NATIONAL UNIVERSITY OF COMPUTER AND EMERGINGSCIENCES

CS3005– Software Design & Architecture Lab

LAB Instructors: Sobia Iftikhar "Sobia.iftikhar@nu.edu.pk"

Lab 09

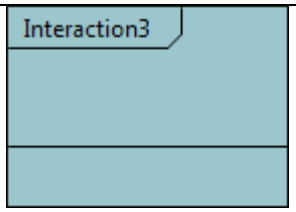
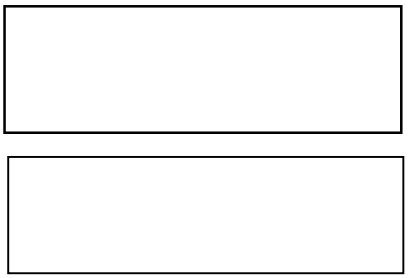
Objectives:

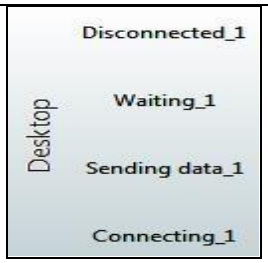
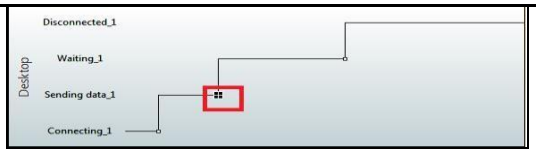
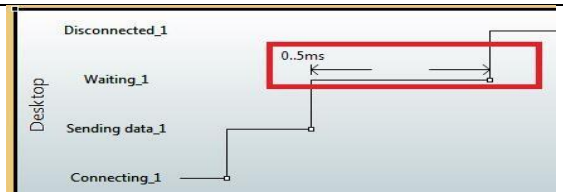
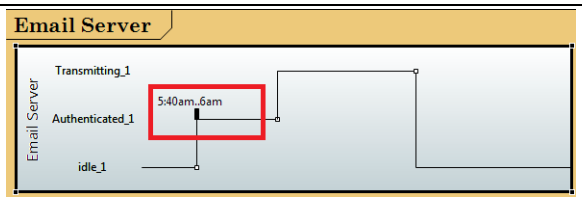
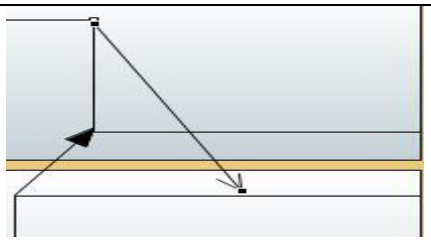
- **Timing Diagram**
- **Component Diagram**
- **Exercise**

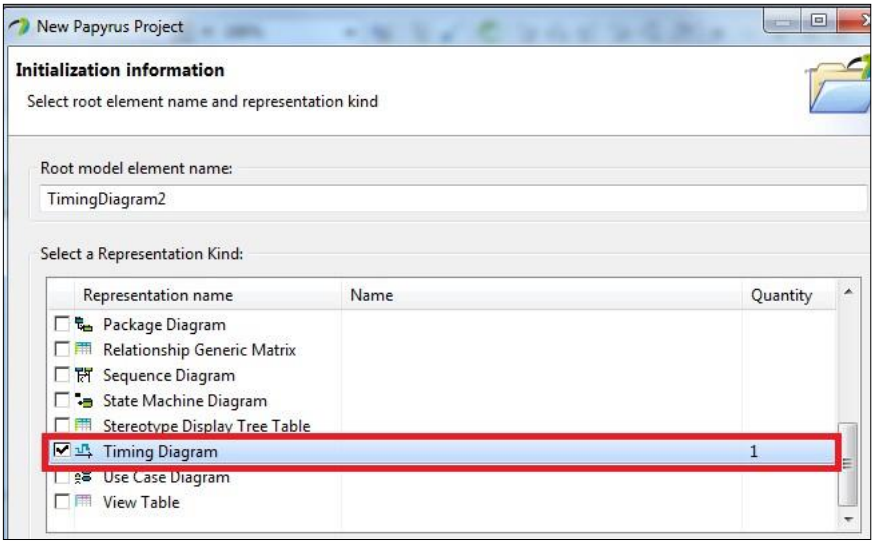
Timing Diagram

Timing diagrams are a type of interaction diagram that emphasize detailed timing specifications for messages. They are often used to model real-time systems such as satellite communication or hardware handshaking. They have specific notation to indicate how long a system has to process or respond to messages, and how external interruptions are factored into execution.

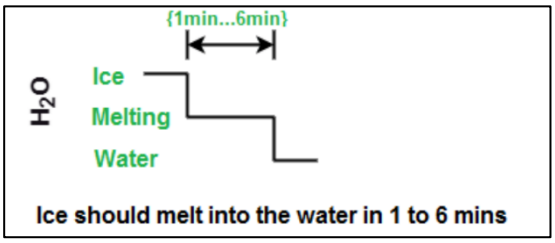
Unlike sequence diagrams, timing diagrams are read left to right rather than top to bottom.

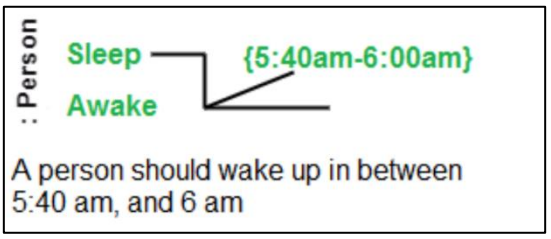
Node type	Notation	Definition
Interaction		The notation shows a rectangular frame around the diagram with a name in a compartment in the upper left corner
Lifeline (full shape/ compact shape)		There are two types of lifelines to show the lifeline of the components

State		States are added for every component
Event		Events are added to set the timing of components
Duration Constraint		The duration constraint is a constraint of an interval, which refers to duration interval. It is used to determine if the constraint is satisfied for a duration or not
Timing Constraint		It is an interval constraint, which refers to the time interval. Since it is a time expression, it depicts if the constraint is satisfied or not.
Synchronous Message/ Reply Message		Messages show interaction among diff lifelines. Message is represented as filled arrow and reply as unfilled arrow

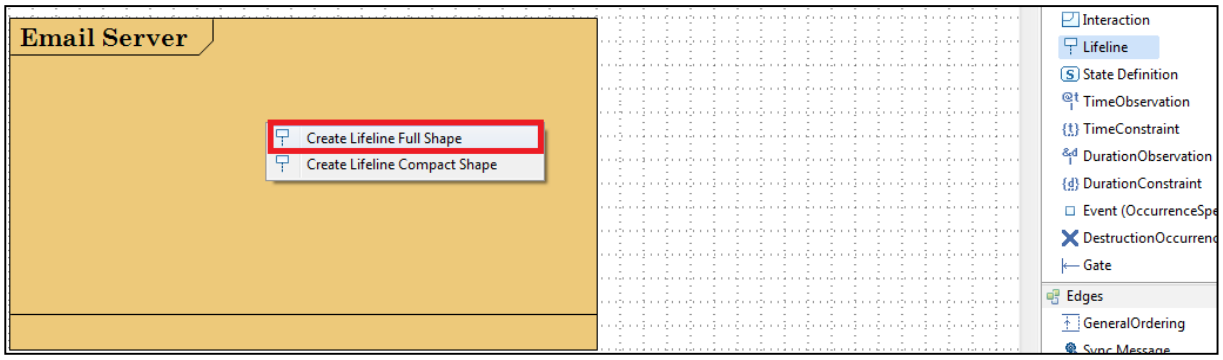


Example



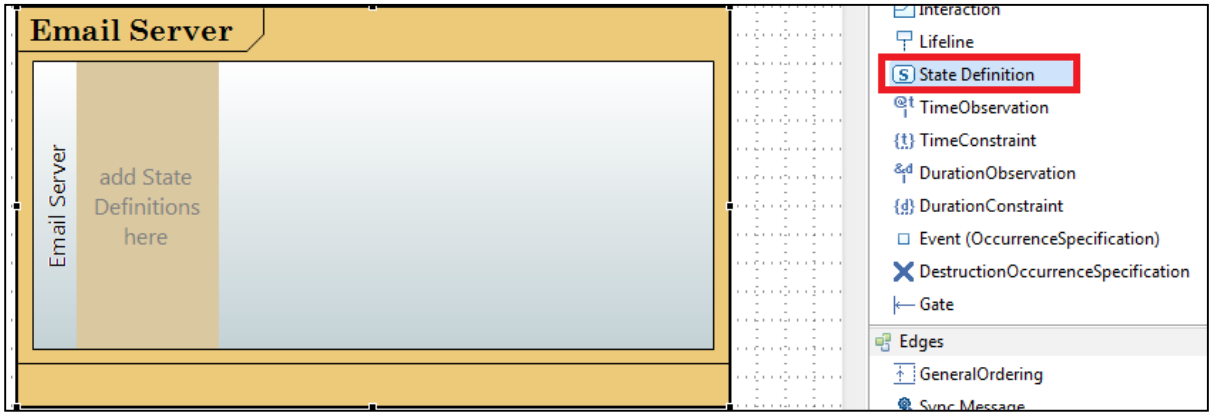


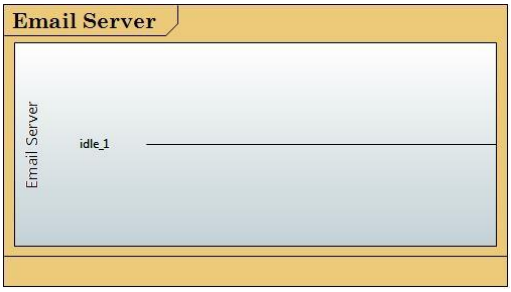
Create Lifeline Full shape

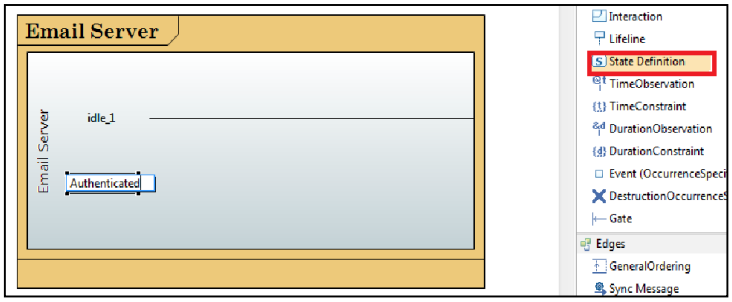


Add lifeline name

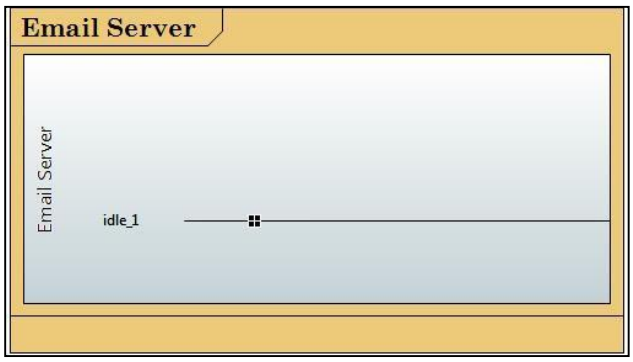
Add states in lifeline

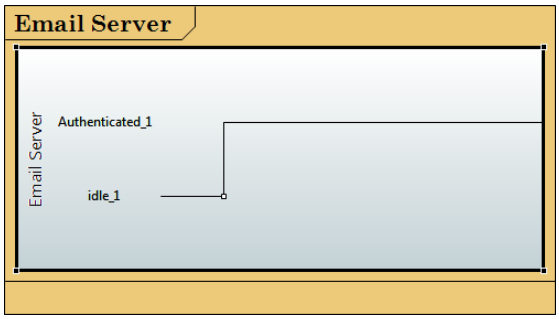




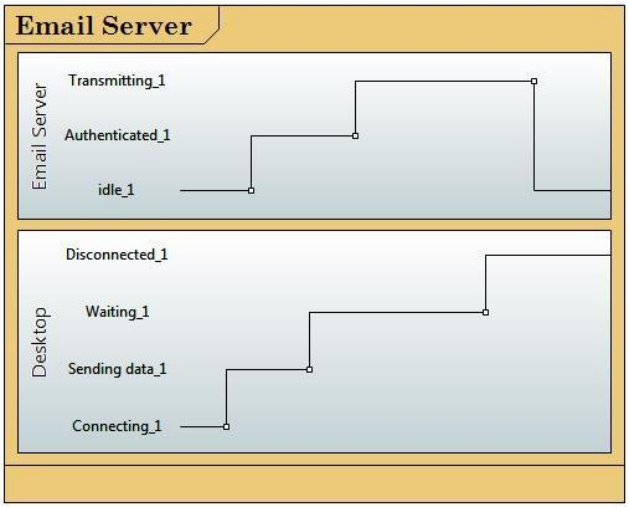


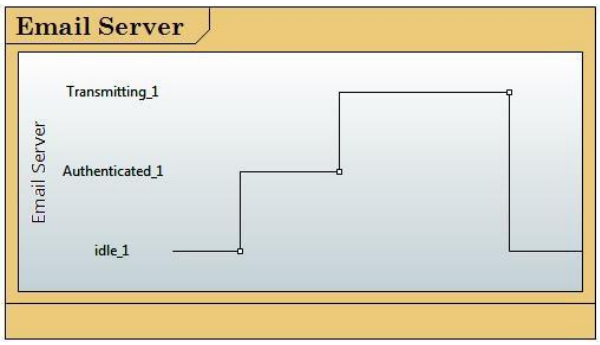
Add Events



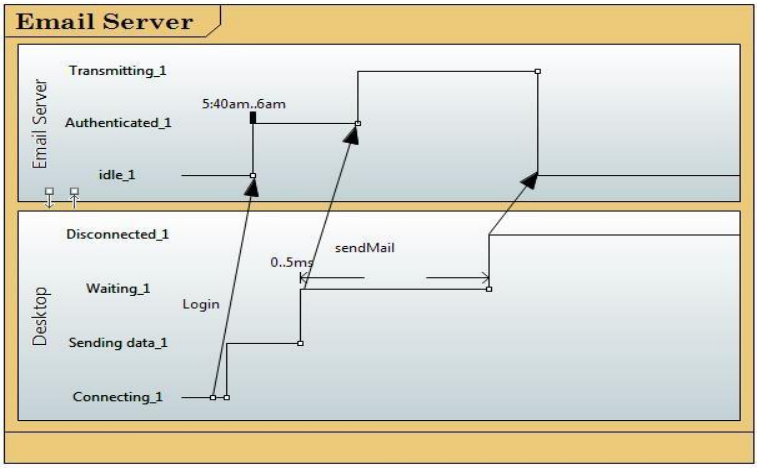


In the same way, add another lifeline Desktop

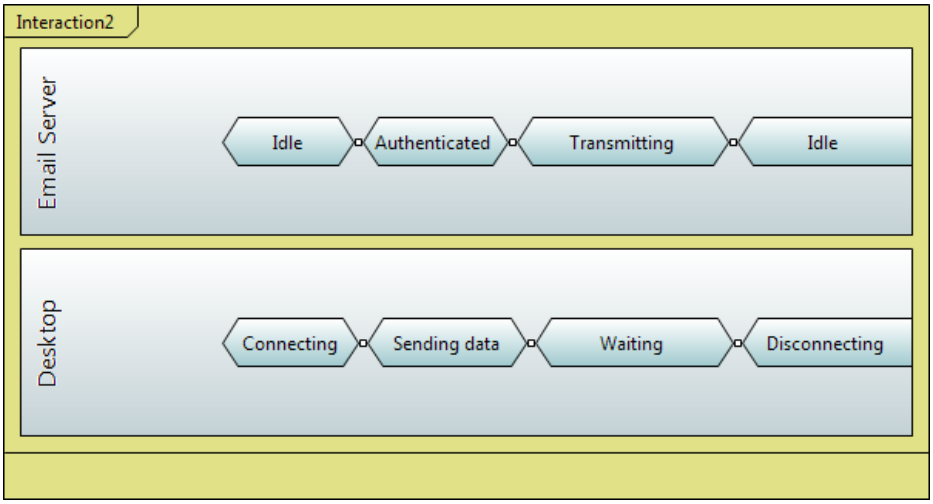




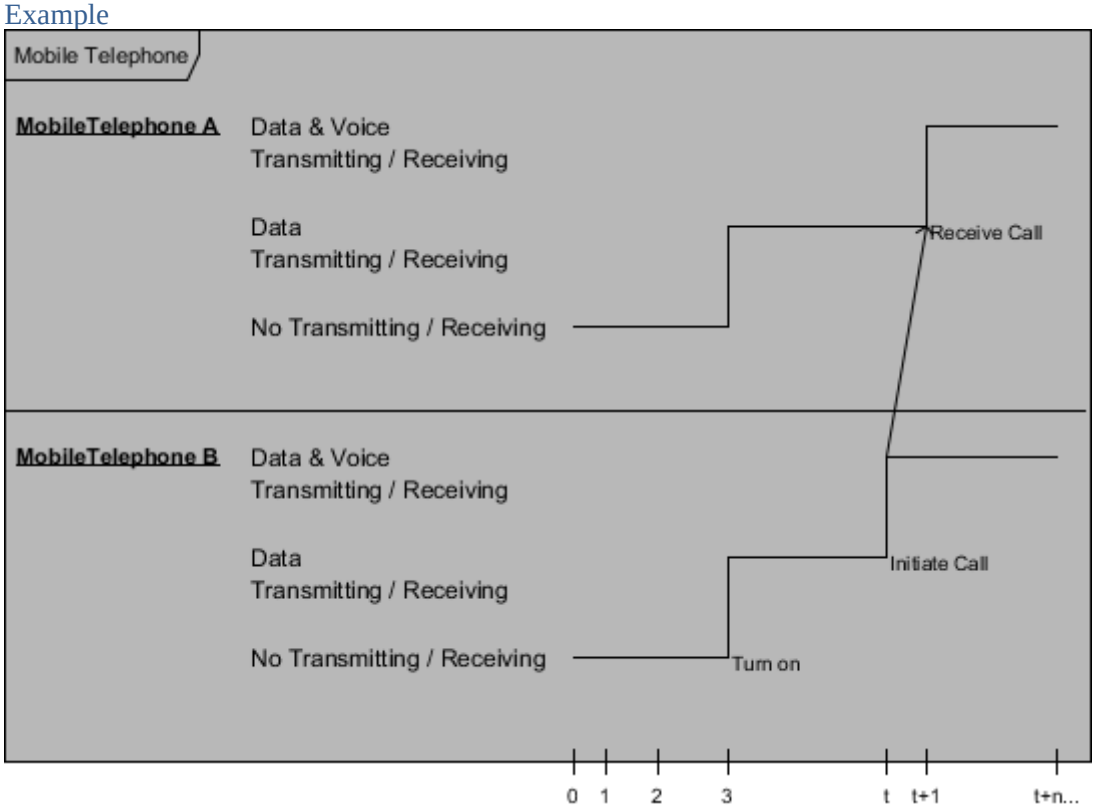
Add duration and timing constraints with messages between lifelines



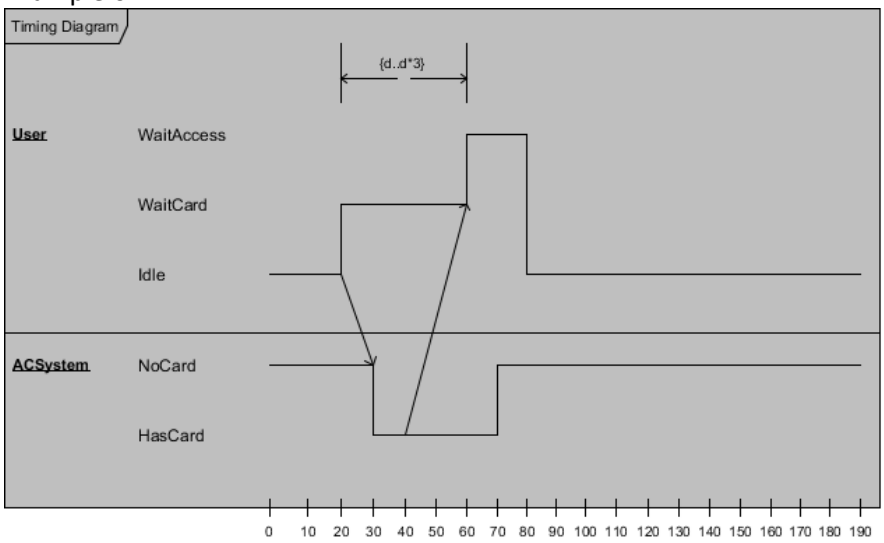
Another to create timing diagram is with compact lifeline shape. It is simpler than full lifeline shape

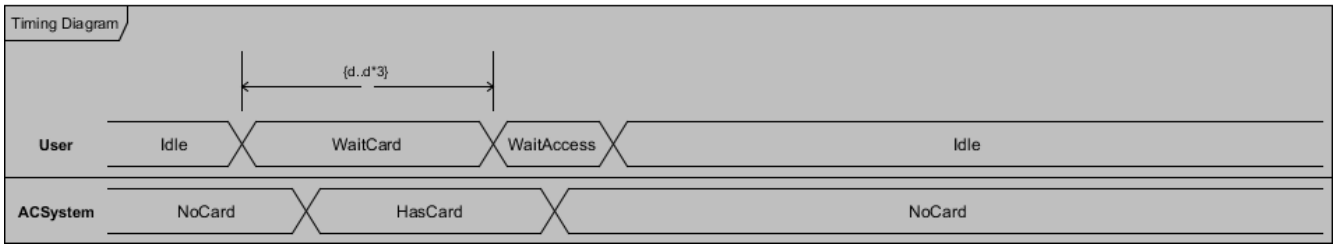


Example



Example 02





Component Diagram

While other UML diagrams, which describe the functionality of a system, component diagrams are used to model the components that help make those functionalities.

Component diagrams are used to visualize the organization of system components and the dependency relationships between them. They provide a high-level view of the components within a system.
The components can be a software component such as a database or user interface; or a hardware component such as a circuit, microchip or device; or a business unit such as supplier, payroll or shipping.

Component diagrams

- Are used in Component-Based-Development to describe systems with Service-Oriented-Architecture
- Show the structure of the code itself
- Can be used to focus on the relationship between components while hiding specification detail
- Help communicate and explain the functions of the system being built to stakeholders

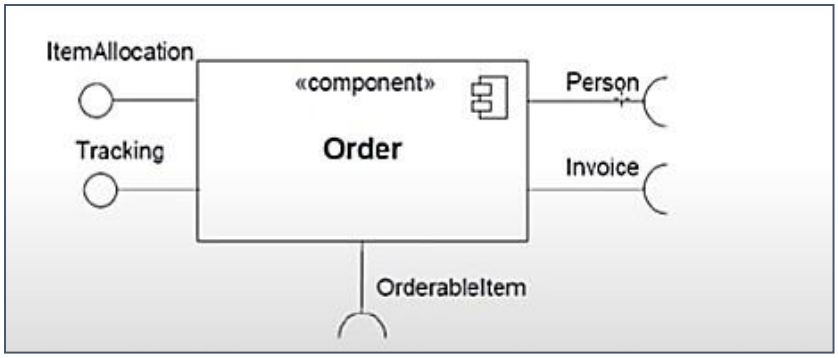
Component Diagram Symbols

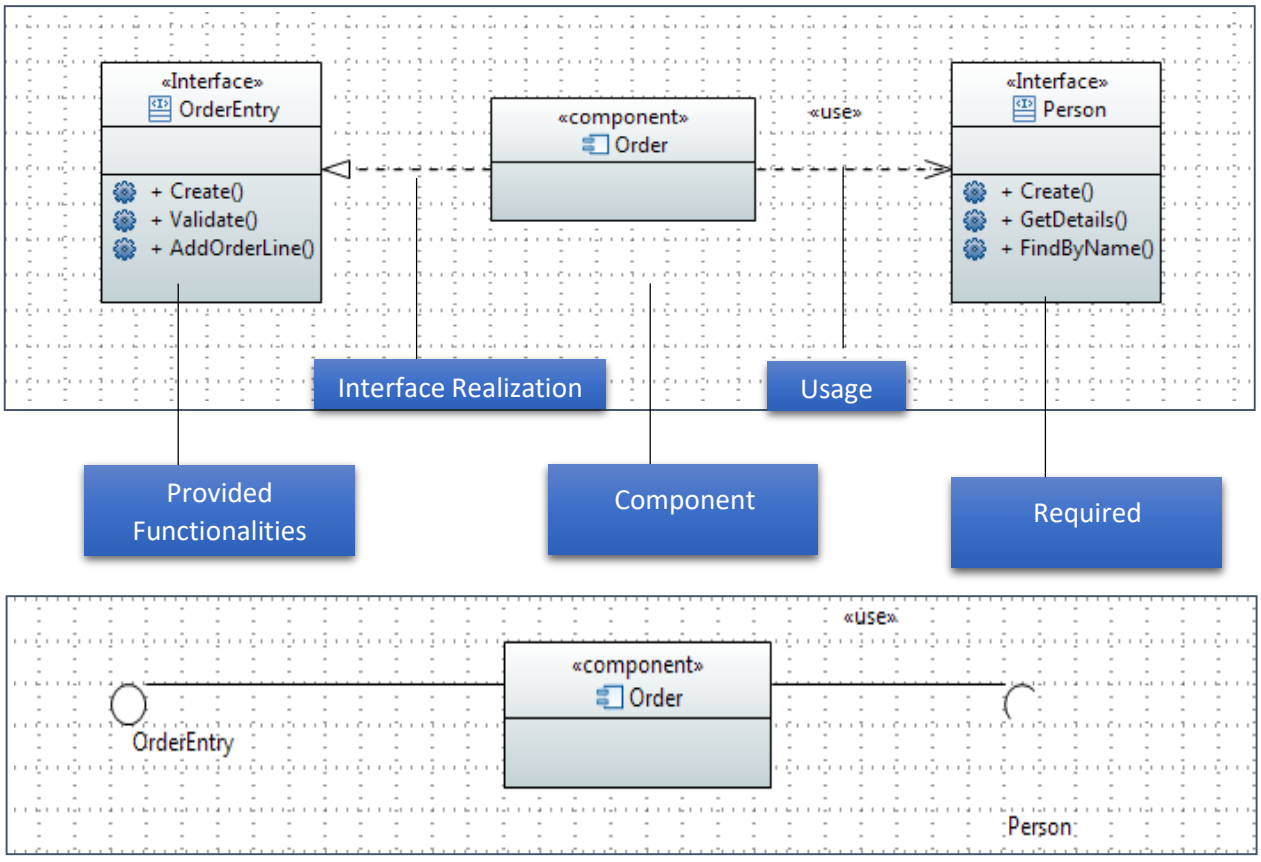
con	Description

	Packages are used to organize your project contents, but when added onto a diagramthey can be use for structural or relational depictions.
	A Component is a modular part of a system, whose behavior is defined by its provided and required interfaces.
	An Interface is a specification of behavior (or contract) that implementers agree tomeet.
	Interface that will be provided as functionality from any component
	Interface will be used by other component
	Ports define the interaction between a classifier and its environment.

Example 01

Required Interfaces and Provided Functionalities Interfaces

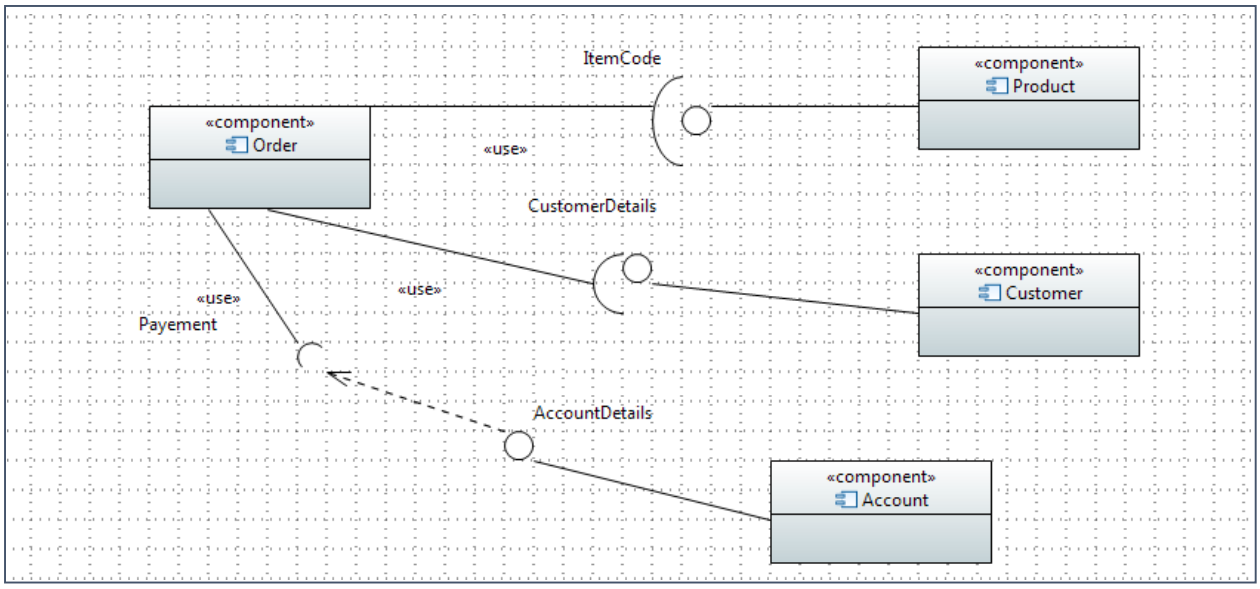




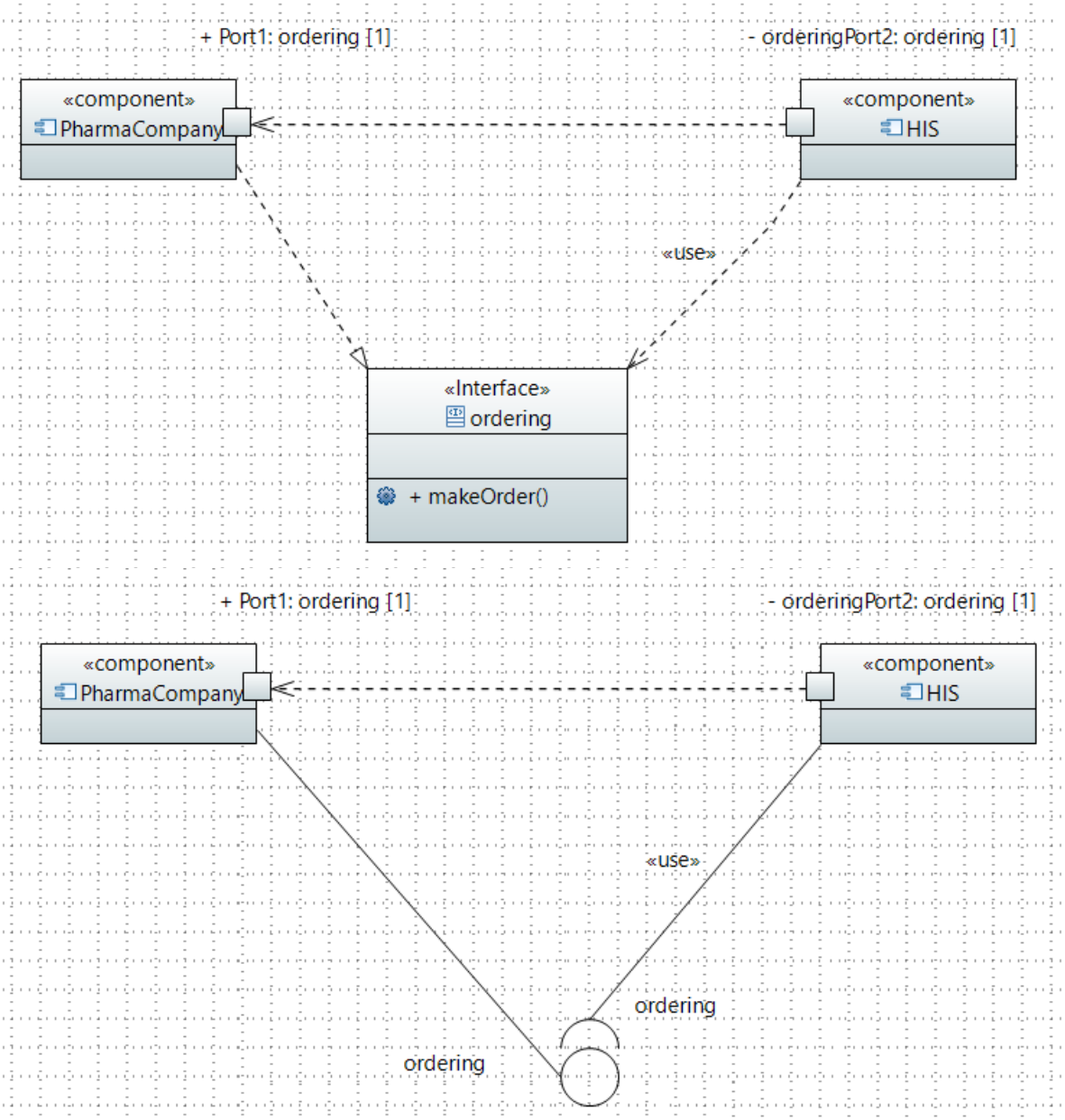
Example 02

This diagram demonstrates a number of components and their inter-relationships.

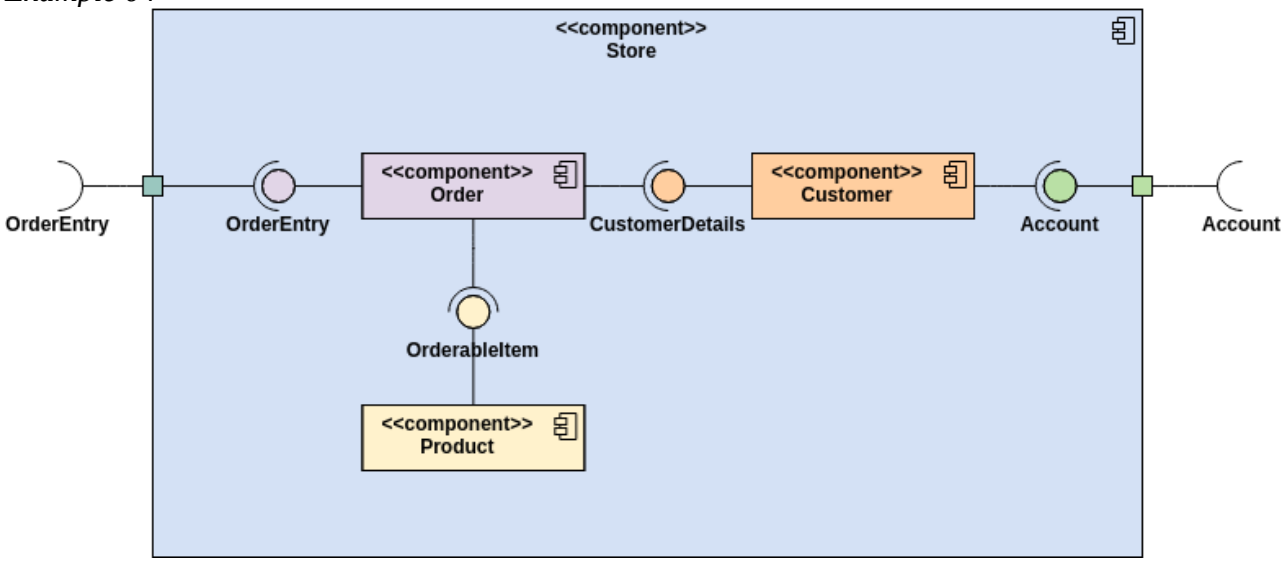
Assembly connectors connect the provided interfaces supplied by Product and Customer to the required interfaces specified by Order. A Dependency relationship maps a customer's associated account details to the required interface Payment, also specified by Order.



Example 03 Health Information System



Example 04


<<component>>
Store
OrderEntry
OrderEntry
<<component>>
Order
CustomerDetails
<<component>>
Customer
Account
Account
OrderableItem
<<component>>
Product