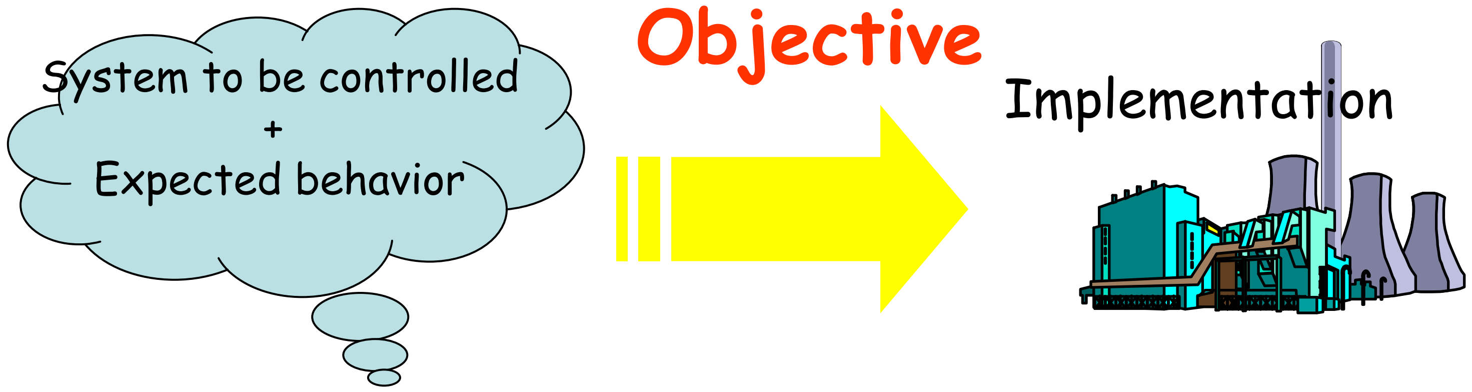
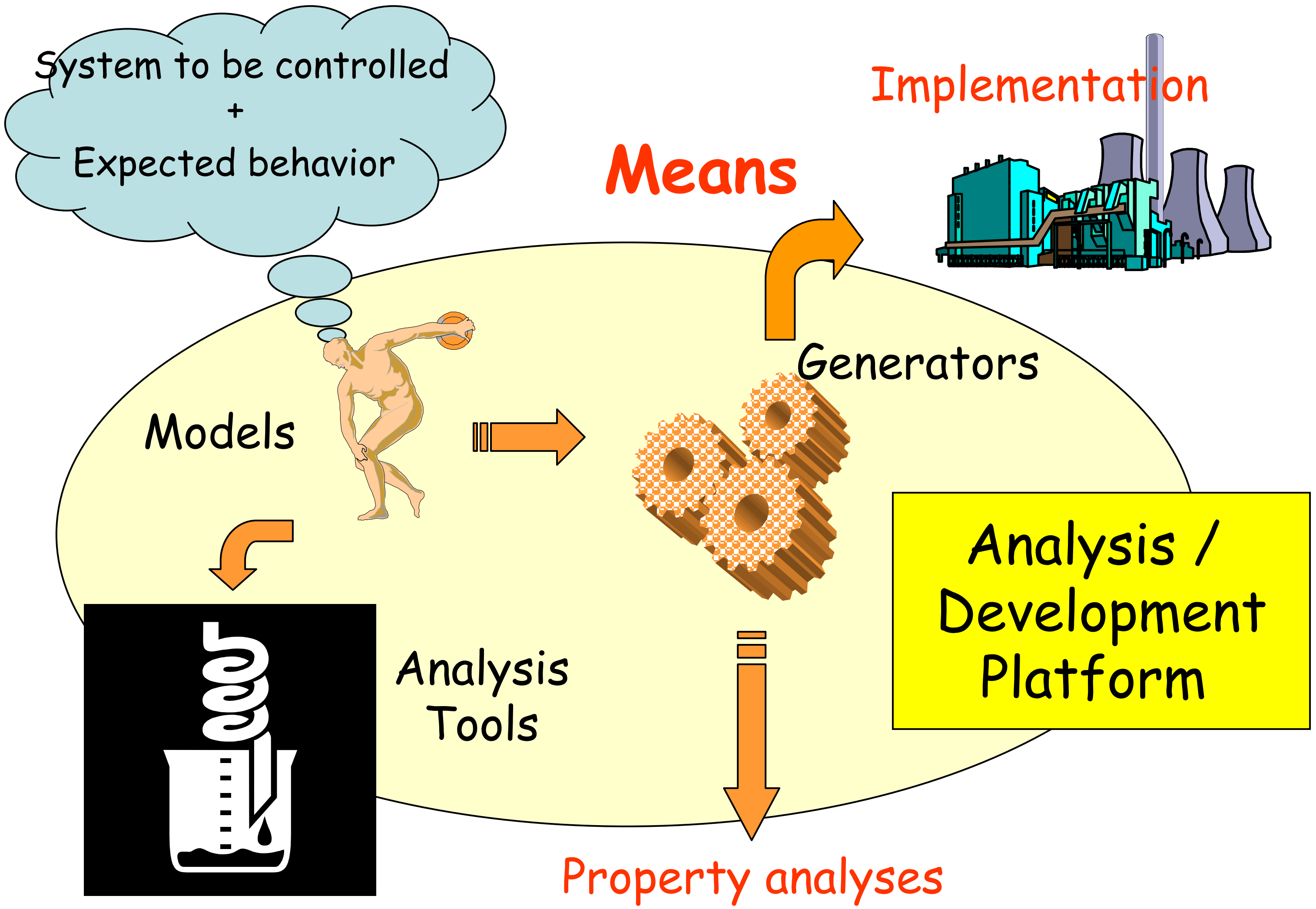
Reactive and Real-Time Systems

Dura lex, sed lex

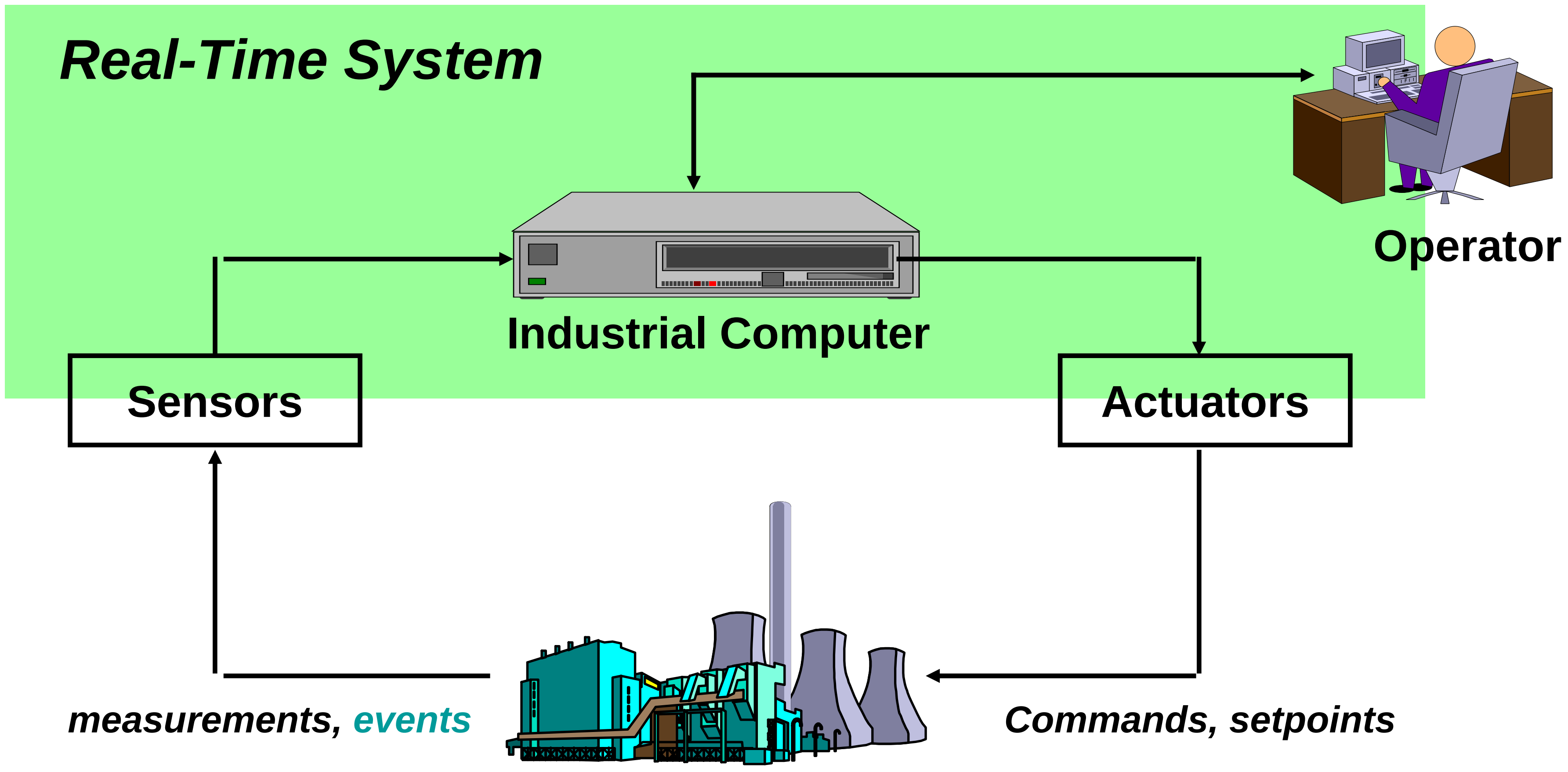
Computer Science and Control



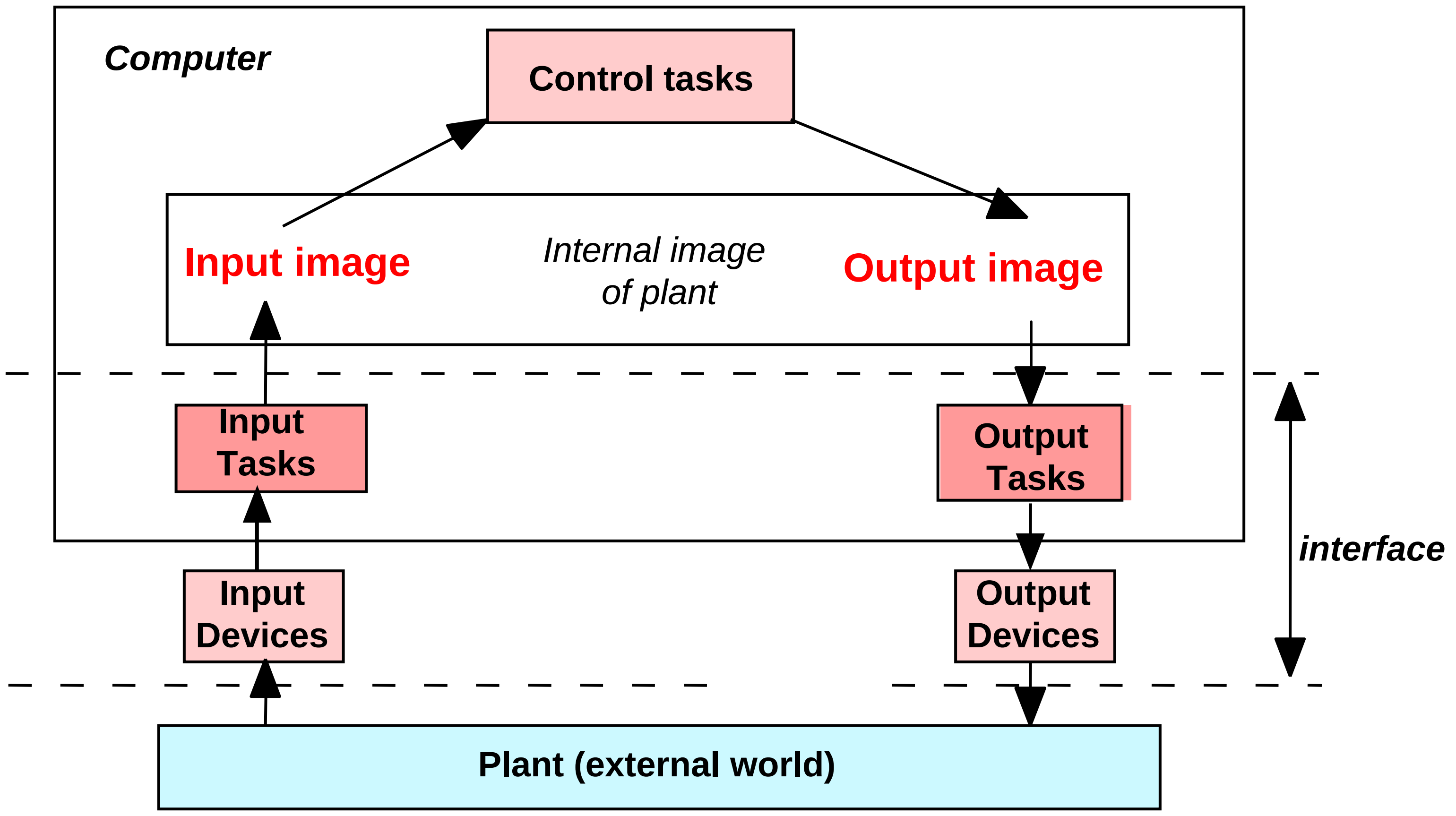
Computer Science and Control



Interactions

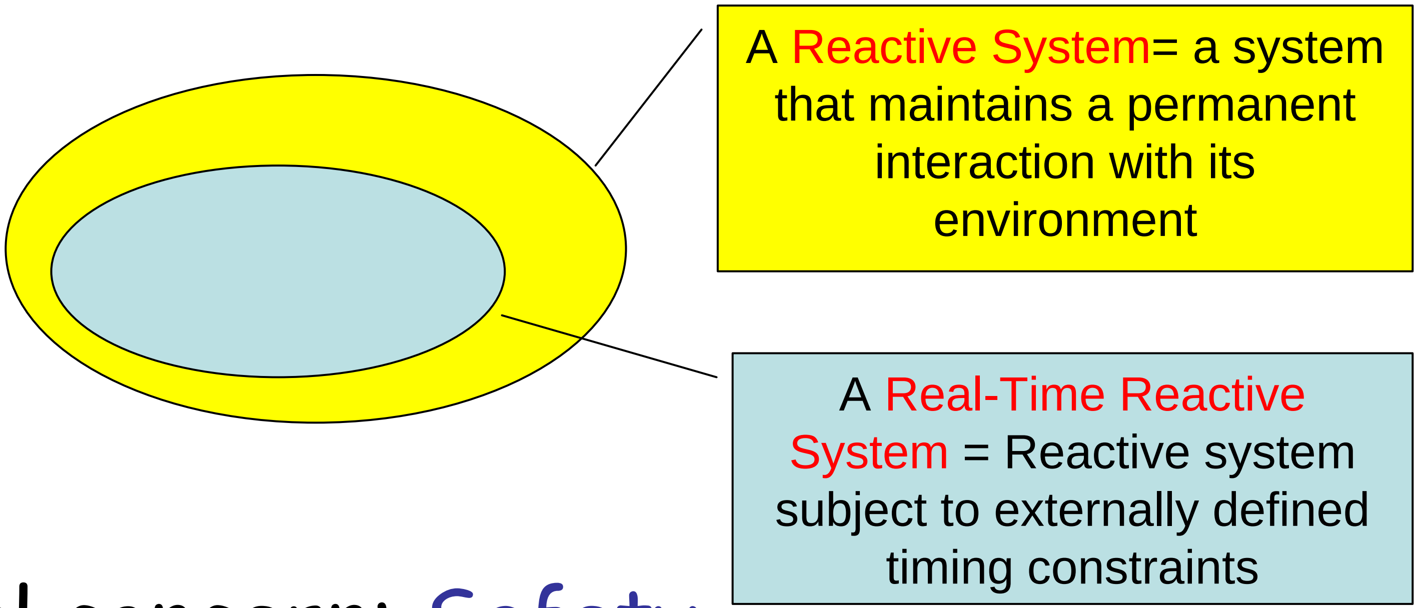


Interfaces



[Bennett]

Reactive & Real-Time Systems


A Reactive System= a system that maintains a permanent interaction with its environment
A Real-Time Reactive System = Reactive system subject to externally defined timing constraints

- Crucial concern: Safety
- Logical correctness essential in all cases
- Temporal correctness: a further requirement for RTS

Embedded Systems

- Most of them are reactive and real-time.
- Interconnected devices that contain software , hardware, electronics,… components.
- *All in all* Computing units are *just another brick in the wall*. (embedded computers)
- Examples: automotive, avionics, cellular phones, smart sensors,… complex digital circuits (System on Chip).

Evolution …

Historically, reactive and real-time applications evolved mostly from the use of analog machines and relay circuits to the use of microprocessors and computers. They did not benefit from the recent progress in programming technology as mush as did other fields.

Although strongly technically related, the various application fields are treated by different groups of people having their own methods and vocabulary, and little relation has been established between them.

The programming tools are still often low-level and specific. [Benveniste & Berry]

… Evolution

The present situation must change rapidly.

The Synchronous Approach is now recognized as a technology of choice for modeling, specifying, validating and implementing real-time embedded applications.

The first part of this course is devoted to synchronous programming and related topics.

Reactive & Real-Time Systems: Main Issues (1)

Most reactive and real-time systems naturally decompose into communicating concurrent components.

All aspects related to concurrency are important (qualitative aspects):

- Communication
- Synchronization
- Organization of the computational flow

Quantitative aspects, imposed by Timing constraints, are mostly related to the speed of computation

Reactive & Real-Time Systems: Main Issues (2)

1. Use modular and formal techniques to specify, implement, and verify programs
2. Encompass within a single framework all reactive aspects
3. Deal with distributed target architectures
4. Preserve determinism whenever possible
5. Consider issues of speed

[Benveniste & Berry]

Real-Time Systems

A short introduction

Real-Time Systems

Any system in which the time at which the output is produced is significant. This is usually because the input corresponds to some movement in the physical world, and the output has to relate to the same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness (Oxford Dictionary of Computing)

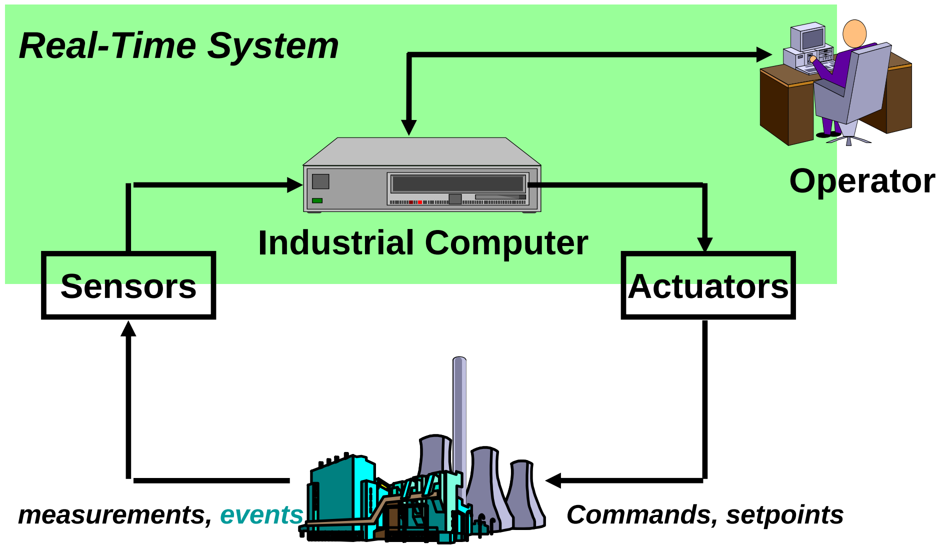
Real-Time systems are those which must produce correct response within a definite time limit. Should computer responses exceed these time bounds then performance degradation and/or malfunction results (Cooling, Software Design for Real-Time Systems, Chapman & Hall, 1991)

A program for which the correctness of operations depends both on the logical results of the computation and the time at which the results are produced.

Embedded Systems

Embedded computers = computer systems in which the computer is just one functional element of a real-time system and is not a stand-alone computing machine.

Example:



Classification of Real-Time Systems (1)

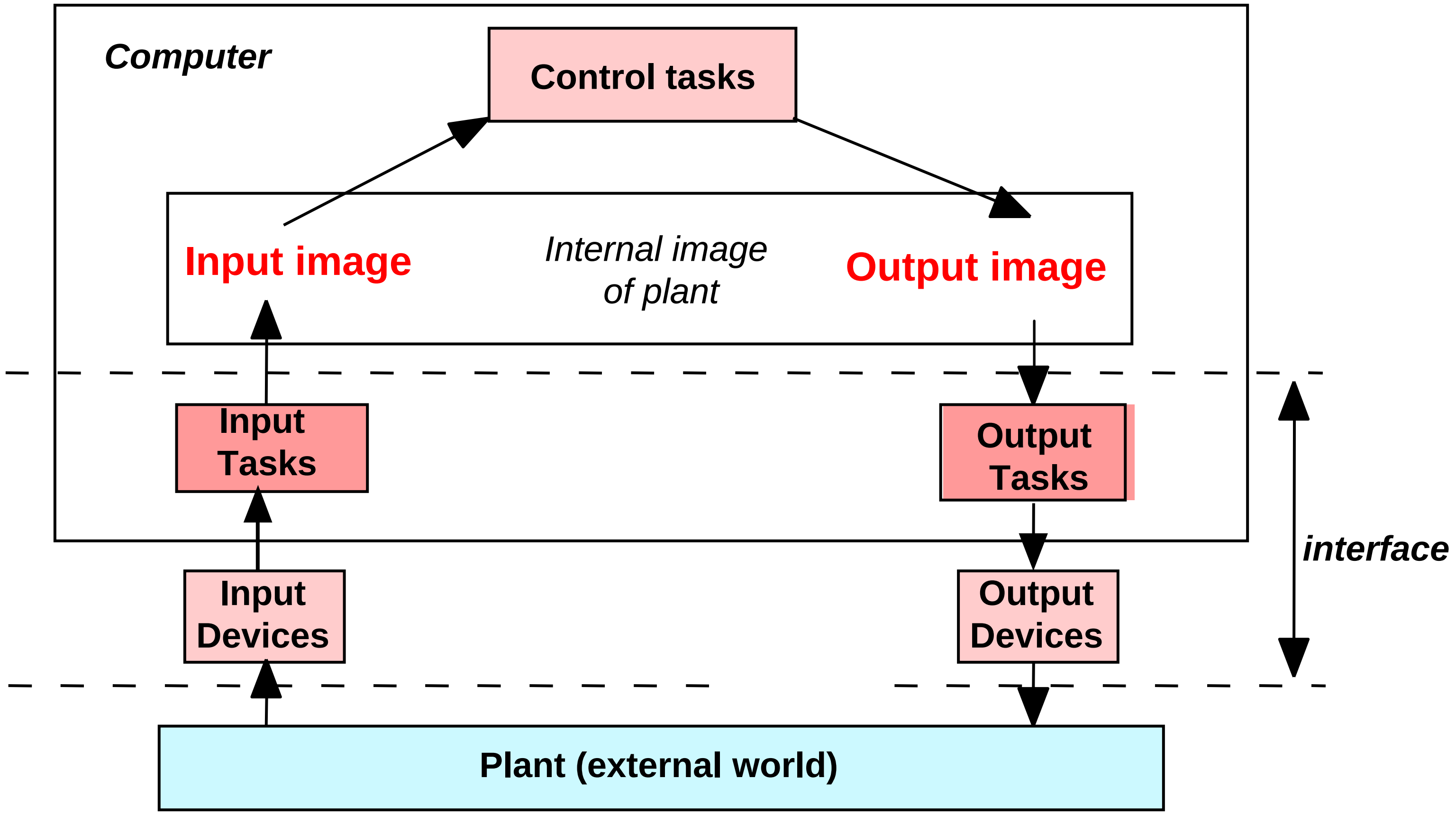
A common feature of RT systems and embedded computers is that the computer is connected to the environment within which it is working by a wide range of interface devices and receives and sends a variety of stimuli.

Input tasks, output tasks, communication tasks are connected by physical devices to processes which are external to the computer. These external processes all operate in their own time-scales and the computer is said to operate in real time if actions carried out in the computer relate to the time-scales of the external processes.

Classification according to the type of synchronization between the external processes and the internal actions (tasks).

[Bennett]

Interfaces



Computer
Control tasks
Input image
Internal image of plant
Output image
Input Tasks
Output Tasks
Input Devices
Output Devices
interface
Plant (external world)

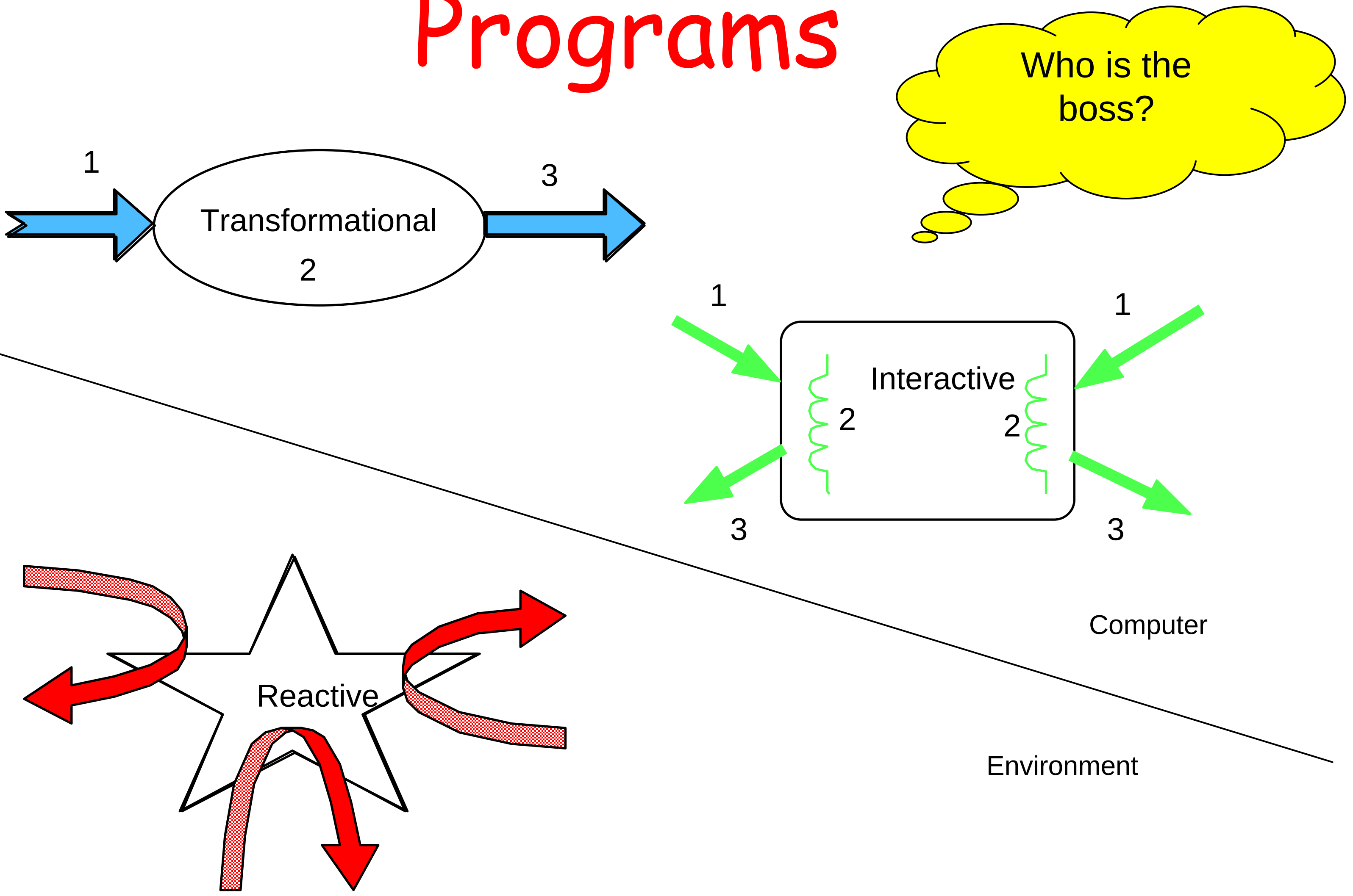
Classification of Real-Time Systems (2)

Clock-based Tasks (cyclic, periodic)
- Time constants (of the controlled system)
- Sampling rate (for feedback control)
- Real-Time clock

Event-based Tasks (aperiodic)
- Response to some event
- Interrupt / Polling
- Events occur at non-deterministic intervals

Interactive systems
- The largest class of RTS
- Requirements: "…the average response time must not exceed …"
- The system responds at a time determined by the internal state of the computer and without any reference to the environment.

Transformational Interactive Reactive Programs



Terms and Concepts

- Timeliness

  The timeliness of an action has to do with the action meeting time constraints such as a deadline.

  The important modeling concerns of timeliness are modeling execution time, deadline, arrival patterns, synchronization patterns, and time sources.

- Responsiveness

  Standard computing systems respond primarily to the user. Real-time systems, on the other hand, may interact with users, but they have more concern for interactions with sensors and actuators.

[Douglass]

Terms and Concepts (2)

- Concurrency

  Concurrency is the simultaneous execution of multiple sequential chains of actions. These actions may execute on one processor (pseudoconcurrency) or multiple processors (true concurrency).

  Scheduling concurrent threads, Event arrival patterns, thread rendez-vous patterns, sharing resources.

[Douglass]

Terms and Concepts (3)

- Predictability

  A key aspect of many real-time systems. This is crucial for many safety-critical and high-reliability systems.

  Memory management

- Correctness and Robustness

  A system is correct when it does the right thing all the time. Such a system is robust when it does the right thing under novel (unplanned) circumstances, even in the presence of unplanned failures of portions of the system.

  Deadlock, exceptional conditions, race conditions

[Douglass]

Terms and Concepts (4)

- Distributed systems
- Fault tolerance and safety

Many embedded systems have high availability requirements. In such systems, it is undesirable for the system to fail.

Many of these systems must not only be reliable, but they must also be safe—that is, if they do fail, they do so without causing injury or loss of life.

Reliabilty, safety, redundancy.

[Douglass]

Terms and Concepts (5)

- Dealing with Resource-limited target environments
  - Embedded systems ship the hardware along with the software as part of a complete system package.
  - Recurring cost = per-shipped-item cost (hw)
  - Development, maintenance cost (sw)
  - Physical size, heat production, weight limitations
  - Often cross-platform development
  - Often design and write software for not yet existing hardware

Real-Time Programming: The State of the Art (1)

- Event-loop
  Condition : actions (or callbacks)
  (+) Simple sequential code
  (-) Non deterministic
- Sequential imperative language + RTOS
  (+) The most common way of doing things
  (-) More or less several loosely connected programs
  (--) Hard to debug and maintain
  (--) Little room for clean automatic program behavior analysis
  (--) Non determinism introduced by the OS
- Finite State Machine
  (+) Deterministic
  (-) No direct support for hierarchy and concurrency
  (--) Highly sensitive to modifications
  (--) Only small FSMs are humanly understandable

Real-Time Programming: The State of the Art (2)

- Petri Nets
  - (+) support concurrency
  - (+) mathematical analysis
  - (-) little modularity
  - (-) usually non deterministic
- Concurrent Programming Languages
  - (?) ADA, OCCAM
  - (?) Java

References

- S. Bennett. “Real-Time Computer Control – An introduction”, 2nd Ed., Prentice Hall, 1994.
- A. Benveniste, G. Berry. “The Synchronous Approach to Reactive and Real-Time Systems”, Proc. IEEE, vol 79, pp1270-1282, Sept. 1991.
- B. P. Douglass. “Doing Hard Time”, Addison-Wesley, 1999.