Under the hood: run-time challenges for time predictable computing in the multicore era

IWES
20 September 2016

Tullio Vardanega
Department of Mathematics
University of Padova, Italy

UNIVERSITÀ DEGLI STUDI DI PADOVA

Inspiring principles /1

- *Separation of concerns*
  - Sharply separate different aspects of SW design and implementation
    - *Vertical* concerns vs *horizontal* concerns
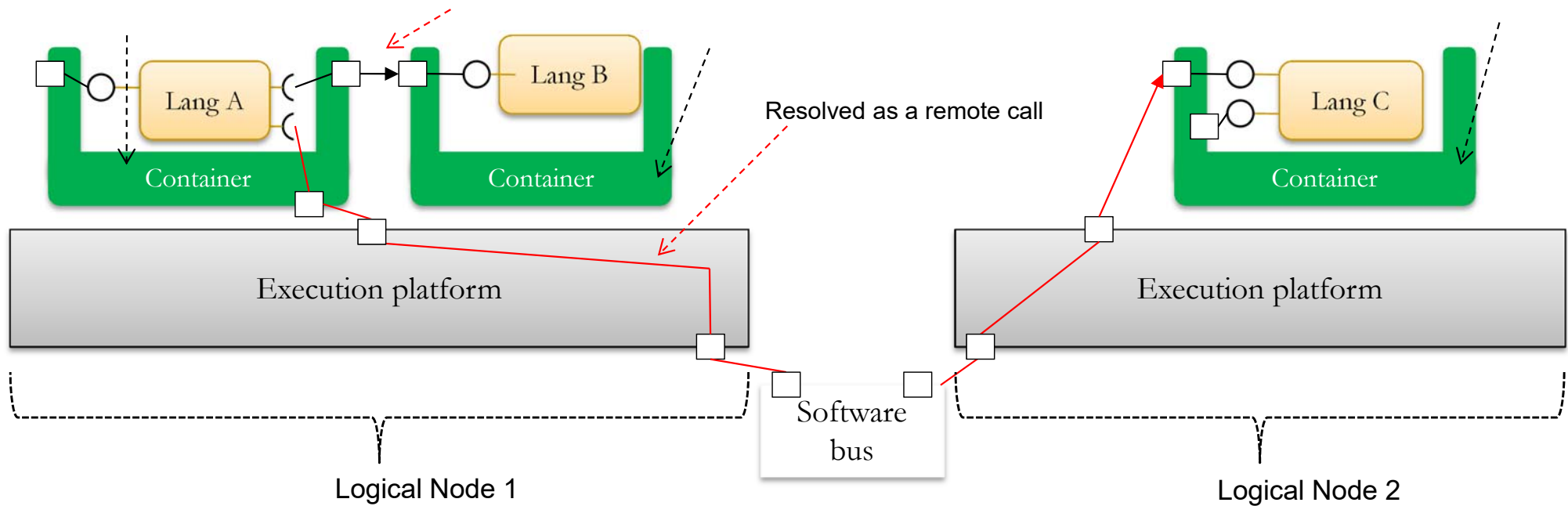   - Each development actor to focus on their own area of expertise
    - For each concern use the best-fit formalisms, tools and verification techniques
  - Achieved using
    - *Design views*
    - *Component model* with containers and connectors

Component model

Resolved as a remote call

Lang A

Lang B

Lang C

Container

Container

Container

Execution platform

Execution platform

Software bus

Logical Node 1

Logical Node 2

Inspiring principles /2

- *Correctness by construction*
  - ❑ Horizontal concern
    - ■ Component assembly follows *interface contracts*
  - ❑ Vertical concern
    - ■ Design-time binding to *models of computation* that have desirable properties

  - ❑ Resist shopping-mall-style feature inflation

Inspiring principles /3

- Seek and embrace *predictable concurrency*
  - Solved problem for single-core processor systems
- Wed high utilization with low-overhead runtimes (for preemptions and migrations)
  - A tough challenge for multicore processor systems
  - Solutions exist but not really in the mainstream
- Devise models of *time-predictable parallelism*
 - An open problem

Inspiring principles /4

- Assure *consistence* between (timing) analysis models and analysed reality

- At the application level you often assume a lot and know little

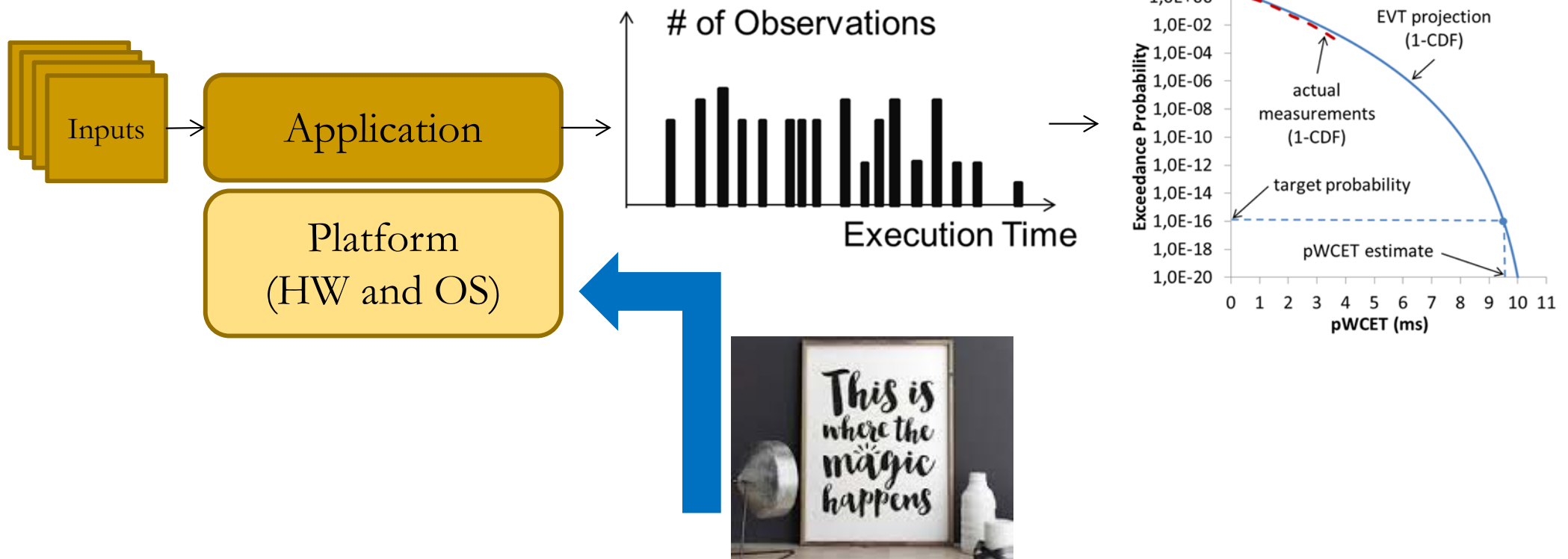 - Preservation of stipulations should happen *under the hood*

    - HW, RTOS, MW

What you assume

What you know

Timing analysis methods

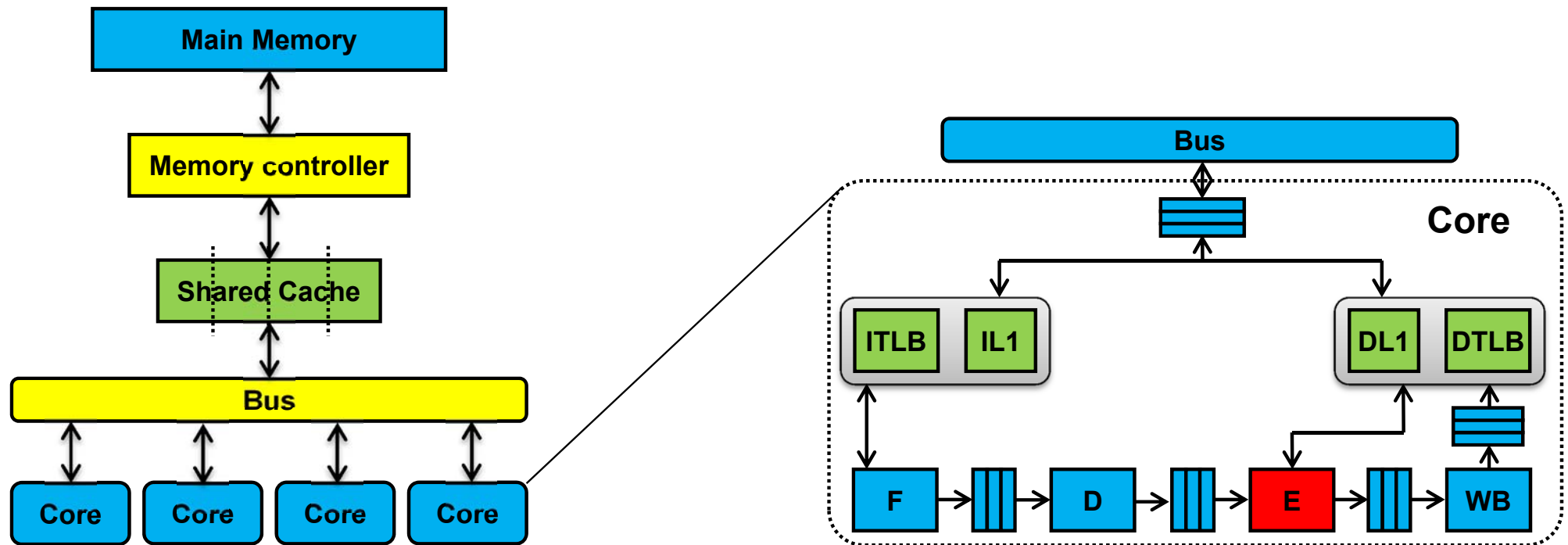Confidence in the results
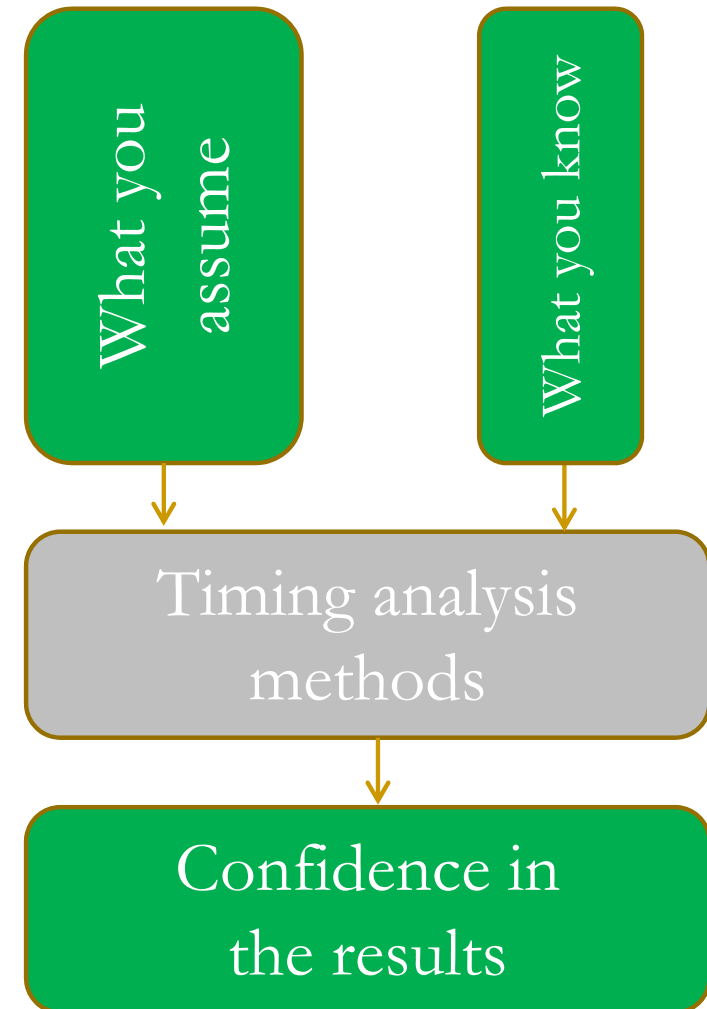
Inspiring principles /5

- *Renounce* strict determinism when achieving it throughout the execution yields inordinately pessimistic worst-case bounds
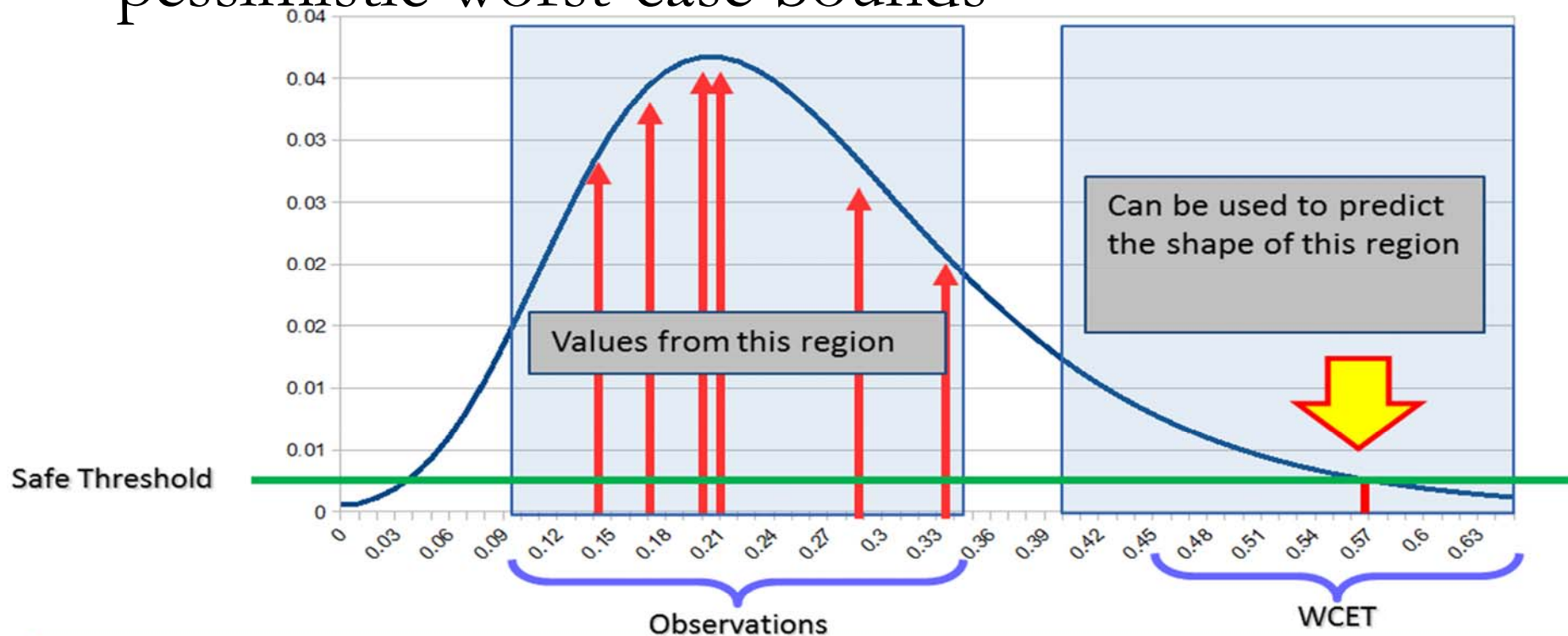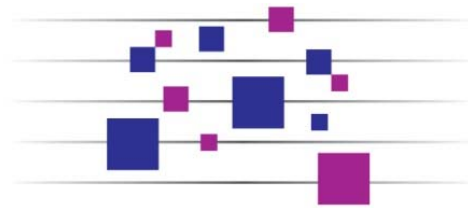
Magic must happen under the hood

What kind of magic?

- *Time randomize* selected processor resources
  - In HW if you can, via transparent SW otherwise
- Make the RTOS operation *time composable* against the application
  - So that measurement observations at application level are not disturbed

Palaestras