Towards Architecture-based Self-Healing Systems

Eric M. Dashofy, André van der Hoek, and Richard N. Taylor

WOSS'02

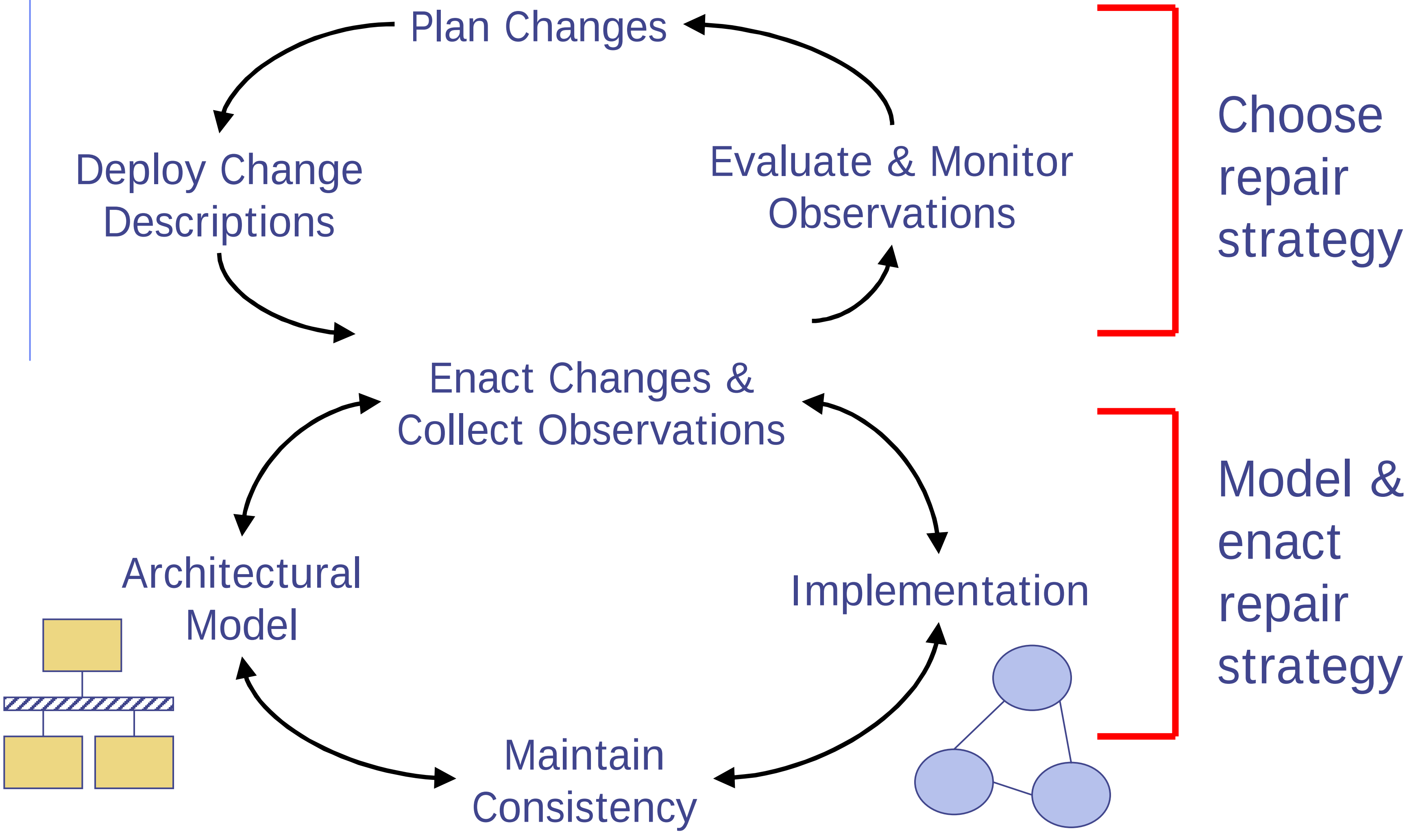
November 18, 2002

What is “self-healing?”

Key Question: What is the difference between a fault-tolerant and a self-healing system?

- Fault-Tolerant
  - Connotes fault-based repair and understanding
  - Faults are likely pre-specified
  - Repair strategies are also pre-specified
- Self-Healing
  - Connotes goal-based repair and understanding
  - Unexpected faults are expected
  - Arbitrary repair strategies constructed at runtime

Overall Vision



Plan Changes

Deploy Change Descriptions

Evaluate & Monitor Observations

Enact Changes & Collect Observations

Architectural Model

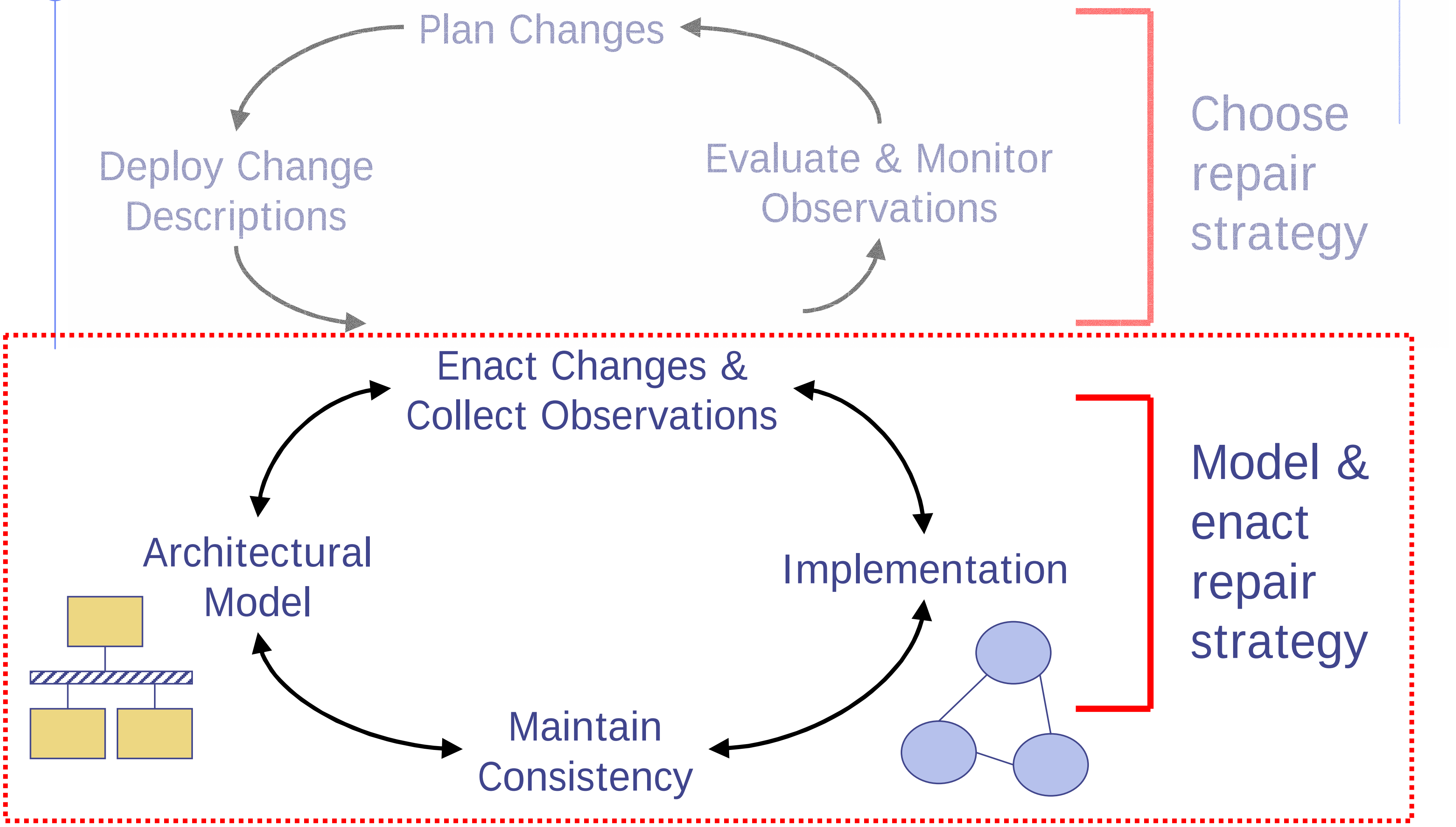
Implementation

Maintain Consistency

Choose repair strategy

Model & enact repair strategy

Our Focus



Plan Changes

Deploy Change Descriptions

Evaluate & Monitor Observations

Choose repair strategy

Enact Changes & Collect Observations

Architectural Model

Implementation

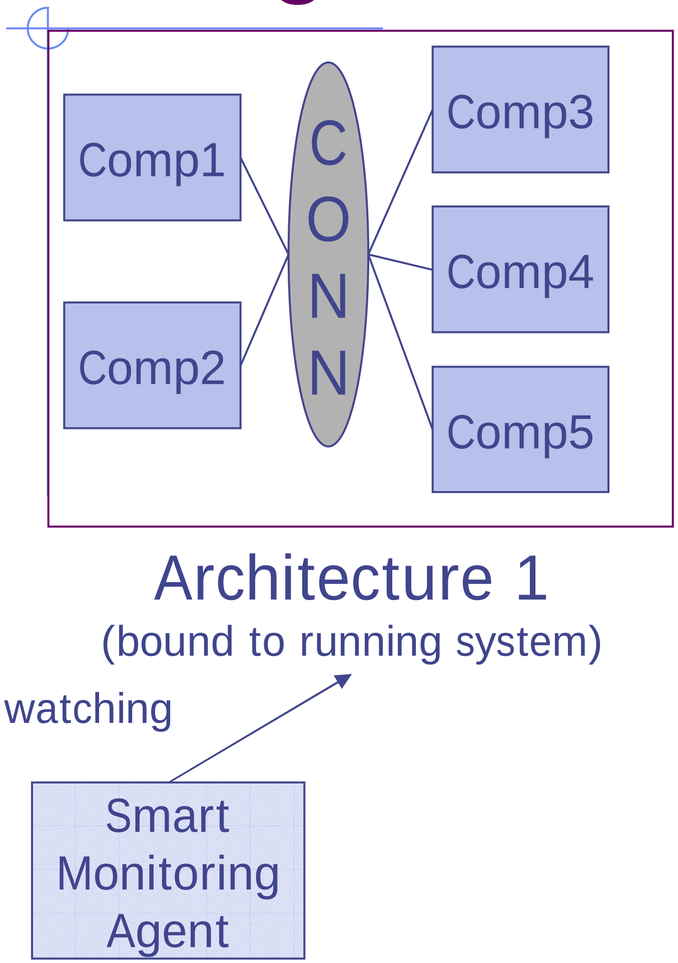
Maintain Consistency

Model & enact repair strategy

Additional Aspects of the Approach

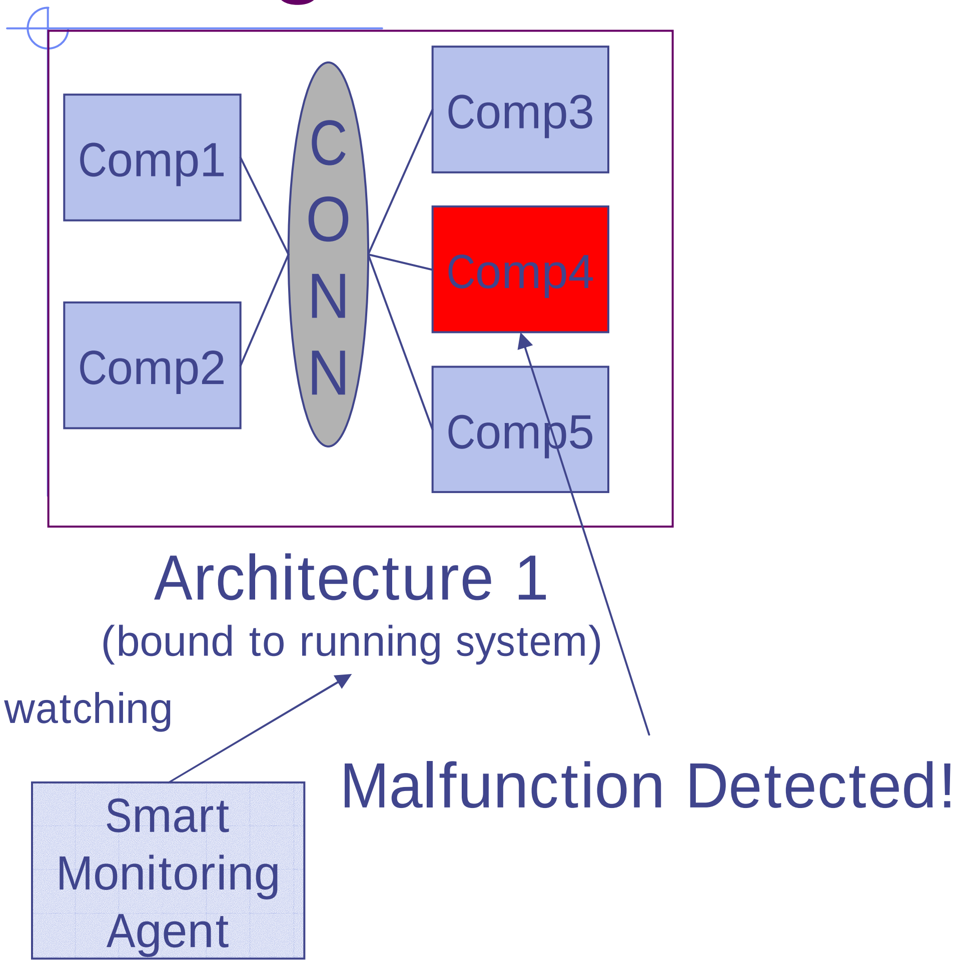
- Architectural Styles
  - Loosely-coupled, event-based
  - Foundation for runtime change
  - Foundation for monitoring
- Systems described in extensible ADL
  - Description accompanies deployed system
  - Repair strategies expressed in terms of architecture description

Expressing Repair Strategies Using Architecture Differencing



Architecture 1
(bound to running system)

Expressing Repair Strategies Using Architecture Differencing



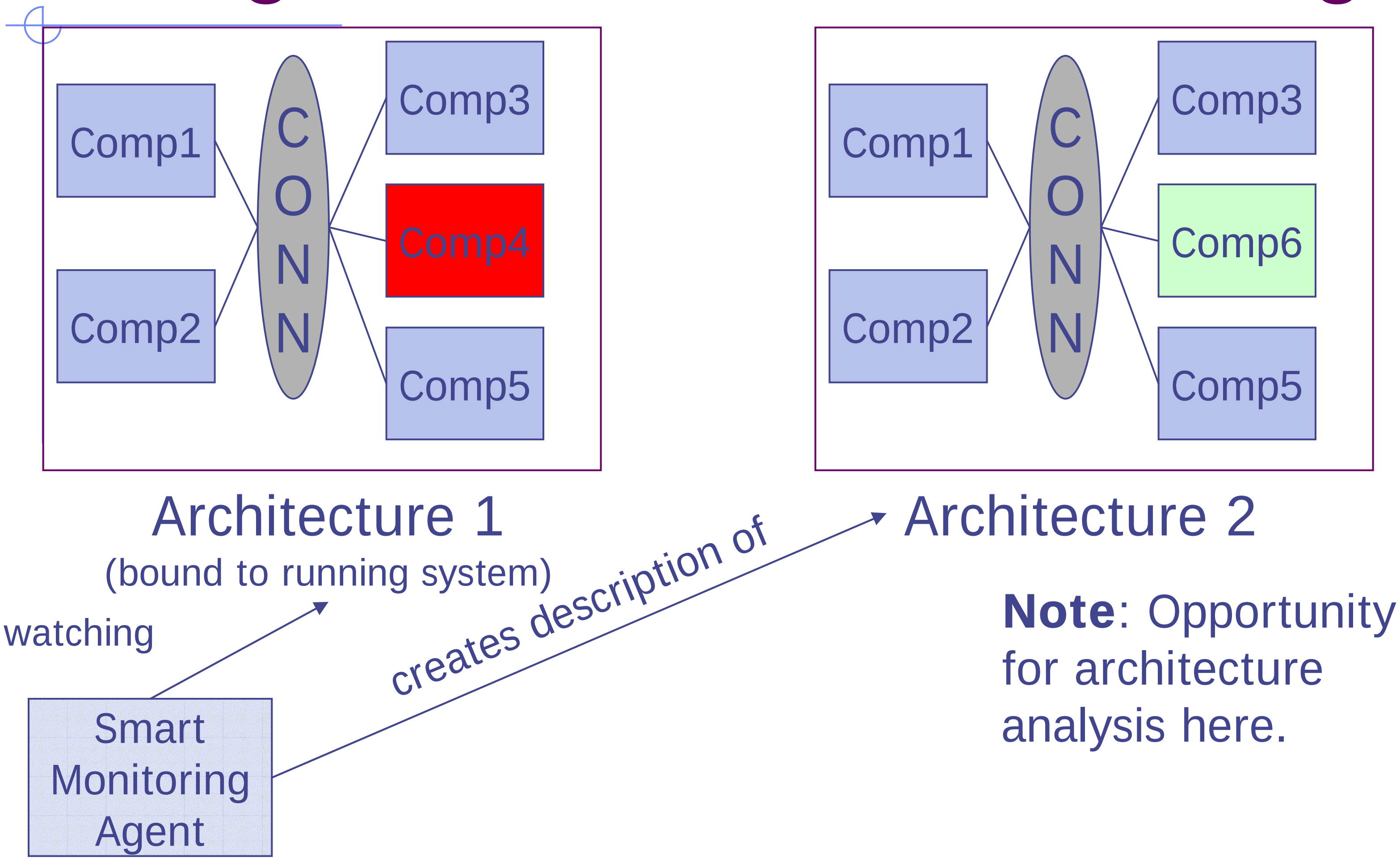
Architecture 1
(bound to running system)

watching

Smart Monitoring Agent

Malfunction Detected!

Expressing Repair Strategies Using Architecture Differencing



Comp1

Comp2

CONN

Comp3

Comp4

Comp5

Architecture 1

(bound to running system)

Comp1

Comp2

CONN

Comp3

Comp6

Comp5

Architecture 2

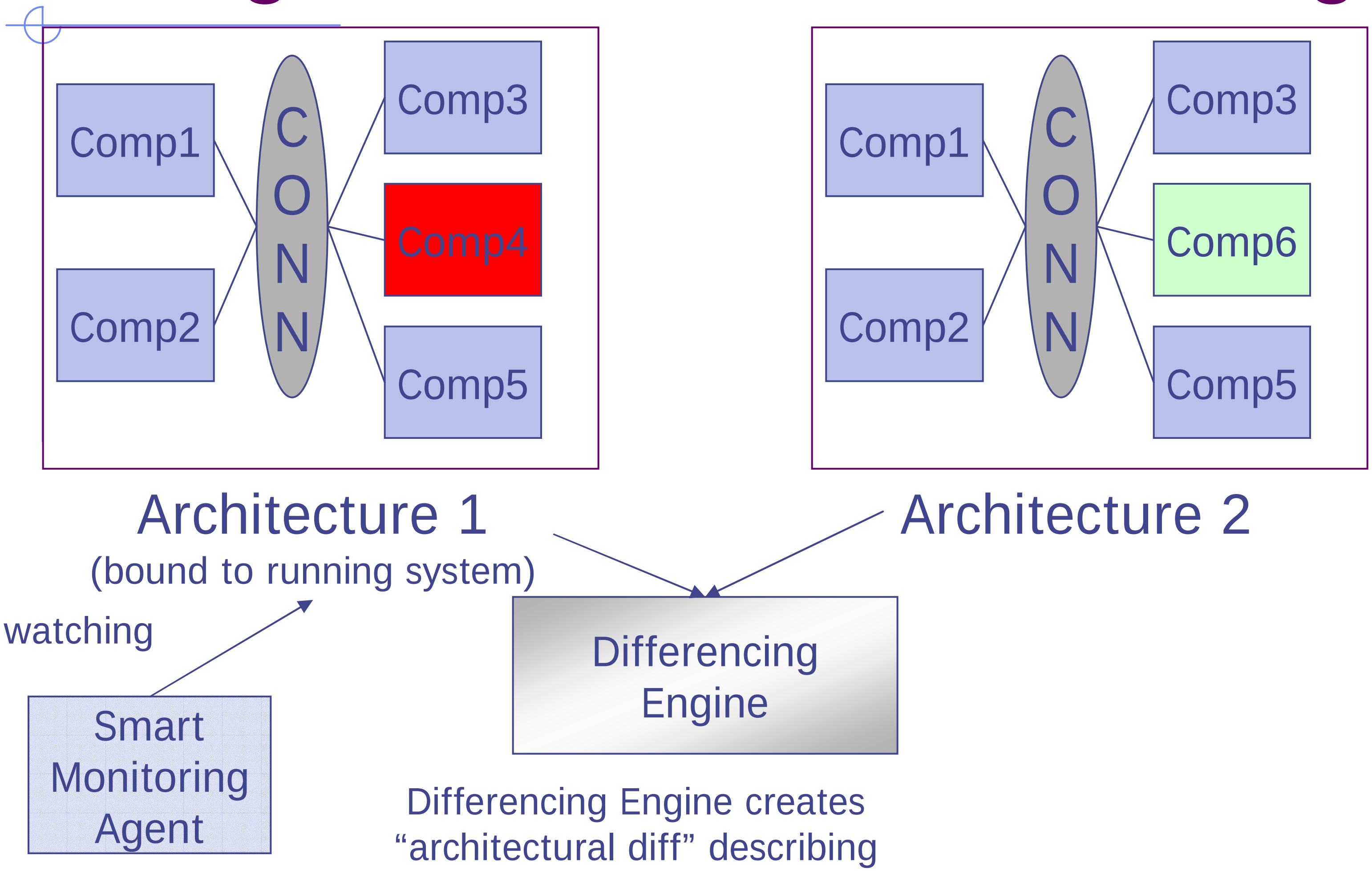
watching

creates description of

Smart Monitoring Agent

Note: Opportunity for architecture analysis here.

Expressing Repair Strategies Using Architecture Differencing



Comp1

Comp2

CONN

Comp3

Comp4

Comp5

Comp1

Comp2

CONN

Comp3

Comp6

Comp5

Architecture 1
(bound to running system)

Architecture 2

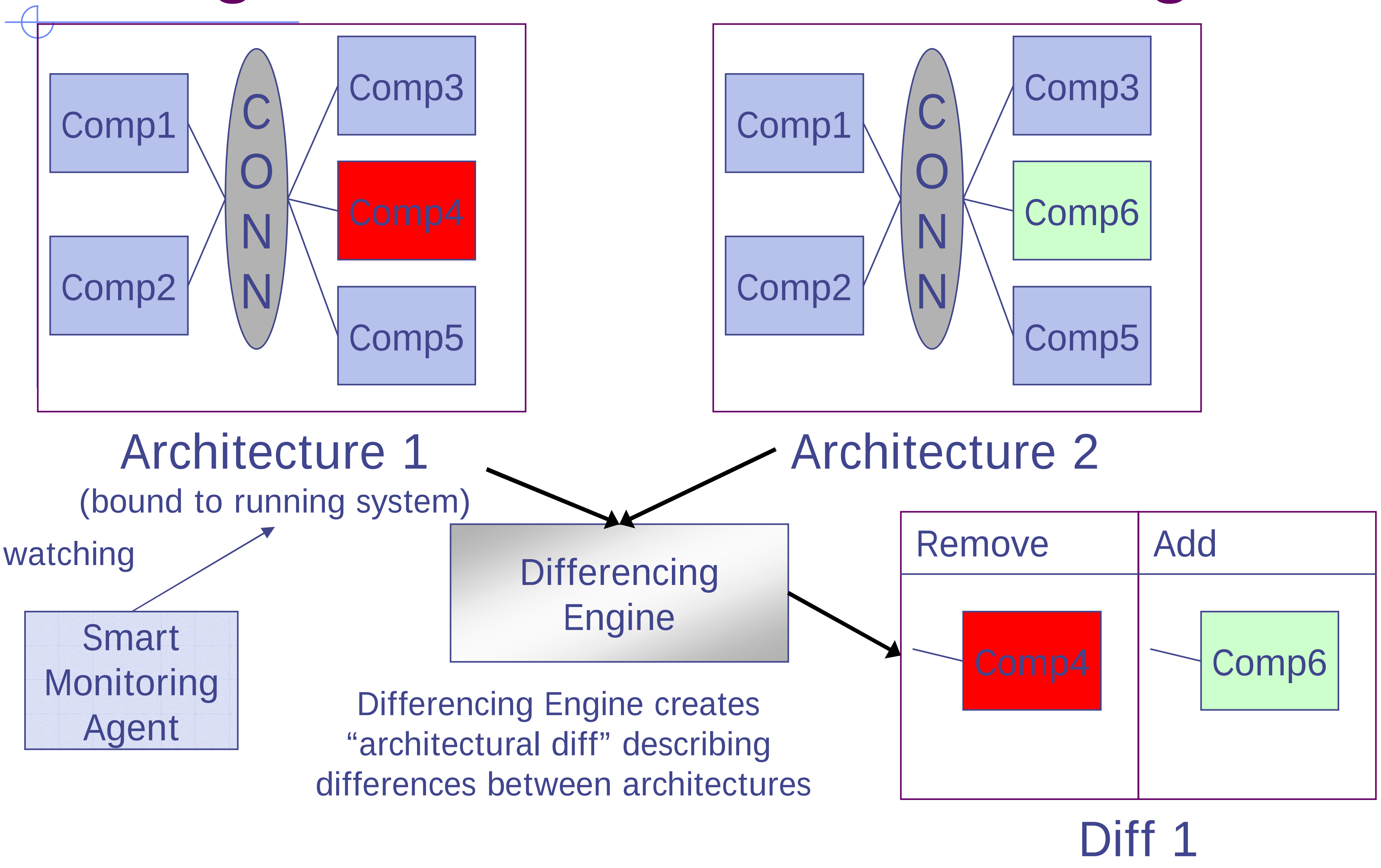
watching

Smart Monitoring Agent

Differencing Engine

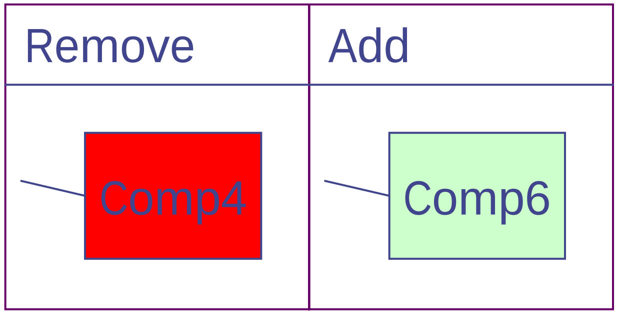
Differencing Engine creates "architectural diff" describing differences between architectures

Expressing Repair Strategies Using Architecture Differencing



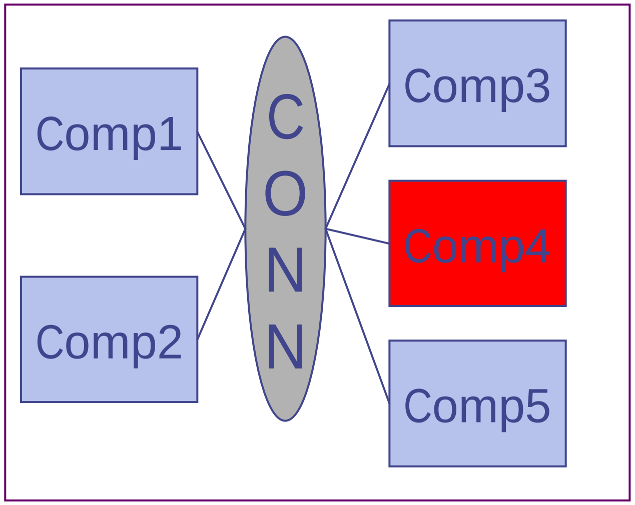
Differencing Engine creates “architectural diff” describing differences between architectures

Effecting Repairs Using Architectural Diffs

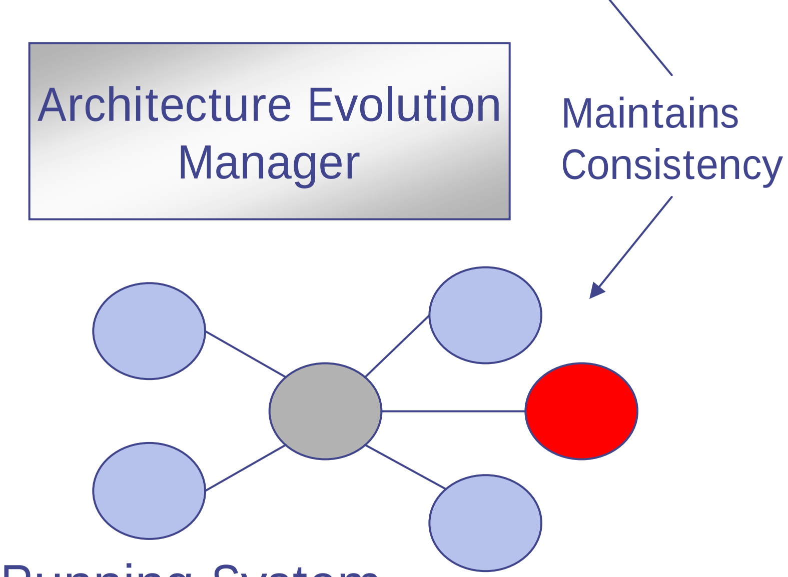


Remove	Add
Comp4	Comp6

Repair Plan 1

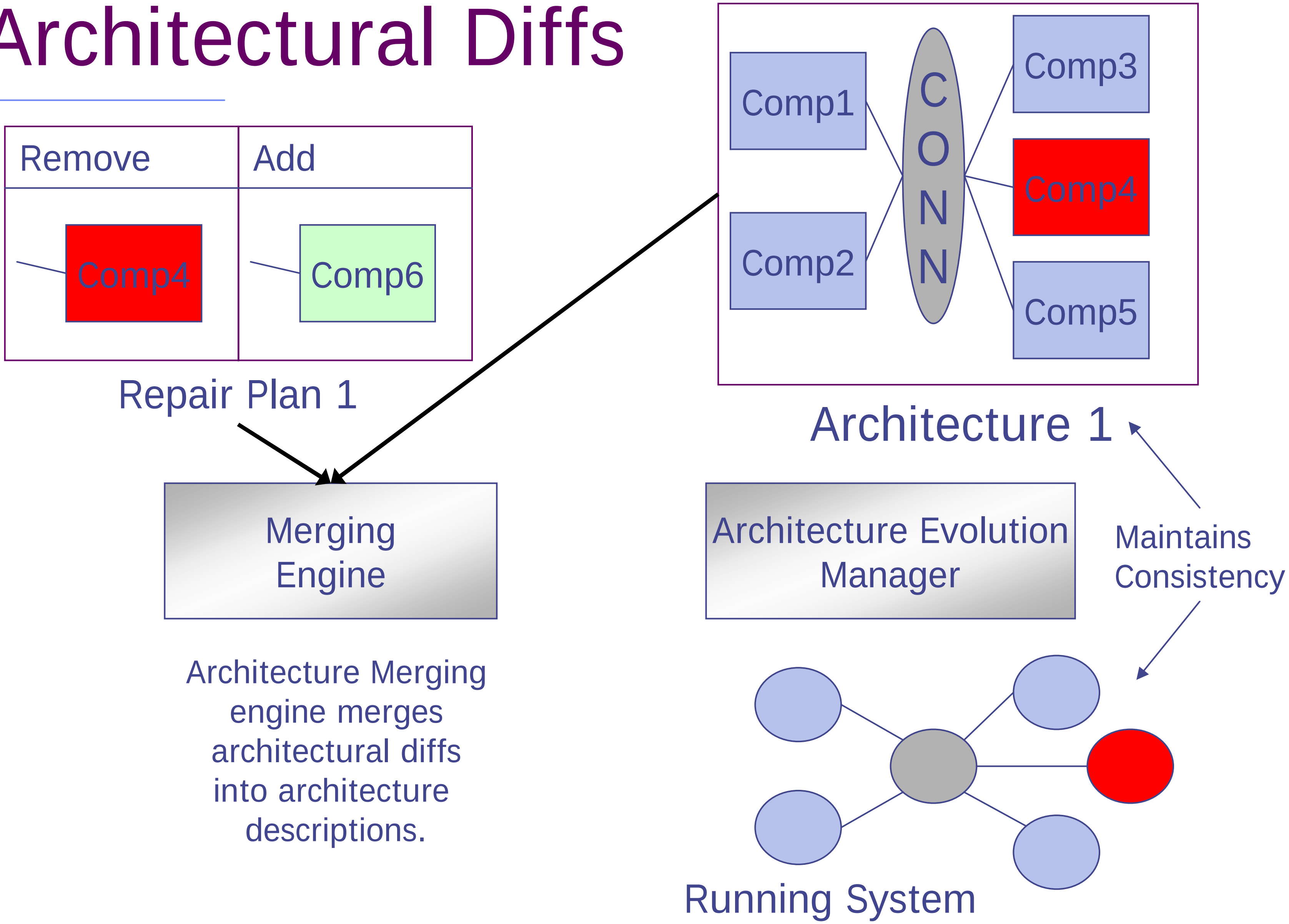


Architecture 1



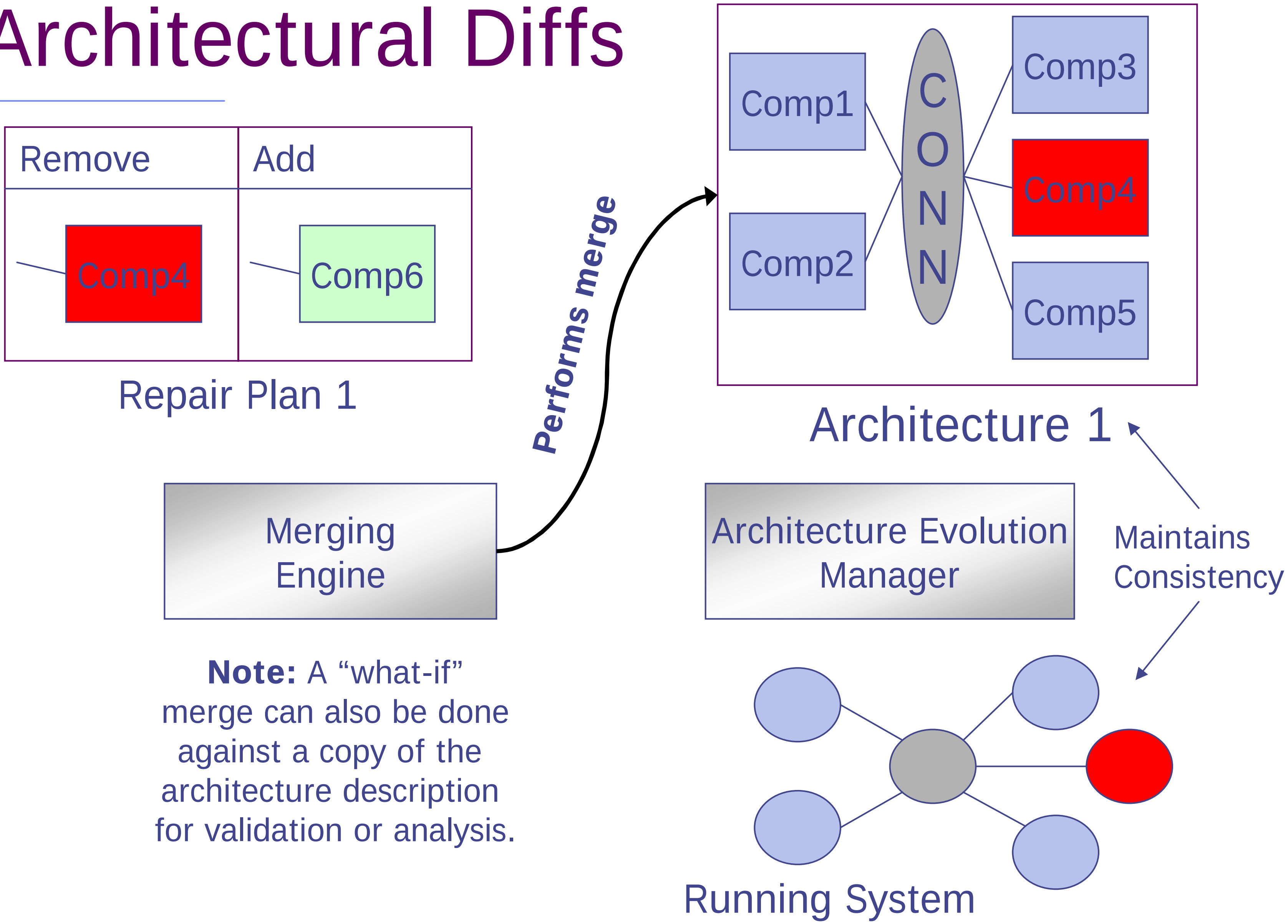
Running System

Effecting Repairs Using Architectural Diffs



Architecture Merging engine merges architectural diffs into architecture descriptions.

Effecting Repairs Using Architectural Diffs



Remove

Add

Comp4

Comp6

Repair Plan 1

Performs merge

Merging Engine

Comp1

Comp2

CONN

Comp3

Comp4

Comp5

Architecture 1

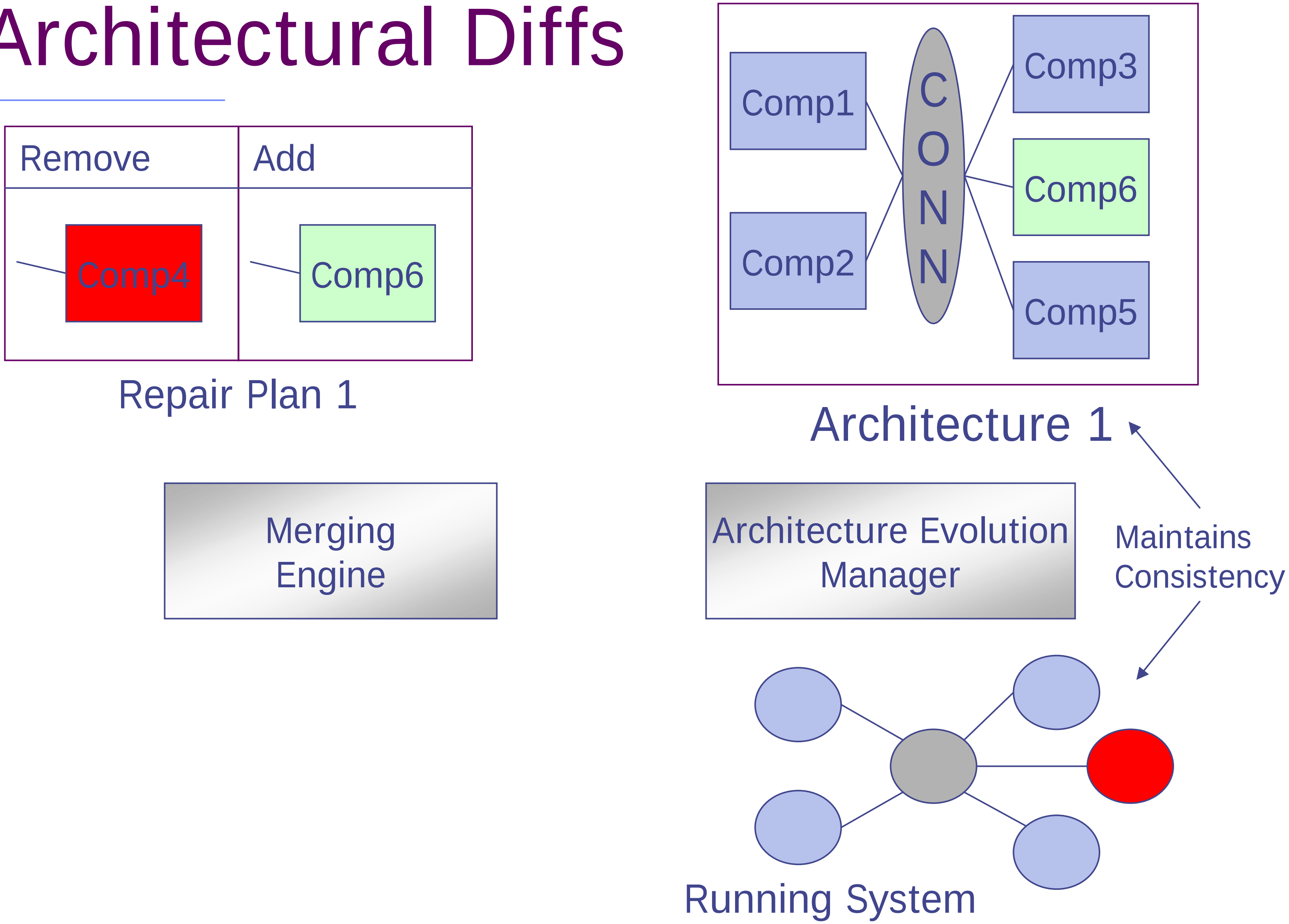
Architecture Evolution Manager

Maintains Consistency

Running System

Note: A "what-if" merge can also be done against a copy of the architecture description for validation or analysis.

Effecting Repairs Using Architectural Diffs



Remove

Add

Comp4

Comp6

Repair Plan 1

Architecture 1

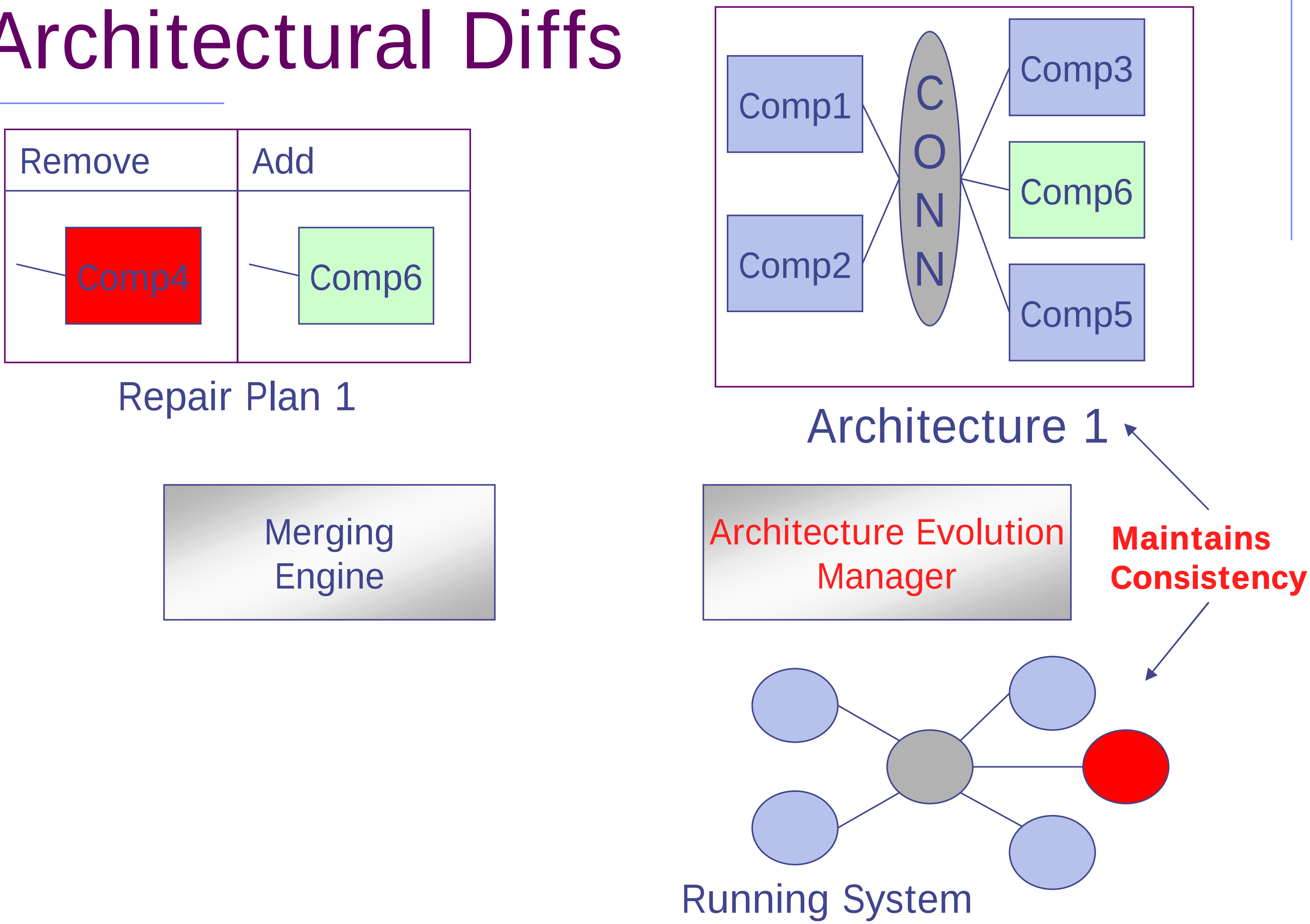
Merging Engine

Architecture Evolution Manager

Maintains Consistency

Running System

Effecting Repairs Using Architectural Diffs



Remove	Add
Comp4	Comp6

Repair Plan 1

Comp1

Comp2

CONN

Comp3

Comp6

Comp5

Architecture 1

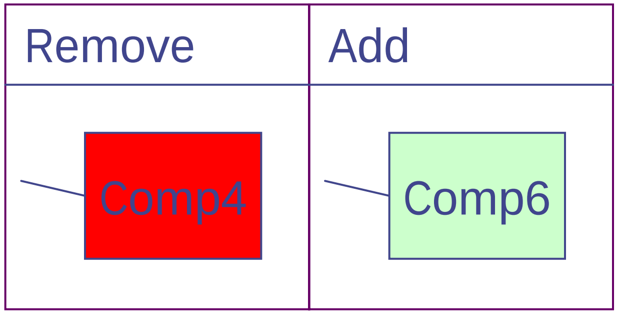
Merging Engine

Architecture Evolution Manager

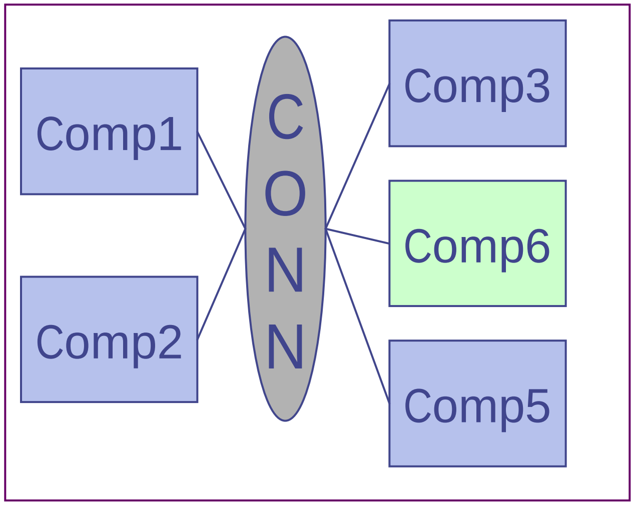
Maintains Consistency

Running System

Effecting Repairs Using Architectural Diffs



Repair Plan 1

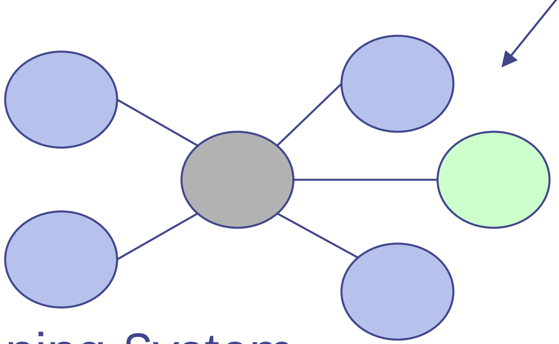


Architecture 1

Maintains Consistency

Merging Engine

Architecture Evolution Manager



Running System

Applications Targeted

- Spacecraft/Spacecraft Ground Systems
  - Architecture modeling formalism, ideas about dynamism already being adopted by MDS project at JPL
- Other component-based, event-driven systems
  - Military command and control
- Multi-agency systems
  - Coalition warfare among allied partners with independently developed systems

Future Work/Top Ideas

- Distributed Dynamism
  - Making repairs in the face of
    - (Partial) link failure,
    - (Partial) node failure
    - Asymmetric connectivity
- Are diffs sufficient as repair plans?
  - Ordering of changes
  - Additional information needed to make changes
- Approaches to quiescence
  - Inspired by Kramer & Magee