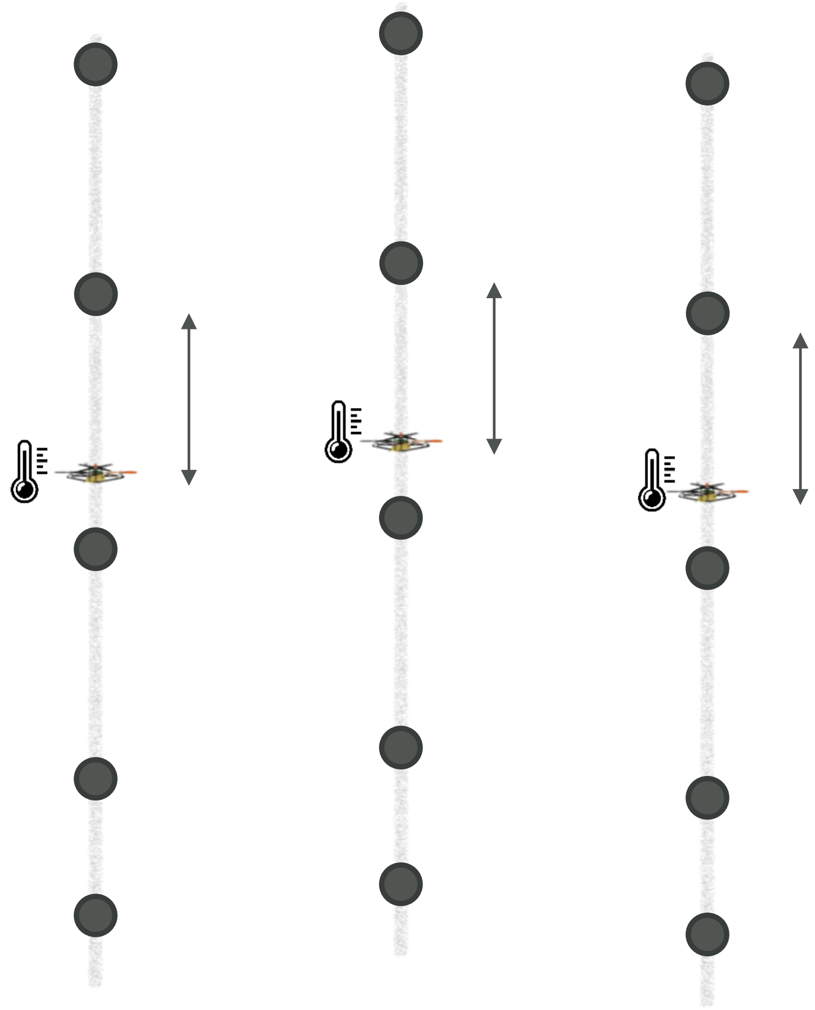
Modeling and Execution of Coordinated Missions in Reconfigurable Robot Ensembles

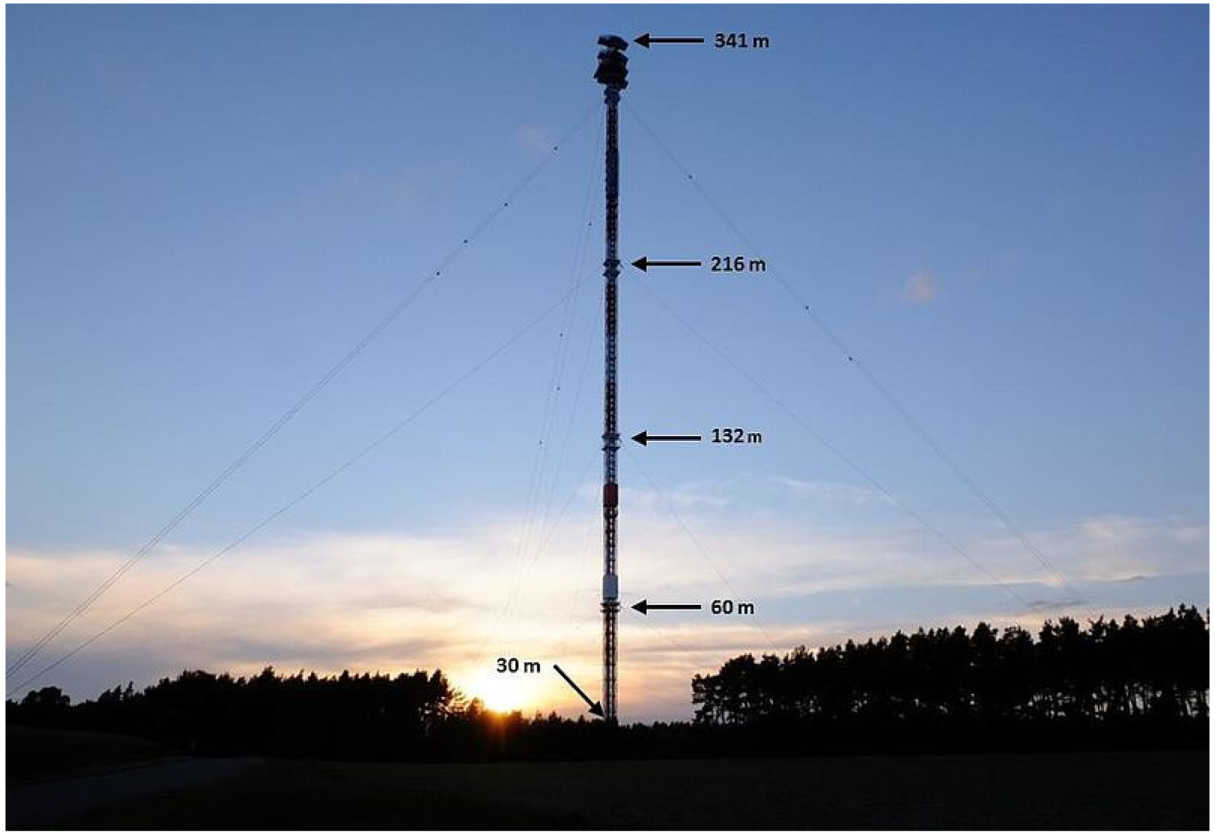
Martin Schörner, Constantin Wanninger, Alwin Hoffmann, Oliver Kosak, Hella Ponsar, Wolfgang Reif

Institute for Software & Systems Engineering

Universität Augsburg University

Motivation: ScaleX 2015 2.0



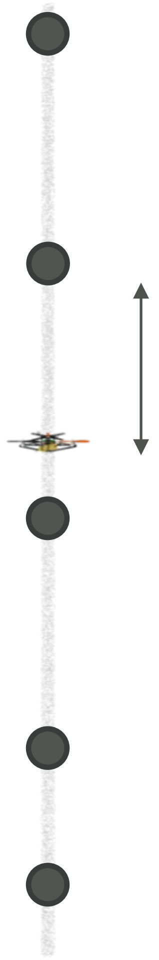


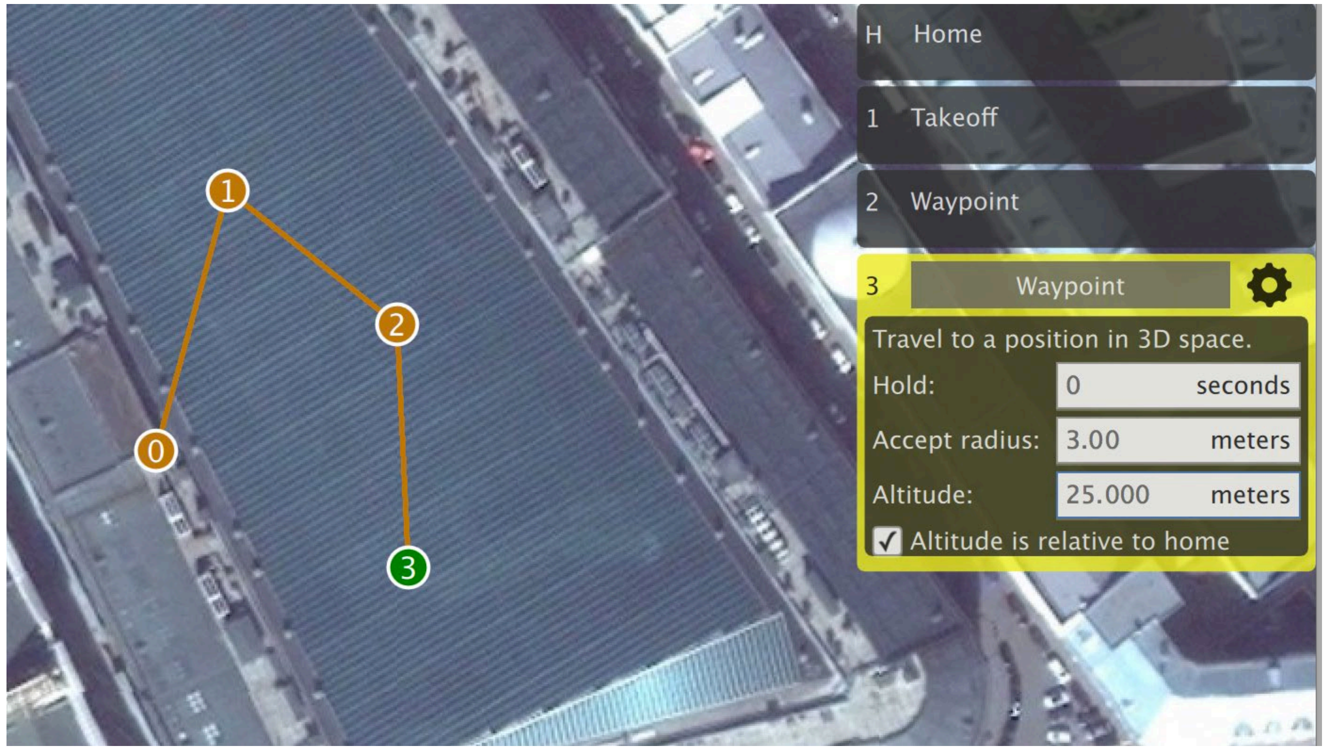
Mission Planning: Static missions

Institute for Software & Systems Engineering

Functionality of missions limited to the feathures that the Flight Controller and the Ground Control Station (GCS) offer (Simple Waypoint Missions)

No support logging of sensor data





Travel to a position in 3D space.

Hold: 0 seconds

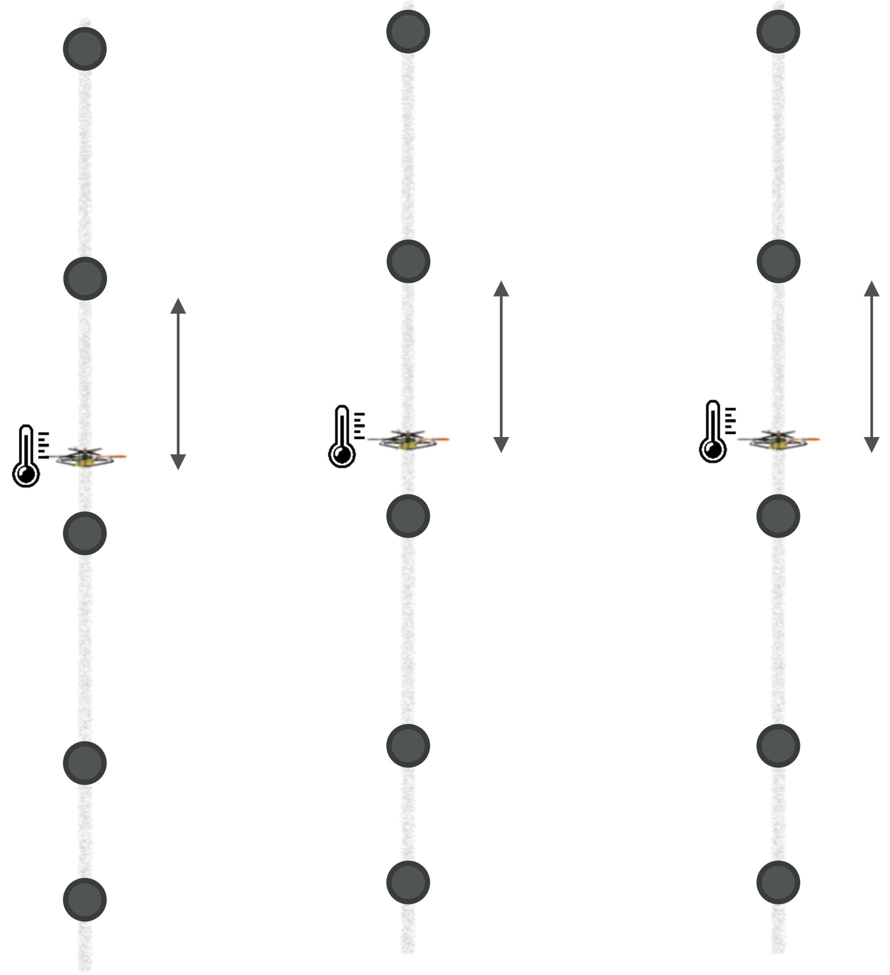
Accept radius: 3.00 meters

Altitude: 25.000 meters

✓ Altitude is relative to home

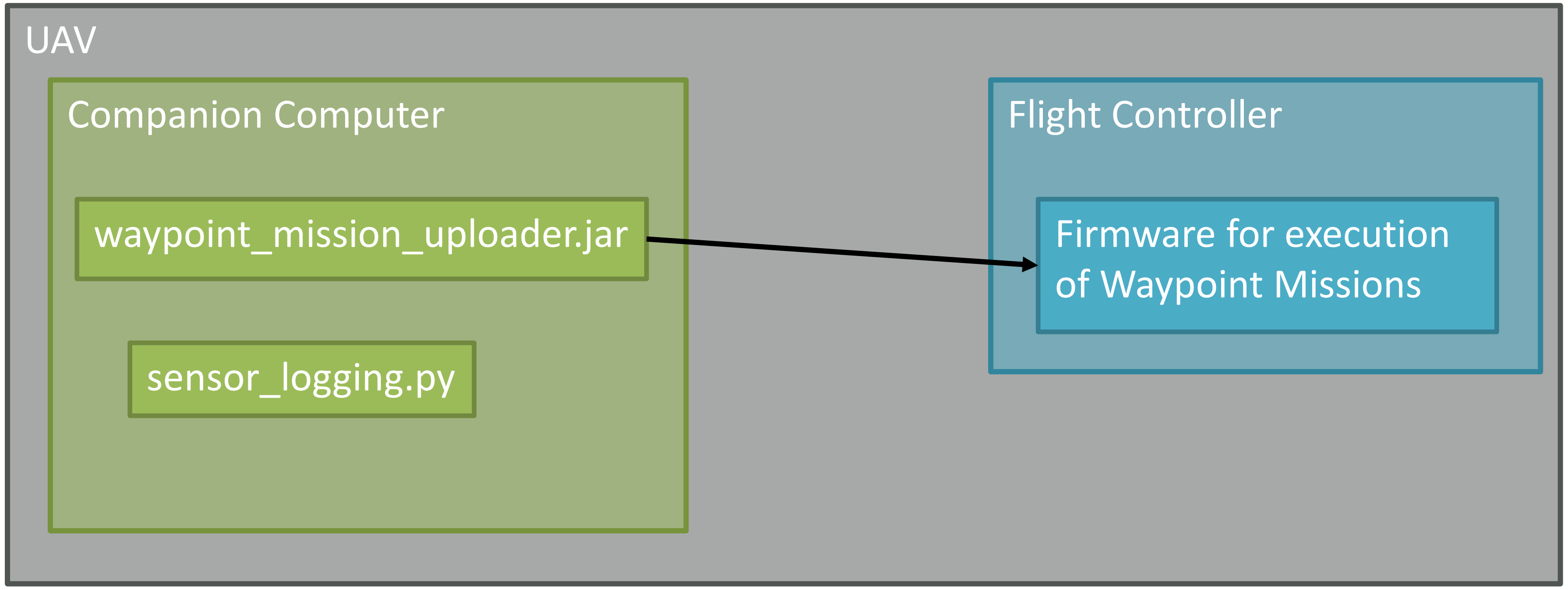
Missionsplanung: No automatic synchronization

„Synchronziation“ realized by manually starting all three missions at the same time and hoping for the best



Mission Planning: Poor extensibility of GCS

- Two seperate hardcoded programs for
  - Downloading and executing mission on flight controller
  - Aggregation of sensor values
- This separation complicates assignment of sensor data and copter / sensor position


UAV
Companion Computer
waypoint_mission_uploader.jar
sensor_logging.py
Flight Controller
Firmware for execution of Waypoint Missions

Scalex 2015: Problems

Institute for Software & Systems Engineering

Scalex 2015 Problems

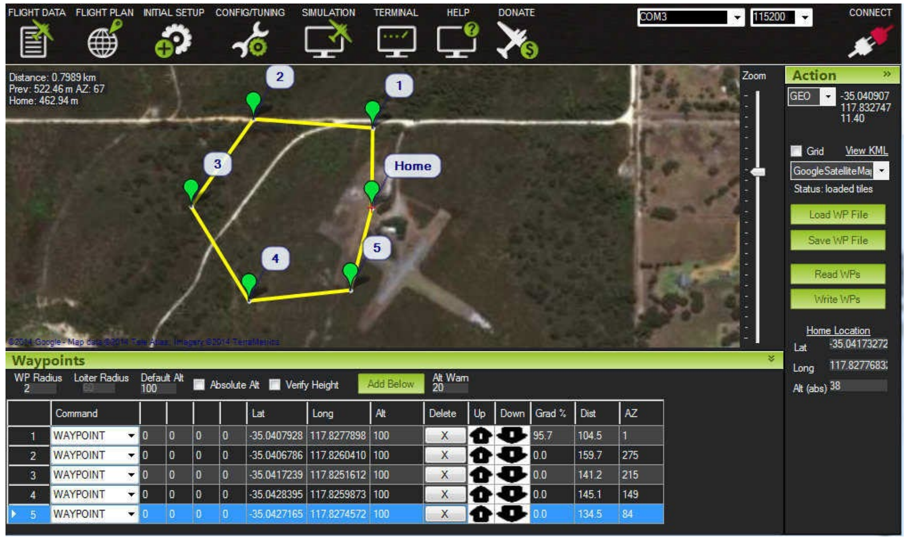
Missions not flexible enough

No synchronization

Hardcoded sensor logging

Mission Planning: State of the Art

- Mission planning is usually done via a ground control software (gcs)
  - Generic GCS: Qgroundcontrol, APM Planner, UgCS
  - Definition of waypoints that need to be reached in a certain sequence
  - Waiting on waypoints
  - Only limited possibilities for other actions (Logging, take pictures, activate sensors)
  - No or minimal swarm / ensemble support



Distance: 0.7989 km
Prev: 522.46 m AZ: 67
Home: 462.94 m

-35.040907
117.832747
11.40

Status: loaded tiles

Home Location
Lat -35.04173272
Long 117.8277683
Alt (abs) 38

Waypoints

WP Radius 2 | Loiter Radius | Default Alt 100 | Absolute Alt | Verify Height | Alt Warn 20

	Command					Lat	Long	Alt	Delete	Up	Down	Grad %	Dist	AZ
1	WAYPOINT	0	0	0	0	-35.0407928	117.8277898	100	X			95.7	104.5	1
2	WAYPOINT	0	0	0	0	-35.0406786	117.8260410	100	X			0.0	159.7	275
3	WAYPOINT	0	0	0	0	-35.0417239	117.8251612	100	X			0.0	141.2	215
4	WAYPOINT	0	0	0	0	-35.0428395	117.8259873	100	X			0.0	145.1	149
5	WAYPOINT	0	0	0	0	-35.0427165	117.8274572	100	X			0.0	134.5	84

Scalex 2015: Requirements for possible solution

Institute for Software & Systems Engineering

Modular mission planning framework

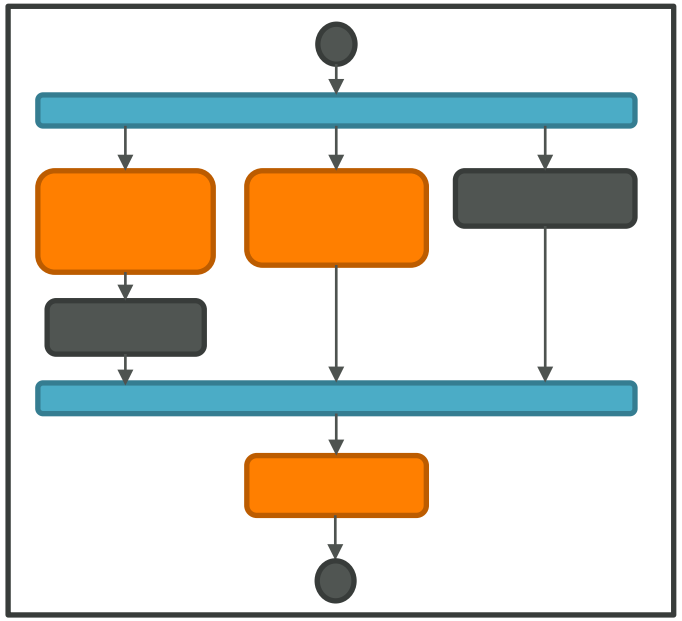
- Set of existing blocks to easily build simple missions
- Synchronize several UAVs with each other
- Possibility to log telemetry and sensor data
- Functionality easily expandable

Mission Planning with Block definition language (BDL)

Institute for Software & Systems Engineering

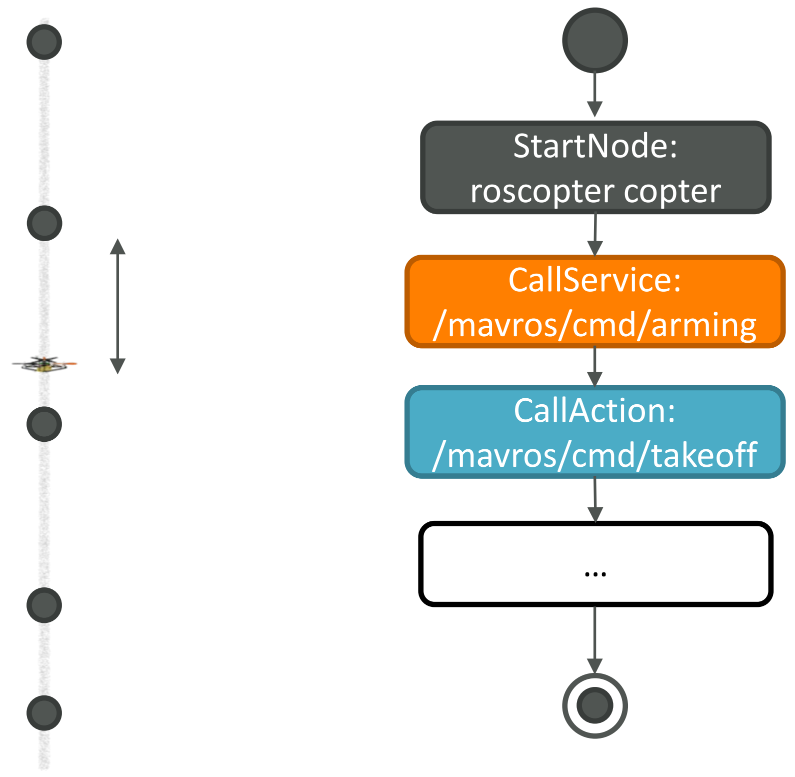
- Definition of a mission via json files
- Addition of new functionalities using Python modules
- Token semantics based on activity diagram
- Parallelism
- Decisions
- Reaction to errors

- Plugins for…
  - … interfacing with ROS2
  - … synchronizing multiple systems



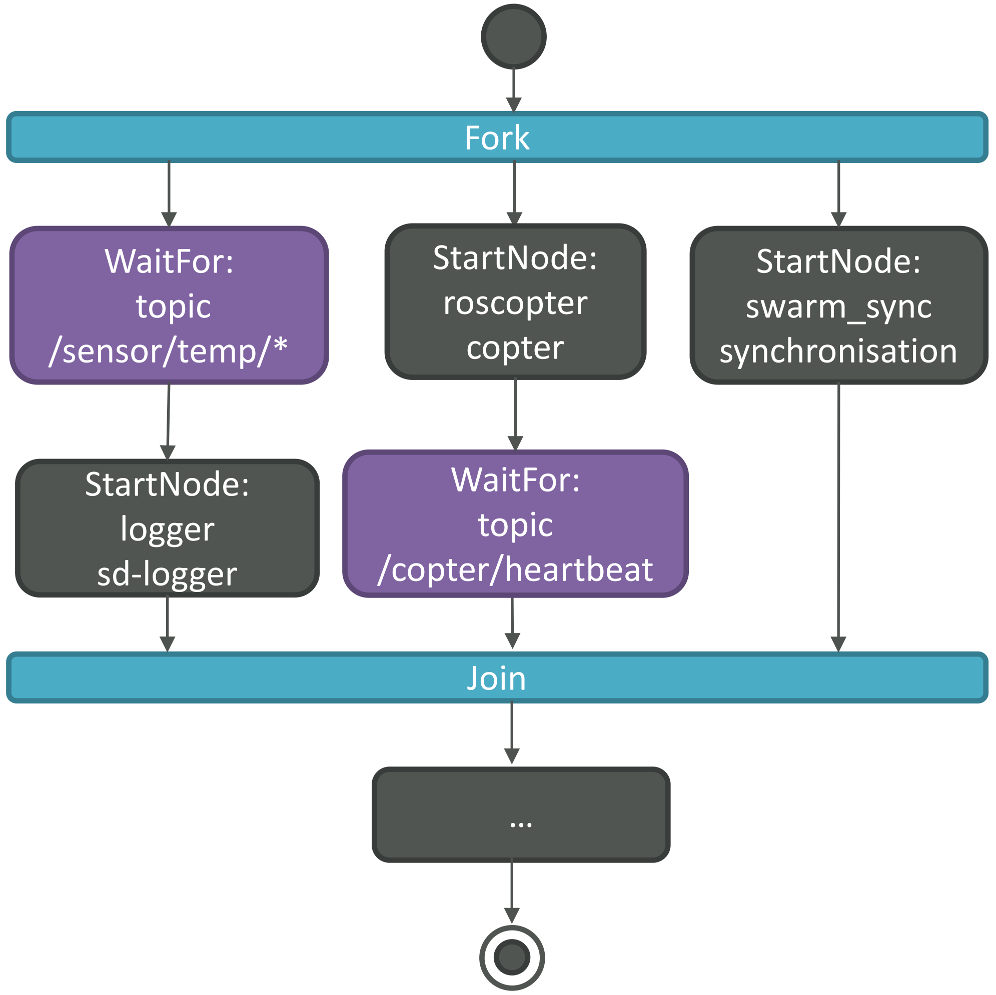
Mission Planning: Modeling a mission

Modeling of missions in ROS2



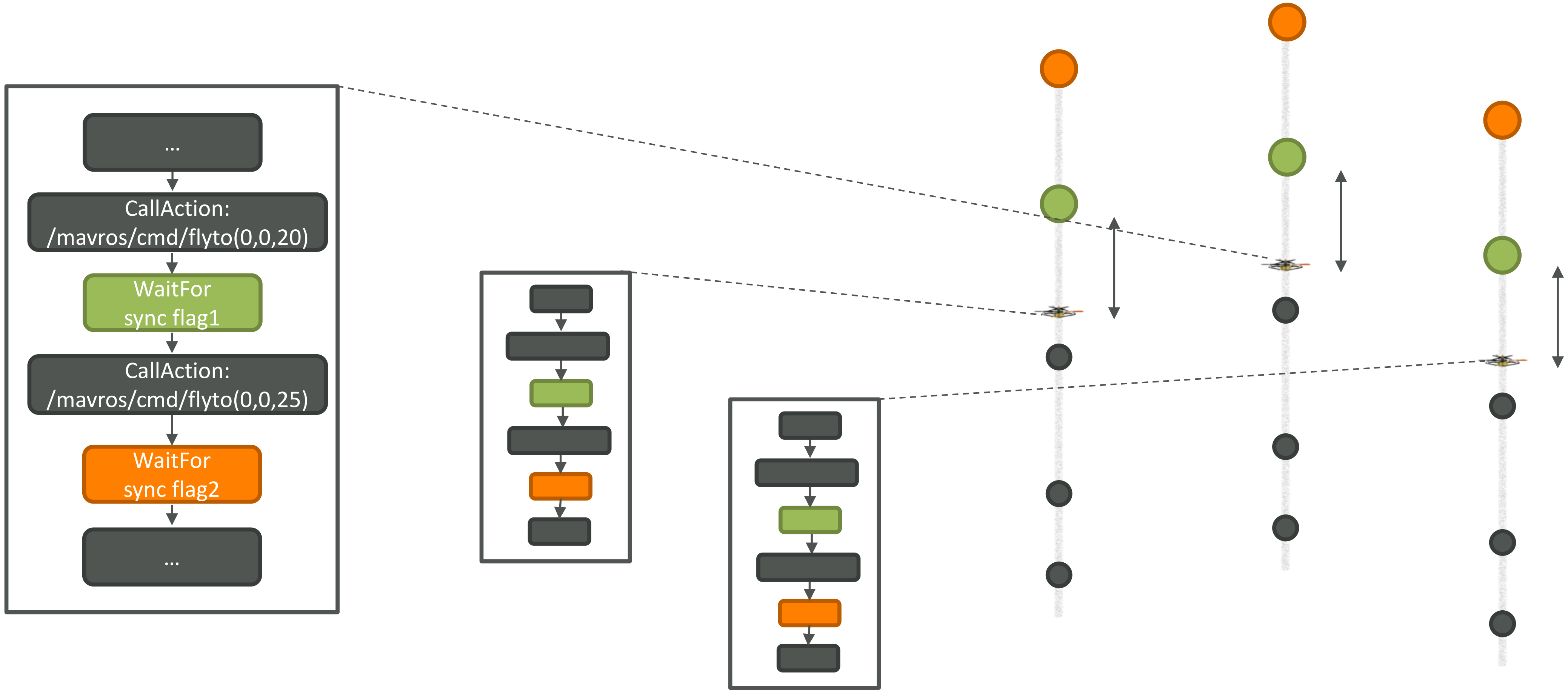
```
"b0": {
   "type": "Start",
   "next": "b1"
 },

 "b1":{
   "type": "StartNode",
   "package": "roscopter",
   "node": "copter",
   "nodename": "copternode1",
   "next": "b2"
 },

 "b2":{
   "type": "CallService",
   "name": "/mavros/cmd/arming",
   "args": "true",
   "next": "b3"
 },
…
```

Mission Planning: Parallel program flows

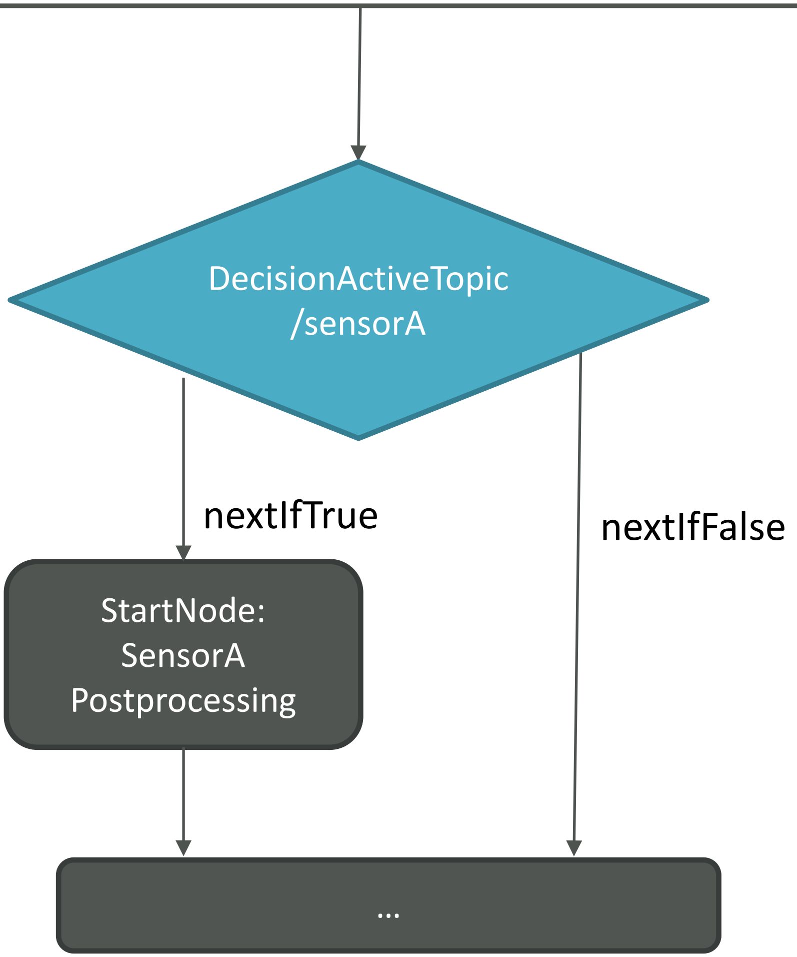

Fork
WaitFor: topic /sensor/temp/*
StartNode: roscopter copter
StartNode: swarm_sync synchronisation
StartNode: logger sd-logger
WaitFor: topic /copter/heartbeat
Join
…

Mission Planning: synchronization

Institute for Software & Systems Engineering


…
CallAction: /mavros/cmd/flyto(0,0,20)
WaitFor sync flag1
CallAction: /mavros/cmd/flyto(0,0,25)
WaitFor sync flag2
…

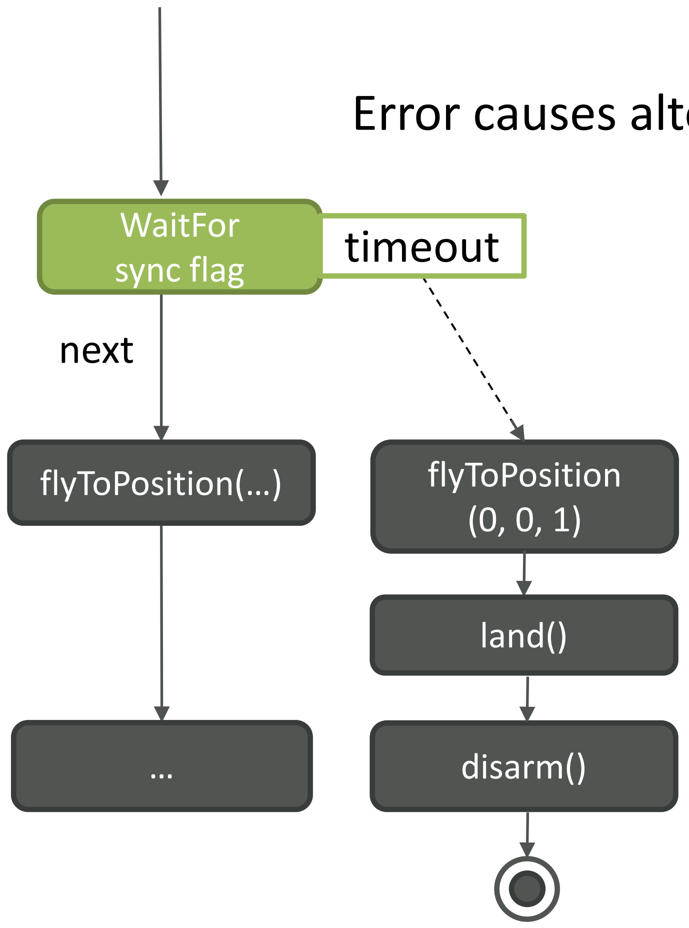
Mission Planning: Decision nodes



```
"b2": {
      "type": "DecisionRandom",
      "chanceForTrue": ".3",
      "nextIfTrue": "b5",
      "nextIfFalse": "b3"
},
```

Mission Planning: Error handling

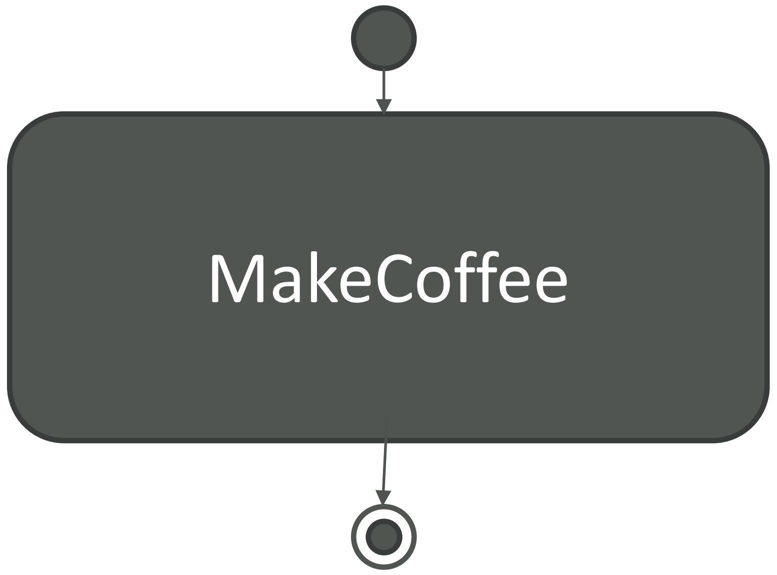
Error causes alternative program flow



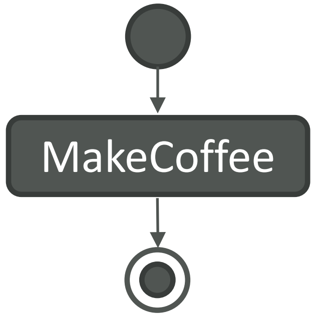
```
"b2": {
        "type": "WaitForSyncFlag",
        "id": "s13",
        "timeout": "15",
        "next": "b7",
        "nextTimeout": "b16"
},
```

Mission Planning: Extensibility

- That‘s nice and all, but i don‘t have a Drone ☹
- Can it make me some Coffee?



Mission Planning: Extensibility



coffee.json

```
"b0": {
     "type": "Start",
     "next": "b1"
  },

"b1": {
   "type": "MakeCoffe",
   "url": "iotcoffemaker.local",
   "type": "strong",
   "next": "b2"
},

"b2": {
     "type": „End",
}
```

coffee.py

```
import mission_player as mp
import coffee_maker as cm

# Define Functionality
def make_coffee(args):
  my_coffeemaker = cm(args["url"])
  my_coffeemaker.make(args["type"])
  return args["next"]

# Add Functionality
mp.add_block("MakeCoffee",make_coffee
)

# Play Mission
mp.play_mission("./coffee.json")
```

Institute for
Software & Systems
Engineering

Modeling and Execution of Coordinated Missions in Reconfigurable Robot Ensembles

Universität
Augsburg
University

Martin Schörner
schoerner@isse.de

Constantin Wanninger
wanninger@isse.de

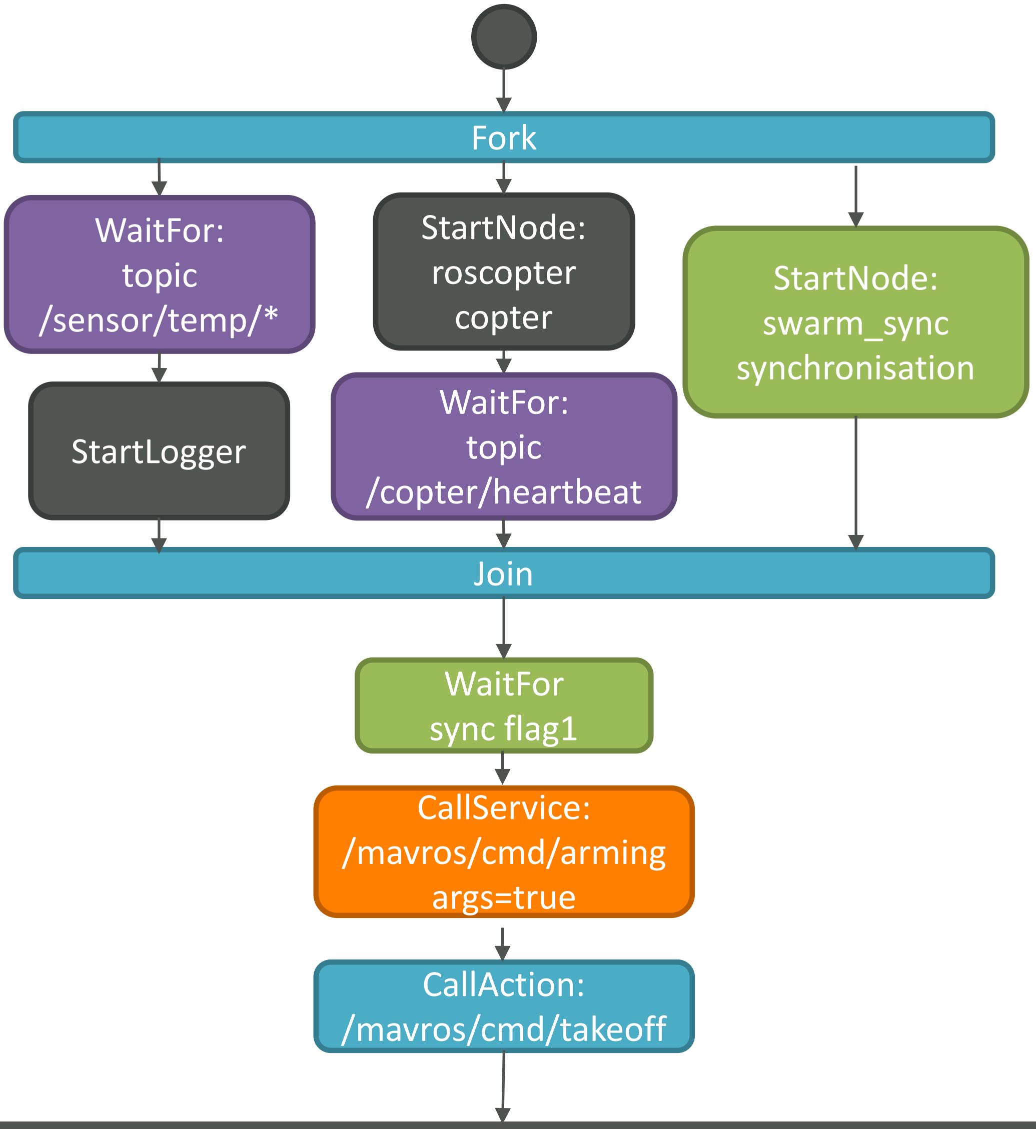
Alwin Hoffmann
hoffmann@isse.de

Oliver Kosak
kosak@isse.de

Hella Ponsar
ponsar@isse.de

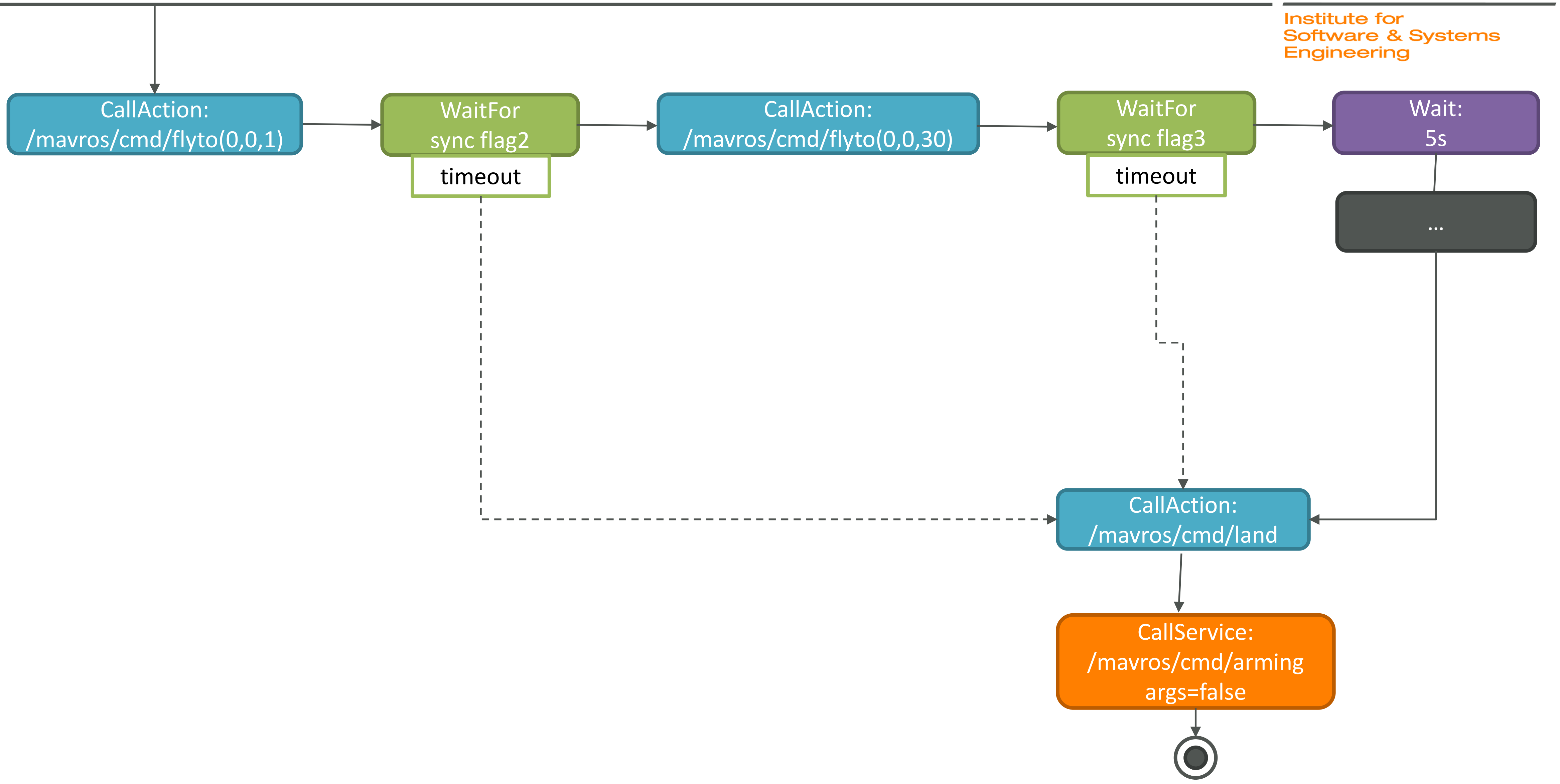
Wolfgang Reif
reif@isse.de

Mission Planning: ScaleX Remastered


Fork
WaitFor: topic /sensor/temp/*
StartNode: roscopter copter
StartNode: swarm_sync synchronisation
StartLogger
WaitFor: topic /copter/heartbeat
Join
WaitFor sync flag1
CallService: /mavros/cmd/arming args=true
CallAction: /mavros/cmd/takeoff

Mission Planning: ScaleX Remastered

Institute for Software & Systems Engineering


CallAction: /mavros/cmd/flyto(0,0,1)
WaitFor sync flag2
timeout
CallAction: /mavros/cmd/flyto(0,0,30)
WaitFor sync flag3
timeout
Wait: 5s
...
CallAction: /mavros/cmd/land
CallService: /mavros/cmd/arming args=false

Mission Planning : State of the Art

- ROS1 Launchfiles
  - XML
  - Allow to start multiple Programs / Nodes at once
  - No chronological order
  - No possiblities for cotrolling the program flow
- ROS2 Launchfiles
  - Python Script
  - Nodes have lifecycles
  - Launch file can react to changes in the lifecycle