Toward Architecture-based Reliability Estimation

Roshanak Roshandel, Nenad Medvidovic

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

ICSE Workshop on Architecting Dependable System (WADS'04),
May 25, 2004

Motivation

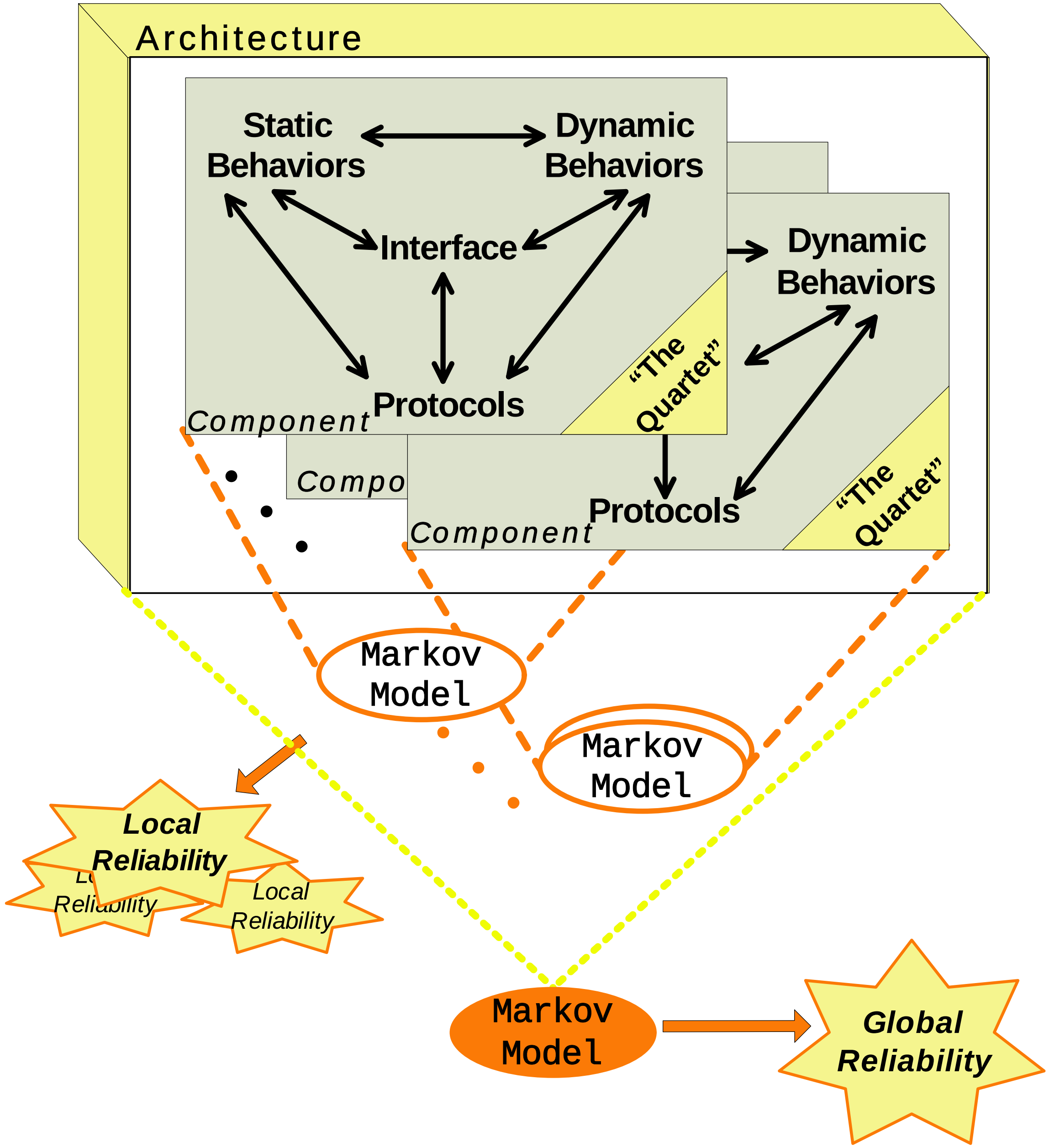
- Software reliability: probability that the system performs its intended functionality without failure
- Software reliability techniques aim at reducing or eliminating failure of software systems
- Complimentary to *testing*, rely on implementation
- **How one goes about building reliable systems? And how to measure early reliability?**

Software Architecture

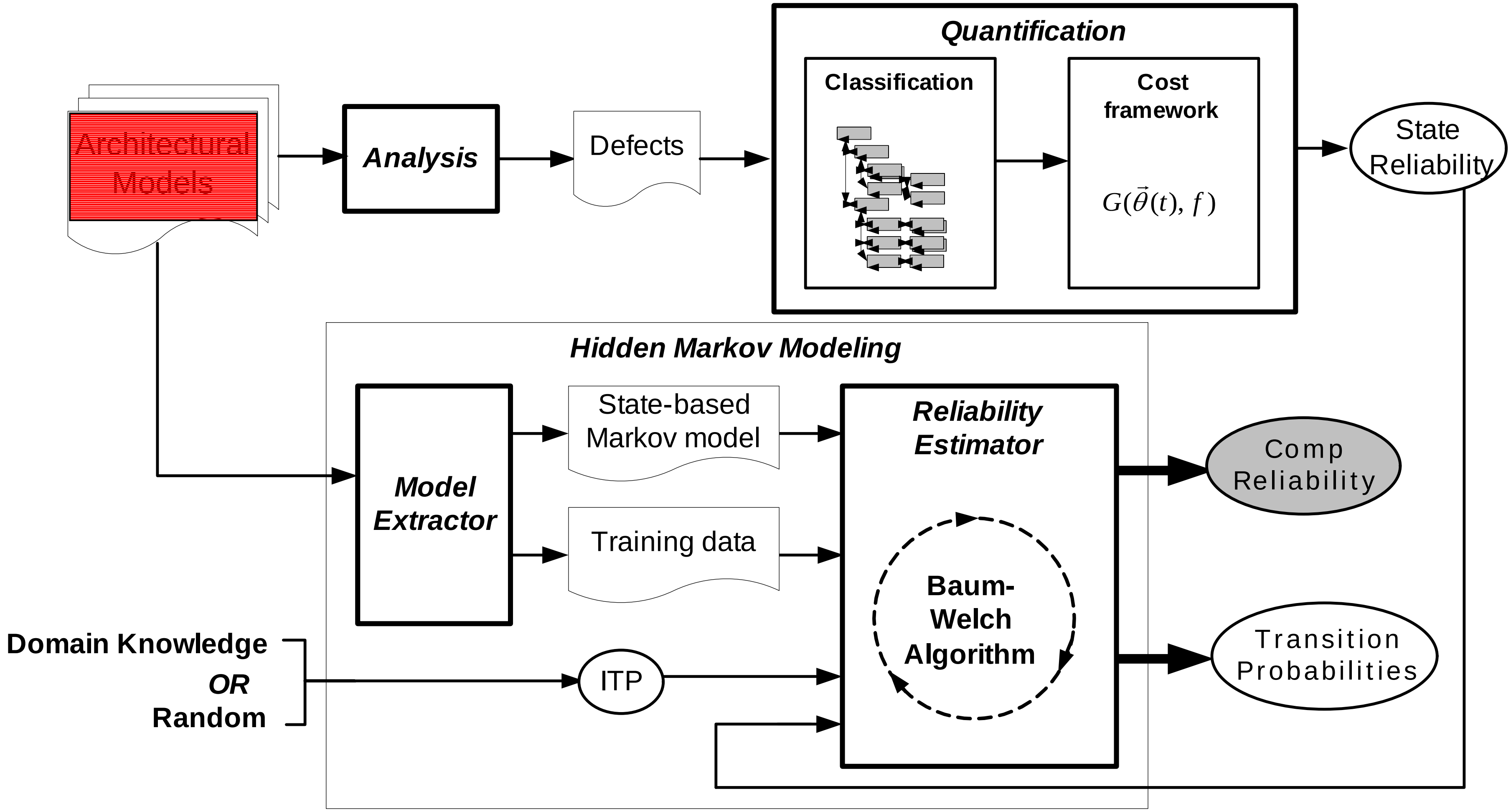
- High-level abstractions describing
  - Structure, Behavior, Constraints
- Coarse-grain building blocks, promote separation of concerns, reuse
  - Components, Connectors, Interfaces, Configurations
- Architectural decisions directly affect aspects of software dependability
  - Reliability
- ADLs, Formal modeling notations, related analysis
  - Often lack *quantification* and *measurement*

Architectural Reliability

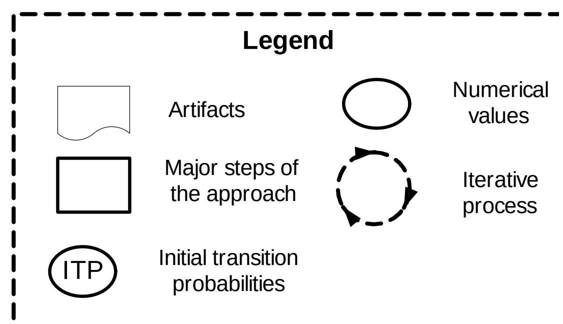
- Lightly explored
- Require availability of implementation to:
  - Build behavioral model of the software system
  - Obtain individual component's reliability
- Software architecture offers compositional approaches to modeling, and analysis
- The challenge is *quantifying* these results
  - Presence of uncertainty
    - Unknown operational profile
    - Improper behavior



Architecture
Static Behaviors
Dynamic Behaviors
Interface
Protocols
Component
"The Quartet"
Dynamic Behaviors
Protocols
Compo
Component
"The Quartet"
Markov Model
Markov Model
Local Reliability
Local Reliability
Markov Model
Global Reliability

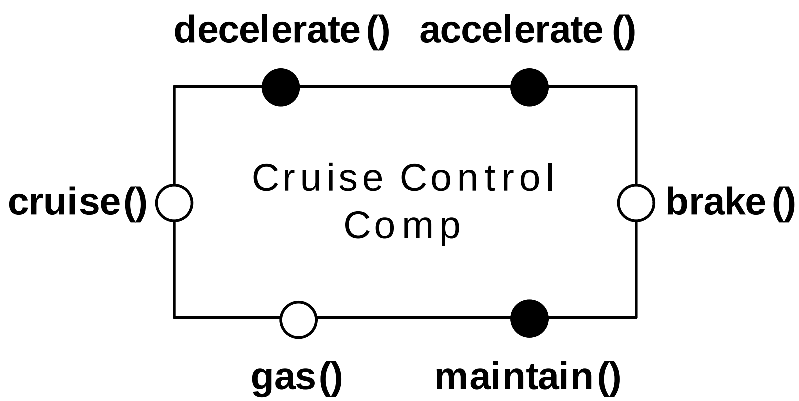


Component Reliability

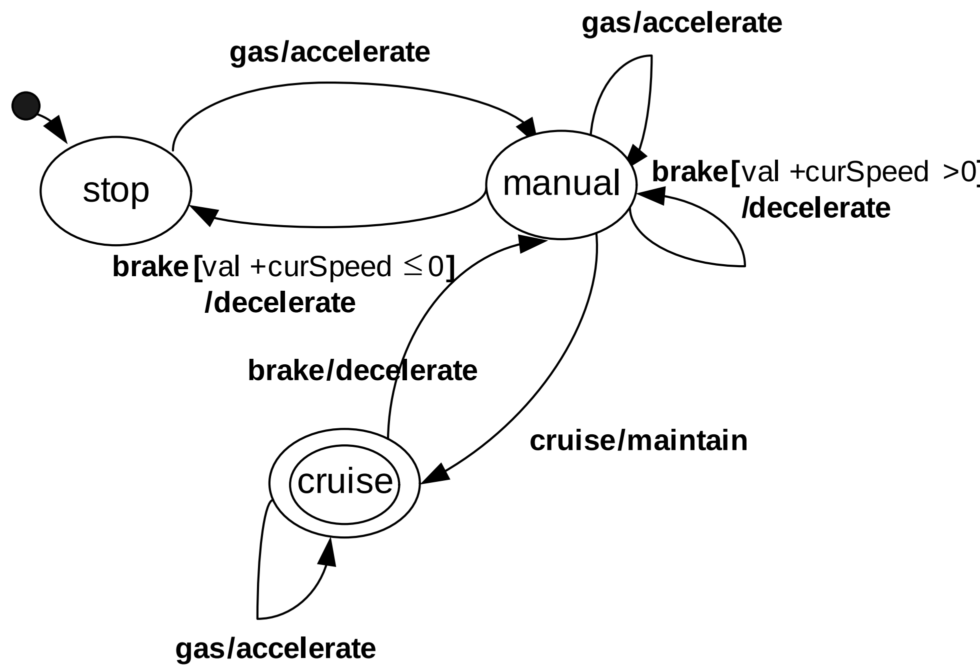


The *Quartet*

1. *Interface* models specify the points by which a component interacts with other components in a system
2. *Static behavior* models describe the functionality of a component discretely, i.e., at particular "snapshots" during the system's execution
3. *Dynamic behavior* models provide a continuous view of *how* a component arrives at different states throughout its execution
4. *Interaction protocol* models provide an *external* view of the component and how it may legally interact with other components in the system



DYNAMIC BEHAVIOR



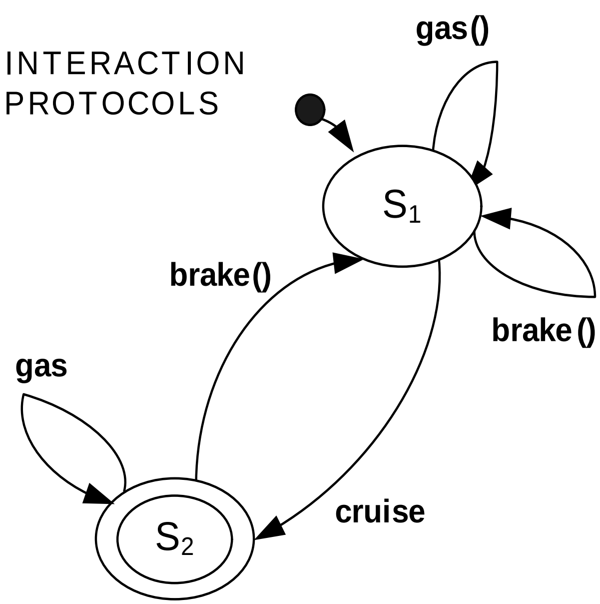
INTERFACES

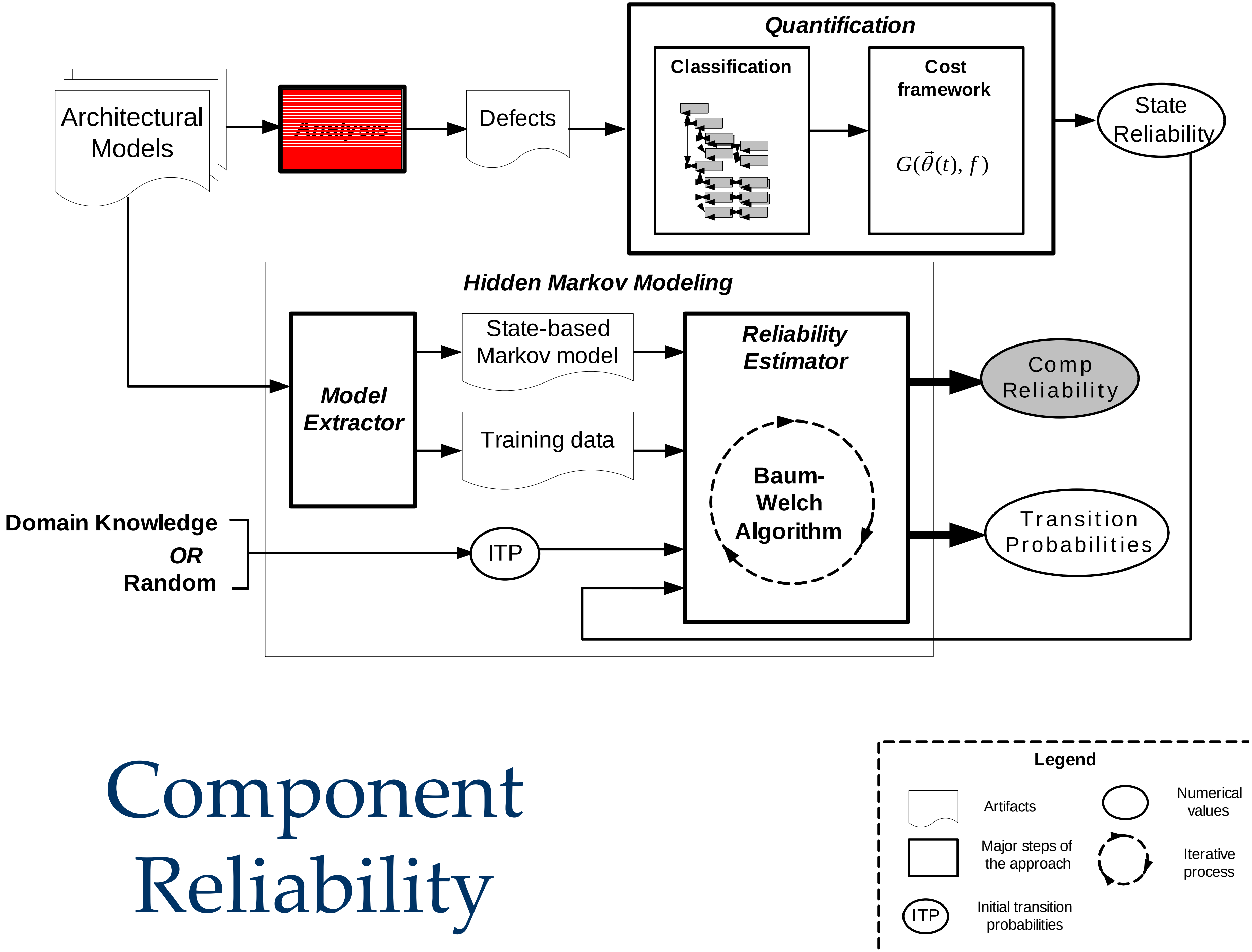
```
PROV gas(val:SpeedType):SpeedType;
PROV brake(val:SpeedType):SpeedType;
PROV cruise(speed:SpeedType);Boolean;
```

STATIC BEHAVIOR

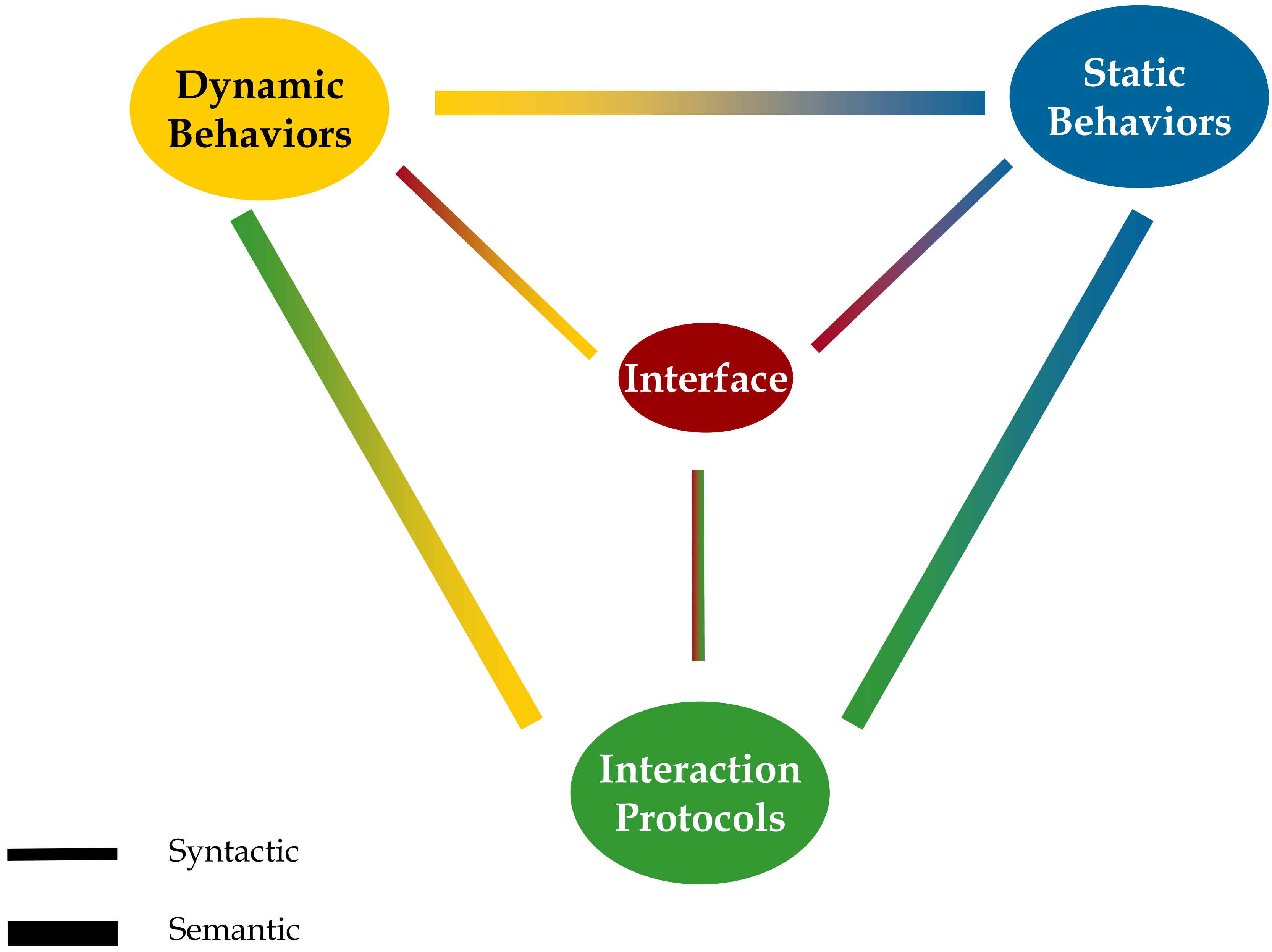
```
STATE-VAR:
   curSpeed:SpeedType;
   isCruising:Boolean;
INVARIANT:
   0 ≤ curSpeed ≤ MAX;
OPERATIONS:
   gas.preCond (val > 0);
   gas.postCond (~curSpeed = curSpeed + val);
   brake.preCond (val < 0);
   brake.postCond (~curSpeed = curSpeed + val
                   AND isCruising = false);
   cruise.preCond (speed > 0);
   cruise.postCond (~curSpeed = speed
                    AND isCruising = true);
```

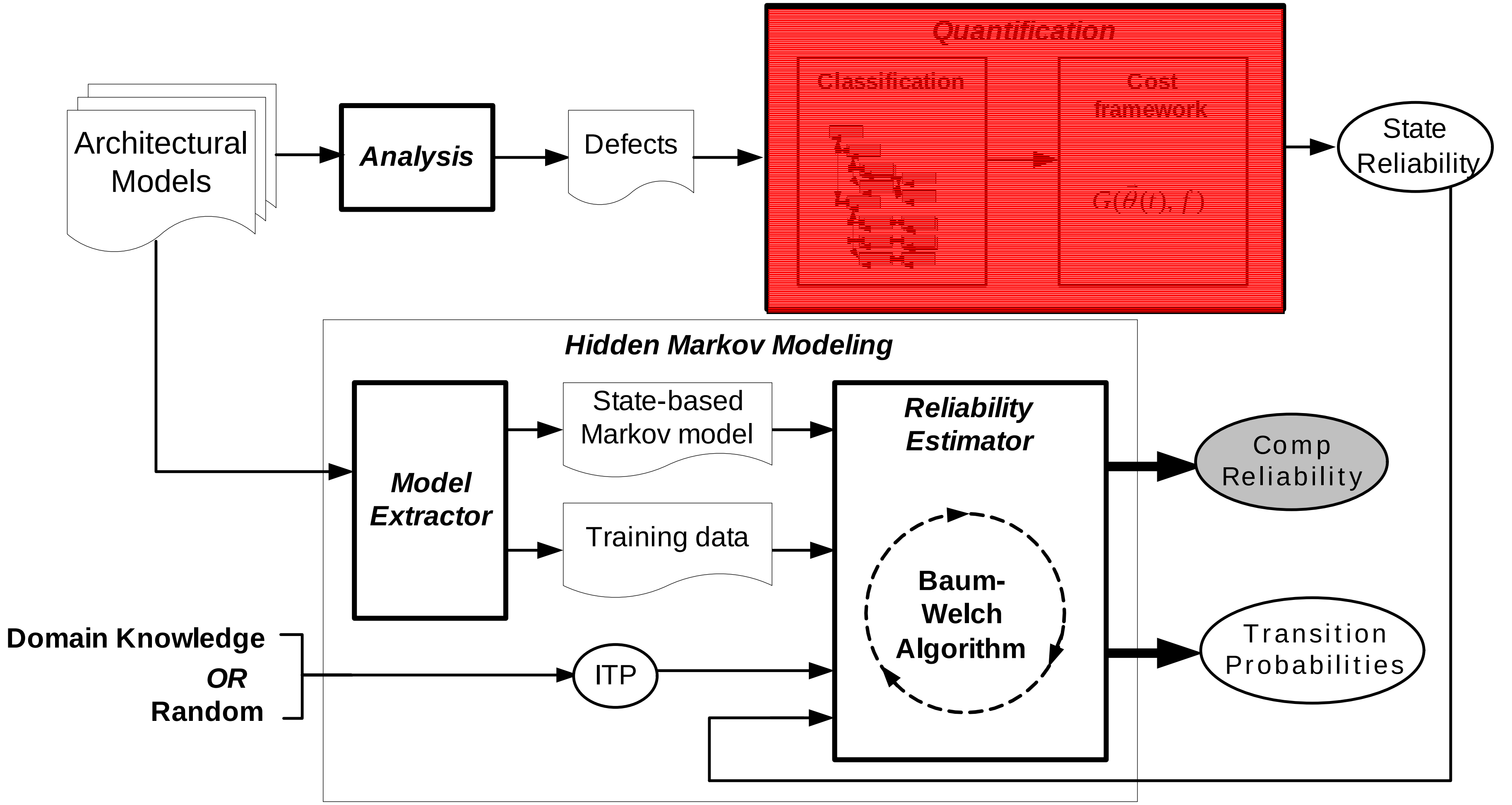
INTERACTION
PROTOCOLS



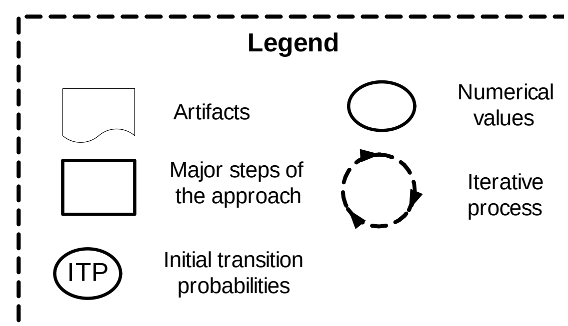


Component Reliability


Dynamic Behaviors
Static Behaviors
Interface
Interaction Protocols
Syntactic
Semantic



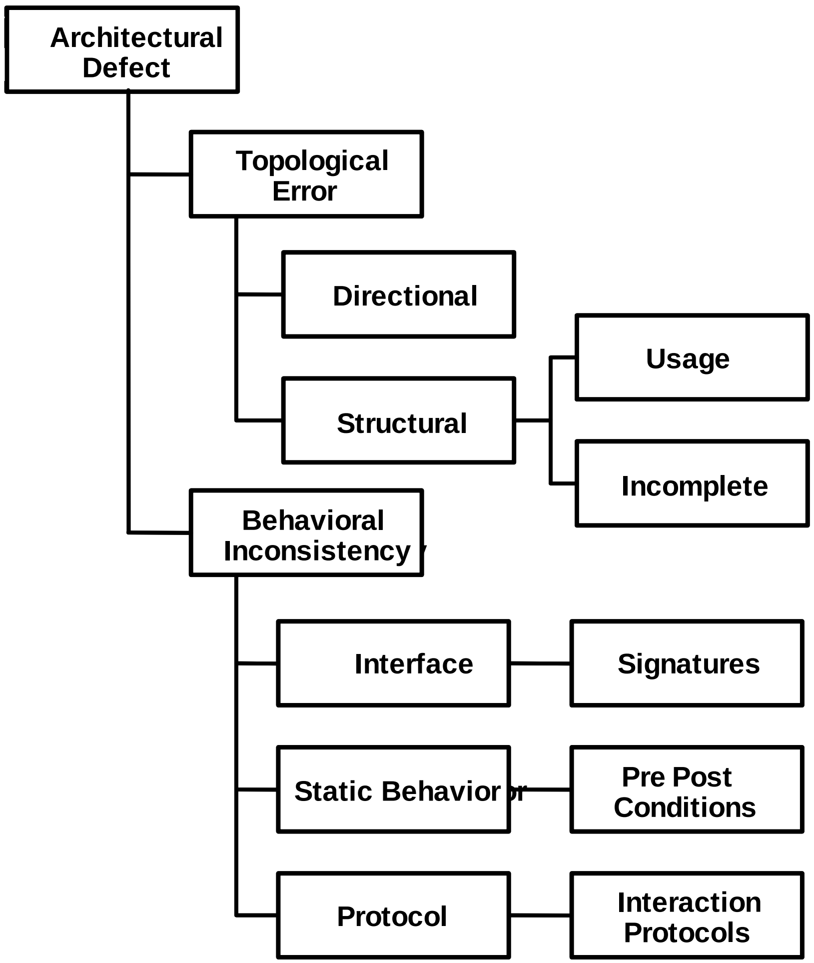
Component Reliability



Defect Quantification

- Architectural defects could affect system Reliability
- Different defects affect the Reliability differently
  - e.g., interface mismatch vs. protocol mismatch
- The cost of mitigation of defects varies based on the defect type
- Other (domain specific) factors may affect the quantification
- **Classification + Cost framework**

Classification + Cost Framework

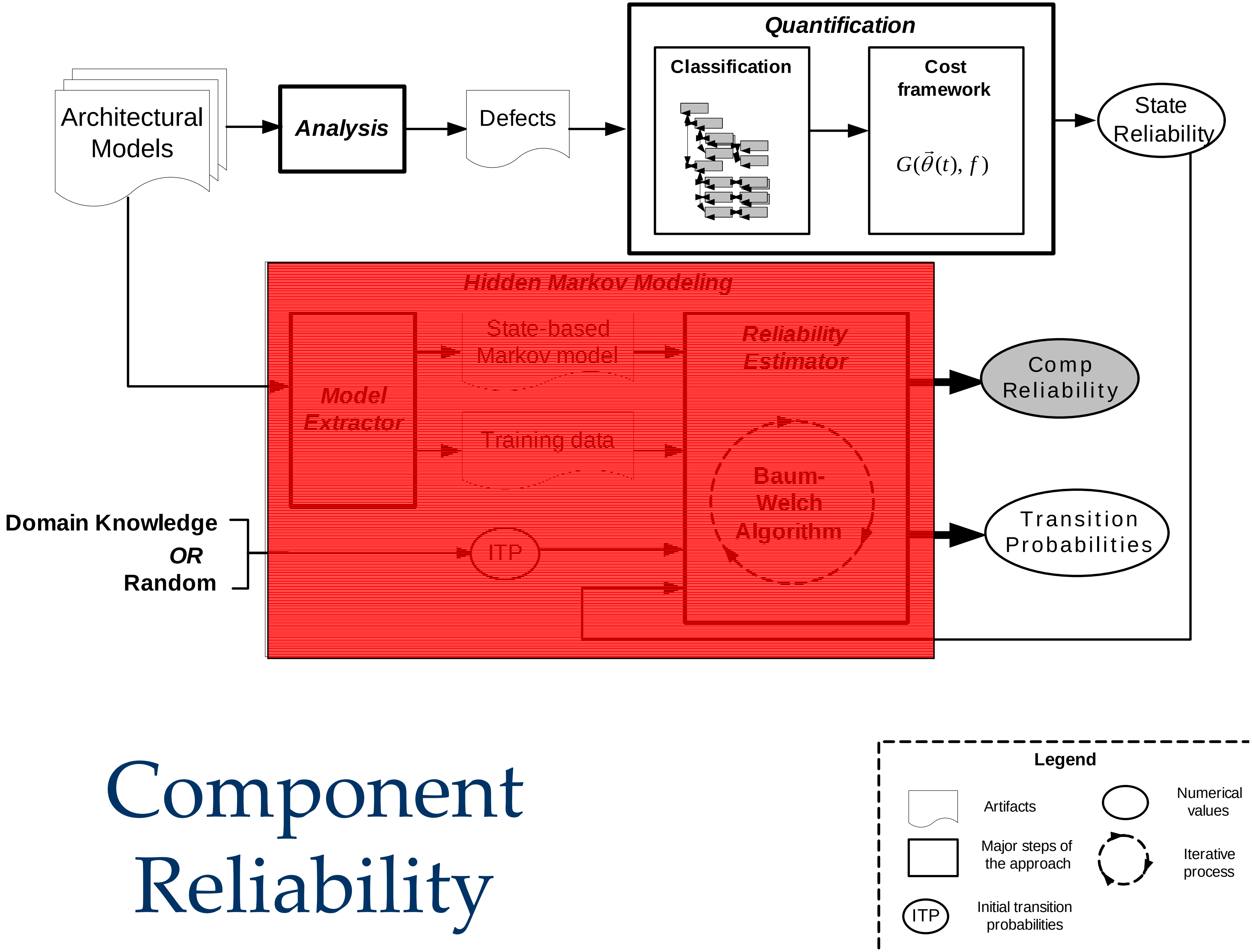


- Pluggable/Adaptable
- Identify the important factors within a domain
- For a defect class *t*

$$c_t = G(\vec{\theta}(t), f),\ where$$

$$\vec{\theta}(t) = [\theta_1(t), \theta_2(t), \ldots, \theta_n(t)]$$

- *f*: Frequency of occurrence
- And $\vec{\theta}(t)$ vector of all relevant factors
- Result will be used in reliability estimation



Component Reliability

Reliability Techniques

- Non-Homogenous Poisson Processes, Binomial Models, Software Reliability Growth Models, …
- Markovian Models
  - Suited to architectural approaches
  - Considers system's structure, compositional
  - Stochastic processes
  - Informally, a finite state machine extended with transition probabilities

Our Reliability Model

- Built based on the *dynamic behavioral model*
- Assume Markov property (Discrete Time Markov Chains)
- Transition probabilities maybe unknown
- Complex behavior results in lack of a correspondence between events and states
- Event/action pairs to describe components' interaction

➔**Augmented Hidden Markov Models (AHMM)**

Evaluation

- Uncertainty analysis
  - Operational profile
  - Incorrect behavior
- Sensitivity analysis
  - Traditional Markov-based sensitivity analysis combined with the defect quantification
- Complexity
- Scalability

Conclusion and Future Work

- Step toward closing the gap between architectural specification and its effect on system's reliability
- Handles two types of uncertainties associated with early reliability estimation
- Preliminary results are promising
- Need further evaluation
- Build compositional models to estimate system reliability based on estimated component reliabilities

Questions?