AVAKSH TECHNOCORP

UML

Introduction to Unified Modeling Language

Ashish Rautela

Experienced professional with a strong proficiency in various technology domains. I have successfully executed multiple projects for Fortune 500 clients and have collaborated with a company accredited at CMM Level 5. My primary focus area is to assist my clients in achieving digital transformation within their business operations.

Vivek Srivastava

Experienced professional with a comprehensive skill set that encompasses various technologies. I possess deep expertise, visionary thinking, and a notable portfolio of innovative projects. My focus is on assisting businesses in achieving their objectives by leveraging technology and domain knowledge.

AVAKSH TECHNOCORP

Agenda

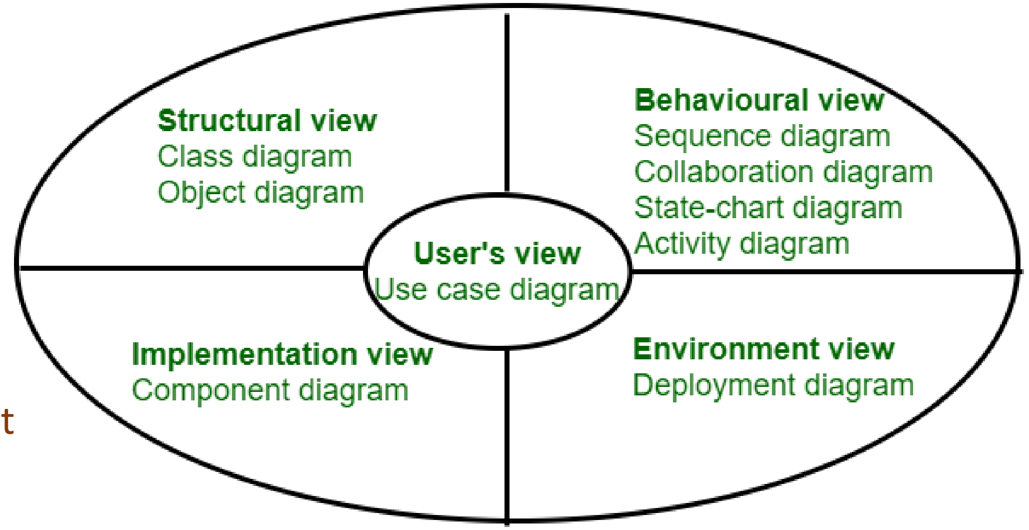
- Introduction to UML
- Evaluation of UML
- UML Diagrams
- Class Diagrams
- Use Case Diagrams
- Activity Diagrams
- Sequence Diagram
- State Machine Diagram
- Component Diagrams and Deployment Diagrams

Introduction to UML

- UML is a language for creating visual models of software and systems.
- It uses diagrams and symbols to represent different aspects of a system's architecture, structure, behavior, and interactions.
- These diagrams are used to communicate, design, analyze, and document software systems and other complex systems in a way that is easily understandable by both technical and non-technical stakeholders.

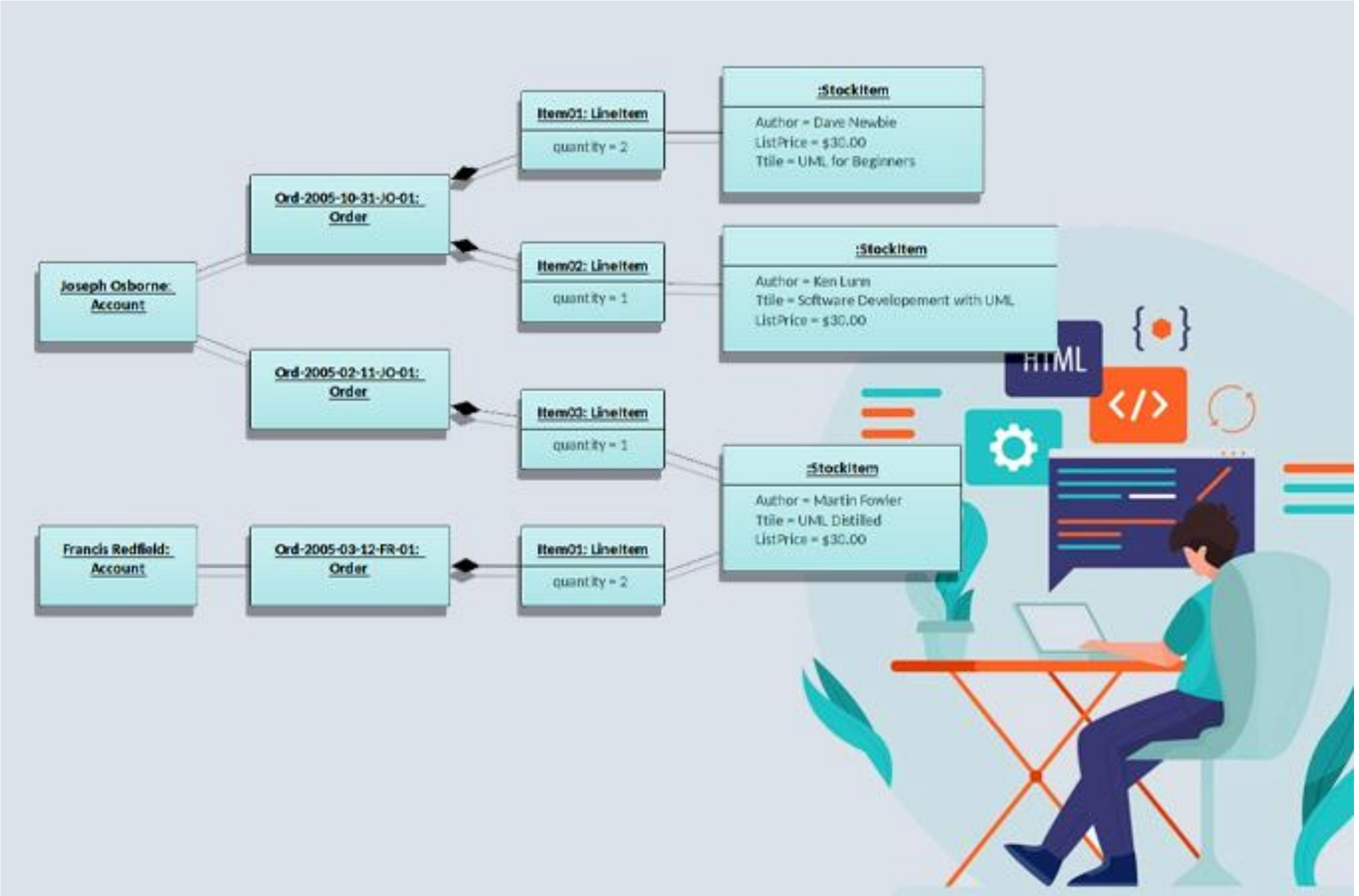
Evaluation of UML

- UML's roots can be traced back to the early 1990s when Grady Booch, James Rumbaugh, and Ivar Jacobson.
- In 1995 UML 0.8 released
- In 1997, the Object Management Group (OMG), an industry consortium, took over the standardization efforts and published UML 1.1 as the first official version of the language.
- UML 1.x included various diagram types like class diagrams, use case diagrams, and collaboration diagrams.
- In year 2005 UML 2.0 released It introduced a more precise and comprehensive specification of modeling concepts and improved support for modeling software architecture.
- UML 2.0 included additional diagram types like component diagrams, sequence diagrams, and activity diagrams
- UML 2.5, released in 2015, improved the language's usability and alignment with modern software engineering practices.
- It introduced features for modeling architectural patterns, better support for modeling at various levels of abstraction, and improved integration with other standards like SysML.



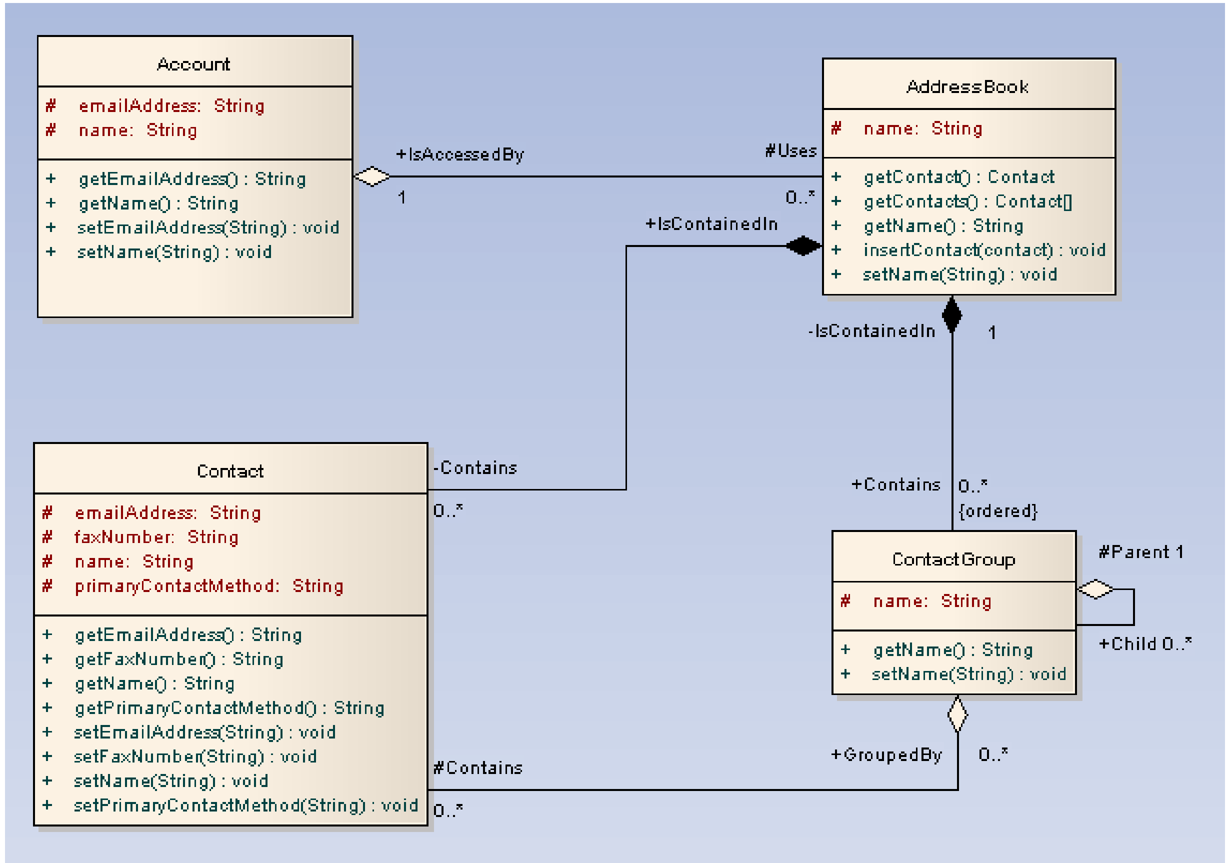
UML Diagrams

- Structural Diagrams
  - Class Diagram
  - Object Diagram
  - Component Diagram
  - Package Diagram
  - Deployment Diagram
- Behavioral Diagrams
  - Use Case Diagram
  - Activity Diagram
  - State Chart Diagram
  - Sequence Diagram
  - Communication Diagram
- Interaction Diagrams
  - Sequence Diagram
  - Communication Diagram
  - Timing Diagram
  - Interaction Overview Diagram
  - Collaboration Diagram
  - Interaction Diagram (UML 2.0)



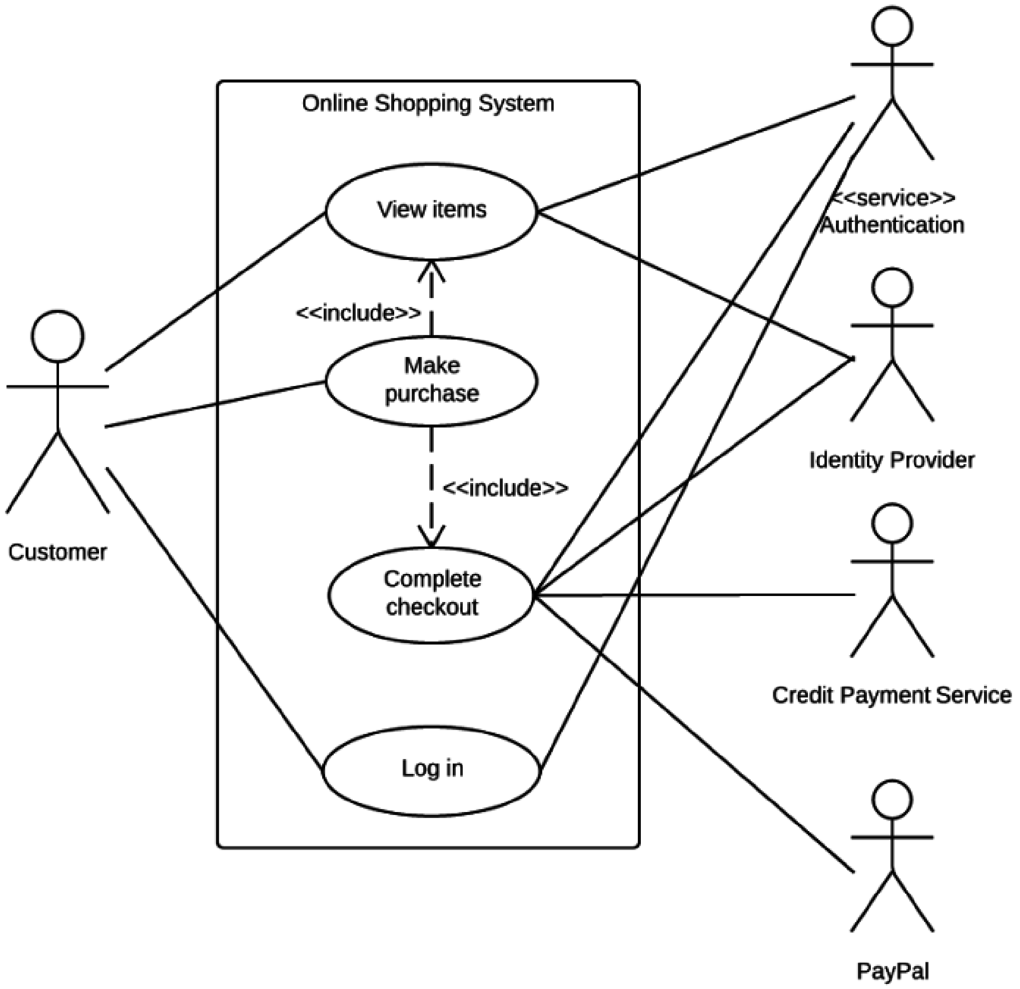
Class Diagrams

- Exploring the structure of class diagrams
- Classes, attributes, and methods
- Relationships (association, aggregation, composition, inheritance).
- Multiplicity and constraints



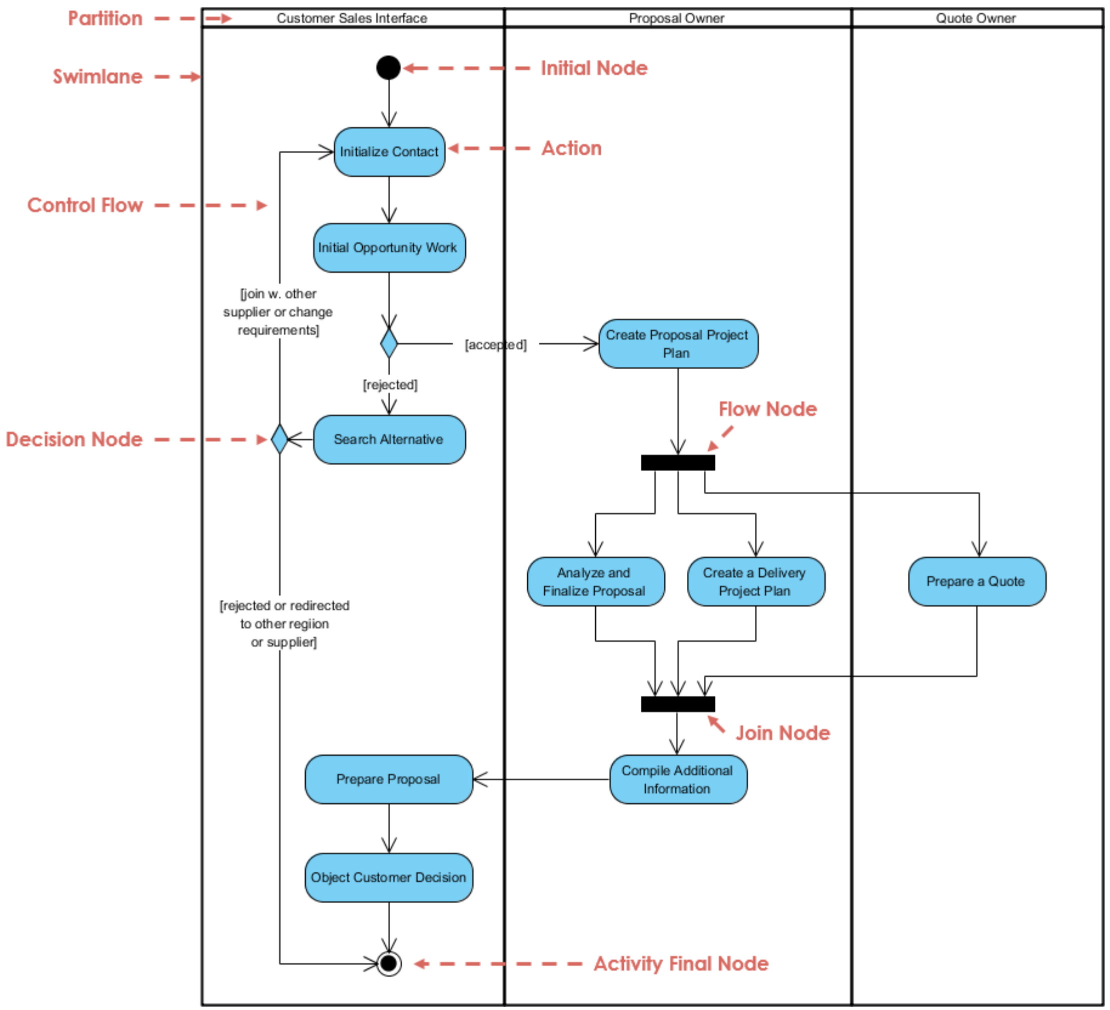
Use Case Diagrams

- Understanding use case diagrams
- Actors and use cases
- Relationships (association, generalization, inclusion, extension)



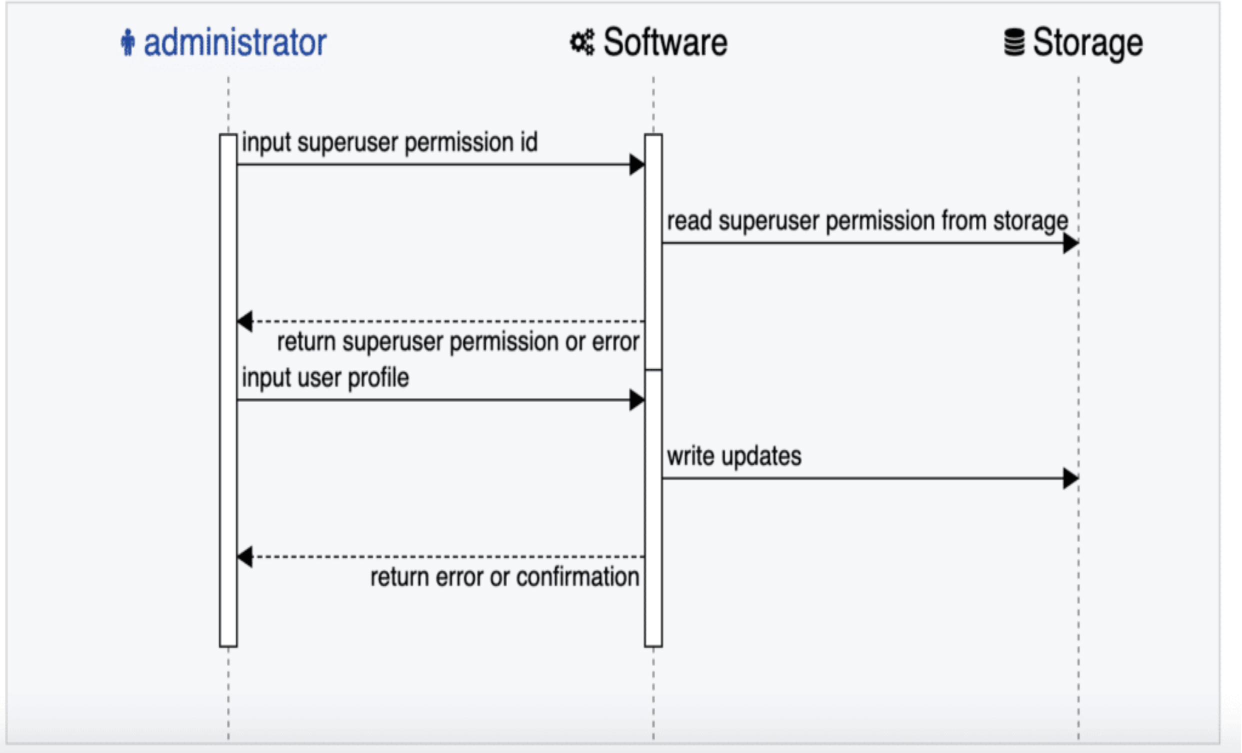
Activity Diagrams

- Dynamic modelling of the system or a process.
- Illustrate the various steps involved in a UML use case.
- Model software elements like methods, operations and functions
- Show the constraints, conditions and logic behind algorithms.
- Flows and transitions



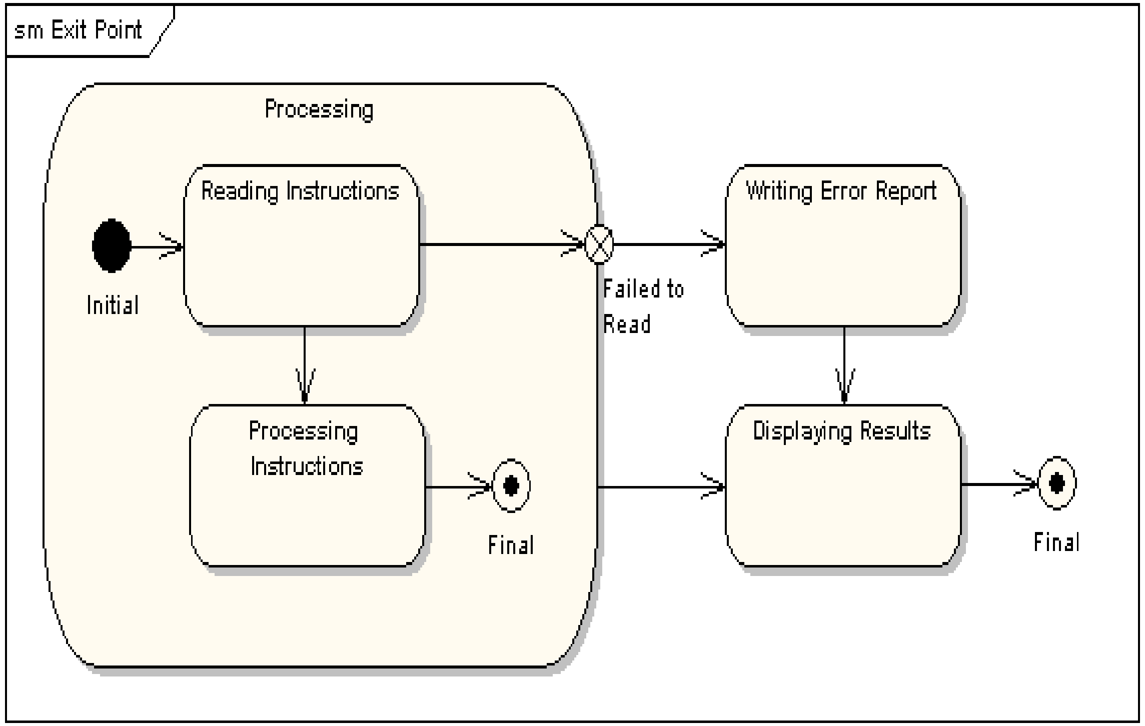
Sequence Diagrams

- Visualizing interactions between objects
- Lifelines, messages, and activation bars.
- Synchronous and asynchronous communication.


administrator
Software
Storage
input superuser permission id
read superuser permission from storage
return superuser permission or error
input user profile
write updates
return error or confirmation

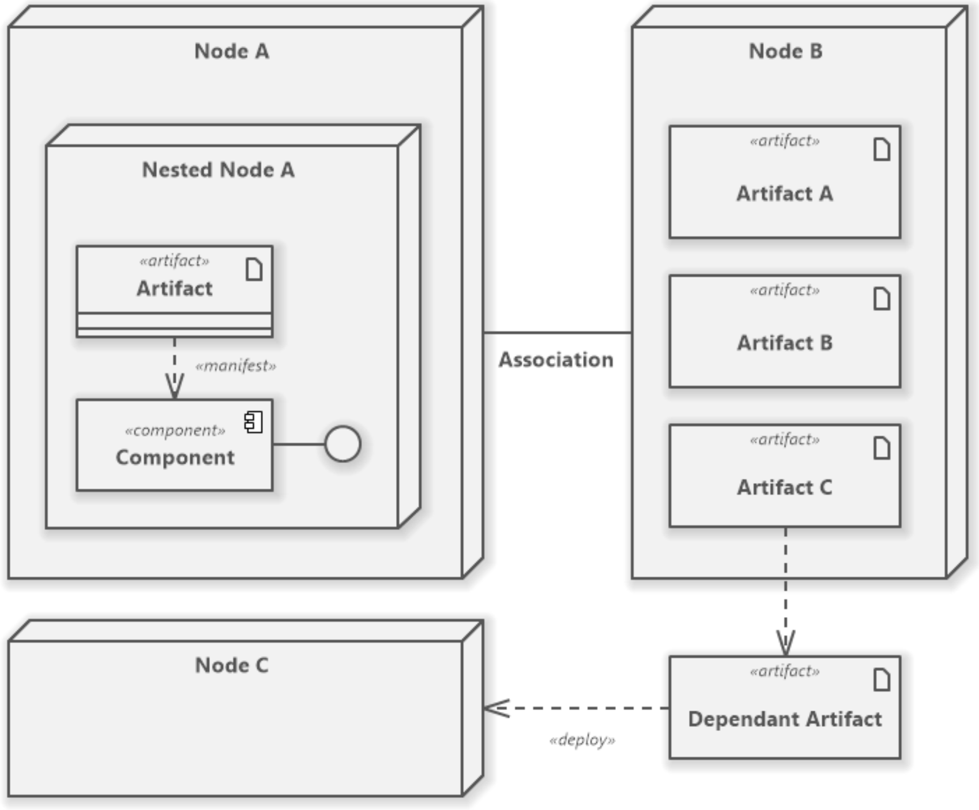
State Chart Diagrams

- Modeling the behavior of objects
- States, transitions, and events
- Hierarchical and concurrent states



Component Diagrams and Deployment Diagrams

- Visualize the hardware topology of a system
- Components, nodes, and interfaces.
- Mapping software components to hardware



Conclusion

- Recap of UML Diagrams
- Best Practices

Q&A

Thank You

viveks@avaksh.com

ashish.rautela@avaksh.com