



McGill

ECSE 421 Lecture 13: Performance Evaluation

ESD Chapter 5

© Peter Marwedel, Brett H. Meyer

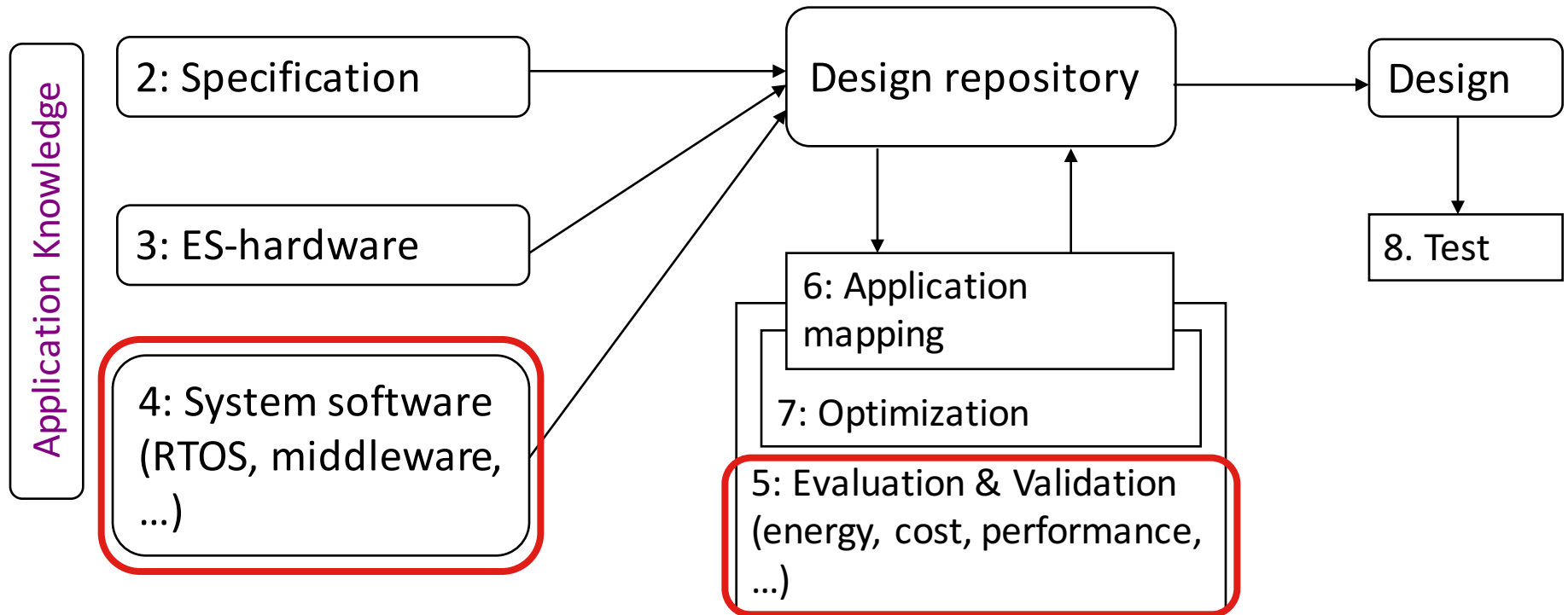
Last Time

- Embedded Operating Systems

Where Are We?

4	T	2-Feb-2016	L06	Discrete Event Models	2.7	4			G: Zaid Al-bayati
	R	4-Feb-2016	L07	DES / Von Neumann Model of Computation	2.8-2.10	5	LA2	LA1	
5	T	9-Feb-2016	L08	Sensors	3.1-3.2	7.3,12.1-6			
	R	11-Feb-2016	L09	Processing Elements	3.3	12.6-12			
6	T	16-Feb-2016		No class				LA2	
	R	18-Feb-2016	L10	More Processing Elements / FPGAs			LA3		
7	T	23-Feb-2016	L11	Memories, Communication, Output	3.4-3.6				
	R	25-Feb-2016		Midterm exam: in-class, closed book			P		Chapters 1-3
	T	1-Mar-2016		No class					Winter break
	R	3-Mar-2016		No class					Winter break
8	T	8-Mar-2016	L12	Embedded Operating Systems	4.1			LA3	
	R	10-Mar-2016	L13	Performance Evaluation	5.1-5.2				
9	T	15-Mar-2016	L14	More Evaluation and Validation	5.3-5.8				
	R	17-Mar-2016	L15	Introduction to Scheduling	6.1-6.2.2				
10	T	22-Mar-2016	L16	Scheduling Aperiodic Tasks	6.2.3-6.2.4				
	R	24-Mar-2016	L17	Scheduling Periodic Tasks	6.2.5-6.2.6				
11	T	29-Mar-2016	L18	HW/SW Partitioning	6.3				
	R	31-Mar-2016	L19	Mapping Applications to Multiprocessors	6.4				
12	T	5-Apr-2016	L20	Intro to Compile-time Optimization	7.1-7.2				
	R	7-Apr-2016	L21	Energy/Memory-aware Compilation	7.3.1-7.3.3				
13	T	12-Apr-2016	L22	Further Optimization	7.3.4-7.4				
	R	14-Apr-2016	L23	Further Optimization					
15	R	28-Apr-2016		Final Exam: closed book, cumulative					9:00 AM

Hypothetical Design Flow



Validation and Evaluation

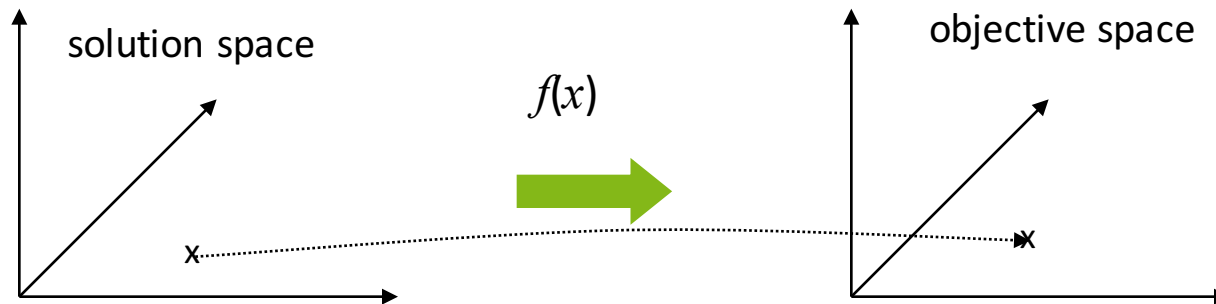
- **Definition:** *Validation* is the process of checking whether or not a certain (possibly partial) design is appropriate for its purpose, meets all constraints and will perform as expected (yes/no decision).
- **Definition:** Validation with mathematical rigor is called *(formal) verification*.
- **Definition:** *Evaluation* is the process of computing quantitative information of some key characteristics of a certain (possibly partial) design.

Multi-objective Evaluation

- Many different criteria are often relevant
 - Average speed
 - Worst case speed
 - Power dissipation
 - Energy consumption
 - Cost, size, weight
 - Radiation hardness
 - Environmental friendliness
- How to compare different designs?
(some designs are “better” than others)

Definitions

- Let X : m -dimensional **solution space** for the design problem
 - E.g.: dimensions correspond to # of processors, size of memories, type and width of busses, *etc.*
- Let F : n -dimensional **objective space** for the design problem
 - E.g.: dimensions correspond to speed, cost, power consumption, size, weight, reliability, ...
- Let $f(x) = (f_1(x), \dots, f_n(x))$ where $x \in X$ be an **objective function**
 - We assume that we are using $f(x)$ for evaluating designs



Pareto Points: Dominance, Indifference

- We assume that, for each objective, a total order $<$ and the corresponding order \leq are defined
- **Definition:** Vector $u=(u_1, \dots, u_n) \in F$ **dominates** vector $v=(v_1, \dots, v_n) \in F \Leftrightarrow$
 u is “better” than v with respect to one objective and not worse than v with respect to all other objectives:

$$\begin{aligned} &\forall i \in \{1, \dots, n\} : u_i \leq v_i \wedge \\ &\exists i \in \{1, \dots, n\} : u_i < v_i \end{aligned}$$

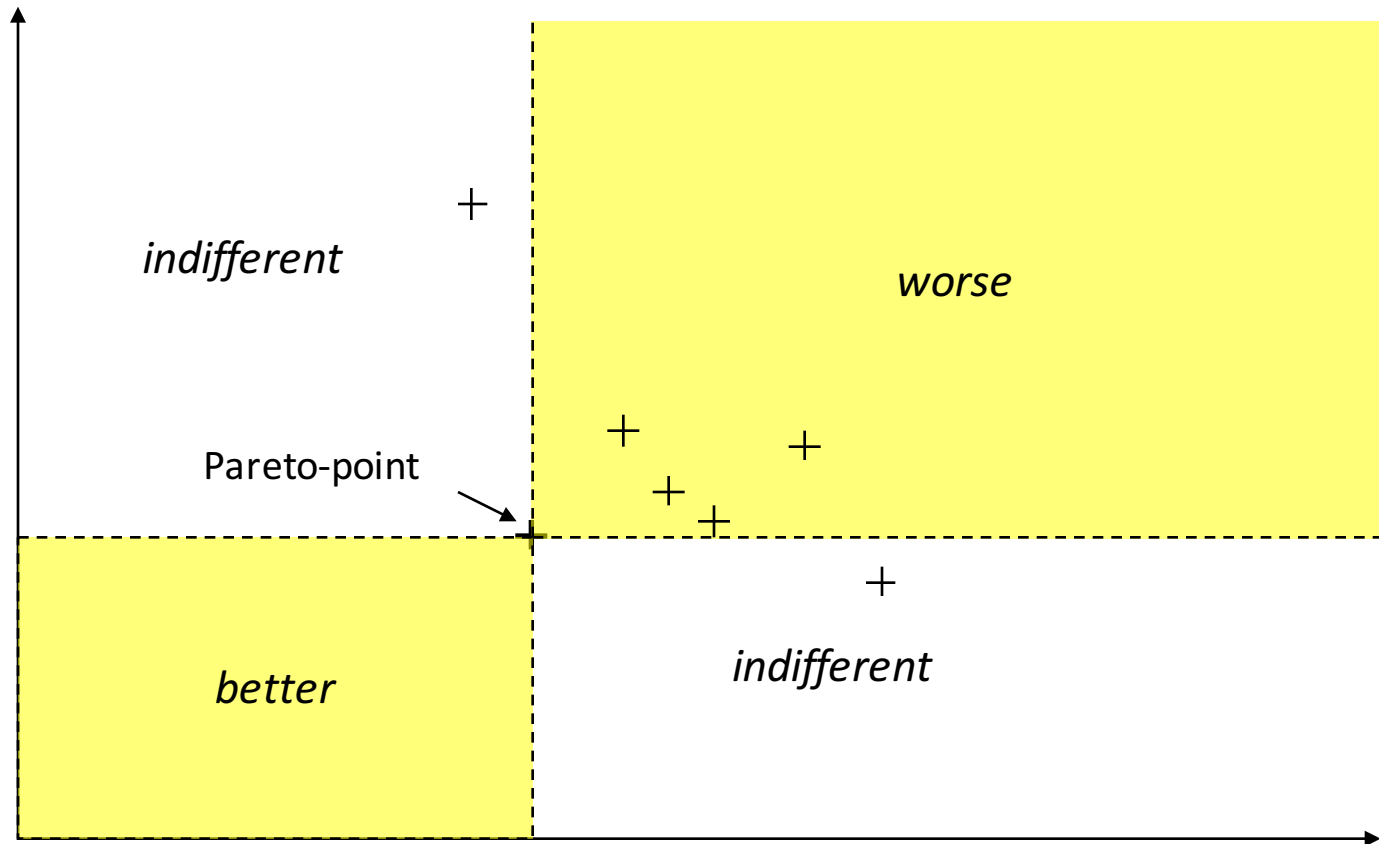
- **Definition:** Vector $u \in F$ is **indifferent** with respect to vector $v \in F \Leftrightarrow$ neither u dominates v nor v dominates u

Pareto Points: Pareto-optimality

- A solution $x \in X$ is called **Pareto-optimal** with respect to $X \Leftrightarrow$ there is no solution $y \in X$ such that $u=f(x)$ is dominated by $v=f(y)$
- **Definition:** Let $S \subseteq F$ be a subset of solutions. v is called a **non-dominated solution** with respect to $S \Leftrightarrow v$ is not dominated by any element $\in S$
- v is called **Pareto-optimal** $\Leftrightarrow v$ is non-dominated with respect to all solutions F

Visualizing Pareto Points

Objective 1
(e.g. energy
consumption)

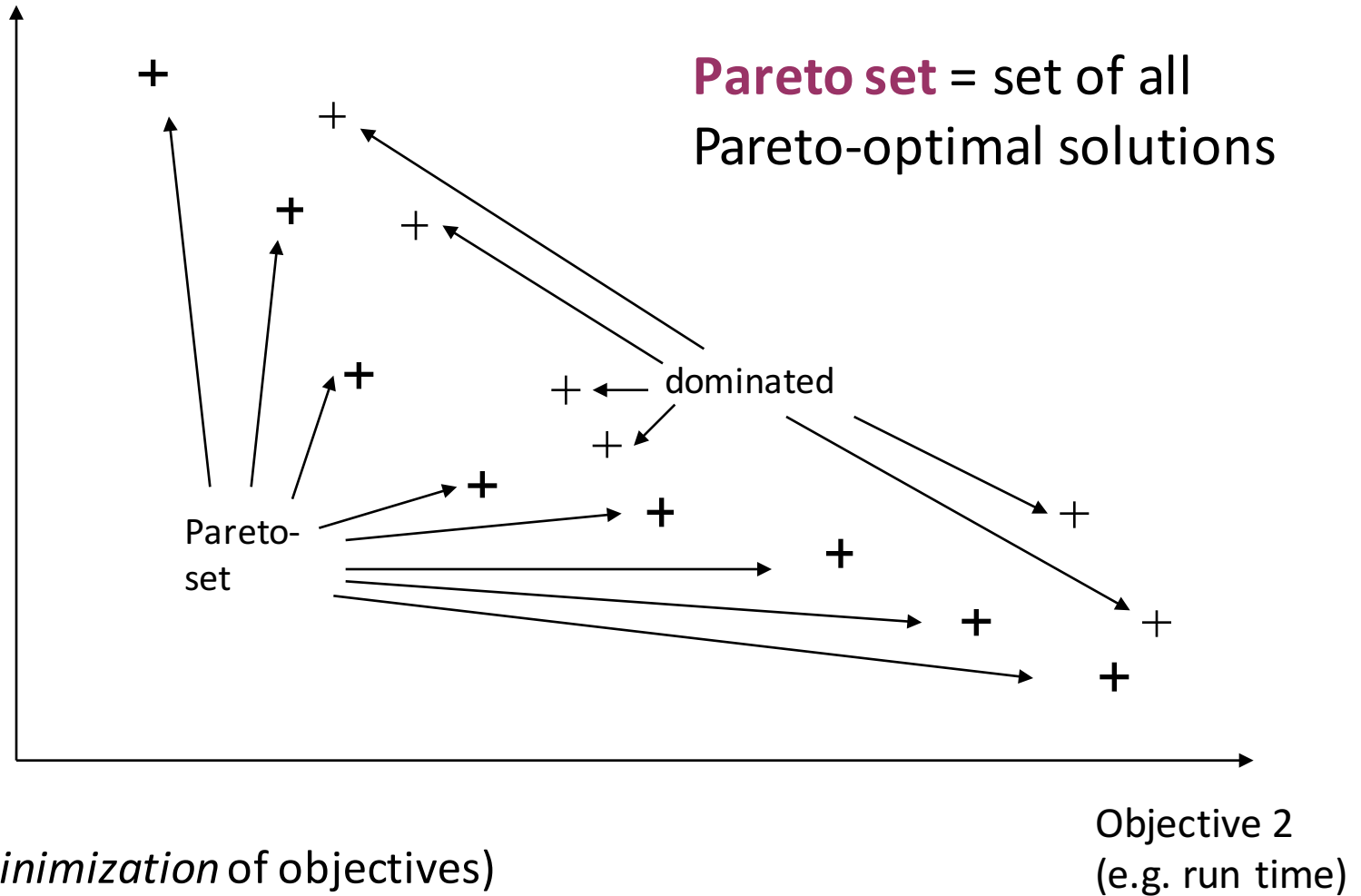


(Assuming *minimization* of objectives)

Objective 2
(e.g. run time)

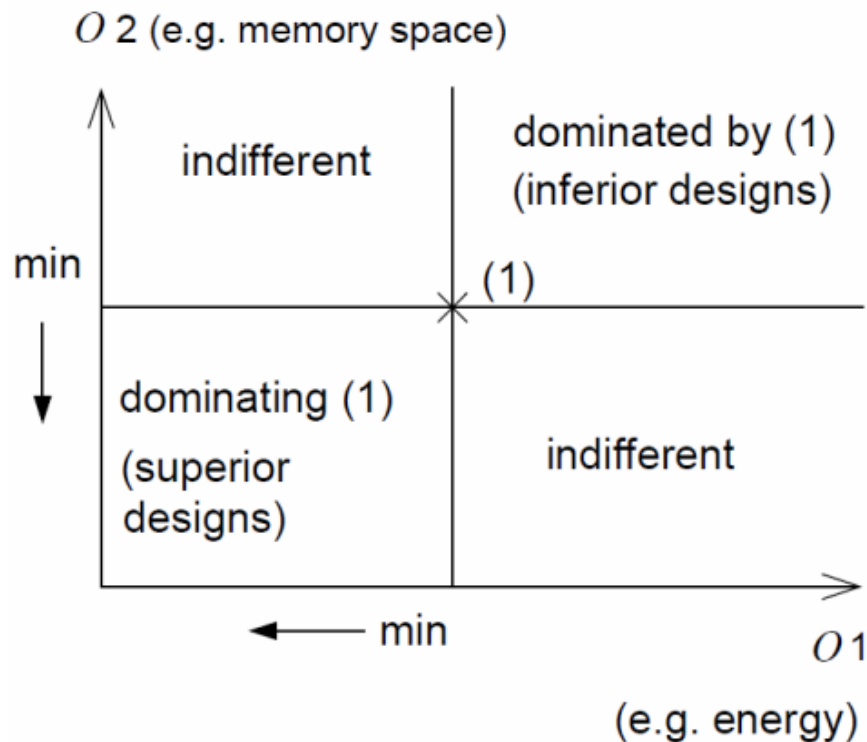
Pareto-optimal Set

Objective 1
(e.g. energy consumption)

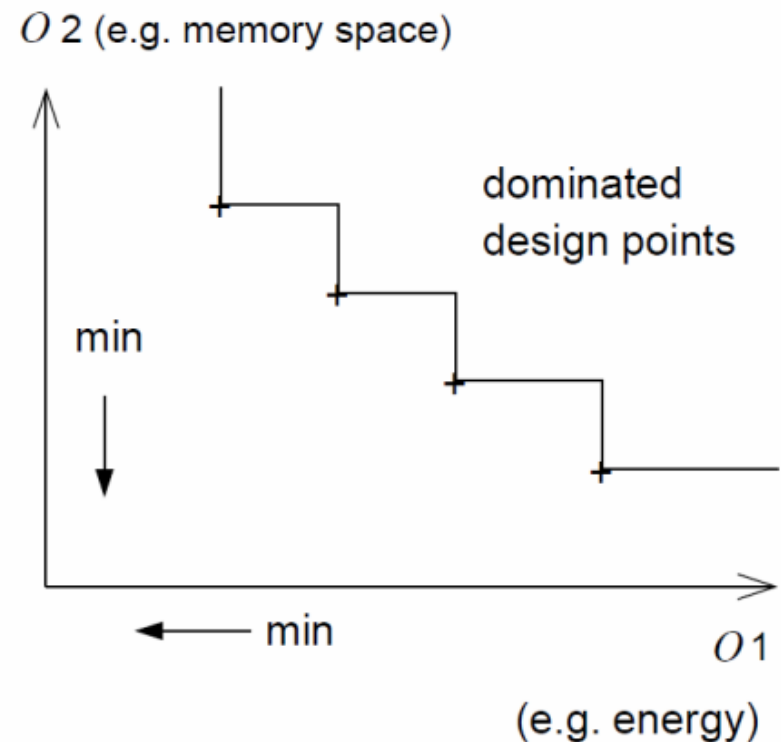


Pareto Points and Pareto Fronts

Pareto Point



Pareto Front



Design Space Evaluation (DSE)

- DSE *based on Pareto-points*
 - The process of finding and returning a set of Pareto-optimal designs to the user
 - Enabling the user to select the most appropriate design
- Lots of research and development in this area
 - General search (*e.g.*, genetic algorithms, simulated annealing, satisfiability)
 - Heuristic search (*i.e.*, special approaches to solve restricted problems)

Multi-Objective Design

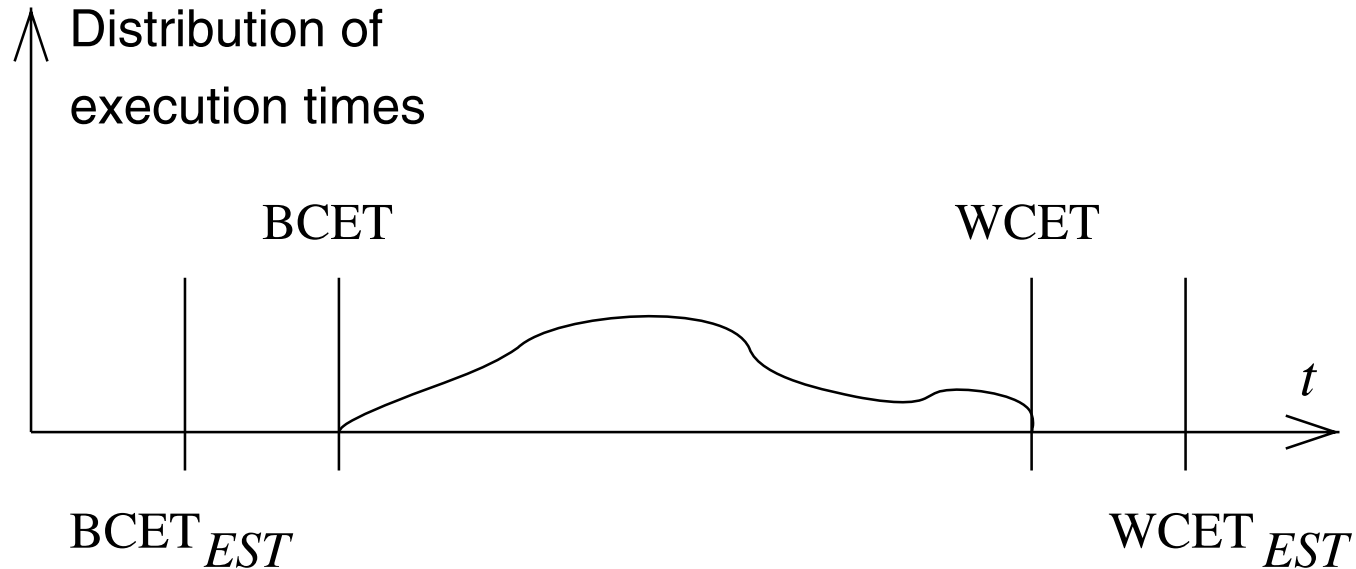
- Different design objectives/criteria are relevant:
 - Average performance
 - Worst case performance
 - Energy/power consumption
 - Thermal behavior
 - Reliability
 - Electromagnetic compatibility
 - Numeric precision
 - Testability
 - Cost
 - Weight, robustness, usability, extendibility, security, safety, environmental friendliness



Performance Evaluation

- Estimated performance values
 - Difficult to generate sufficiently precise estimates
 - Balance between run-time and precision
- Accurate and precise performance values
 - As accurate and precise as the input data
 - *Accuracy*: distance from estimate to real value
 - *Precision*: distance between estimates of real value
- Two performance metrics:
 - Average case execution time
 - Worst case execution time

Worst/best Case Execution Times



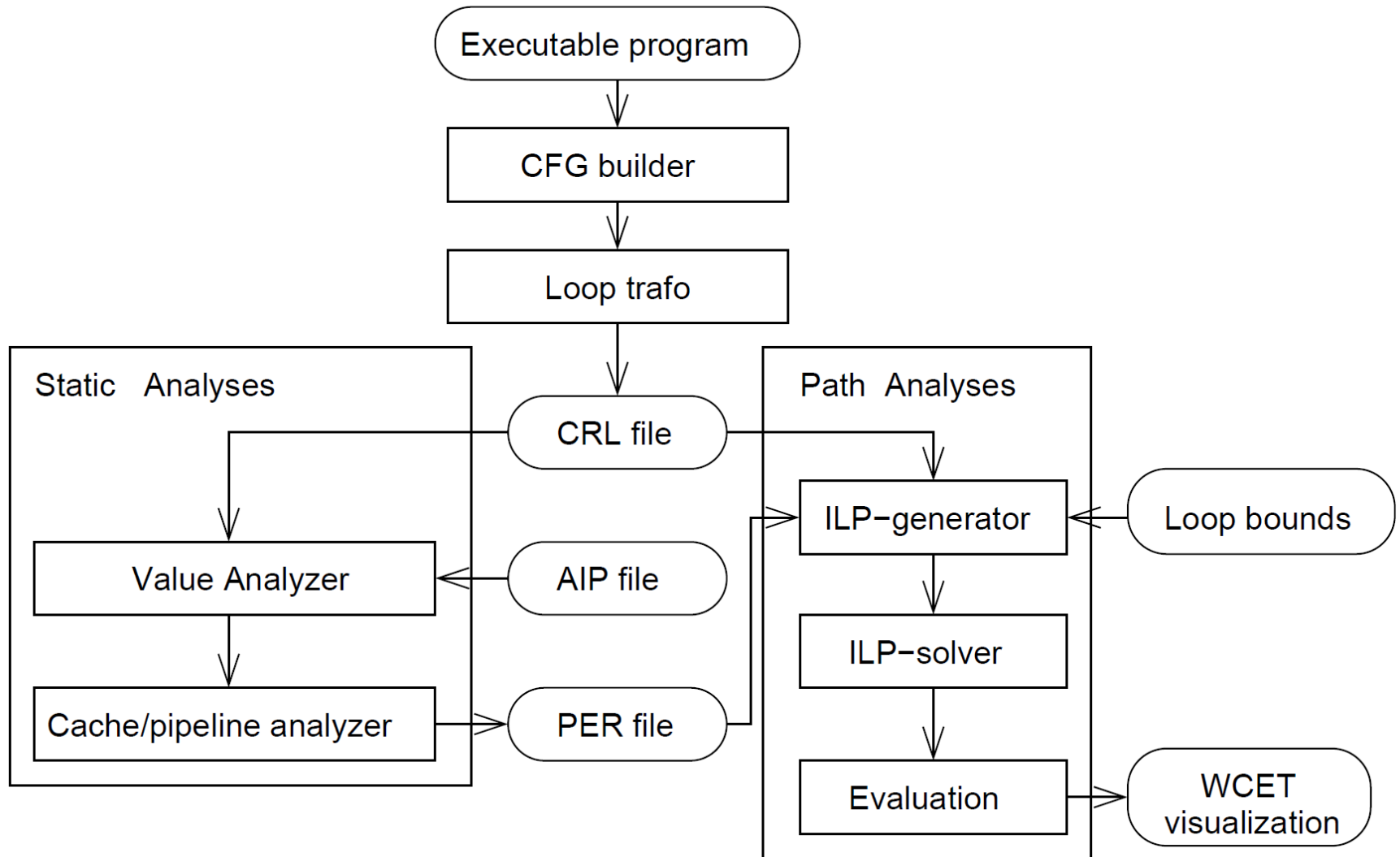
- Requirements for WCET estimates:
 - Safeness: $WCET \leq WCET_{EST}$!
 - Tightness: $WCET_{EST} - WCET \rightarrow \text{minimal}$

Worst Case Execution Times (2)

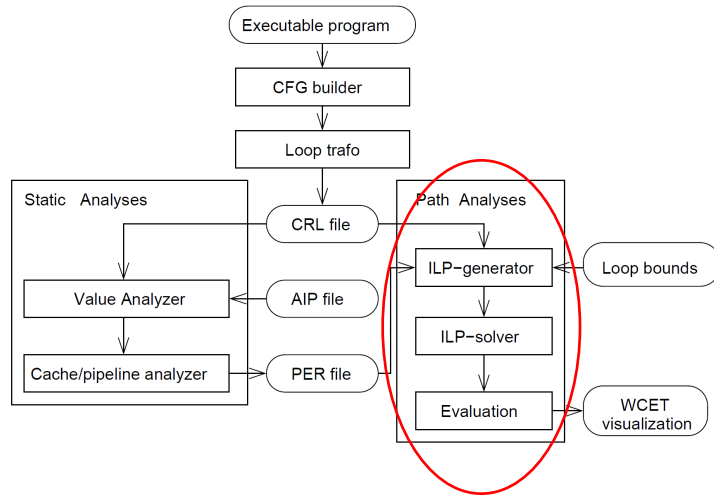
- Estimating WCET is *computationally complex*
- In the general case
 - Whether a bound exists is undecidable
- For restricted programs
 - Simple for “old” architectures
 - Very complex for new architectures with pipelines, caches, interrupts, virtual memory, *etc.*
- Approaches:
 - For hardware: requires detailed timing behavior
 - For software: requires availability of machine programs; complex analysis (see, e.g., www.absint.com)



WCET Estimation: AiT (AbsInt)



Integer Linear Programming (ILP) Model



- Objective function
 - Execution time as a function of the execution time of blocks
 - To be maximized
- Constraints
 - Dependencies between blocks
- Avoids explicit consideration of all paths (too complex)
 - Called *implicit path enumeration* technique

Example (1)

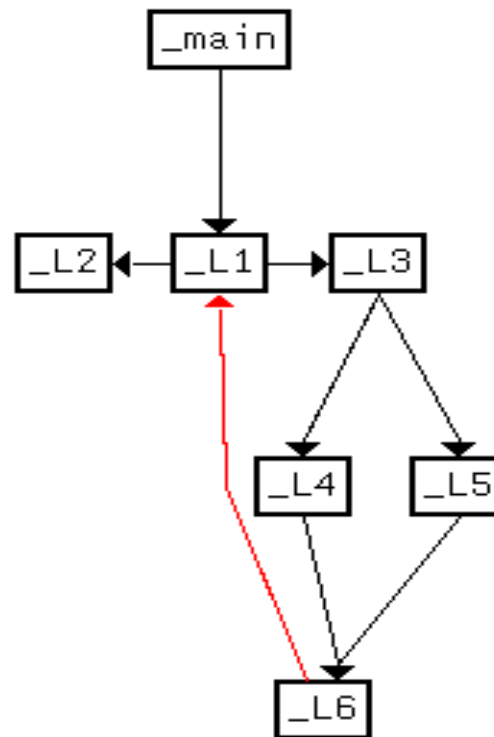
Program

```
int main()
{
    int i, j = 0;

    _Pragma( "loopbound min
              100 max 100" );
    for ( i = 0; i < 100; i++ ) {
        if ( i < 50 )
            j += i;
        else
            j += ( i * 13 ) % 42;
    }

    return j;
}
```

CFG



WCETs of BB (aiT 4 TriCore)

```
_main: 21 cycles
_L1: 27
_L3: 2
_L4: 2
_L5: 20
_L6: 13
_L2: 20
```

Example (2)

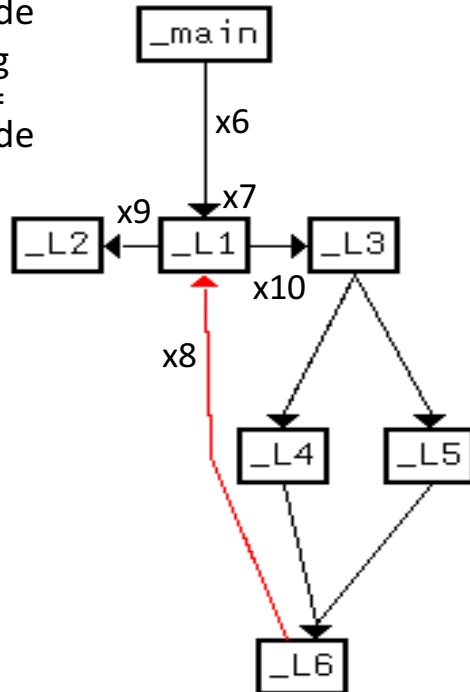
- Virtual start node
- Virtual end node
- Virtual end node per function

Variables:

- 1 variable per node
- 1 variable per edge

Constraints: “Kirchhoff” equations per node

- Sum of incoming edge variables = flux through node
- Sum of outgoing edge variables = flux through node



`_main`: 21 cycles
`_L1`: 27
`_L3`: 2
`_L4`: 2
`_L5`: 20
`_L6`: 13
`_L2`: 20

ILP

```

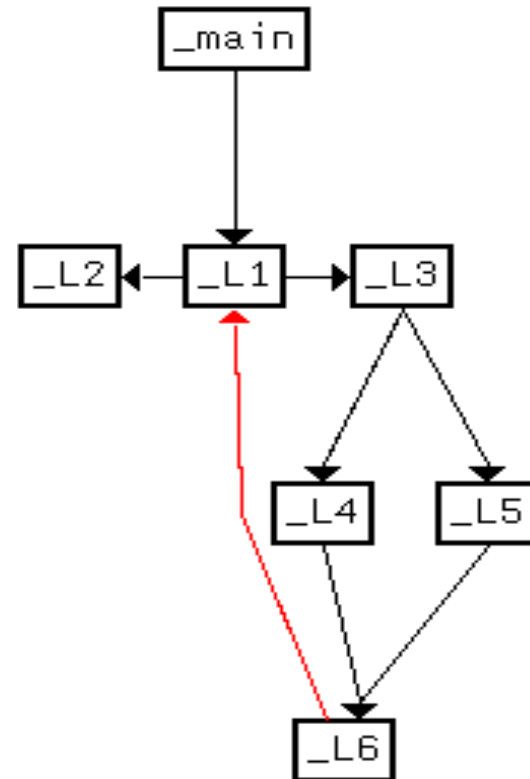
/* Objective function = WCET to be maximized*/
21 x2 + 27 x7 + 2 x11 + 2 x14 + 20 x16 + 13 x18 + 20 x19;
/* CFG Start Constraint */ x0 - x4 = 0;
/* CFG Exit Constraint */ x1 - x5 = 0;
/* Constraint for flow entering function main */
x2 - x4 = 0;
/* Constraint for flow leaving exit node of main */
x3 - x5 = 0;
/* Constraint for flow entering exit node of main */
x3 - x20 = 0;
/* Constraint for flow entering main = flow leaving main */
x2 - x3 = 0;
/* Constraint for flow leaving CFG node _main */
x2 - x6 = 0;
/* Constraint for flow entering CFG node _L1 */
x7 - x8 - x6 = 0;
/* Constraint for flow leaving CFG node _L1 */
x7 - x9 - x10 = 0;
/* Constraint for lower loop bound of _L1 */
x7 - 101 x9 >= 0;
/* Constraint for upper loop bound of _L1 */
x7 - 101 x9 <= 0; ....
  
```

Example (3)

Value of objective function: 6268

Actual values of the variables:

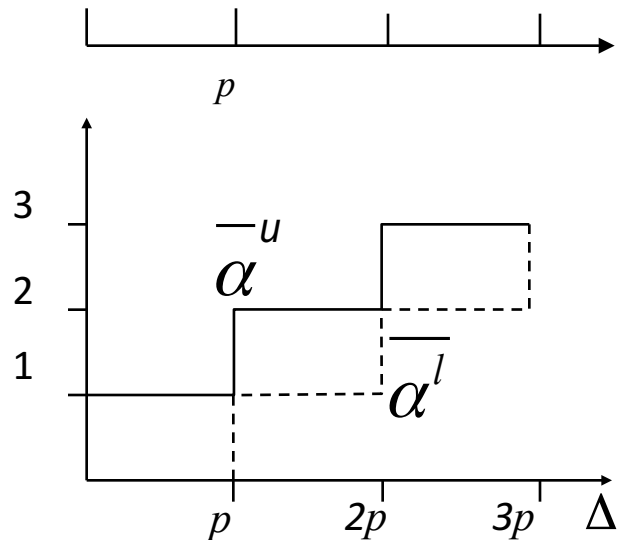
x2	1
x7	101
x11	100
x14	0
x16	100
x18	100
x19	1
x0	1
x4	1
x1	1
x5	1
x3	1
x20	1
x6	1
x8	100
x9	1
x10	100
x12	100
x13	0
x15	0
x17	100



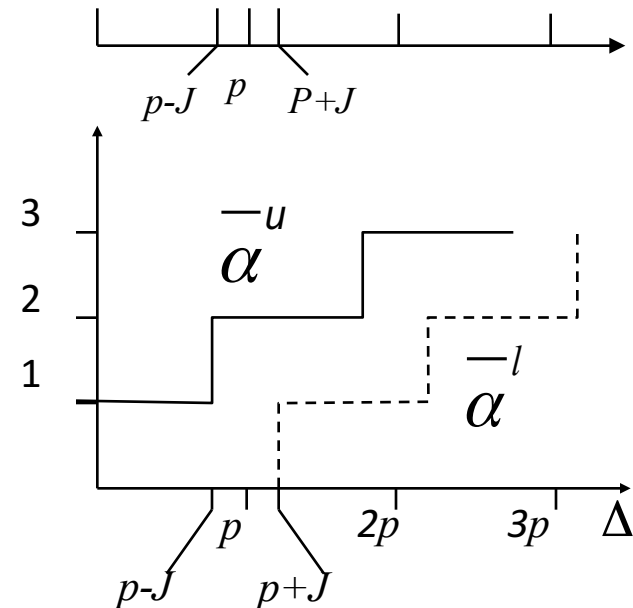
RTC/MPA: Arrival Curves

- Real-time Calculus/Modular Performance Analysis
 - Thiele *et al.* (ETHZ)
- Extended **network calculus**
 - Arrival curves describe the bounds on events arriving in some interval Δ
- Examples:

periodic event stream

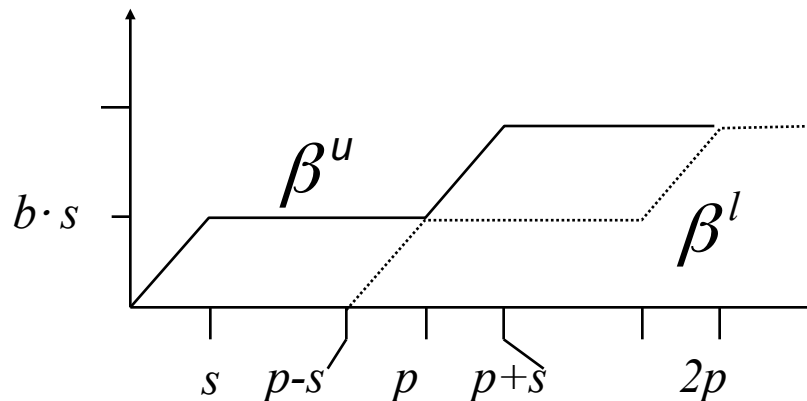
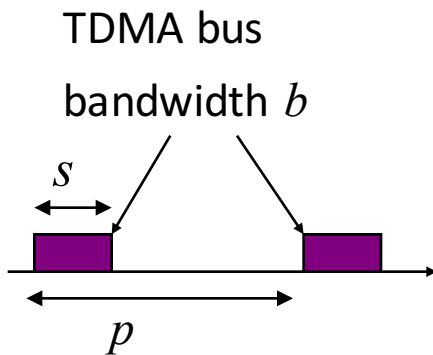


periodic event stream with jitter



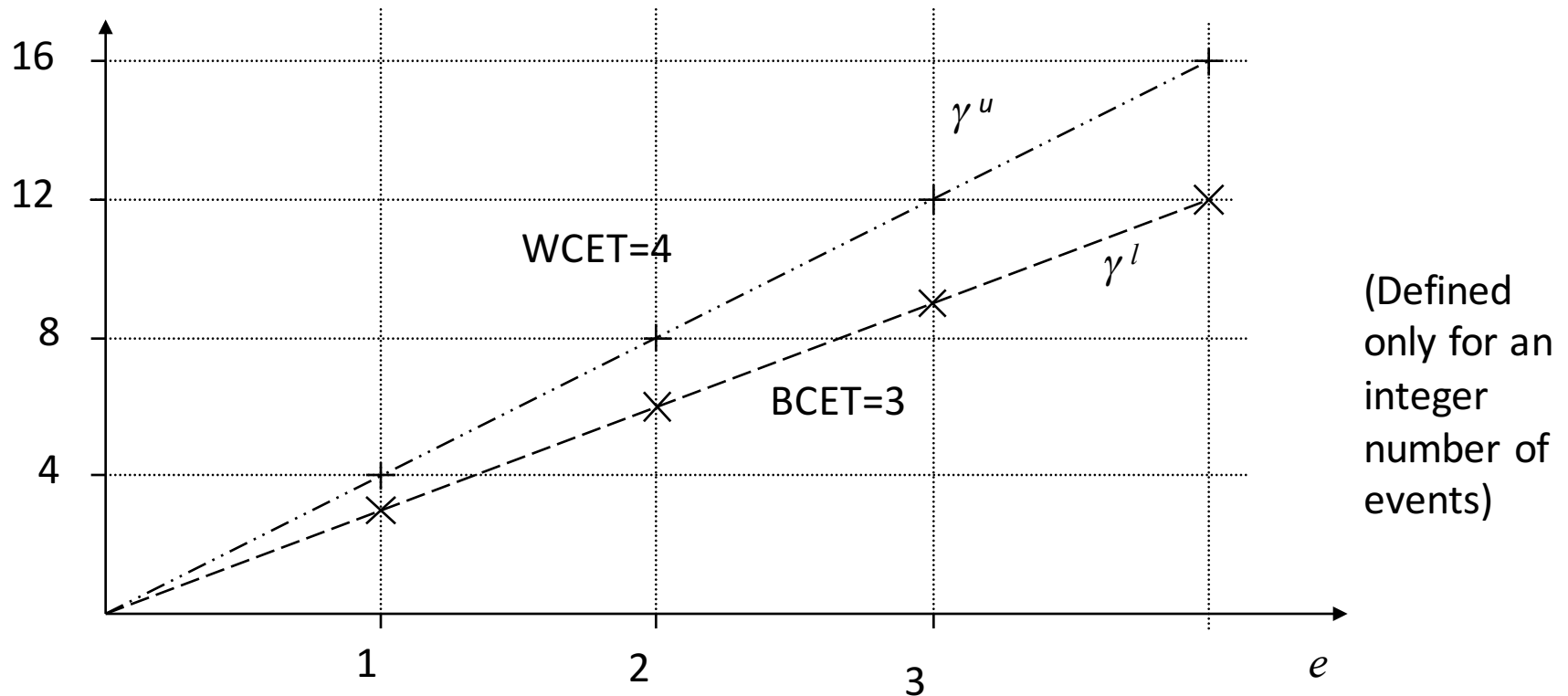
RTC/MPA: Service curves

- Service curves β^u and β^l respectively describe the maximum and minimum service capacity available in some time interval Δ
- Example:



RTC/MPA: Workload Characterization

- γ^u and γ^l respectively describe the maximum and minimum service capacity required as a function of the number of events e
- Example:



RTC/MPA: Workload Required

- Upper and lower bounds on incoming workload

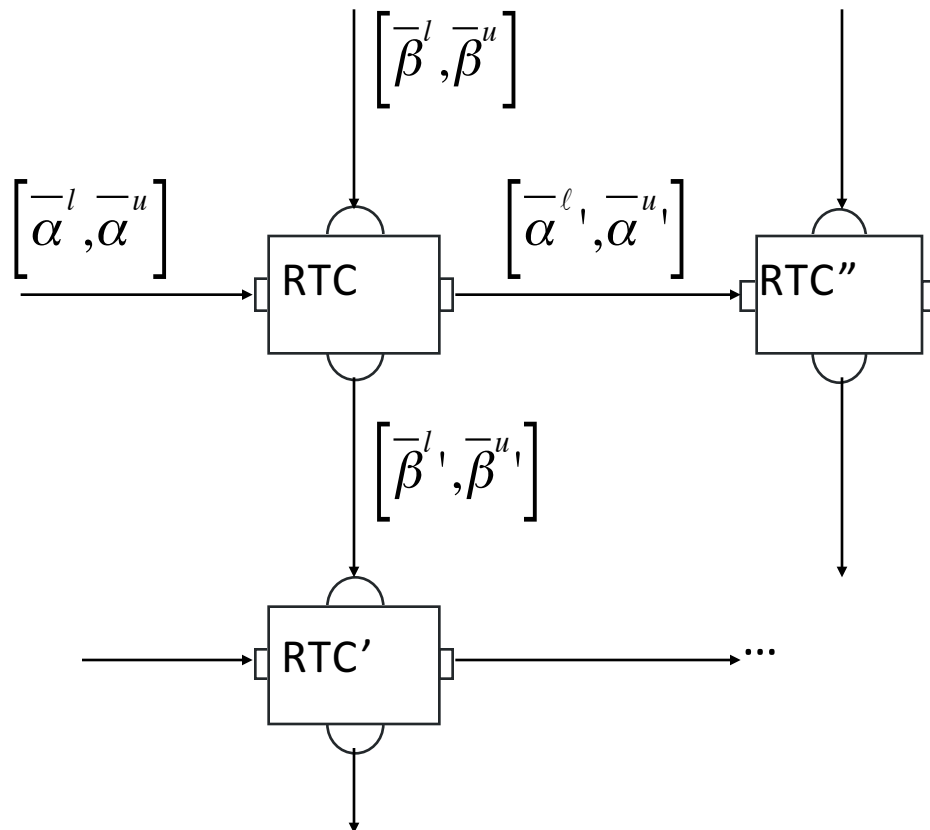
$$\alpha^u(\Delta) = \gamma^u(\overline{\alpha^u}(\Delta)) \qquad \alpha^l(\Delta) = \gamma^l(\overline{\alpha^l}(\Delta))$$


- Upper and lower bounds on the number of events that can be processed

$$\overline{\beta^u}(\Delta) = (\gamma^l)^{-1}(\beta^u(\Delta)) \qquad \overline{\beta^l}(\Delta) = (\gamma^u)^{-1}(\beta^l(\Delta))$$

RTC/MPA: System of RT Components

- Incoming event streams and available capacity are transformed by real-time components



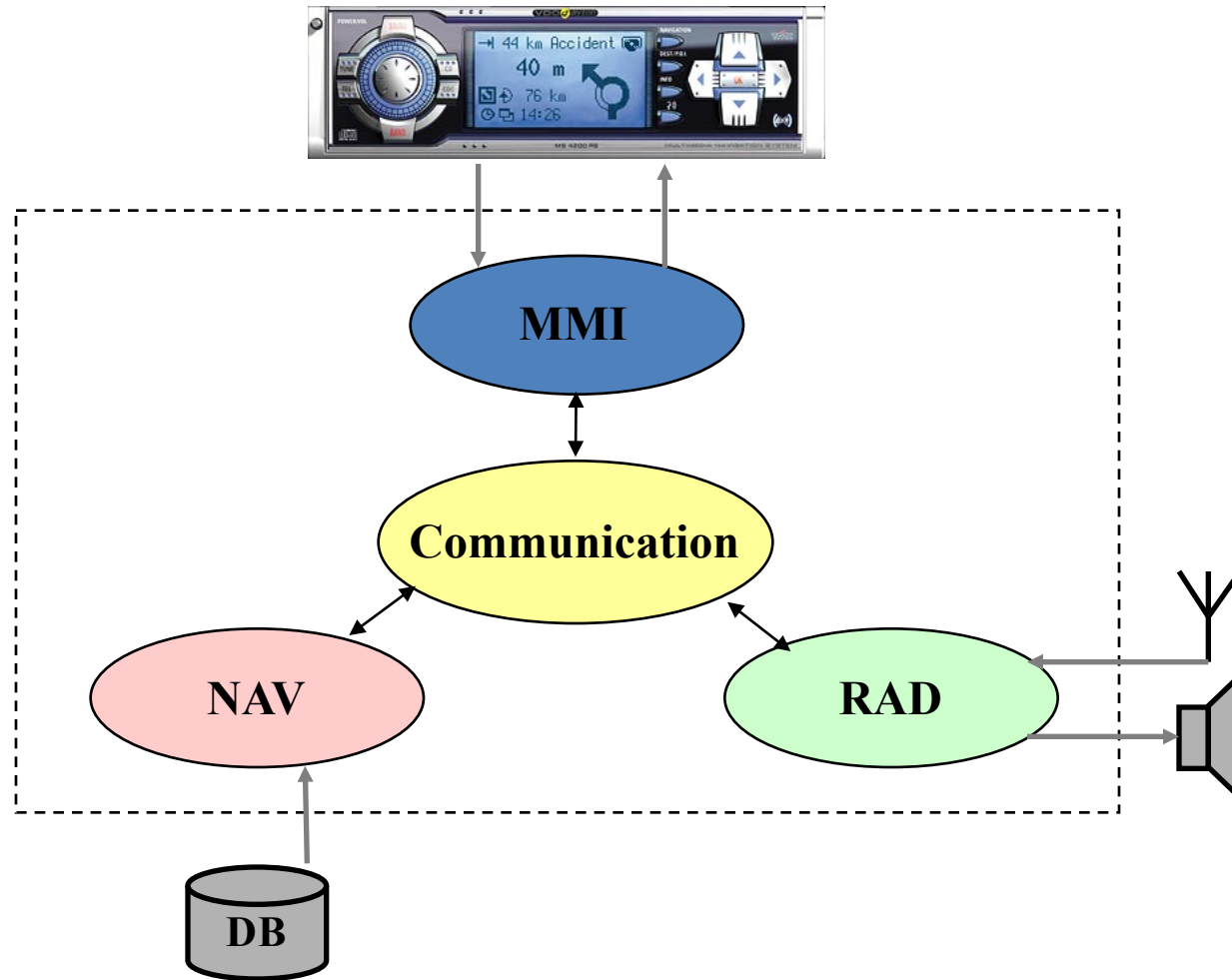
Theoretical results allow the computation of properties of outgoing streams 

Example: In-Car Navigation System

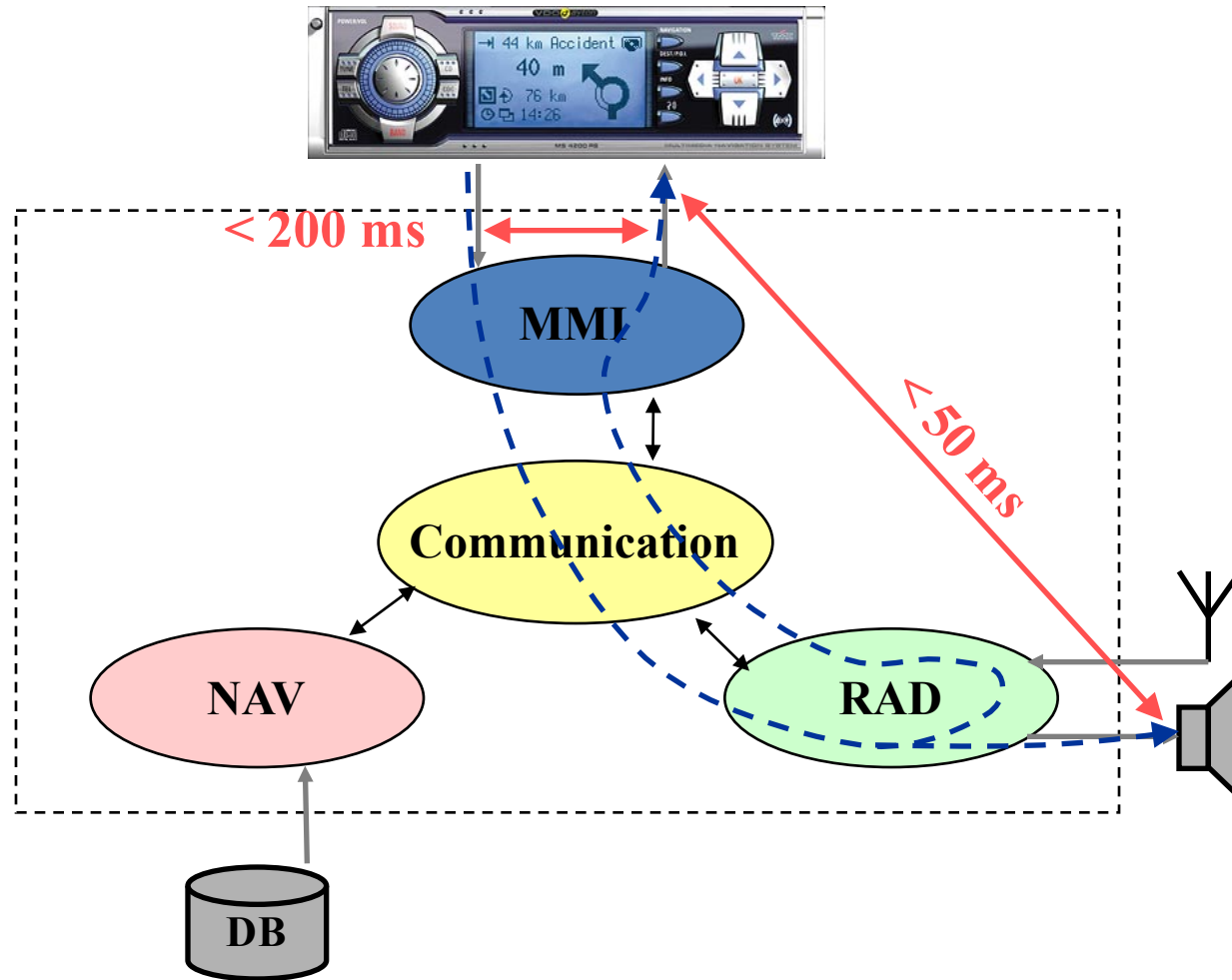
- Car radio with navigation system
 - User interface should be responsive
 - Traffic messages (TMC) must be handled in a timely way
 - Several applications may execute concurrently



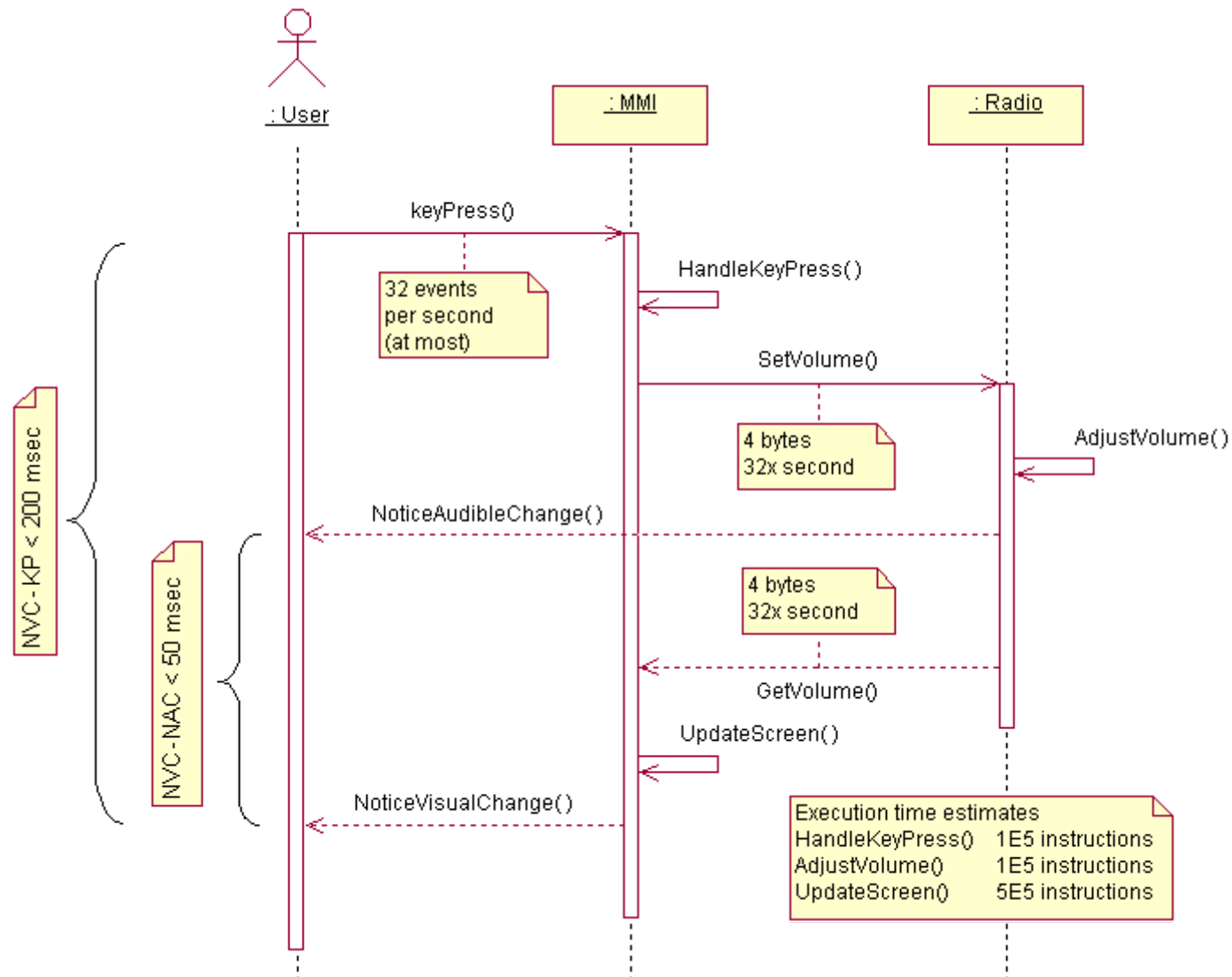
System Overview



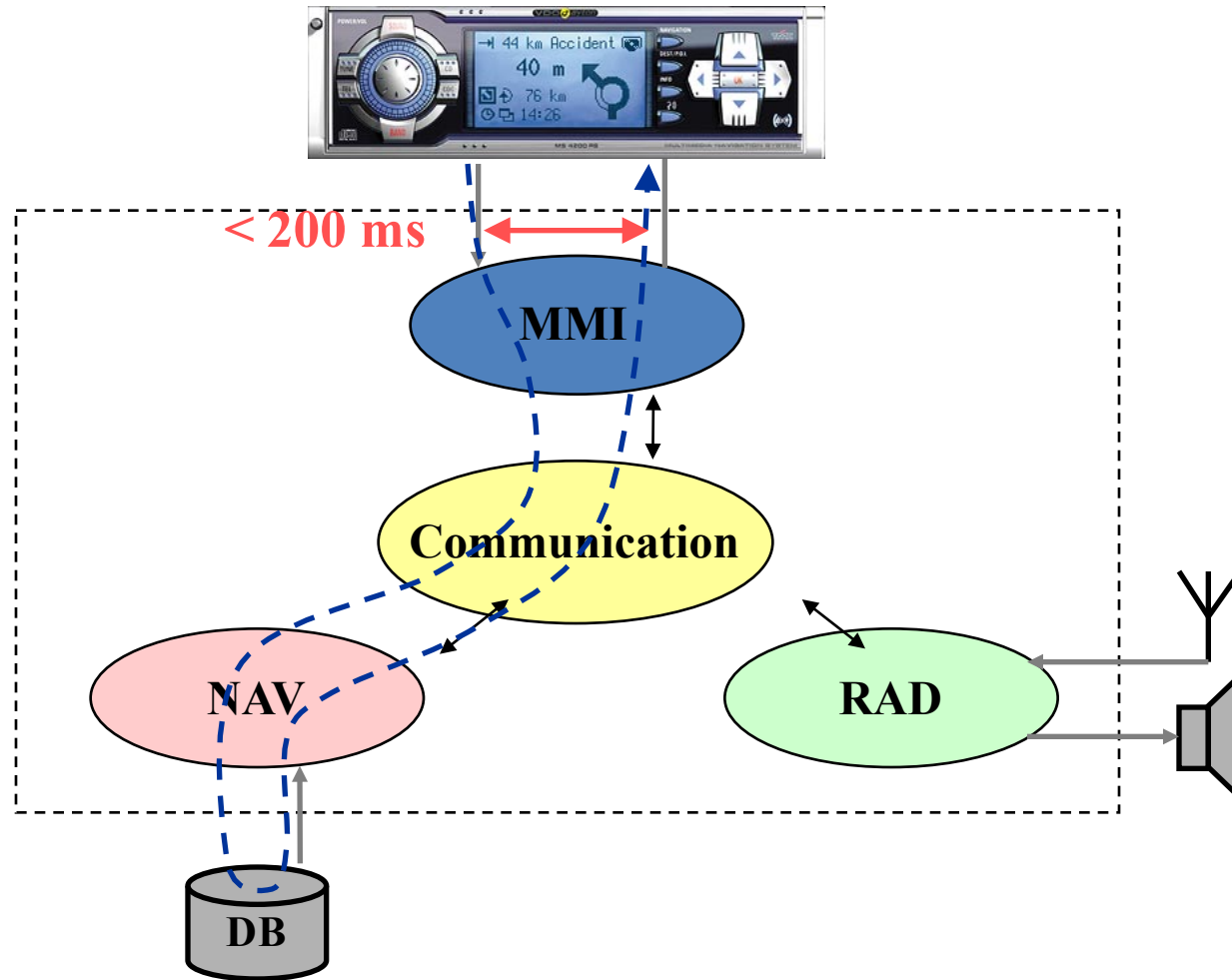
Use Case 1: Change Audio Volume



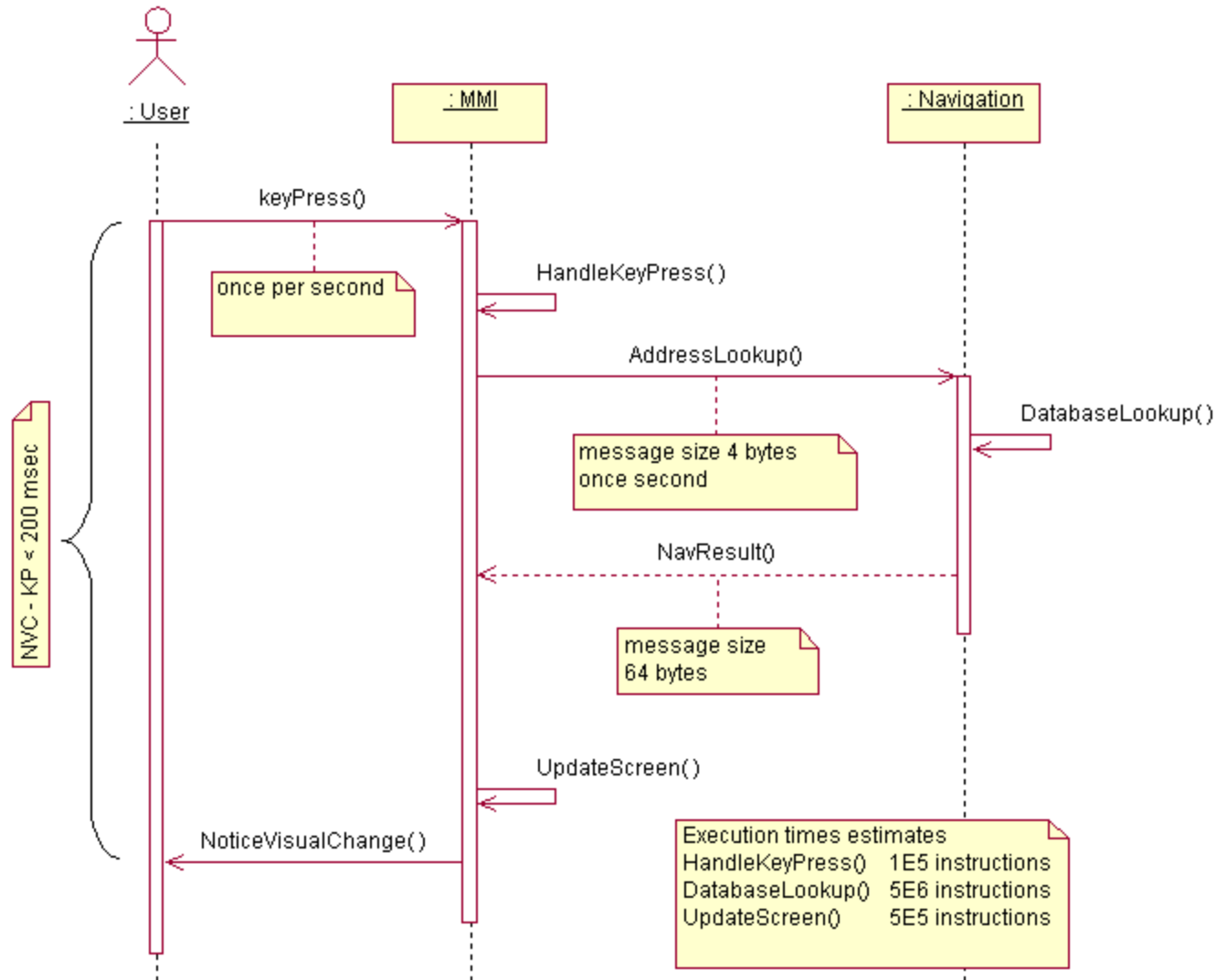
Use Case 1: Change Audio Volume



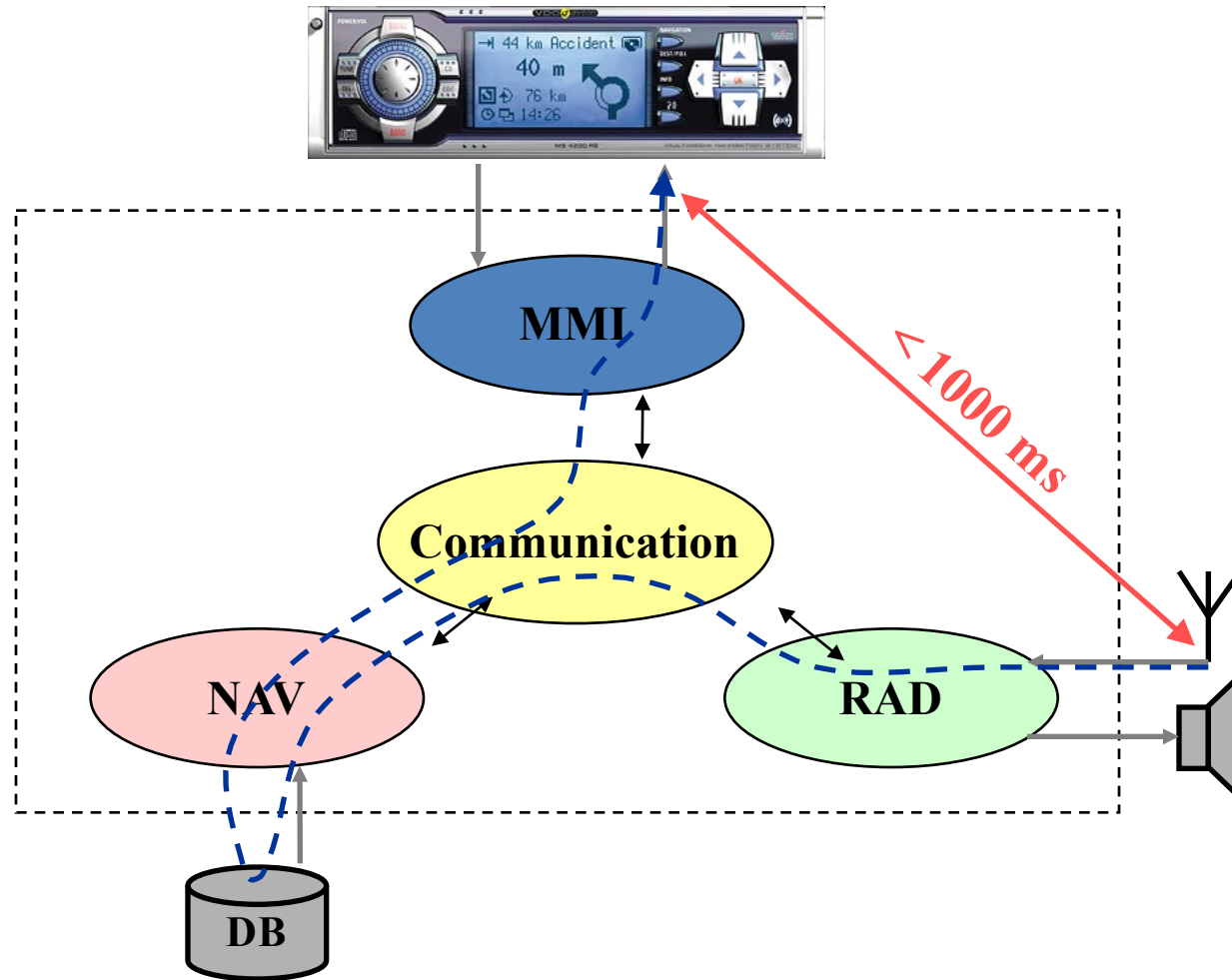
Use Case 2: Lookup Destination Address



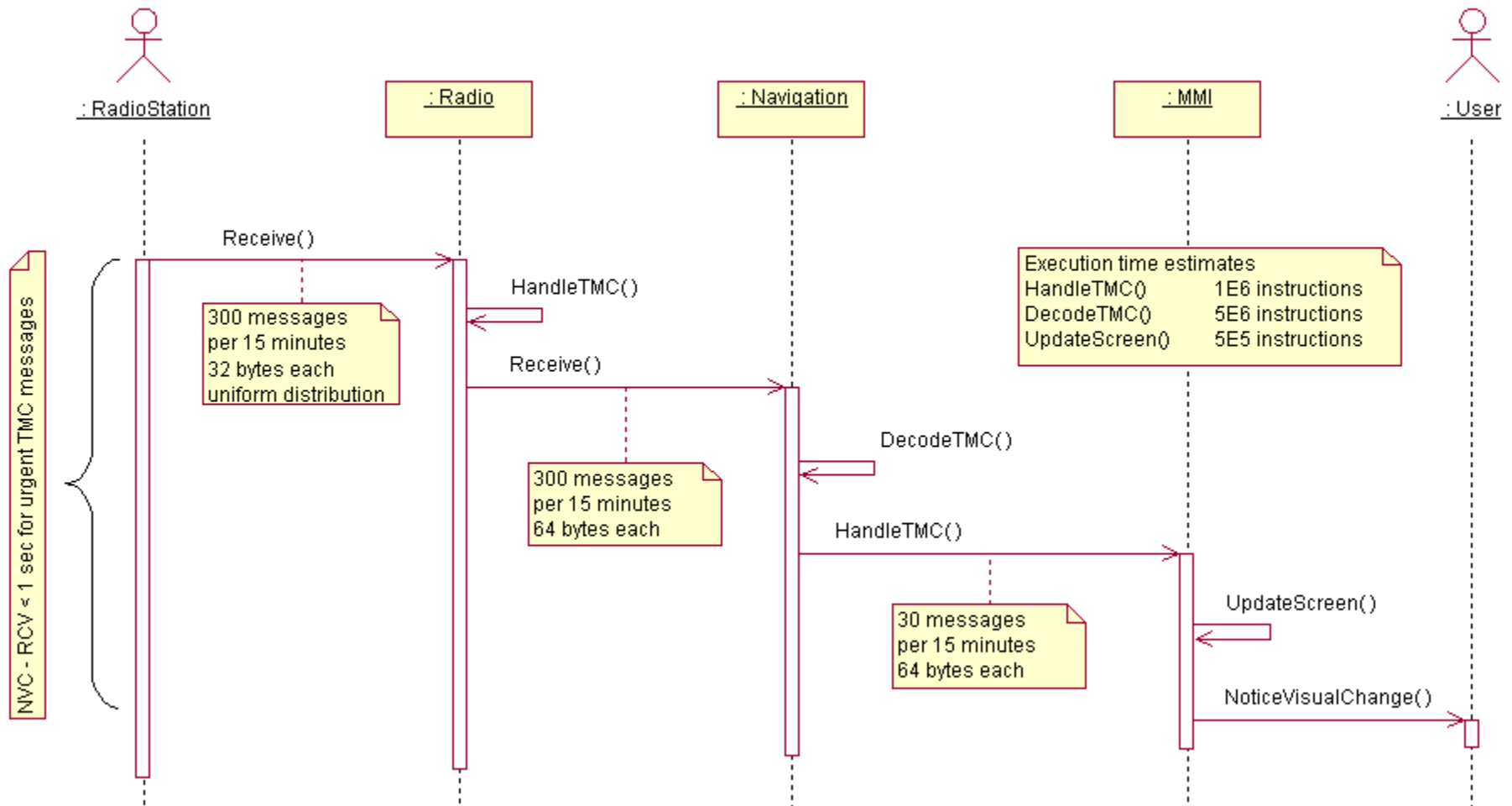
Use Case 2: Lookup Destination Address



Use Case 3: Receive TMC Messages

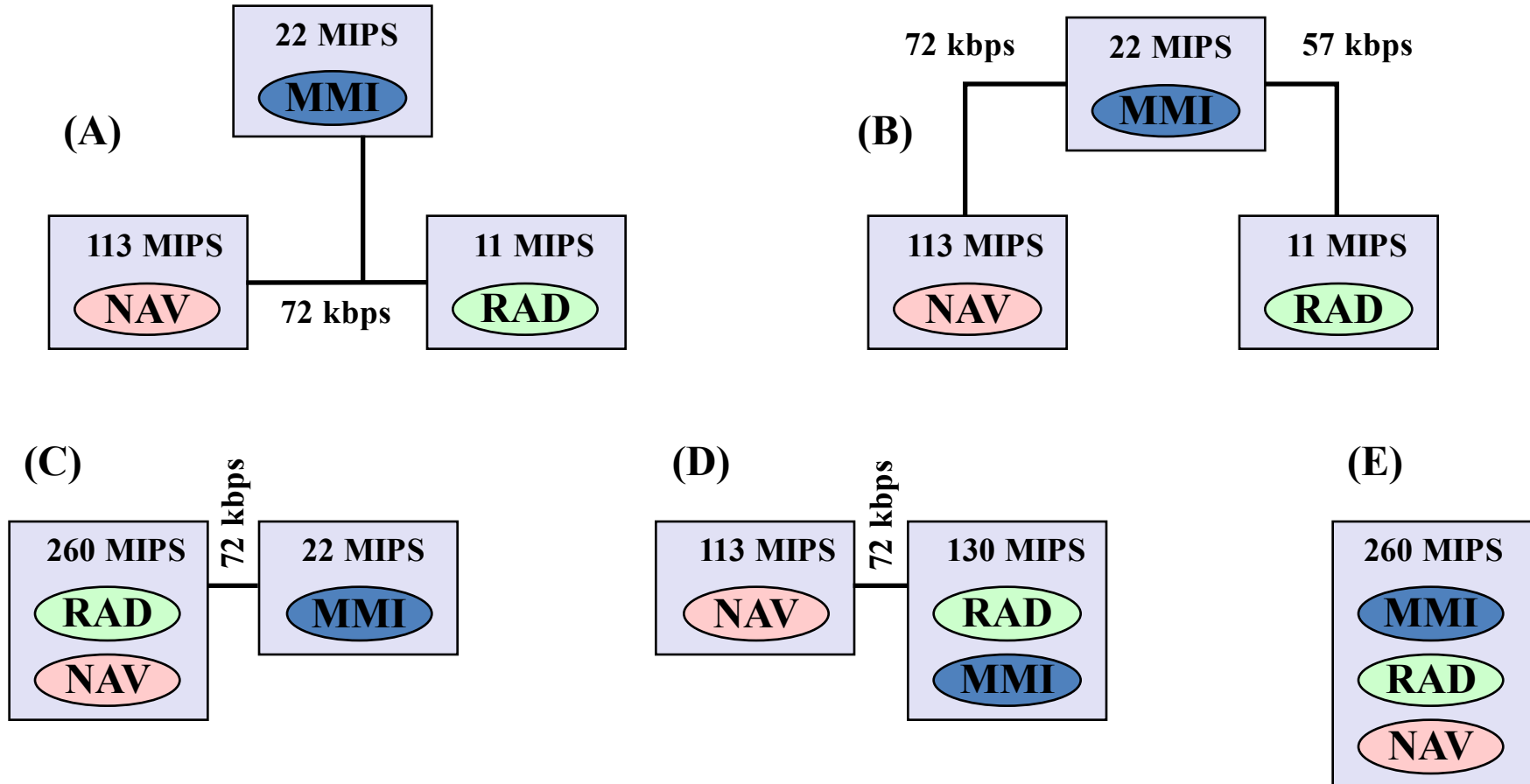


Use Case 3: Receive TMC Messages



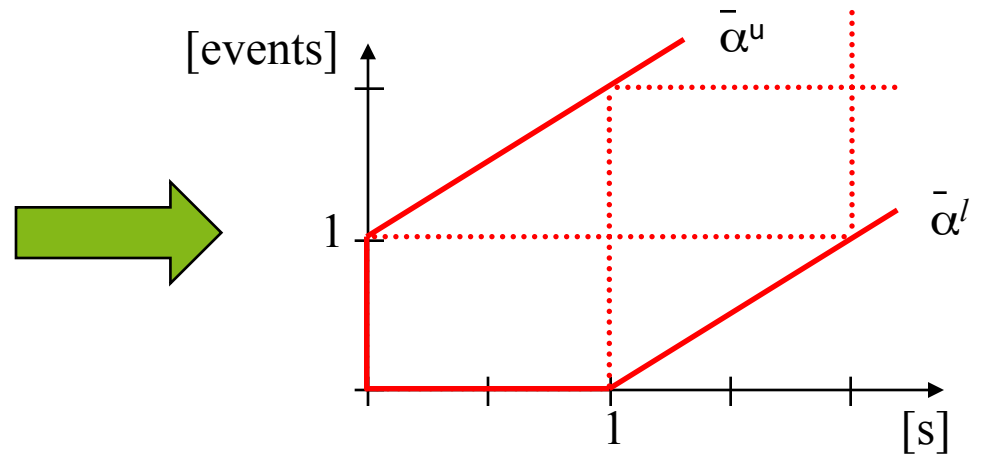
Proposed Architecture Alternatives

© Thiele, ETHZ



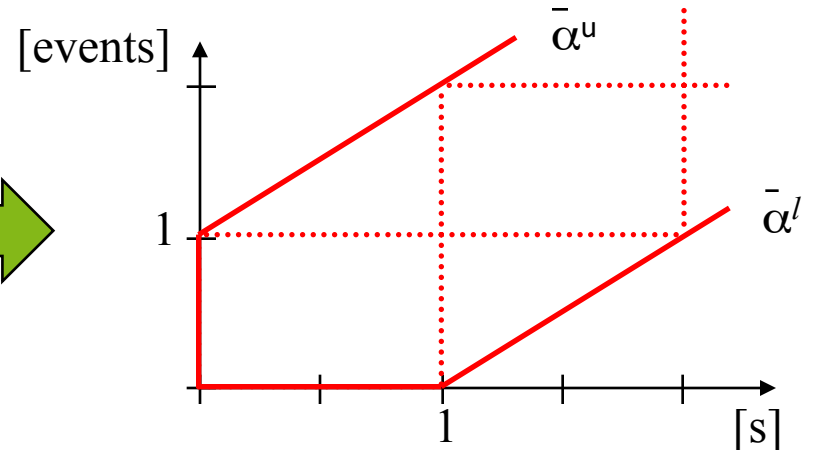
Step 1: Environment (Event Streams)

- Event Stream Model
 - *E.g.* Address Lookup
 - (1 event / sec)

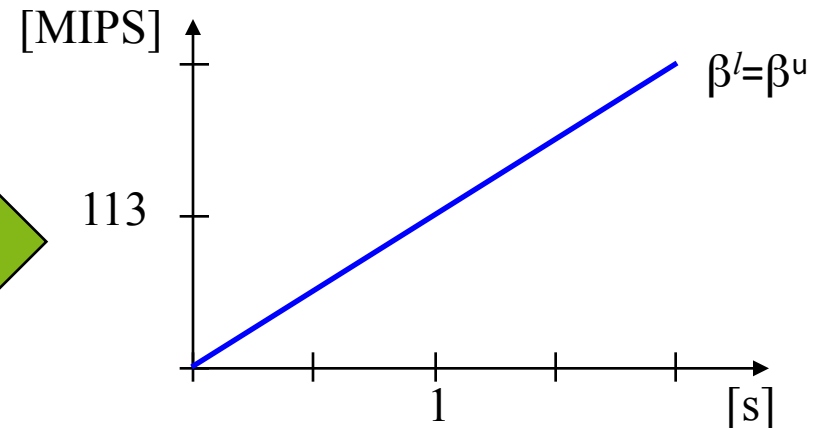


Step 2: Architectural Elements

- Event Stream Model
 - *E.g.* Address Lookup
 - (1 event / sec)



- Resource Model
 - *E.g.* unloaded RISC CPU
 - (113 MIPS)

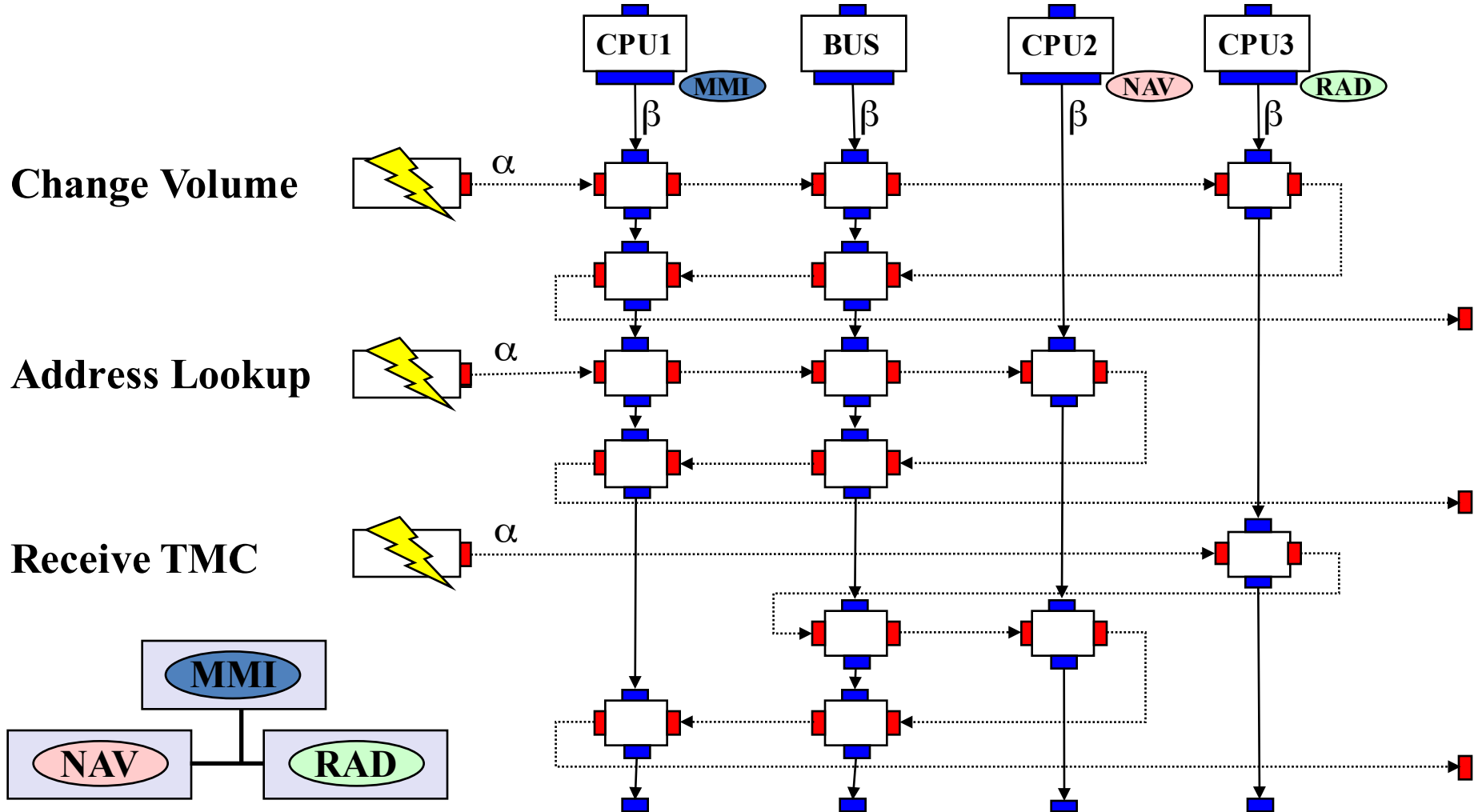


Step 3: Mapping / Scheduling

- Rate Monotonic Scheduling
(Pre-emptive fixed priority scheduling):
 - Priority 1: Change Volume ($p=1/32$ s)
 - Priority 2: Address Lookup ($p=1$ s)
 - Priority 3: Receive TMC ($p=6$ s)

Step 4: Performance Model

© Thiele, ETHZ



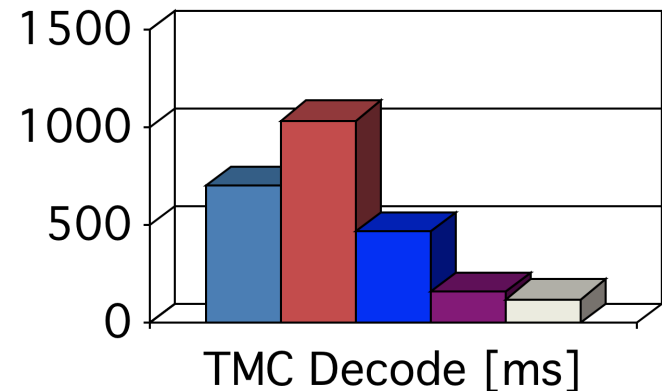
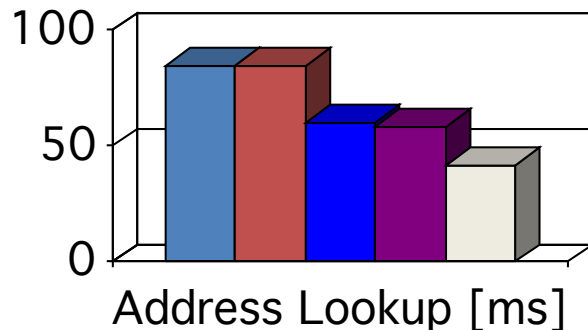
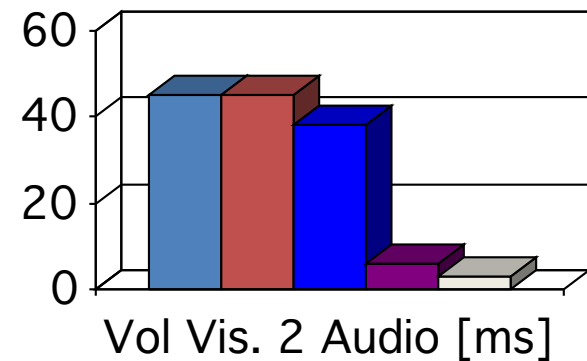
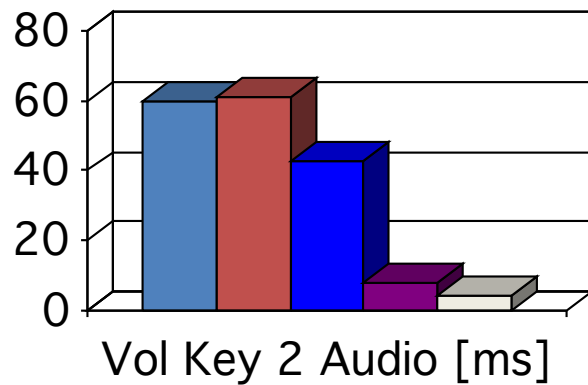
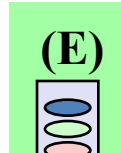
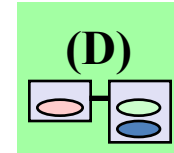
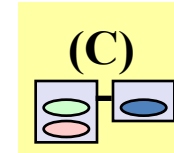
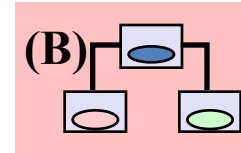
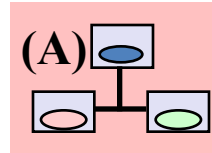
Analysis – Design Question 1

- How do the proposed system architectures compare in respect to end-to-end delays?

Analysis – Design Question 1

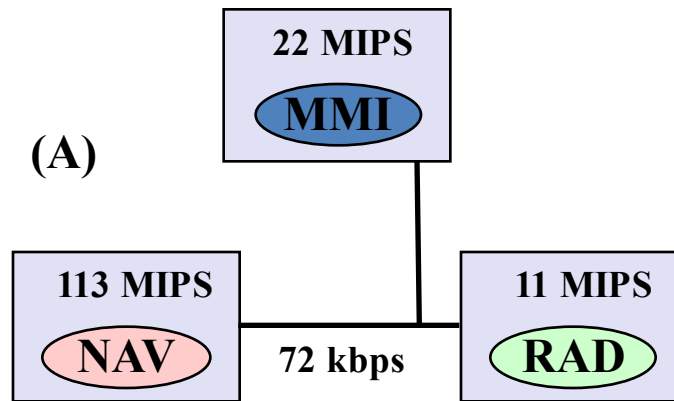
© Thiele, ETHZ

- End-to-end delays:



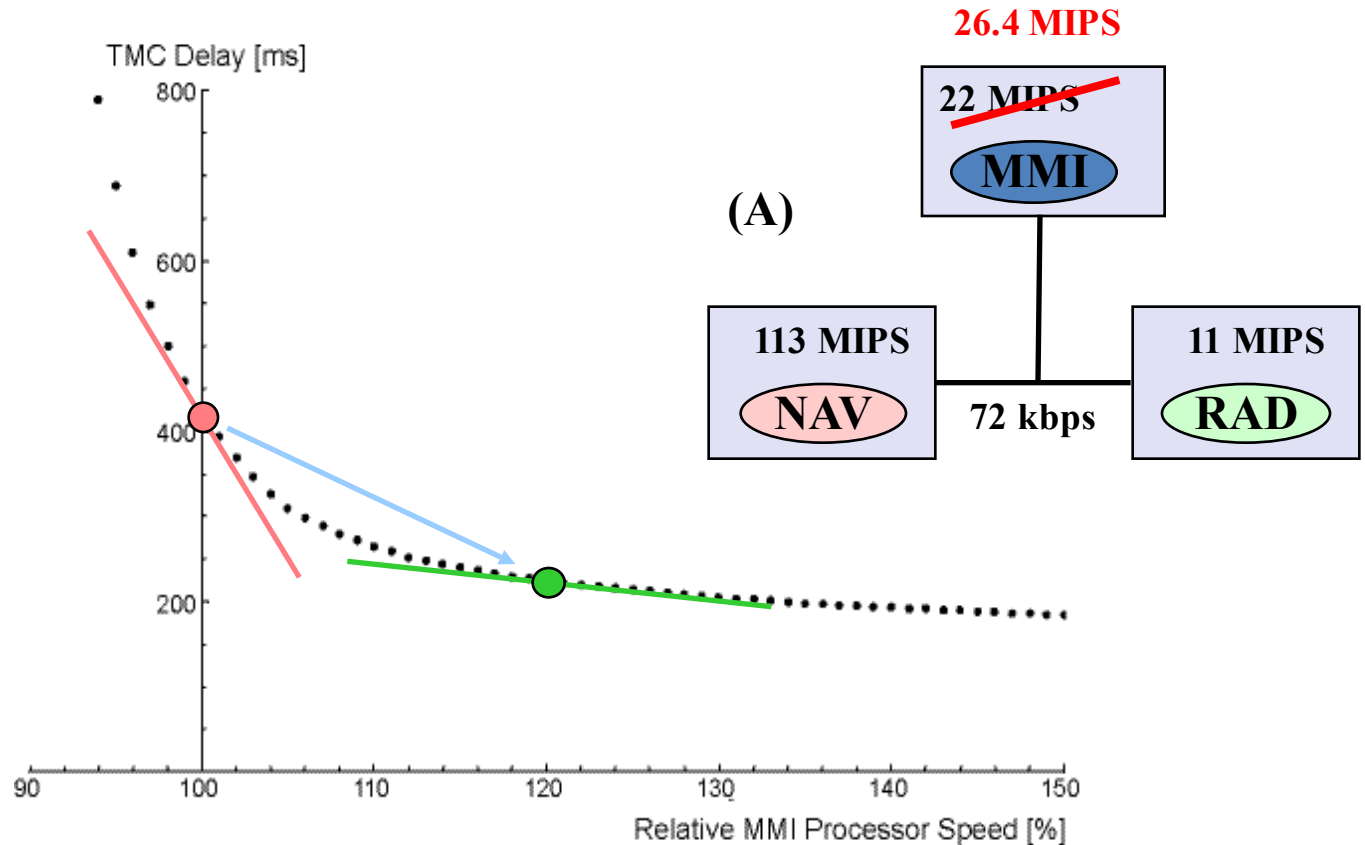
Analysis – Design Question 2

- How robust is architecture A?
- Where is the bottleneck of this architecture?



Analysis – Design Question 2

- TMC delay vs. MMI processor speed:



© Thiele, ETHZ

RTC/MPA: Conclusions

- Easy to construct models (~ half day)
- Evaluation speed is fast and linear with model complexity (~ 1 s per evaluation)
- Needs very little information to construct early models
 - E.g., message sequence charts
 - ⇒ Fits early design cycle very well
- While the mathematics are very complex
 - The method is easy to use
- Download the free Matlab Toolbox:
 - <http://www.mpa.ethz.ch/Rtctoolbox/Overview>

© Thiele, ETHZ

Summary

- Evaluation and Validation
 - In general, multiple objectives
 - Pareto optimality
 - Design space evaluation (DSE)
- Performance analysis
 - Trade-off between speed and accuracy
- Computation of worst case execution times
 - Cache/pipeline analysis
 - ILP model for computing application WCET from BB WCET
- Real-time Calculus
 - Mathematical models of event arrival, service time, and available resources
 - Combine models to derive system properties

Next Time

- Energy- and power estimation
- Modeling thermal behavior
- Reliability
- Validation
- Chapter 5.3-8