Architecture Patterns

Rylan Perumal
slides adapted from
Prof. van Zyl, Ivan
Marsic and and Mark
Richards

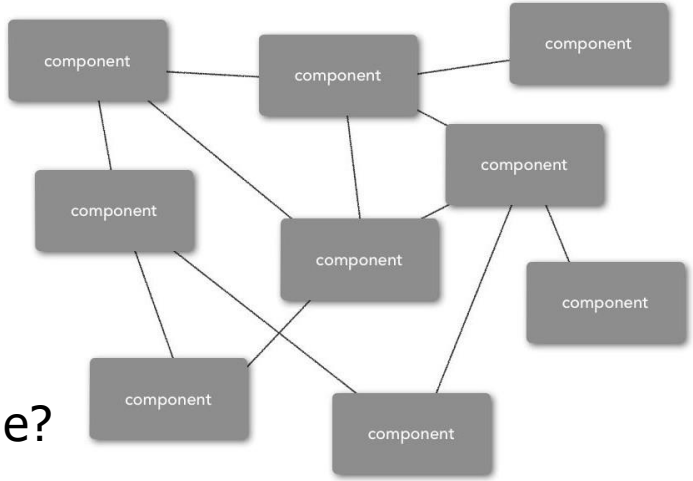
“Humans are pattern-seeking story-telling animals, and we are quite adept at telling stories about patterns, whether they exist or not.” :- Michael Shermer

Topics/Recap:

- Introduction
- Layered Architecture Pattern
- Event-driven Architecture Pattern
- Microkernel Architecture Pattern
- Service Oriented Architecture Pattern
- Space-Based Architecture Pattern

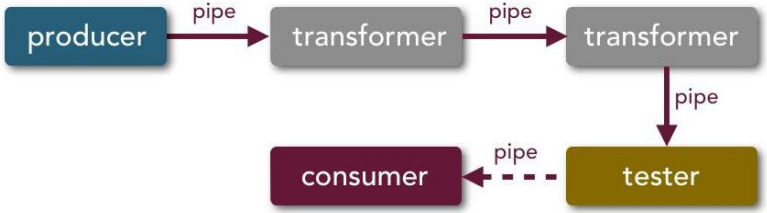
How do we even Architecture?

- how are components classified?
- how do components interact?
- does the architecture scale?
- how responsive is the architecture?
- is there a logical flow to the components?
- what are the deployment characteristics?
- how does the architecture respond to change?
- is the architecture extensible and if so how?
- how maintainable is the architecture?

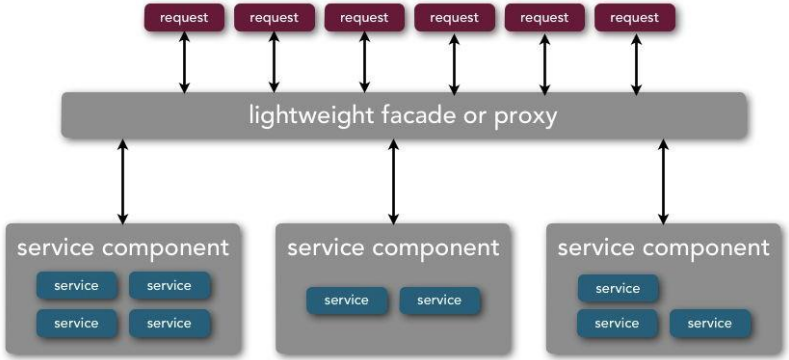


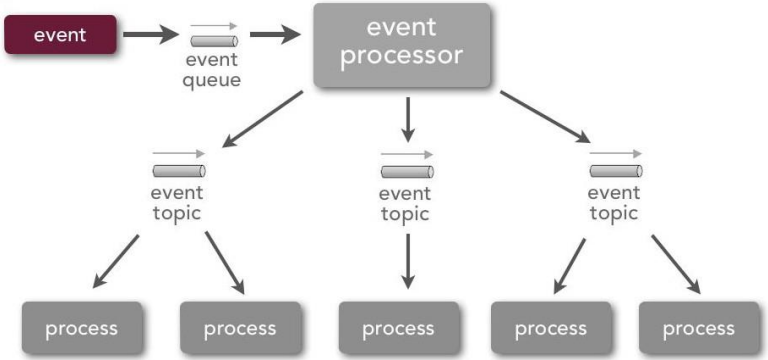
Architecture Patterns: Part 1

Pipe and Filter



Service Oriented

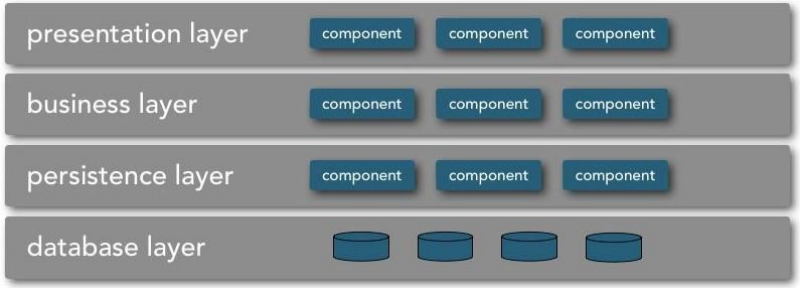




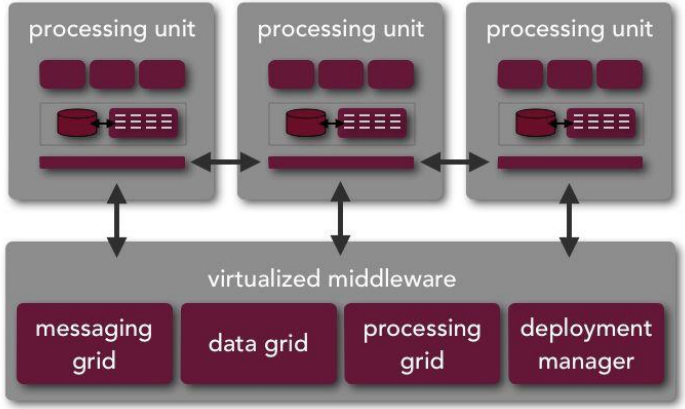
Event-Driven

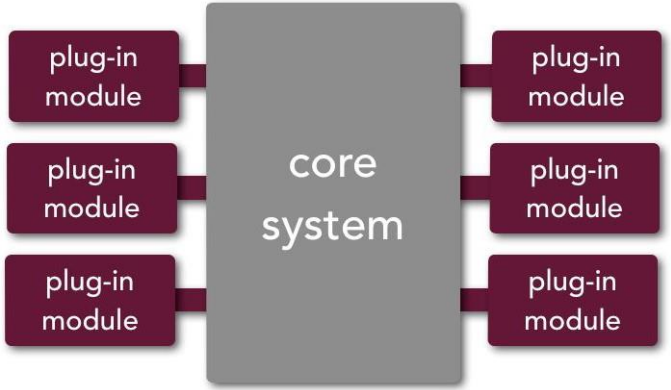
Architecture Patterns: Part 2

Layered



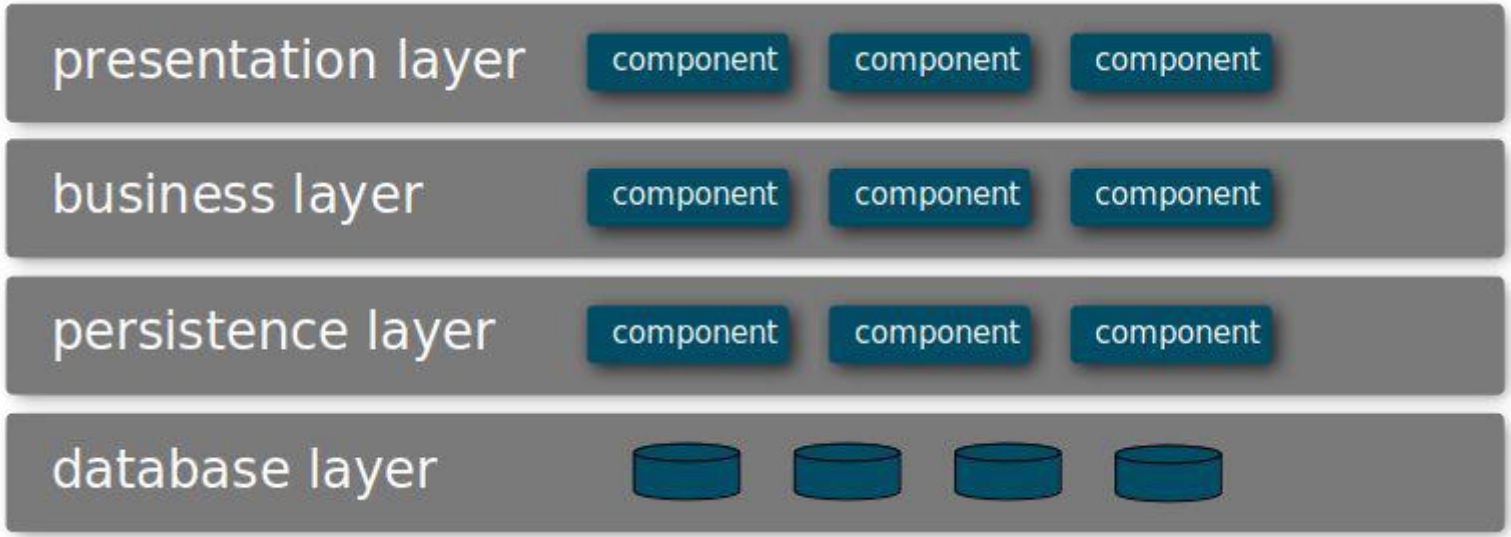
Space-Based



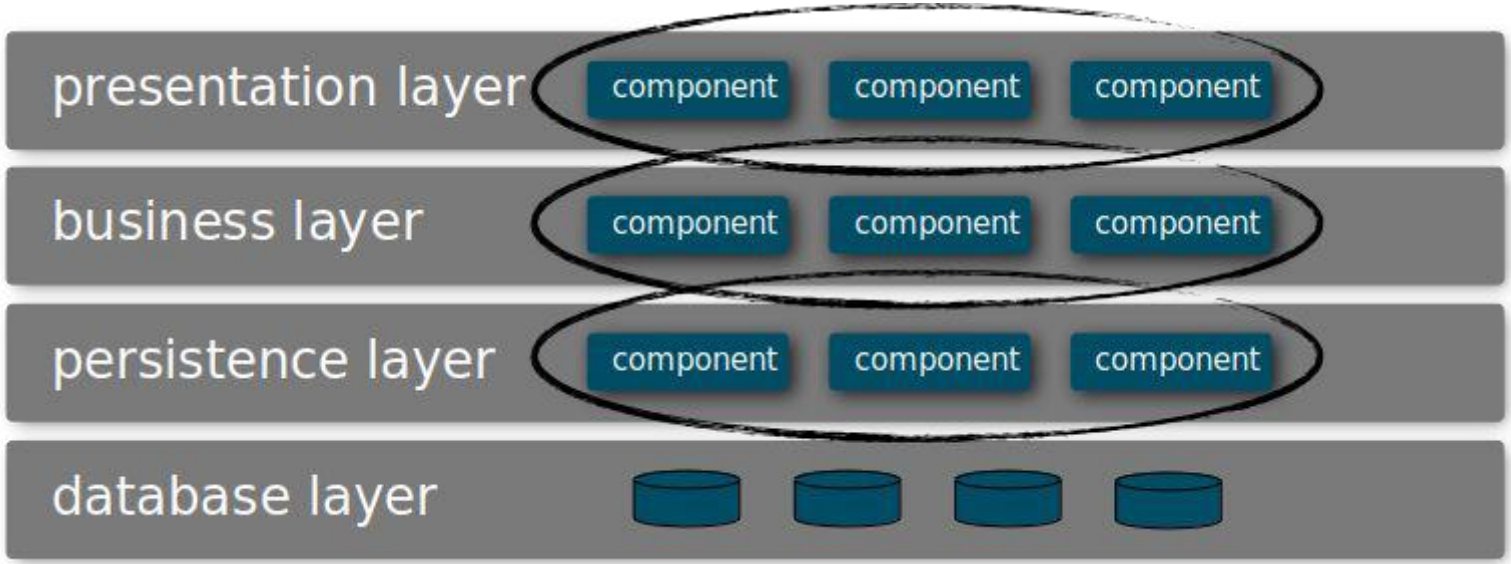


Microkernel

Layered Architecture

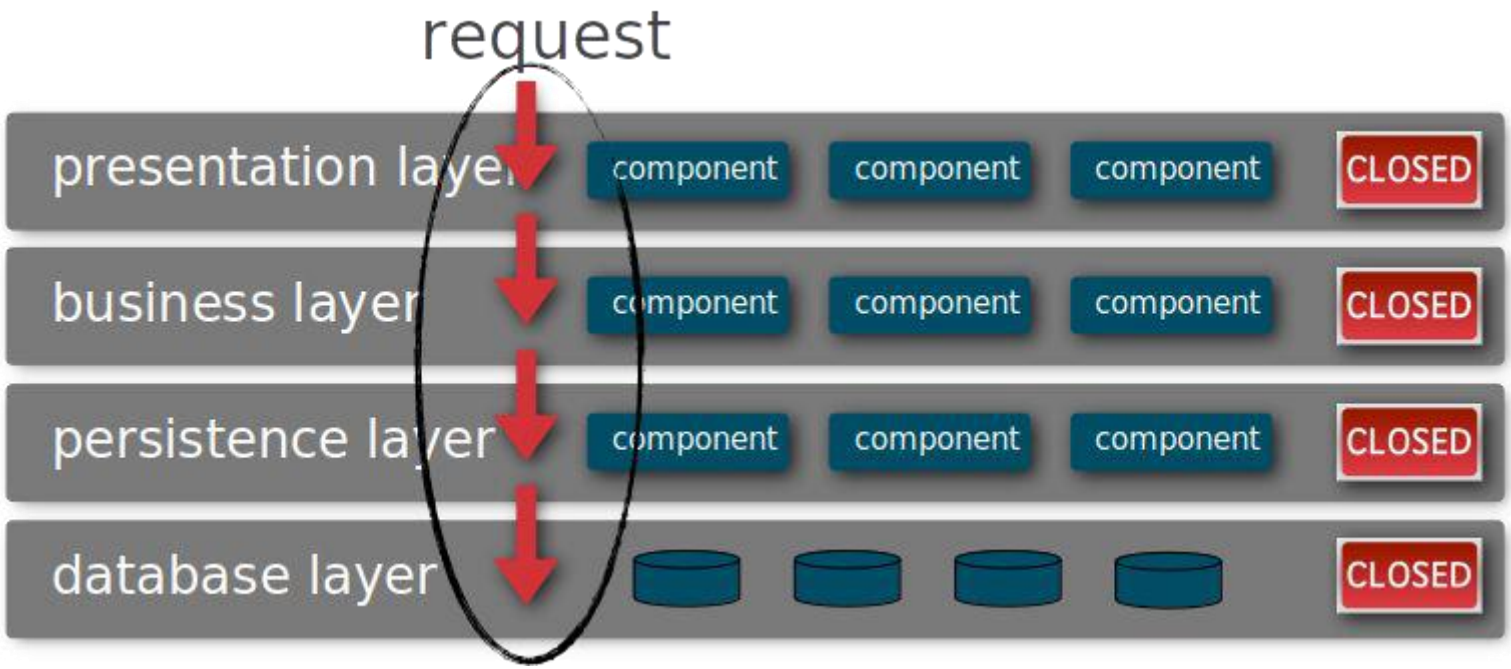


Layered: Separation of Concerns



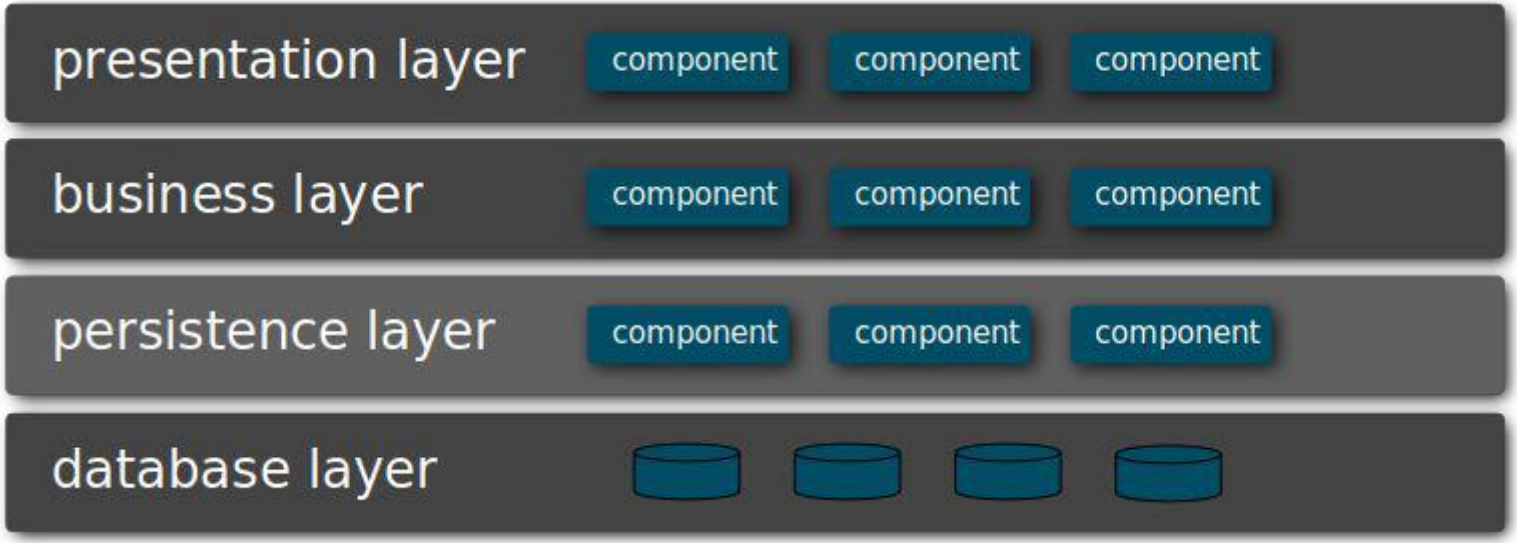
Components within a specific layer deal only with logic that pertains to that layer

Layered Architecture: Closed



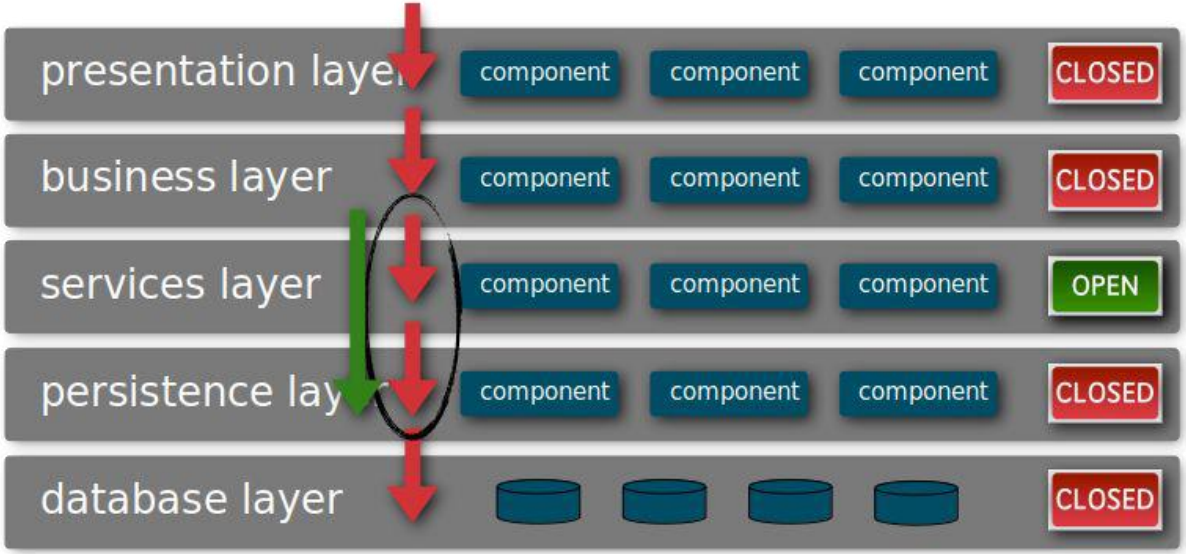
Closed layer means that a request moves from layer to layer

Layered: Isolation



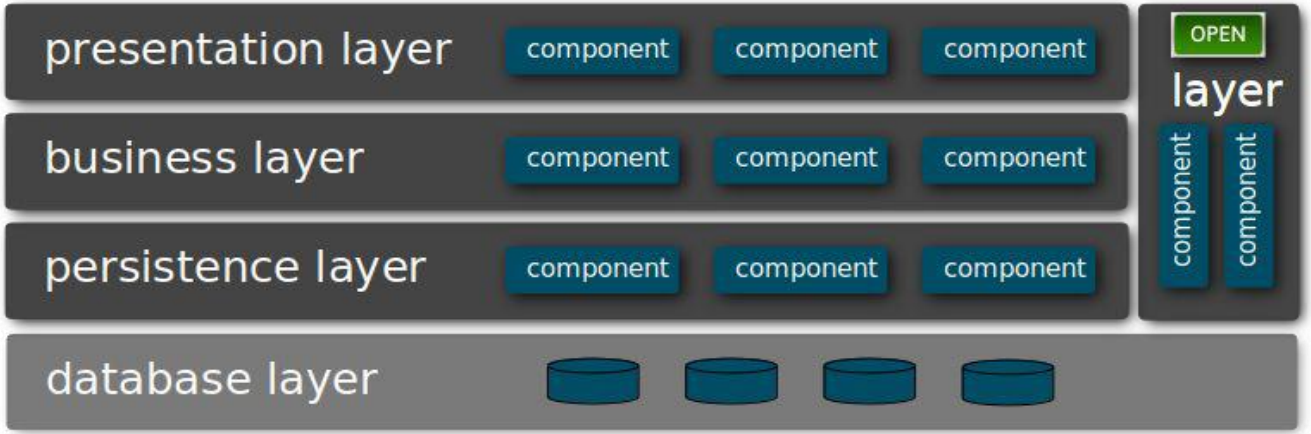
Changes made in one layer of the architecture generally don’t impact or affect components in other layers

Layered: Hybrids and Variants



Open layer means that a request can move past the open layer to the next layer

Layered: Hybrids and Variants

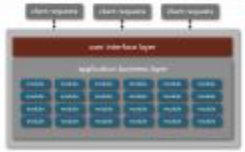


Specific components can be defined to pass through all the layers, we can separate these components into a separate layer.

Layered: Considerations

good general purpose architecture and a good starting point for most systems

watch out for the architecture sinkhole anti-pattern



tends to lend itself towards monolithic applications

Layered: Analysis

overall agility

deployment

testability

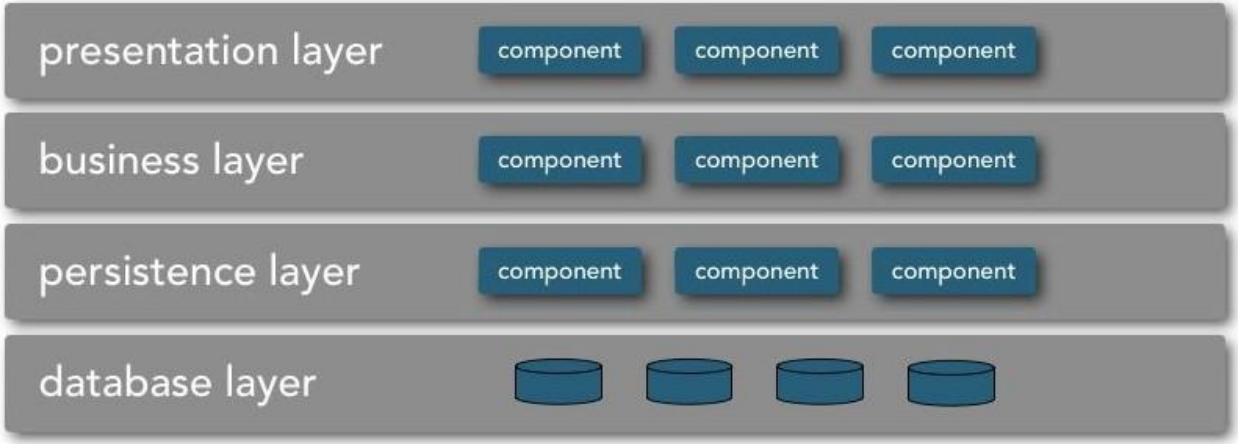
performance

scalability

development

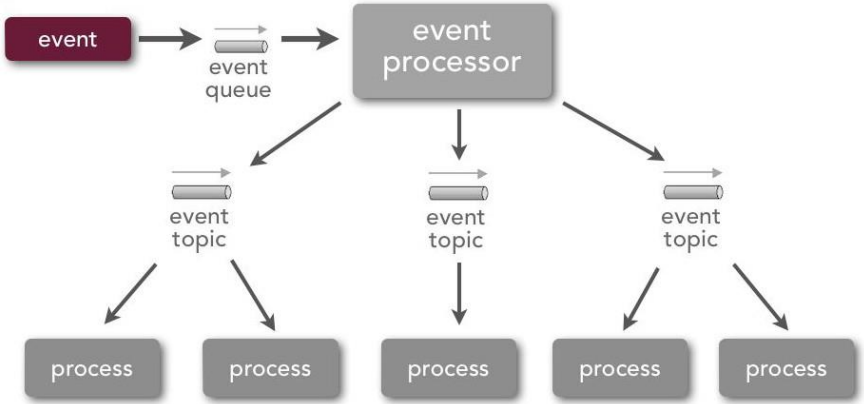
complexity

loose coupling

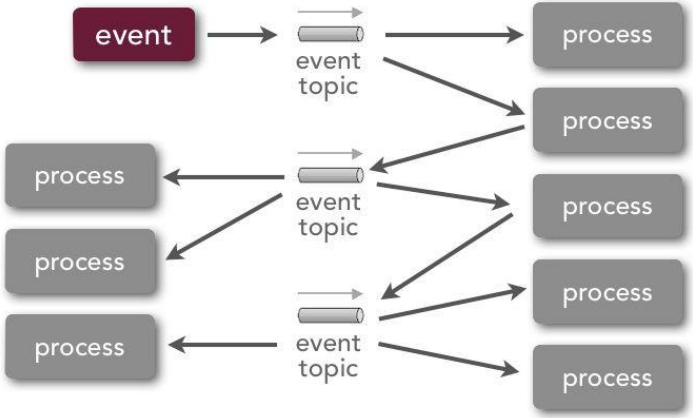


Event-driven Architecture

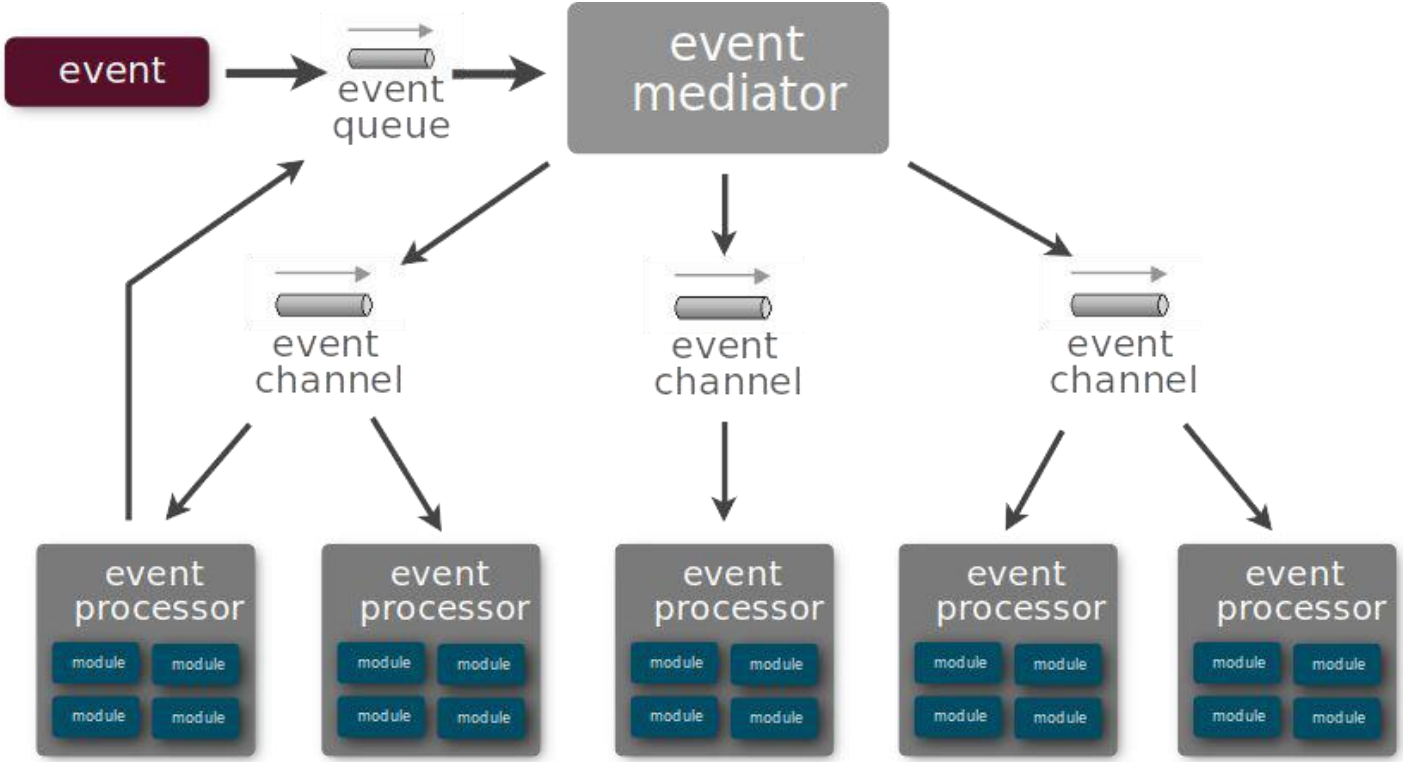
Mediator Topology



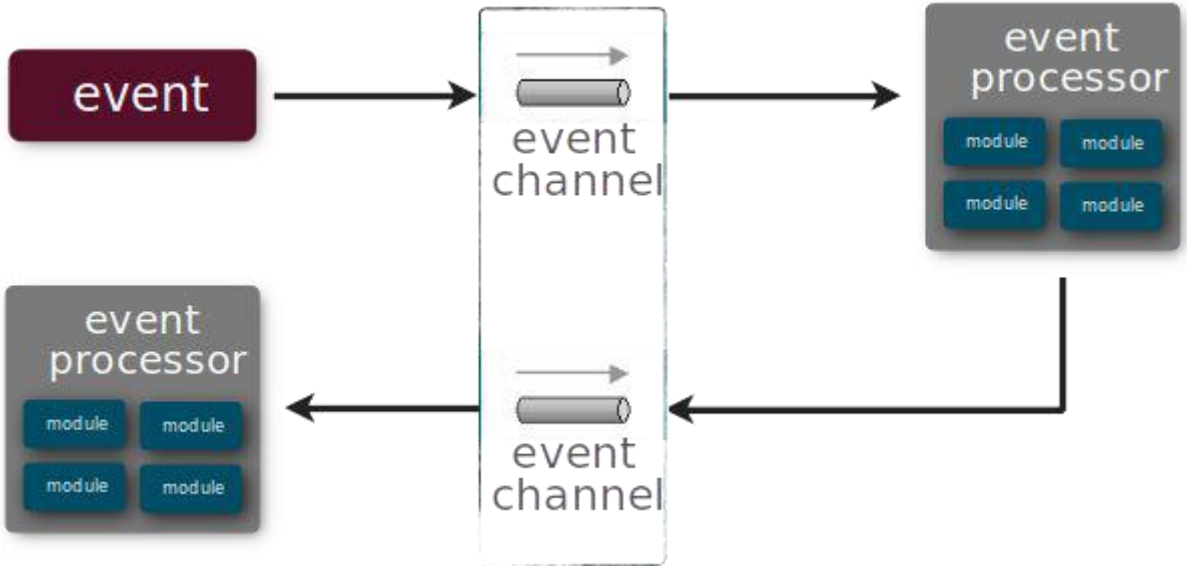
Broker Topology



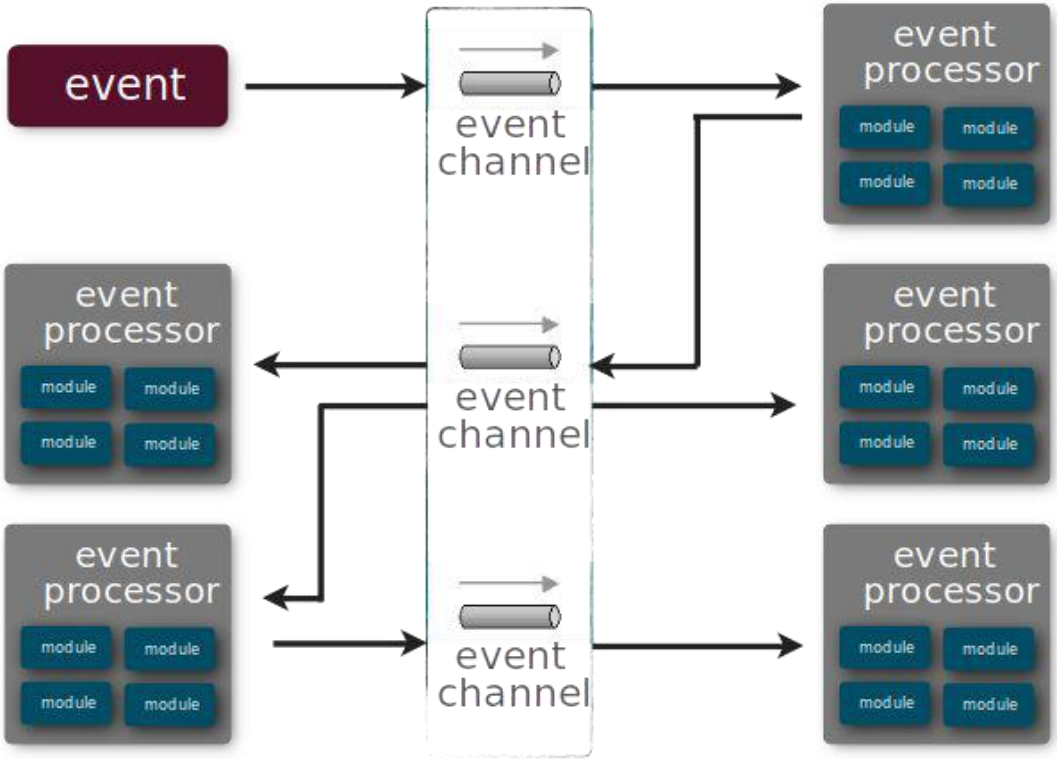
Event-driven: Mediator Topology



Event-driven: Broker Topology



Event-driven: Broker Topology



Event-driven: Considerations

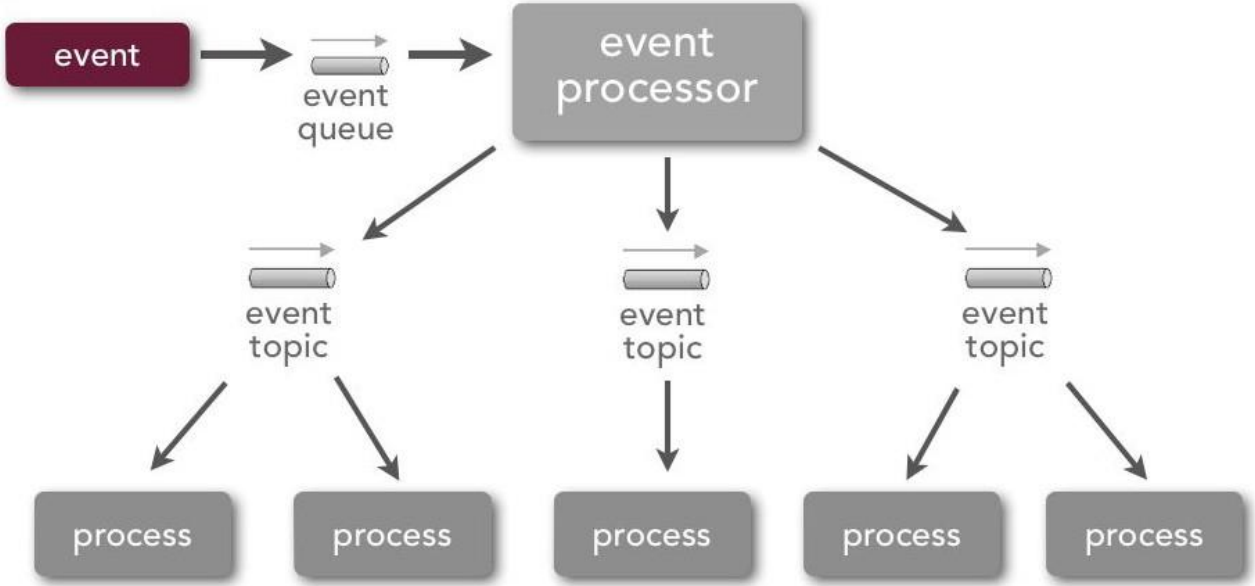
contract creation, maintenance, and versioning can be difficult

must address remote process availability or unresponsiveness

reconnection logic on server restart or failure must be addressed

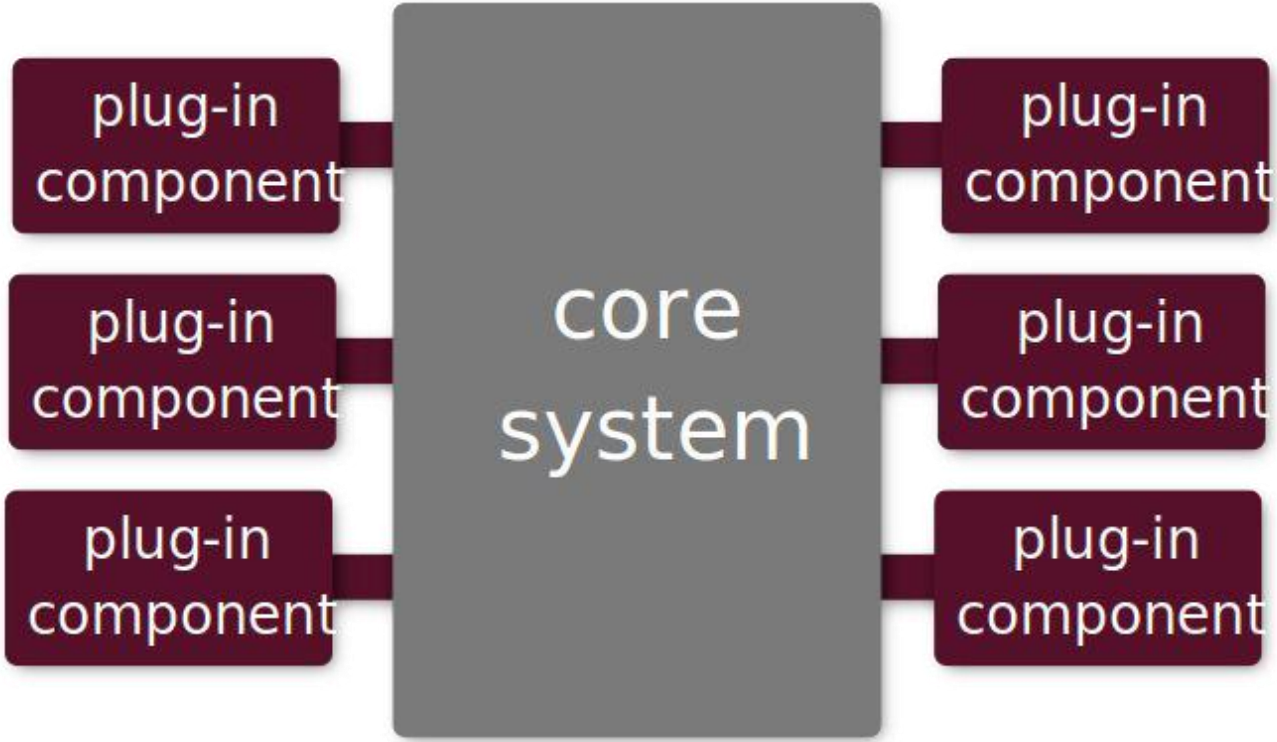
Event-driven: Analysis

- overall agility
- deployment
- testability
- performance
- scalability
- development
- complexity
- loose coupling



Microkernel Architecture

(a.k.a. plug-in architecture pattern)



Microkernel: Architectural Components

core system

minimal functionality to run system

general business rules and logic

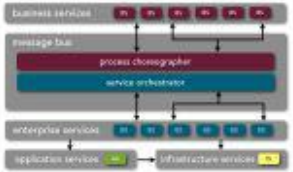
no custom processing

plug-in module

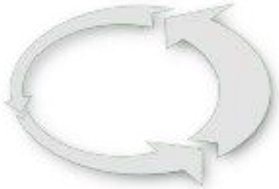
standalone independent module

specific additional rules or logic

Microkernel: Considerations



can be embedded or used as part of another pattern



great support for evolutionary design and incremental development

great pattern for product-based applications

Microkernel: Analysis

overall agility

deployment

testability

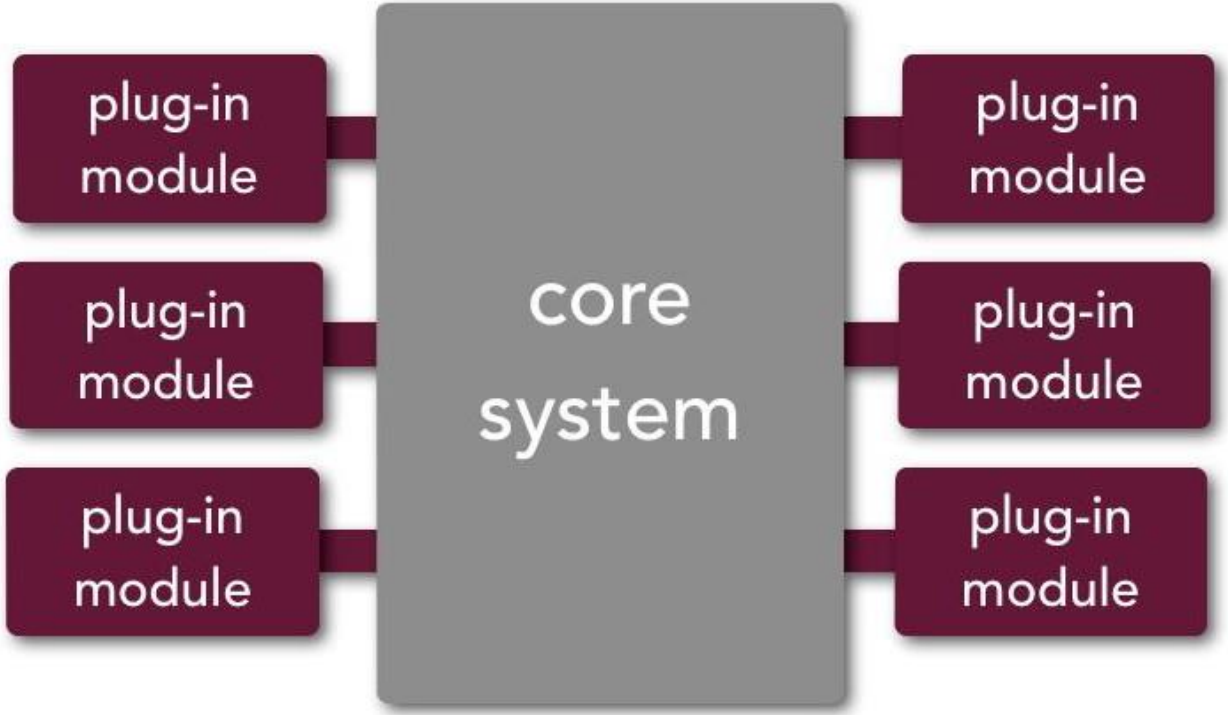
performance

scalability

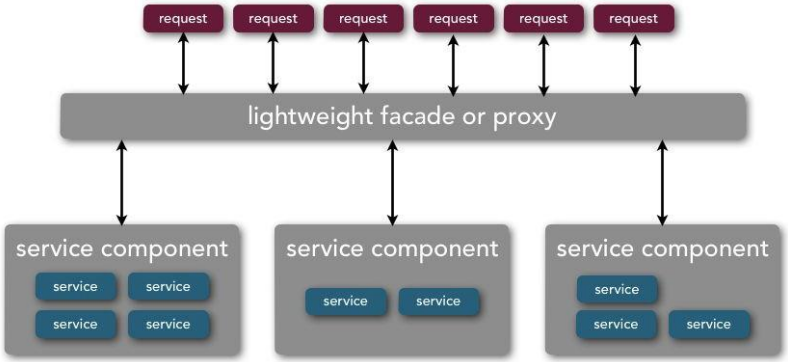
development

complexity

loose coupling



Service Oriented/Microservice Architecture



Space Based Architecture