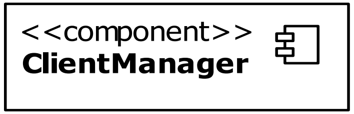
SE 211
Software Specification and Design II

(Additional) UML Diagrams

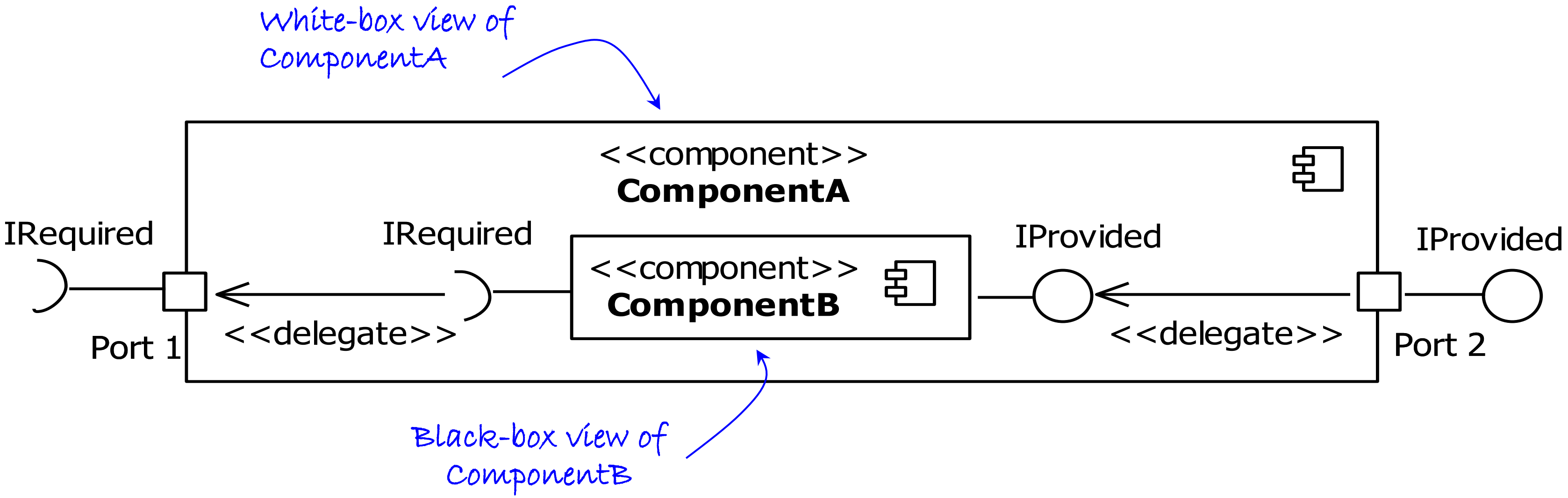
UML Component Diagrams

- A component represents a modular part of a system.
- Component diagrams are used to model software as group of components connected to each other through well-defined interfaces.
- Component diagrams help decompose systems and represent their structural architecture from a logical perspective.
- A component is represented using a box with the keyword <<component>> and optional component icon on the top right corner.



UML Component Diagrams (cont'd)

- Components can be modeled using an external *black-box* or internal *white-box* approach.
  - *Black-box* approach hides the component's internal structure.
    - Components interact with each other only through identified interfaces.
  - *White-box* approach shows the component's internal structure (e.g., realizing classifiers).

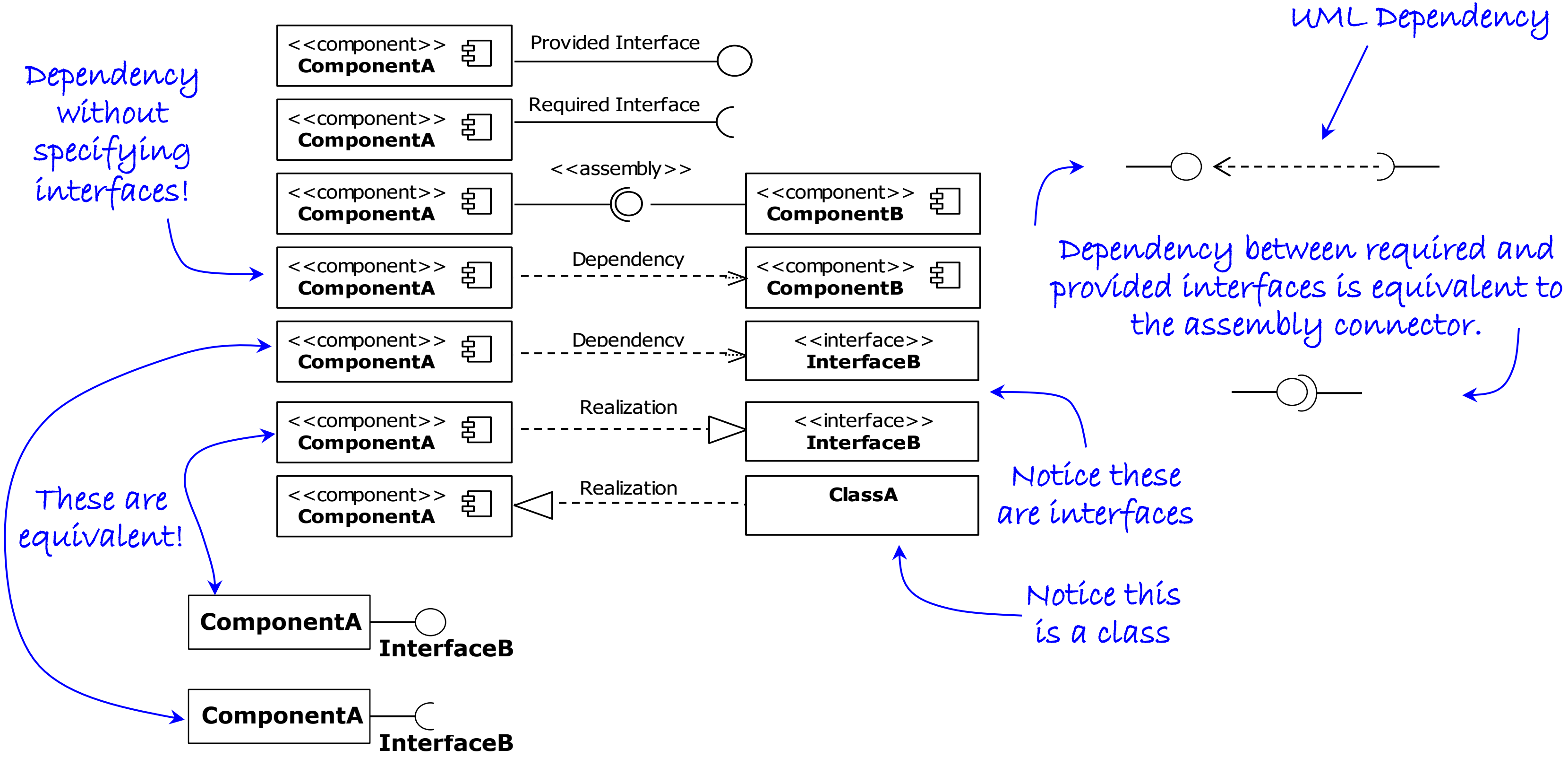


UML Component Diagrams (cont’d)

- Component interfaces are classified as provided or required interfaces.
  - Provided interfaces are used by other external components to interact with the component providing the services.
  - Required interfaces are those the components need to carry out their functions.

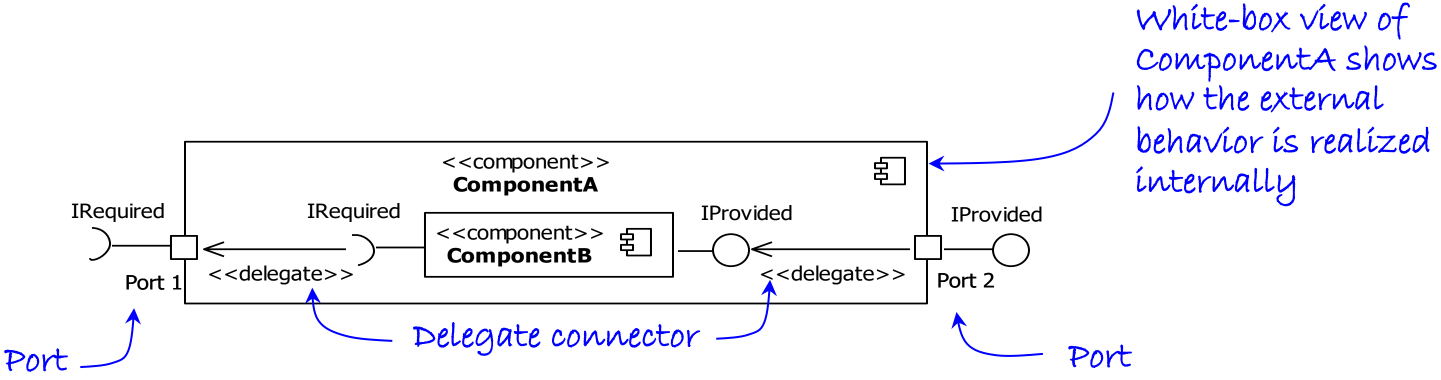
UML Component Diagram (cont'd)

- UML relationships applied to components



UML Component Diagram (cont'd)

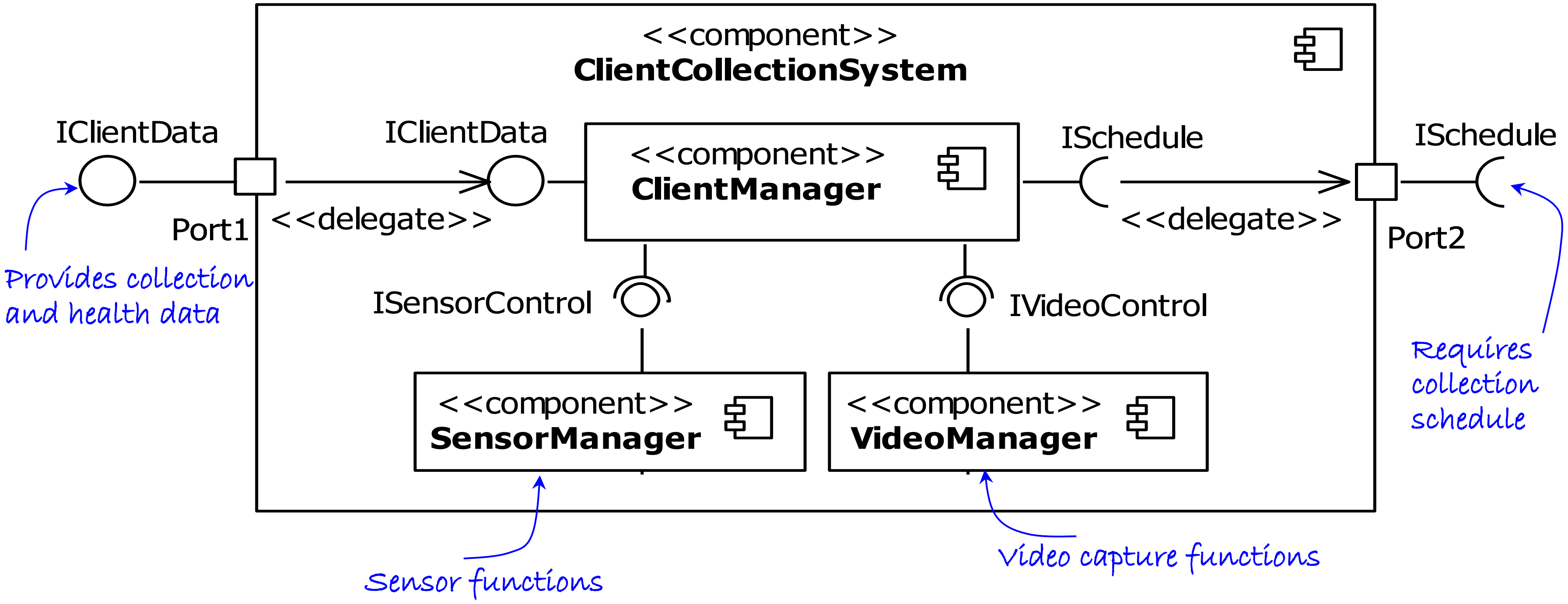
- Two more important concepts used in component diagrams are:
  - Ports
    - Abstraction used to model access points for allowing the external environment to access the component's services and for allowing components to interact with their external environment.
    - Modeled using a small square at the boundary of the component box.
    - Ports can be named, e.g., port names below are Port 1 and Port 2.
  - Delegation connectors
    - Used to model the link between the external provided interfaces of a component to the realization of those interfaces internally within the component.
    - Similarly, delegation connectors model the link between internally required interfaces to ports requiring the interface from external components.
    - Modeled using a directed arrow with the stereotype <<delegate>>



Component Diagram Example

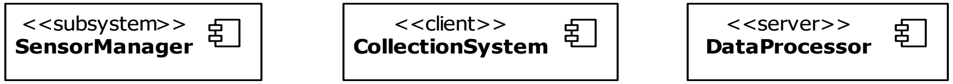
- What does a component diagram look like for a system with the following desired properties:
  - A data collection system equipped with:
    - Sensors
    - Video capture capabilities
  - Automatic collection at specific times of the day.
    - Collection schedules need to be provided to the system.
  - The system must make available the data collected.
    - Both sensor and video data.
    - Also, health data about the system
      - Events, problems, etc.

Component Diagram Example (cont'd)

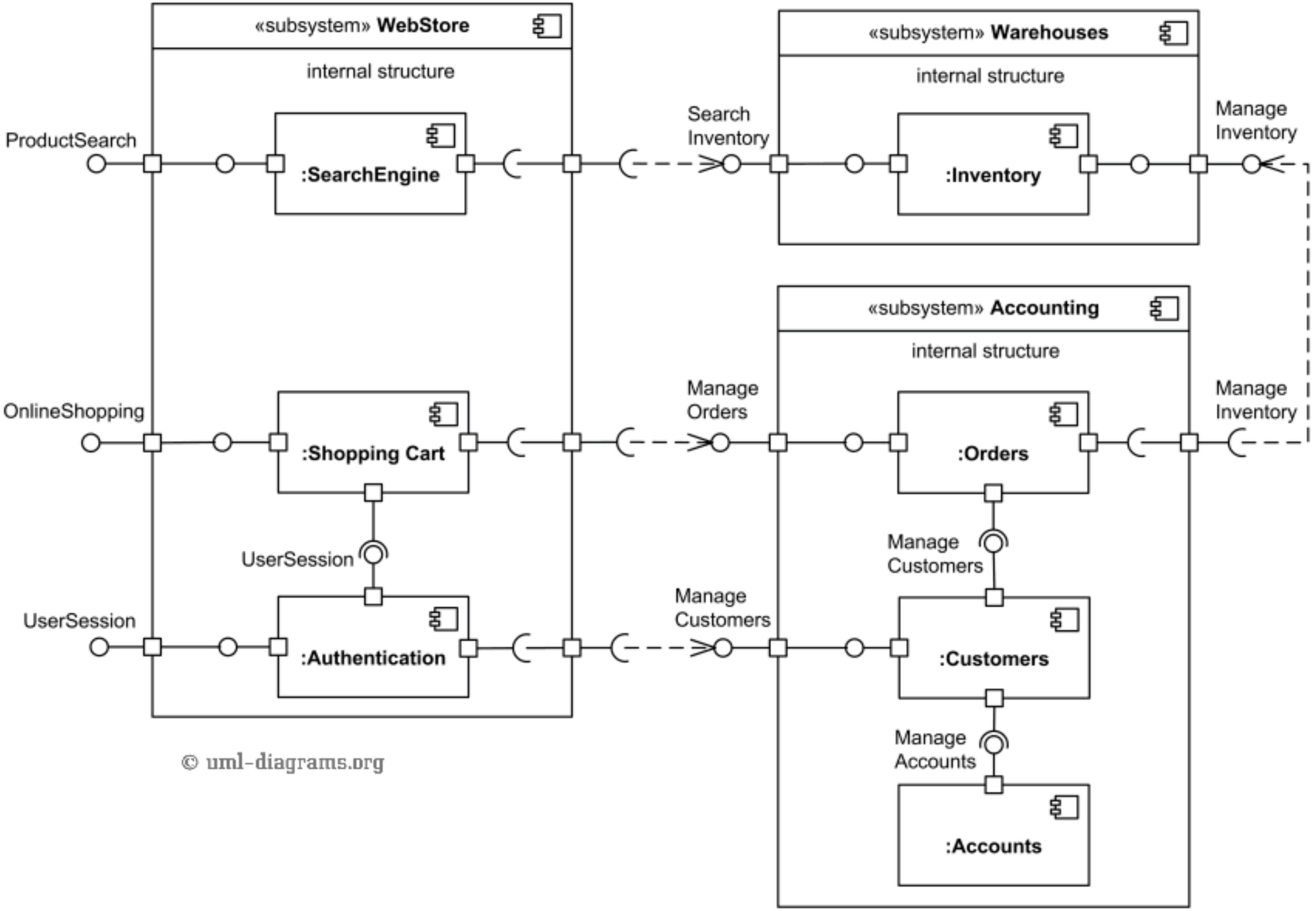


UML 2.3 Enhancing Features

- Stereotypes
  - Mechanism for extending UML by adding information that gives existing UML elements (both classifiers and relationships) a different meaning, therefore creating a semantically different element for modeling application-specific concepts.
  - Modeled as existing UML elements with the <<stereotype>> mechanism.

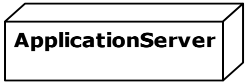


Sample Component Diagram



UML Deployment Diagrams

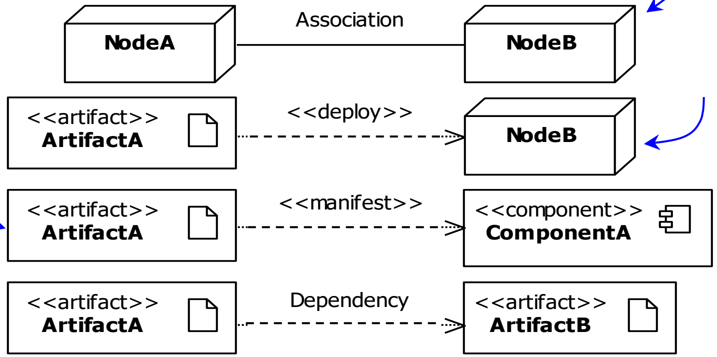
- Deployment diagrams are used to model the physical realization of software systems.
  - They provide the means to visualize and evaluate the environment in which software executes.
  - They model nodes and the interfaces between them.
    - A node is a computational resource that host software artifacts for execution.
    - A node is a represented by a named cube.



- Deployment diagrams also include artifact and components and depict how all of these work together from a system deployment perspective.
- UML relationships applied to the these classifier is presented below.

UML Deployment Diagrams (cont'd)

There is a communication path between Nodes A and B



NodeA

Association

NodeB

<<artifact>>
ArtifactA

<<deploy>>

NodeB

<<artifact>>
ArtifactA

<<manifest>>

<<component>>
ComponentA

<<artifact>>
ArtifactA

Dependency

<<artifact>>
ArtifactB

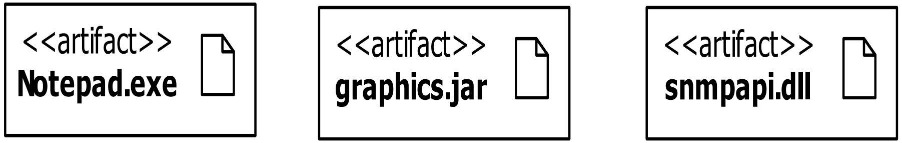
ArtifactA is deployed on NodeB

ArtifactA embodies ComponentA

ArtifactA depends on ArtifactB

UML Deployment Diagrams (cont'd)

- A UML artifact is used to model physical units of information that form part of the software system, such as binary executable files, configuration files, scripts, .jar files, .dll, etc.
- An artifact is represented using a rectangle with the keyword <<artifact>>

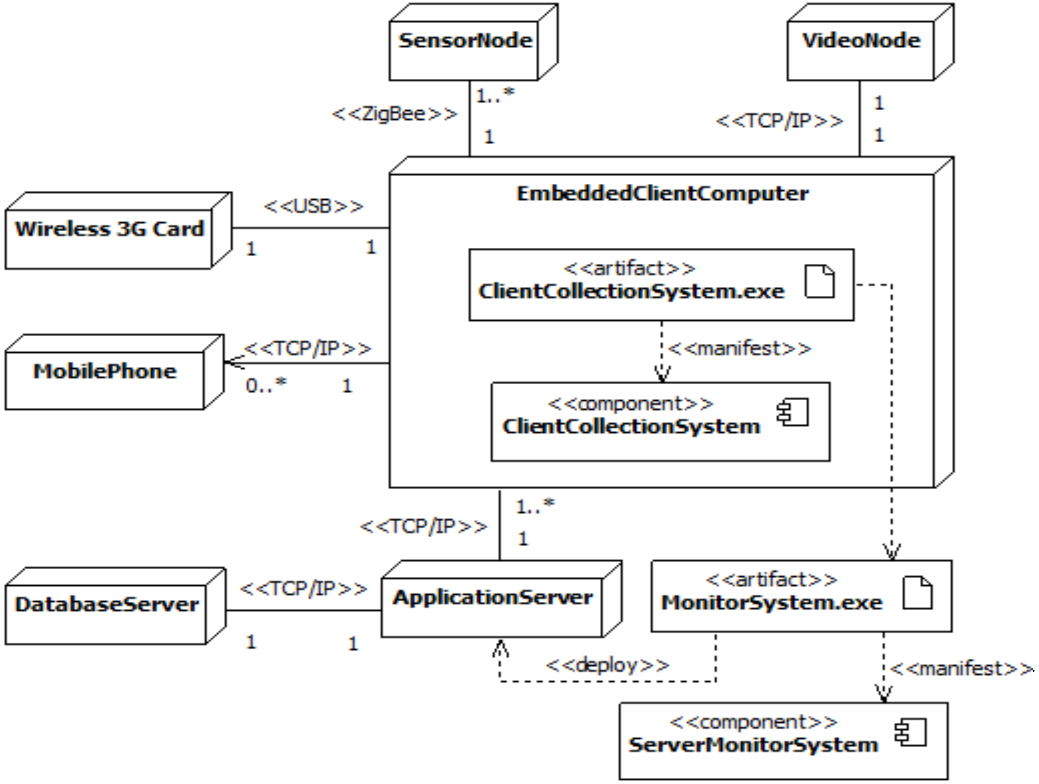


<<artifact>>
Notepad.exe

<<artifact>>
graphics.jar

<<artifact>>
snmpapi.dll

Example of UML Deployment Diagram


SensorNode
VideoNode
<<ZigBee>>
1..*
1
<<TCP/IP>>
1
1
EmbeddedClientComputer
Wireless 3G Card
<<USB>>
1
1
<<artifact>>
ClientCollectionSystem.exe
MobilePhone
<<TCP/IP>>
0..*
1
<<manifest>>
<<component>>
ClientCollectionSystem
<<TCP/IP>>
1..*
1
DatabaseServer
<<TCP/IP>>
1
1
ApplicationServer
<<artifact>>
MonitorSystem.exe
<<deploy>>
<<manifest>>
<<component>>
ServerMonitorSystem