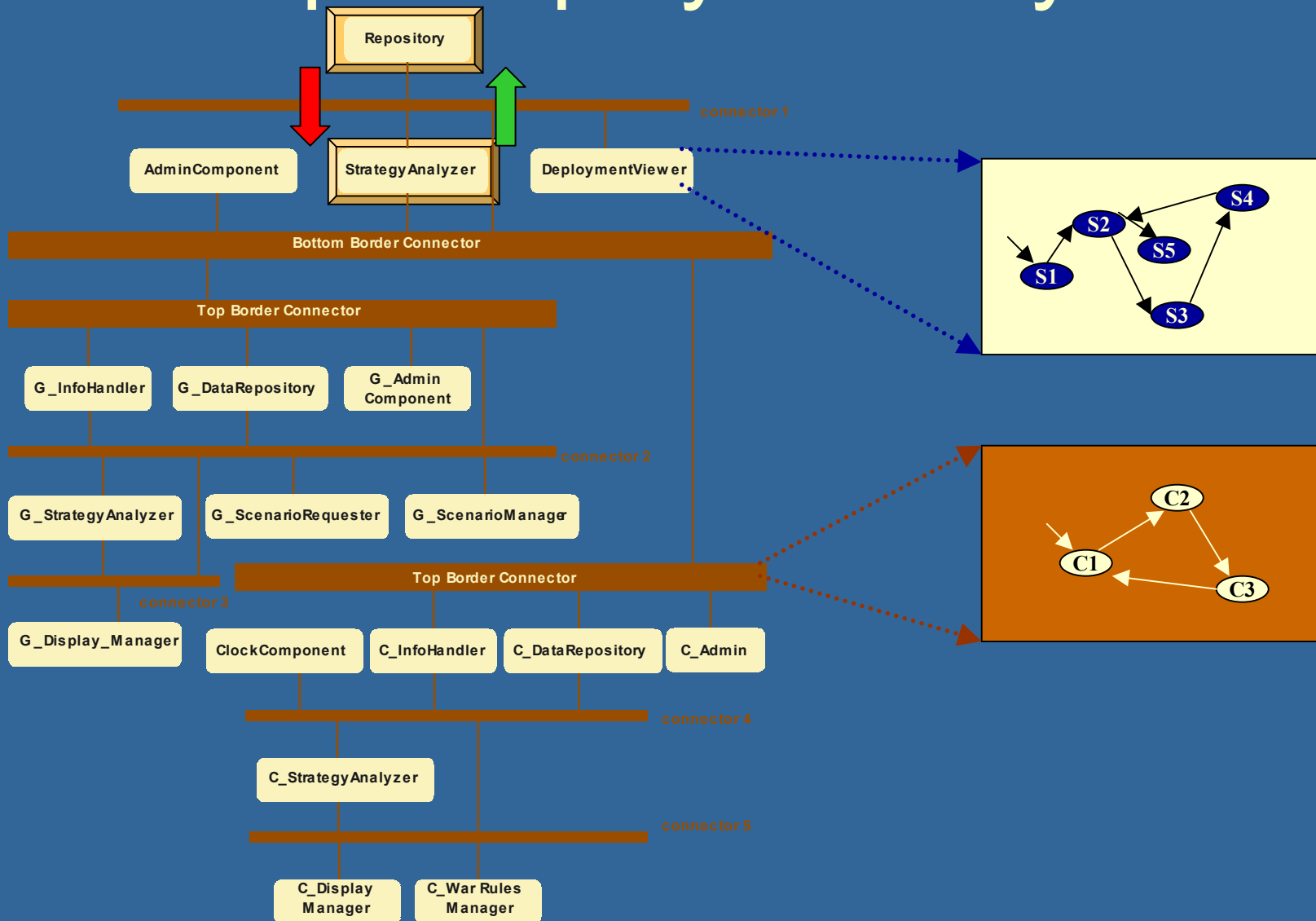
# Coupling Static and Dynamic Modeling

- Expand the scope of architectural analysis
- Static → C2SADEL
  - Design the topology, separate concerns, specify behavioral “snapshots”
- Dynamic → StateCharts
  - Well known, powerful dynamic modeling technique
  - One of the modeling notations in UML
  - Extend FSM with concurrency and hierarchy

# Mapping

C2SADEL	StateCharts
Architecture	Composite State Machine
Components	State Machine
Connectors	State Machine
State variables	State's variable
Invariants	Implicit in designing the states
Required Interface/Operation	Events
Provided Interface/Operation	Actions or events
Interface/Operation parameters	Parameters on transitions
Pre-conditions	Guards
Post-conditions	State entry variable

# Troops Deployment System



## Repository

### Operation

prov pAskMapInfo  
prov pResultMap

connector1

## StrategyAnalyzer

### Operation

req rAskMapInfo  
req rResultMap

### Repository



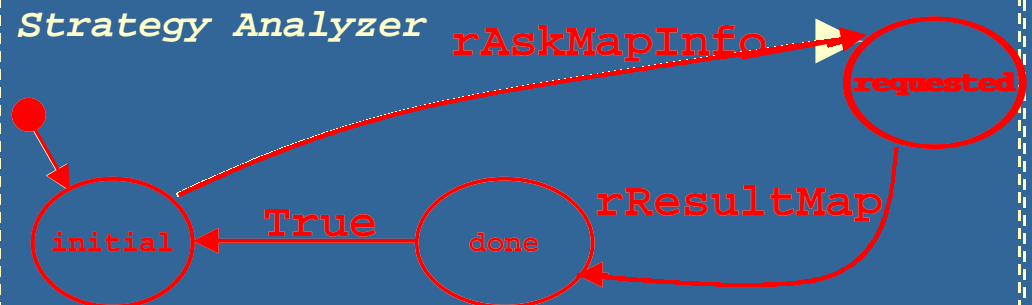
### connector1

rAskMapInfo/ pAskMapInfo



### Strategy Analyzer

rAskMapInfo



# Tool Support

- Integration of Argus-I (UC Irvine) and DRADEL (USC)
- Combine structural and static analysis with dynamic and run-time behavior modeling and analysis

**DRADEL**

Parse File    Check Constr    Type Check    Compliance    Gen Code    **Statecharts**    EXIT

File: c:\troshanak\spec\pc\_arch.c2

**STATUS**

Parsing ...  
 Completed  
 Generating Statechart info ...  
 Completed

**ARCHITECTURAL TYPE MISMATCH**

TypeMismatch:  
 cannot match component PC\_Repository's operation opUpdateData  
 under typing relationship: beh0

```

Component StrategyAnalyzer is {
state {
  status:Integer;
  theRow: Integer;
  theCol: Integer;
  theType:Integer;
  valid :Boolean;
}
invariant {
  status > greater 0 \and (status < less
  \and (theRow < less 64) \and (theCol < le
}
interface {
  req irMapInfo : MapInfo(id: Integer);
  req irMapResult : MapInfo();
  prov ipUpdateData:UpdateData(row:Integer
  col:Integer, type:Integer);
}
operations {
  op irMapInfo: {
    ...
  }
  op irMapResult: {
    ...
  }
  op ipUpdateData: {
    let r: Integer;
    c: Integer;
    t: Integer;
    post (theRow = r)\and (theCol = c)
    \and (theType = t);
  }
}
map {
  irGetMapInfo -> orGetMapInfo();
  irGetMapResult -> orGetMapResult(id -> i);
  ipUpdateData -> opUpdateData(row -> r,
  col -> c, type -> t);
}
}
  
```

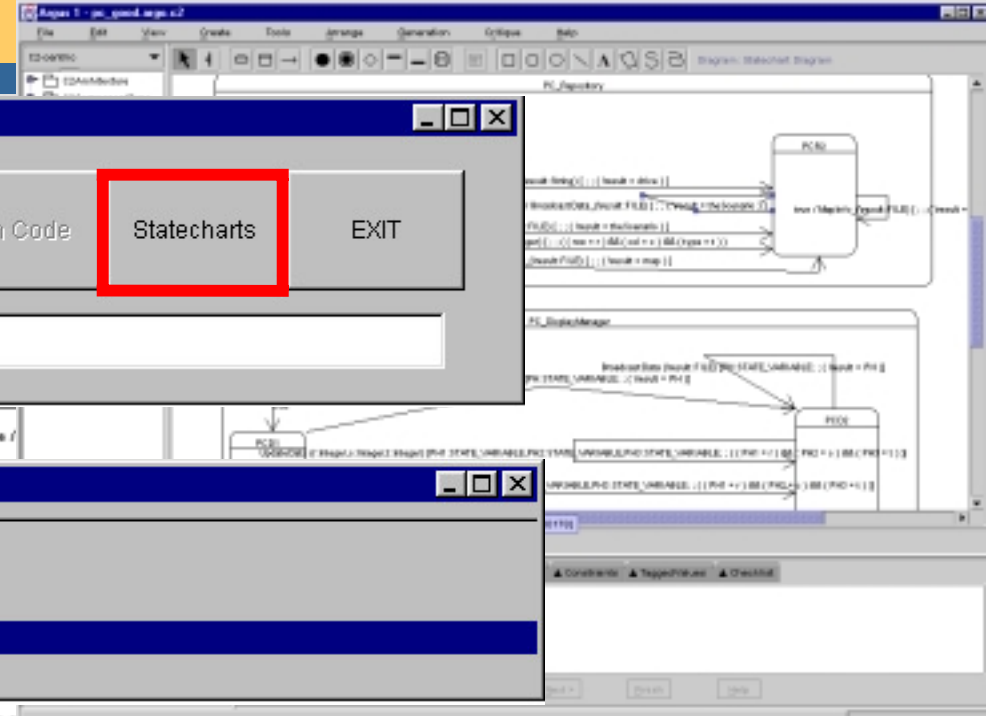
To Do Item

No To Do Item selected

```

  prov opMapInfo: {
    let i: Integer;
    post (i <> null);
  }
  prov opMapResult: {
    post (\result = theMap);
  }
  ...
  map {
    irMapInfo -> opMapInfo(id -> i);
    irMapResult -> opMapResult();
  }
  ...
}
  
```

< Back    Next >    Finish    Help



# Summary

- Ability to analyze the behavior both statically and dynamically
- Consistency of architectural models and implementation level artifacts
- *Extensions*: Subtyping, Interaction protocols, Code generation
- The framework is being applied to JPL's Mission Data System