The Logical Execution Time Paradigm

Christoph Kirsch
Universität Salzburg

ESWEEK Tutorial, Taipei, October 2011

People

- Arkadeb Ghosal (HTL)
- Thomas Henzinger (Giotto, HTL)
- Ben Horowitz (Giotto)
- Daniel Iercan (HTL)
- Eduardo Marques (HTL)
- Marco Sanvido (Giotto)
- Ana Sokolova (HTL)
- ...

LET Programming

Giotto

[EMSOFT 2001, Proceedings of the IEEE 2003]

HTL

[EMSOFT 2006, RTSS 2009]

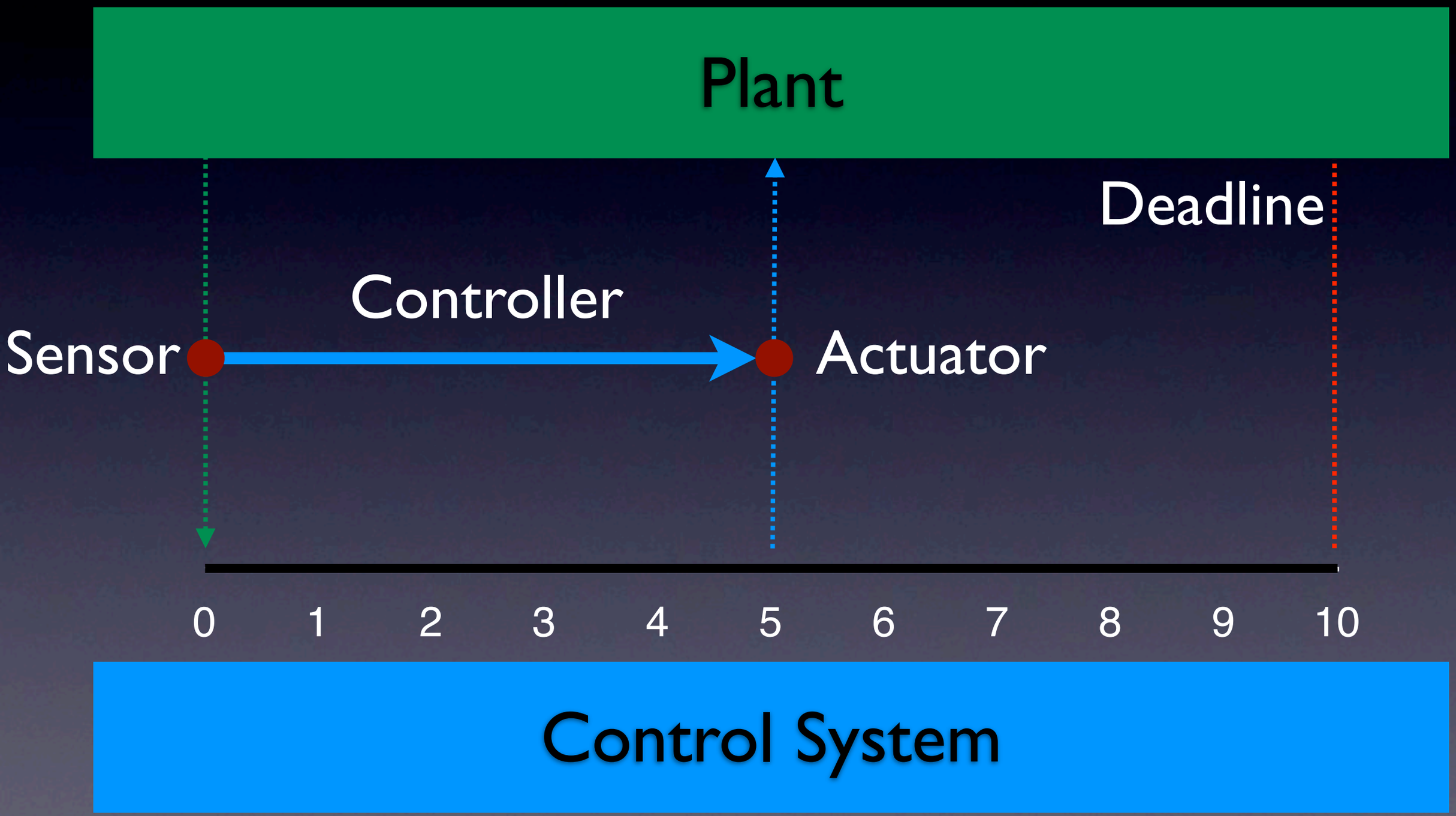
Exotasks

[LCTES 2007, TECS 2008]

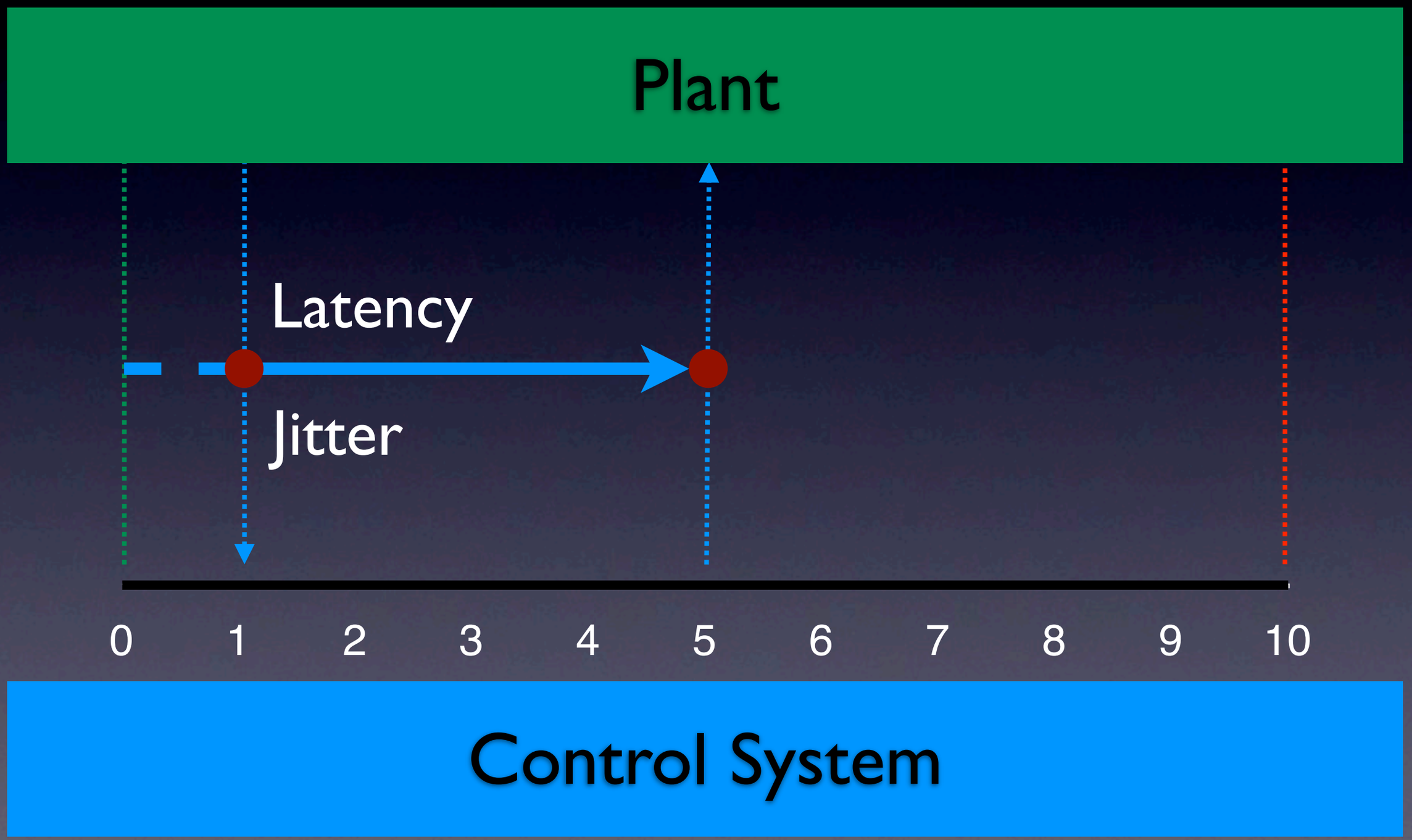
Tiptoe

[IIES 2009, SIES 2009, RePP 2009]

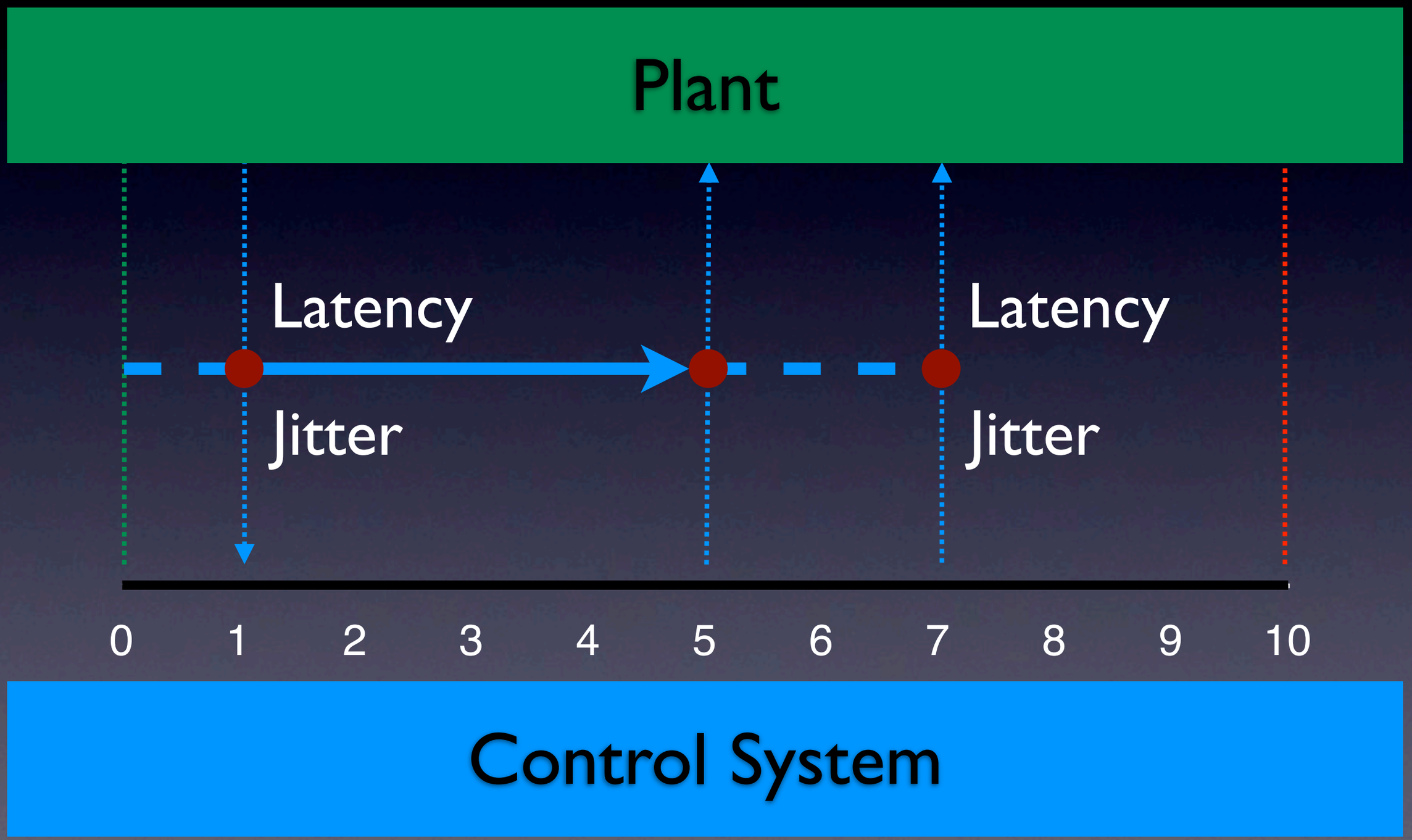
Control Software



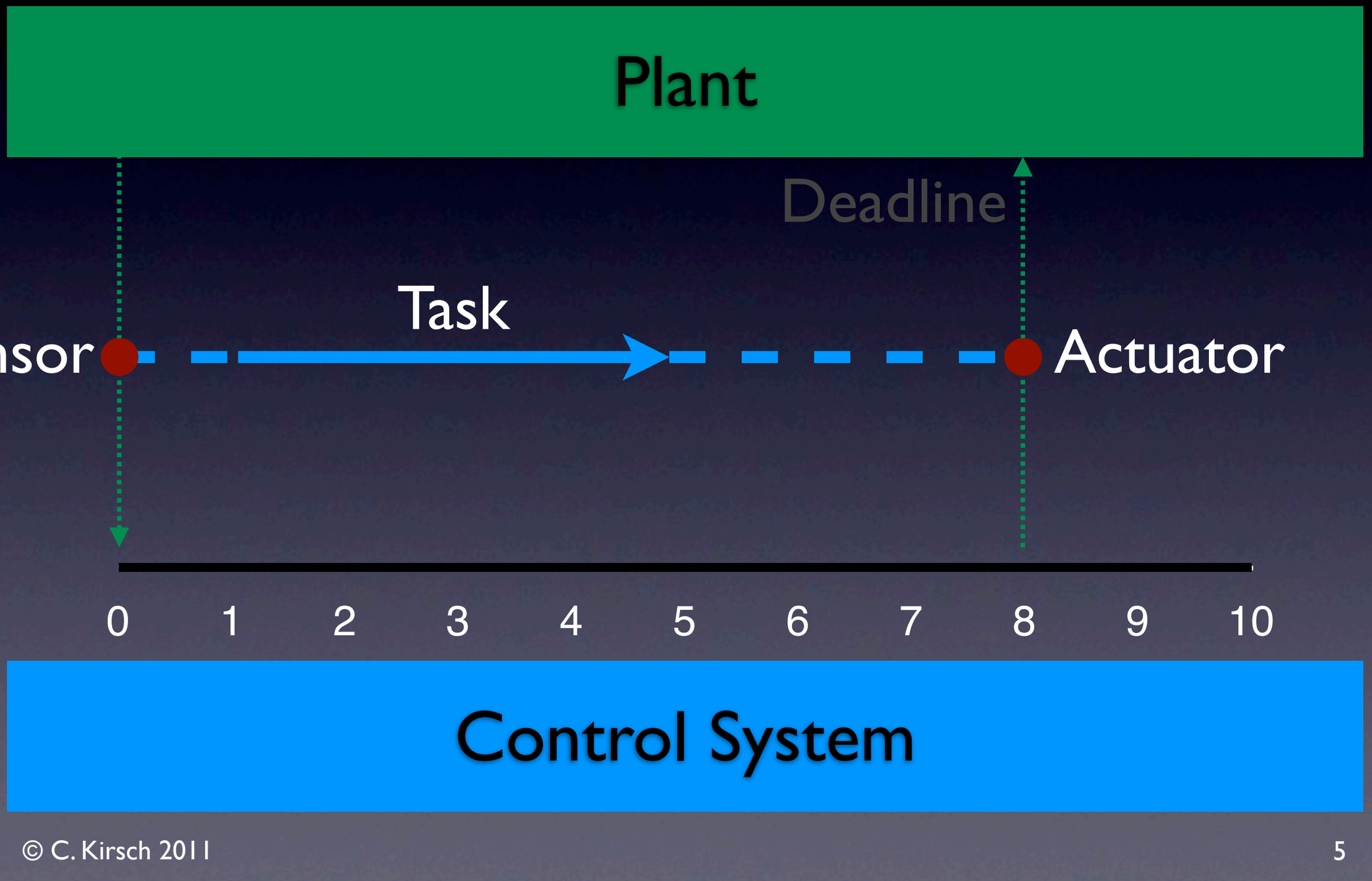
Control Software



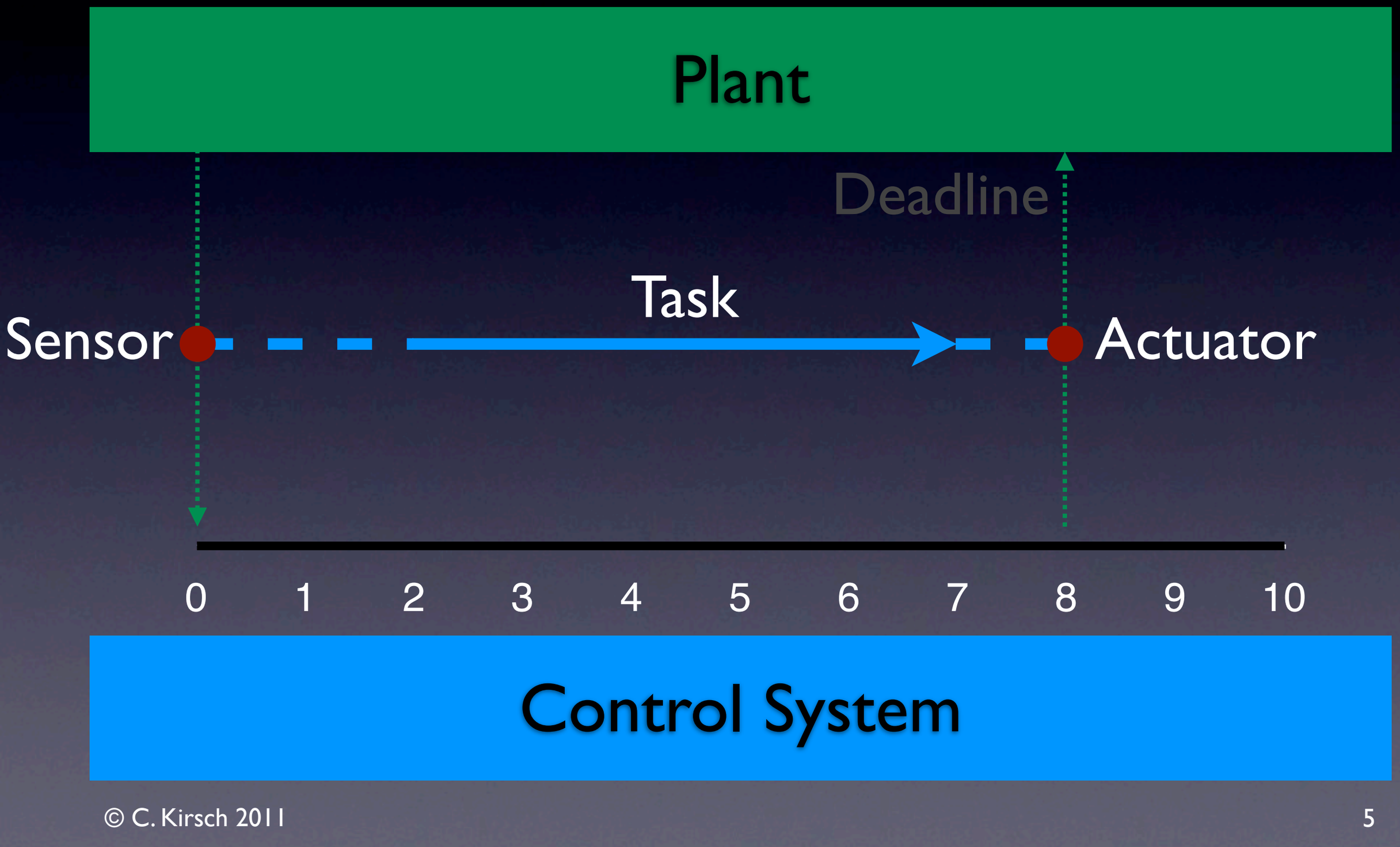
Control Software



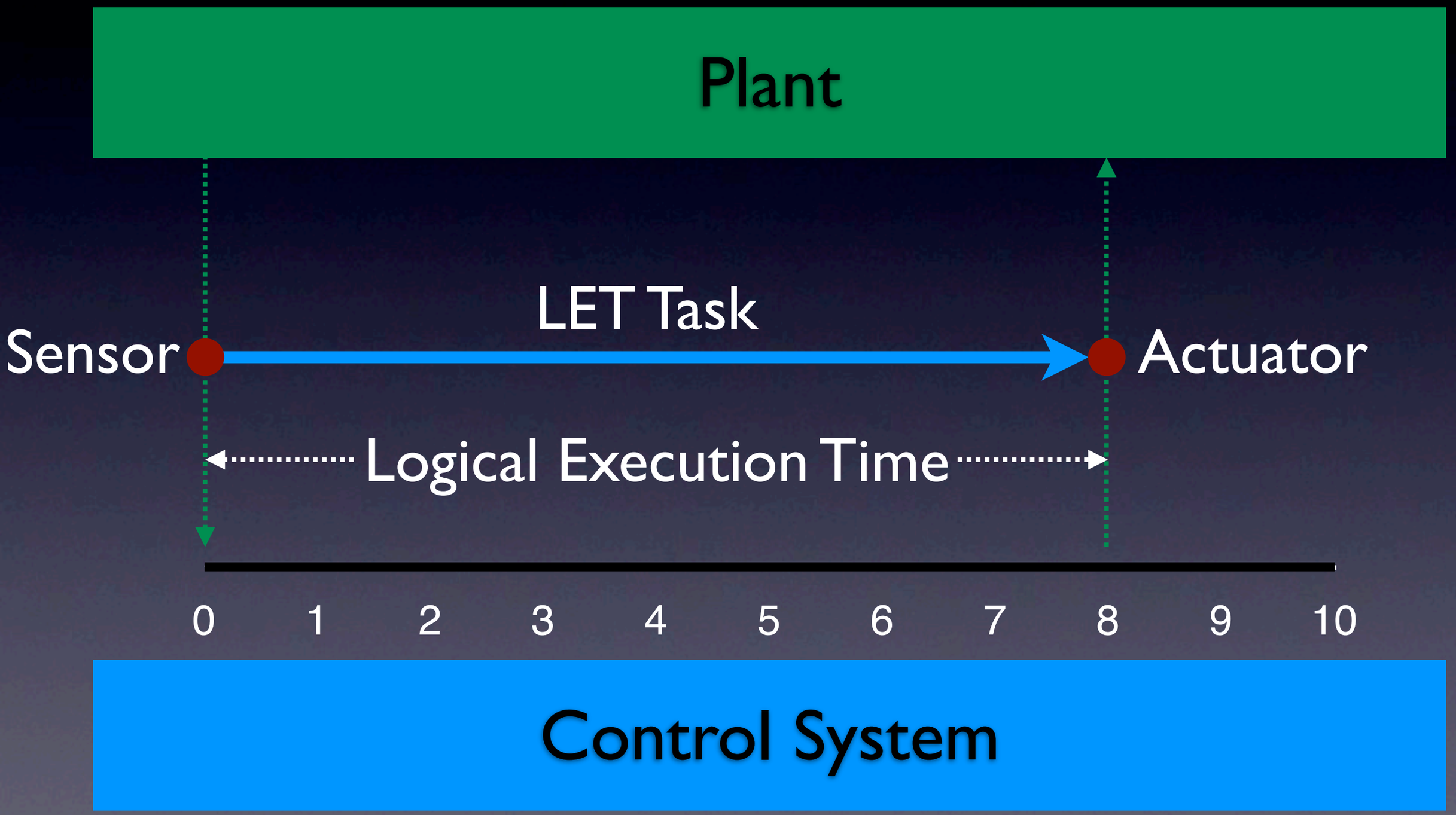
Non-Determinism



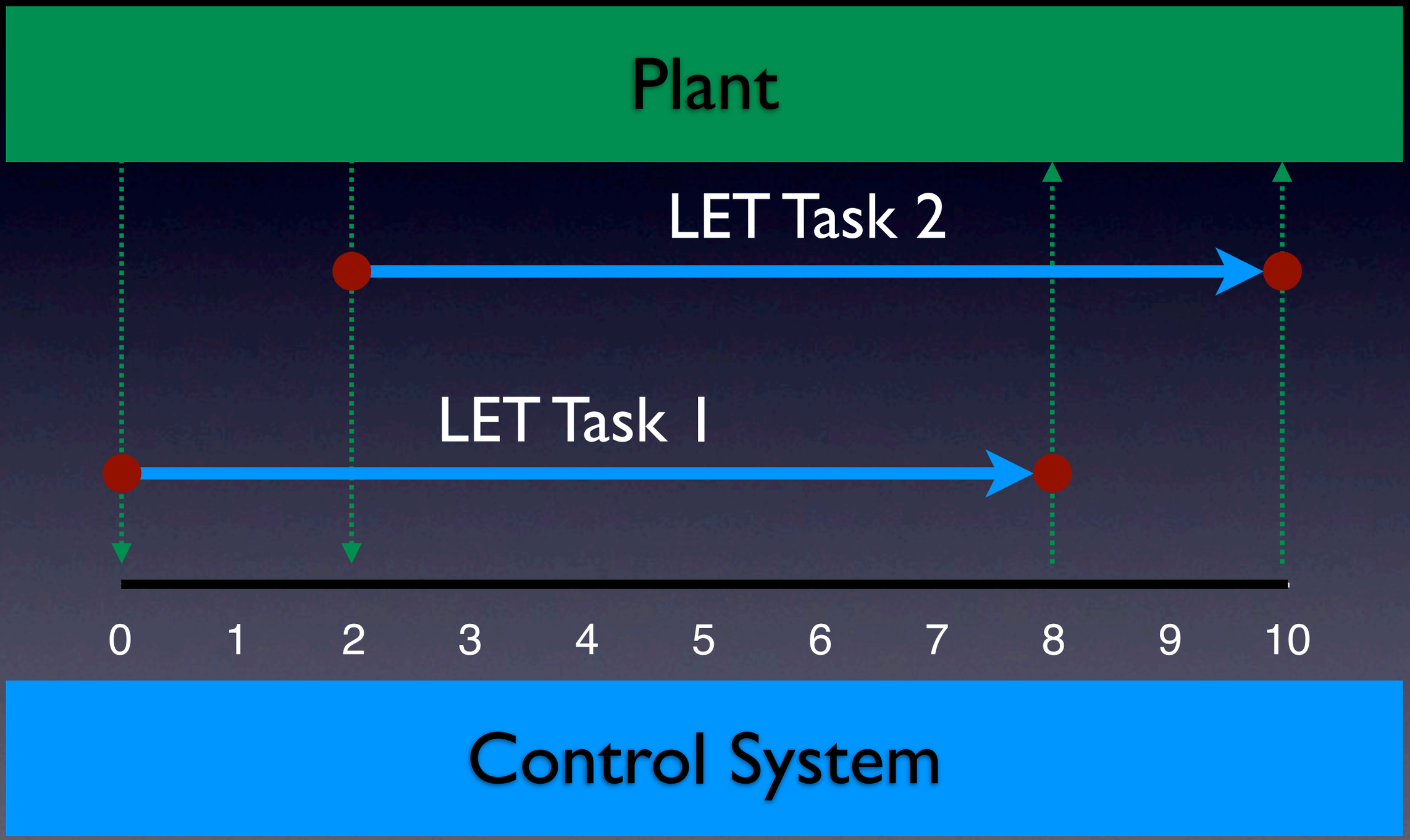
Non-Determinism



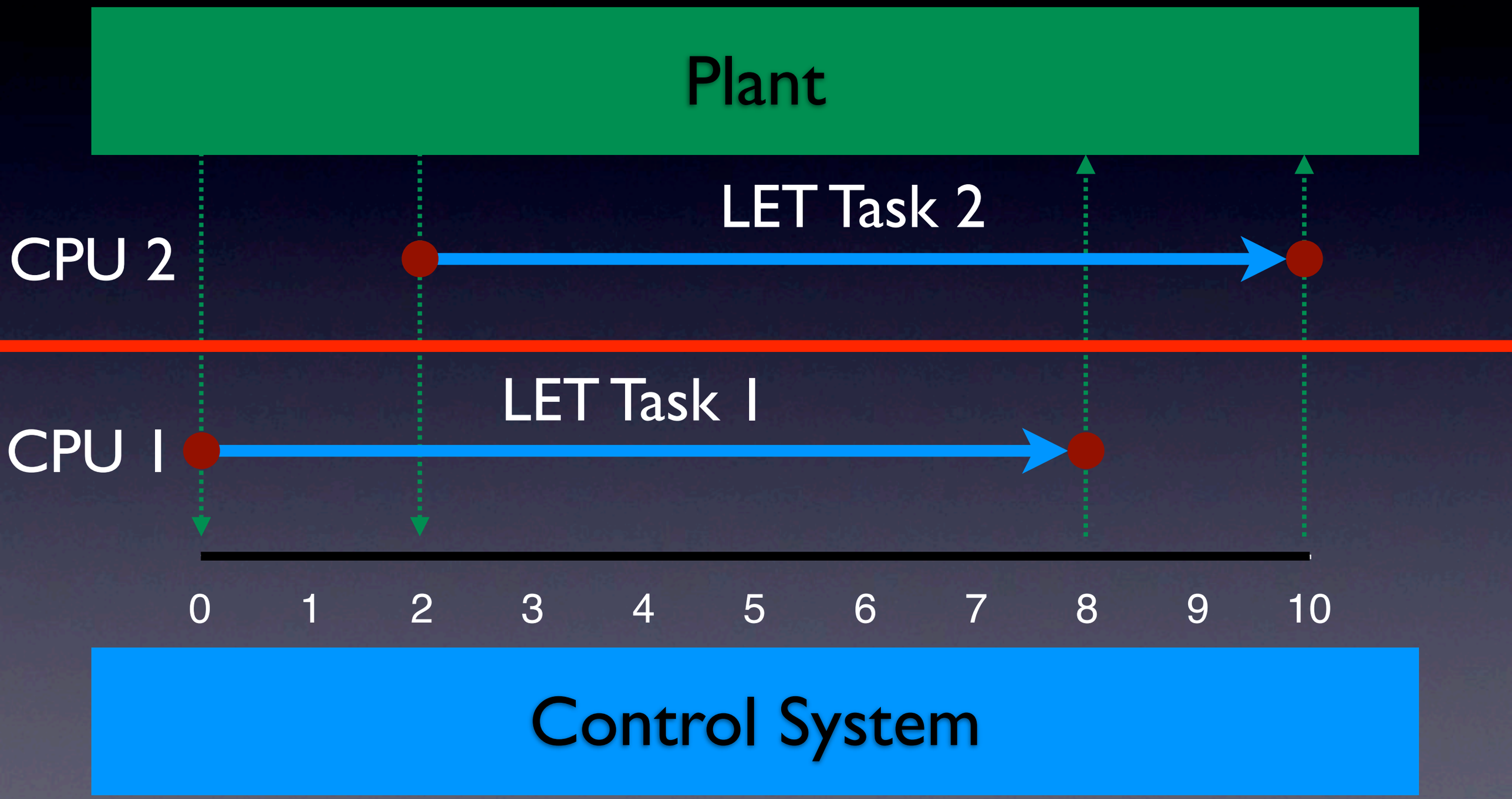
Logical Execution Time



Concurrency



Distribution



A LET Program Incorporates Logical Execution Time as First-Class Concept

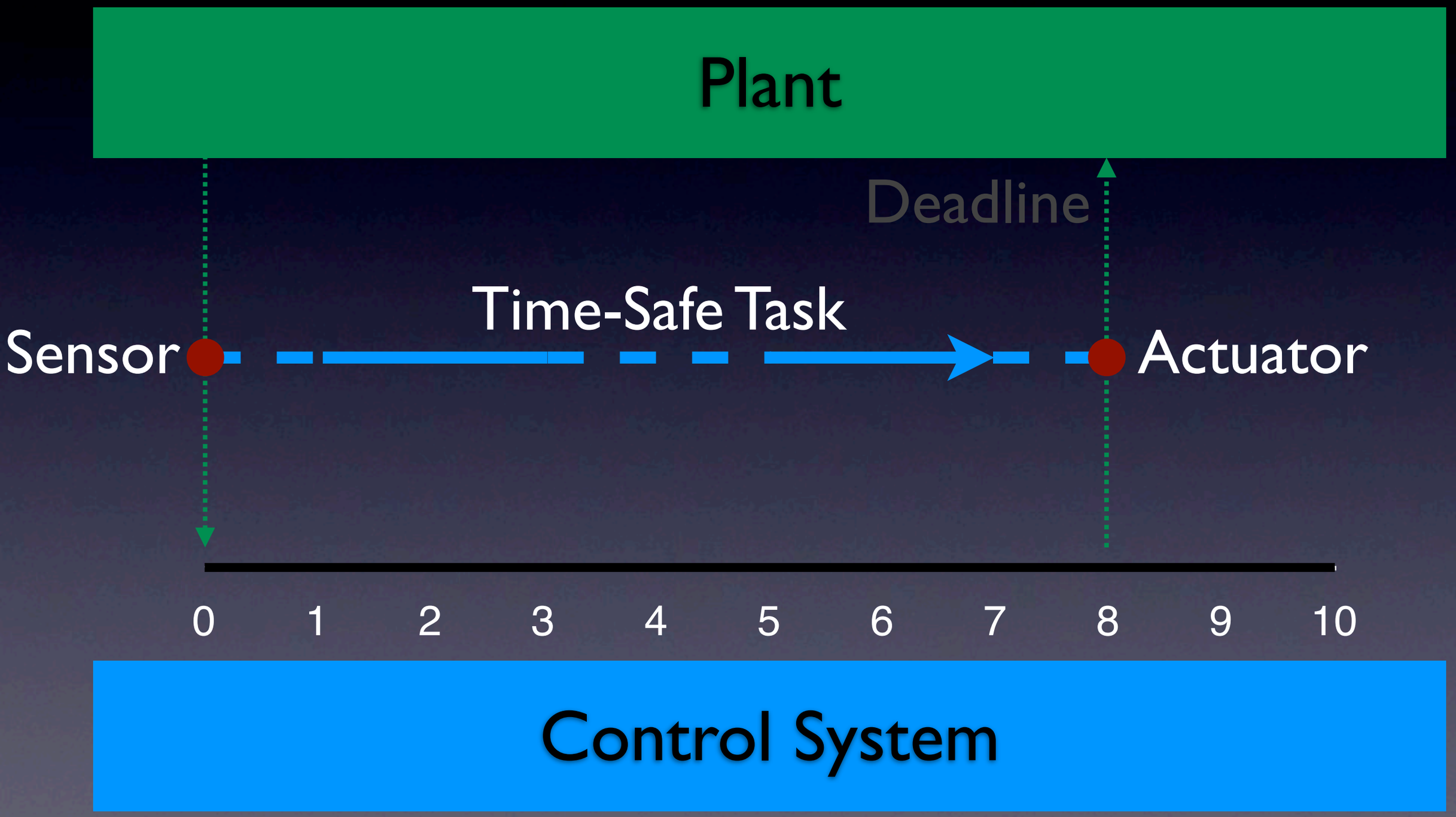
Definition

Plant

A LET program's I/O behavior is *input-determined* if,
for all sequences *I* of input values and times,
the system always produces
unique sequences *f(I)* of output values and times.

Control System

Time Safety



Result

Plant

A LET program's I/O behavior
is input-determined on any platform
that runs the program time-safely.

[with Henzinger, Horowitz at EMSOFT 2001 and
in Proceedings of the IEEE 2003]

Control System

Giotto on ETHZ Helicopter, 2001

From Control Models to Real-Time Code Using Giotto

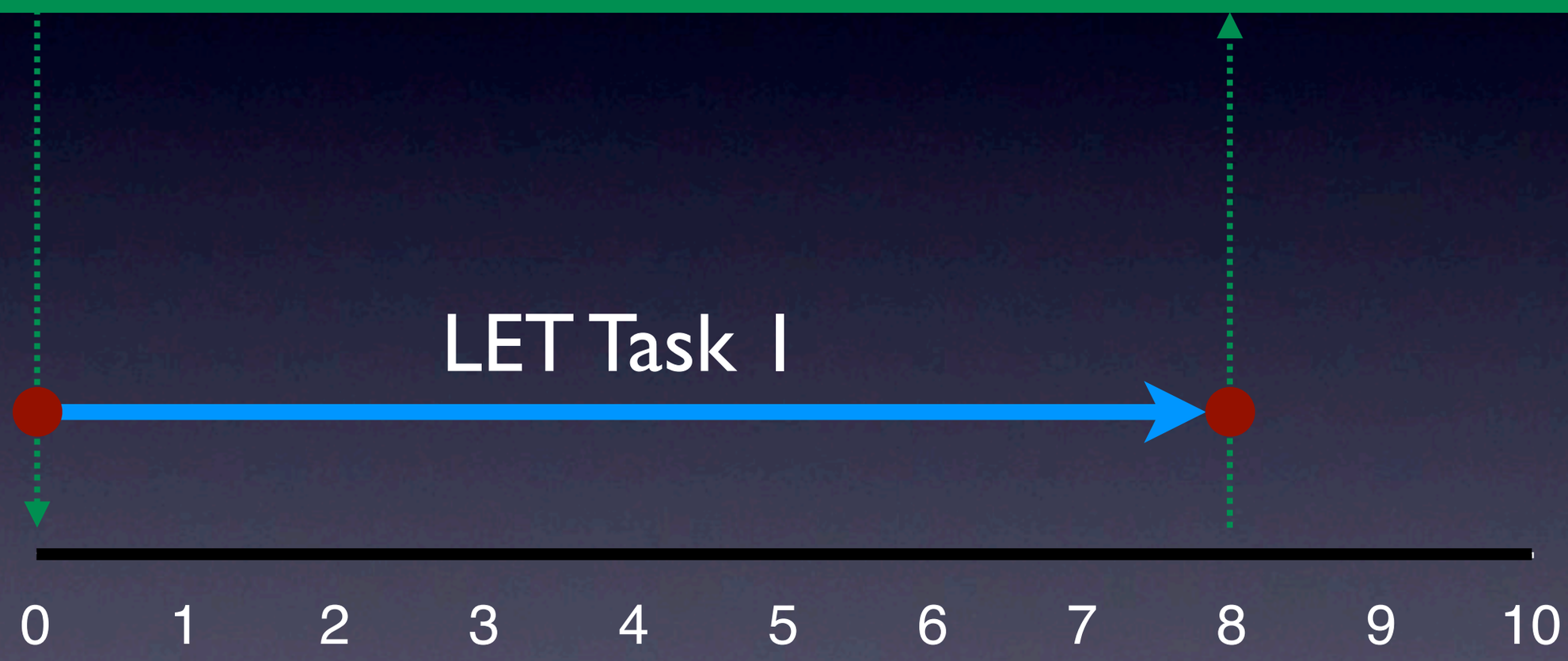
[with Henzinger, Sanvido, Pree in the IEEE Control Systems Magazine, 2003]

HTL:
A Hierarchical Coordination Language for Interacting Real-Time Tasks

[with Ghosal, Henzinger, Iercan, and Sangiovanni-Vincentelli at EMSOFT 2006]

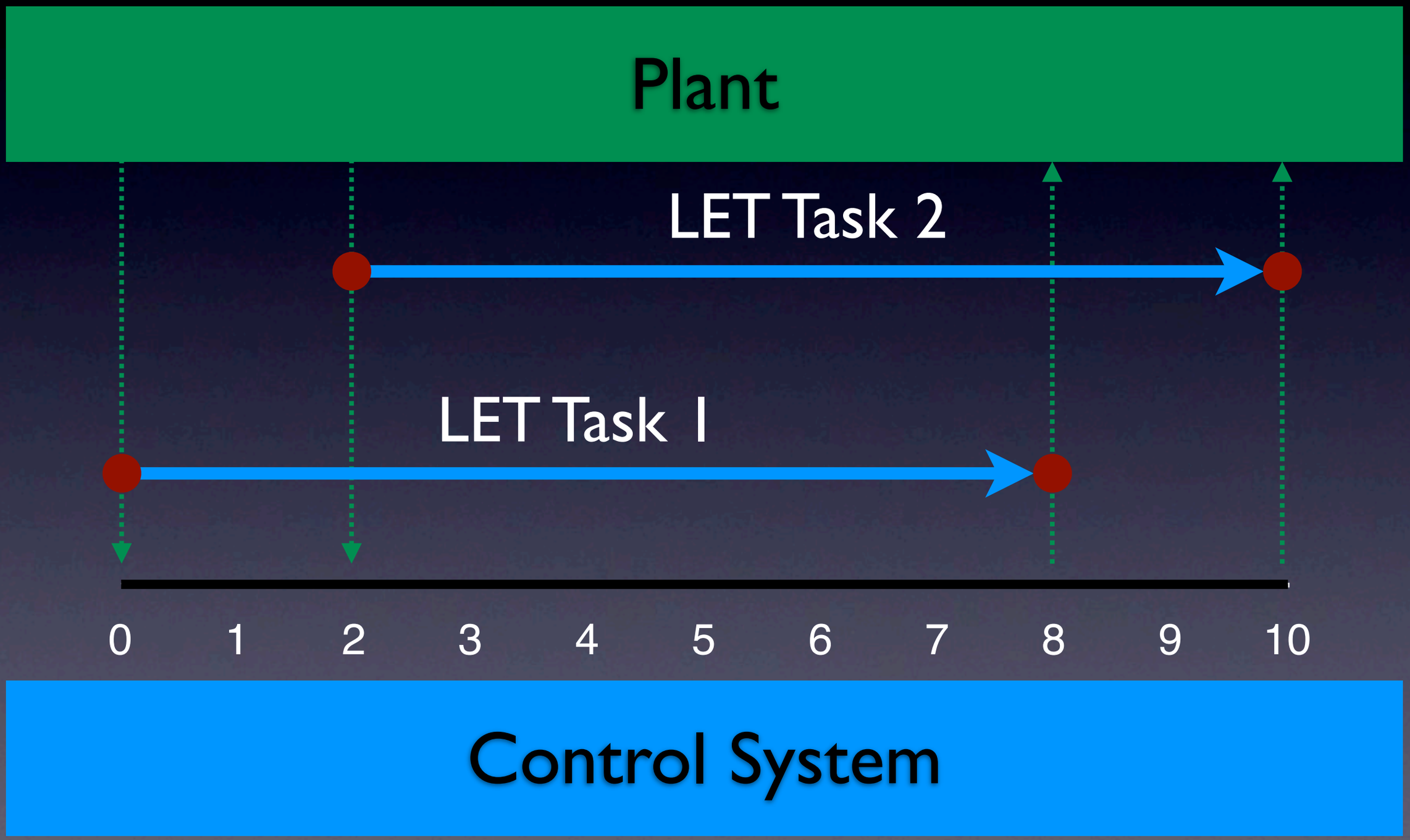
Design: Adding Tasks

Plant



Control System

Design: Adding Tasks



Observation

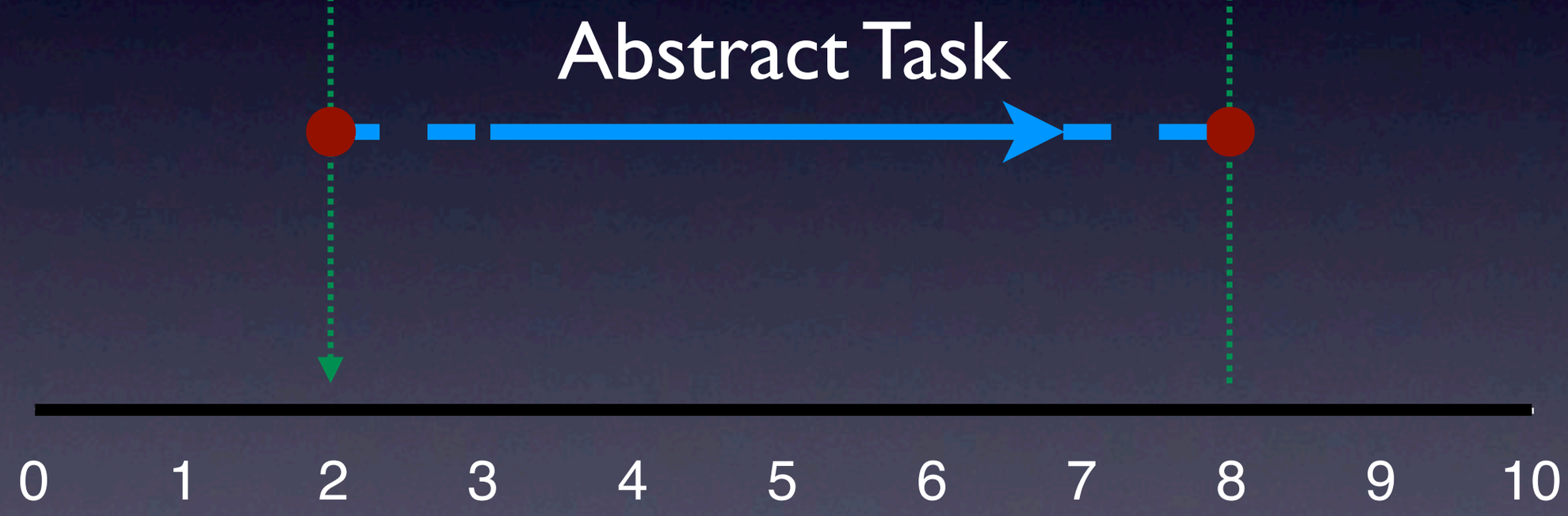
Plant

A LET program's existing I/O behavior does not change by adding new tasks.

Control System

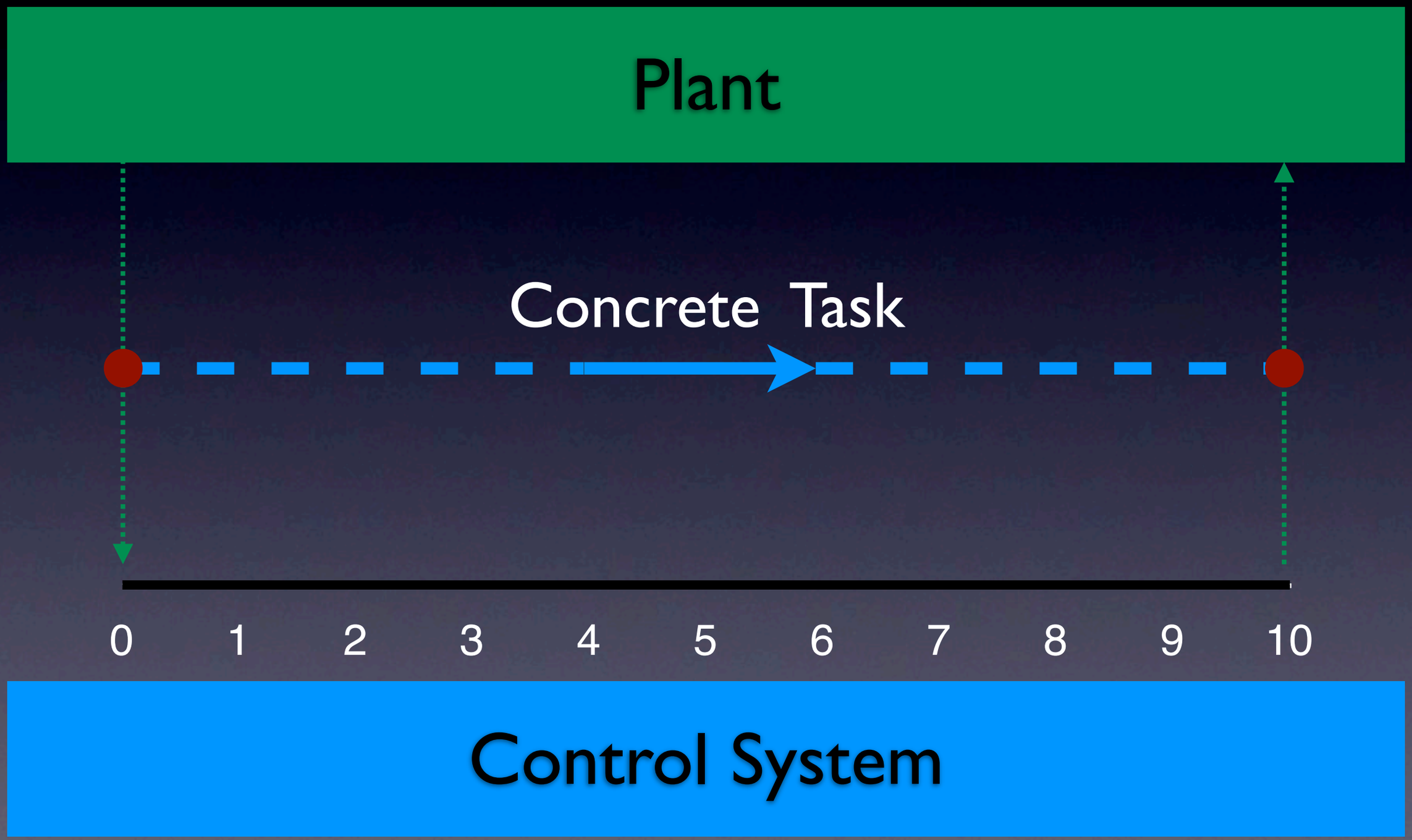
Analysis: Refining Tasks

Plant



Control System

Analysis: Refining Tasks



Result

Plant

A concrete LET program is time-safe if it refines a time-safe, abstract LET program.

[with Ghosal, Henzinger, Iercan, Sangiovanni-Vincentelli at EMSOFT 2006]

Control System

Distributed, Modular HTL

[with Henzinger, Marques, Sokolova at RTSS 2009]

Complexity

φ	C	$\mathcal{D}^{\mathcal{A}}_{\varphi}(C,P)$	$\mathcal{C}^{\mathcal{A}}_{\varphi}(C,P)$	$\overline{\mathcal{C}}^{\mathcal{A}}_{\varphi}(P)$
Well-formedness	any	C	$n^C_{m\downarrow}\ n_T\ n_p$	$n^P_{m\downarrow}\ n_T\ n_p$
Race freedom	top	P	$n^C_{T\uparrow}\ n_w + n_M\ n_c$	$n^P_{T\uparrow}\ n_w + n_M\ n_c$
	ref.	C	1	
Transmission safety	any	C	1	n_c
Time safety	top	P	$(n_m\ \Delta_{max})^{n_M}$	$(n_m\ \Delta_{max})^{n_M}$
	ref.	C	1	
Code generation	any	C	$n^C_{m\downarrow}\ (n_T\ n_a + n_m)$	$n^P_{m\downarrow}\ (n_T\ n_a + n_m)$

n_a	number of communicator accesses per task	n_c	number of communicators	n_M	number of modules per program
n_m	number of modes per module	n_p	number of ports per task	n_T	number of tasks per mode
n_w	number of communicator writes per task	$n^C_{m\downarrow}$	total number of modes in C	$n^P_{m\downarrow}$	total number of modes in P
$n^C_{T\uparrow}$	number of top-level tasks in C	$n^P_{T\uparrow}$	number of top-level tasks in P	$\Delta_{\max}$	maximal value of mode periods

Result I

Plant

Checking time-safety (and transmission-safety) of HTL programs is modular (below top level).

[with Henzinger, Marques, Sokolova at RTSS 2009]

Control System

Result II

Plant

Generating distributed code for HTL programs is modular.

[with Henzinger, Marques, Sokolova at RTSS 2009]

Control System

The JAviator

javiator.cs.uni-salzburg.at

Quad-Rotor Helicopter

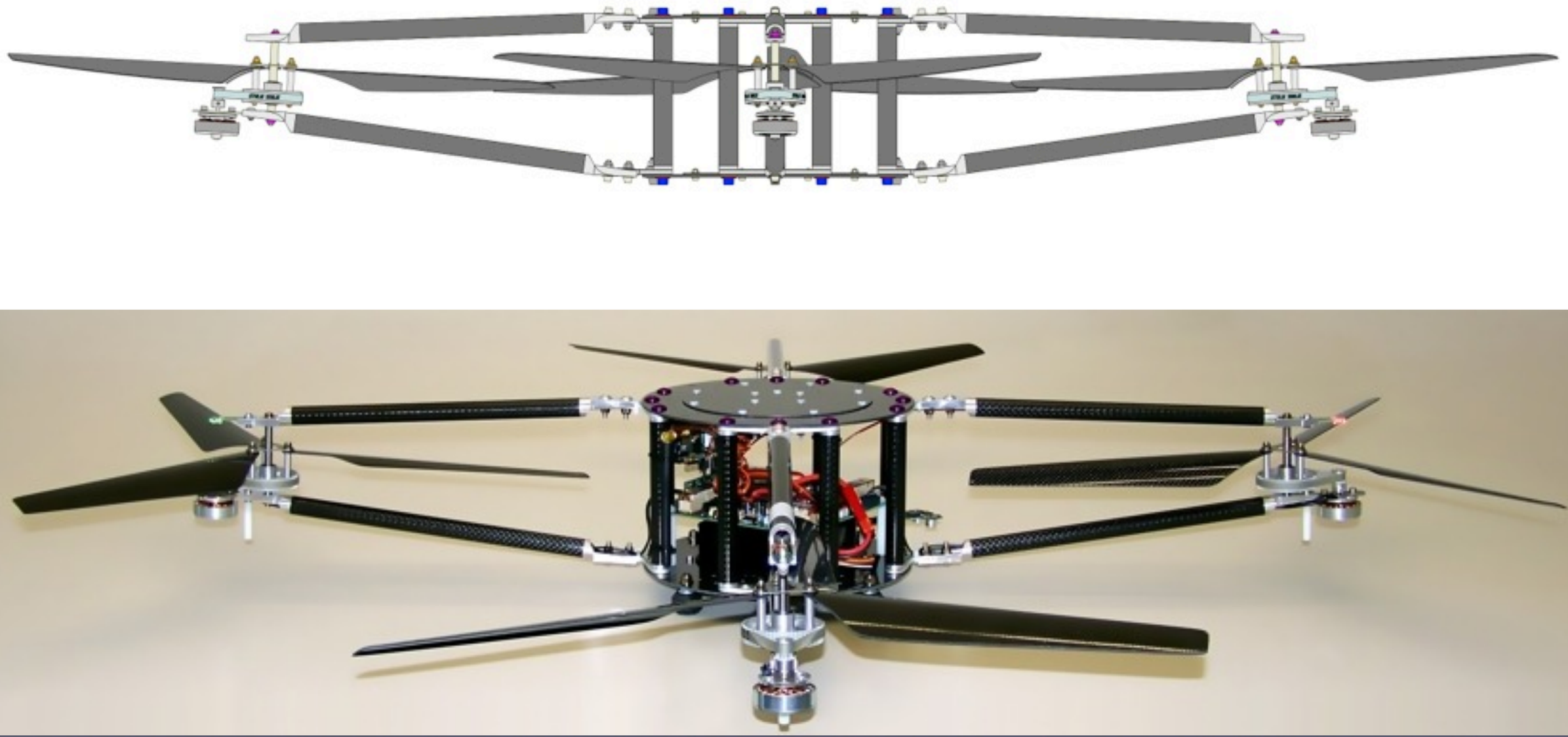
[AIAA GNC 2008]

- all carbon, titanium, aluminum design
- custom motors

- 1.3m diameter
- ~2.2kg weight
- +2kg payload

- ~40min (empty)
- ~10min (full)

Open Source Blueprints



15V
10A
35g
1Kg

Outdoor Flight Salzburg Controller

Thank you