

# Simulation Verification and Validation by Dynamic Policy Enforcement

W.T. Tsai, X. Liu, Y. Chen, R. Paul

Software Research Laboratory

Computer Science & Engineering Department

Arizona State University

# Introduction

Develop a general purpose simulation framework

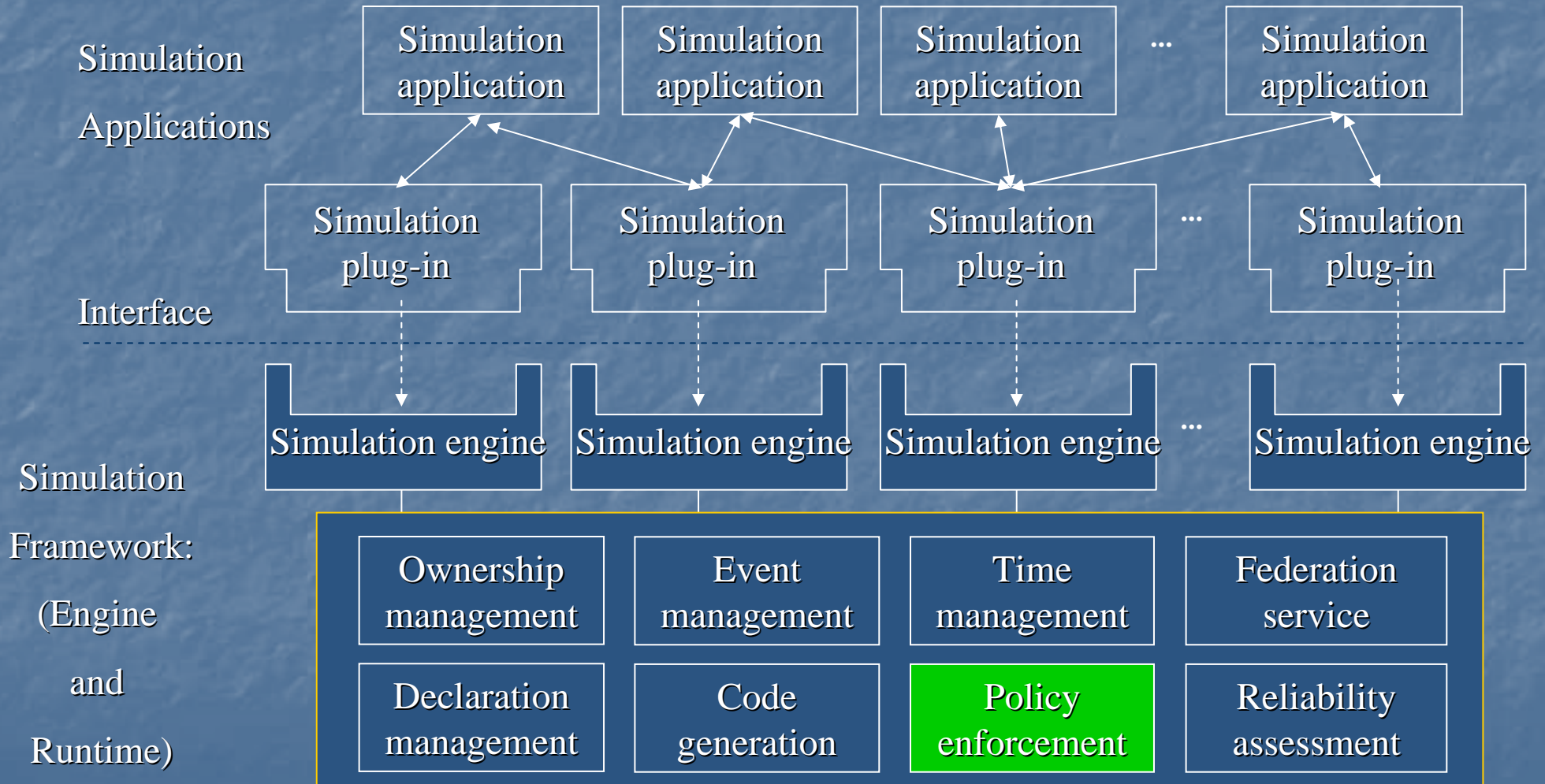
## ➤ Features

- Web-based in Service-Oriented Architecture
- Support distributed simulation with multi-agents
- Dynamic verification and validation
- Dynamically extendable

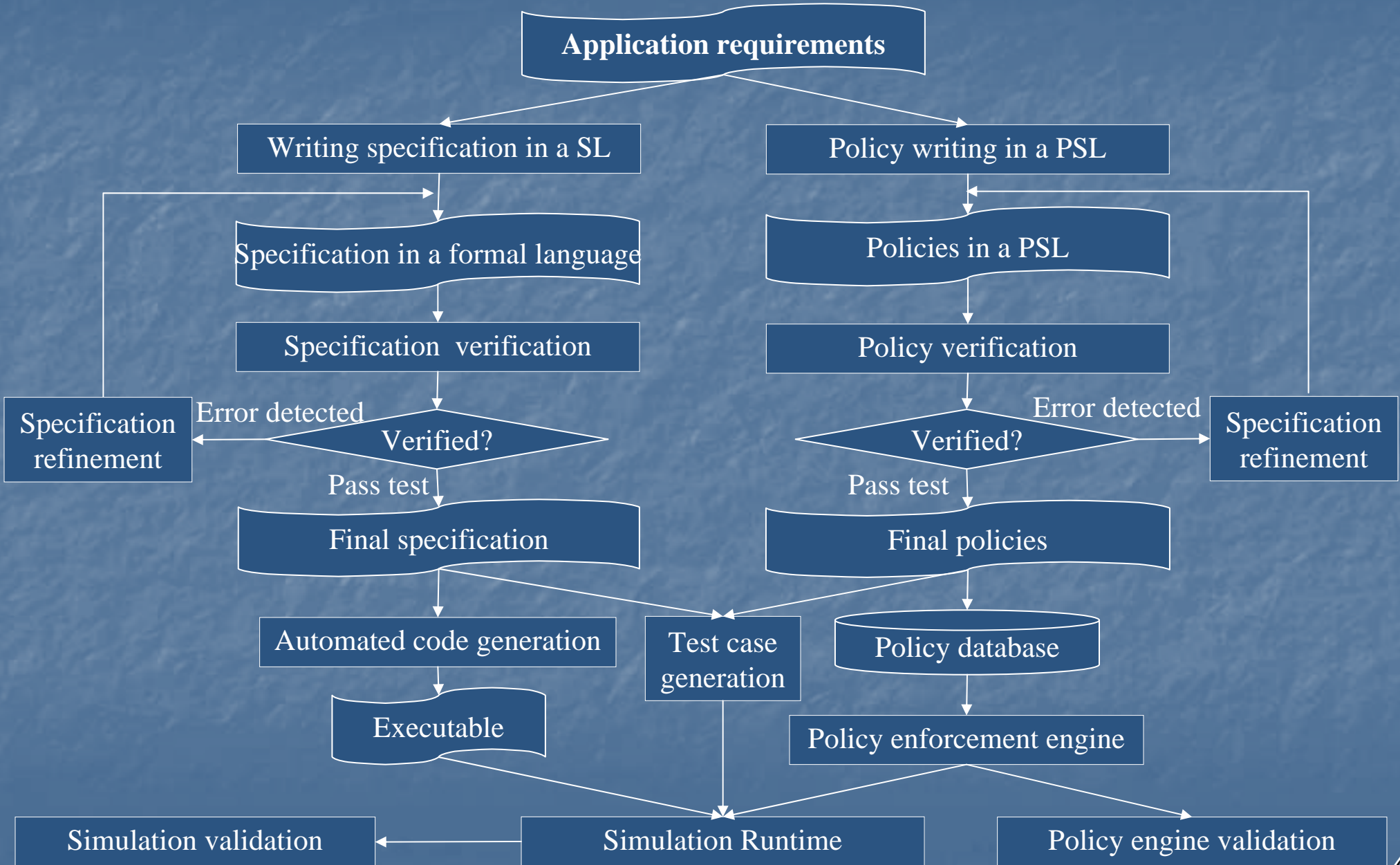
## ➤ Three application areas

- Web services simulation environment
- Embedded system simulation environment
- Command and Control system simulation environment

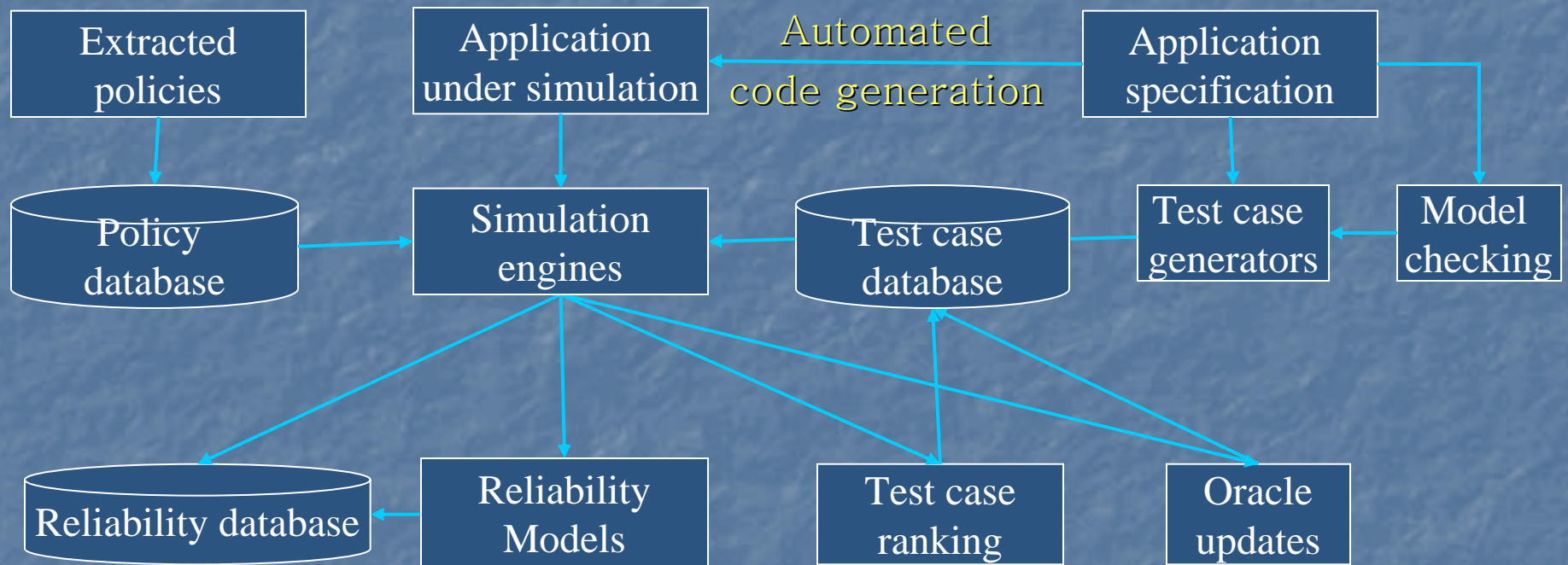
# Service-Oriented Simulation Framework



# System Development for Policy-Based Computing

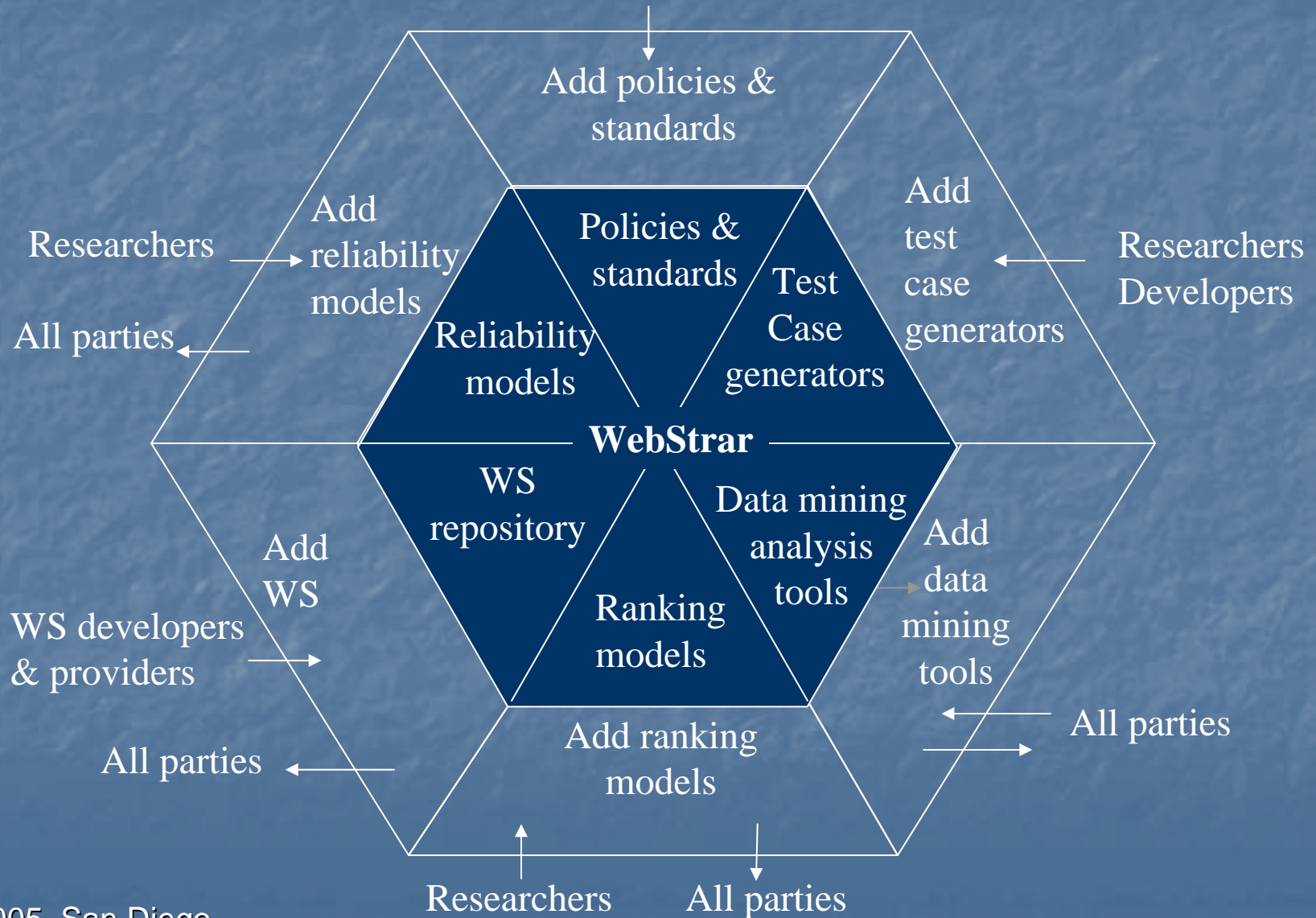


# The Framework with Multiple Functions



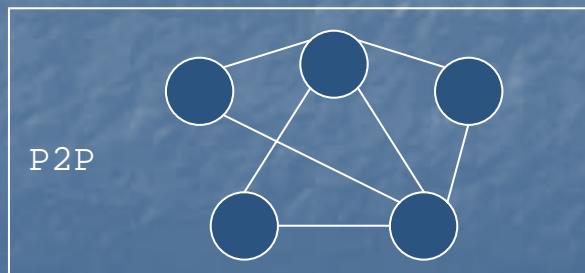
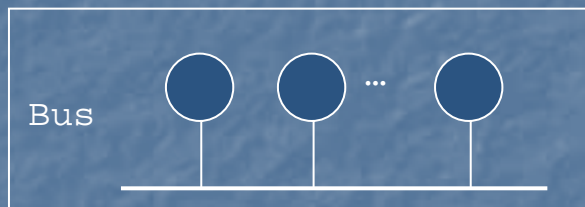
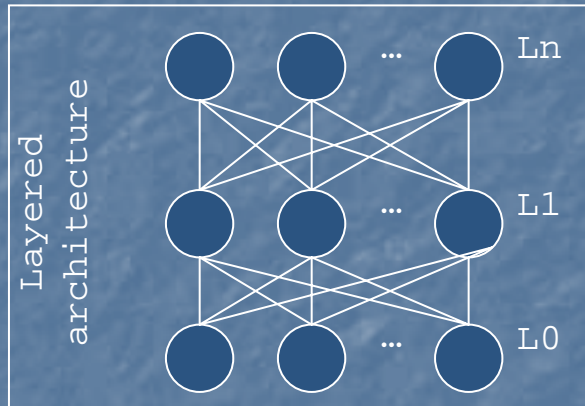
# Extendable Simulation

Regulators and standard organizations, e.g., FAA, FDA, DoD, ISO, IEEE, NASA, NIH, OMG



# Using Templates to Build Different Application Platforms

## Architecture selection



## Connection selection

Specify layer Ln nodes



Specify layer L1 nodes



Specify layer L0 nodes

Specify bus type



Specify all nodes

Specify all links



Specify all nodes

## Feature selection

- Distributed scheduling
- Logging
- Visualization
- Policy enforcement**
- Redundant execution
- Reconfigurable
- Auto code generation

- Distributed scheduling
- Logging
- Visualization
- Policy enforcement
- Arbitration
  - CSMA/CD
  - token
- Data rate
  - 10 Mb
  - 100 Mb
  - 1 Gb

- Logging
- Visualization
- Policy enforcement

# Policy Specification (Positive Obligation Policies)

Template

```
Tanks.e2e Policy01
1 MUSTDO
2 {
3   definedOn ROLE
4   triggeredBy EVENT
5   do ACTION
6   on CONDITION
7 }
```

Example

```
Items List Policy01
1 MUSTDO
2 {
3   definedOn ROLE:SupportingArmsCoordinator
4   triggeredBy EVENT:E07_ReceiveCFF
5   do ACTION:A_IssueFireOrder
6   on CONDITION:TRUE
7 }
```

# Policy Specification (Negative Obligation Policies)

Template

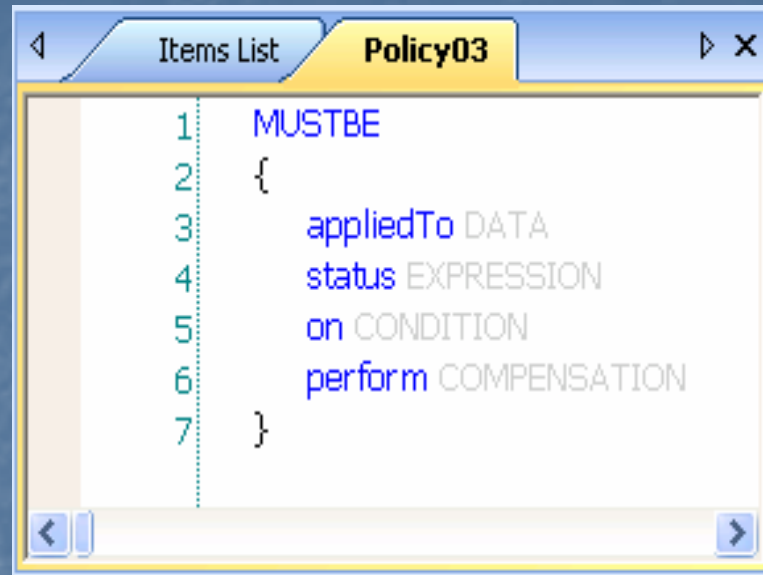
```
Tanks.e2e Policy02
1 MUSTNOTDO
2 {
3   definedOn ROLE
4   do ACTION
5   on CONDITION
6   perform COMPENSATION
7 }
```

Example

```
Tanks.e2e Policy02
1 MUSTNOTDO
2 {
3   definedOn ROLE:MainBattleTank
4   do ACTION:A_MBTRectMission
5   on CONDITION:MainBattleTankACanShoot
6   perform COMPENSATION:Warning
7 }
```

# Policy Specification (System Constraints)

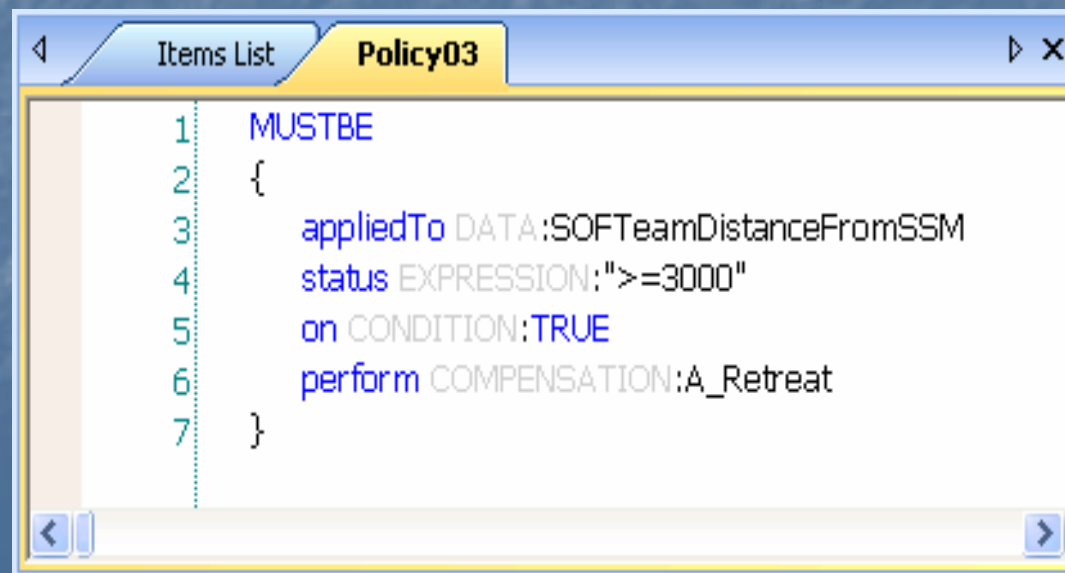
Template



The screenshot shows a window titled 'Policy03' with a tab labeled 'Items List'. The main content area contains a policy template with the following text:

```
1  MUSTBE
2  {
3    appliedTo DATA
4    status EXPRESSION
5    on CONDITION
6    perform COMPENSATION
7  }
```

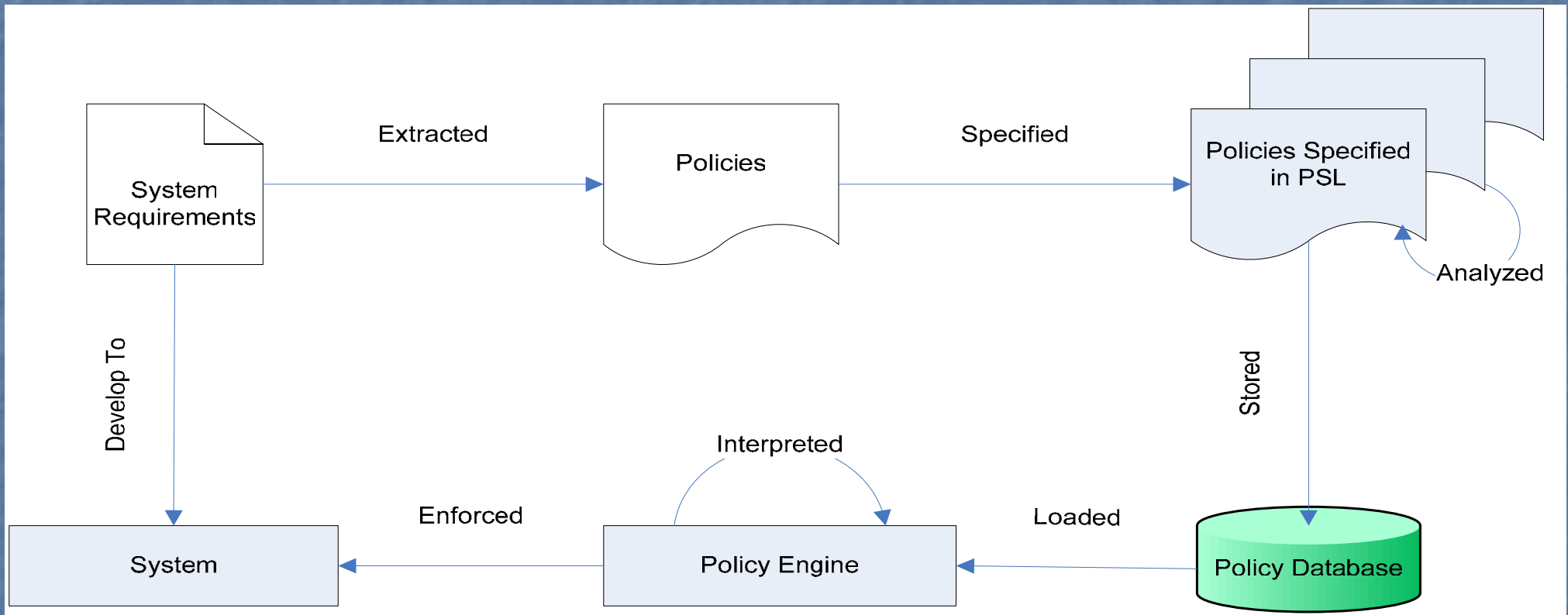
Example



The screenshot shows a window titled 'Policy03' with a tab labeled 'Items List'. The main content area contains an example policy with the following text:

```
1  MUSTBE
2  {
3    appliedTo DATA:SOFTeamDistanceFromSSM
4    status EXPRESSION:">=3000"
5    on CONDITION:TRUE
6    perform COMPENSATION:A_Retreat
7  }
```

# Policy Enforcement Framework



# Overhead of Policy Enforcement

Sample Application	Criteria	Policy disabled	Policy enabled	Hard-coded Policies
Space Complexity	Memory Size (KB)	5323	5918	5599
	Compared to No Policies	100%	111%	105%
	Memory Composition	SIM + SC	SIM + SC + PE	SIM + SC + HCP
Time Complexity	Simulation Time (ms)	2722	5002	2962
	Compared to No Policies	100%	184%	109%
	Time Composition	SC	SC + PE	SC + HCP
Comments	SIM = Simulator    SC = System Scenarios    PE = Policy Engine HCP = Hard-Coded Policies			

# Conclusions

We developed a simulation framework

- Service-oriented architecture
- Distributed simulation of different applications
- Templates for platform building
- System specification and verification tools
- Policy specification and verification tools
- Dynamic policy enforcement and verification
- Policy experiments