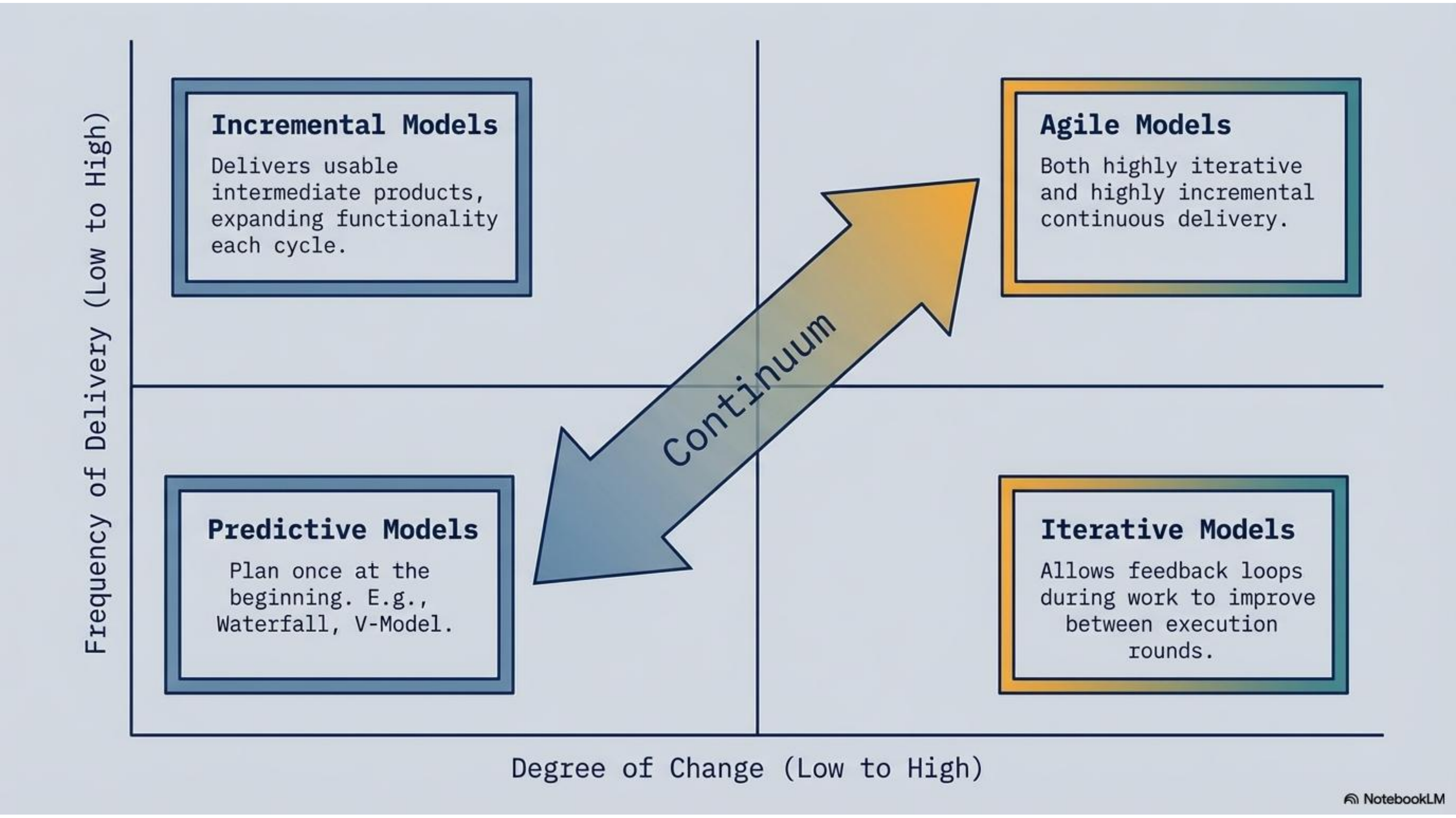
Noti Haim

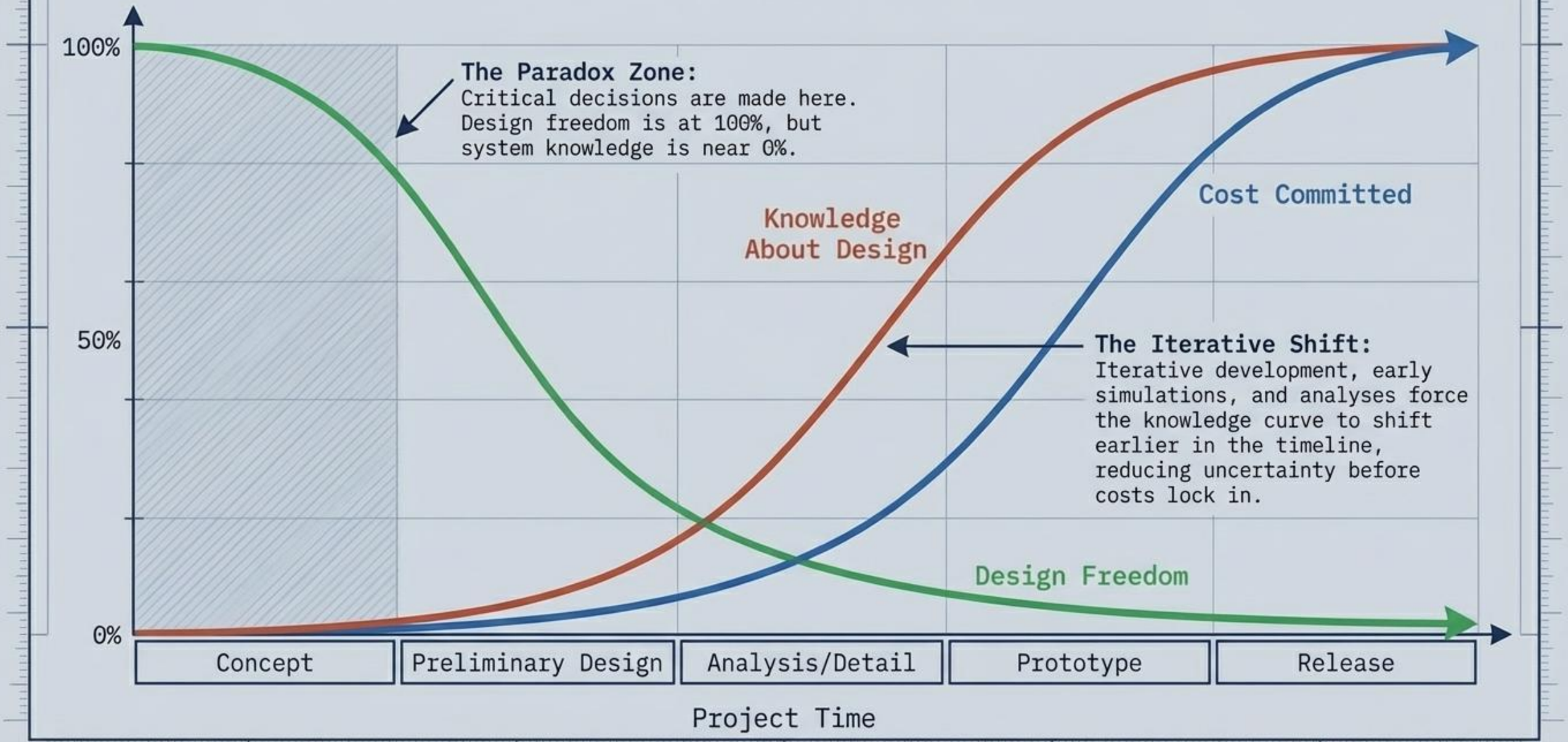
Navigating SDLC Methodologies

A Comparative Guide to System Development Life Cycle Models

REQUIREMENTS | DESIGN | TESTING | IMPLEMENTATION


Incremental Models
Delivers usable intermediate products, expanding functionality each cycle.
Agile Models
Both highly iterative and highly incremental continuous delivery.
Continuum
Predictive Models
Plan once at the beginning. E.g., Waterfall, V-Model.
Iterative Models
Allows feedback loops during work to improve between execution rounds.
Frequency of Delivery (Low to High)
Degree of Change (Low to High)

The Core Challenge: The Knowledge Paradox



Evaluation Framework

Standardized criteria for comparing SDLC methodologies.

Requirements Approach

How the model handles initial specifications and adapting to requirement changes.

Design & Implementation

The frequency, phasing, and overlap of the actual build process.

Testing Strategy

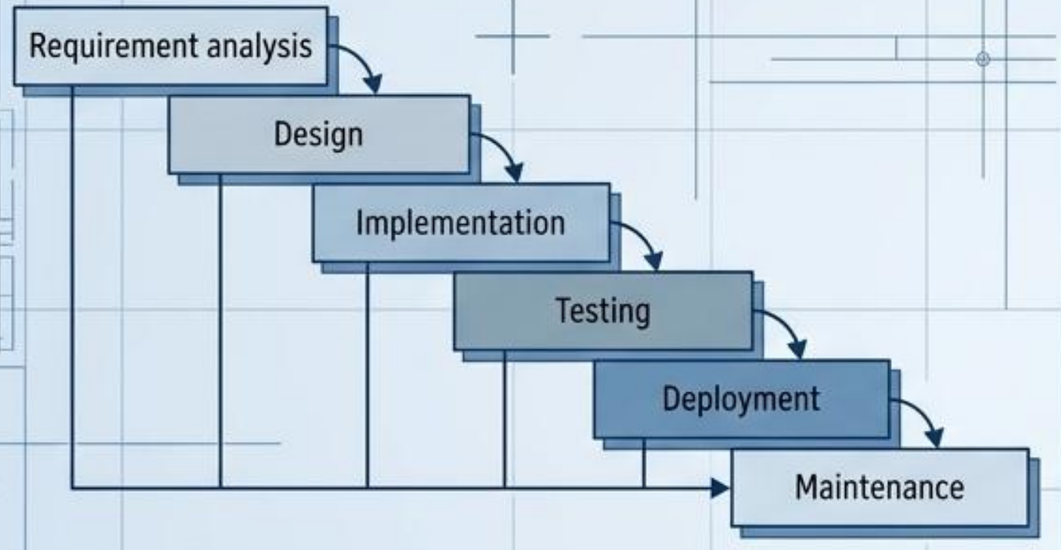
When and how validation occurs—whether in a single phase, multiple levels, or continuously.

Ideal Use Case

The specific project constraints, risk levels, and clarity prerequisites suited for the model.

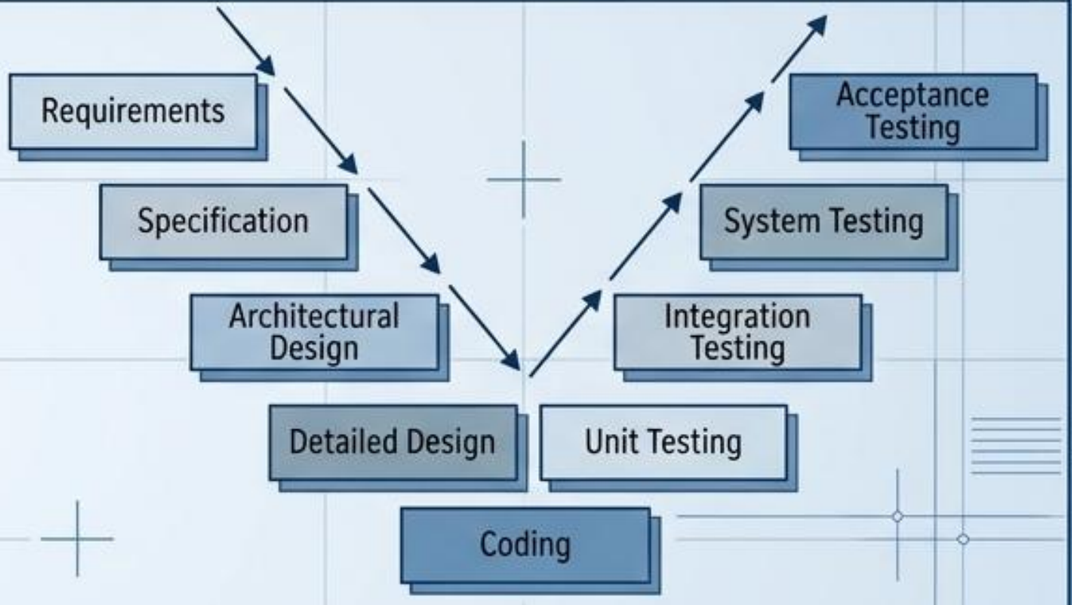
Matrix 1: Predictive & Single-Delivery Models

Waterfall Model



DELIVERY:	"Single final product. Activity triggers sequentially."
REQUIREMENTS:	"1 Phase."
DESIGN/IMPL:	"1 Phase."
TESTING:	"1 Phase."
WHEN TO USE:	"Small projects, crystal-clear requirements, low risk. Inflexible and largely outdated for complex systems."

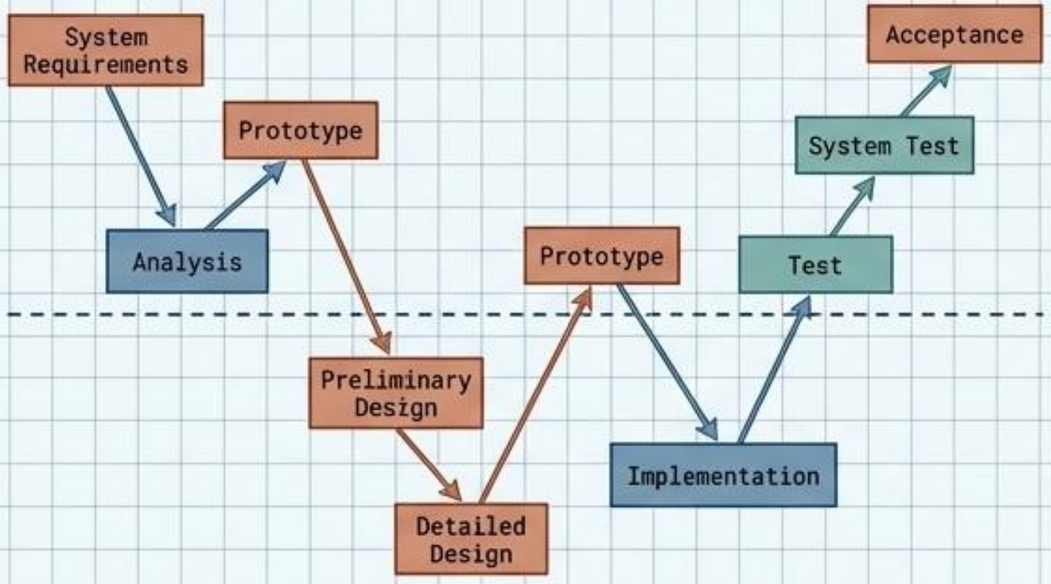
V-Model



DELIVERY:	"Single final product. Working system appears late."
REQUIREMENTS:	"1 Phase."
DESIGN/IMPL:	"1 Phase."
TESTING:	"Conducted across multiple levels (Unit, Integration, System, Acceptance)."
WHEN TO USE:	"Low risk and clear requirements. Note: This is the standard baseline model for Rafael projects. Highly sensitive to changing requirements."

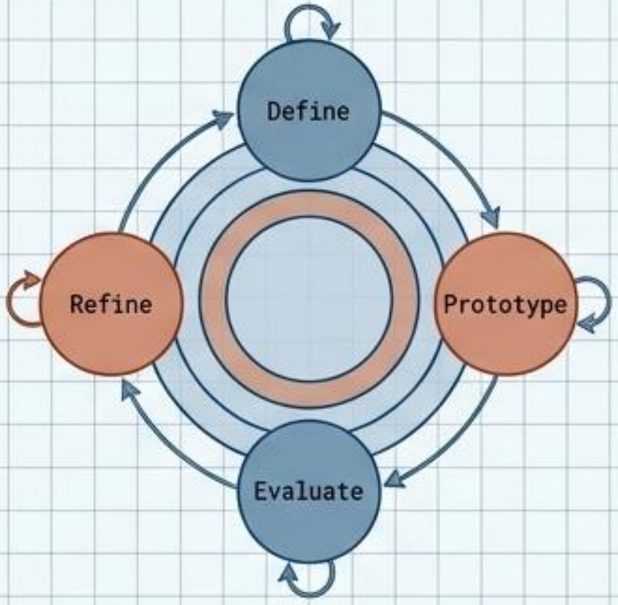
Matrix 2: Prototyping & Risk-Reduction Models

Sawtooth Prototyping Model



APPROACH:	Early reduction of main risks through prototypes of growing complexity.
REQUIREMENTS:	1 Phase.
DESIGN/IMPL:	1 Phase (sometimes requires more design cycles).
TESTING:	1 Phase (at multiple levels, sometimes more).
WHEN TO USE:	Projects with specific identified risks. Warning: Temporary prototype designs may inadvertently become the final product if not properly managed.

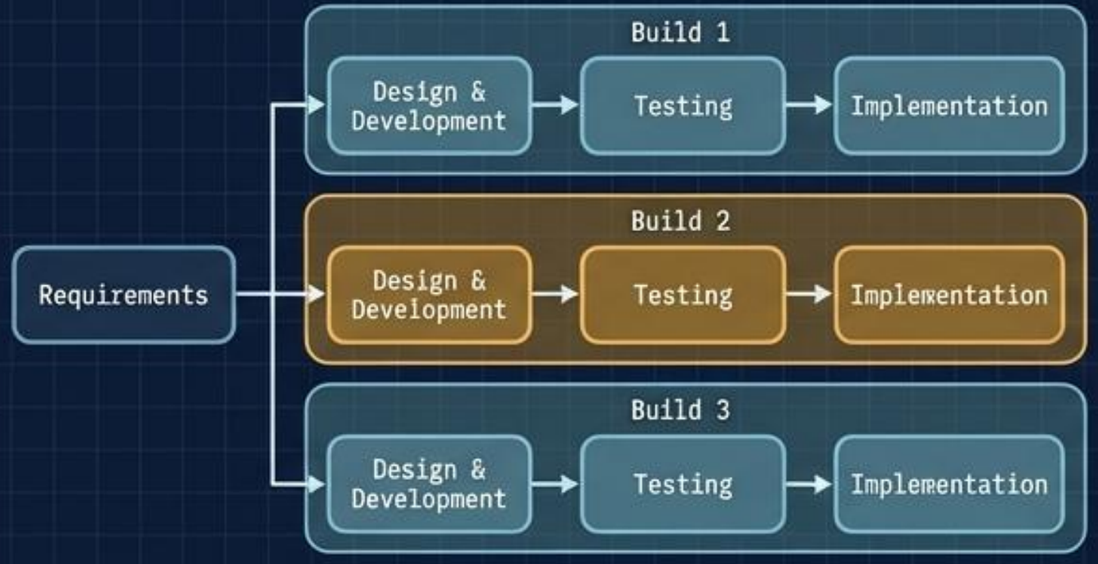
Evolutionary Prototyping Model



APPROACH:	Tests critical functions early. Highly involves the customer to refine unclear needs.
REQUIREMENTS:	1 Phase.
DESIGN/IMPL:	1 Main Phase + smaller design packages for iterations.
TESTING:	Conducted inside every single iteration.
WHEN TO USE:	High-risk projects where customer requirements are not finalized or fully understood.

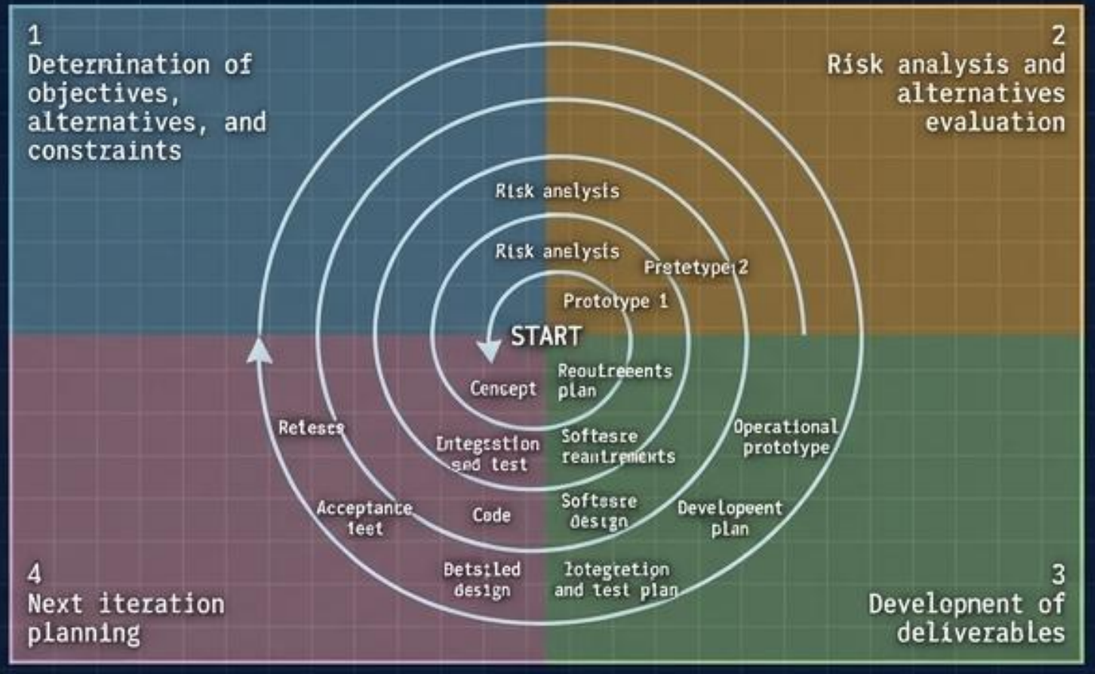
Matrix 3: Parallel & Phased Delivery Models

Incremental Development



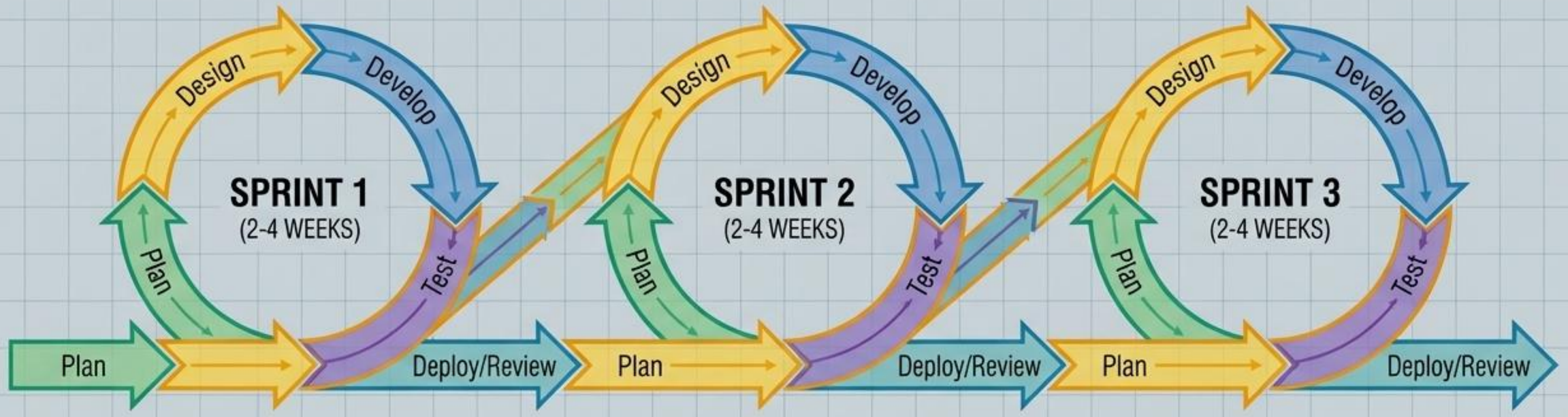
APPROACH:	Develops distinct technologies or functionalities in parallel without overlap.
REQUIREMENTS:	1 Phase.
DESIGN/IMPL:	Multiple parallel activities.
TESTING:	Multiple parallel activities.
WHEN TO USE:	When Time-to-Market (TTM) is critical and system architecture supports high modularity (e.g., developing day and night cameras simultaneously).

Spiral Model



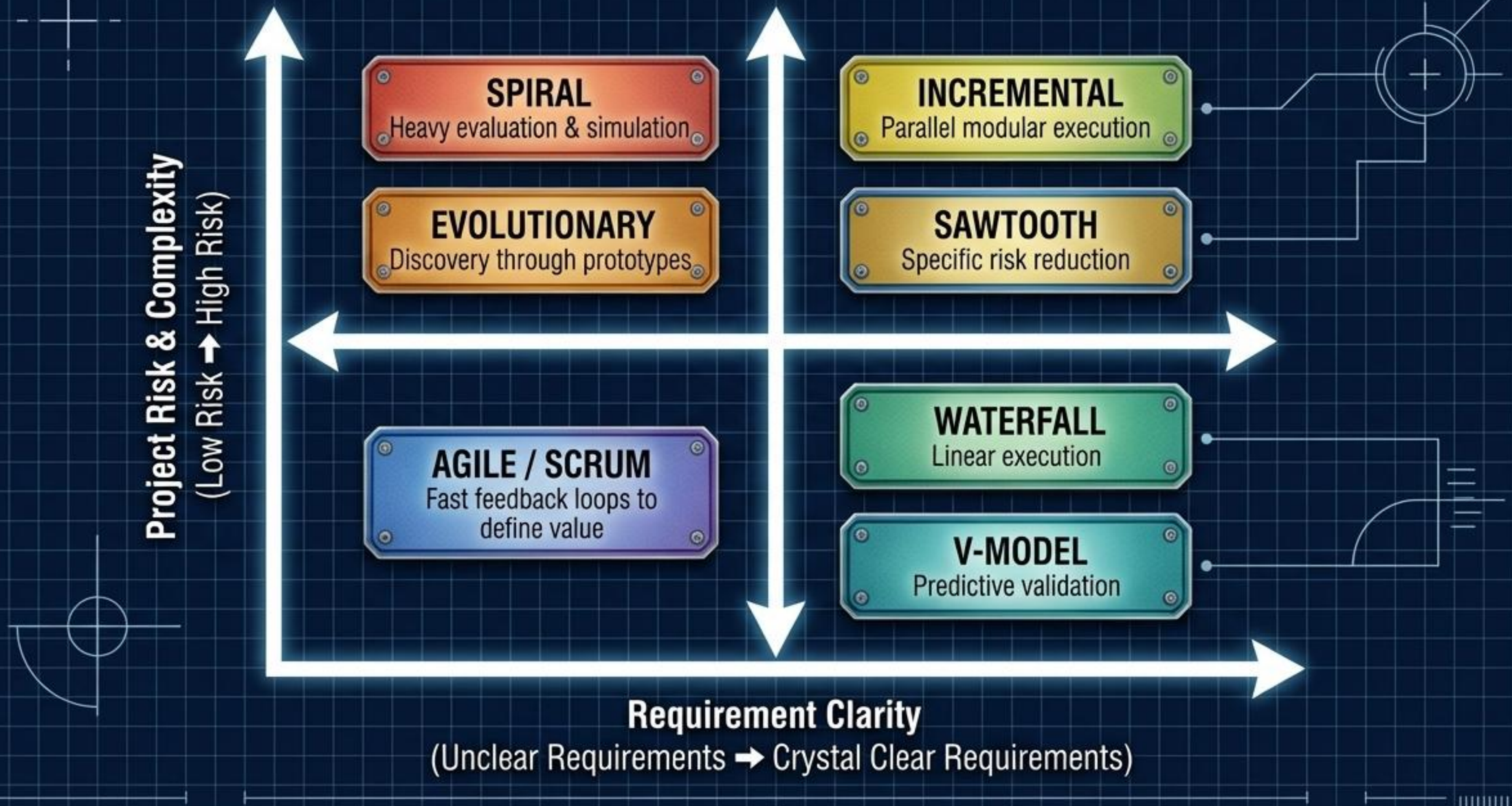
APPROACH:	Each cycle includes full requirements, heavy risk analysis, simulation, build, and test phases.
REQ / DESIGN / TEST:	Multiple occurrences, completely dependent on the number of iterations.
WHEN TO USE:	Startups, large projects divisible into smaller components, or when business/technical baselines are entirely unclear.

Matrix 4: High-Frequency Agile Frameworks



APPROACH:	Divides development into highly structured, time-boxed Sprints (2-4 weeks). Focuses on continuous delivery of working value.
REQUIREMENTS:	1 overarching phase, but highly adaptable and easily changed throughout.
DESIGN & TESTING:	Multiple actions for small scopes, with frequent overlaps.
WHEN TO USE:	Projects requiring intense customer communication and early working deliverables. Requires highly flexible system architecture.
KEY CONSTRAINT:	Demands strict iteration management. If the customer lacks focus, the project can easily drift in the wrong direction.

Master Diagnostic: Selecting by Risk & Clarity



Trade-offs: The Price of Agility

The Value

Roboto Mono

1. Requirements easily pivot based on market reality.
2. Continuous customer feedback loop.
3. Early and consistent value realization.

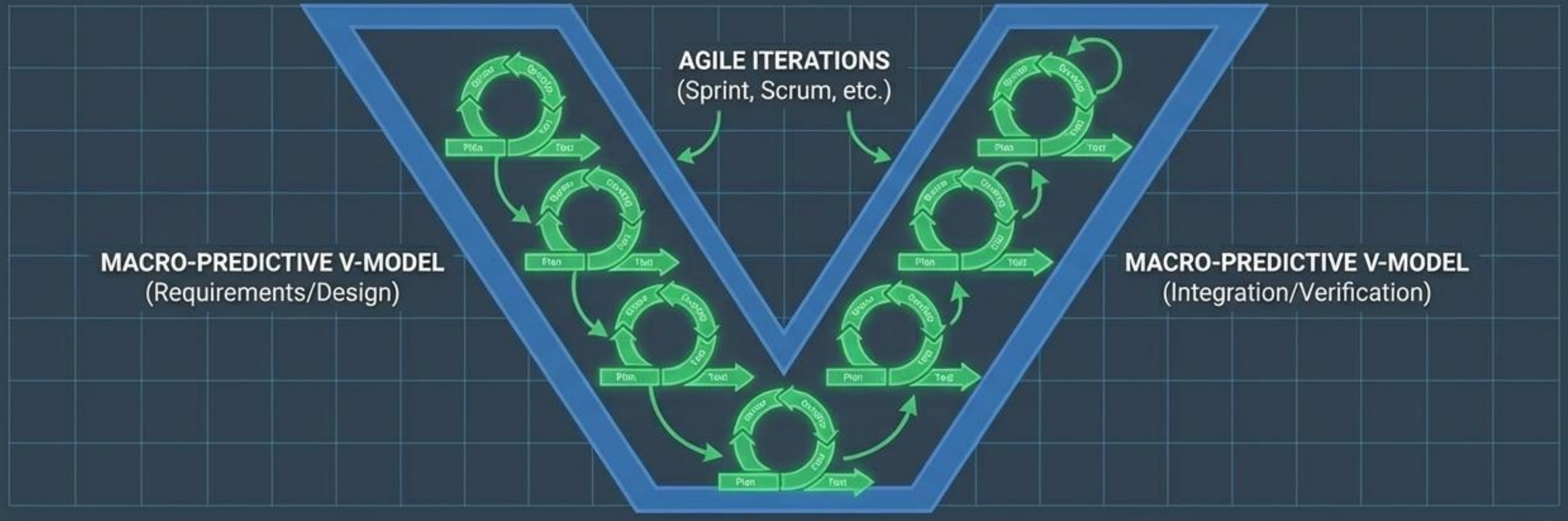
The Cost

Roboto Mono

1. Requires intense, continuous customer availability (if the customer lacks focus, the project drifts).
2. Difficult to implement in heavy hardware or physical systems engineering.
3. Requires masterful iteration management; poor Scrum execution leads to architectural chaos.

Synthesis: Functionality over Time

Real-World Application: The Hybrid Reality



Choosing the wrong model is expensive. But choosing only one model is often impossible.

Application Example: Complex defense or aerospace systems frequently use a rigid V-Model for the overarching hardware and physical systems integration, while running Agile/Scrum internally for the software and digital interface components. Structure dictates outcomes.

Architecting for Uncertainty: Core Principles

01

Respect the Paradox.

Acknowledge that initial knowledge is low. Do not lock in massive decisions early unless requirements are absolute.

02

Match Model to Risk.

Use Predictive models for clarity, Prototyping for risk, Incremental for speed, and Agile for adaptation.

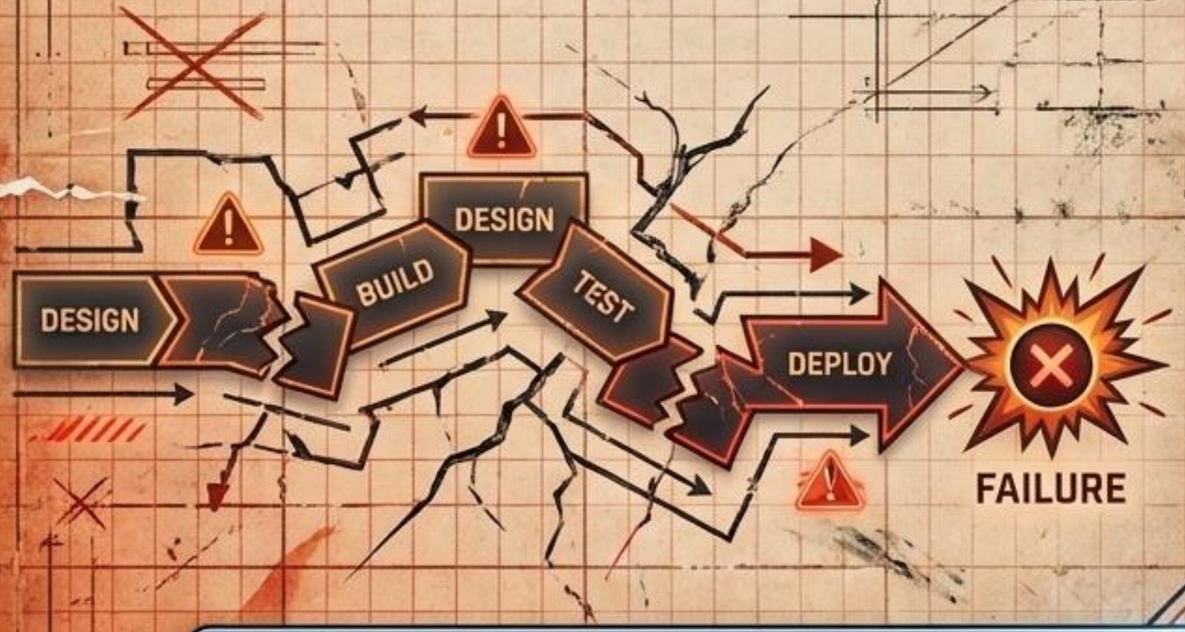
03

Structure Dictates Outcome.

The SDLC is not administrative overhead; it is the first and most critical engineering decision of your project. Adjust it to fit reality.

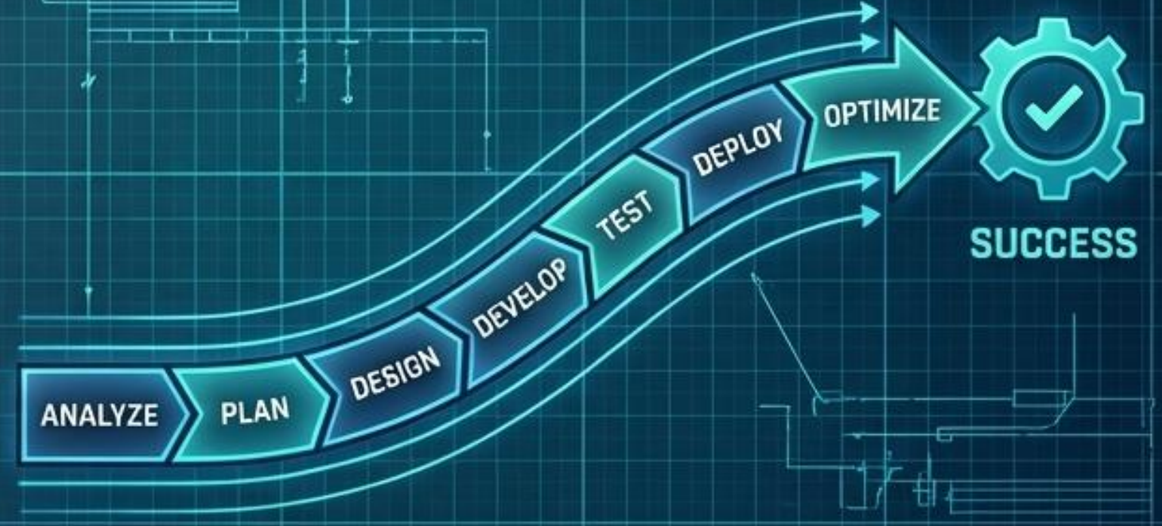
THE WARNING

SELECTING THE WRONG SDLC MODEL IS ONE OF THE MOST EXPENSIVE AND IRRECOVERABLE MISTAKES A PROJECT CAN MAKE. PURE METHODOLOGIES RARELY SURVIVE CONTACT WITH REALITY WITHOUT ADAPTATION.



THE DIRECTIVE

INVEST TIME IN STRATEGIC PLANNING. ANALYZE YOUR RISK, MANDATE, AND REQUIREMENT CLARITY TO DELIBERATELY SELECT, COMBINE, AND TAILOR YOUR DEVELOPMENT MODEL BEFORE EXECUTION BEGINS.



TRUE PROJECT SUCCESS REQUIRES MATCHING THE ARCHITECTURAL BLUEPRINT TO THE OPERATIONAL REALITY.