SOFTWARE DEVELOPMENT PROCESSES

DR. VIVEK NALLUR

VIVEK.NALLUR@SCSS.TCD.IE

OUTLINE OF THIS TALK

- Software Development Processes
  - Waterfall
  - V-Model
  - Incremental
  - Prototyping
  - Spiral

WHAT IS A SOFTWARE DEVELOPMENT PROCESS?

Also known as Software Lifecycle, the development process is a structure imposed on the development of a software product/project.

ACTIVITIES

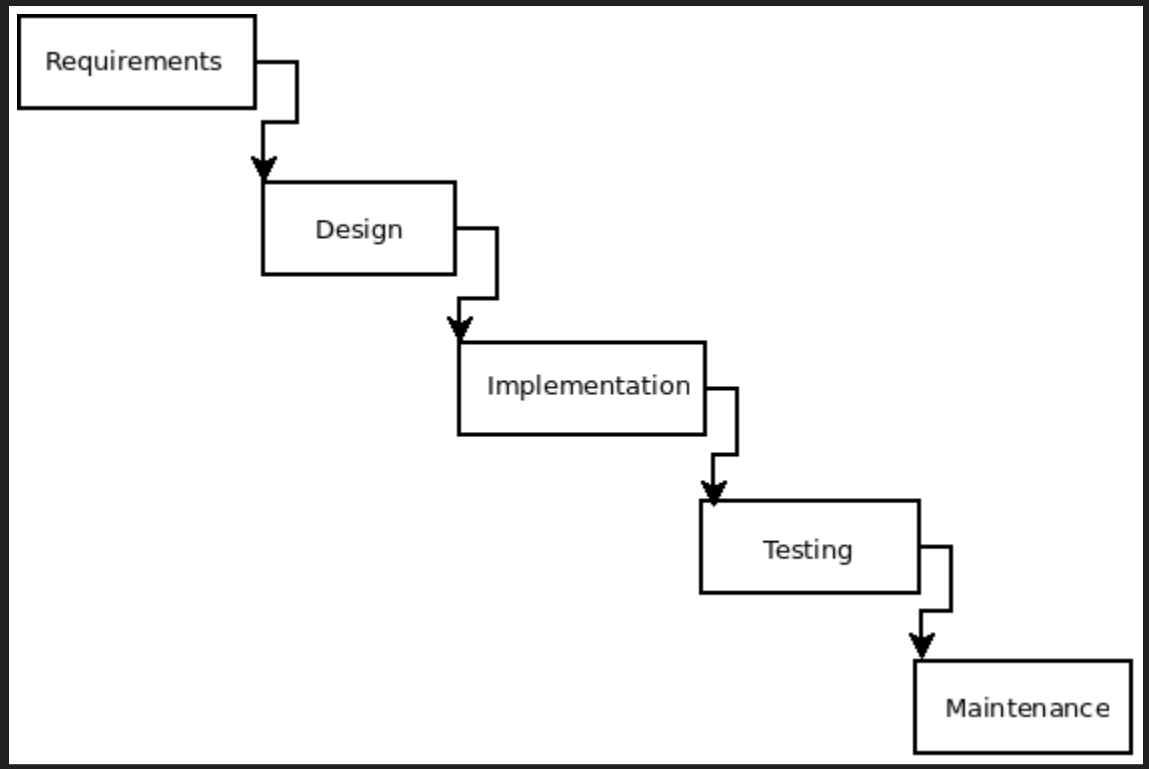
Requirements	Extracting and specifying the requirements of the desired software
Architecture	An abstract representation of the structural and communication patterns of the software
Implementation	Translating the design into code
Testing	Verification and validation of the software
Maintenance	Fixing discovered bugs, enhancing the software for new requirements

PROCESS MODELS

Predictable and repeatable ways to sequence the activities, such that software meets expectations in terms of functionality, cost, delivery schedule, etc.

Two major models of development: *linear* and *non-linear*

THE (IN)FAMOUS WATERFALL MODEL



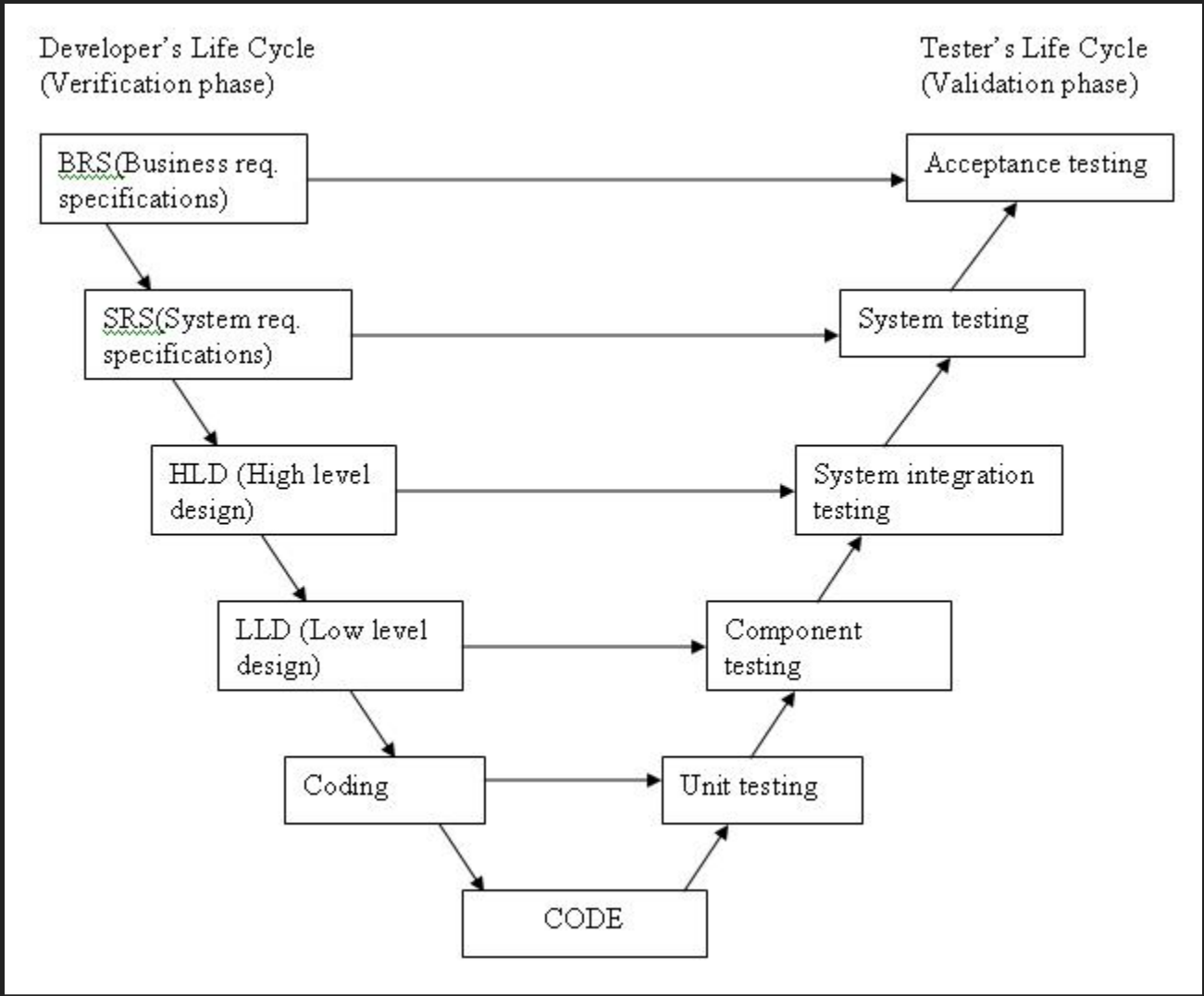
WATERFALL: CHARACTERISTICS

- *Sequential*: Each phase *must* be completed before the next phase can begin
- *Easy*: Ease of understanding, use, and management
- *Well-specified*: Each phase has specific deliverables, which can be reviewed for completeness

WATERFALL: PROS / CONS

Works well when well-specified	Can't go back from Testing to Architecture/Design
Easy to implement	Difficult to deal with ambiguity
Works for short-projects	Long, on-going projects not well-suited

THE V-MODEL


Developer's Life Cycle
(Verification phase)
Tester's Life Cycle
(Validation phase)
BRS(Business req. specifications)
Acceptance testing
SRS(System req. specifications)
System testing
HLD (High level design)
System integration testing
LLD (Low level design)
Component testing
Coding
Unit testing
CODE

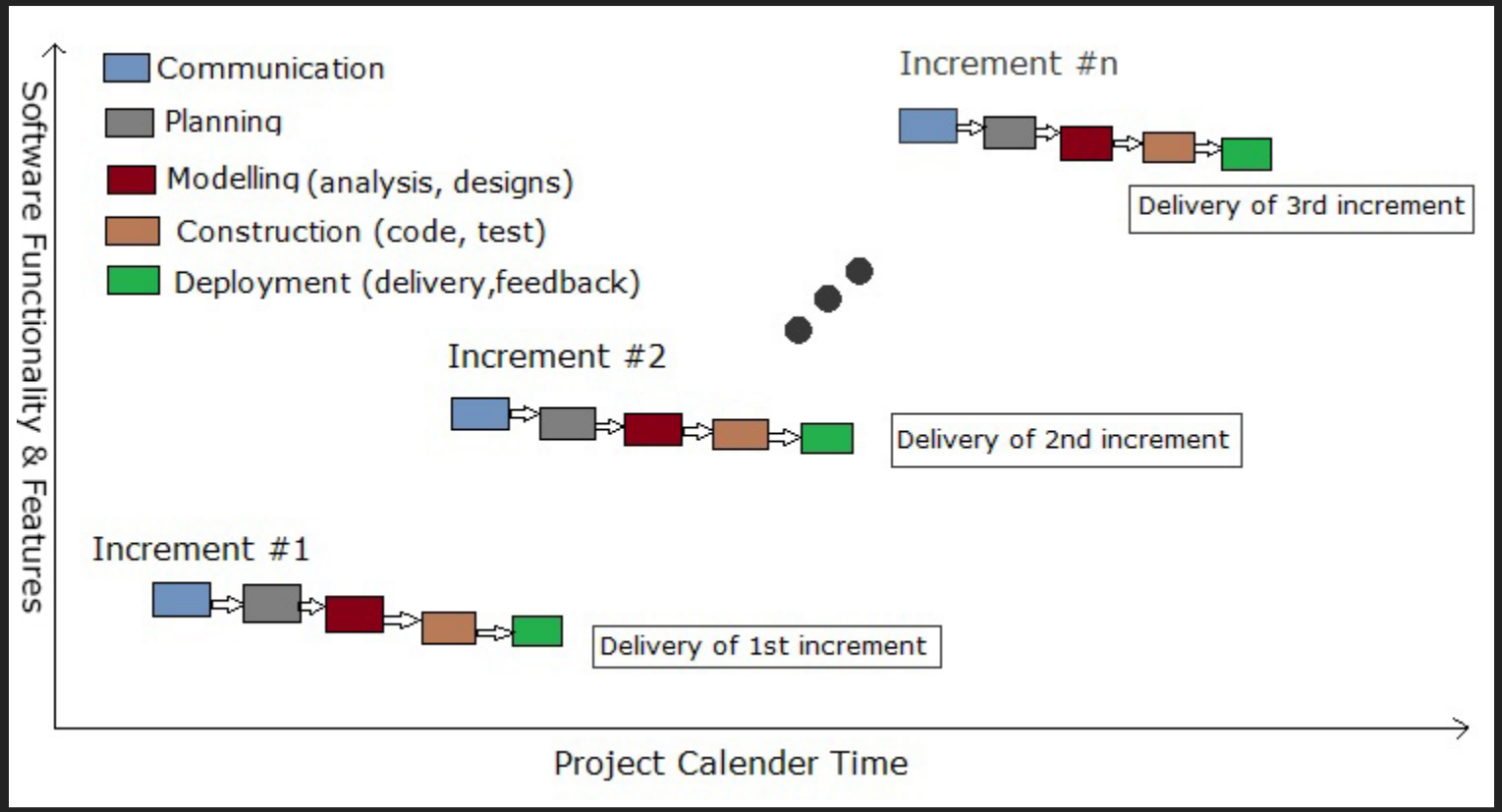
V-MODEL: CHARACTERISTICS

- *Sequential*: (Like the waterfall) a sequence of activities with no loops
- *V&V*: Focusses on verification and validation as the primary feedback mechanism
- *Split between Architecture and Design*: Explicitly separated architecture, its acceptance, and design and its acceptance

V-MODEL: PROS / CONS

Simple and Easy	Fairly rigid
Test Design happens before coding	No prototypes produced
Avoids downward flow of defects	Any change requires re-start of entire model

INCREMENTAL MODEL



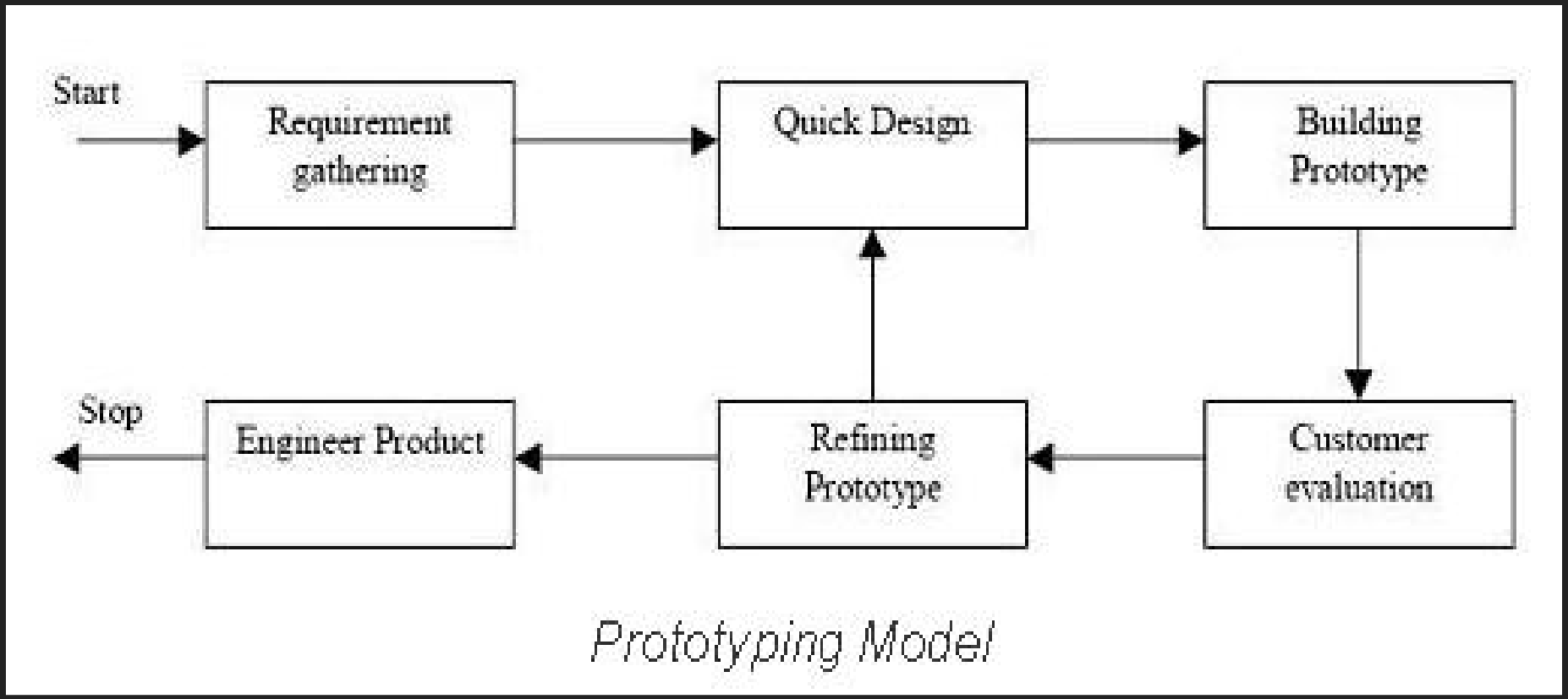
INCREMENTAL MODEL: CHARACTERISTICS

- *Multi-Waterfall*: Effectively reproduces multiple waterfall cycles
- *Working Software Early*: Each build results in working software
- *Multiple Integration Cycles*: Each deployed build must integrate the previous ones

INCREMENTAL MODEL: PROS / CONS

Initial product delivery is fast	Expert planning needed
Easier to test and debug	Cost may exceed Waterfall costs
Changes can be made to product before final delivery	Architecture can be affected badly

PROTOTYPE MODEL



Prototyping Model

PROTOTYPE: CHARACTERISTICS

- *Incomplete versions*: Deliberately incomplete to achieve fast user feedback
- *High ambiguity in domain*: The user cannot articulate exactly what's needed
- *Requires buy-in from client*: Client must agree up-front to use and review incomplete software

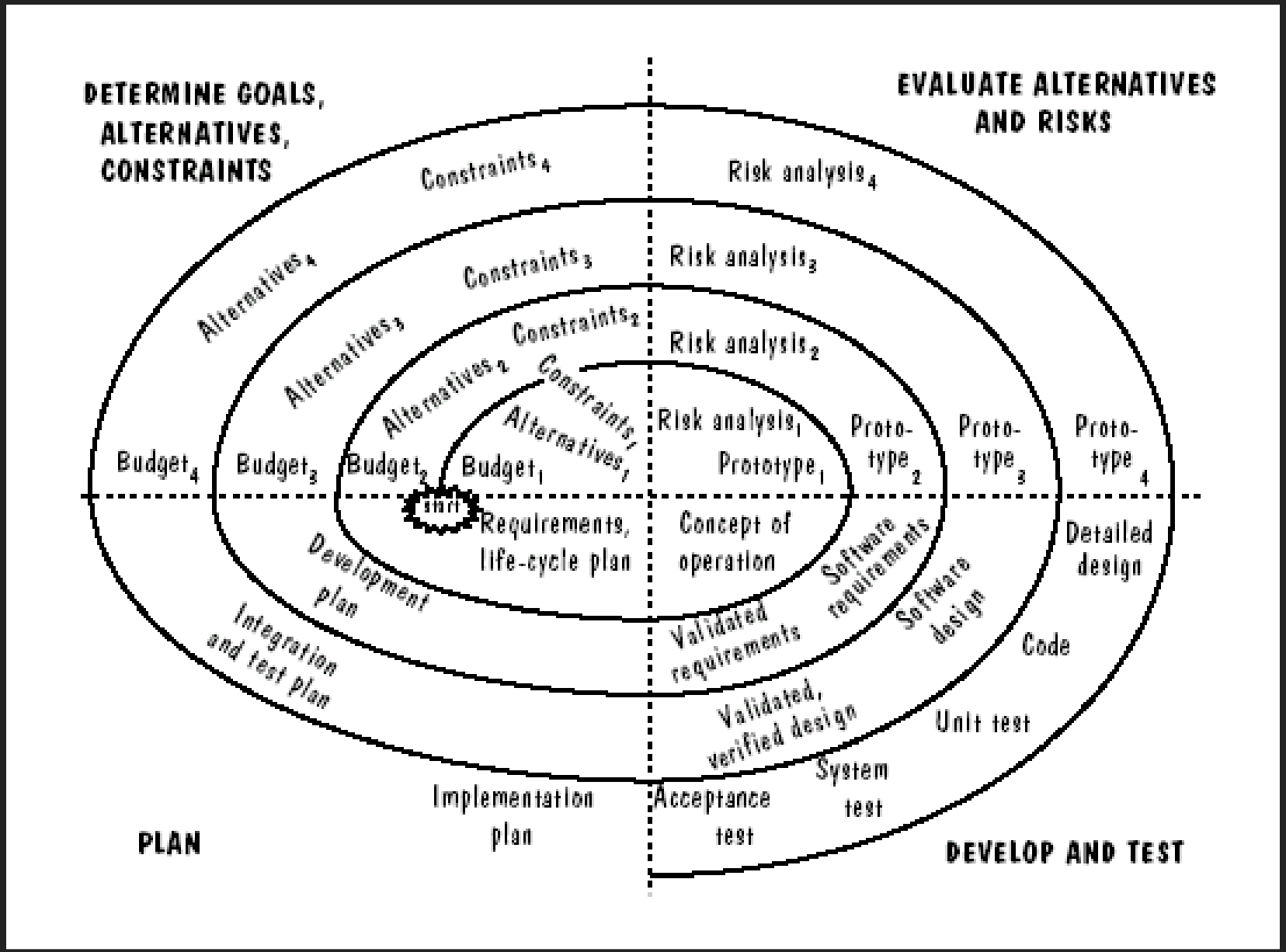
TYPES OF PROTOTYPING

Breadth	Usage
Horizontal	Throwaway
Vertical	Evolutionary

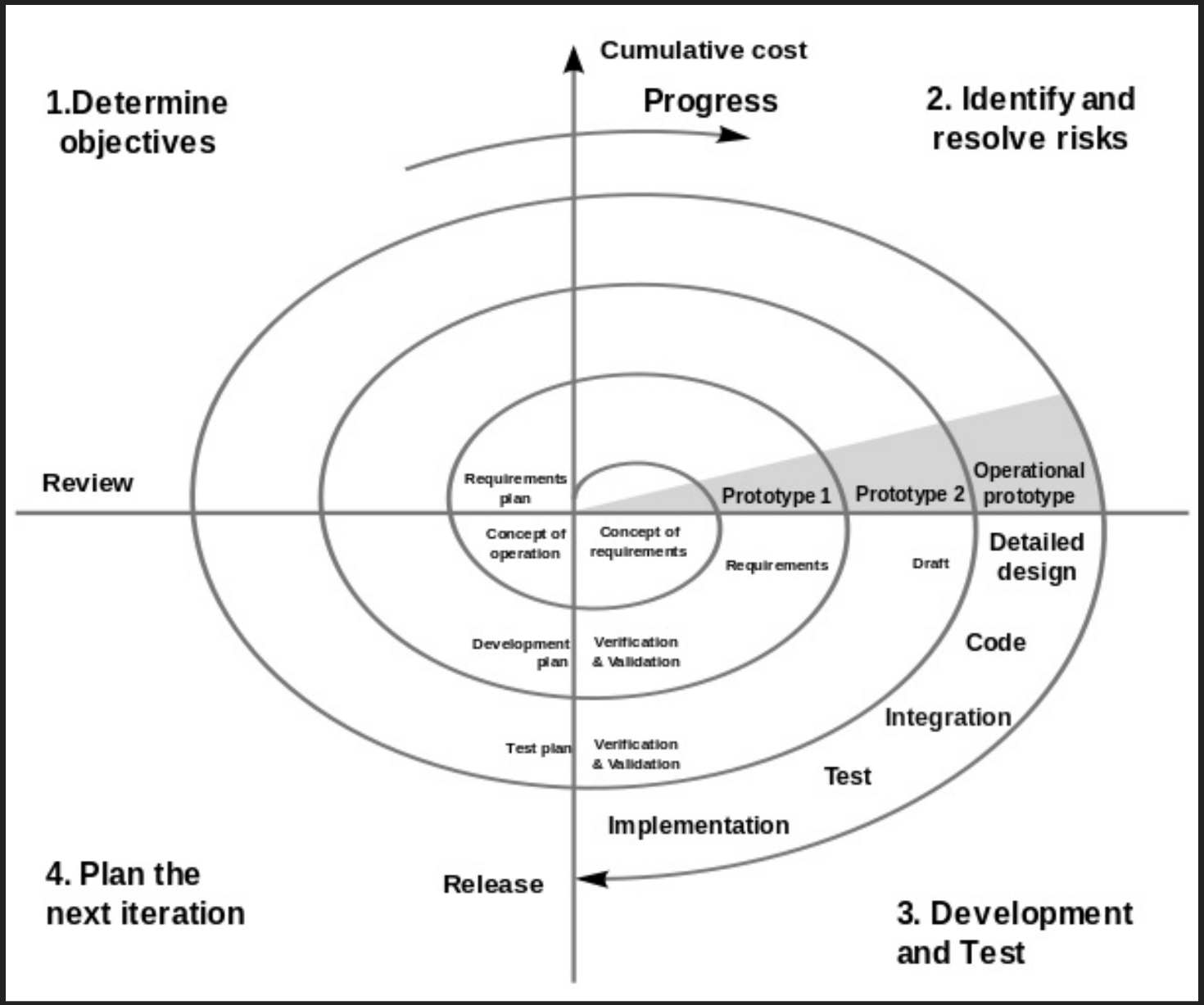
PROTOTYPING: PROS / CONS

Active involvement prevents future disappointment	Scope creep increases complexity
Missing functionality identified easily	Leads to implementing-and-repairing way of building
Errors detected very early	Incomplete or inadequate problem analysis

SPIRAL MODEL



SPIRAL MODEL - CLASSIC



SPIRAL MODEL: CHARACTERISTICS

- *Process Generator*: Any of the linear models can be adopted in any phase
- *Risk-driven*: Every phase is concerned with eliminating the most dominant risks
- *Use of Anchor Points*

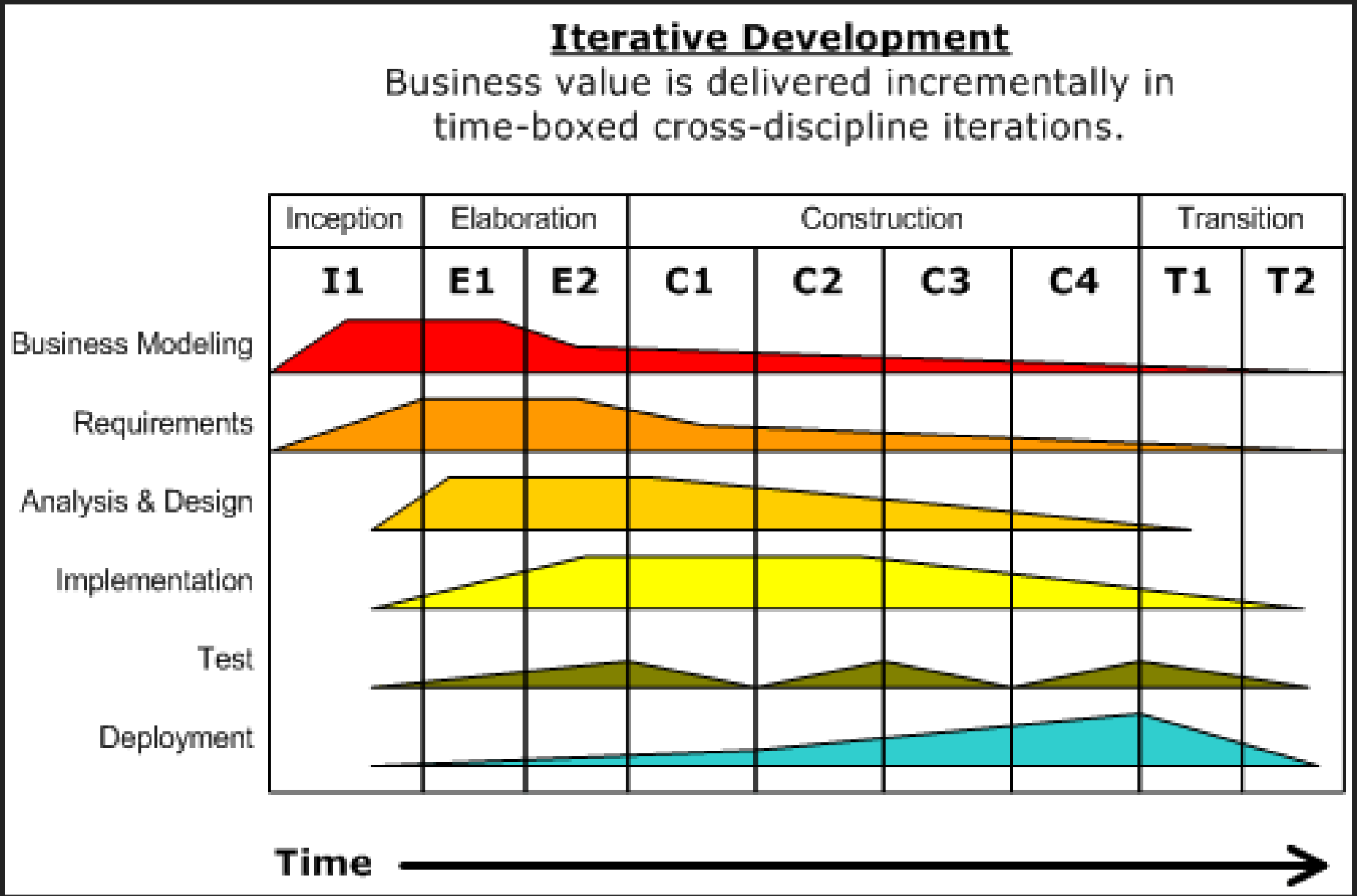
ANCHOR POINTS

- Lifecycle Objectives - Sufficient definition of mgmt and technical approach to achieve everyone's objectives?
- Lifecycle Architecture - Sufficient definition of all risks and mitigation plans created?
- Initial Operational Capability - Do all stakeholders (site, users, operators, maintainers) agree that software meets their objectives?

SPIRAL: PROS / CONS

Active avoidance of risk	Costly model to use
Strong approval and documentation control	Fairly specialized expertise required
Additional functionality can be added later	Only works for large or medium-large projects

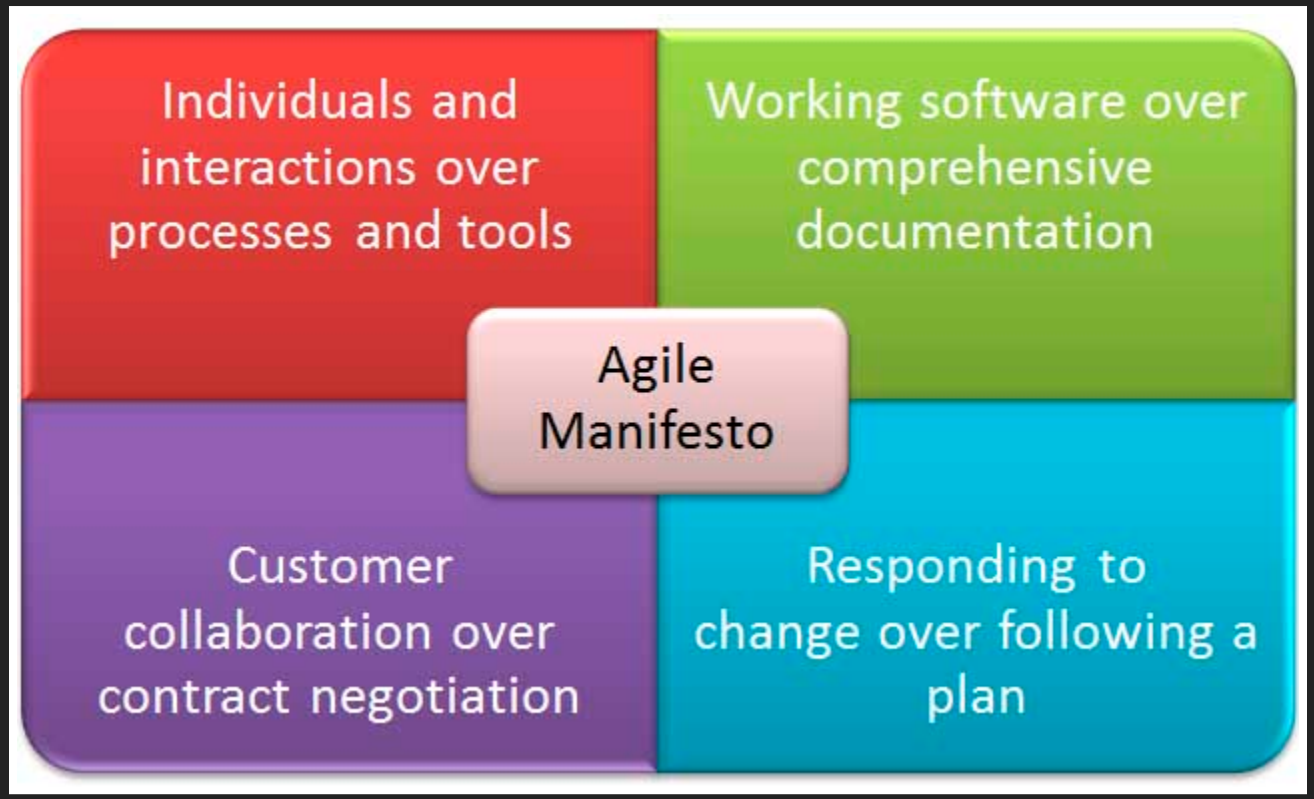
RATIONAL UNIFIED PROCESS



RUP - THE FOUR PHASES

- *Inception*: Scope the system to produce initial costings, estimates, assessment of risks.
- *Elaboration*: Mitigate risk items, perform domain analysis, create executable architecture.
- *Construction*: Building the system components, first external release of software.
- *Transition*: Make system available to, and understood by user. Validation, training, beta-testing.

AGILE MANIFESTO



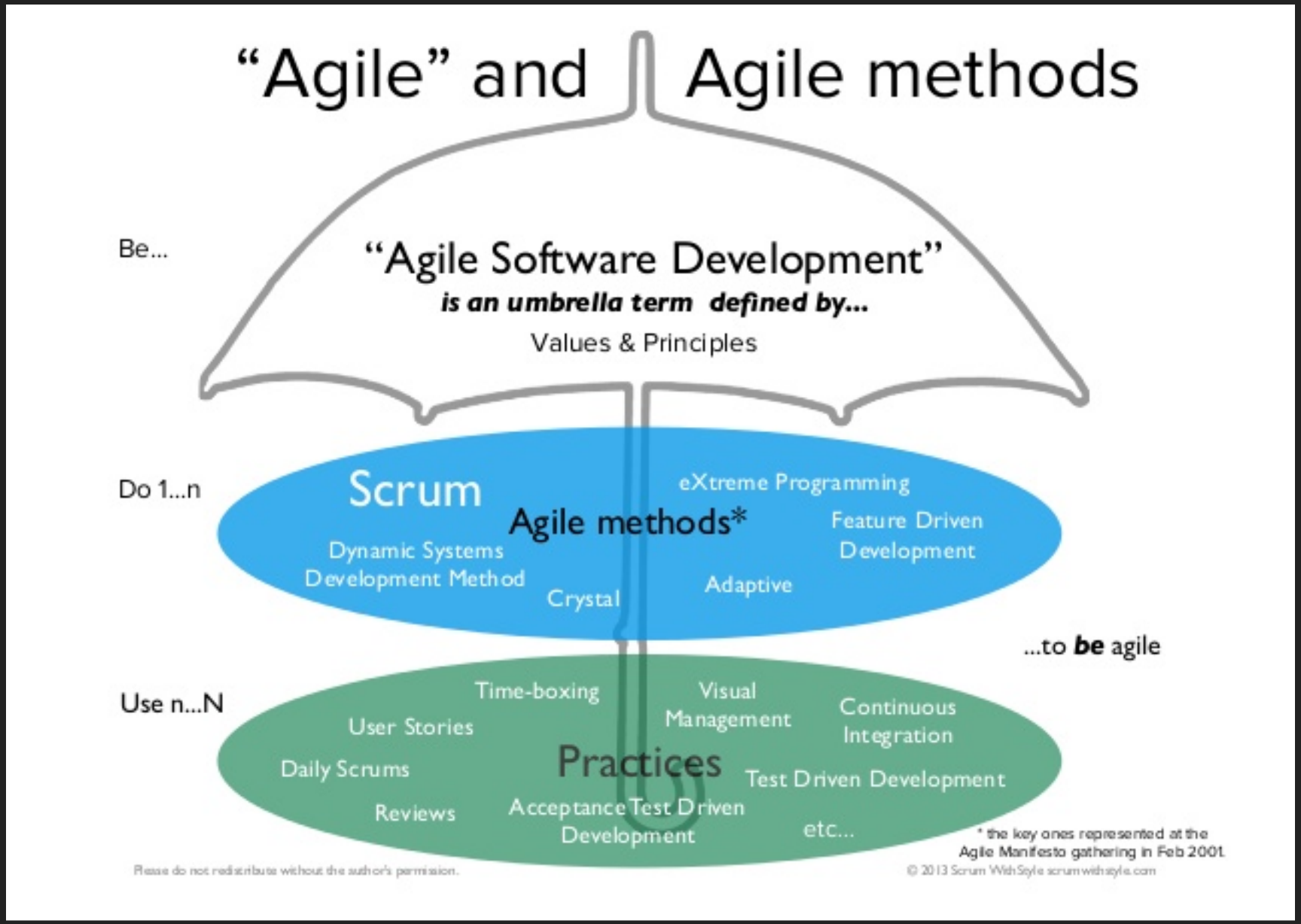
AGILE: CHARACTERISTICS

- Iterative and incremental development
- Self-organizing, cross-functional teams
- Adaptive velocity-based planning
- Repeatable, time-boxed iterations

MAIN METHODOLOGIES

- Scrum
- XP (eXtreme Programming)
- Kanban
- FDD
- TDD, AUP, DSDM, Crystal,

AGILE UMBRELLA



Next class: Scrum & XP

ABOUT THE ASSIGNMENTS

- Since everyone is an expert at estimating their velocity
  - Put theory into practice in a mini-project
  - Use Scrum
- Starts after 'study-week'
  - Team registration deadline: 14-Nov-2016
  - Project selection deadline: 14-Nov-2016

FOR MORE INFORMATION

https://www.scss.tcd.ie/Vivek.Nallur/teaching/

THAT'S ALL, FOLKS!

Questions? Comments?