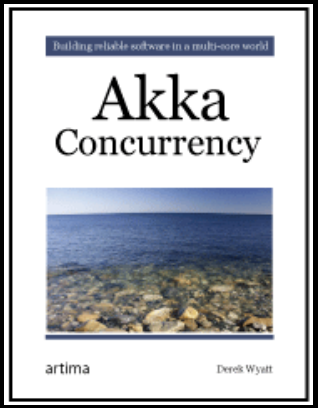
Actors

Correl Roush

October 21, 2015



Akka Concurrency

Derek Wyatt

- Concurrency through messaging
- Concurrency through delegation
- Delegation for safety
- Doing one thing at a time
- Reactive programming

You send an actor a message that tells it to do something, which it does presumably quickly and well, and then it tells you what it did. You can scale this model out to thousands or millions (or billions?) of actors and many orders of magnitude more messages and your applications are still reasonable, not to mention huge and fast.

There's nothing wrong with creating child actors for the sole purpose of putting them in harm's way. In fact, it's a very good thing. So don't be afraid of giving birth to an actor only to have him meet his ultimate demise micro-seconds later. He's more than happy to give his life in the service of his parent's good.

Actors only do one thing at a time; that's the model of concurrency. If you want to have more than one thing happen simultaneously, then you need to create more than one actor to do that work.

Actor programming is reactive programming. *Another way to say this is that it's* event-driven programming. *Event-driven programming has been with us for a long time, but it's arguably never been epitomized as much as with actor programming. The reason for this is that actors naturally sit there just waiting for something to happen (i.e., waiting for a message). It's not the act of* sending *a message that's important; it's the act of* receiving *one that really matters.*

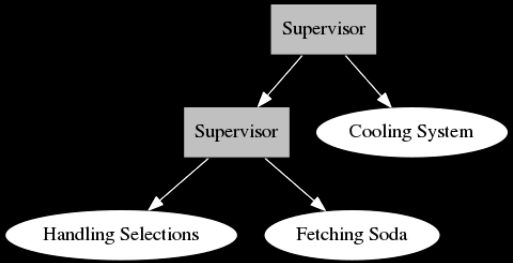
Designing for Actor Based Systems

Martin J. Logan

One process for each truly concurrent activity in the system.

VENDING MACHINE ACTIVITIES

- Putting coins into the slot
- Handling coins
- Handling selections
- Fetching the coke and putting it into the pickup tray
- Cooling the soda


Supervisor
Supervisor
Cooling System
Handling Selections
Fetching Soda

LANGUAGES AND TOOLS

Erlang Provides lightweight processes and transparent distribution

OTP Provides frameworks for actors and supervisors via callback modules.

Scala A hybrid functional language on the JVM

Akka a toolkit and runtime for building highly concurrent, distributed, and resilient message-driven applications