Embedded Actors — Towards Distributed Programming in the IoT

Raphael Hiesgen, Dominik Charousset, Thomas C. Schmidt

HAW Hamburg

raphael.hiesgen@haw-hamburg.de

September 2014, ICCE-Berlin
Agenda

1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
The Internet of Things (IoT)

- Characteristics
  - Cooperatively process complex duties
  - Dependent on machine-to-machine communication
  - Connected to the Internet
  - Implemented with networking standards
  - Often constrained environment

- Developing for the IoT
  - Fall back to low-level programming
  - Hand-crafted network code
  - Barely portable code

⇒ Raise the level of abstraction through the actor model
Agenda

1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
The Actor Model

- Isolated, concurrent software entities: actors
- Network-transparent message passing
- Divide & conquer via “spawn”
- Strong, hierarchical failure model
- Re-deployment at runtime
Limitations

- Actors not yet established in the native programming domain
  - Need to broaden range of applications
  - Deploy actors in performance-critical systems

- Actors not available for embedded systems
  - Why not model the “Internet of Things” as network of actors?
  - HW platform should not dictate programming model
The C++ Actor Framework

- The C++ Actor Framework (CAF) is an C++11 actor system
- Previously named libcppa
- Efficient program execution
  - Low memory footprint
  - Fast, lock-free mailbox implementation
- Targets both high-end and low-end computing
  - Multi-core & many-node systems
  - Embedded HW, e.g., running RIOT\textsuperscript{1}

\textsuperscript{1}http://www.riot-os.org
Memory Usage

![Box plot diagram showing memory usage for different programming languages. The diagram compares CAF, scala, and erlang, with measurements in MB. The box plot includes the 5th percentile, median, mean, 25th percentile, 75th percentile, and 95th percentile.](image-url)
Agenda

1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
Communication in the IoT

- Loosely coupled
  - Unreliable links
  - Infrastructure failure
- Challenges
  - Error propagation for non-hierarchical systems
  - Secure and authenticated connectivity
- Multiple open standards available to meet these challenges
Standards for the IoT

- Bluetooth low energy
  - Supported by all major OSs
- IEEE 802.15.4
  - WPAN for embedded devices
  - 127 bytes frame size
- IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)
  - IPv6 compatibility
  - Header compression
- UDP / Datagram Transport Layer Security (DTLS)
  - Features of TLS with datagrams
  - Reordering, retransmission and fragmentation for the handshake
- Constrained Application Protocol (CoAP)
  - Request-response model adapted from HTTP
  - Works asynchronously over datagram protocols
  - Offers reliability through Confirmable messages (CON)
Agenda

1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
CAF Network Stack

C++ Actor Framework

TCP
IPv4 / IPv6
Ethernet / WLAN

CoAP
DTLS
UDP
6LoWPAN
802.15.4 / Bluetooth LE
Messaging Architecture Between Actors

- Map CoAP messages to CAF
  - Reliability (CON) → synchronous messages
  - Unreliability (NON) → asynchronous messages

- Handle small frame sizes
  - Compress meta-information to slim down headers
  - Type-exchange & annotation introduces more state
  - Fragmentation on the application layer (CoAP block messages)

- Concept for error-propagation
  - No longer connection oriented
  - Based on asynchronous transactions (CoAP)
  - Take unreliable messages into account
Agenda

1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
Packet Flow Evaluation

- Proof-of-concept implementation
  - Network stack composed of Ethernet, UDP and CoAP
  - Based on libcoap \(^2\)
- Bitrate & message count compared to TCP based impl.
- Raspberry Pi sends 10 bytes to a desktop PC every 100 ms
  - Characteristic scenario for sensor nodes
- Trace network traffic (handshakes, ACKs, messages, ...)

\(^2\)http://libcoap.sourceforge.net
Packet Flow
Packet Size Distribution

The diagram shows the packet size distribution for CoAP + UDP and TCP. The x-axis represents the packet size in bytes, ranging from 0 to 260. The y-axis shows the count of packets. The diagram includes bars for both CoAP + UDP and TCP, with CoAP + UDP represented by darker gray bars and TCP by white bars. The distribution for CoAP + UDP has a peak at 80 bytes, while TCP shows a peak at 140 bytes.
1. The Internet of Things (IoT)
2. The Actor Model
3. Communication in the IoT
4. Messaging Architecture
5. Evaluation
6. Conclusion & Outlook
Conclusion & Outlook

- Development for the IoT requires specialized knowledge
  - Network communication & synchronization
  - Port software to different hardware
- The actor model abstracts over distributed systems
- Adjusted the CAF network stack to the IoT
- Proof-of-concept with first measurements

Future work
- Concept for error-propagation
- Additional CoAP drafts (CoCoA, block messages, ...)
- Evaluation of packet loss, message sizes
- Adaption to RIOT
Thank you for your attention!

Website: http://actor-framework.org

Sources: https://github.com/actor-framework

iNET working group: http://inet.cpt.haw-hamburg.de