Implementation models in UML: Component and Deployment Diagrams

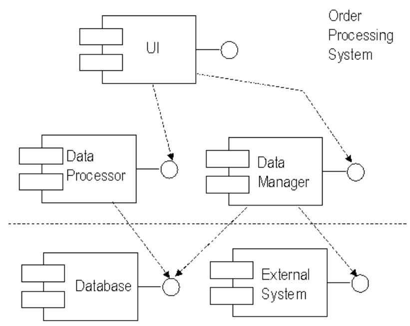
CS3 / SEOC1

Note 14

UML Implementation Diagrams *show aspects of implementation, including source code structure and run-time implementation structure. Implementation diagrams come in two forms: (i) component diagrams show the structure of the code itself; (ii) deployment diagrams show the structure of the run-time system.*

Component Diagrams

A component diagram shows the dependencies among software components, including source code, binary code and executable components. Some components exist at compile time, some exist at link time, and some exist at run time; some exist at more than one time.



Components have...

Interfaces: (can be) represented

Context dependencies:

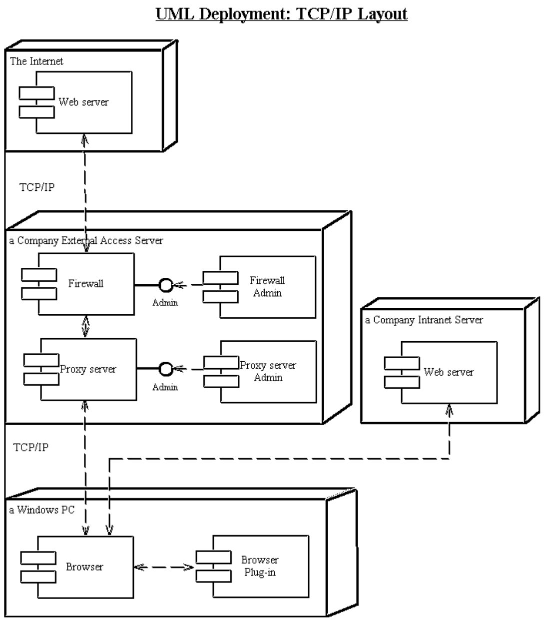
- **implementation-specific:** shown on diagram
- **use-contexts:** may be described *elsewhere.* For example: accompanying documentation, use cases, interactions diagrams (c.f. design patterns), ...

Deployment Diagrams

A Deployment diagram shows the configuration of run-time processing elements and the software components, processes, and objects. Software component instances represent run-time manifestations of code units. Components that do not exist as run-time entities do not appear on these diagrams. These components should be shown on component diagrams.

In other words, a deployment diagram shows your system's hardware, the software installed on that hardware, and the middleware that connects the disparate machines together. A deployment model is a collection of one or more deployment diagrams with their associated documentation.

Example Deployment Diagram



Deployment Modelling should consider:

- What existing systems will system need to interact or integrate with?
- How robust does system need to be (e.g. redundant hardware in case of a system failure)?
- What and who will connect to or interact with system, and how will they do it)?
- What middleware, including the operating system and communications approaches and protocols, will system use?
- What hardware and software will users directly interact with (PCs, network computers, browsers, . . .)?
- How will you monitor the system once deployed?
- How secure does the system need to be (needs a firewall, physically secure hardware, . . .)?

Deployment *Planning* considers:

- How will your system be installed?
  - Who will install it?
  - How long should it take to install?
  - Where can the installation possibly fail?
    - How do you back out if the installation fails?
    - How long does it take to back out?
  - What is your installation window (during what time period can you install your system)?
  - What backups do you need before installation?
    - Do you need to do a data conversion?
  - How do you know that the installation was successful?
- If different versions of the system will be in production at the same time, how will you resolve differences?
- What physical sites do you need to deploy to and in what order?
  - How will you train your support and operations staff?
  - Do you need to deploy a production support system so that the support staff uses their own environment to simulate problems?
- How will you train your users?
  - What documentation, and in what formats and languages, do your users, and support and operations staff need?
  - How will updates to the documentation be deployed?

Modelling Business Process

- Business modelling using nodes and components is an effective means of capturing non-computer based processes and entities.
- This can be done very early in development, to complement the use case model and other business modelling
- components are the business procedures and documents; the nodes ("run-time structure" are the organisation units and resources (human and other) of the business.