Introduction to Reactive Systems

Understanding the Concepts and Real-Life Applications

By Dr Varsha Singh on 17/01/2025

Content

- What are Reactive Systems?
- Characteristics of Reactive Systems
- Real-Life Examples of Reactive Systems
- Key Concepts and Design Principles
- Challenges in Reactive Systems
- Summary

What are Reactive Systems?

- Systems that continuously respond to external events or stimuli.

Focus: Event-driven, real-time, continuous operation.

Key Question: How do these systems react to the changing environment?

Characteristics of Reactive Systems

Event-Driven: Respond to external events or changes.

Real-Time: Provide quick responses to inputs.

Always On: Continuous operation without interruption.

Autonomous: Can adapt and make decisions without human intervention.

Key Concepts in Reactive Systems

Event: An external stimulus that triggers a reaction.

State: The current condition or status of the system.

Action: The response of the system to an event.

Feedback Loop: The continuous cycle of receiving inputs, processing them, and reacting.

Real-Life Example

Autonomous Vehicles: Vehicles that navigate and make decisions based on real-time data.

- **Events**: Pedestrians, traffic signals, road conditions.
- **Reaction**: Braking, steering adjustments, speed changes.
- **Importance**: Safety and real-time decision-making.

Other real life examples

- Air Traffic Control Systems
- Smart Thermostats
- Stock Trading Systems etc.

Design Principles of Reactive Systems

Responsiveness: Must respond in a timely manner.

Scalability: Should handle increased load without failure.

Elasticity: Can adapt to changing conditions.

Availability: Continually operational with minimal downtime.

Fault Tolerance: Resilient to failures in critical components.

Challenges in Reactive Systems

Latency: Delays in processing and response time.

Complexity: Difficulty in managing interactions between components.

Scalability: Ensuring the system works efficiently at large scales.

Synchronization: Ensuring proper order of events and actions.

Concurrency: Handling multiple events simultaneously.

Reactive Systems vs. Traditional Systems

Traditional Systems: Request-response models, often synchronous.

Reactive Systems: Event-driven, asynchronous.

Comparison: Performance, flexibility, fault tolerance.

Case Study: Industrial Control Systems

Overview: Systems in factories and power plants that react to changes in machinery or environment.

Events: Pressure, temperature, equipment failure.

Reaction: Adjusting machinery, triggering alarms, shutting down systems for safety.

Technologies Supporting Reactive Systems

Microservices Architecture: Decomposing applications into independent, reactive services.

Reactive Programming: Techniques like RxJava, ReactiveX for building event-driven systems.

Event-Driven Architecture (EDA): Systems designed around events and reactions.

Real-World Applications

Smart Cities: Traffic management, environmental monitoring, public safety.

Healthcare Systems: Patient monitoring, real-time alerting, adaptive treatment.

IoT (Internet of Things): Connected devices reacting to environmental changes.

Summary

- Reactive systems continuously respond to real-time events and inputs.
- **Key Characteristics**: Event-driven, real-time, scalable, and fault-tolerant.
- **Examples**: Autonomous vehicles, stock trading, air traffic control.
- **Challenges**: Handling latency, complexity, scalability, and concurrency.

Thank You!