Autonomic Management of GCM/Proactive Components

INRIA/NICLabs SCADA Associate Team

16 June 2014

Autonomic Computing

Ability of a computer resource to adapt itself to changes in the runtime environment or in the desired quality of services.

- Response to the complexity in the maintenance of systems
- Based on the idea of self-governing systems
- Requires high-level objectives from an administrator

Autonomic Computing

Ability of a computer resource to adapt itself to changes in the runtime environment or in the desired quality of services.

- Response to the complexity in the maintenance of systems
- Based on the idea of self-governing systems
- Requires high-level objectives from an administrator

How?

Autonomic Computing

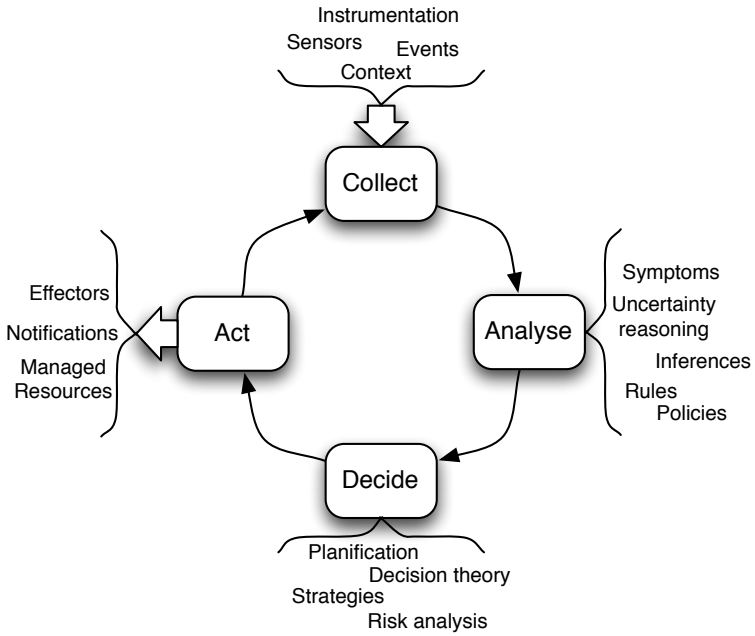
Ability of a computer resource to adapt itself to changes in the runtime environment or in the desired quality of services.

- Response to the complexity in the maintenance of systems
- Based on the idea of self-governing systems
- Requires high-level objectives from an administrator

How?

- *feedback control loop*
- Implementation referenced as *MAPE autonomic control loop*:

  "Monitor, Analyze, Plan & Execute"



Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

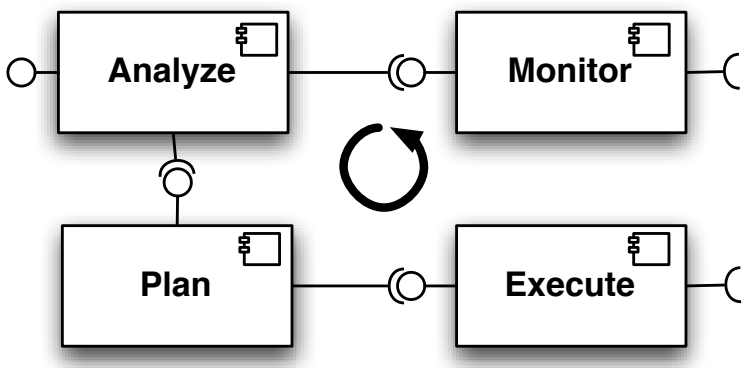
- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

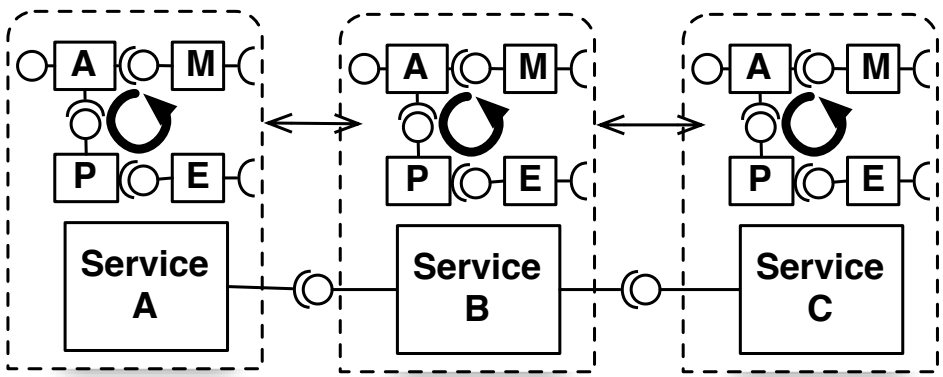


Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

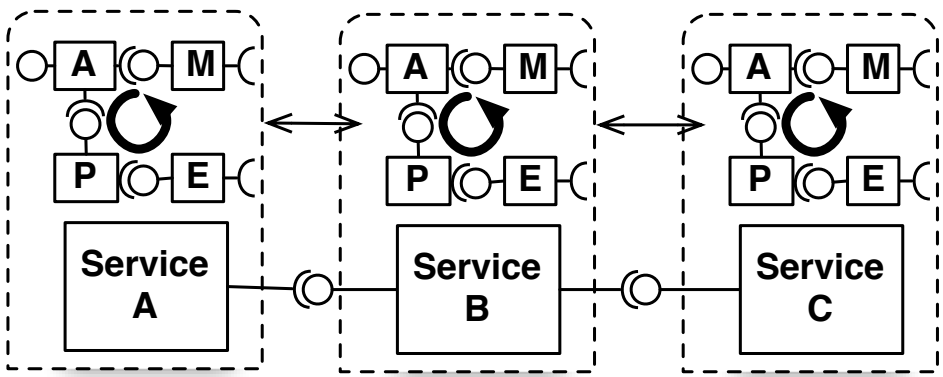


Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

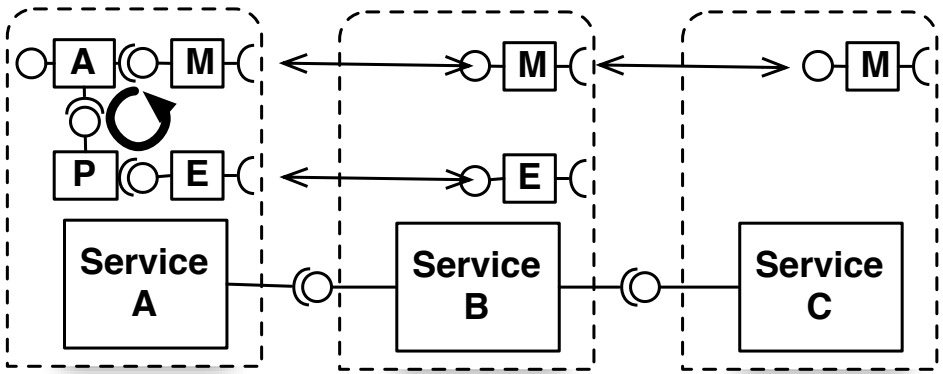


Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

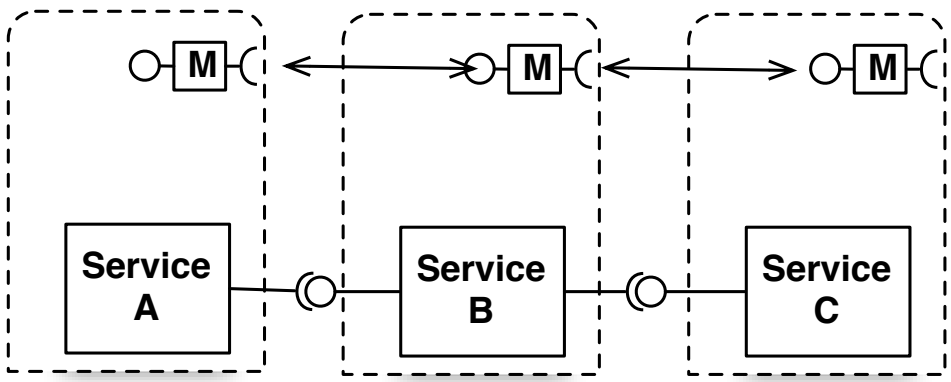


Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop

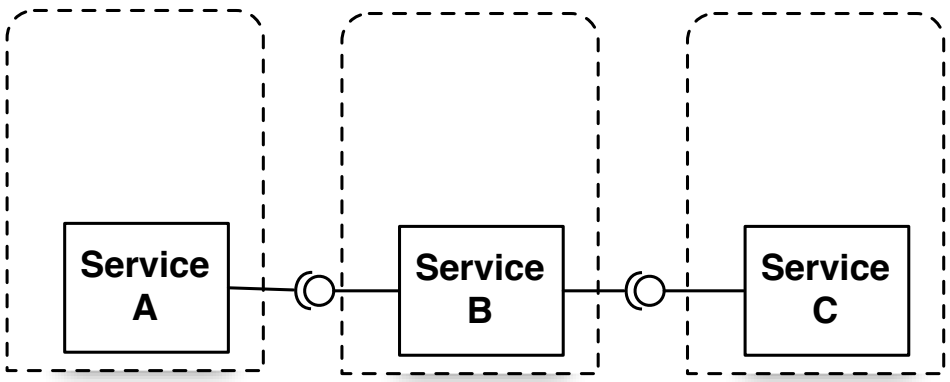


Design

Using Components to Provide a Flexible Adaptation Loop to Componente-based SOA Application

C. Ruz, F. Baude, B. Souvan

- Implement each phase of the autonomic control loop by a different component.
- Attach these components to each managed service.
- Allow dynamically reconfiguration of the autonomic control loop



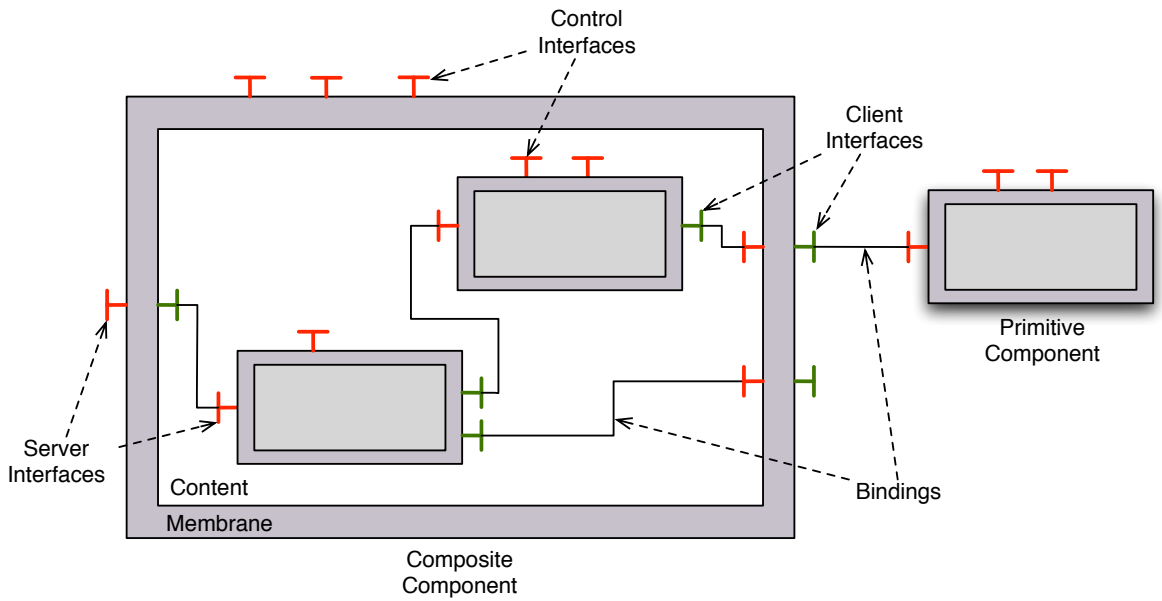
Goal

Goal:

Implement the autonomic control loop to provide Autonomic Management features to GCM/ProActive components

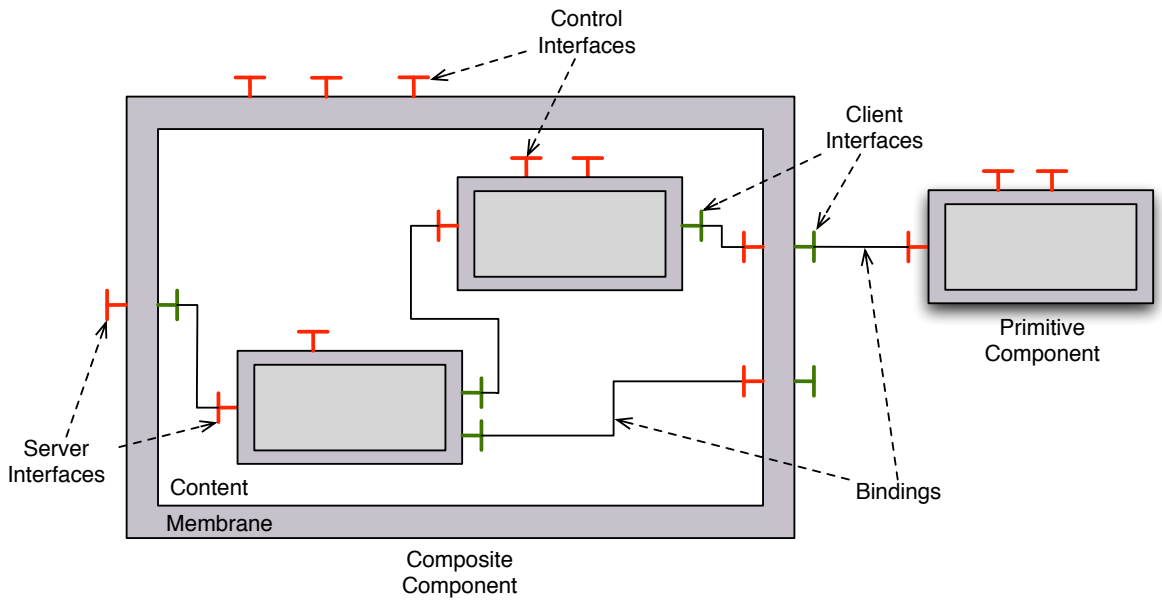
Implementation: Technical Background

- Grid Component Model (GCM)
  - Extension of the Fractal Component Model
    - Support for distributed deployment
    - Support for collective communications
- Using the GCM/ProActive reference implementation
  - Based on asynchronous active objects, and futures



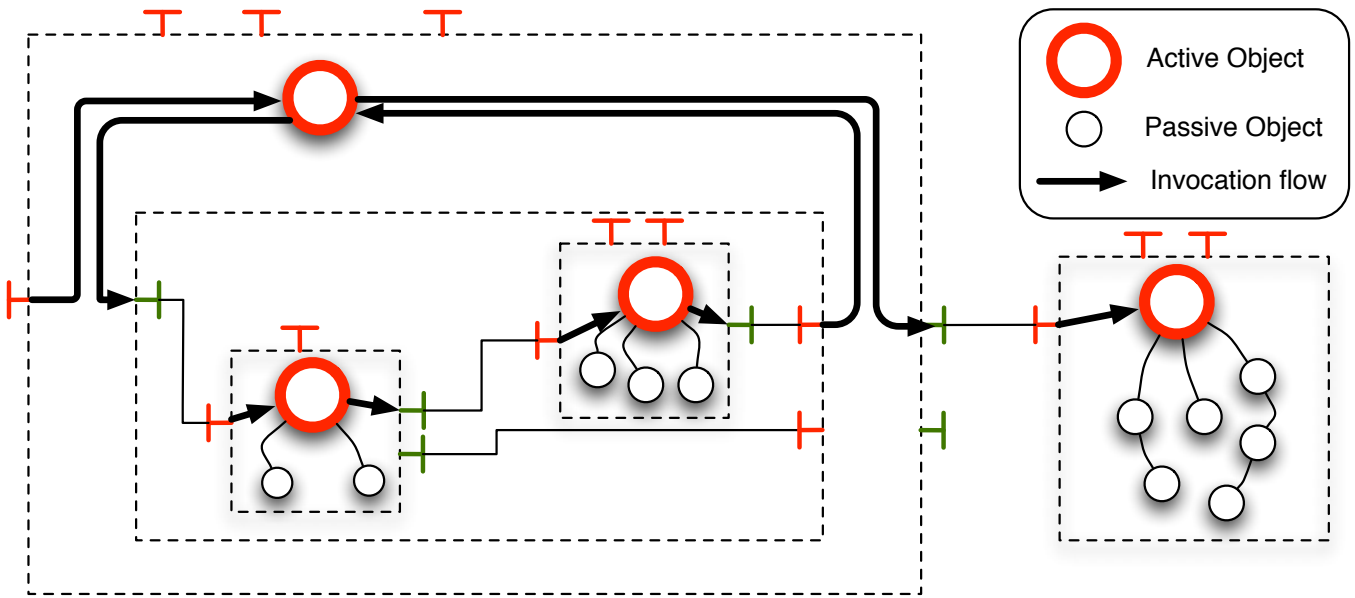
Implementation: Technical Background

- Grid Component Model (GCM)
  - Extension of the Fractal Component Model
  - Support for distributed deployment
  - Support for collective communications
- Using the GCM/ProActive reference implementation
  - Based on asynchronous active objects, and futures



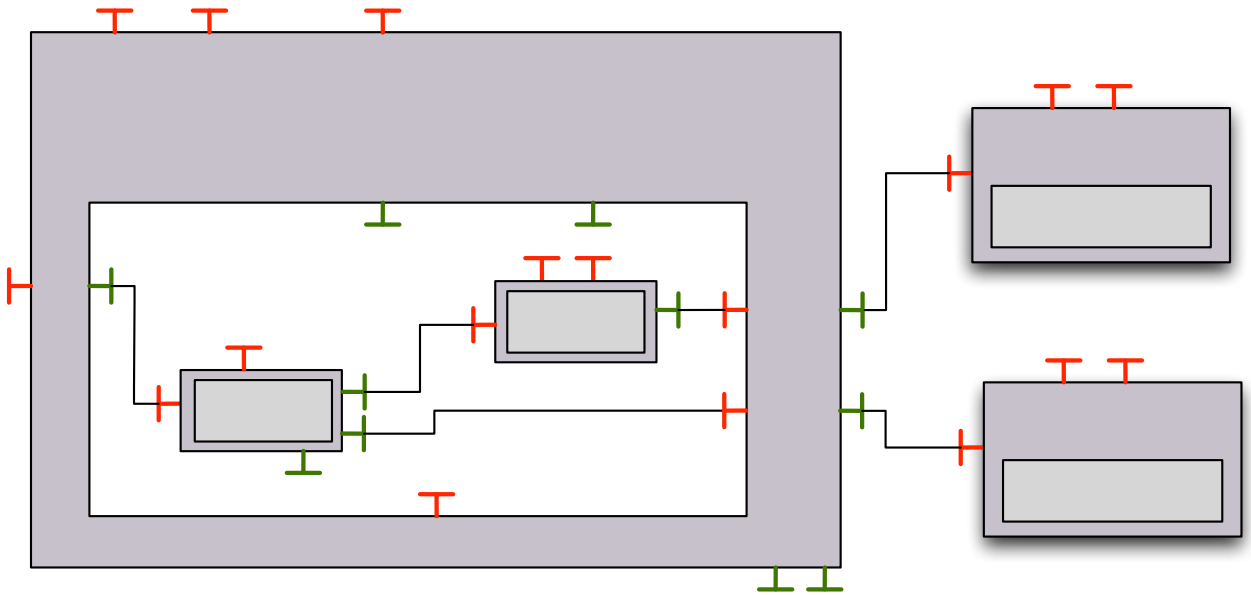
Implementation: Technical Background

- Grid Component Model (GCM)
  - Extension of the Fractal Component Model
  - Support for distributed deployment
  - Support for collective communications
- Using the GCM/ProActive reference implementation
  - Based on asynchronous active objects, and futures



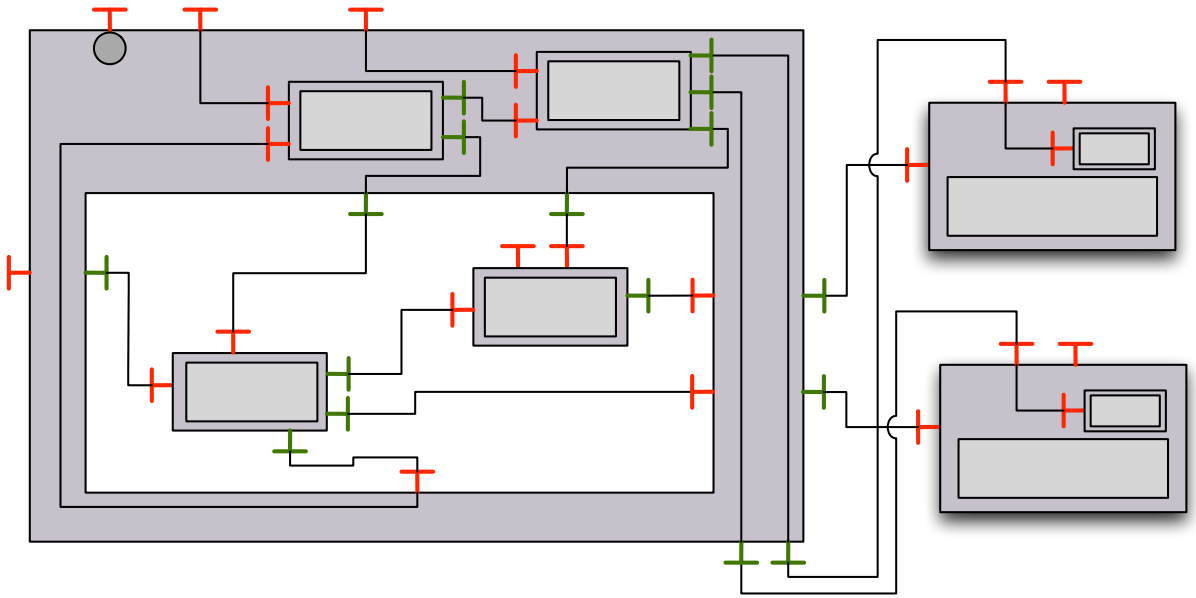
Implementation: Technical Background

- Grid Component Model (GCM)
  - Separation between F and NF concerns (Naoumenko, 2010)



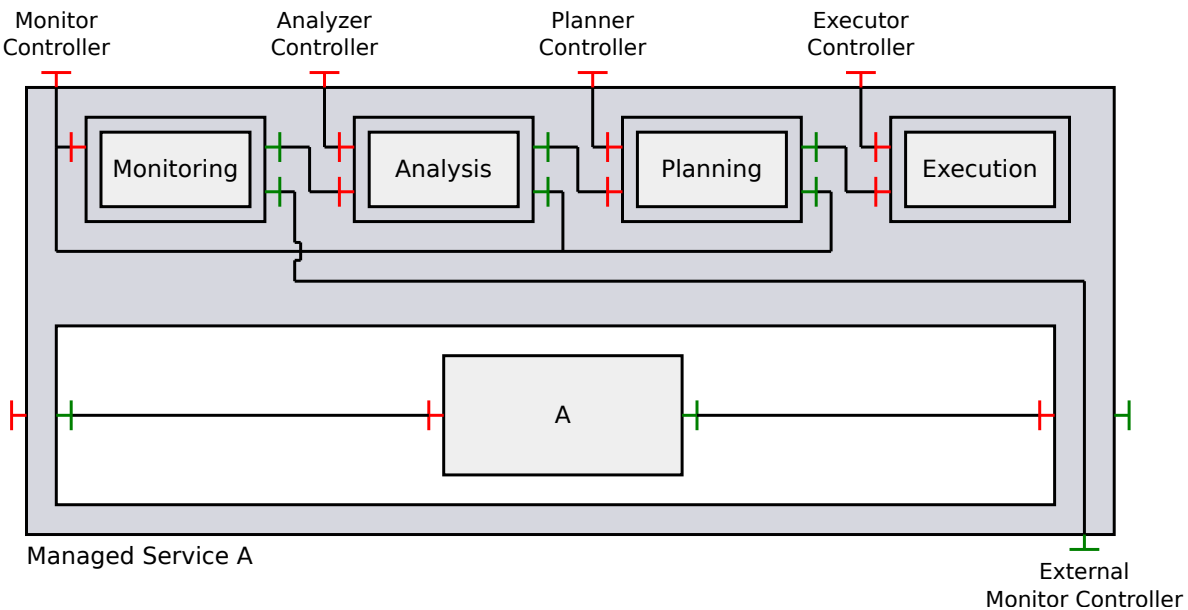
Implementation: Technical Background

- Grid Component Model (GCM)
  - Separation between F and NF concerns (Naoumenko, 2010)

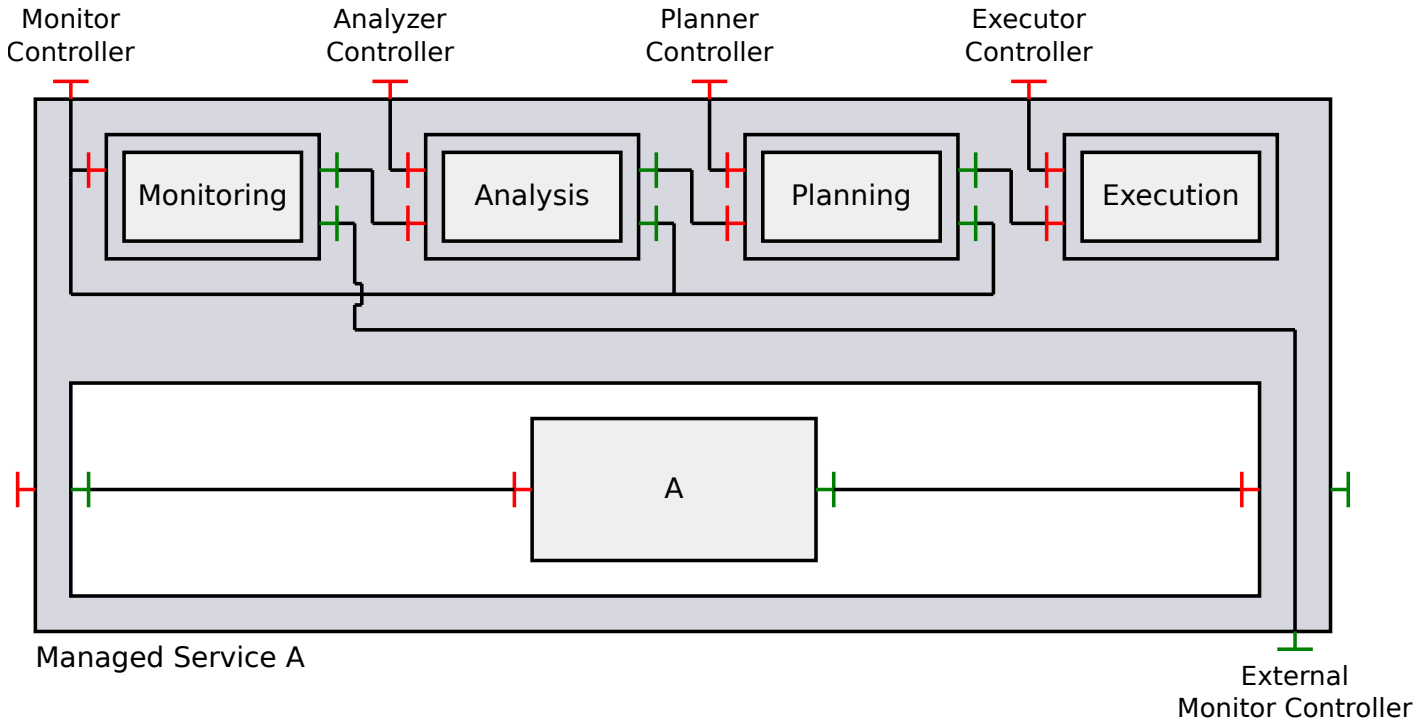


Implementation: Component Controllers

- MAPE Components attached to GCM membranes
- Using NF server and client interfaces
- Definition of an API to manipulate MAPE components

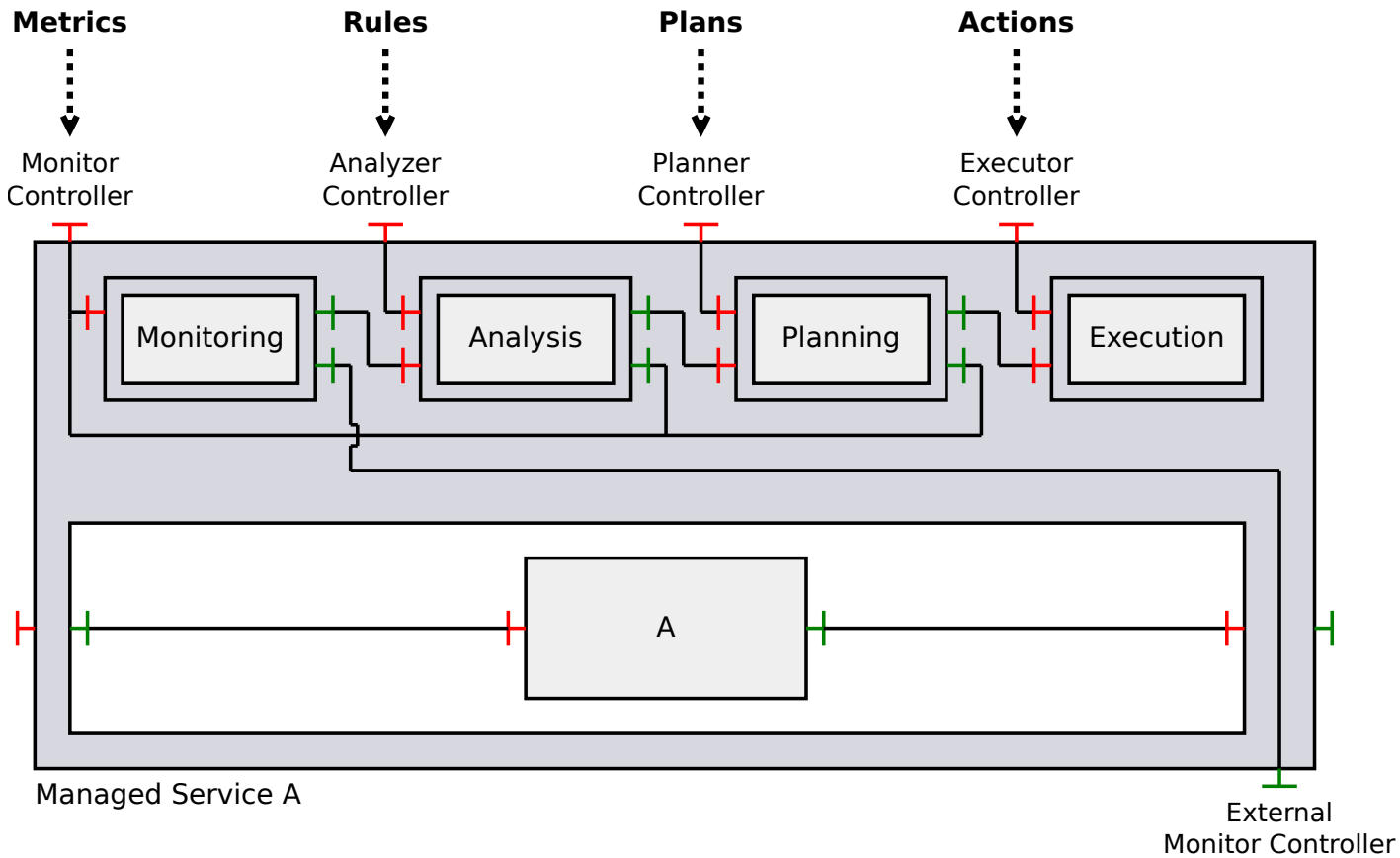


Implementation: Component Controllers



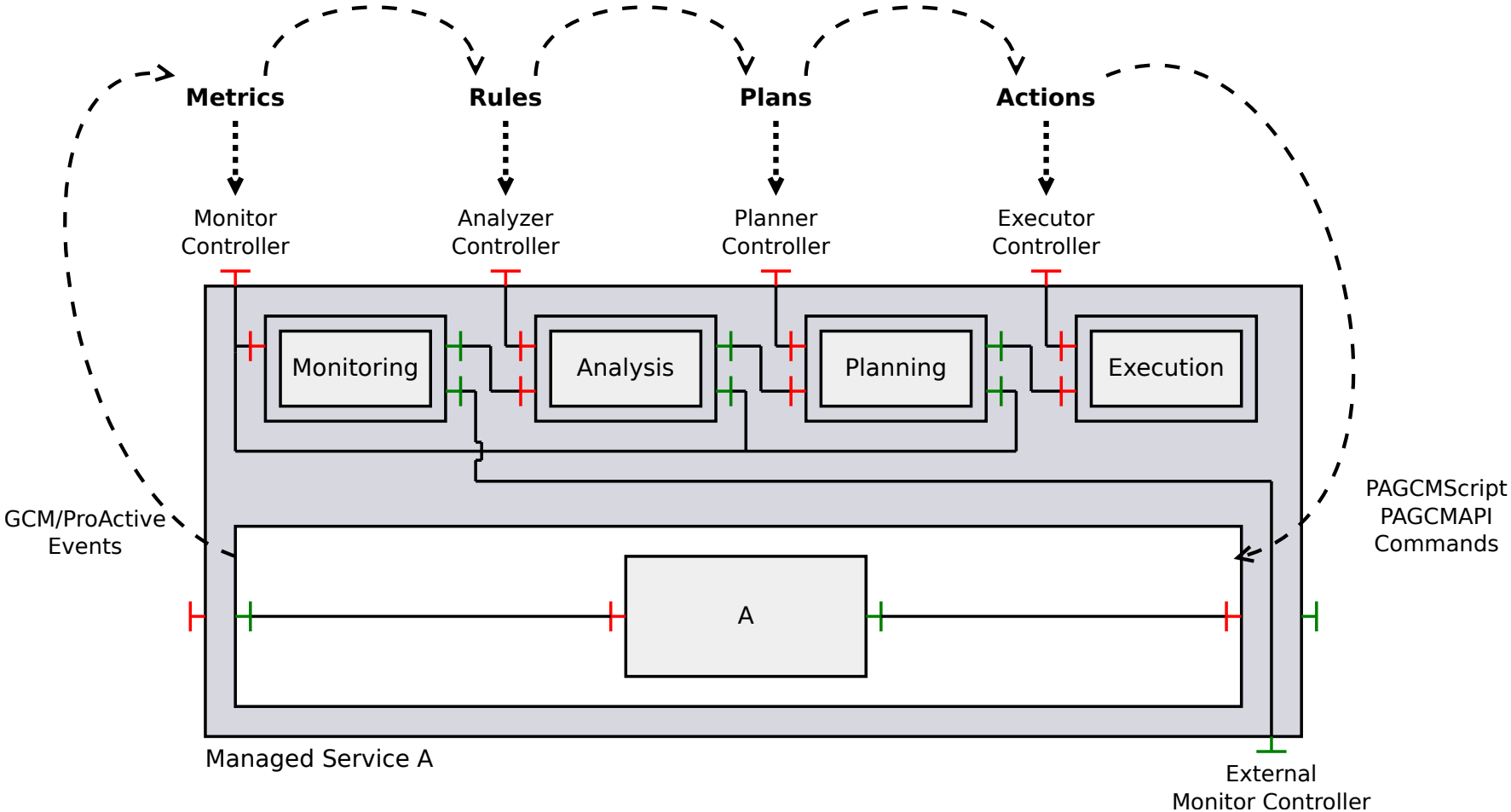
- (Metrics, Rules, Plans, Actions) = High-Level Objectives

Implementation: Component Controllers



- (Metrics, Rules, Plans, Actions) = High-Level Objectives

Implementation: Component Controllers

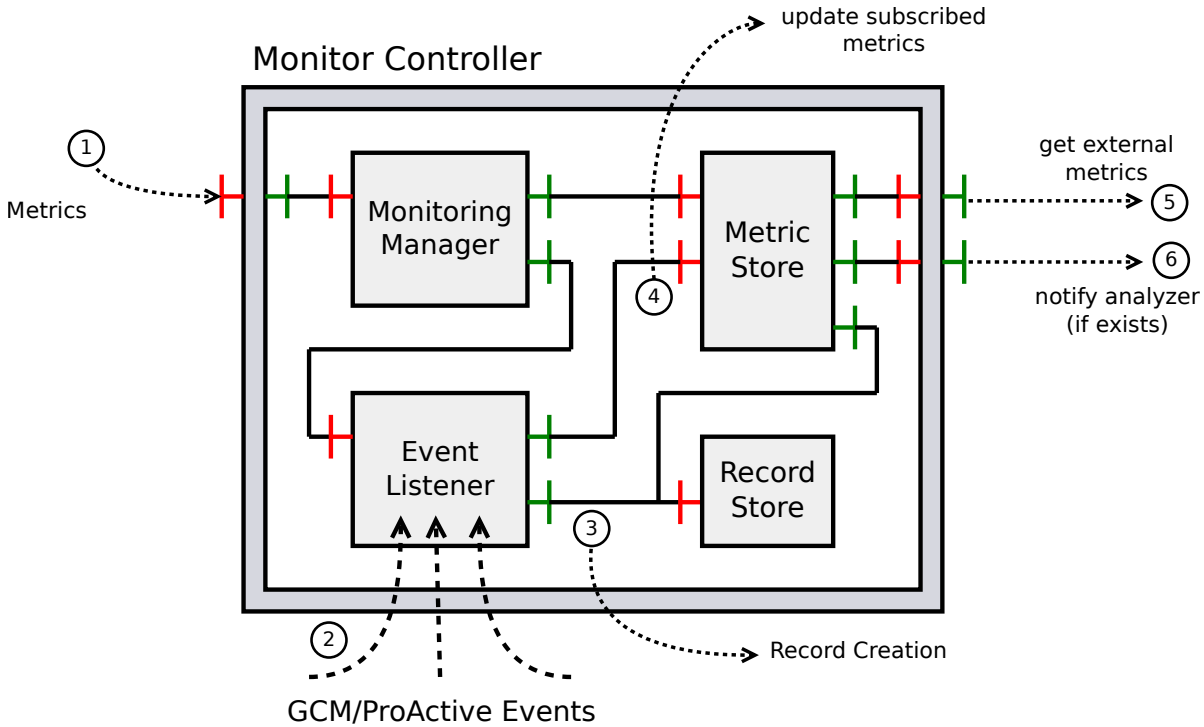


- (Metrics, Rules, Plans, Actions) = High-Level Objectives

Monitoring Component

Collection, storage, computation of metrics

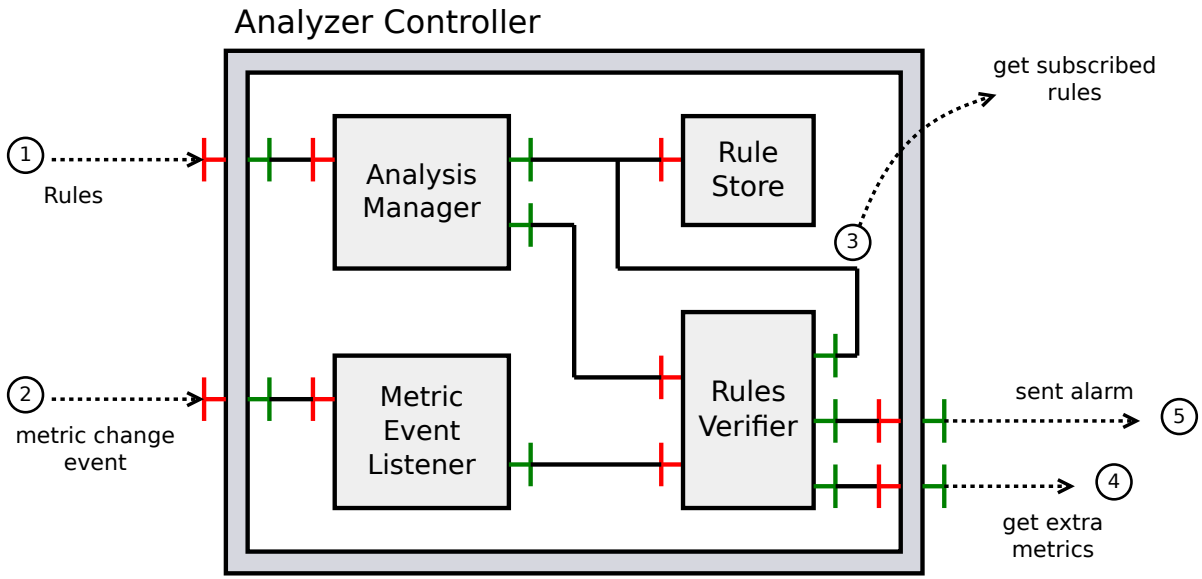
- Collecting JMX events from GCM/ProActive
- Supports insertion/removal of metrics
- Notifies active metrics changes



Analysis Component

Checking of conditions and generation of alarms

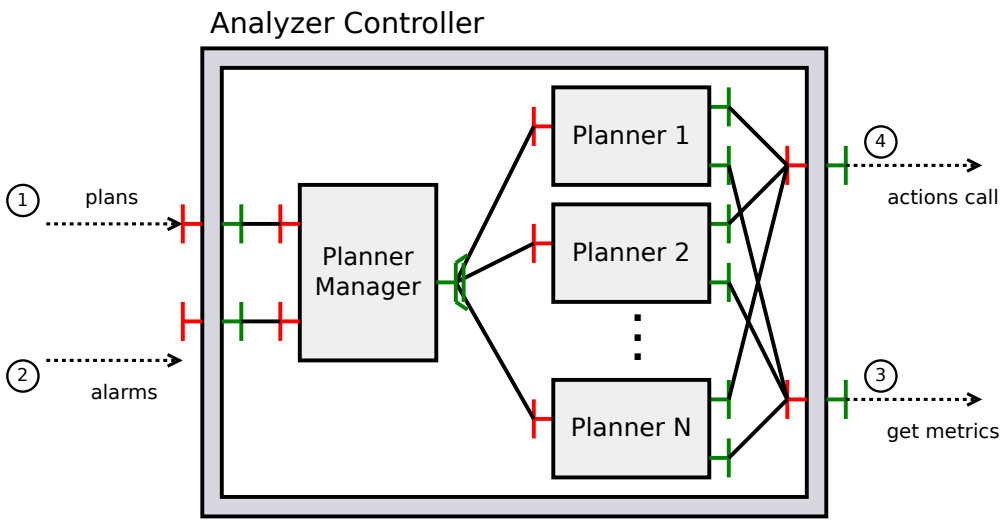
- Rules subscribe to Metrics
- Sends an Alarm object if necessary



Planning Component

Execution of planning algorithms (strategies)

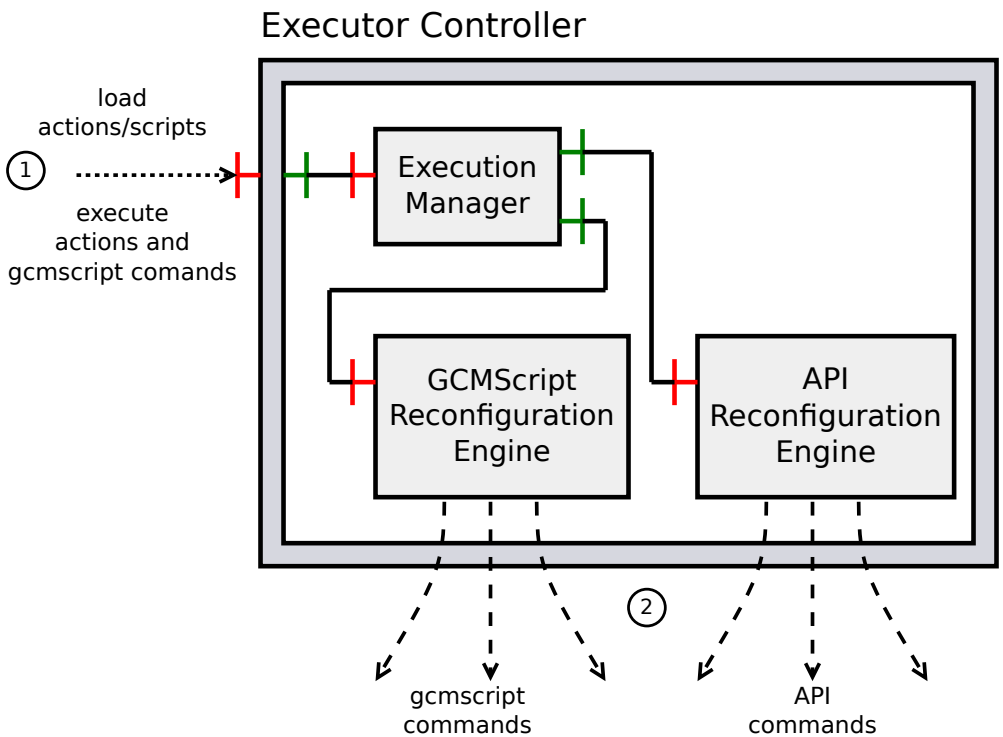
- Associates an Alarm to one or more strategies
- Support for multiple strategies using multicast interfaces
  - Selection, parallel execution of strategies



Execution Component

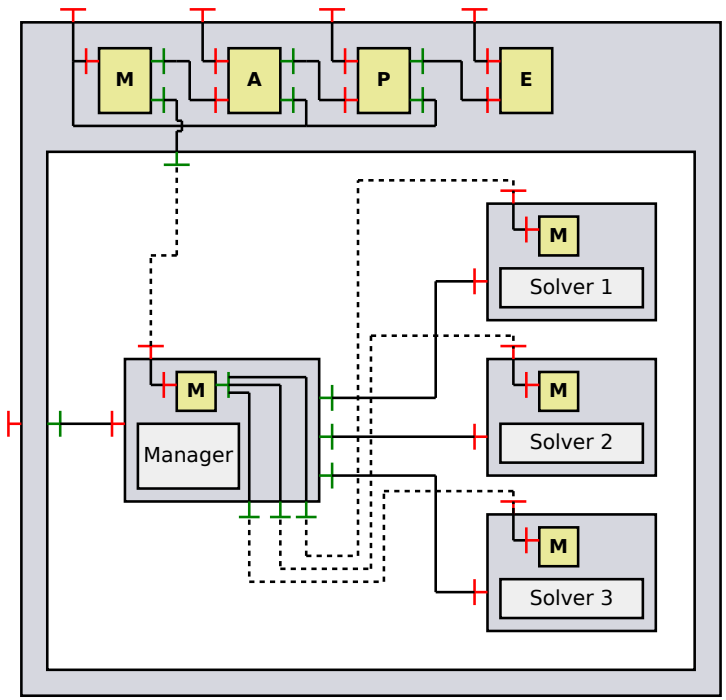
Execution of Actions over the component/service

- Support to execute reconfigurations using the GCM/ProActive API (Java code embedded in a Action object)
- Support to execute reconfigurations using PAGCMScript language code (extends of FScript).



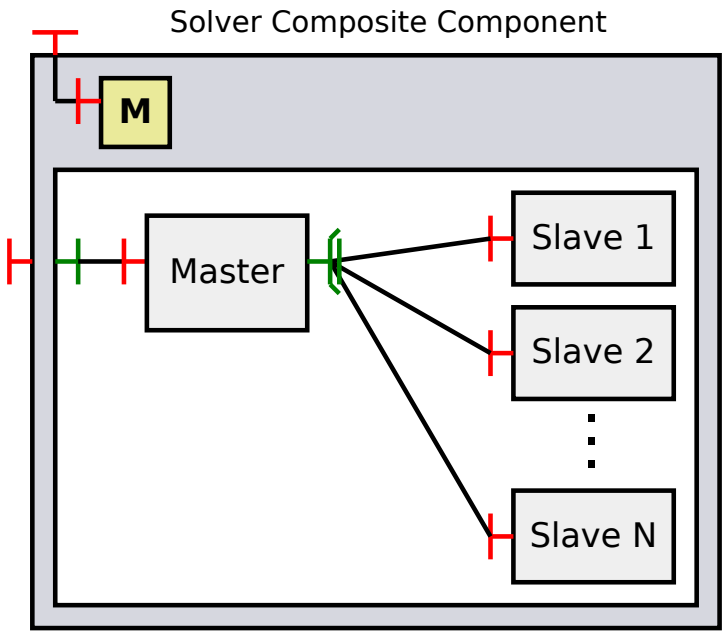
Use Case

- MD5Hash brute force cracker
- Multiactive service
- Each Solver deployed on on a different machine
- Each Solver has several workers (Slaves)

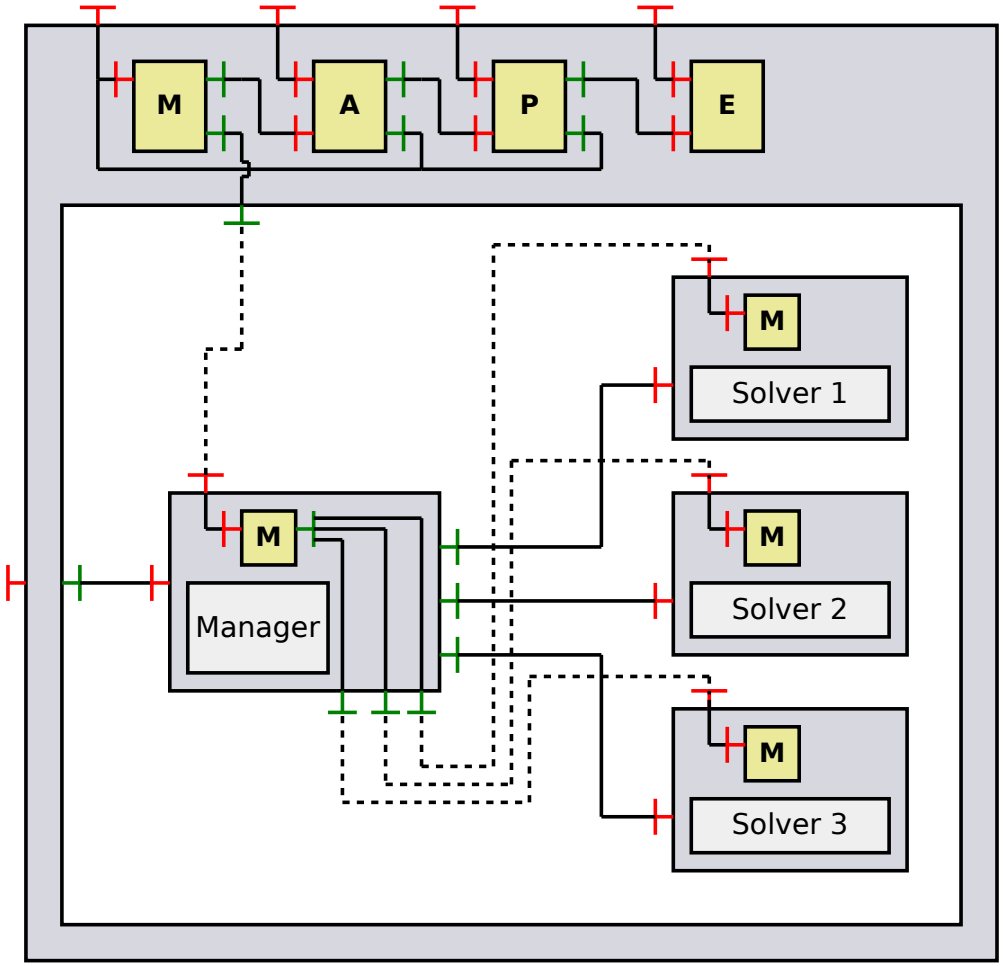


Use Case

- MD5Hash brute force cracker
- Multiactive service
- Each Solver deployed on on a different machine
- Each Solver has several workers (Slaves)

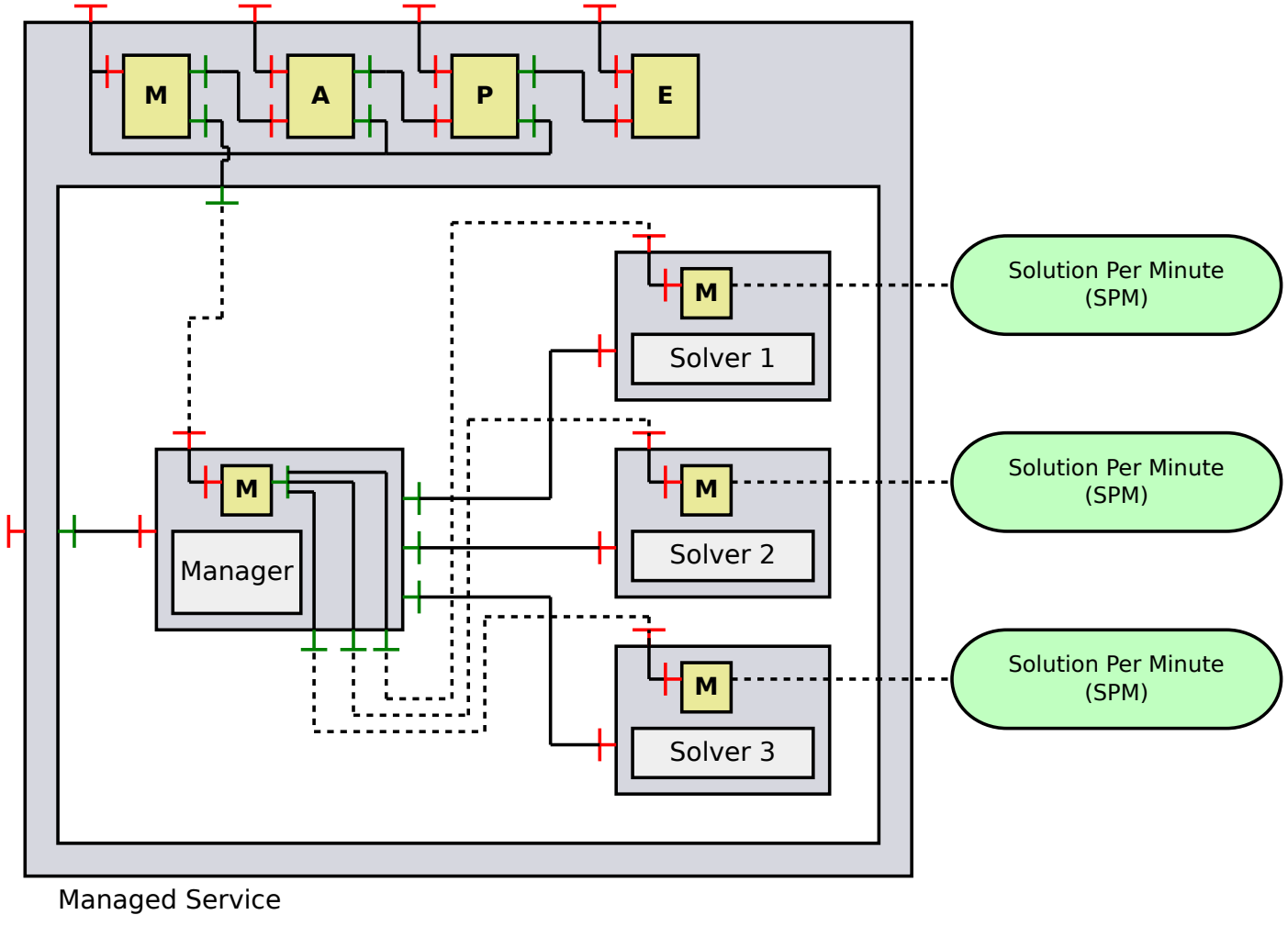


Use Case

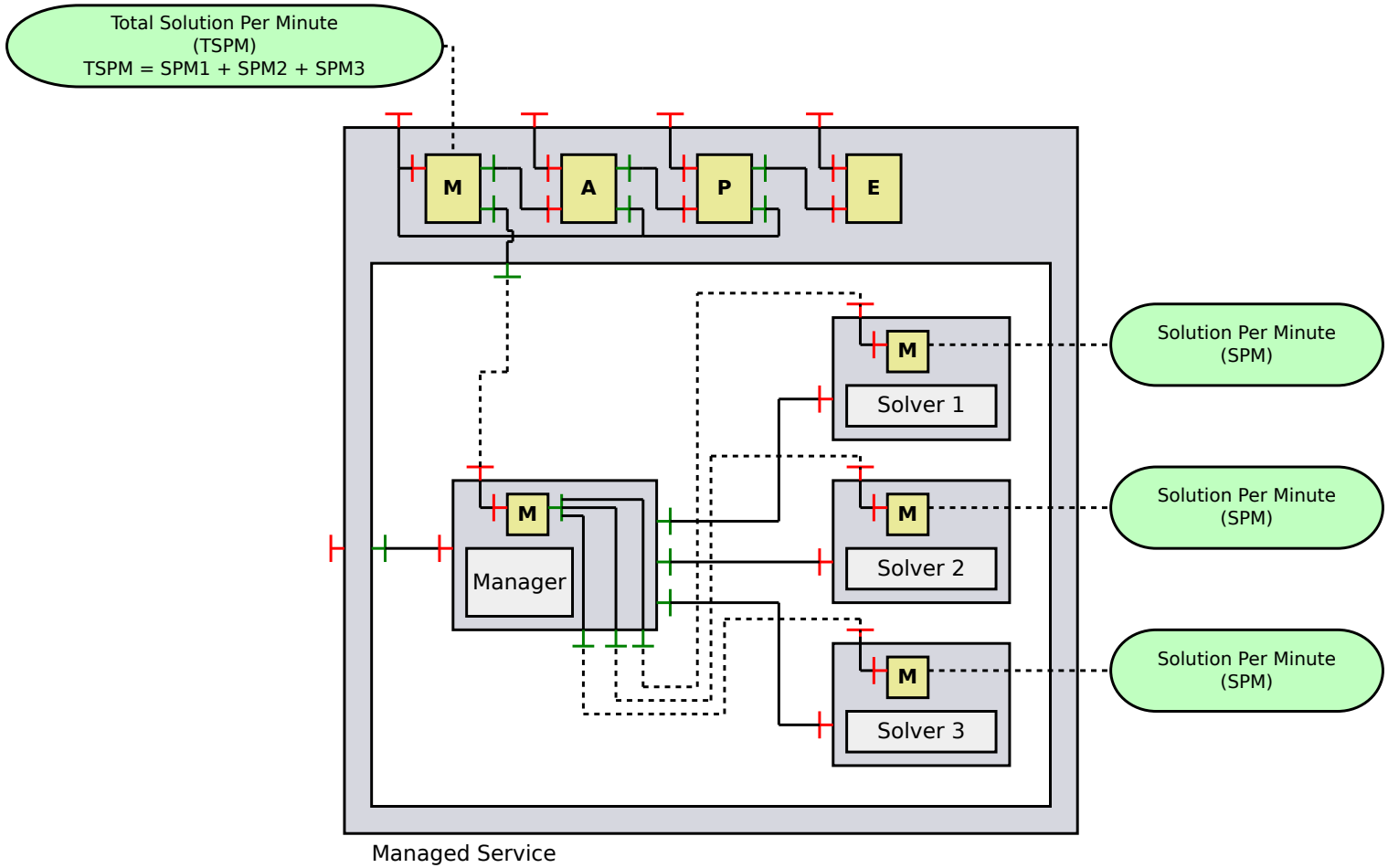


Managed Service

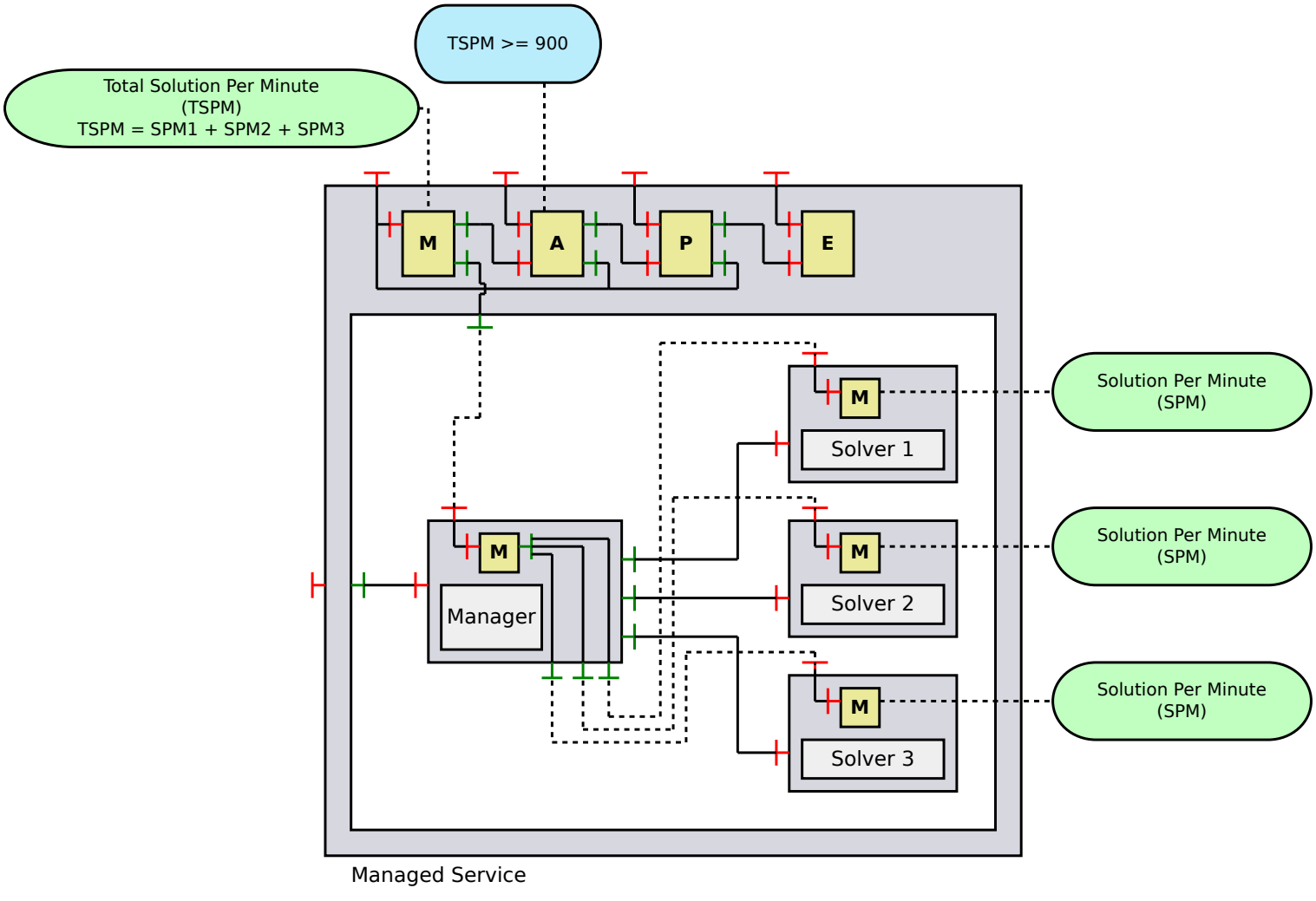
Use Case


M
A
P
E
M
Solver 1
M
Manager
M
Solver 2
M
Solver 3
Solution Per Minute (SPM)
Solution Per Minute (SPM)
Solution Per Minute (SPM)
Managed Service

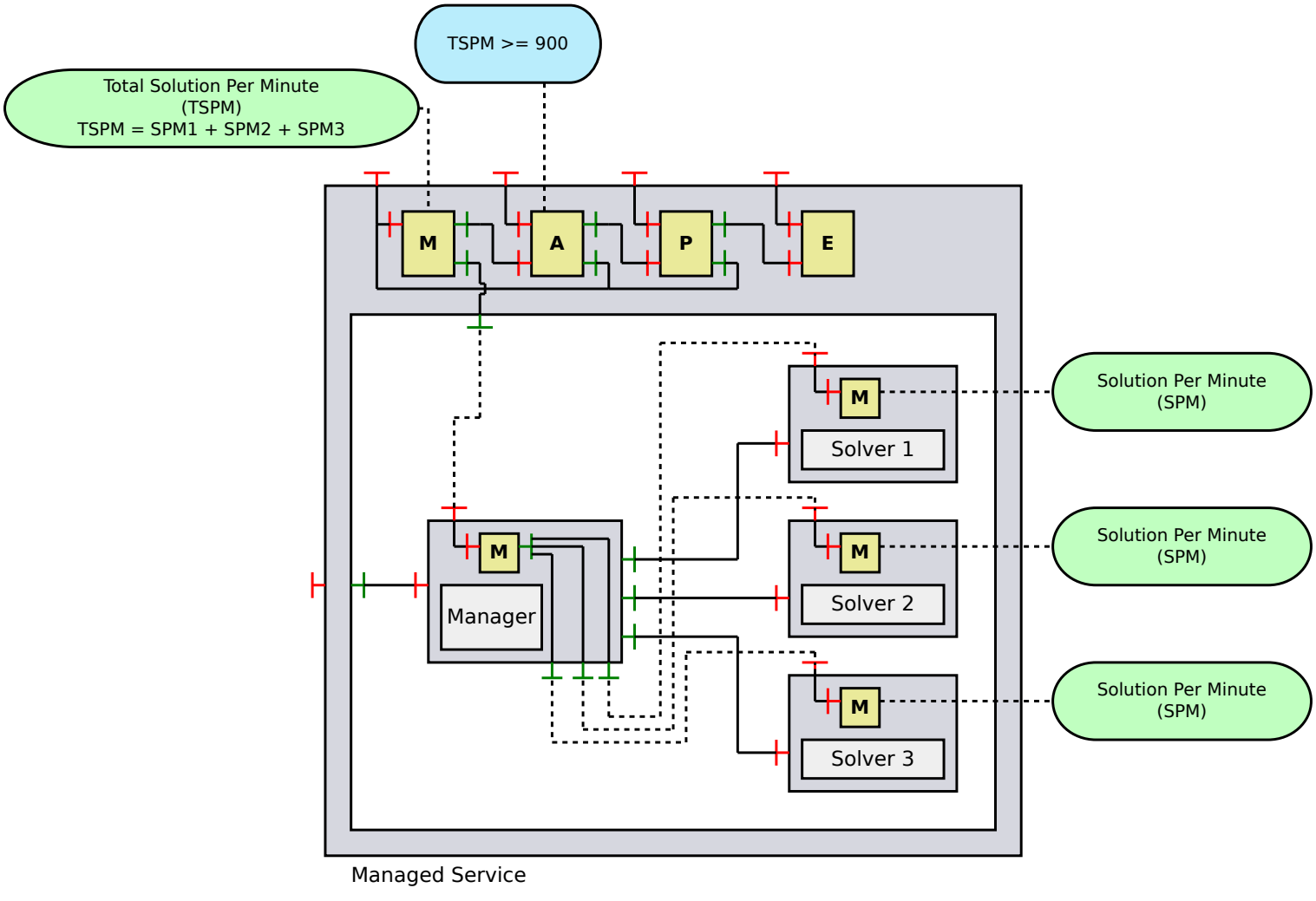
Use Case


Total Solution Per Minute
(TSPM)
TSPM = SPM1 + SPM2 + SPM3
M
A
P
E
M
Solver 1
Solution Per Minute
(SPM)
M
Manager
M
Solver 2
Solution Per Minute
(SPM)
M
Solver 3
Solution Per Minute
(SPM)
Managed Service

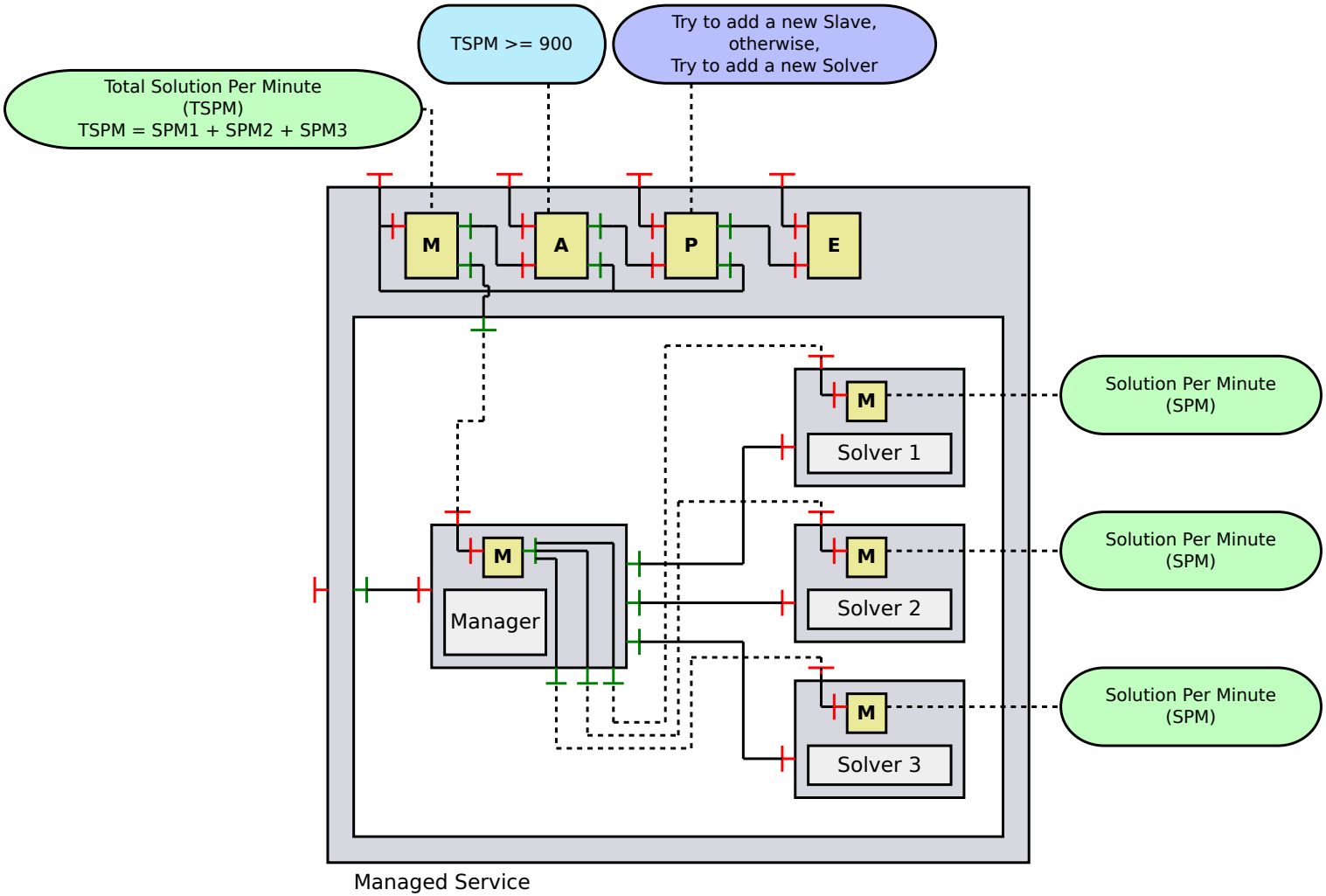
Use Case


TSPM >= 900
Total Solution Per Minute
(TSPM)
TSPM = SPM1 + SPM2 + SPM3
M
A
P
E
M
Solver 1
Solution Per Minute
(SPM)
M
Manager
M
Solver 2
Solution Per Minute
(SPM)
M
Solver 3
Solution Per Minute
(SPM)
Managed Service

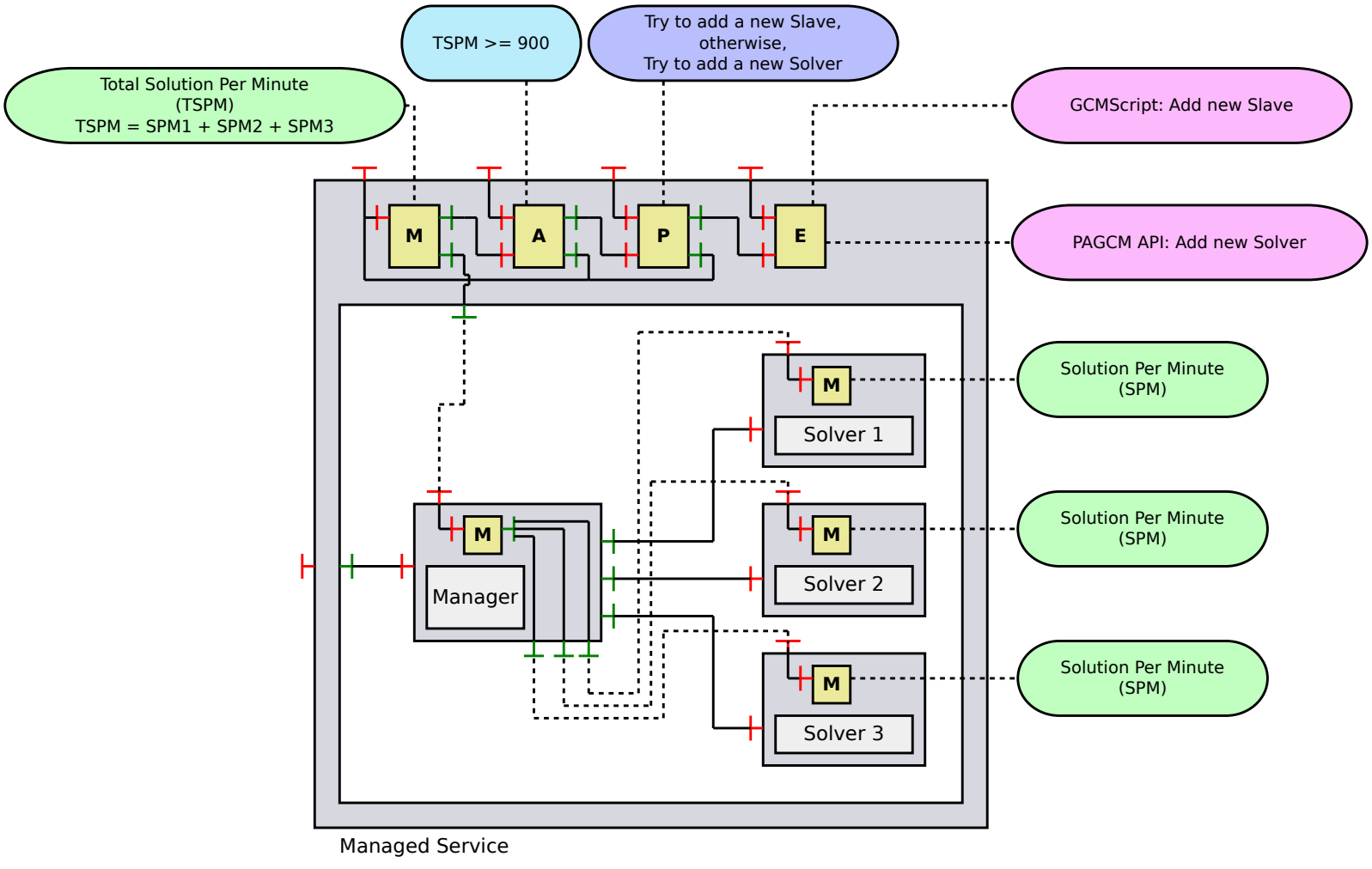
Use Case


TSPM >= 900
Total Solution Per Minute
(TSPM)
TSPM = SPM1 + SPM2 + SPM3
M
A
P
E
M
Solver 1
Solution Per Minute
(SPM)
M
Manager
M
Solver 2
Solution Per Minute
(SPM)
M
Solver 3
Solution Per Minute
(SPM)
Managed Service

Use Case


TSPM >= 900
Try to add a new Slave,
otherwise,
Try to add a new Solver
Total Solution Per Minute
(TSPM)
TSPM = SPM1 + SPM2 + SPM3
M
A
P
E
M
Solver 1
Solution Per Minute
(SPM)
M
Manager
M
Solver 2
Solution Per Minute
(SPM)
M
Solver 3
Solution Per Minute
(SPM)
Managed Service

Use Case

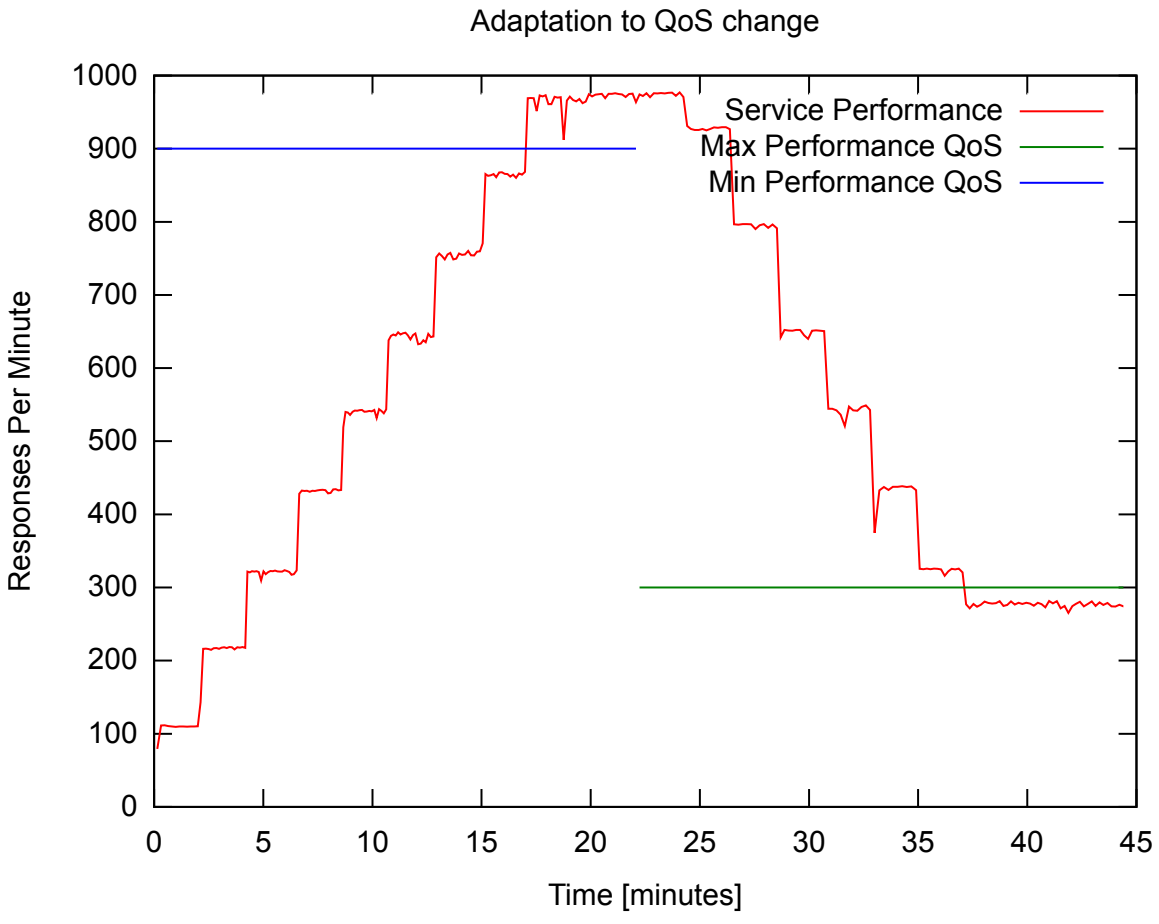

TSPM >= 900
Try to add a new Slave,
otherwise,
Try to add a new Solver
Total Solution Per Minute
(TSPM)
TSPM = SPM1 + SPM2 + SPM3
GCMScript: Add new Slave
M
A
P
E
PAGCM API: Add new Solver
M
Solver 1
Solution Per Minute
(SPM)
M
Manager
M
Solver 2
Solution Per Minute
(SPM)
M
Solver 3
Solution Per Minute
(SPM)
Managed Service

Use Case

Run parameters:

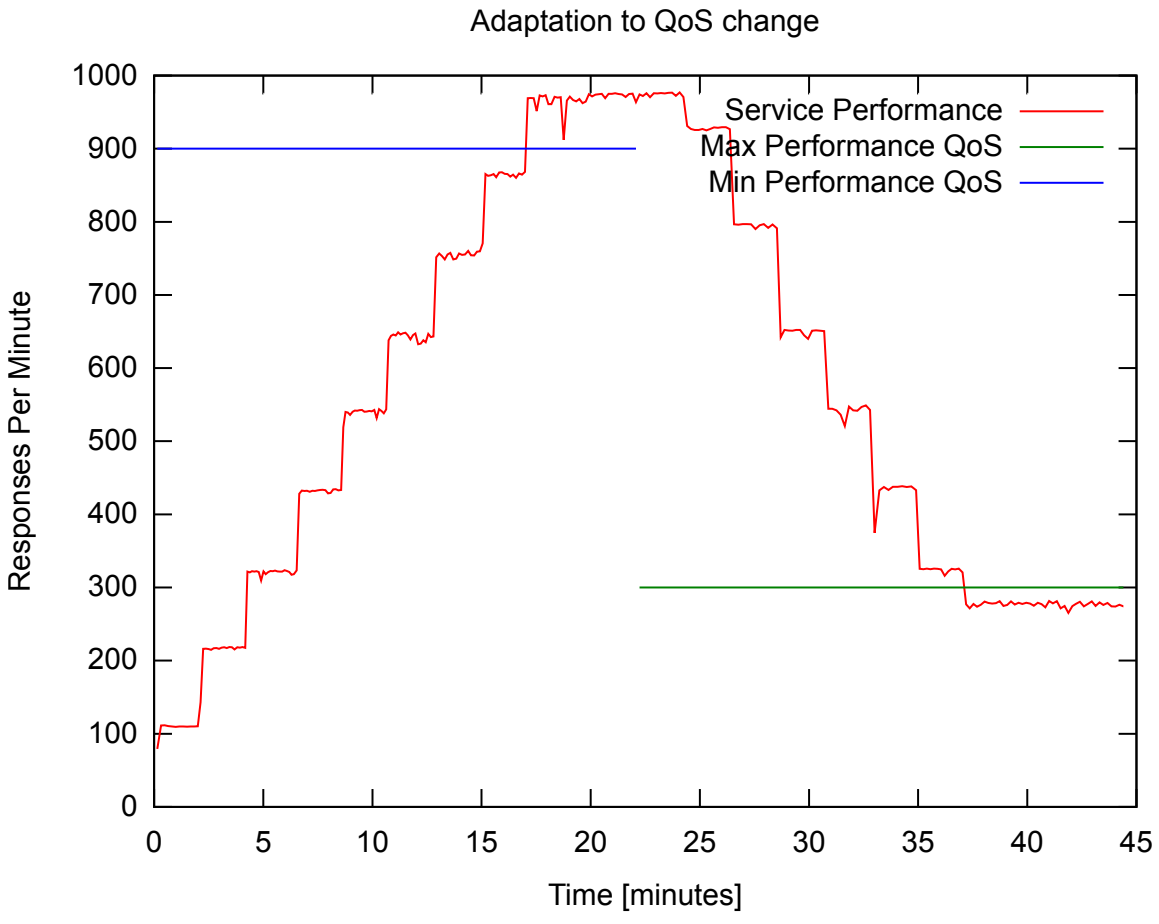
- Maximum number of Solvers $= 3$
- Maximum number of Slaves per Solver $= 3$
- Starting with 1 Solver and 1 Slave

Use Case: Results



- QoS desired change after minute 22 to "TSPM <= 300"

Use Case: Results



- QoS desired change after minute 22 to "TSPM <= 300"

Current Work

Current Work

- Allows instantiation of managed components using ADL descriptor file only.
- More and different examples
- Benchmarking