# Practical - 05.

- Subject: Object oriented modeling and design.
- Assignment title - Draw component diagrams assuming reuse of existing components with new ones.
- Semester / Year - VII, Fourth year.
- Instructor - Saiprasad Bhise.
- Submission date: 15/09/2025.

1] OBJECTIVE :-

- Understand and apply UML component diagram standards for system architecture modeling.
- Illustrate system decomposition into reusable and new components.
- Communicate system structure focusing on modularity, reuse and integration points.

2] Problem statement.

Draw one or more component diagrams representing the architectural structure of a chosen application Assume the system is build by reusing existing components alongside a few newly developed ones.

3] Introduction to component diagram modelling -

Component diagrams show how a system is divided into components and how these components interact.

They empathize the physical and logical organization of software and hardware elements.
They are essential for planning system reuse, understanding dependencies and managing large-scale development.
In UML, components are modular parts of a system with well defined interfaces.

4] THEORY AND BEST PRACTICES –

UML Elements –

- Component – Modular part of a system encapsulating functionality, represented by a rectangle with a component symbol.
- Interface – provided or required by components, represented by lollipop or socket notation.
- Dependancy – shows that one component uses or depends on another.
- Package – Optional grouping of related components.
- Ports – Interaction points of a component defining interfaces.
- Notation and naming –
- Component names should be meaningful and capitalized.

- Interface names describe provided or required functionality
- Show clear dependencies and usage between components

Layout -

- Group reused/third party/external components distictively.
- Position new/custom components centrally or logically.
- Use clear dependency arrows and interface connectors.

5.] Assignment workflow -

1. System definition and boundary - briefly describe system scope and focus.
2. Identify existing and new components - list reusable components and new components.
3. Define interfaces - Specify interface components provide and require.
4. Draw component diagrams - Using UML notation, illustrate the components their interfaces and dependencies.
5. Document component descriptions - Provide detailed templates for at least two key components describing their roles, interfaces and dependencies.
6. Stakeholder validation - Describe how architectural reviews and feedback sessions would be conducted to validate component design.

6] Recommended Tools -
- UML Tools - Draw IO, Microsoft Visio, Lucid chart.
- Collaborative review tools - Miro, confluence.

7] System Description -
Whatsapp is a cross platform messaging application supporting text, multimedia, voice and video communication. The system is composed of reused components such as encryption libraries, database management systems, and network protocols along with custom-developed components specific to whatsapp business logic and UI.

8] Assignment requirements -

1. System boundary - Clearly define scope.
2. Components - Identify reused and new components.
3. Interfaces - Clearly specify provided / required interfaces.
4. dependencies - Correctly model usage and dependency relationships.
5. Diagrams - UML standard precise and well structured
6. Templates - Detailed description of component responsibilities, interfaces and reuse rationale.
7. Stakeholder validation - Reflection or example of feedback for architectural validation.
8. Documentation - professional formatting and readability.

9] Sample component diagram overview -

Components -

- Reused Components:
  - EncryptionLibrary (provides IMessageEncryptor interface).
  - Database Engine (provides IDatastore interface).
  - Network protocol (provides INetworkHandler).

- New components -
  - Message controller (required IMessageEncryptor, IDataStore, INetwork Handler, IMessage Handler).
  - User Interface (requires IMessage Handler).

- Key Interfaces -
  - IMessage Encryptor.
  - IDataStore.
  - INetwork Handler.
  - IMessage Handler.

10] Component Template structure -

Component Name - Message Controller.

- Description - Central component managing message creation, encryption, storage and transmission.
- Provided Interfaces - IMessage Handler.
- Required Interfaces - IMessage Encryptor, IDataStore, INetwork Handler,
- Dependencies - Uses Encryption library, Database Engine.
- Business rules - enforces message format and delivery guarantees.
- Reuse rationale - Leverages existing encryption and storage for security and reliability.

- Component Name - Encryption Library. (Reused).

- Description - Provides encryption / decryption services.
- Provided Interfaces - IMessage Encryptor.
- Dependencies - None (external third party library).
- Business rules - Supports AES 256. encryption standards.
- Reuse Rationale - Avoids reinventing cryptographic functionality.

11] Stakeholder Validation -

- Conduct architecture review sessions with developers, security experts and product owners.
- Validate reuse feasibility and component interactions.
- Gather feedback for interface completeness and dependency management.
- Refine diagrams and templates accordingly.

12] CONCLUSION -

Component diagrams are vital for visualizing system architecture, promoting modularity and facilitating component reuse.
Proper modeling of components, interfaces and dependencies supports maintainability and scalability key for large systems. Mastering component diagrams strengthens a ability to design robust, reusable and extensible software architecture.