Code Mobility

Konstantin Selyunin
e1228206@student.tuwien.ac.at
Igor Pelesić
igor.pelesic@gmail.com
Miljenko Jakovljević
micky686@gmail.com

December 5, 2012

Outline

1. Introduction
   - Code mobility overview
   - Level of abstraction
   - Requirements
2. System architecture
   - General overview
   - Agents
   - Platform
     - Scheduler
     - Execution Layer
   - Communication Protocol
3. Project management
4. Tools

Code mobility overview
Concept of code mobility

Concept of code mobility

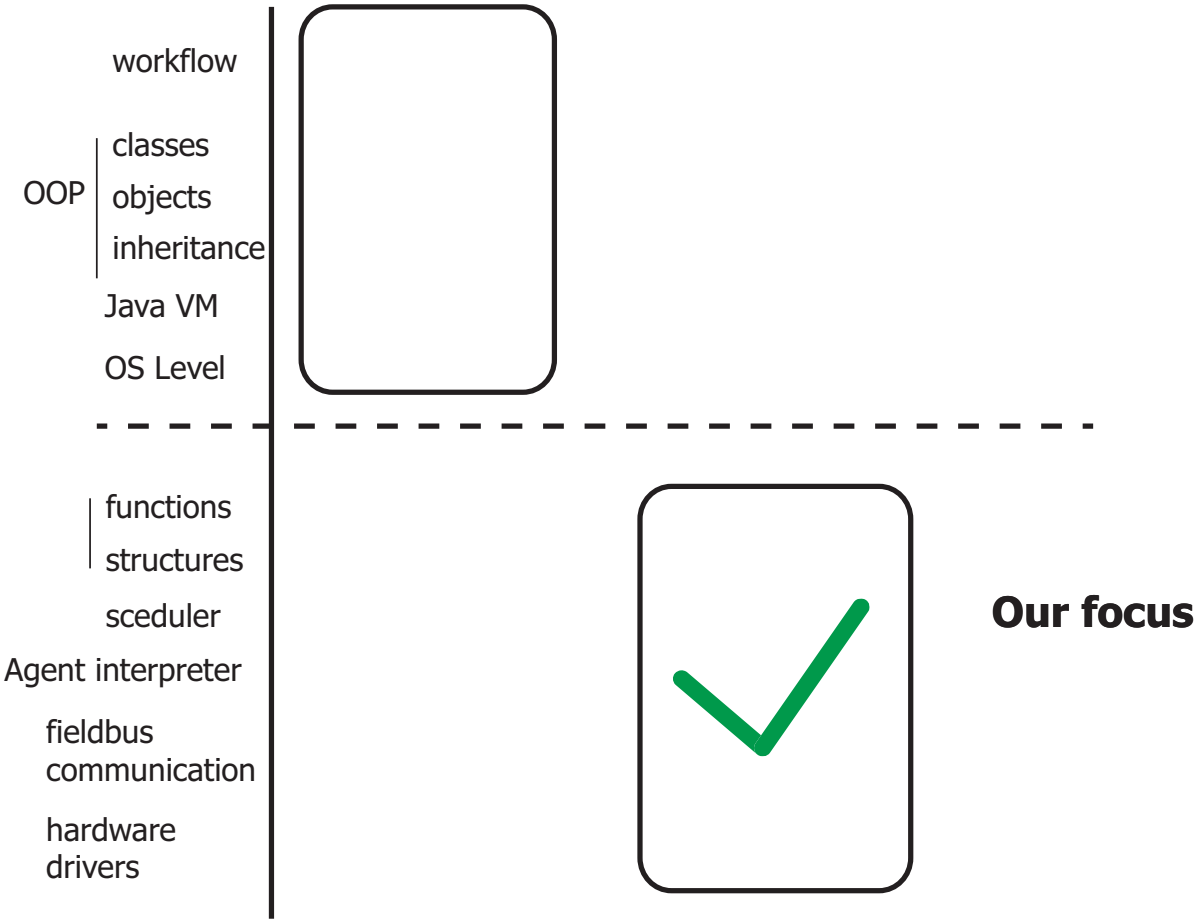
- Mobile agent
- *Strong* and *weak* code mobility
- Layered architecture

Advantages of code mobility

- Move code close to resources
- Enable client customization of remote resources
- Performance gains

[FPV98, BCMV06]

Level of abstraction



Requirements

- Agents:
  - simple language
  - support mobility and message exchange
- Platform:
  - execute agents concurrently
  - provide hardware services to agents
- Communication:
  - transfer agents & state *strong mobility*
  - transfer messages between platforms
  - cross board communication via Zigbee

General overview

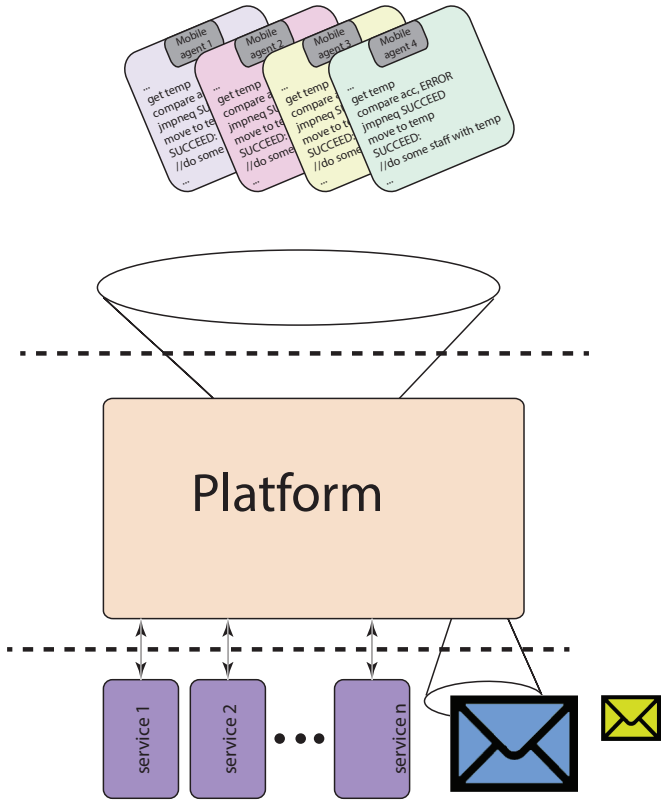
3 layered architecture:

- Agent level
- Platform level
- communication & drivers

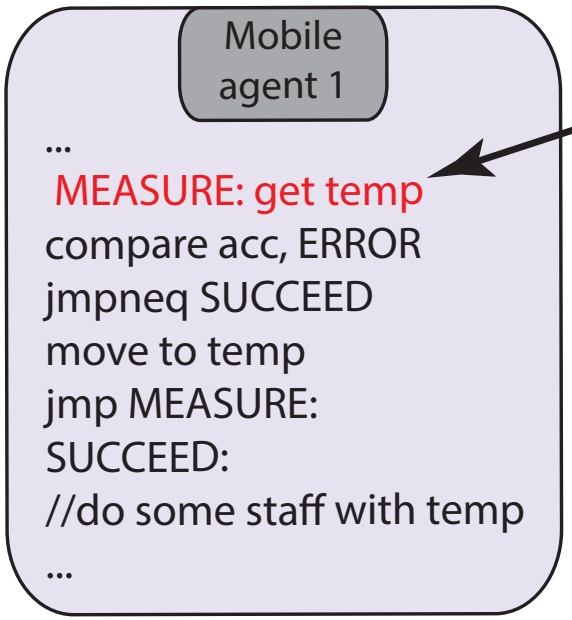
General overview

3 layered architecture:

- Agent level
- Platform level
- communication & drivers



Agents



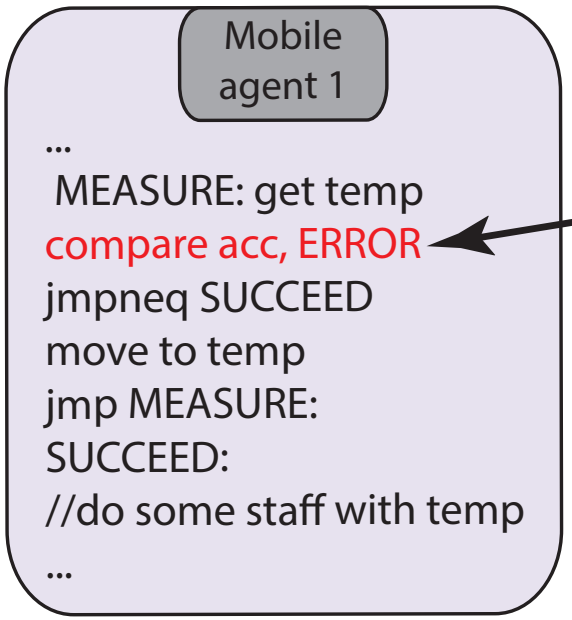
Get temperature value

Platform can provide this service?

yes: do staff

no: move agent to another platform

Agents



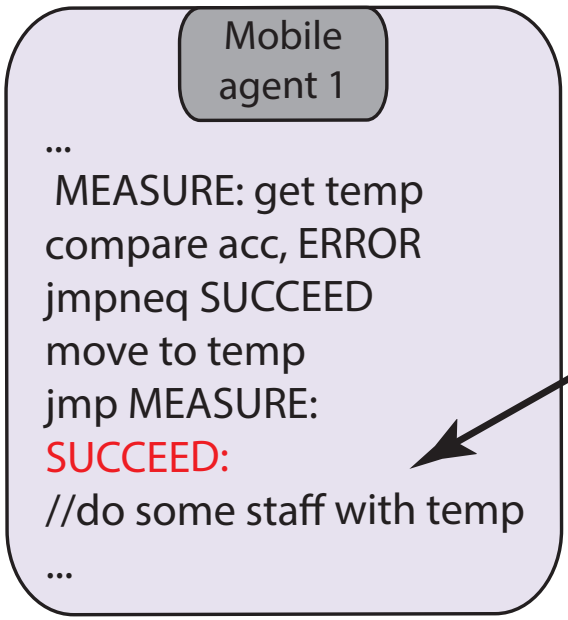
Get temperature value

Platform can provide this service?

yes: do staff

no: move agent to another platform

Agents



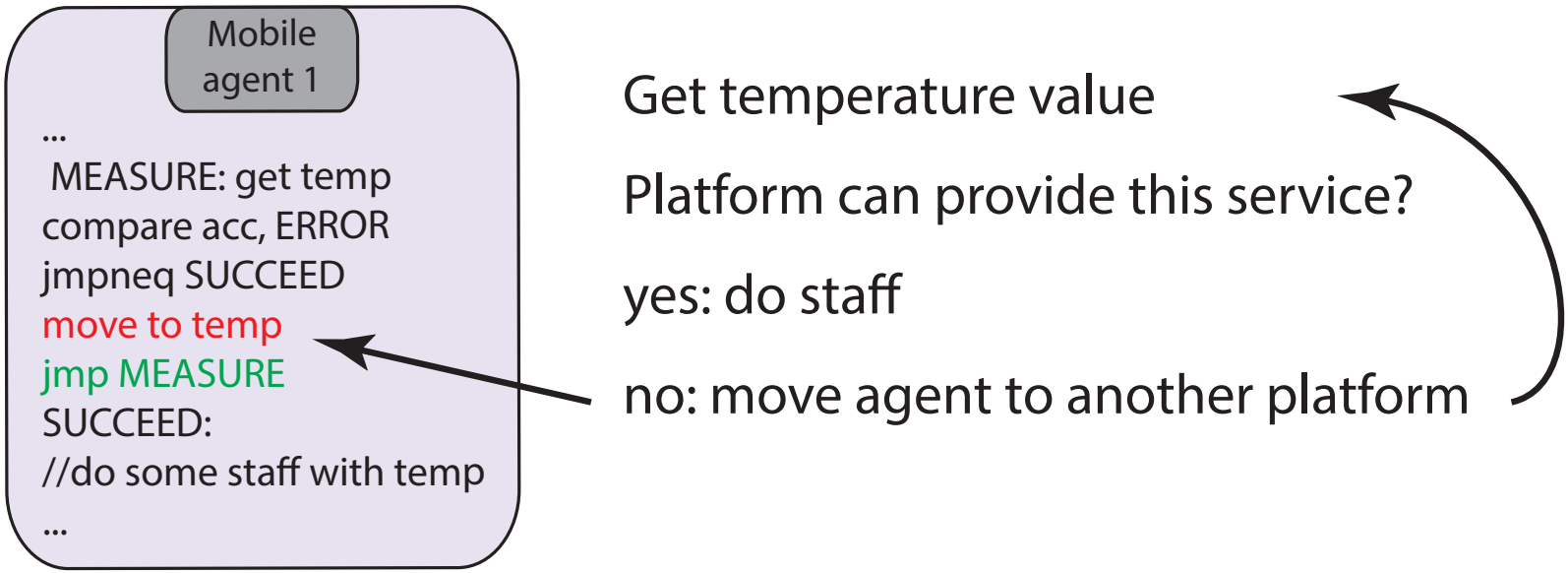
Get temperature value

Platform can provide this service?

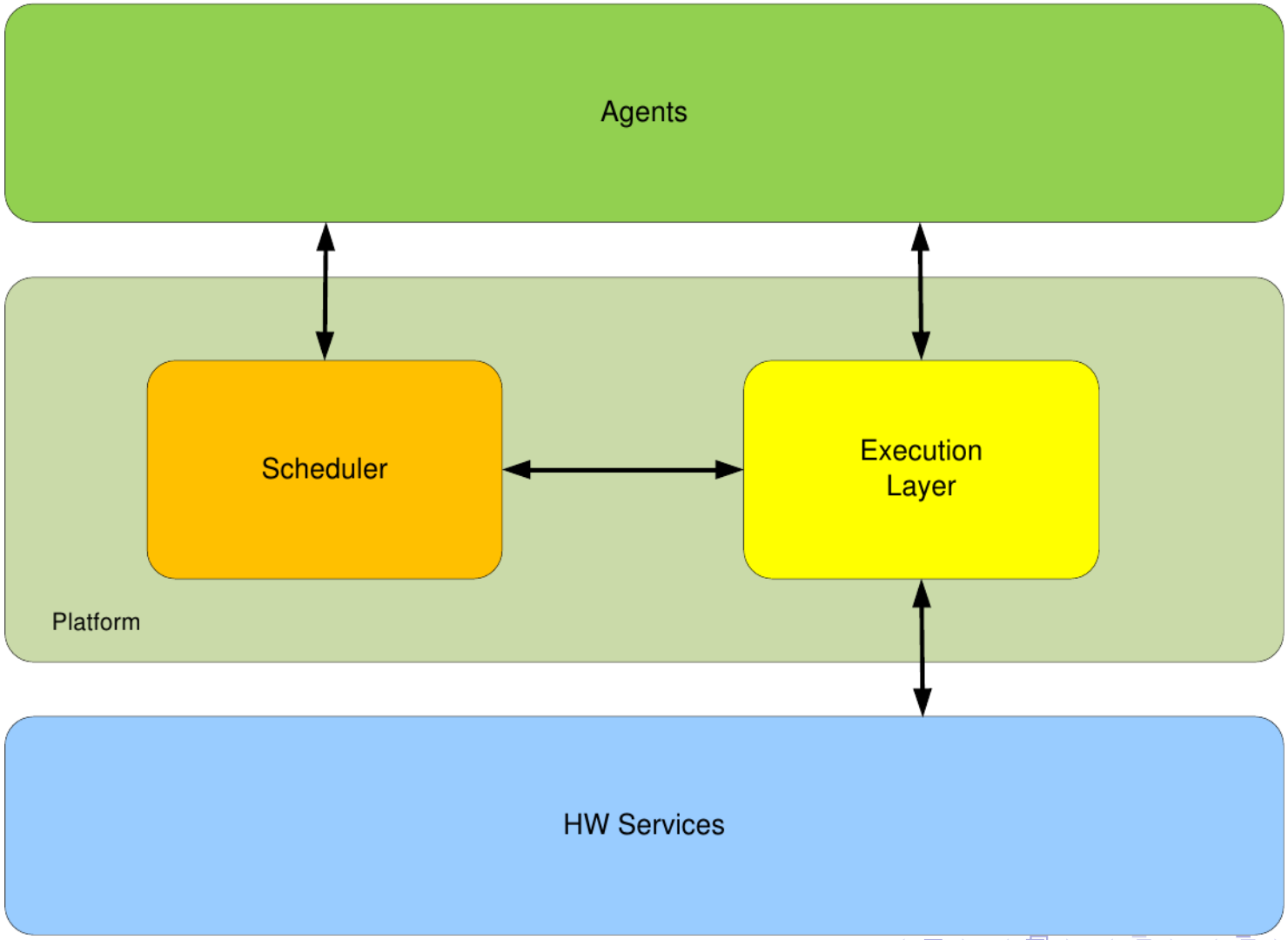
yes: do staff

no: move agent to another platform

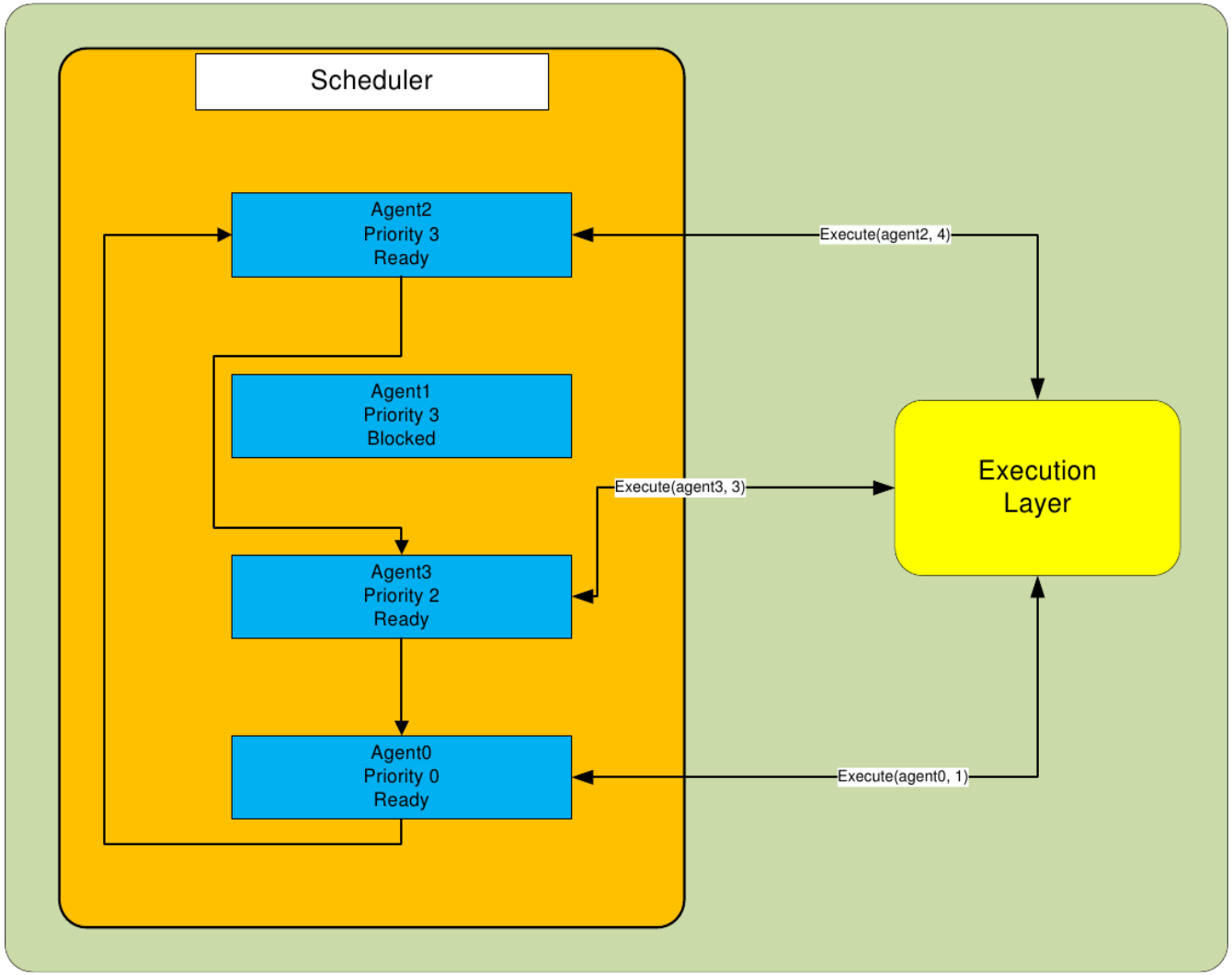
Agents


Mobile agent 1
...
MEASURE: get temp
compare acc, ERROR
jmpneq SUCCEED
move to temp
jmp MEASURE
SUCCEED:
//do some staff with temp
...
Get temperature value
Platform can provide this service?
yes: do staff
no: move agent to another platform

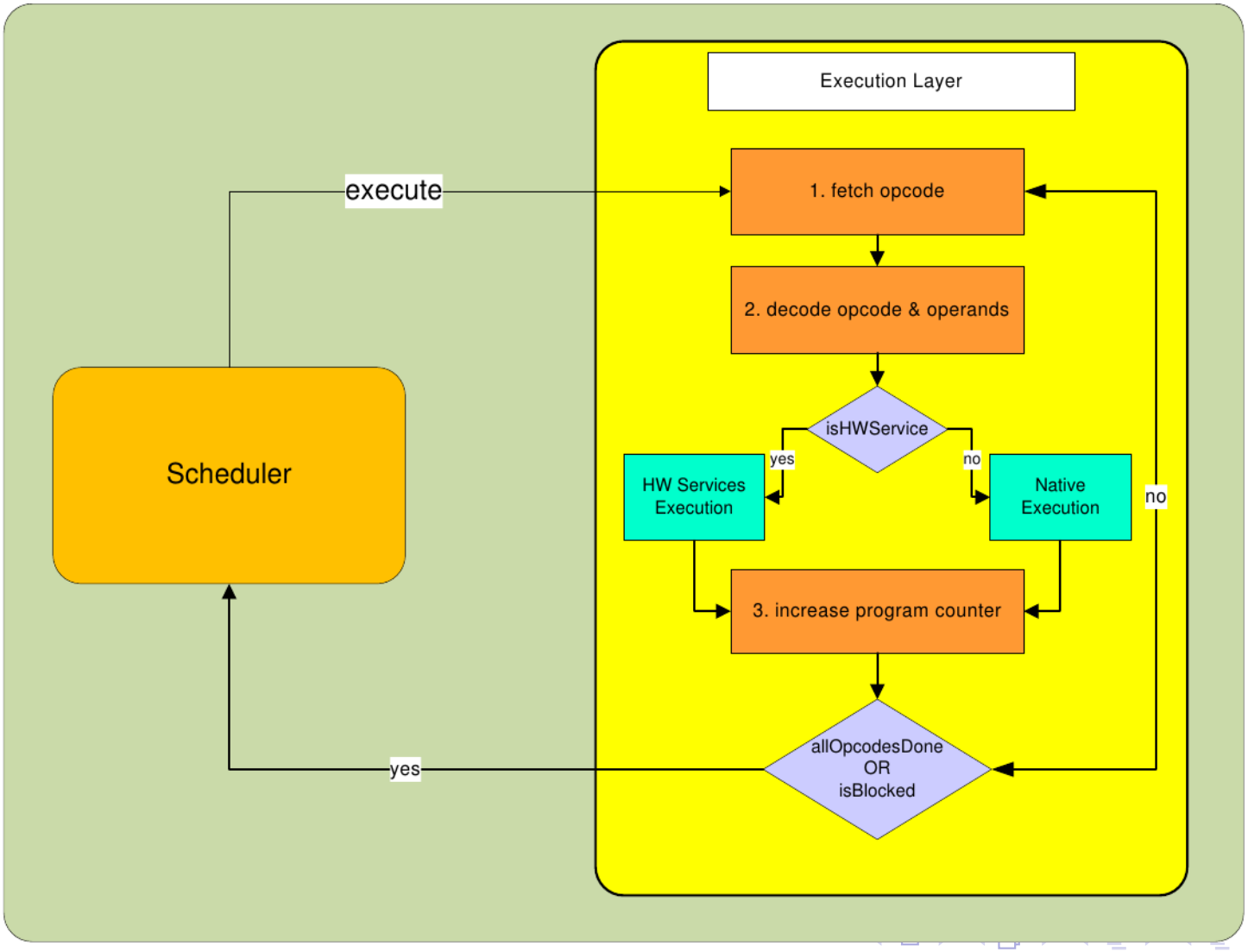
Platform



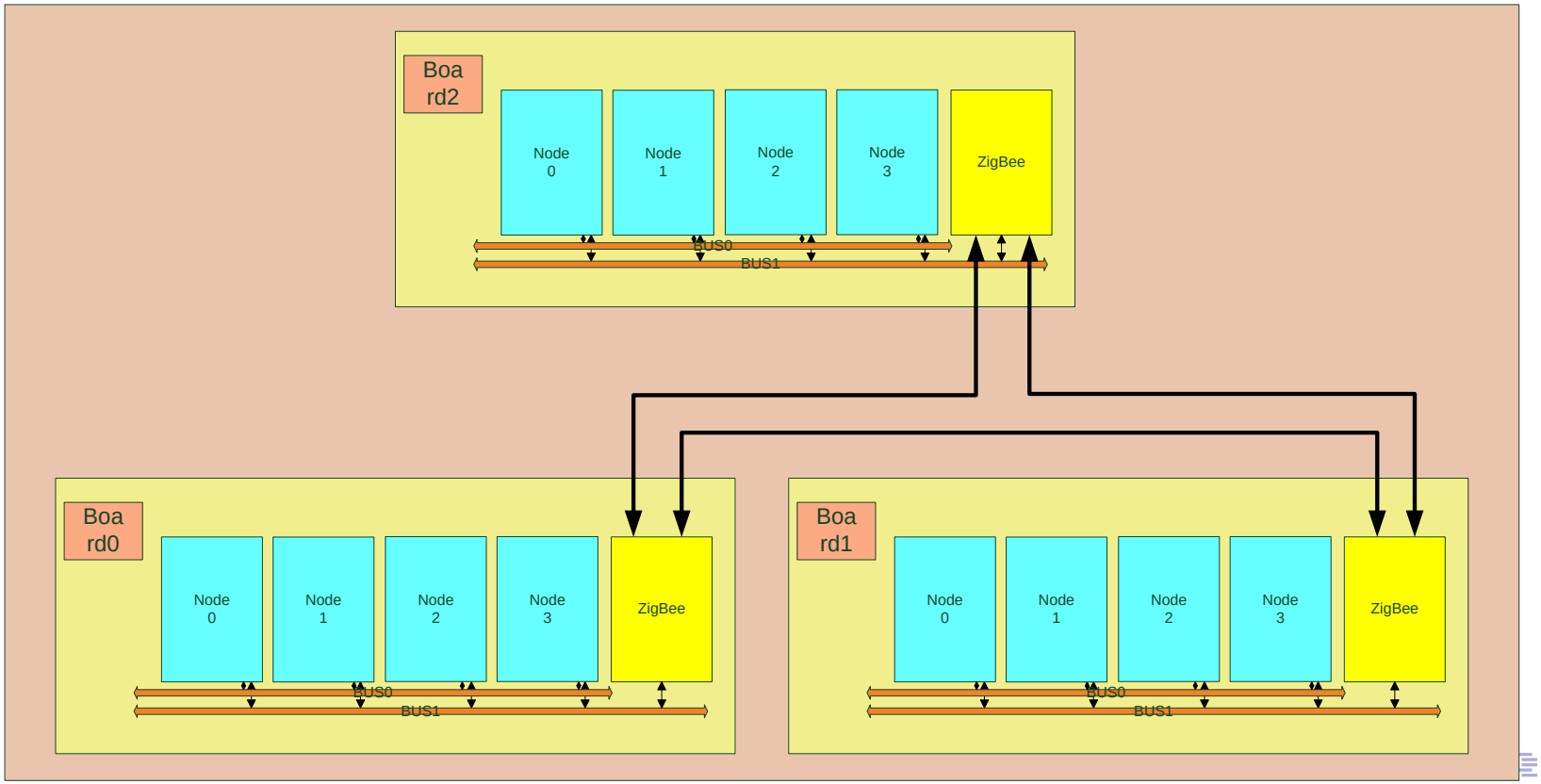
Scheduler


Scheduler
Agent2
Priority 3
Ready
Agent1
Priority 3
Blocked
Agent3
Priority 2
Ready
Agent0
Priority 0
Ready
Execute(agent2, 4)
Execute(agent3, 3)
Execute(agent0, 1)
Execution
Layer

Execution Layer



Communication Protocol
Communication Participants



Protocol Design

Requirements

Local *on-board* communication

Remote communication

Sending agent code

Sending application data

Principles

Layered design

Fairness in network access

Composability with *Zigbee*

Acknowledgement and retry

Transmission Layers

Byte	MSB	LSB
0	*destination node*	*payload length*
1	*data*	
|	...	
14	*data*	
15	*crc*	

Figure : Low Level Datagram

Byte	MSB	LSB
0	*destination node*	*payload length*
1	*source node*	*destination board*
2	*source board*	*packet type*
3	*frame id*	
4	*packet id high*	
5	*packet id low*	
6	*empty*	
7	*data*	
|	...	
14	*data*	
15	*crc*	

Figure : High Level Datagram

Network Configuration

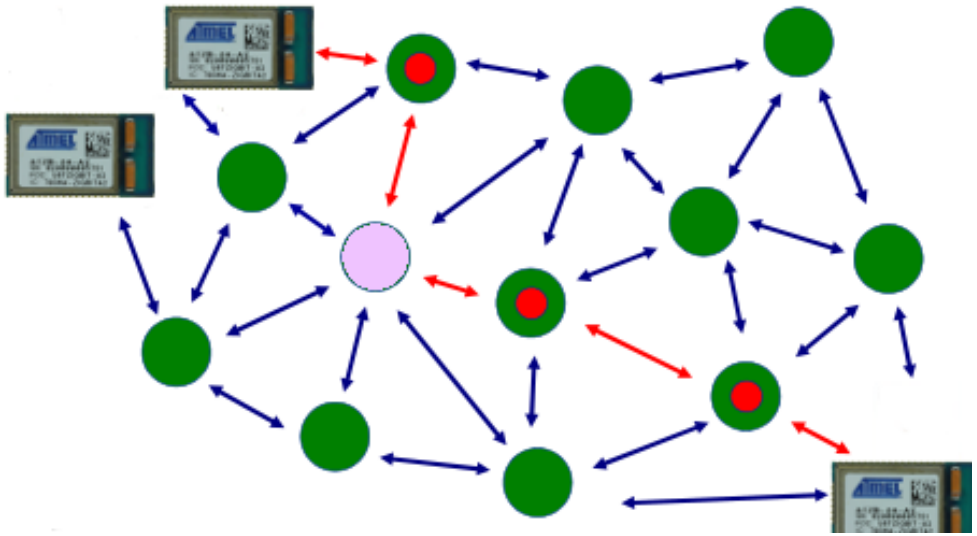


Figure : Zigbee Mesh Network

Zigbee Network Configuration

Rerouting Example

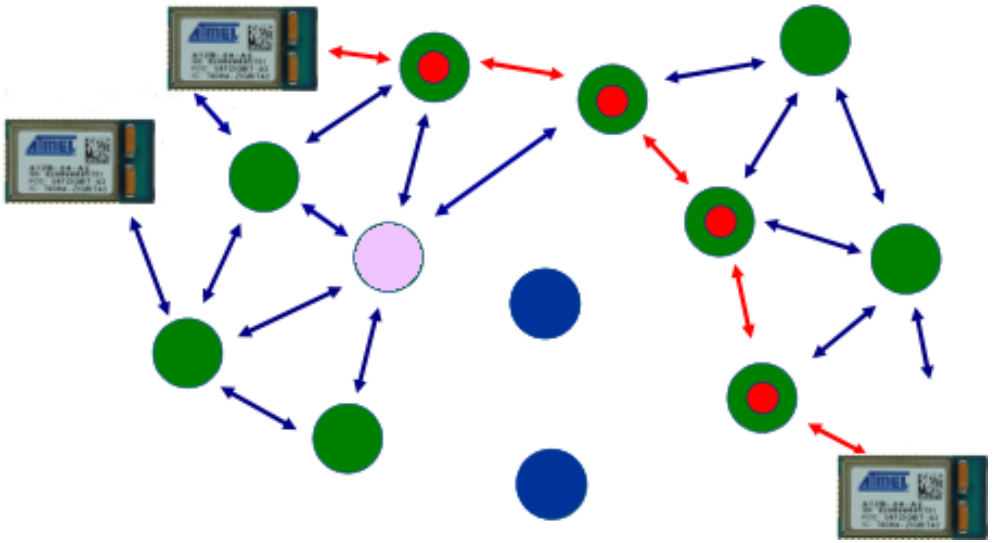


Figure : Network after rerouting

- Network Coordinator
- Failed Node
- Network Router
- Message Route

Milestones

Phase 1. Product outline and information gathering

Phase 2. Application requirements and specification

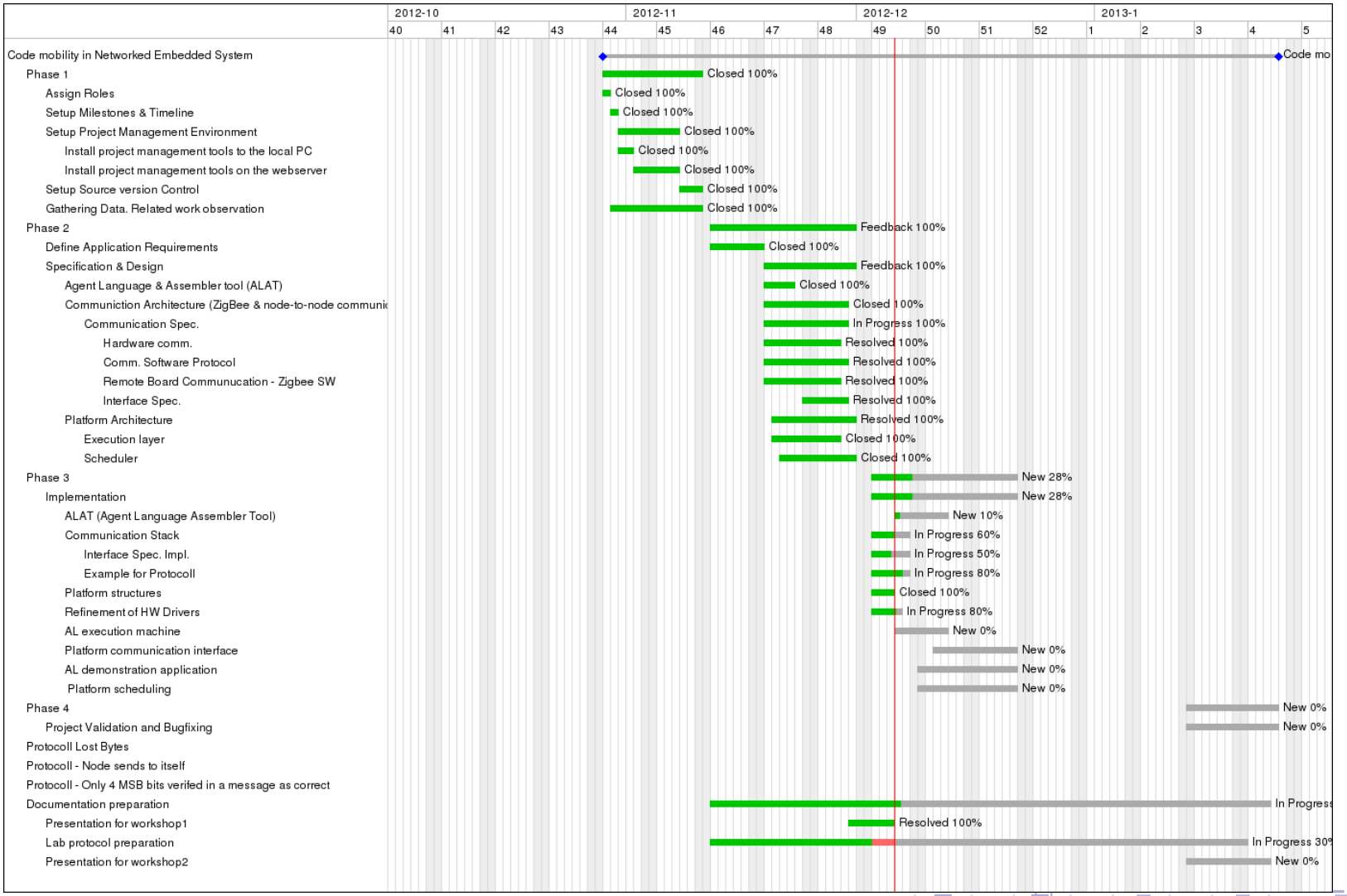
Phase 3. Implementation

Phase 4. Validation and analysis

Workpackages

	Name	Dates	Interdependencies	Deliverables
WP1	Documentation	all	25.10.12 - 15.01.13	D1.1 Lab protocol
				D1.2 specification
				D1.3 workshop1
				D1.4 workshop2
WP2	Adaption of drivers		10.12 - 15.12	D2.1 hardware drivers
WP2	Agent language tool		6.12 - 10.12	D2.1 Agent language assembler tool
WP4	Communication	D2.1		Protocol
WP5	Platform	WP2, WP4	10.12 - 21.12	D3.1 Platform

Gantt diagram



Tools

Version control	git
Documentation & code repository	github
File sharing	amazon s3
Project management	redmine

http://nes2012group4.herokuapp.com/

Code generation	SCADE
Editors	Emacs
	gedit

References

Ezio Bartocci, Flavio Corradini, Emanuela Merelli, and Leonardo Vito.
Model driven design and implementation of activity-based applications in hermes.
Proceedings of the 7th WOA 2006 Workshop, From Objects to Agents (Dagli Oggetti Agli Agenti), Catania, Italy, September 26-27, 2006, 2006.

Alfonso Fuggetta, Gian Pietro Picco, and Giovanni Vigna.
Understanding code mobility.
IEEE Transactions on Software Engineering, 24:342–361, 1998.

Questions

Roles

Konstantin Selyunin	
Project manager	internal coordination
	defining tasks
	control meeting deadlines
Igor Pelesić	
System architect	technical decisions
	determine technical part of the project
Miljenko Jakovljević	
Documentation responsible	Lab protocol
	documentation decisions

Goals

Our goals:

- Design, implement and evaluate code mobility system on ESE Board
- Hardware drivers & mobile agents & communication
- Master project management skills

Use cases

Risk management

High-level agent language