Component and Deployment Diagrams

Both are implementation-type diagrams

Component diagrams

- Modern software development proceeds via components (e.g. Table, data file, executable, DLL, document etc) which is particularly important in team-based development efforts.

Component diagrams

- Why do you model components:
  - Clients can see the structure of the finished system
  - Developers have a structure to work toward
  - Technical writers who have to provide documentation and help files can understand what they are writing about
  - You are ready for reuse

UCSC

Component diagrams

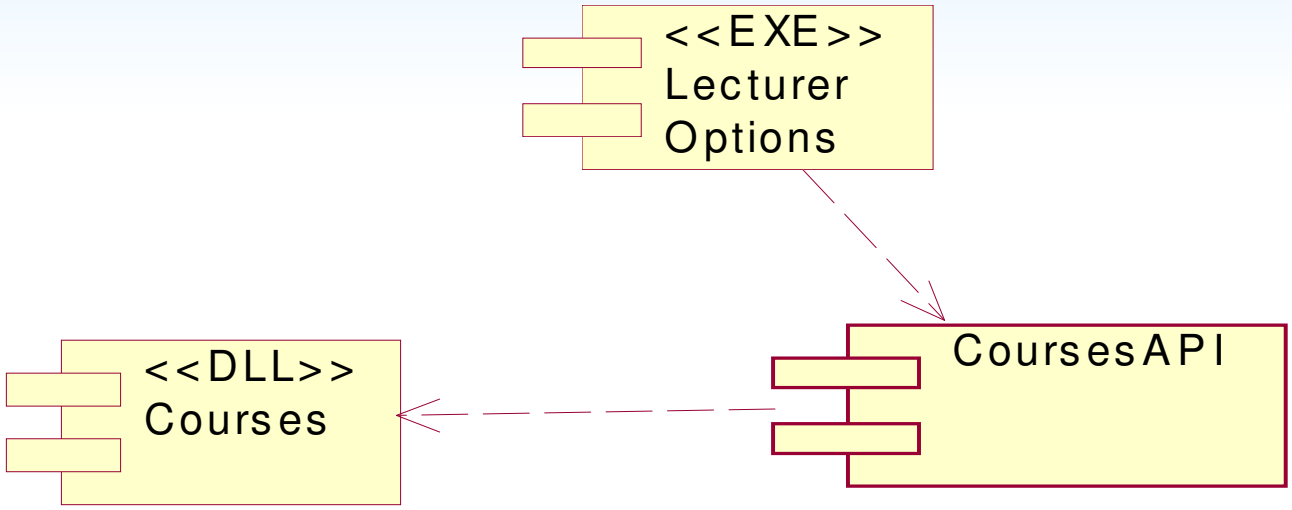
- An Implementation-type diagram that is used to show the physical architecture of the software of the system.

UCSC

Component diagrams

- Component Diagrams contain mainly *components*, *interfaces* and *relationships*.
- Components are related via dependency relationships.
- Run time components show the mapping of classes to run time libraries such as:
  - Java applets
  - Active-X components
  - DLLs

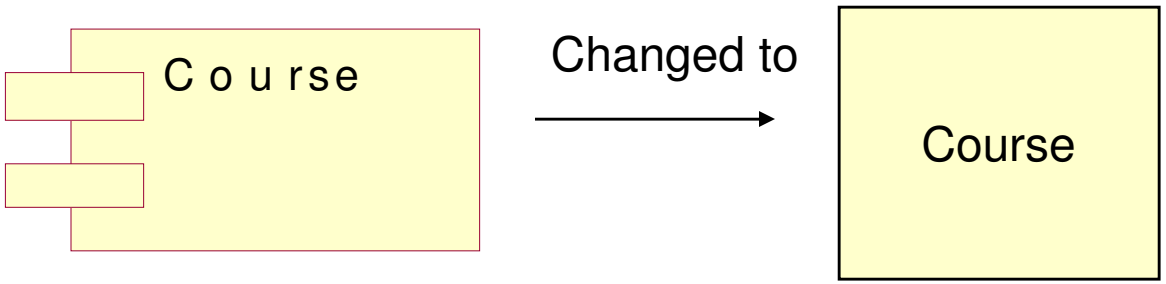
Component diagram
UML 1.x



UCSC

Component diagrams

- Component symbol in UML 2.0 is different.



UCSC

Deployment diagrams

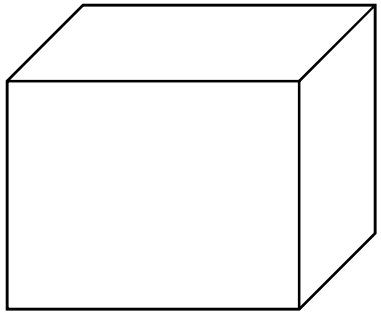
- An implementation type diagram that describe the physical architecture of the hardware and software in the system.
- It can show
  - the computers and devices,
  - their connections with one another
  - the software that sits on each machine

UCSC

Deployment diagrams

- Used for distributed systems only.
- The main hardware item is a *node*, a generic name for any kind of computing resource.

UML node icon



UCSC

Deployment diagrams