Understanding Actor Lifecycles and States

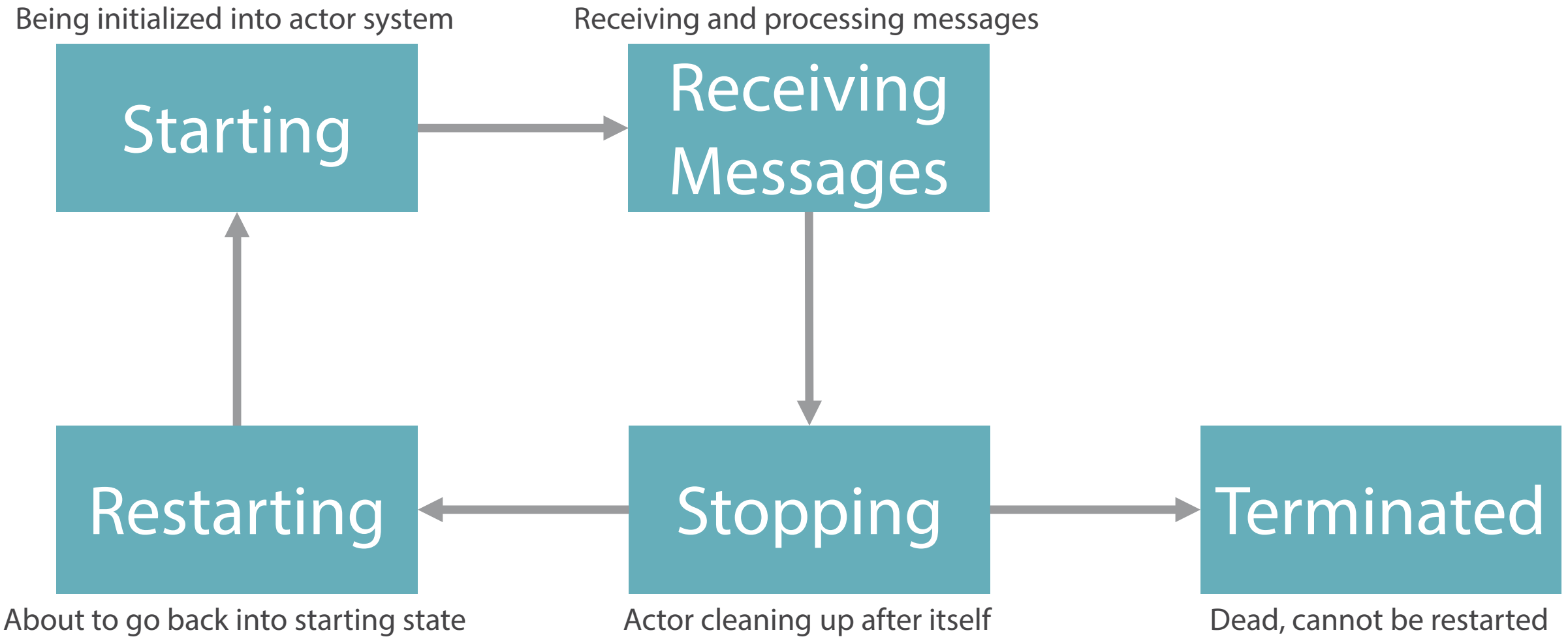
Jason Roberts

@robertsjason | dontcodetired.com

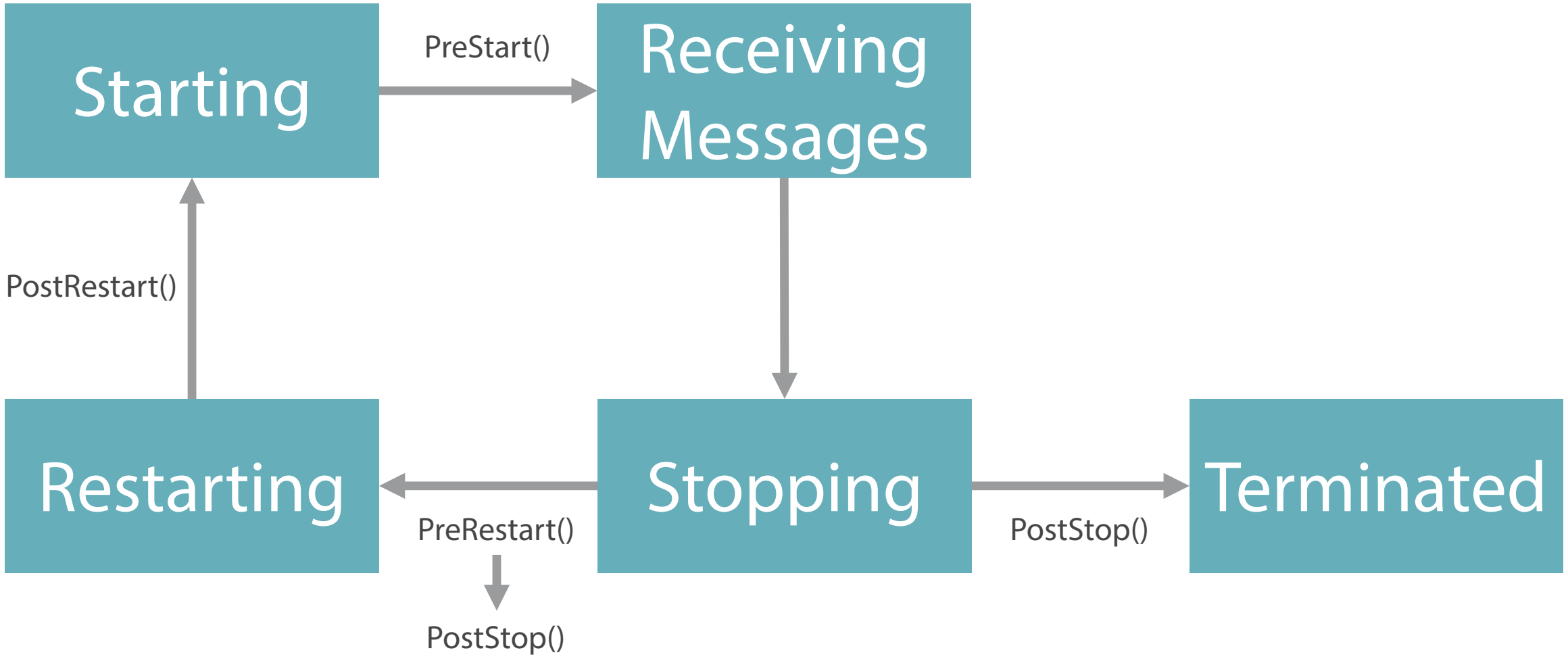
Overview

- Actor instance lifecycle phases
- Lifecycle hook methods
- Terminating actor instances
- Terminating actor hierarchies
- Sending PoisonPill messages
- Switchable actor behaviours

Actor Instance Lifecycle



Lifecycle Hook Methods



Lifecycle Hook Methods

- PreStart()
  - Called before actor instance receives first message
  - Custom initialization code, getting actor ready to start receiving messages
  - E.g. opening/creating files, system handles, etc.
- PostStop()
  - Called after the actor has been stopped and is not receiving messages any more
  - Custom cleanup code
  - E.g. release system resources/handles such as file system

Lifecycle Hook Methods

- PreRestart()
  - Called before actor begins restarting
  - Allows code to do something with current message/exception
  - E.g. Save current message for reprocessing later when actor restarts
- PostRestart()
  - Called after PreRestart() and before PreStart()
  - Allows code to do something with exception
  - E.g. Additional custom diagnostic/logging

Overriding Lifecycle Hook Methods

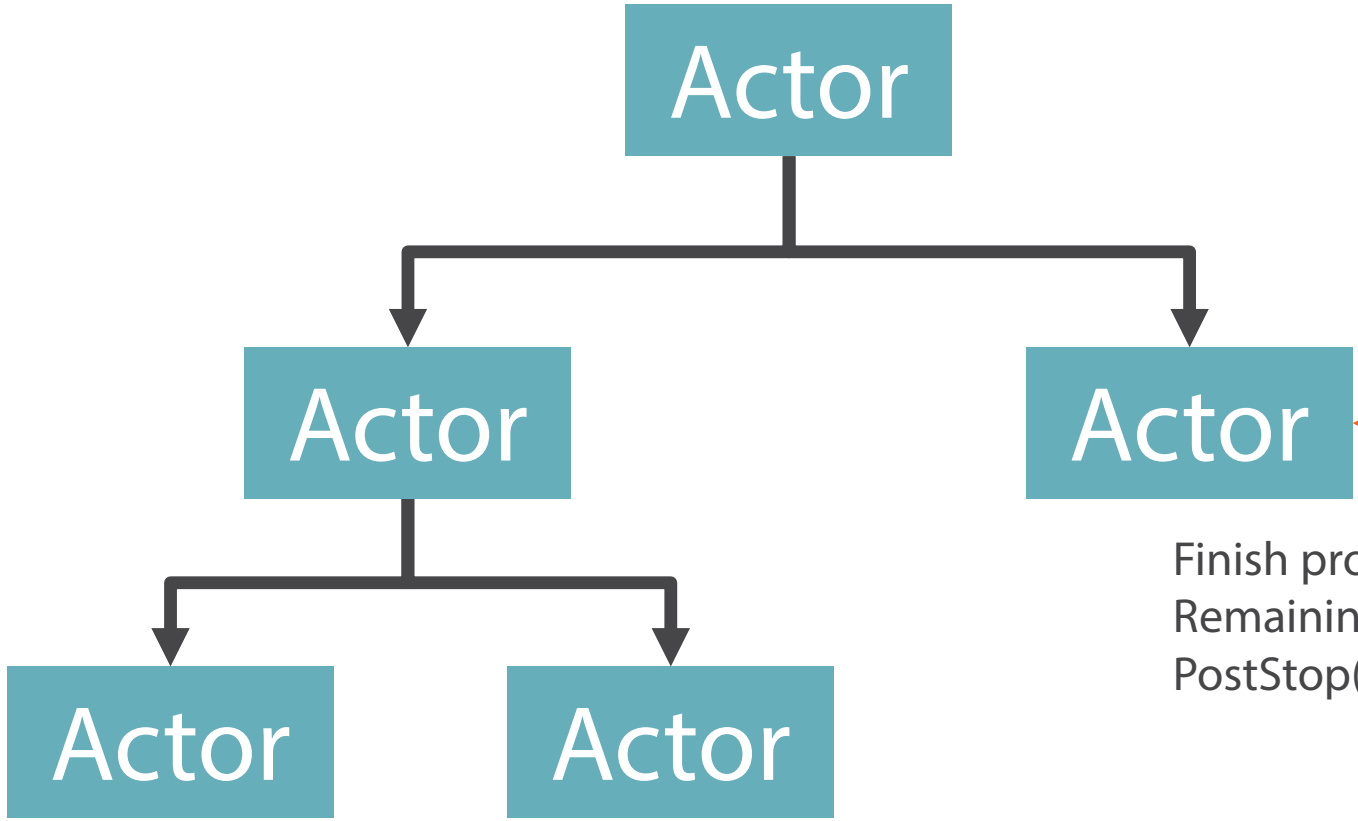
Add lifecycle hooks for PlaybackActor

Override PreStart() and PostStop()

Override PreRestart()

Override PostRestart()

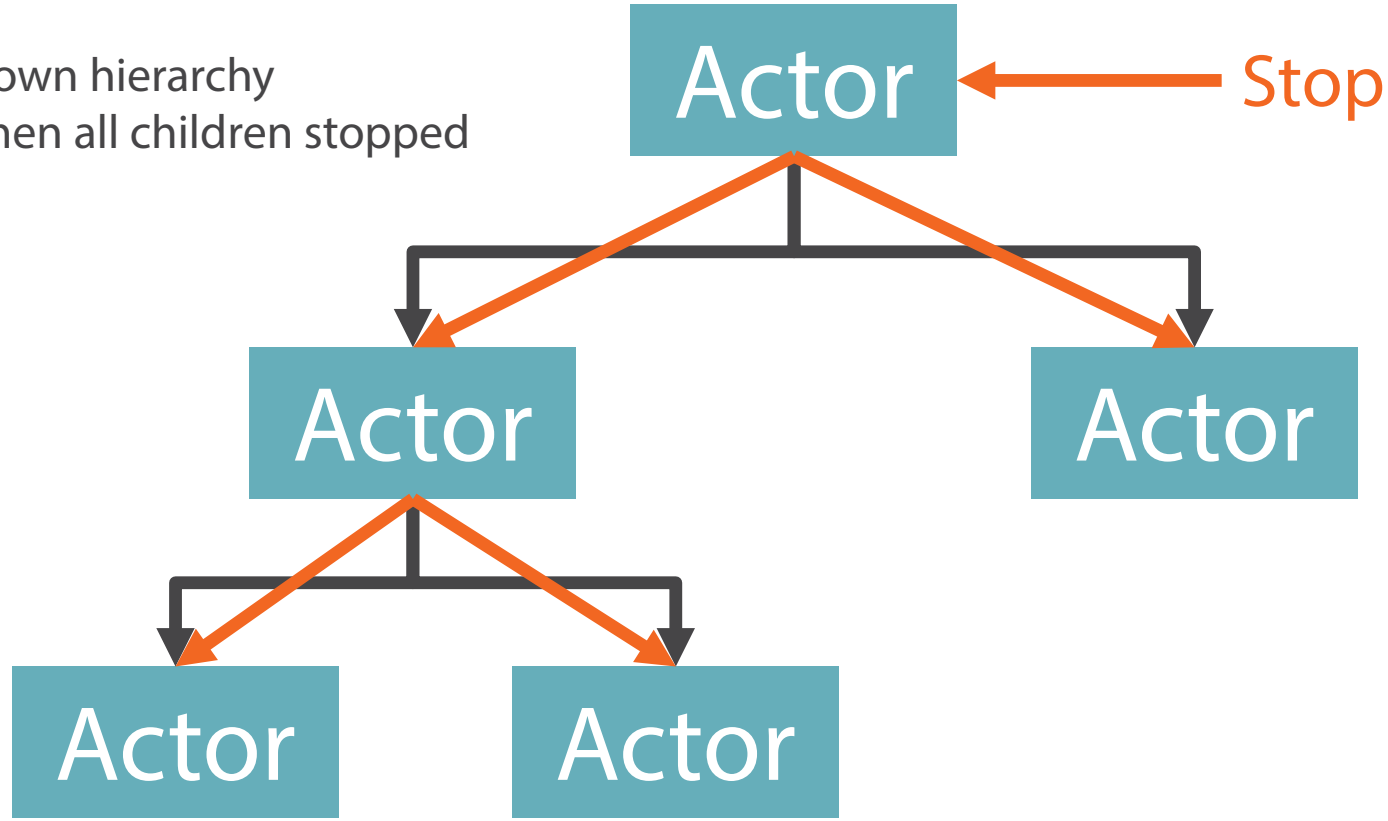
Terminating Actors and Hierarchies



Finish processing current message (if any)
Remaining messages won't be processed
PostStop() hook method called

Terminating Actors and Hierarchies

Tell children to stop
Shutdown “flows” down hierarchy
PostStop() called when all children stopped



Terminating Actors and Hierarchies

- MovieStreamingActorSystem.Shutdown();
- Child terminated automatically by its parent on exception
- Child terminated by us manually in parent code
  - Context.Stop(someChildActorRef);
- Send a special Akka.NET PoisonPill message to an actor
  - someActorRef.Tell(PoisonPill.Instance);
- Termination of an actor happens asynchronously
- To manually terminate and wait use GracefulStop
  - await someActorRef.GracefulStop(TimeSpan.FromMinutes(1));

Sending a Poison Pill Message

Switchable Actor Behaviour

Receive & react to messages

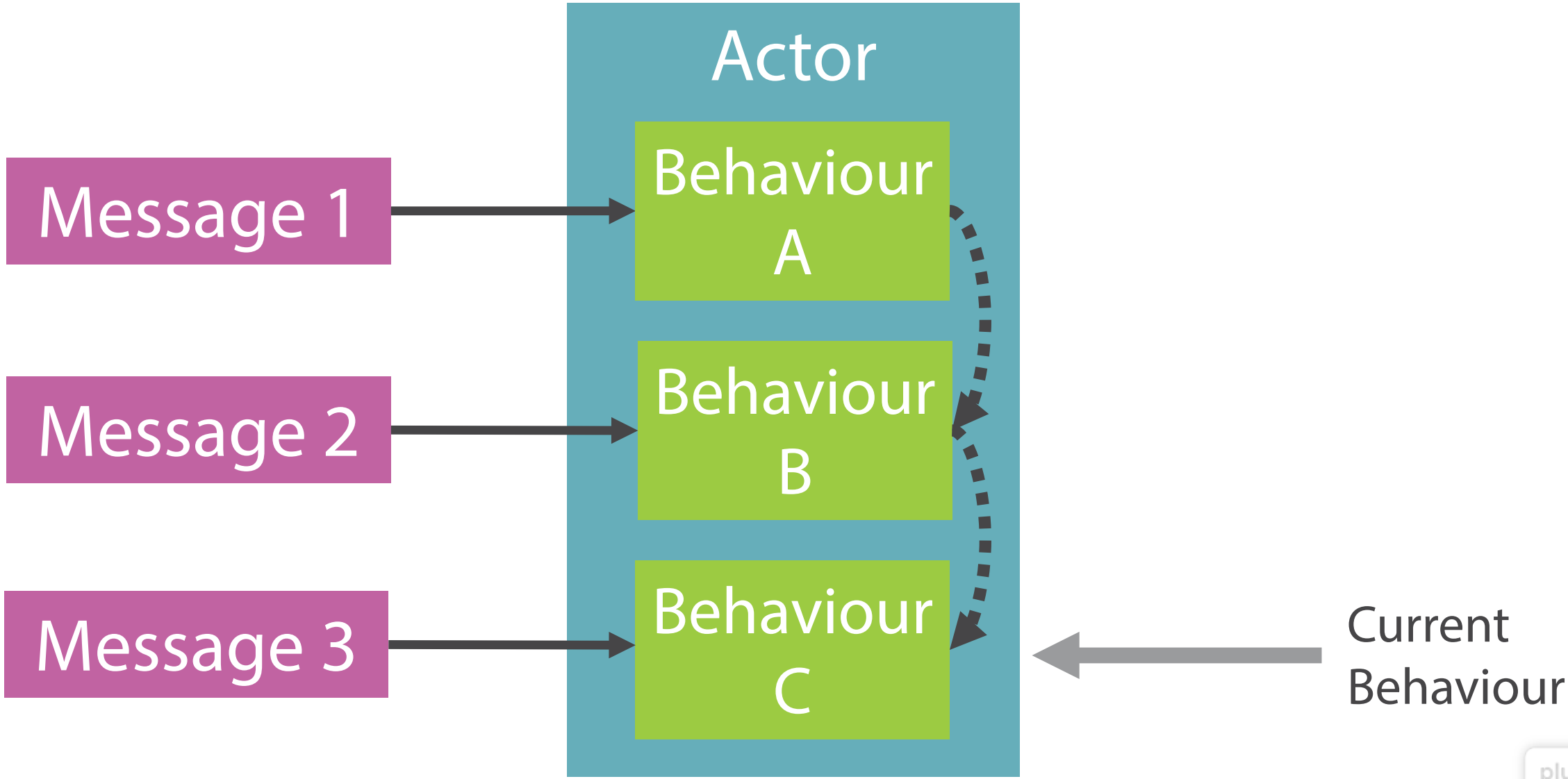
Change behaviour for next message

Actor

Create more actors

Send messages to other actors

Switchable Actor Behaviour



Akka.NET Behaviour Switching API

- Switching to new explicitly specified behaviour
  - Become() method
  - Existing configured behaviour not remembered
- Using the behaviour stack
  - BecomeStacked() switches to new behaviour and pushes existing behaviour down the behaviour stack
  - UnbecomeStacked() pops current behaviour off stack and the previously pushed behaviour is restored

The change in behaviour applies to the next message that gets processed

> Switchable behavior [sic] is one of the most powerful and fundamental capabilities of any true actor system. It's one of the key features enabling actor reusability, and helping you to do a massive amount of work with a very small code footprint.
>
> — petabridge.com (bit.ly/petaquote)

Creating a UserActor

Add a StopMovieMessage class

Add a UserActor class

Implement logic without using switchable behaviors

Refactoring to Use Switchable Behaviours

Create 2 methods to represent stopped and playing behaviours

Remove if statements

Become()

Summary

Actor instance lifecycle phases

PreStart(), PostStop(), PreRestart()
PostRestart()

Terminating actor instances & hierarchies

playbackActorRef.Tell(PoisonPill.Instance);

Become()

Next:

Creating Actor Hierarchies and Isolating Faults