Organic Fault-Tolerant Control Architecture for Robotic Applications

Werner Brockmann, Erik Maehle
University of Lübeck
Institute of Computer Engineering
Director: Prof. Dr.-Ing. E. Maehle

Karl-Erwin Großpietsch
Fraunhofer-Institut für
Autonome Intelligente Systeme
Schloss Birlinghoven, Sankt Augustin

Karlsruhe, 14./15.07.2005

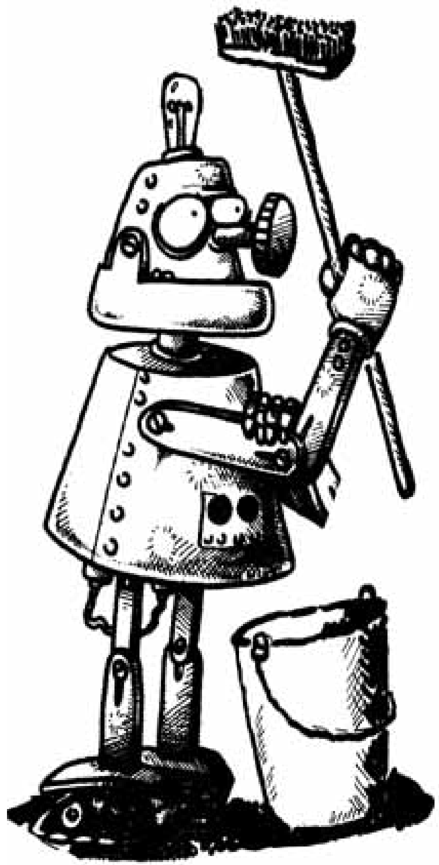
- Introduction
- ORCA Architecture
- Application Example
- Methods
- Conclusion

Motivation

Intended applications: autonomous mobile robots in human environments

-> unstructured, dynamically changing environment

-> no explicit model of the environment



-> complex control systems

-> no explicit fault model

fault-tolerance, safety

engineering bottleneck

Organic Systems

Organic systems adapt dynamically to environment and malfunctions

- uncertainties
- unknown environment
- unforeseen situations

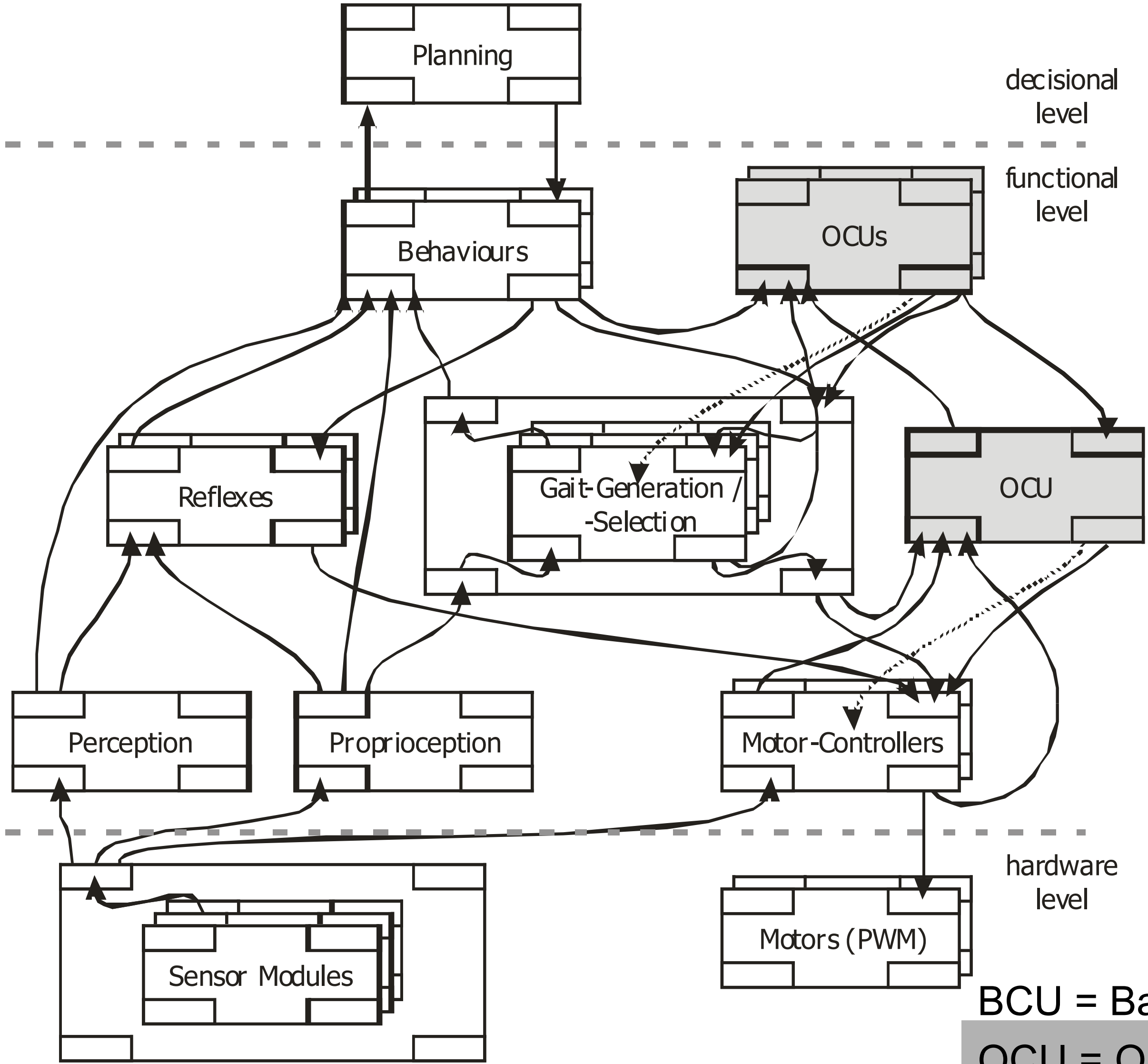
- errors
- faults

Our approach: **controlled self-organization**

Inspiration: autonomic nervous system and immune system

Aim:
- detect, react, and adapt to malfunctions
- avoid critical system states at any time
- low cost implementation and engineering

ORCA - Organic Robot Control Architecture

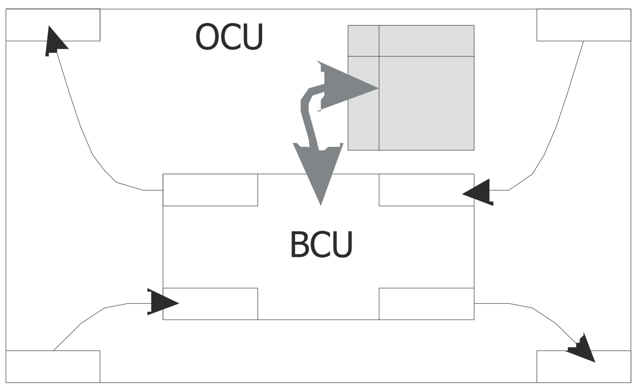


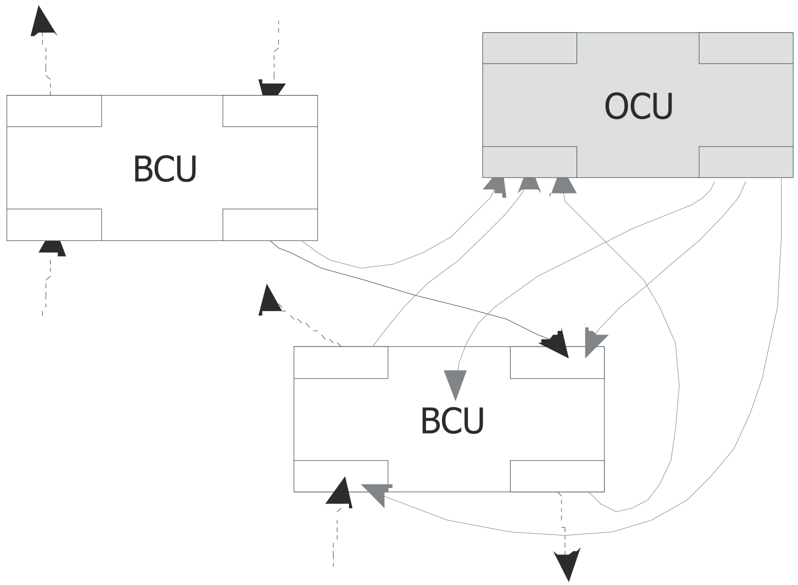
BCU = Basic Control Unit

OCU = Organic Control Unit

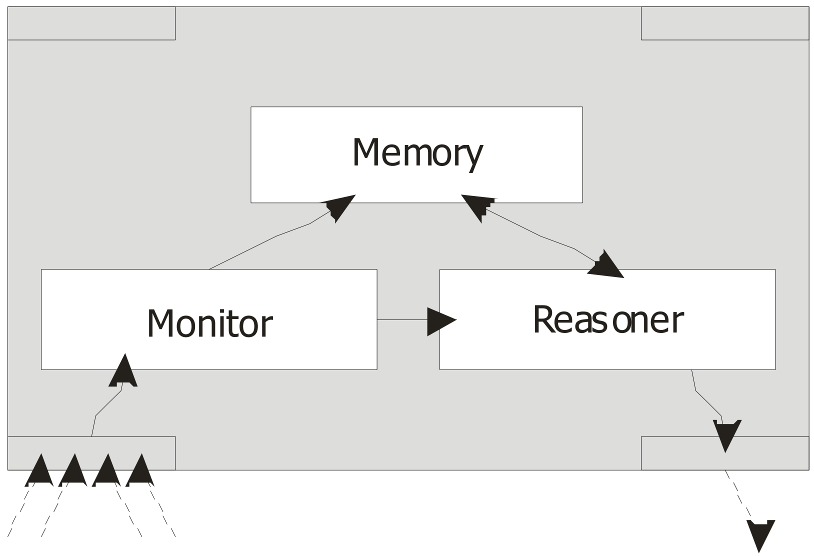
OCU-BCU-Interaction

Software architecture DAISY (Distributed Architecture of Intelligent SYstems)





OCU-Architecture



- Monitor: anomaly detection
- Memory: short term history
- Reasoner: hard real-time determination of a counteraction

Climbing Robot DEXTER

(DEXTerous Electric Rover)

Technical data:

- size 36,5×22×13 cm^3
- weight about 3 kg
- 4 active degrees of freedom
- passive suction cups
- IR- and US-sensors on feet

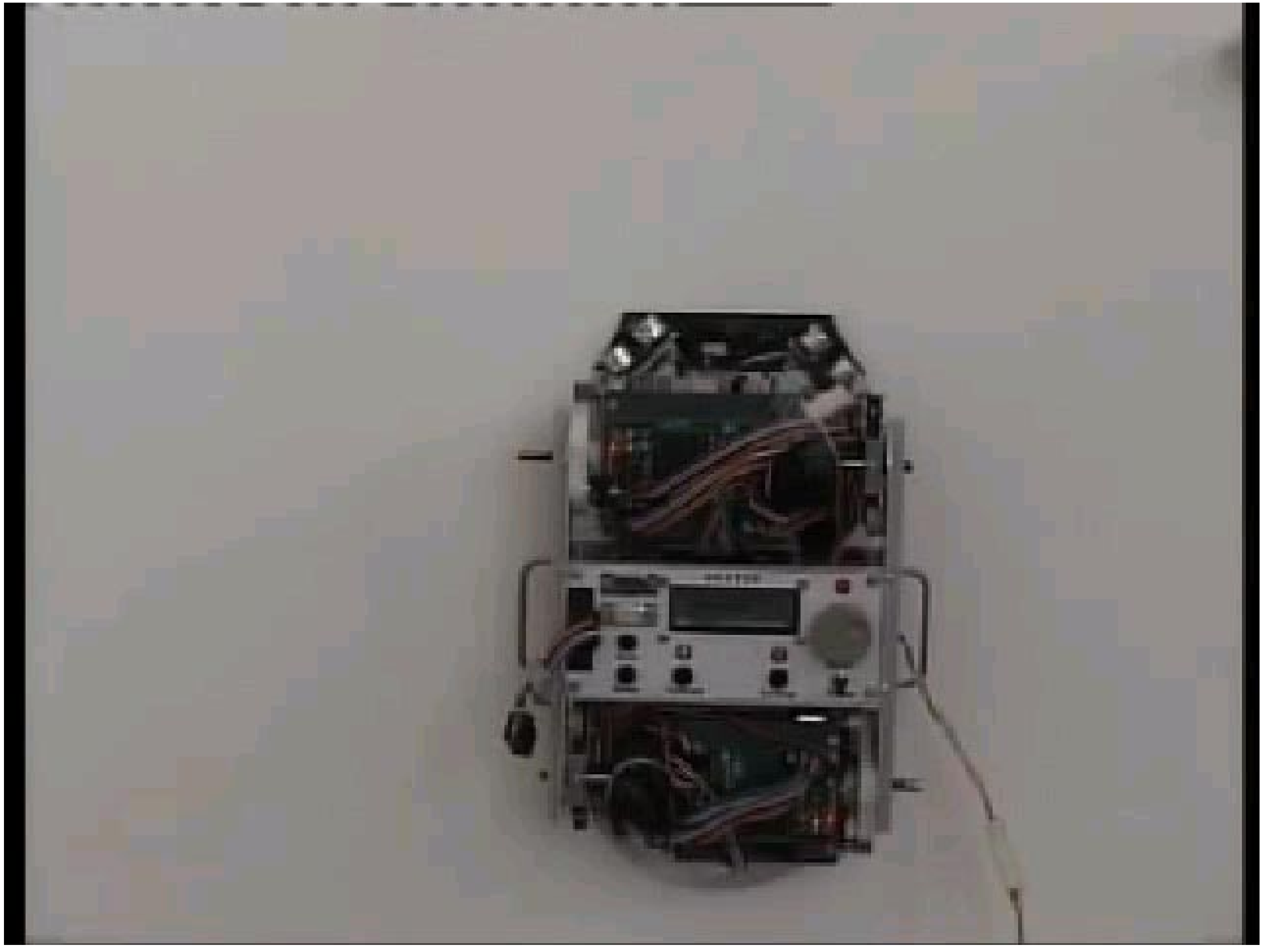
Characteristics:

- distributed control system
- alternative behaviors
- incomplete perception

Application Example

Climbing Robot DEXTER

(DEXTerous Electric Rover)



Methodological Approaches

- signals reflecting the ´health´ of a signal or a BCU

  e.g. - noisiness, confidence of output results,

  - load state; error state

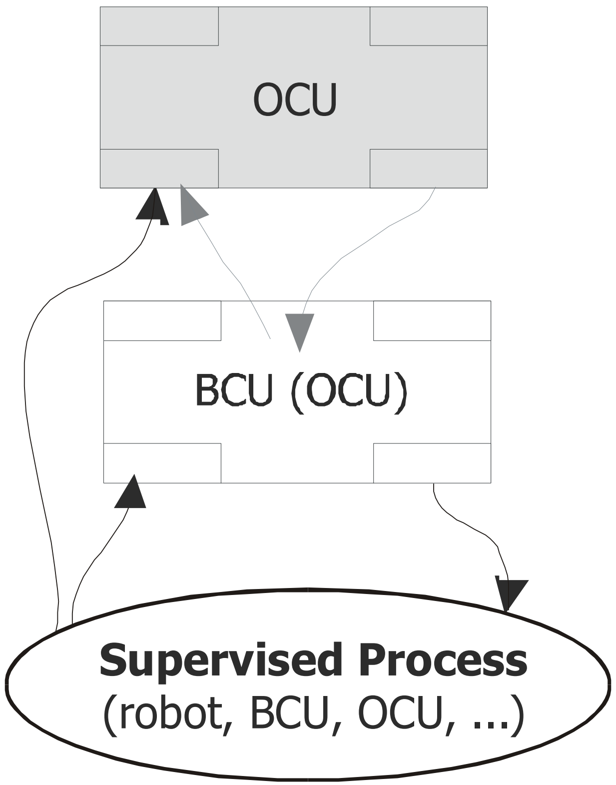
- adaptive action selection

```
IF   movement=blocked
THEN activation of commanded behaviour -=5%,
     activation of non-commanded behaviour +=5%
```

- learn to treat malfunctions

  -> **learning** at BCU- and OCU-level

Supervised Learning



- no formal models

- fast adaptation

- manageability

- controlled self-organization (safety)

    -> hybrid crisp-fuzzy systems

- high-dimensional problems

    -> adaptive filters

Hybrid Crisp-Fuzzy Systems

Key features:

- **granular computing** (partitioning of the input space)
- mixed crisp and fuzzy processing
- normalization of rule-base
- rule-based specification as well as learning
- very fast computation, hard real-time learning
- decomposition to fight complexity
- individual **attributes** for each partition

-> tag-bit for **guided online-learning**

-> **safety warranty**

Adaptive Filters

Key features:

- complexity reduction of the control space
- weighted summation out of „**history array**"
- learning of **linear parameters**
- minimizing the difference between desired and measured system behaviour
- at higher system levels, combination with hybrid crisp-fuzzy systems
- used for
  - -> health monitoring of local components
  - -> compensation of small local malfunctions

Conclusion

Application: fault-tolerant autonomous mobile robots

-> flexible, but safe adaptation to malfunctions (unforeseen events, errors, faults, …)

Architecture: mimic autonomic nervous system and immune system

-> **ORCA-architecture + learning**

Methods: - ´health´-status

- **controlled self-organisation**

-> hybrid crisp-fuzzy systems

-> adaptive filters