

Code Mobility

Konstantin Selyunin

`e1228206@student.tuwien.ac.at`

Miljenko Jakovljević

`micky686@gmail.com`

Igor Pelesić

`igor.pelesic@gmail.com`

December 4, 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

Our goal:

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

Concept of code mobility

Mobile agent

Strong and *weak* code mobility

Layered architecture

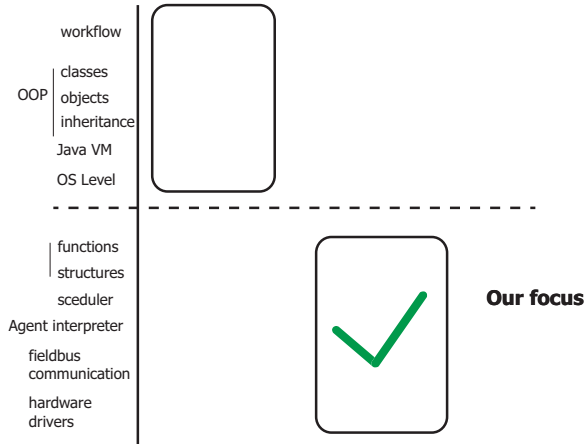
Advantages of code mobility

Move code close to resources

Enable client customization of remote resources

Performance gains

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:

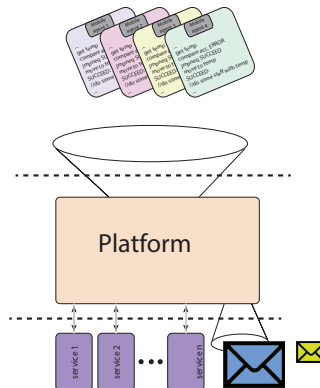
- Agent level
- Platform level
- communication & drivers



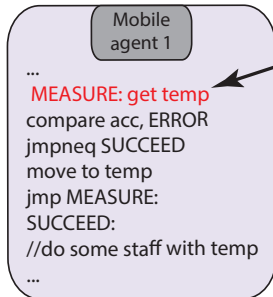
General overview

3 layered architecture:

- Agent level
- Platform level
- communication & drivers



Agents



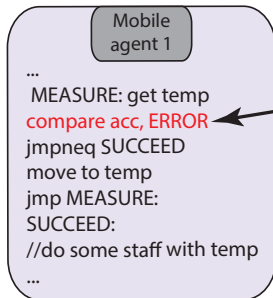
Get temperature value

Platform can provide this service?

yes: do staff

no: move agent to another platform

Agents



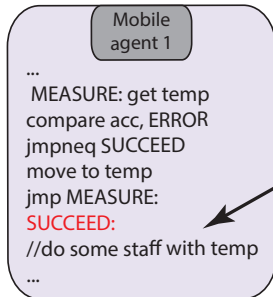
Get temperature value

Platform can provide this service?

yes: do staff

no: move agent to another platform

Agents



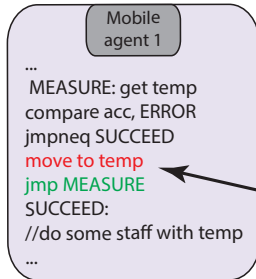
Get temperature value

Platform can provide this service?

yes: do staff

no: move agent to another platform

Agents

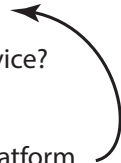


Get temperature value

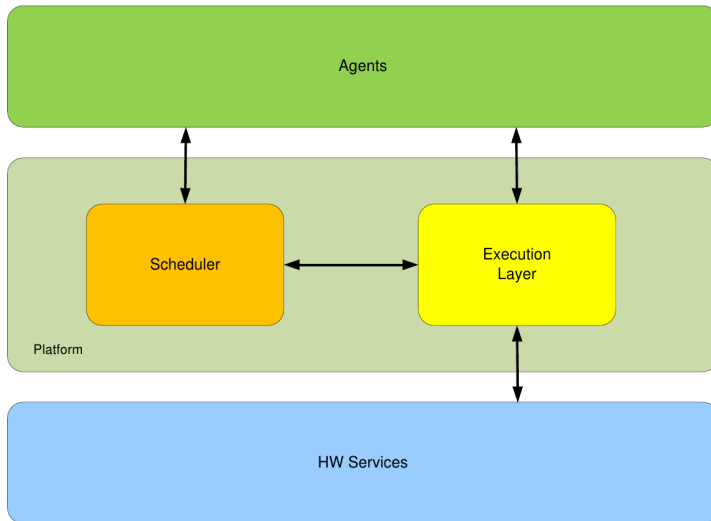
Platform can provide this service?

yes: do staff

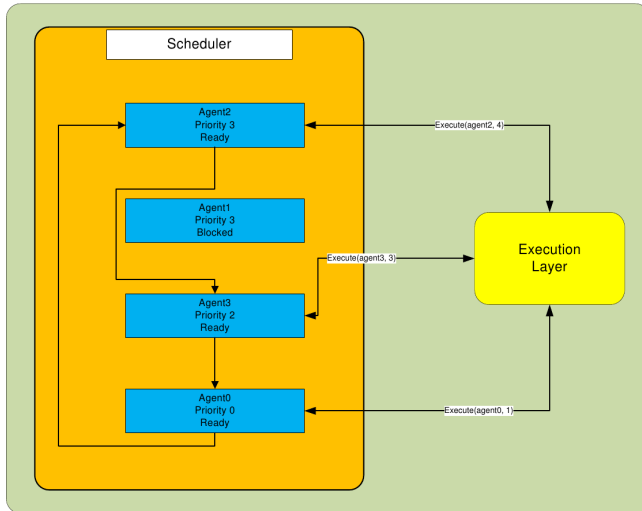
no: move agent to another platform



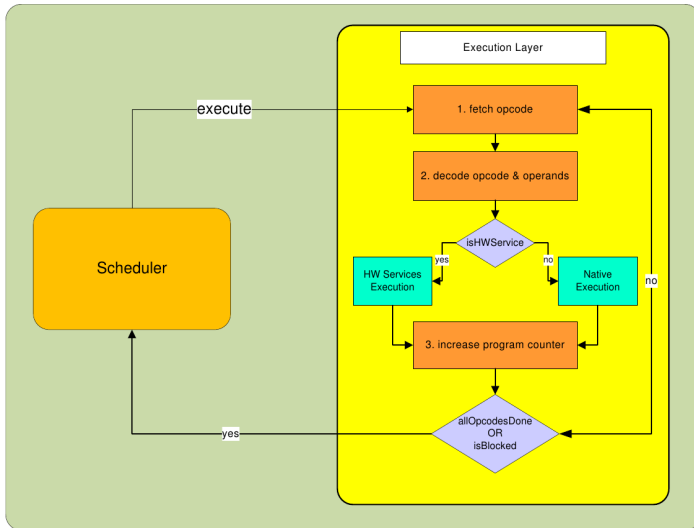
Platform



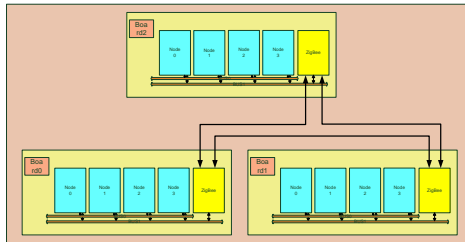
Scheduler



Execution Layer



Communication Requirements



- Inter-node and Inter-board communication
- Correct sending of large amounts of agent code
- Timely delivery of control messages
- Bridging between local and wireless network

Transmission Layers

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

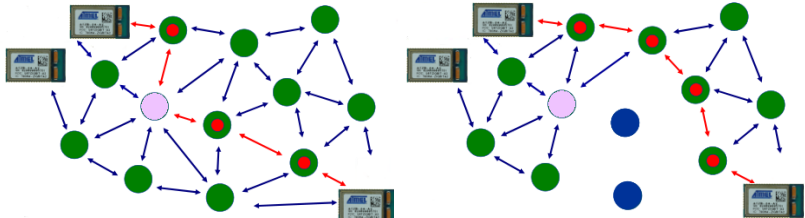
Byte	MSB	LSB
0	destination node	payload length
1	source node	destination board
2	source board	packet type
3	frame id	
4	packet id high	
5	packet id low	
6	empty	
7	data	
1	...	
14	data	
15	crc	

Figure: Lower-and Upper Level Datagram

- Lower level: priority based CSMA/CA protocol
- Upper level: routing and frame composition
- Network bridge: Zigbee access on application layer

Network Topology

Fault Tolerant Routing



- Zigbee Network Coordinator
- Zigbee Network Router
- Disabled Node

Milestones



Phase 1. Product outline and information gathering



Phase 2. Application requirements and specification



Phase 3. Implementation

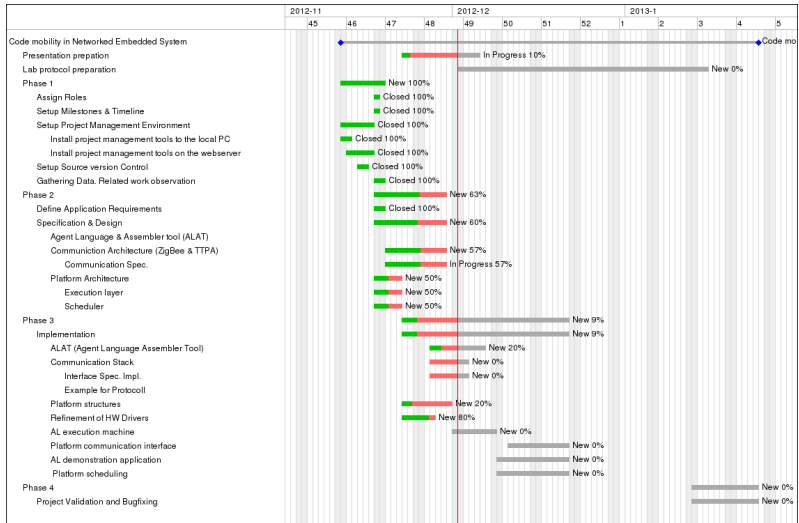


Phase 4. Validation and analysis

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