Reactive Applications: What if Your Internet of Things has 1000s of Things?

dean.wampler@typesafe.com

@deanwampler

Photographs licensed from iStock unless otherwise noted.
Characteristics of Large IoT

- Large number of “nodes”.
- Potentially large number of messages to/from service providers and managers.
- Message sizes usually small.
- Resilience requirements vary.
Characteristics of IoT

• Response times from:
  • *Real time*: \(\mu\)-seconds for avionics.
  • *Human time*: 10s-100s of milliseconds.
  • *Phoning home*: no response or slow response okay.
• *Connectivity*: Intermittent to always on.
Examples
Med. Devices, IT Appliances

Photos:
http://www.oncolink.org/treatment/images/us-machine.jpg
Med. Devices, IT Appliances

- **Phone home** with status updates.
- Diagnose pending problems.
- Learn client usage patterns.
- Stable internet connection.

Most of the time, the messages are fire and forget. Round-trip messages might include queries for updates and subsequent downloads.
Trucks, Farm Equipment

image: http://upload.wikimedia.org/wikipedia/commons/6/60/Modern-tractor.jpg
Trucks, Farm Equipment

- *Phone Home* to report movements determined using GPS.
- Optimize routing.
- Spy on drivers?
- Occasional network.

The data can be uploaded in batch. It’s usually not required for real-time analytics.
Remote Sensors

Used to monitor earthquakes and nuclear test ban compliance. Another example is the growing network of Tsunami detectors.

Image: http://upload.wikimedia.org/wikipedia/commons/6/60/Modern-tractor.jpg
Remote Sensors

- *Human to Real Time*: trigger alert systems.
  - Earthquake warning systems.
  - Nuclear test pinpointing - test ban compliance.
- Reliable networks

The data can be uploaded in batch. It's usually not required for real-time analytics.
The Core Infrastructure

The case for a Reactive implementation.
The Core Infrastructure

Reactive - the system responds to events quickly, rather than driving system activity
Why a Manifesto?

• **Reactive** has been trending up:
  • Growing popularity of event-driven systems like Node.js, Erlang, Akka.
  • Evangelism: Erik Meijer, Jonas Bonér, Martin Thompson...
  • Define the “buzz word” preemptively...

[Typesafe logo]
Responsive

Scalable

Resilient

Event-Driven

reactivemanifesto.org

Typesafe
Event Driven

Reactive Applications scale up \textit{and} down on demand

• \textit{Asynchronous Programming}:  
  • Transparently leverage all cores on each CPU.  
  • Avoid resource contention; no blocking!  
  • Add/remove servers dynamically.
Event Driven

Reactive Applications respond to changes in the world around them

- *Messages* are passed between services and subsystems.
- *Asynchronous* and *non-blocking* throughout.
- You define the *workflow*; the runtime decides how to *schedule* those tasks.
Resilient

Reactive Applications are architected to handle failure at all levels

- **Bulkheads**: contain “damage”.
  - Within one process.
  - Within one server.
  - Within one datacenter.
  - …

Bulkheads are built into ships, for example, to contain leaks to a small section without compromising the whole ship. Firefalls perform a similar function.

Image: [http://axion.physics.ubc.ca/titanic/03_bulkhead.jpg](http://axion.physics.ubc.ca/titanic/03_bulkhead.jpg)
Scalable

Reactive Applications scale up and down on demand

• **Asynchronous Programming:**
  • Leverage all cores on a CPU.
  • Avoid resource contention.
  • Add/remove servers dynamically.

Typesafe
Scalable

Reactive Applications scale up \textit{and} down on demand

- \textit{Horizontal Scaling}:
  - Add servers, clusters.
  - \textasciitilde{}Linear performance, load gain?
Responsive

Reactive Applications are always available & provide low-latency responsiveness

• No SPOFs:
• No bottlenecks.
• Tuned for performance.
• Minimized latency.

Latency sources include garbage collection pauses, resource contention, network partitions, and bottlenecks.
So, a Reactive Application:

- Is *reactive* from top to bottom.
- *Decouples* event generation and processing.
- *Minimizes* the weakest link in the chain to match *Amdahl’s Law*.
For More…

- See this *Martin Thompson Interview*:
  - [infoq.com/interviews/reactive-system-design-martin-thompson](http://infoq.com/interviews/reactive-system-design-martin-thompson)
Typesafe Stack

- Actors are asynchronous and communicate via message passing
- Supervision and clustering in support of fault tolerance
- Purely asynchronous and non-blocking web frameworks
- No container required, no inherent bottlenecks in session management
- Asynchronous and immutable programming constructs
- Composable abstractions enabling simpler concurrency and parallelism
Typesafe Stack

- Actors are asynchronous and communicate via message passing
- Supervision and clustering in support of fault tolerance

- Purely asynchronous and non-blocking web frameworks
- Supervision and clustering in support of fault tolerance

- Asynchronous and immutable programming constructs
- Composable abstractions enabling simpler concurrency and parallelism
Typesafe Stack

- Actors are asynchronous and communicate via message passing
- Supervision and clustering in support of fault tolerance
- Purely asynchronous and non-blocking web frameworks
- No container required, no inherent bottlenecks in session management
- Asynchronous and immutable programming constructs
- Composable abstractions enabling simpler concurrency and parallelism
http://typesafe.com/platform/getstarted
Reactive Coursera Course

Principles of Reactive Programming

coursera.org/course/reactive

Started November 4th,
7 weeks long
Thank You!
Questions?

dean.wampler@typesafe.com
@deanwampler