unic²ast

Simulation and Benchmarking of IoT Device Usage Scenarios Using Zephyr and Qemu

Steffen Späthe, Bill Schirrmeister, Frank Geyer

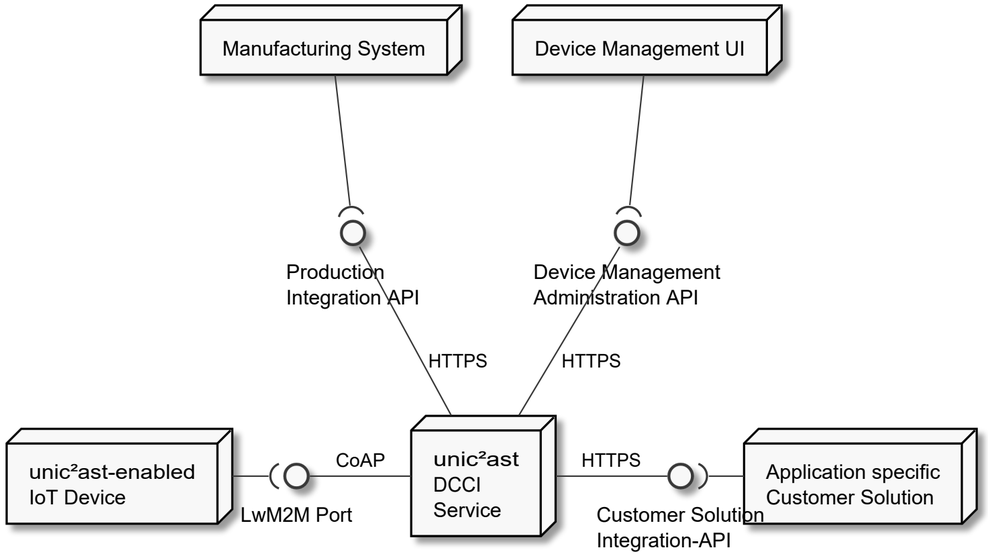
Overview

- Project "unic²ast"
- The Challenge
- Approaches For Load Testing Setup
- Evaluation of our Approach
- Conclusion

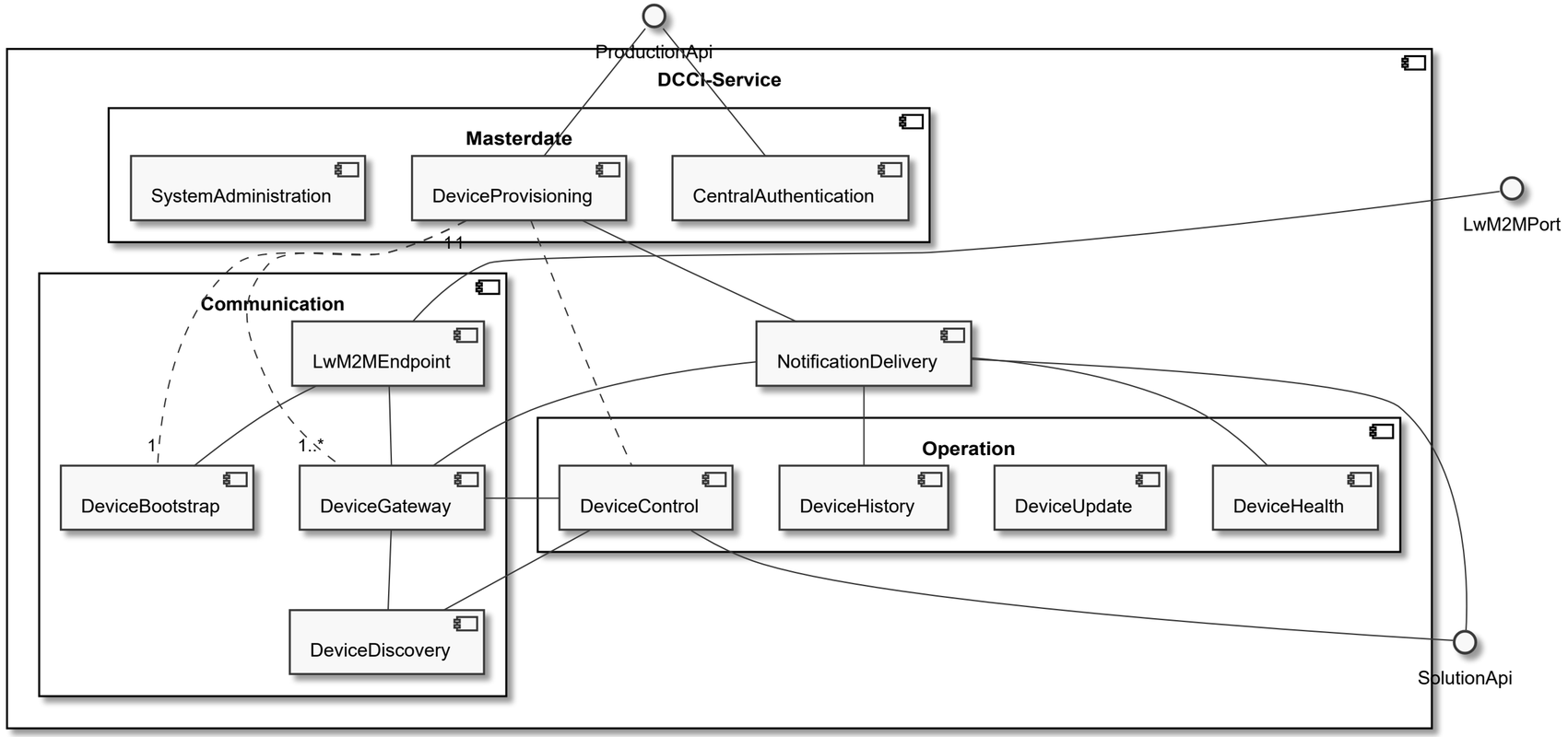
unic²ast

- UNIversal Command and Control infrASTructure
- Function block for IoT devices
- Application-agnostic backend

unic²ast Context



unic²ast Architecture (Level 1)



unic²ast dcci service - whitebox

The Challenge

How to proof the capabilities?

Our quality goals:

- Manage thousands of devices in parallel
- Handle at least 10 messages per device per second

Approaches For Load Testing Setup

- Network simulators (ns-3, OMNeT++)
- Real devices
- Virtual devices
  - Synthetic client devices
  - Emulated devices

Network Simulators

Approach: Implement server and devices inside a network simulator

- Non-realtime, event based simulation
- Scalability is primarily limited by processing time
- BUT:
  - Lots of work to implement simulation (server and devices)
  - Statistics about the real behavior of service internal components **does not exist**
- Conclusion: **We need to use the real server.**

Synthetic Client Devices

Approach: Implement a software, which behaves like a real device

- Some work in high level programming language
- BUT: *no* correlation between simulated and real device behaviour

Real Devices

Approach: Use real devices, they ~~behave like~~ are real devices

- Best behavior matching
- Minimal implementation effort
- BUT: *not* scaleable

Emulated devices

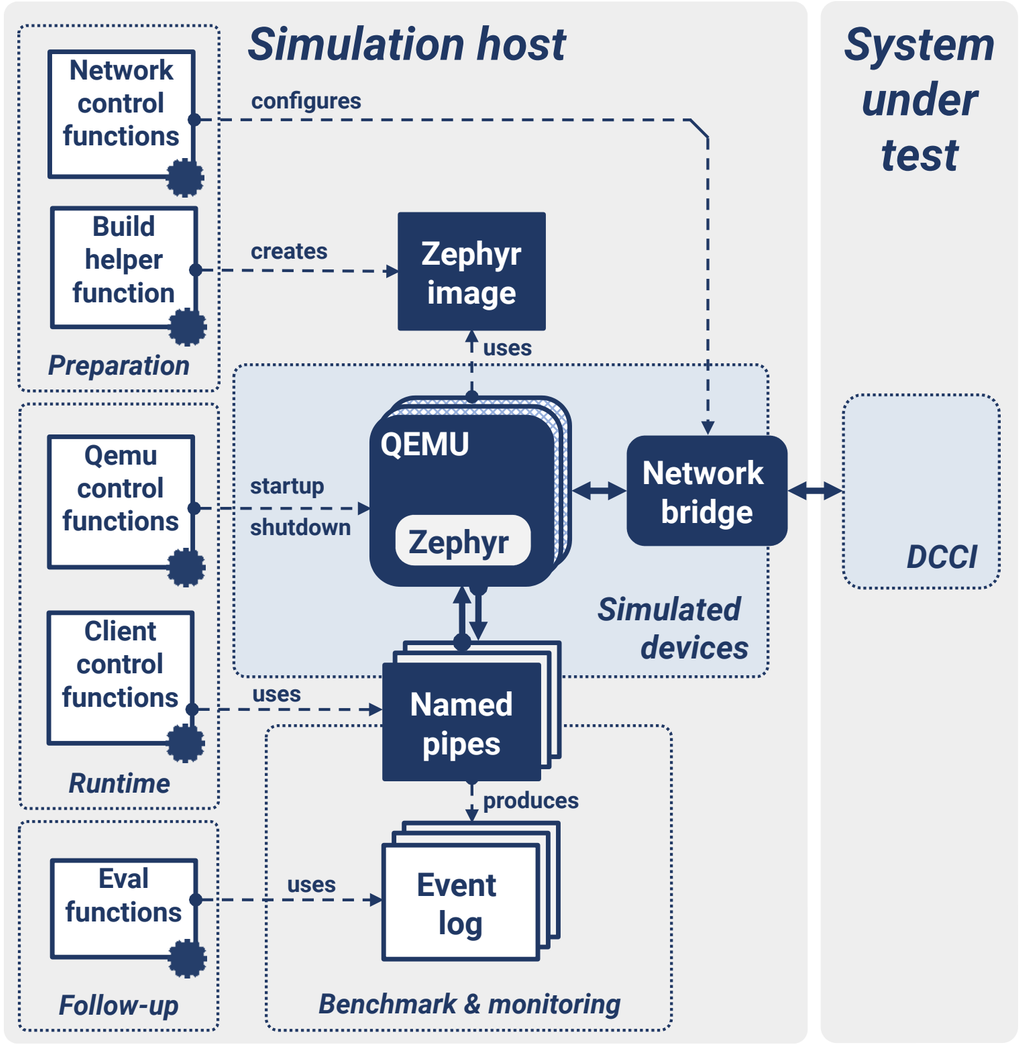
Approach: Use real software but emulated devices

- Emulated devices behave similar to original devices
- Ability to run multiple simulated device instances on one host
- Real LwM2M connection to the system under test
- Easy parameterization of the device instances
- Retrieval of runtime information after startup
- Automation of the device instance lifetime and the test procedure

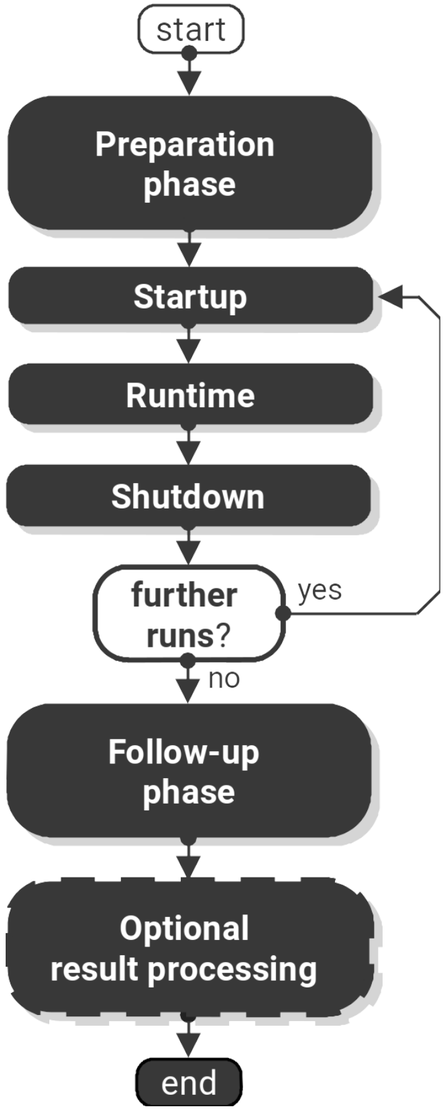
Emulated Devices

- **Zephyr OS** based firmware for IoT devices
- Zephyr OS supports a wide variety of CPU architectures and embedded boards
- With **qemu_x86** usable via x86-Emulation on Linux hosts
  - Network via SLIP and QEMU Ethernet

Simulation Landscape



Coordinator Workflow



Evaluation: Emulated Devices

- Zephyr and qemu_x86 work good together
- Some obstacles with network setup (QEMU x86)
- Quite memory intensive (per 16MB RAM device about 25MB host RAM)
  - 8 core CPU, 32GB Host runs about 1.000 emulated devices

unic²ast

Evaluation: Bash Based Coordinator

- Easy proof-of-concept approach
- Worked out quite well
- PoC needs some refactoring efforts
- Lack of integrated data acquisition and processing functions

Conclusion

- Our chosen approach based on Bash-based orchestration which controls emulated devices with Zephyr OS works
- Bash is not the best way to orchestrate performance tests
- Zephyr and QEMU is a great combination
- To increase the number of emulated devices multiple hosts support is required

Thank you very much for your interest.

Steffen Späthe, Bill Schirrmeister, Frank Geyer

steffen.spaethe@uni-jena.de