Design and implementation of configuration synchronization and WebGUI for decentralized multi LiDAR setups

Uday Navaneeth Menon
Advisor: Johannes Meßmer
Chair of Decentralized Systems Engineering
https://dse.in.tum.de/

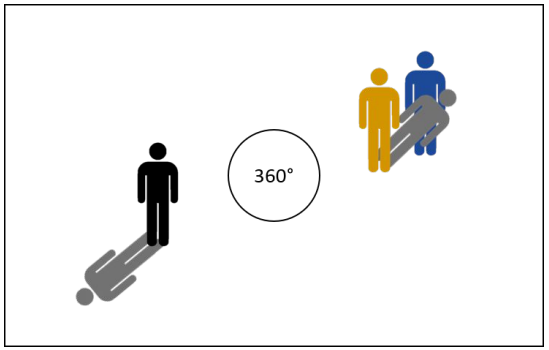
15.12.2021 – 15.08.2022

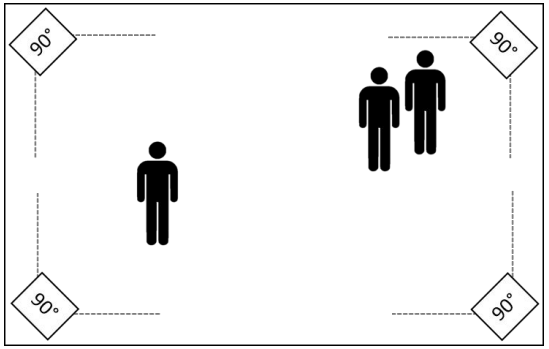
Outline

- **Motivation**
- Overview
- Background
- Design
- Implementation
- Summary
- Future work

Motivation: Research context

- Current LiDAR technologies
  - Mechanical rotating LiDAR
  - Solid state LiDAR
- Distributed LiDAR system (Swarm)
- Possible applications of Swarm
  - People counting
  - Volume measurement
  - Perimeter security
  - Traffic management





Problem statement

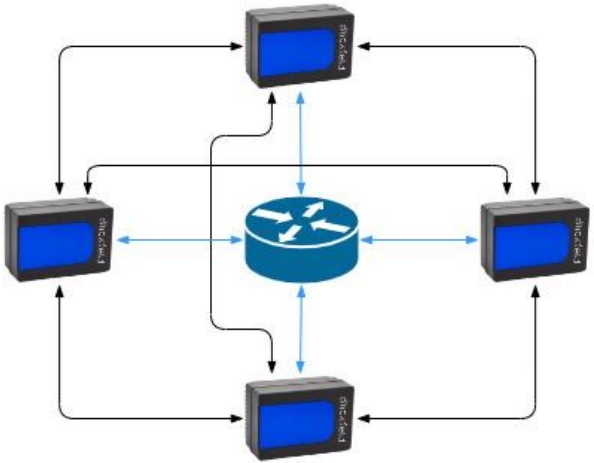
- Configuration synchronization
  - Discovery service to search the network for LiDAR devices and Swarm clusters
  - Configuration service to synchronize configuration of all devices and maintain homogeneity of cluster
- WebGUI
  - Designing a modular system to help improve scalability
  - Component based interface design to facilitate reusability and a streamlined process of adding new elements
  - User experience to enable users to work with the device with little to no training needed

Outline

- Motivation
- **Overview**
- Background
- Design
- Implementation
- Summary
- Future work

Overview

- Blickfeld Qb2 [2]
  - Solid-state LiDAR
  - On-device processing
  - Software defined LiDAR
- Swarm
  - Distributed LiDAR cluster
  - Same physical location
  - Combined output

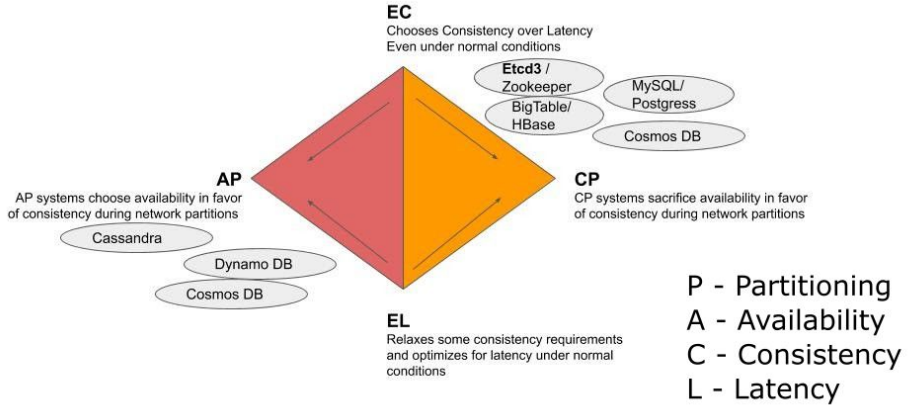


Outline

- Motivation
- Overview
- **Background**
- Design
- Implementation
- Summary
- Future work

Background

- Network discovery
  - Zero configuration networking
  - Multicast Domain Name System
- Distributed key-value storage
- grpc Remote Procedure Call



PACELC representation of key-value storage technologies [1].

Outline

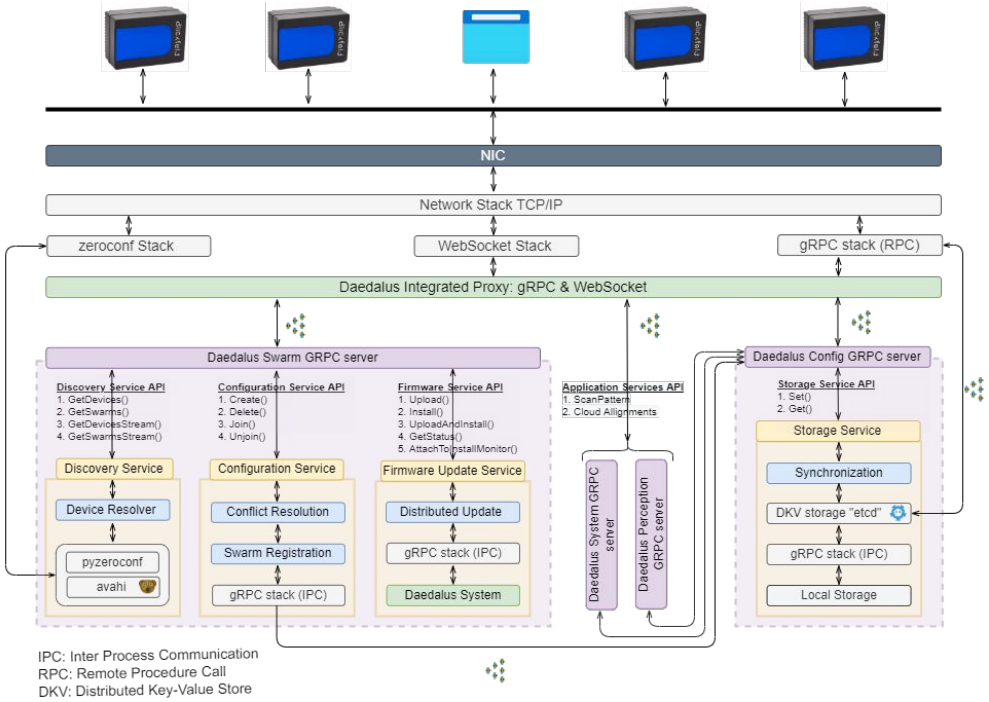
- Motivation
- Overview
- Background
- **Design**
- Implementation
- Summary
- Future work

Design overview: Swarm

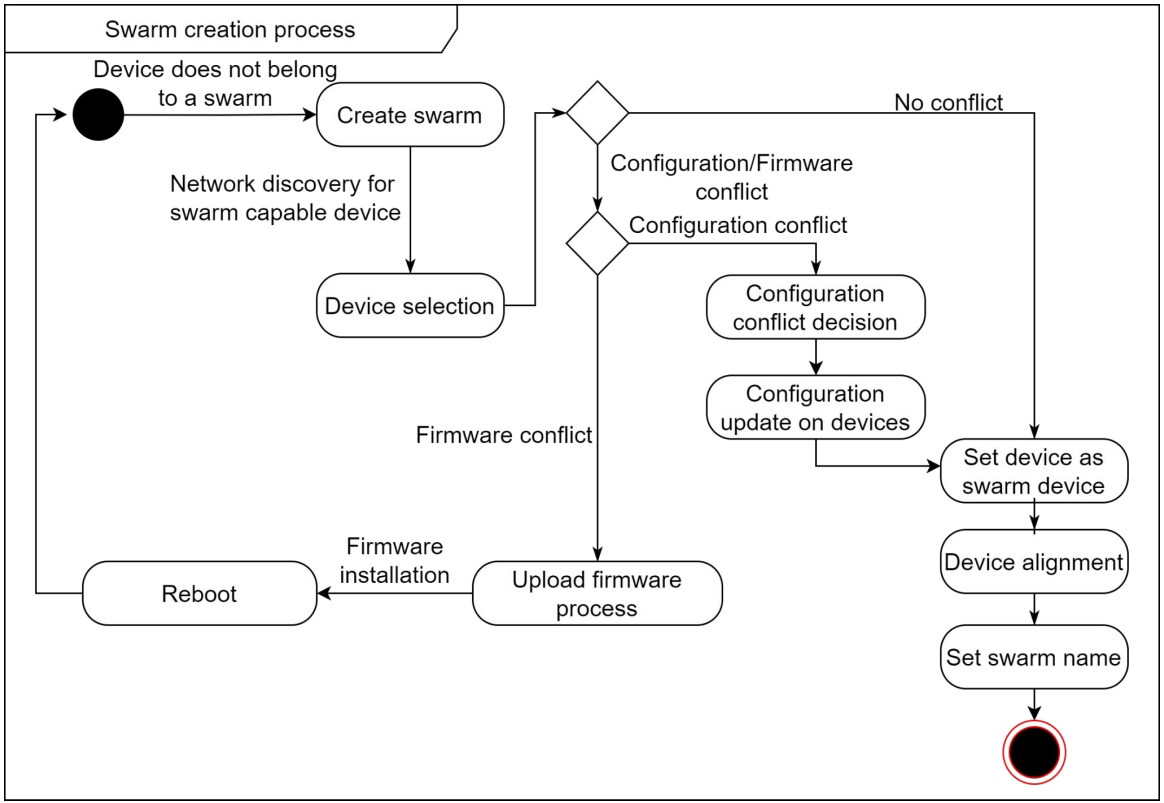
- Swarm core functions
  - Registration
  - Alignment of point clouds
  - Configuration synchronization
  - Distributed configuration storage
  - Maintenance
- Swarm rules
  - Number of devices
  - Positioning
  - Connection
  - Configuration

Design overview: Swarm

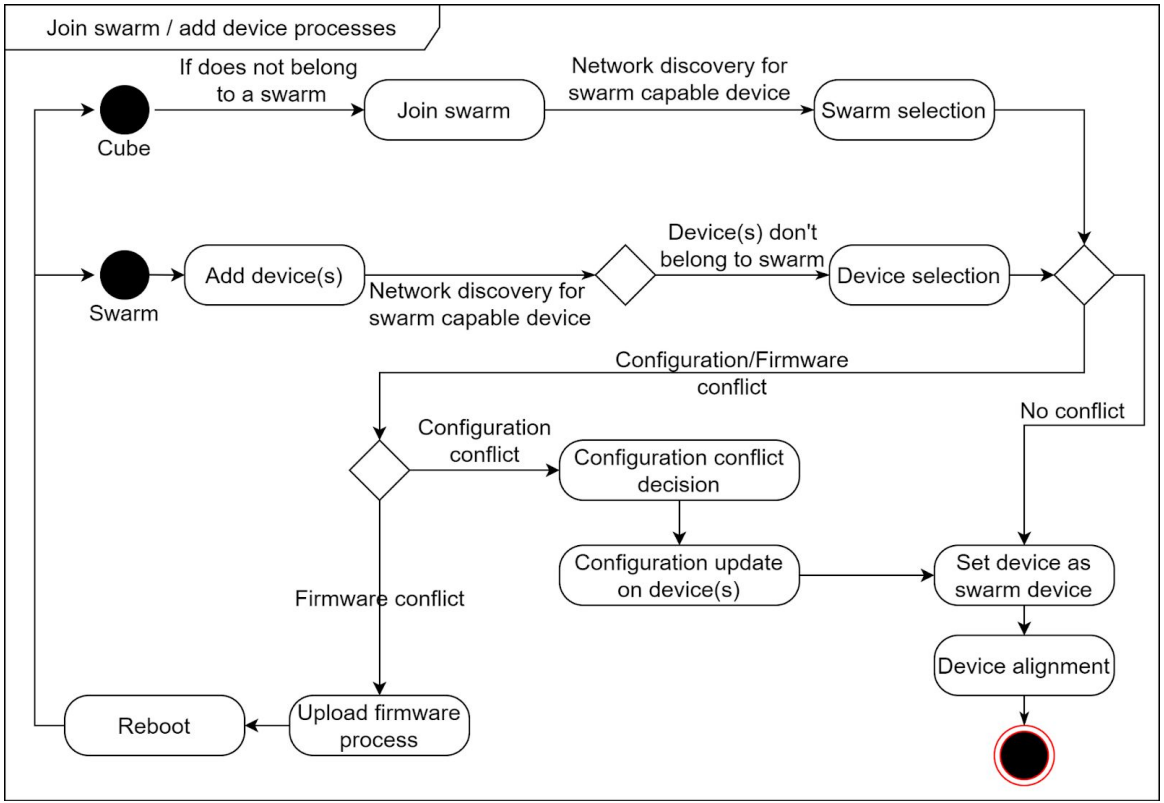
- Discovery service
  - Python zeroconf
- Configuration service
  - Swarm identifier
  - Distributed configuration storage
  - Time synchronization
  - Frame synchronization
  - Scan pattern settings



Swarm workflows

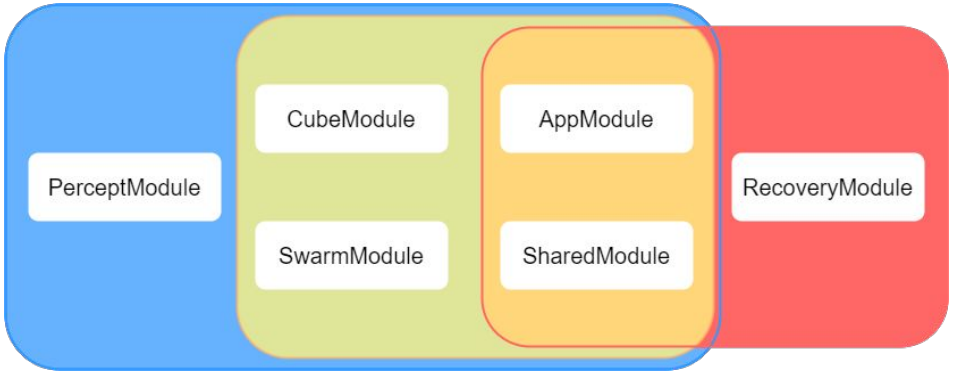


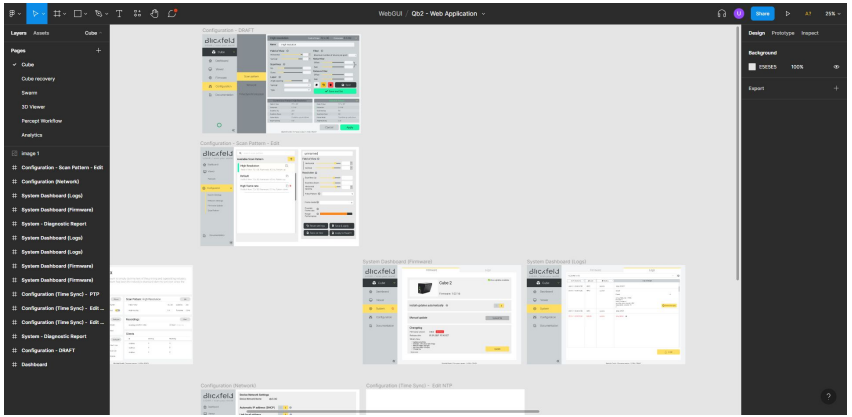
Swarm workflows



Design overview: Web-GUI

- Design goals
  - Modular structure
  - Development rules
  - Plugin based architecture
- Design process
- User experience (UX)
- User interface (UI)



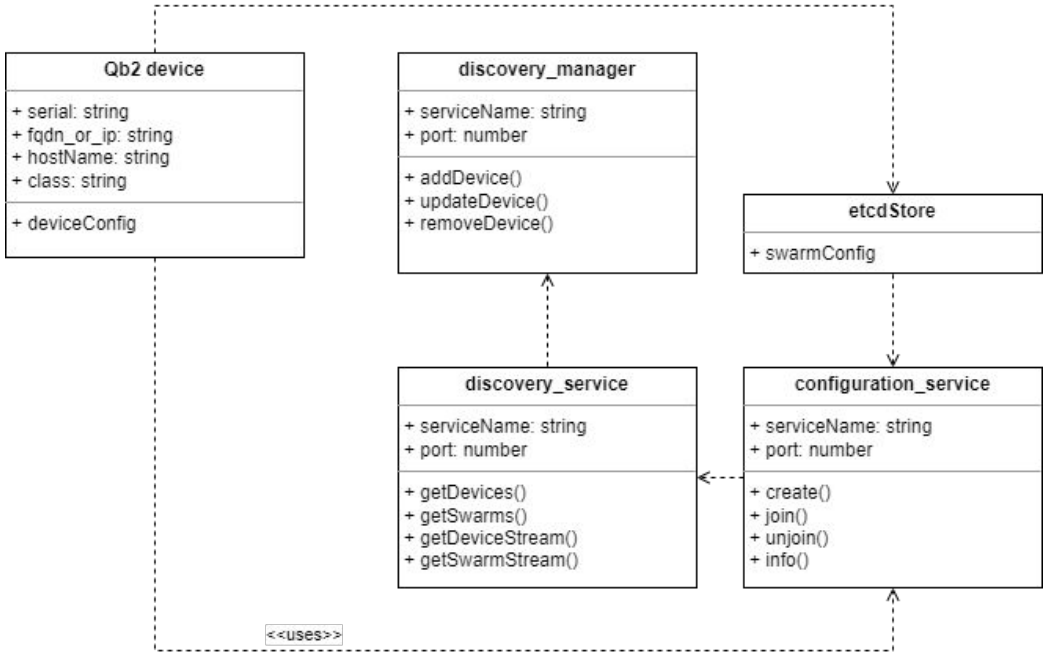


Outline

- Motivation
- Overview
- Background
- Design
- **Implementation**
- Summary
- Future work

Implementation: Configuration synchronization

- Discovery service
  - Add device
  - Remove device
  - Update device
- Configuration service
  - Create
  - Join and Unjoin
  - Configuration synchronization



Result

These results were obtained by using gWhisper.

```
| devices[4/6] = {DevicesEntry}
| | key.. = "256014ac131b4238b0ce754dd213a4be"
| | value = {Device}
| | | fqdn_or_ip... = "172.19.0.5"
| | | name......... = "qemux86-76.local."
| | | serial_number = "256014ac131b4238b0ce754dd213a4be"
| | | class........ = QB2
| devices[6/6] = {DevicesEntry}
| | key.. = "5cdd5c9bbe2a49a88fee7c95ba5d476a"
| | value = {Device}
| | | fqdn_or_ip... = "172.19.0.2"
| | | name......... = "qemux86-77.local."
| | | serial_number = "5cdd5c9bbe2a49a88fee7c95ba5d476a"
| | | class........ = QB2
```

List of devices on network: **blickfeld.swarm.services.Discovery GetDevices**

Discovery service returns all the Qb2 devices and Swarms that exist on the network

```
Received message:
| clusters[0/0] = {}
```

List of Swarms on network:
blickfeld.swarm.services.Discovery GetSwarms

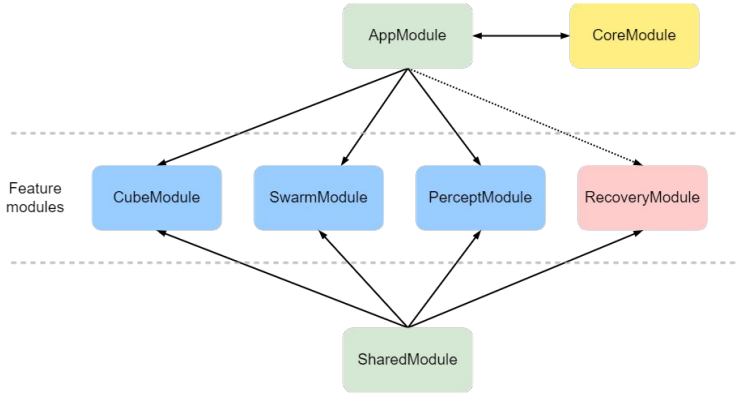
Using the
“blickfeld.swarm.services.Cluster Create”
service

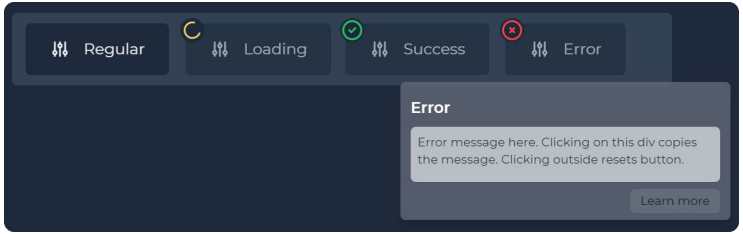
```
| clusters[1/1] = {ClusterInfo}
| | members[1/2] = {ClusterMember}
| | | id = "5cdd5c9bbe2a49a88fee7c95ba5d476a"
| | | ip = "172.19.0.2"
| | members[2/2] = {ClusterMember}
| | | id = "256014ac131b4238b0ce754dd213a4be"
| | | ip = "172.19.0.5"
```

List of Swarms on network after swarm is created

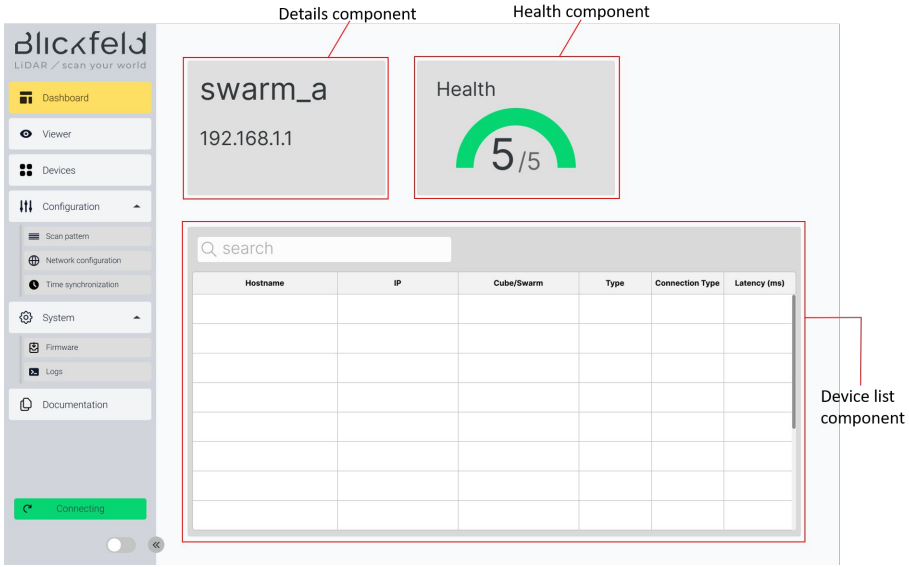
Implementation: WebGUI

- Technologies
  - Angular
  - Typescript
  - Tailwind CSS
- Connection
  - Websocket gRPC proxy
  - Routing
- Dynamic components

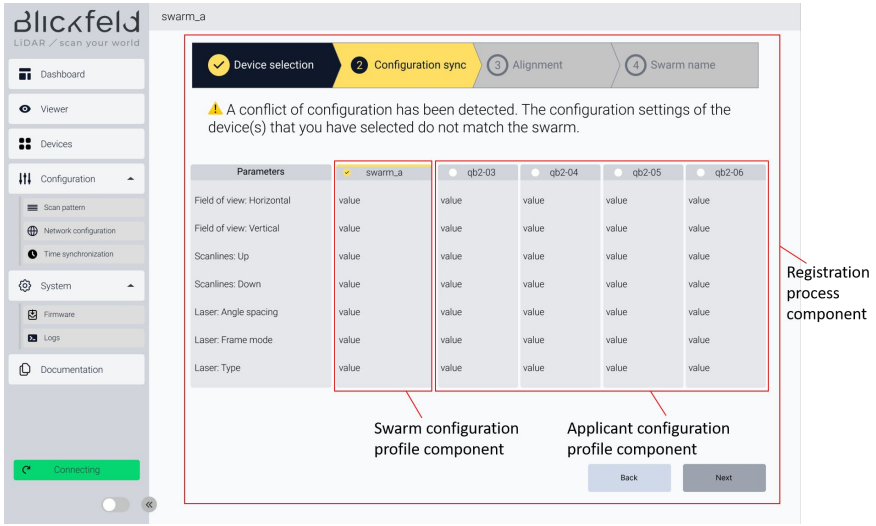




Result



Swarm dashboard



Configuration conflict resolution

Outline

- Motivation
- Overview
- Background
- Design
- Implementation
- **Summary**
- Future work

Summary

The proposed concept of decentralized LiDAR system (Swarm) provides a scalable and flexible solution for combining outputs from multiple LiDAR devices

In this project the focus was on

- Designing initial architecture
- Discovery service
- Configuration service
- Modular web interface
- Dynamic components

Outline

- Motivation
- Overview
- Background
- Design
- Implementation
- Summary
- **Future work**

Future Work

- Combining point cloud data
- Perception processor

References

[1] D. Ongaro and J. Ousterhout. “In Search of an Understandable Consensus Algorithm (Extended Version).” In: Stanford University.

[2] Blickfeld GmbH. Qb2. https://www.blickfeld.com/lidar-sensor-products/qb2/