Package, Component and Deployment Diagrams

PB007 Software Engineering I

Bruno Rossi

19. 12. 2016

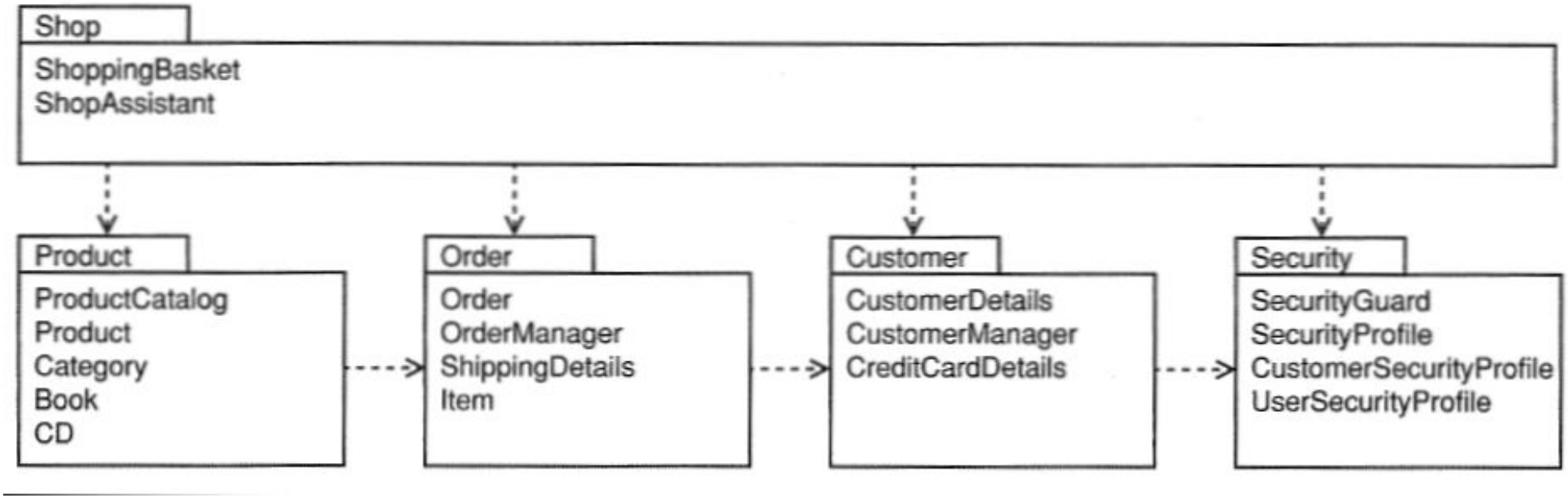
Package Diagrams

Package diagrams display groups (packages) of related elements and the dependencies between them.

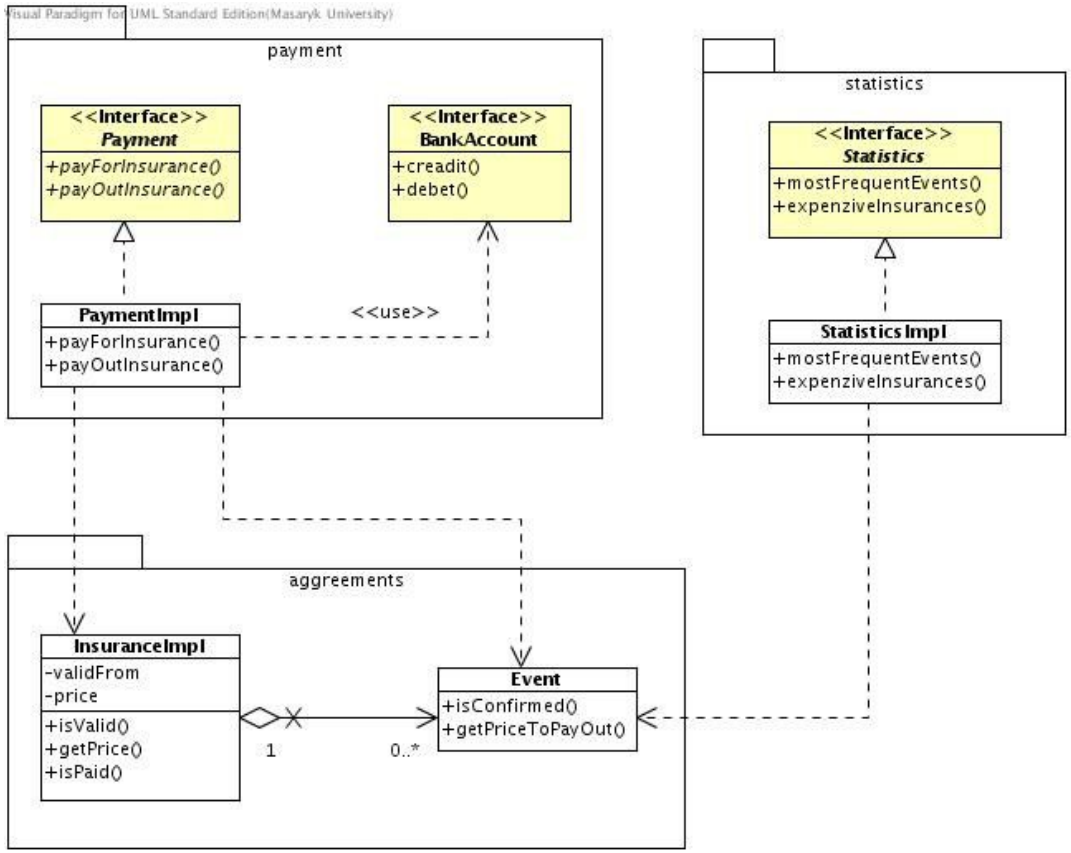
The basic elements:

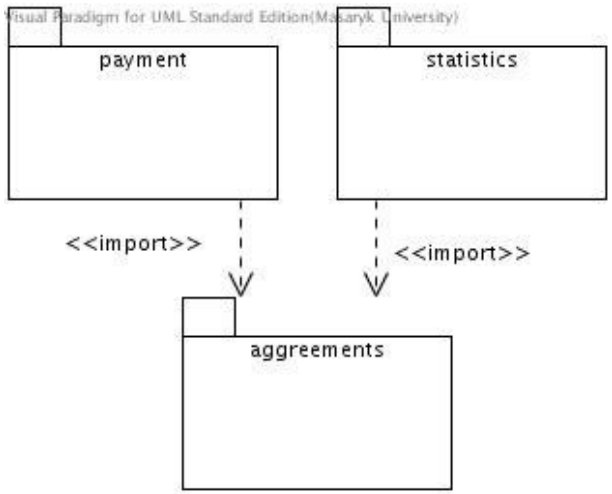
- **Packages** - represent a logical mechanism for grouping related model elements (classes, objects, instances of use, ...), plus they define their namespace.
- **Dependencies** - indicate that the elements in one package depend on elements in another package. Depending on the type it can be further specified as stereotype (`use`, `import`,...)

Package Diagrams - example 1

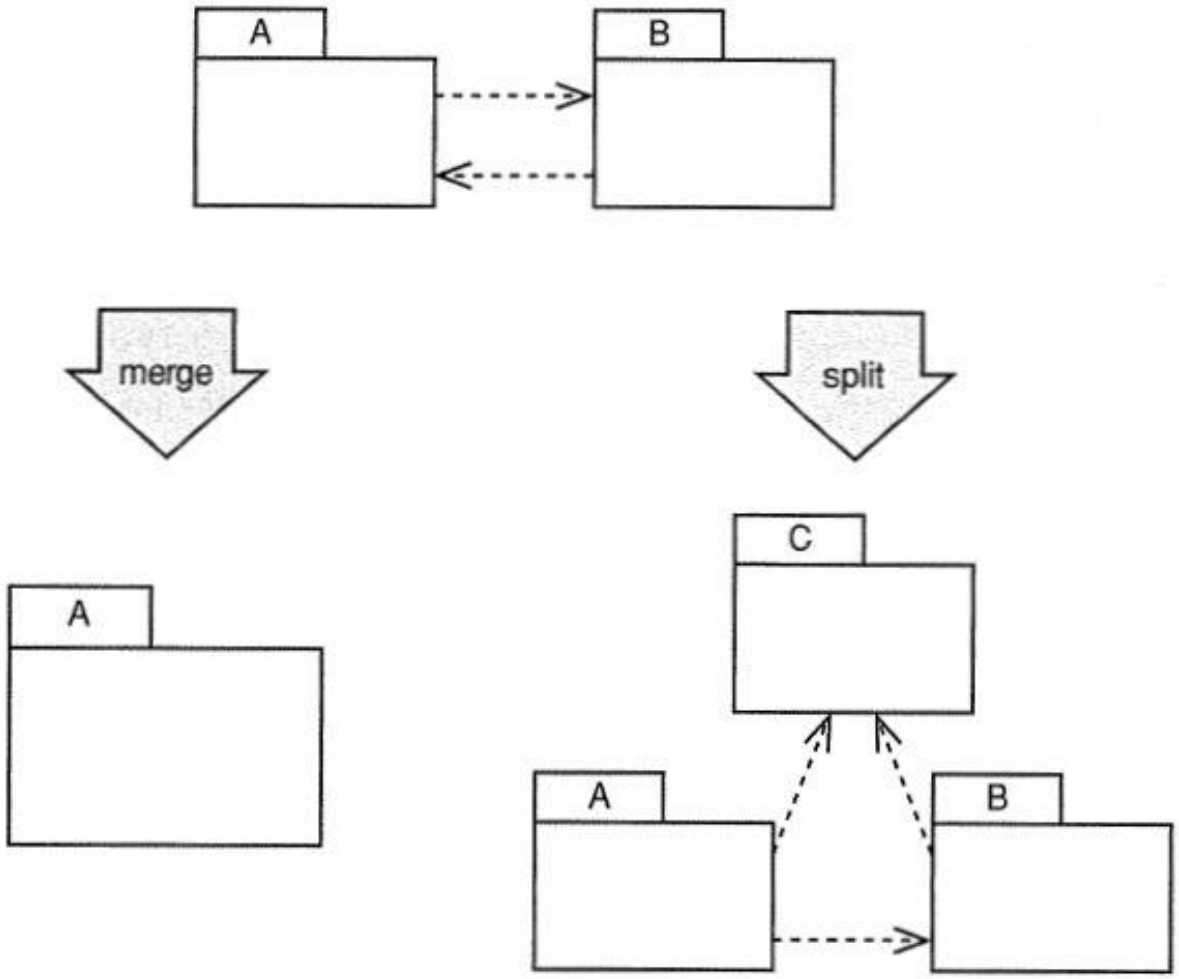


Package Diagrams - example 2





Package Diagrams - circular dependencies



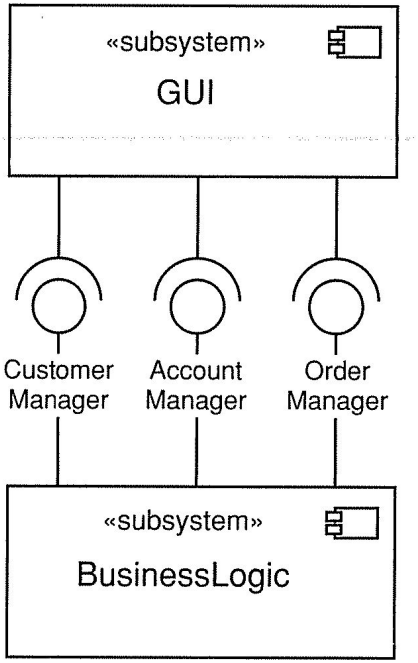
Component diagram

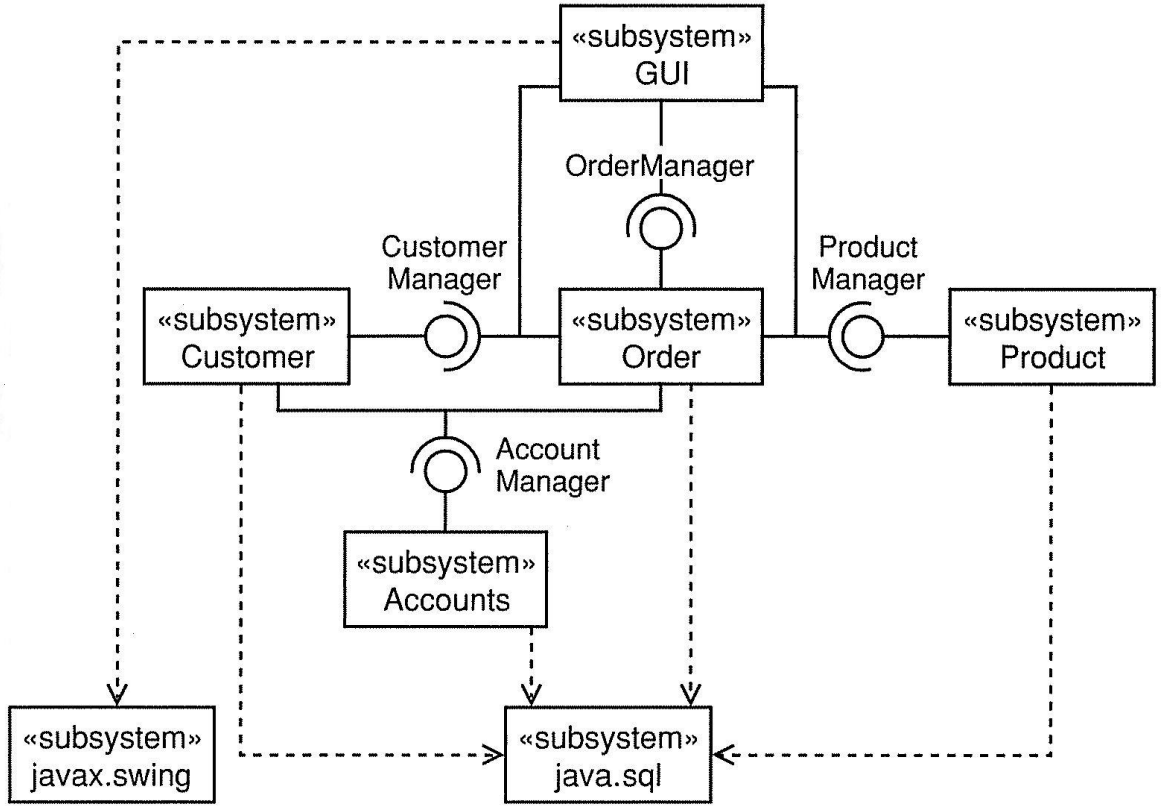
Component diagrams show how to (hierarchically) distribute the system into separate parts and communication links between them, that all define the system architecture.

The basic elements:

- **Component** - software components physically separate parts of the system that are internally coherent and externally communicate only through defined interfaces.
  - Can be *physical* (e.g. EJB) or *logical* (e.g. subsystem)
  - Can be composed of other, nested, components
- **Interfaces** - interfaces for communication between components.
  - We distinguish *required* interfaces and *provided* interfaces
- **Relations between interfaces** - connection between the *required* interface and the *provided* interface.

Component Diagram - example





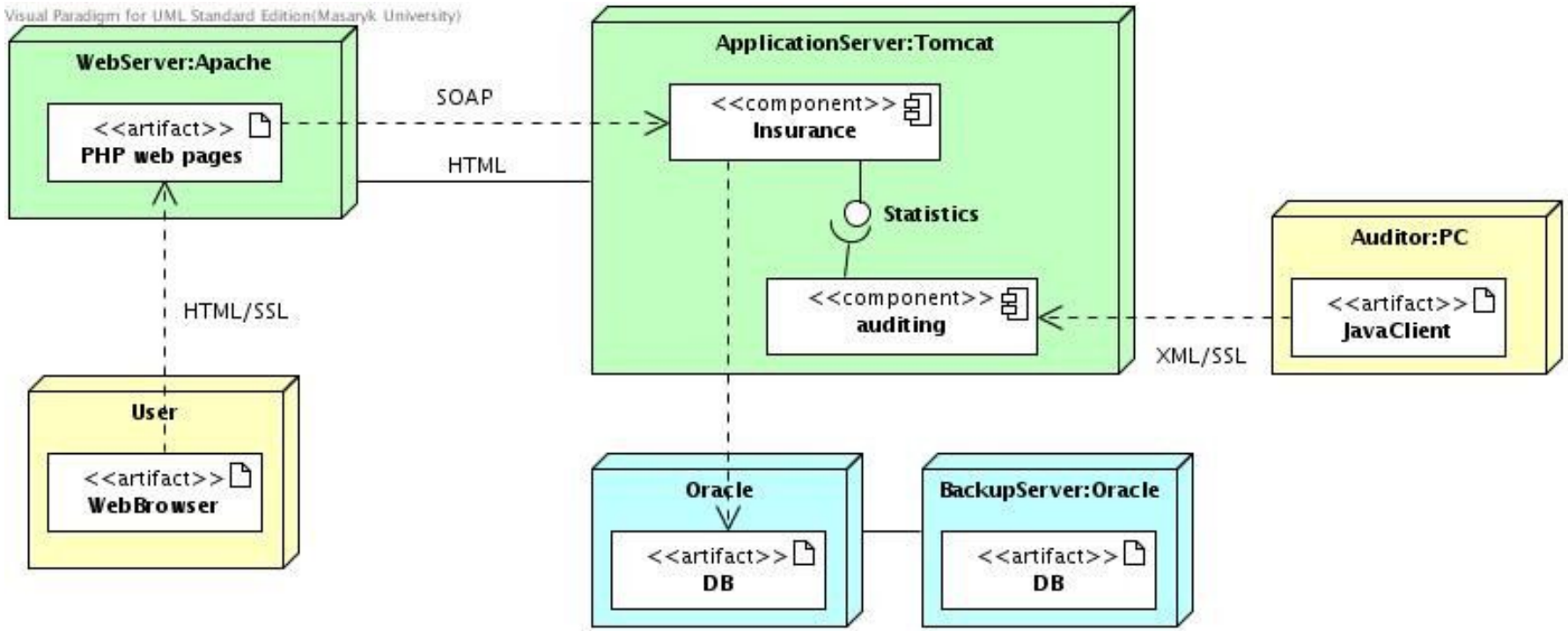
Deployment Diagram

Deployment diagrams show the way in which the software architecture will be mapped to the hardware.

Basic elements:

- **Nodes** - critical computing resources that will be placed on different parts of the system. Can be further specified with stereotypes, for example `device` or `execution environment`
- **Components/artefacts**
- **Interfaces** – interfaces for communication with components
- **Associations/dependencies** - connections between nodes (communication channels) and dependencies between components / artifacts. May contain the name of the communication protocol.

Deployment Diagram - example



Tasks

- Divide the class into packages according to the type of usage and draw dependencies between packages. Use stereotypes.
- Think about what components / subsystems will comprise your system and by means of which interfaces they will communicate.
- Create a deployment diagram of the proposed system.
- Finalize the project - remove old diagrams, check all the charts for consistency.
- Upload the **FINAL PDF report** into folder for (**Week 12**). PLEASE ensure all diagrams are included!
  Deadline: Saturday, 17.12.16 23:59

Customization of PDF Reports