Distributed & Embedded Real-Time Systems

(0640751)

Lecture (3)

Concepts of Real-Time Computer Control Systems

Prof. Kasim M. Al-Aubidy
Philadelphia University-Jordan

Lecture Outline:

- Concepts of computer control systems.
- Analog and digital control.
- Data acquisition system.
- Sequence control.
- Direct digital control.
- Adaptive control.
- Supervisory control.
- Centralized and distributed computer control.
- Human computer interface.

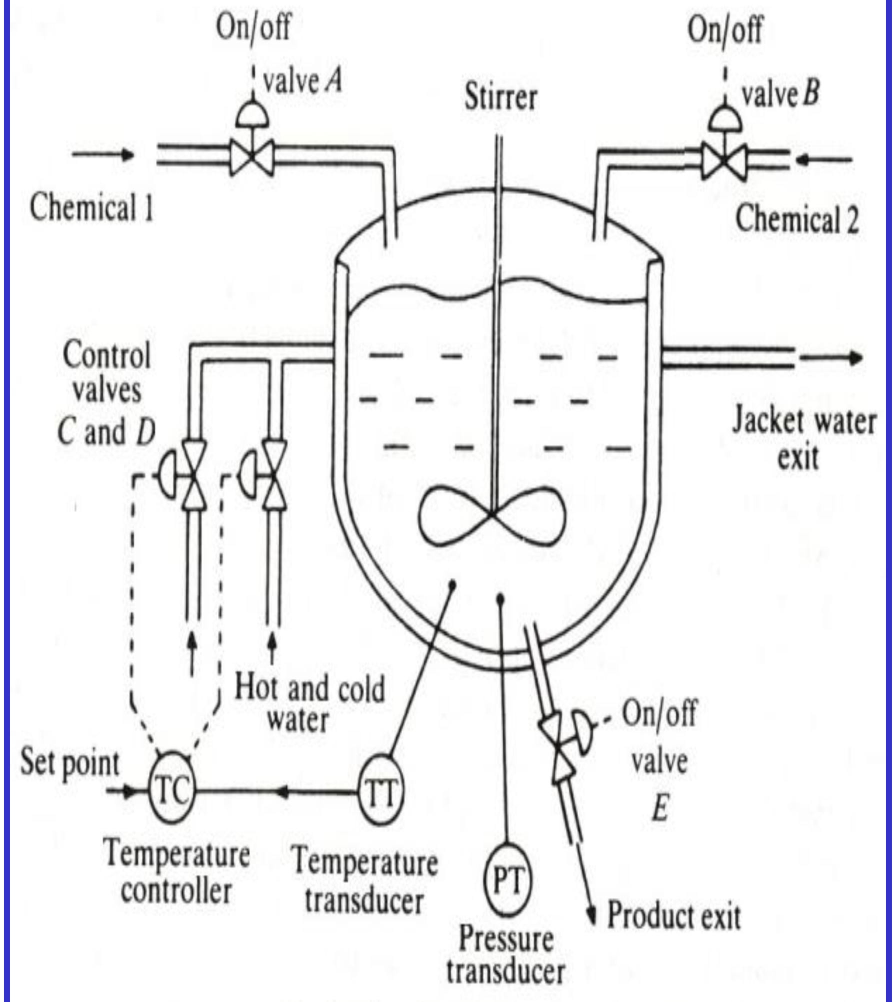
- The activities being carried out by a computer , in a RTS, will include the following:
  - Data acquisition .
  - Sequence control .
  - Direct digital control (DDC).
  - Supervisory control (SC).
  - Data analysis .
  - Data storage .
  - Human – computer interface (HCI).
- The objectives of using a computer in a RTS will include the following :
  - Efficiency of operation
  - Ease of operation .
  - Safety.
  - Improved products
  - Reduction in waste .
  - Reduced environmental impact .
  - Reduction in direct labor .

Sequence Control:

Sequence control systems are widely used in the food processing and chemical industries.

The procedure for this simple reactor are:

1. Open valve A.
2. Check the level of chemical 1.
3. Start the stirrer to mix the chemical reactor.
4. Repeat steps 1 and 2 with valve B.
5. Switch ON the PID controller.
6. Monitor the reaction temp, when it reaches the set-point, start a timer.
7. When the timer indicates that the reaction is complete, switch OFF the controller and open valve C to cool down the reactor contents. Switch OFF the stirrer.
8. Monitor the temp, when the contents have cooled, open valve E to remove the product from the reactor.

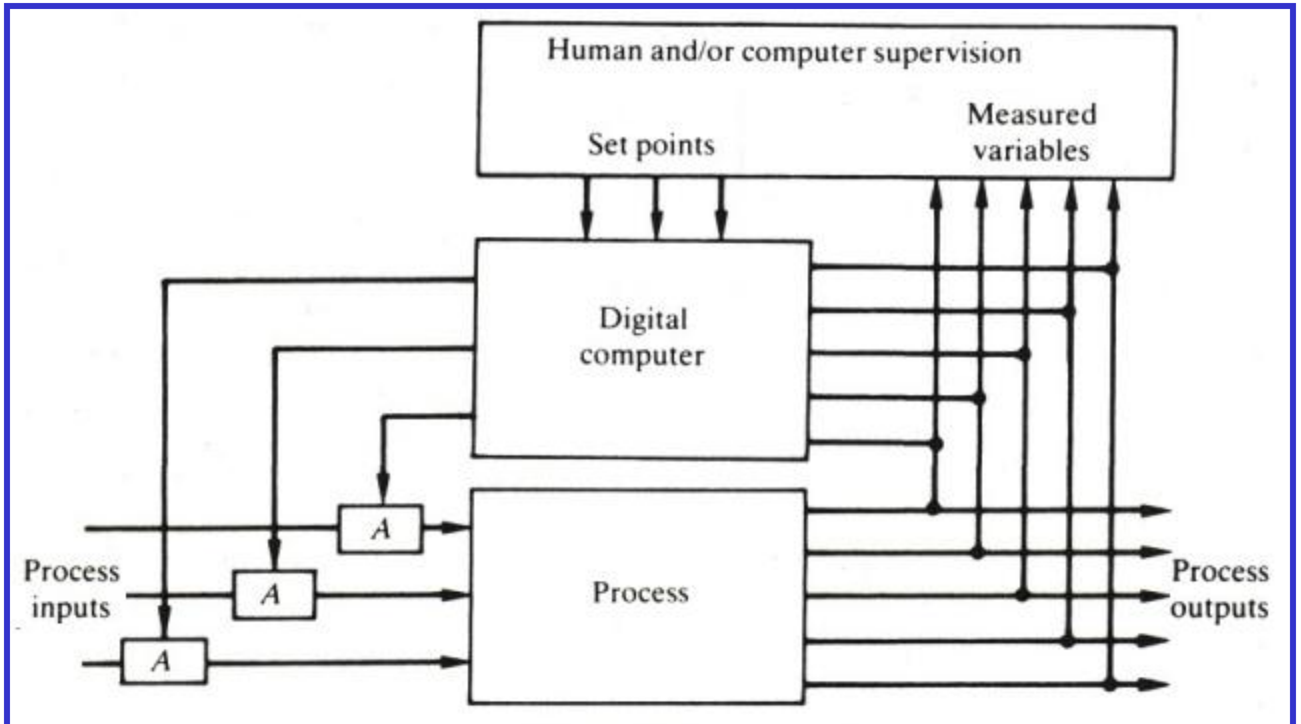


A simple chemical reactor vessel

Direct Digital Control (DDC):

- The computer is in the feedback loop of the system. It is a critical component in terms of the reliability of the system.
- In the event of a failure of the computer, the system remains in a safe condition.
- The advantages for DDC over analog control are:

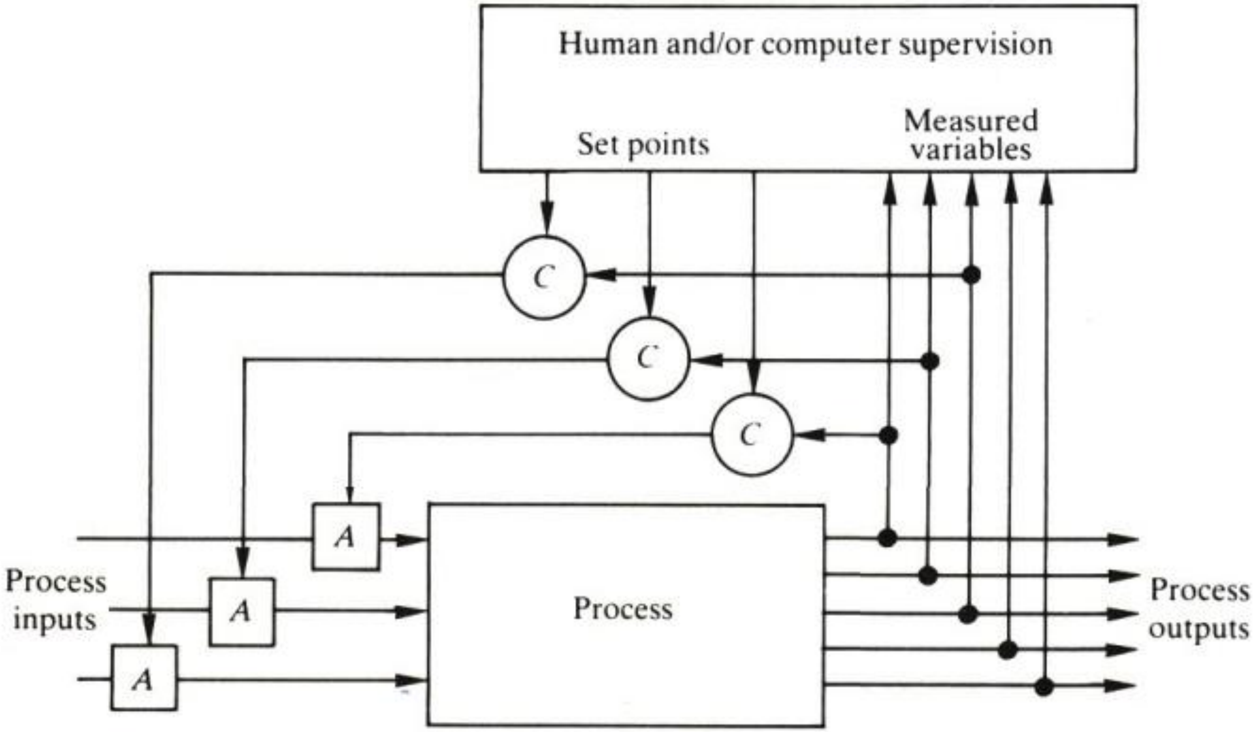
1. Cost. 2. Performance. 3. Safety.



Direct digital control.

Supervisory Control:

- Many of early computer control schemes used the computer in a supervisory role and not for DDC. The main reason for this were;
  1. Computers were not always very reliable and caution dictated that the plant should still be able to run in the event of a computer failure.
  2. Computers were very expensive and it was not economically viable to replace the analog control equipment in current use.

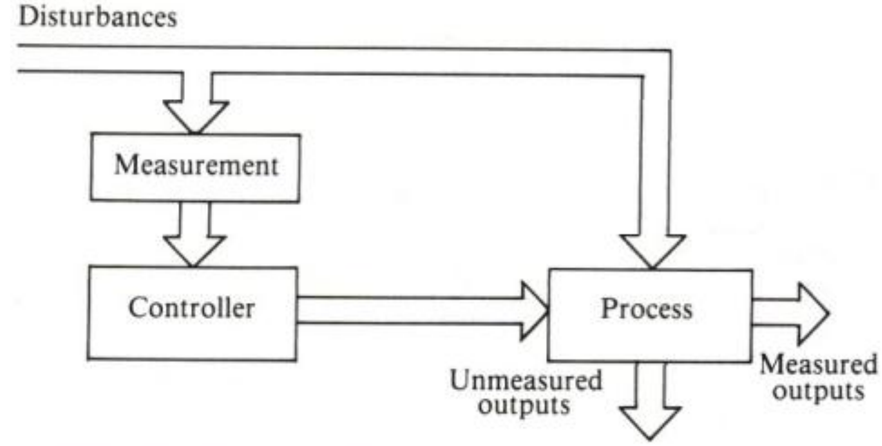


Supervisory control.

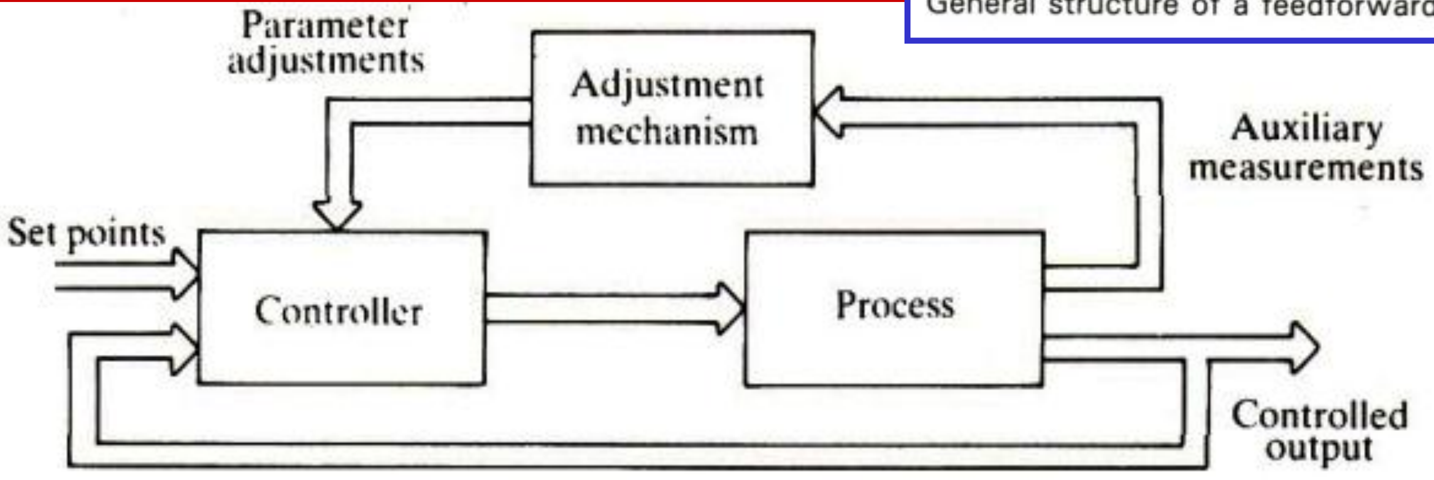
Adaptive Control:

Adaptive control can take several forms. Three of the most common are:

1. Preprogrammed adaptive control.
2. Self-tuning control.
3. Model-reference adaptive control.



General structure of a feedforward control configuration.

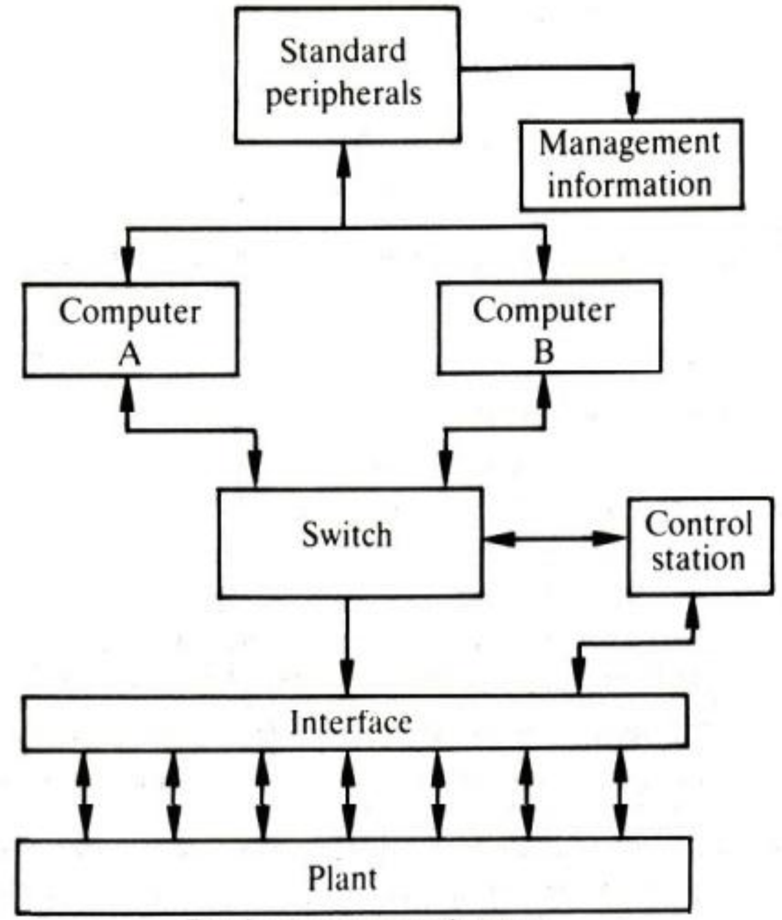


Programmed adaptive control

Centralized Computer System:

- Most of the 1960s computer control systems implied the use of one central computer for the control of the whole plant. The reason for this was largely financial (computers were expensive).
- By 1970 the cost of computer hardware had reduced to such an extent that it became feasible to consider the use of dual computer systems.
- Automatic failure and change-over equipment when used becomes a critical component.
- The continued reduction of the cost of hardware and the development of the microprocessor has made multi-computer systems feasible. These fall into two types:

1. Hierarchical systems : tasks are divided according to function, e.g.: one computer performing DDC.
2. Distributed systems : many computers perform essentially similar tasks in parallel.



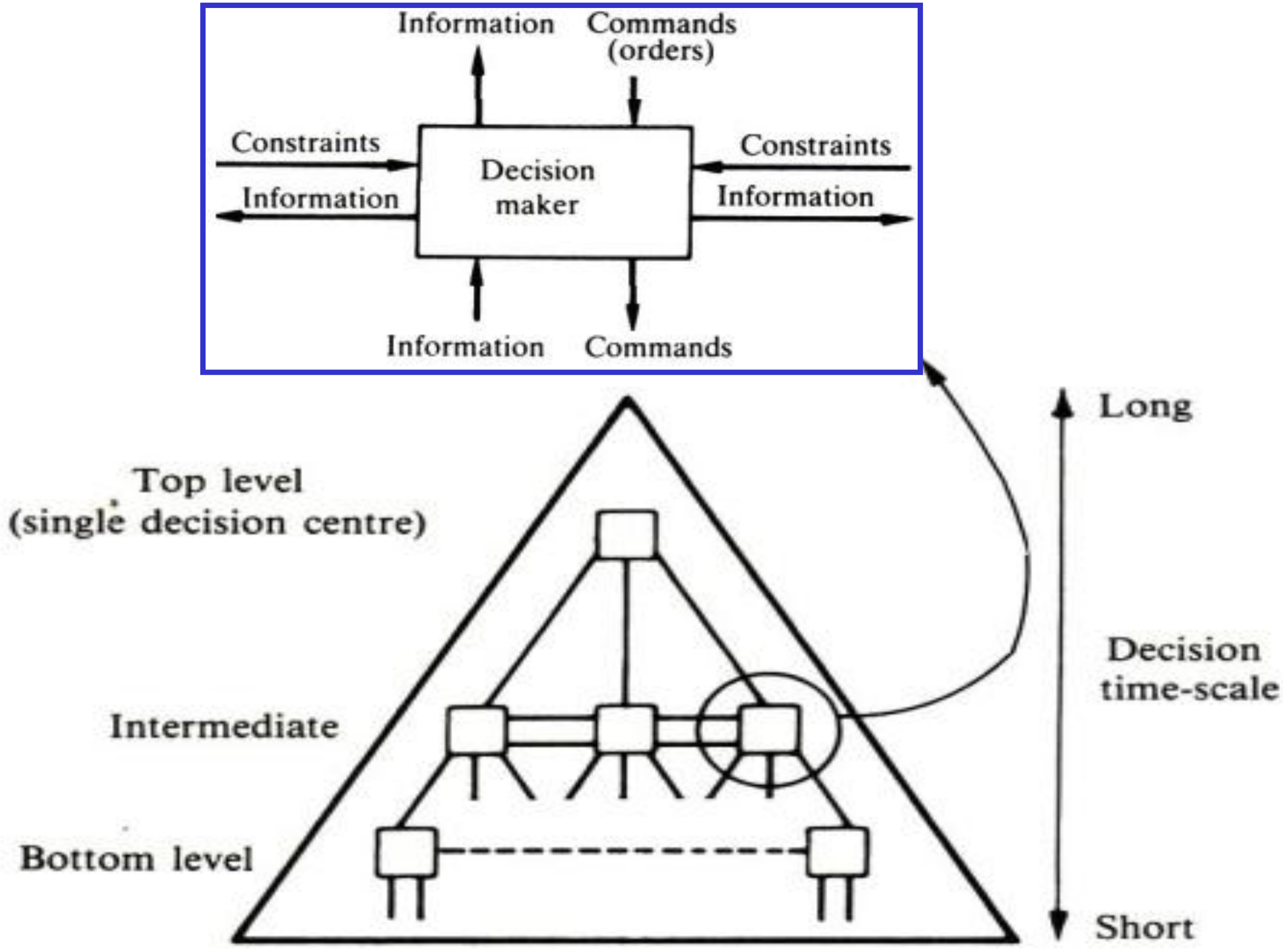
Dual computer scheme

Mlti-Compter Systems:

- Several computers can be configured for real-time computer control applications.
- These include dual computer systems to increase reliability, and distributed and hierarchical configurations.

1. **Hierarchical Systems:** tasks are divided according to function, for example; one computer performing DDC, other performing sequence control, other performing supervisory control..
2. **Distributed Systems:** many computers perform essentially similar tasks in parallel.

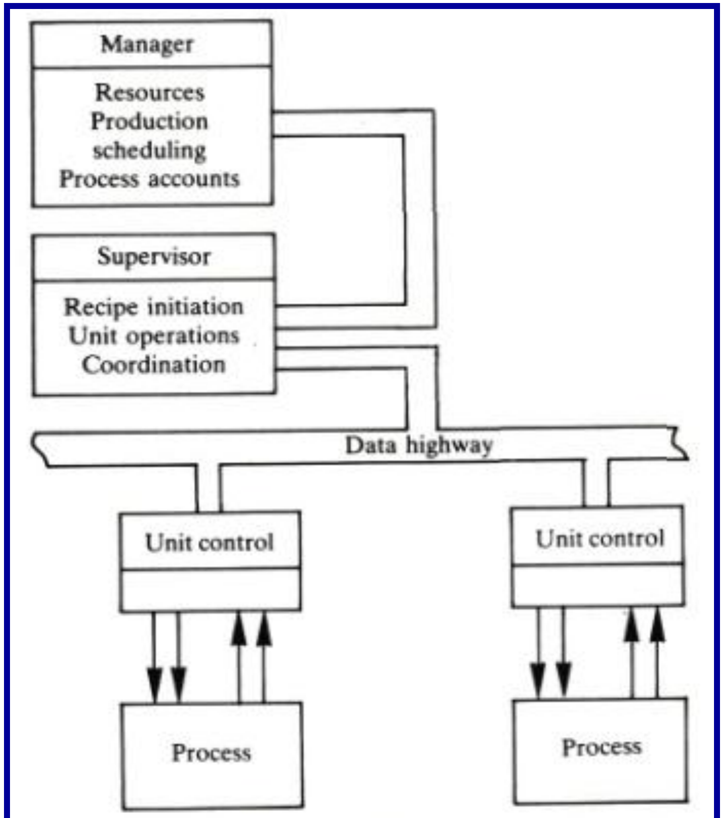
Hierarchical Decision Making:



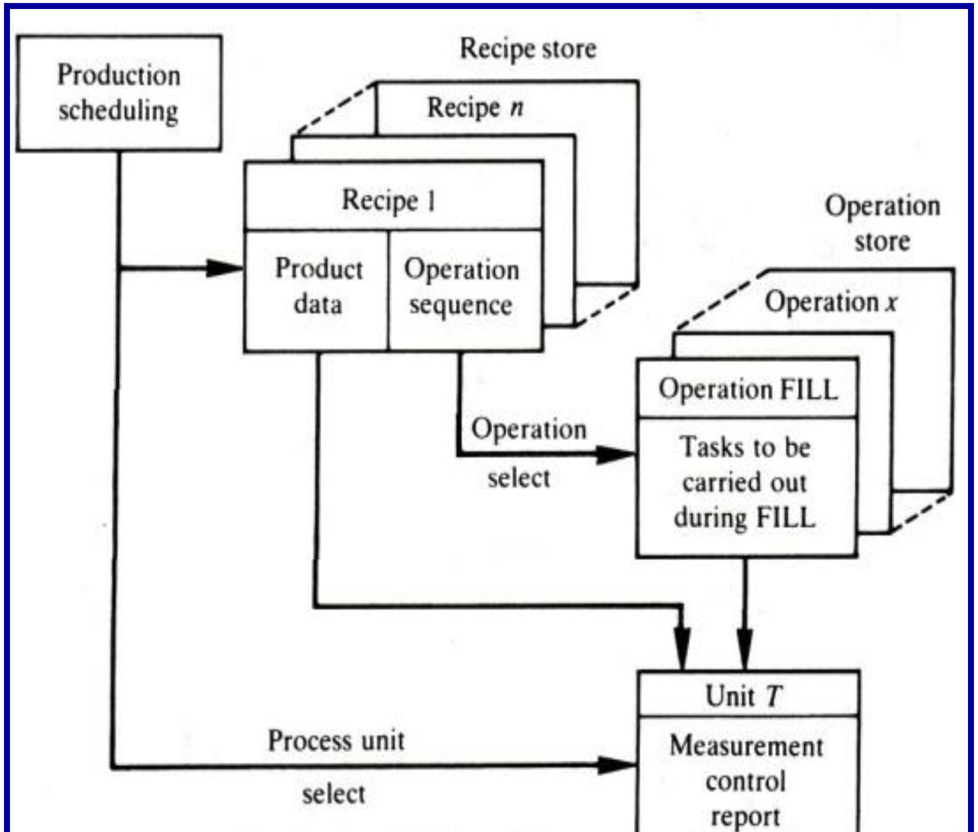
Hierarchical decision-making

Hierarchical System: An Example

- A typical example of a hierarchical system is the batch system given below.
- It has three levels; Manager, Supervisor, and unit Control.
- It is assumed that single computers are used for manager and supervisor functions, and that for each processing unit a single unit control computer is used.



Batch control using a hierarchical system.



Batch control (software)

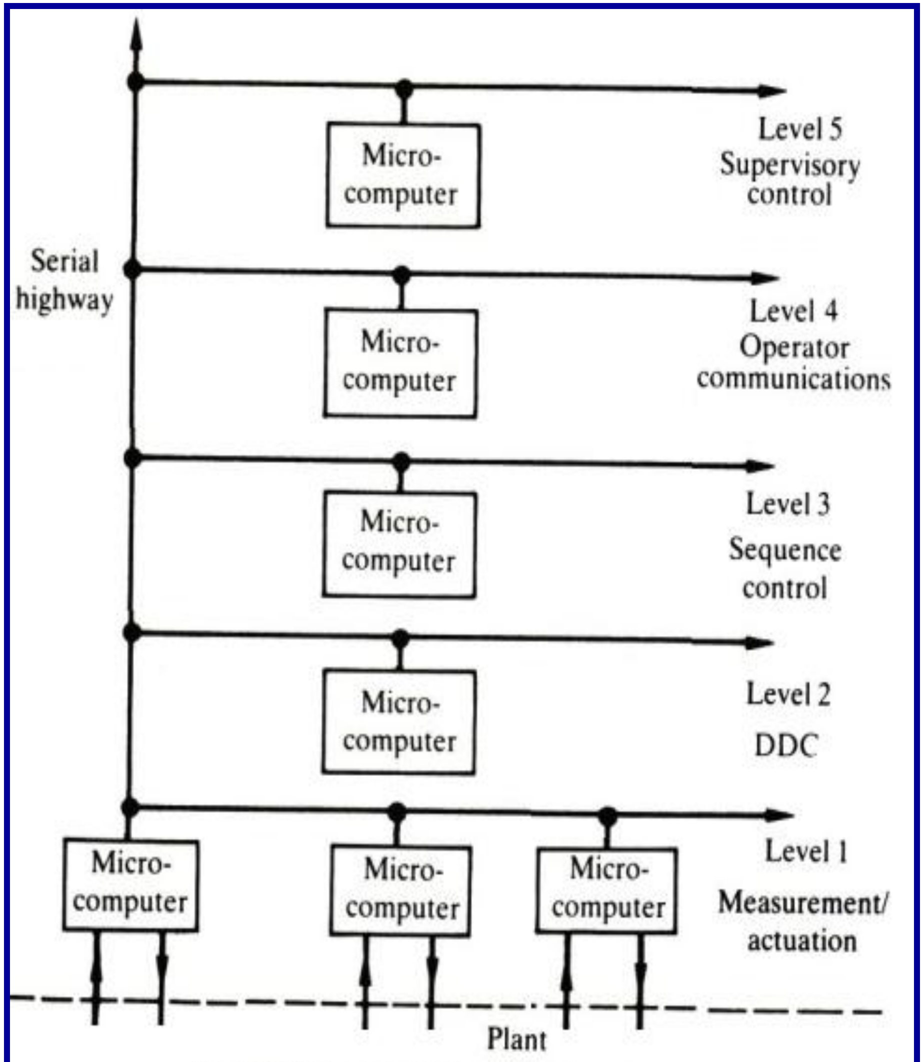
Distributed Systems:

In real-time systems , consider:

- Each unit is carrying out essentially similar tasks to all the other units.
- In the event of failure or overloading of a particular unit all or some of the work can be transferred to other units.

Advantages:

1. Sharing of tasks between μCs.
2. More flexible than using one μC.
3. Failure of a unit will cause less disruption.
4. It is easier to make changes .
5. Linking by serial highway means that the computer units can be widely dispersed .



A distributed and hierarchical system

The Essence of Model Building:

• A **model** is a thoughtful simplification of reality with the objective of explaining a chosen property of reality that is relevant for a particular purpose.

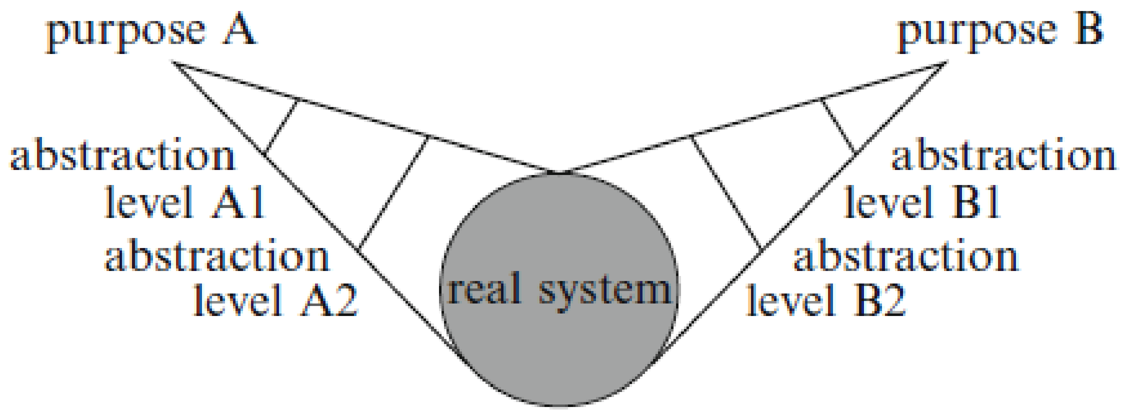
• At the start of any modeling activity, a clear purpose of the model must be established.

> Example: The purpose of a model of behavior of a real-time system is to provide answers to the question at what points in real-time will the computer system produce what kind of outputs. If our computer system is a System-on-Chip (SoC) with a billion transistors, then we must find a hierarchy of behavioral models to meet our purpose.

• Starting with basic-level concepts that are essential for understanding a domain.

• More general concepts can be formed by abstraction, and more concrete concepts can be formed by refinement (improvement).

Hierarchy of models:



Example: Two hierarchies of models that are introduced to serve two purposes, purpose A and purpose B.

•Purpose A could refer to a hierarchy of **behavioral models**, while purpose B could refer to a hierarchy of **dependability models** of the same real system.

•At the top of each hierarchy is the stated purpose. i.e., the questions that must be answered.

•The different levels of the hierarchy are introduced to support a stepwise improvement of the stated question considering more detail.

•At the low end of the hierarchy is the real system. The analysis is substantially simplified if the structure of the model corresponds with the structure of the system.

The major challenge:

- The major challenge of design is the **building of an embedded computer system that provides the intended behavior** (i.e. the service) under given constraints and where relevant properties of this system can be modeled at different levels of abstraction by models of adequate simplicity.
- There are many different purposes that give rise to a hierarchy of models of an ERTS. Examples are: behavior, reliability, man–machine interaction, energy consumption, physical dimension, cost of manufacturing, or cost of maintenance. Out of these, the most important one is the model of behavior.
- In RTS, **behavior specifies the output actions of a computer system as a consequence of the inputs**, the state and the progression of real-time. Output actions and input can be captured in the concepts of input messages and output messages.

Complex Systems:

A system is classified as complex if we are not in the position to develop a set of simple models to explain the structure and behavior of the system. Examples for complex systems are the earth's climate and weather, the global economy, living organisms, and many others.

- The four strategies to simplify a complex system in order that it can be processed by the limited cognitive capabilities of humans are:
  1. abstraction,
  2. partitioning,
  3. isolation, and
  4. segmentation.

How Can We Achieve Simplicity?

Simplicity is achieved by considering the following design principles:

1. Principle of Abstraction: The introduction of a component (a hardware/software unit) as a basic structural and computational unit makes it possible to use.
2. Principle of Separation: it is sometimes called principle of partitioning.
3. Principle of Causality: The deterministic behavior of basic mechanisms makes it possible that a causal chain between a cause and the consequent effect can be established.
4. Principle of Segmentation: it suggests that hard-to-understand behavior should be decomposed into a serial behavioral structure such that a sequential step-by-step analysis of the behavior becomes possible. Each step requires only the investigation of the limited context that is of relevance at this step.

5. Principle of Independence: it suggests that the interdependence of architectural units should be reduced to the necessary minimum that is required by the application.
6. Principle of Observability: it suggests methods to observe the external behavior of any component without a probe effect.
7. Principle of a Consistent Time: it suggests that a global time base should be introduced in the distributed computer system such that system can be established on the basis of global time-stamps.

Characteristics of simple versus difficult tasks

Characteristics of a simple task	Characteristics of a difficult task
Static: The properties of the task do not change over time.	Dynamic: The properties of the task are time dependant.
Discrete: The variables that characterize the task can only take values from discrete sets.	Continuous: The domain of the variables is continuous.
Separable: Different subtasks are nearly independent. There is only a weak interaction among tasks.	Non-separable: Different subtasks are highly interactive. It is difficult to isolate the behavior of a single task.
Sequential: Behavior can be understood by a sequential step-by-step analysis.	Simultaneous: Many concurrent processes interact in generating visible behavior. Step-by-step analysis is difficult.
Homogeneous: Components, explanatory schemes, and representations are alike.	Heterogeneous: Many different components, explanatory schemes, and representations.
Mechanism: Cause and effect relations dominate.	Organicism: Behavior characterized by a multitude of feedback mechanisms.
Linear: Functional relationships are linear.	Non-linear: Functional relationships are non-linear.
Universal: Explanatory principles do not depend on context.	Conditional: Explanatory principles are context dependent.
Regular: Domain characterized by a high regularity of principles and rules.	Irregular: Many different context dependent rules.
Surface: Important principles and rules are apparent by looking at observable surface properties.	Deep: Important principles are covert and abstract and not detectable when looking at surface properties.