(**NB for ST2)**

For you to view the details of that design.

**Component:** Populate the architecture and play a role in achieving a role of achieving objectives.  
  
  
What are different roles of system components?: what are contributing to the objectives of the system/architecture (what u must look at in ST2 Q).

**Views**

The ways in which we can view components.  
  
From OOM:   
-This component has some type of class it can collaborate with.  
-We look at how they depend on eachother  
  
Traditional View  
-We just look at how the componenet performs a function to complete an object of the system.

Process  
-look atr processes

**Component-Level Design**

What steps are taken when designing the design

1.Identify all classes relating to problem domain (what are classes)  
-is it a part of a class, or is it a class, then it is a component (from business process)  
  
2. Identify Business Rules.  
  
3.Elaborate on the classes.

(**WE** will be asked give components (not these 3 steps) – but we must use them to actually get/identify them)

**Specialised component Design**

**-Can be used as general advise for** components

**Component for WebApp**

A cohesive function that manipulates content or provides some type of data processing.  
**-Anything that has content/data processing/manipulation is a WebApp (NB). This is for the what type of component Q.**

It is cohesive

Web apps can be **developed in parallel to ensure consistency**

**Mobile component**

-interface  
-business layer: consume components  
-Data layer  
(These 3 are mobile components)

**Component-Based Development (NB need to know very well – 100% in ST2)**

We have to build high quality systems in a short amount of time.

CBSE: **Process that emphasizes reusable components.**-This allows us to develop faster.  
-Reduce Lead time: It is faster to build a new application from a pool of existing components.  
-Greater ROI: Sometimes it is cheaper to purchase a component, than developing it yourself.  
Leveraged costs of developing components: Reusing components in many applications allows the cost to be spread.

Enhanced Quality: If you learn from pre-existing components (where they success or fail), allows improved components ect  
Maintenance of components: You can maintain components up to a point, bit sometimes you have to replace components.  
  
**Component-Based Development (risks of above)**

-Component Selection: Sometimes its difficult to pick the right component of the system environment is weird (falls outside the pattern).  
-Component Integration: Trying to integrate with another society   
-Quality:  
-Security: You do not know what vulnerabilities are with that component  
-Evolution: When you change, sometimes a component becomes too low level/not fasisticated enough for the system

**Domain Engineering (not in ST2)**

Reusing domain knowledge so we can develop a new system.  
  
Fundamentally experience allows for better doing of something better.  
  
-Identify:  
-Construct:  
-Catalogue:  
-Disseminate: