Component-based software engineering (CBSE) is an approach to software development that relies on the reuse of entities called ‘software components.

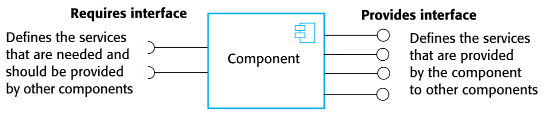
CBSE essentials:

- ✧ Independent components specified by their interfaces.
- ✧ Component standards to facilitate component integration.
- ✧ Middleware that provides support for component inter-operability.
- ✧ A development process that is geared to reuse.

A software component is a software element that conforms to a component model.

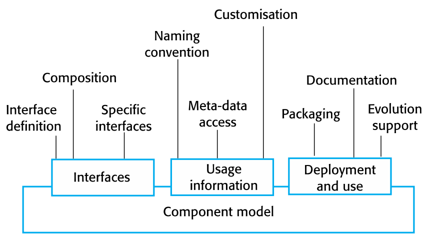
A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only.

- ✧ Provides interface:
  - ▪ Defines the services that are provided by the component to other components.
- ✧ Requires interface:
  - ▪ Defines the services that specifies what services must be made available for the component to execute as specified.


Requires interface
Defines the services that are needed and should be provided by other components
Component
Provides interface
Defines the services that are provided by the component to other components

A component model is a definition of standards for component implementation, documentation and deployment. The component model specifies how interfaces should be defined and the elements that should be included in an interface definition.

Elements of a component model:



To use services provided by a model, components are deployed in a container.

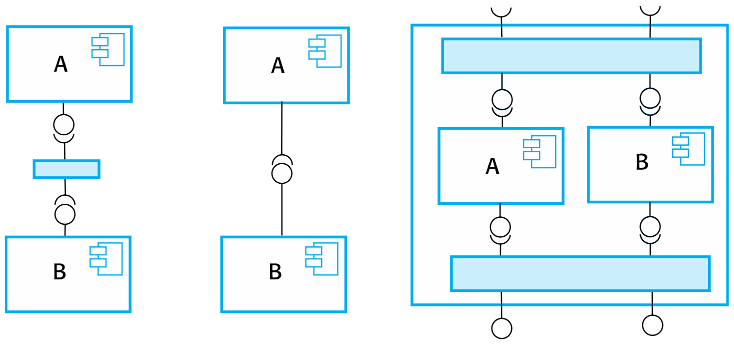
- ✧ Development for reuse
  - This process is concerned with developing components or services that will be reused in other applications. It usually involves generalizing existing components.
- ✧ Development with reuse
  - This process is the process of developing new applications using existing components and services.

Component composition:

The process of assembling components to create a system.

Types of composition:

- ✧ Sequential composition where the composed components are executed in sequence. (1)
- ✧ Hierarchical composition where one component calls on the services of another. (2)
- ✧ Additive composition where the interfaces of two components are put together to create a new component. (3)



(1) (2) (3)

Glue code: allows components to work together.

Interface incompatibility:

- ✧ Parameter incompatibility where operations have the same name but are of different types.
- ✧ Operation incompatibility where the names of operations in the composed interfaces are different.
- ✧ Operation incompleteness where the interface of one component is a subset of the requires interface of another.

The Object Constraint Language (OCL) has been designed to define constraints that are associated with UML models.