Module 7
Patterns – Part 4

BITS Pilani
Pilani|Dubai|Goa|Hyderabad

Harvinder S Jabbal
SSZG653 Software Architectures

innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Map-Reduce Pattern

Map-Reduce Pattern

innovate achieve lead

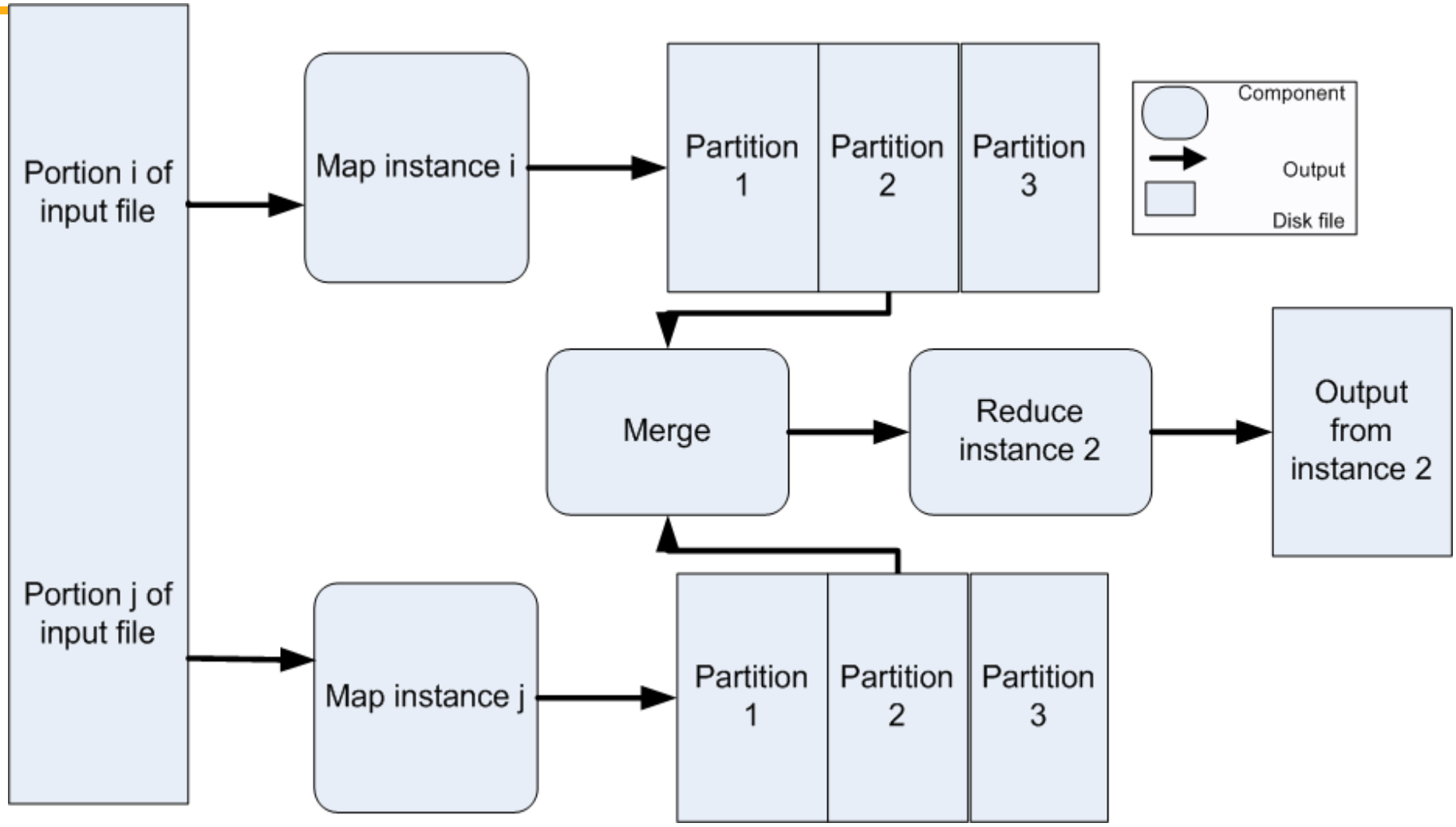
Context: Businesses have a pressing need to quickly analyze enormous volumes of data they generate or access, at petabyte scale.

Problem: For many applications with ultra-large data sets, sorting the data and then analyzing the grouped data is sufficient. The problem the map-reduce pattern solves is to efficiently perform a distributed and parallel sort of a large data set and provide a simple means for the programmer to specify the analysis to be done.

Solution: The map-reduce pattern requires three parts:

- A specialized infrastructure takes care of allocating software to the hardware nodes in a massively parallel computing environment and handles sorting the data as needed.
- A programmer specified component called the map which filters the data to retrieve those items to be combined.
- A programmer specified component called reduce which combines the results of the map

Map-Reduce Example



Map-Reduce Solution - 1

Overview: The map-reduce pattern provides a framework for analyzing a large distributed set of data that will execute in parallel, on a set of processors. This parallelization allows for low latency and high availability. The map performs the extract and transform portions of the analysis and the reduce performs the loading of the results.

Elements:

- Map is a function with multiple instances deployed across multiple processors that performs the extract and transformation portions of the analysis.
- Reduce is a function that may be deployed as a single instance or as multiple instances across processors to perform the load portion of extract-transform-load.
- The infrastructure is the framework responsible for deploying map and reduce instances, shepherding the data between them, and detecting and recovering from failure.

Map-Reduce Solution - 2

Relations:

- Deploy on is the relation between an instance of a map or reduce function and the processor onto which it is installed.
- Instantiate, monitor, and control is the relation between the infrastructure and the instances of map and reduce.

Constraints:

- The data to be analyzed must exist as a set of files.
- Map functions are stateless and do not communicate with each other.
- The only communication between map reduce instances is the data emitted from the map instances as <key, value> pairs.

Weaknesses:

- If you do not have large data sets, the overhead of map-reduce is not justified.
- If you cannot divide your data set into similar sized subsets, the advantages of parallelism are lost.
- Operations that require multiple reduces are complex to orchestrate.

innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Multi-Tier Pattern

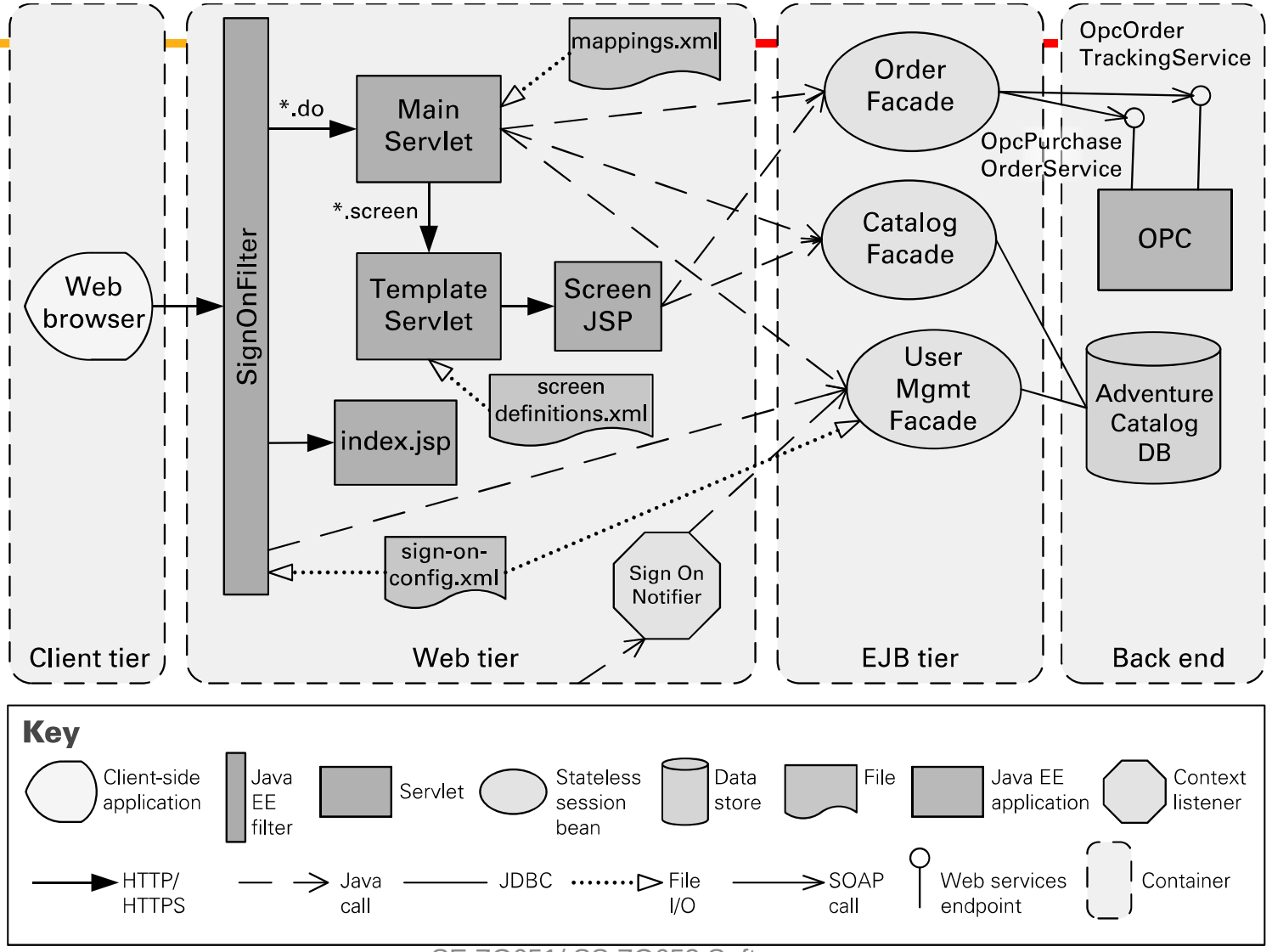
Multi-Tier Pattern

Context: In a distributed deployment, there is often a need to distribute a system's infrastructure into distinct subsets.

Problem: How can we split the system into a number of computationally independent execution structures—groups of software and hardware—connected by some communications media?

Solution: The execution structures of many systems are organized as a set of logical groupings of components. Each grouping is termed a tier.

Multi-Tier Example



Multi-Tier Solution

Overview: The execution structures of many systems are organized as a set of logical groupings of components. Each grouping is termed a *tier*.

Elements:

- *Tier,* which is a logical grouping of software components.

Relations:

- *Is part of,* to group components into tiers.
- *Communicates with,* to show how tiers and the components they contain interact with each other.
- *Allocated to,* in the case that tiers map to computing platforms.

Constraints: A software component belongs to exactly one tier.

Weaknesses: Substantial up-front cost and complexity.

innovate achieve lead

BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Tactics and Patterns

Relationships Between Tactics and Patterns

innovate achieve lead

Patterns are built from tactics; if a pattern is a molecule, a tactic is an atom.

MVC, for example utilizes the tactics:

- Increase semantic coherence
- Encapsulation
- Use an intermediary
- Use run time binding

Tactics Augment Patterns

Patterns solve a specific problem but are neutral or have weaknesses with respect to other qualities.

Consider the broker pattern

- May have performance bottlenecks
- May have a single point of failure

Using tactics such as

- Increase resources will help performance
- Maintain multiple copies will help availability

Tactics and Interactions

Each tactic/pattern has pluses (its reason for being) and minuses – side effects.

Use of tactics can help alleviate the minuses.

But nothing is free…

Tactics and Interactions - 2

A common tactic for detecting faults is Ping/Echo.

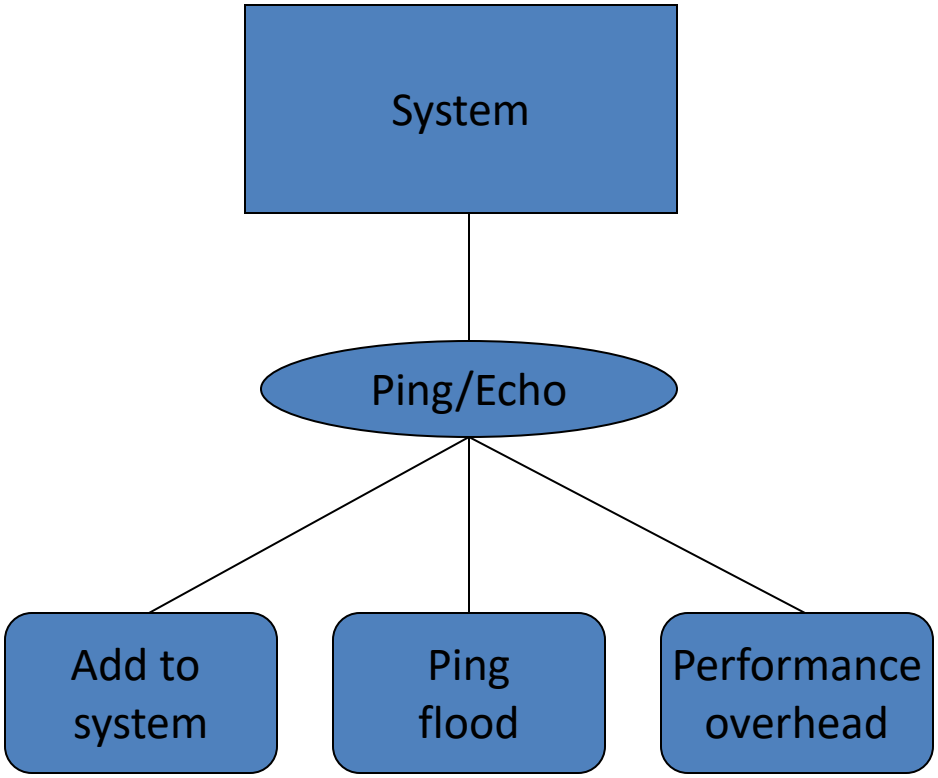
Common side-effects of Ping/Echo are:

security: how to prevent a ping flood attack?

performance: how to ensure that the performance overhead of ping/echo is small?

modifiability: how to add ping/echo to the existing architecture?

Tactics and Interactions - 3



Tactics and Interactions - 4

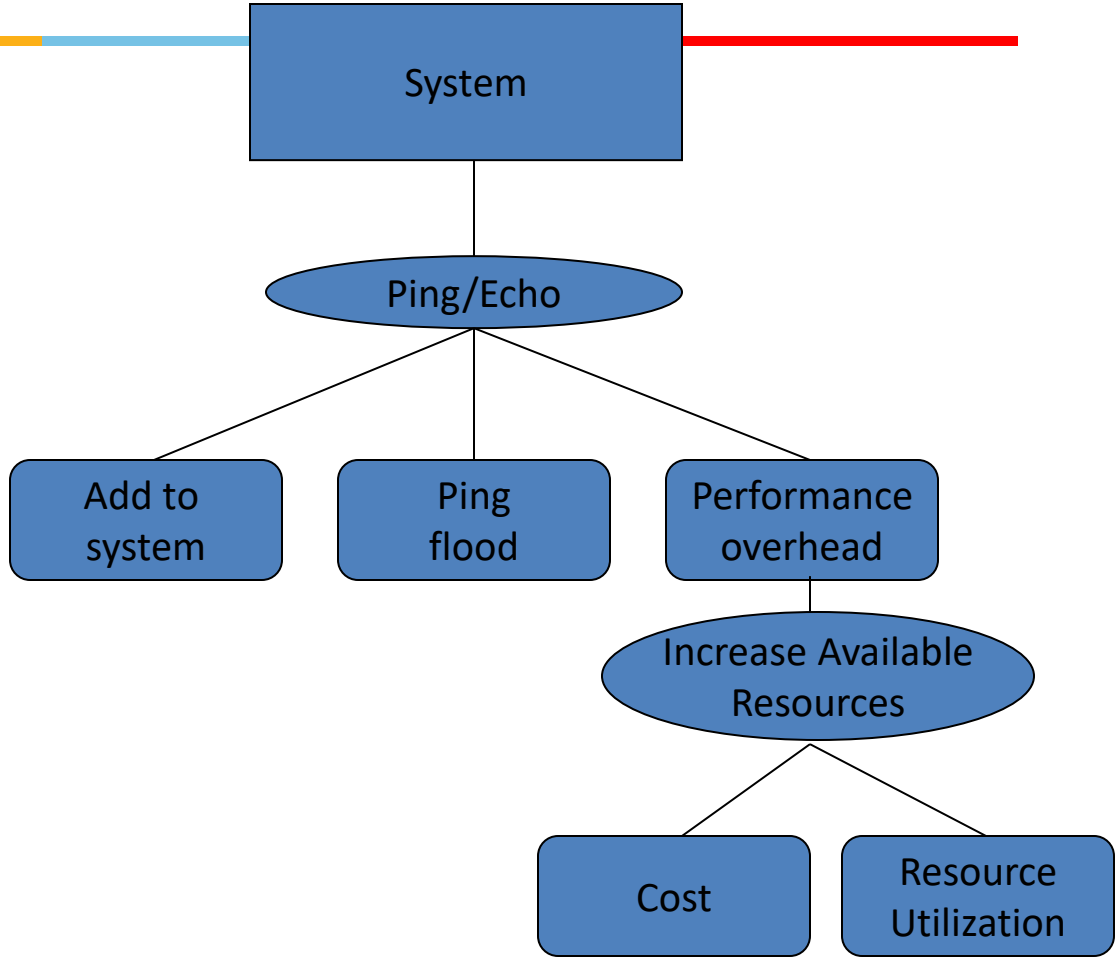
A tactic to address the performance side-effect is “Increase Available Resources”.

Common side effects of Increase Available Resources are:

cost: increased resources cost more

performance: how to utilize the increase resources efficiently?

Tactics and Interactions - 5



Tactics and Interactions - 6

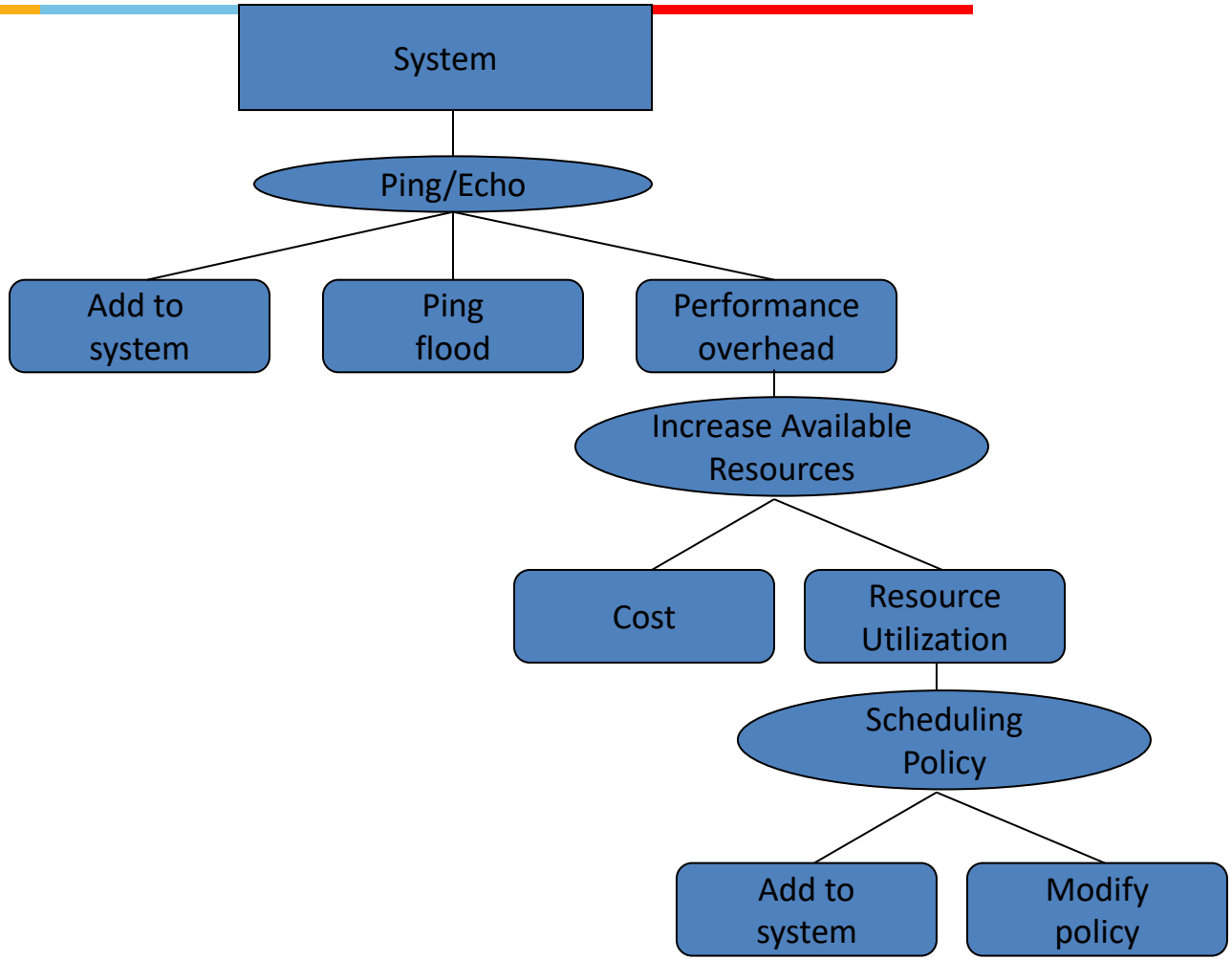
A tactic to address the efficient use of resources side-effect is “Scheduling Policy”.

Common side effects of Scheduling Policy are:

modifiability: how to add the scheduling policy to the existing architecture

modifiability: how to change the scheduling policy in the future?

Tactics and Interactions - 7



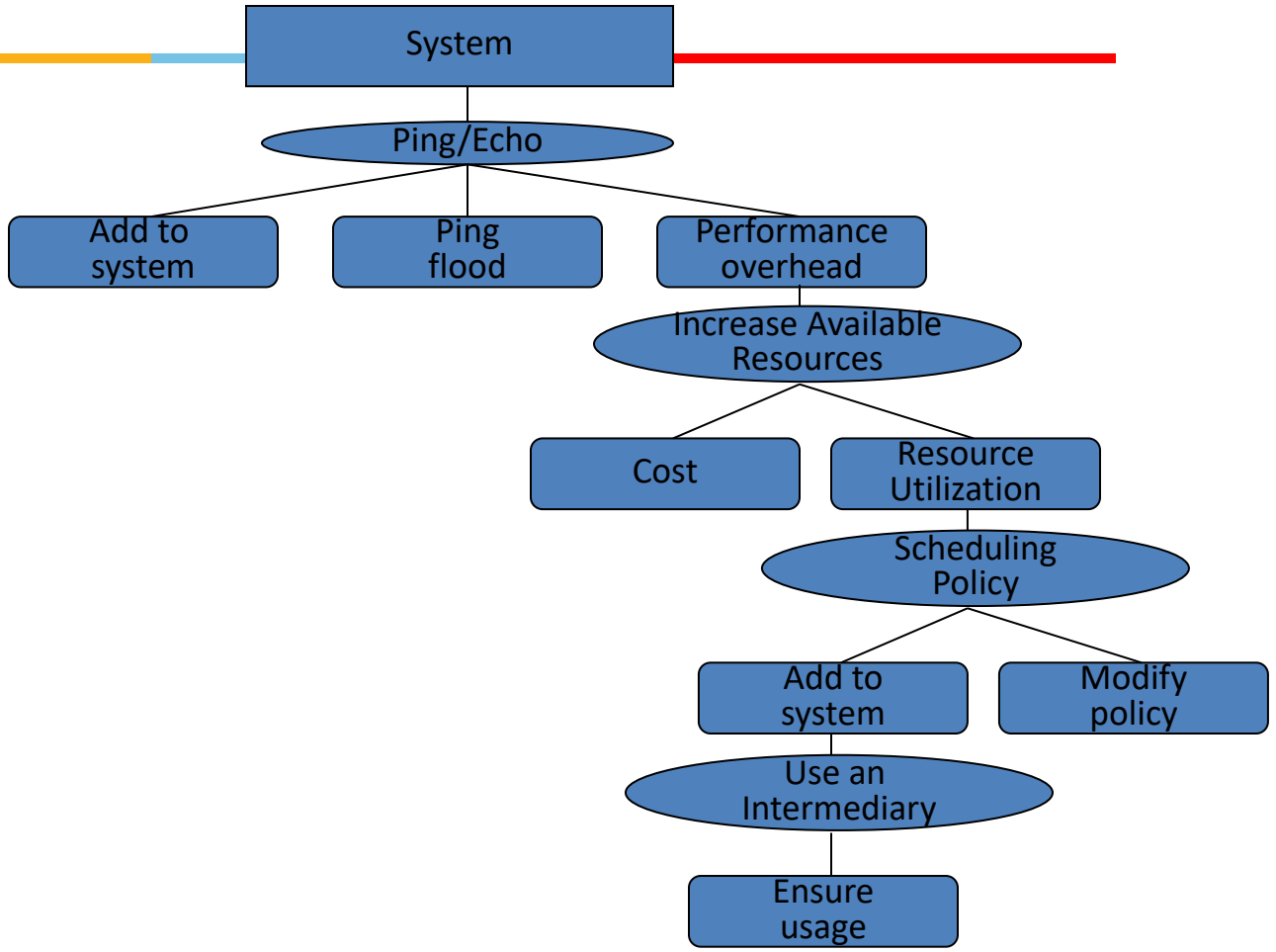
Tactics and Interactions - 8

A tactic to address the addition of the scheduler to the system is “Use an Intermediary”.

Common side effects of Use an Intermediary are:

modifiability: how to ensure that all communication passes through the intermediary?

Tactics and Interactions - 9



Tactics and Interactions – 10.

innovate achieve lead

A tactic to address the concern that all communication passes through the intermediary is “Restrict Communication Paths”.

Common side effects of Restrict Communication Paths are:

performance: how to ensure that the performance overhead of the intermediary are not excessive?

Note: this design problem has now become recursive!

How Does This Process End?

Each use of tactic introduces new concerns.

Each new concern causes new tactics to be added.

Are we in an infinite progression?

No. Eventually the side-effects of each tactic become small enough to ignore.

Summary

An architectural pattern

- is a package of design decisions that is found repeatedly in practice,
- has known properties that permit reuse, and
- describes a *class* of architectures.

Tactics are simpler than patterns

Patterns are underspecified with respect to real systems so they have to be augmented with tactics.

- Augmentation ends when requirements for a specific system are satisfied.