Real Time System Design

Introduction to Real Time System

1. Definition

A real time system is any information processing system which has to respond to externally generated input stimuli within a finite and specified period. Also it could be define as the ability of the system to guarantee a response after a (domain defined) fixed time has elapsed.

The hardware of a computer solves problems by repeated execution of machine - language instructions, collectively known as software. Software is traditionally divided into *system programs* and *application programs*. System programs consist of software that interfaces with the underlying computer hardware, such as device drivers, interrupt handlers, task schedulers, and various programs that act as tools for the development or analysis of application programs. These software tools include *compilers*, which translate high - level language programs into assembly code; *assemblers*, which convert the assembly code into a special binary format called object or machine code; and *linkers/locators*, which prepare the object code for execution in a specific hardware environment. An *operating system* is a specialized collection of system programs that manage the physical resources of the computer. As such, a real - time operating system is a truly important system program.

Application programs are programs written to solve specific problems, such as optimal hall - call allocation of an elevator bank in a high - rise building, inertial navigation of an aircraft, and payroll preparation for some industrial company. Certain design considerations play a role in the design of system programs and application software intended to run in real - time environments. The notion of a

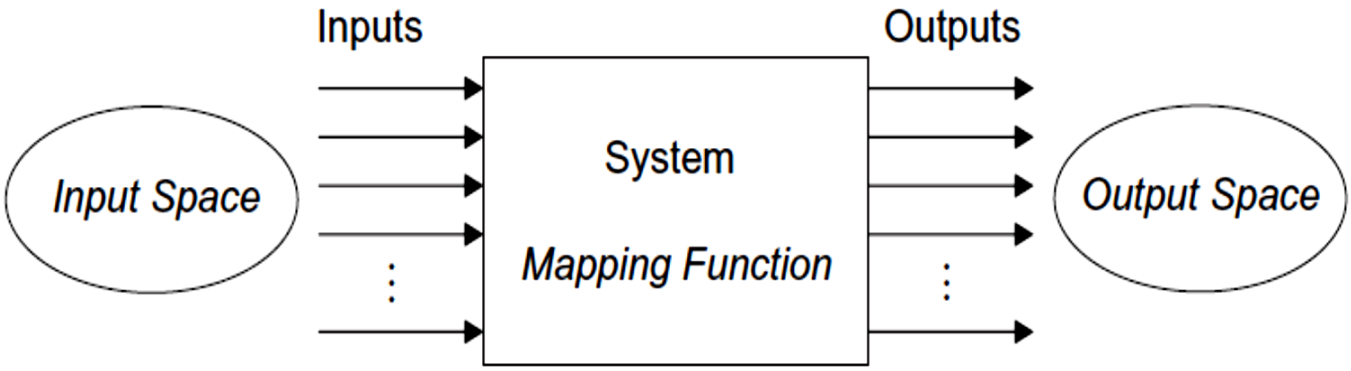
"system" is central to software engineering, and indeed to all engineering, and warrants formalization.

1.1 What is a System?

A system is a mapping of a set of inputs into a set of outputs. When the internal details of the system are not of particular interest, the mapping function between input and output spaces can be considered as a black box with one or more inputs entering and one or more outputs exiting the system as shown if Fig.(1).

There are five general properties that belong to any "system":

1. A system is an assembly of components connected together in an organized way.
2. A system is fundamentally altered if a component joins or leaves it.
3. It has a purpose.

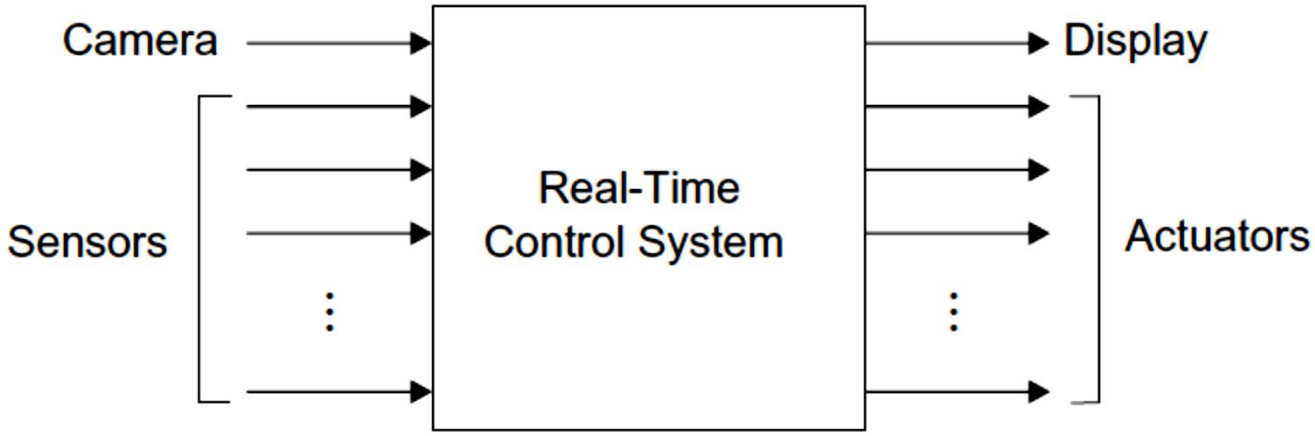


4. It has a degree of permanence.
5. It has been defined as being of particular interest.

Fig.(1): A general system with inputs and outputs.

Modeling a real - time (control) system, as in Fig.(2) , is somewhat different from the more traditional model of the real - time system as a sequence of jobs to be scheduled and performance to be predicted, which is comparable with that shown in Fig.(3). The latter view is simplistic in that it ignores the usual fact that

the input sources and hardware under control may be highly complex. In addition, there are other, “sweeping” software engineering considerations that are hidden by the model shown in Fig.(3). Look again at the model of a real - time system shown in Fig.(2). In its realization, there is some inherent delay



between presentation of the inputs (excitation) and appearance of the outputs (response).

Fig.(2): A real - time control system including inputs from a camera and multiple sensors, as well as outputs to a display and multiple actuators.

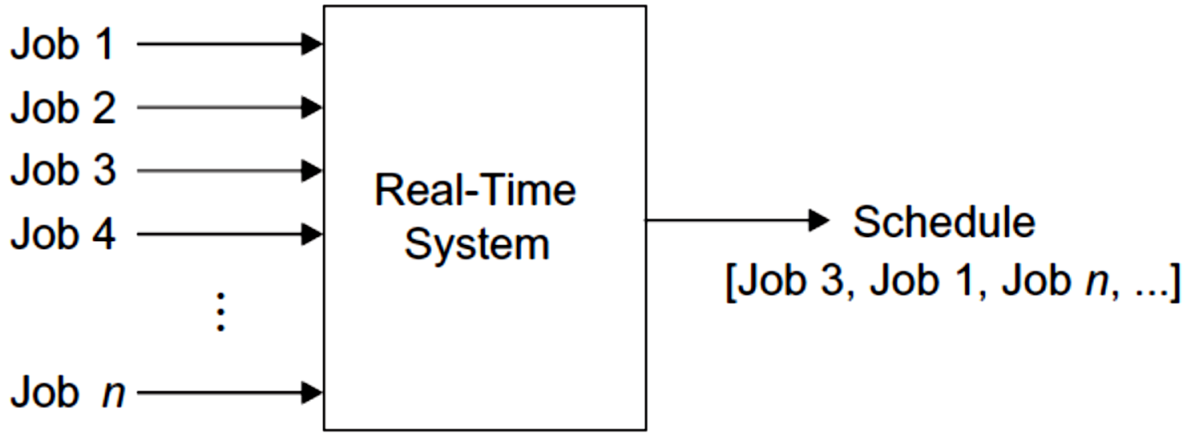


Fig.(3): A classic representation of a real - time system as a sequence of schedulable jobs.

1.2 Real Time Aspects

While speed is indeed fundamental to real-time performance, speed alone is not real-time. The four aspects of real-time performance are:

- ❖ Speed
- ❖ Responsiveness
- ❖ Timeliness
- ❖ Graceful adaptation.

Speed is the rate of execution of tasks. Tasks could refer to problem-solving tasks (large or small) or event processing tasks.

Responsiveness is the ability of the system to stay alert to incoming events. Since an interactive real-time system is primarily driven by external inputs, the system should recognize that such input is avail able. It may not necessarily process the new event right away; that may depend upon the criticality relative to other events the system is currently processing.

Timeliness is the ability of the system to react to and meet deadlines, which can be achieved by processing the more critical tasks first. Criticality of tasks may be defined by several factors, either domain dependent or domain independent or both.

Graceful adaptation is the ability of the system to reset task priorities according to changes in work load or resource availability. Graceful adaptation allows for a smooth transition across potential perturbations to the system. Without this ability, the system may waste

Processing resources as well as time by executing less relevant tasks or waits for scarce resources.

So, the correctness depends not only on the logical result but also the time it was delivered and failure to respond is as bad as the wrong response.

1.3 Deadline and Real Time System Patterns

Is there a pattern?

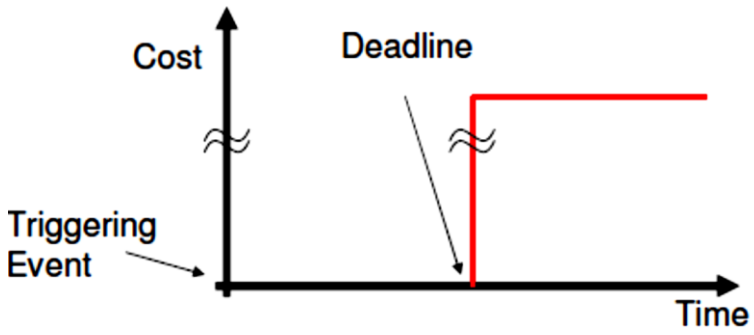
- ❖ Hard real-time systems
- ❖ Soft real-time systems
- ❖ Firm real-time systems

A ***deadline*** is a given time after a triggering event, by which a response has to be completed.

Hard real-time systems

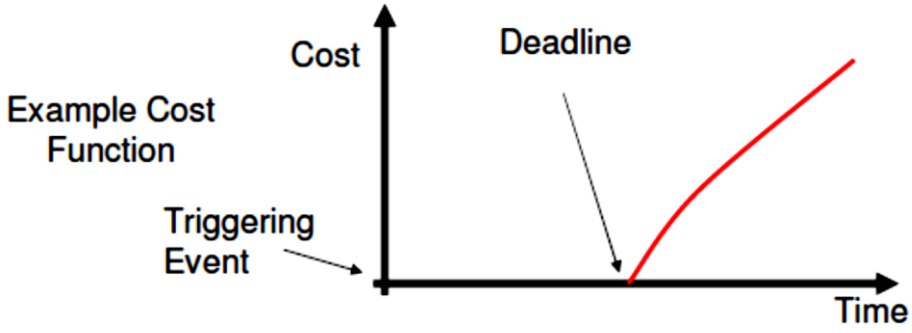
- ♣ A system whose operation is degraded if results are not proceeded according to specified timing requirements.
- ♣ System response occurs within a specified deadline. Failure to meet such timing requirement can have catastrophic consequences.
- ♣ Systems where it is absolutely imperative that responses occur within the required deadline.

  Example: Flight control systems, automotive systems, robotics etc.



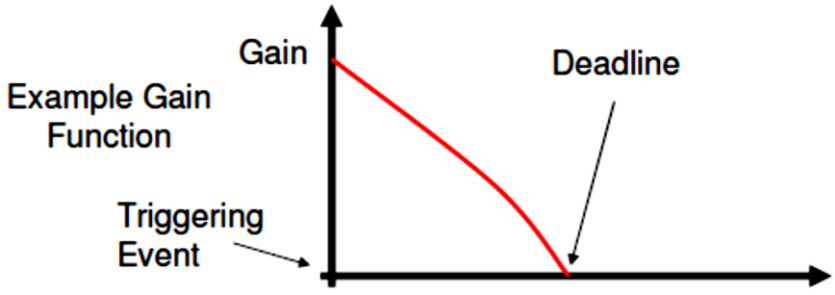
Soft real-time systems

- ♣ A system whose operation is incorrect if results are not produced according to the timing constraints. Catastrophic results will happen then.
- ♣ The response times are important but not critical to the operation of the system. Failure to meet the timing requirements would not impair the system.
- ♣ Systems where deadlines are important but which will still function correctly if deadlines are occasionally missed.
- ♣ **Example:** Banking system, multimedia etc.



Firm teal-time systems

- ♣ There is no value for a response that occurs past a specific deadline. Failure to meet the timing requirements is undesirable.



1.4 Real Time System Structure, Requirements, and Characteristics

RTS Structure:

- Hardware (CPU, I/O Devices, Memory …. etc)
  - Single CPU or more.
  - Clock selection.
- A real time operating system: function as standard OS, with predictable behavior and well-defined functionality.
- A collection of RT tasks/processes (share resources; communicate/synchronize with each other and the environment).

RTS Requirements:

1. Sufficiently fast (processing, communication …. etc).
2. Predictable resource sharing and timing.

RTS characteristics:

- **Large and complex:** Vary from a few hundred lines of assembler or C to 20 million lines of the estimated Space Station Freedom.
- **Concurrent control of separate components:** Devises operate in parallel in the real-world; better to model this parallelism by concurrent entities in the program.
- **Facilities to interact with special purpose hardware:** need to be able to program devices in reliable and abstract way.
- **Mixture of Hardware/Software:** some modules implemented in hardware, even whole system.
- **Extreme Reliability and Safety:** real-time systems typically control the environment in which they operate; failure to control can result in loss of life, damage to environment or economic loss.

- **Guaranteed Response Times:** We need to be able to predict with confidence the worst case response times for systems; efficiency is important but predictability is essential.