RRLAB
THE ROBOTICS RESEARCH LAB

# Robot Architecture

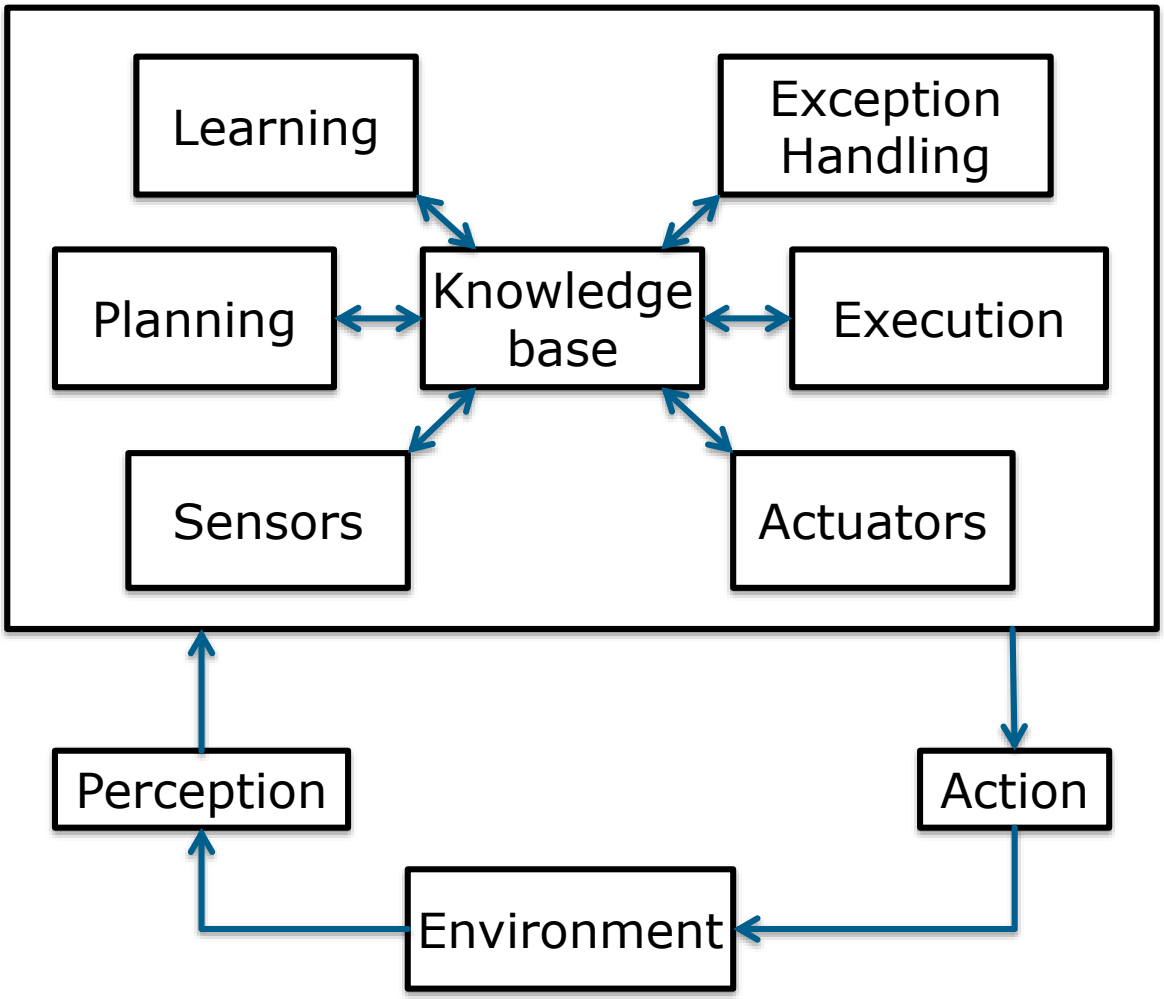

**Prof. Dr. Karsten Berns**
Robotics Research Lab
Department of Computer Science
University of Kaiserslautern, Germany

# Contents:

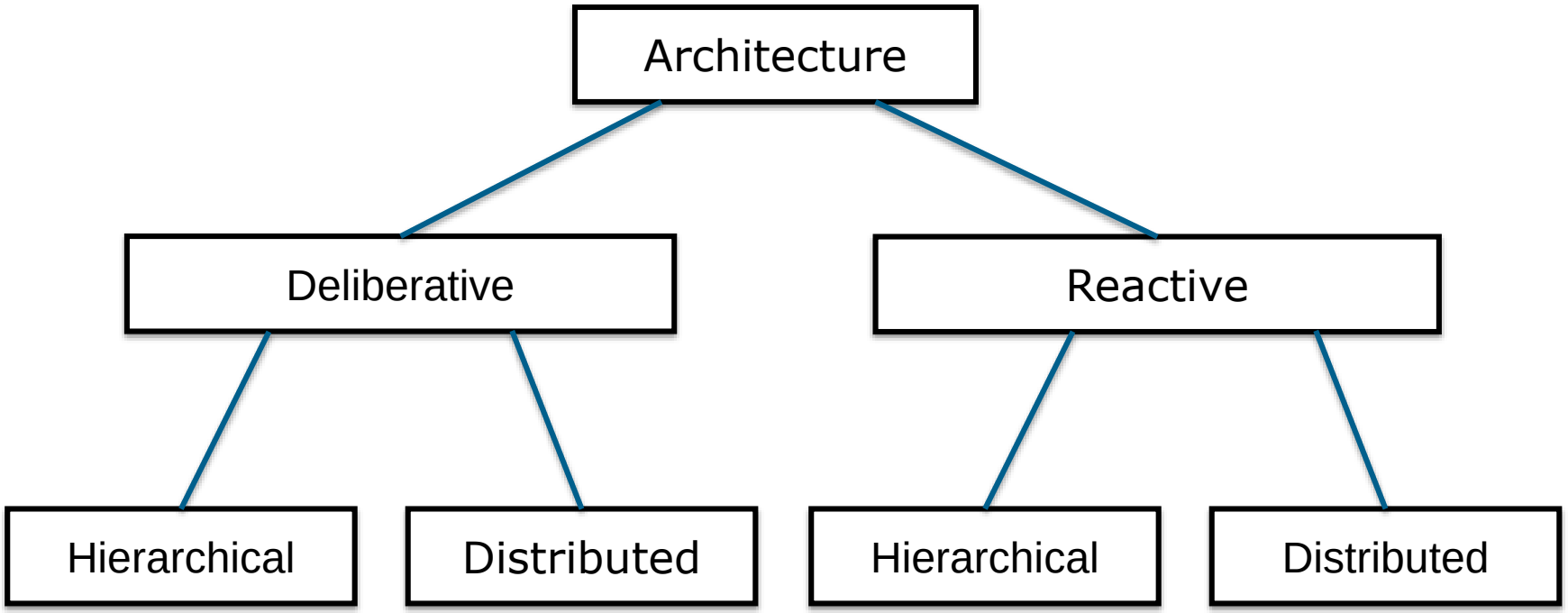
# Motivation

- General capabilities of a robotic system
- Schematic visualization of the 4 main architectures:
  - Hierarchical deliberative/function based architectures
  - Distributed deliberative/function based architectures
  - Hierarchical reactive/behavior based architectures
  - Distributed reactive/behavior based architectures

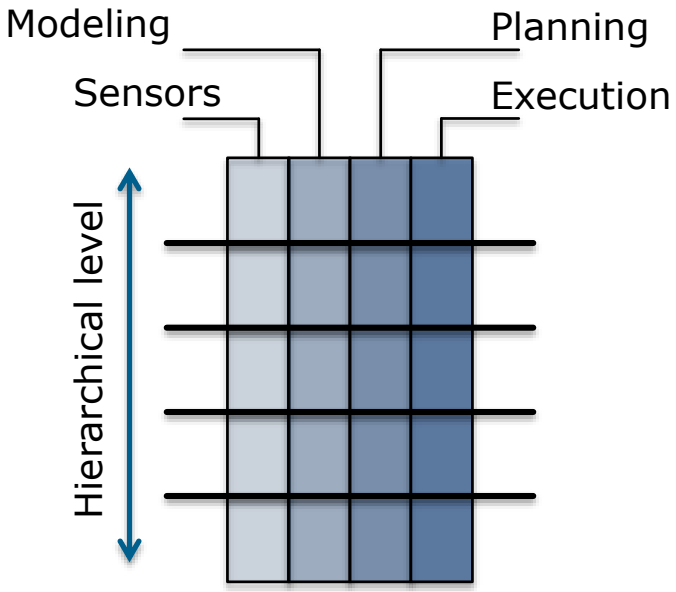
# General Capabilities of a Robotic System



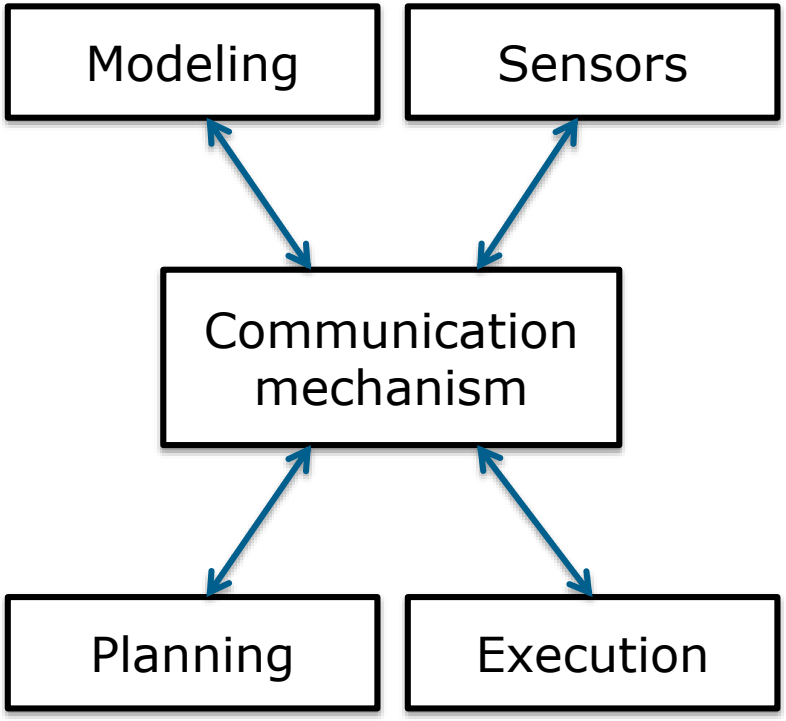
# Main Architectures: Classification



# Main Architectures: Schematic Visualization

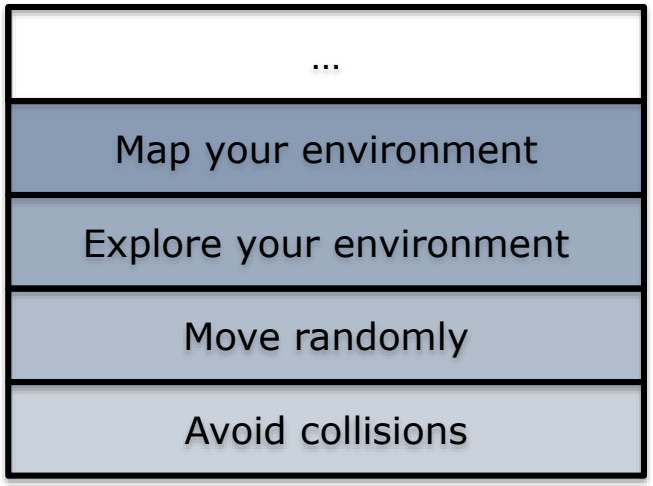


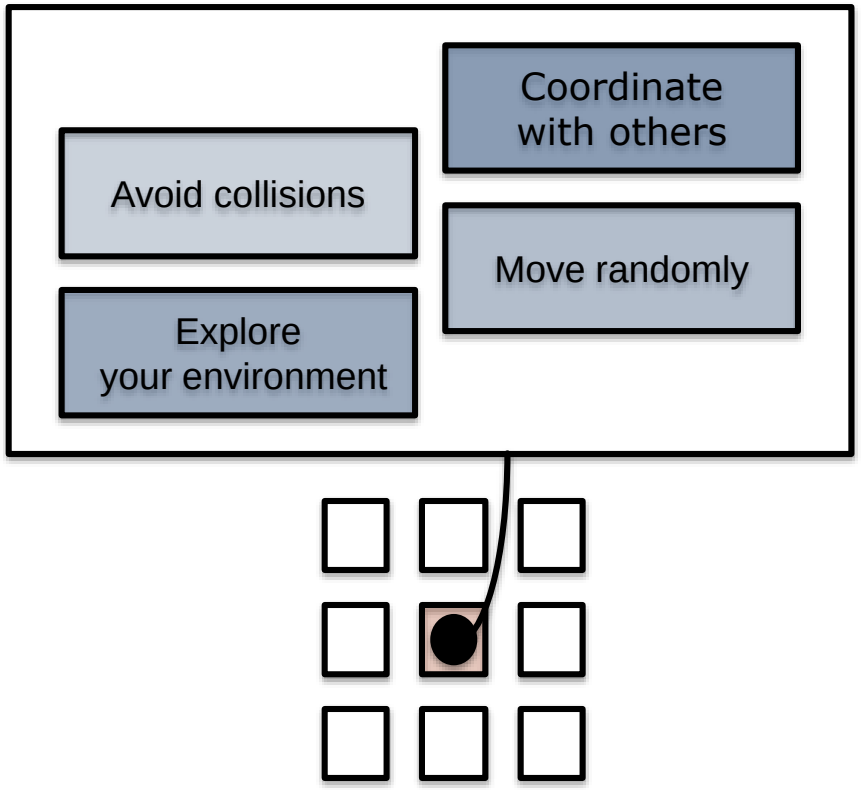

Hierarchical deliberative




Distributed deliberative

# Main Architectures: Schematic Visualization



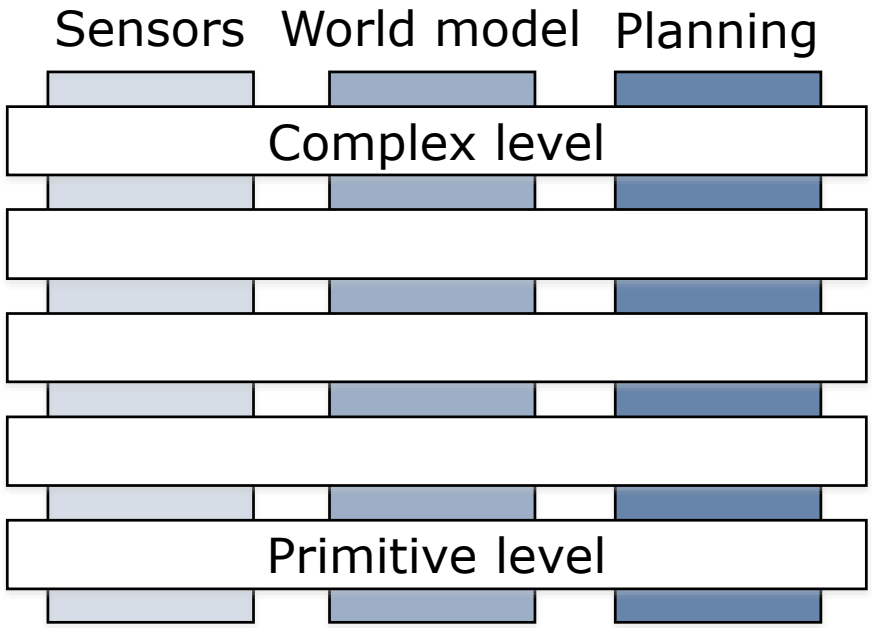




Hierarchical reactive

Distributed reactive

# Hierarchical Deliberative Architectures

- RCS: Real-time Control System
- Reference-Arch. for development of intelligent control system
- Combination of planning and reactive control structures
- Hierarchical ordered set of nodes
- Example: NASREM-Model
  (Albus, J. S.; McCain, H. G.; Lumini, R; "NASA / NBS Standard Reference Model for Telerobot Control System Architecture" NBS Technical Note 1235, 1987)



# Hierarchical Deliberative Architectures

- Divide in 4 to 6 levels
- 3 Modules per level
  - Sensor processing module $G_i$
  - World model and reference data module $M_i$
  - Task division, planning and execution module $H_i$
- Modules are ordered hierarchical by levels

# Modules

- Planning module
  - Plans and supervises the execution of actions
  - Considers data from world model and reference data module
- Sensor processing module
  - Processes and filters sensor data
  - Compares observations with internal world model
  - Determines deviations
  - Detects events, objects and situations

# Modules

- World model module
  - Refreshes reference data with sensor data
  - Predicts sensor results
  - Simulation of results in hypothetical plans
  - Information on the world (variables, objects, rules, maps)
- User interface
  - Interaction with processes or knowledge base

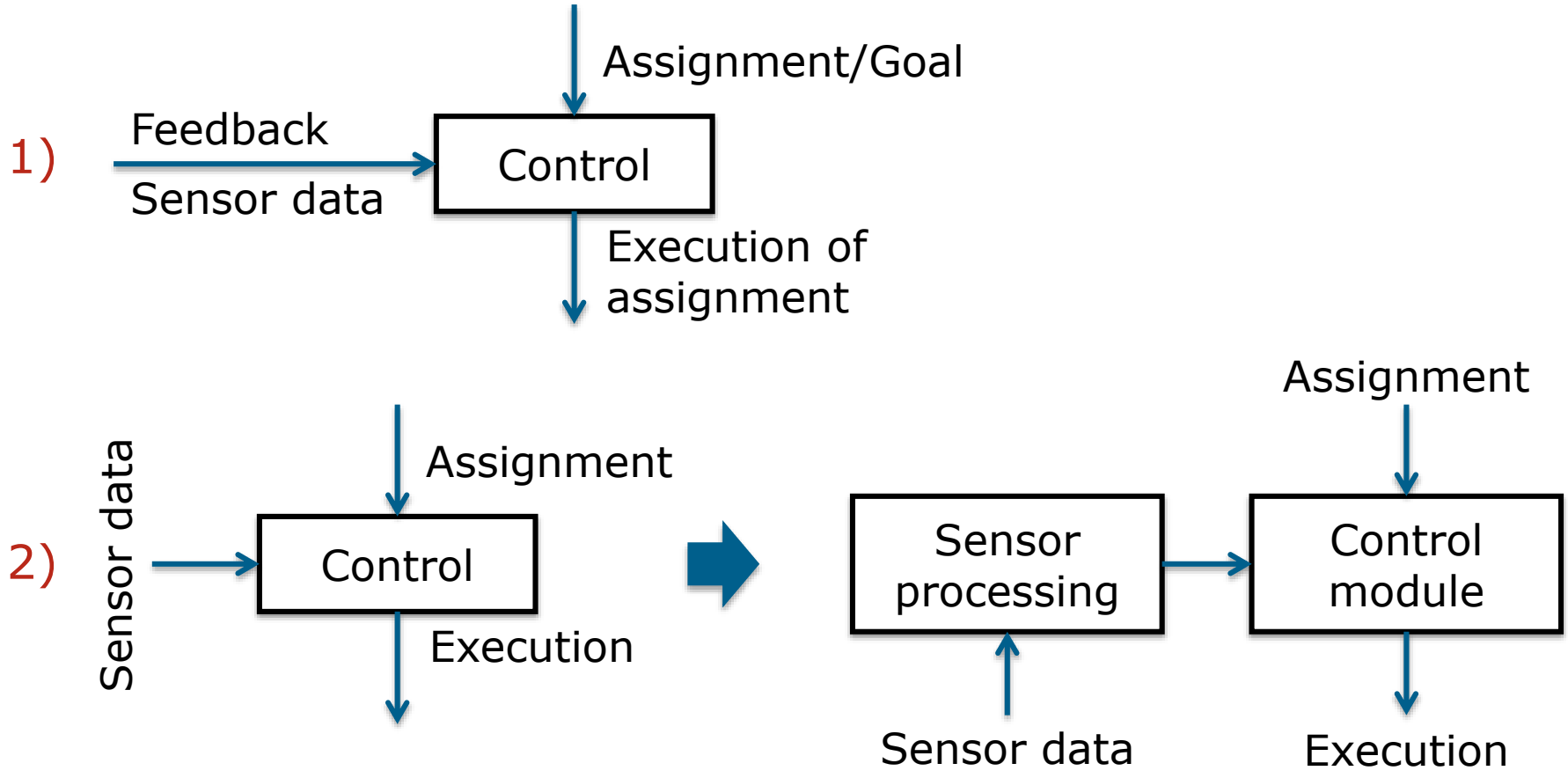
# Modules: Communication between Modules

- $H$-Modules: Commands to $H$-modules of lower levels
- $G$-Modules: Status/Sensor information to higher levels
- In between levels: Between $G$- and $M$- as well as $M$- and $H$-modules
- No special communication mechanisms
- Example
  - Point-to-point-connection
  - Network connection
  - Shared memory

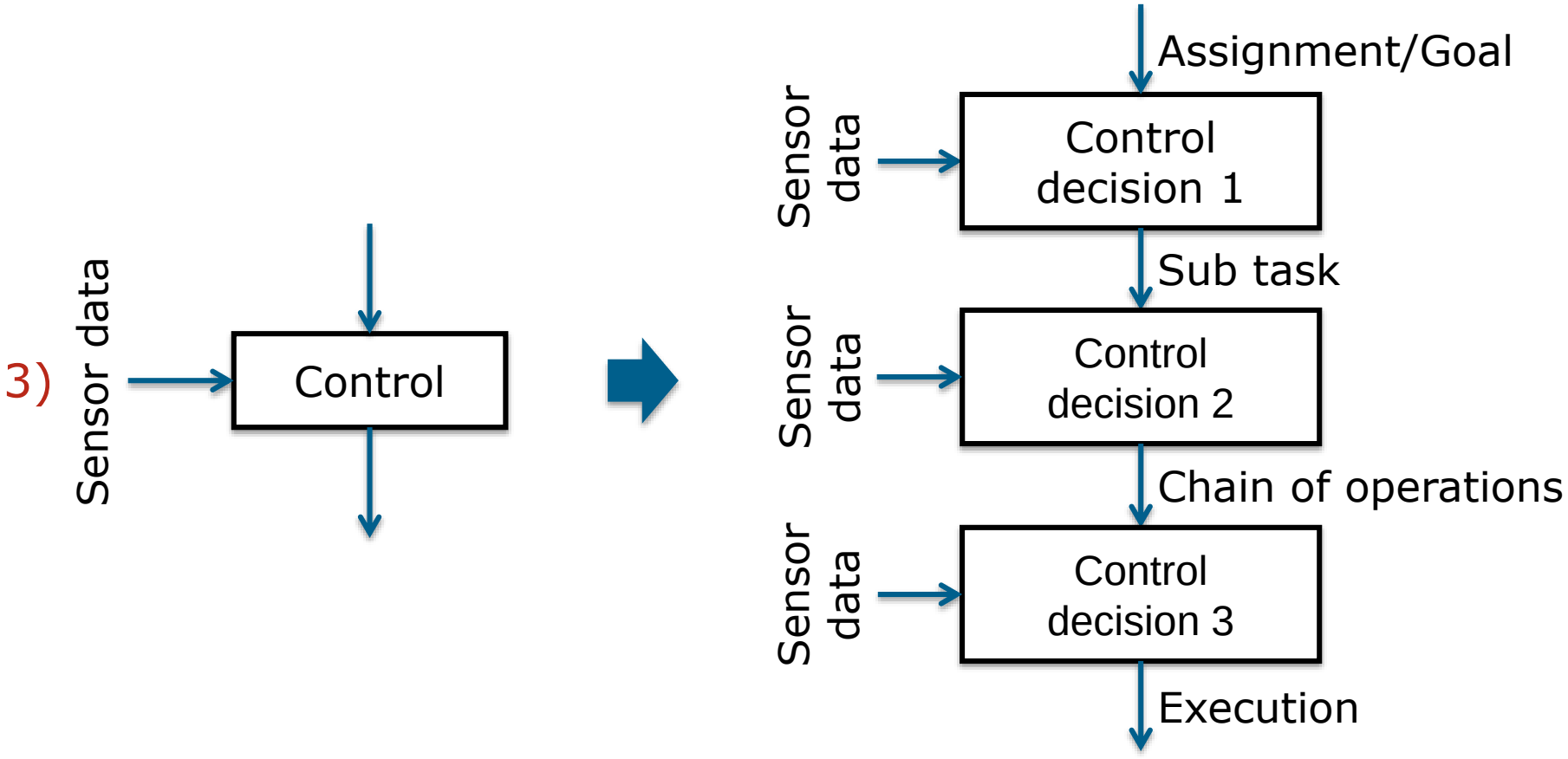
# Nodes in RCS

- Implementation of functions inside of nodes (e.g. as extended finite automaton)
- Defined initial state
- Input from at least one input buffer
- State change based on read information
- State change initiates action
- Storing of output in output buffer
- Communication system distributes information from output buffer in corresponding input buffer
- A-/synchronous control of finite automaton
- Execution frequency depends on level

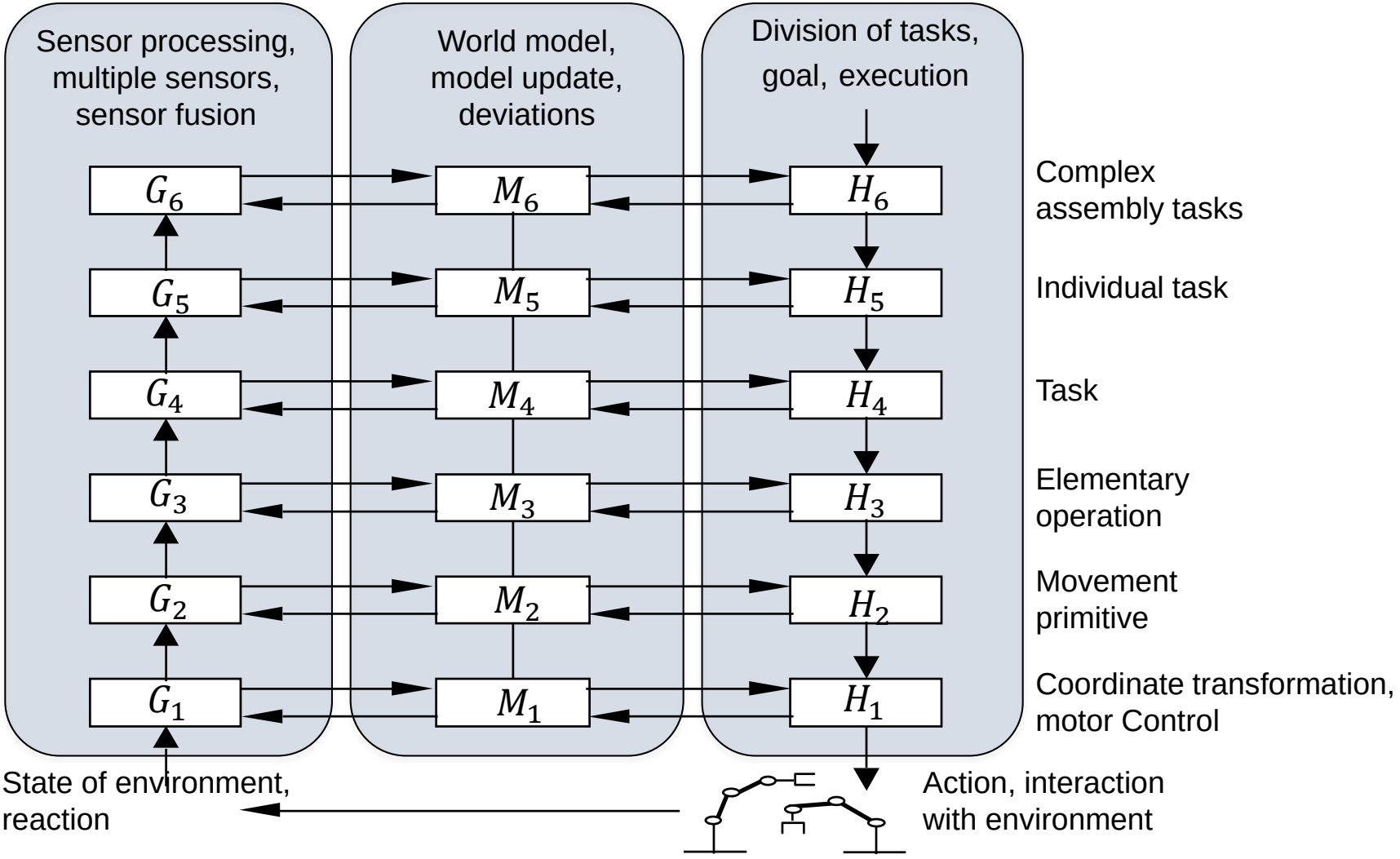
# Hierarchical Structured Sub-Functions



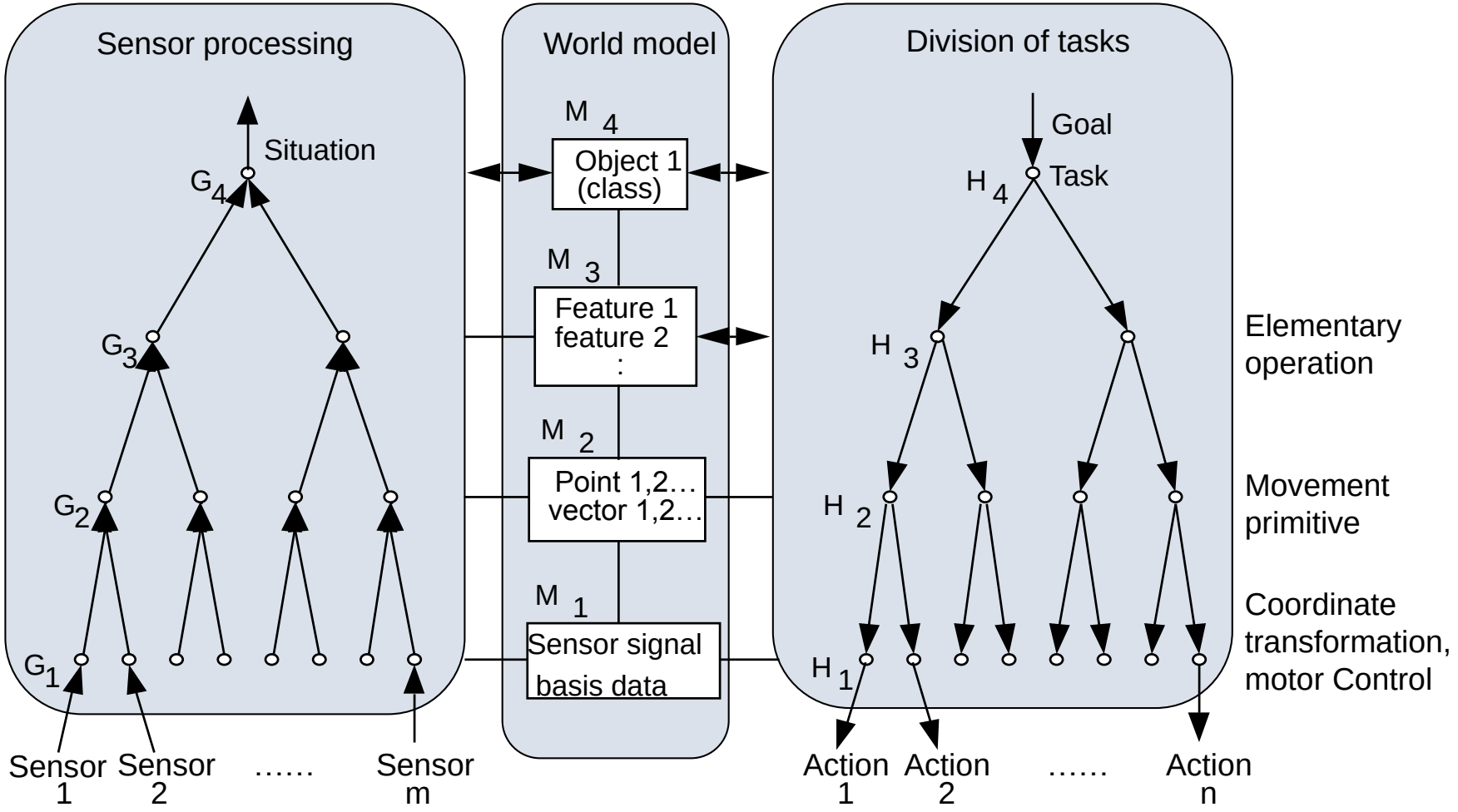
# Hierarchical Structured Sub-Functions



# Modules of a Robot Architecture

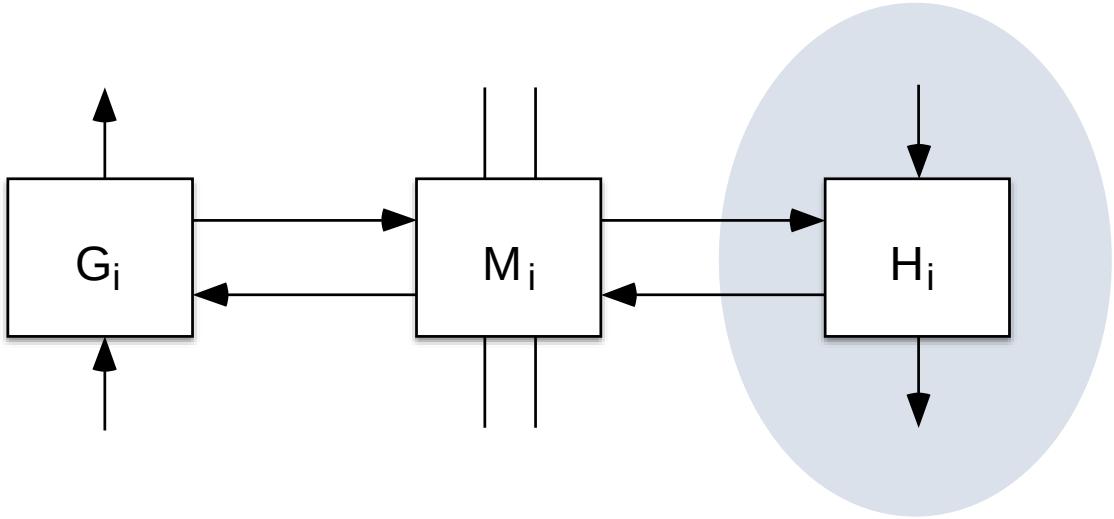


# Sensor Processing, Planning and Division of Tasks

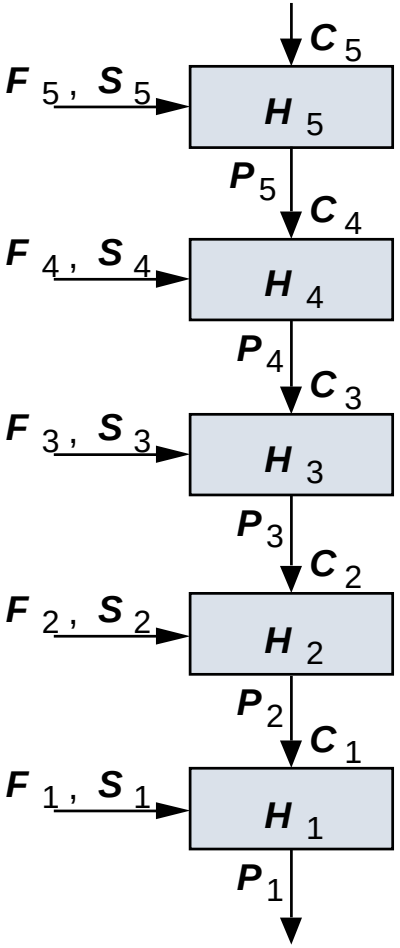


# $H$-Module

- Contains tasks from upper levels
- Divides tasks in subtasks with data from $M$-module
- Updates $M$-module
- Subtasks are forwarded to lower levels

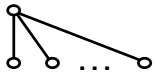


# $H$-Module: Sensor Processing, Planning and Task Division



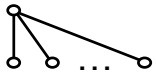

Complex task

**C** 5 : Connect part A with part B



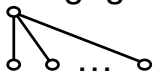
Task level

**C** 4 : Grip part B



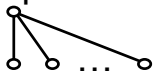
Elementary operations

**C** 3 Move end-effector to engagement position of part B



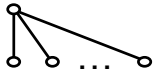
Geom. trajectory, adaptation, force control

**C** 2 : Frame part B, determine position of engagement



Joint-trajectory, servo control

**C** 1 : Joint vector for robot (TCP in frame B), ...

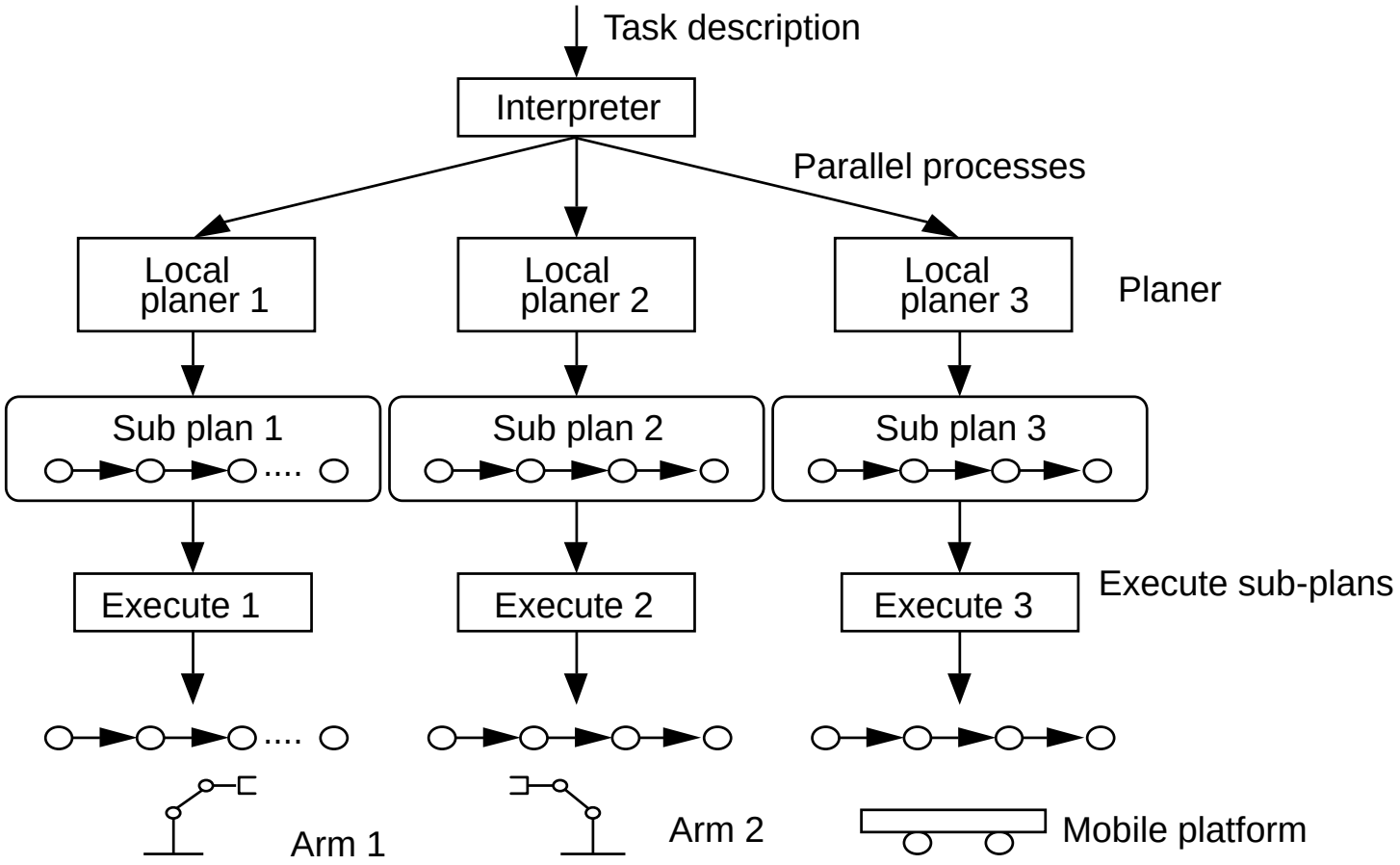


Series of moments

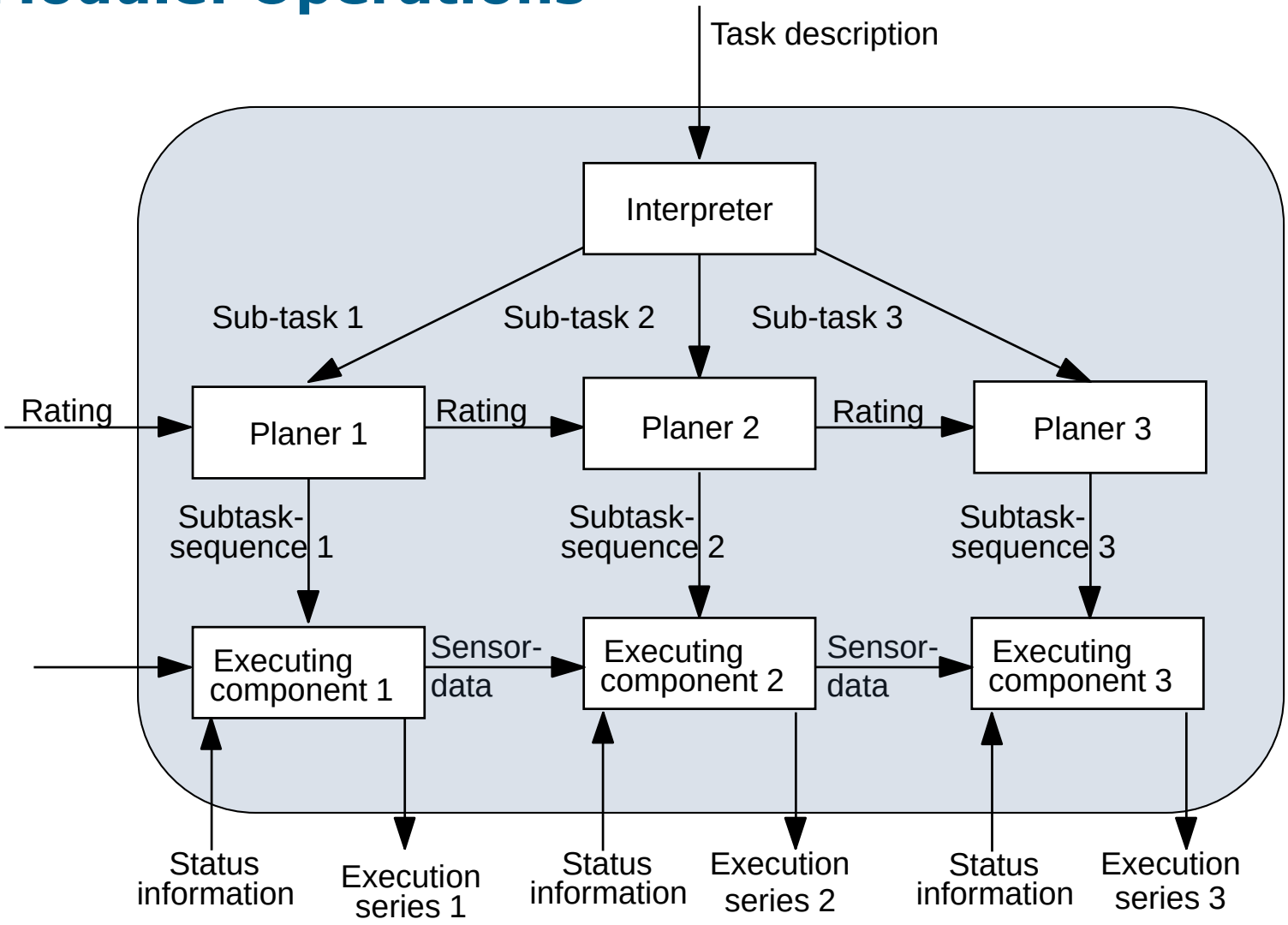
# $H$-Module: Task Division

- Output vector $P_i \sim C_{i-1}$
- Assignment vector $C_i \rightarrow$ Series of $P_{i+1}$
- Sensor vector $F_i$
- Input vector $S_i = C_i + F_i$
- Operator $H_i$
  - $P_i^k = H_i\left(s_i^{k-1}\right)$
  - $P_i(t) = H_i\left(S_i(t - \Delta t)\right)$

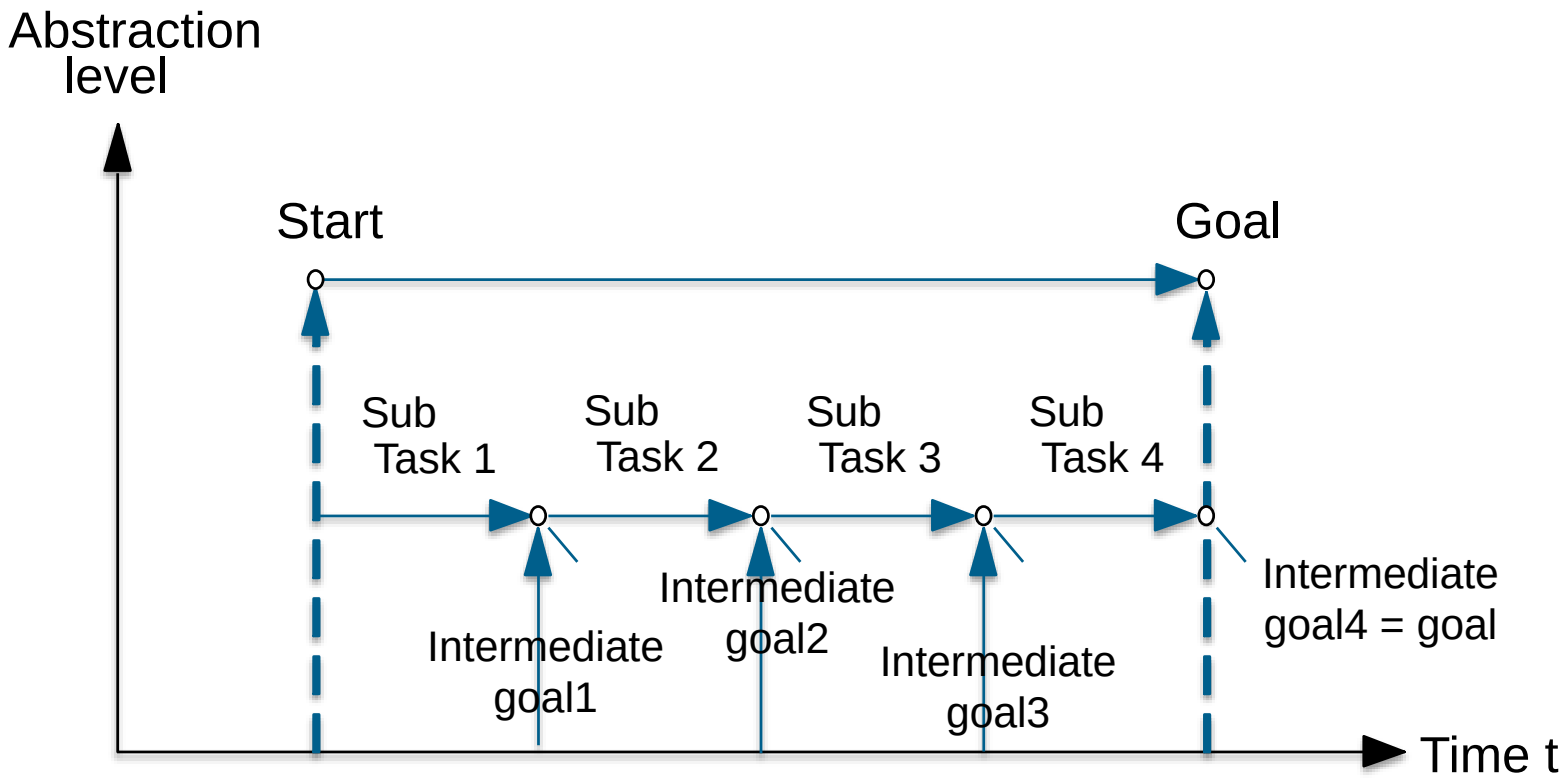
# $H$-Module: Task Division



# $H$-Module: Operations

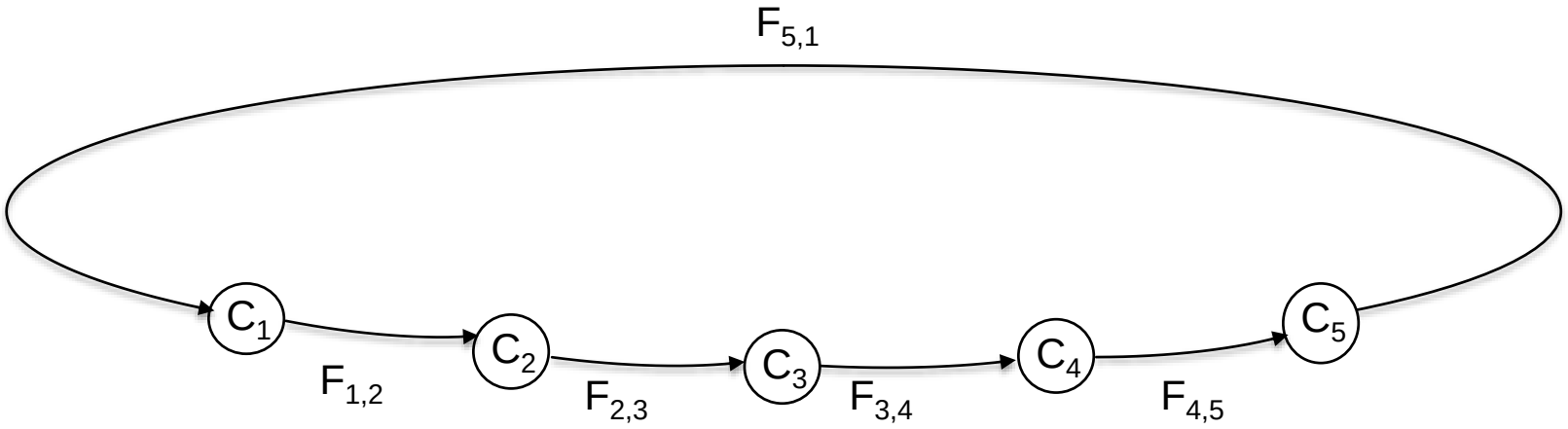


# $H$-Module: Task Division into sub-tasks

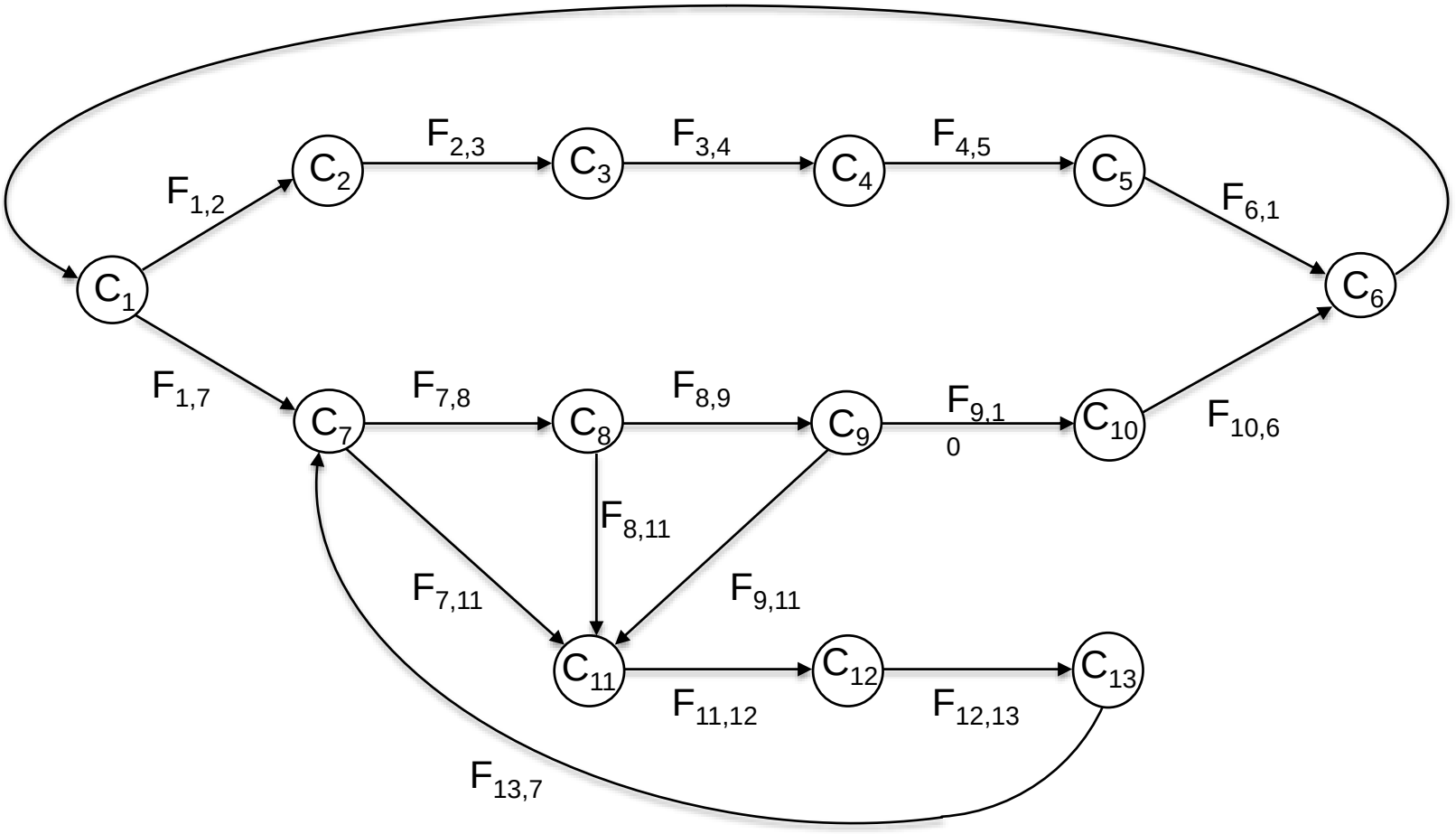


# $H$-Module: Input Series as Linear Graph

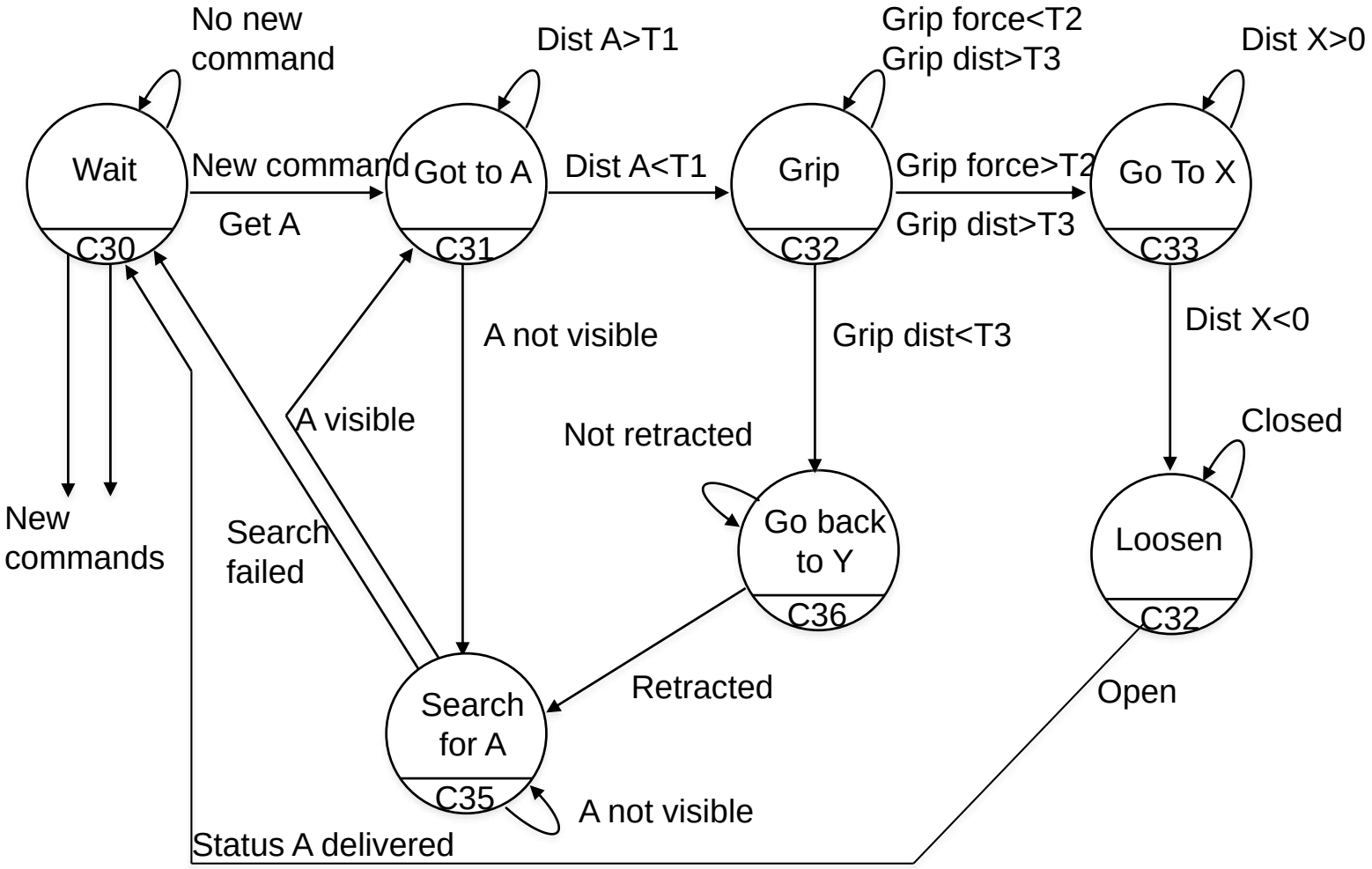
- $C_i$: Command $i$
- $F_{i,j}$ : Transition from $i$ to $j$



# $H$-Module: Input Series as Tree

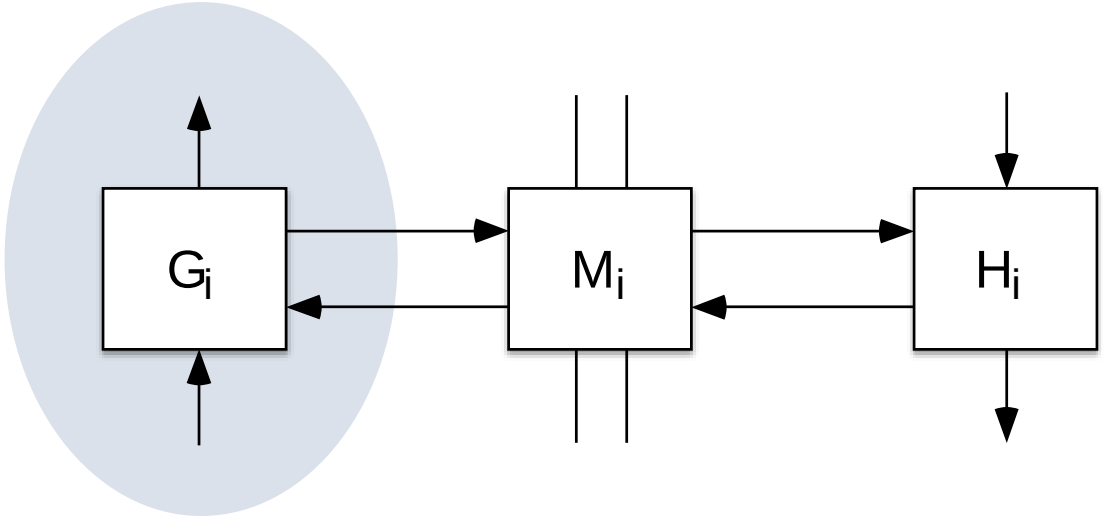


# $H$-Module: Example Assembly Sequence



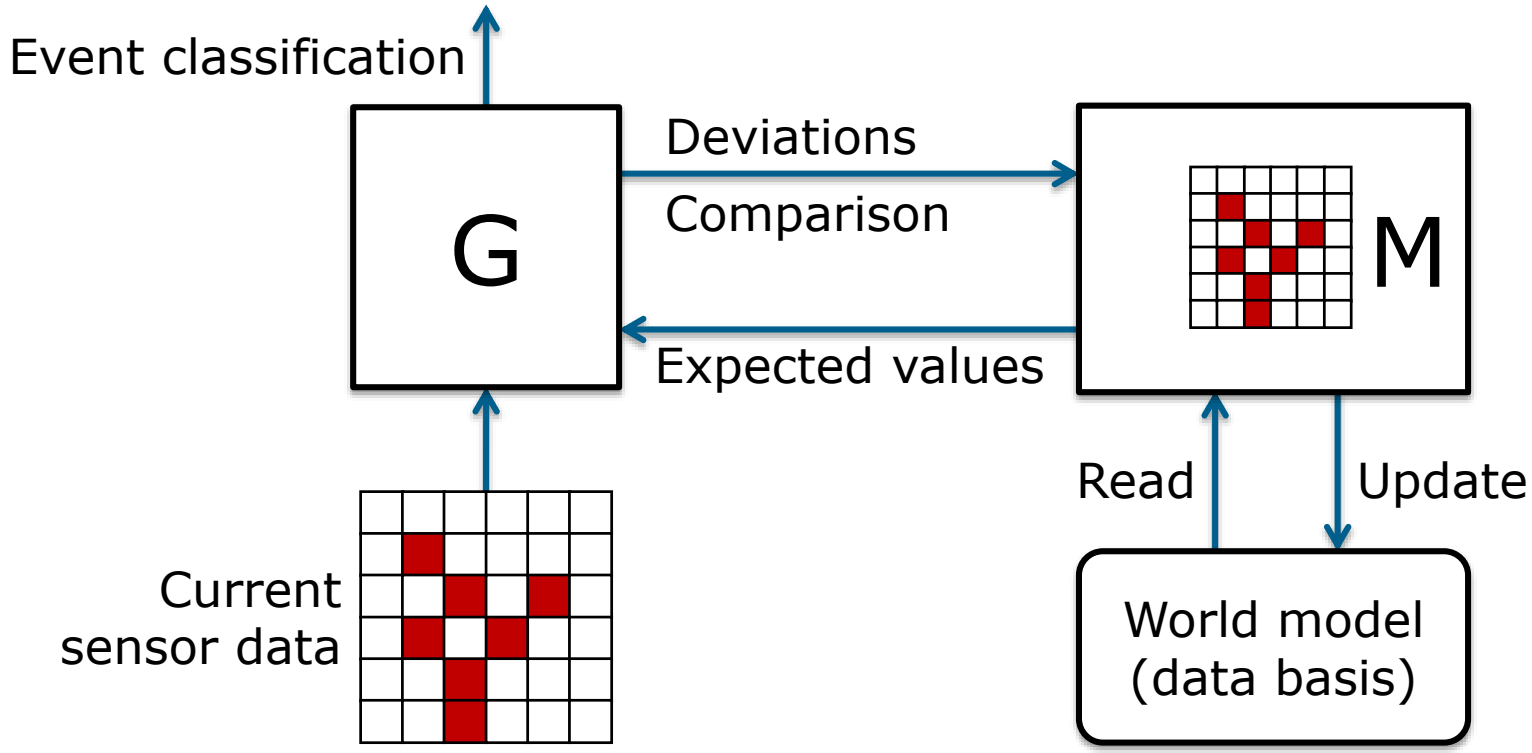
# $G$-Module

- Contains status/sensor information from lower levels
- Process Data in $M$-Module
- Update $M$-module
- Send information to next higher level

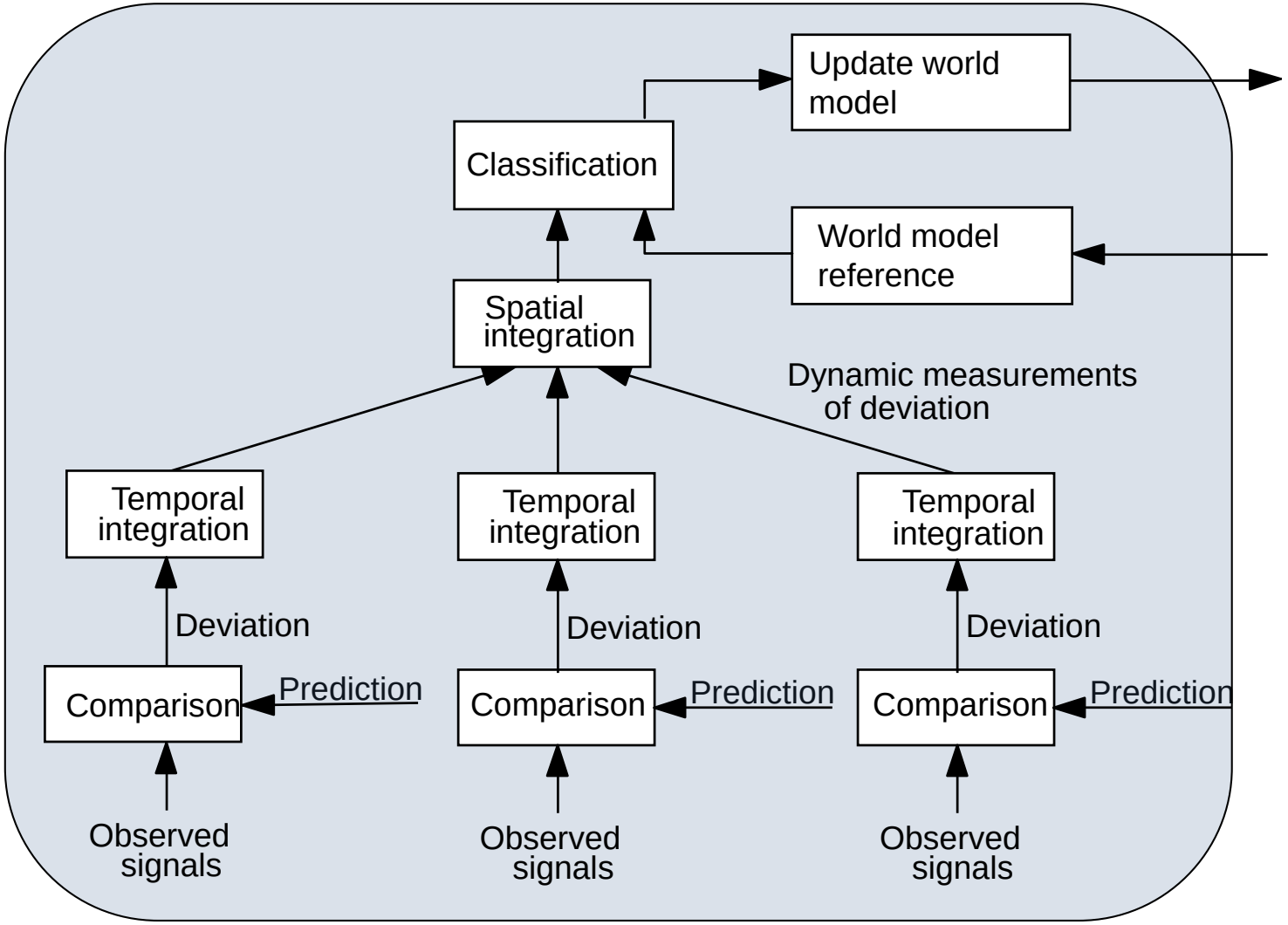


# $G$-Module: Compare Model With Measurement

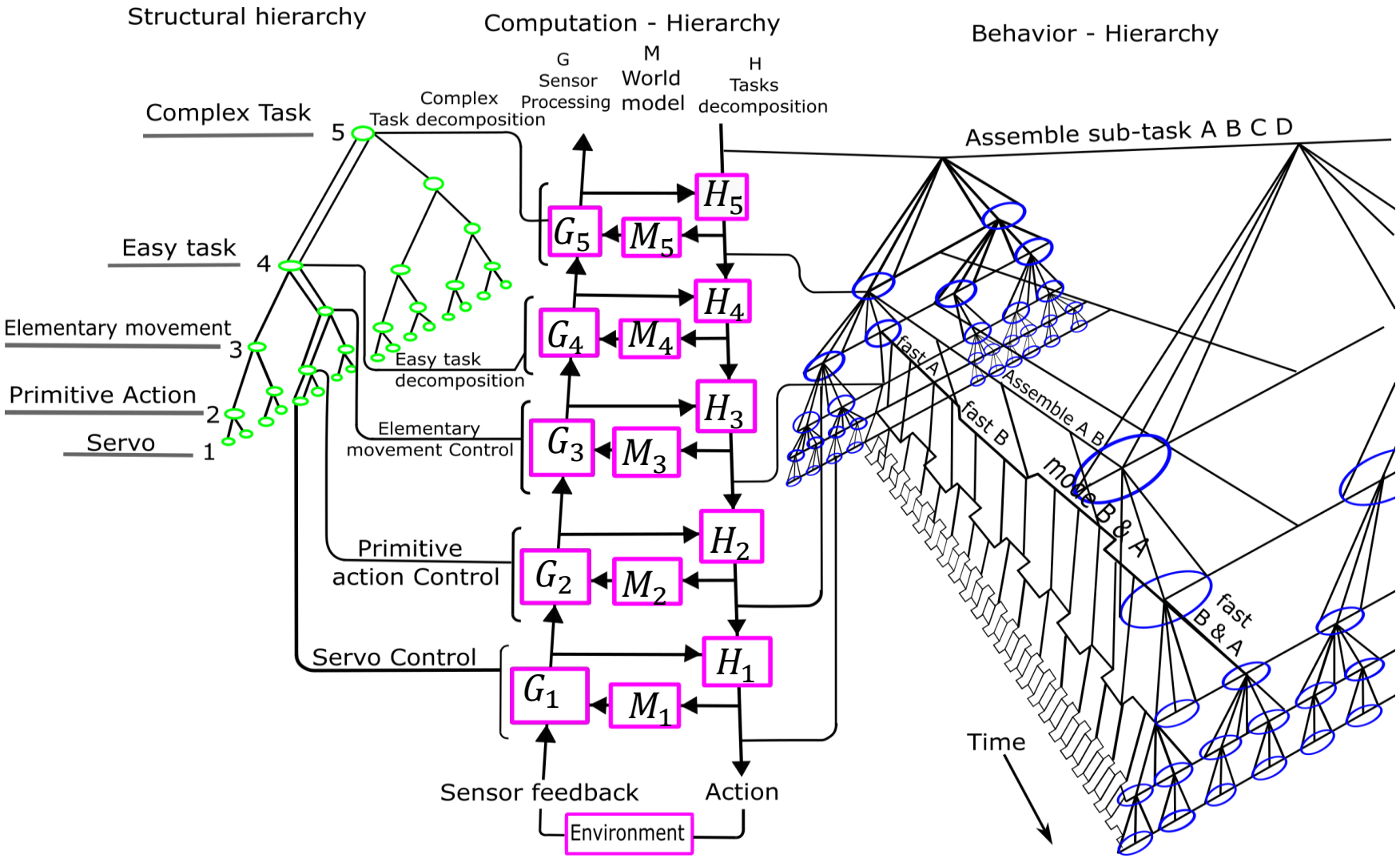
Compare predicted model states with measured ones



# G-Module: Signal Processing

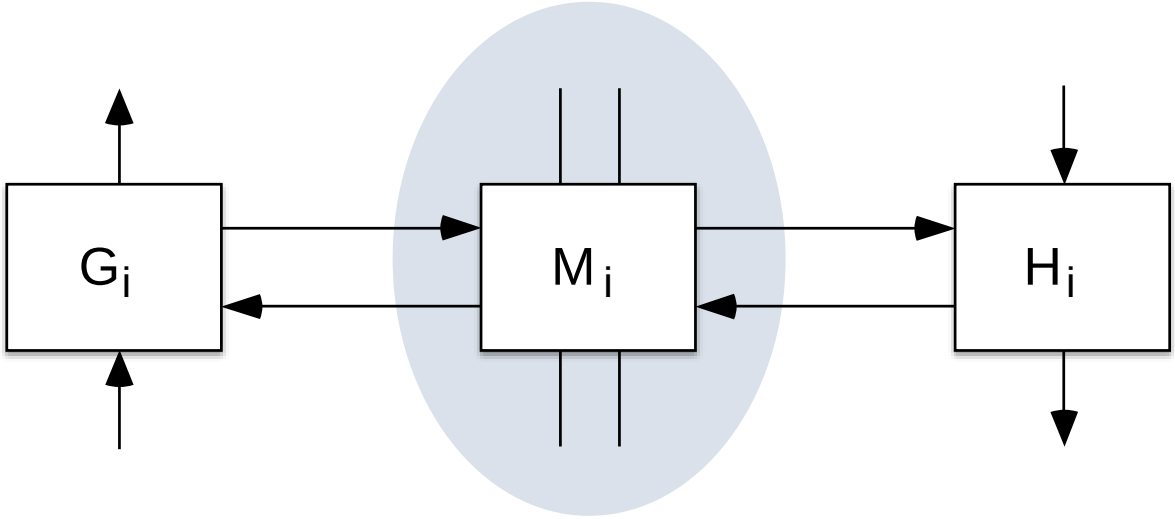


# $G$-Module: Application

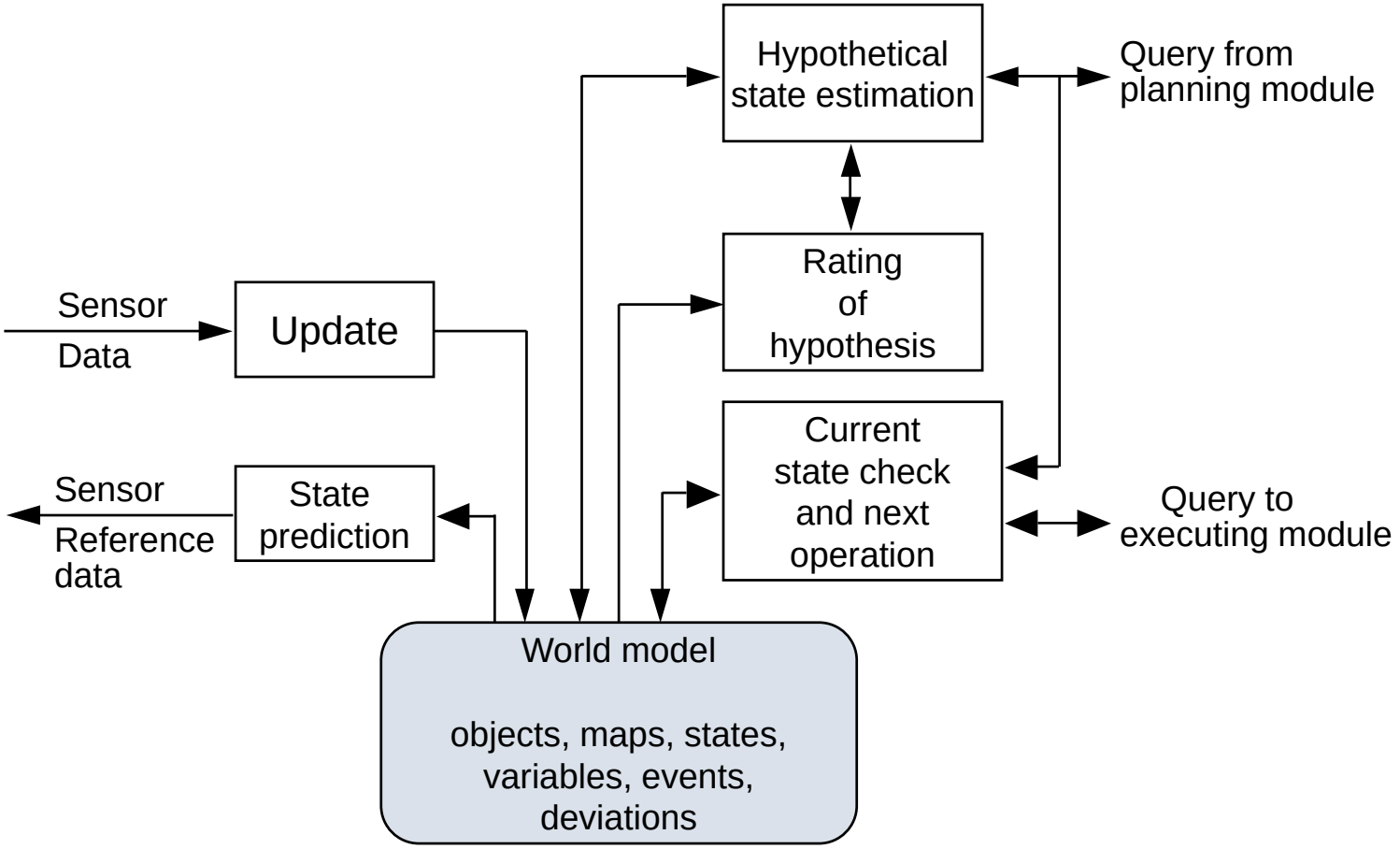


# $M$-Module

- Contains data with corresponding abstraction level
- $G$- and $H$-Modules extract this data
- Error Detection by comparing predicted ($M$-module) and actual data ($G$-module)

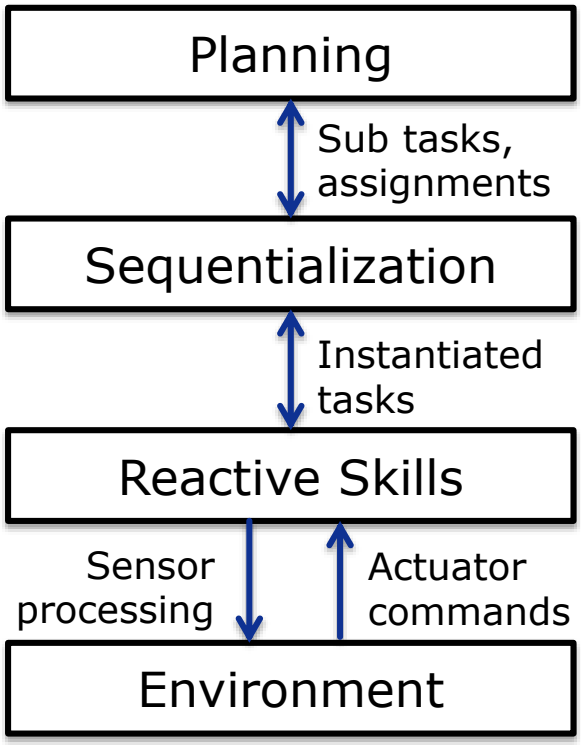


# $M$-Module: Operations on the World Model



# Example: T3 – Intelligent Control Architecture

- Developed by NASA and
  Metrica Inc. Robotics and Automation Group
- Three integrating levels
  - Dynamic reprogrammable set of reactive skills
  - Coordination via skill-manager
  - Sequential control for activation and deactivation of skills to solve specific tasks
  - Reactive action packages (RAPs)
  - Planning components to determine goals, resources and timings (adversarial Planner (AP))

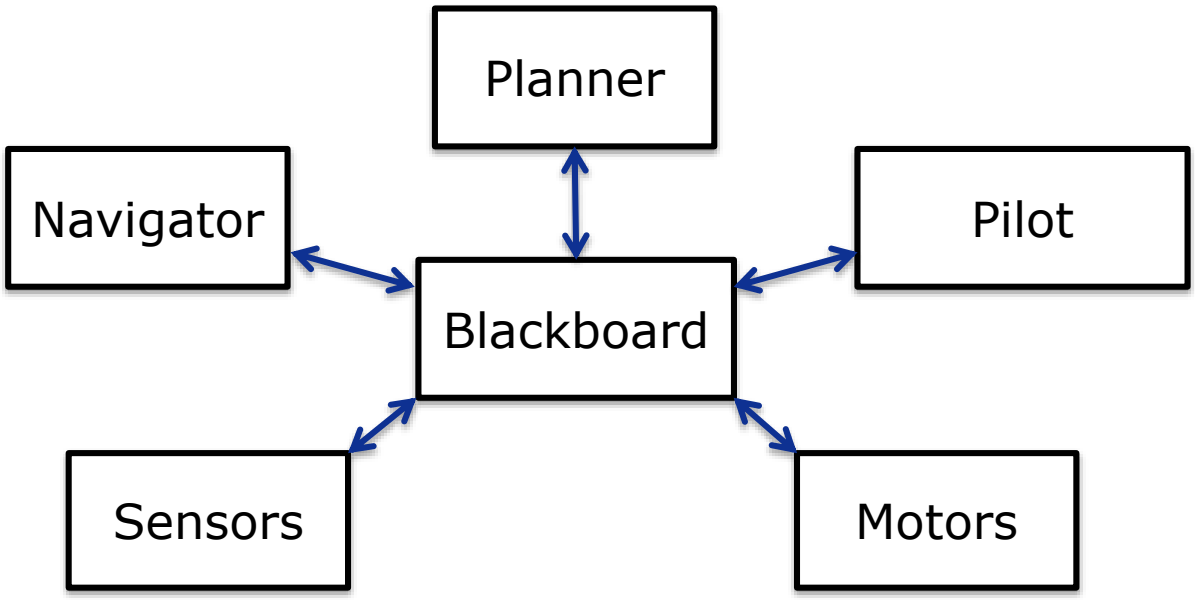


# Example: T3 – Sequential Control

- Task division (series of actions or RAPs)
- Further division and execution of RAPs on the sequential level by the RAP-interpreter
- Activation of even monitors to notify the sequential control of certain events
- Activated skills effect the events and the environment
- Sequential level terminates actions, if …
  - … monitored events occur
  - … certain time limit is exhausted
  - … changes in the plan are send by the deliberation level

# Distributed Deliberative Architectures

- Set of specialized subsystems
- Communication of central component
- Example: Nav-Lab-System



# Hierarchical reactive Architectures

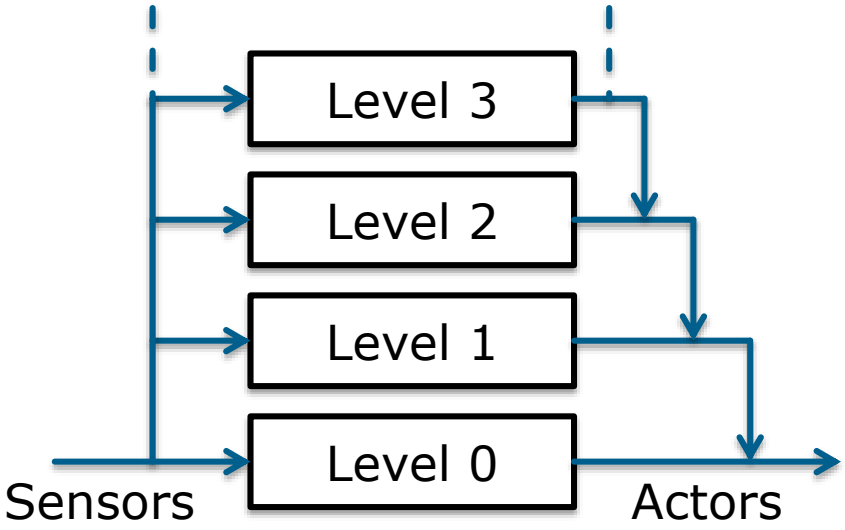
- Control via behaviors/reflexes
- Pattern: System reacts non sensor stimulus
- Arrangement in behavior levels
- Hierarchical structure on competence level
- Example: Subsumption architecture,
  behavior Scheme as finite state machine

# HV-Arch.: Requirements on Control System

- Multiple Goals: Robot can pursue multiple, potentially contradicting, goals
- Multiple Sensors: Fusion of multiple sensors (problem: Inconsistency)
- Robustness: Robust with respect to noise and sensor failure
- Additivity: Possibility to add new sensors or computing power

# Subsumption Architecture

- Dividing architecture in different levels of competence
- On each level the behavior system models a complete control system
- Behaviors of higher level can overwrite output of lower levels



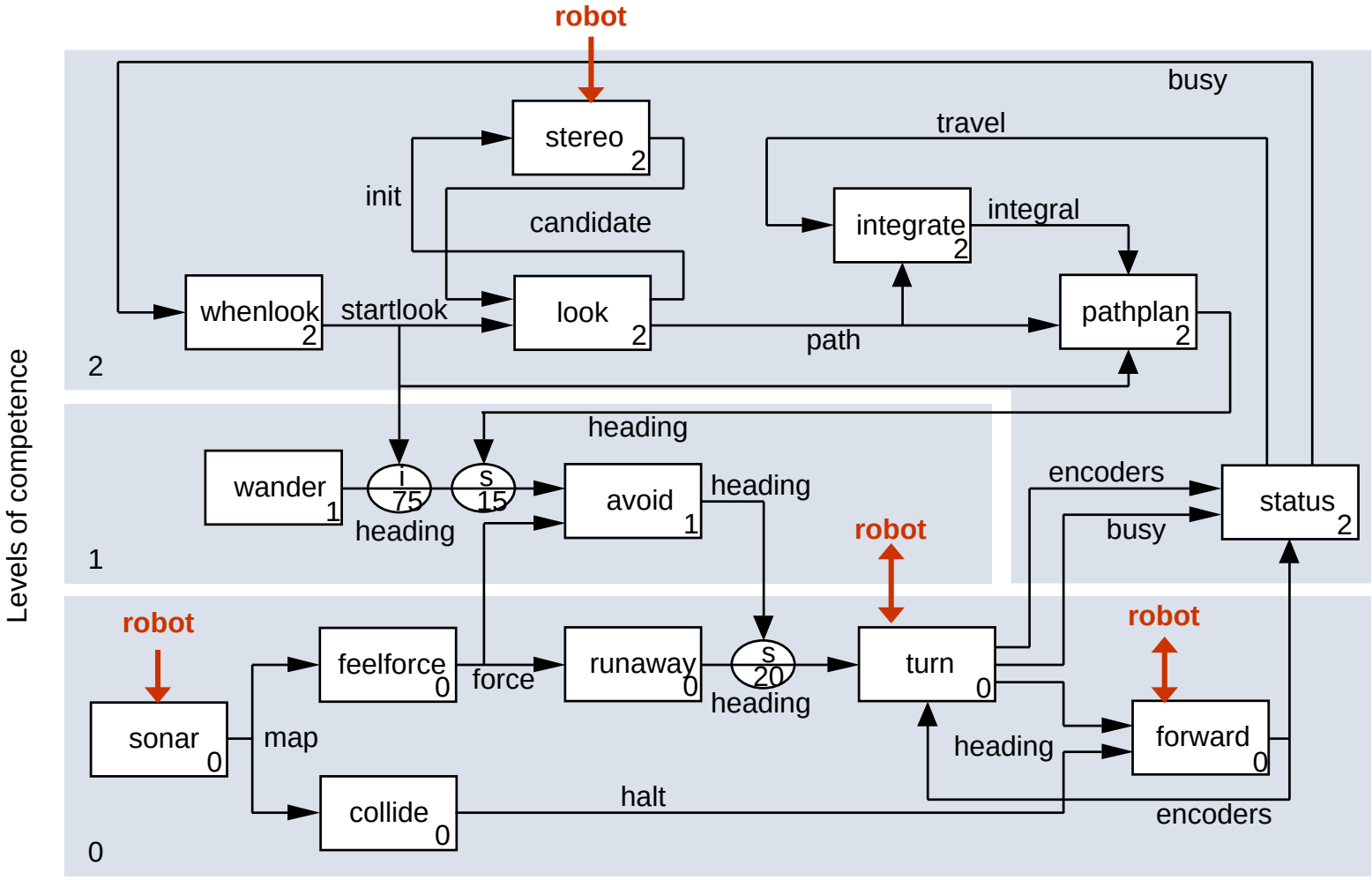

Hierarchical setup

# Subsumption Architecture: Mobot-System

Hierarchical structure of competence level of the robot Mobot

1. Collision avoidance
2. Random movement in environment
3. Exploration of environment
4. Mapping and path planning
5. Capture changes in local environment
6. Detect known objects and deduct effects on object oriented tasks
7. Create plan and execute it
8. Deduct effects on environments and resulting modifications in plans

# Subsumption Architecture: Mobot-System



# Distributed Reactive Architectures

- Independent subsystems with identical/competing behavioral patterns
- Coordination via a behavior
- Prioritizing by arbitration process (e.g. decaying activation)
- Example: Multi-agent-system

# Attributes of Multi-Agent-Systems

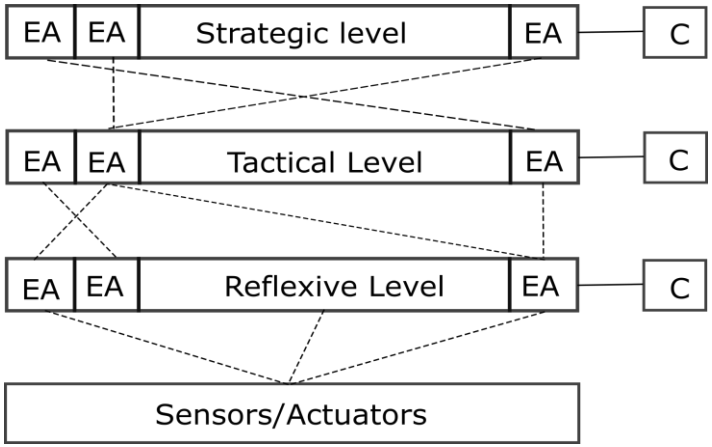
- Self organizing
- Negotiations
- Associative memorization of information
- Pattern recognition

# CoMRoS: Multi-Agent-Robot-Architecture

- CoMRoS: *CO*operative *M*obile *RO*bots *S*tuttgart
- Goals
  - Universal architecture for autonomous mobile robots
  - Predictable/Computable distributed negotiations
  - Reusability
  - Dynamic configuration
  - Robustness
  - Safety
- Scenarios
  - Driving in formation
  - Cooperative transport
  - RoboCup

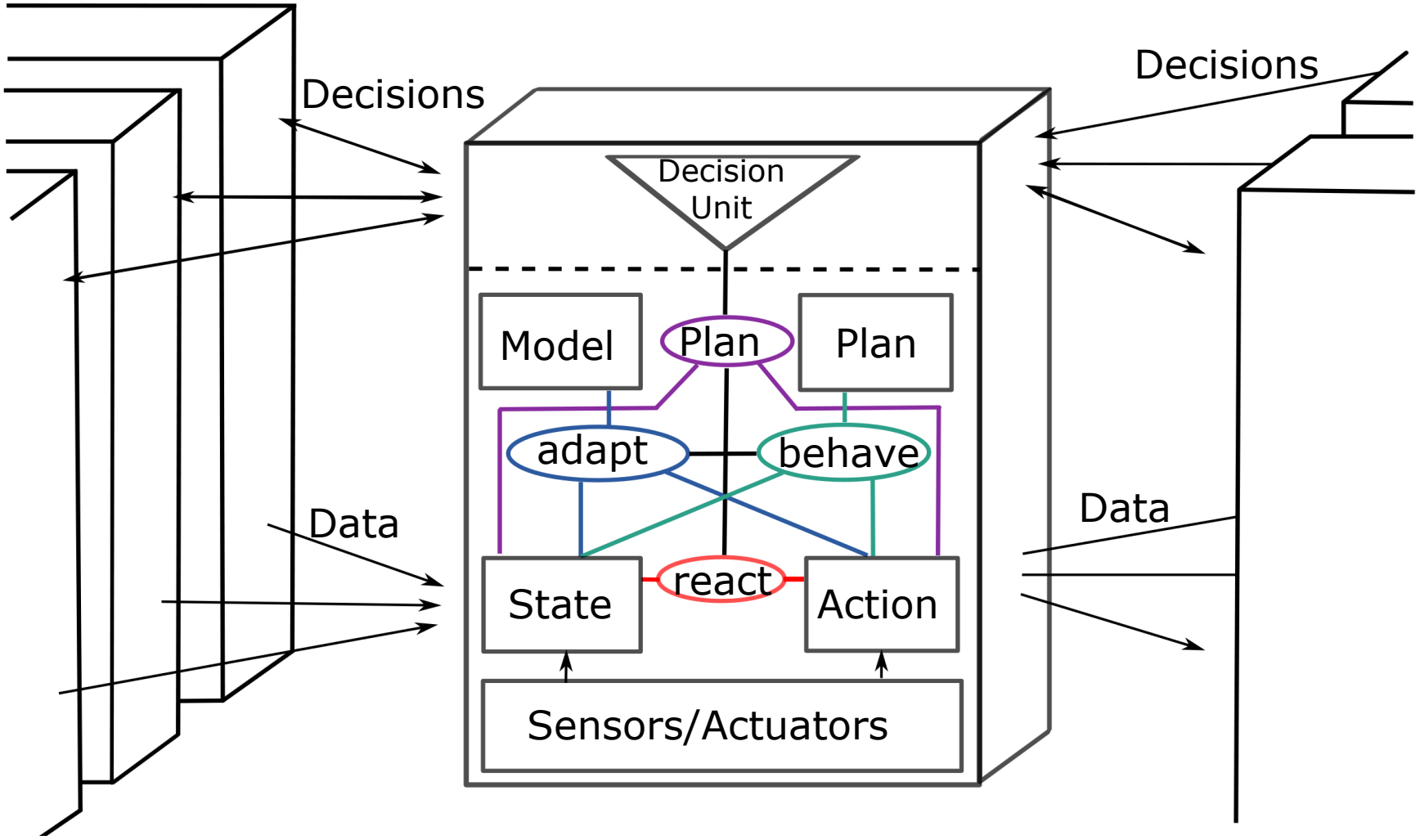
# CoMRoS: Multi-Agent-Robot-Architecture

- Division in three levels
  - Strategic Level: Complete planning of actors
  - Tactic Level: Context aware planning, control of execution, contains assignments from strategical level
  - Reflex Level: Process with mainly reactive behaviors, partially directly connected to Hardware
- Each level contains multiple software components (elementary-agents or autonomous cycles)



# CoMRoS: Multi-Agent-Robot-Architecture

- Elementary-agents (EA)
  - Autonomous
  - Dividable into brain (decision-unit) and body (sensor-actor-cycle)
- Function and structure of a EA
  - Sensor-actor-cycle
  - Connection of sensors and actors define autonomous areas of EA
- Decision unit
  - Controls abstract control-cycle
  - Leads negotiation between agents
- Interaction between decision units is based on dynamically configurable networks


Decisions
Decisions
Decision Unit
Model
Plan
Plan
adapt
behave
Data
Data
State
react
Action
Sensors/Actuators

# CoMRoS: Multi-Agent-Robot-Architecture

- Nodes of sensor-actor-cycle
  - World model
  - Plans
  - Actions
  - States
- Node connection by four cycles
  - Reactions: Feedback between sensors and actors
  - Behaviors: Defines the typical behavior of agents
  - Adaption: Updates the structure of the model
  - Planer: Creates new plans for agents

# Coming up next …

*Programming*