IMPLEMENTATION DIAGRAMS

TOPIC # 15(B)

CHAPTER 37 – UML DEPLOYMENT AND COMPONENT DIAGRAMS – CRAIG LARMAN

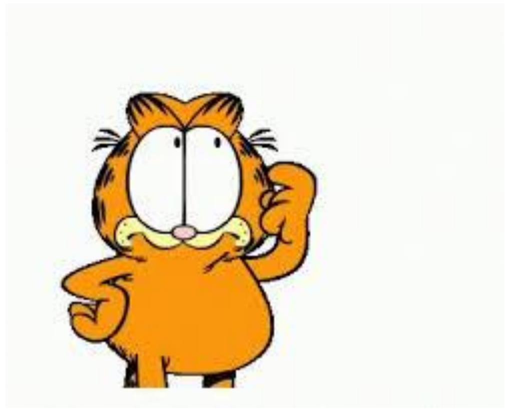
DEPLOYMENT DIAGRAM

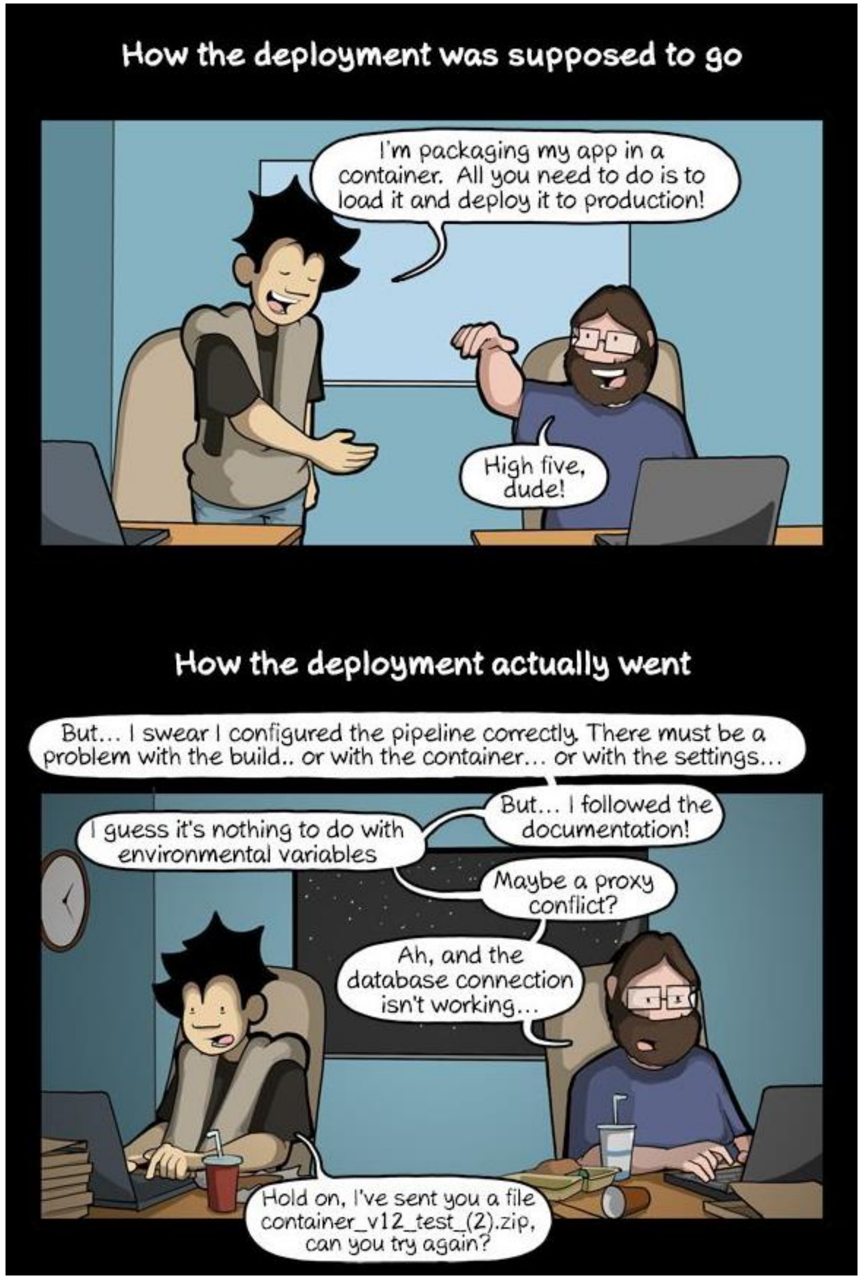
FALL 2020

Contents

- What is software deployment?
- Introduction to software deployment diagram
- Applications of deployment diagram
- Guidelines for deployment diagram
- Essentials of deployment diagram
- Real life example

What is Software Deployment?




How the deployment was supposed to go
I'm packaging my app in a container. All you need to do is to load it and deploy it to production!
High five, dude!
How the deployment actually went
But... I swear I configured the pipeline correctly. There must be a problem with the build.. or with the container... or with the settings...
But... I followed the documentation!
I guess it's nothing to do with environmental variables
Maybe a proxy conflict?
Ah, and the database connection isn't working...
Hold on, I've sent you a file container_v12_test_(2).zip, can you try again?

- Software deployment includes
  - Steps
  - Processes
  - Activities that are required to make a software system or update available to its intended users

Deployment Diagram

- It describes an aspect of the software system itself
- It is a diagrammatic representation of physical deployment information generated by the software program on hardware components
- ** The information generated by the software program is called an ‘Artifact’

Why do we need deployment diagrams?

- To show which software elements are deployed by which hardware elements.
- To illustrate the runtime processing for hardware.
- To model physical hardware elements and the communication paths between them
- To plan the architecture of a system

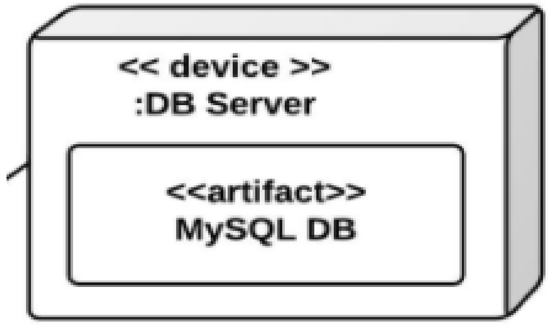
Essentials of Deployment Diagram

- A variety of UML shapes make up deployment diagrams
  - Artifact
  - Association
  - Component
  - Dependency
  - Interface
  - Node
  - Node as container
  - Package
  - Sterotypes

Artifact

- A product developed by the software, symbolized by a rectangle with the name and the word “artifact” enclosed by double arrows.

Communication Path

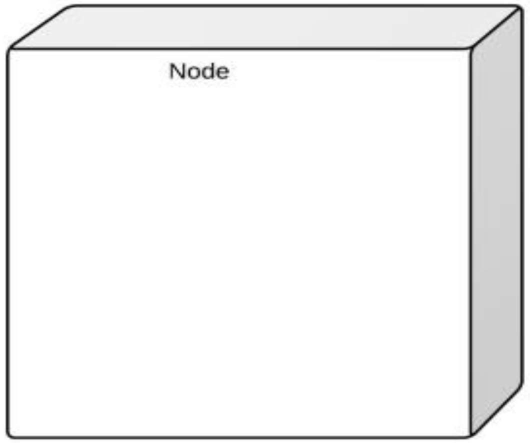


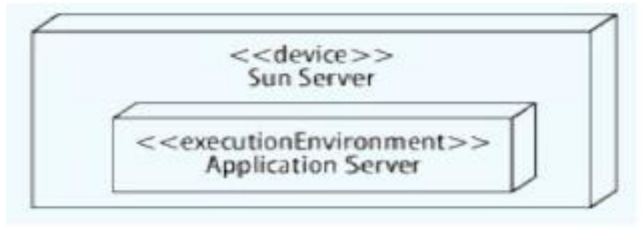
- A straight line that represents communication between two device nodes

Node

- Hardware or software object, shown by a 3D three-dimensional box
- Hardware Nodes can be
  - Server, desktop, PC etc
- Software Nodes also known as Execution node can be
  - OS
  - J2EE Container
  - Web Server
  - Application Server

<< device >>
:DB Server





Association

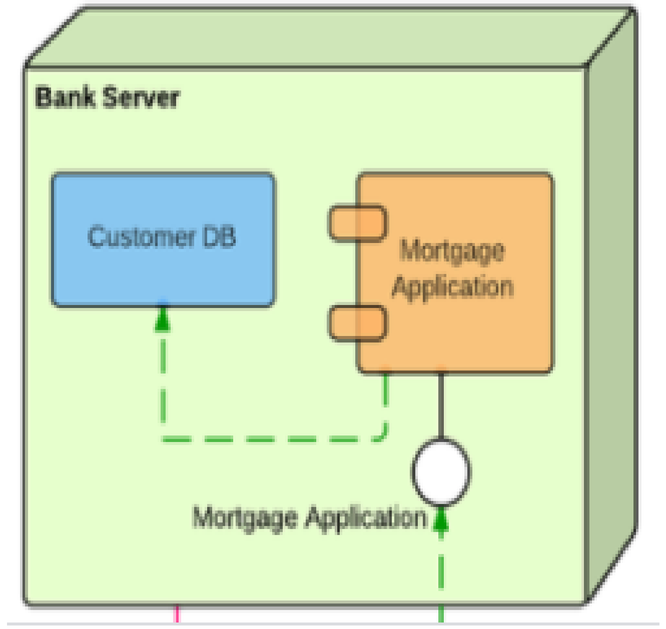
- A line that indicates a message or other type of communication between nodes

Component

- A rectangle with two tabs that indicates a software element

Dependency

- A dashed line that ends in an arrow, which indicates that one node or component is dependent on another



Interface

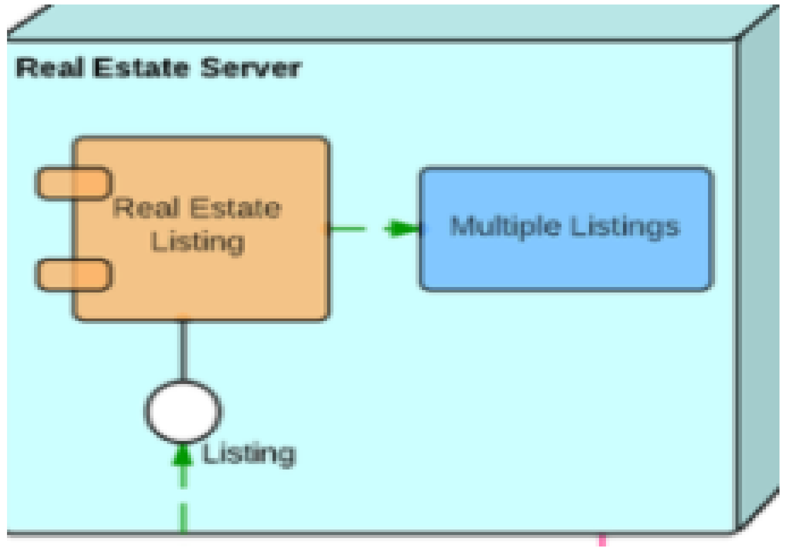
- A circle that indicates a contractual relationship.

Node as container

- A node that contains another node inside of it

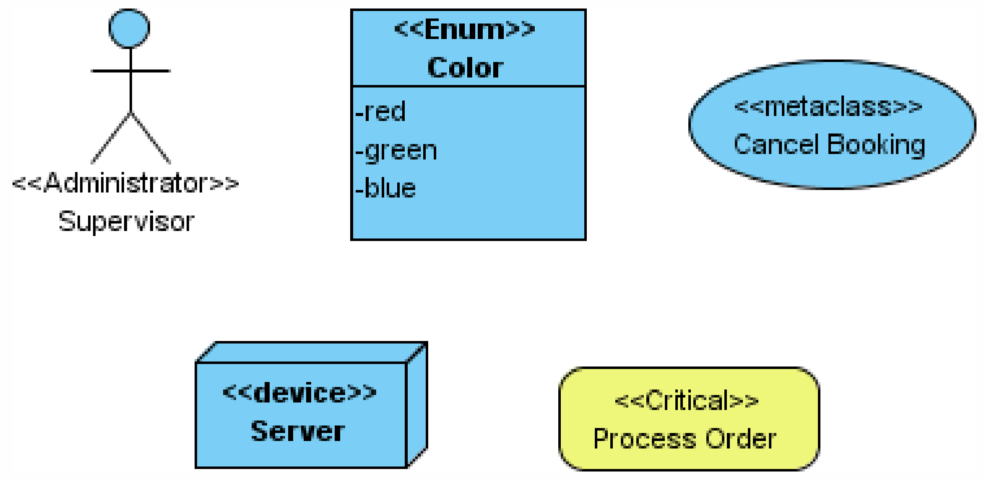
Package

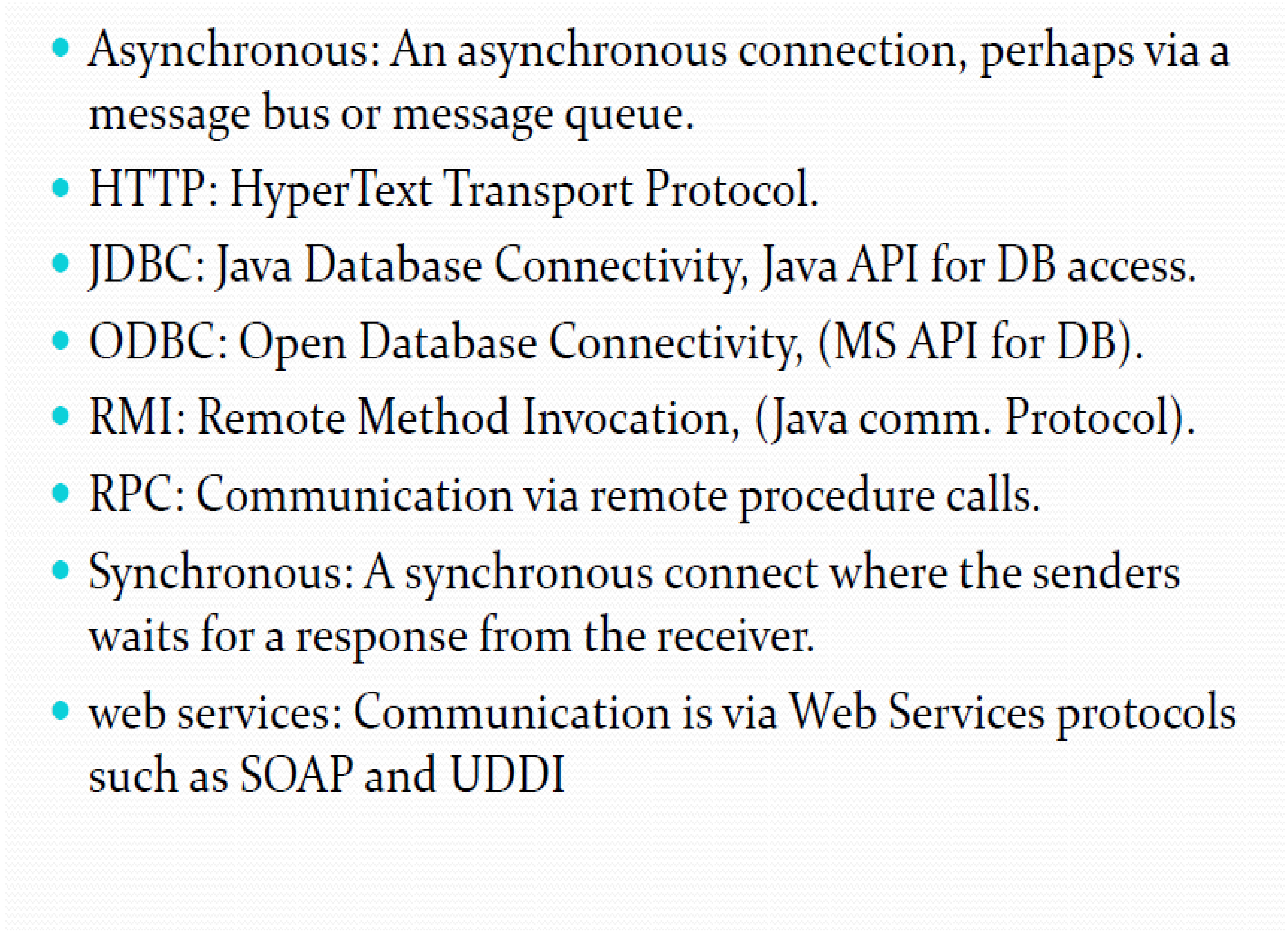
- A file-shaped box that groups together all the device nodes to encapsulate the entire deployment



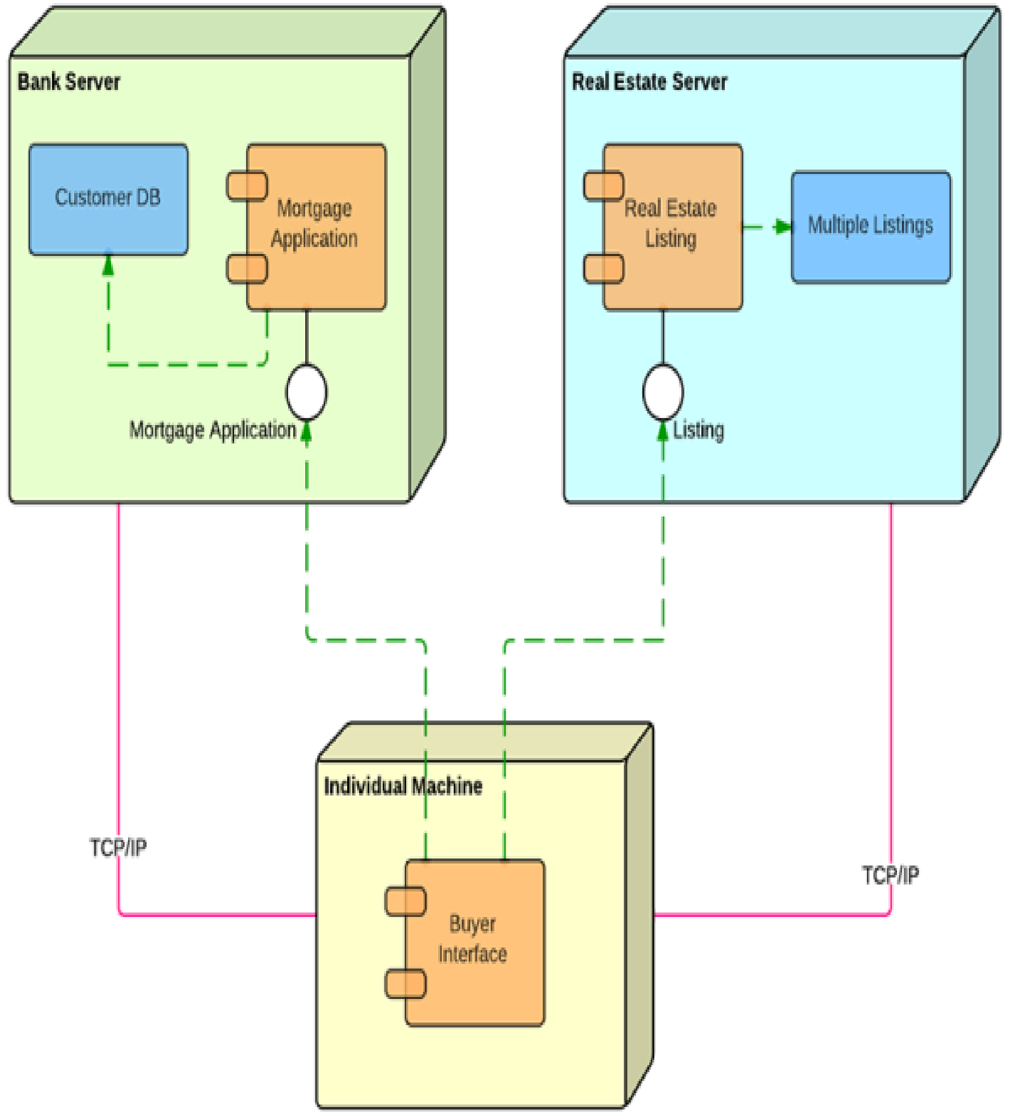
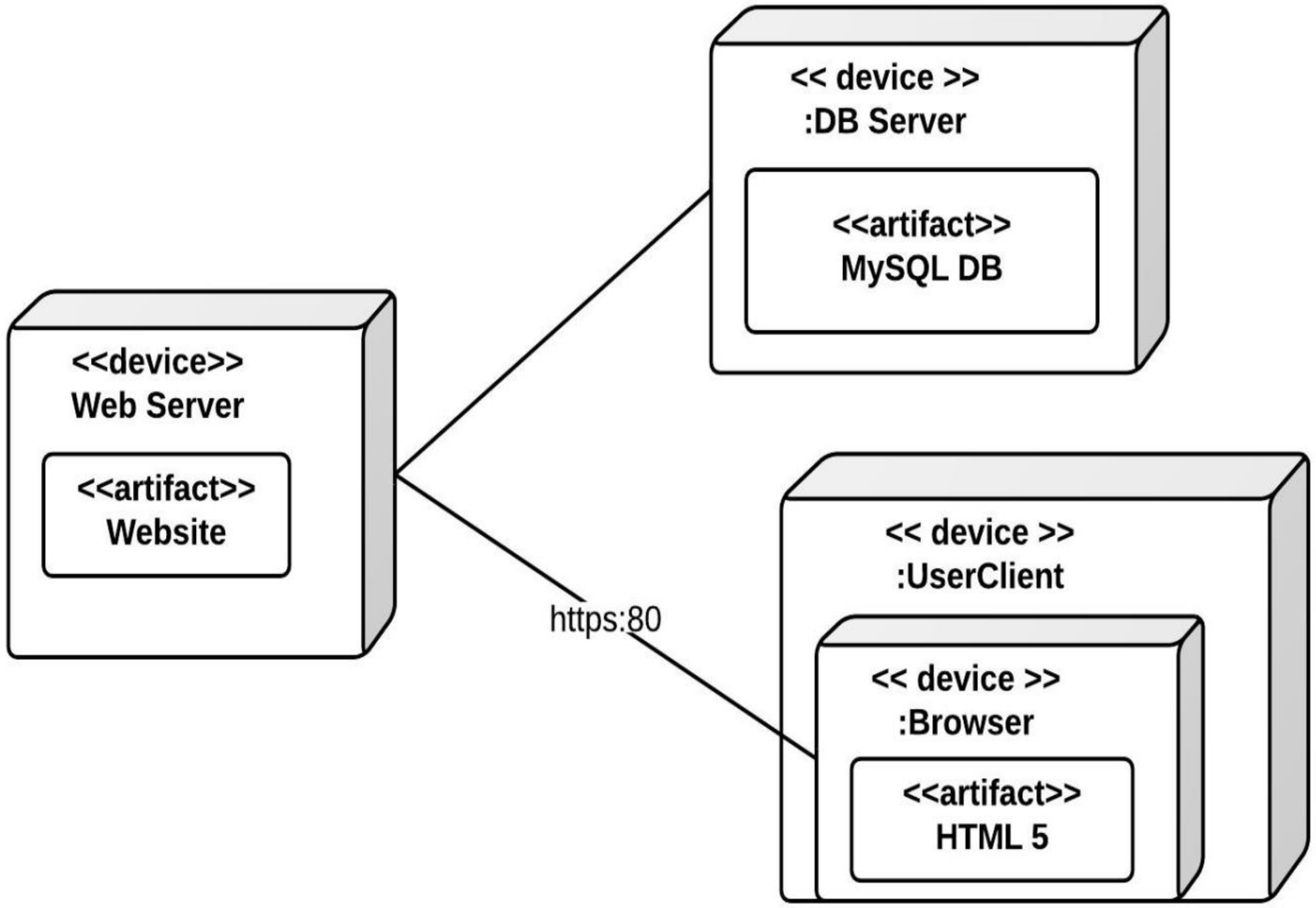
Stereotypes

- Stereotypes is extensibility mechanisms in UML which allows designers to extend the vocabulary of UML in order to create new model elements.
- By applying appropriate stereotypes in your model you can make the specification model comprehensible





- Asynchronous: An asynchronous connection, perhaps via a message bus or message queue.
- HTTP: HyperText Transport Protocol.
- JDBC: Java Database Connectivity, Java API for DB access.
- ODBC: Open Database Connectivity, (MS API for DB).
- RMI: Remote Method Invocation, (Java comm. Protocol).
- RPC: Communication via remote procedure calls.
- Synchronous: A synchronous connect where the senders waits for a response from the receiver.
- web services: Communication is via Web Services protocols such as SOAP and UDDI


Bank Server
Customer DB
Mortgage Application
Mortgage Application
Real Estate Server
Real Estate Listing
Multiple Listings
Listing
Individual Machine
Buyer Interface
TCP/IP
TCP/IP

<<device>>
Web Server
<<artifact>>
Website
<< device >>
:DB Server
<<artifact>>
MySQL DB
https:80
<< device >>
:UserClient
<< device >>
:Browser
<<artifact>>
HTML 5

What to consider before heading towards drawing a deployment diagram?

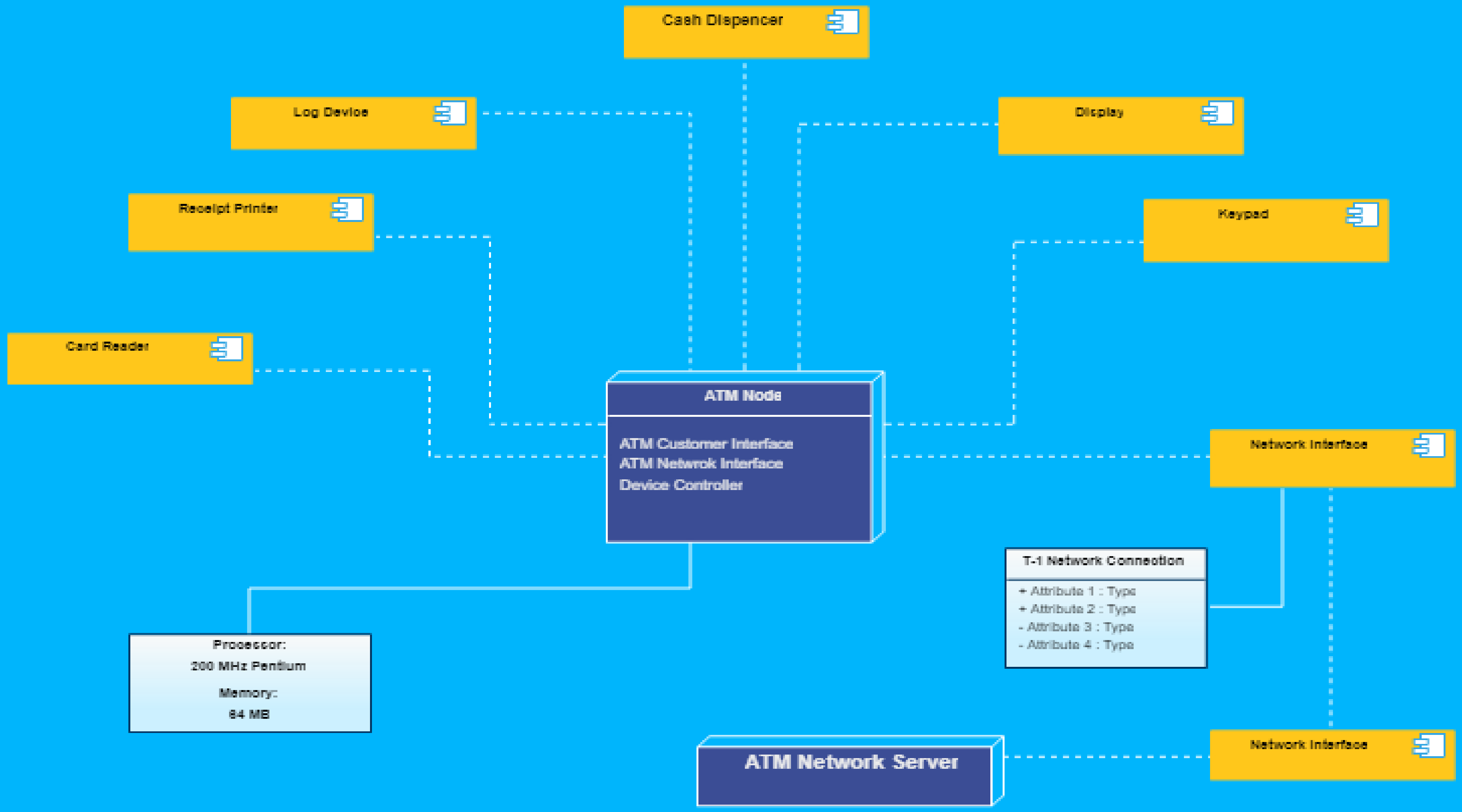
1. **Have you identified the scope of your system?**
2. **What are the limitations of your physical hardware?**
3. **Which distribution architecture are you using?**
4. **Do you have all the nodes you need? Do you know how they are all connected?**
5. **Do you know which components are going to be on which nodes?**

How to draw a deployment diagram

- Decide on the purpose of the diagram.
- Add nodes to the diagram.
- Add communication associations to the diagram.
- Add other elements to the diagram, such as components or active objects, if required.
- Add dependencies between components and objects, if required.

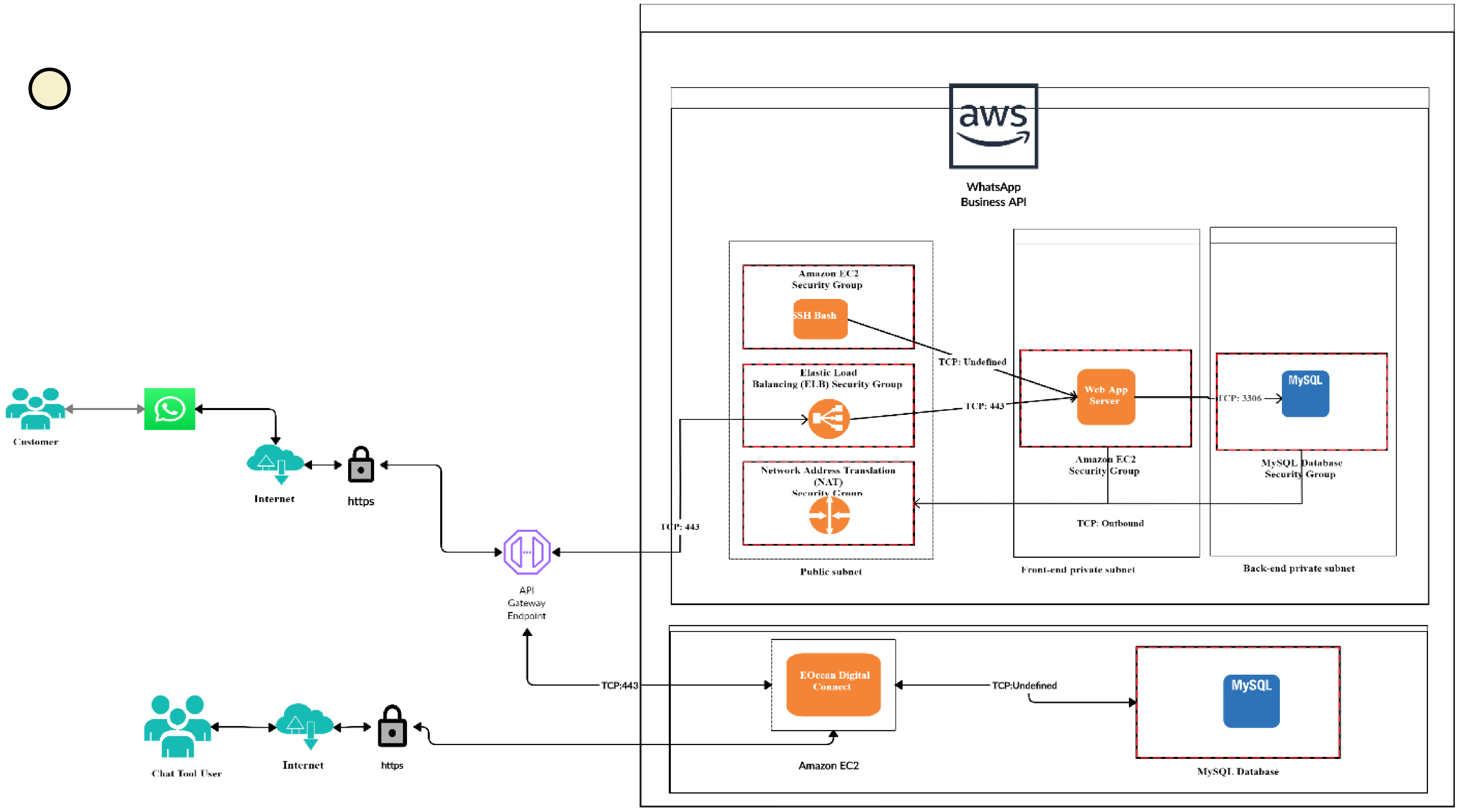
Scenario

- Let’s consider an ATM machine.
- How many node will we have in the deployment diagram of ATM machine
  - ATM Node
  - Network Node
- Usually ATMs have two interfaces i.e Customer & network interface and one device controller underneath it.
- The ATM machine itself have many hardware attached with it including
  - Bio-metric detector
  - Card reader
  - Receipt printer
  - Keypad
  - Display screen
  - Card dispenser


Cash Dispencer
Log Device
Display
Receipt Printer
Keypad
Card Reader
ATM Node
ATM Customer Interface
ATM Netwrok Interface
Device Controller
Network Interface
T-1 Network Connection
+ Attribute 1 : Type
+ Attribute 2 : Type
- Attribute 3 : Type
- Attribute 4 : Type
Processor:
200 MHz Pentium
Memory:
64 MB
ATM Network Server
Network Interface

Real Life Example : Live WhatsApp Agent Chat Tool

- The screen next shows not so accurate but a cloud deployment diagram.
- This application have two interface i.e the agent interface and the WhatsApp interface where the customer sends/receive responses from the agent
- The entire application is hosted on AWS cloud and connected via one API gateway.
- The Node container represents the WhatsApp API container holding all essential components for the API to be hosted
- The second Node container holds components of the web application that connects to the API gateway
- All of the nodes are connected through internet externally and internally via tcp/ip

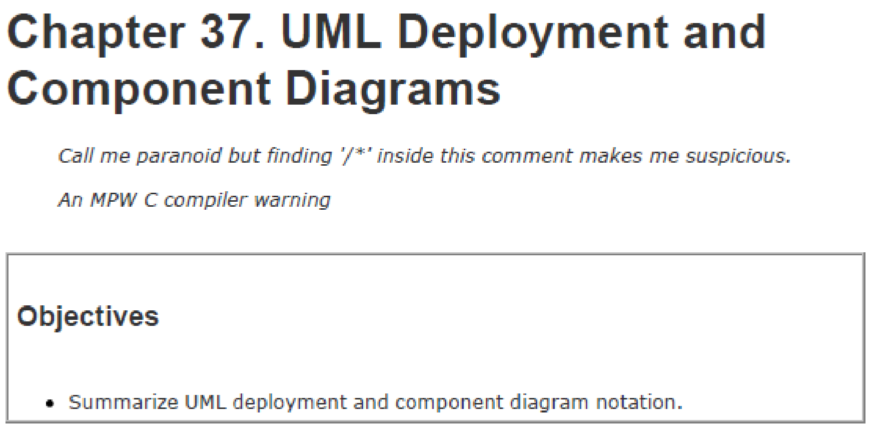

aws
WhatsApp Business API
Amazon EC2 Security Group
SSH Bash
Elastic Load Balancing (ELB) Security Group
Network Address Translation (NAT) Security Group
Public subnet
TCP: Undefined
TCP: 443
Web App Server
Amazon EC2 Security Group
TCP: Outbound
Front-end private subnet
TCP: 3306
MySQL
MySQL Database Security Group
Back-end private subnet
Customer
Internet
https
TCP: 443
API Gateway Endpoint
TCP:443
EOcean Digital Connect
Amazon EC2
TCP:Undefined
MySQL
MySQL Database
Chat Tool User
Internet
https

References

- https://www.sumologic.com/glossary/software-deployment/
- https://www.lucidchart.com/pages/uml-deployment-diagram
- https://www.visual-paradigm.com/tutorials/how-to-draw-deployment-diagram-in-uml/
- https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-deployment-diagram/
- https://creately.com/app/

READING

Chapter 37 – Applying UML and Pattern by Craig Larman 3rd Edition



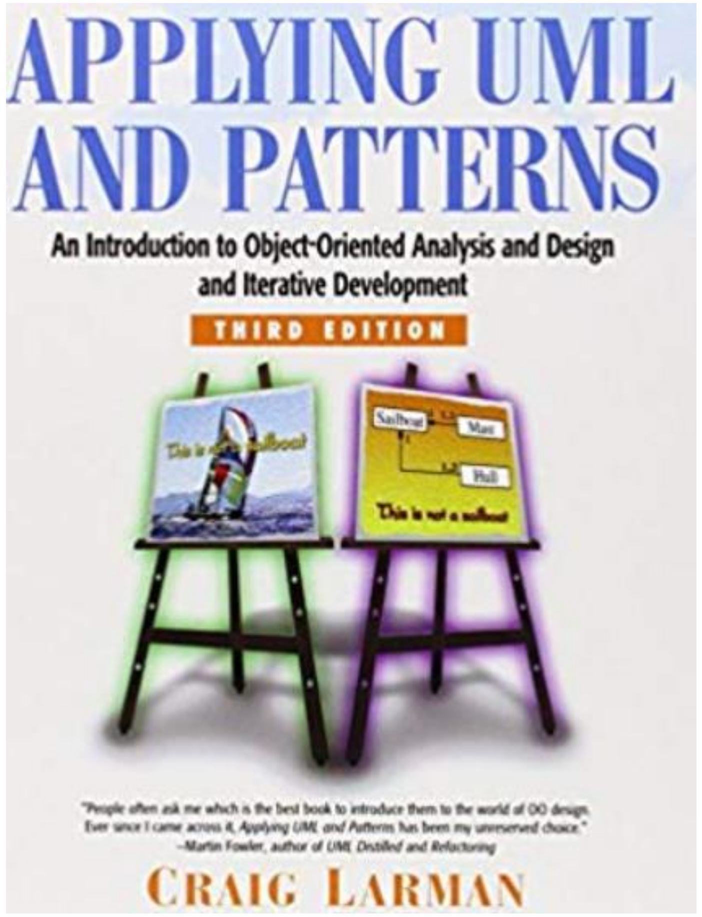
Chapter 37. UML Deployment and Component Diagrams

Call me paranoid but finding '/' inside this comment makes me suspicious.*

An MPW C compiler warning

Objectives

- Summarize UML deployment and component diagram notation.


APPLYING UML AND PATTERNS
An Introduction to Object-Oriented Analysis and Design and Iterative Development
THIRD EDITION
This is not a sailboat
CRAIG LARMAN