Object Oriented Analysis and Design with Java

UE20CS352

Prof. Nivedita Kasturi
Department of Computer Science and Engineering

Acknowledgements: Significant portions of the information in the slide sets presented through the course in the class, are extracted from the prescribed text books, information from the Internet. Since these are only intended for presentation for teaching within PESU, there was no explicit permission solicited. We would like to sincerely thank and acknowledge that the credit/rights remain with the original authors/creators only

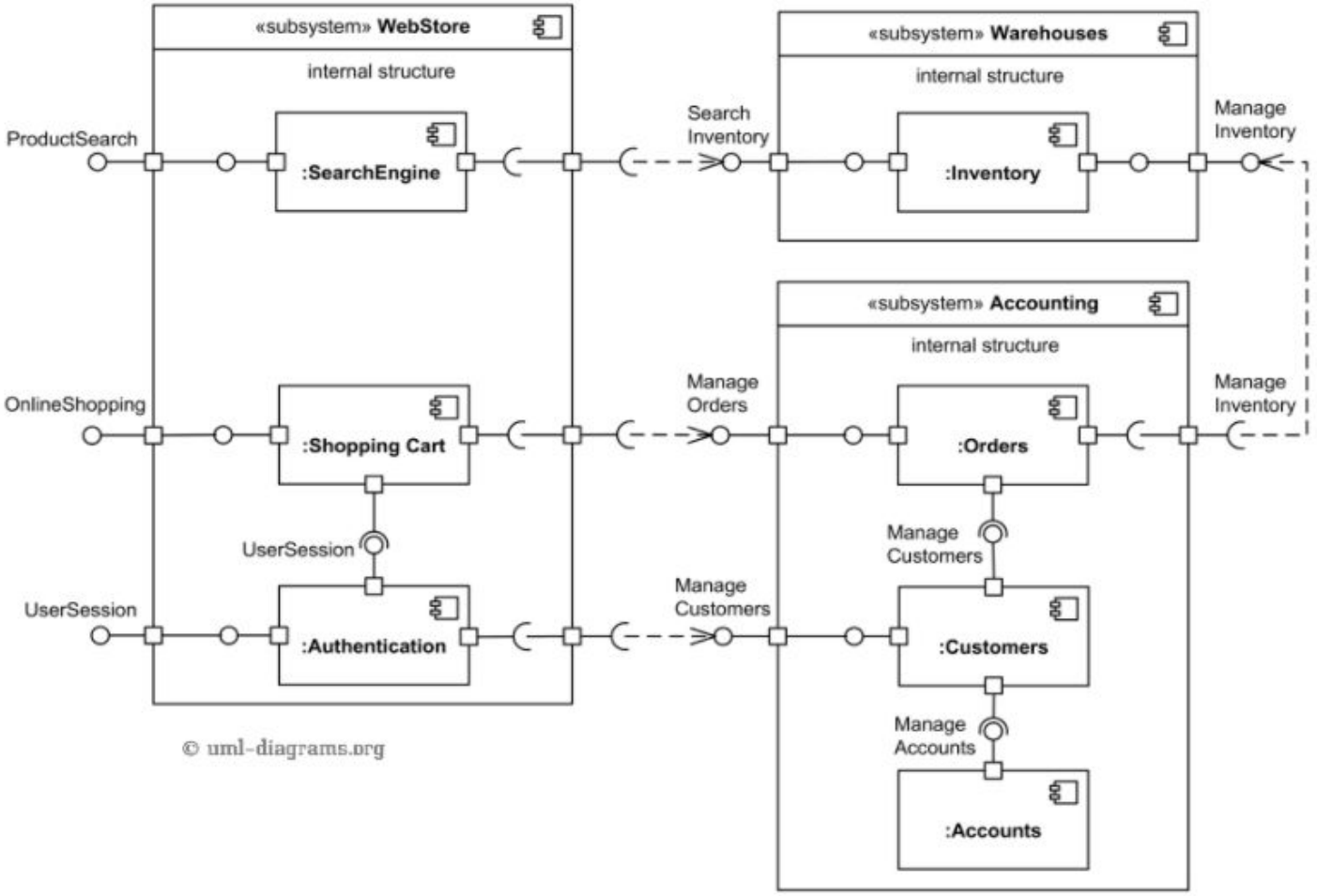
PES UNIVERSITY ONLINE

UE20CS352: Object Oriented Analysis and Design with Java

Component Model

Prof. Nivedita Kasturi
Department of Computer Science and Engineering

UML Component Diagram



Online shopping UML component diagram example with three related subsystems - WebStore, Warehouses, and Accounting.

- Used for modelling large systems into smaller subsystems which can be easily managed
- Used to represent different components of a system.
- When modelling large OO-systems, break down the system into manageable subsystems.

What is Component in OOAD?

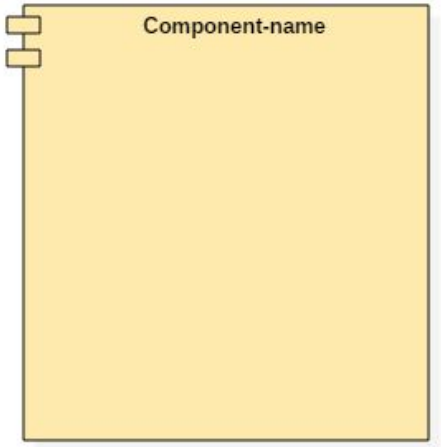
- Component is a replaceable and executable piece of a system whose implementation details are hidden.
- Component provides the set of interfaces that a component realizes or implements.
- Components require interfaces to carry out a function.
- It is a modular part of a system that encapsulates its contents.
- They are the logical elements of a system that plays an essential role during the execution of a system.
- A component is similar to a black box whose external behaviour is defined by a provided interface and required interfaces.

Structure of a UML Component

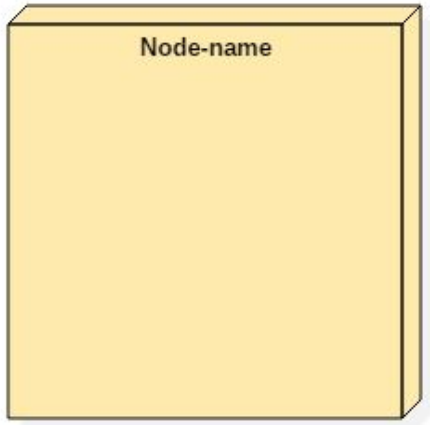
- A component is represented with classifier rectangle stereotypes as: << component >>:
- Component details are hidden for the outside world.
- The name of a component is placed at the center of a rectangle.
- A component icon is displayed at the upper right corner of a rectangle, which is optional.

Component Diagram Notations

Component Notation in Component Diagram

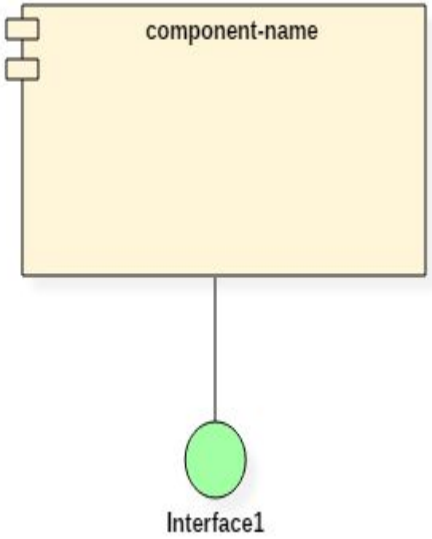


Node Notation in Component Diagram



Interface in Component Diagram

- The Interface is a named set of public features.
- It separates the specification of functionality from its implementation by a class diagram or a subsystem.
- An interface symbol cannot be instantiated.
- It declares a contract that may be realized by zero or more classifiers such as a class or a subsystem.
- Anything that realizes an interface accepts the functionalities of the interface and agrees to abide by the contract defined by the interface.
- If the implementation language does not support interfaces the use abstract classes, interfaces are named just like classes, in **UpperCamelCase**.



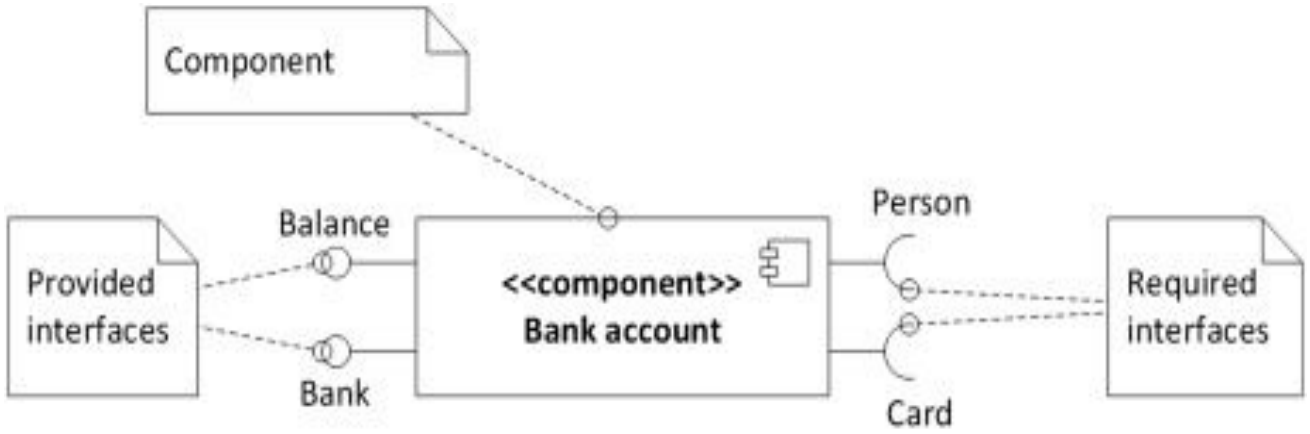
Interface in Component Diagram

The two types of Interfaces in Component Diagram are **Provided interfaces and Required interfaces**

We can connect provided and required interfaces using assembly connector.

Advantages: It increases the flexibility and extensibility of a class and it decreases the implementation dependencies.

Disadvantages: Extra flexibility leads to complex classes and too many interfaces make systems hard to understand.

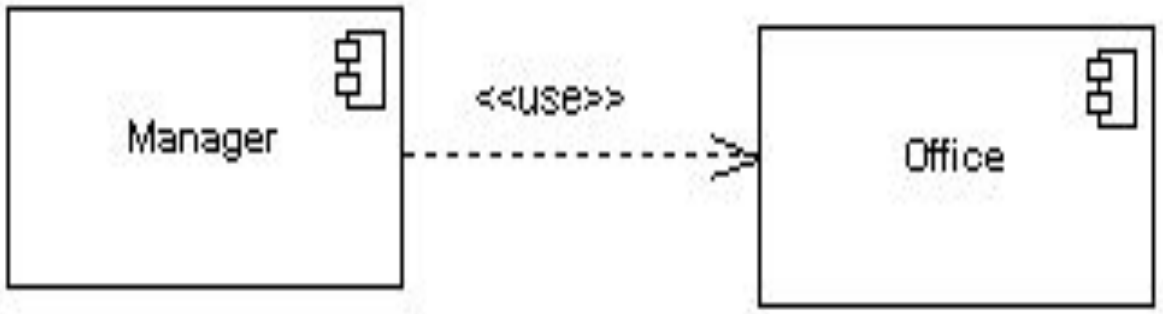


Component Diagram – Usage Connector

Components can be connected by usage dependencies.

Usage Dependency

- A usage dependency is relationship which one element requires another element for its full implementation
- It is a dependency in which the client requires the presence of the supplier
- It is shown as a dashed arrow with a <<use>> keyword
- The arrowhead point from the dependent component to the one of which it is dependent on



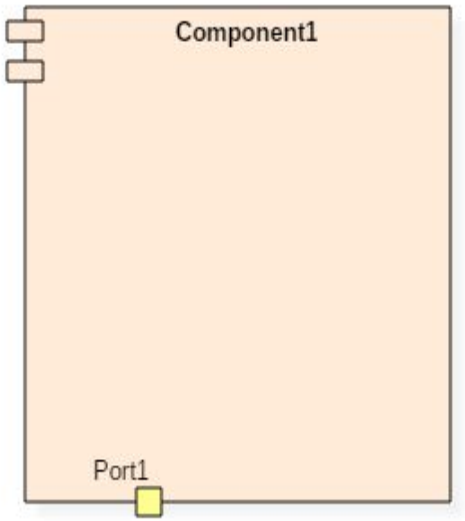
Component Diagram Subsystems

- It is a component base that acts as a decomposition unit for larger systems.
- It is a logical construct which is used to break down an extensive system into smaller systems which are known as subsystems.
- This process makes it easy to manage each subsystem efficiently.
- A subsystem cannot be instantiated during runtime, but their contents can be initialized. When subsystems are connected, it creates a single system.

Port in Component Diagram

- A Port is an interaction point between a classifier and an external environment.
- It groups semantically cohesive set of provided and required interfaces.
- A port can be used in UML without specifying the name of the port.
- A port may have visibility. When a port is drawn over the boundary of a classifier, then it means that the port is public. It also means that all the interfaces used are made as public.
- When a port is drawn inside the classifier, then it is either protected or private.
- A port also has multiplicity that indicates the number of instances of the port classifier will have.

A port in UML diagram is denoted as given below:



Here, the port1 is drawn over the boundary, which means it has visibility as public.

How to Draw Component Diagram

A component is nothing but an executable piece of a system. Various components together make a single system. Component diagrams are useful during the implementation phase of any system.

Step 1) Before modeling the component diagram, one must know all the components within the system. The working of each component should be mentioned. Component diagrams are used to analyze the execution of a system.

Step 2) One should also explore each component in depth to understand the connection of a component to other physical artifacts in the system.

Step 3) Identifying the relationship amongst various artifacts, libraries, and files are the essential things required during modeling of a component diagram.

Why use Component Diagram?

- UML component diagrams have significant importance.
- Component diagram variously differs from other diagrams. While other diagrams are used to represent the system, working of a system or the architecture of a system.
- Component diagrams are used to describe the working and behaviour of various components of a system.
- It represents how each component acts during the execution of a system.

Why use Component Diagram?

- These are the static diagrams of the unified modelling language.
- A component diagram is used to represent the structure and organization of components during any instance of time.
- Component diagrams are used for modelling the subsystems.
- These subsystems collectively represent the entire working view of any system.
- A single component cannot visualize the whole system, but the collection of multiple components can.

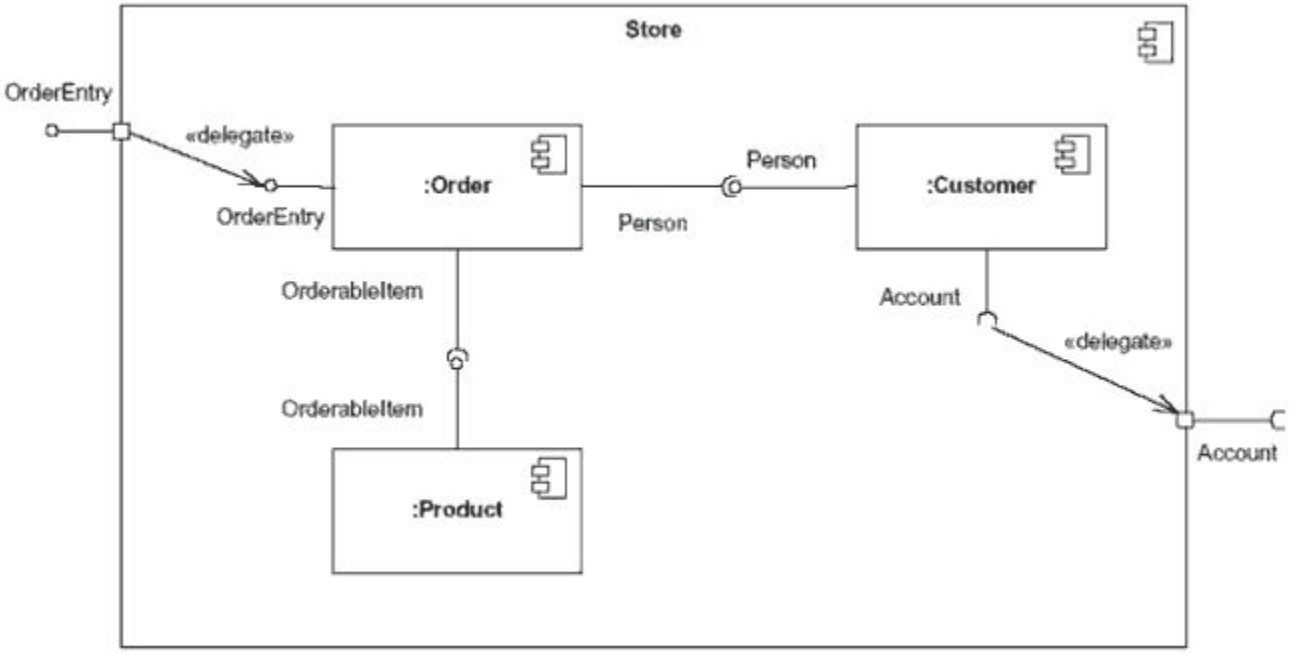
Why use Component Diagram?

So, Component Diagrams are used for:

- To represent the components of any system at runtime.
- It helps during testing of a system.
- It visualizes the connection between various components.

Internal and External view of a Component

Internal View

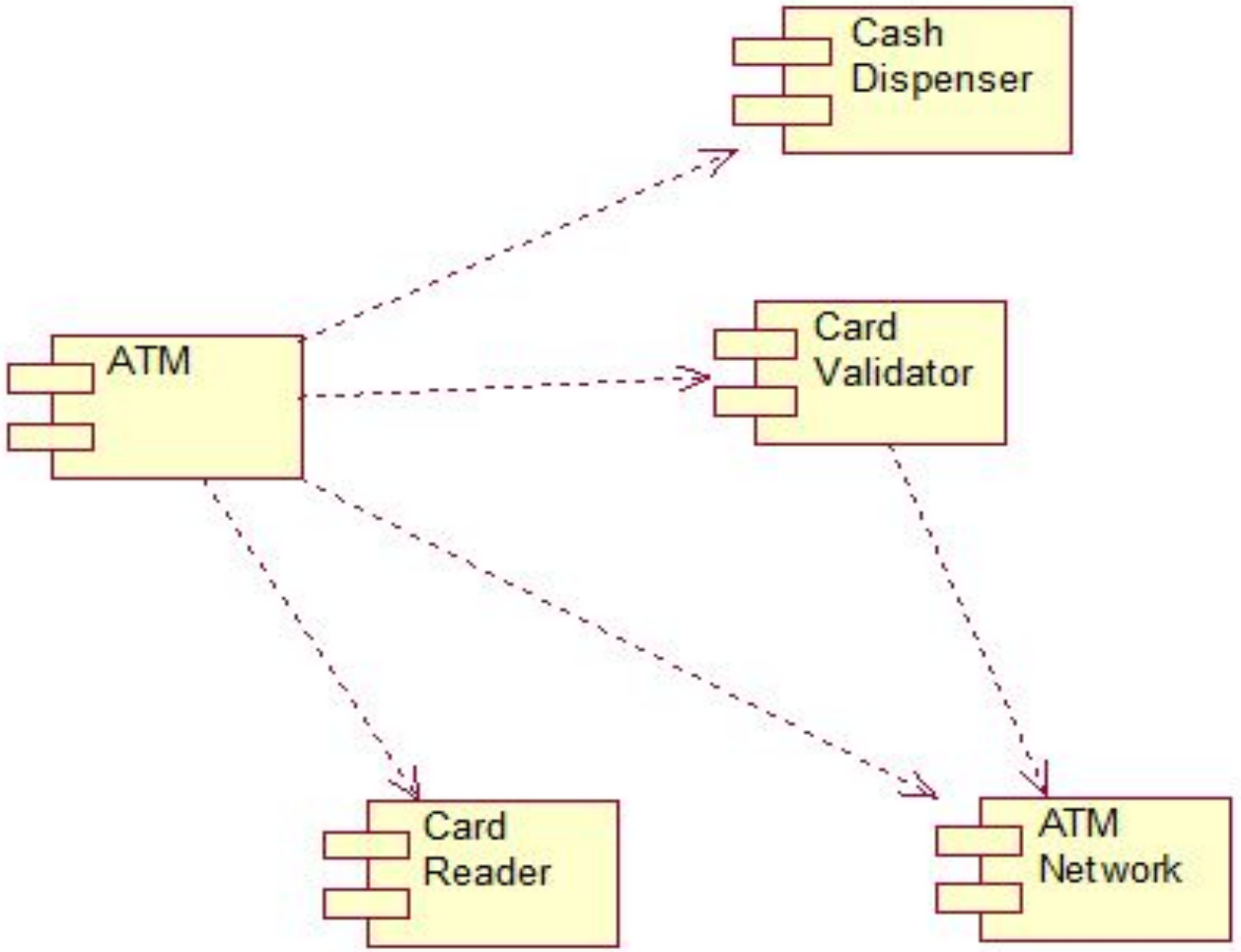


An internal view (or white box view) of a component is where the realizing classes/components are nested within the component shape

External View

An external view (or black box view) shows publicly visible properties and operations

Example – Component Diagram of an ATM



Summary

- A component is a replaceable and executable piece of a system.
- A component provides the set of required interfaces that a component realizes or implements.
- These are the static diagrams of the unified modeling language.
- It is a modular part of a system that encapsulates its contents.
- Component diagrams are used to represent the working and behavior of various components of a system.
- Various components together make a single system.

Reference

- https://www.guru99.com/component-diagram-uml-example.html

THANK YOU

Prof. Nivedita Kasturi

Department of Computer Science and Engineering

niveditak@pes.edu