**Software Engineering**

Summary of Chapter

19 & 20

MD. ABU TALHA

Reg. - 2012331008

**Summary Of Chapter 19(***Service-oriented Architecture* ***):***

Service-oriented architecture is an approach to software engineering where reusable, standardized services are the basic building blocks for application systems. Services may be classified as utility services that provide a general-purpose functionality, business services that implement part of a business process, or coordination services that coordinate the execution of other services.

A critical distinction between a service and a component as defined in CBSE is that services are independent and do not have a requires interface.

Task-oriented services are those associated with some activity and entity oriented services are like objects. Utility or business services may be entity or task-oriented but the coordination services are always task-oriented.

Service testing is intended to find defects and demonstrate that a system meets its functional and non-functional requirements. The non-functional behavior of the service is unpredictable because it depends on load. If services have to be paid for as used, testing a service may be expensive.

**Summary of Chapter 20(***Embedded Systems****):***

Embedded systems are designed to handle a particular task within a larger mechanical or electrical system. Today’s embedded systems are everywhere – in hybrid vehicles, aircraft, medical devices, traffic lights, factories and videogame consoles. Embedded systems generally run continuously and do not terminate. There may be physical limitations like power that affect the design of a system.

In embedded systems producer processes collect data and add it to the buffer. Consumer processes take data from the buffer and make elements available.

The design process for embedded systems is a systems engineering process that has to consider, in detail, the design and performance of the system hardware. Real-time systems are therefore usually designed as cooperating processes with a real-time executive controlling these processes. UML state diagrams may be used to show the states and state transitions in a real-time system.

In most real-time systems, there will be several classes of periodic process, each with different periods (the time between executions), execution times and deadlines (the time by which processing must be completed). Periodic processes are executed at pre-specified time intervals.