Architectural Design

Introduction

Introduction

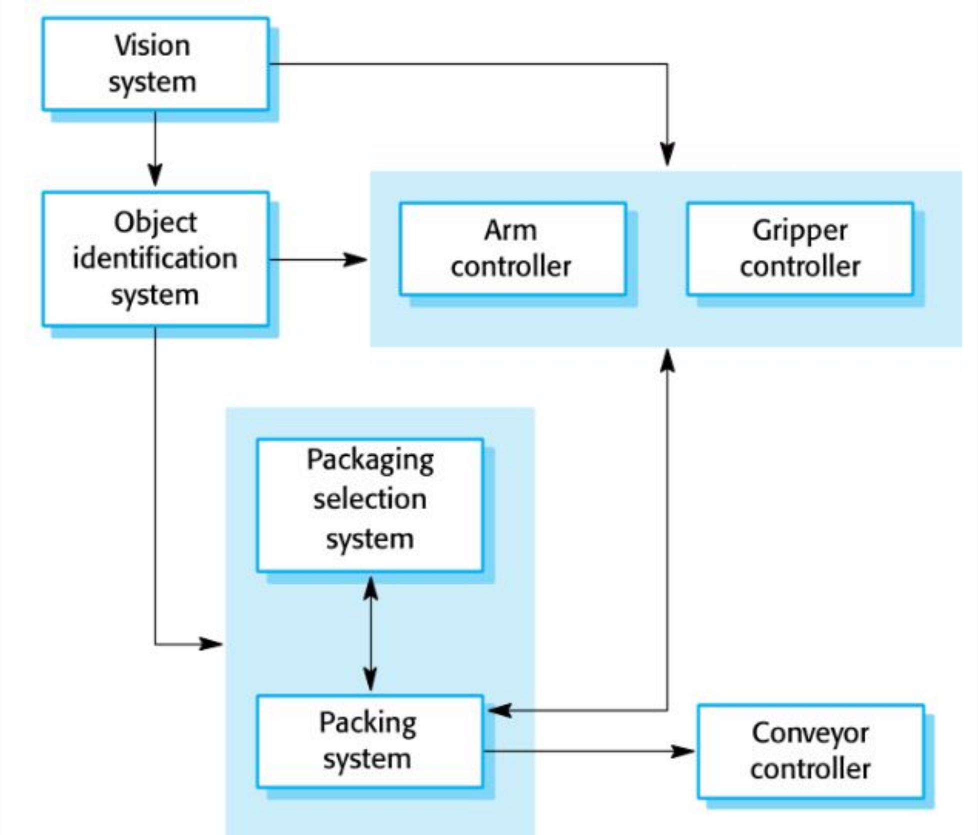
1st stage in software design process.

Describes major components, their relationships, and how they interactions.

System is organized in a set of communicating components.

Much like UML

Example – Packing Robot Control System



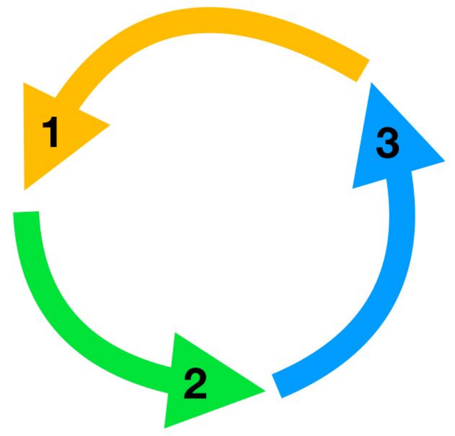
Architecture in Agile

Design architecture in early stages.

Refactor code along the cycle.

But Refactoring is expensive.

Many components are affected



Architecture & RE

Requirement Engineering is overlapped with architectural design.

Specification always includes design information.

High level features – main architectural components.

You can also identify subsystems in requirements.

Non-Functional Requirements

Requirements that apply on the system as a whole.

Represented well by architecture.

Like, robustness, performance, maintainability etc.

Architectural Design

Abstractions

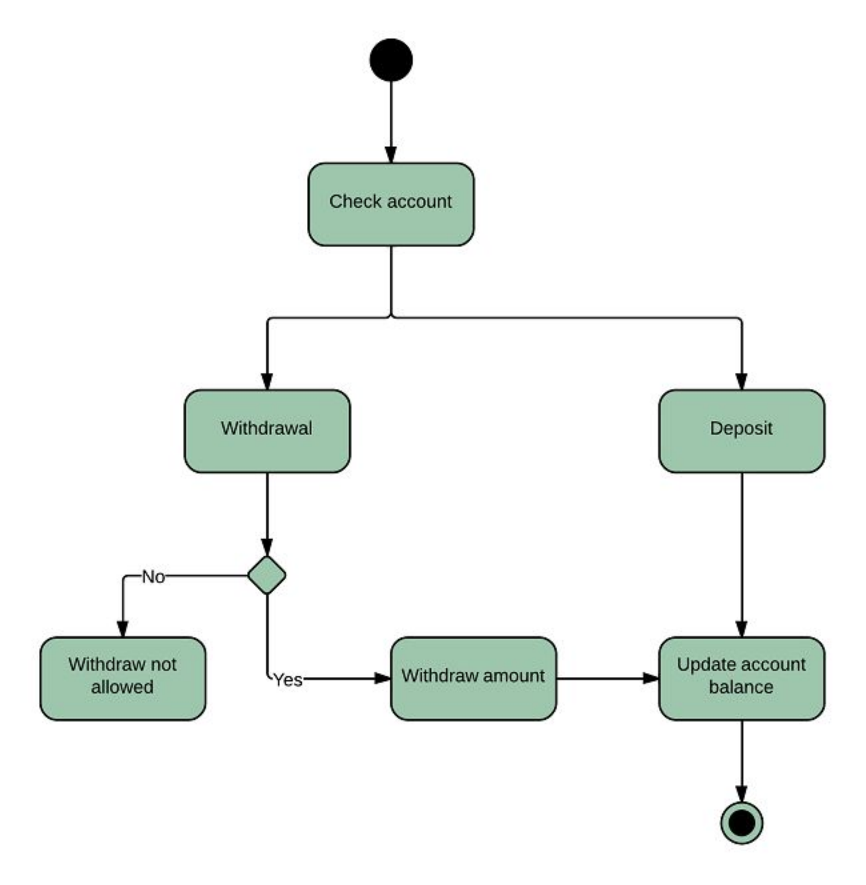
Small Abstractions

Architecture of individual programs.

Decompose an individual program into components.

Example: Activity diagram

Checkout Process Example



Large Abstractions

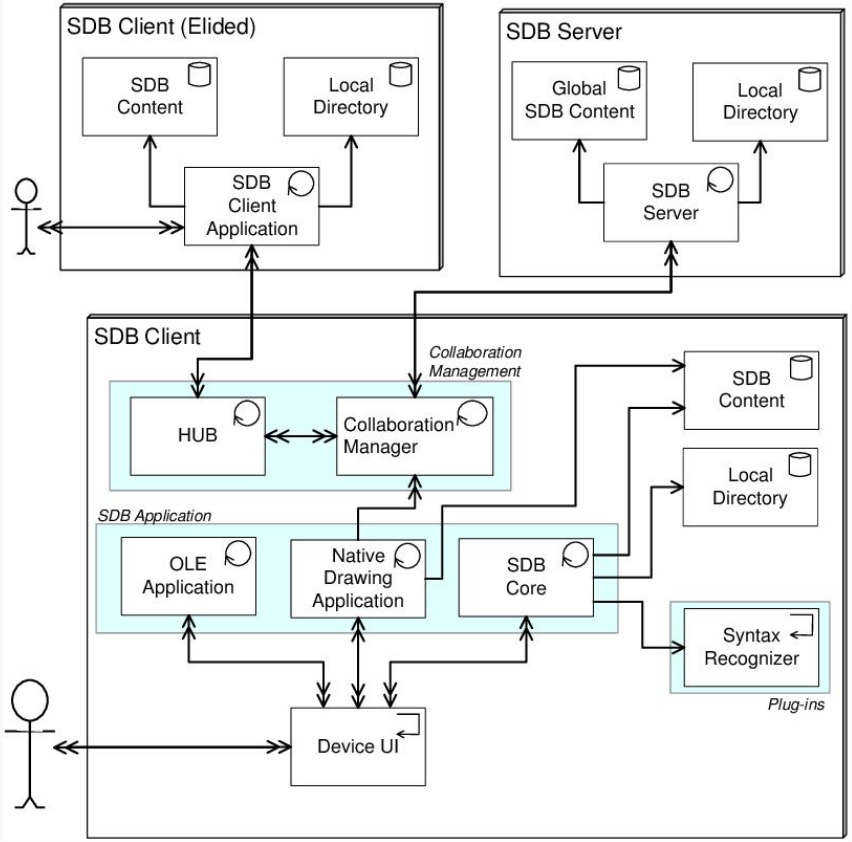
Architecture of complex enterprise systems

Include systems, programs, and components.

Distributed over different computers.

Example: Component diagram

A Client Server Architecture



Architectural Design

Importance & Advantages

Advantages

High Level Representation.

Good Communication with Stakeholders.

System Analysis – are requirements even possible?

Reusability across similar applications.

Uses

Project Planning.

Assigning Tasks.

System as a whole is discussed without understanding implementations.

Architectural Design

Representation

Representation

Informal block diagrams showing entities and relationships.

Components are represented as Box.

Sub components are represented as nested boxes.

Arrows depicts the data and control signals flowing from components.

Drawbacks

Very abstract.

They do not show nature of component relationships.

But useful for:

Communication with stakeholders

Project planning.

Architectural Design

Design Decisions

A Creative Process

No formulaic design process.

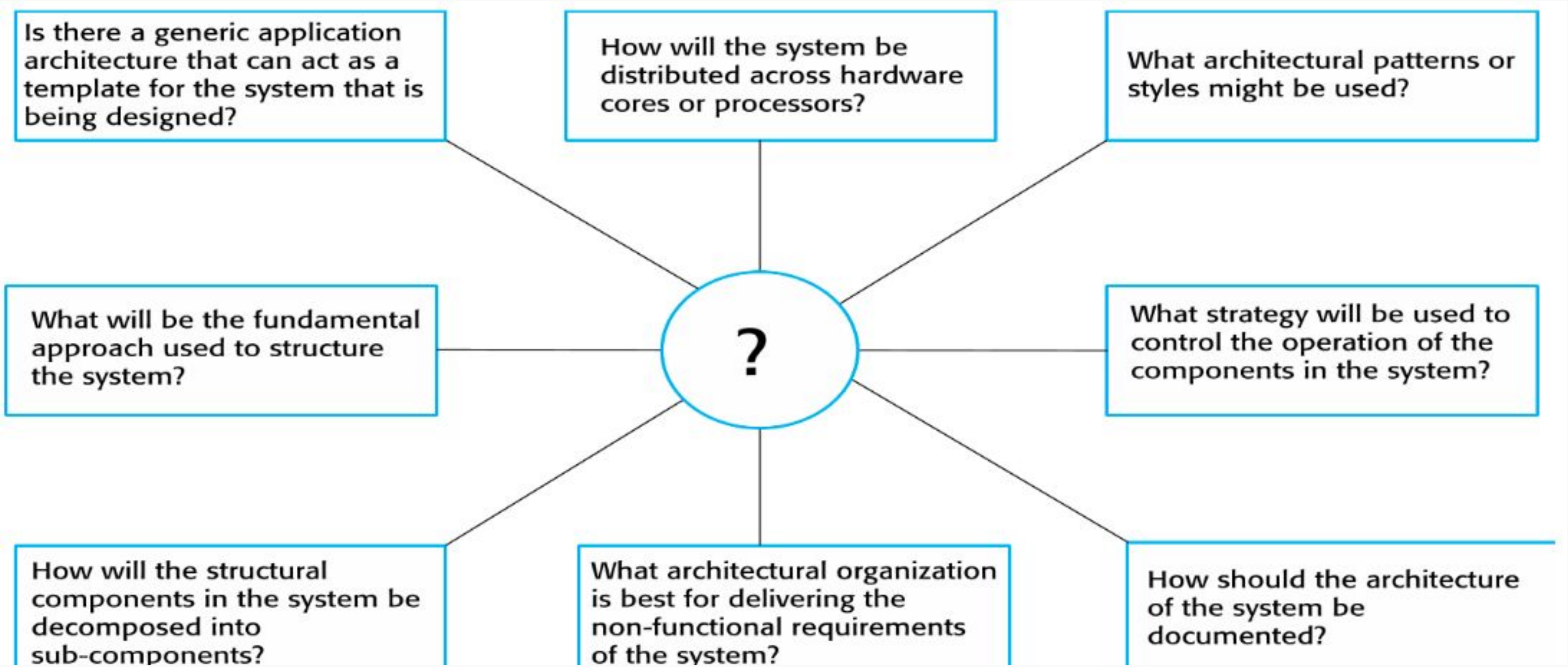
Depends upon: System type, Background, Experience, Requirements

Series of decisions to made.

Not a sequence of activities to be performed.

Common decisions span all design processes.

Decision Making Questions



Is there a generic application architecture that can act as a template for the system that is being designed?

How will the system be distributed across hardware cores or processors?

What architectural patterns or styles might be used?

What will be the fundamental approach used to structure the system?

What strategy will be used to control the operation of the components in the system?

How will the structural components in the system be decomposed into sub-components?

What architectural organization is best for delivering the non-functional requirements of the system?

How should the architecture of the system be documented?

Architectural Design

Reusability

Architecture Reuse

Same domain systems have similar architectures.

Application are built around a core architecture.

There can be changed to satisfy particular customer requirements.

Example: Ecommerce apps have CRUD operations.

Architectural Design

What to do in Critical NFRs

See Slides

Architectural Views

Introduction

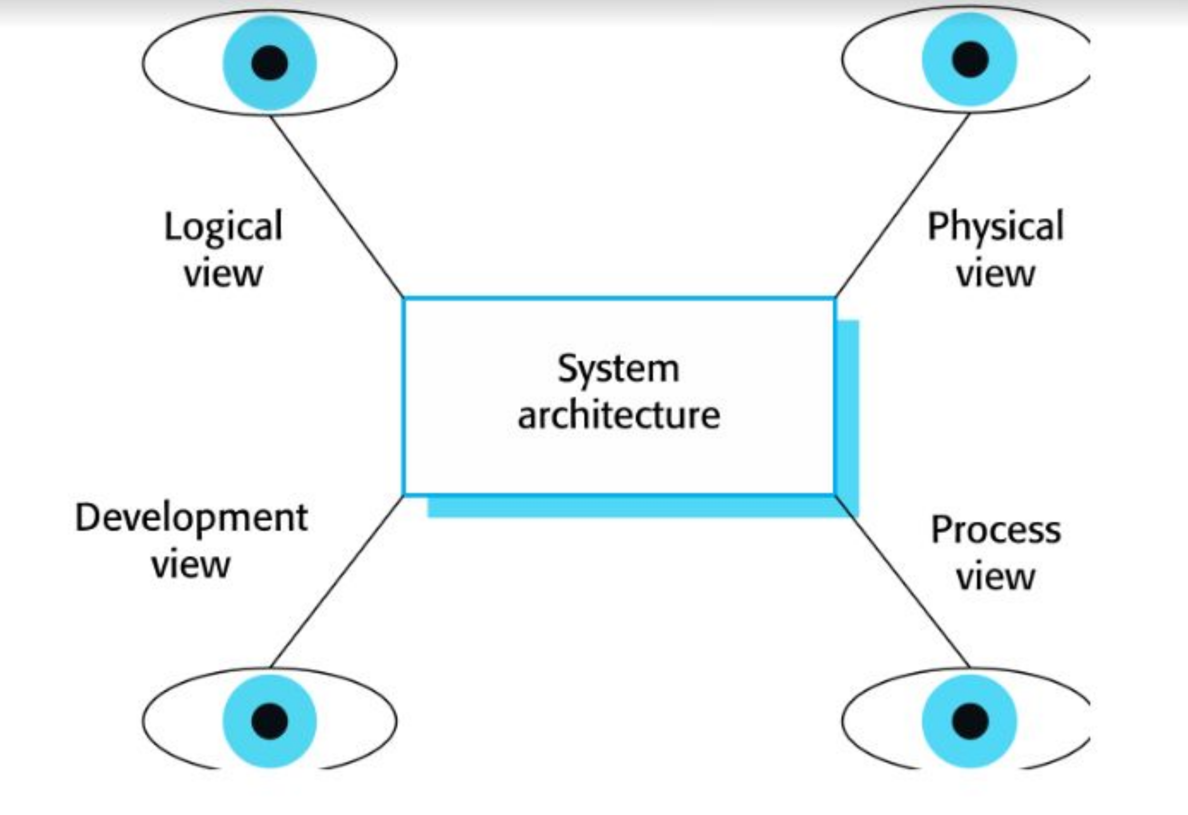
Architectural Views

Architecture can have many perspectives.

Impossible to show all perspectives in one diagram.

Multiple views to show different perspectives.

Multiple Views



Architectural Views

4+1 views

Logical View

Relates system requirements to entities.

Abstractions in the system as objects or classes.

Example: Class diagrams and state diagrams.

Process View

System interactions with processes at run-time.

Explains the processes and how they communicate.

Useful analysing NFRs like availability / performance.

Examples: sequence diagram, communication diagram, and activity diagram.

Development View

System from the standpoint of a programmer.

Shows breakdown of software components.

Example: Component diagram.

Physical View

System from the standpoint of a system engineer.

Shows how hardware and software are distributed and connected.

Example: Deployment diagram.

Representing Views

Is UML an appropriate notation?

UML is designed for OOP so it is more implementation oriented, not abstract.

UML also increases time.

Architectural description languages (ADLs) are too domain specific.

They enforce rules & guidelines.

Better to go with UMLs diagrams

Architectural Patterns

Architectural Patterns

A way of representing, sharing and reusing information. Separation and independence.

Pattern description include:

- Pattern name
- Brief description
- Graphical model
- Examples
- Pros & cons.