SSW-555: Agile Methods for Software Development

Introduction

Instructor: Prof. Zhongyuan Yu

School of Systems and Enterprises

Stevens Institute of Technology

This Lecture is based on material from Prof. Jim Rowland

Course Basics

Instructor: Dr. Zhongyuan Yu

Contact Info: zyu7@stevens.edu

Office Hours: Tuesday 11am-12pm available on Zoom or by appointment

Location: Gateway North 204

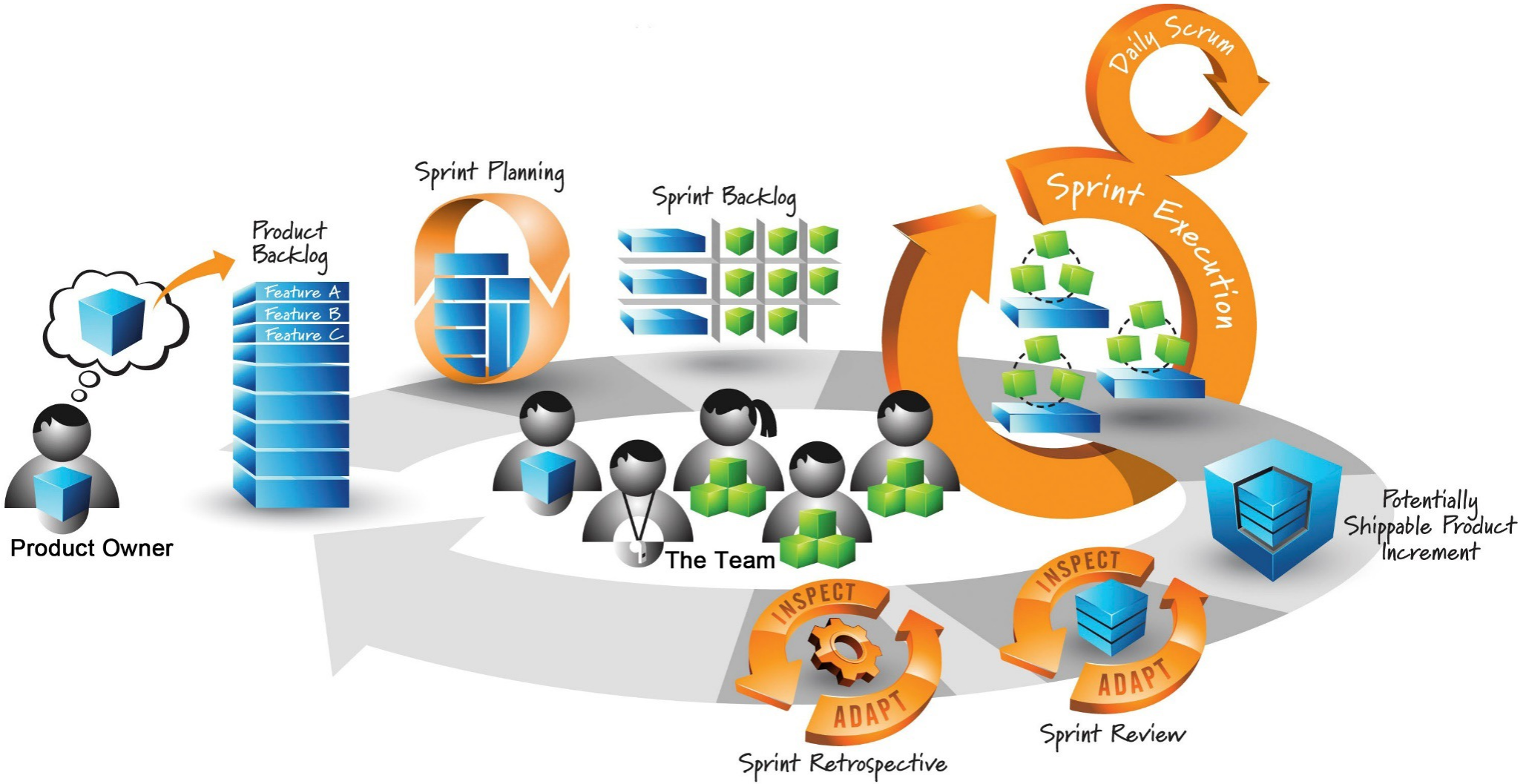
Prerequisite(s): Programming experience, preferably Python

Cross-listed with: CS 555

What this course is about?

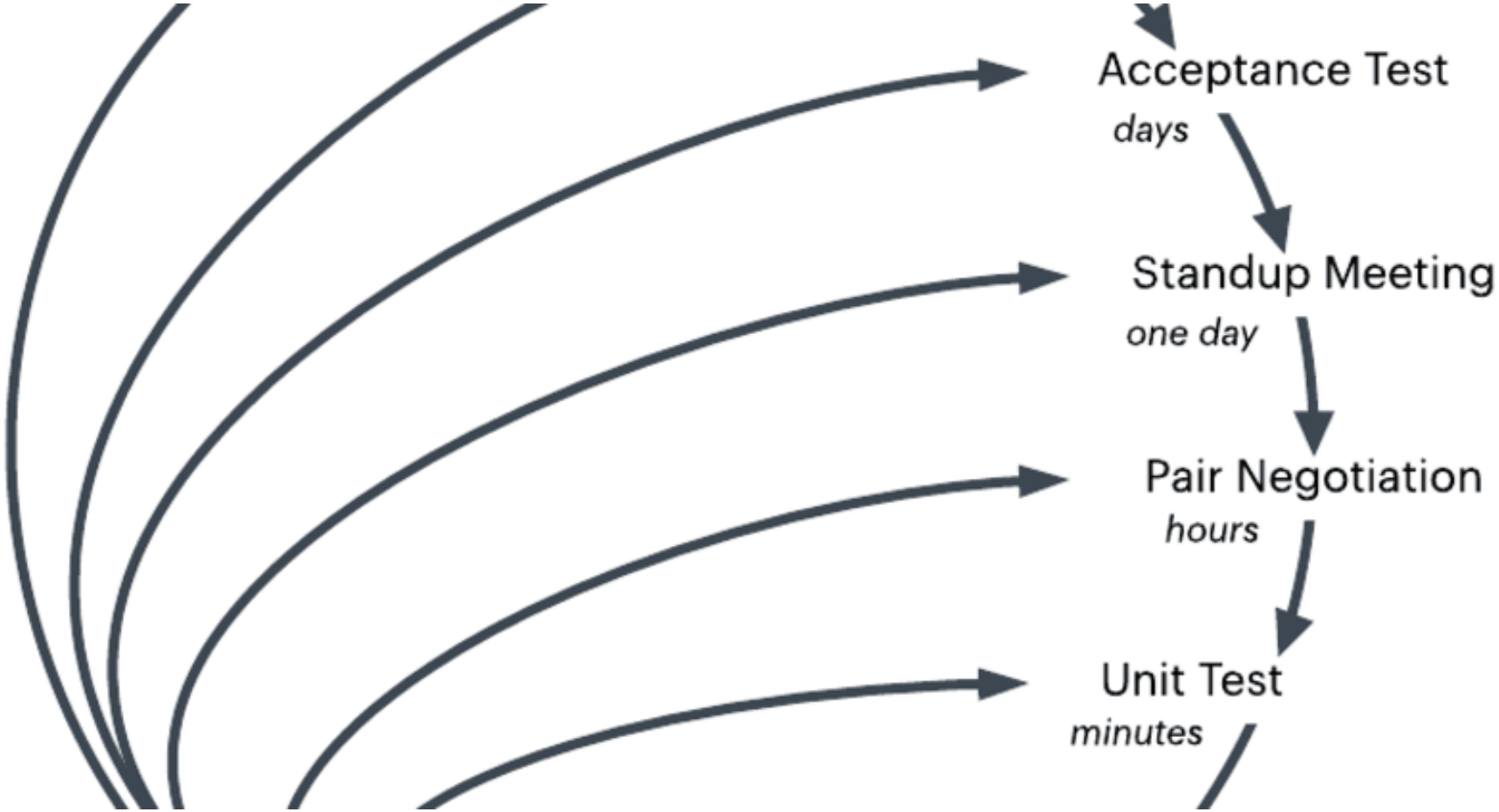
- This course examines agile methods to understand how rapid realization of software occurs most effectively.
- The agile development is contrasted with teams following more traditional methodologies that emphasize planning and documentation.
- Students will learn agile development principles and techniques covering the entire software development process.

Scrum



http://www.innolution.com/essential-scrum/table-of-contents/chapter-2-scrum-framework

Extreme Programming (XP)

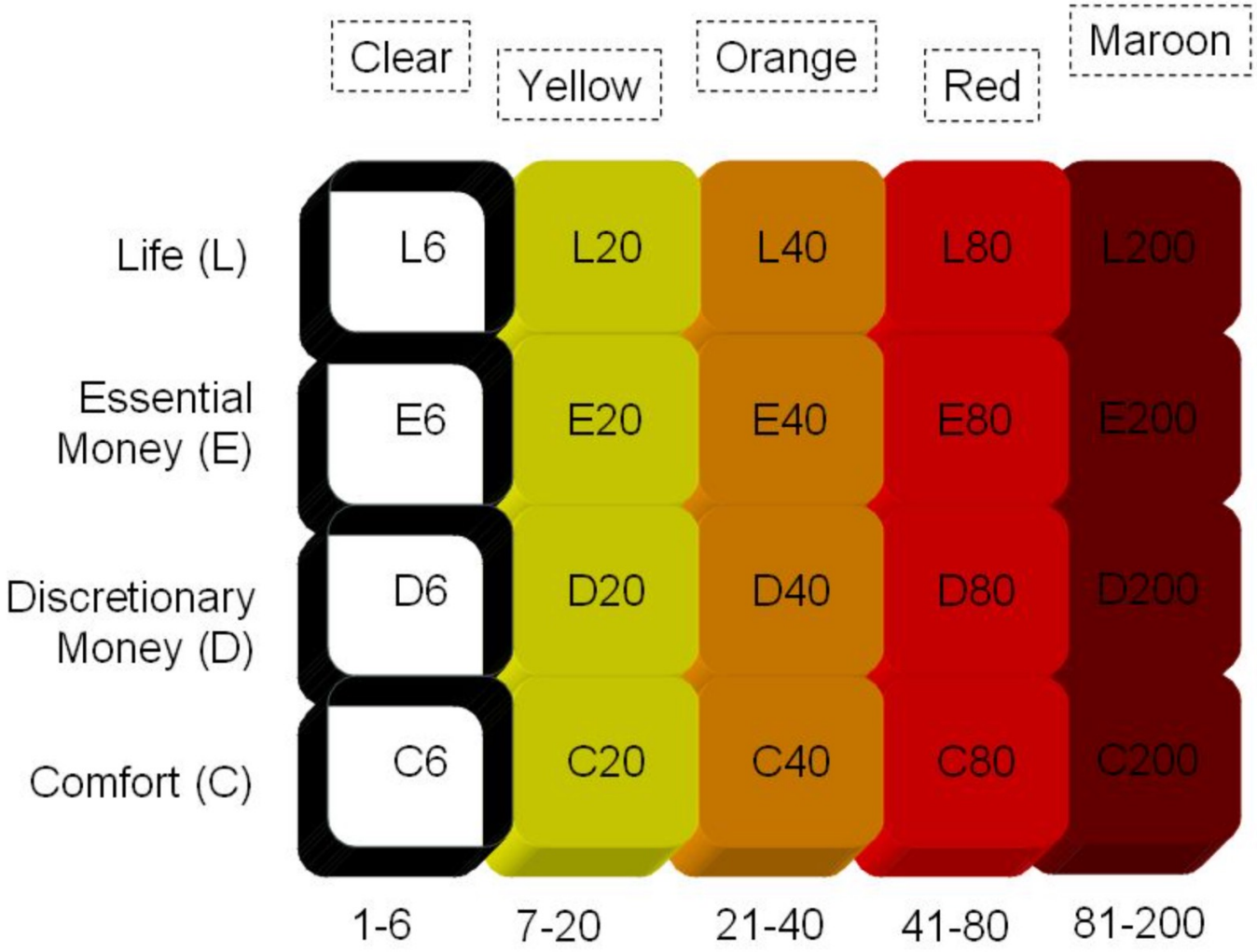


https://dev.to/noeza/reflecting-on-extreme-programming-xp-24pg

Lean

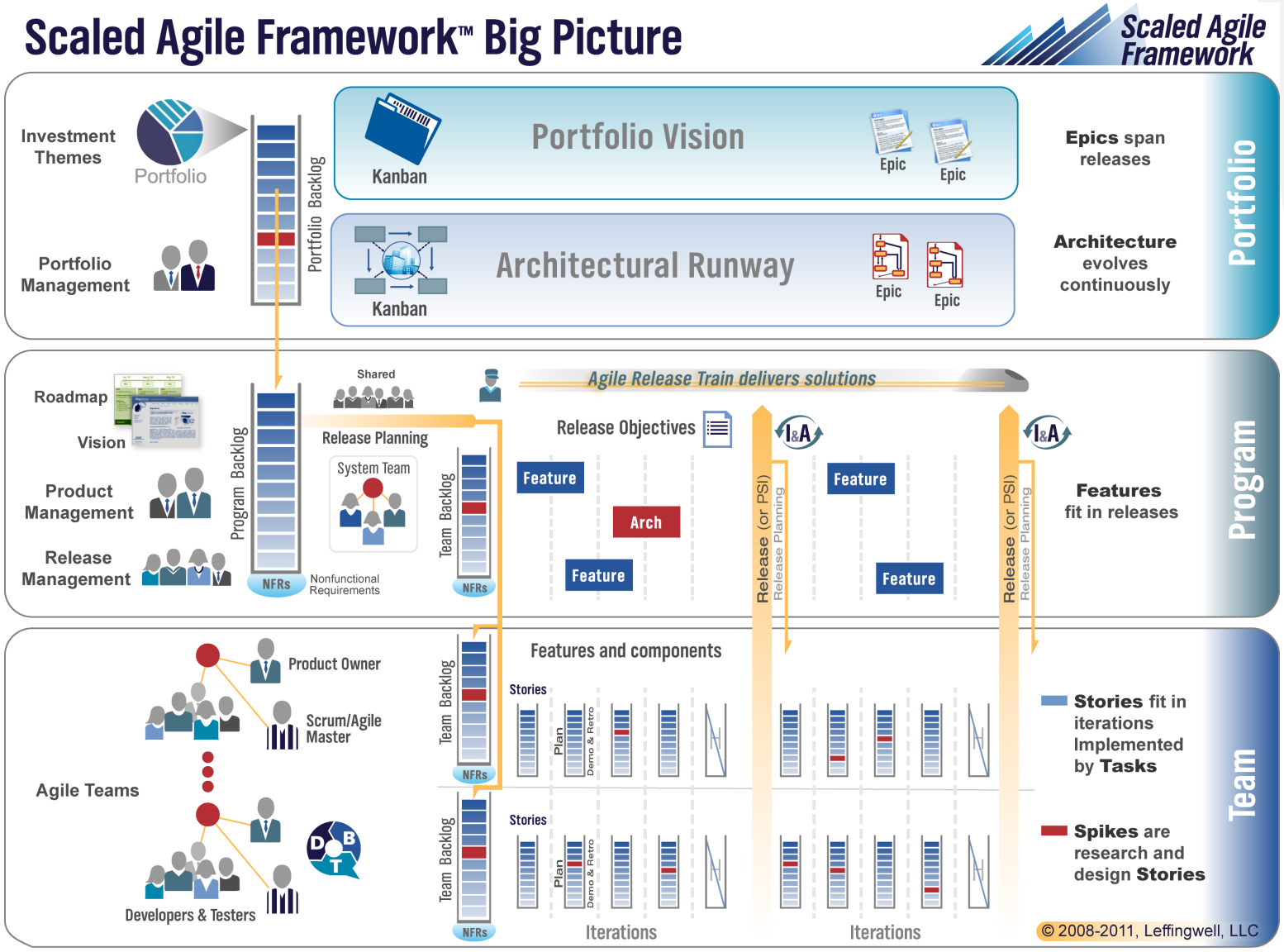
https://kruschecompany.com/lean-software-development/

Crystal



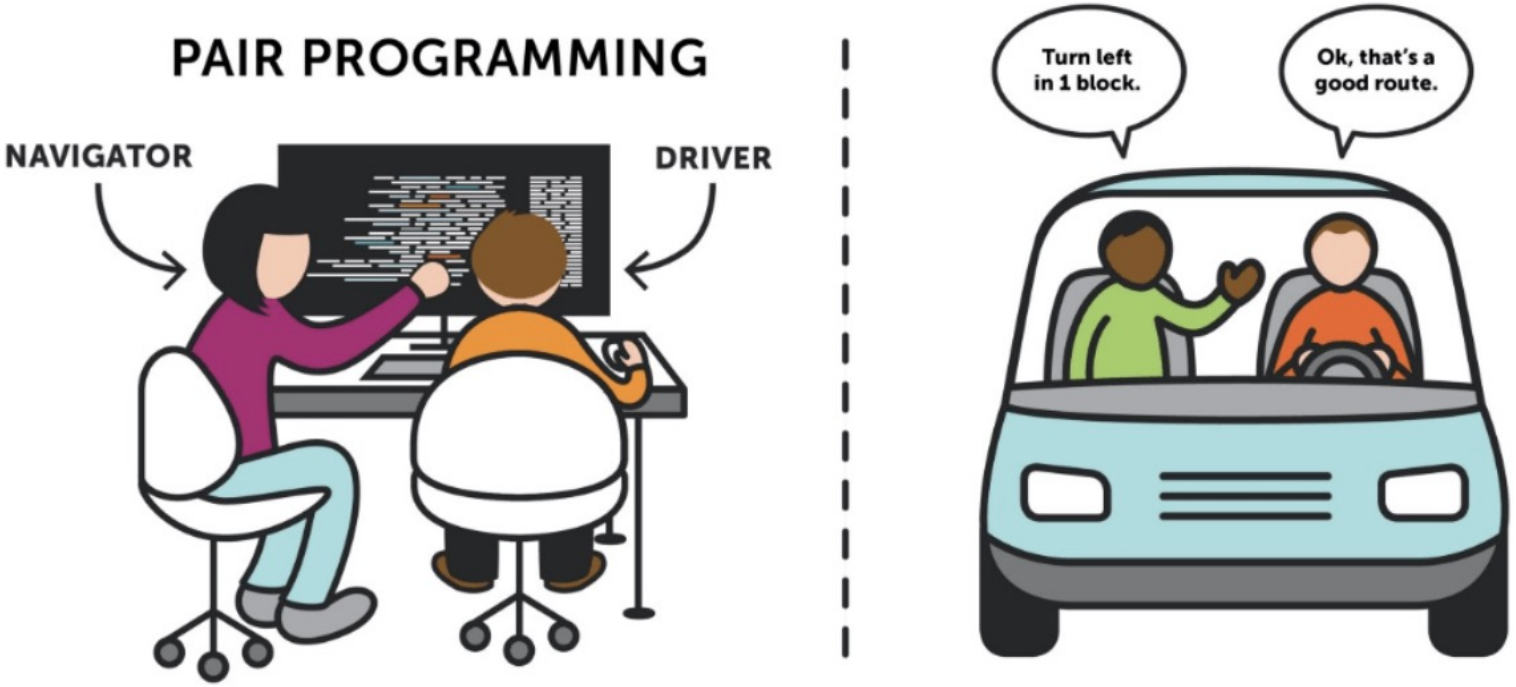
https://en.wikiversity.org/wiki/Crystal_Methods

SAFe



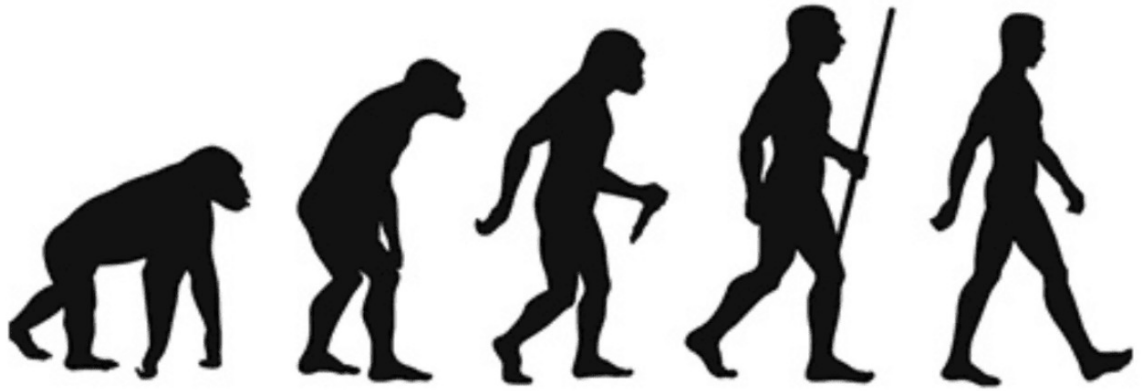
See also www.scalingsoftwareagility.wordpress.com and Leffingwell, D. *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*, Addison-Wesley (Pub. 2011) v 0.8

Pair Programming



https://unruly.co/blog/article/2019/08/27/what-is-pair-programming/

Refactoring



https://lvivity.com/what-is-code-refactoring

Course Schedule (subject to change)

Week	Date	Additional Comments	Assignments Due	Topic	Reading(s)	Project
0	Thursday, January 12, 2023			Orientation		
1	Thursday, January 19, 2023			Course Overview		
	Thursday, January 26, 2023		2/1/23	XP	[Boehm2002][RUP2003]	
2	Thursday, February 2, 2023		2/8/23	User Story	[Ambler 2004][Grenning 2002]	
3	Thursday, February 9, 2023			Project Release		Preparation for project
4	Thursday, February 16, 2023		2/22/23	Scrum	[Schwaber 2013]	Group formation Due
5	Thursday, February 23, 2023		3/1/23	Testing	[Gamma 1998]	Start of Sprint 1
6	Thursday, March 2, 2023		3/8/23	Pair Programming	[Williams 2001]	
7	Thursday, March 9, 2023			Refactoring	[Fowler 1999] Chapter 2 -3	Start of Sprint 2
8	Thursday, March 16, 2023	Spring Recess; No Classes				
9	Thursday, March 23, 2023		3/29/23	Lean	[Poppendieck 2012]	
10	Thursday, March 30, 2023		4/5/23	Crystal	[Wikiversity on Crystal]	Start of Sprint 3
11	Thursday, April 6, 2023		4/12/23	Feature-Driven Development (FDD)	[Palmer 2013]	
12	Thursday, April 13, 2023		4/19/23	DSDM	[Clifton2003]	Start of Sprint 4
13	Thursday, April 20, 2023		4/26/23	Scaled Agile Framework (SAF)	[Leffingwell 2009]	
14	Thursday, April 27, 2023			Zoom Project Presentation		End of Project
15	Thursday, May 4, 2023		5/8/22	Final Exam		

Assignments

- Reading
- Quiz (Due date: 11:59 pm the day before the lecture day, weekly)
- Social Media Discussion (Due date: 11:59 pm the day before the lecture day, total 3 times in a semester)
- Project (Due date: 11:59 pm on the lecture day, bi-weekly)

Readings and Quizzes

Reading assignment each week

- Suggested to be completed before attending lectures
- See Canvas Modules for the weekly reading assignments

Short online quiz covering the reading and lecture material

- To help you retain the material

Be sure to answer all questions in your own words --- don't copy from the lecture notes or other online sources including Open AI.

Social Media

- How many hours do you spend on Social Media each day?
- Reallocate time for enhancing your career
  - New technologies
  - New approaches, tools, and techniques
  - Pulse of the industry
- Total of 3 assignments to share an article of your choice

Team Project

- We will practice Extreme Programming and Scrum on a software project throughout the course
- The project will be released by industry or academic leaders, tentatively scheduled on the third week of the course (check calendar)
- Midterm short feedback report & QA is expected after Sprint 2, and the final presentation is scheduled at the end of the semester (check calendar)
- How project is evaluated:
  - Software content 50%
  - XP and Scrum process 50%
  - Team member evaluation Adjusted (20%)
- All deliverables are team assignments:
  - Only one member of the team needs to submit the deliverable to the Canvas
  - We will use GitHub for Configuration Management
- Sign up for a GitHub account if you don't already have one

Late Policy and Assignment Policy

- All assignments are due at the same time each week if any, so you can plan your schedule

- Late assignments
  - **Less than 24 hours: 5% deduction**
  - **1-3 days: 10% deduction**
  - **4-7 days, 30% deduction**
  - **Greater than a week, you will receive 0 for the corresponding assignment, if without written explanation in advance regarding a situation of unavoidable emergency**

- The lowest quiz grade will be dropped for final homework calculation.

Grading

- Grades will be based on:
  - **Assignments (30%)**
  - **Project and Presentation (40%)**
  - **Final Exam (20%)**
  - **Bi-Weekly Social Media (10%)**
  - **Participation and Bonus points (+)**

- Grades will be posted on Canvas

Cheating

- ***Cheating will NOT be tolerated***
- ALL work is expected to be in your own words
- all quiz answers
- all exam answers
- all programming
- Copying from any source is considered cheating
- providing a reference does not excuse this (do not exceed 20%)
- Consequences of cheating may include:
  - receiving a grade of 0 for an assignment
  - receiving a grade of "F" for the course

Acknowledgements

Lecture material comes from a variety of sources, including:

- https://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm
- Software Engineering, 10th Edition, Ian Sommerville
- Scott W. Ambler www.ambysoft.com/surveys/
- "Get ready for agile methods, with care" by Barry Boehm, *IEEE Computer*, January 2002.
- http://agilemanifesto.org/
- http://www.extremeprogramming.org/
- http://www.tutorialspoint.com/extreme_programming/
- https://www.infoq.com/articles/reifer-agile-study-2017

Today’s Topic

- Software development method comparison
  - Waterfall Model
  - V (Verification/Validation) Model
  - Boehm’s Spiral Model
  - Iterative models
  - Agile Methods
  - Big Bang/Chaos

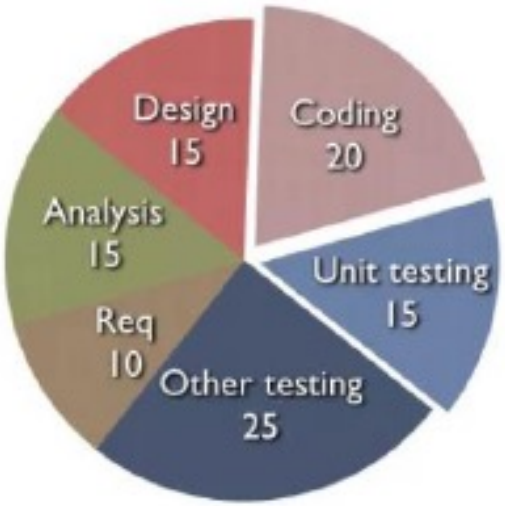
Software Development = Coding ?

Programming (coding) of software is an important part of any software effort.

- 100% effort
- 80% effort
- 50% effort
- 25% effort
- 15% effort

Software Development > Coding

Programming (coding) of software is an important part of any software effort.

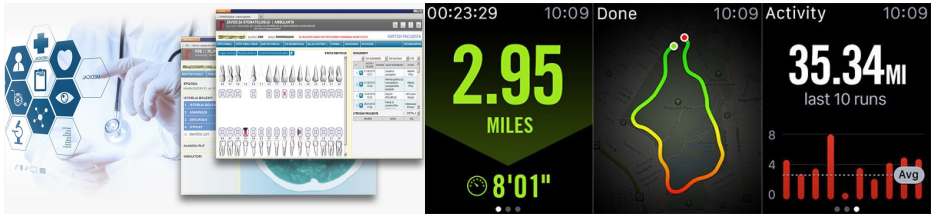


- 100% effort
- 80% effort
- 50% effort
- **25% effort**
- **15% effort**

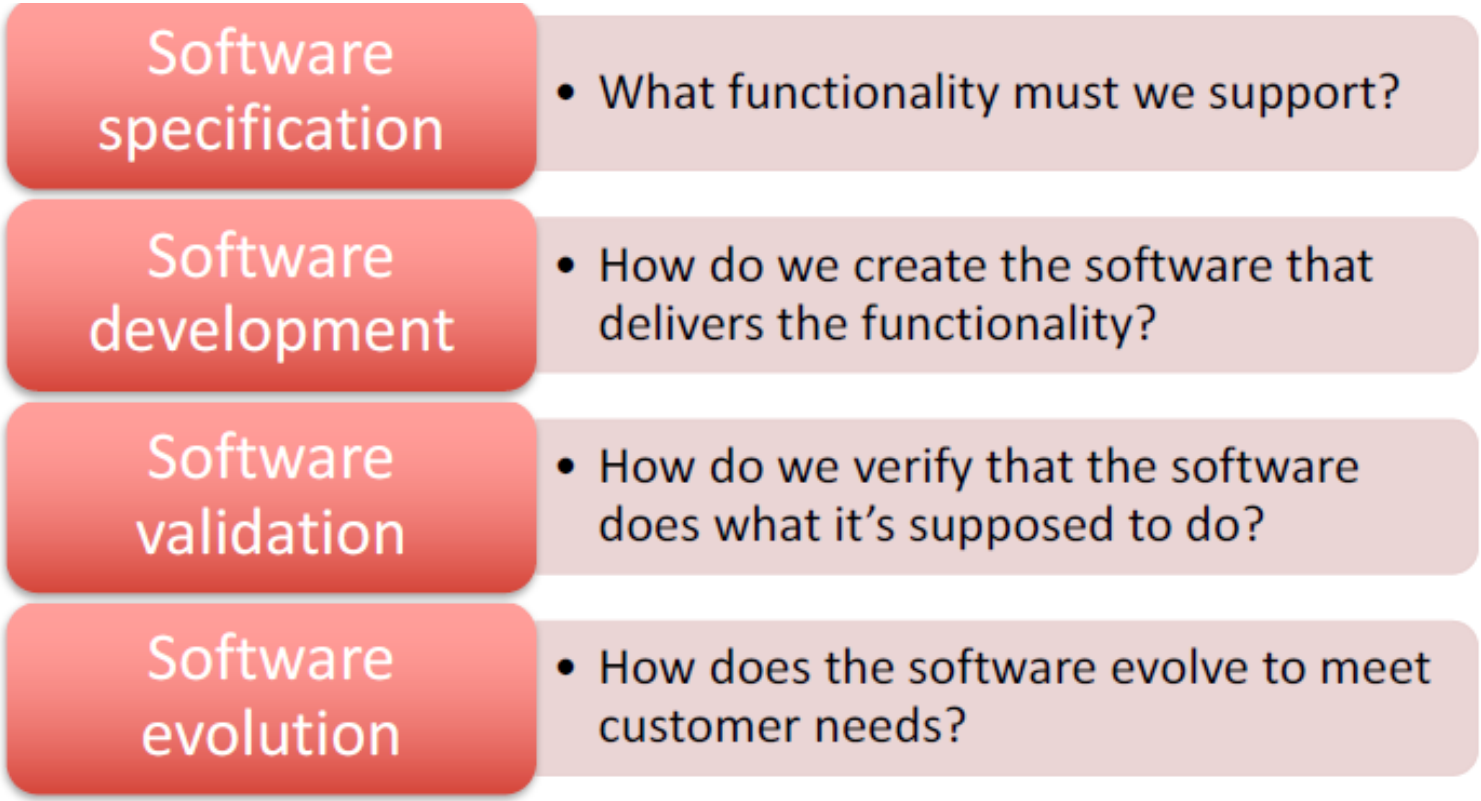
- It is usually less than **25%** of the total effort.
- On large projects, it is less than **15%** of the total effort.

How much planning?

- What's the right level of planning for software projects?
  - It depends on the task!
- How should we decide?
  - What's the domain?
  - How complete are the requirements?
  - How stable are the requirements?
  - What's the cost of doing the wrong thing?
  - What's the cost of doing the right thing too slowly?
  - What are the risks?
  - What are the rewards?

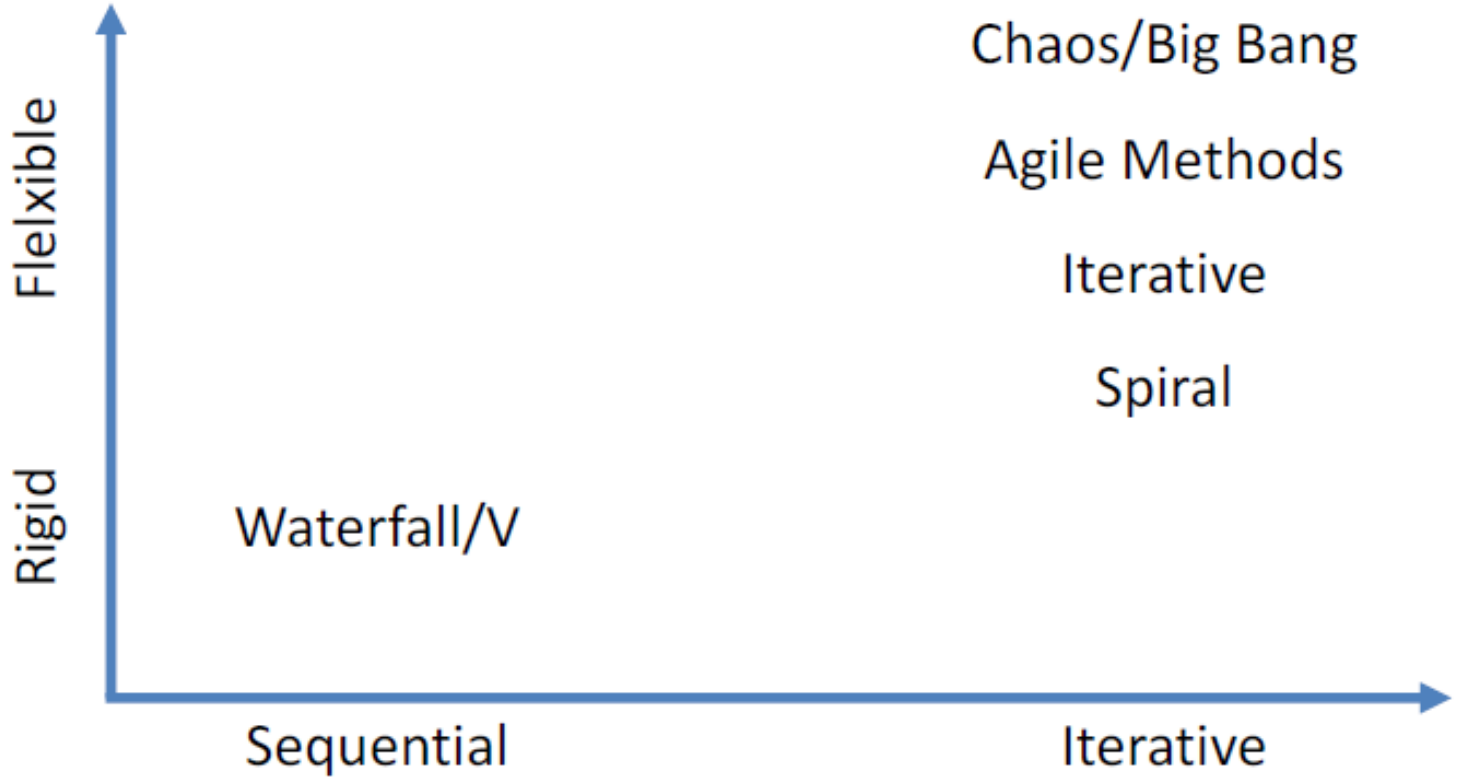


Software Development Life Cycle (SDLC)

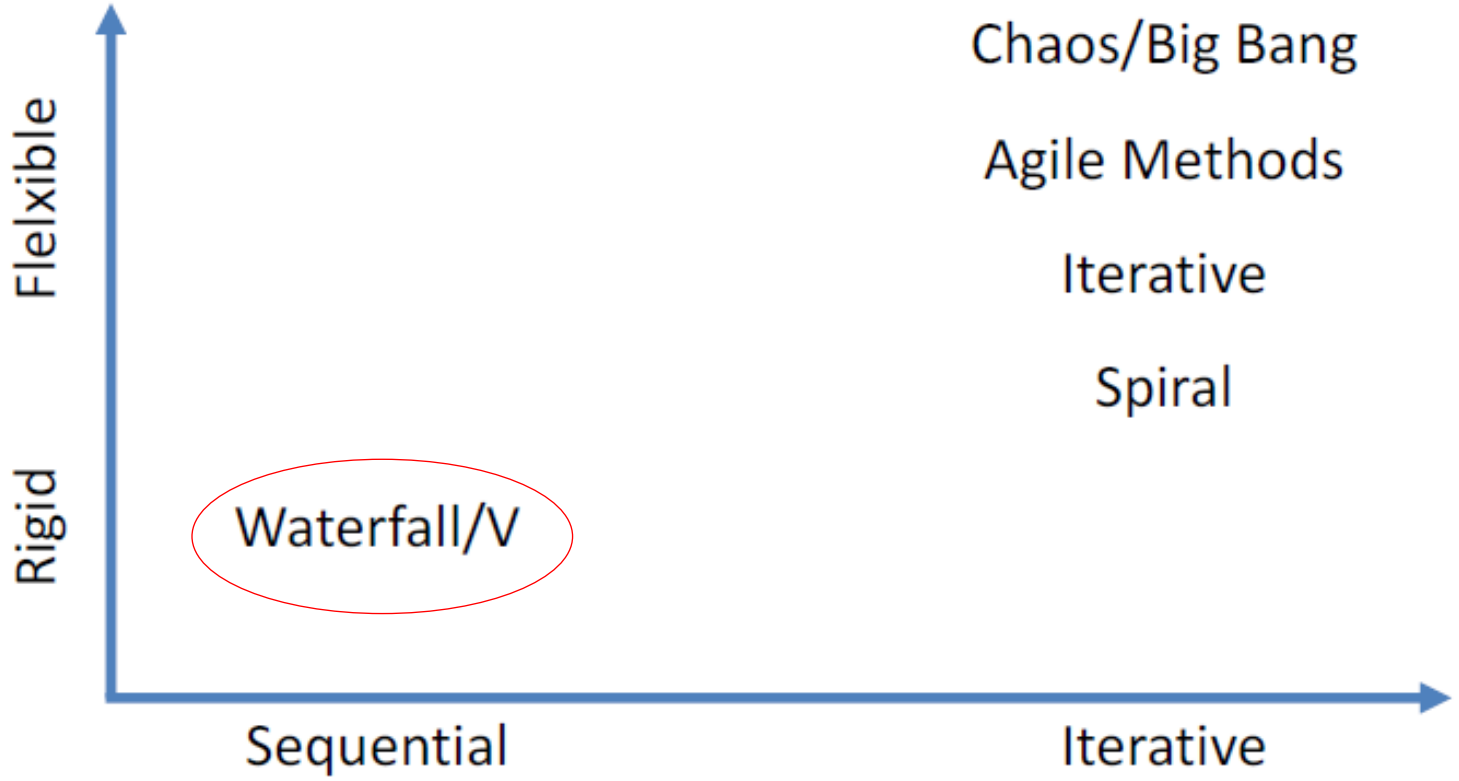


Source: Software Engineering, 10th Edition, Ian Sommerville

SDLC Methods



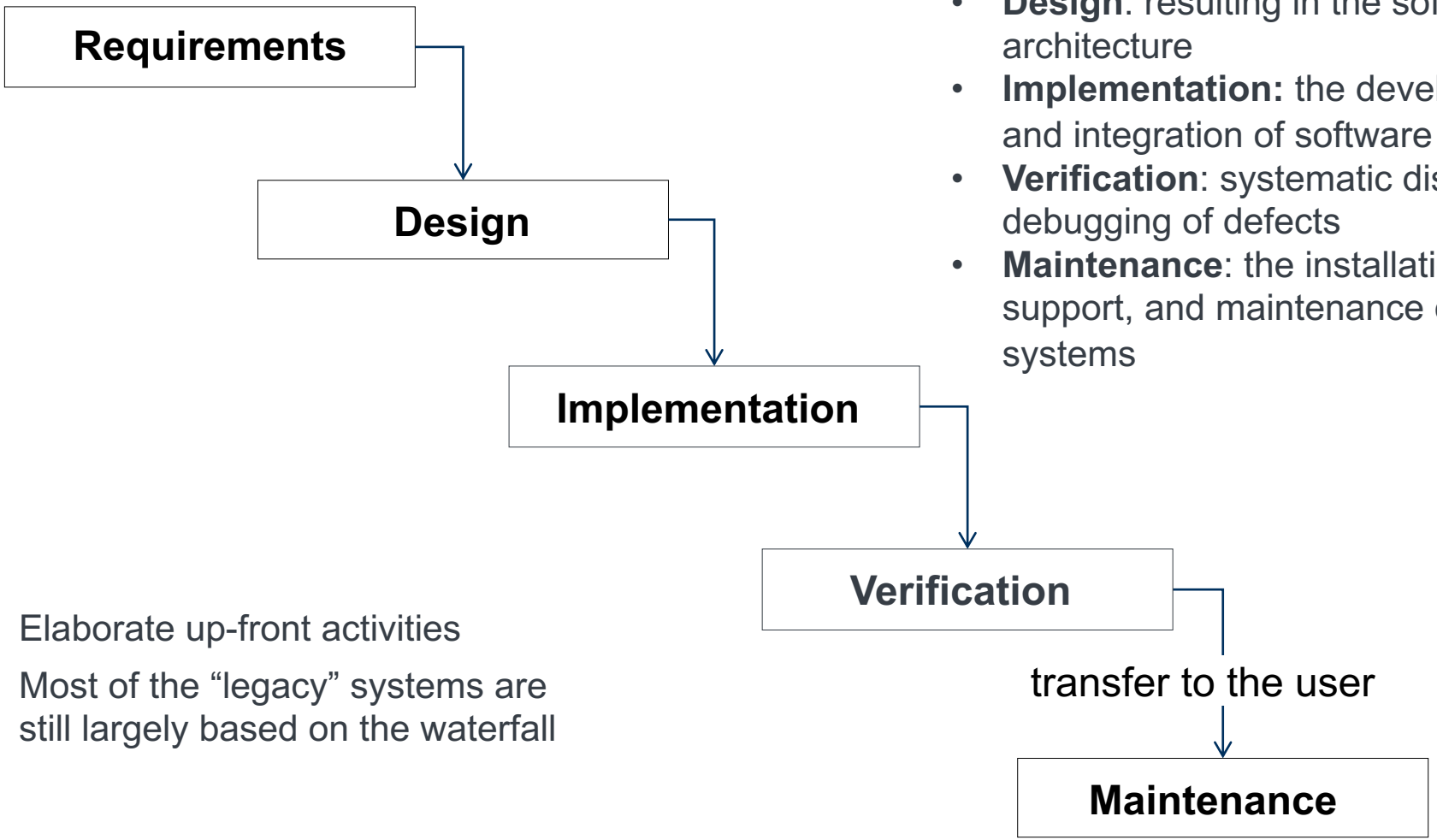
SDLC Methods



Meet the Famous Waterfall Model

A sequential (non-**iterative**) design process, seen as flowing steadily downwards (like a **waterfall**).

Meet the Famous Waterfall Model

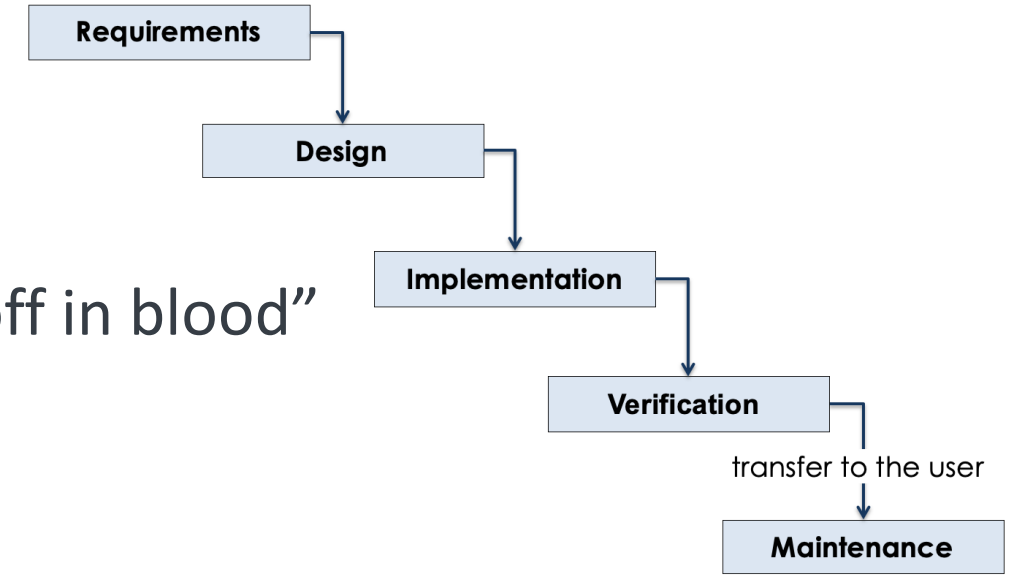


- **Requirements**: captured in a product requirements document
- **Design**: resulting in the software architecture
- **Implementation:** the development, proving, and integration of software
- **Verification**: systematic discovery and debugging of defects
- **Maintenance**: the installation, migration, support, and maintenance of complete systems

- Elaborate up-front activities
- Most of the “legacy” systems are still largely based on the waterfall

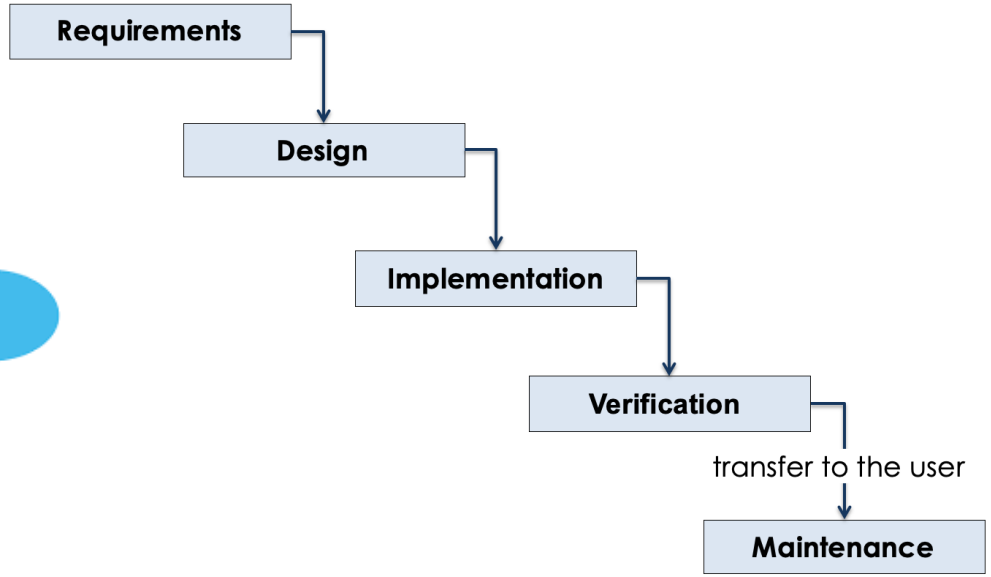
Meet the Famous Waterfall Model

- Sufficient planning
- Popular with traditional engineering problems, e.g. building a bridge
- Very formal process
  - Extensive documentation
  - Serial execution
  - Strict paper work “signed off in blood”



Meet the Famous Waterfall Model

- When is waterfall appropriate?



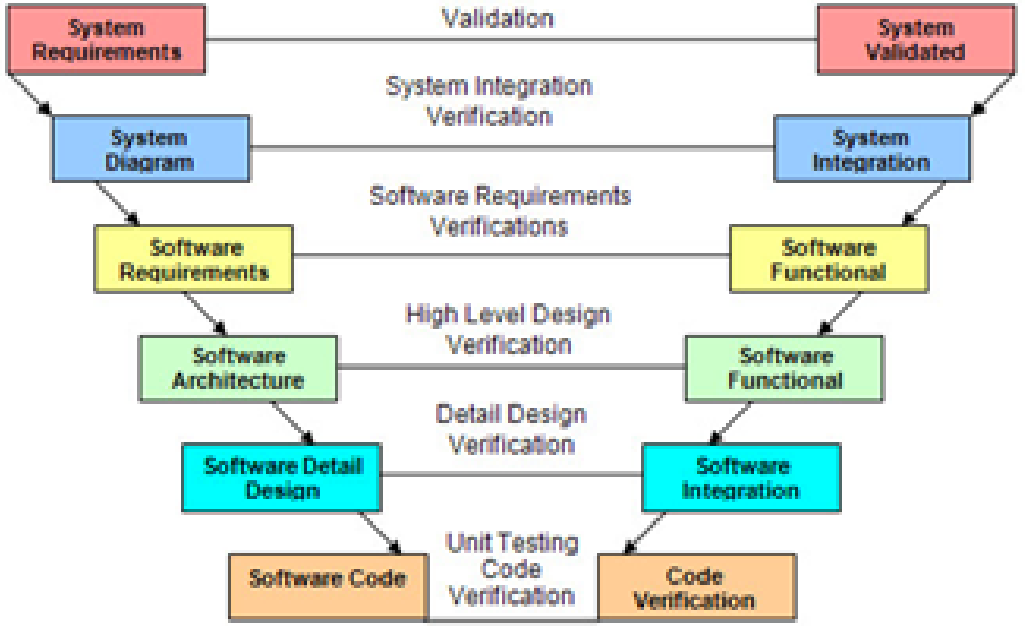
Meet the Famous Waterfall Model

When is waterfall appropriate?

- Must get it right the first time!
  - Interacting with hardware systems, which are difficult to change
- Cost of failure is very high
  - Critical systems with safety or security requirements, e.g., airplanes, self-driving cars…
- Complete and stable requirements
  - Very large, multi-organizational software projects

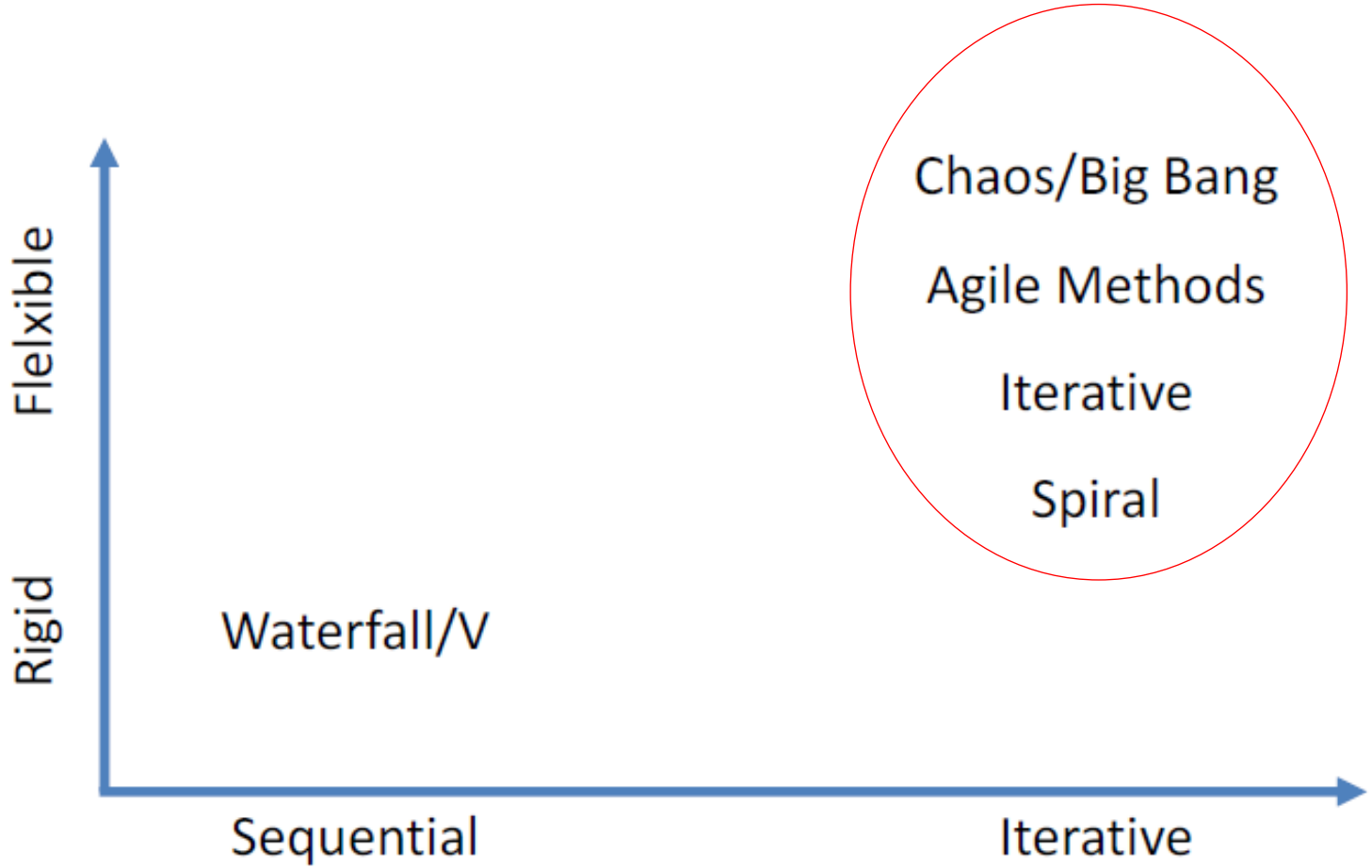
V (Verification/Validation) SDLC Model

- Add verification/testing to each step of the Waterfall Model
- Must complete rigorous testing before proceeding to next step



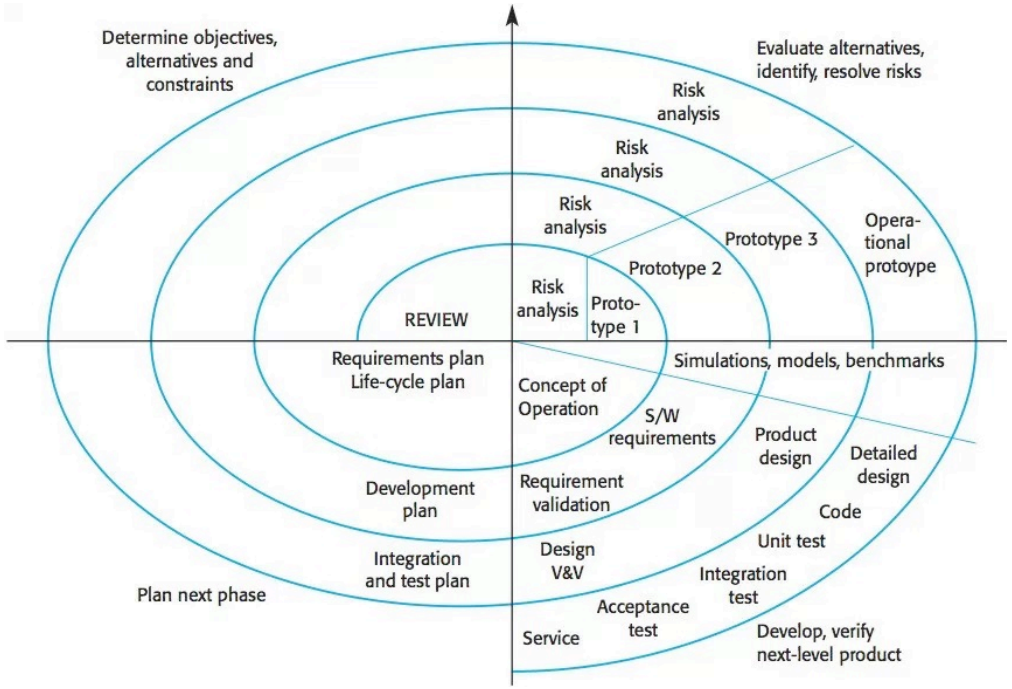
https://sites.google.com/site/advancedsofteng/softwareacquisition/software-development-lifecycle-approaches

SDLC Methods



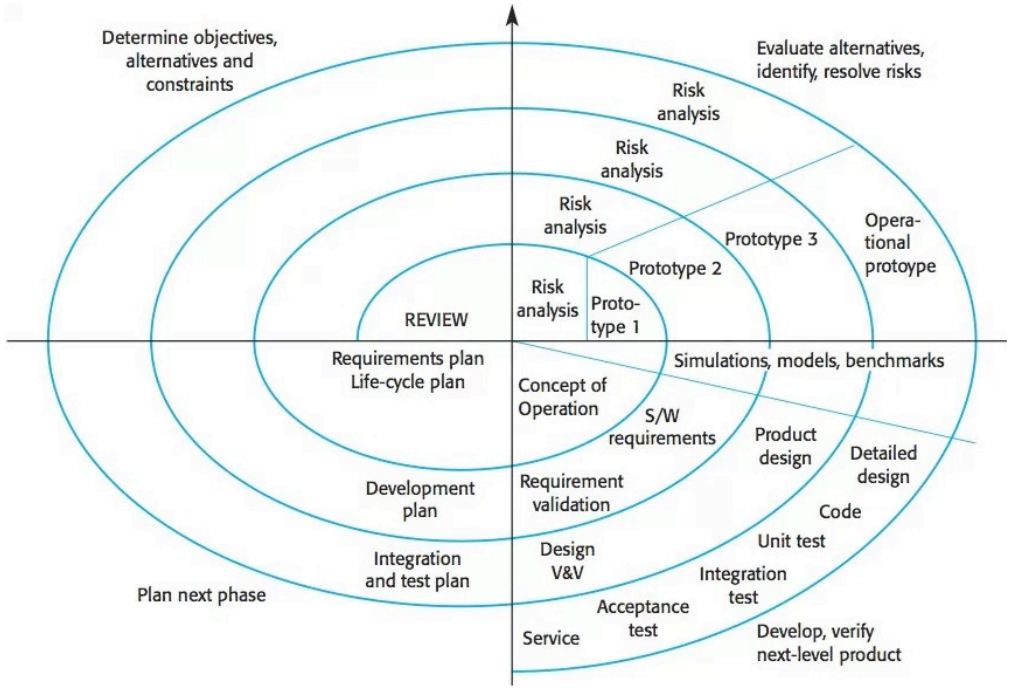
Boehm's Spiral Model

- Limitations of waterfall?



http://iansommerville.com/software-engineeringbook/web/spiral-model/

Boehm's Spiral Model

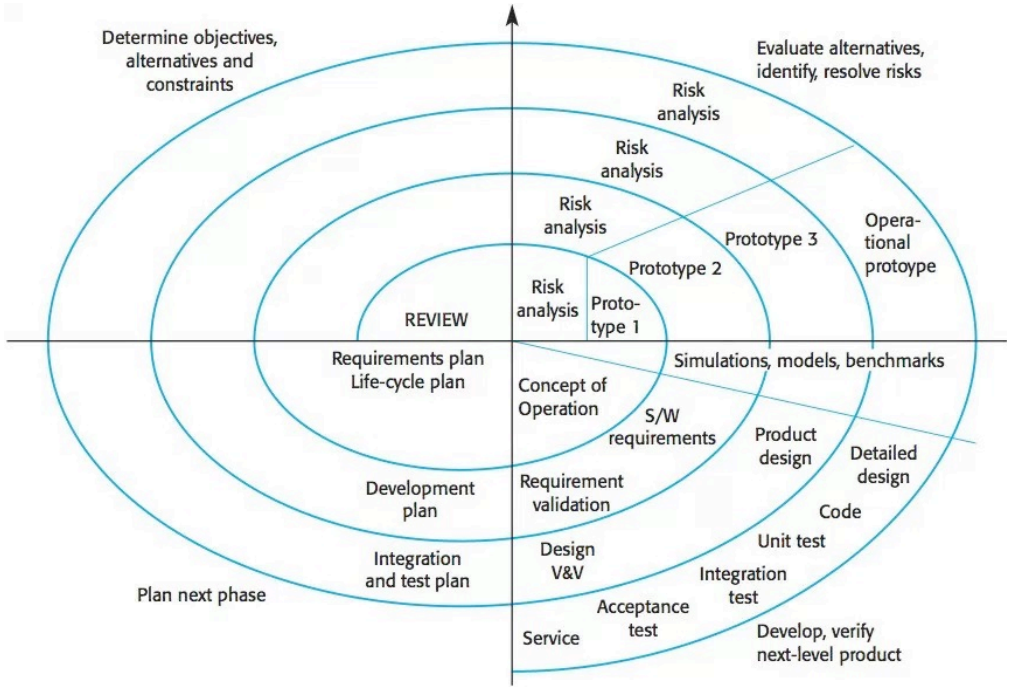


http://iansommerville.com/software-engineeringbook/web/spiral-model/

- Limitations of waterfall?

Hard to get it right the first time and changing requirements…

Boehm's Spiral Model



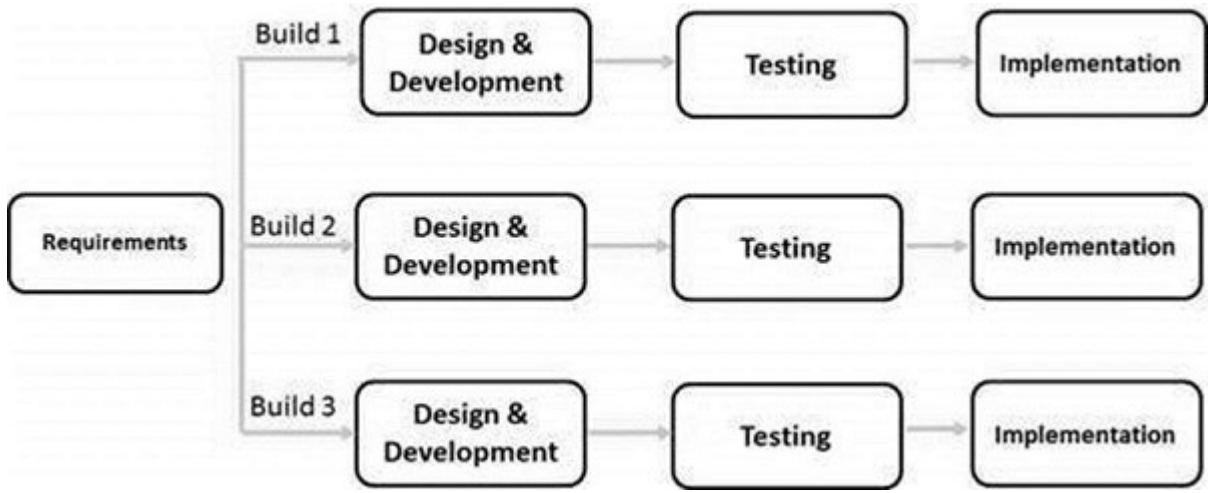
http://iansommerville.com/software-engineeringbook/web/spiral-model/

- Limitations of waterfall?

Hard to get it right the first time and changing requirements…

- Incremental development and interactions
  1. Objective setting
  2. Risk assessment and reduction
  3. Development and validation
  4. Planning for next iteration

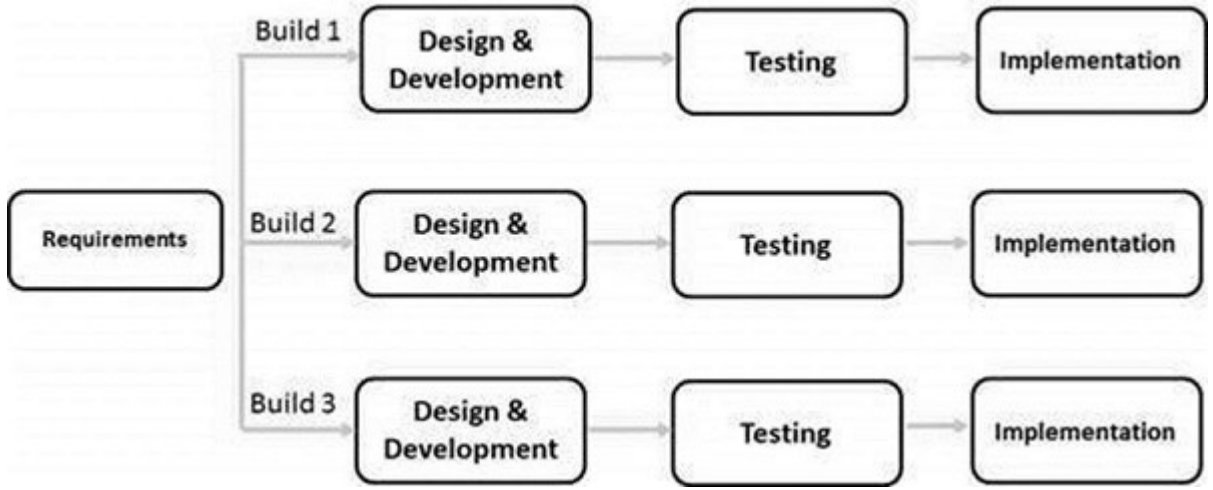
Iterative models SDLC



https://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm

- Identify requirements up front
  - Mostly stable but may change…
- Build subsets of requirements
  - Sequentially or in parallel with multiple groups of developers

Iterative models SDLC

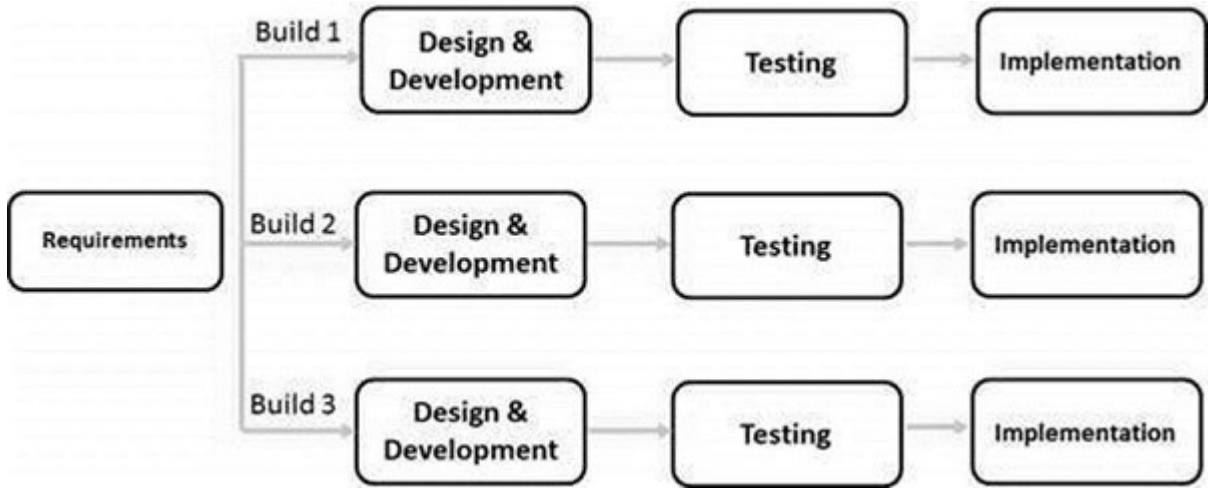


Benefits?

https://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm

- Identify requirements up front
  - Mostly stable but may change…
- Build subsets of requirements
  - Sequentially or in parallel with multiple groups of developers

Iterative models SDLC



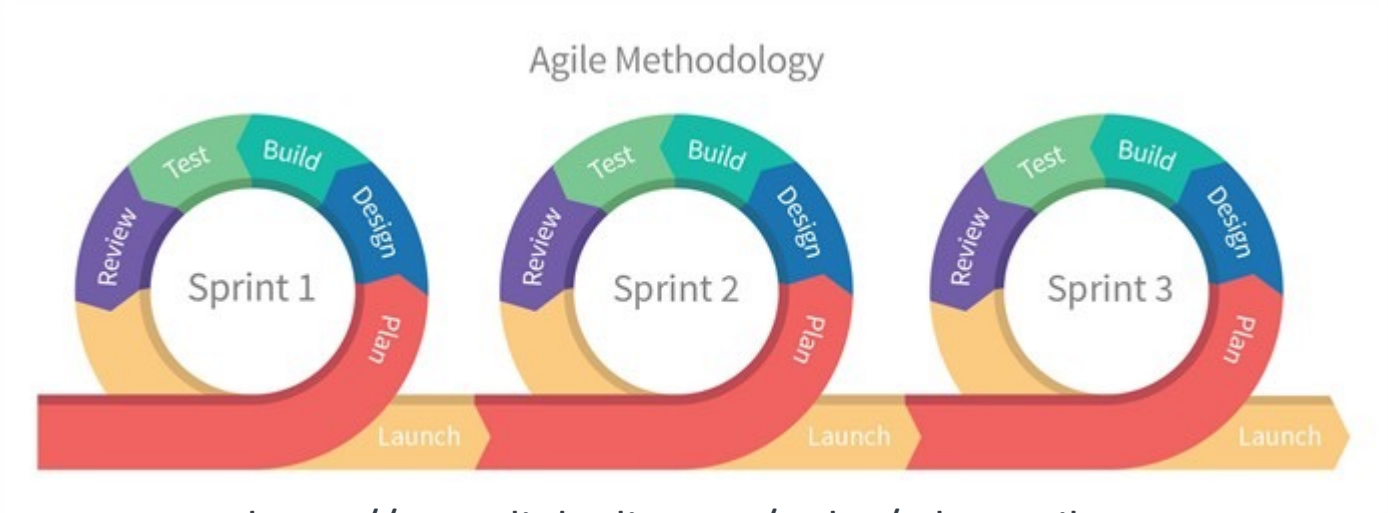
https://www.tutorialspoint.com/sdlc/sdlc_quick_guide.htm

- Identify requirements up front
  - Mostly stable but may change…
- Build subsets of requirements
  - Sequentially or in parallel with multiple groups of developers

Benefits?

- Parallel effort supported by multiple teams
- Delivers early functionality to customers for review

Agile Methods SDLC

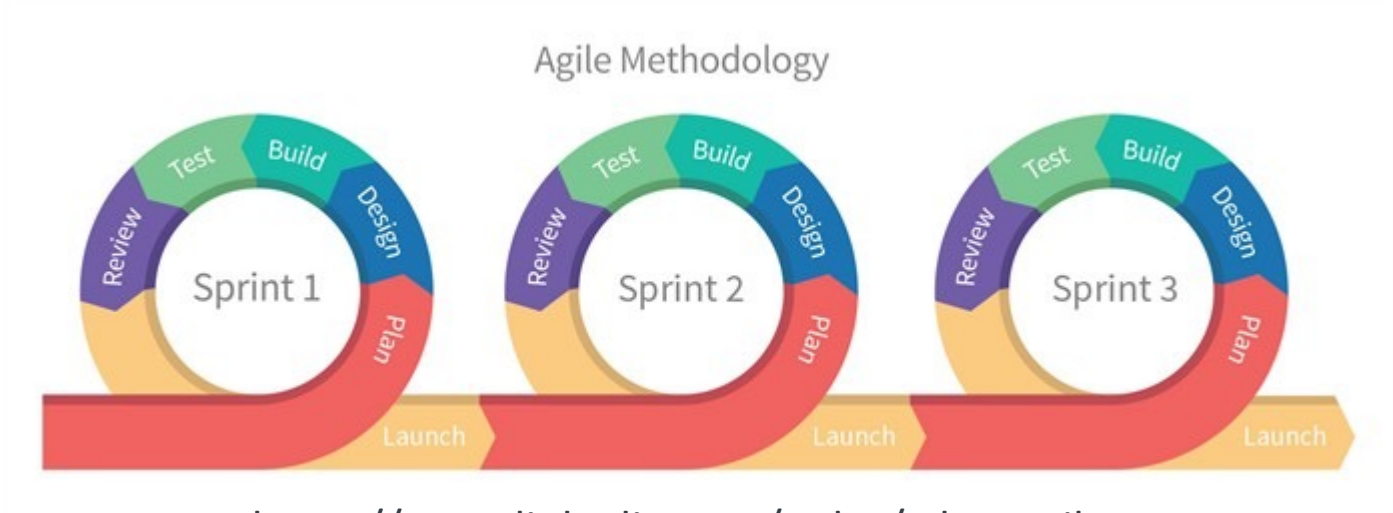


https://www.linkedin.com/pulse/what-agile-methodologydisadvantage-waterfall-model-bikesh-srivastava

"Agile software Development" is an umbrella term for frequent iterative and incremental programming development approaches.

e.g. Extreme Programming (XP), Scrum, Crystal, DSDM, Lean, FDD

Agile Methods SDLC



https://www.linkedin.com/pulse/what-agile-methodologydisadvantage-waterfall-model-bikesh-srivastava

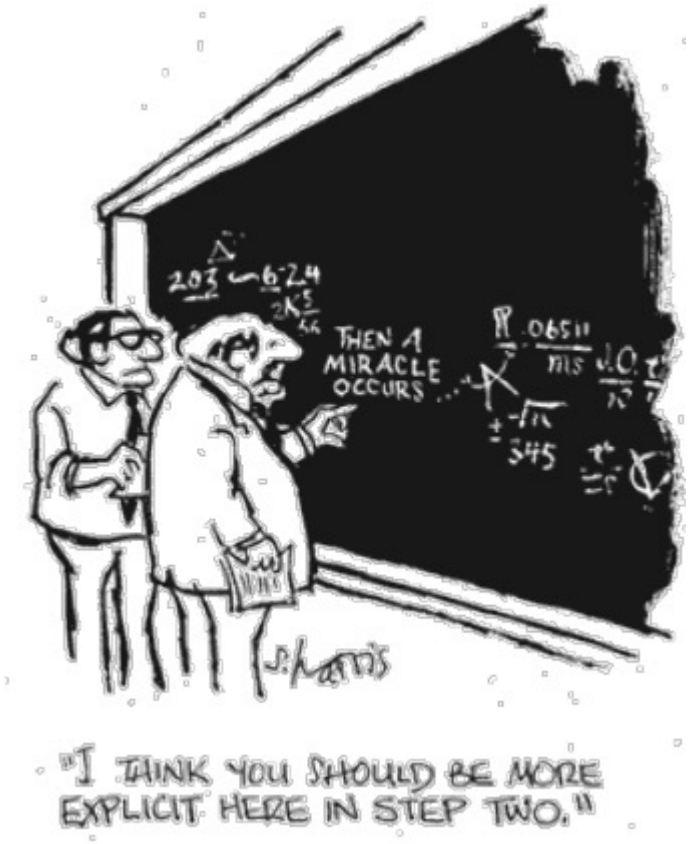
- Frequent iterations and deliverables
- Close collaboration between developers and customers
- Support changing requirements
- Frequent retrospection: learn and improve from experience

Reasons to Use Agile Methods

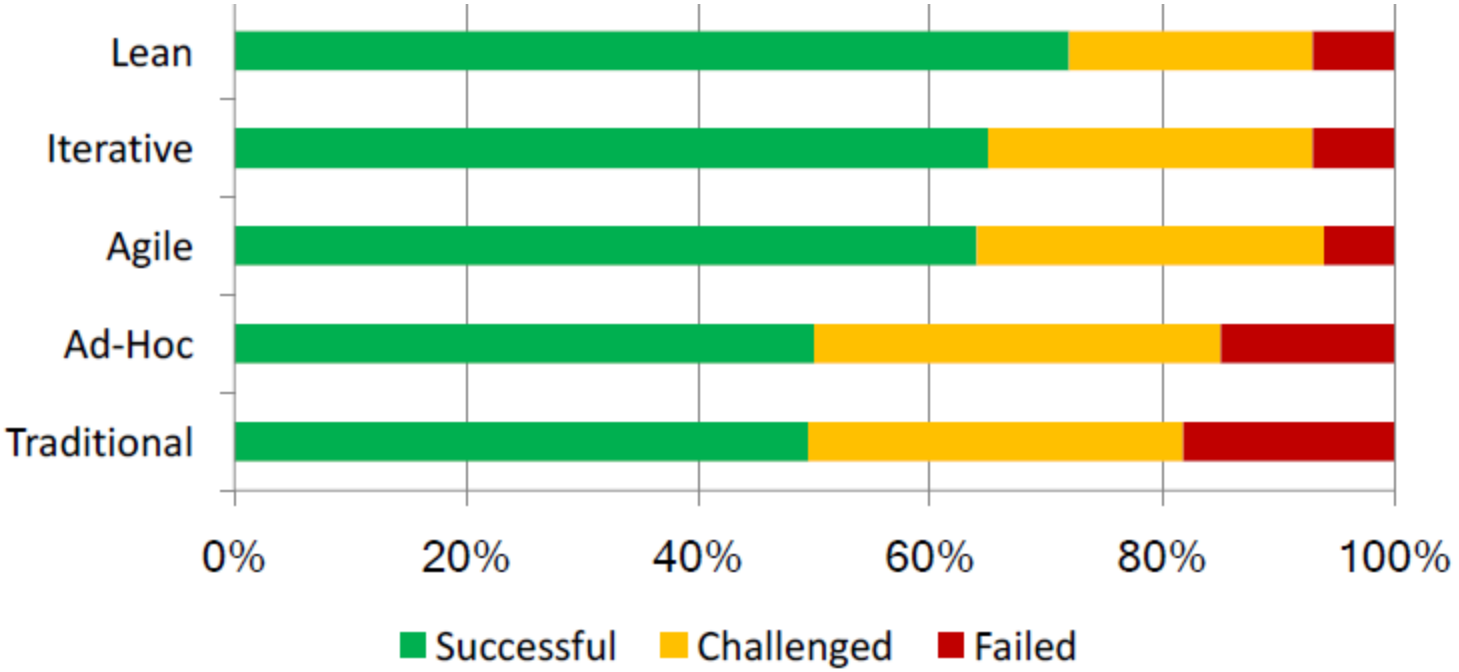
- Big, upfront planning is not practical because of unstable (changing) and ambiguous requirements
- Delivery through small baby steps through iterative and incremental development to reduce the chances of risk.
- Visibility with customers: customers are part of the team instead of being purely observers.
- Frequent reflections by the project teams
  - What are we doing well?
  - What can we improve?

Big Bang/Chaos SDLC

- Little to no planning
- Figure it out as you go
- Typically used for very small projects (e.g. course projects…, small start-ups)
- Not highly recommended…



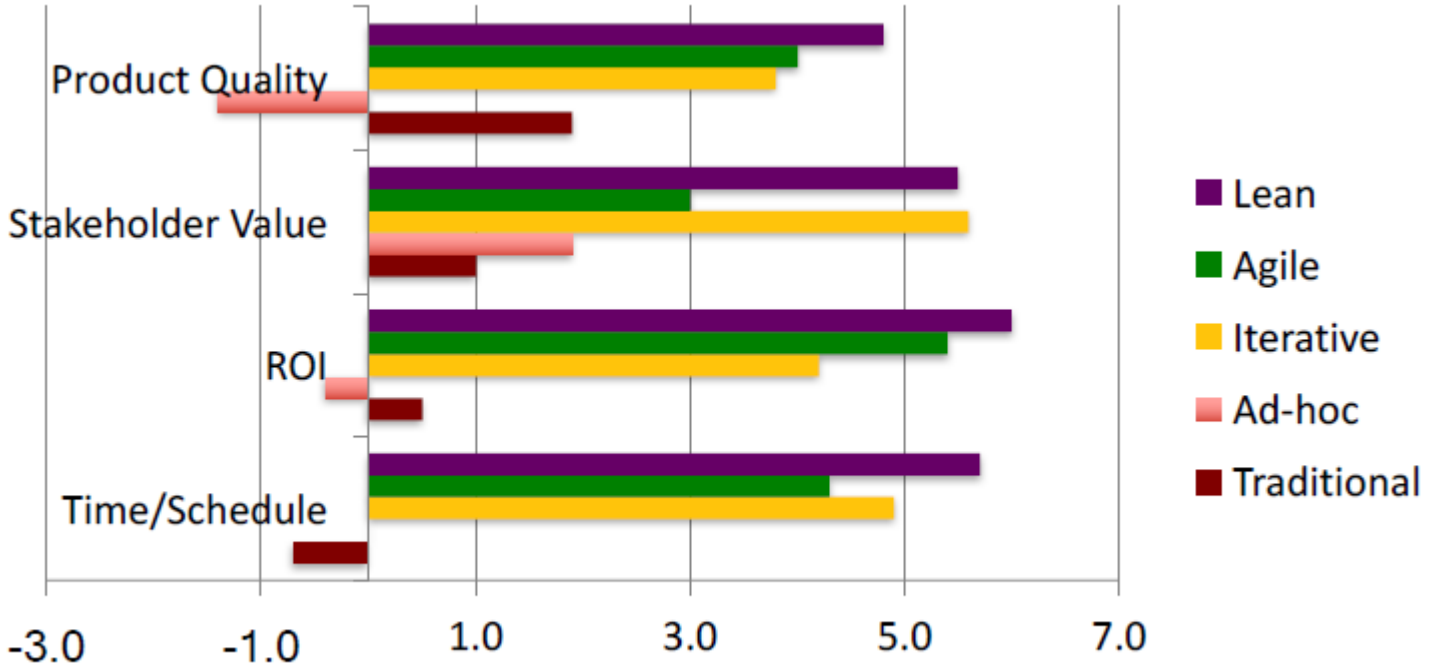
Comparing Software Development Paradigms: 2013



SCOTT AMBLER + Associates

Copyright 2014 Scott W. Ambler www.ambysoft.com/surveys/

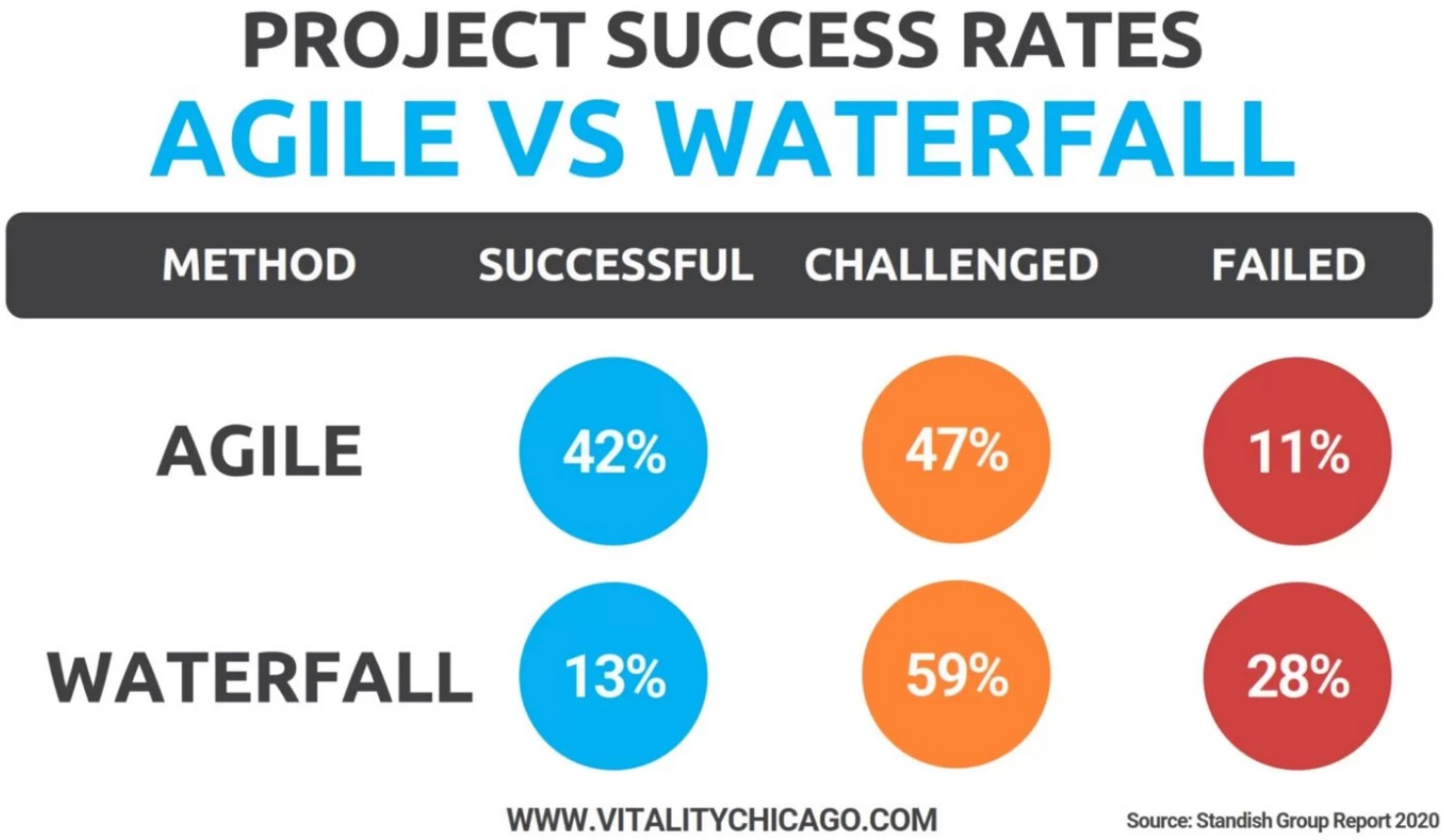
Comparing Delivery Paradigms



SCOTT AMBLER + Associates

Copyright 2014 Scott W. Ambler www.ambysoft.com/surveys/

Comparing Software Development Paradigms: 2020



Chaos 2020 Beyond Infinity. This report is based on an impressive database of 50,000 projects.

STEVENS
INSTITUTE OF TECHNOLOGY
1870

THANK **YOU**

Stevens Institute of Technology
1 Castle Point Terrace, Hoboken, NJ 07030