Information Society Technologies

ARTIST Workshop at DATE'06
W4: "Design Issues in Distributed, Communication-Centric Systems"

Optimisation of Robust Communication-Centric Systems

Rolf Ernst, Arne Hamann
TU Braunschweig

Overview

- motivation
- robustness in communication centric design
- robustness metrics and optimization
- experiments
- conclusion

Motivation

- design properties are subject to modifications
  - during the design process
    - *refinement of early design data estimations*
    - *refinement and changes of specification*
    - *exchange of platform components*
      - *– replace processor or memory type*
  - in the product lifecycle
    - *product updates (HW, firmware and SW)*
    - *integration of new components or subsystems*
    - *change in the environment*
      - *– applications (smartphone), technical system (motor speed)*
  - in the field
    - *dynamic systems*
    - *unplanned environment situations (resilience)*
- such changes introduce uncertainties and increase design risk
- find approaches to analyze and reduce risk

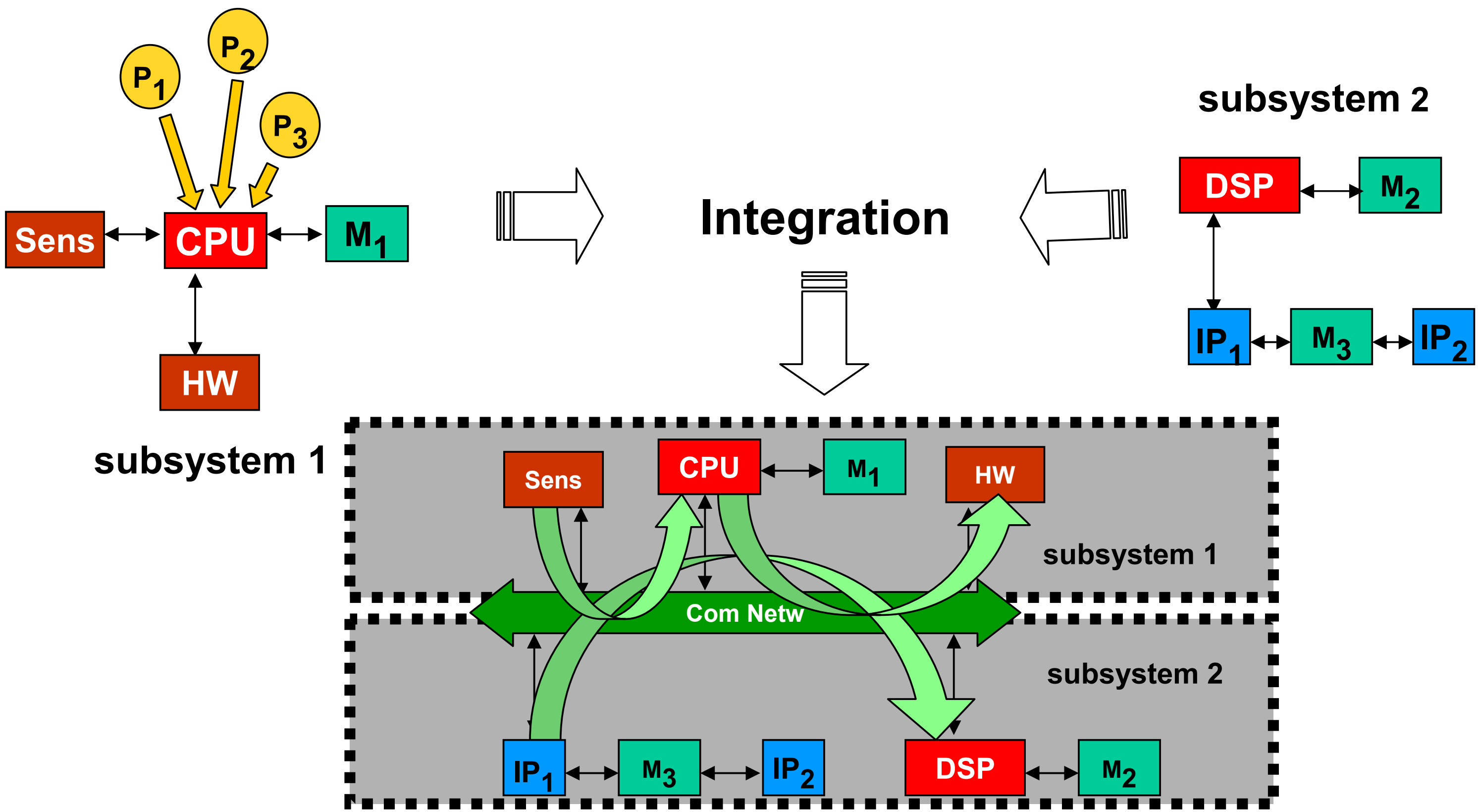
Embedded system robustness

- defining robustness – first approach
  - intuitive: a system is robust that provides required functionality and meets **contraints** under **design property modifications**
  - *robustness to HW failures not considered in the sequel*
- many different approaches to improving robustness
  - system learning and adaptation (control application)
  - statistical process optimization (e.g. Taguchi Method)
  - design centering (analog design)
- what approach is suitable to embedded systems?
  - what are the **constraints** that we want to consider?
  - what **design property modifications** should be included?
  - what **models** are appropriate?

Design properties considered

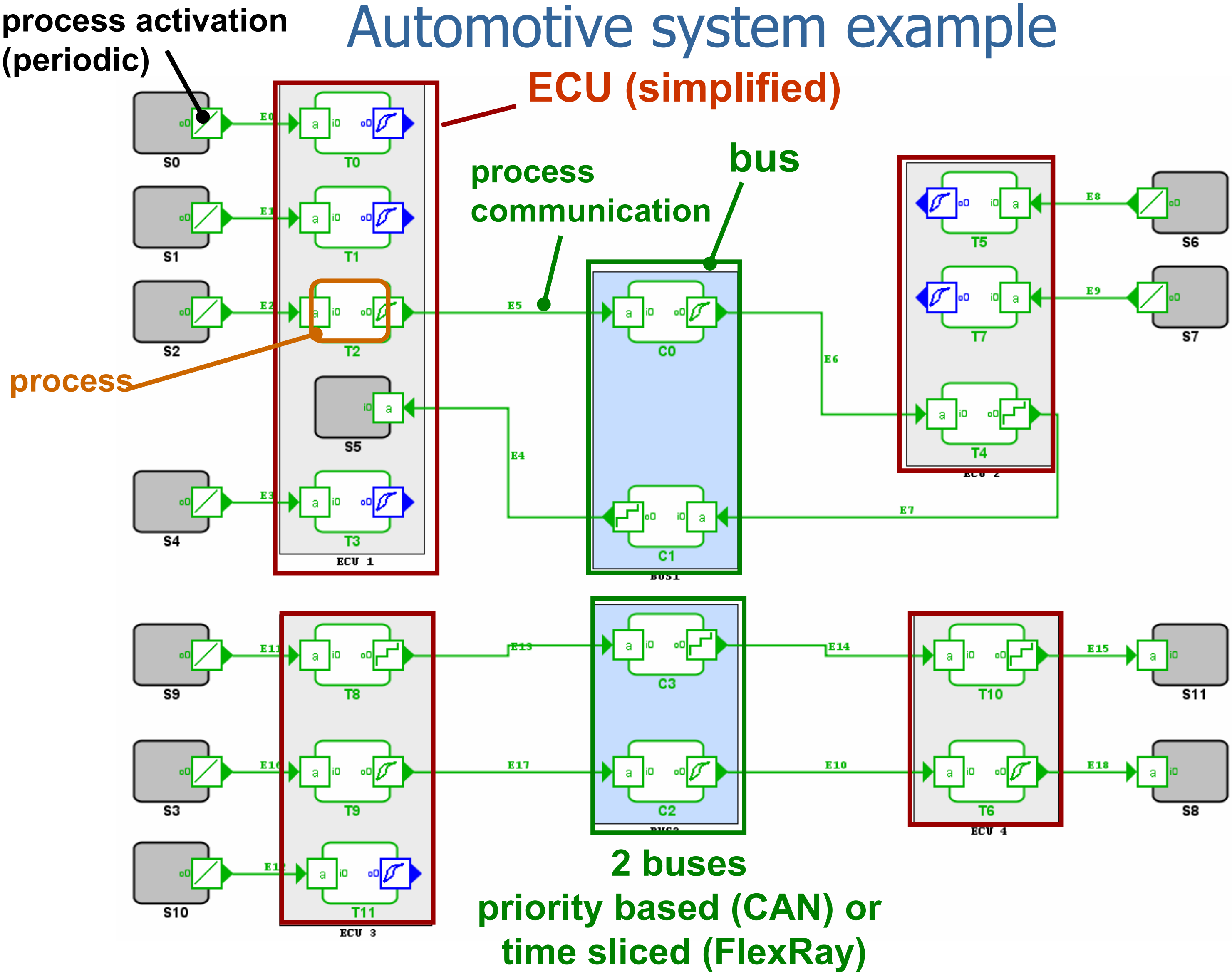
- in principle, all design data can be subject to design robustness consideration - complex issue
- here we assume
  - fixed architecture
  - fixed mapping of functions to components
- **modification** of performance related SW and HW component properties
  - platform component performance (processor and communication links)
  - execution times of individual processes
  - process communication volumes
- considered **constraints**
  - focus on **real-time systems**
    – consider worst case behavior (rather than e.g. average)
  - max. response times
  - end-to-end deadlines

Communication centric design as integration problem



- complex dependencies as a integration result
- major robustness issue

Automotive system example



Automotive example - explanations

- electronic control unit functions are jointly specified by OEM and supplier
- buses (or bus networks with gateways) are used for systems integration
  - design parameter: bus priorities, time slice, cycle time
- design scenario 1:
  - parameters are defined and fixed early at design time
  - not modified later to reach compatibility for variants and later updates
  - state of the practice
- design scenario 2:
  - update parameters during product life cycle (e.g. new version of an existing car type) or in the field

Robustness metrics

- robustness metrics shall be based on the “slack” of a system property
- **def. 1: Slack**
  - given

    a constrained system S

    a parameter configuration c

    a system property p $\in$ S
  - we define

$$slack_{p;c} = \frac{\left| v_c^+(p) - v(p) \right|}{v(p)} * 100$$

were v(p) is the current value of p and $v_c^+(p)$ is the maximum property value for p not leading to constraint violations

Robustness metrics – static design robustness

- **def. 2: Static Design Robustness (SDR)**
  - given
    - *a constrained system S*
    - *a parameter configuration c*
    - *a set of system properties* $P = \{p_1, \ldots, p_n\}$
    - *a set of (user defined) weights* $W = \{w_1, \ldots, w_n\}$
  - we define SDR as the weighted set of slacks

$$SDR_{P;c} = \frac{\sum_{i=1}^{n} w_i * slack_{p;c}}{\sum_{i=1}^{n} w_i}$$

  - SDR is relevant to design scenario 1 and measures the overall slack in case one of the considered properties is modified later
  - alternative: geometric mean value

Robustness metrics – static and dynamic

- **SDR** is relevant to design scenario 1 and measures the overall slack in case one of the considered properties is modified later
- to anticipate and include *potential* parameter adaptations in later design phases or in the field, we *need a metric that includes potential designer or system counteractions in case a system property is modified later*
- *for that purpose, we must*
  - *identify such potential counteractions*
  - *include their effect in the metric*
- *potential counteractions can e.g. be found by system optimization assuming modified system properties*

Robustness metrics – dynamic design robustness

- **def. 2: Dynamic Design Robustness (DDR)**
  - given

    a constrained system S

    a system property p

    a set of potential parameter configurations $C = \{c_1, \ldots, C_m\}$

    The slack vector $V = \{slack_{p;c1}, \ldots slack_{p;cn}\}$
  - we define DDR as the slack of the configuration that allows the maximum p modification

$$DDR_{p;C} = \max_{c \in C}(slack_{p;c})$$

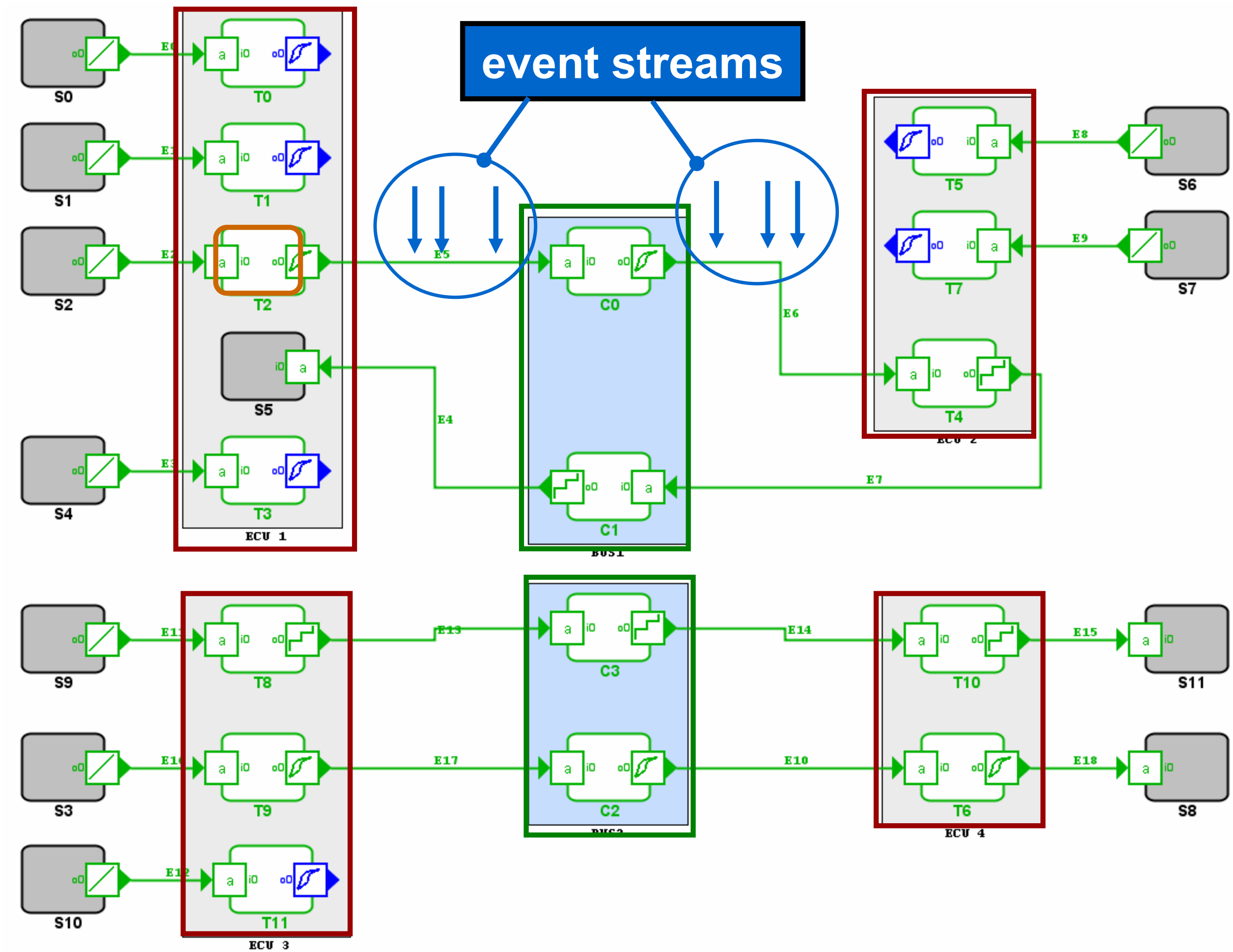
  - DDR is not unique but depends on the set of available configurations ("counteractions") in C

    DDR is maximal if C contains c with maximum possible $slack_{p;c}$

Models for robustness metrics

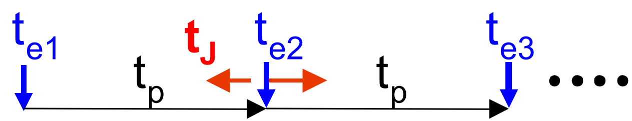
- simulation
  - not possible because property changes are not supported in general (if code available at all)
- simple models capturing average loads of processors or communication links
  - often used in architecture design
  - do not consider scheduling influences – not appropriate
- event and response time models of schedulability analysis
  - suitable

Automotive example event streams

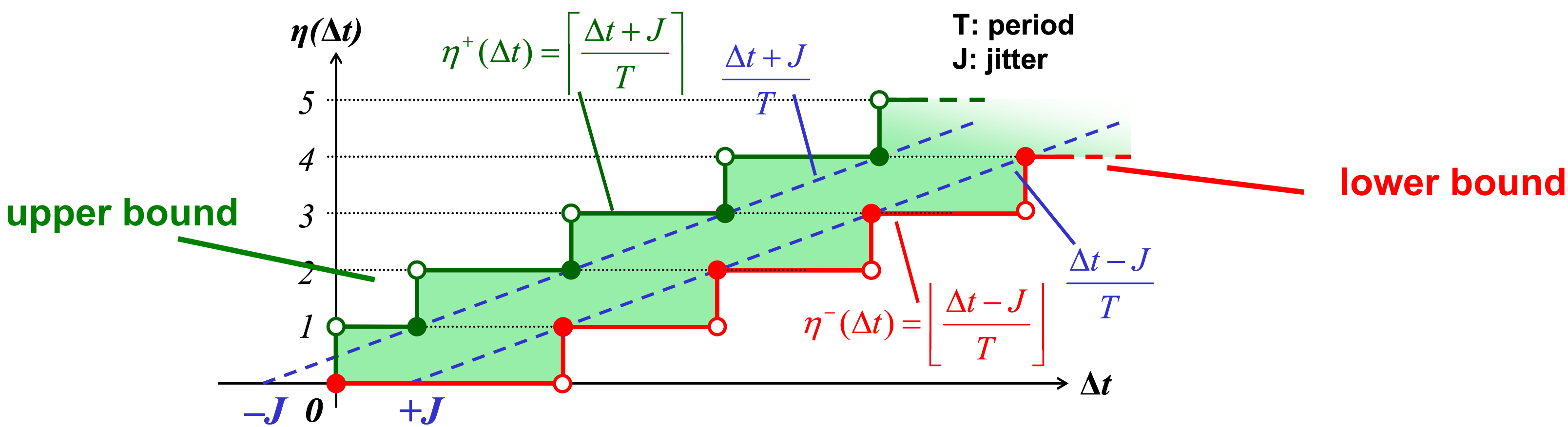


Event models

- event stream model w. parameters
  - individual events replaced by stream variables with parameters period, jitter, min. distance, …

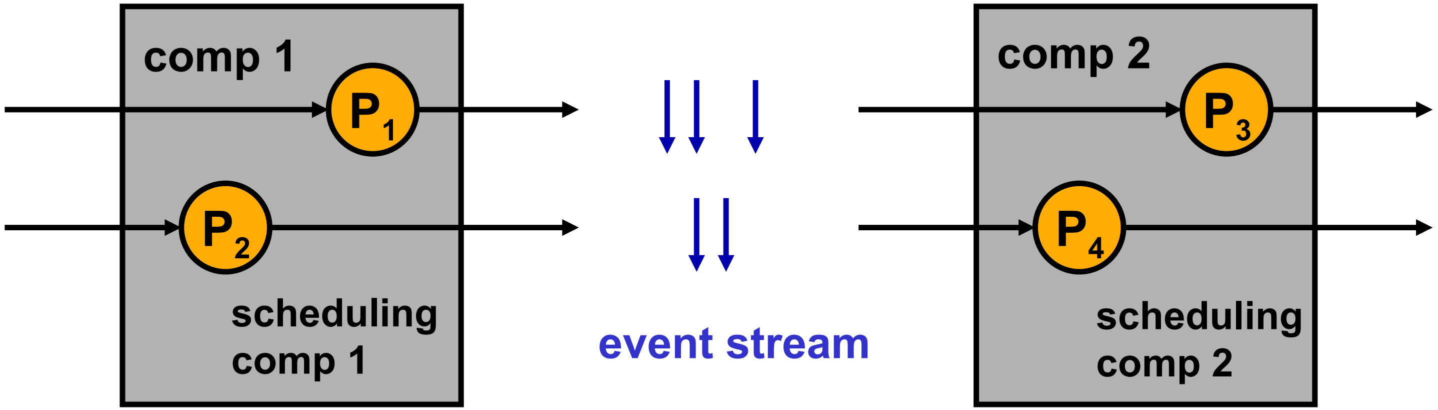


- Network Calculus
  - individual events replaced by sum of events in sliding time window Δt



System analysis using compositional approach

❖ independently scheduled subsystems are coupled by data flow

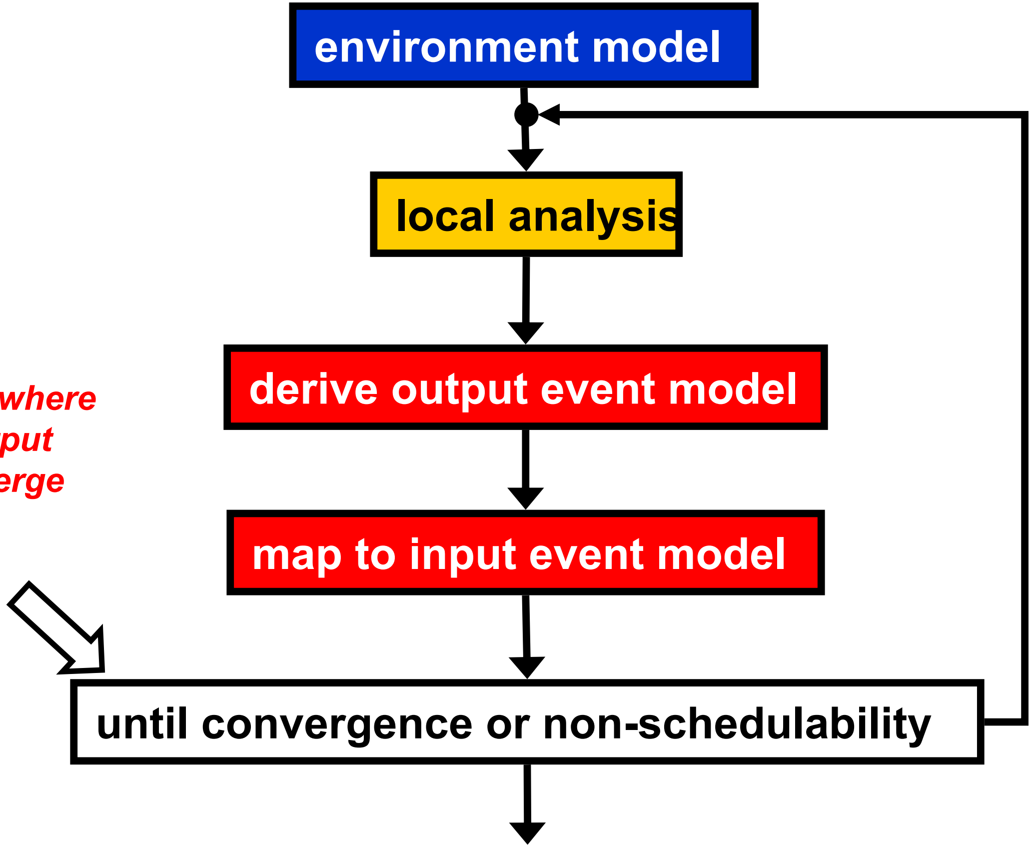


⇒ subsystems coupled by stream of data

⇒ interpreted as activating events

⇒ coupling corresponds to event propagation

Compositional analysis principle



find fix point where input and output models converge

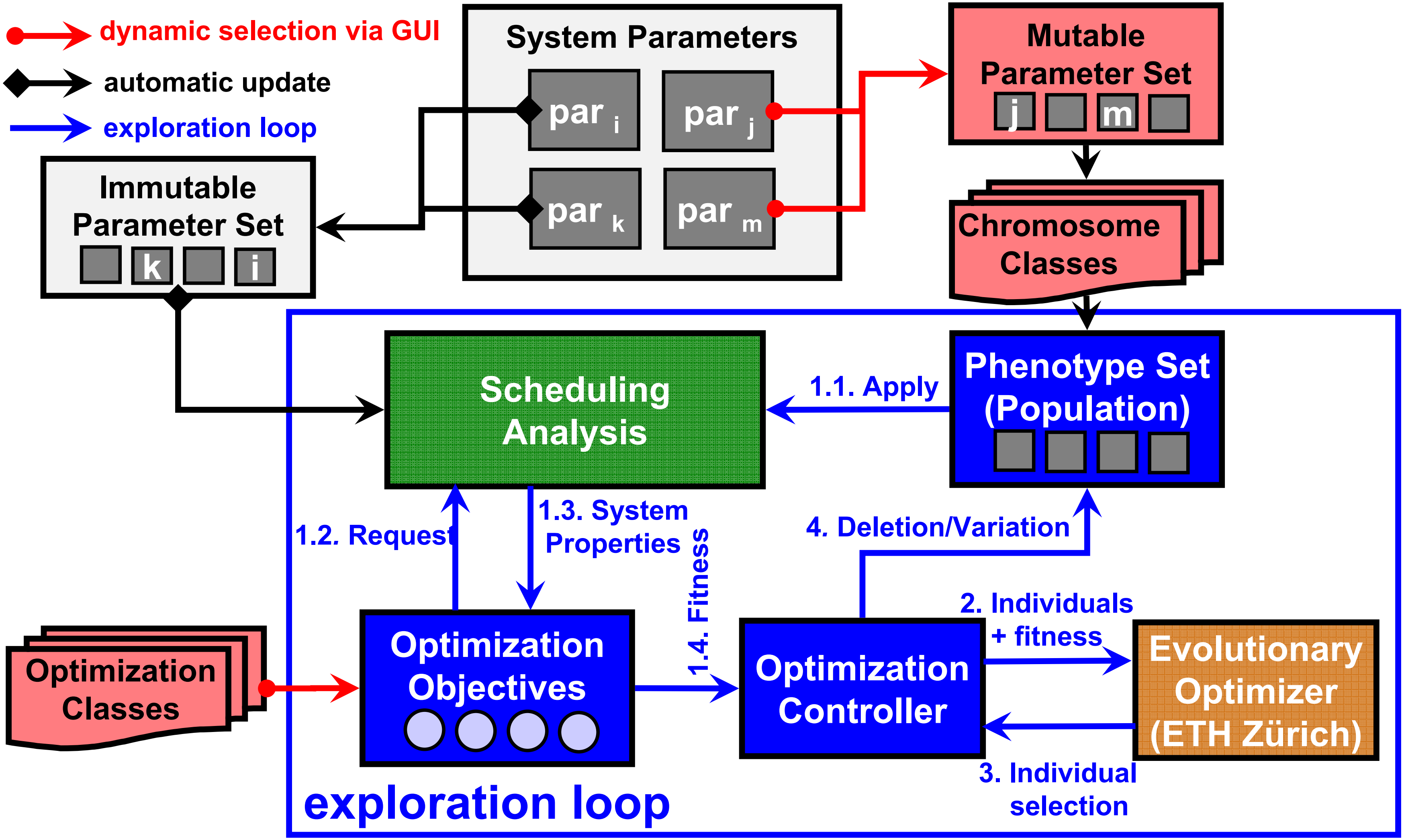
- **flexible and modular !**

Application to robustness analysis and optimization

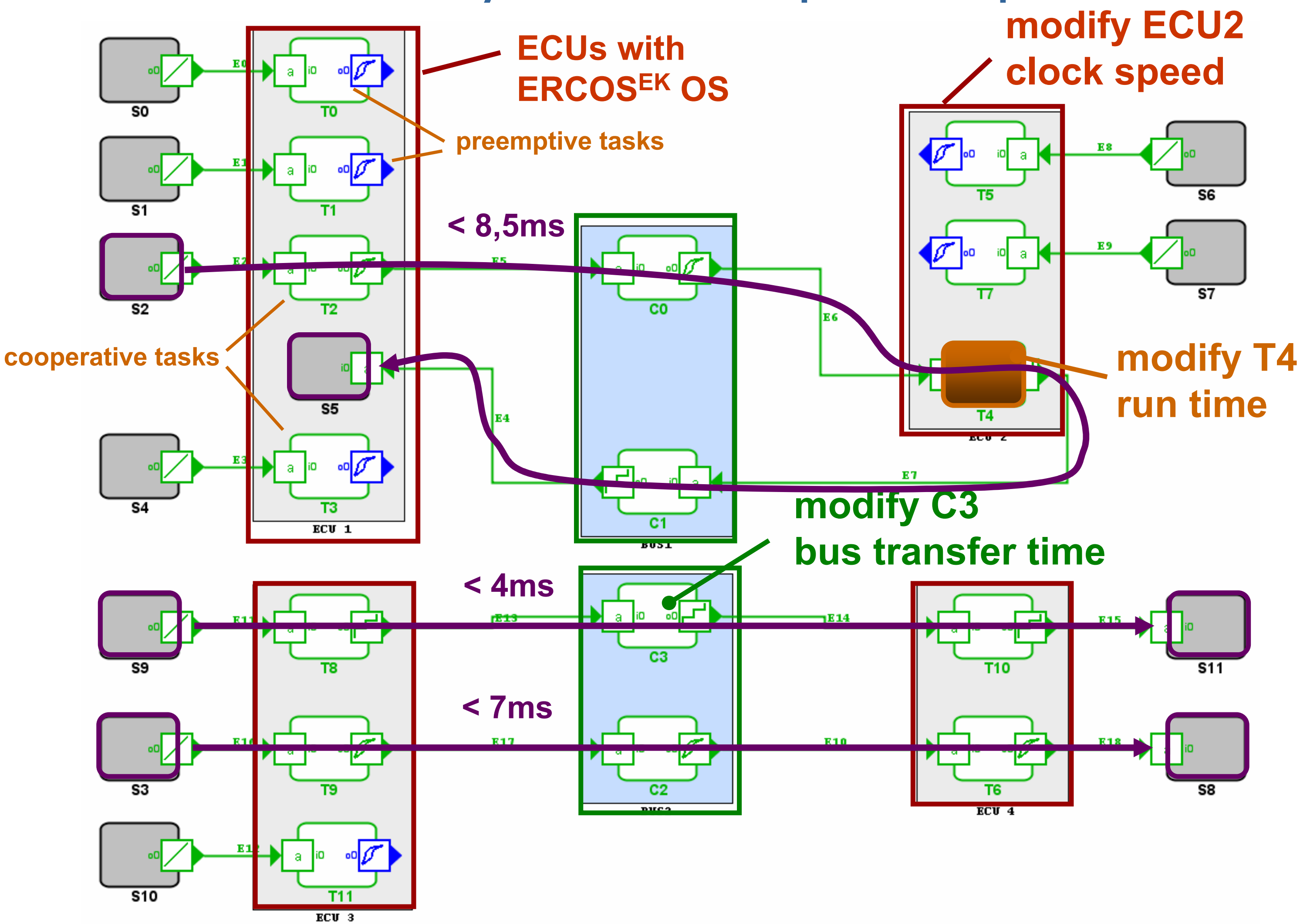
- sensitivity analysis
  - binary search to determine slack and SDR
- automated design space exploration
  - uses evolutionary optimizer
  - to maximize SDR
  - to generate a „good“ configuration set C for DDR determination

    pareto optimization approximates maximum DDR

Design space exploration framework



Automotive system example - experiments

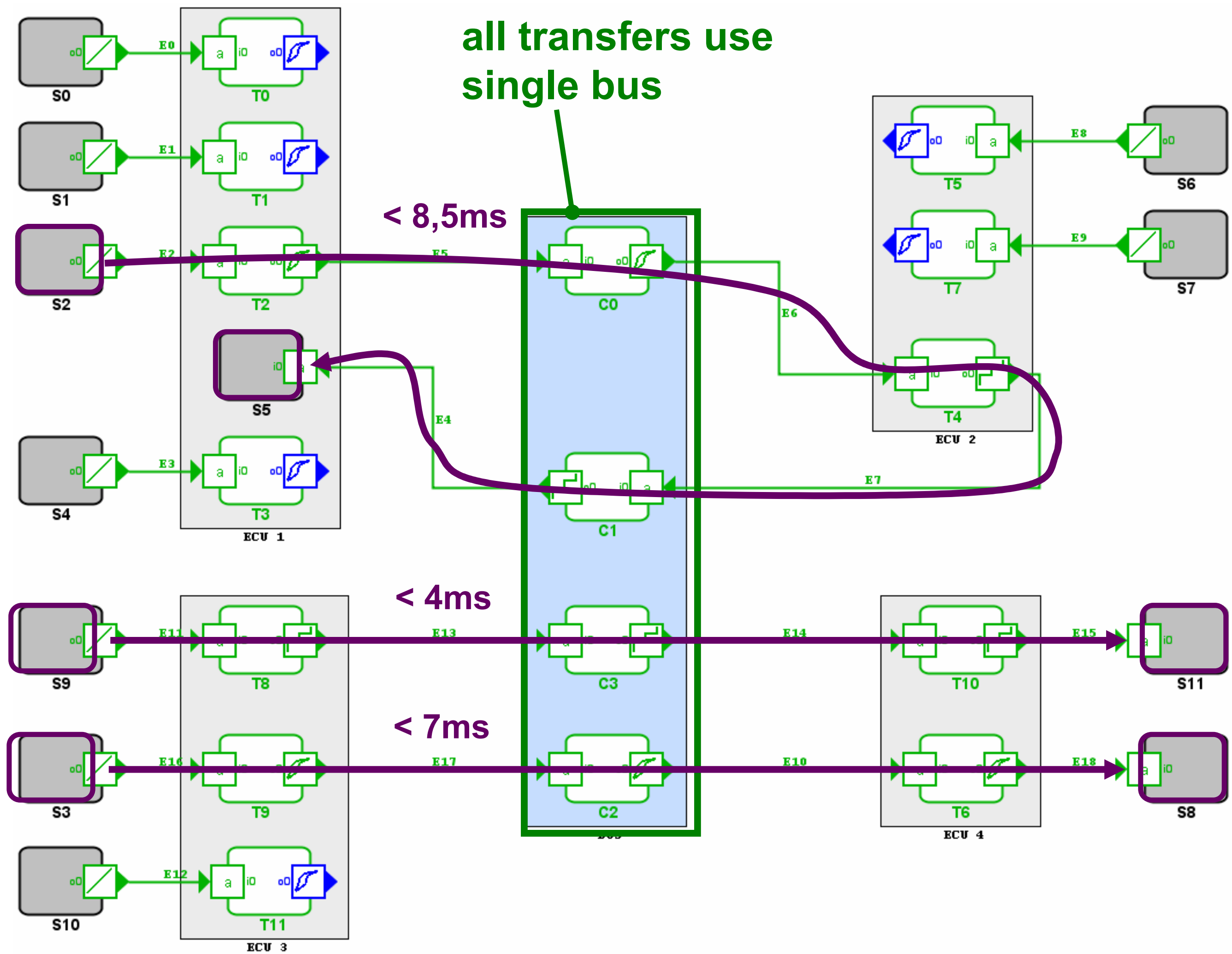


Exp. 1: Design robustness for bus w. priorities

	WCET T4 (slack)	WCET C3 (slack)	Speed ECU2 (slack)	SDR Metric
Original Configuration (Pareto-optimal with respect to timing)	28.75%	3000%	12%	1013.58
Optimized for SDR (all $w_i = 1$)	62.5%	5900%	28%	1996.83
DDR	86.25%	5900%	35%	n.a.

- significantly higher robustness when parameters are optimized for maximum SDR rather then just for minimum response time
- bus and ECU load identical in each column

Example system with single bus



Design robustness – single bus time triggered

	WCET T4 (slack)	WCET C3 (slack)	Speed ECU2 (slack)	SDR Metric (wi = 1)
Original Configuration (Pareto-optimal with respect to timing)	27,5%	750%	12%	200,875
Optimized for SDR – bus w. priorities (wi = 1)	50%	4900%	18%	1247,25
Optimized for SDR bus time triggered	30%	1400%	12%	593
DDR – bus w. priorities	81,25%	4900%	29%	N.A.

❖ higher robustness of SDR optimized system remains under higher load, dynamic configuration efficiency is increasing

Conclusion

- formal methods for communication centric embedded system optimization
- introduced metrics to quantify and optimize embedded system robustness
- distinguish two design scenarios with different flexibility to change system parameters in later design phases
- first experiments at an automotive example show that optimization for robustness can be effective

Further reading

- www.symta.org
- www.symtavision.com
- www.mpa.ethz.ch