Cyber-Physical Systems, Robots and Embedded Software in Java

Anders P. Ravn
Aalborg University

BEST Summer school
July 2013

JAcknowledgement: Jens Lyngsø, Piotr Makowski, Jan D. Bendtsen, Thomas Bak, Hans J. Andersen, Svend Christensen, Thomas Bygholm, Tomas Kalibera, Stephan Korsholm, René R.Hansen, Martin Schoeberl, Hans Søndergaard, and Bent Thomsen.

Water in – water out – how ?

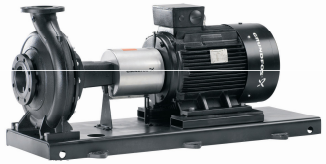
Necessary Components





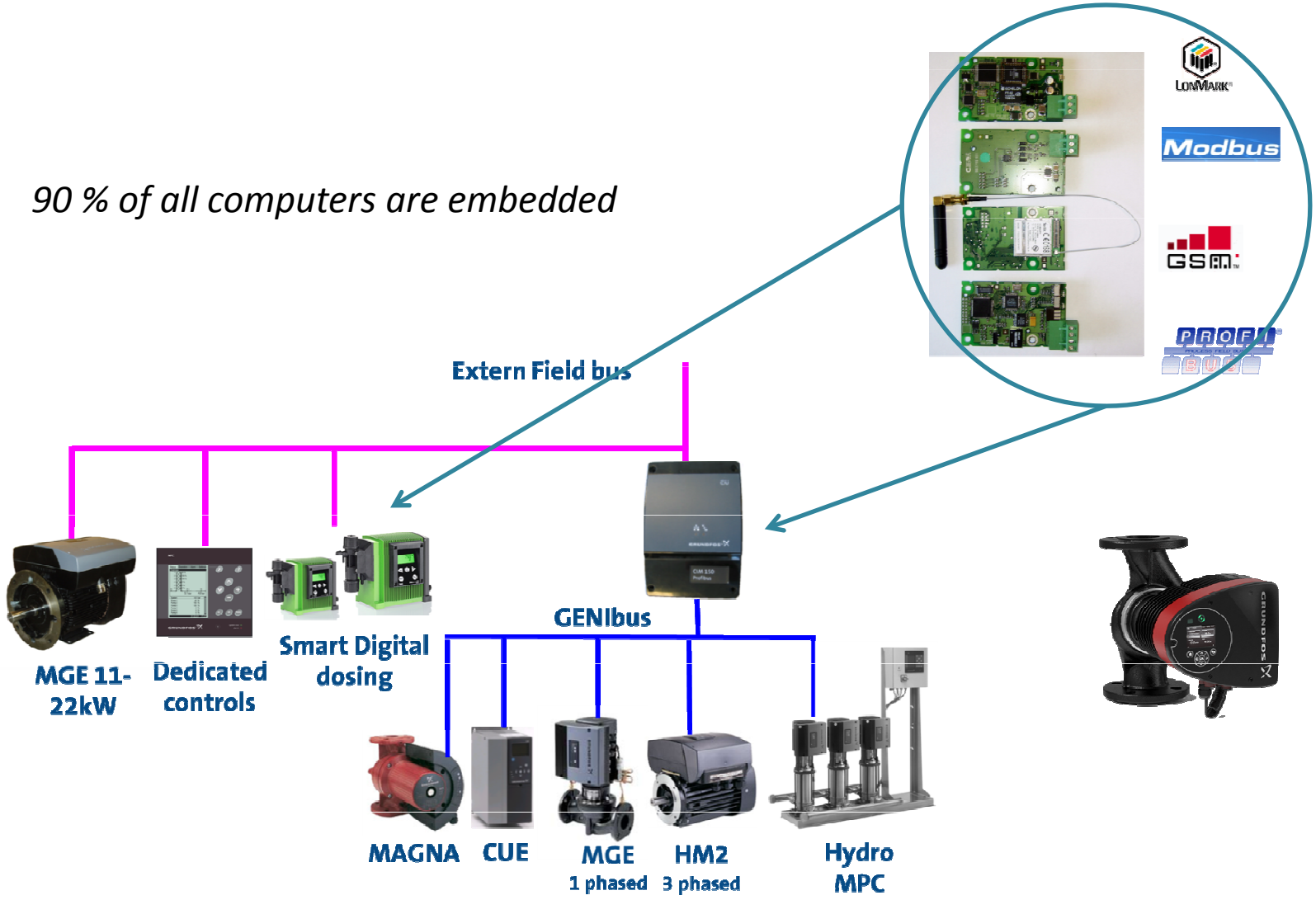






Embedded Computers

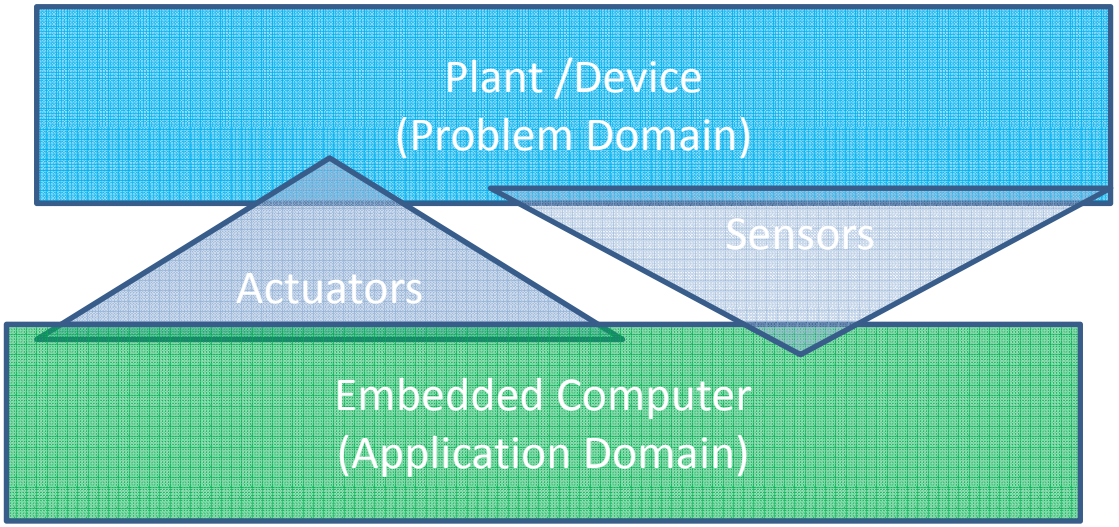
90 % of all computers are embedded



A cyber-physical system

"Cyber-Physical Systems (CPS) are integrations of computation and physical processes"

[Lee, 2007]

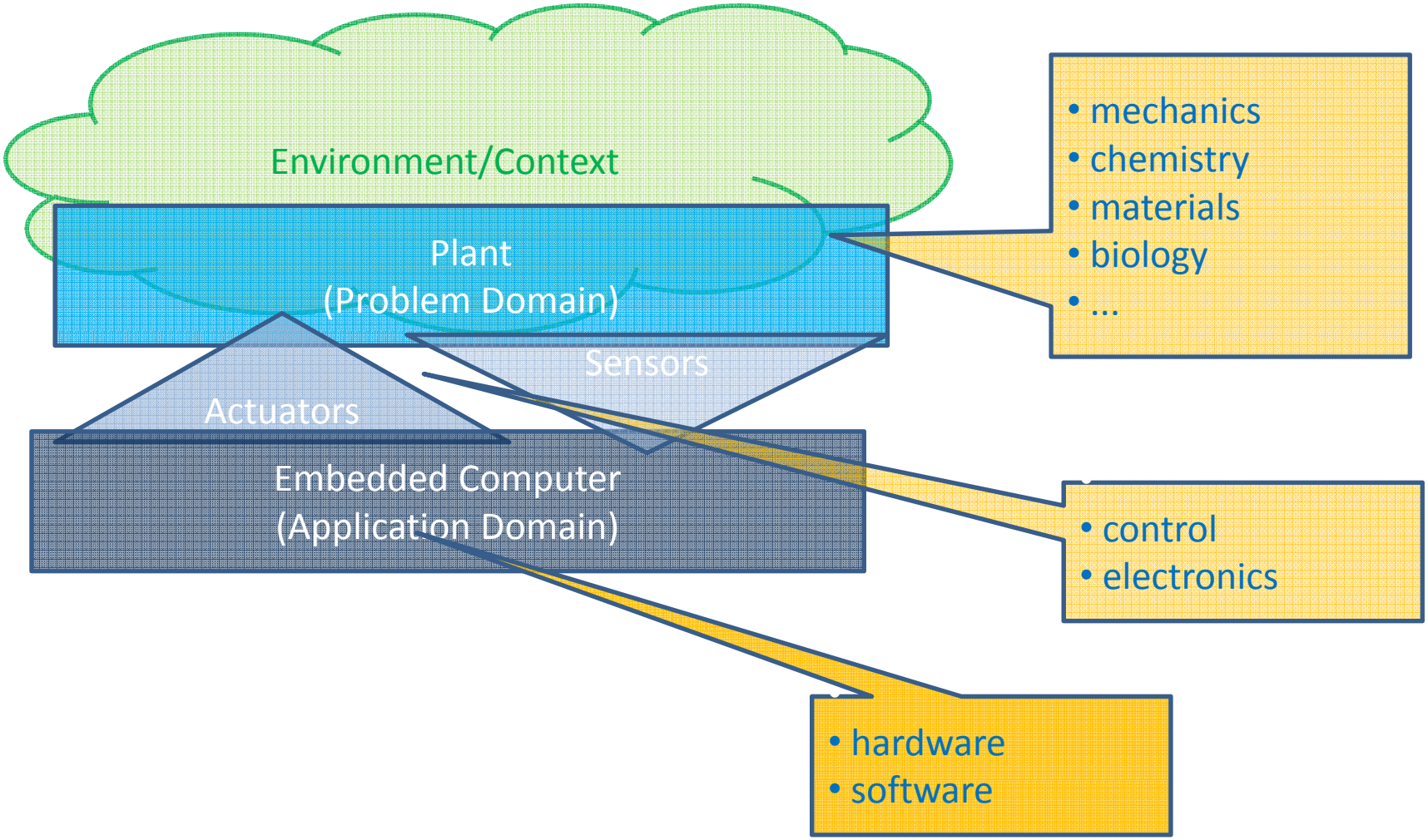


Peter Marwedel: *Embedded System Design:*

Embedded Systems Foundations of Cyber-Physical Systems

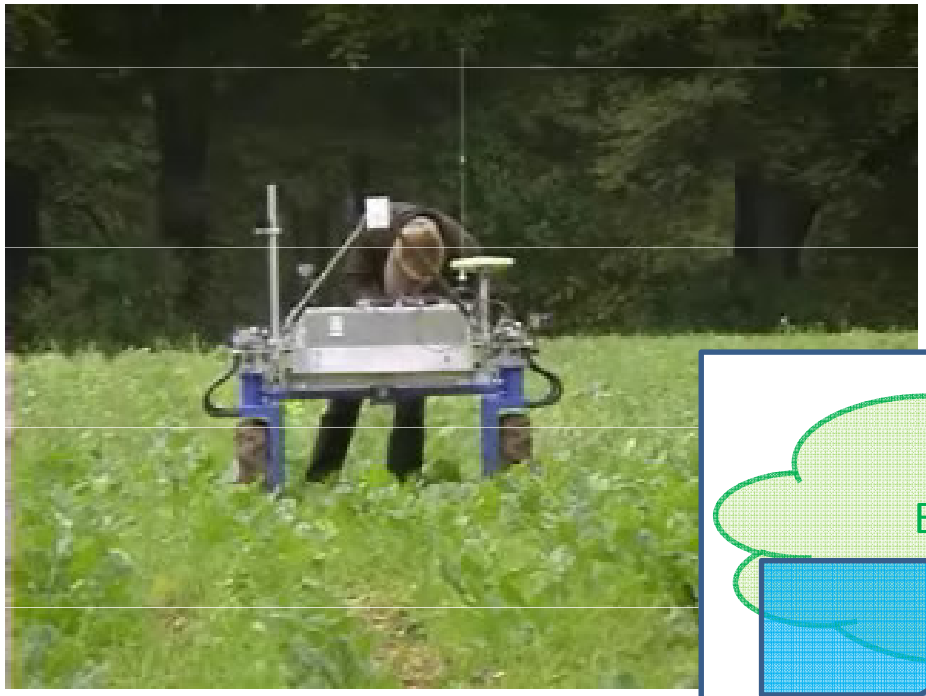
(2nd edition) Springer-Verlag, 2011.

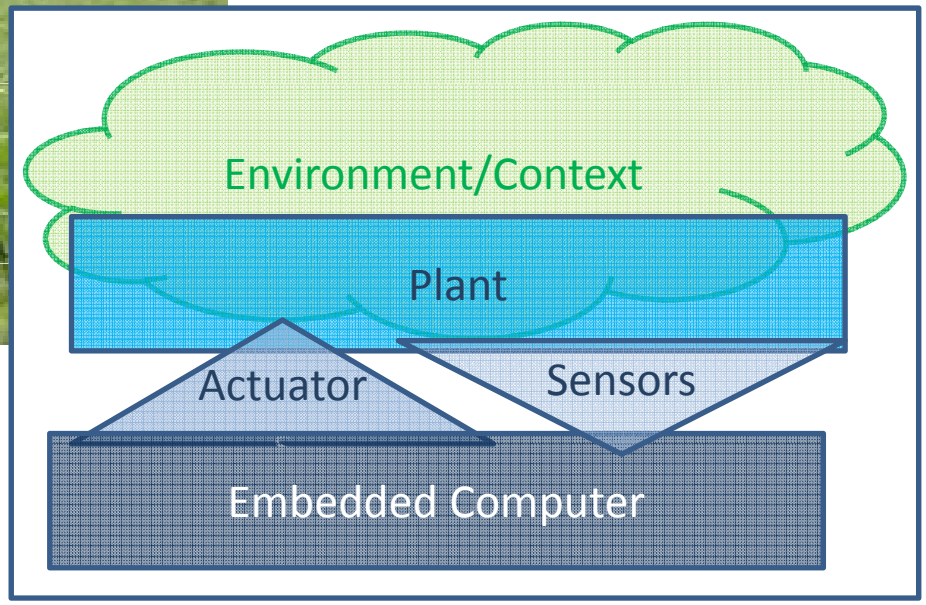
Engineering - who does what ?



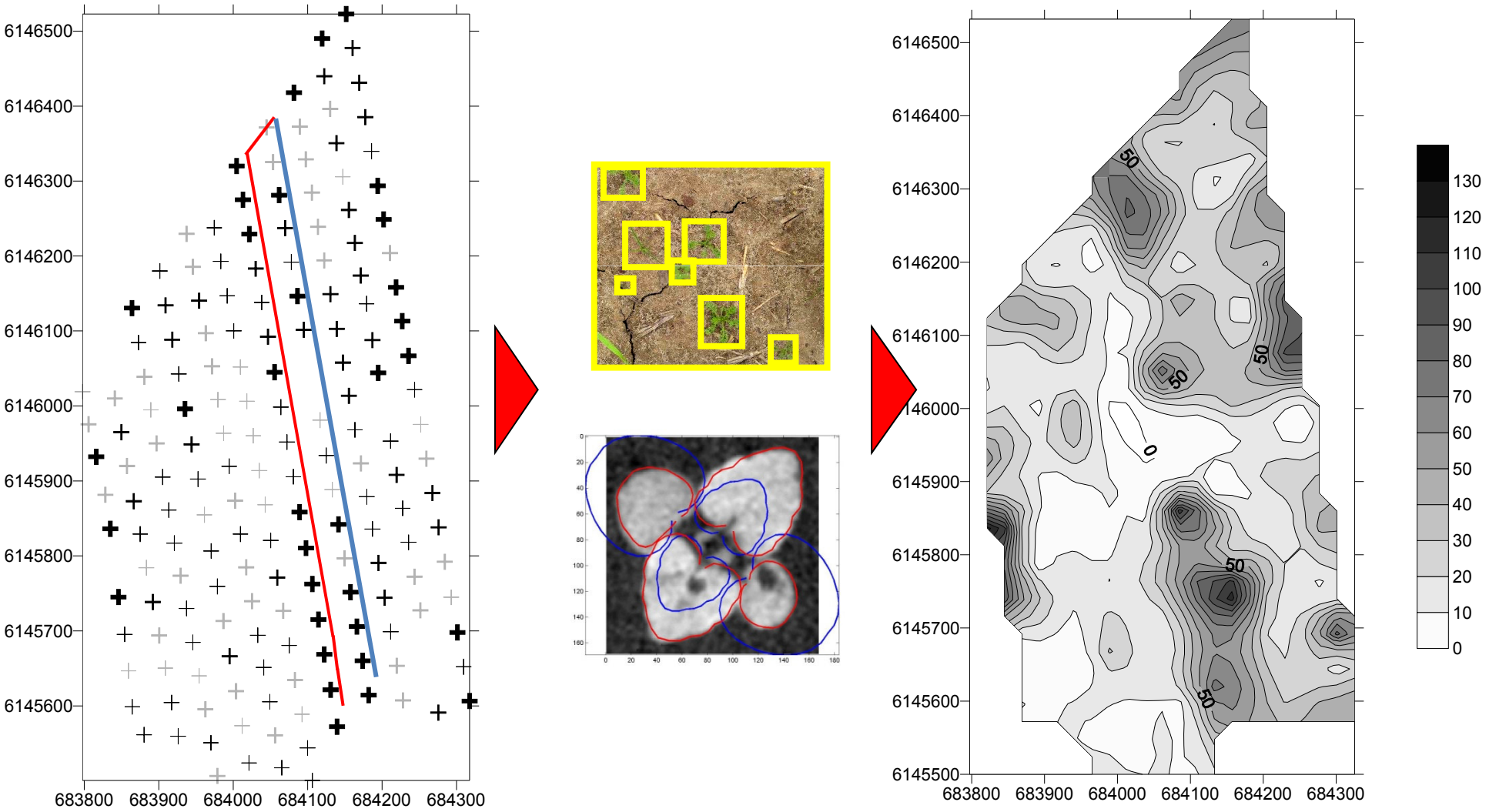
A robot -

Robot – CP-system





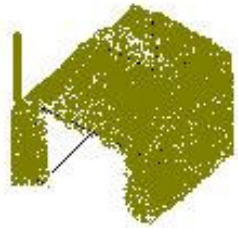
Mission



(Heisel, T., et al, *Weed Research*, 1996, Vol. 36: 325-337)

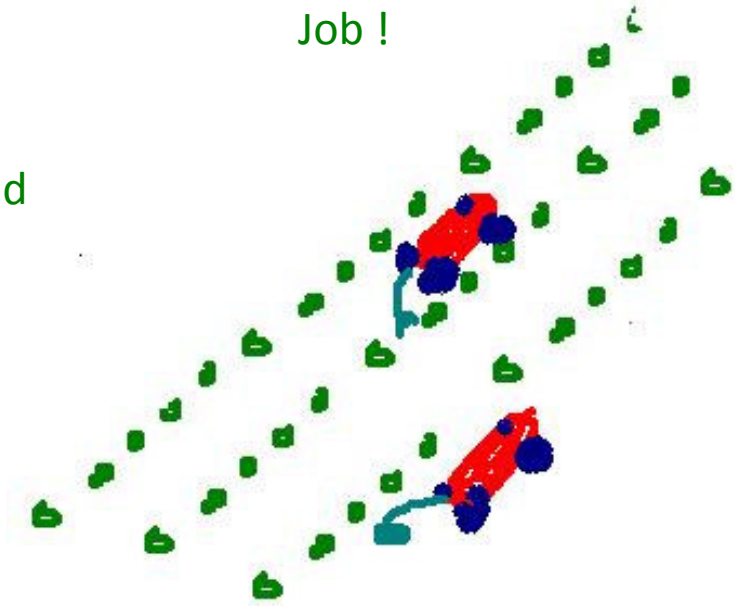
The Concepts

Time, Position



Job !

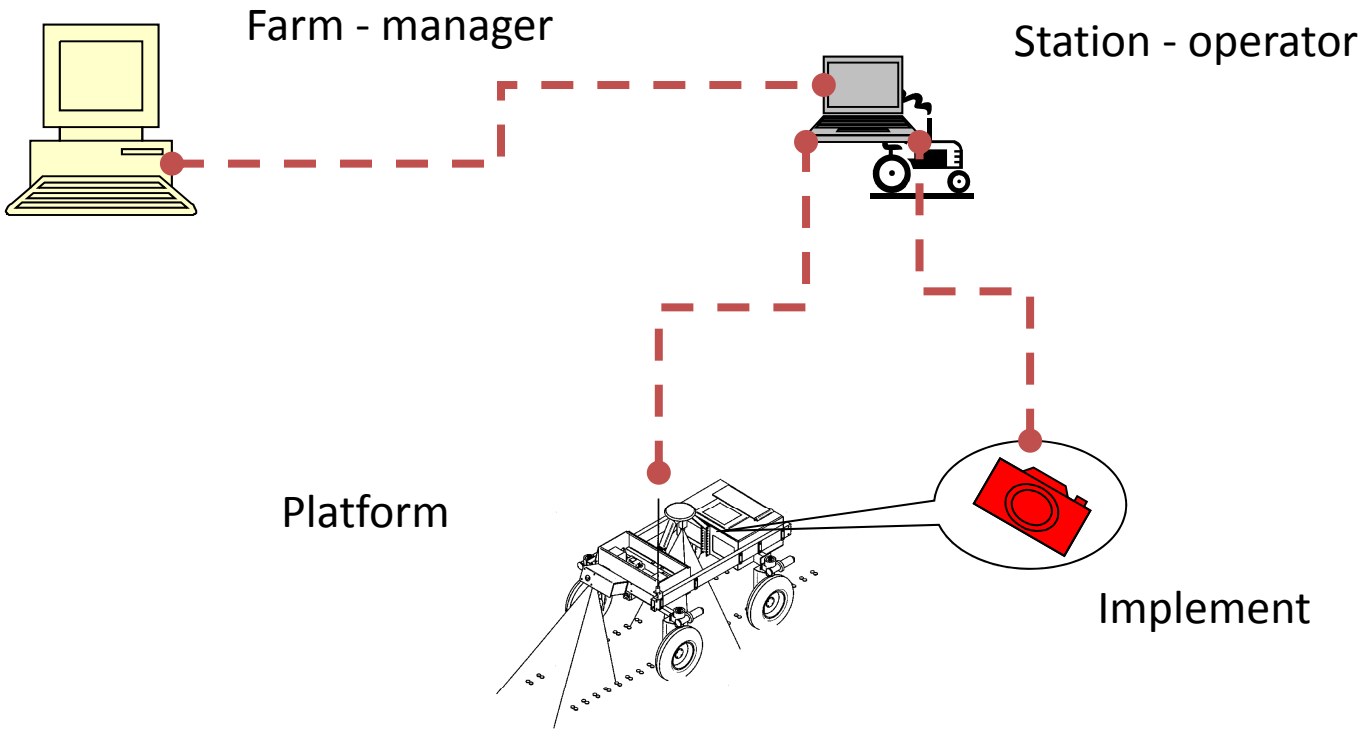
Field



Farm Management

(October 1999)

API main objects



How do we make it a system ?

Object Oriented Analysis and Design

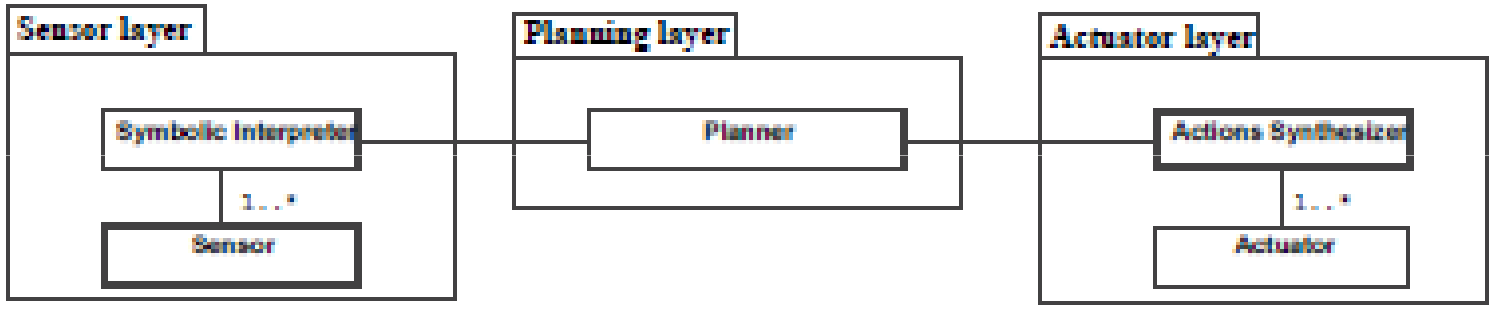
1. **Identfy the Problem Domain (Plant in Context)**
   - rich picture
   - system definition
   - plant model and identification
2. **Identify the Application Domain (Functionality/Control)**
   - functions (use cases)
   - temporal constraints
   - interfaces to the plant (actuators and sensors)
3. **Design**
   - architecture
   ...

MDD, SysML, …

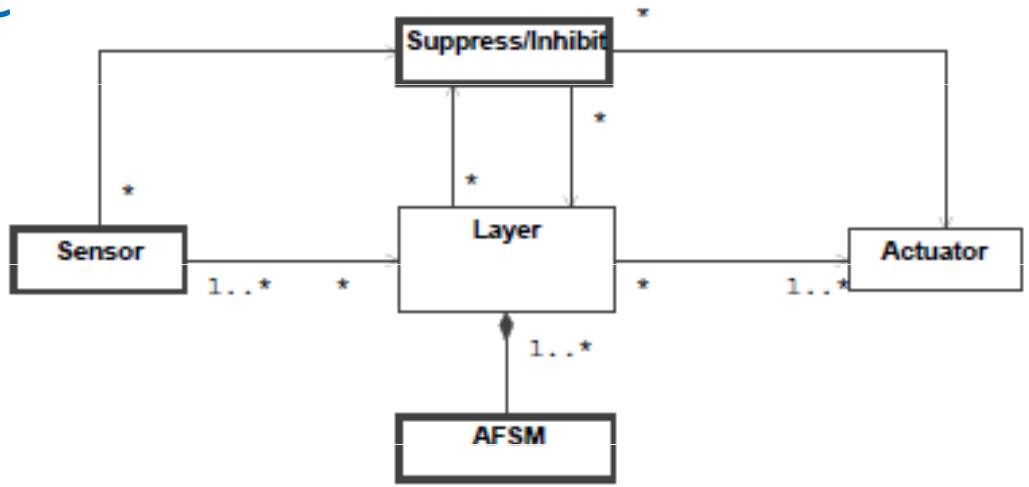
Lars Mathiassen, Andreas Munk-Madsen, Peter Axel Nielsen and Jan Stage, *Object-oriented Analysis and Design*, MARKO Publishing, Aalborg 2000.

A note on robot architecture

Deliberative



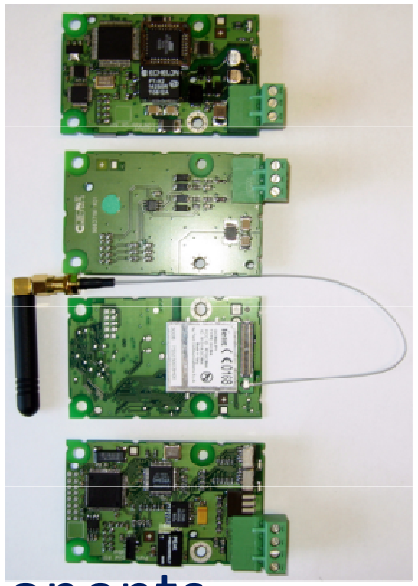
Reactive



Embedded Software

"Embedded software is software integrated with physical processes. The technical problem is managing time and concurrency in computational systems".

E. A, Lee: *The future of embedded software,*
ARTEMIS Conference, Graz, 2006.



Characteristics of a Real-Time Embedded System

- Timing Constraints
- Dependability Requirements
- Concurrent control of separate components
- Facilities to interact with special purpose hardware

Alan Burns and Andy Wellings:
Real-Time Systems: Ada 95, Real-Time Java and Real-Time POSIX
(4th ed), Addison-Wesley, 2010

C versus Java

C

- Well known
- Mature compilers
- Close to the processor
- Liberal typing and checks

Java

- Yet new
- Mostly interpreted
- Platform independent
- Object oriented
- Strict typing and checks
- Concurrency
- Automatic Memory allocation

- ***Timing Constraints***
- ***Dependability Requirements***
- ***Concurrent control of separate components***
- ***Facilities to interact with special purpose hardware***

C dominates embedded software programming - why?

”The UNIX kernel consists of about 10.000 lines of C code and about 1.000 lines of assembly code.

The assembly code can be further broken down into 200 lines included for efficiency (they could have been written in C) and 800 lines to perform hardware functions not possible in C.”

K. Thompson: UNIX Implementation,
The Bell System Technical Journal, **57**, 6, 1978

Real-Time Java
- timing and memory constraints

- Real-Time Specification for Java (2002),
  Java Community Process, JSR-1 – real-time specification for Java.

  high resultion time, clocks, real-time threads, schedulers, memory areas
- Ravenscar Java: A high-integrity profile for real-time Java (2005) Jagun Kwon, Andy Wellings, Steve King

  restrictions on threads, schedulers, memory areas
- Safety Critical Java (2013?) Java Community Process,

  predictability and analyzability

Safety Critical Java

- Periodic Event Handlers
- Aperiodic Event Handlers

collected in a Mission

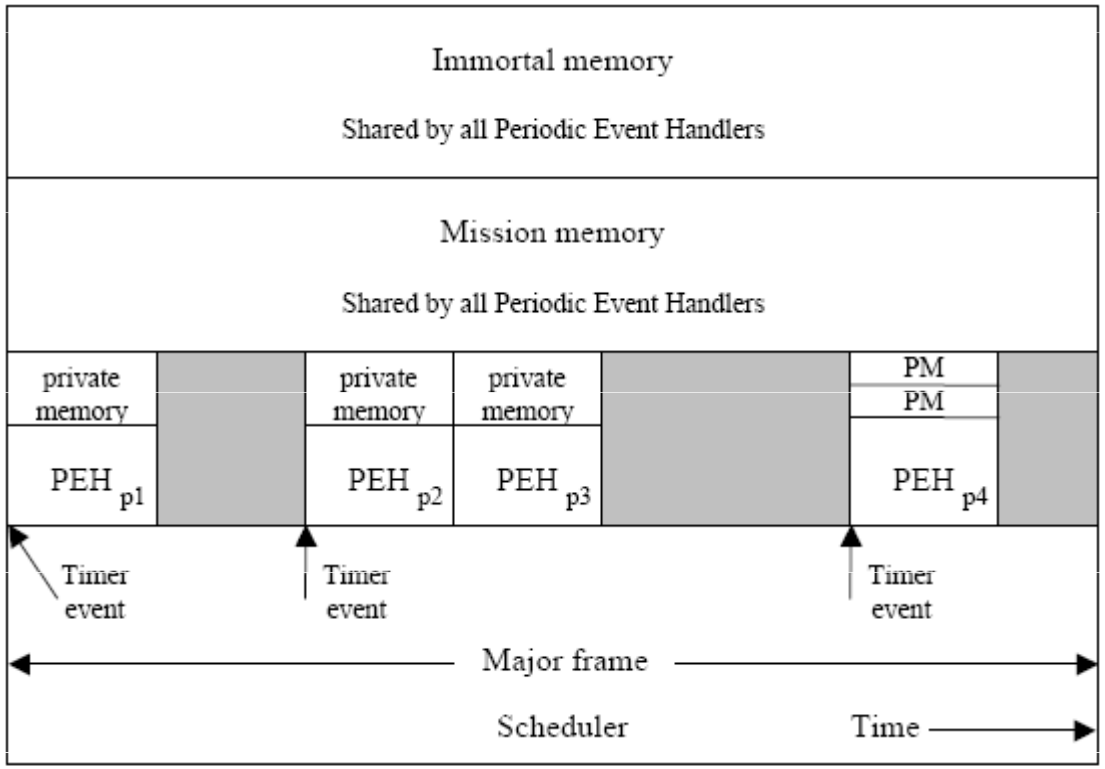
Each handler

- ➢ has a private Memory
- ➢ is Scheduled

A mission

- ➢ has a mission Memory with synchronized shared objects

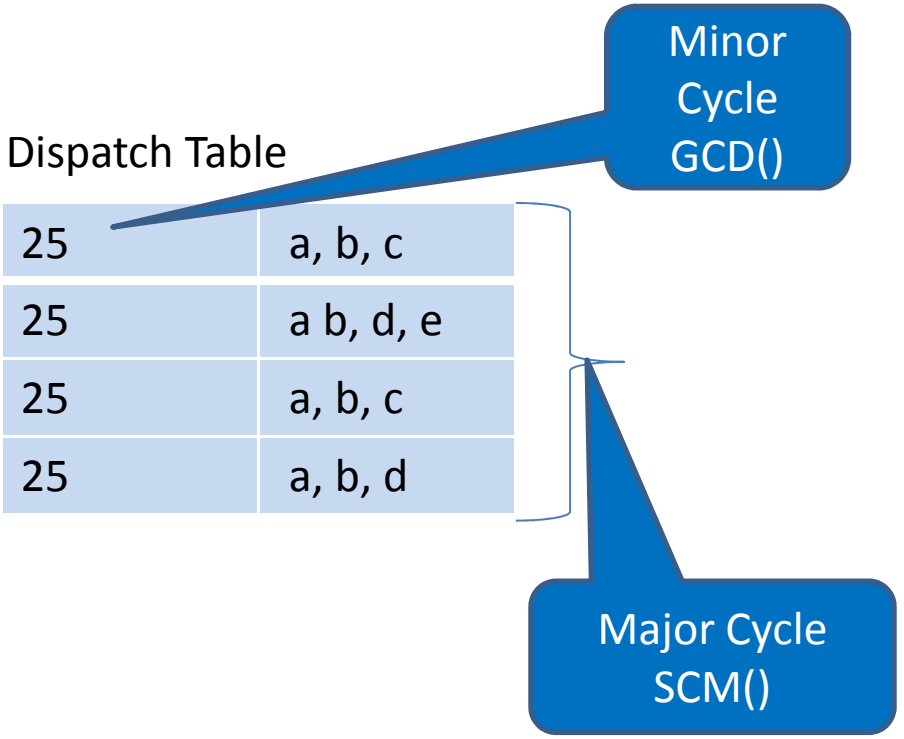
Level 0: cyclic executive



Cyclic Executive

Table driven static scheduling

Task	Period	WCET
a	25	10
b	25	8
c	50	5
d	50	4
e	100	2



A. Burns and A. Wellings*: Real-Time Systems and Programming Languages,* Ch 11.1

Level 0 – Safety Critical Java

```
public final class CyclicSchedule {
  CyclicSchedule(Frame [] frames) { … }
public final class Frame
  Frame(RelativeTime duration,
        PeriodicEventHandler [] handlers)
```

25	a, b, c
25	a b, d, e
25	a, b, c
25	a, b, d

Handlers

```
public PeriodicEventHandler(
                PriorityParameters priority,
                PeriodicParameters release,
                StorageParameters storage)

public abstract void handleAsyncEvent();
```

Periodic Parameters

```
public class PeriodicParameters  {

public PeriodicParameters(RelativeTime start,
                          RelativeTime period)
 { … }

public PeriodicParameters(RelativeTime start,
                          RelativeTime period,
                   RelativeTime deadline,
                   AperiodicEventHandler missHandler)
 { … }
```

Priority Parameters

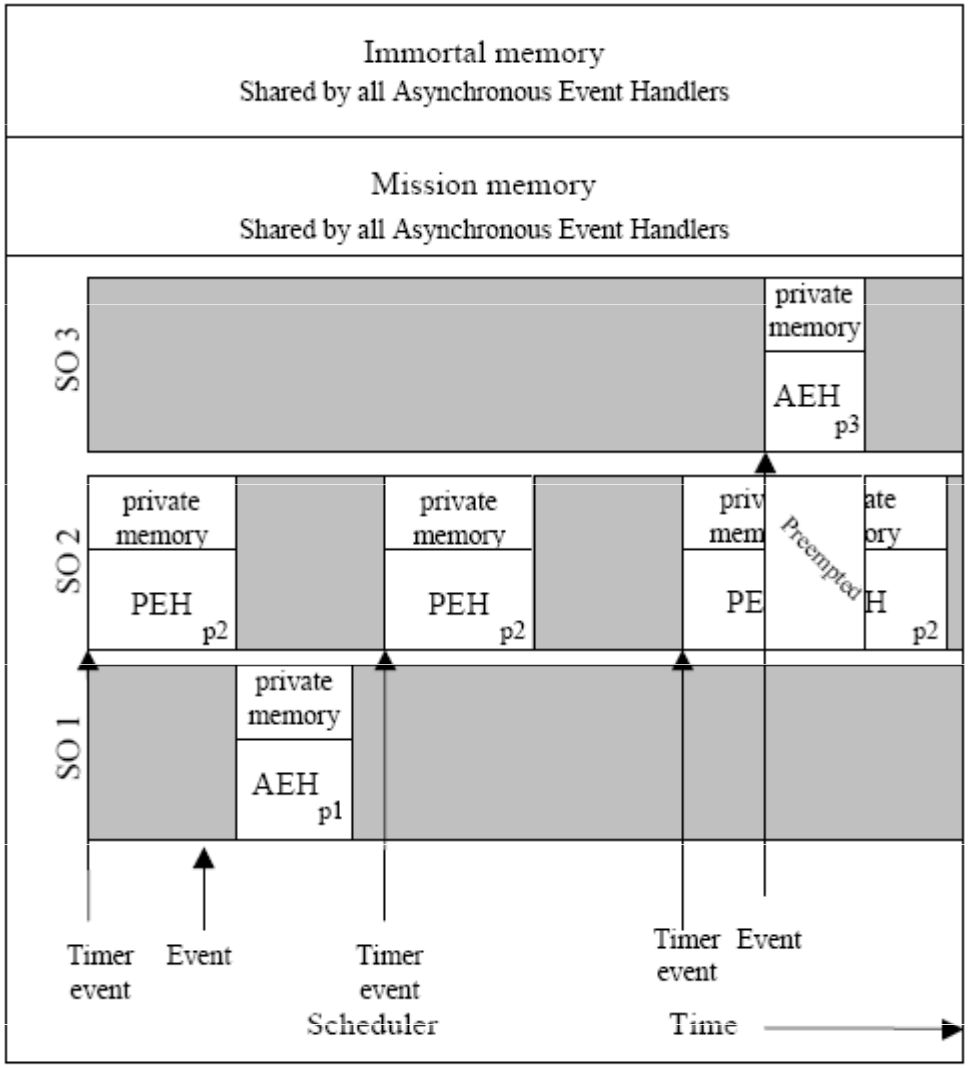
public class PriorityParameters extends SchedulingParameters {

public PriorityParameters(int priority)
{ … }

PriorityScheduler.instance(). getMaxPriority()

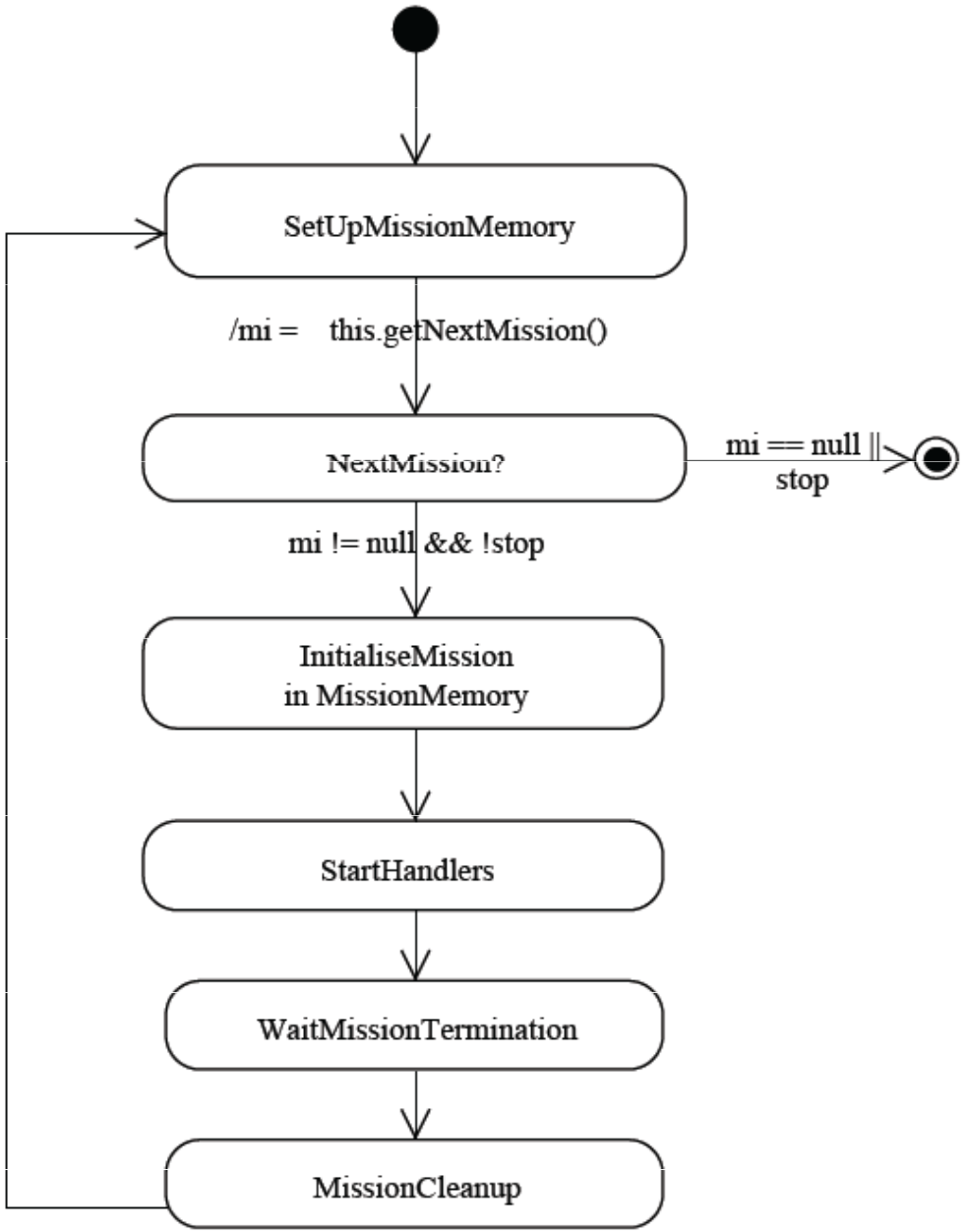
PriorityScheduler.instance(). getMinPriority()

Level 1: fixed-priority preemptive scheduler



SO 3 has the highest priority

Missions


SetUpMissionMemory
/mi = this.getNextMission()
NextMission?
mi == null ||
stop
mi != null && !stop
InitialiseMission
in MissionMemory
StartHandlers
WaitMissionTermination
MissionCleanup

Analysing R-T properties for FPP

B	Worst-case blocking time for the process
C	Worst-case computation time (WCET)
D	Deadline of the process
I	The interference time of the process
P	Priority assigned to the process
R	Worst-case response time of the process
T	Minimum time between releases(process period)

$$R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j$$

Summary of Topics

- Cyber-physical systems
- Robots
- OOAD
- Robot Architecture
- R-T Programs
- R-T Program Schedulability Analysis