DodOrg

Digital On-Demand Computing Organism
DodOrg

Startkolloquium – DFG SPP 1183 “Organic Computing”

Karlsruhe, July 15th 2005

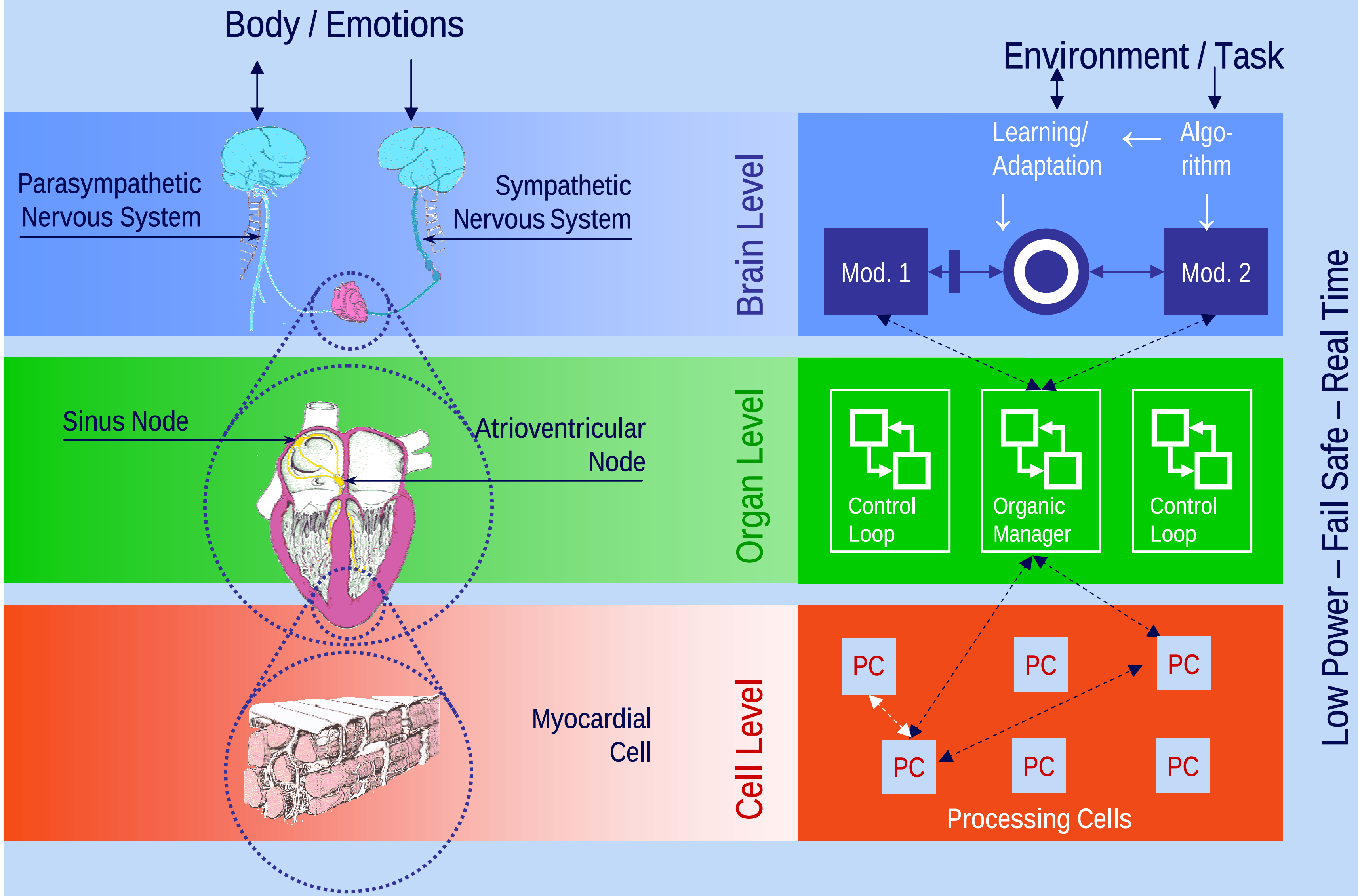
Jürgen Becker [1], Kurt Brändle [1], Uwe Brinkschulte [2], Jörg Henkel [2], Wolfgang Karl [2], Heinz Wörn [2]

[1] Fakultät für Elektrotechnik und Informationstechnik

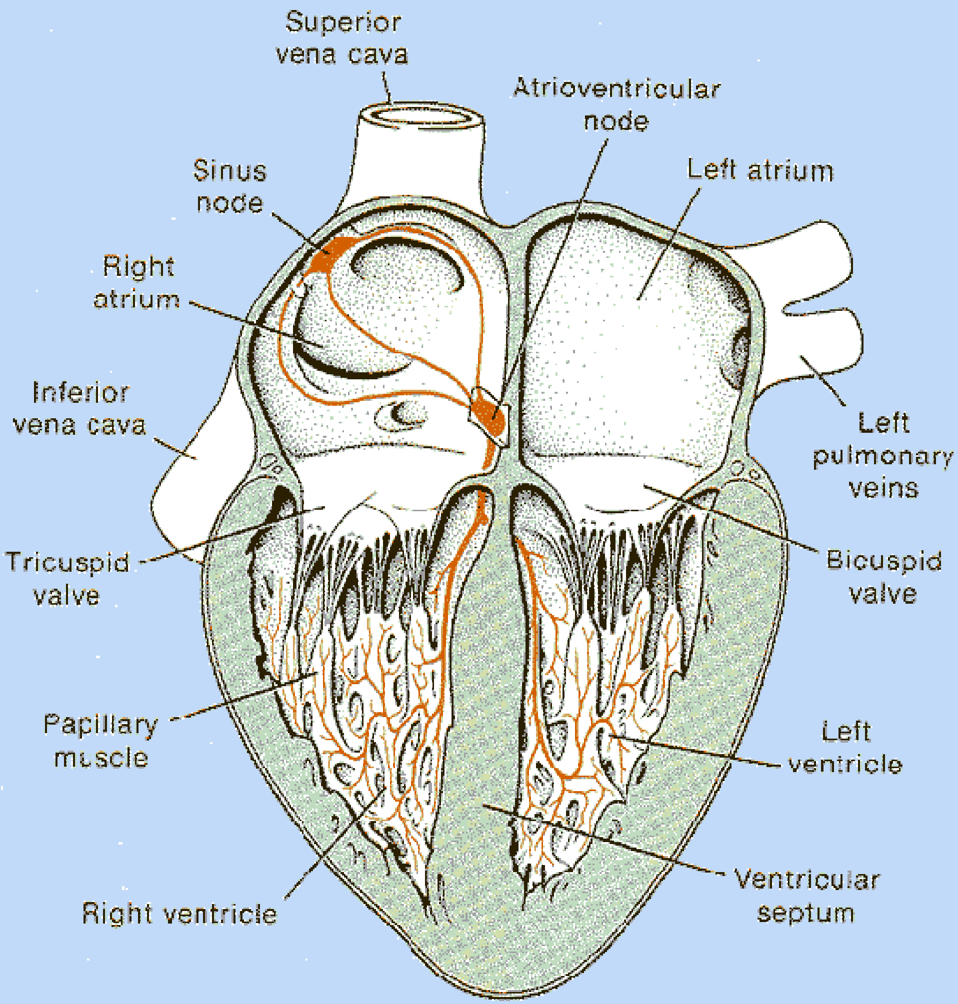
[2] Fakultät für Informatik

Universität Karlsruhe (TH)

From Biology towards an Organic Computing System



The Mammalian Heart
Anatomy

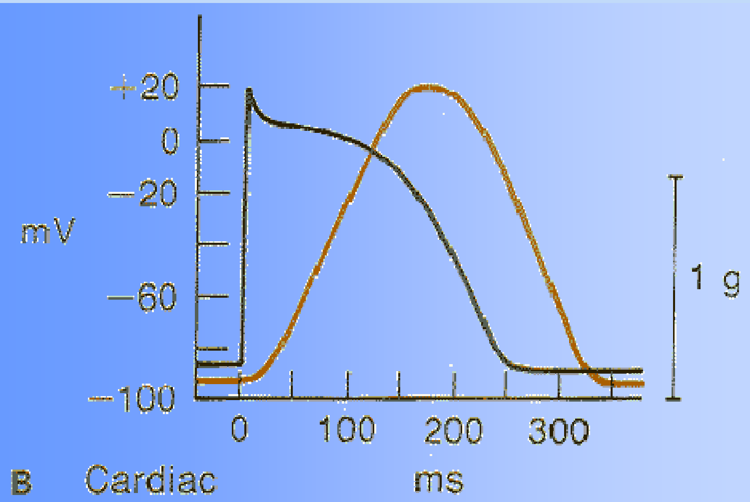


Mammalian heart as model for DodOrg research application

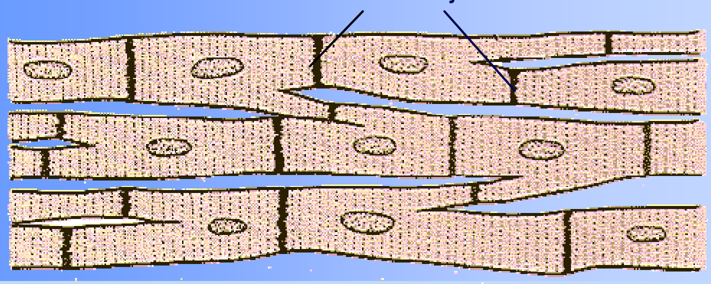
- Matches the proposed hardware structure very well
- Hierarchical order of functionality
- Some features overlap all levels, such as low power consumption and fail safe

Inspired by nature, but not copying it

DodOrg



Relationship between electrical (black) and mechanical (brown) activity of cardiac muscle fibers (below)



Lowest level: Cardiac muscle fibers

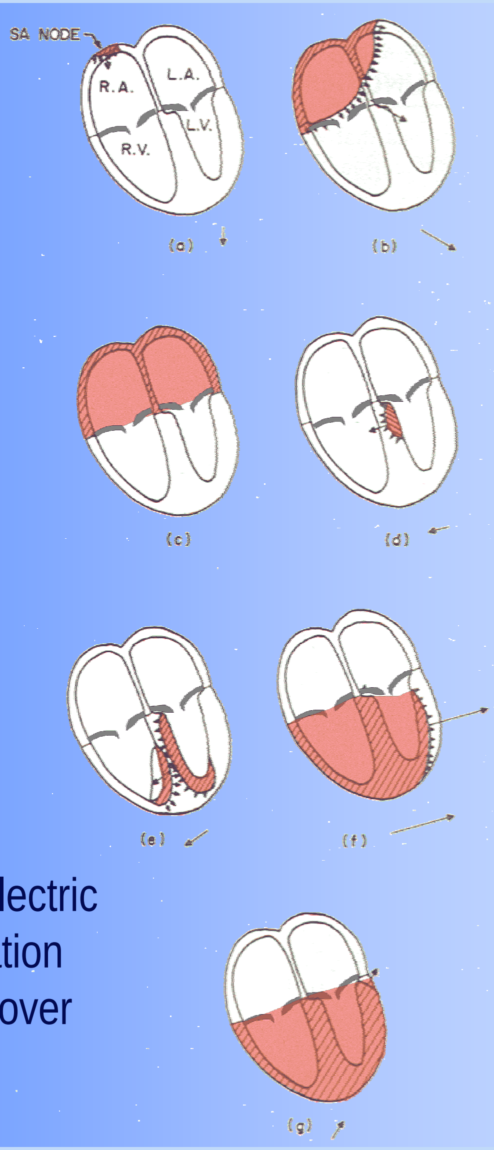
Task: Contraction in response to electric stimulation

- Specific autonomous functionality
- Inflexible response in course and time to electric depolarization of the cellular membrane
- Depolarization transmitted to neighbor cells

"Self-healing"

- In case of malfunction or failure neighbor cells will take over the tasks

Comparable to Processing Cells



Wave of electric depolarization sweeping over the heart

Middle level: Coordination of the contraction of different myocard areas

Task: Achieve pumping movement

- Sinus node initiates contraction of atria
- Electric excitation spreads over the muscles of both atria to the atrio-ventricular node
- Atrio-ventricular node passes wave of excitation via the His bundle to the tip of the heart
- Contractions of the ventricles start from the tip and expand towards the ventricular basis

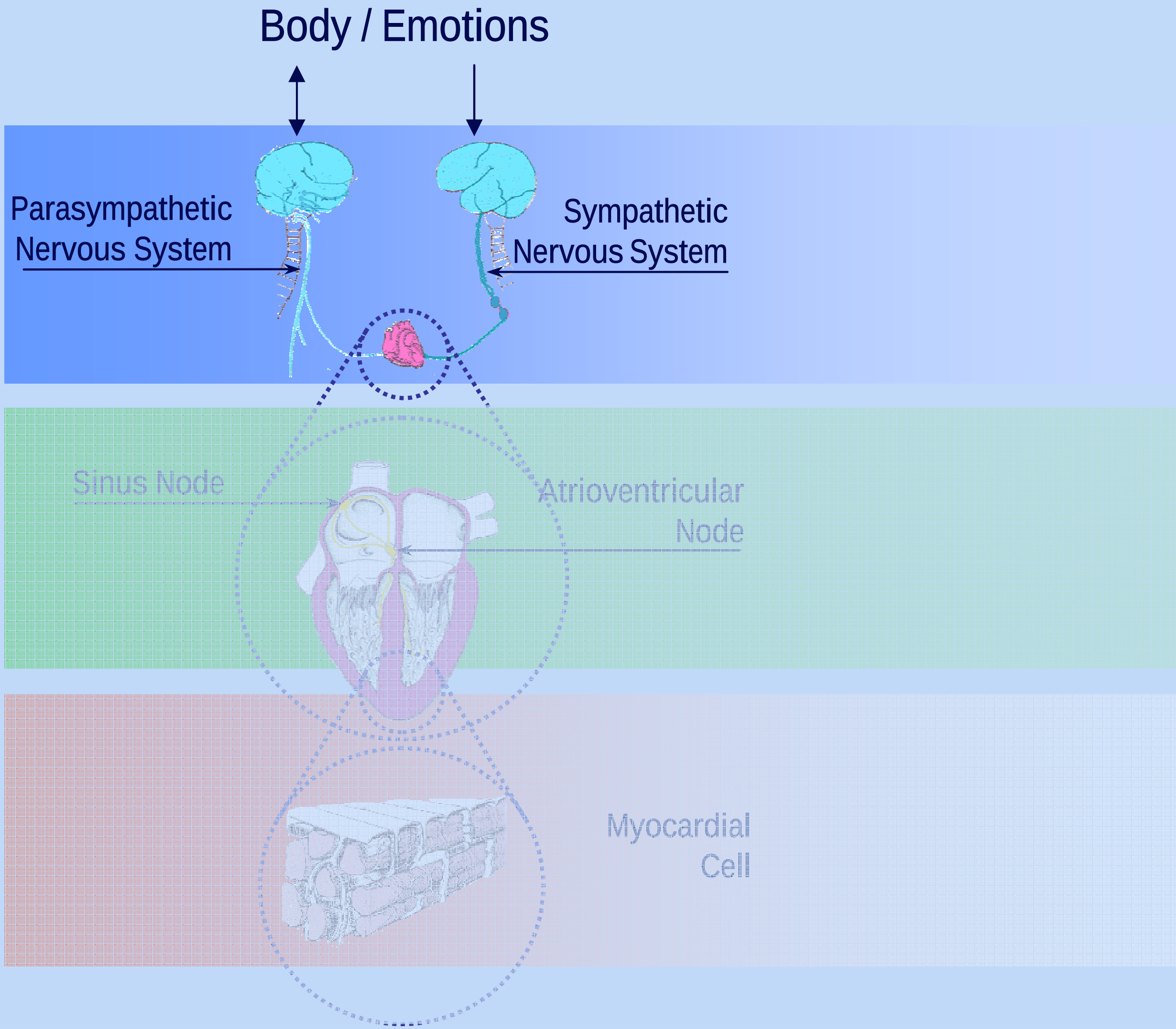
Comparable to middleware

Top level: Outside events can influence heart action

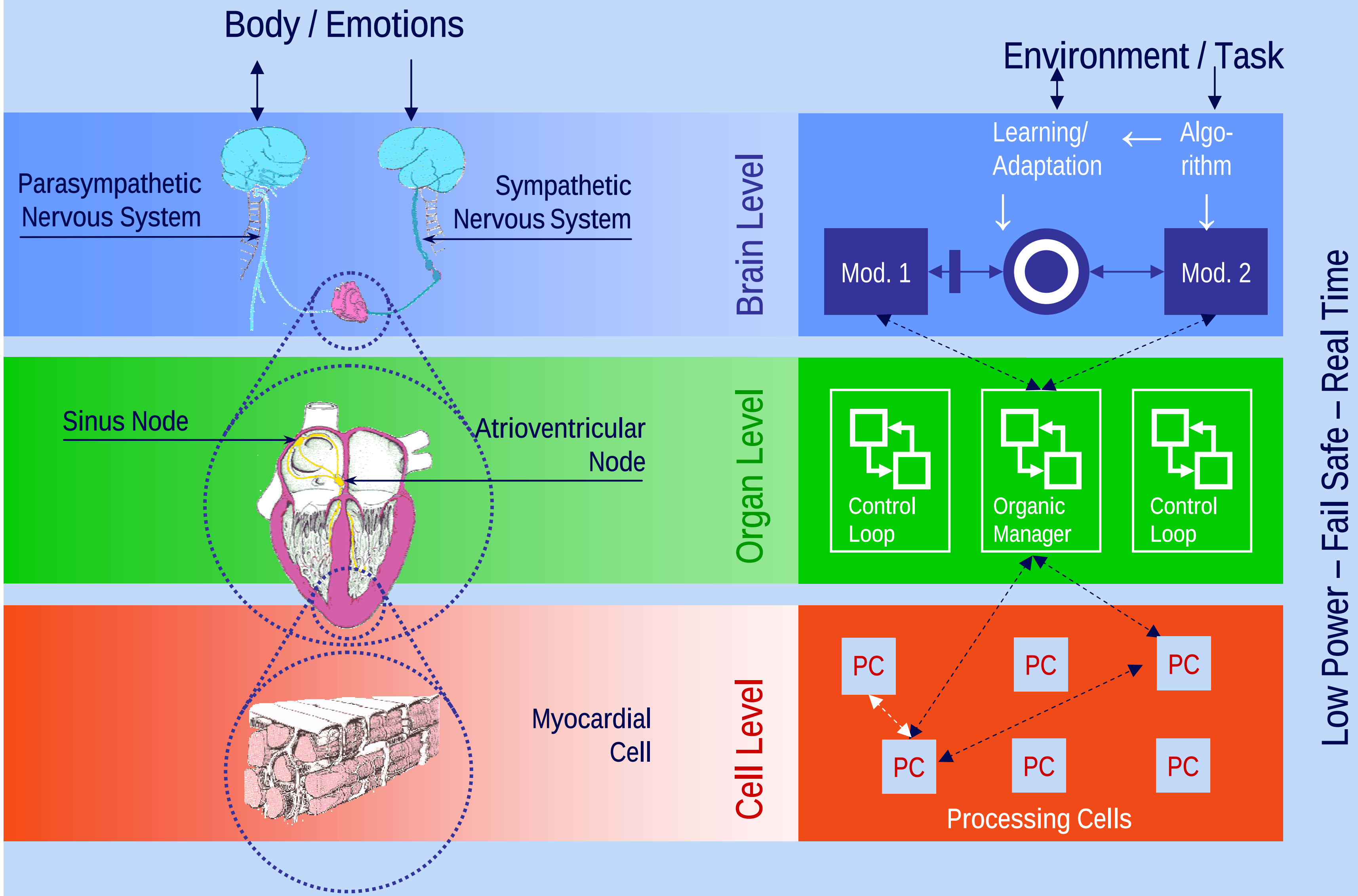
Task: Nervous control of the heart function by the vegetative nervous system

- Sympathetic nervous system increases heart rate and stroke volume
- Parasympathetic nervous system decreases heart rate and stroke volume

Comparable to controlling application

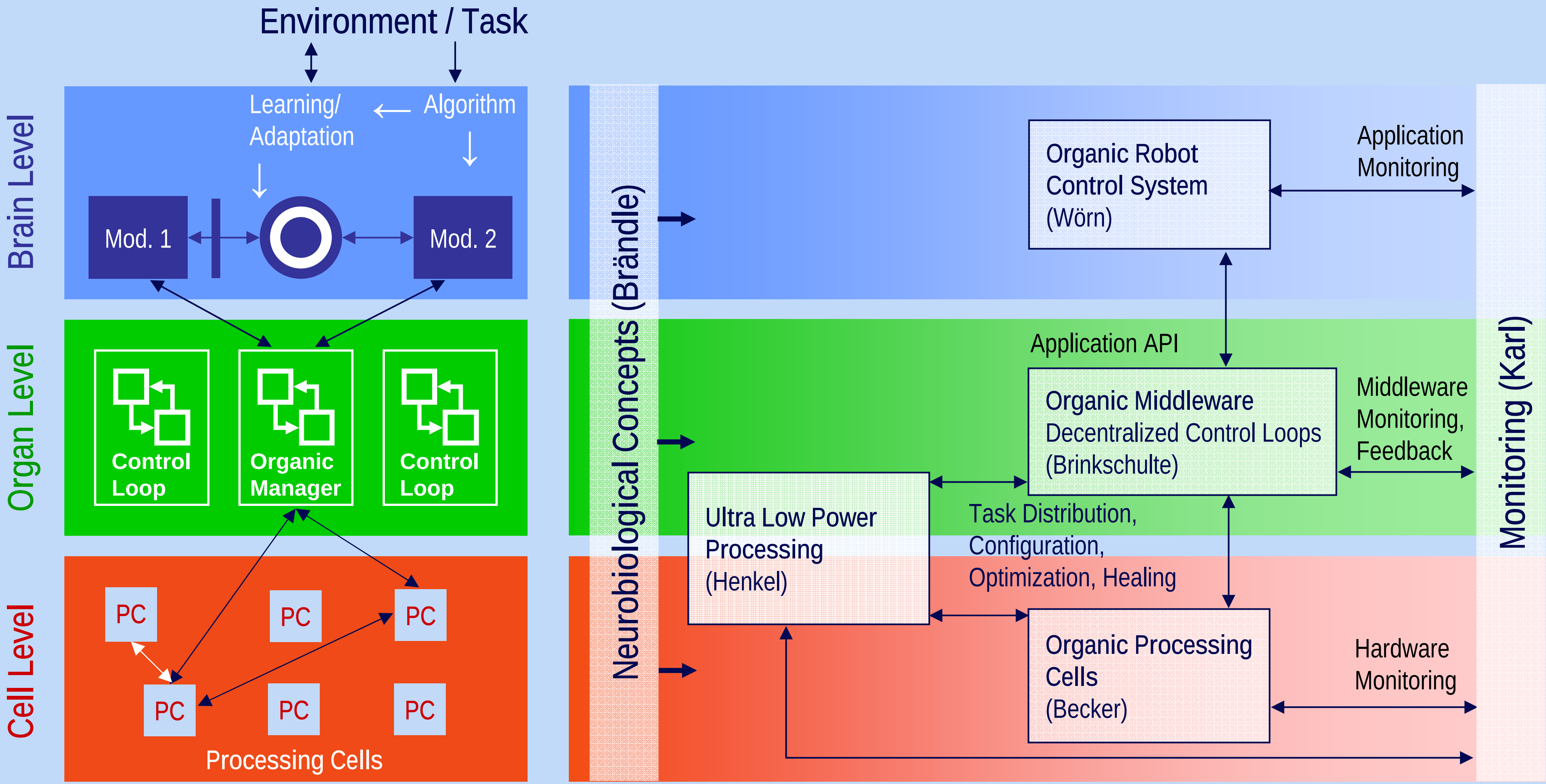


From Biology towards an Organic Computing System



Organization of Sub-Projects

DodOrg



Sub-Project "Organic Processing Cells" (Prof. Becker)

DodOrg

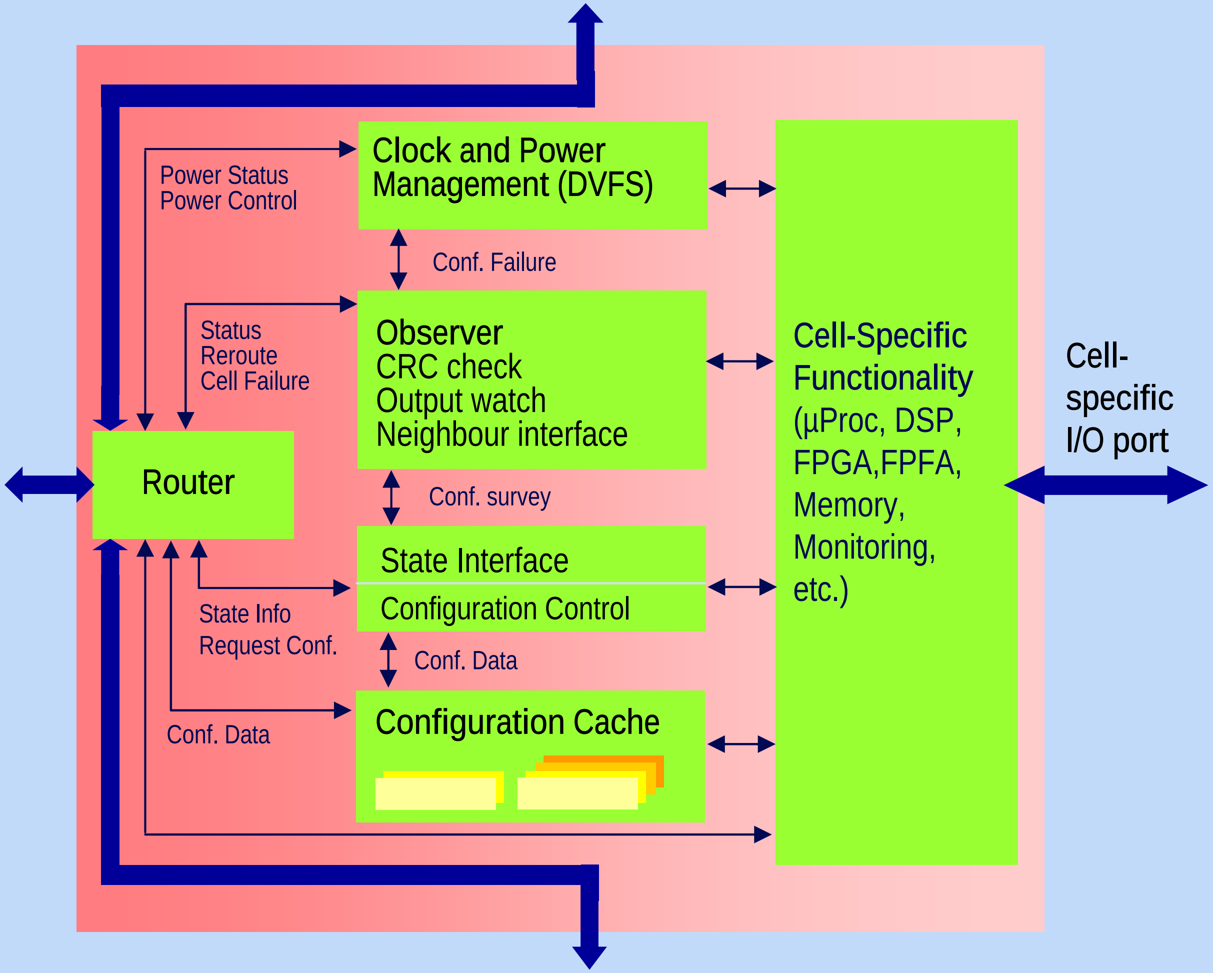
Modularity

- Same "blue print" for all cells
- Common infrastructure
- ▶ Cells can easily take over for defective neighbors
- ▶ Interface for higher-level functions (middleware, monitoring) stays the same

Local intelligence

- Power management
- Basic monitoring facilities
- Configuration
- Router
- Built into each cell

Biological cells are also based on common "blue print" / differentiate only during development



Sub-Project "Organic Processing Cells" (Prof. Becker)

DodOrg

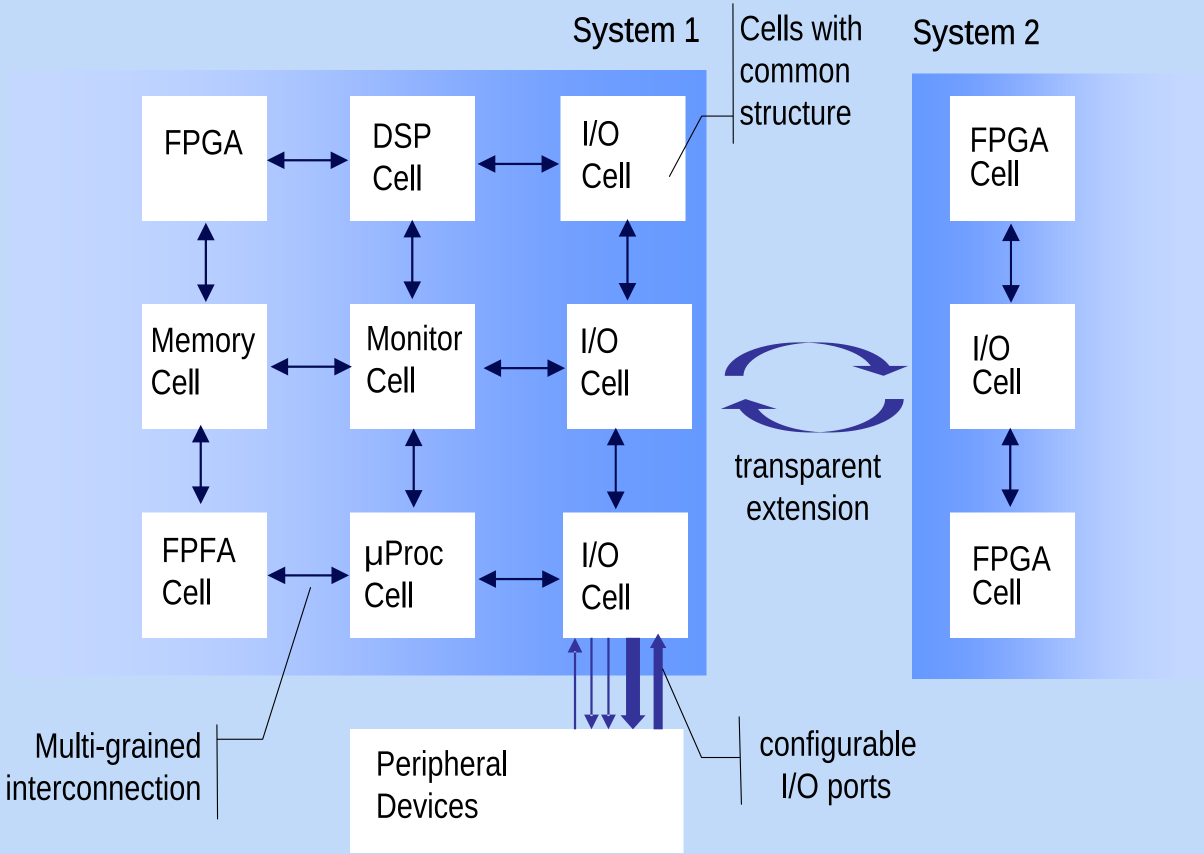
Limited set of cell-specific types of functionality
- Microprocessor / DSP
- FPGA / FPFA
- I/O

Trade-off possible with respect to flexibility of overall system

Multiple implementations of algorithms
- "Self-healing"
- Example: FPGA cell might need to take over from microprocessor cell

Communication facilities transparent across Systems

Basis for organic middleware



Sub-Project “Low power through swap-based system architecture” (Prof. Henkel)

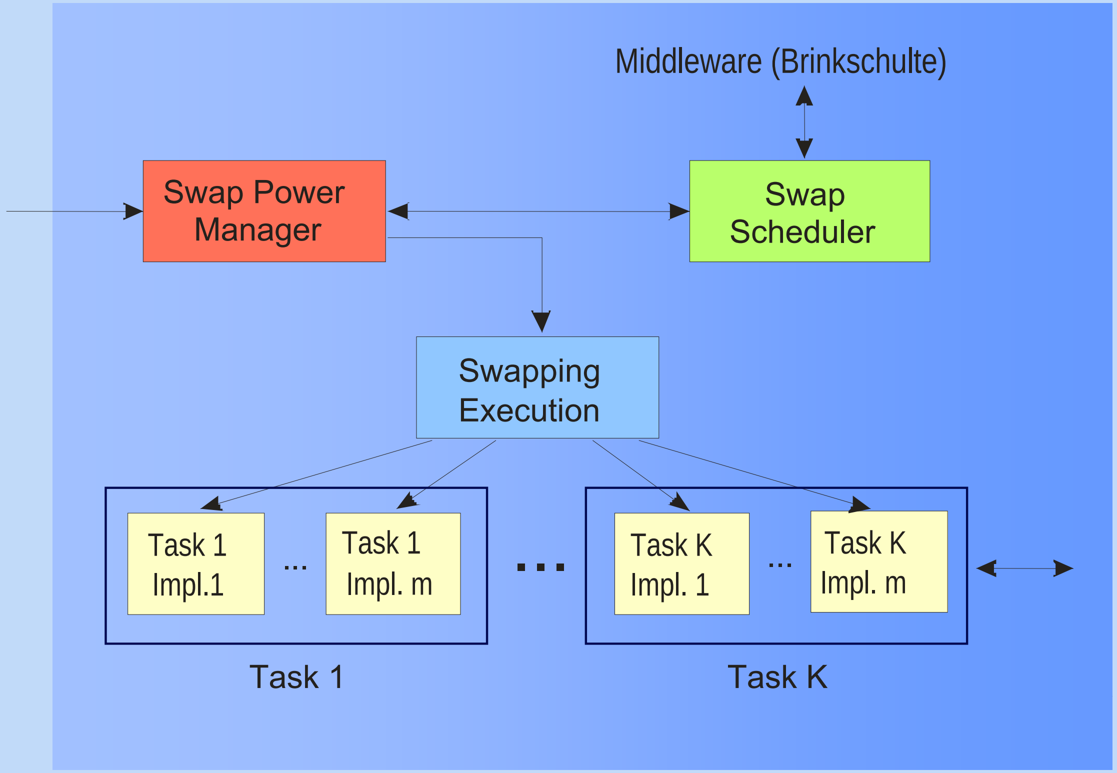
DodOrg

Basic idea

- System should be able to adapt to:
  - Varying (over time) requirements in terms of performance, functionality etc
- The adaptation through “swapping-on-the-fly” ensures that the system is running at any time with minimum resource requirements (ideally without any overhead)

► minimizes power consumption

- Info about system state is retrieved through monitoring



Requirements

- Multiple implementations of one and the same task need to be at the system’s disposal
- At most one of them is in running state at a certain time

Swapping options

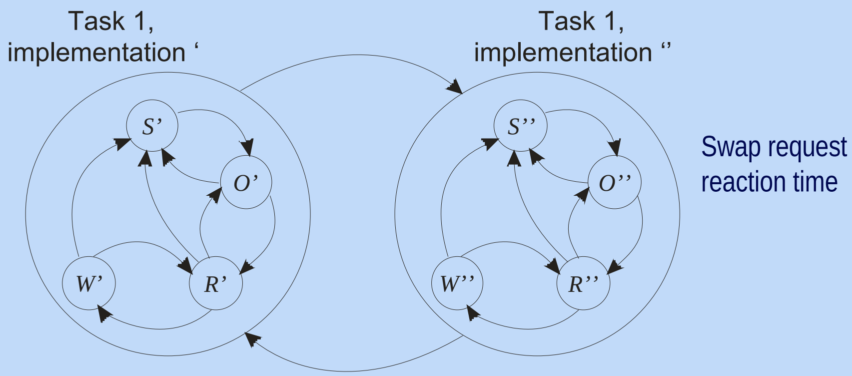
1. A physical implementation can be: software (RAM-based), custom hardware, re-configurable hardware etc. and combinations thereof
2. Alternative algorithmic implementations

- Algorithmic and physical implementation differ significantly in relevant characteristics like power consumption, performance etc.

Core components

- <u>Swap Power Manager</u>: Determines which of multiple task implementations is active and which state that task should assume (see fig below)
- <u>Swap Scheduler</u>: Determines a low power schedule for the swapping process considering partial overlaps, swap process time etc.
- <u>Swap Execution</u>: Seamless swapping of diverse (physical, algorithmic) implementations

“Swapping-on-the-fly” components are located between organic middleware and organic reconfigurable fabric



	Power States	Implementation ‘’			
		S”	O”	R”	W”
Implementation ‘	S’	(x)	(x)	x	x
	O’	(x)	(x)	x	x
	R’	x	x	-	-
	W’	x	x	-	-

Sub-Project "Monitoring"
(Prof. Karl)

DodOrg

Monitoring Approach

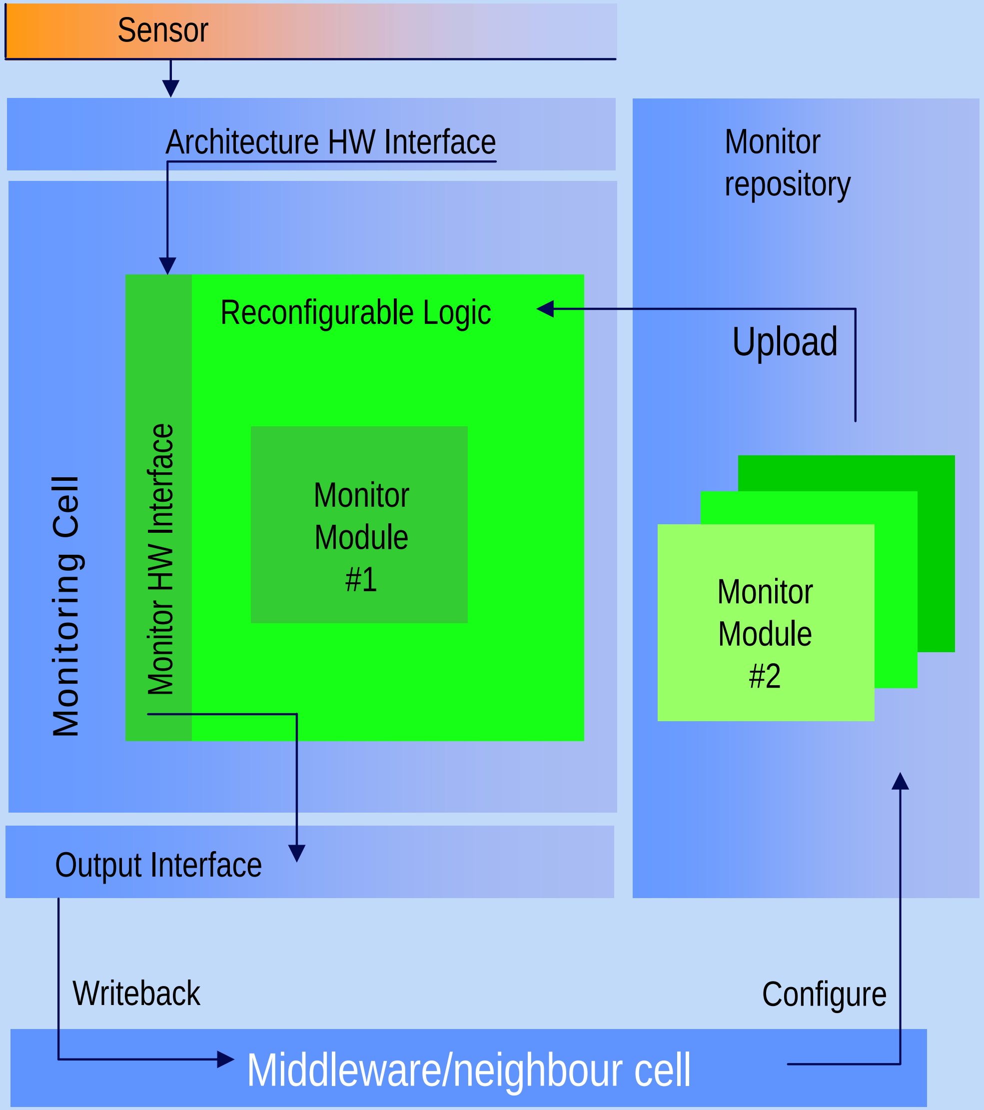
A generic and standardized monitoring cell capable of holding an adaptable, reconfigurable monitoring device

Domain-specific Monitor Analysis Modules

- To be loaded into the cells
- Extract, process, and store system status at its source
- Generate all information needed by the middleware to conclude a more reasonable configuration

Correlation of monitoring information from several sources to assemble global system state

A standardized API to query monitoring information across monitoring cells



Sub-Project “Monitoring”
(Prof. Karl)

DodOrg

Monitoring Challenges

- Reconfigurability and adaptability of the monitoring itself
- Ability to hold any signals and to process status information at all levels and from all components of the system
- Efficient processing of raw data, at the data source
- Presentation of a single API
  - Allowing the middleware to access any system performance data

Monitoring Framework

- Wide and uniform base for access to monitoring information
- Efficient data pre-processing
- Correlation in space and time
- Problem-specific data aggregation

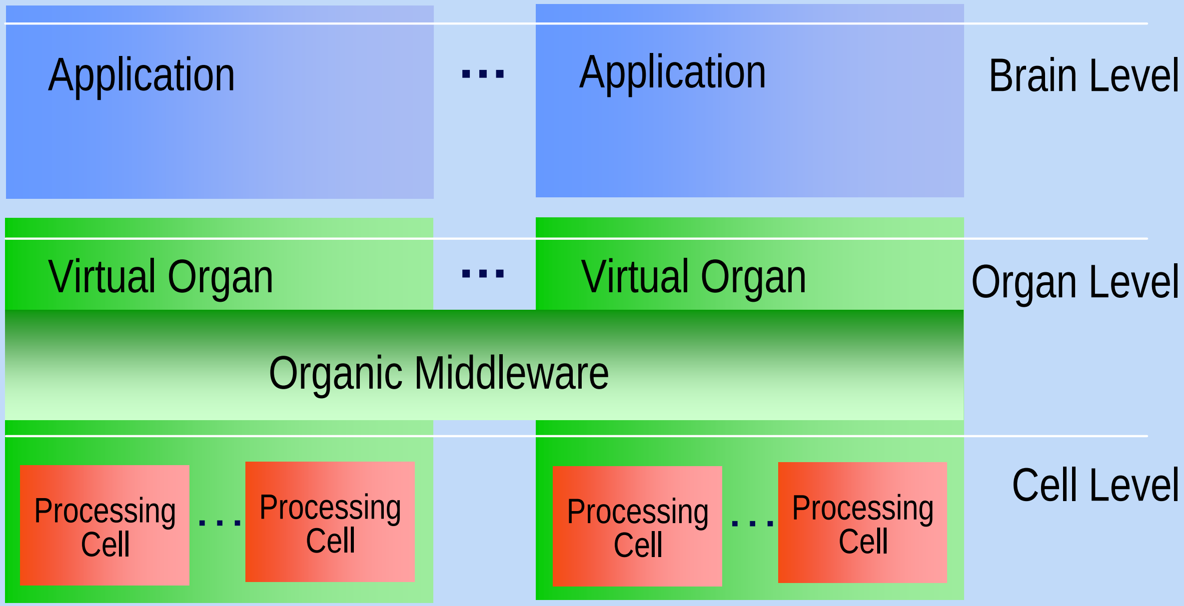
Interaction between single processing cells is coordinated by organic middleware

- ▶ The organic middleware forms virtual organs in this way.

Since there are **continuous modifications** in the system (by the environment and also by the brain level), the organic middleware must have the possibility to interfere in the system at each point of time (configuration, optimization).

▶ The organic middleware implements control loops, which fulfill these functionalities.

▶ There are de-central control loops to compensate the failing of single components and to ensure continued operation of the whole system.



Sub-Project "Organic Robot Control" (Prof. Wörn)

DodOrg

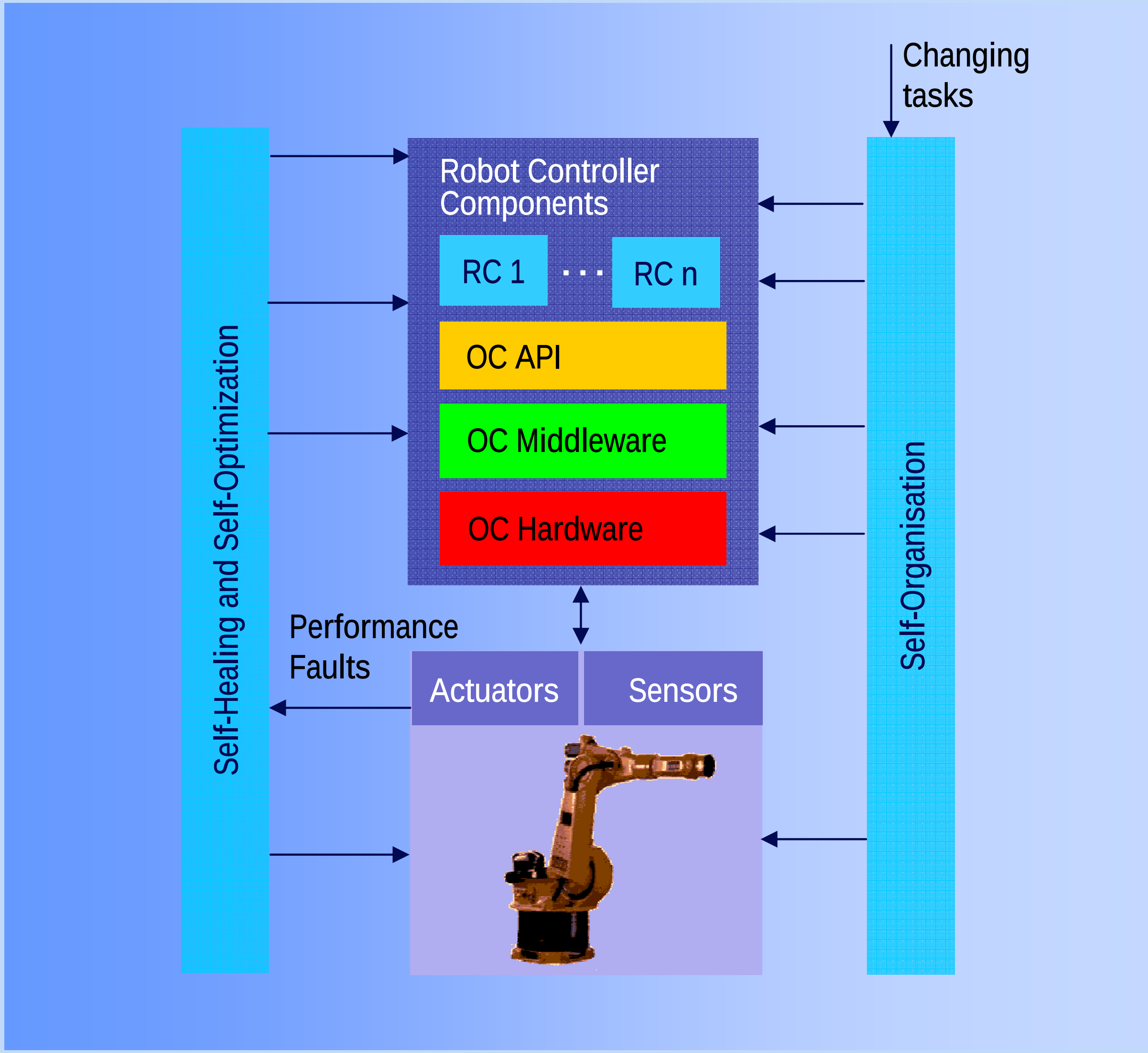
Monolithically structured robot controls can only be adapted and enhanced with high efforts

Frequent reconfiguration in factory automation necessary

- Short product life cycles, mass customization
- Flexibility when material and resource bottlenecks occur
- High availability, fault tolerance

Research approach

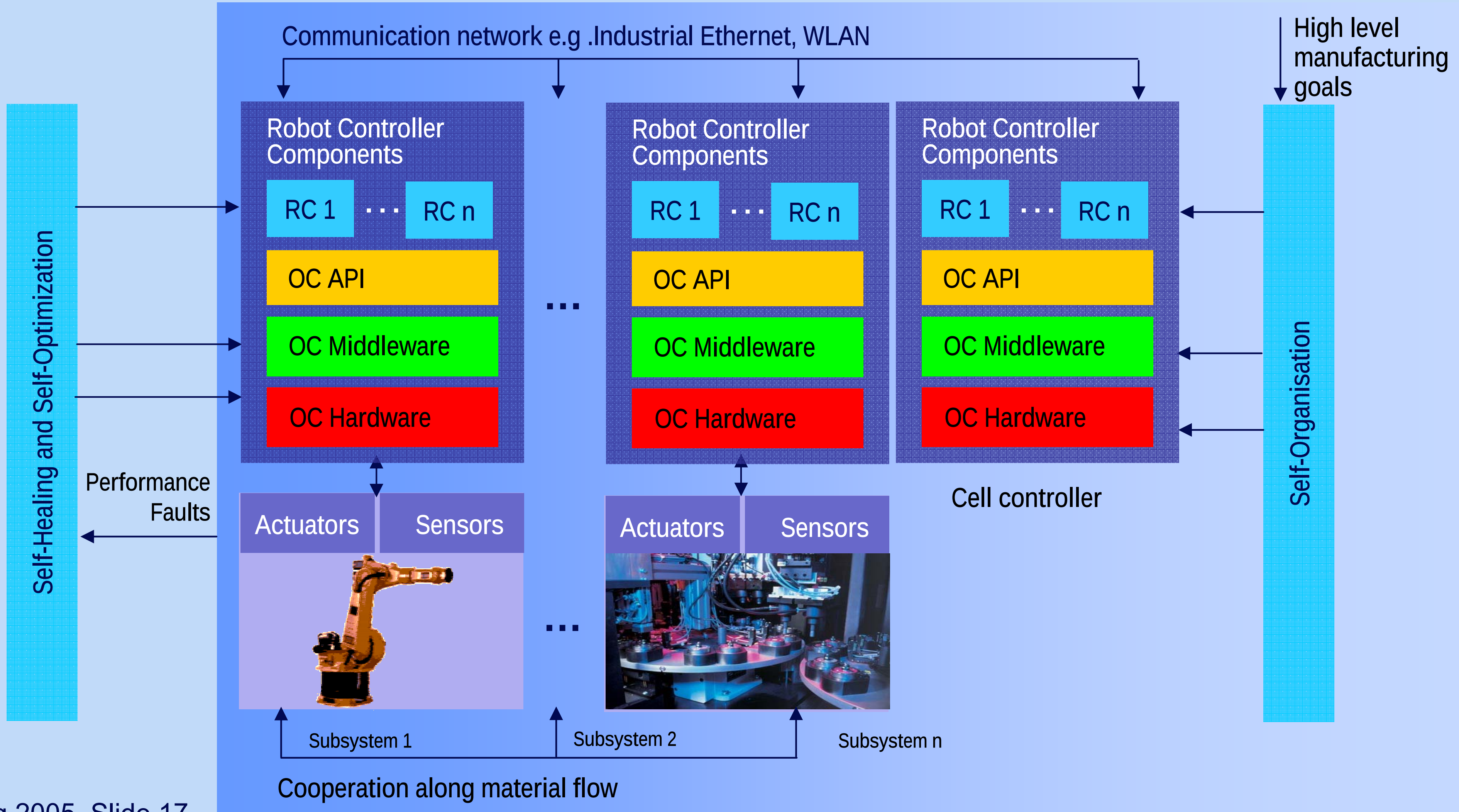
- Self-configuration and self-organization based on knowledge and on rules
- Component-based architecture
- Extensibility and interchangeability of components from different vendors (plug and work)



Sub-Project "Organic Robot Control" – "Organic Robot Swarm" (Prof. Wörn)

DodOrg

Development of swarm strategies and models for cooperating robots
(will be done in a later phase of the project)



Conclusion and Outlook

Heart as a model to learn from

- Hierarchical approach
- Cross-hierarchy features
- Neural control

Comprehensive project comprising hardware, middleware and application

- Multiple cells with common interface allow for basic self-healing/self-(re)configuration
- Low-power swapping and scheduling techniques for optimal resource usage
- Middleware shows emergent behavior through accelerator/suppressor concept
- Monitoring ensures system stability on all levels
- Application demonstrates feasibility for large-scale real-world examples

Research challenges

- Self-x and predictable behavior
- Ease observation of emergent behavior through adaptive monitoring
- Adaptation towards varying power/performance constraints
- Common interface/protocol specification
- Decentralized control loops
- Use digital organisms to control robots/robot swarms