Organic Computing - Controlled Self-organisation

Hartmut Schmeck, Institute AIFB, Karlsruhe Institute of Technology - KIT

Joint work with Christian Müller-Schloer, Institut SRA, Leibniz Universität Hannover

Die Kooperation von Forschungszentrum Karlsruhe GmbH und Universität Karlsruhe (TH)

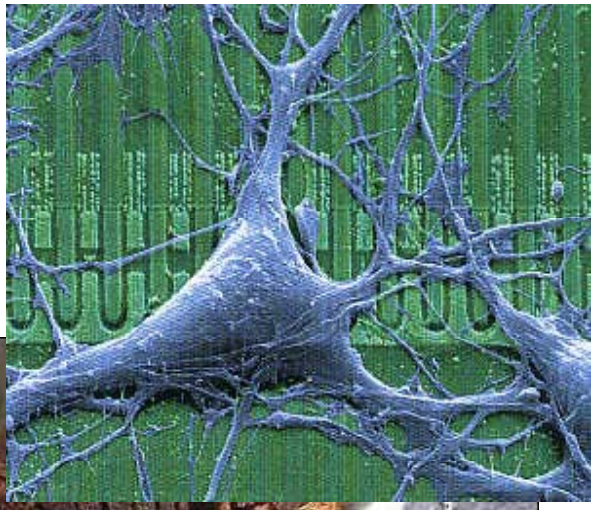
Forschungszentrum Karlsruhe in der Helmholtz-Gemeinschaft

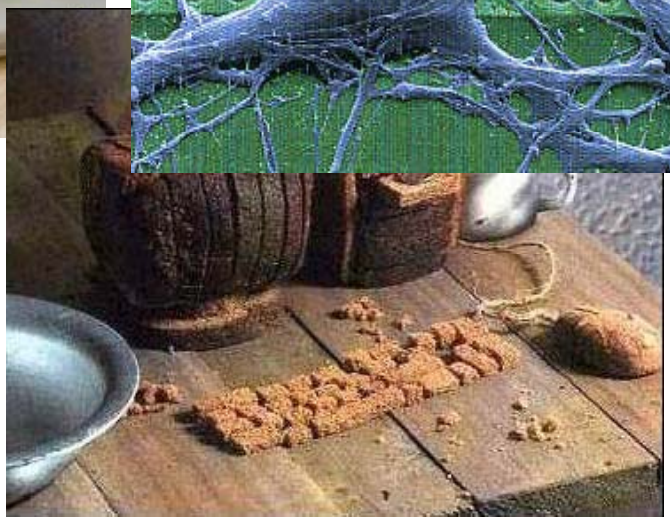
Universität Karlsruhe (TH) Forschungsuniversität • gegründet 1825

Overview

- Motivation
- Organic Computing
- System Model, State Spaces
- Some Key Properties of OC systems
- Degree of (Controlled) Self-organisation
- Challenges for Research
- Concluding Remarks

What Organic Computing is *not about*





So, what is it about?

- Collections of **intelligent (embedded) systems**
  (scenarios like **smart house, car, office, factory, shop, healthcare**,…
  …**ubiquitous, pervasive computing**).
- Potentially **unlimited networks** (large number, mobility)
- **Spontaneous local interaction,** leading to unexpected global behaviour (**emergent phenomena** as a result of **self-organisation**)
- **Robust services** in dynamically changing environments
  (e.g. mobile communication).
- **Flexible behaviour** as a reaction to varying external constraints
  (e.g. traffic light control)
- **Design, management and acceptance problems** wrt increasingly **complex systems**
  → **Controllability? Trustworthiness?**

⇒ **We have to come up with good ideas for**

- **designing, managing, and controlling unlimited, dynamical networks of intelligent devices,**
- **utilising the available technology for the utmost benefit to humans.**

Origin of Organic Computing

Workshops of the GI-/ITG-Sections on Computer Engineering in 2002

- Information technology is moving towards the ubiquitous networked computer.
- Complex ubiquitous systems need new concepts for organization and user interfaces to remain manageable and controllable.
- Future computer systems have to be designed with respect to human needs.
- Future computer systems have to be trustworthy.
- Future computer systems have to be robust, adaptive, and flexible.
- **Systems having these properties will be life-like.**
  We call them *Organic Computer Systems*.

GI/ITG Position paper 2003:

Vision for System Architecture > 2010

- **Organic Computer Systems**
  - will possess lifelike properties.
  - will consist of autonomous and cooperating sub systems and will work, as much as possible, in a self-organised way.
  - will adapt to human needs,
  - will be robust, adaptive, and flexible,
  - will be controlled by objectives (“goal-driven”),
  - will provide customized service in a user-friendly way,
  - will be trustworthy.
- **Self-organisation** allows for adaptive and context aware behaviour:
  - self-configuring
  - self-optimizing
  - self-healing
  - self-protecting
  - self-explaining
  - self-managing
  - …

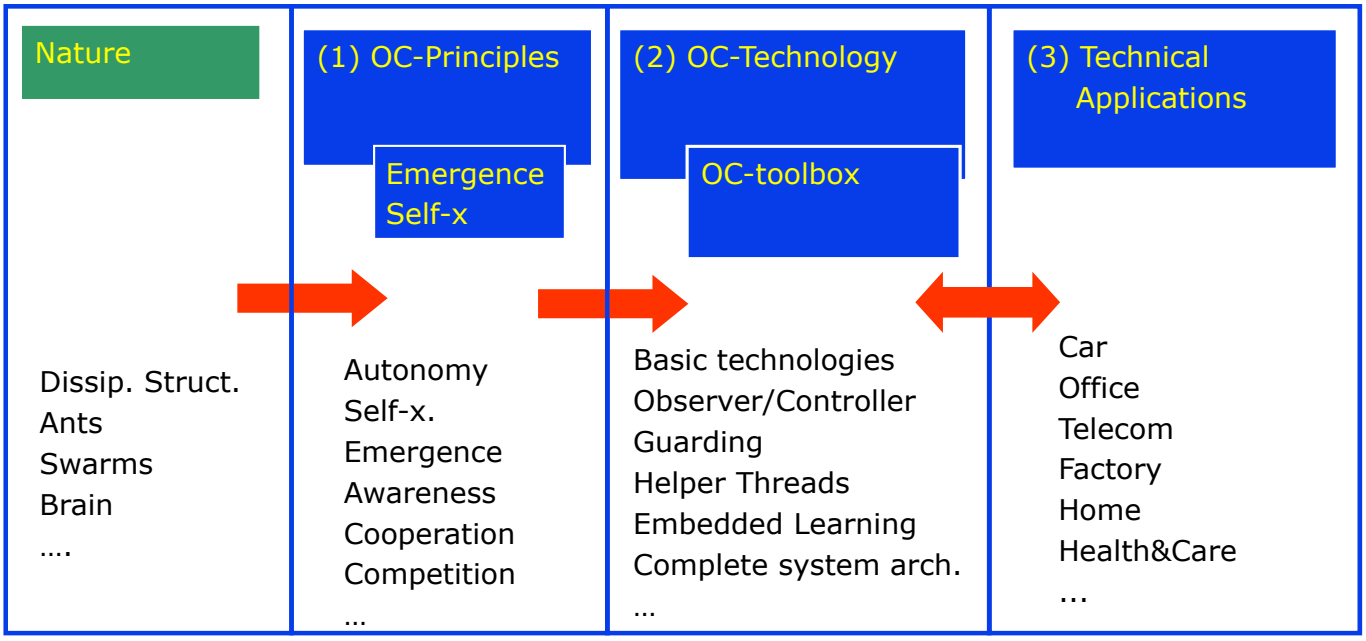
Organic Computing

It is not the question,
whether adaptive and self-organising systems
will emerge,
but ***how*** they will be designed and controlled.

German Framework for Research on OC :

DFG priority program 1183 „Organic Computing“ (2005 – 2011)
18 projects, ~2 Mio€ per year

www.organic-computing.de/SPP



Evaluation of progress

- How far did we get already?
- Needs a clear understanding of the vision of organic computing and its defining properties
  - self-x
  - adaptive
  - robust
  - controllable
  - trustworthy
  - respecting human needs
- Should consider related work in neighboring areas like
  - autonomic computing
  - ubiquitous computing
  - pervasive computing
  - …

Interpretations of Organic Computing

- **Organic Computing** is a form of biologically-inspired computing with organic properties. These include:
  - the growth, life or development cycle
  - the ability to adapt, learn, and evolve
  - emergent behaviour or emergent properties
  - steady change or growth, as opposed to instant change
  - regulatory feedback
  - composed of heterogeneous (diverse) parts

(taken from Wikipedia)

Remark

- Organic Computing should not be reduced to „biologically inspired computing“.
- But: Nature is full of interesting examples of systems with self-x properties. **Therefore, nature definitely may be a valuable source of inspiration for the engineering of organic computing systems.**

Interpretations of Organic Computing (2)

- Once fully developed, Organic Computing will be a conceptual framework, indeed a branch of science, that will form the basis for understanding the organic structure of Life on its molecular, organismic, cognitive and societal levels, and for building an organically structured information technology.
  (taken from www.organic-computing.org)

Remark:
This is far beyond the scope and claim of the „Organic Computing Initiative“.
(influenced by the „original“ definition of Organic Computing)

- A system is called organic if all of its components and subsystems are well coordinated in a purposeful manner.
  Organic structures realize themselves as hierarchically nested processes, structured such as to be able to meet upcoming challenges by goal-oriented reactions.
  (taken from www.organic-computing.org)

Remark:
This is too weak, too few systems are excluded.

Interpretations of Organic Computing (3)

- A system is called organic, if it is a distributed, self-organizing system.
  (cited from a recent discussion)

Remark:
This is too simplified, important properties are missing

- **adaptive**, i.e. capable of adjusting its behaviour to dynamically changing environmental objectives,in particular, capable of **learning**
- **robust**, i.e. providing a requested service even under variations of internal or external parameters,
- various **self-x** properties
- **trustworthy**, i.e. reacting in a predictable way, showing exactly the expected behaviour,
- **controllable**, i.e. allowing for external observation of essential system parameters and for external control actions modifying the behaviour of the system.

Related Notion: Autonomic Computing System

www.research.ibm.com/autonomic

1. Needs to "know itself" - its components, current status, ultimate capacity, and all connections to other systems to govern itself (**self-aware**).
2. Configures and reconfigures itself under varying (and in the future, even unpredictable) conditions (**self-configuring**).
3. Always looks for ways to optimize its workings (**self-optimizing**).
4. Recovers from routine and extraordinary events that might cause some of its parts to malfunction (**self-healing**).
5. Is an expert in self-protection, detects, identifies and protects itself against various types of attacks (**self-protecting**).
6. Knows its environment and the context surrounding its activity, and acts accordingly - in a word, adapting to the environment (**adaptive**).
7. Functions in a heterogeneous world and implements **open standards** - it cannot, by definition, be a proprietary solution.
8. Anticipates the optimized resources needed while keeping its complexity hidden - without involving the user in that implementation (**autonomous**).

Autonomic vs. Organic Computing

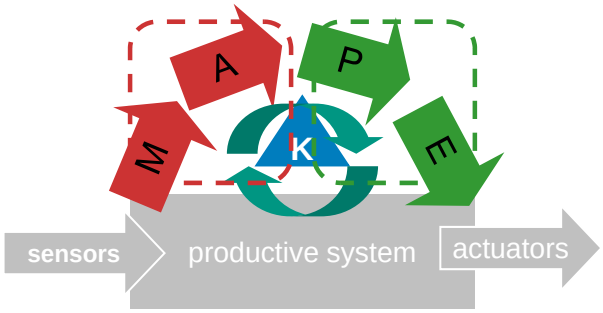
- Autonomic Computing is focussing on enterprise server architectures *(although its principles are applicable in other areas, too)*
- Organic Computing is focussing on collections of interacting intelligent devices (in technical applications). *(although its principles are applicable to monolithic systems, too)*
- Autonomic Computing removes human users from the system control loop *(no explicit human interference or control).*
- Organic Computing emphasizes the interaction with human users and respecting their needs.
- Organic Computing allows for (and requires) controlled self-organisation *(which sounds like a contradiction – traditionally, a system is either self-organised or controlled, but not both!)*

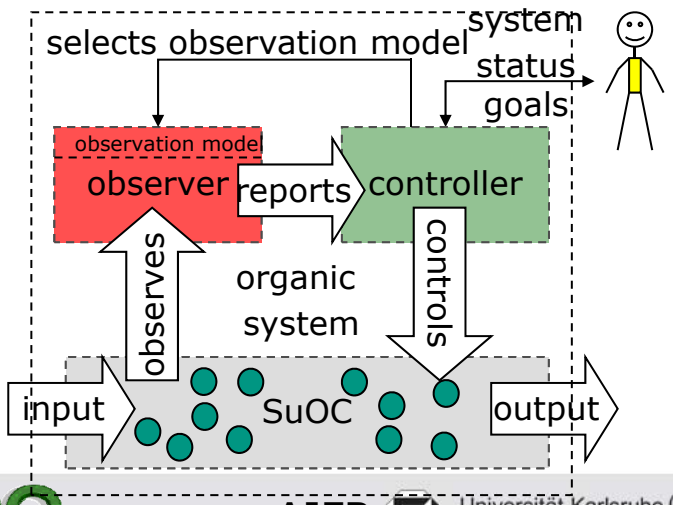
⇒ many similarities (in particular, self-x properties), few (but some essential) differences

AC/OC (Generic) Concepts for Control

- IBM's **MAPE cycle** for autonomic computing
  - Monitor
  - Analyze
  - Plan
  - Execute
  - Knowledge

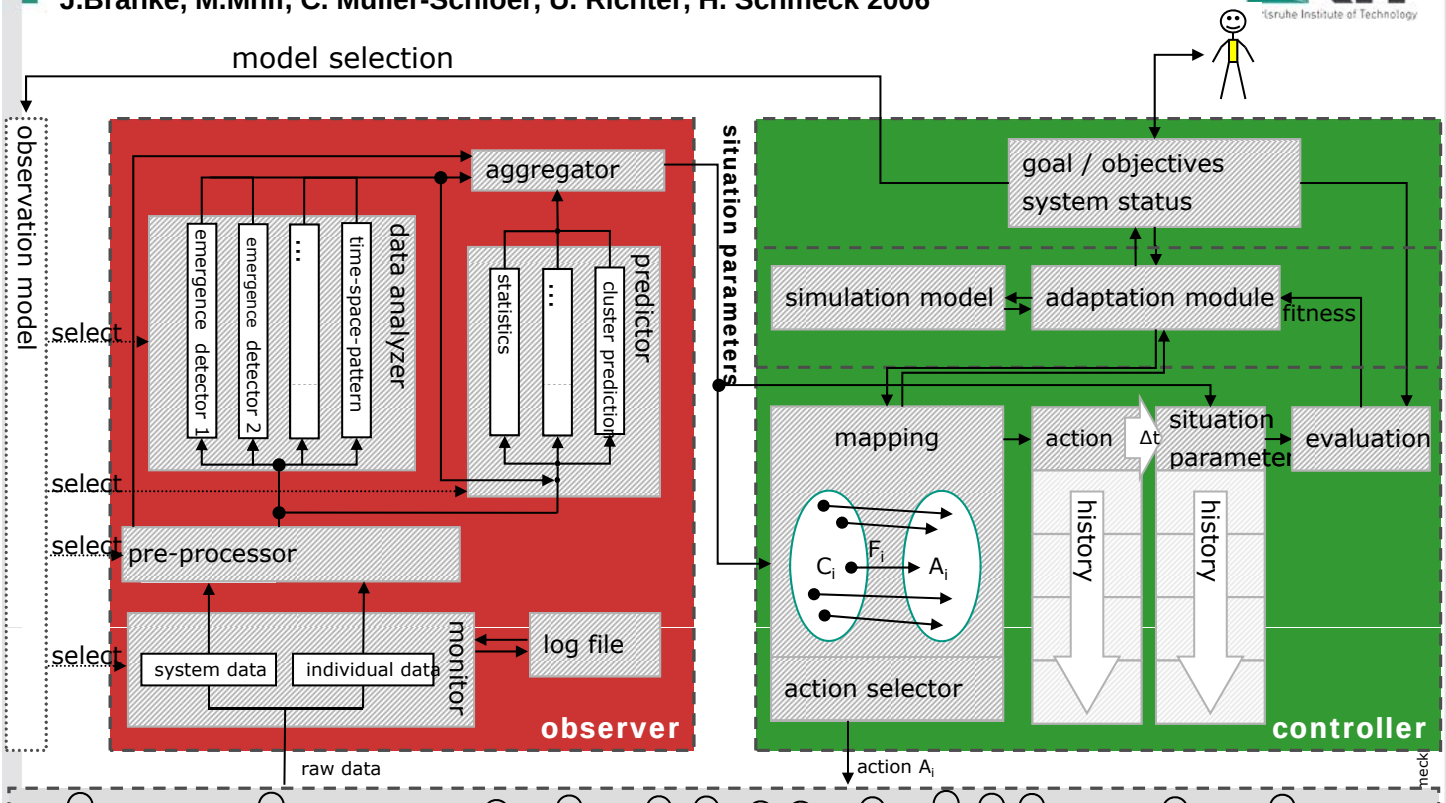
  (called "autonomic element")
- System under observation and control (SuOC)
  - A set of interacting elements/agents.
  - Does not depend on the existence of observer/controller.
- Distributed and/or central **observer/controller-architecture**
  - Driven by external goals
- Multilevel organization





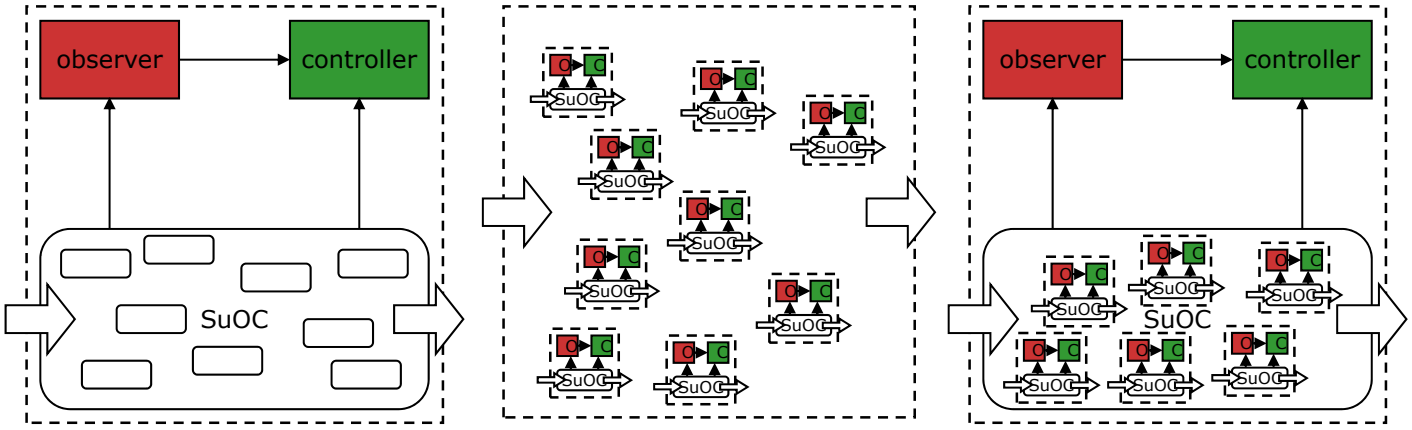
Generic O/C-Architecture

J.Branke, M.Mnif, C. Müller-Schloer, U. Richter, H. Schmeck 2006



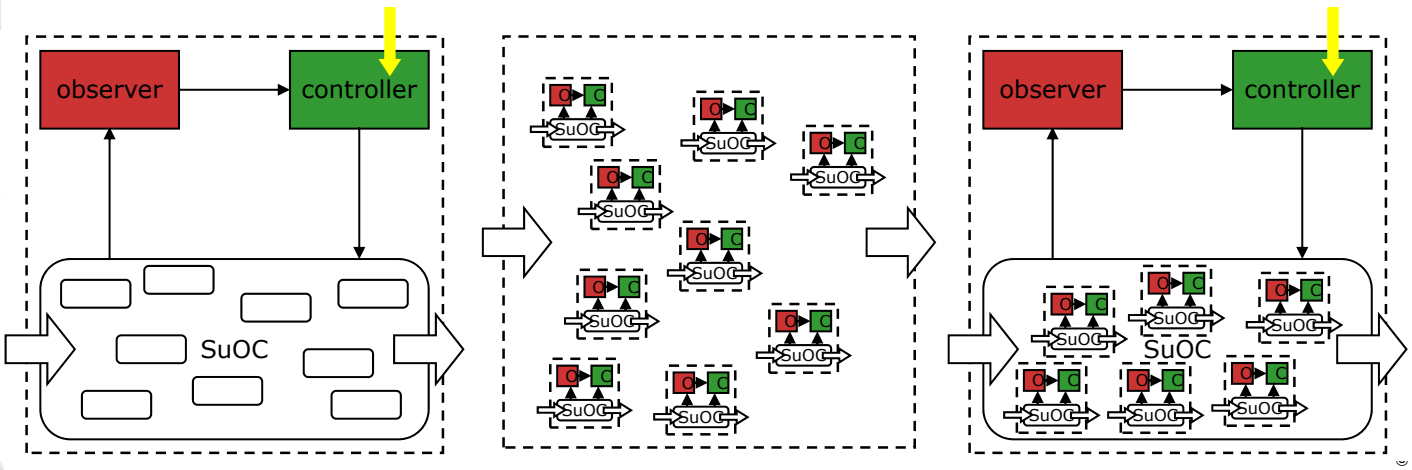
Realisation of OC systems

1. **Central**: One observer/controller for the whole system.
2. **Distributed**: An observer/controller on each system component.
3. **Multi-level**: An observer/controller on each system element as well as one for the whole system.



Types of control actions

1. **Control the environment** (e.g. speed limit in traffic)
2. **Control the communication** (messages, addresses, neighborhoods,…)
3. **Control the local behavior of components** (reconfigure HW, update software, modify skills, set new local objectives,..)



Evaluation of AC / OC systems

- Statements on key properties of OC systems need a clear understanding of their meaning.

Therefore:

- We need a common understanding and quantitative measures of
  - Robustness
  - Adaptivity
  - Flexibility
  - Autonomy
  - (Controlled) Self-organisation
  - Trust
  - …

Definitions of Self-organisation

Definitions of **Self-organisation** in the Web :

- www.geos.ed.ac.uk/homes/mnaylor/Glossary.html:
  The ability of certain non-equilibrium systems to develop structures and patterns in the absence of external control or manipulation.
- wordnet.princeton.edu/perl/webwn
  self-organization: organizing yourself (especially organizing your own labor union)
- en.wikipedia.org/wiki/Self-organisation:
  Self-organization refers to a process in which the internal organization of a system, normally an open system, increases automatically without being guided or managed by an outside source. Self-organizing systems typically (though not always) display emergent properties.

Definitions of Self-organisation

- http://pespmc1.vub.ac.be/SELFORG.html *(Principia Cybernetica Web)*: Self-organization is a process where the organization (constraint, redundancy) of a system spontaneously increases, i.e. without this increase being controlled by the environment or an encompassing or otherwise external system
- http://www.calresco.org/sos/sosfaq.htm#1.2 : The essence of self-organization is that system structure often appears without explicit pressure or involvement from outside the system. In other words, the constraints on form (i.e. organization) of interest to us are internal to the system, resulting from the interactions among the components and usually independent of the physical nature of those components. The organization can evolve in either time or space, maintain a stable form or show transient phenomena. General resource flows within self-organized systems are expected (dissipation), although not critical to the concept itself.

Definitions of Self-organisation

Gero Mühl et al. (2006)

A system can be called **self-organising**

"if it is

(i) **self-managing**
[the system adapts to its environment without outside control],

(ii) **structure-adaptive**
[the system establishes and maintains a certain kind of structure (e. g. spatial, temporal), providing the system's primary functionality], and

(iii) **employs decentralised control**
[the system has no central point of failure]."

Definitions of Self-organisation

Giovanna Di Marzo Serugendo et al. (2005):

- **Self-organisation in Engineered Systems**

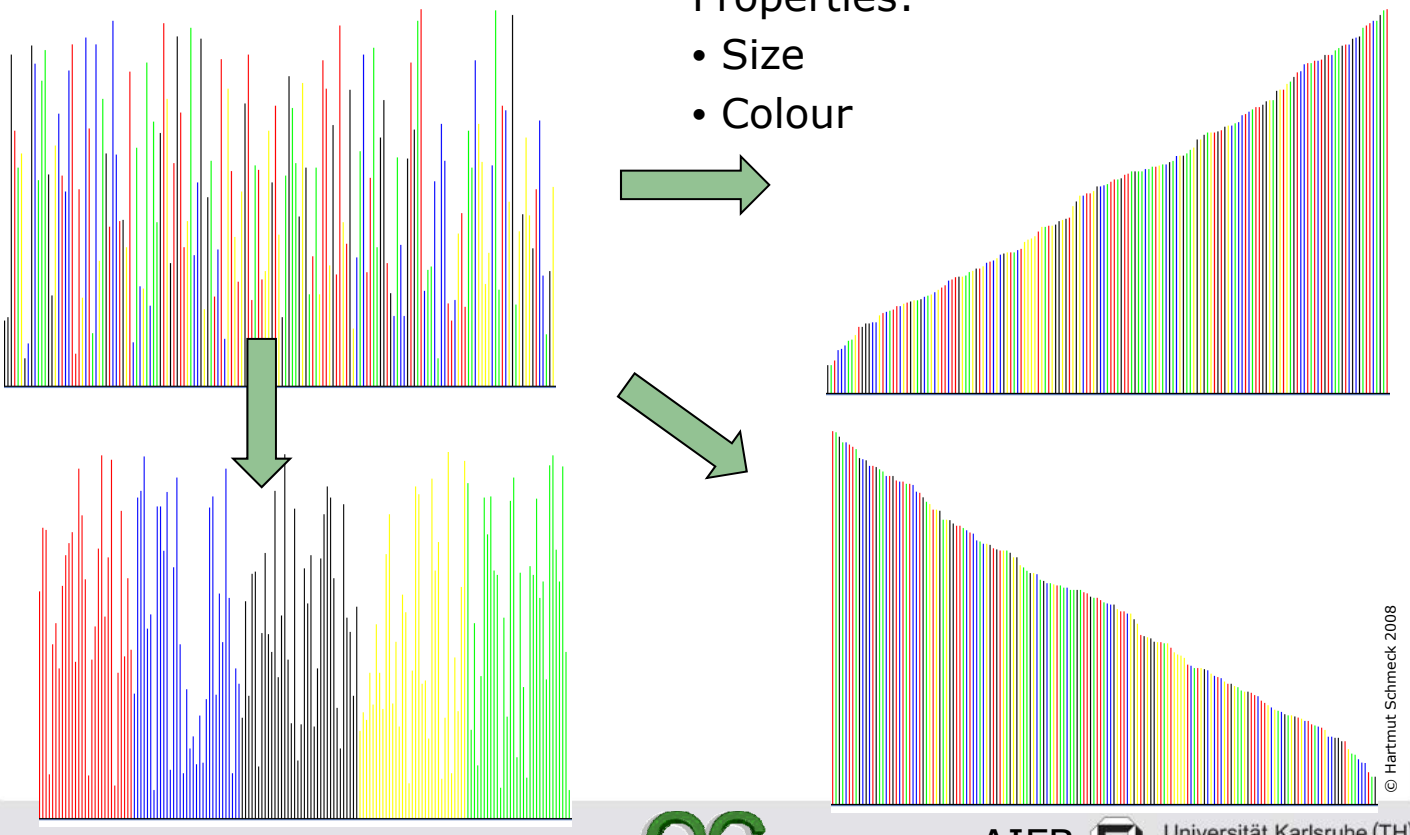
"**Self-organisation** is the **process** enabling a system to change its organisation in case of environmental changes **without explicit external command** ."

"**Strong self-organising** systems are those systems where there is re-organisation with no explicit central control, either internal or external."

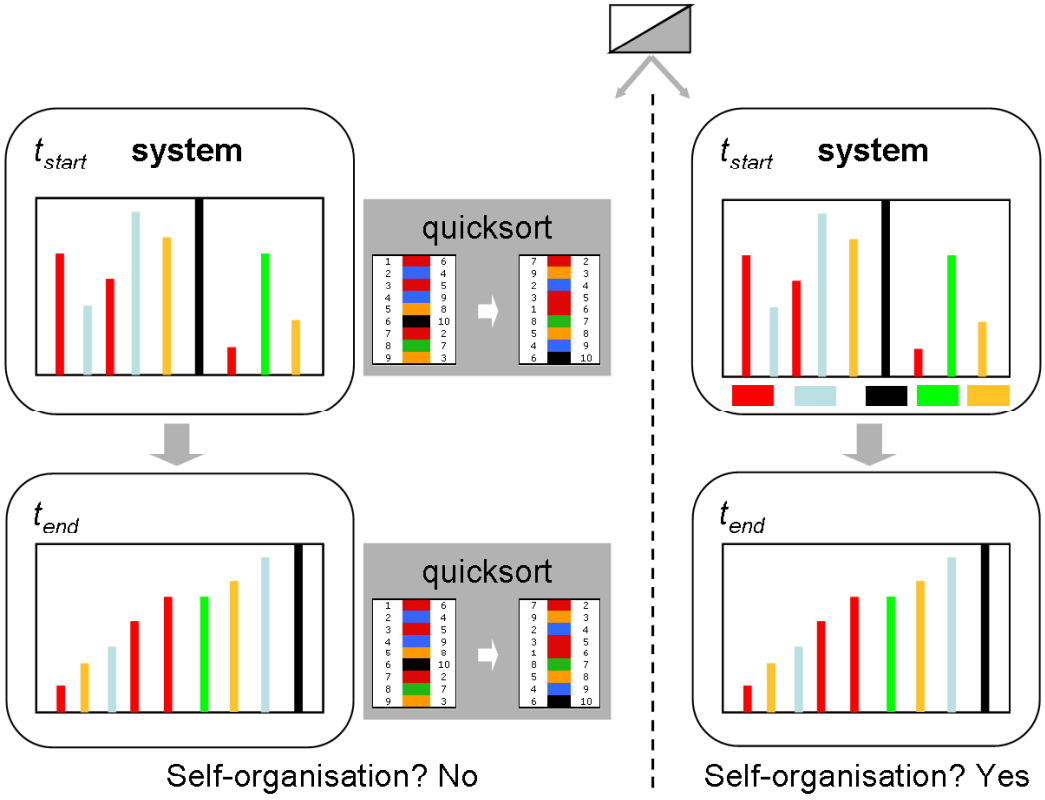
"**Weak self-organising** systems are those systems where, from an internal point of view, there is re-organisation under an internal central control or planning."

Example Scenario: Simple Sorting

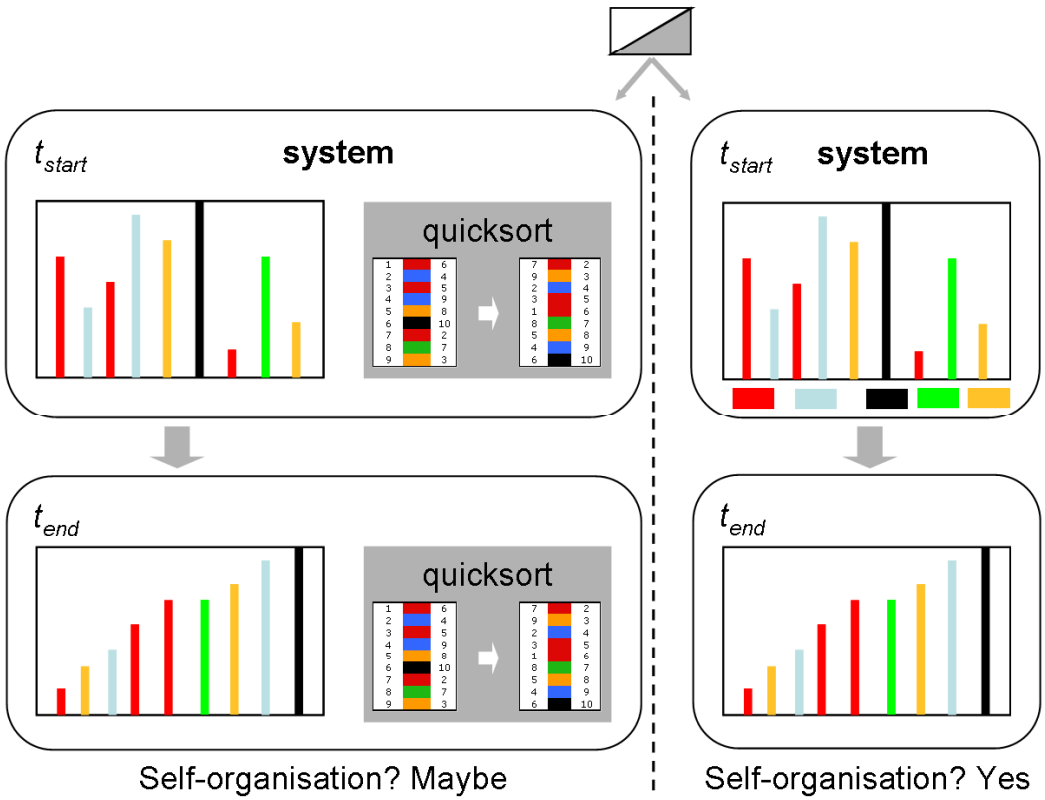
Properties:
- Size
- Colour



Example Scenario: Simple Sorting – Control Structures?

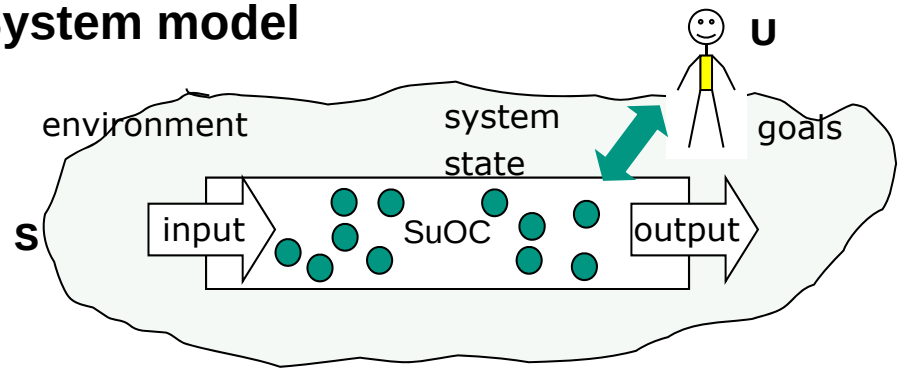


Example Scenario: Simple Sorting – Control Structures?



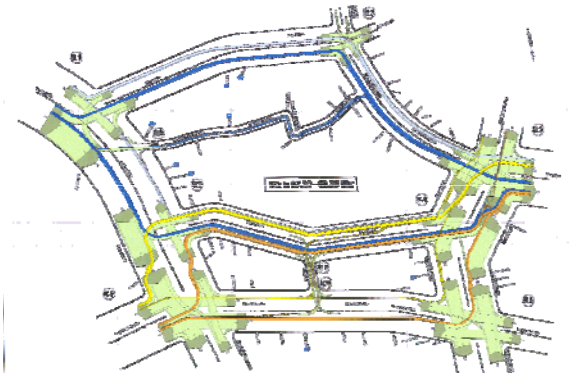
© Hartmut Schmeck 2008

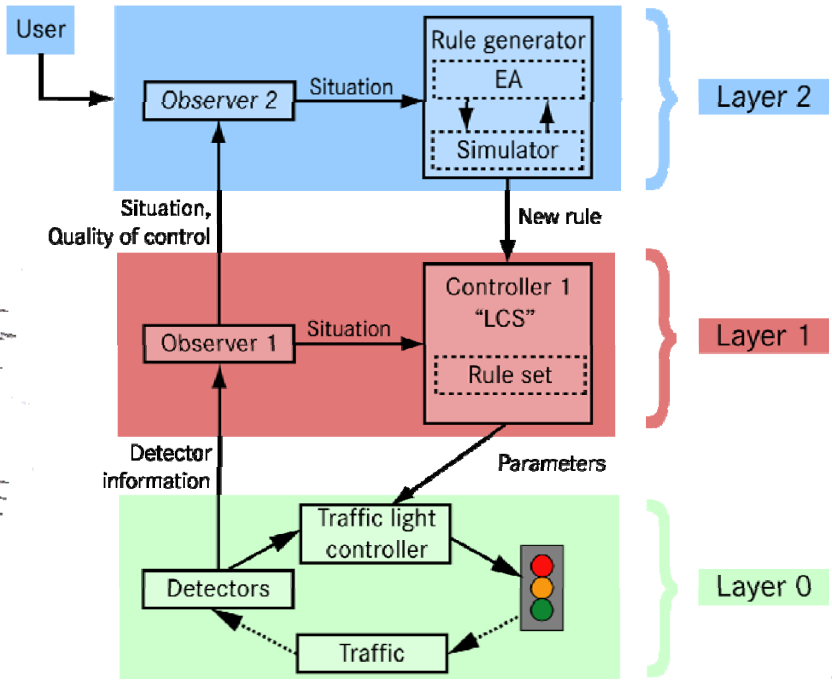
System model

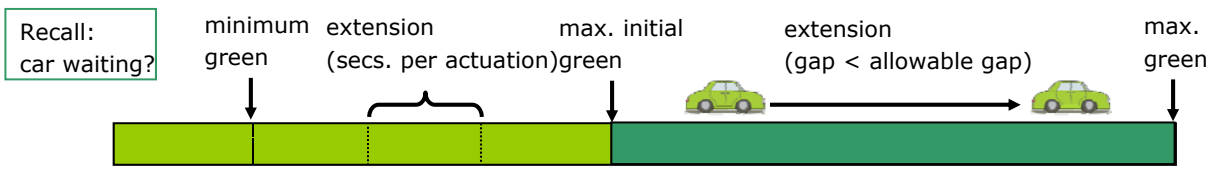


- "System" **S** with input (sensors), output (actuators)
- Current state z(t) described by values of a set of attributes / parameters
  - structure (nodes and edges with attributes)
  - input, output
  - "environmental" parameters
  - → **state space** of S
- "external observer" (user) **U**
  with potential access to (almost) all the parameters
- U provides system objectives (goals)
  (**evaluation criteria η, mapping state space into IR^k**)
- U may control the system (modify parameter values)

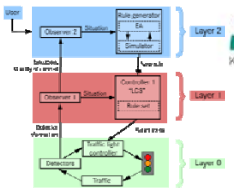
Example Application: Organic Traffic Control



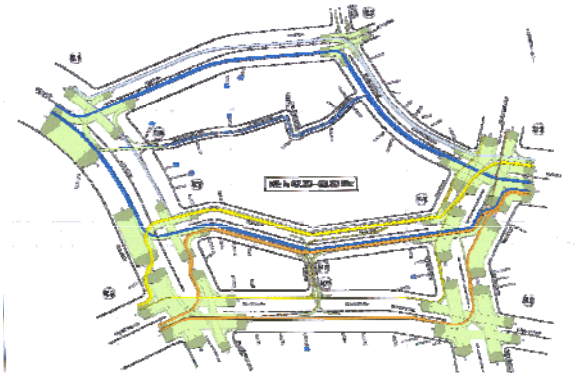




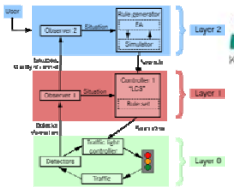
Example Application: Organic Traffic Control



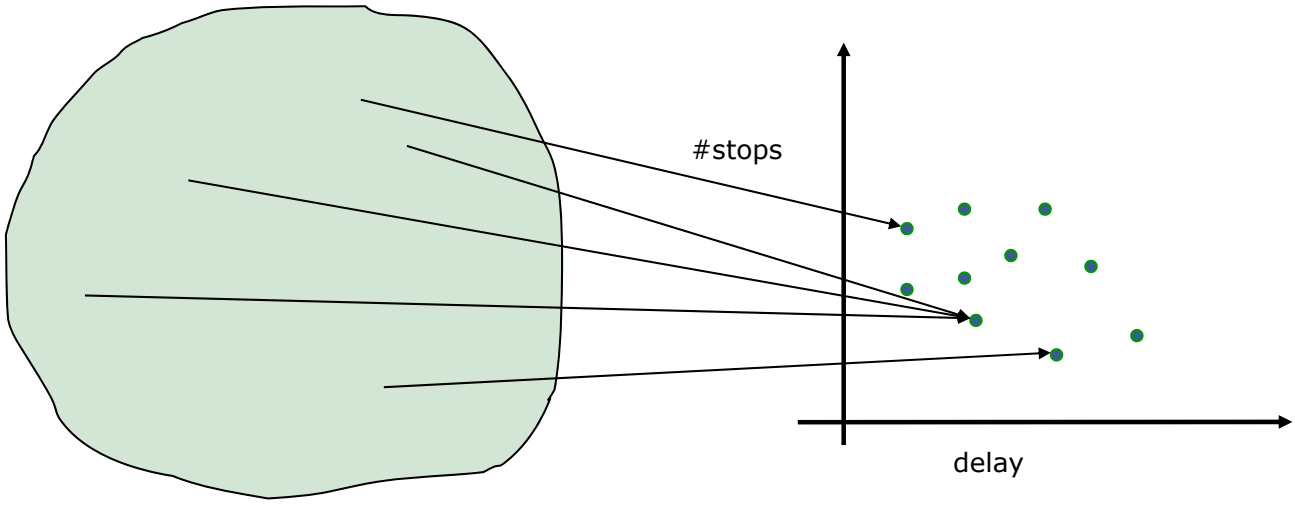
- State space attributes for one intersection:
  - Street network data
  - Detector information (#cars, arrival times, speed, gaps,…)
  - Traffic light controller settings
  - Rule base (layer 1 data)
  - Simulator data, EA parameters
  - Objectives (delay, #stops,…)



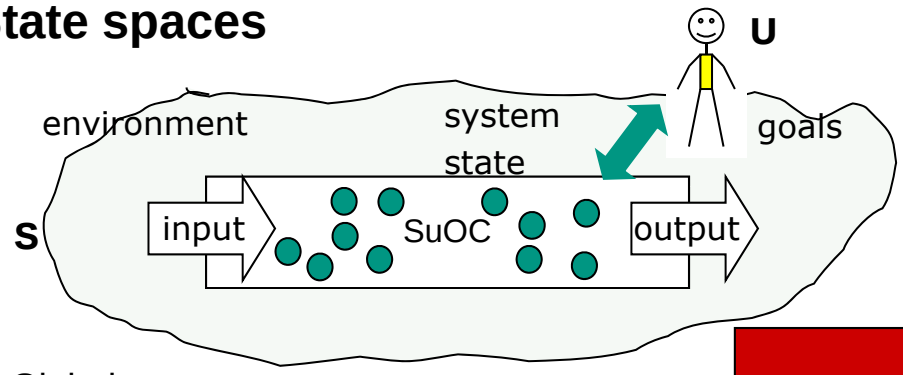
Example Application: Organic Traffic Control



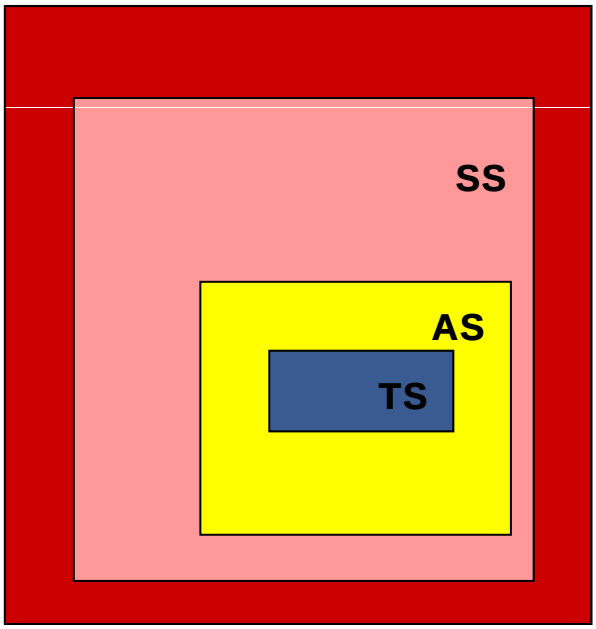
- Characterization of system properties
  - in state space
  - –in objective space



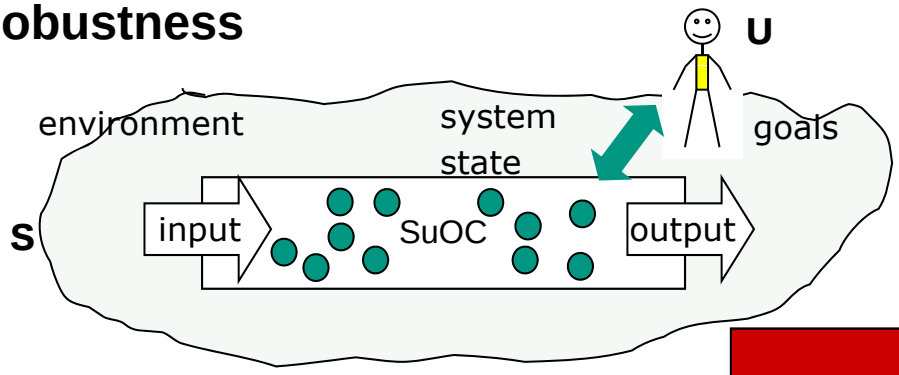
State spaces



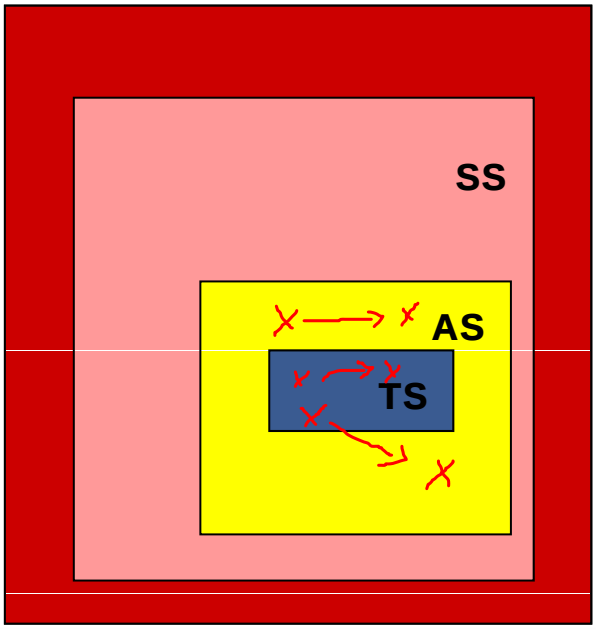
- Global state space
- **Target Space TS**:
  „zero space“ of evaluation criteria η
- **Acceptance Space AS**
  with respect to **acceptance criteria** θ
  (threshold), determined by U
- **Survival Space SS**
  States, from which AS may be reached by control actions.
- The rest:
  states with “unrepairable” defects (“**dead zone**”)
- **Disturbance** δ (or “modification”)
  mapping state space into itself



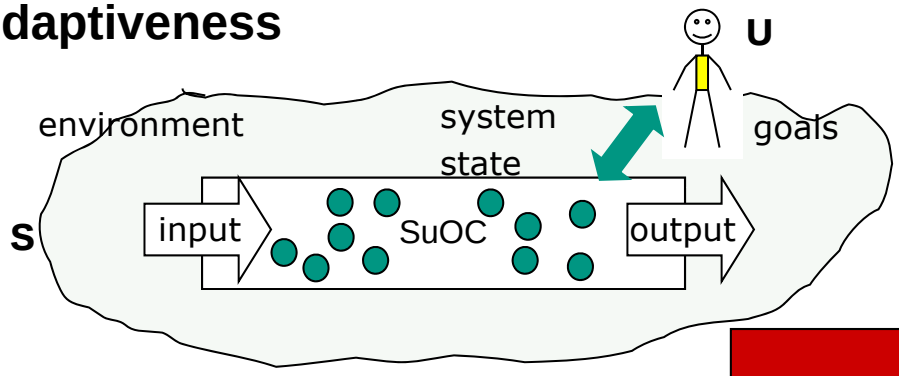
Robustness

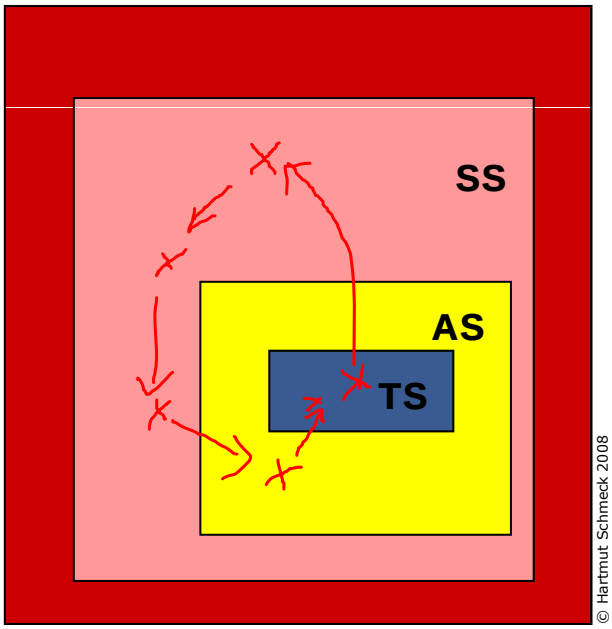


- Let D be a set of disturbances.
- S is **(strongly) robust wrt. D** iff every $\delta \in D$ maps TS and AS into itself, resp..
- S is **weakly robust wrt. D** iff every $\delta \in D$ maps AS into AS.



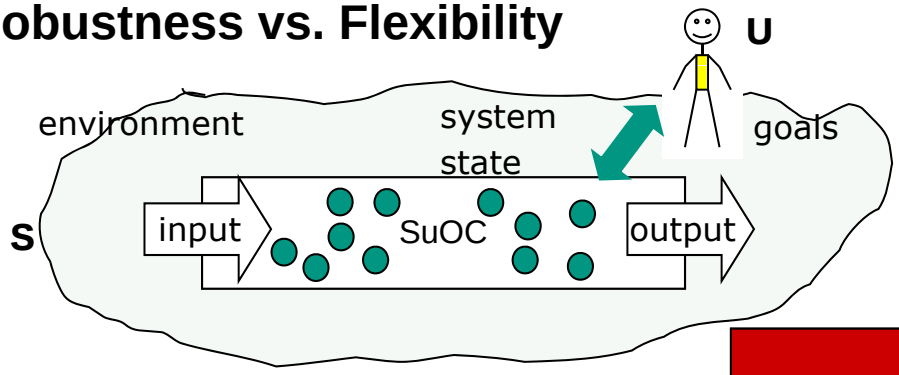
Adaptiveness

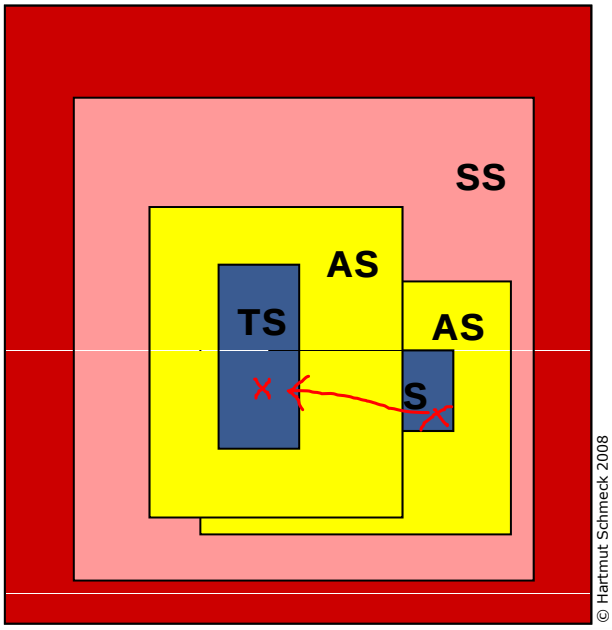




- Let D be a set of disturbances.
- S is **adaptable wrt. D** iff
  every $\delta \in D$ maps AS into SS and
  S can return into AS (by external control).
- S is **adaptive wrt. D** iff
  S is adaptable wrt. D and after every disturbance it returns to AS without external control actions.

Robustness vs. Flexibility

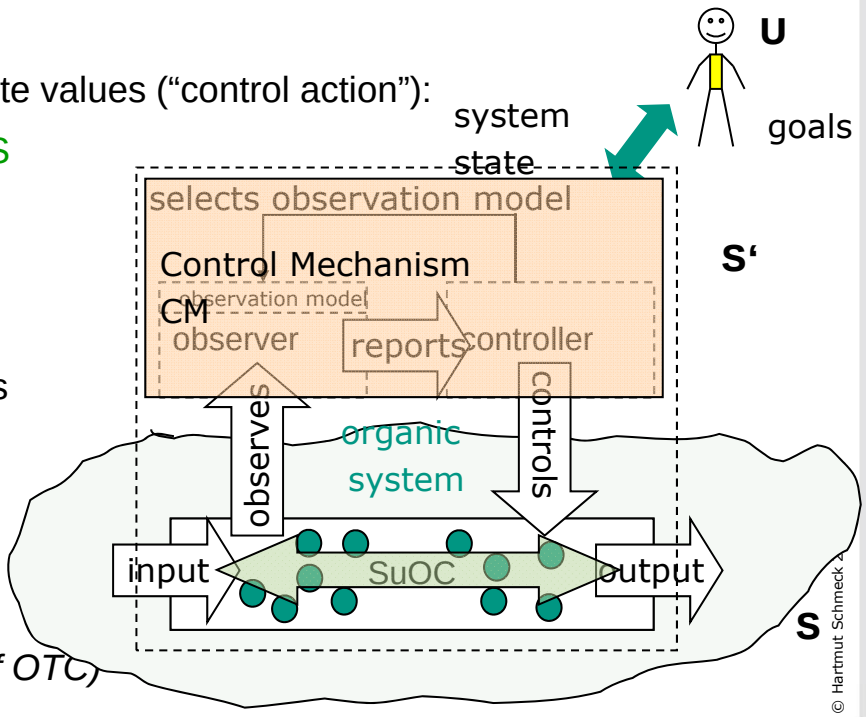




- Distinguished based on type of disturbance:
  - Modification of state attributes without changes in objectives η:
    → **Robustness**
  - Modification of objectives only (→ changes in target and/or acceptance space)
    → **Flexibility**

Control mechanisms

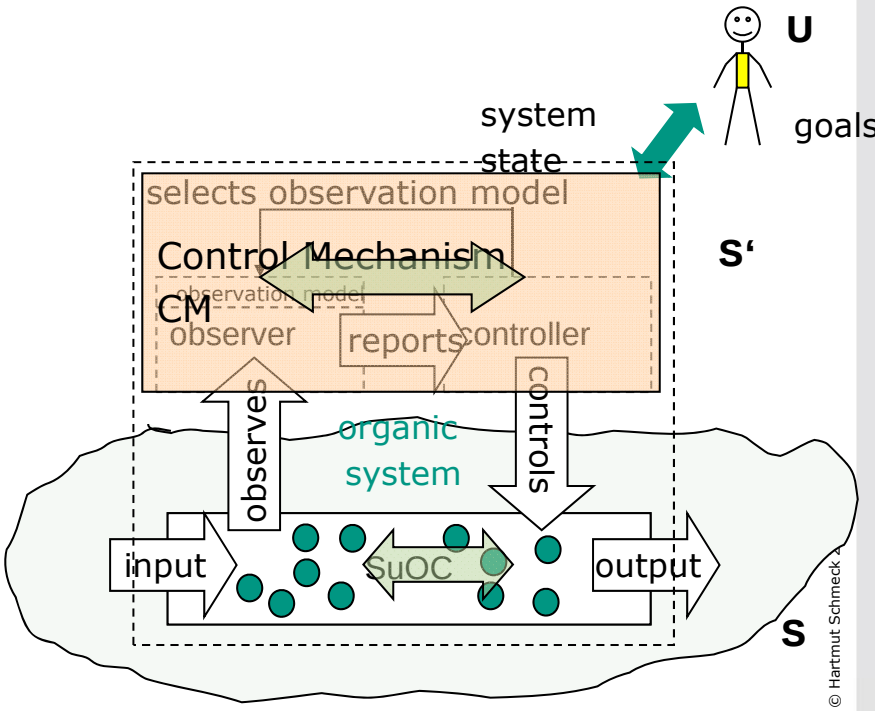
- Extend S with **“control mechanism” CM**
  → combined system S‘
  (e.g. O/C architecture)
- CM observes the state of S.
- CM may modify certain attribute values (“control action”):
- The **configuration space** of S is spanned by some of the state space attributes.
  (e.g. traffic light parameters, speed limits,…)
- Every **action** of CM influences a selection of the attributes of the configuration space.
  (e.g. length of green phases)
- CM has an **action repository B**.
  (current rule base of layer 1 of OTC)



AIFB Universität Karlsruhe (TH) Forschungsuniversität • gegründet 1825

Control mechanisms

- Extend S with **“control mechanism” CM**
  → combined system S‘
  (e.g. O/C architecture)
- S‘ has configuration space (for external control) with attributes of CM and of S.
- User U has possibilities for (external) control of S‘
  - Providing objectives **control actions c_{obj}**
  - Influencing (internal) parameters of S, **actions c_{low}**



AIFB Universität Karlsruhe (TH) Forschungsuniversität • gegründet 1825

Control mechanism

- Variability of the **(internal) configuration space** of S:
  $\mathbf{V_i}$ = log #configurations of S
- Variability of the (**external) configuration space** of S‘:
  $\mathbf{V_e}$ = log #configurations of S‘

Assumption:
There should be a complexity reduction from lower to upper layers (higher level of abstraction)

Define:

Complexity reduction

$$R = V_i - V_e$$

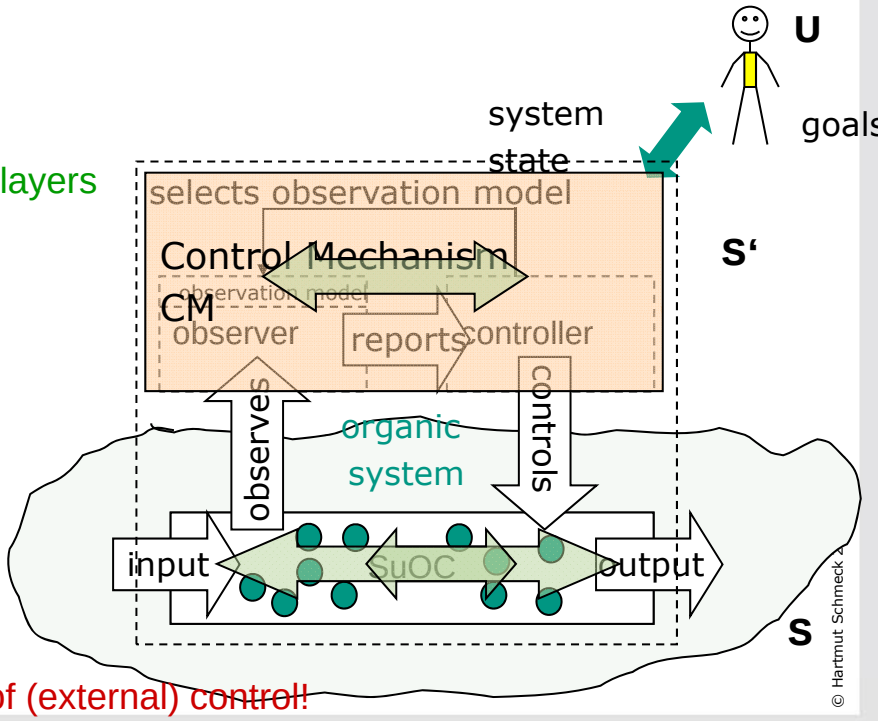
Degree of Autonomy

$$\alpha = R / V_i \leq 1$$

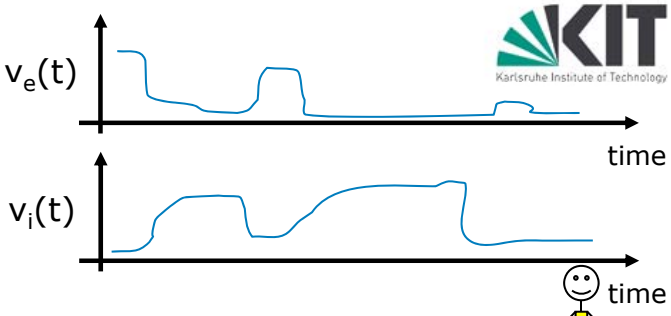
potentially $\alpha < 0$!

Full autonomy at $\alpha = 1$

But : undesirable, no possibility of (external) control!



Dynamic degree of autonomy



Consider **effective flow of control!**

$v_i(t)$ = #bits of (internal) action at time t

$v_e(t)$ = #bits of (external) action at time t

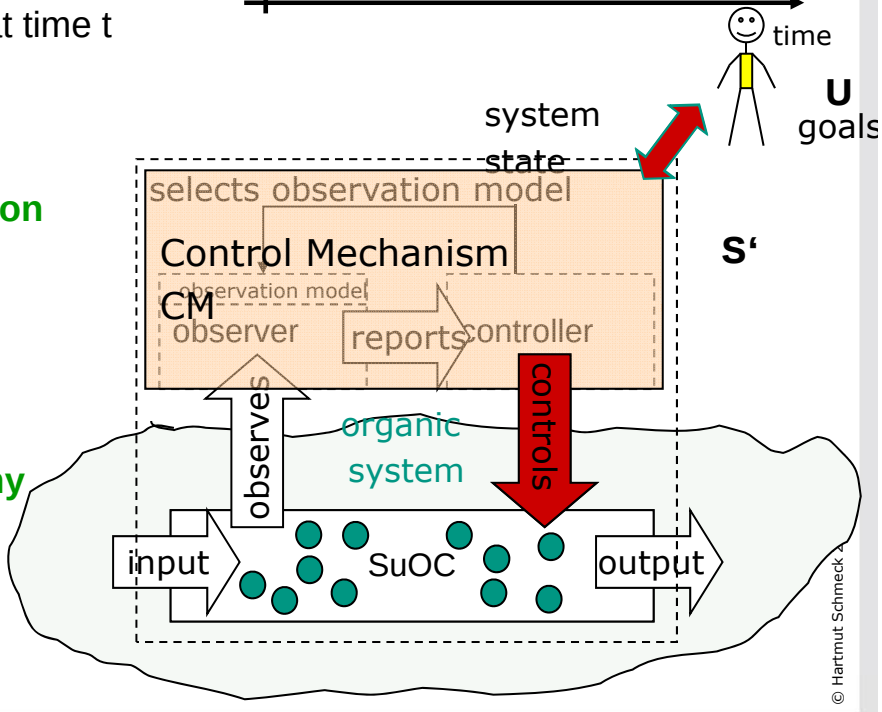
Define

dynamic complexity reduction
in time interval $[t_1, t_2]$

$$r = \int_{t_1}^{t_2} (v_i(t) - v_e(t))dt$$

dynamic degree of autonomy

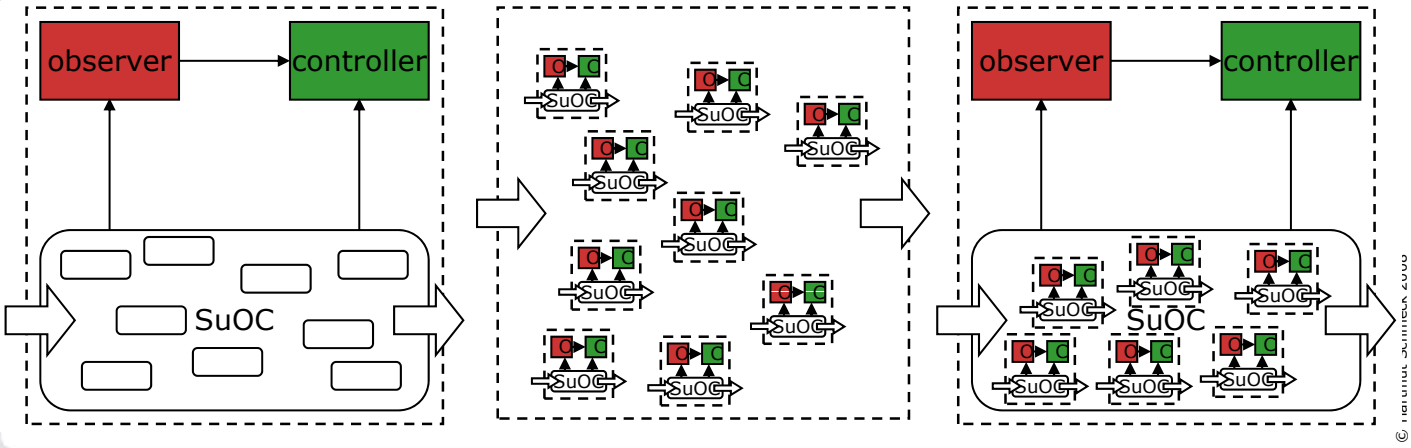
$$\beta = \int_{t_1}^{t_2} (v_i(t) - v_e(t))dt \Big/ \int_{t_1}^{t_2} v_i(t)dt$$



Degree of Self-organization

Let S be adaptive, consisting of m elements (m > 1) with high degree of autonomy and (distributed) k CMs k≥ 1

- Then let (k : m) be the **degree of self-organization**.
- S is **self-organized**, iff k >1
- S is **strongly self-organized**, iff k ≥ m
- S is **weakly self-organized**, iff k=1



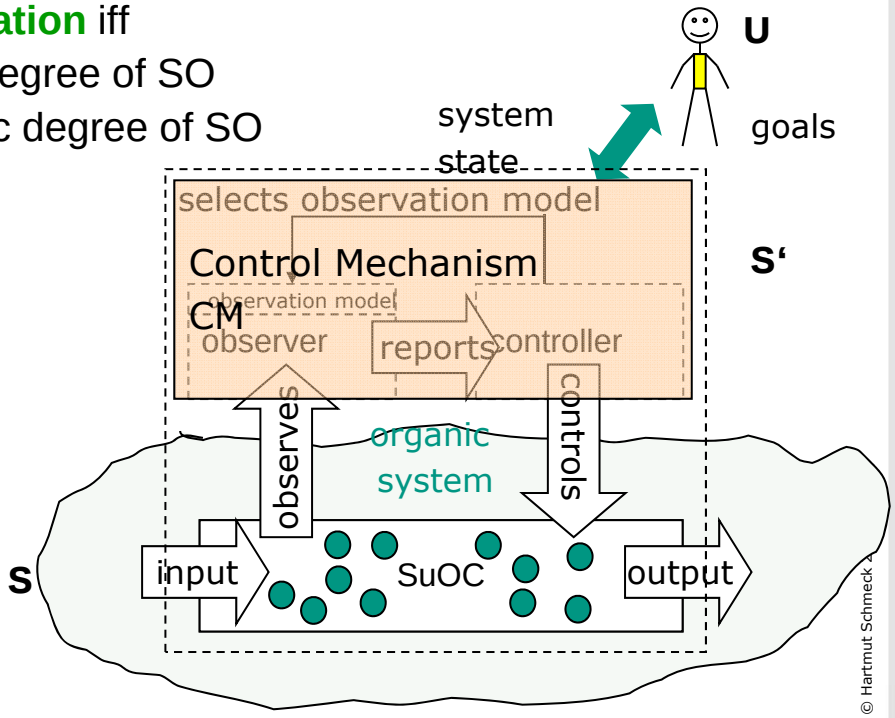
Controlled Self-organization

Definition:

A self-organizing system allows for **controlled self-organization** iff

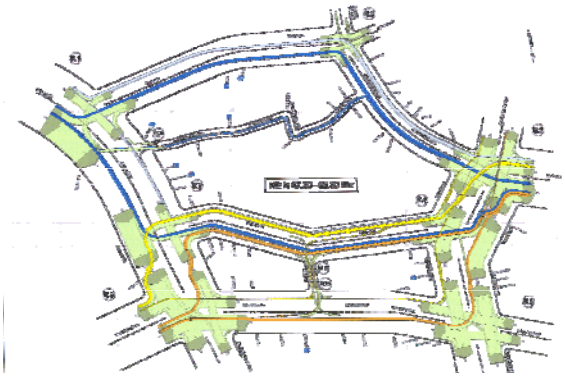
(i) It has a small static degree of SO

(ii) It has a high dynamic degree of SO

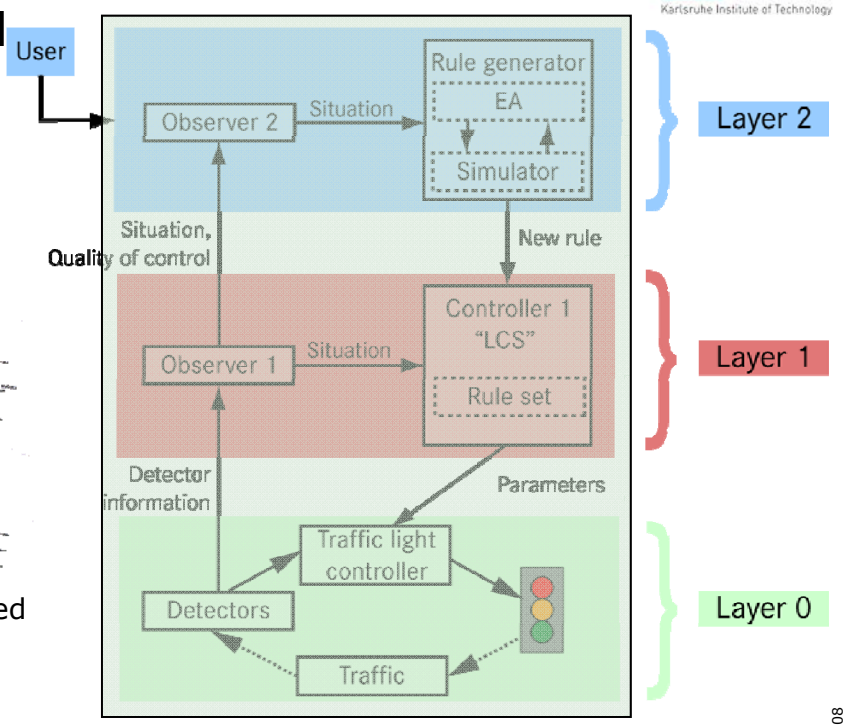


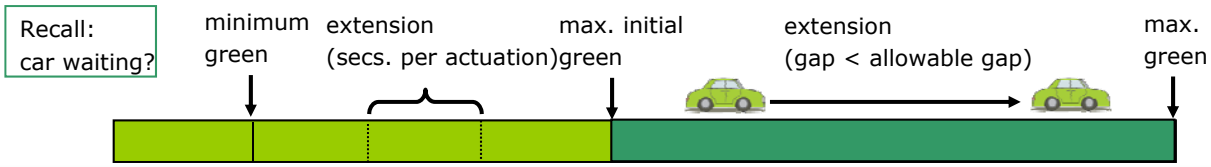
Example Application: Organic Traffic Control

- Robust
- Adaptive
- Flexible



- Single node: weakly self-organized
- Street network: strongly self-organized





Challenges for Research on OC/AC systems (or on self-organizing, adaptive systems)

- **Learning**:
  - Potential of online- and offline learning
  - Collaborative learning
- **Coordination and collaboration**
  - Typical patterns of c & c in OC systems
  - Benefits, necessity of c & c
- **Design**
  - Finding the right balance between explicit design and degrees of freedom
  - Finding the right separation of concerns in hierarchical OC systems
- **Cognition**
  - Finding out "the needs of human users" (or, of the environment).
  - Detecting anomalies, distinguishing the "good" from the "bad".

Challenges for research on OC/AC systems (2)

- **Control**
  - Finding the right balance between “SO” and “control” phases.
  - Does “control by objectives” work?
- **Trust**
  - Can OC-systems be trustworthy?
  - Trust engineering?
- **Assessment**
  - Can there be “service level agreements” with guaranteed performance for OC-systems?
  - Benefits of AC / OC versus “standard” designs?
  - Benchmark applications for AC / OC
- …

Conclusion

- Organic Computing could be a perfect response to the challenges imposed by increasingly complex networks of increasingly more intelligent components (but not the only one!)
- Human needs should be the driving force of technical innovations – organic computing could be our chance to get human centered and manageable systems, operating to serve our needs.
- Self-x properties and controllability are orthogonal requirements, but we need them both.
- We have made already some considerable progress towards understanding effects of self-organisation, how to design OC systems.
- **There is still a long road ahead to a true realisation of our vision of organic computing, filled with a long list of fascinating challenges for research!**

Thanks for your attention!

Questions?