MÁSTER UNIVERSITARIO EN SISTEMAS ESPACIALES
POR LA UNIVERSIDAD POLITÉCNICA DE MADRID

POLITÉCNICA

2020.04.20

Data handling
Real-time systems
Real-time analysis

Juan Antonio de la Puente juan.de.la.puente@upm.es

STRAST

Real-time requirements for tasks

T = period
D = deadline
C = processor time
R = response time

- Activation pattern
  - periodic: jobs are activated with a period *T*
  - sporadic: events occur with a minimum inter-arrival time *T*
  - other: aperiodic, random, etc.
- Deadline
  - each job has a deadline *D* relative to its release time
- Computation time
  - each job consumes a certain amount *C* of processor time
- The main real-time requirement is that for all jobs of every task the response time $R \leq D$
  - $C \leq R$ is implied

R (RESPONSE) <= D (DEADLINE)

C (processor time) <= R (RESPONSE)

Validating real-time requirements

- High-integrity real-time systems are required to have a guaranteed real-time behaviour in all operating conditions
  - including worst-case situations
- Tests are not enough
  - how can we be sure that the worst case is covered?
- Analytical methods are preferred
  - analyse timing behaviour from the program text
  - do not execute the program (static analysis)

Single task systems

R (RESPONSE) <= D (DEADLINE)

C (processor time) = R (RESPONSE)
<-- because there's no other task

WORST CASE:
R = C =WCET (Worst-Case Execution Time)

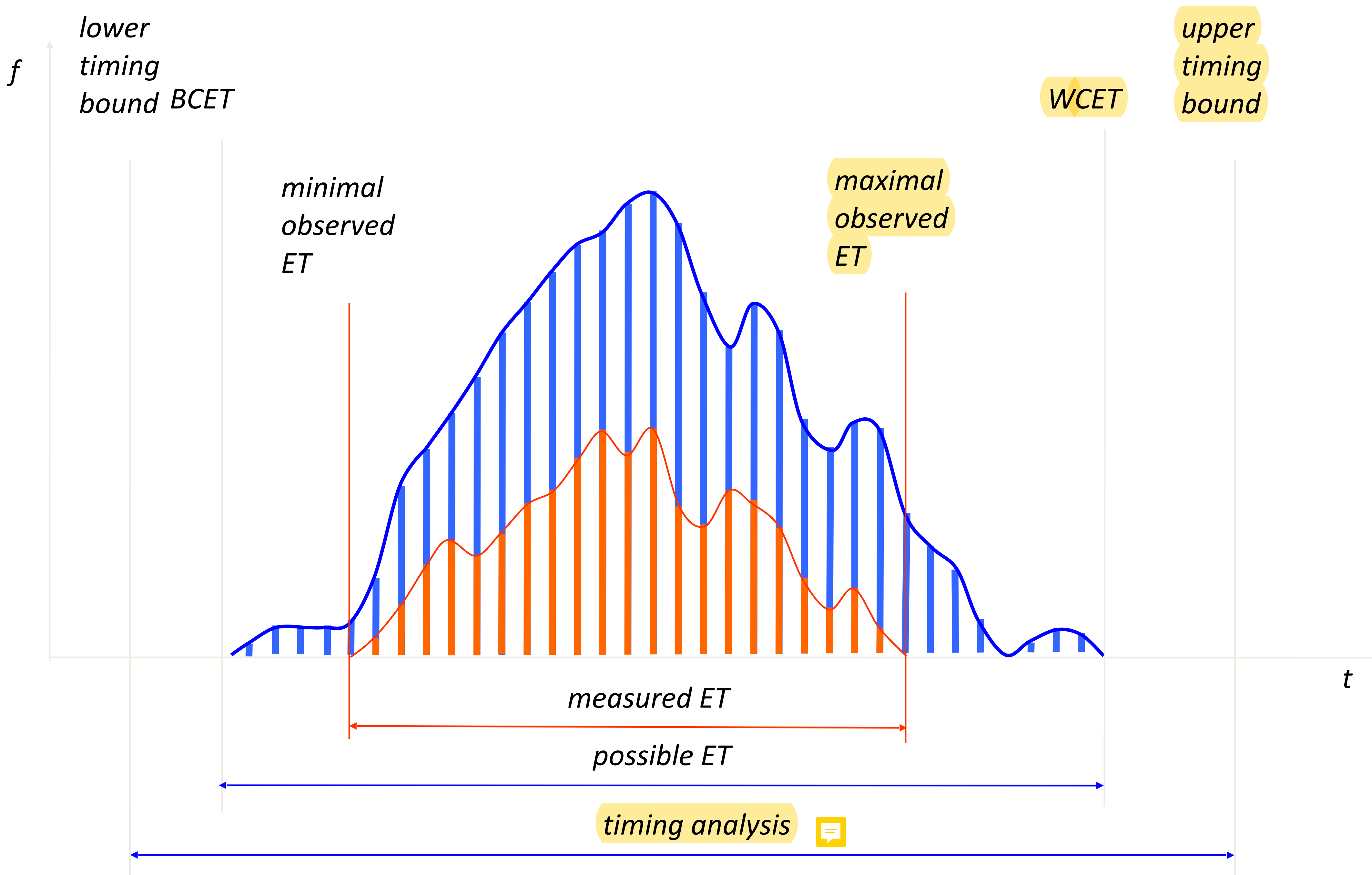
- Start with a simple system with only one task
  - no need for OS
- Feasibility condition
  - $R \leq D$ (but $R = C$ as there are no other tasks)
- The worst case is then
  - $R = C = WCET$

  where WCET is the worst-case execution time of all the task jobs
  - notice that the execution time may vary from one job to another

Calculating WCET

- The execution time of a code segment may be variable due to
  - program structure: different execution paths
    - e.g. if-then-else, loops, etc.
  - hardware acceleration mechanisms
    - e.g. caches, pipelines, speculative branching
- Calculate an estimate of WCET that is
  - pessimistic
    - to be sure that the real WCET is covered
  - tight
    - to use efficiently processor time

Execution time values



WCET estimation

- Static analysis
  - build execution graph, find longest path, add C for elementary blocks
  - requires a model of the processor hardware
- Dynamic analysis
  - measure execution time of running software
  - not sure that worst-case is covered
- Hybrid approach
  - find longest paths by analysis and measure execution time
  - many runs required to ensure that WCET is covered

Response time analysis

- The execution of a task suffers interference
  - pre-emption from higher priority tasks
- For FPPS/ICPP, the response-time equation for a task τ_i is bounded by

$$R_i = C_i + B_i + \sum_{j \in \mathrm{hp}(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j$$

  where

  - R_i is an upper bound of the response time of τ_i
  - C_i is the worst-case execution time of τ_i
  - B_i is the worst-case blocking time of τ_i
  - T_i is the period (or minimal separation) of τ_i
  - hp(i) is the set of all tasks having a higher priority than τ_i
- The equation can be solved by linear iteration

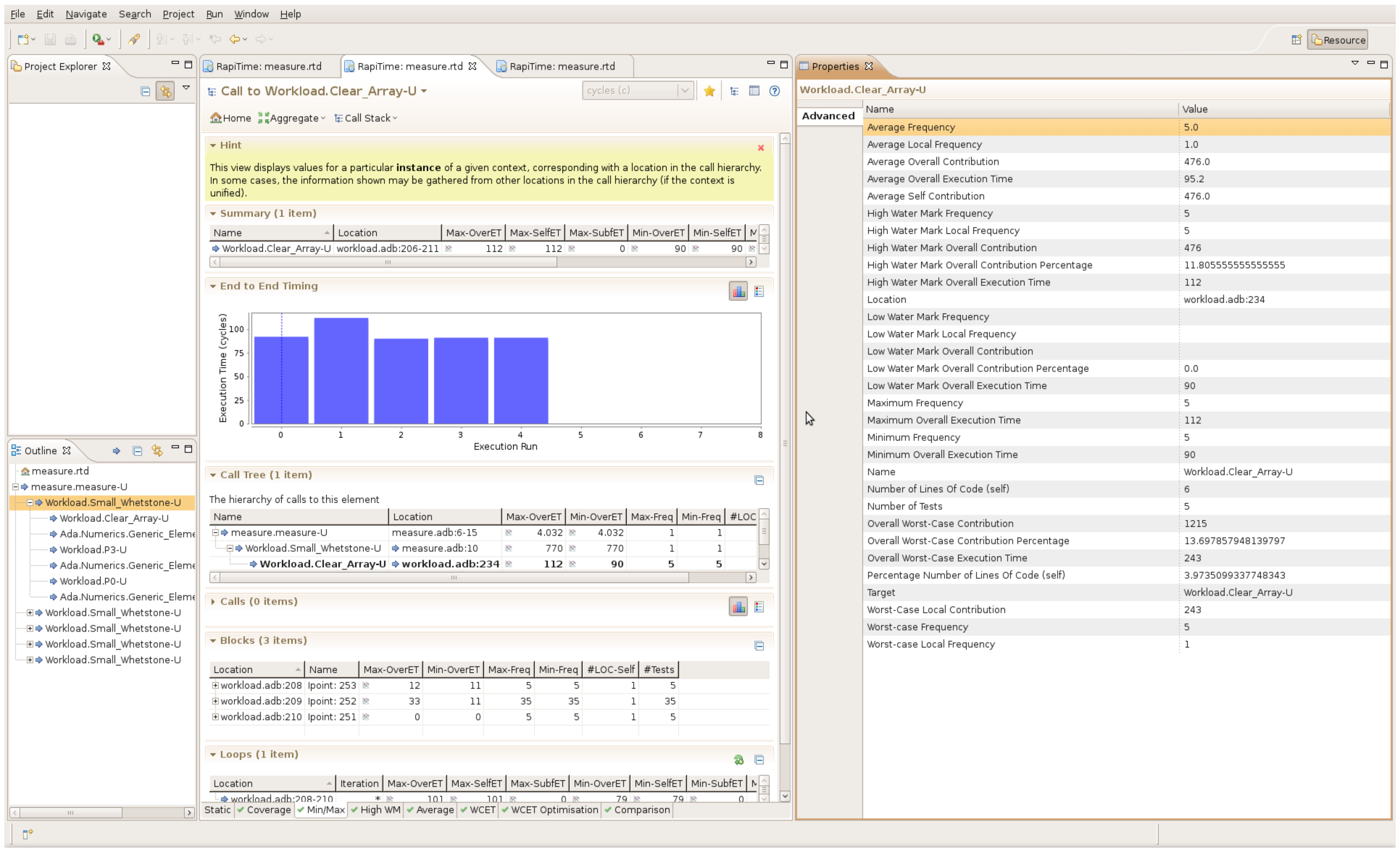
The Ravenscar computational model

- Restrictions on the structure of the program in order to enable RTA
  - static set of tasks
  - inter-task communication restricted to shared data with mutually exclusive access
    - no conditional synchronisation
  - FPPS + ICPP
- Further restrictions to enable WCET estimation
  - e.g. no dynamic storage
- Can be checked at compilation time

```
pragma Profile(Ravenscar);
```

Tools

- WCET analysis: RapiTime



- RTA / MAST