
akka

Акка — высочайшая гора в Швеции.

Технология до 2010 года — часть рантайма Scala.

Martin Odersky — автор языка, Scala

Coursera Scala курс:
https://www.coursera.org/learn/progfun1

Модель Акторов

Основная философия: Все есть Актор

Актор — вычислительная сущность, которая может отправлять
и принимать сообщения. Приняв сообщение, может:
* отправить сообщения
* создать других акторов
* изменить свое состояние

Актор обладает адресом, по которому он доступен для получения сообщений

Alan Kay, 2015: Power of Simplicity
https://www.youtube.com/watch?v=NdSD07U5uBs

Erlang

Agner Krarup Erlang

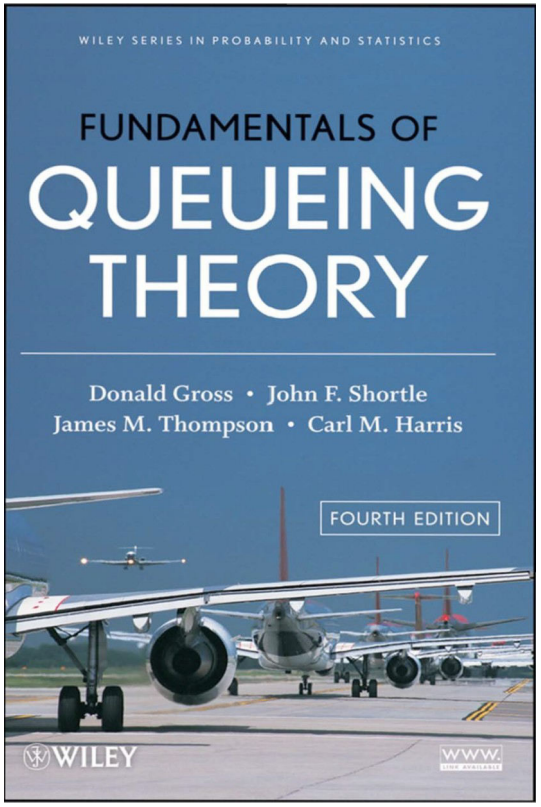
Erlang: The Movie
https://www.youtube.com/watch?v=uKfKtXYLG78

C10K problem → C10M problem

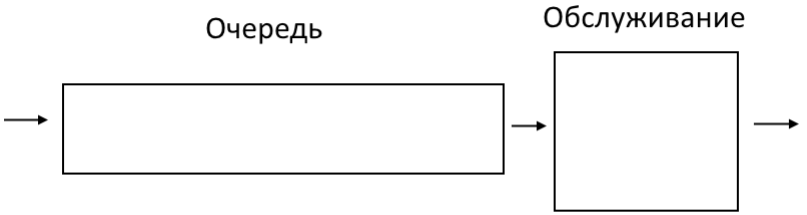
2010: Watsapp — 2M конкурентных соединений (Erlang, FreeBSD)

9 Nines:
https://en.wikipedia.org/wiki/High_availability#Percentage_calculation

ТМО: теория массового обслуживания
Queueing Theory



http://lmgtfy.com/?q=Fundamentals+of+queueing+theory



T нахождения заявки = T очереди + T обсулижвания

Формула Литтла
(метод вычисления производительсности СМО)

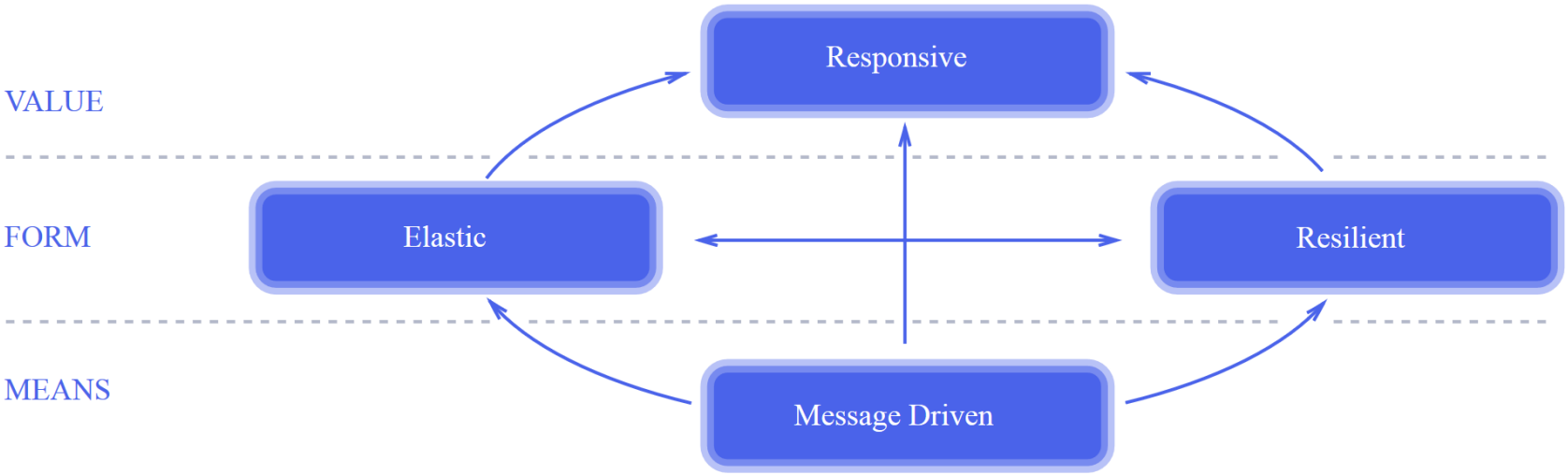
$L = \lambda W$

L — среднее количество заявок в системе
λ — поток входящих заявок
W — среднее время ожидания в очереди

$\rho = \lambda/\mu$ — интенсивность трафика

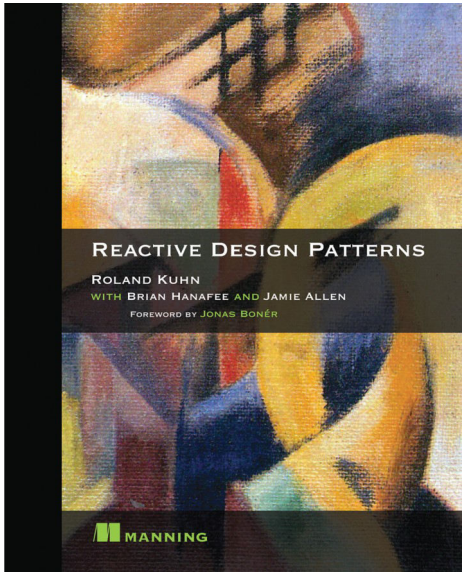
Reactive manifesto

https://www.reactivemanifesto.org/



Responsive	— Реагировать на пользователя.
Resilient	— Реагировать на отказы и оставаться доступной
Elastic	— Реагировать на изменения в нагрузке.
Message-driven	— Реагировать на поступающие запросы.

Reactive design patterns



http://lmgtfy.com/?q=Reactive+Design+patterns

The Simple Component pattern
Компонент должен делать только одну вещь, но делать ее полностью

The Error Kernel pattern
В супервизорской иерархии, супервизов отвечает за состояние приложения, делегируя рисковые операции подчиненным

The Let-It-Crash pattern
Начальное состояние — самое чистое. Лучше рестартовать, чем пытаться исправить

The Circuit Breaker pattern
Исключайте сервисных поставщиков, имеющих тендецию Фейлить

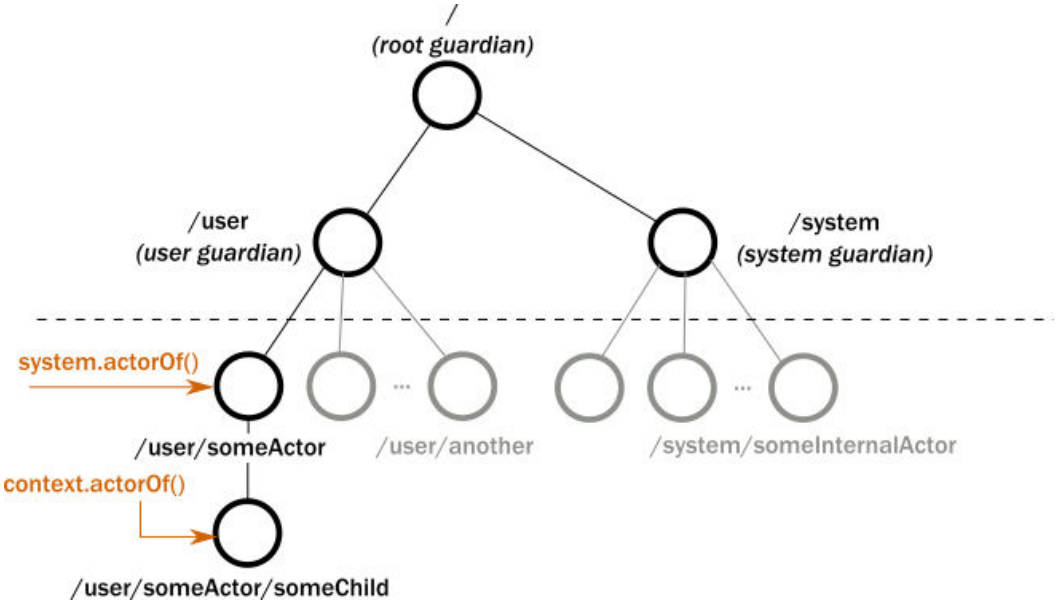
The Pull pattern
Получайте информацию о состоянии сервиса перед заливкой большого объема данных

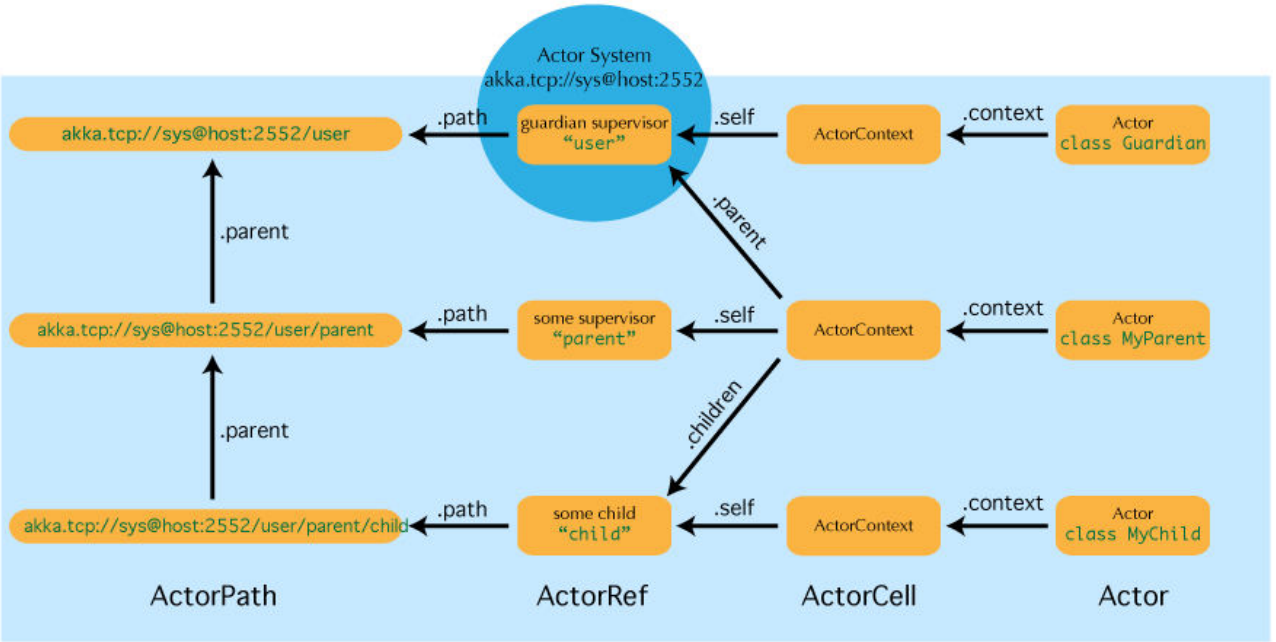
The Throttling pattern
Снижайте нагрузку на сервис в соответствии с контрактами

Akka Framework

Основные компоненты:

* Actors
* Fault tolerance
* Dispatchers
* Mailboxes
* Routing
* Persistence



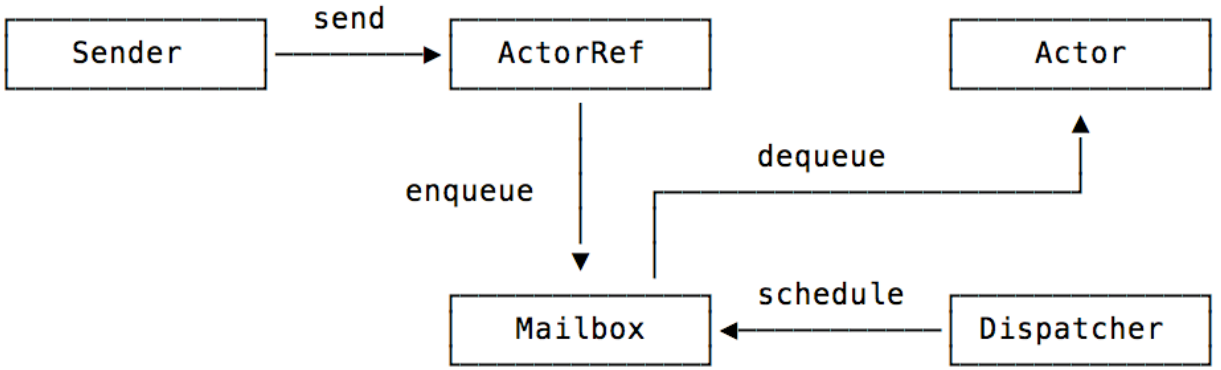


Доки:
https://doc.akka.io/docs/akka/current/index.html

Akka Framework

Основные компоненты:

* Actors
* Fault tolerance
* Dispatchers
* Mailboxes
* Routing
* Persistence



Доки:
https://doc.akka.io/docs/akka/current/index.html

Akka Streams

Основные компоненты:

Source	Входная точка, источник элементов для обработки
Flow	Компонет процесса обработки элементов
Materializer	Система акторов, влияющая на среду обработки данных в организованном потоке
Sink	Выходная точка, которая запускает поток

Доки:

https://doc.akka.io/docs/akka/2.5/stream/index.html

По теме

Netflix Zuul2. Async and CPU-bound tasks

https://medium.com/netflix-techblog/zuul-2-the-netflix-journey-to-asynchronous-non-blocking-systems-45947377fb5c

Alpakka

https://developer.lightbend.com/docs/alpakka/current/

Что почитать