Basic Concepts on RTS

Real-Time Operative Systems Course

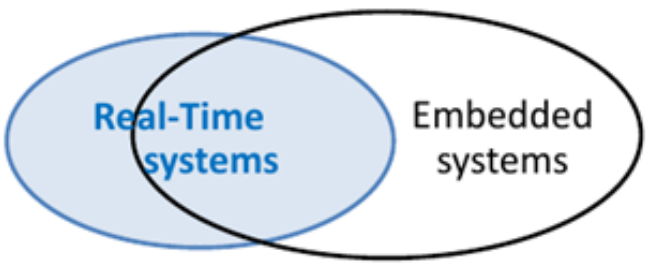
Paulo Pedreiras, DETI/UA/IT

September 21, 2022

Embedded vs Real-Time systems

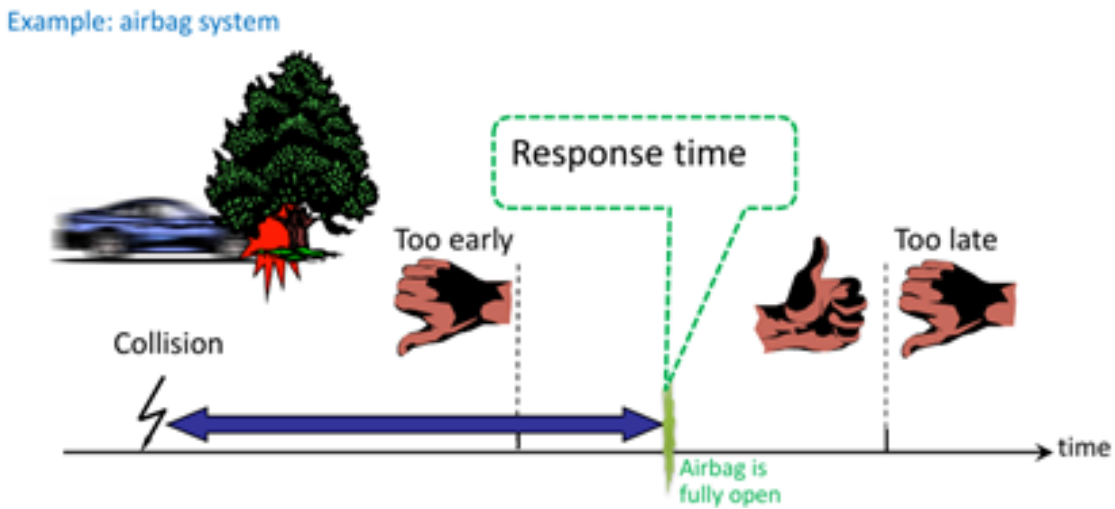
Real-Time and Embedded Systems are often confused. **But they are not the same ...**

- Real-time And Embedded:
  - Nuclear reactor control, Flight control, Mobile phone
- Real-time, but not embedded:
  - Stock trading system, Video streaming/processing
- Embedded, but not real-time:
  - Home temperature control, Washing machine, refrigerator, etc.



A few definitions related with "Real-Time"

- There is a wide variety of definitions related to Real-Time, systems dealing with Real-Time and the services that they provide.
- All of these definitions have in common the fact that express the **dependency of a computer system on the time, as it exists in a particular physical process** .



Definitions related with "Real-Time"

- Real-Time Service or Function
  - Which must be performed or provided within finite time intervals imposed by a physical process
- Real-Time System
  - One who contains at least one feature of real-time or at least providing a service of real-time
- Real-Time Science
  - Branch of computer science that studies the introduction of Real-Time in computational systems.

Real-Time Computation

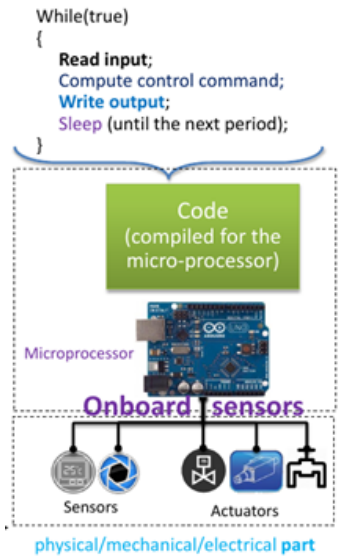
- The computation results must be
  - Logically correct
  - Delivered on time

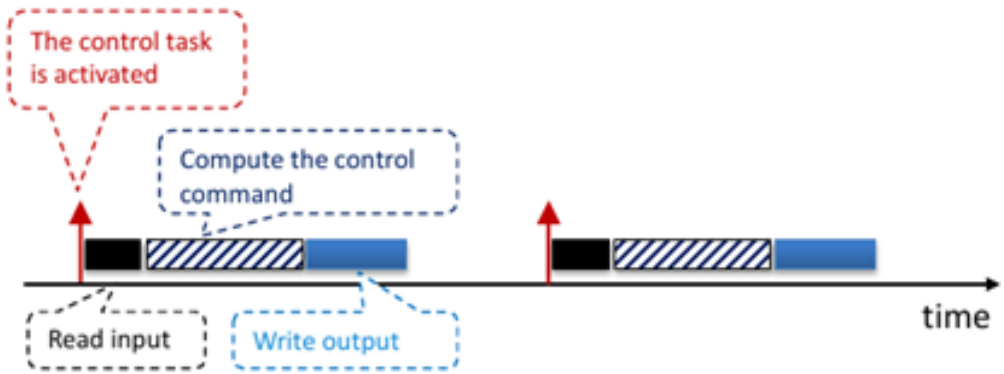
(Stankovic, 1988)

Timeliness  Logic Correction

Real-Time systems - implementation

- Simple case, with an infinite loop:

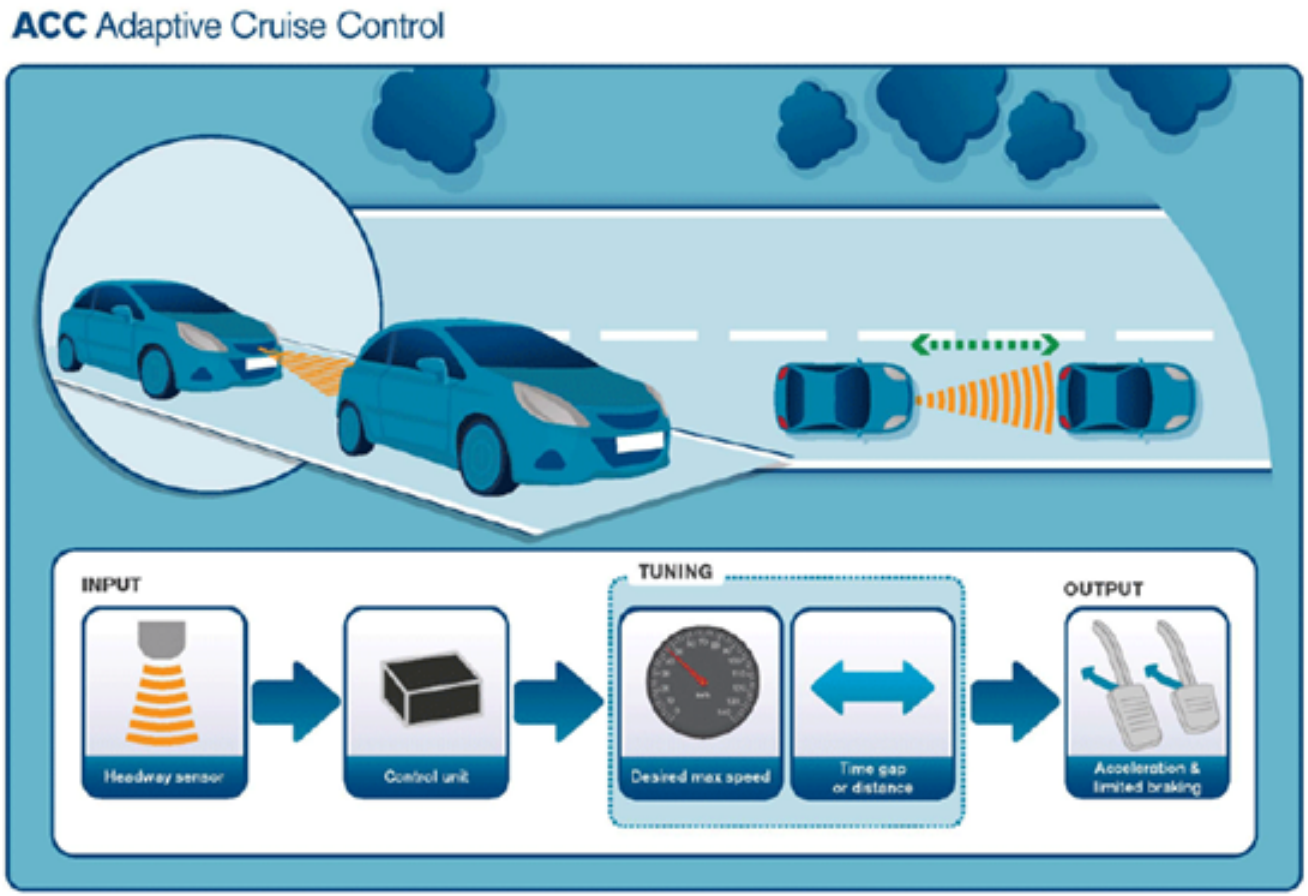




- Looks simple and predictable, right?

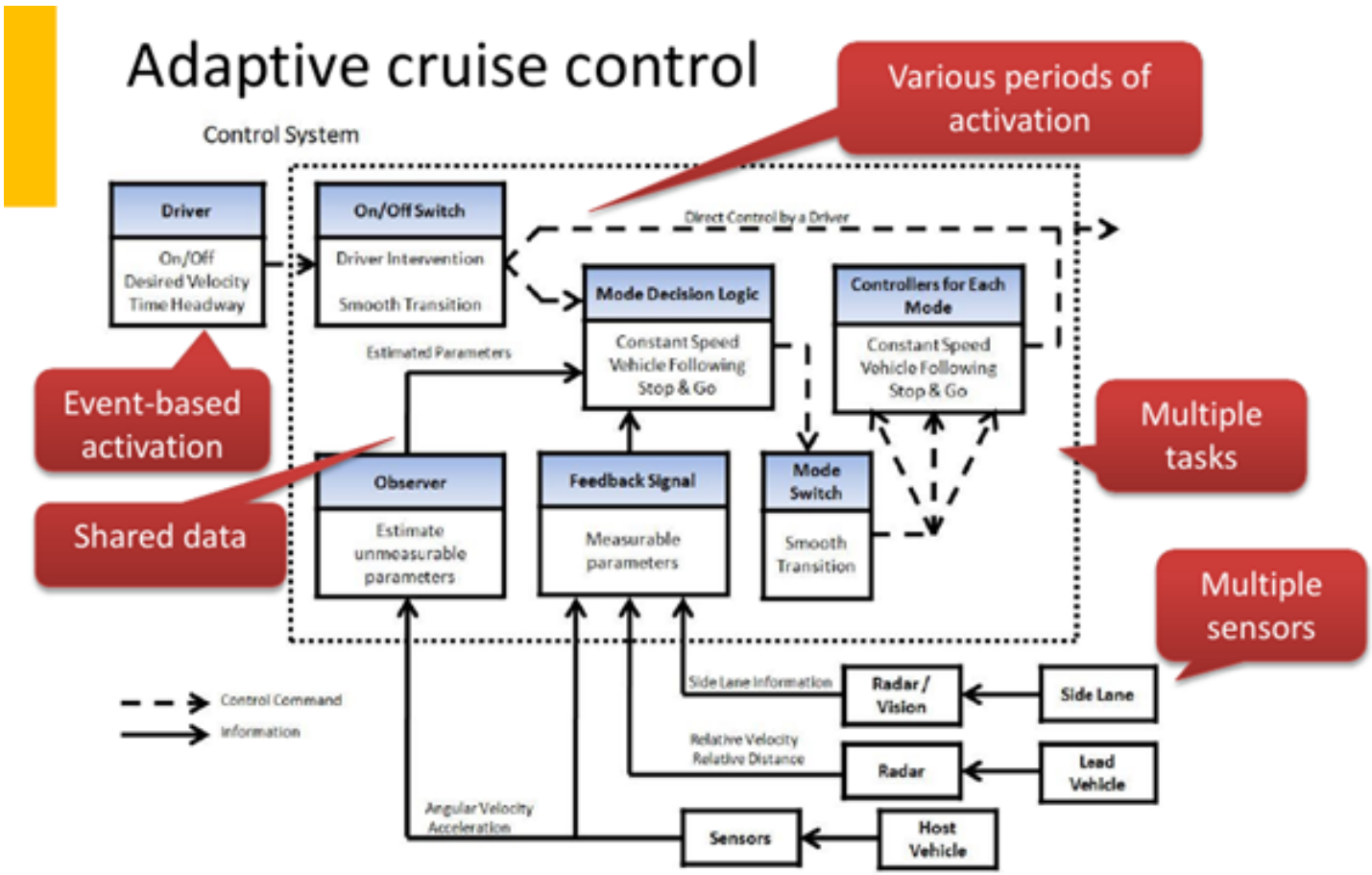
Real-Time systems - implementation

- But it can get **much more** complicated ...



Real-Time systems - implementation

- Does is still look simple and predictable?



https://codingcommunity.de/jaguar-cruise-control-diagram.html

Objective of the study of RTS

- Generally, when a computer system monitors the state of a given physical process and, if necessary, acts on it in time, then it is a real-time system.
- All living beings are real-time in relation to their natural habitats, which are its "real-time system".
- On the other hand, when we build (programmable) machines to interact with physical processes, we need to use Operative Systems, Programming Techniques and Analysis Methods that allow us to have confidence in its ability to carry out **correct and timely actions** .

Objective of the study of RTS

- The main objective of the study of Real-Time Systems is thus the development of techniques for
  - Design,
  - Analysis, and
  - Verification

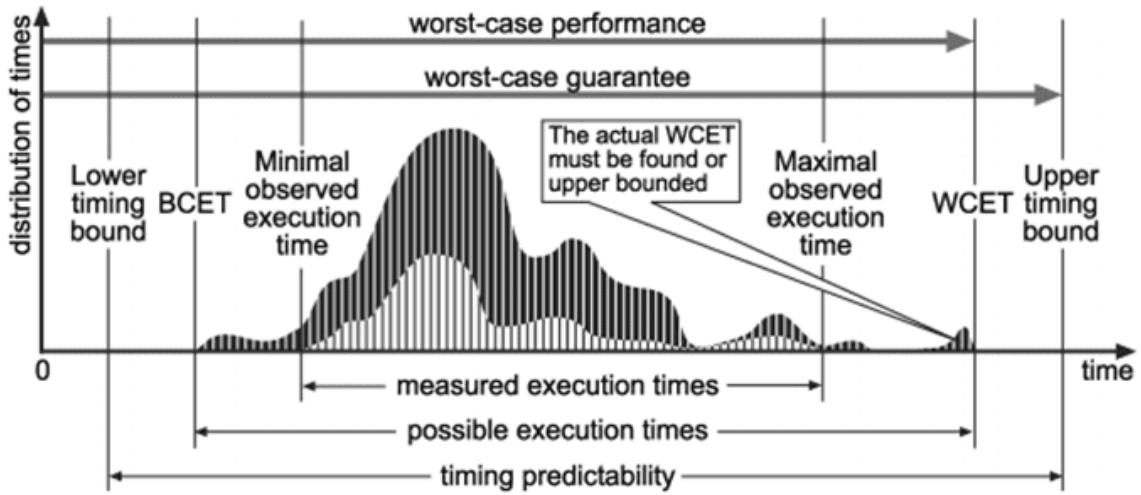
  that allow to assure that a given (computer, in our case) system has an appropriate timing behavior, satisfying the requirements imposed by the dynamics of the system with which it interacts

Objective of the study of RTS

- Regarding the computational activities of RTS, the main aspects to consider are:
  - Execution time
  - Response time
  - Regularity of periodic events

Objective of the study of RTS

Despite essential, these aspects are far from trivial:



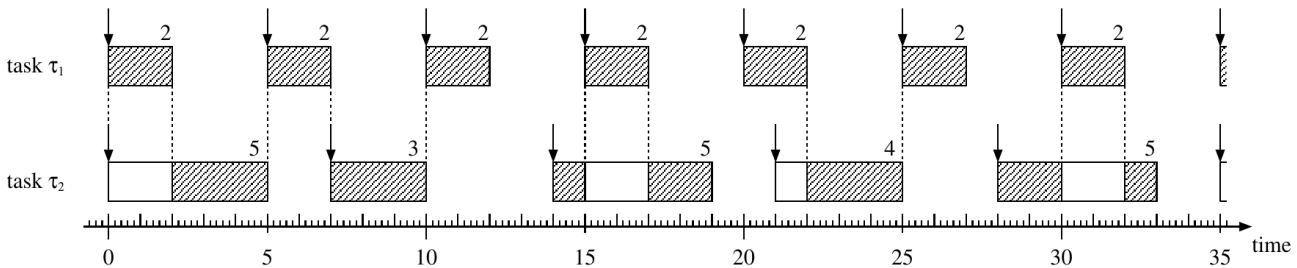
Analysing Extreme Value Theory Efficiency In Probabilistic Wcet Estimation With Block Maxima, ISSN:2448-0959

Reasons for this behaviour include:

- Execution time
  - Code structure (language, conditional execution, cycles)
  - DMA, cache, pipeline
  - Operative System or kernel (system calls)

Objective of the study of RTS

- Response time and regularity
  - Interrupts
  - Multi-tasking
  - Access to shared resources (buses, communication ports, ...)



Impact of interference on the Worst-case Response Time

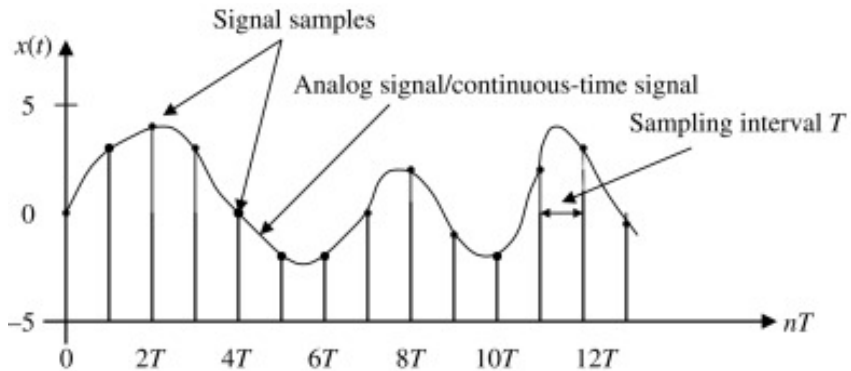
"Worst-case response time analysis of real-time tasks under fixed-priority scheduling with deferred preemption revisited", Reinder J. Bril et al

Requirements of Real-Time Systems

- The requirements commonly imposed to real-time systems are of three types:
  - Functional
  - **Temporal** (our main focus)
  - Dependability

Functional requirements

- Data gathering
  - Sampling of system variables (real-time entities), both analog and discrete
- Supervise and Control
  - Direct access to sensors and actuators
- Interaction with the operator
  - System status information, logs, support to correct system operation, warnings, ...

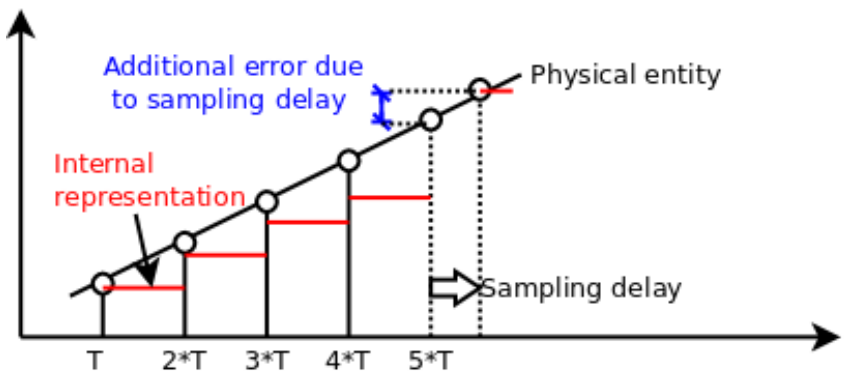


Periodic sampling illustration

Functional requirements

Data gathering

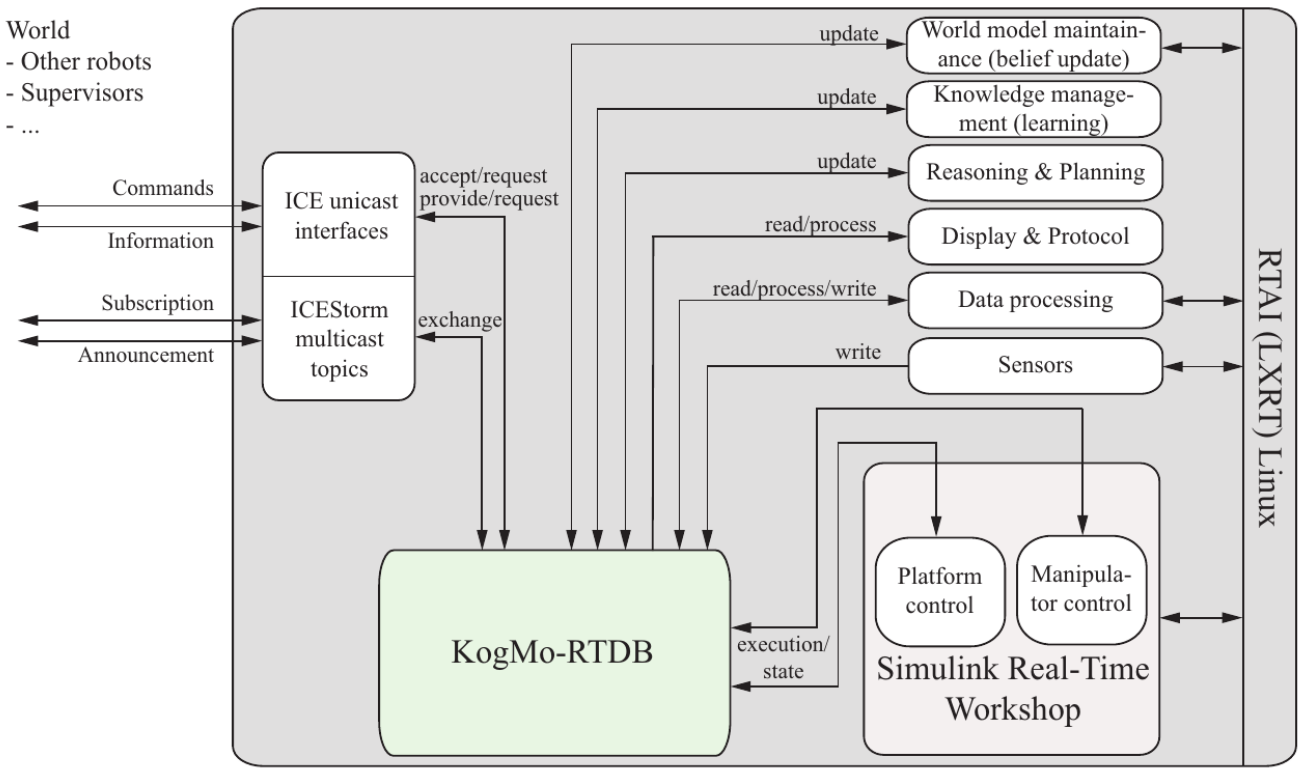
- The Real-Time computer operates on **local images** (internal variables) that represent the physical entities
- Each image of a real-time entity has a limited time validity, due to the temporal dynamics of the physical process
- The set of images of the real-time entities forms a **Real-Time Database**
- The real-time database must be updated to keep consistency between the physical world and its the internal representation



Impact of time delay in sampling error

Functional requirements

Illustration of a RTDB for Multi-Robot Systems



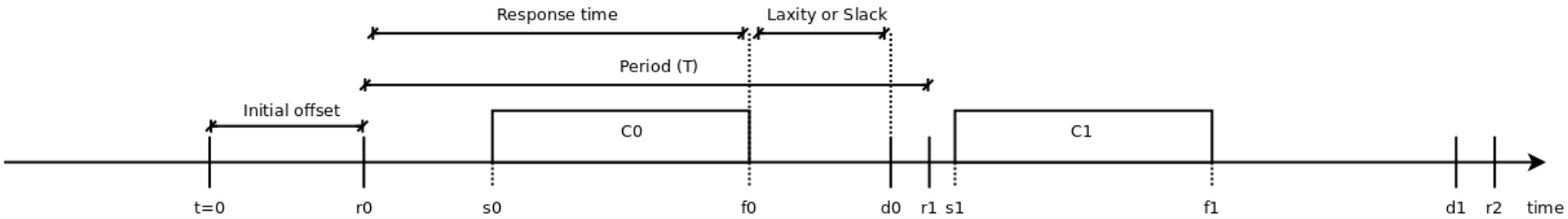
An Architecture for Real-Time Control in Multi-robot Systems, DOI:10.1007/978-3-642-10403-9_5

Temporal requirements

- Usually arise from the physical dynamics of the process to be managed or controlled
- Impose restrictions:
  - Delays the observation of the system state
  - Delays computing the new control values (acting)
  - Variations of previous delays (jitter)
- that must be followed in **all instances** (including the worst case) and not only on average

Temporal requirements

Terminology (1/2)



Initial offset (ϕ) : time before the first release/activation (job) of a task.

Period (T) : time between successive jobs of a task. Can be a Minimum Inter-Arrival time (MIT) for sporadic tasks.

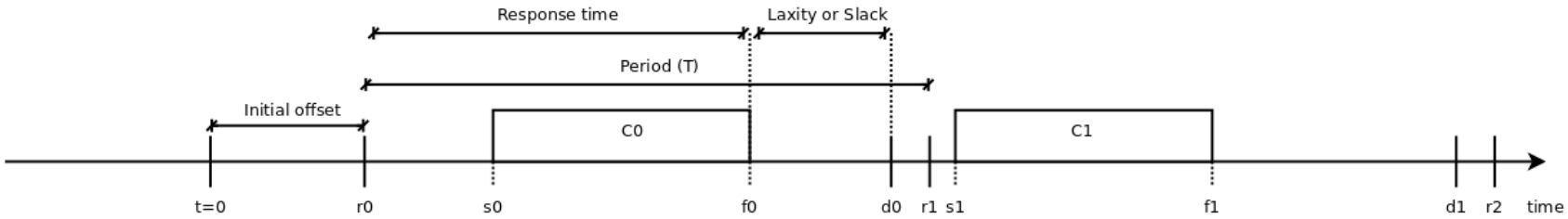
Release/activation/arrival time (r_i, a_i) : time instant of the i_{th} job of a task. $r_i = \phi + k \cdot T_i$ for periodic tasks.

Start time (s_i) : time at which the i_{th} job of a task start executing.

Finish/completion time (t_i) : time instant in which the i_{th} job of a task terminates.

Temporal requirements

Terminology (2/2)



Execution/computation time (C_i) :time necessary to the processor for executing the task instance without interruption.

Absolute deadline (d_i) : time instant by which the i_{th} execution of a task must complete

Response time (R_i) : time elapsed between the release of the i_{th} job of a task and its completion ($R_i = f_i - r_i$)

Slack/Laxity (L_i) : maximum time a task can be delayed on its activation to complete within its deadline ($L_i = d_i - r_i - C_i$)

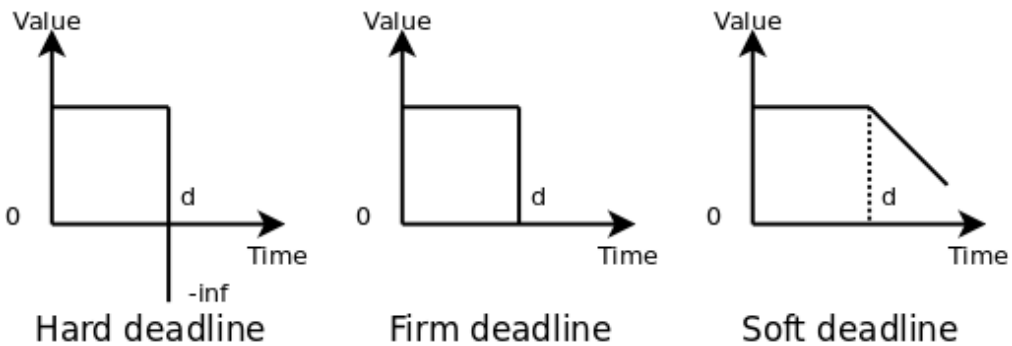
Temporal requirements

Classification of the temporal constraints, according with the **usefulness** of the result:

Soft: temporal constraint in which the result retains some utility to the application, even after a temporal limit D, although affected by a degradation of quality of service.

Firm: temporal constraint in which the result loses any usefulness to the application after a temporal limit D.

Hard: temporal restriction that, when not met, can lead to a catastrophic failure.



Temporal requirements

Classification of the Real-Time Systems, according with the **temporal** constraints:

Soft Real-Time: The system only has **firm or soft** real-time constraints (e.g., simulators, multimedia systems)

Hard Real-Time The system has **at least one hard real-time constraint** . These are the so-called safety-critical systems (e.g. airplane control, missile control, nuclear plants control, control of dangerous industrial processes)

Best Effort: The system **is not subject** to real-time constraints

Dependability requirements

- Real-time systems are typically used in critical applications, in which failures may endanger human lives or result in high economic impact/losses.
- This results in a requirement of **High Reliability**
  - Hard real-time systems have typically ultra-high reliability requirements ($< 10^{-9} failures/hour$)
  - Cannot be verified experimentally!
  - Validation requires solid analytic support (among other aspects)

Dependability requirements

- Important aspect to consider in safety-critical systems:
  - **Stable interfaces** between the critical and the remaining subsystems, in order to avoid error propagation between each other.
  - **Well defined worst case scenarios** . The system must have an adequate amount of resources to deal with worst case scenarios without resorting to probabilistic arguments, i.e. must provide service guarantees even in such scenarios.
  - Architecture composed of autonomous subsystems, whose properties can be checked independently of the others (**composability**).

Summary

- Notion of real-time and real-time system
- Antagonism between real-time and best effort
- Objectives of the study of RTS – how to guarantee the adequate temporal behavior
- Aspects to consider: execution time, response-time and regularity of periodic events
- Requirements of RTS: functional, temporal and dependability
- Constraints soft, firm and hard, and hard real-time vs soft real-time
- The importance of consider the worst-case scenario