Akka.NET Persistence Fundamentals

INTRODUCTION TO PERSISTENT ACTORS

Jason Roberts

.NET MVP

@robertsjason dontcodetired.com

Overview

Why persistent actors?

Architectural overview

The event journal store

Supported persistence stores

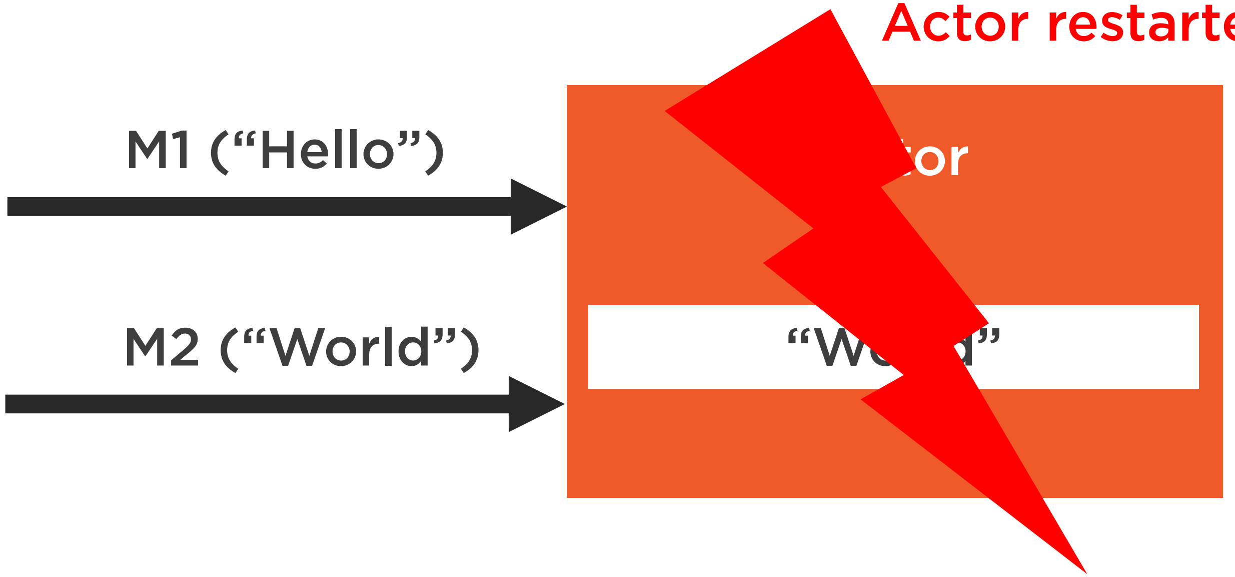
Snapshots

At-Least-Once Message Delivery

Course outline

Demo state loss in non-persistent actors

Why Persistent Actors?

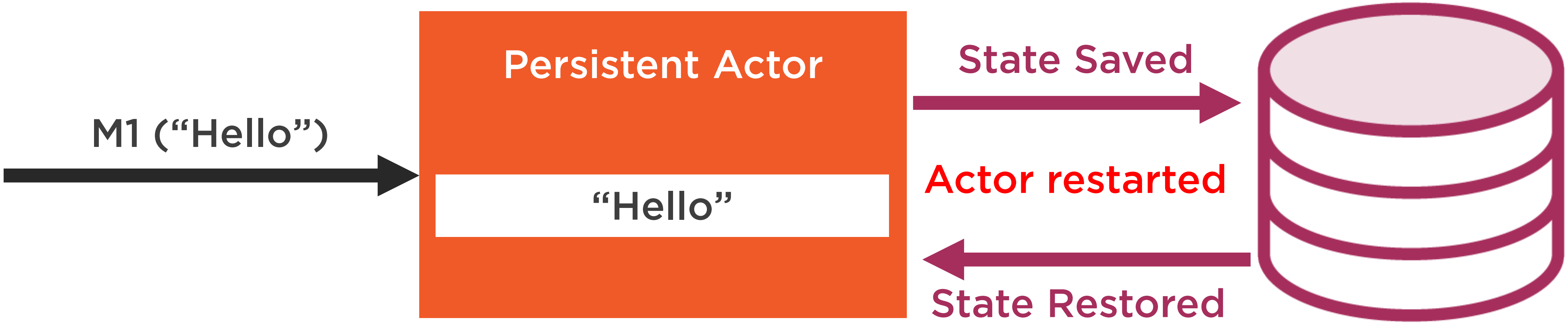


Actor restarted

Internal actor exception
.NET process crash
Server restart/crash

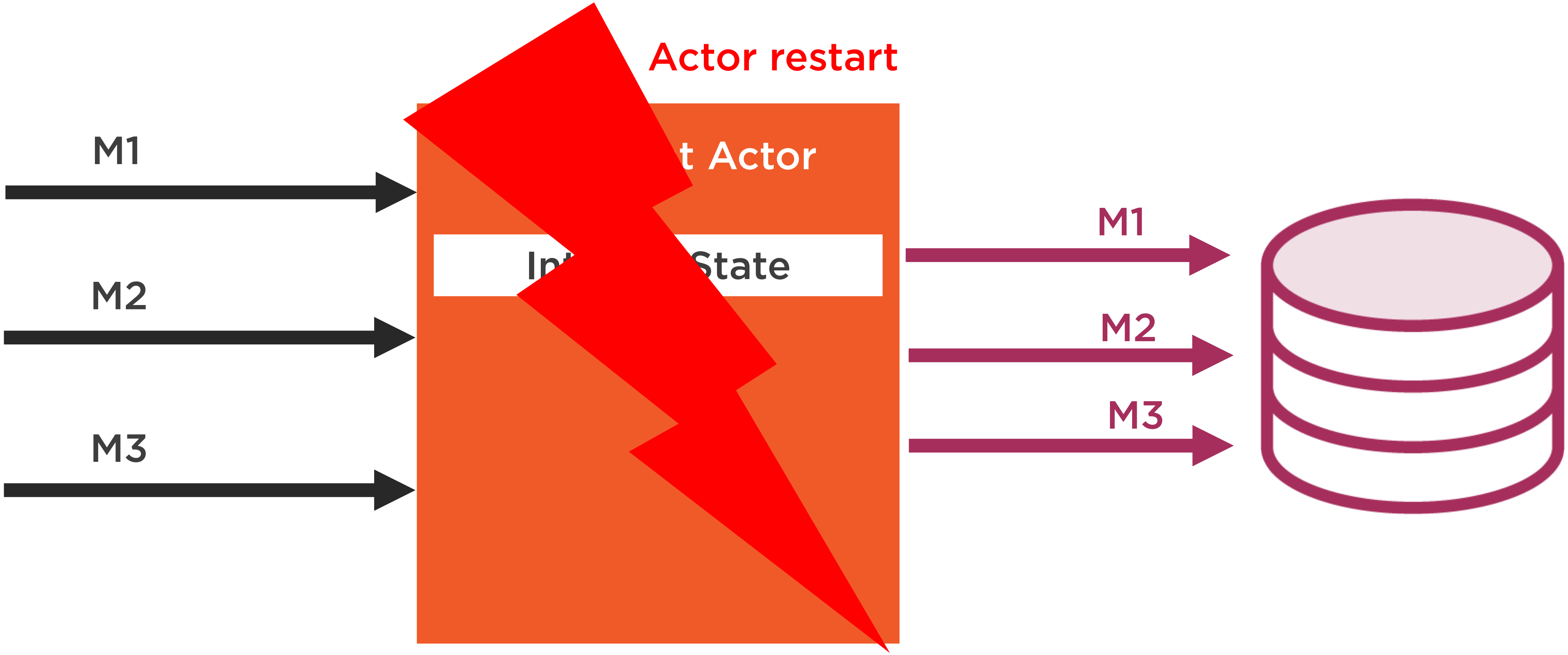
Internal state lost

Why Persistent Actors?

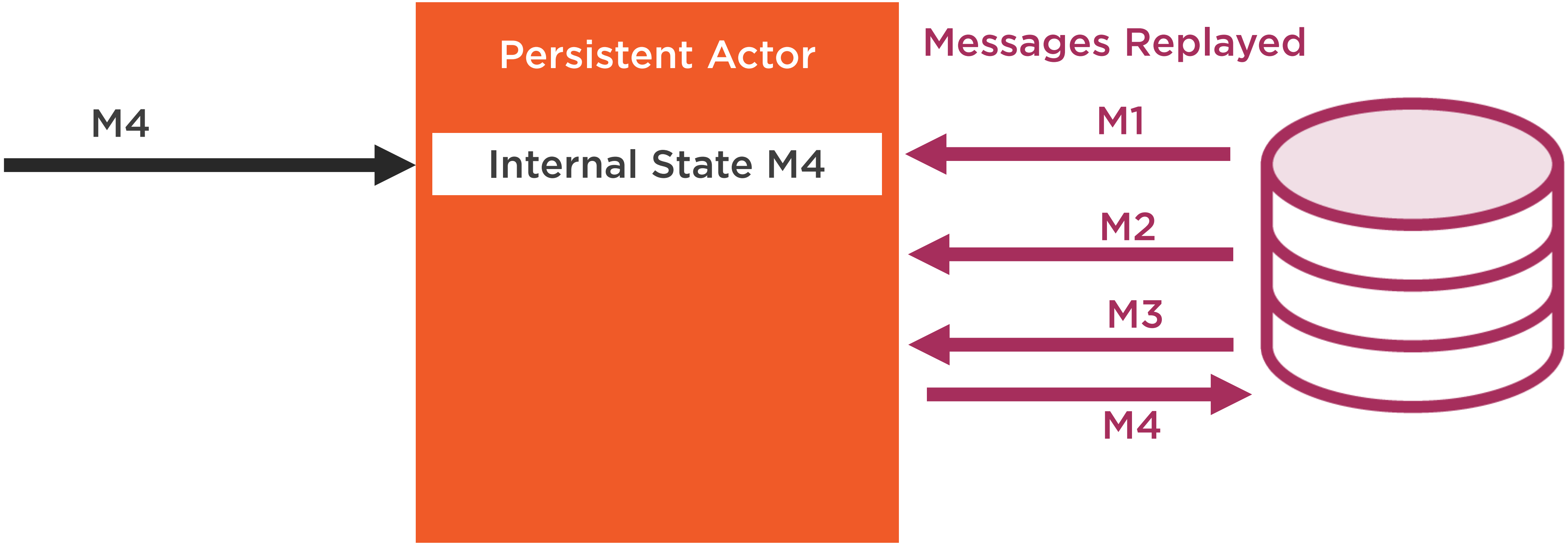


Akka.Persistence allows the creation of actors whose internal state can be preserved and can be restored after an actor has restarted.

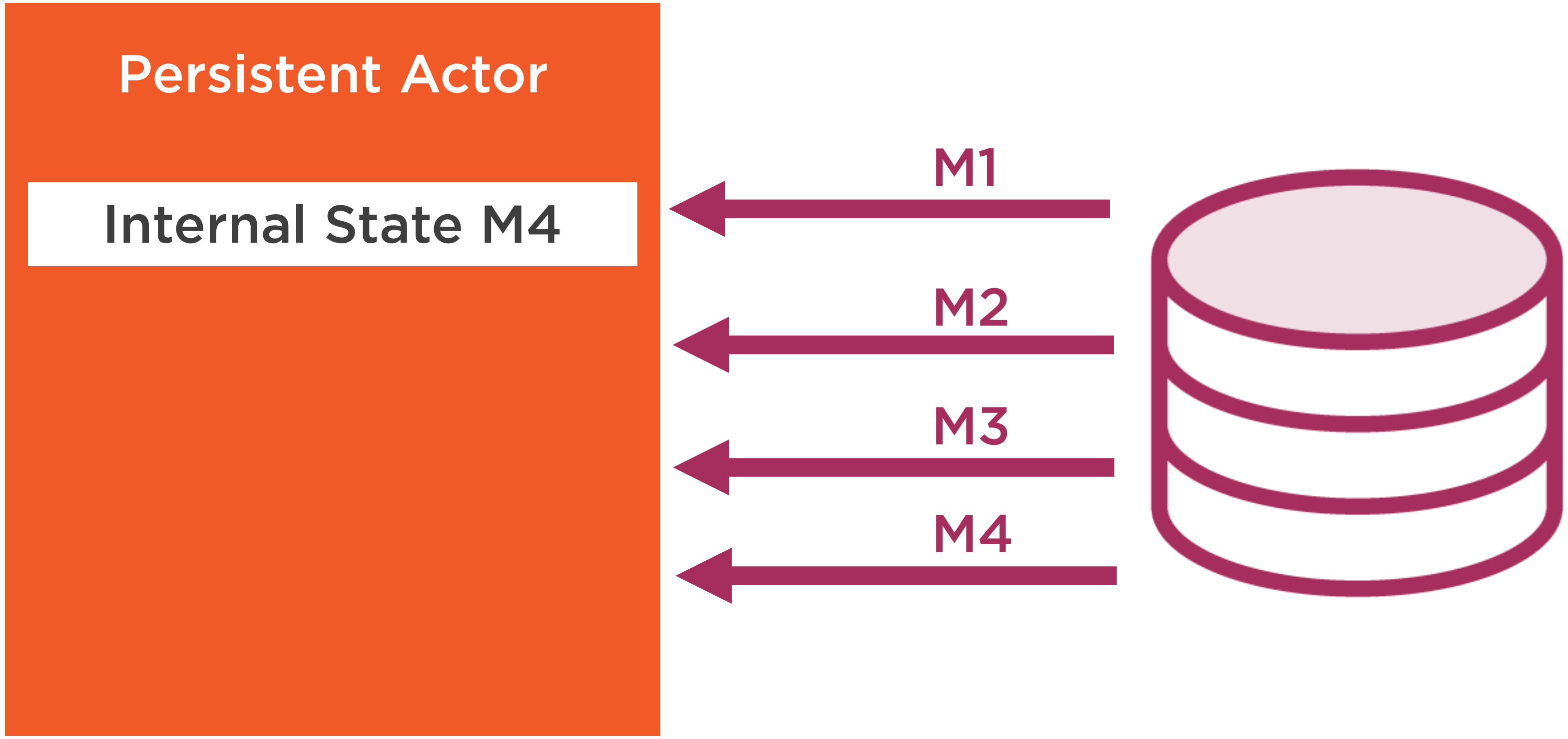
Architectural Overview



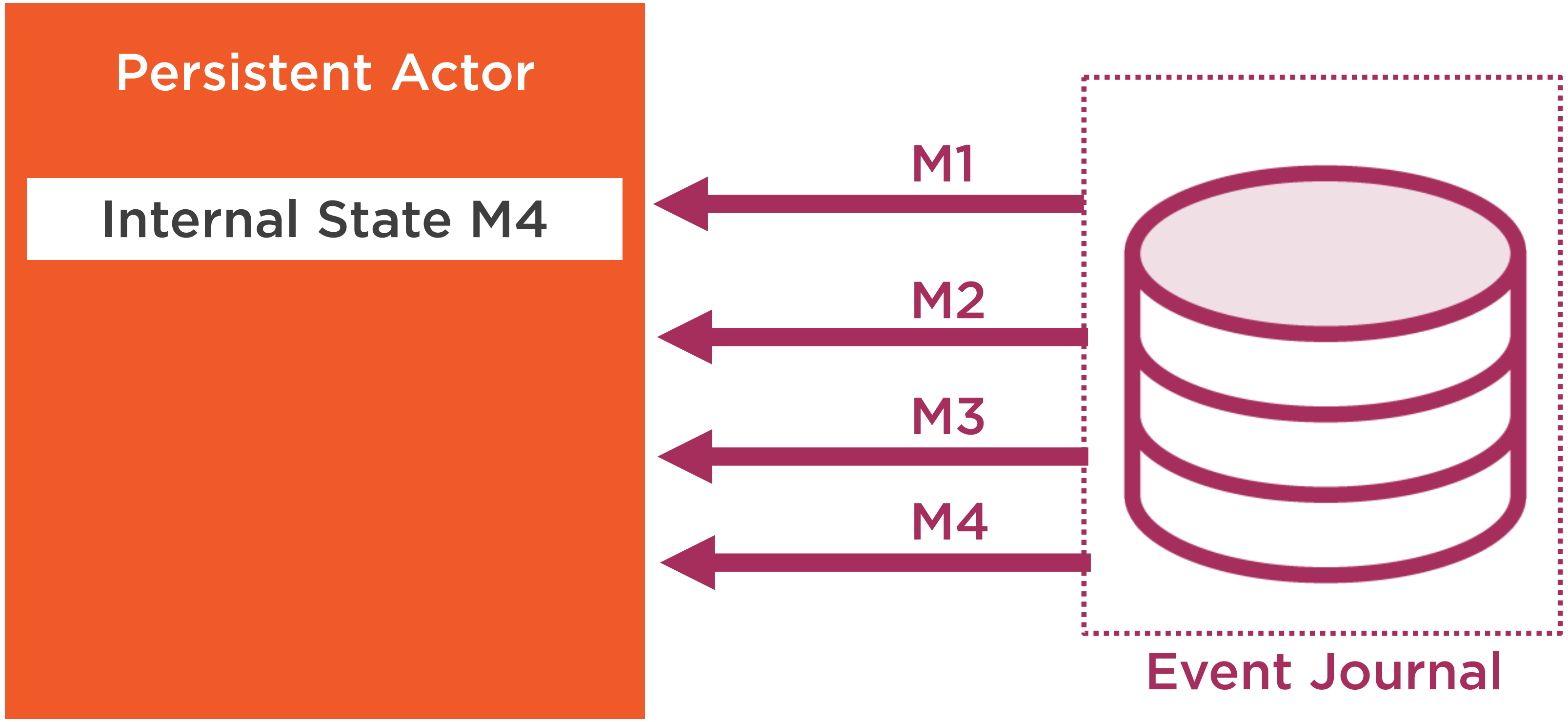
Architectural Overview



Architectural Overview



The Event Journal Store



Persistent Actor

Internal State M4

M1

M2

M3

M4

Event Journal

Holds messages (events) received by actor

Actor state not stored directly

Actor state restored by replaying events

Append only (immutable) store

Stored events never modified

Events can be deleted

High throughput / transaction rates

Choose which messages are persisted

“Commands” & “Events”

Supported Persistence Stores

Plugin model

Abstracts underlying journal store

Actor code same regardless of store

Journal plugin specified in configuration

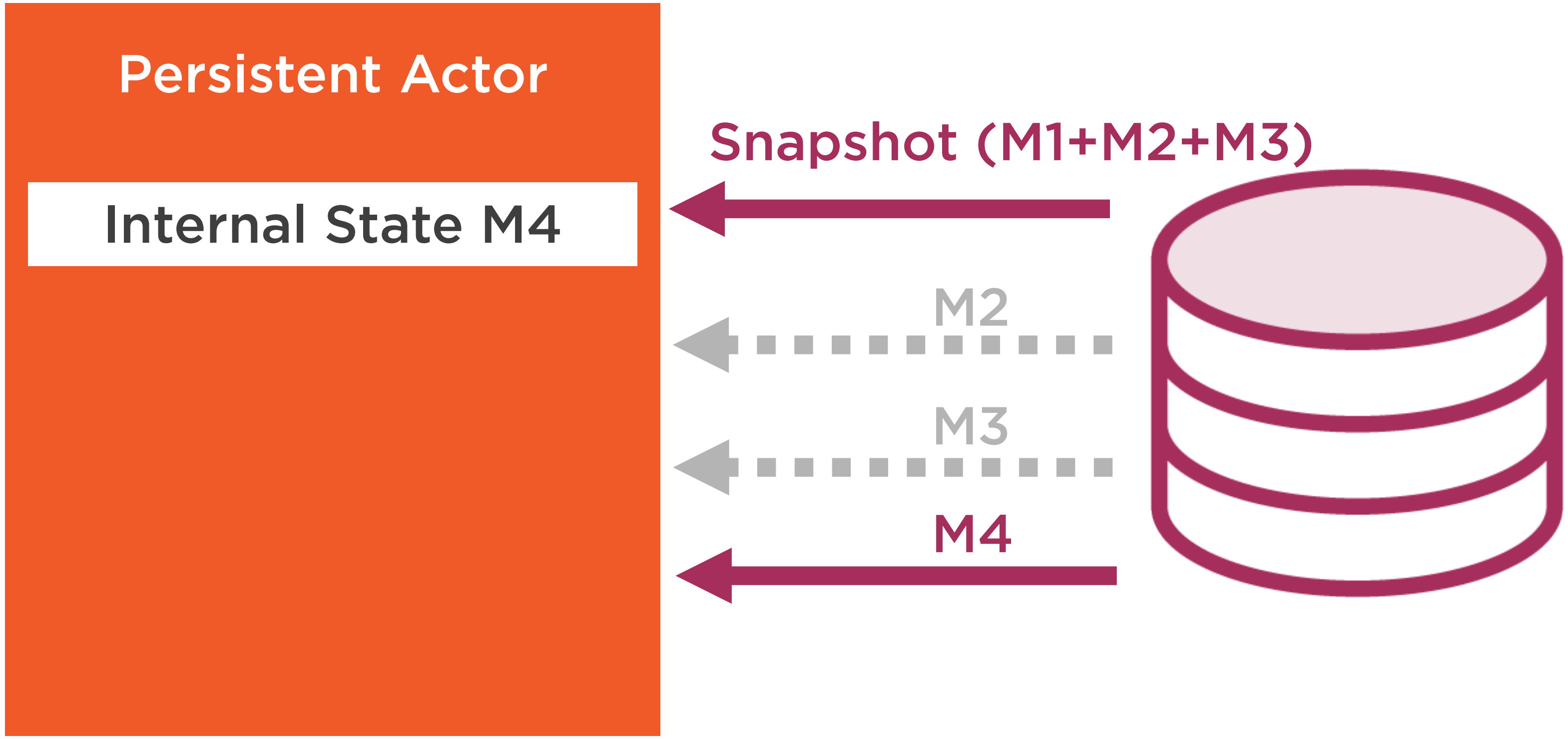
Additional NuGet packages for stores

- SqlServer
- MySql
- Cassandra
- MongoDB
- Redis

Implement custom stores

Snapshots

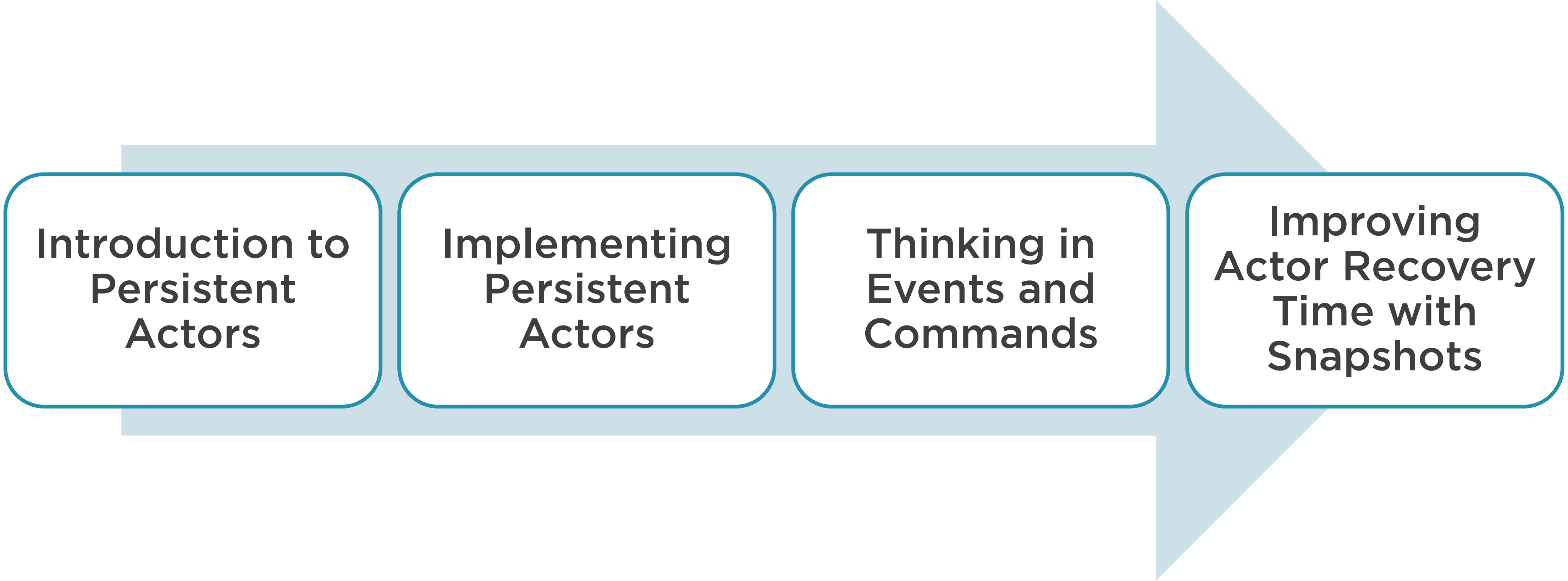
Snapshots decrease the time taken to recover actor state



At-Least-Once Delivery

The Akka.Persistence NuGet package also adds support for at-least-once message delivery. When implemented, the sending actor will resend messages to the destination actor automatically if they are not received.

Course Outline



Introduction to Persistent Actors

Implementing Persistent Actors

Thinking in Events and Commands

Improving Actor Recovery Time with Snapshots

Demo

State Loss in Non-Persistent Actors

Actors in online game

PlayerActor

Internal state:

- Player name
- Current player health

Receive<HitMessage>

Reduces player's health

Receive<CauseErrorMessage>

Actor restart - state loss

Restart .NET process – state loss

Summary

Persistent actors allow us to retain state

Actor restart (actor error / process crash)

Architectural overview

Messages (events) written to journal store

SqlServer, MySql, MongoDB, etc.

Snapshots to increase recovery speed

At-least-once message delivery

Demo state loss in non-persistent actors

PlayerActor lost health state

Next:

Implementing Persistent Actors