Distributed Actors

Frank Lefebvre / try! Swift World

@franklefebvre

Agenda

- Presentation
- From Local to Distributed
- Actor System Internals
- MultipeerActorSystem

Sample Project

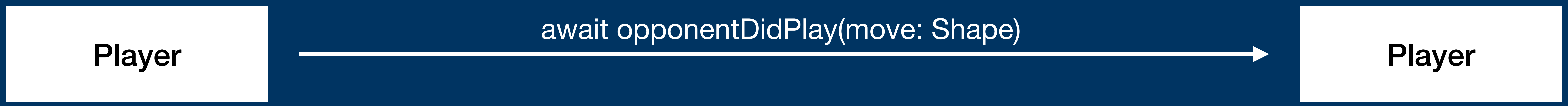
- https://github.com/franklefebvre/DistributedActorsWorkshop.git

From Local to Distributed

Distributed Actor Systems

Method call

Local Actor



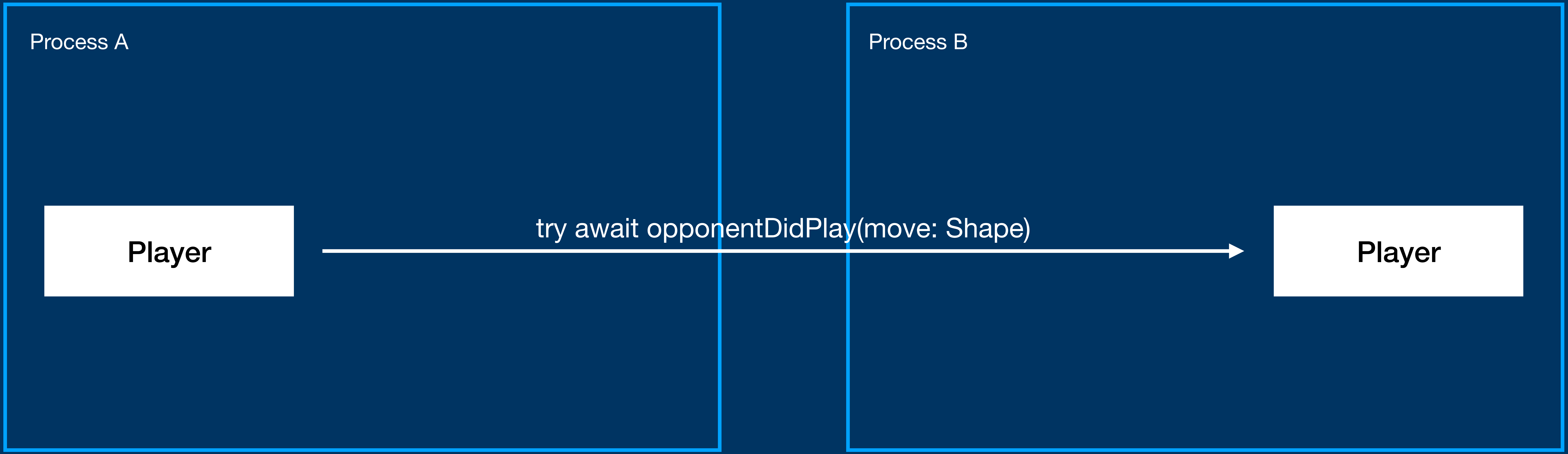
Player

await opponentDidPlay(move: Shape)

Player

Method call

Distributed Actor



Process A

Process B

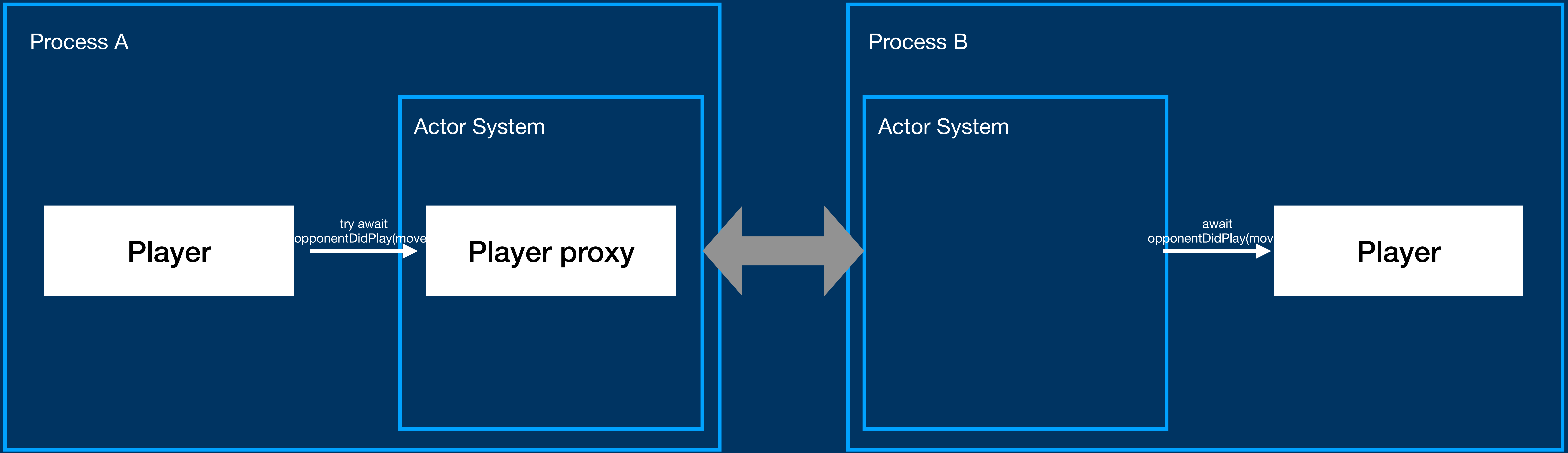
Player

try await opponentDidPlay(move: Shape)

Player

Method call

Distributed Actor

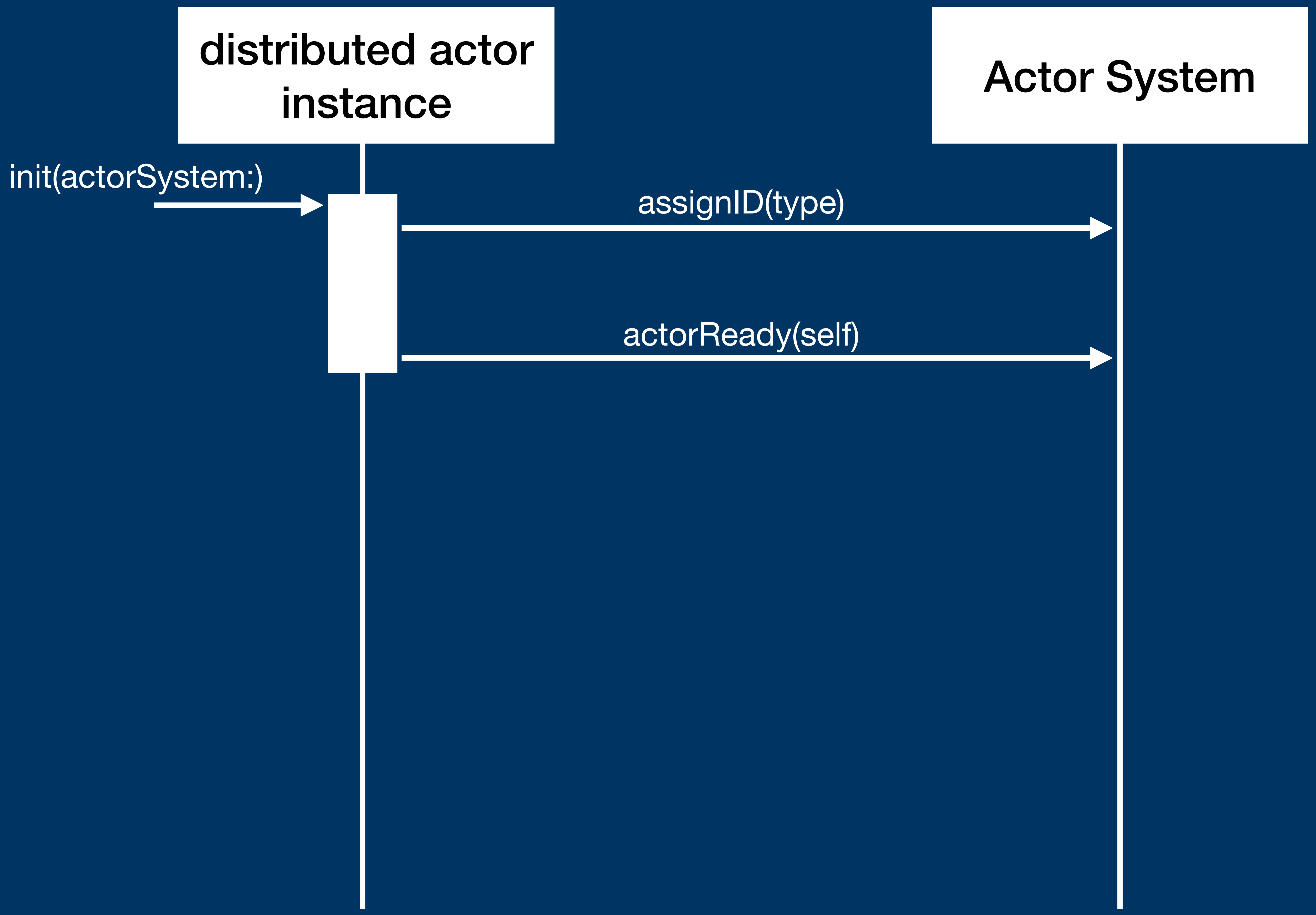


Distributed Actor System

- Actor Identification
- Invocation encoding/decoding
- Transport

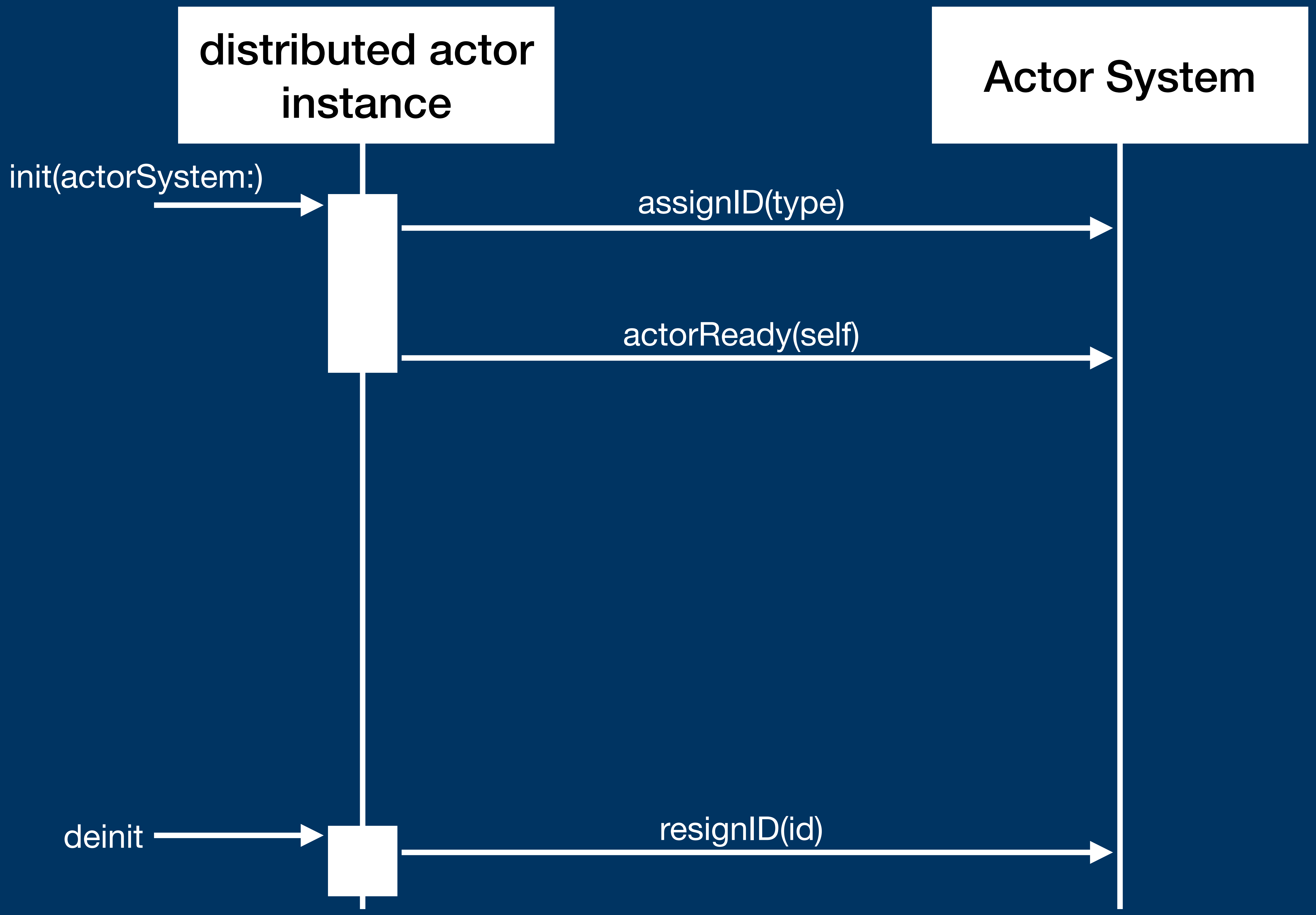
Actor Identification

ActorID: lifecycle



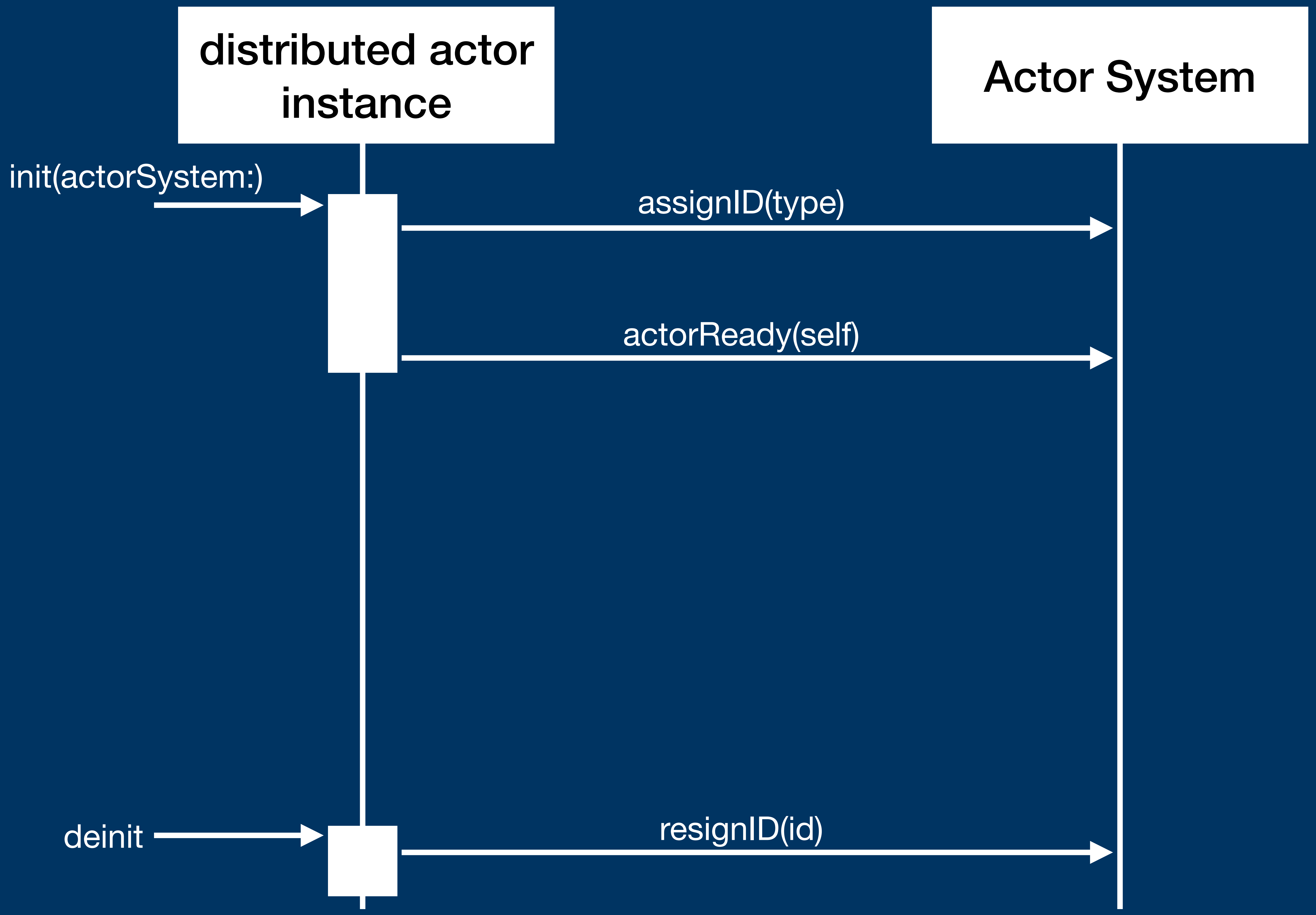
Actor Identification

ActorID: lifecycle



Actor Identification

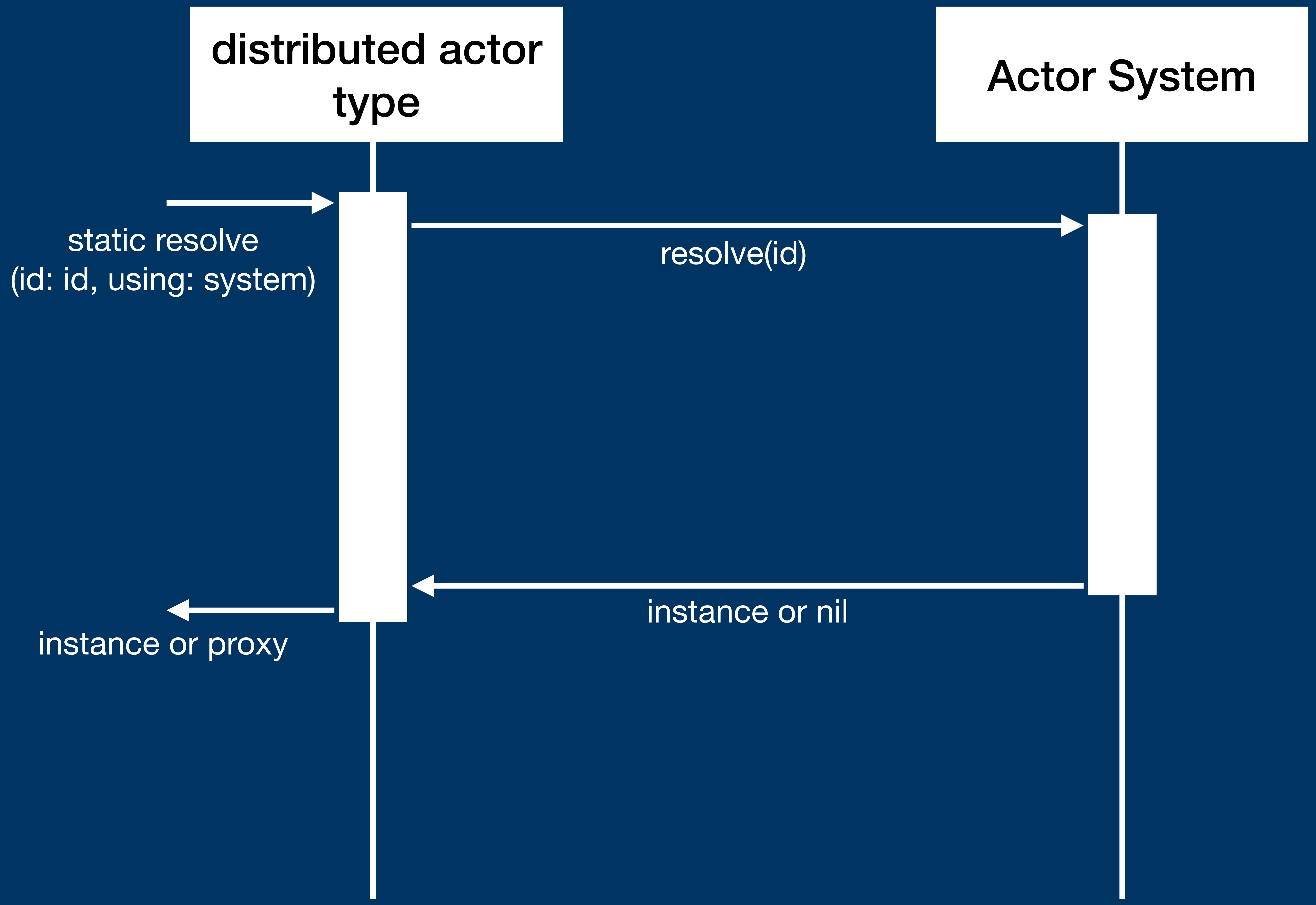
ActorID: lifecycle



- Synthesized at init/deinit time
- Local-only
- Synchronous
- ActorID must be serializable

Actor Identification

ActorID: resolving



distributed actor type

Actor System

static resolve
(id: id, using: system)

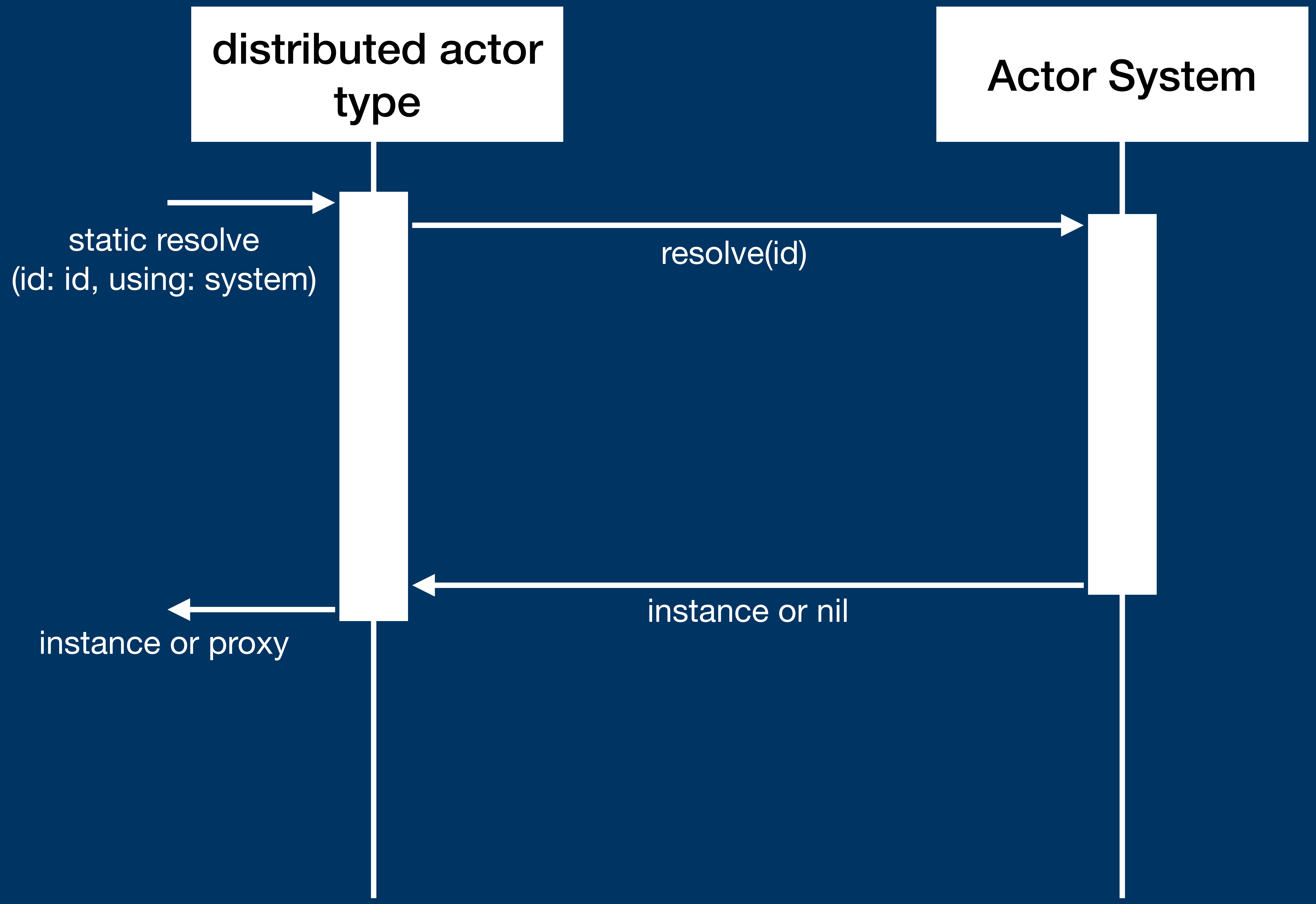
resolve(id)

instance or nil

instance or proxy

Actor Identification

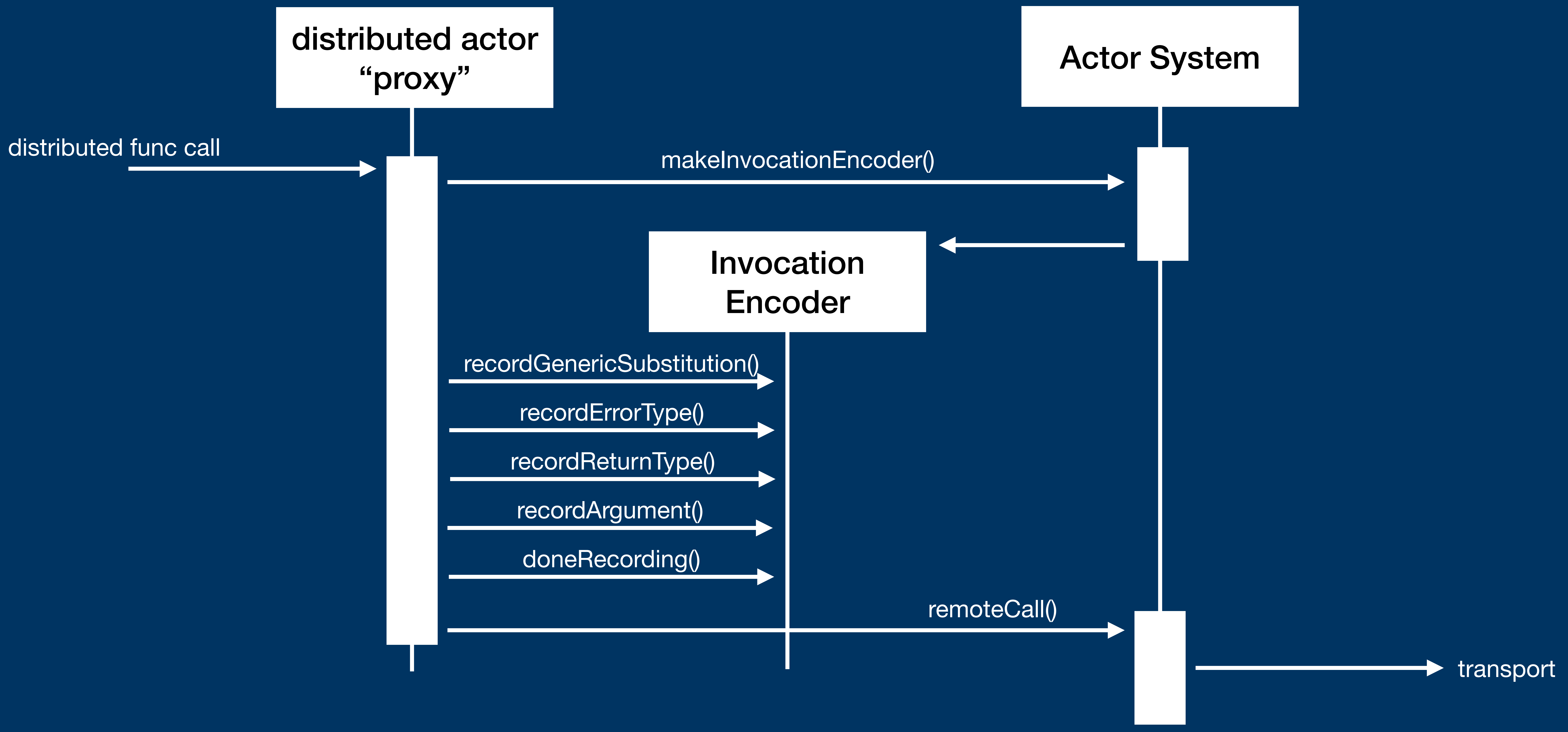
ActorID: resolving



- resolve() is always synchronous
- No network access

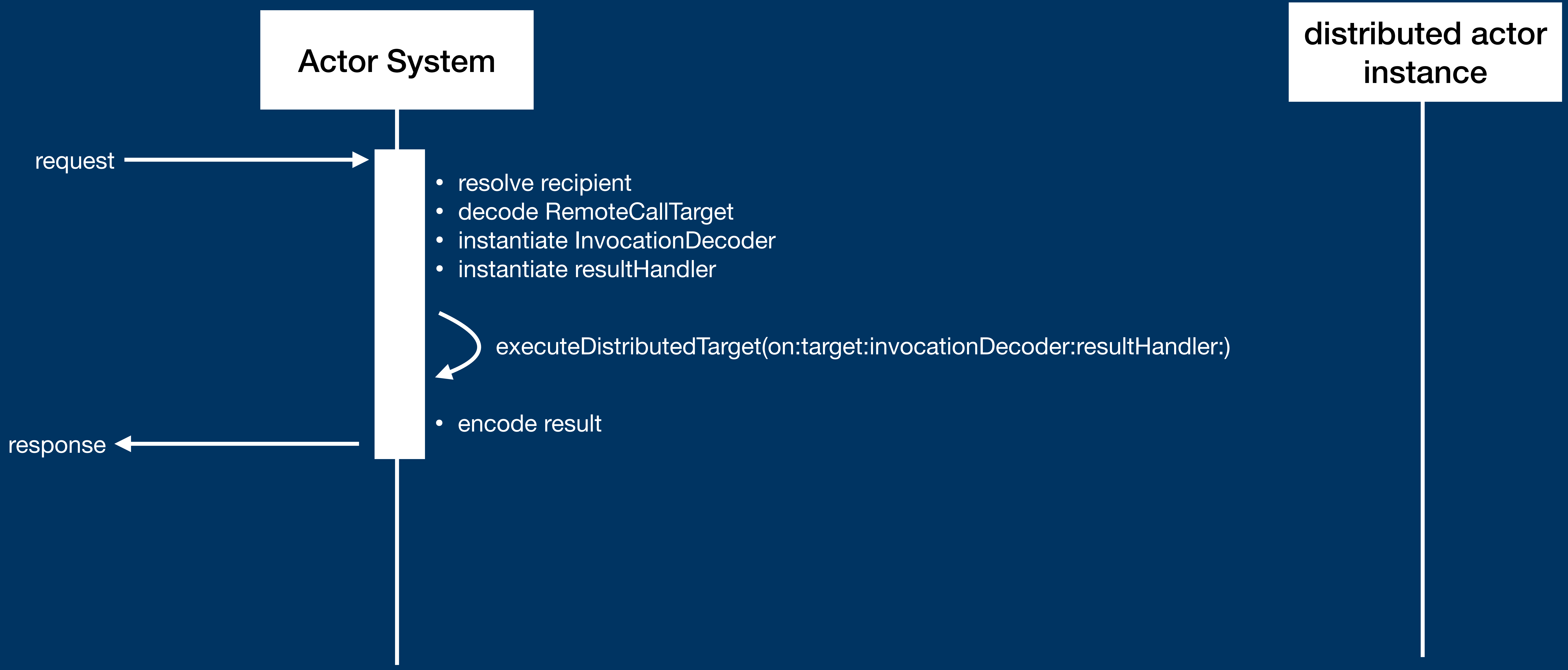
Invocation (sender)

InvocationEncoder



Invocation (receiver)

InvocationDecoder, ResultHandler



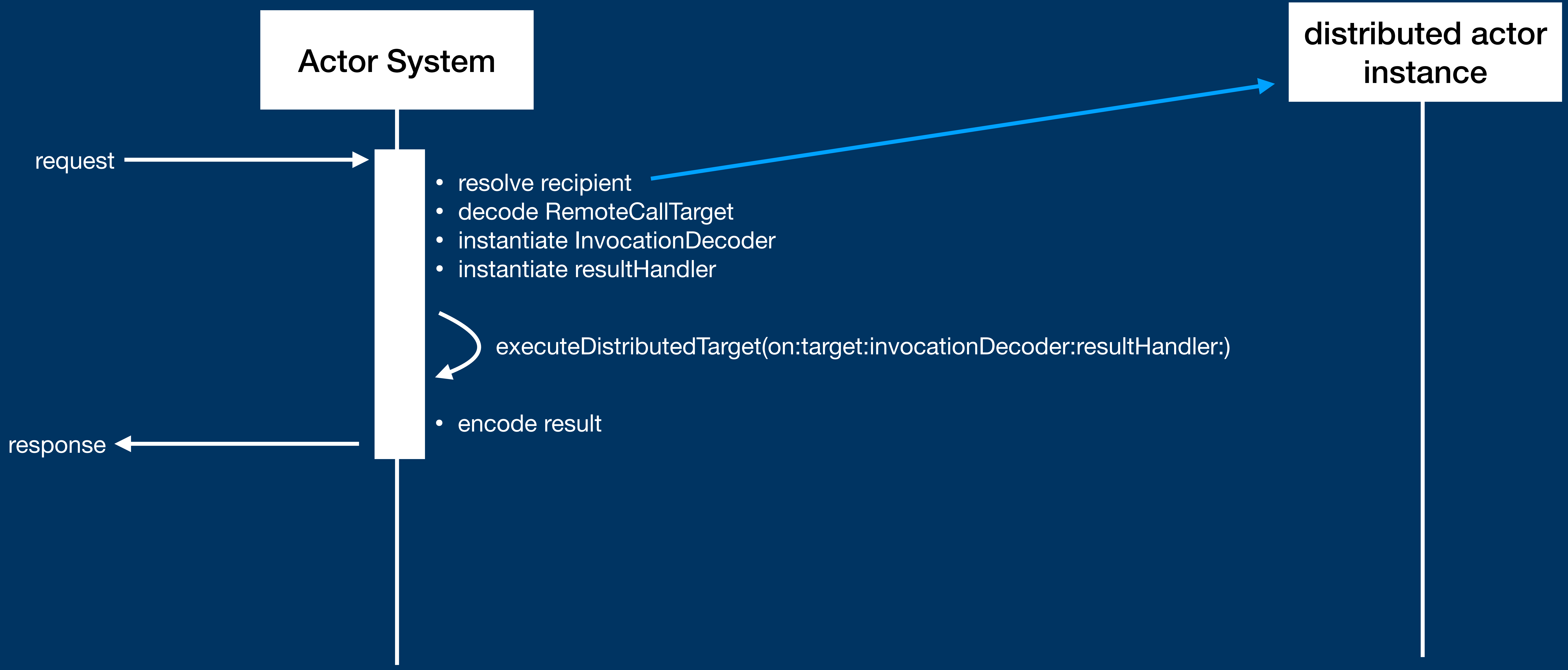
Actor System
distributed actor instance
request
response
executeDistributedTarget(on:target:invocationDecoder:resultHandler:)

- resolve recipient
- decode RemoteCallTarget
- instantiate InvocationDecoder
- instantiate resultHandler

- encode result

Invocation (receiver)

InvocationDecoder, ResultHandler



Actor System

distributed actor instance

request

- resolve recipient
- decode RemoteCallTarget
- instantiate InvocationDecoder
- instantiate resultHandler

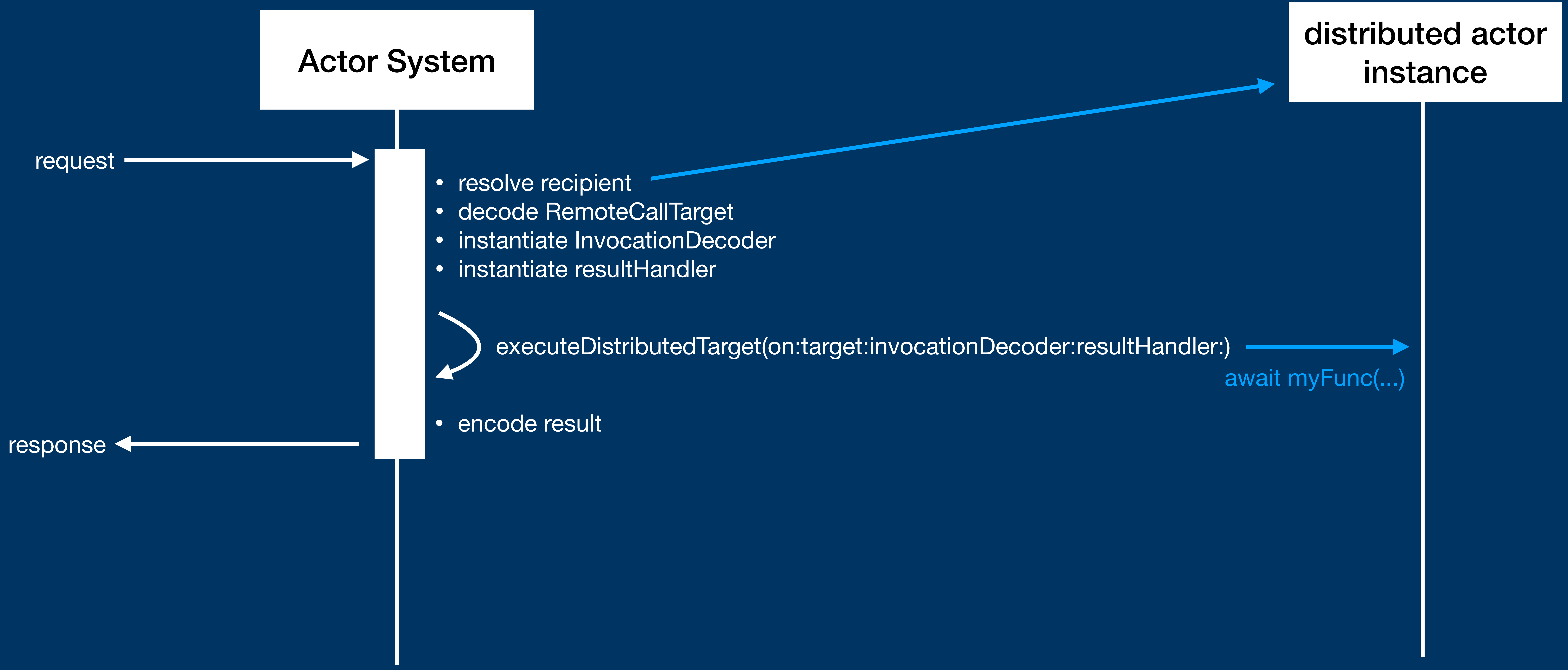
executeDistributedTarget(on:target:invocationDecoder:resultHandler:)

- encode result

response

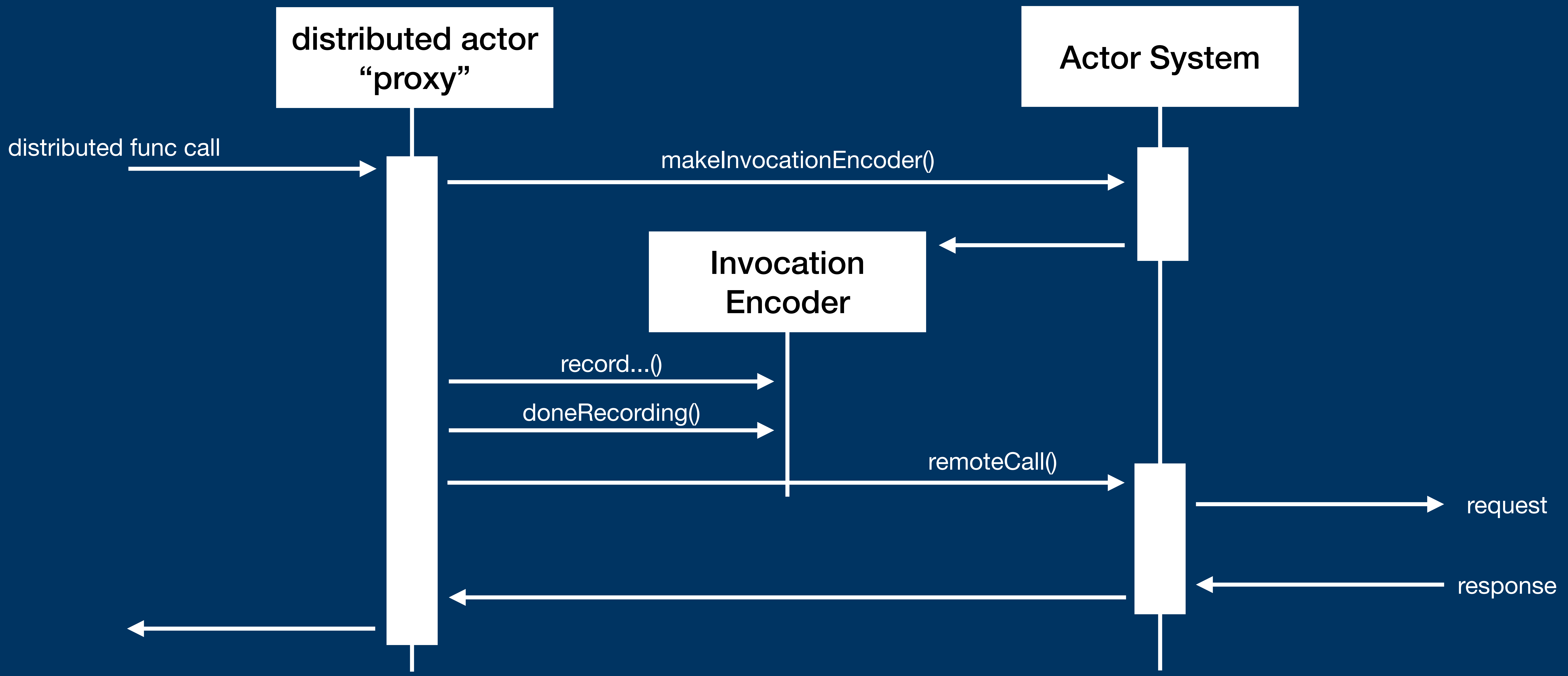
Invocation (receiver)

InvocationDecoder, ResultHandler


Actor System
distributed actor instance
request
resolve recipient
decode RemoteCallTarget
instantiate InvocationDecoder
instantiate resultHandler
executeDistributedTarget(on:target:invocationDecoder:resultHandler:)
await myFunc(...)
encode result
response

Invocation (sender)

Retrieving results



Hands On

More Contents

- Introducing Swift Distributed Actors
    - https://www.swift.org/blog/distributed-actors/
- Swift Evolution Proposals
    - SE-336 - Distributed Actor Isolation
    - SE-344 - Distributed Actor Runtime
- WWDC 2022
    - Meet Distributed Actors in Swift