ARCHITECTURAL DESIGN

Prepared By: Natabar Khatri

New Summit College

Unit 6

INTRODUCTION TO ARCHITECTURAL DESIGN

- Focuses on organizing a software system and designing its overall structure.
- Defines the **high-level structure** of the software system, including its major components and how they interact
- Bridges **requirements engineering** and **detailed design**.
- Outputs an architectural model that represents the system's organization as interconnected components.
- Typical Diagram includes:
  - Layered Architecture
  - Microservices Map
  - Technology Stack Overview
  - Component Diagrams
  - Deployment Diagrams

THE ROLE OF ARCHITECTURAL DESIGN

- **Early Architecture is Critical:**
  Agile values adaptability but avoids incremental architecture changes (costly!).
- **Challenge:**
  Modifying architecture requires system-wide changes.

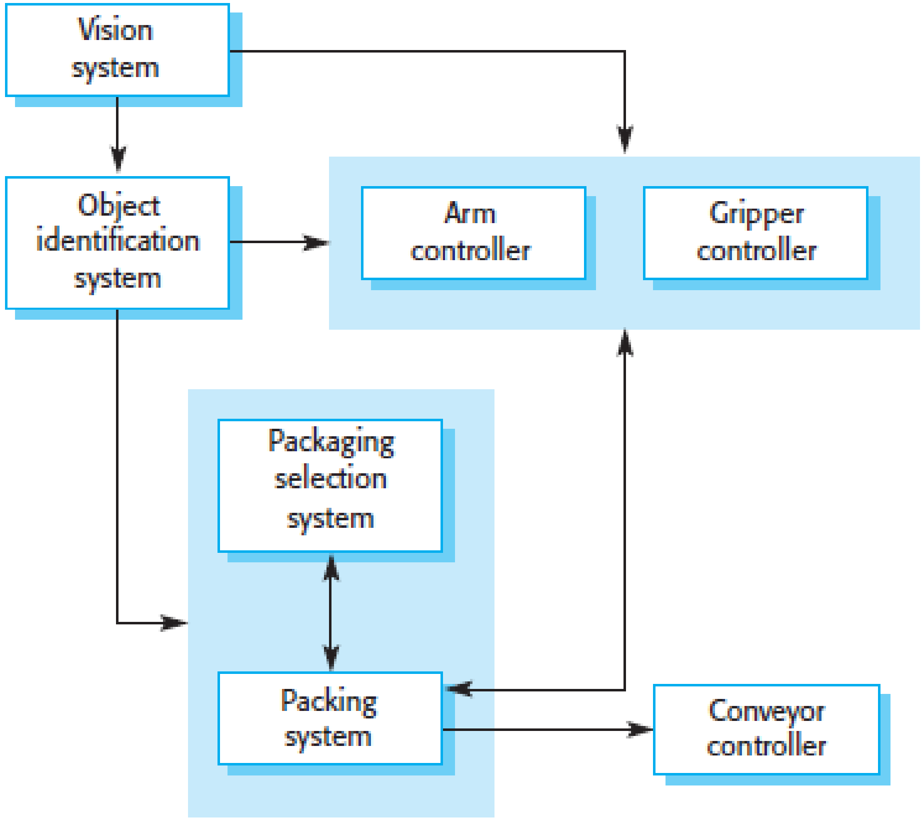

Fig: Packing robot control system architecture

RELATIONSHIP WITH REQUIREMENTS ENGINEERING

- Requirements often propose abstract architectures.
- However, there is significant overlap between requirements engineering and architectural design.
- Grouping system functions aids discussions with stakeholders about requirements

Levels of Architectural Abstraction

- Architecture operates at two levels:
  - **small (individual programs)**
  - Focuses on individual programs and their decomposition into components.
  - This level deals with how a single program is structured internally.
  - **large (complex enterprise systems).**
  - Concerns complex enterprise systems that may include multiple programs and components
  - often distributed across different computers and potentially managed by different organizations

IMPORTANCE AND BENEFITS

- **Importance of Software Architecture**
- Architecture impacts non-functional characteristics like performance, robustness, and maintainability.
- Non-functional requirements significantly influence architectural decisions.

- **Benefits of Architectural Design**
  - **Stakeholder communication**: The architecture serves as a high-level representation that facilitates discussion among various stakeholders.
  - **System analysis**: Early architectural design requires analysis that helps ensure the system can meet critical requirements.
  - **Large-scale reuse**: Architectural models enable reuse across systems with similar requirements, supporting approaches like product-line architectures.

REPRESENTING SYSTEM ARCHITECTURE

- System architectures are commonly modeled using informal block diagrams like in Packing robot control system architecture
  - Boxes represent components
  - Nested boxes indicate component decomposition
  - Arrows show data or control flow between components

Dual Purpose of Architectural Models

Architectural models serve two primary purposes:

- **Facilitating design discussions**: The high-level abstraction helps stakeholders understand and discuss the system without getting bogged down in details.
- **Documenting the architecture**: More detailed models provide a complete description of components, interfaces, and connections to support system understanding and evolution.

ARCHITECTURAL DESIGN DECISIONS

Architectural design is a creative decision-making process impacting system structure and development.

Key Architectural Decisions

- The architectural design process revolves around answering several fundamental questions:
- **Architecture Reuse**: Whether existing architectural templates can serve as a foundation for the new system. Many systems within the same application domain share similar architectures that reflect core domain concepts. Application product lines exemplify this approach, building variants from a common core architecture.
- **System Distribution**: How the system will be allocated across hardware resources. While embedded systems and personal applications might not require distributed architectures, most large systems must address distribution across multiple computers, significantly impacting performance and reliability.

ARCHITECTURAL DESIGN DECISIONS

- **Architectural Patterns**: The selection of appropriate architectural styles (such as client-server or layered architectures) that provide proven organizational structures for the system. These patterns capture successful architectural approaches used across various systems.
- **Structural Approach**: The fundamental strategy for organizing system components, including how they will be decomposed into subcomponents and how control will be managed between them.
- **Non-functional Requirements Alignment**: Determining the architectural organization best suited to deliver critical non-functional requirements like performance, security, or maintainability.
- **Documentation Strategy**: Deciding how to effectively capture and communicate the architectural design.

ARCHITECTURAL PATTERNS AND STYLES

- Architectural patterns (or styles) provide templates for system organization that have proven successful in various contexts. These patterns help answer several of the fundamental architectural questions by offering:
  - Proven structural approaches
  - Established component relationships
  - Documented strengths and weaknesses
  - Guidance on applicability to different system types
- The choice of architectural style directly influences how components are structured, decomposed, and controlled within the system.

ALIGNING ARCHITECTURE WITH NON-FUNCTIONAL REQUIREMENTS

- The architecture must be carefully designed to support the system's non-functional requirements, which often involve trade-offs between competing priorities:
- **Performance**: Favors architectures with:
  - Localized critical operations
  - Fewer, larger components
  - Minimized network distribution
  - Potential for component replication
- **Security**: Requires:
  - Layered structures
  - Protected core components
  - Rigorous validation mechanisms
  - Clear security boundaries

ALIGNING ARCHITECTURE WITH NON-FUNCTIONAL REQUIREMENTS

- **Safety**: Benefits from:
  - Co-located safety-critical operations
  - Minimal safety-critical components
  - Built-in protection systems
  - Clear failure modes
- **Availability**: Needs:
  - Redundant components
  - Hot-swappable elements
  - Minimal single points of failure
- **Maintainability**: Thrives with:
  - Fine-grained, self-contained components
  - Clear separation of concerns
  - Minimal shared state
  - Well-defined interfaces

ARCHITECTURAL EVALUATION

- Evaluating architectural designs presents challenges since the ultimate test occurs during system operation. However, architects can perform preliminary evaluations by:
- Comparing designs against reference architectures
- Assessing alignment with generic architectural patterns
- Reviewing pattern characteristics against system requirements
- Considering documented experiences with similar architectures

THE CORE ACTIVITIES OF THE ARCHITECTURAL DESIGN PROCESS

- **Requirements Analysis and Partitioning**
- **Architectural Synthesis / Conceptual Design**
- **Modeling and Documentation**
- **Analysis and Evaluation**
- **Implementation and Realization**
- **Evolution and Maintenance**