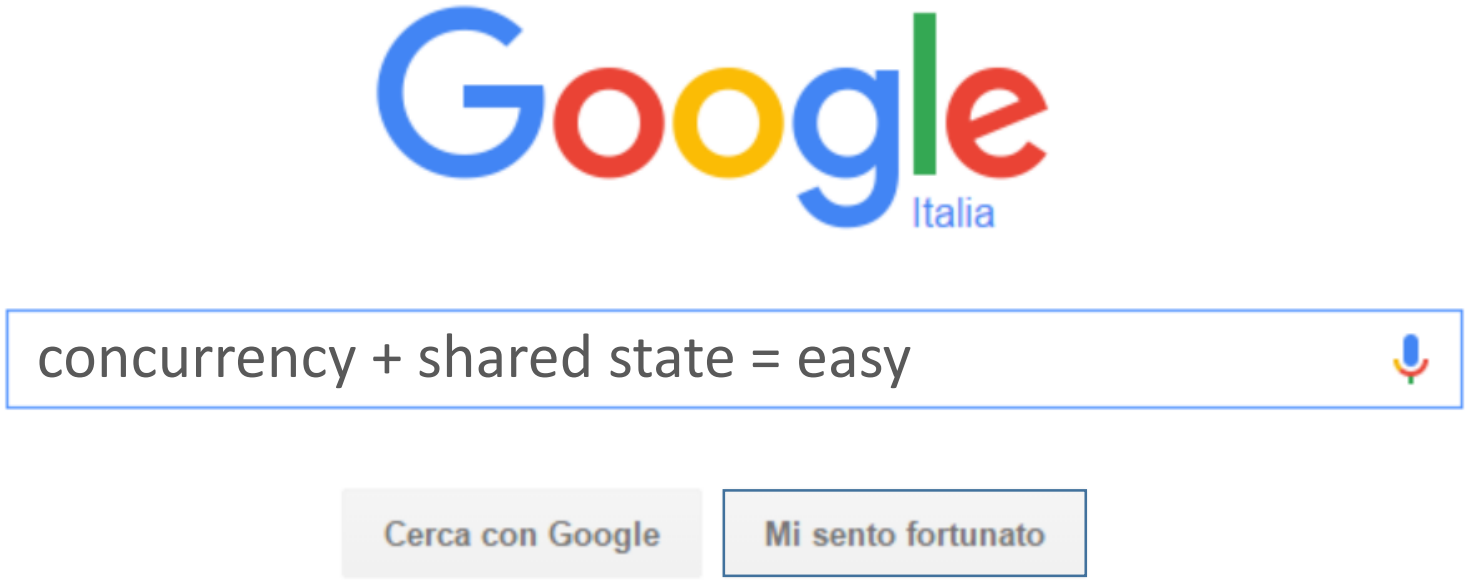
Un «actor» model per amico

Alessandro Melchiori

19 LUGLIO

Give it a try...


Google
Italia
concurrency + shared state = easy
Cerca con Google
Mi sento fortunato


CHE CAVOLO STAI DICENDO
WILLIS?
memegenerator.net

Shared state and concurrency

- Synchronizing shared state
- Coordinating work among threads
- Use synchronization mechanisms
- Debugging..WTF?
- …

WRITING THREAD-SAFE CODE IS HARD

TheSpeaker.AboutMe();

- Alessandro Melchiori
- Partner @ CodicePlastico srl
- Software Architect @ Particular Software

- @amelchiori
- alessandro.melchiori@particular.net
- http://melkio.codiceplastico.com
- http://github.com/melkio

TheTalk.About();

- Actor model: introduzione
- Akka.Net
- Azure Service Fabric: Reliable Actors

Actor model

The actor model in computer science is a mathematical model of concurrent computation that treats "actors" as the universal primitives of concurrent computation: in response to a message that it receives, an actor can make local decisions, create more actors, send more messages, and determine how to respond to the next message received

https://en.wikipedia.org/wiki/Actor_model

Actor model

The actor model in computer science is a mathematical model of concurrent computation that treats "actors" as the universal primitives of concurrent computation: in response to a message that it receives, an actor can make local decisions, create more actors, send more messages, and determine how to respond to the next message received

https://en.wikipedia.org/wiki/Actor_model

Alan Key and OOP

«... I thought of objects being like biological cells and/or individual computers on a network, only able to communicate with messages (so messaging came at the very beginning -- it took a while to see how to do messaging in a programming language efficiently enough to be useful).»

«... OOP to me means only messaging, local retention and protection and hiding of state-process, and extreme late-binding of all things.»

http://userpage.fu-berlin.de/~ram/pub/pub_jf47ht81Ht/doc_kay_oop_en

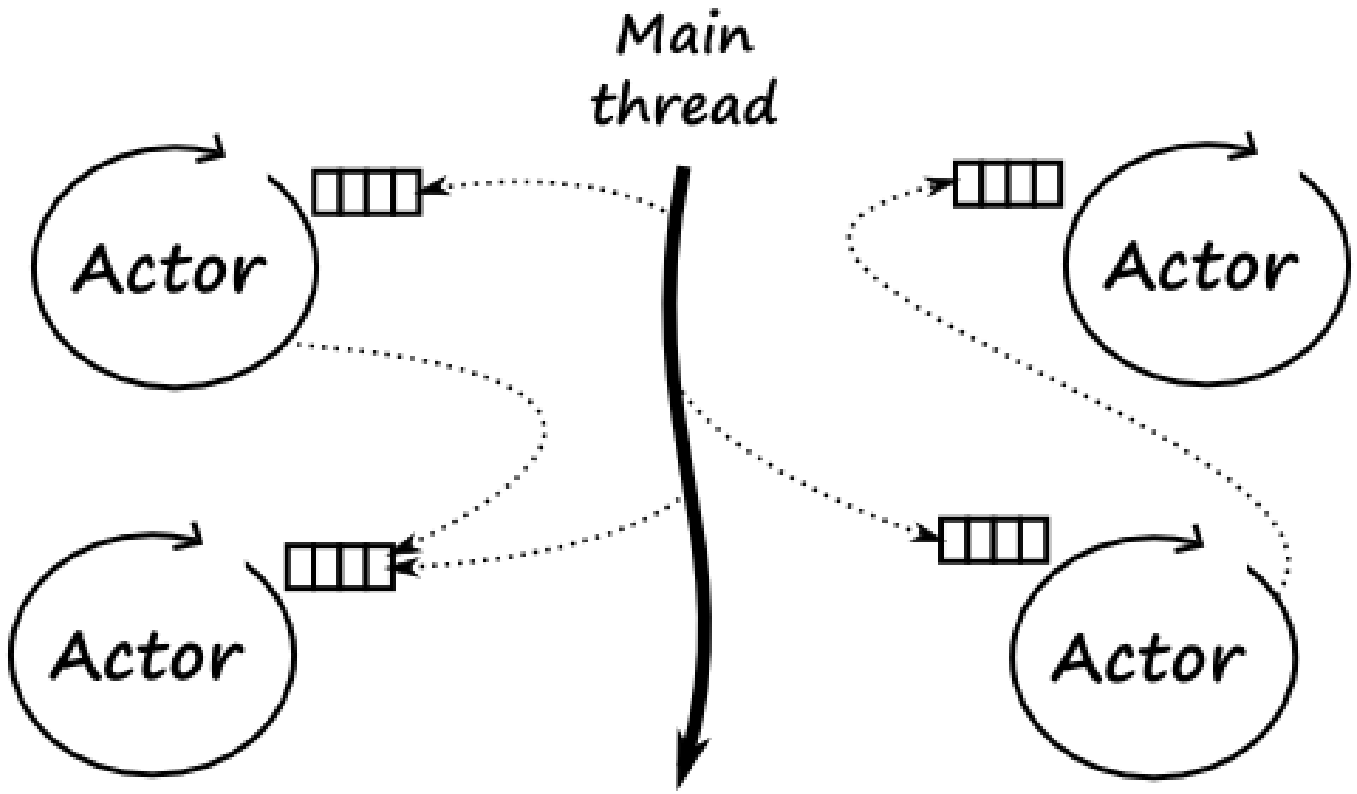
Alan Key and OOP

«... I thought of objects being like biological cells and/or individual computers on a network, only able to communicate with messages (so messaging came at the very beginning -- it took a while to see how to do messaging in a programming language efficiently enough to be useful).»

«... OOP to me means only messaging, local retention and protection and hiding of state-process, and extreme late-binding of all things.»

http://userpage.fu-berlin.de/~ram/pub/pub_jf47ht81Ht/doc_kay_oop_en

Actor Model



Actor Model

- The only way to interact with an actor is to send it a message
- Messages should be immutable
- All computation is in response to a message
- An actor is «potential energy». A message turns it into «kinetic energy»

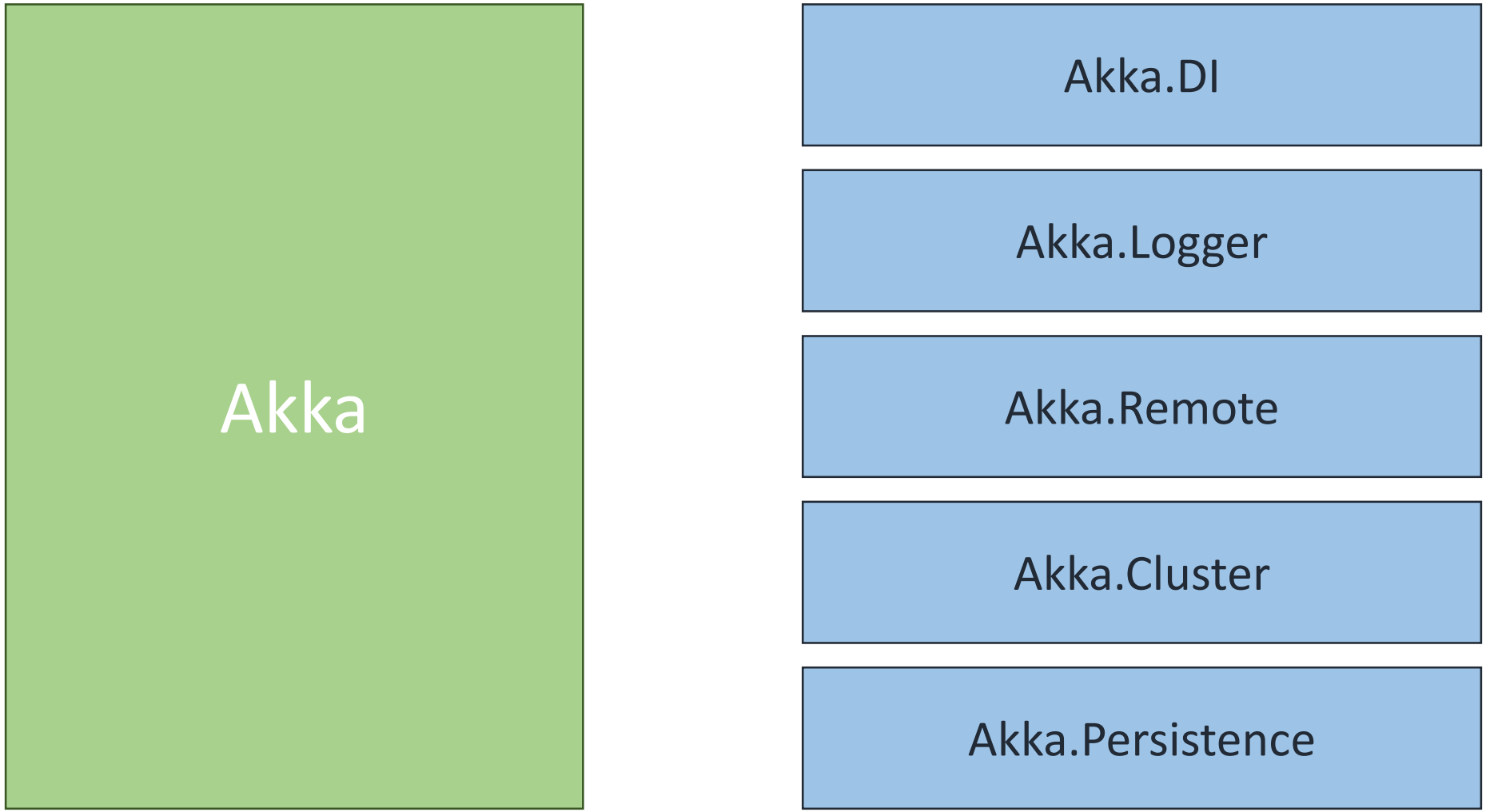
Implementations

- Erlang
- Akka (Scala / Java)
- Orleans
- Akka.Net *(why not NAkka?)*
- Reliable Actors (Azure Service Fabric)

Implementations

- Erlang
- Akka (Scala / Java)
- Orleans
- Akka.Net
- Reliable Actors (Azure Service Fabric)

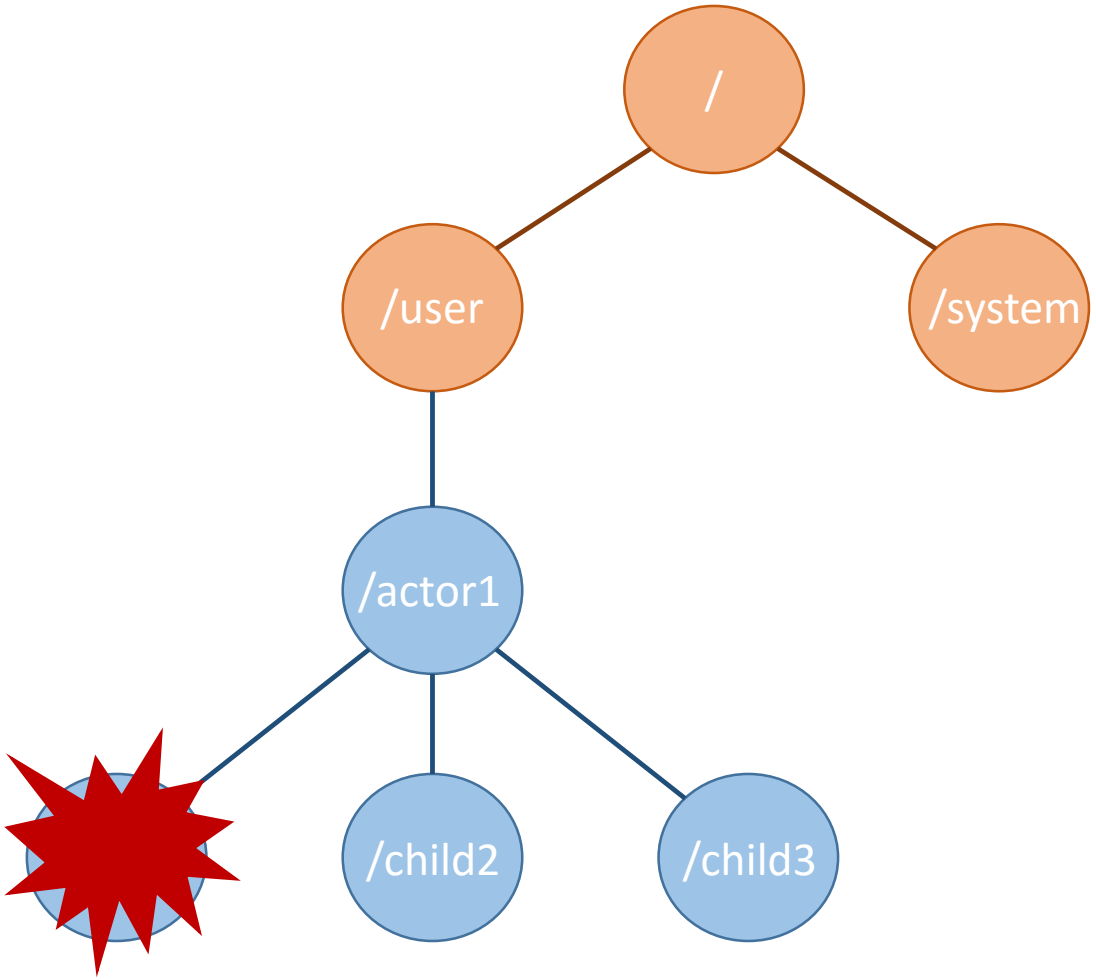
Akka.Net



1 - demo

Intro to Akka.Net

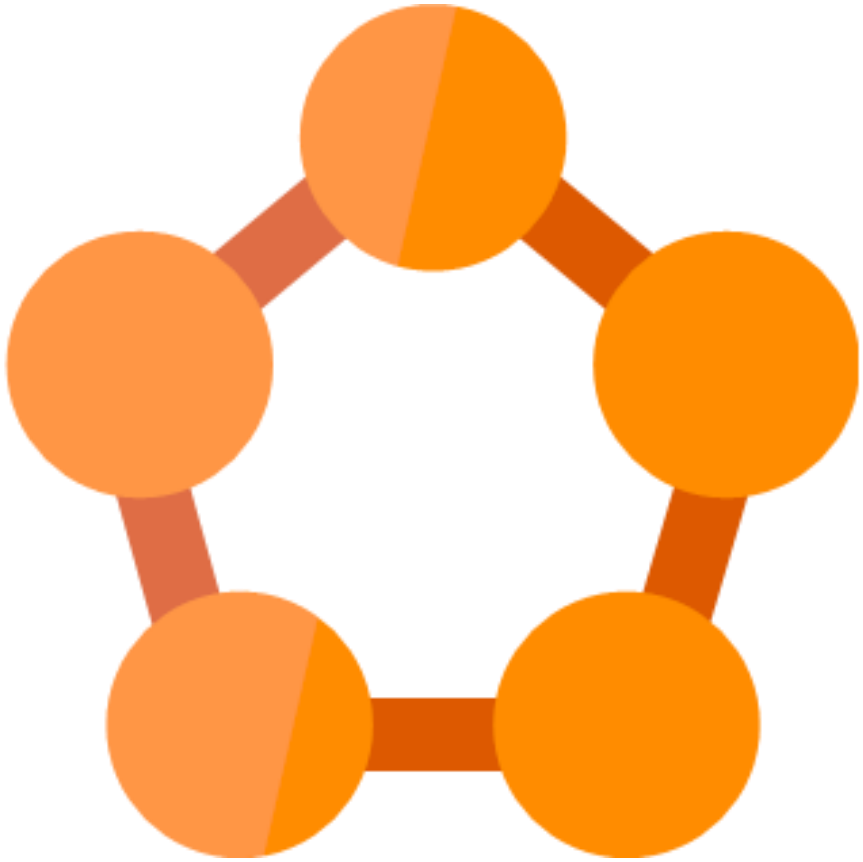
Akka.Net



- Hierarchy
- ActorSelection
- Supervisor Strategy

2 - demo

ActorSelection, ActorPath and SupervisorStrategy

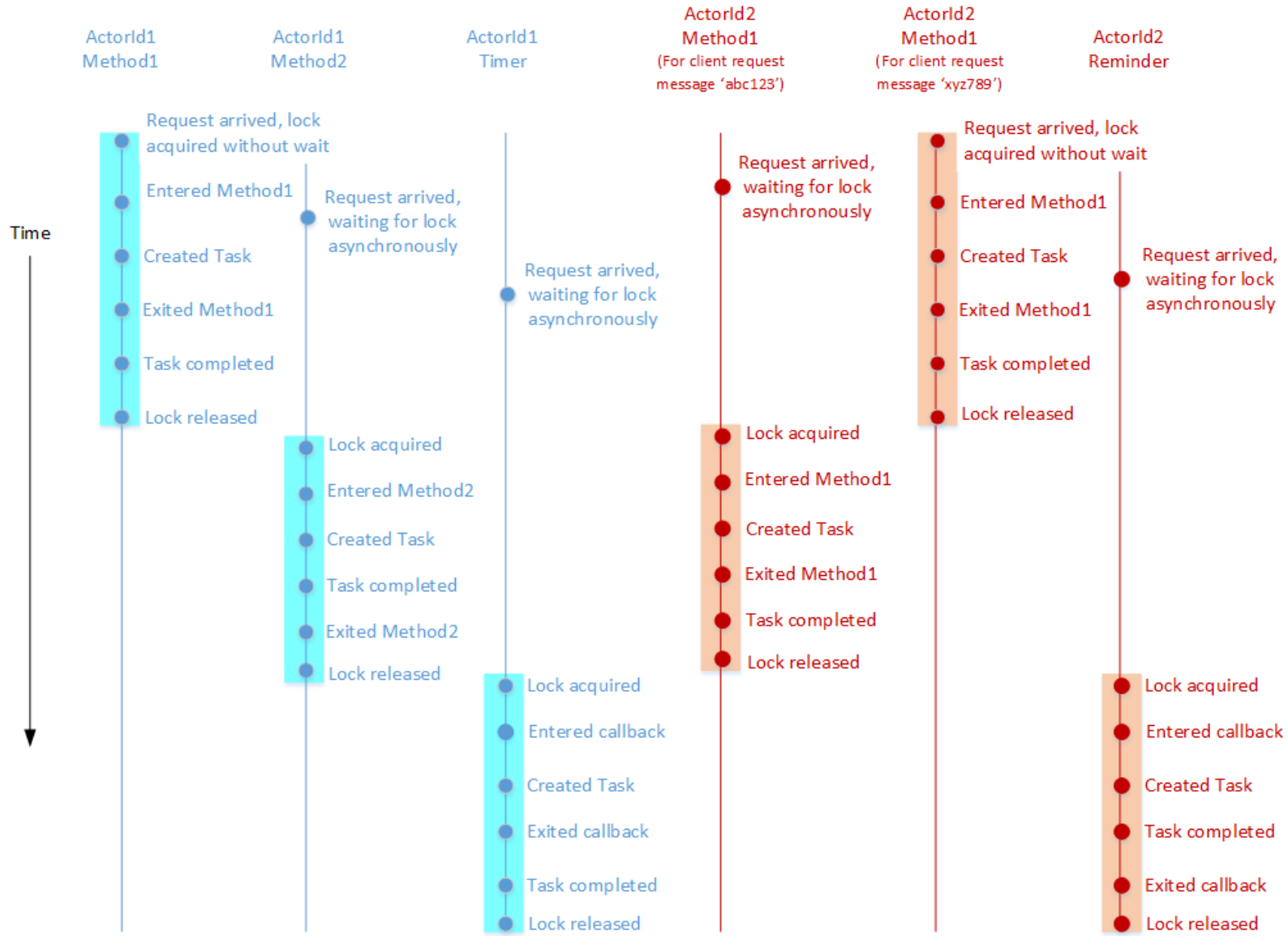


Reliable Actors
Azure Service Fabric

Reliable Actors

- *Virtual* Actor Model: actor's lifetime is not tied to their in-memory representation
- Each Reliable Actor service is a partitioned, stateful *Reliable Service*
- To provide scalability and reliability, *Service Fabric* distributes actors throughout the cluster and automatically migrates them from failed nodes to healthy ones as required

Reliable Actors



3 - demo

Intro to Reliable Actors

Thank you! Questions?

https://twitter.com/ugidotnet


HAPPY
Birthday!