Module 7

Deployment Diagrams

Vasco Amaral
vma@fct.unl.pt

Deployment is the process of distributing artifacts over nodes, or artifact instances over node instances

Deployment diagrams

- A type of diagram used to model a system's hardware topology and software deployment
- Models the architecture of a given system in execution time
  - Presents a static view of the configuration in execution time of the processing nodes and of the distribution of the components that are executed in these corresponding nodes

Deployment diagrams

- Examples of nodes:
  - server, client, modem, printer, etc.
- The deployment diagrams show:
  - hardware
  - software (installed in the hardware)
  - middleware (used to connect together the different machines involved)

Deployment diagrams map the software architecture to the hardware architecture

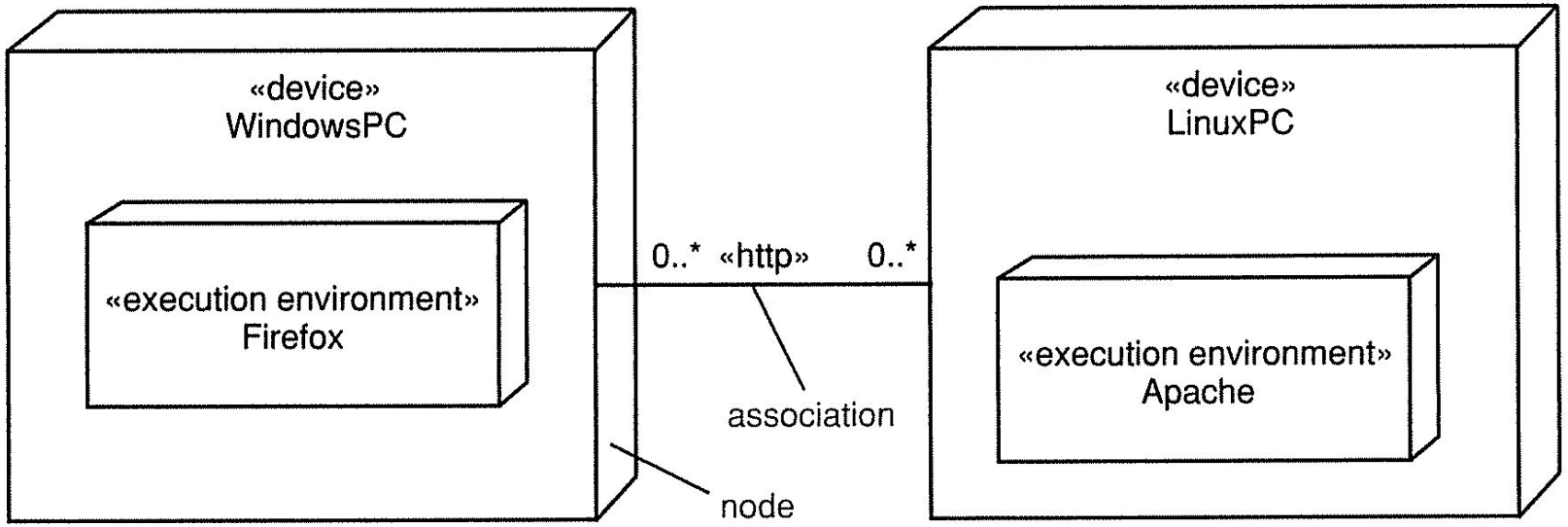
- Two types of installation diagrams:
  - **Descriptive**: contains nodes, relations between nodes and artifacts
    - Example of node: PC
    - Example of artifacts: jar archive
  - **Instantiated**: contains nodes instances, links between instance nodes and artifacts nodes (which can be anonymous)
    - Example of nodes: Miguel's PC
    - Example of artifacts: a particular jar archive

The construction of deployment diagrams is a two-step process

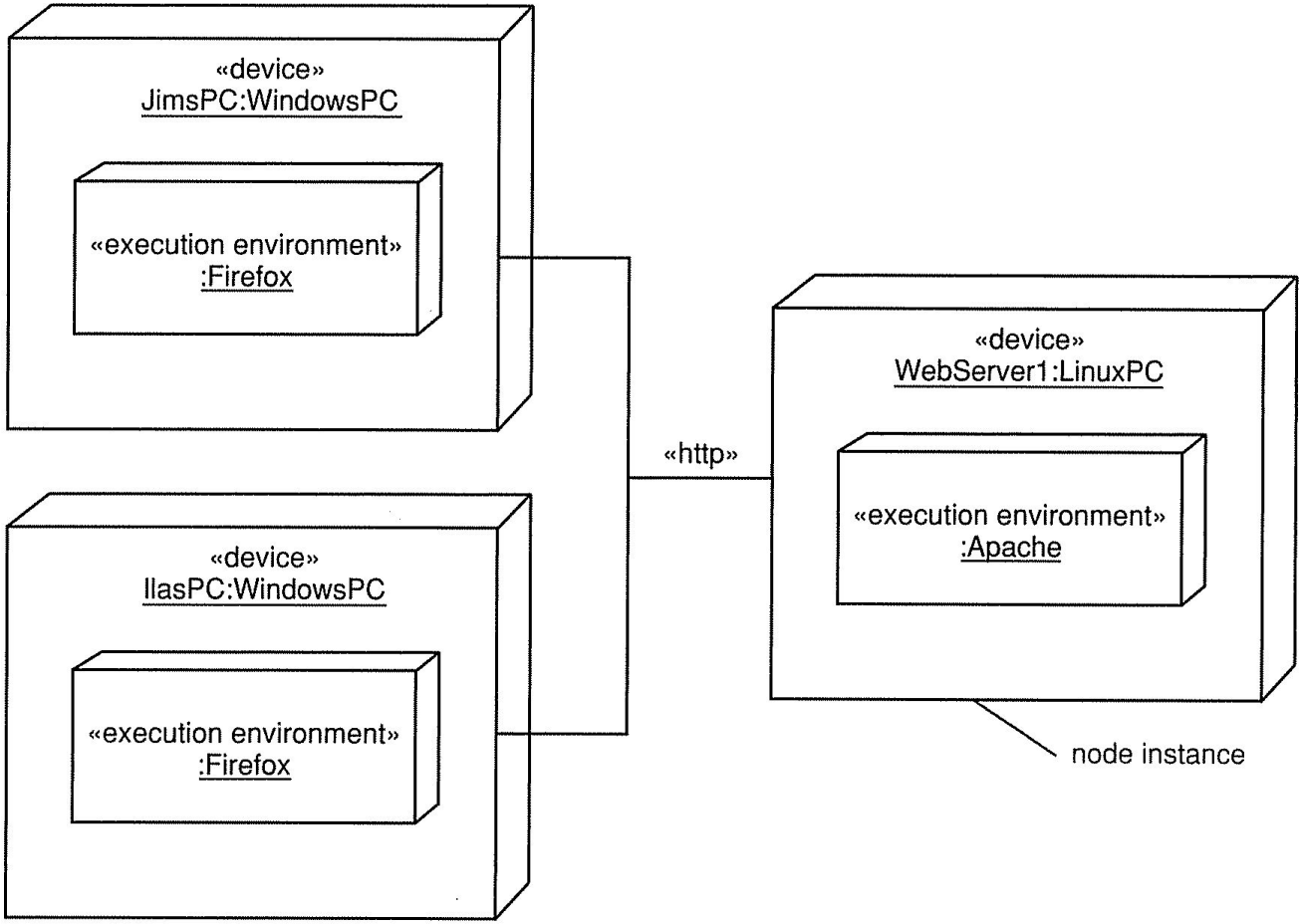
- First version during the design phase (**descriptive diagram**) with the goal to help on the design process of the hardware architecture
- Refinements show one or more instantiation forms, using anonymous instances (**instantiated diagrams**)
  - When the hardware details in the installation place are known, the instantiated diagram can discard anonymous instances and start to use the nodes ids and specific artifacts to be used

A node represents a computational resource

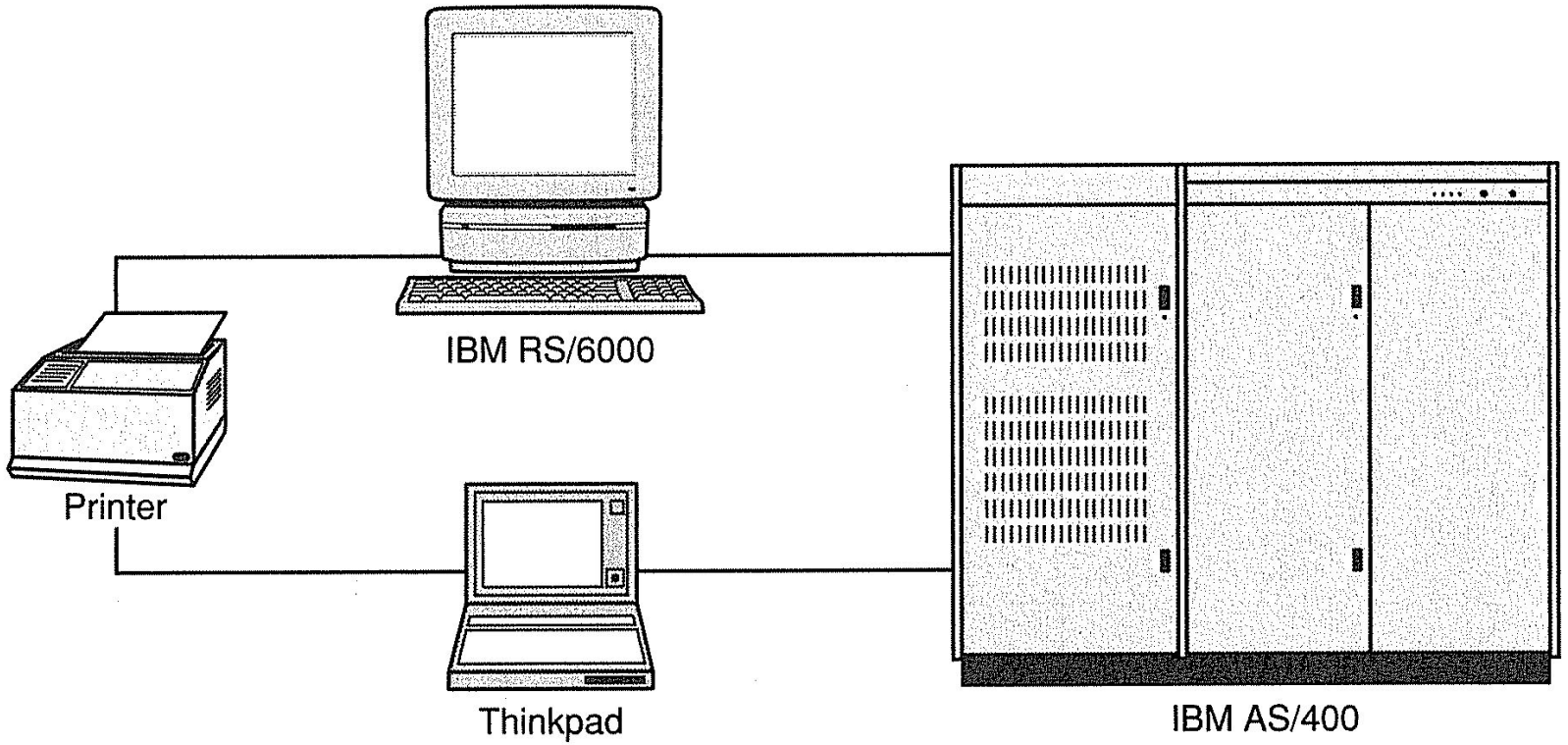
- Artifacts can be deployed in a node for execution
  - <<device>> represents a physical device
  - <<execution environment>> represents an execution environment for software



A node instance represents a specific computational resource



Assign icons to stereotyped nodes in deployment diagrams to increase readability

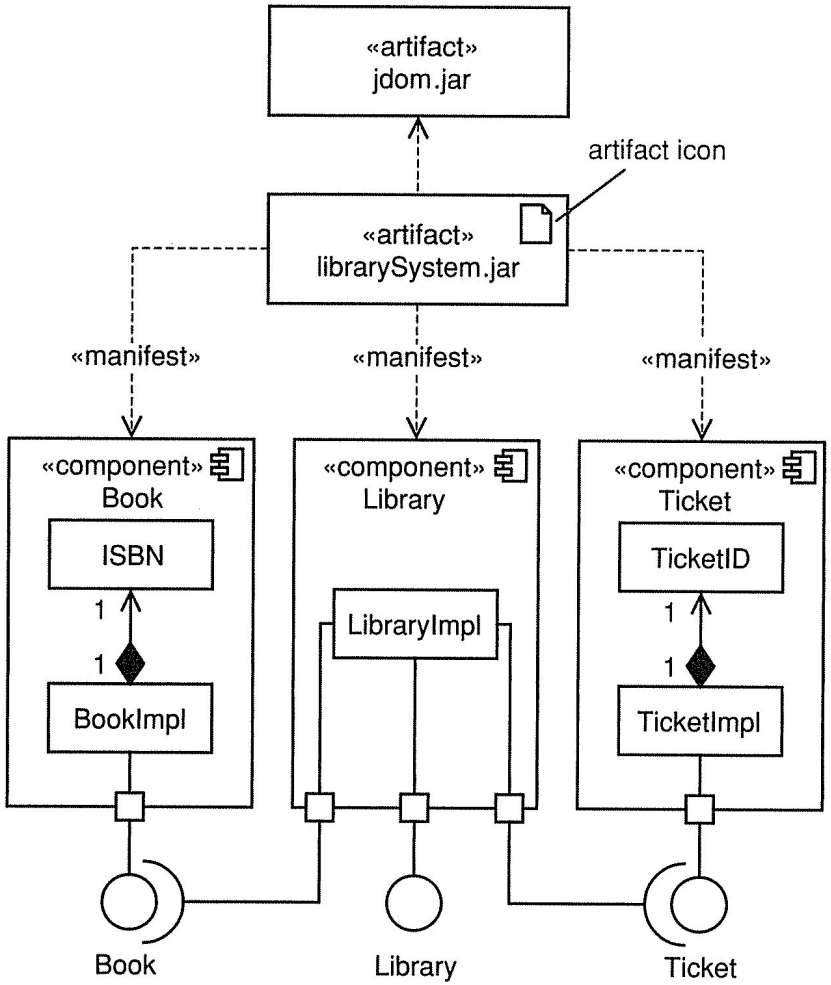


Artifacts represent the specification of real-world things such as a file

- source files
- executable files
- scripts
- database tables
- documents
- outputs of the development process (e.g. a UML model)

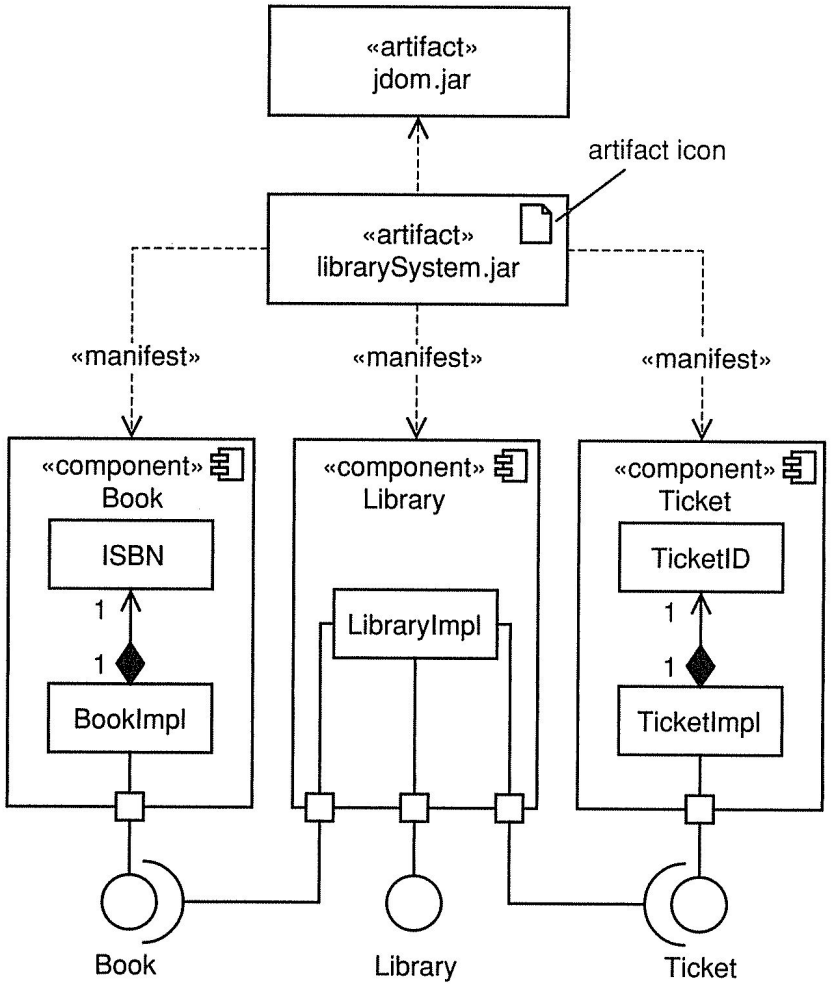
An artifact instance is a specific instance of a particular artifact

- Artifacts can manifest on one or more components
  - librarySystem.jar manifests three white-box components (Book, Library and Ticket)



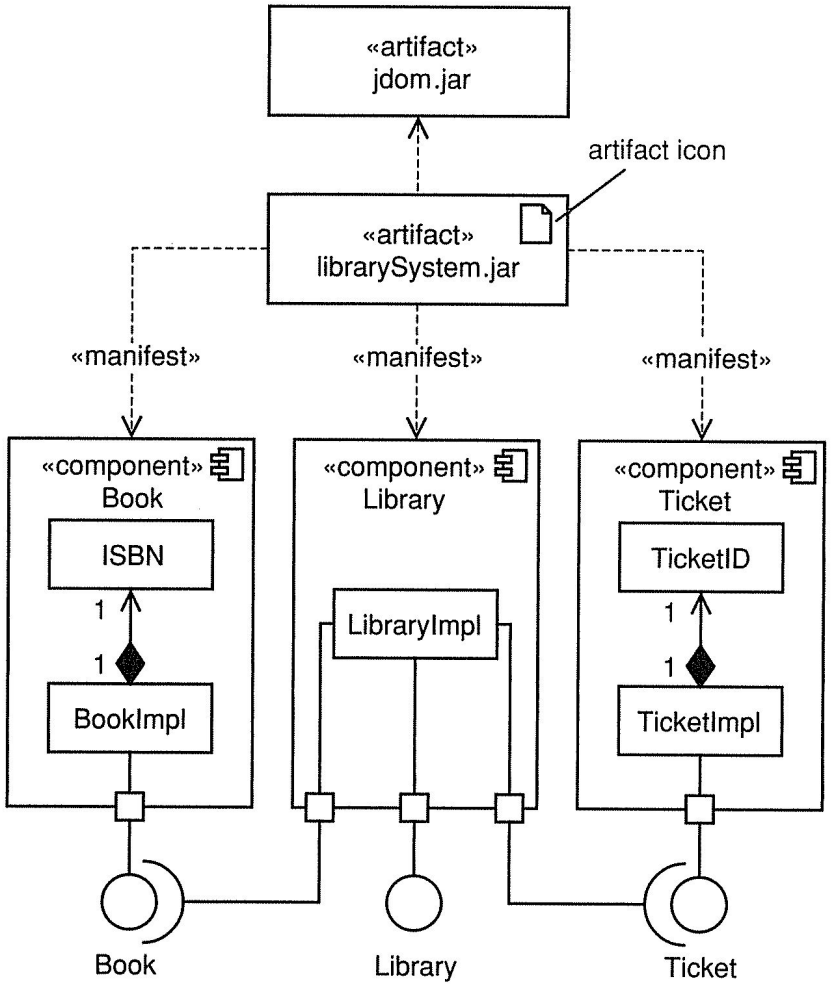
Artifacts may depend on other artifacts

- librarySystem.jar depends on jdom.jar

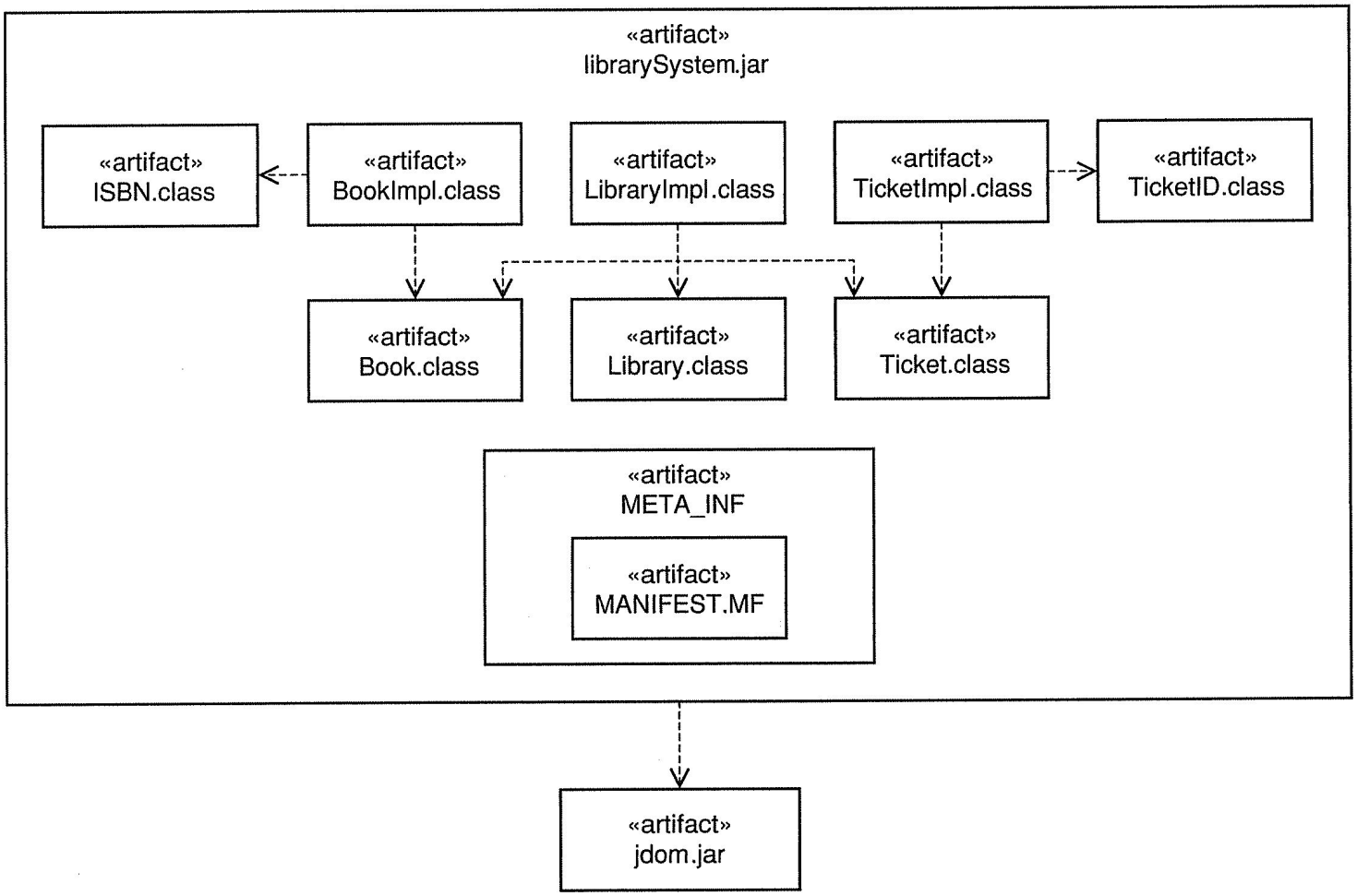


To create this jar, we perform 2 steps

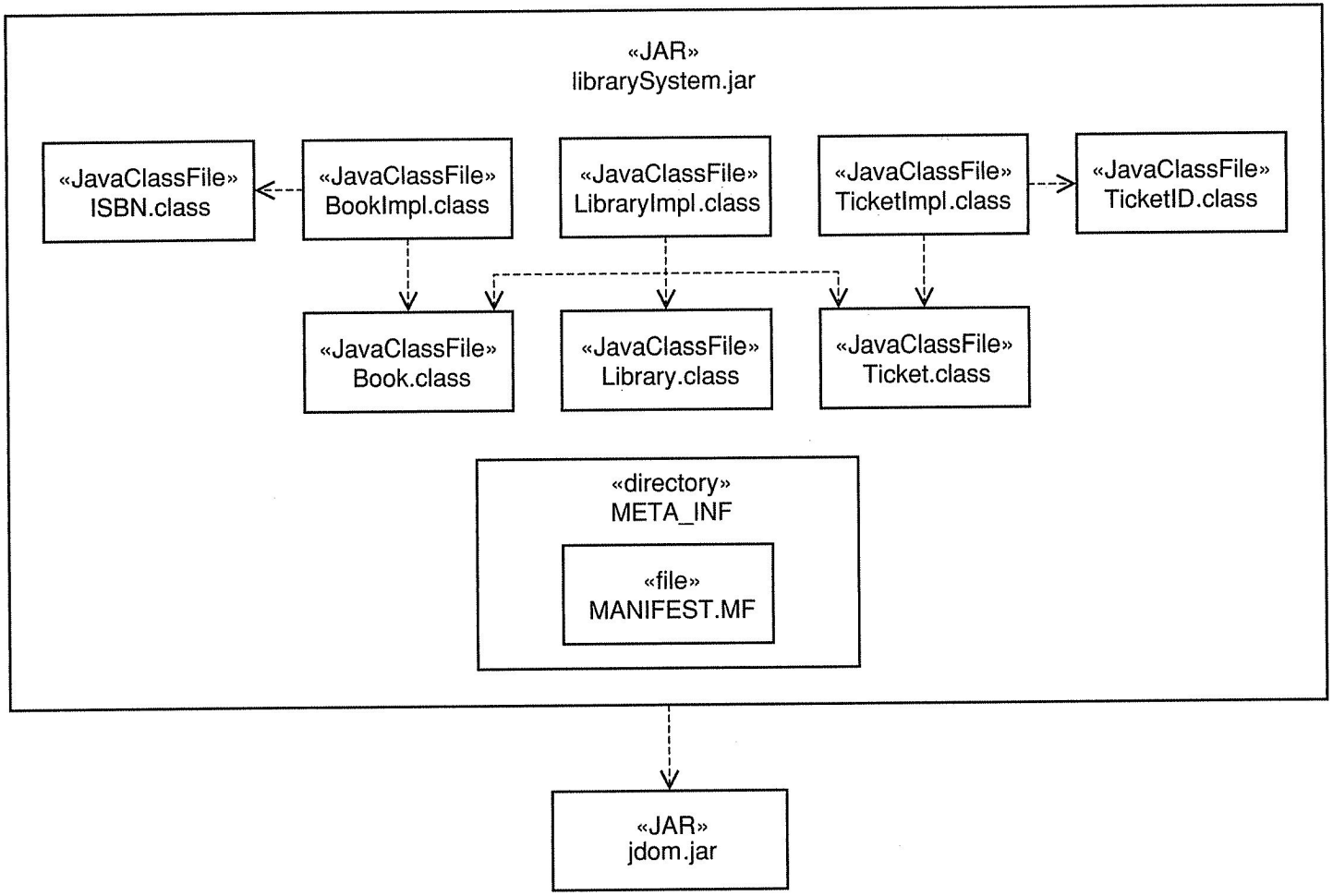
1. Compile the Java source files for the classes Book, ISBN, BookImpl, Library, LibraryImpl, Ticket, TicketId, and TicketImpl
2. Create the JAR file from the compiled files



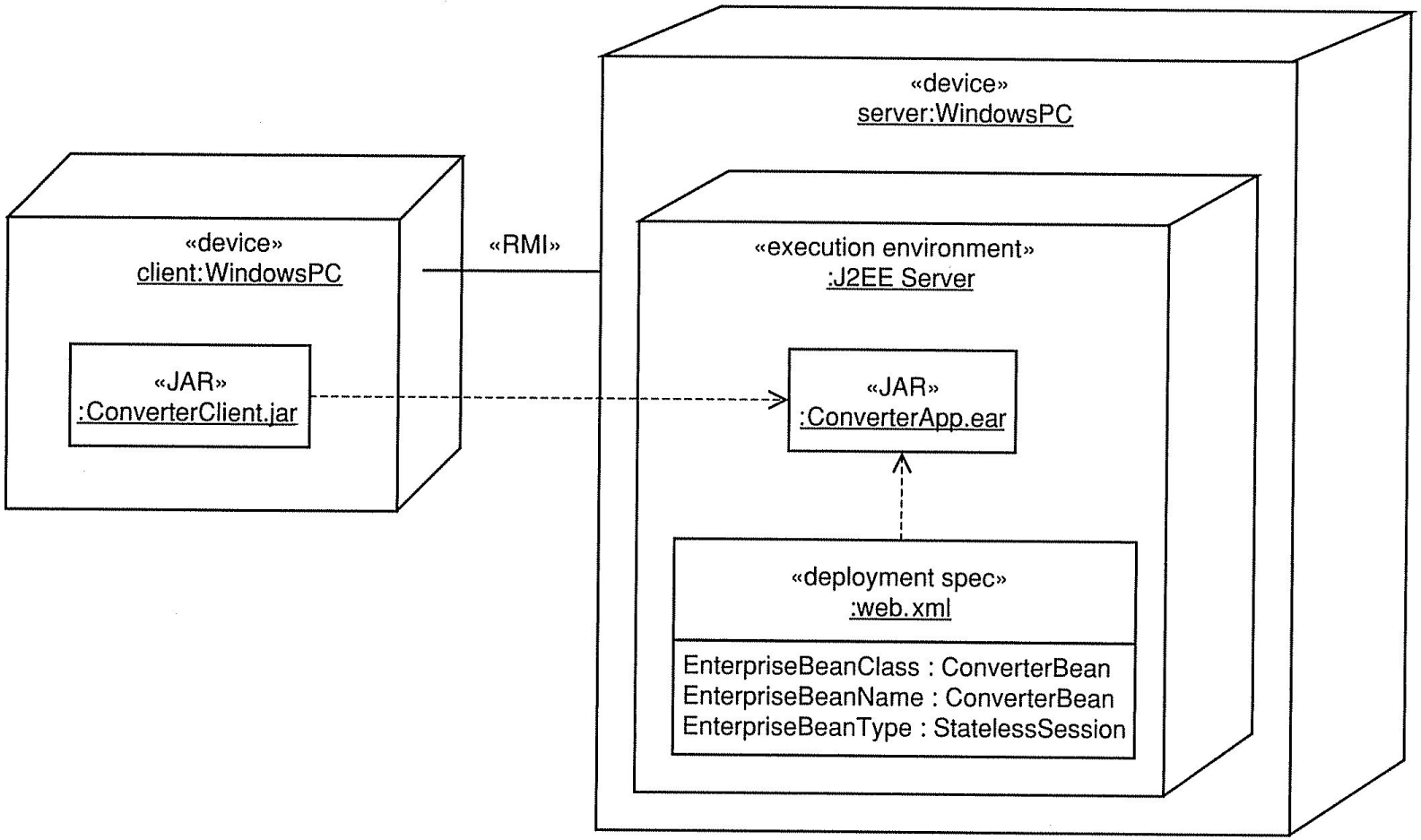
This would be the structure of the corresponding JAR file



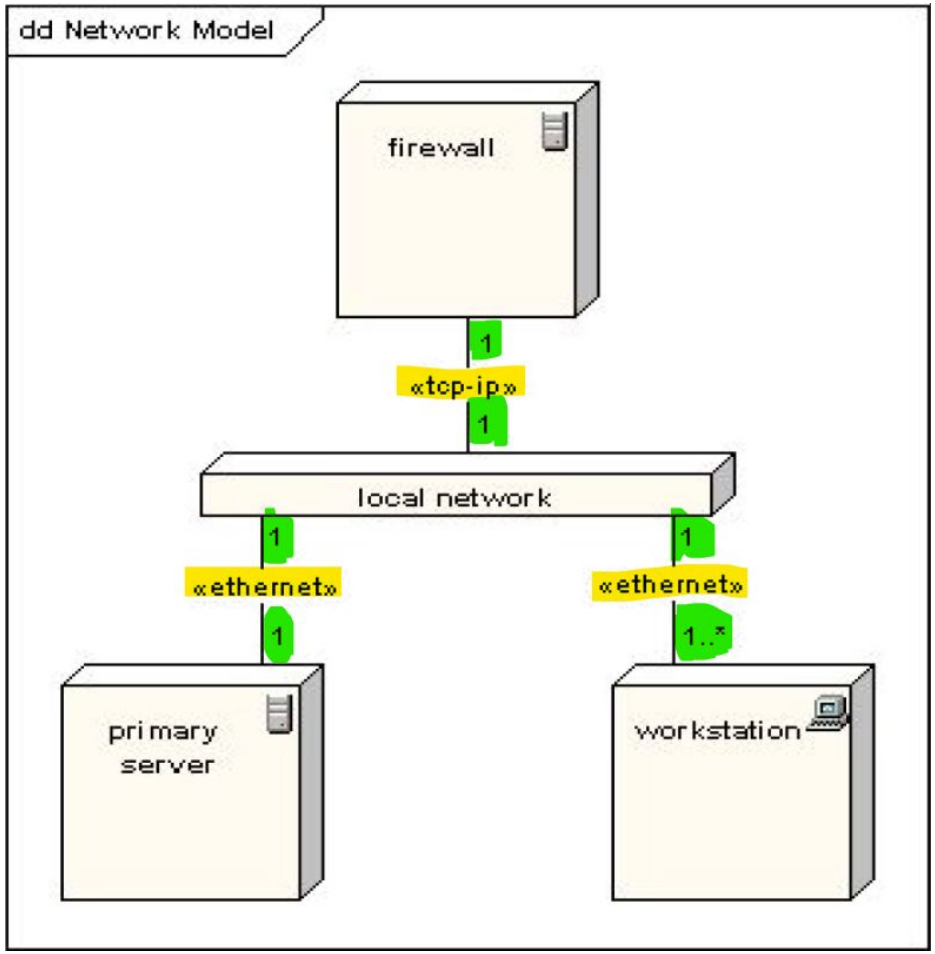
We can make this clearer by applying more specific stereotypes (here, Java stereotypes)



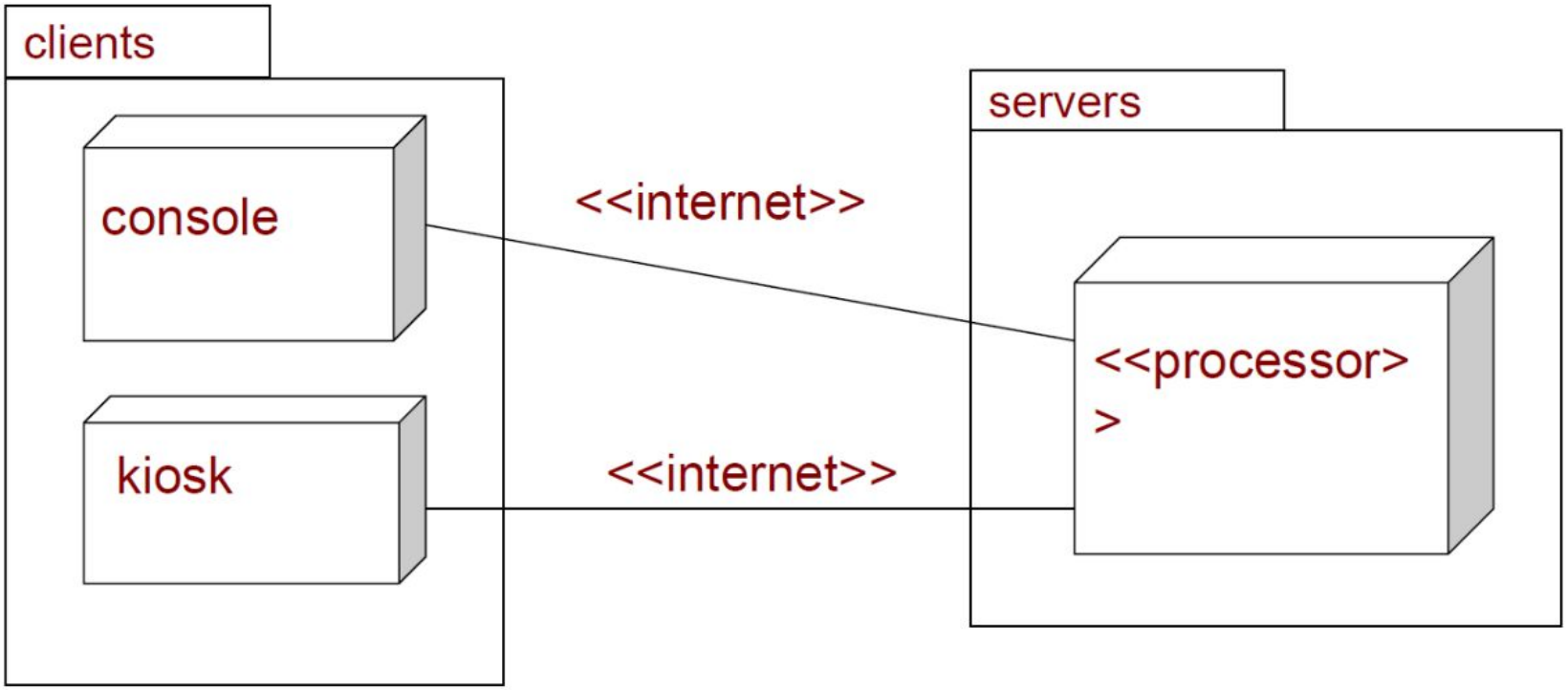
Deployment diagram



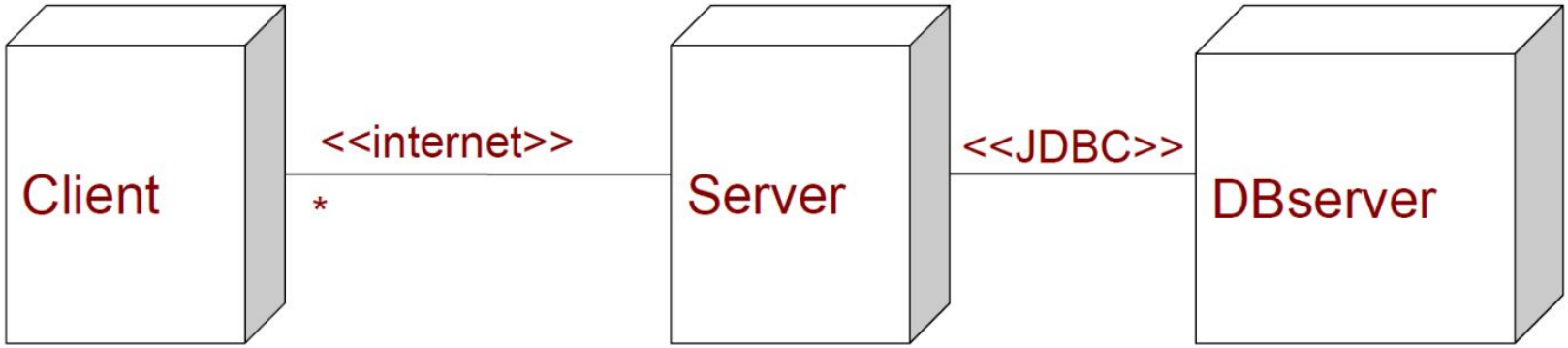
We can represent network protocols and multiplicities



Packages can be used to structure different types of nodes (here, clients *vs.* servers)



Another example, with a client-server architecture



Bibliography

Jim Arlow and Ila Neustadt, “UML 2 and the Unified Process”, Second Edition, Addison-Wesley 2006

- Chapter 24