Architecture-Driven Self-Adaptation and Self-Management in Robotics Systems

George Edwards, Joshua Garcia, HosseinTajalli, Daniel Popescu, Nenad Medvidovic, Gaurav Sukhatme

Computer Science Department
University of Southern California

{gedwards, joshuaga, tajalli, dpopescu, neno, gaurav}@usc.edu

Brad Petrus

Bosch Research and Technology Center

brad.petrus@us.bosch.com

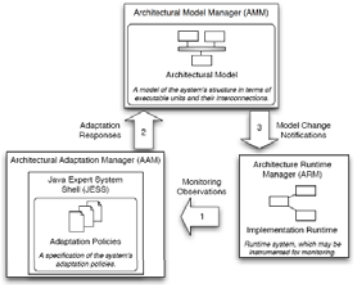
Mobile Robotics Systems

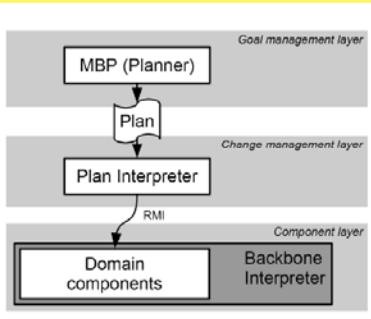
- Difficult or impossible to suspend execution for modifications and upgrades
  - Require **run-time adaptation**
- Heterogeneous, mobile, ubiquitous
  - Management and direction of run-time adaptations must be **decentralized** and **autonomous**

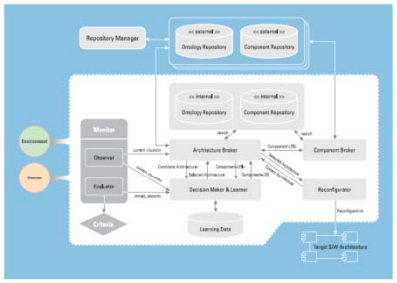
Motivation

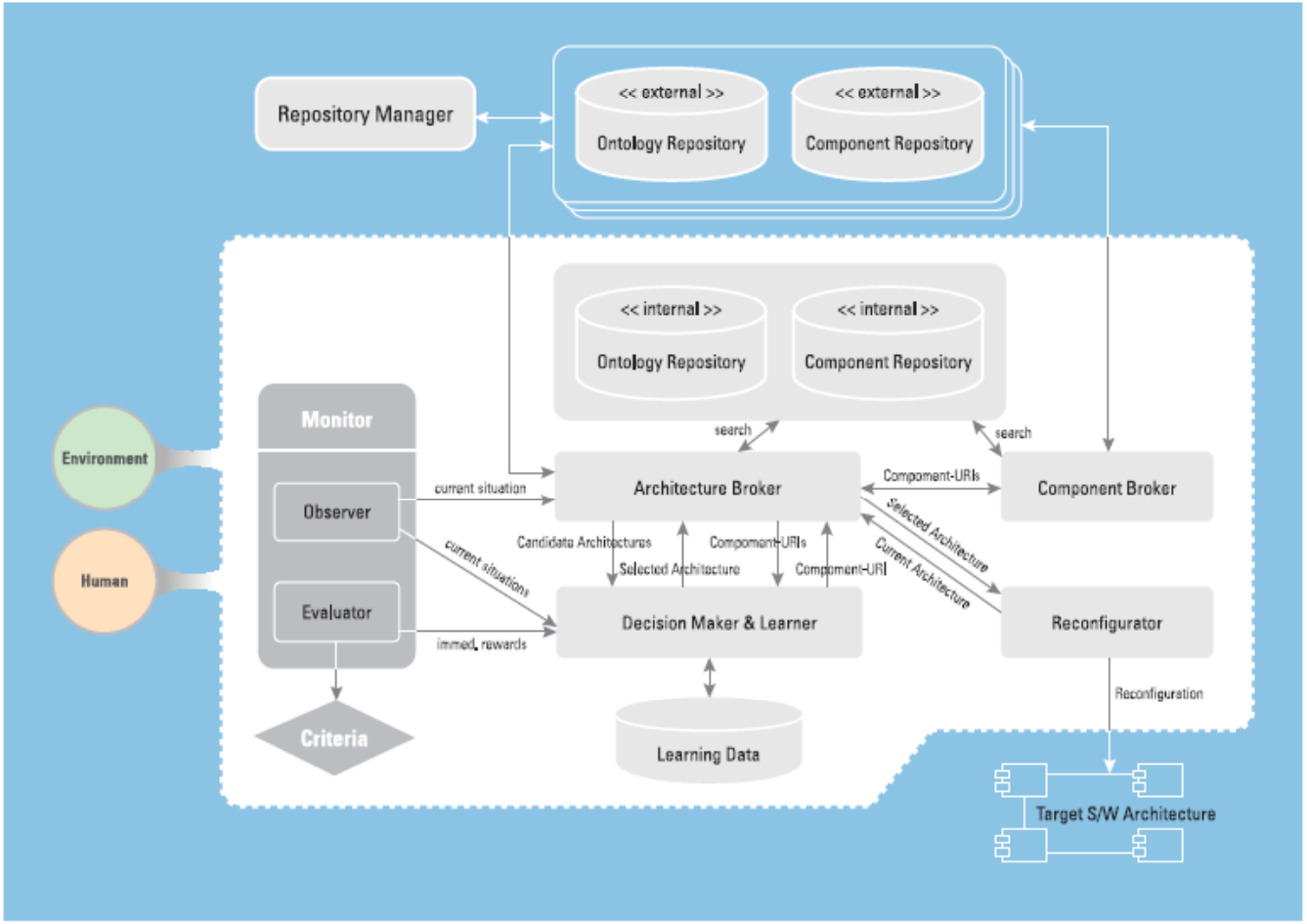
- **Observation:** Several architecture-driven adaptation frameworks with common design principles proposed
- **Problem:** How can we factor out and codify the common principles among these frameworks?
- **Solution:**
  - Define an architectural style
  - Extend an architectural middleware

Self-Adaptive Architectures









D. Sykes et al. ***From Goals to Components: A Combined Approach to Self-Management***. In Proceedings of SEAMS 2008, pages 1–8.
pages 105–112.

Commonality Among Approaches

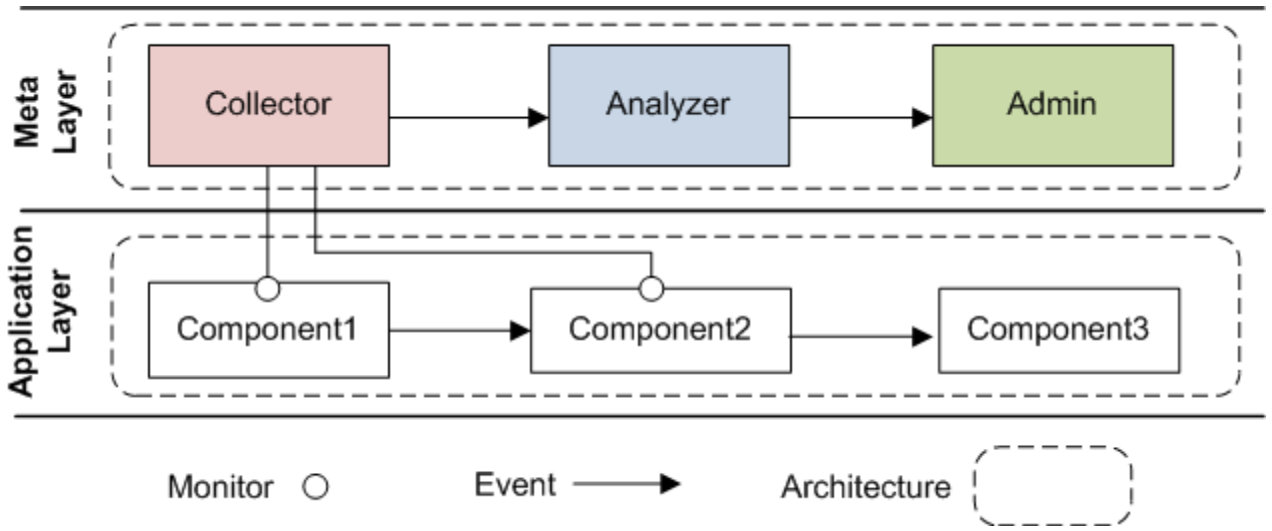
- Architecture-driven
  - **Architectural elements** as targets for **adaptation operations**
- Layered organization
  - **Application layer** implements app requirements
  - **Control layer** implements adaptation capabilities
- Adaptation activities
  - **Collecting** information
  - **Analysis** and decision making
  - **Administering** desired changes

Adaptive Layered Architectures

- Layering relationship:
  - In traditional layered systems, components **invoke the services of** the layer below
  - In adaptive layered systems, components **monitor, manage, and adapt** the layer below
- Defined roles for meta-level components:
  - **Collectors** gather information
  - **Analyzers** make decisions
  - **Admins** execute changes

Meta-Level Components

- **Architecturally aware**: have direct access to the run-time instantiations of a system's architectural elements
- Can **hierarchically** and/or **recursively** monitor, manage, and adapt other meta-level components

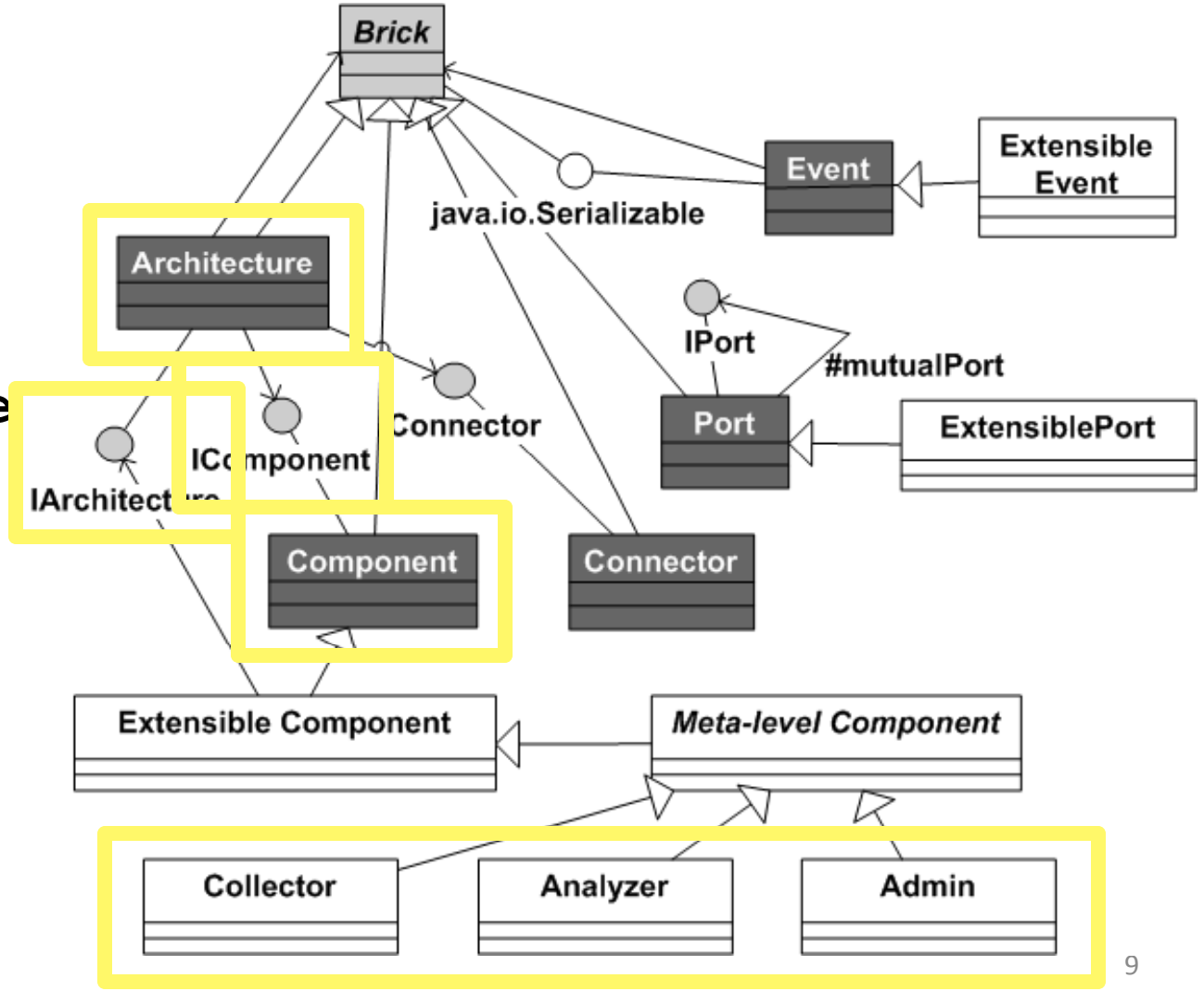


Implementation Approach

- Differentiate meta-level components from application components
  - Define **architectural roles** and **constraints**
- Provide programming constructs for developing application-specific meta-level components
  - **Provide role capabilities** off-the-shelf
  - **Enforce constraints** at run-time
- Leave points of variability undefined/unimplemented

Implementation Support – Prism-MW

- Meta-level components possess a reference to `IArchitecture` interface
- Meta-level components implement `IComponent` interface



Collector

- Gather data about components at the layer below
  - Structural and behavioral information
- Install monitors in application elements
  - Application developers implement monitors based on application-specific requirements
- Aggregate monitor data and transmit to Analyzer
- Hold a read-only reference to the application architecture representation

Analyzer

- Evaluate **adaptation policies** based on monitor data and **construct adaptation plans**
- Developers implement mechanisms for evaluating policies and constructing plans
- Adaptation plans passed to Admins
- Hold a read-only reference to the architecture representation

```
public interface Policy {
    public boolean isTriggered(CollectorData data);
    public Plan computeAdaptationPlan(IArchitecture arch);
}

public class EnergyPolicy implements Policy {

    private int THRESHOLD = 500;

    public boolean isTriggered(CollectorData data) {
        if (data.getType().equals(CollectorDataTypes.ENERGY_TYPE)) {
            Integer currentEnergy = (Integer) data.value();
            return currentEnergy < THRESHOLD;
        } else {
            return false;
        }
    }

    public Plan computeAdaptationPlan(IArchitecture arch) {
        return DeploymentCalculator.computeOptimizedPlan(
                OptimizationGoal.SAVING_ENERGY, arch);
    }
}
```

Admin

- Execute **adaptation plans** created by Analyzers
  - Default adaptation mechanisms are built-in
  - Add, remove, connect, disconnect
- Invoke specialized interfaces provided by application elements
- Hold a writable reference to the application architecture representation

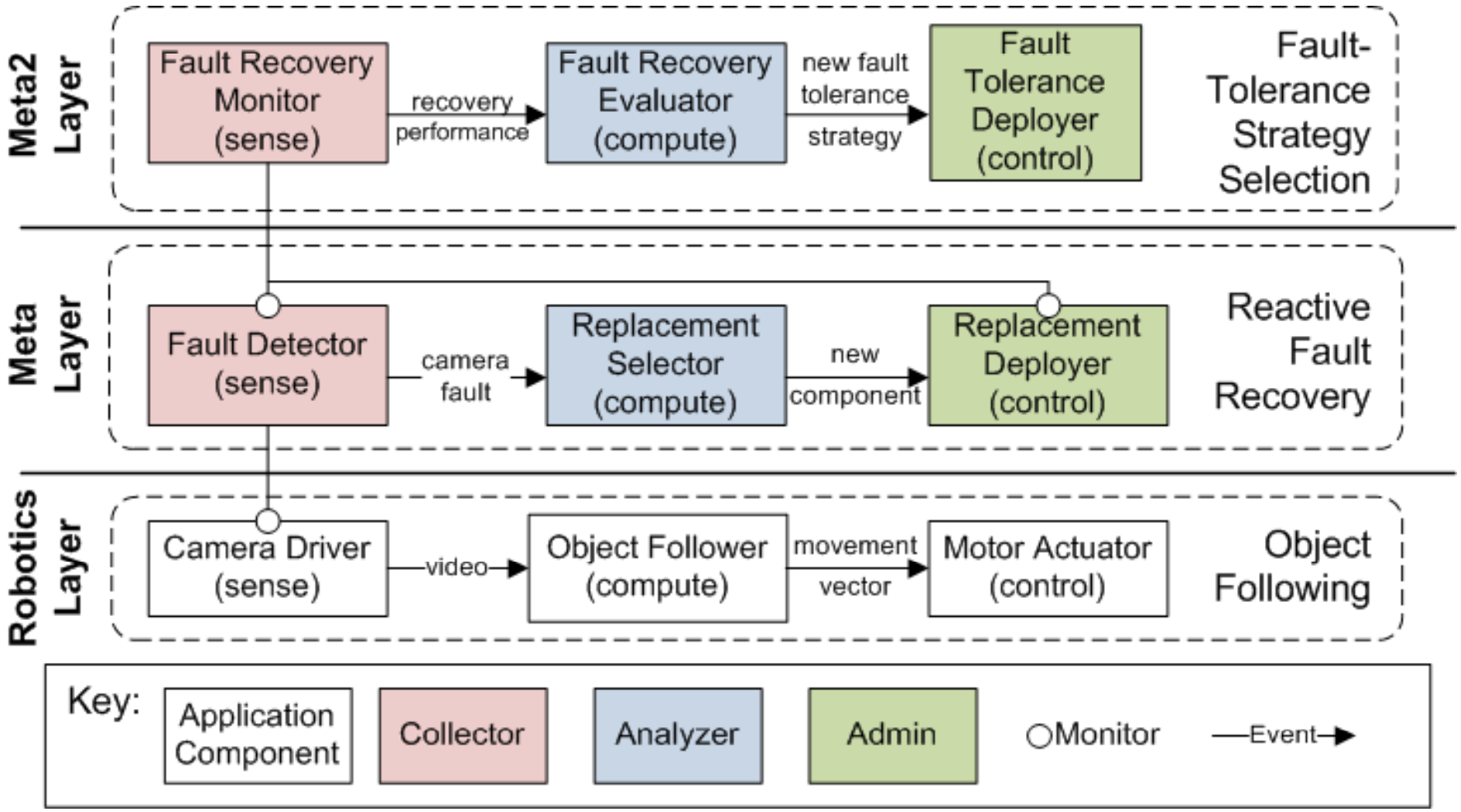
```
public class Admin extends AbstractImplementation {

    private IArchitecture arch;

    private void removeComponent(IComponent comp) {
        List<IPort> ports = comp.getPorts();
        for (IPort aPort : ports) {
            arch.unweld(aPort, aPort.getMutualPort());
        }
        arch.remove(comp);
    }

    public void execute(Plan adaptationPlan){
    ...
```

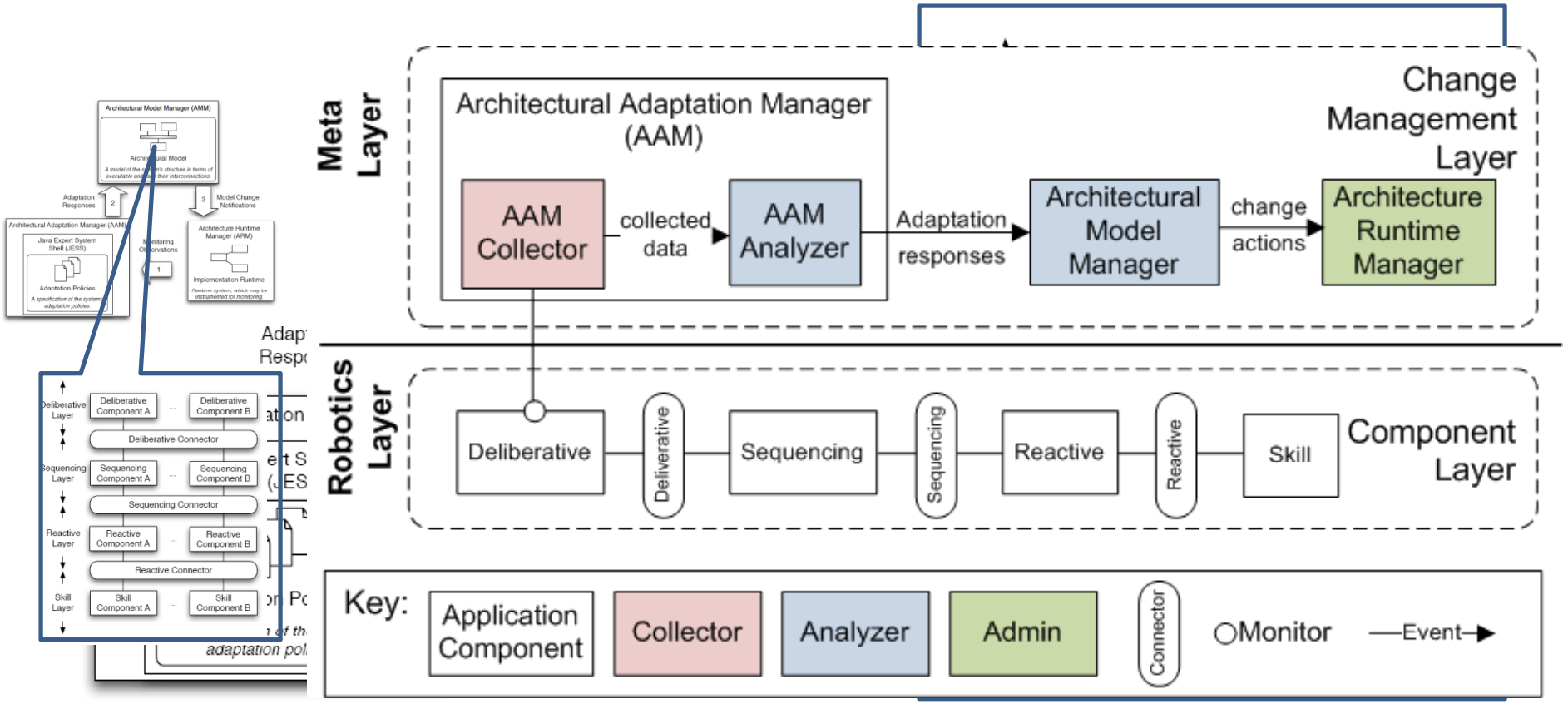
Fault Tolerance Example



Variations on Adaptive Layering

- Layer definition
  - An arbitrary number of layers may exist
  - A layer may contain only a subset of roles
- Composition of roles
  - Meta-level components may perform more than one role
- Distribution
  - Meta-level components may interact across host/network boundaries
- Multiplicity of interactions
  - Meta-level component interactions may be N-to-M

Mapping PBAAM to Adaptive Layering



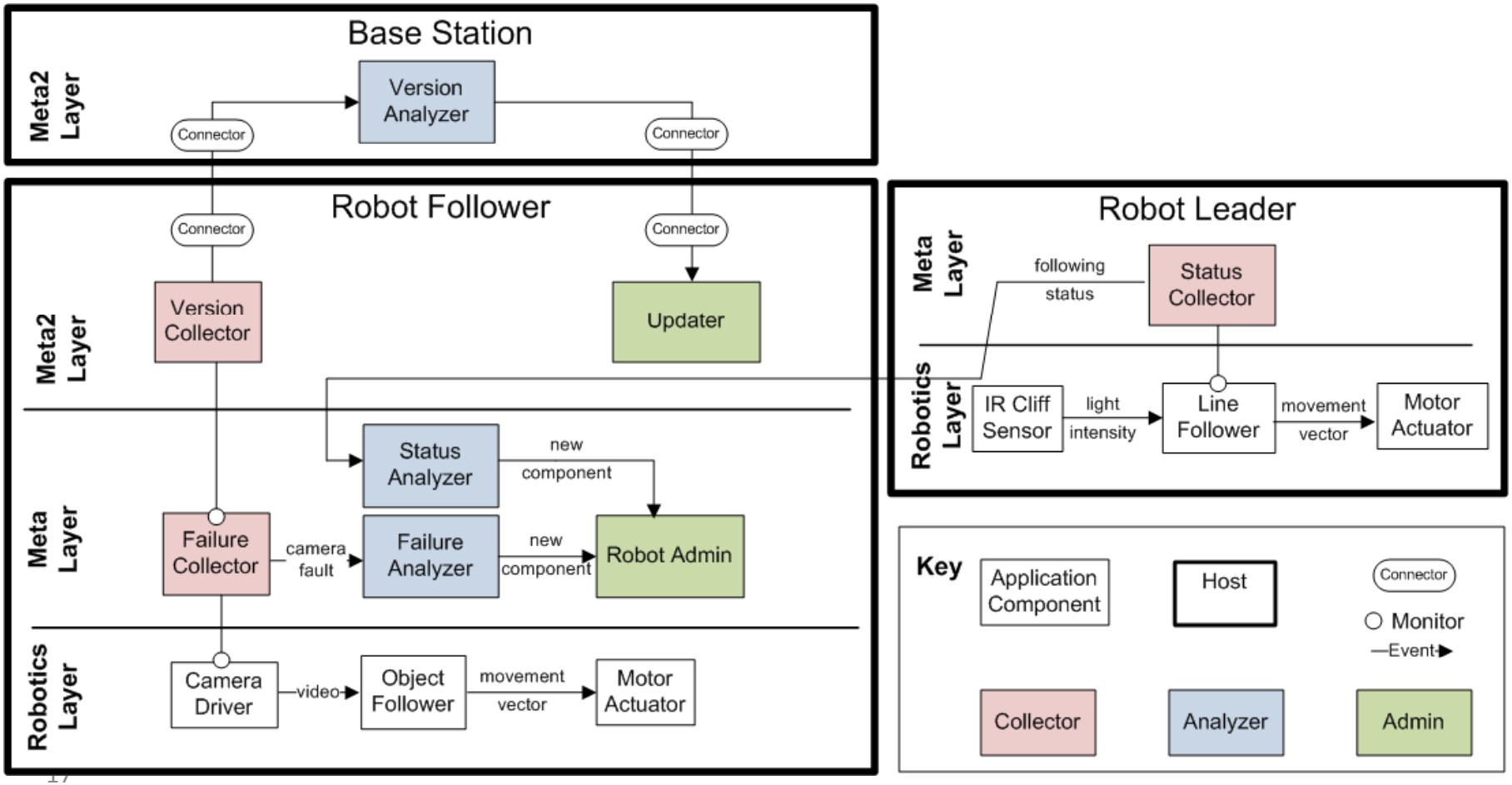
Demonstration Scenario

- Platoon of autonomous robots
  - Experience hardware failures
  - Encounter base stations, stationary sensors

- Several types of adaptivity
  - Ensuring correct operation (**fault tolerance**)
  - Modifying functionality (**dynamic update**)
  - Adding new capabilities (**resource discovery**)
  - Moving components (**redeployment**)

Play video

Demonstration Scenario



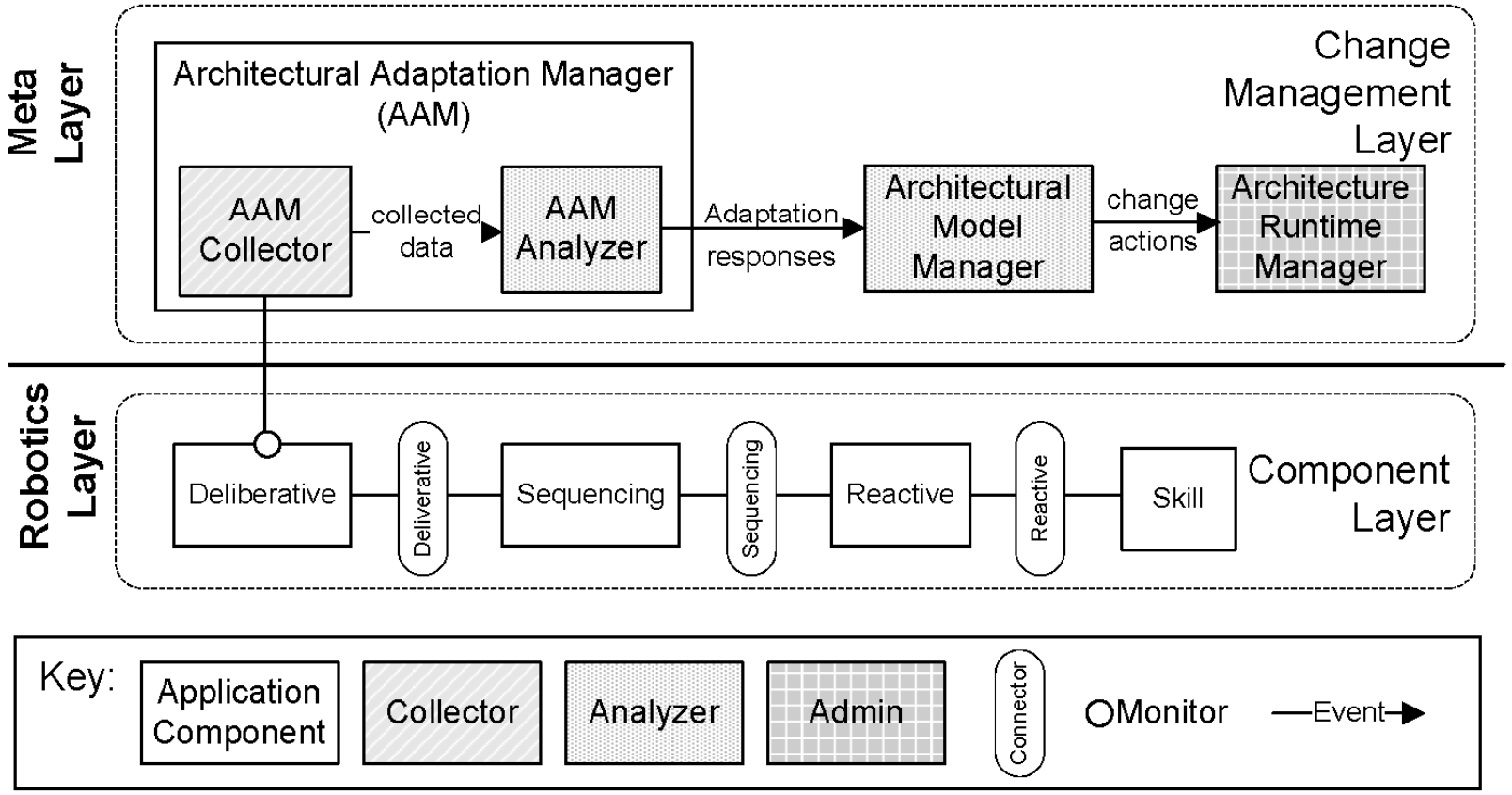
Conclusions

- The adaptive layered style is intended to capture the common design decisions used in self-adaptive architectures
- We defined a set of roles and constraints for adaptive layered systems
- We implemented support for adaptive layered systems in an architectural middleware by providing role capabilities and enforcing constraints

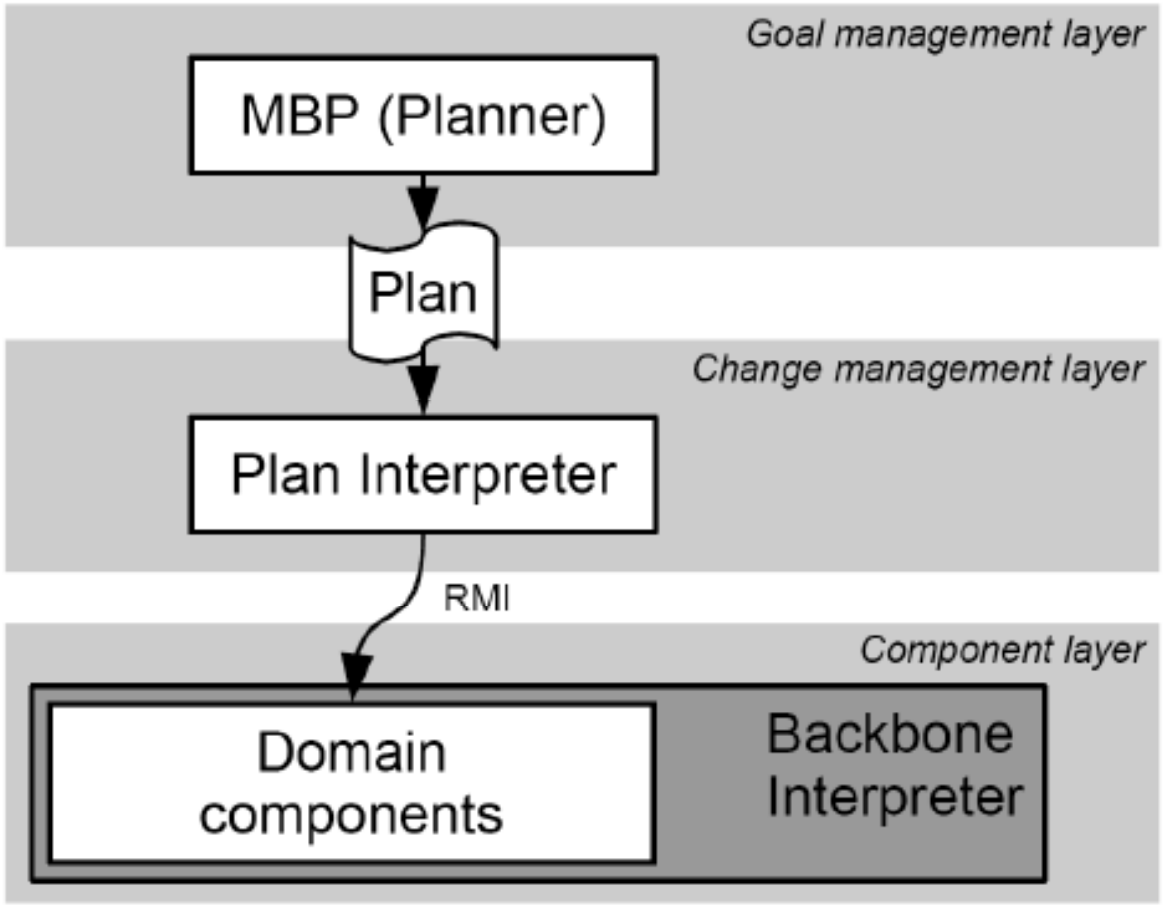
Questions?

Backup Slides

Application to PBAAM



Three Layer Reference Model



Application to Three Layer Reference Model