End-to-End Design and Analysis of Embedded Real-Time Systems

Faculty: Kang G. Shin

Grad students: Sam Gu, Sharath Kodase

Real-Time Computing Laboratory

EECS Department

The University of Michigan

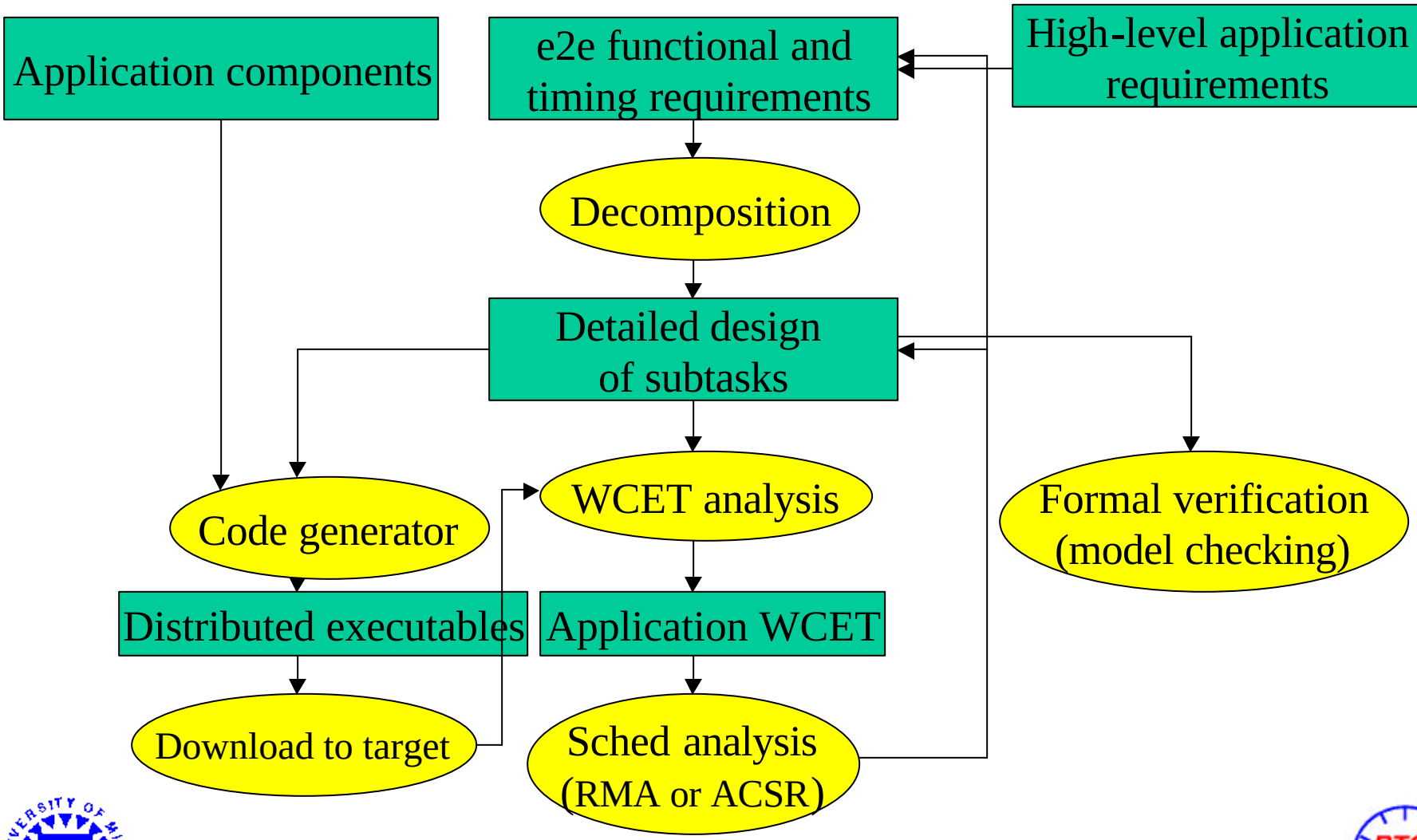
Ann Arbor, MI 48109-2122

http://www.eecs.umich.edu/~kgshin

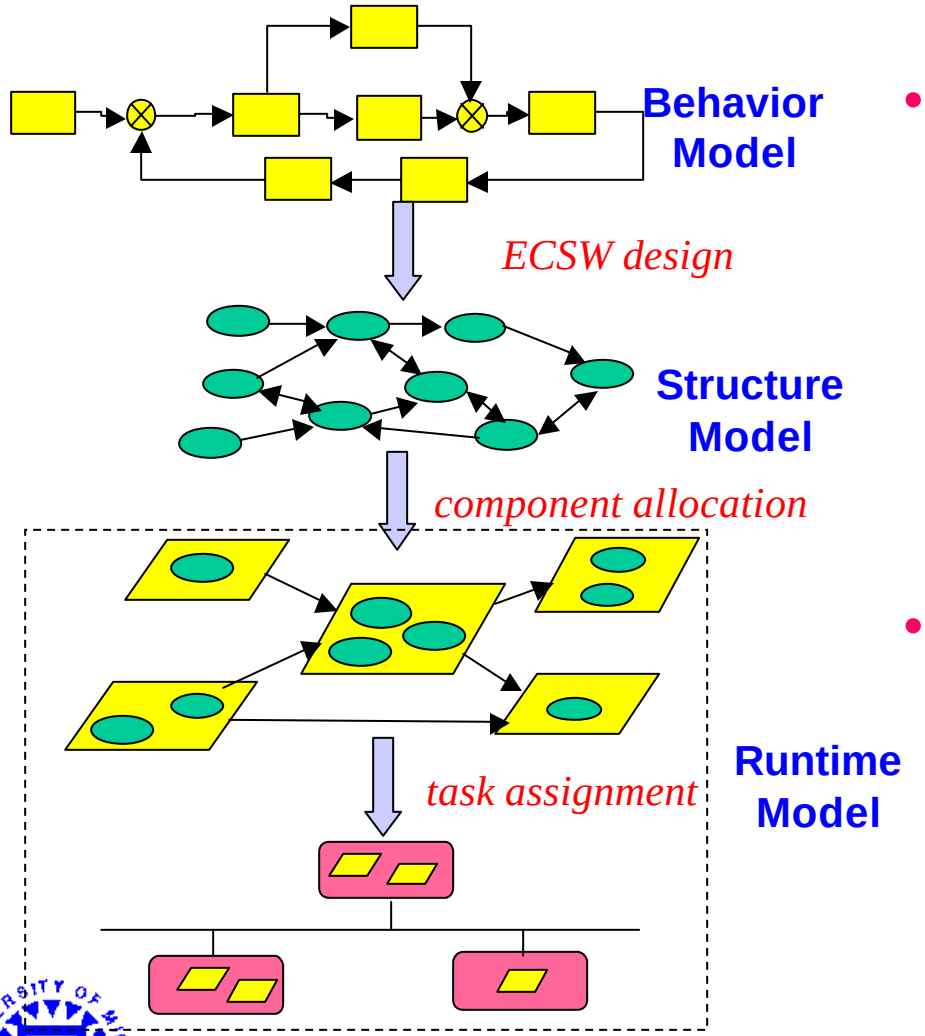
Objectives

- Enable the designer to express system behavior and e2e RT constraints at a higher level, closer to domain knowledge and further away from implementation details.
- Automate the process of mapping from application structure models to runtime models subject to high-level e2e timing constraints.
- Provide formal semantics for e2e extensions to enable effective formal analysis.

Proposed Design Flow

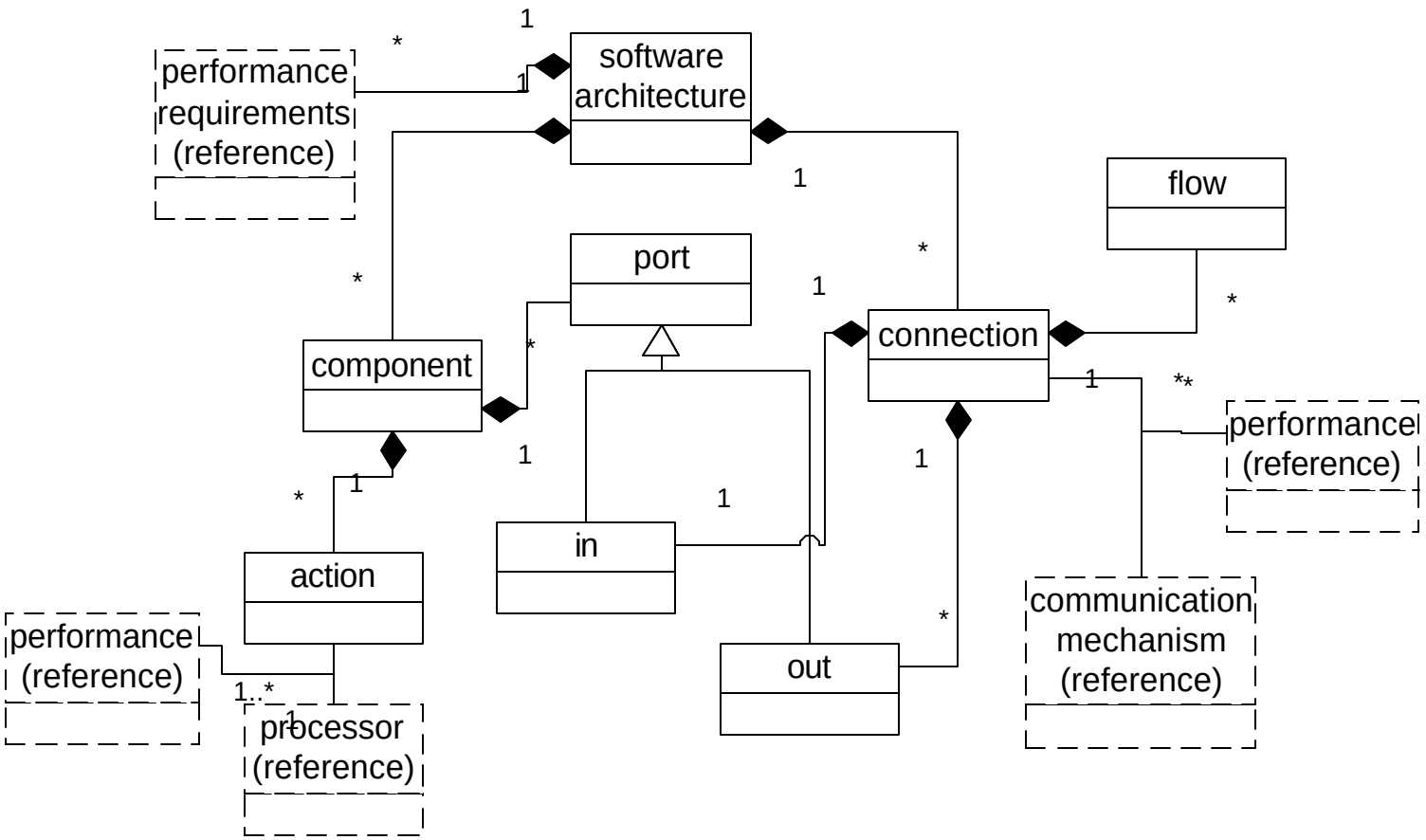


Model Transformation and Integration

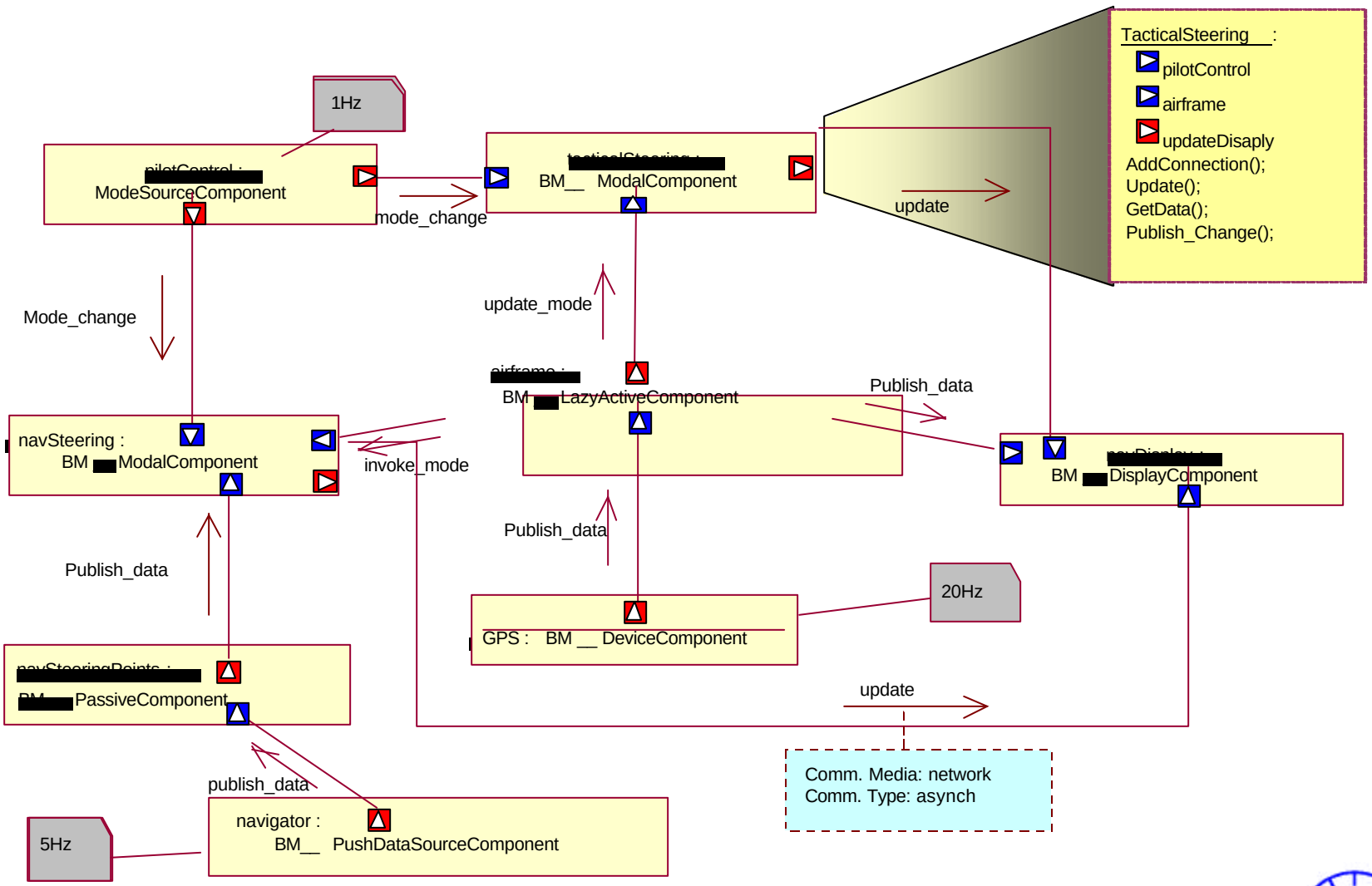


- Functional design model includes
  - Behavior model: control specifications
  - Structure model: components/subsystems
  - Runtime model: task graph
- Non-functional issues should be considered during transformation, especially for runtime model

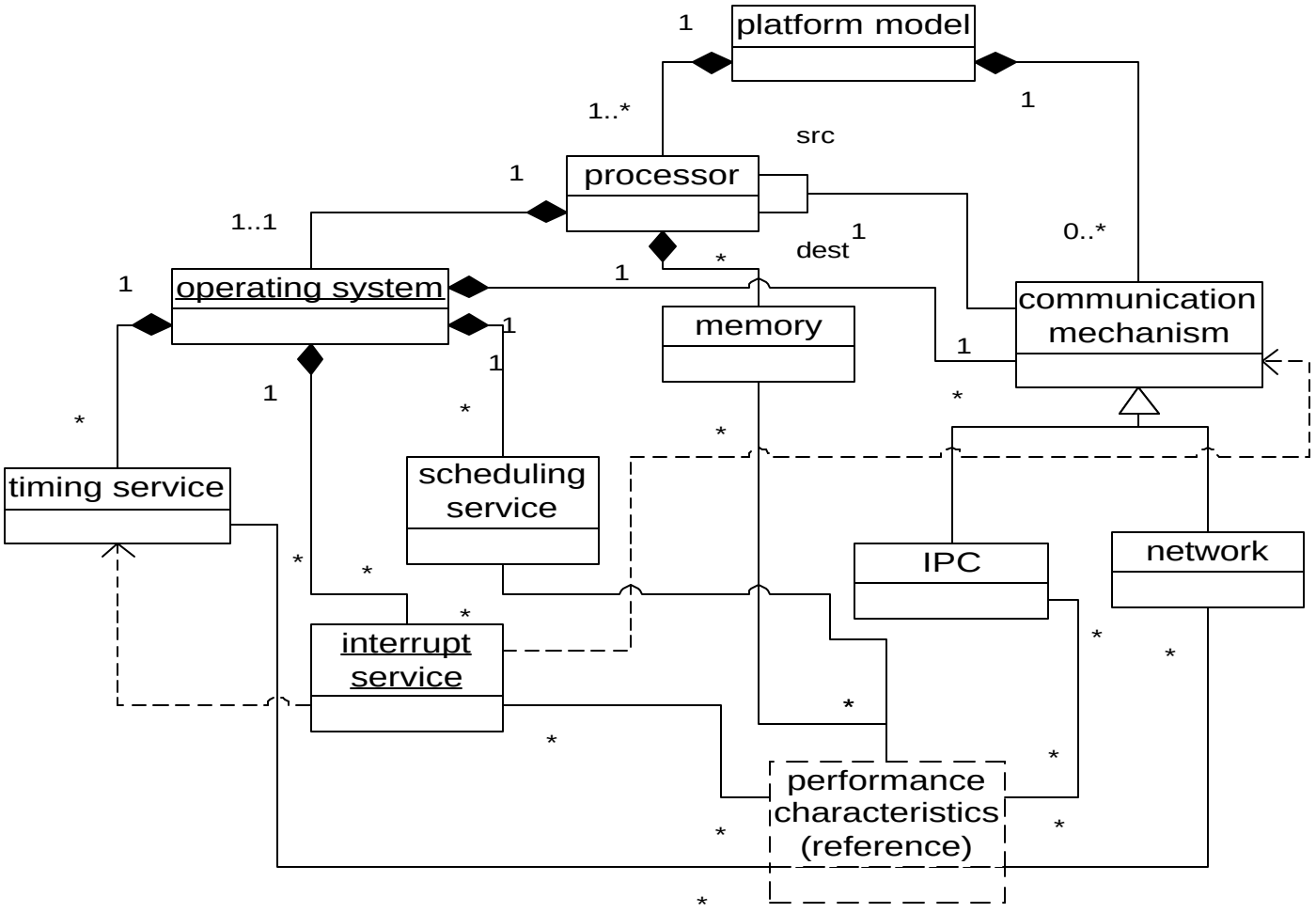
Software Structure Meta-Model



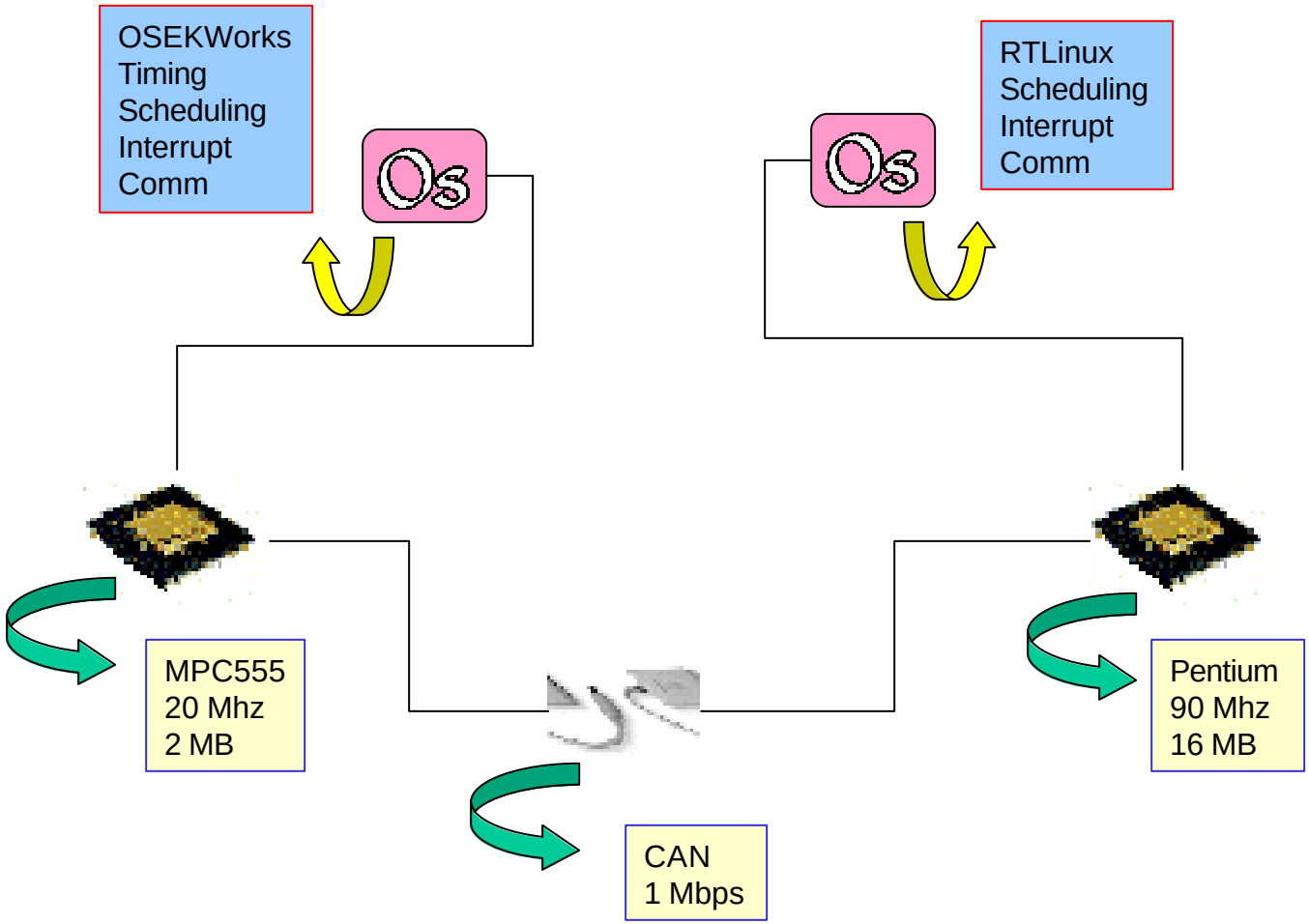
Structure Model Example



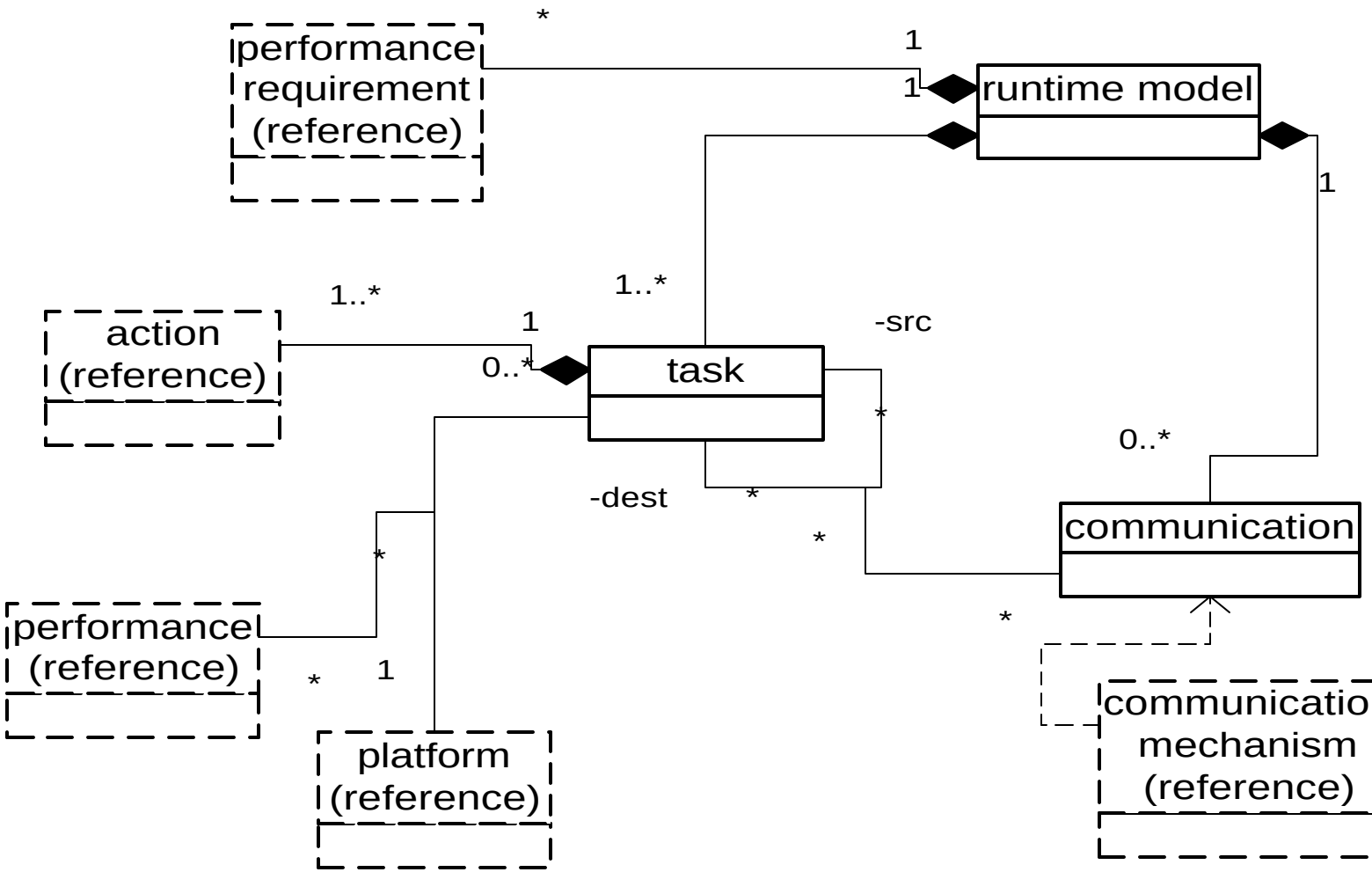
Platform Meta-Model



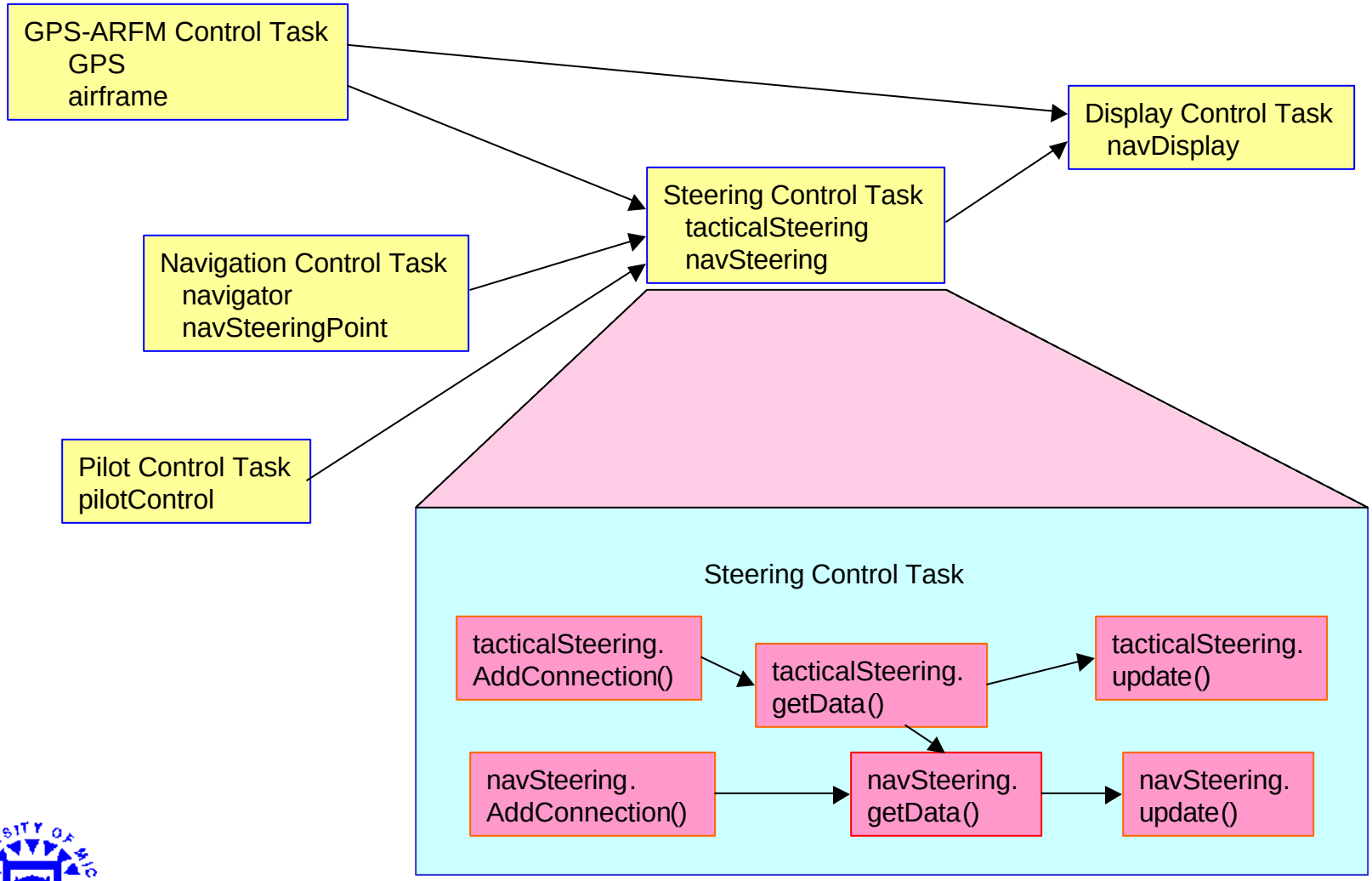
Platform Model Example



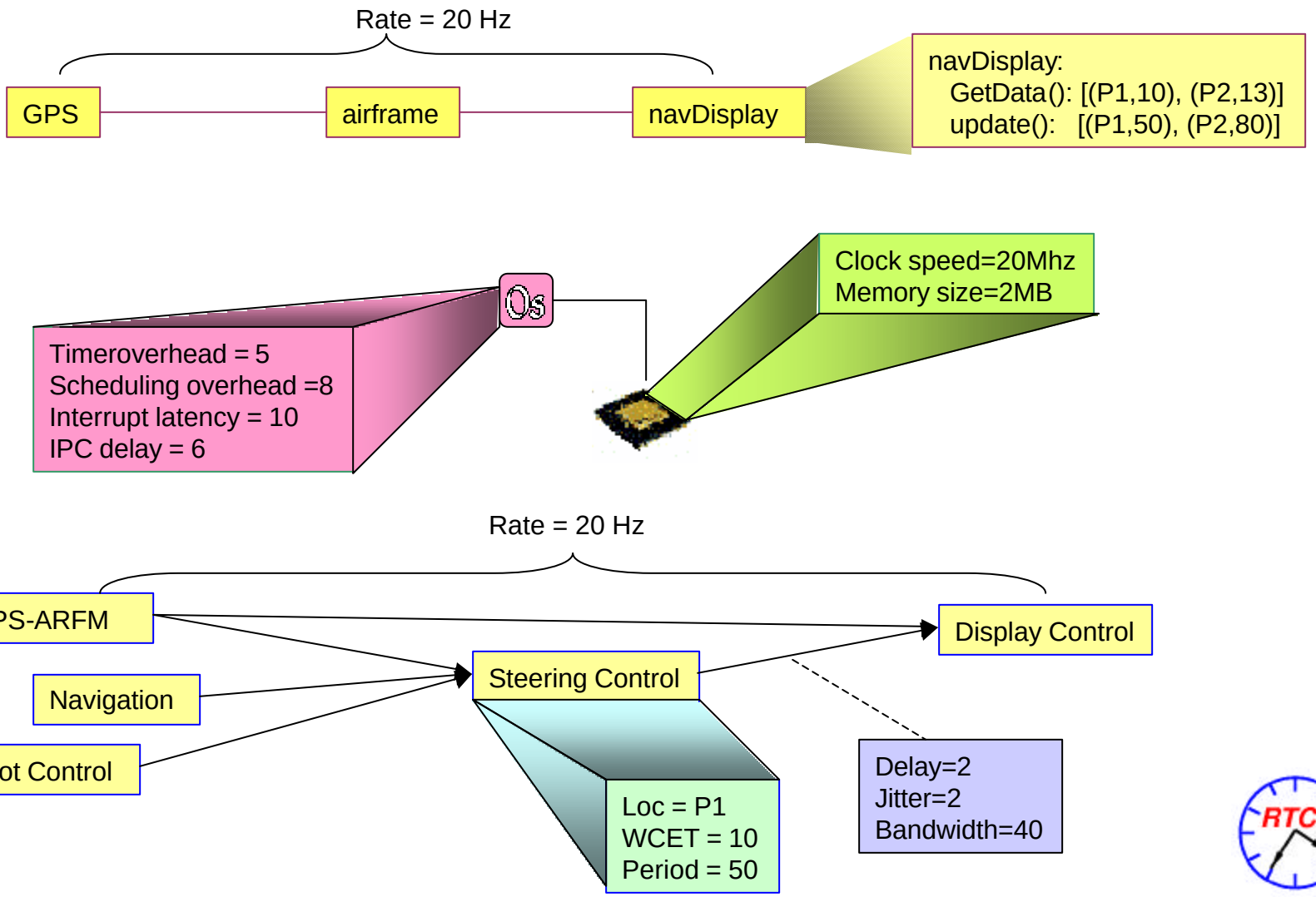
Runtime Meta-Model



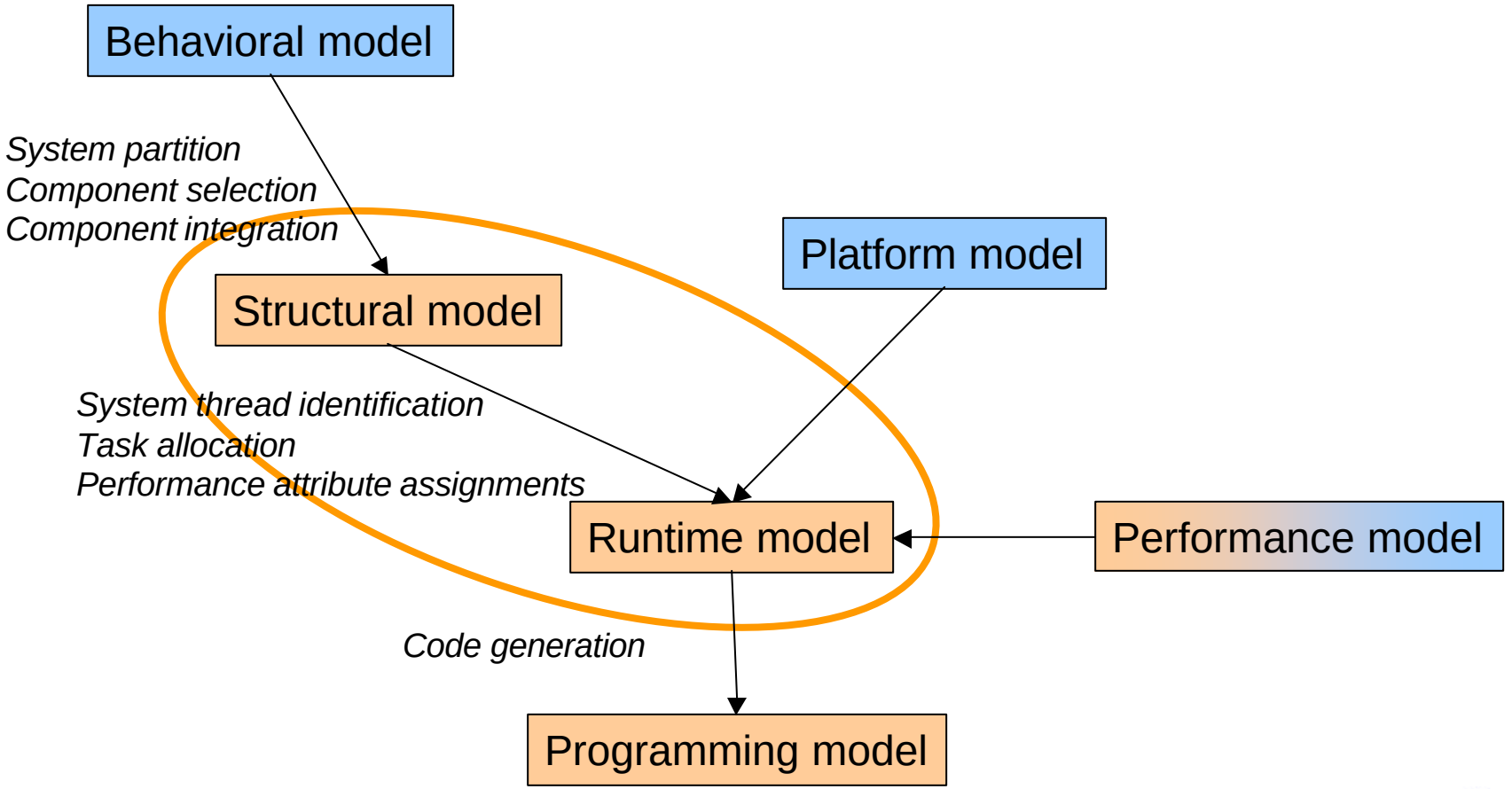
Runtime Model Example



Runtime Model with Timing Annotations



Transformation Between Models



Transformation Algorithms

- 2-step process
  - Task construction
  - Timing assignments
- Break dependencies by considering function only in the first step, and then performance in the second
- Design involves multiple iterations of 2-step process

Task Construction

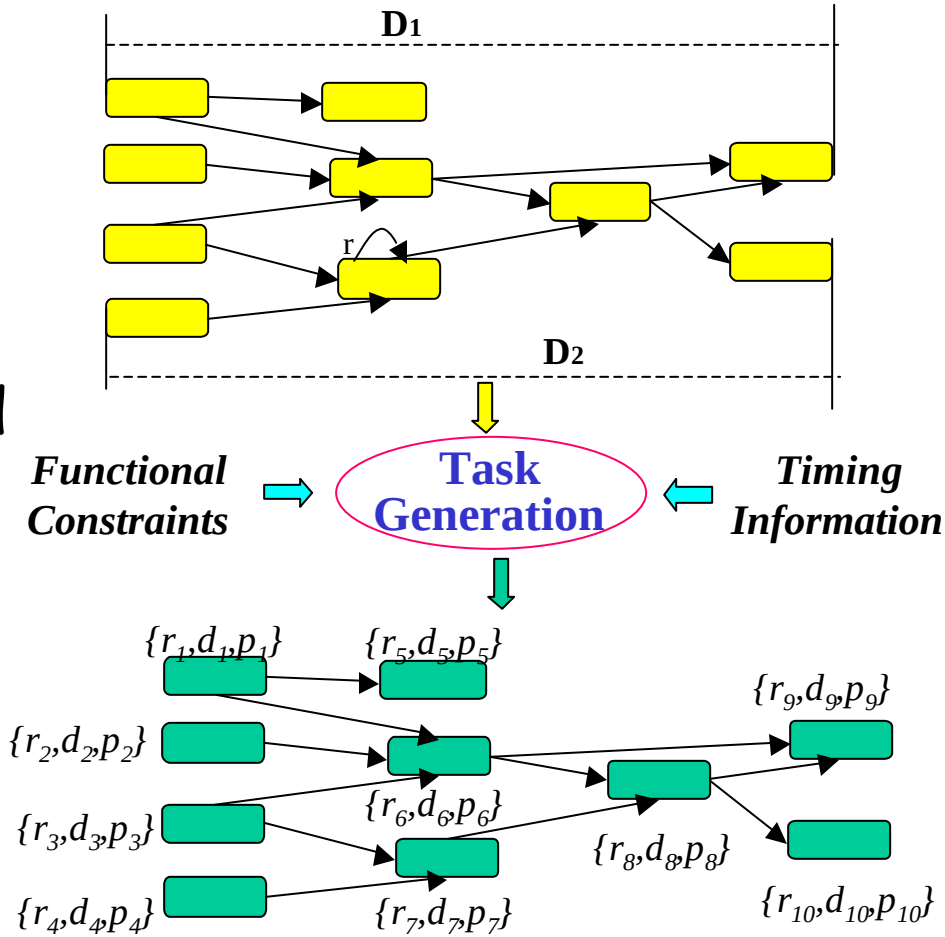
- Input: structural and platform models
- Process:
  - Find e2e transactions (a.k.a. execution path)
  - Allocate actions in transactions to platform
    - Maximize utilization while preserving schedulability
    - Refine later with communication cost
  - Group actions on the same processor to form tasks
    - Actions in the same transaction should be in one task
    - Actions with the same priority should be in one task
  - Allocate shared components in the faster task
  - Construct task graph
    - Derive dependencies according to structural model
    - Assign timing constraints to e2e tasks

Timing Assignment

- Input: Task graph with e2e timing constraints
- Process
  - Compute task WCET=Σe_{action}
  - Find critical execution path P in task graph
  - Distribute e2e deadline over P
  - Break task dependencies by adding shared buffer
  - Combine tasks with the same rate on the same processor
  - Verify the satisfaction of timing constraints
  - Refine assignment by shortening the period of task(s) on P until
    - All constraints are satisfied
    - Task set is found unschedulable (need more resource)

Timing Specification and Assignment

- Requirements are usually given in an e2e form or a rate for each component
- End-to-end constraints should be partitioned and assigned to each activity
- Schedules have to consider these timing spec as requirements



Deadline Distribution

- Objective
  - Partition constraints at higher-level for timing assignments and scheduling
- Deadline distribution supports hierarchical partitioning of constraints

Inputs

· A task graph with WCET

· Timing constraints:

· e2e constraints: given a sensor signal change X, the new command for actuator Y has to be outputted within t time units

· Rate constraints: task T has to be executed at a particular rate R to satisfy the requirements of component C inside it

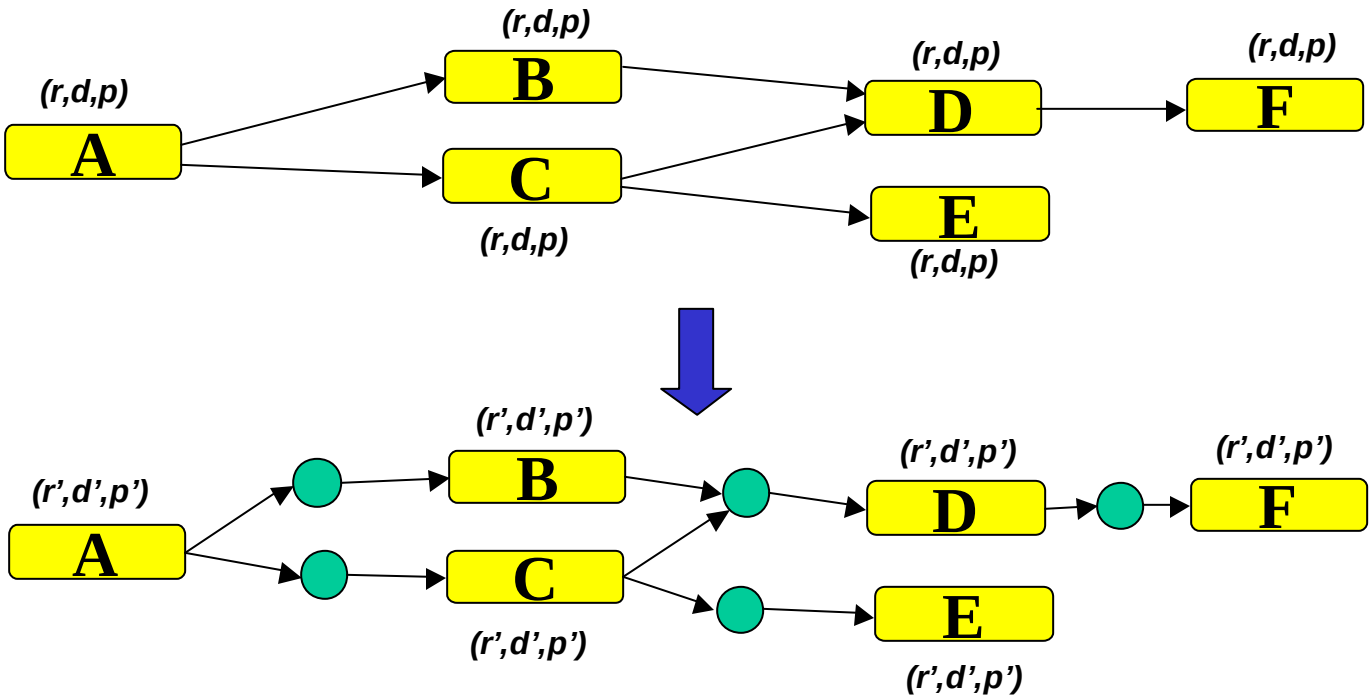
Outputs

· Deadlines and release times for all intermediate subtasks

Deadline, release time, rate, and WCET are sufficient for any scheduling algorithm to generate a schedule

Task Dependency Resolution

- Task dependencies should be broken to support scalable scheduling and allocation algorithms
- Shared buffers are used to break dependencies
- After introducing shared buffers, rates need to be regenerated
- Tasks are clustered to reduce resource consumption



Real-Time Analysis

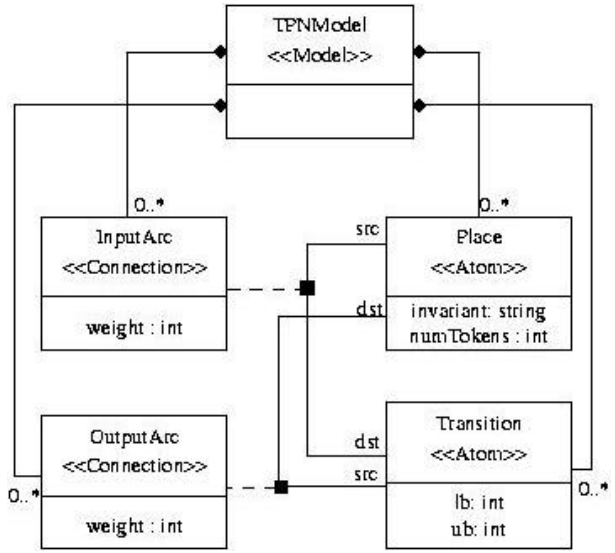
- Schedulability Analysis
  - Commonly used scheduling policies: RMA, EDF, DMA, etc.
  - Processor utilization
  - Resource consumption by
    - Application tasks
    - System software (OS and middleware)
    - Communication messages
- Two approaches:
  - Generalized Rate Monotonic Analysis
  - ACSR/VERSA

Formal Analysis

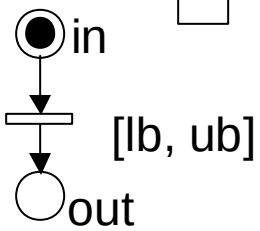
- Map event-triggered software model in UML Interaction Diagrams to Timed Petri-Nets.
- Syntax-directed automated mapping from TPN to Timed Automata, implemented in GME via mapping between meta-model elements.
- Use an existing model-checker UPPAAL to check for system property violations.
- Map counter-examples back into UML environment.

Automated Transformation

TPN Meta-Model



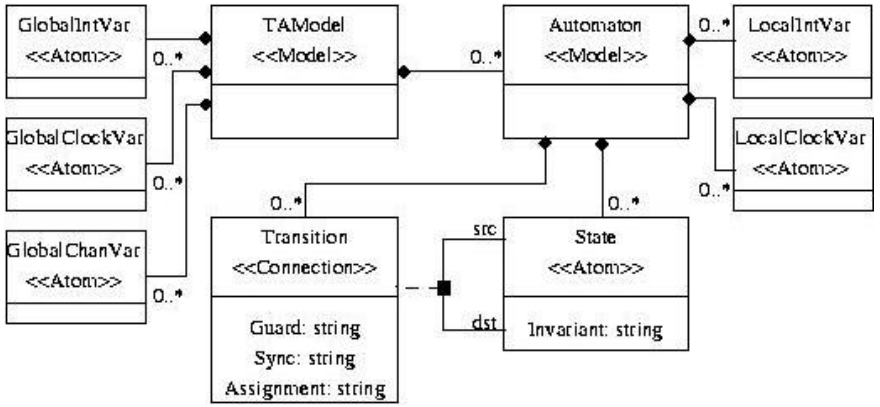
Conforms To



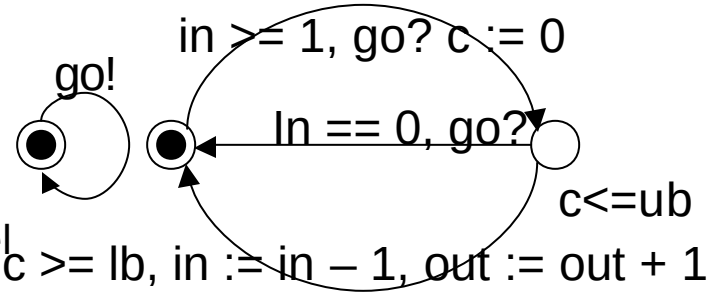
TPN Model

Transformation via mapping of meta-model elements

TA Meta-Model

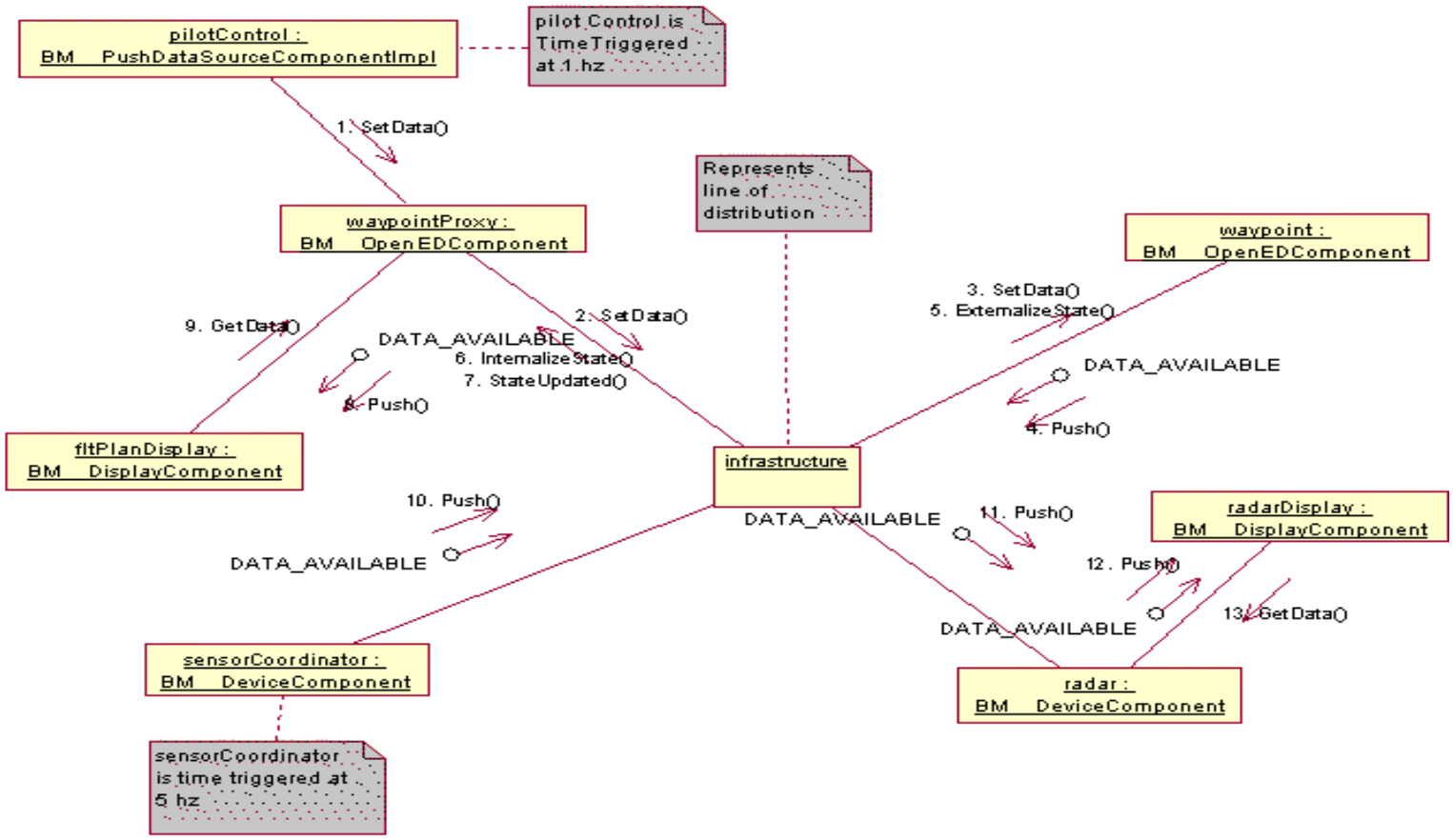


Conforms To

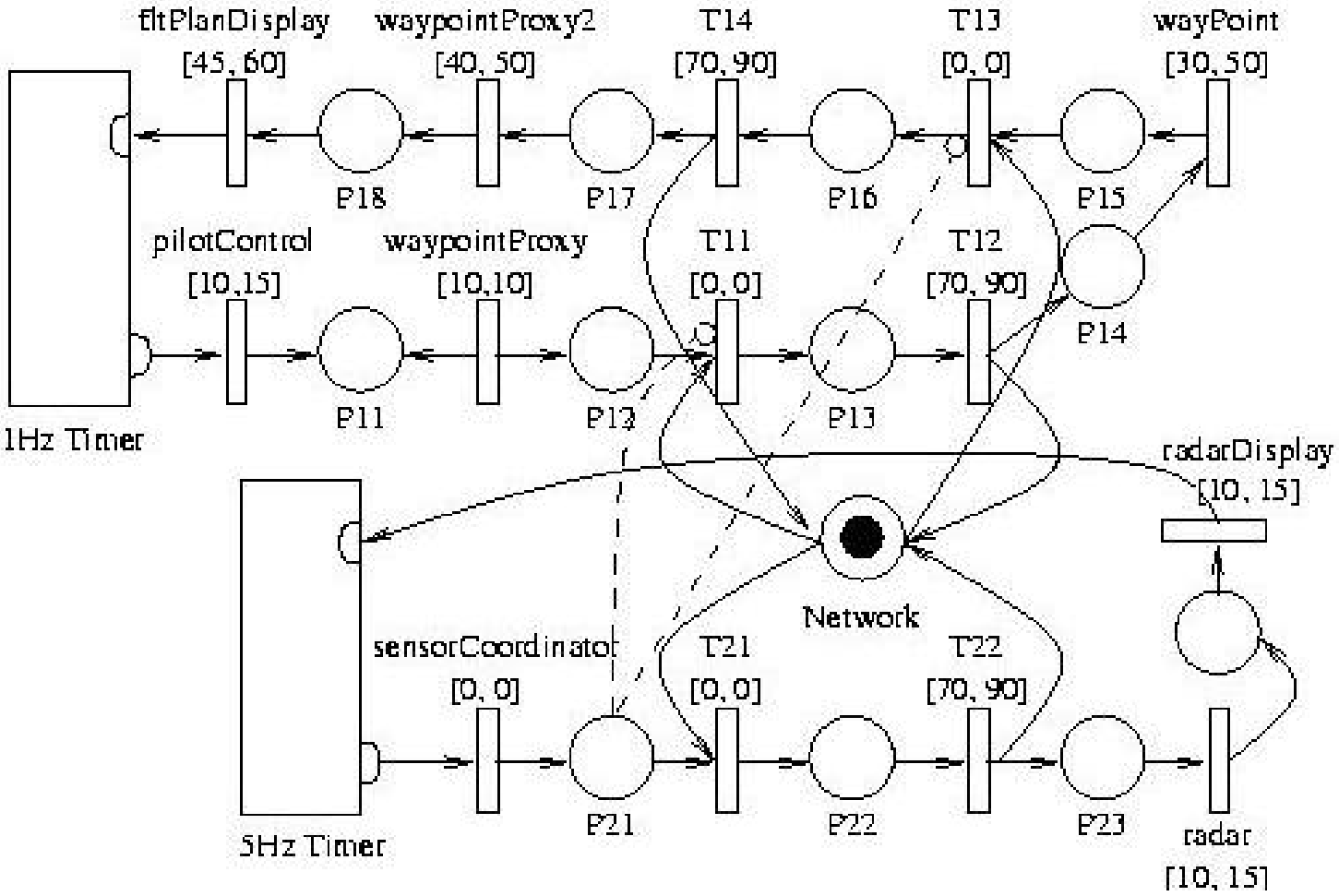


TA Model

An Avionics Scenario in UML

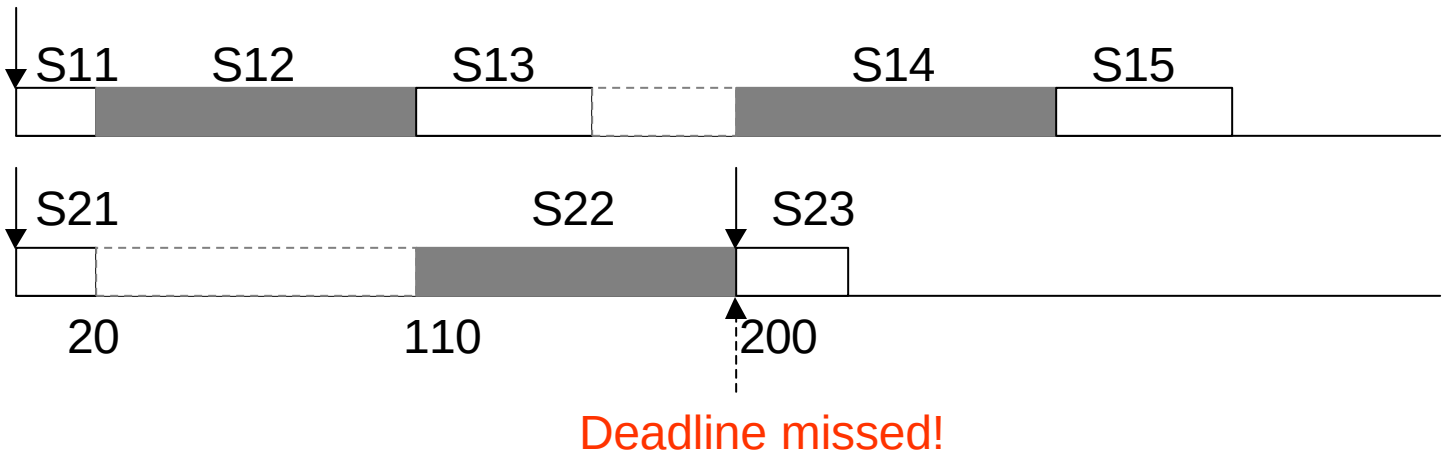


TPN Model of EDG Scenario



Model Checking Results (Sample)

- Transform TPN to TA, then use UPPAAL for model-checking.
- End-to-end delay range of 1Hz thread is [275, 525] ms.
- 5Hz thread with deadline=200 ms has frame-overrun.
- UPPAAL can give a diagnostic trace for the execution scenario that leads to frame overrun.



Conclusions

- ESW development is a multi-phase multi-iteration process, and requires integration of tools based on heterogeneous models
  - Require a common modeling framework
  - Require information loop
- The proposed framework supports semi-automated model transformation
  - Demonstrated by translating structural model to runtime model while meeting e2e timing constraints
- Formal verification by automated transformation from UML models to TPN and TA.

Publications

- An Integrated Approach to Modeling and Analysis of Embedded Real-Time Systems Based on Timed Petri-Nets. ICDCS 2003.
- Analysis of Event-Driven Real-Time Systems with Time Petri-Nets. DIPES 2002.
- Integrated Modeling and Analysis of Computer-Based Embedded Control Systems. ECBS 2003.
- Improving Scalability of Task Allocation and Scheduling in Large Distributed Real-Time Systems using Shared Buffers. IEEE RTAS 2003.
- Transforming Structural Model to Runtime Model of Embedded Software with Real-Time Constraints. DATE 2003.
- Automating embedded software construction and analysis with design models. Euro-uRapide 2003

END

- Questions?

Timing Assignment

- Input:
  - Dependent task graph with timing constraints
  - Platform
- Process
  - Compute task WCET=Σe_{action}
  - Find critical execution path P in task graph
  - Distribute end-to-end deadline over P
  - Break task dependencies by adding shared buffers
  - Combine tasks with the same rate on the same processor
  - Verify the satisfaction of timing constraints
  - Refine assignment by shortening the period of task(s) on P until
    - All constraints are satisfied
    - Task set are unschedulable (need more resource)