Embedded Software Engineering

3 Unit Course, Winter 2010

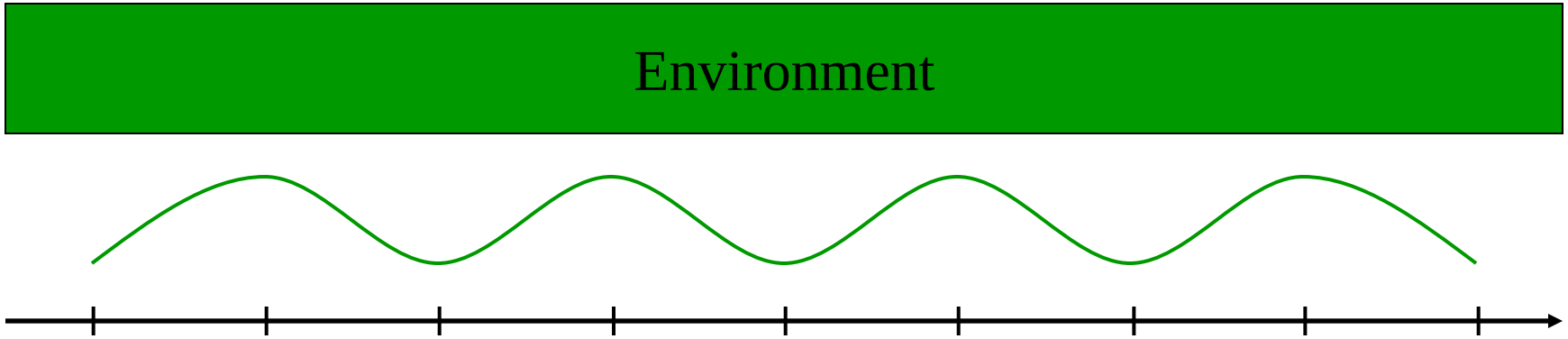
CS Department, Univ. of Salzburg

RT Scheduling

Christoph Kirsch

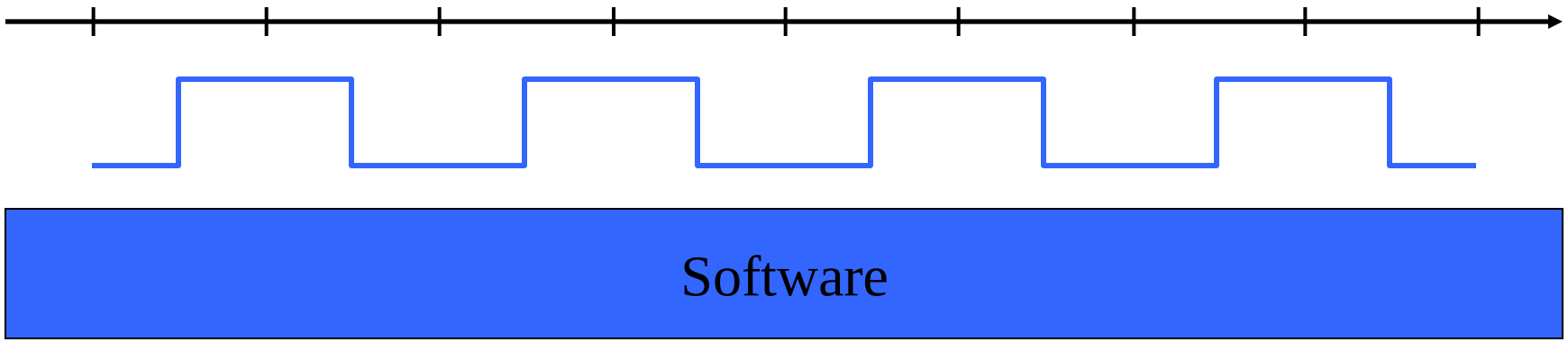
www.cs.uni-salzburg.at/~ck/teaching/ESE-Winter-2010

Platform Time is Platform Memory

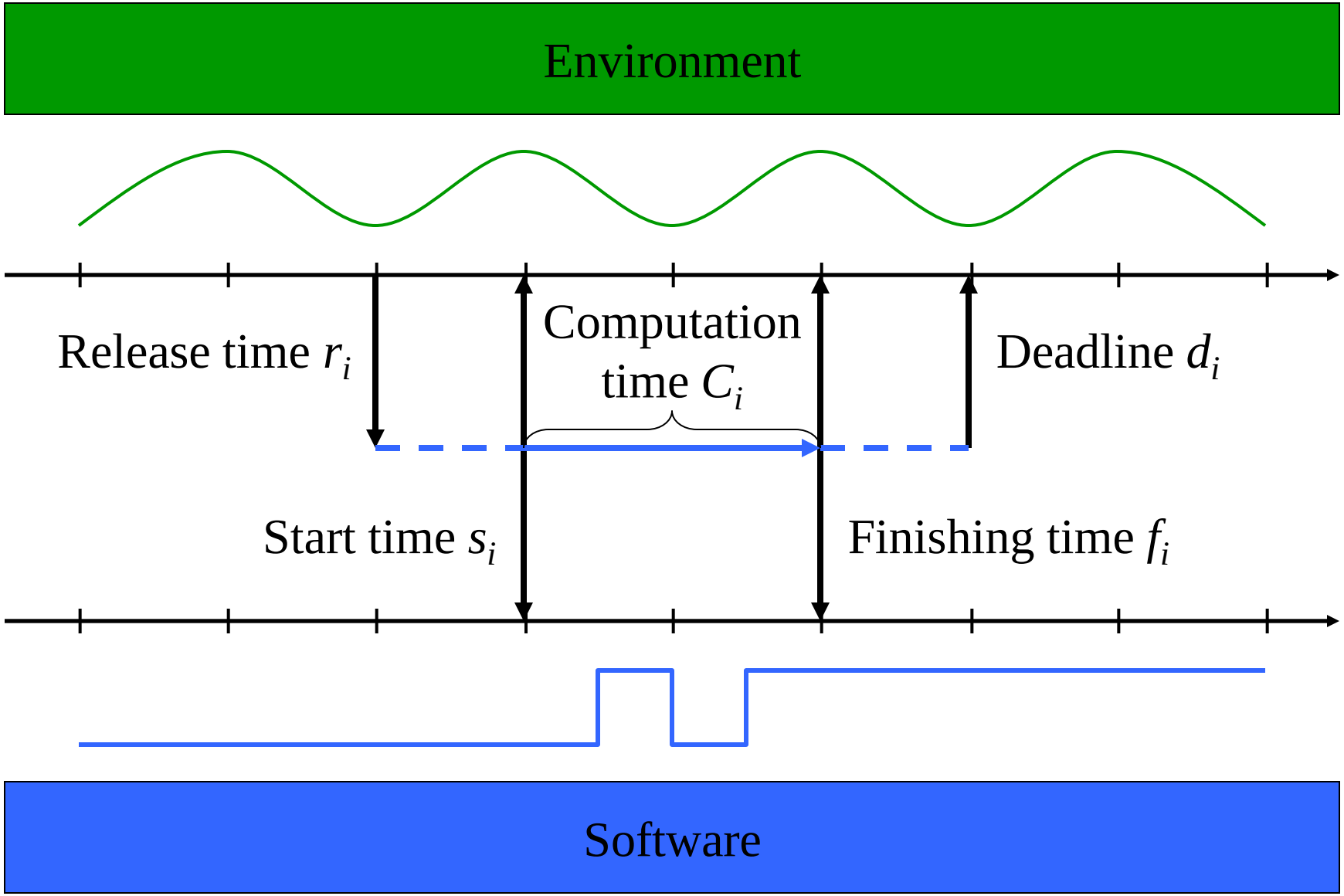


- Programming as if there is enough platform time

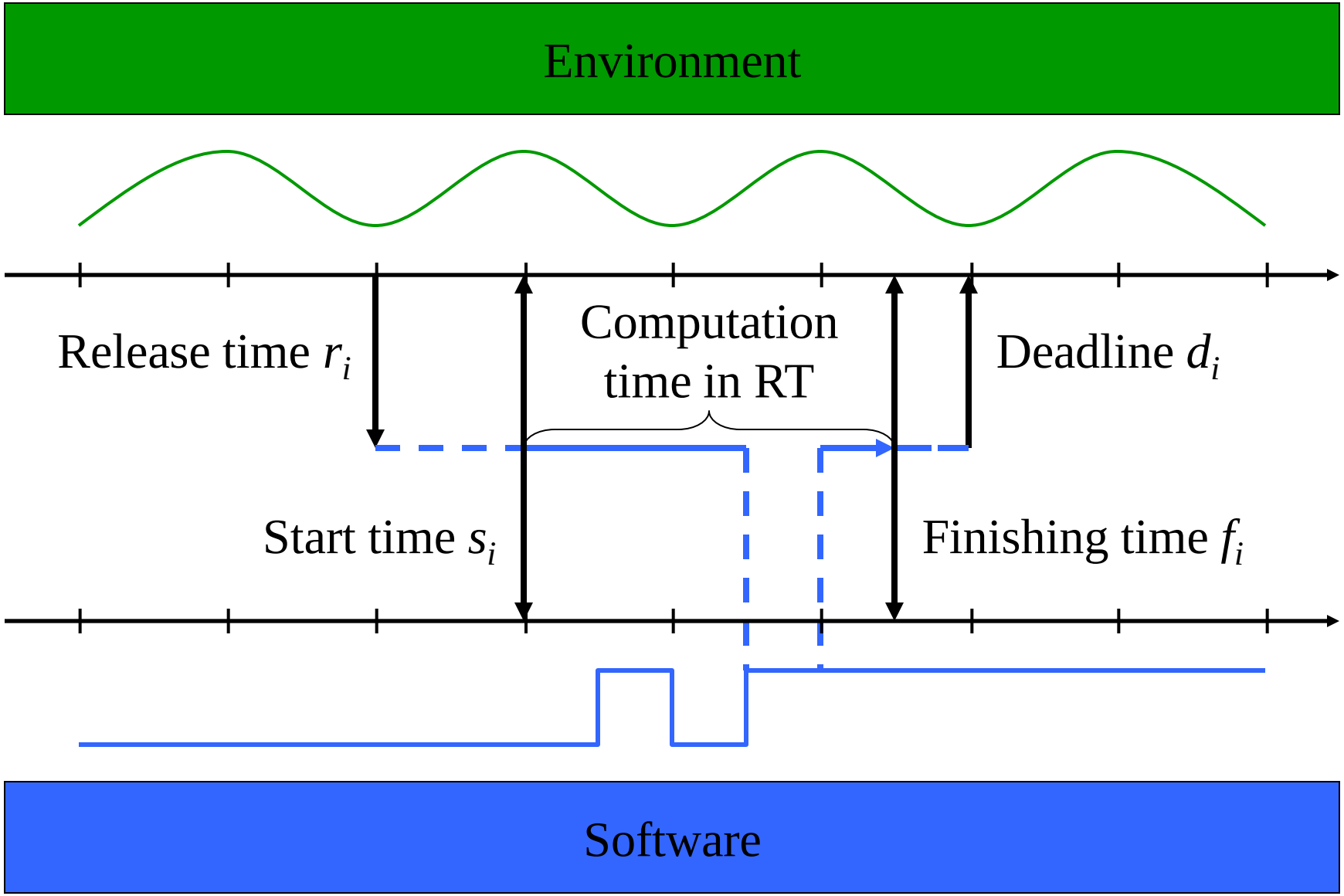
- Implementation checks whether there is enough of it



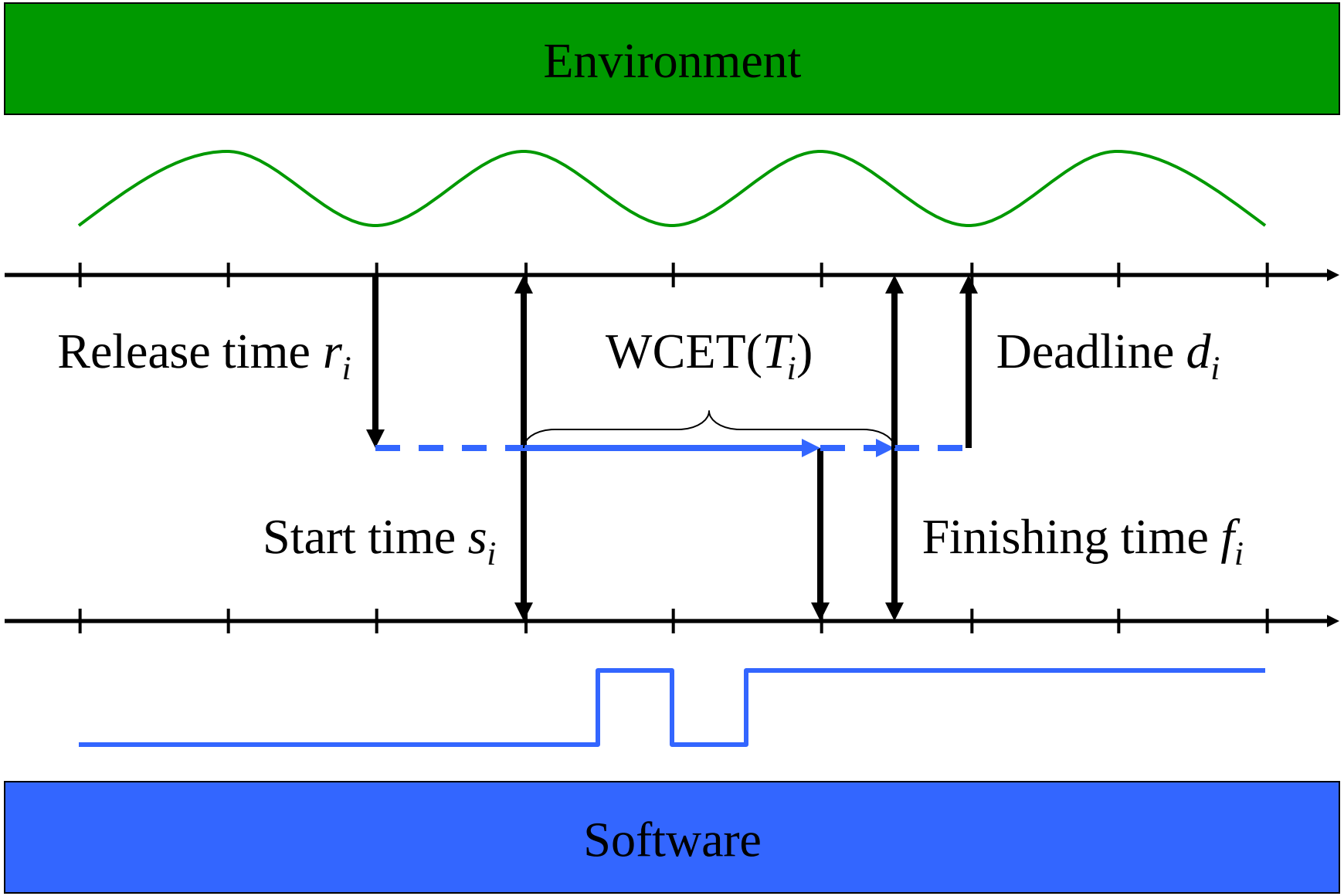
A Task T_i


Environment
Release time r_i
Computation time C_i
Deadline d_i
Start time s_i
Finishing time f_i
Software

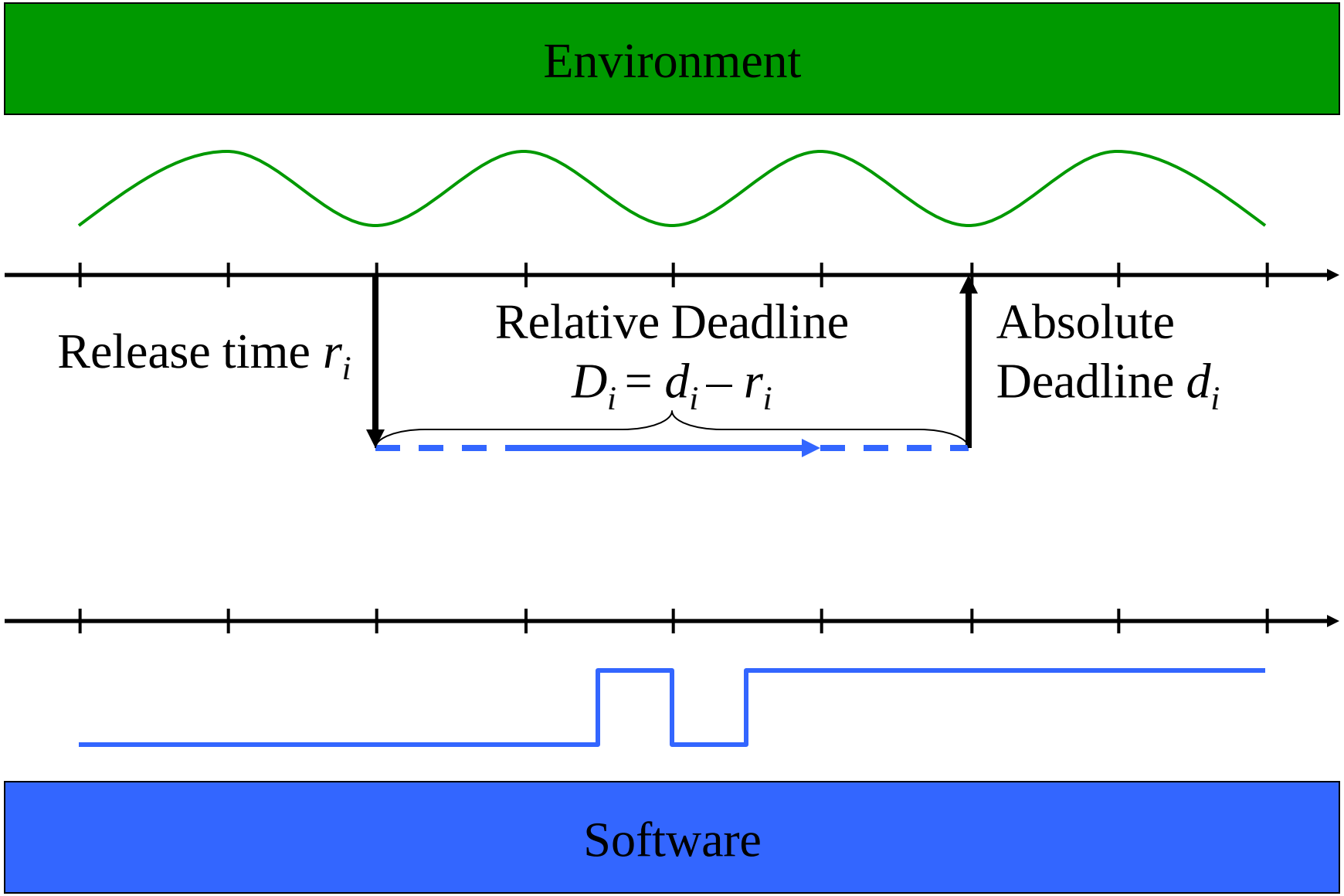
Preemption


Environment
Release time r_i
Computation time in RT
Deadline d_i
Start time s_i
Finishing time f_i
Software

Worst-Case Execution Time: WCET(T_i)



Relative Deadline D_i



Some Vocabulary for a Task T_i

- *Lateness*: $L_i = f_i - d_i$ is the delay of T_i's completion with respect to its deadline; negative L_i mean early completion

- *Laxity* (*Slack time*): $X_i = D_i - C_i$ is the maximum time T_i can be delayed on its start to complete within its deadline

Triggering a Task T_i

- *Periodically*: A *periodic task* T_i is a task with a-priori known release times regularly activated at a constant rate P_i
  - The first release time r_i is called the *phase* ϕ_I
  - The release time of the n-th instance is given by $r_i + (n - 1)\, P_i$
  - P_i is called the *period* of T_i
- *Sporadically*: A *sporadic task* T_i is a task with a minimum (*interarrival*) time between any two release times
- *Aperiodically*: An *aperiodic task* T_i is a task without any constraints on the release times

Definition: Schedule

- A *schedule* for a set T of tasks and a set S of *shared resources* is a function that maps a shared resource $s \in S$ for any given (discrete) time instant to a possibly empty subset of T (Non-Determinism)

- A *feasible* schedule is a schedule in which each task can complete within its deadline

Schedulability Test vs. Scheduling Algorithm

- A *schedulability test* determines the existence of a feasible schedule for a given set of tasks and shared resources
- A schedulability test can be an *exact*, *sufficient*, or *necessary* condition for the existence of a feasible schedule

---

- A *scheduling algorithm* computes a (possibly infeasible) schedule
- A scheduling algorithm is called *optimal* with respect to a *cost function* if it minimizes that cost function
- A scheduling algorithm is called *optimal* with respect to *feasibility* if it always computes a feasible schedule provided that schedule exists