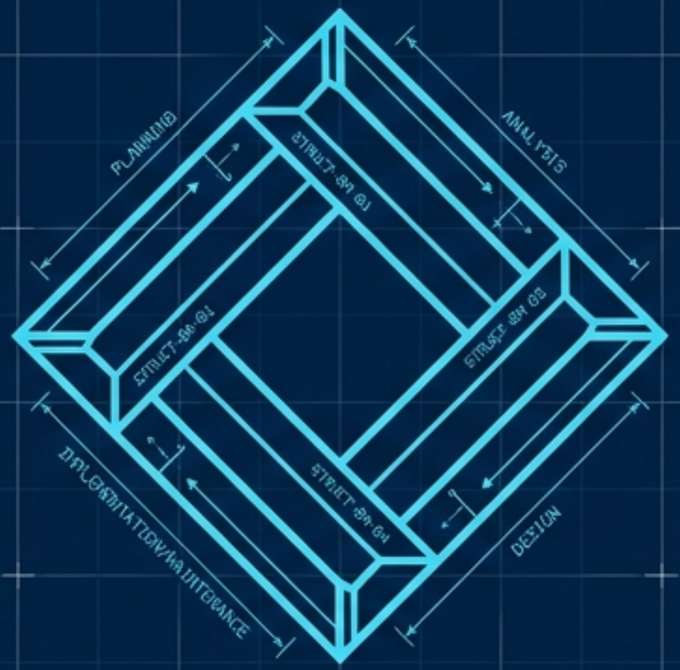
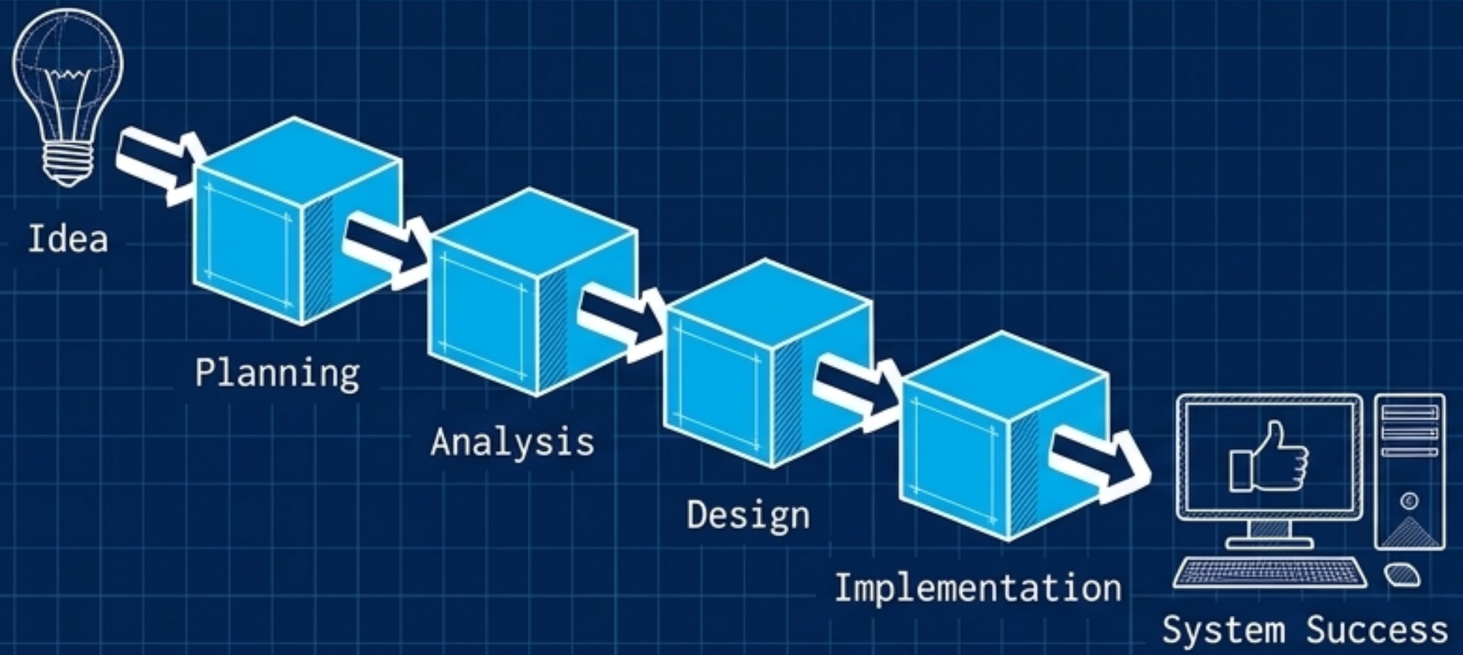


Basic Concepts of Software Development

[Mastering the SDLC, Feasibility Analysis, and Development Methodologies]

The Systems Development Life Cycle (SDLC)



The process of determining how an Information System (IS) supports business needs, designing it, building it, and delivering it.

The Systems Analyst

- Analyzes business situations
- Identifies improvement opportunities
- Designs IS solutions

Phase 1: Planning

Understanding WHY we build and HOW to manage it.

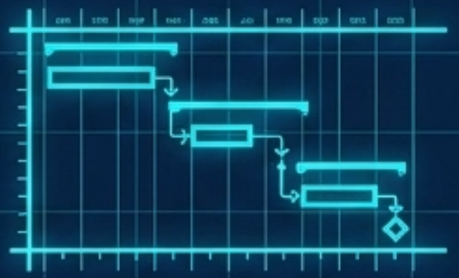
1. Project Initiation

Identifies business value.

Deliverable: System Request

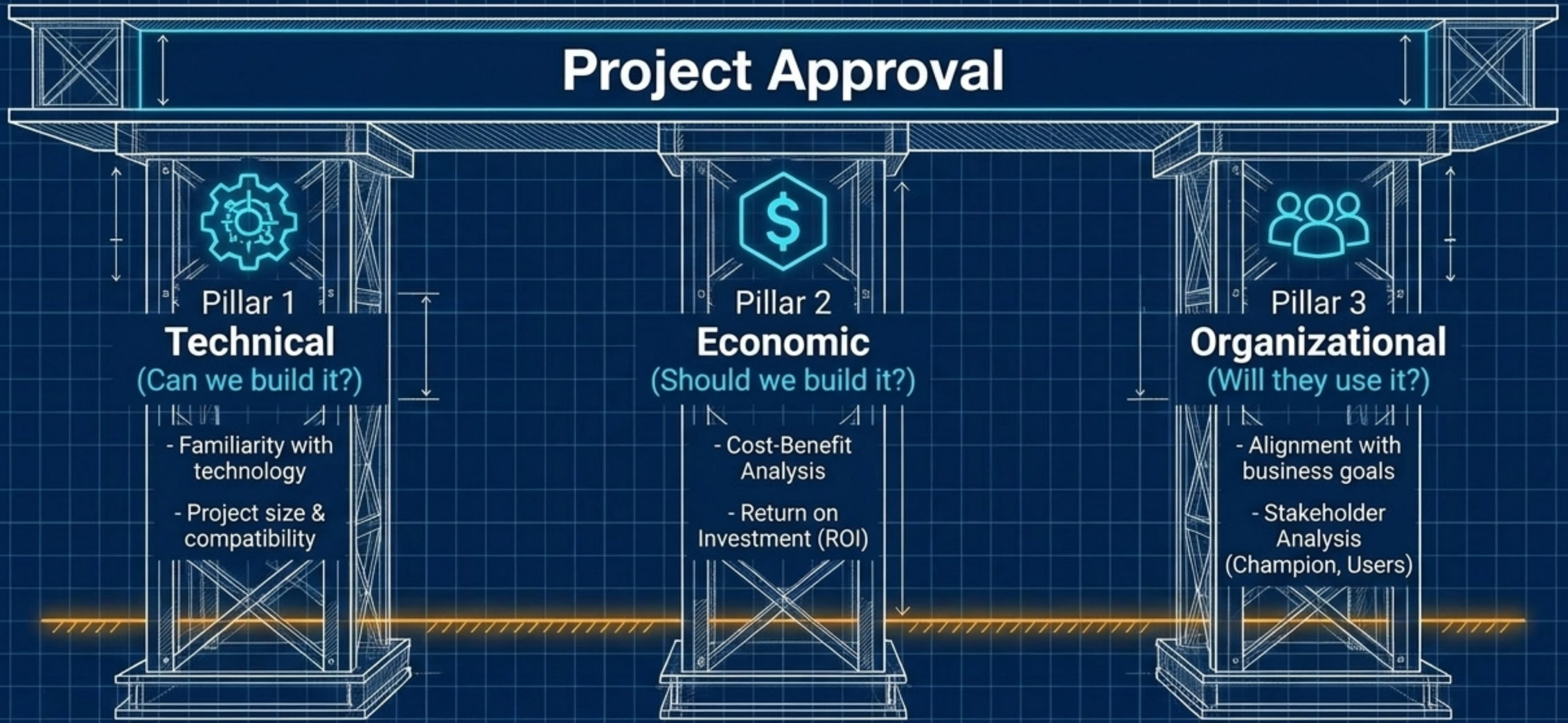
2. Project Management

Creates work plan, staffs project, implements control.



Deliverable: Project Plan

The Checkpoint: Feasibility Analysis



Deep Dive: Economic Feasibility

Costs

Development Costs

One-time expenses (e.g., developer salaries, hardware purchases).

Operational Costs

Ongoing expenses (e.g., operations staff salaries, software licensing).

Benefits

Tangible Benefits

Easily quantified and measured (e.g., 20% operating cost reduction).

Intangible Benefits

Intuitive value, hard to measure (e.g., improved customer satisfaction).

Phase 2: Analysis

Answering Who, What, Where, and When.

Techniques

- Interviews
- Joint Application Design (JAD)
- Questionnaires
- Document Analysis
- Observation

1 Analysis Strategy

Study the current system (As-Is) and envision the new system (To-Be).

2 Requirements Gathering

Develop concepts and analysis models.

3 System Proposal

Final deliverable presented to project sponsor for approval.

Phase 3: Design

Deciding HOW the system will operate.

1. Design Strategy

Clarifies if the system is built in-house, outsourced, or bought pre-written.

2. Architecture Design

Describes hardware, software, and network infrastructure.

3. Database & File Specifications

Defines exactly what data will be stored and where.

4. Program Design

Defines what programs need to be written and what they will do.

Phase 4: Implementation

Usually the longest & most expensive phase

Building and Delivering the System

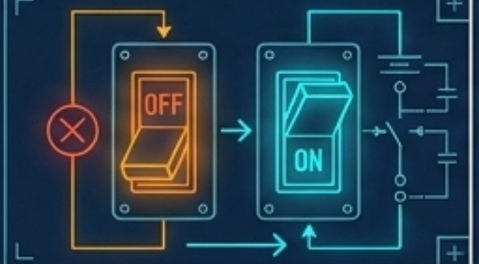
1. System Construction

The system is built and tested (Unit, Integration, System, Acceptance testing).

2. Installation

The old system is turned off and the new one is turned on.

(Requires a conversion strategy).

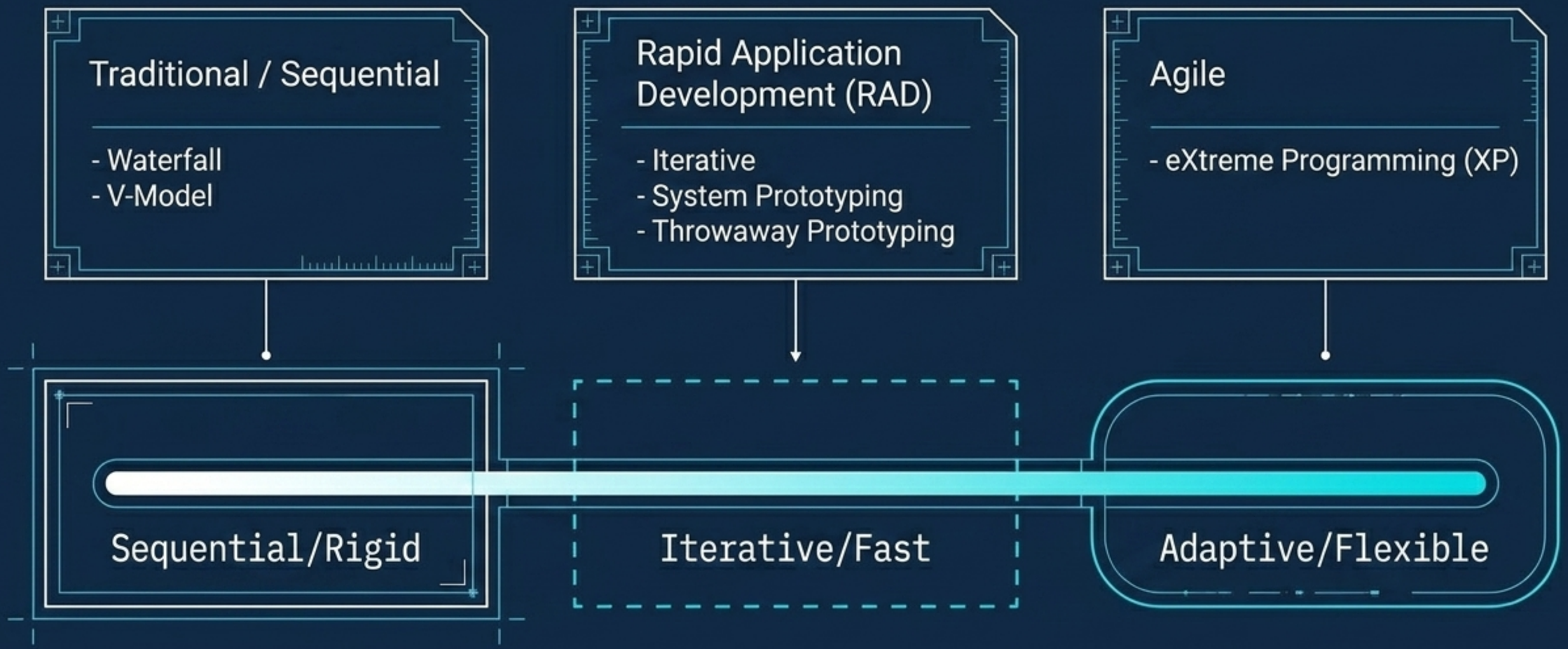


3. Support Plan

Post-implementation review and identifying needed changes.

System Development Methodologies

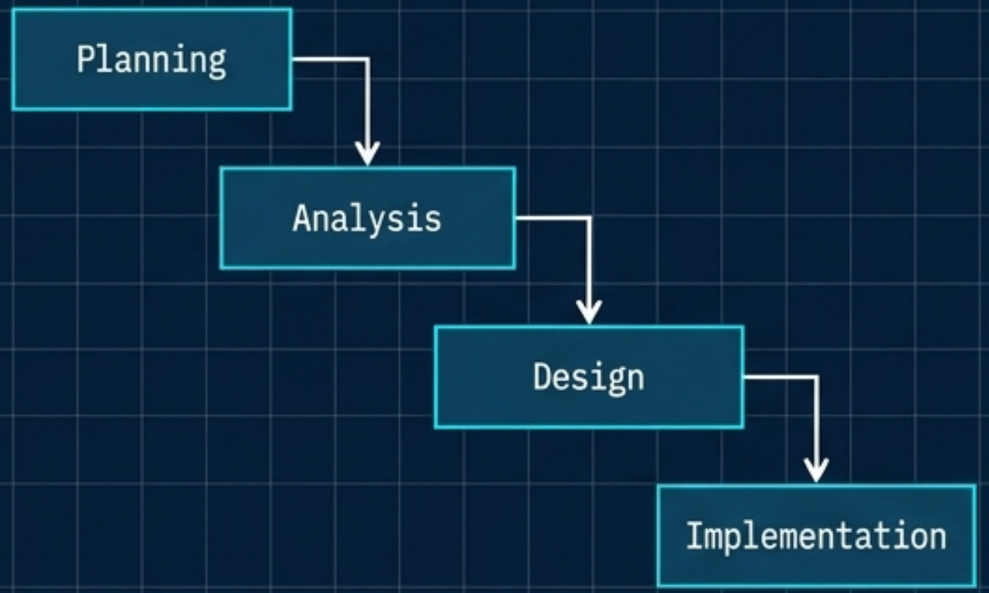
Formalized approaches to executing the SDLC.



Traditional Approaches

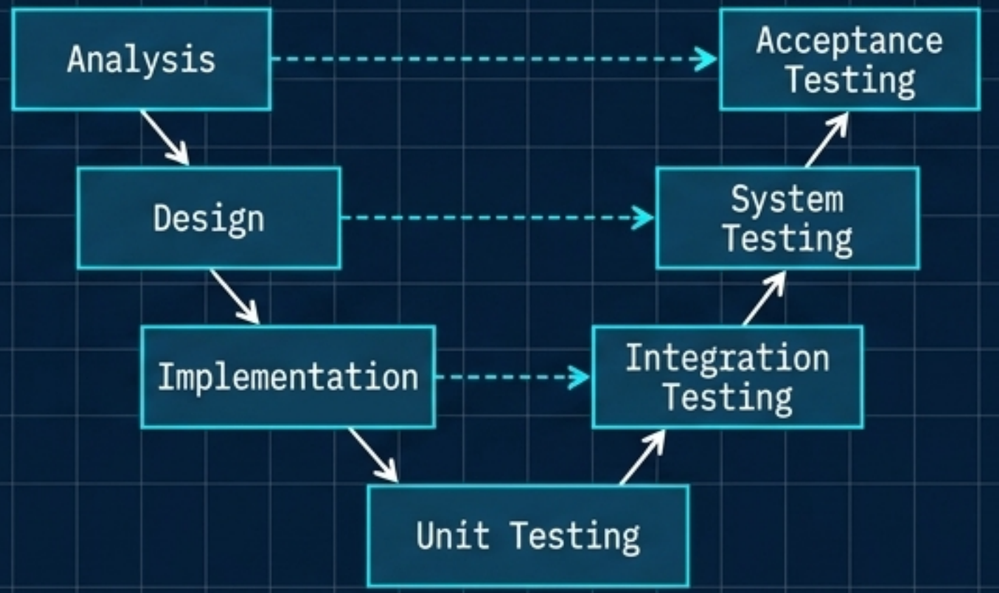
Strict, sequential flow with upfront planning.

Waterfall Development



Flow: Strict order, no overlap.
Pros: High control, upfront requirements.
Cons: Rigid, difficult to adapt to changes.

V-Model



Flow: Waterfall variation with explicit focus on Testing.
Pros: Maximizes test effectiveness; ensures high quality.
Best for: Systems requiring ultimate reliability.

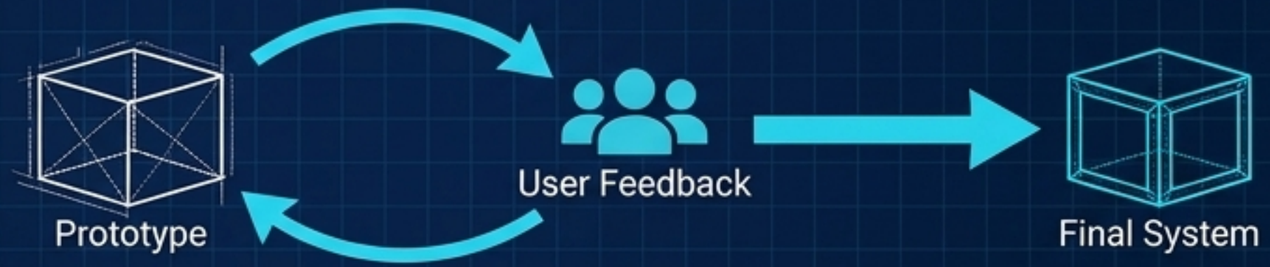
Rapid Application Development (RAD)

Getting a visible system to the user faster.

Iterative Development

Breaks project into sequential versions. V1 shapes V2.

System Prototyping



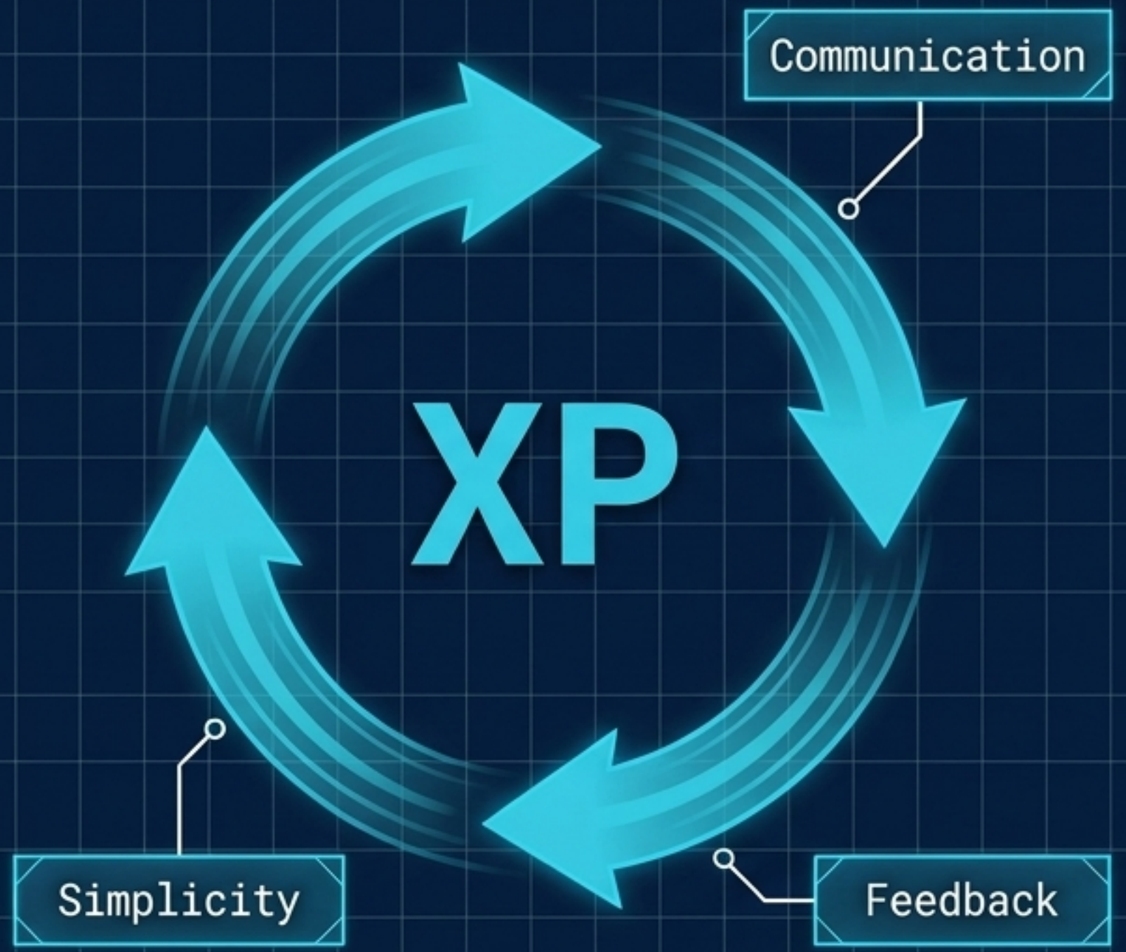
A quick and dirty version is built for user feedback, then refined into the final system.

Throwaway Prototyping

A prototype explores challenging technical issues or unclear requirements, then is discarded. The actual system is built properly afterwards.

Agile Development

Streamlining the SDLC through adaptability.



eXtreme Programming (XP)

Focus: Continuous testing, simple coding, close user interaction.

Best for: Highly motivated, cohesive, and experienced teams.

Warning: Produces very little design documentation (relies on code documentation). Maintenance can be extremely difficult for large systems.

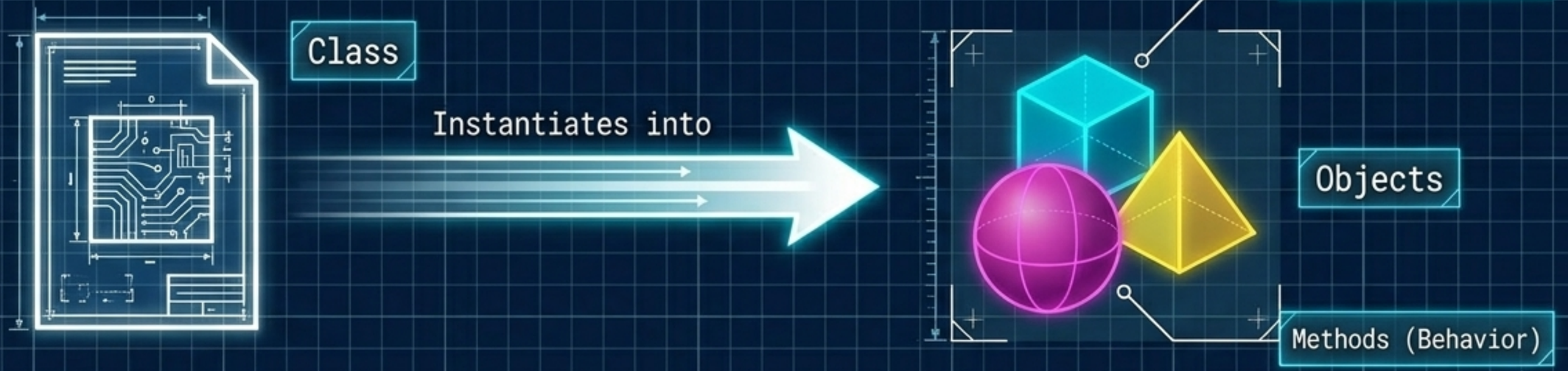
The Ultimate Decision Matrix

Selecting the Appropriate Development Methodology

Characteristic	Waterfall	V-Model	Iterative	System Proto	Throwaway Proto	Agile
Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent
Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Poor
Complex System	Good	Good	Good	Poor	Excellent	Poor
Reliable System	Good	Excellent	Good	Poor	Excellent	Good
Short Schedule	Poor	Poor	Excellent	Excellent	Good	Excellent

Enrichment: Object-Oriented (OO) Basics

Bridging system design and implementation.



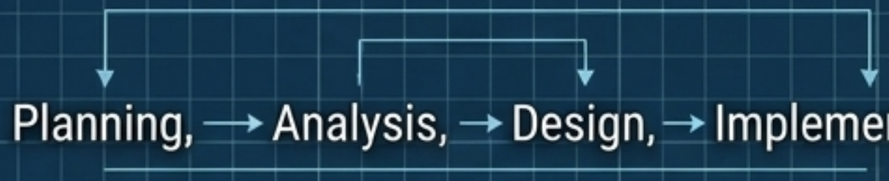
Class:	A blueprint or template for creating objects (e.g., Class Student).
Object:	An instance of a class (e.g., Jane, a specific student).
Attribute:	The data or properties of a class (e.g., student_mark).
Method:	The behavior or functions of a class (e.g., markLab()).
Relationship:	How classes interact with one another (e.g., Inheritance, Association).

Synthesis: From Idea to System Success

1

The Architecture

SDLC provides the strict phases: Planning, Analysis, Design,



2

The Gatekeeper

Feasibility Analysis ensures Technical, Economic, and Organizational viability.

3

The Playbook

Methodologies dictate execution.

- Need reliability? → V-Model
- Unfamiliar Tech? → Throwaway Prototyping
- Unclear Specs? → Agile / Prototyping

ULTIMATE TAKEAWAY:

MASTER THE WHY, UNDERSTAND THE HOW, AND CHOOSE THE RIGHT PLAYBOOK.