
SPRING FRAMEWORK
GURU

The Reactive Manifesto

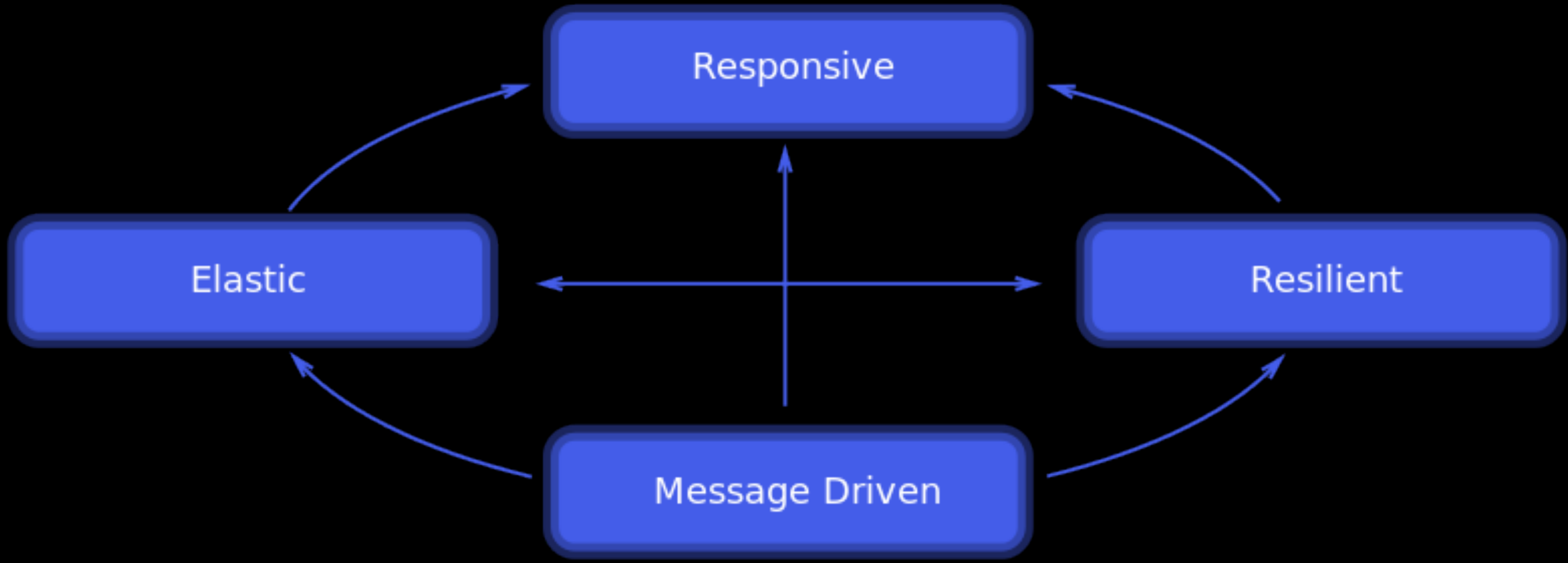
Reactive Manifesto - History

- Originally published in 2013.
- Available at www.reactivemanifesto.org
- Authors: Jonas Bonér (Akka founder), Dave Farley, Roland Kuhn, and Martin Thompson
- The term “Reactive” is getting a bit overloaded in the IT community.

“Reactive”

- Reactive Systems - Architecture and Design
  - ie Cloud Native
- Reactive Programming
  - Generally Event Based
- Functional Reactive Programming (FRP)
  - Often confused with Reactive Programming

Reactive Manifesto



Reactive Manifesto - Responsive

- The system responds in a timely manner.
- Responsiveness is the cornerstone of usability and utility.
- Responsiveness also means problems may be detected quickly and dealt with effectively.
- Responsive systems provide rapid and consistent response times.
- Consistent behavior simplifies error handling, builds end user confidence, and encourages further interaction.

Reactive Manifesto - Resilient

- System stays responsive in the face of failure.
- Resilience is achieved by replication, containment, isolation, and delegation.
- Failures are contained within each component.
- Parts of the system can fail, without compromising the system as a whole.
- Recovery of each component is delegated to another.
- High-availability is ensured by replication where necessary.

Reactive Manifesto - Elastic

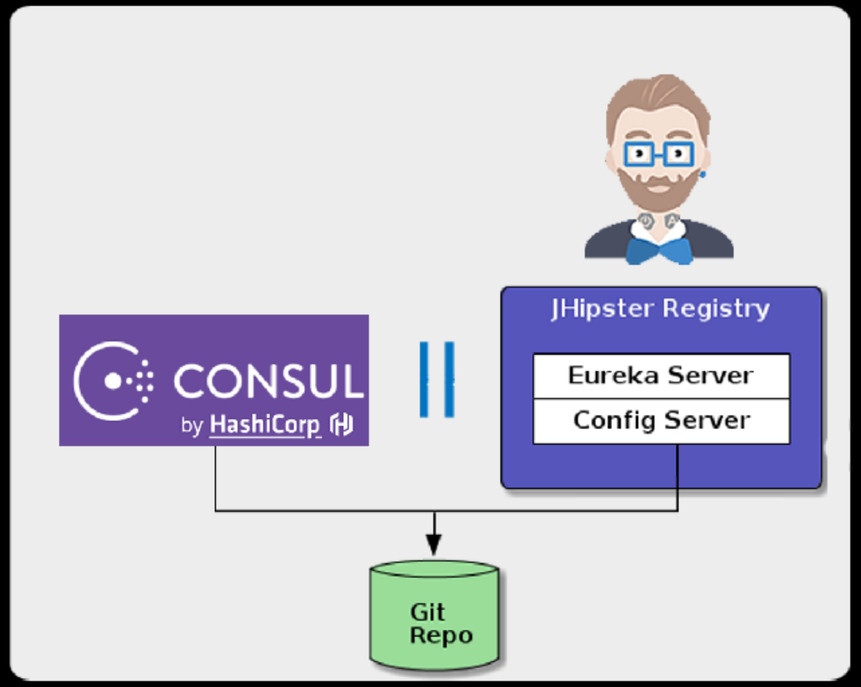
- The system stays responsive under varying workload.
- Reactive Systems can react to changes in the input rate by increasing or decreasing resources allocated to service inputs.
- Reactive Systems achieve elasticity in a cost effective way on commodity hardware and software platforms.

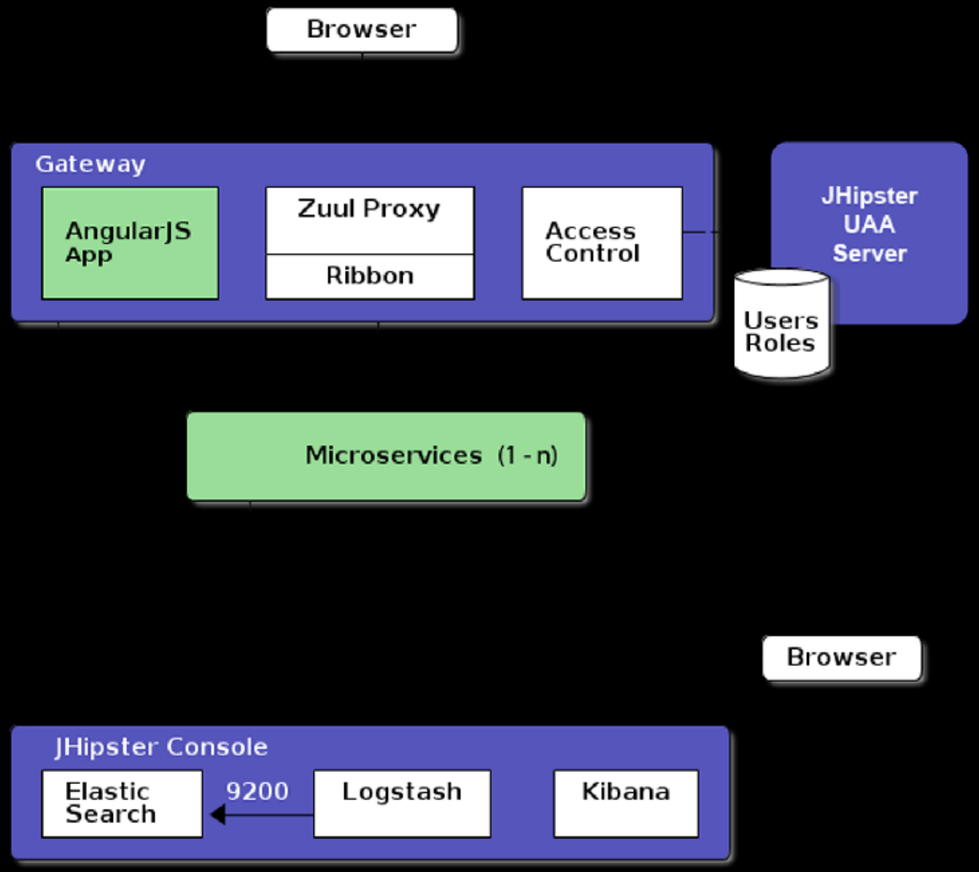
Reactive Manifesto - Message Driven

- Reactive Systems rely on asynchronous message passing to establish a boundary between components.
  - This ensures loose coupling, isolation, and location transparency.
- Message passing enables load management, elasticity, and flow control.
- Location transparent messaging makes management of failures possible
- Non-blocking communication allows recipients to only consume resources while active, leading to less system overhead.

Spring Cloud Native

NETFLIX OSS + + docker





elastic + + kibana

Reactive Programming with Reactive Systems

- Reactive Programming is a useful implementation technique.
- Reactive Programming focuses on non-blocking, asynchronous execution - a key characteristic of Reactive Systems.
- Reactive Programming is just one tool in building Reactive Systems.


SPRING FRAMEWORK
GURU