Agile

- How would you implement agile methodologies and tools for web projects?
- What do you see as the benefits and challenges to doing this?

What is Agile?

The term *agile* (sometimes written *Agile*) was popularised by the *Agile Manifesto (2001)*,[which defines those values and principles. Agile software development frameworks continue to evolve, two of the most widely used being Scrum and Kanban. (wikipedia)

Agile Manifesto

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

The 12 principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximising the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organising teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Why Agile?

“We were doing incremental development as early as 1957, in Los Angeles, under the direction of Bernie Dimsdale at IBM's Service Bureau Corporation. All of us, as far as I can remember, thought waterfalling of a huge project was rather stupid, or at least ignorant of the realities. I think what the waterfall description did for us was make us realize that we were doing something else, something unnamed except for 'software development.’" - Gerald M. Weinberg

Agile vs Waterfall

Both Agile and waterfall have the same SDL (Software Development Lifecycle):

1. Analysis
2. Design
3. Code
4. Test

The main difference is that Agile works in a series of relatively short but incremental iterations, Waterfall spans the length of the project, where everything is delivered usually in one stage.

Agile Methodologies

1. Lean - origins 1948 - 70's
2. Kanban - origins 1948 - 70's
3. Scrum - origins 1948 70's - 1986
4. XP - eXtreme Programming - 90's
5. DSDM- Dynamic Systems Development - 1994
6. FDD - Feature Driven Development - 1997
7. Crystal - 2001

Lean

Originated from Japanese Car Manufacturer - derived from the principles of the Toyota Production Model or Just in Time (1948 - 1975). Lean methodology eliminates waste through such practices as selecting only the truly valuable features for a system, prioritising those selected, and delivering them in small batches.

- Eliminating Waste
- Amplifying Learning
- Deciding as Late as Possible
- Delivering as Fast as Possible
- Empowering the Team
- Building Integrity In
- Seeing the Whole

Kanban

- Visualise what you do today (workflow): seeing all the items in context of each other can be very informative
- Limit the amount of work in progress (WIP): this helps balance the flow-based approach so teams don‘t start and commit to too much work at once
- Enhance flow: when something is finished, the next highest thing from the backlog is pulled into play

Scrum

Scrum for software was directly modelled after "The New New Product Development Game" by Hirotaka Takeuchi and Ikujiro Nonaka published in the Harvard Business Review in 1986.

Nonaka was hired by the Japanese government after World War II to help analyse why they lost the war. He does not use a computer and to him, Scrum is only indirectly related to software. It is directly related to leadership and running the top companies in the world.

- Stand Ups
- Product Owner
- Product Backlog
- Cross Functional Teams
- No additional functionality can be added to the sprint except by the team
- Proven to scale

XP - eXtreme Programming

Chrysler 1990’s
Emphasis on shorter product lifecycles and improving speed to market

- Planning Game
- Small Releases
- Customer Acceptance Tests
- Simple Design
- Pair Programming
- Test-Driven Development
- Refactoring
- Continuous Integration
- Collective Code Ownership
- Coding Standards
- Metaphor
- Sustainable Pace

DSDM - Dynamic Systems Development Method

Group of companies met in London circa 1994 including BA, American Express, Oracle, Butler Group
Evolution from RAD (Rapid Application Development)

- MoSCoW Method
- 80% of system can be developed in 20% of the time
- low priority items added to backlog, can be removed to prevent from delivering high value items

FDD - Feature Driven Development

FDD was initially devised by Jeff De Luca to meet the specific needs of a 15-month, 50-person software development project at a large Singapore bank in 1997

- Domain Object Modeling
- Developing by Feature
- Component/Class Ownership
- Feature Teams
- Inspections
- Configuration Management
- Regular Builds
- Visibility of progress and results

Crystal

Alistair Cockburn, the originator of Crystal, has released a book, Crystal Clear: A Human-Powered Methodology for Small Teams. Emphasises:

- Collaboration
- Teamwork
- Simplicity

The use of the word Crystal comes from the gemstone where, in software terms, the faces are a different view on the “underlying core” of principles and values. The faces are a representation of techniques, tools, standards and roles.

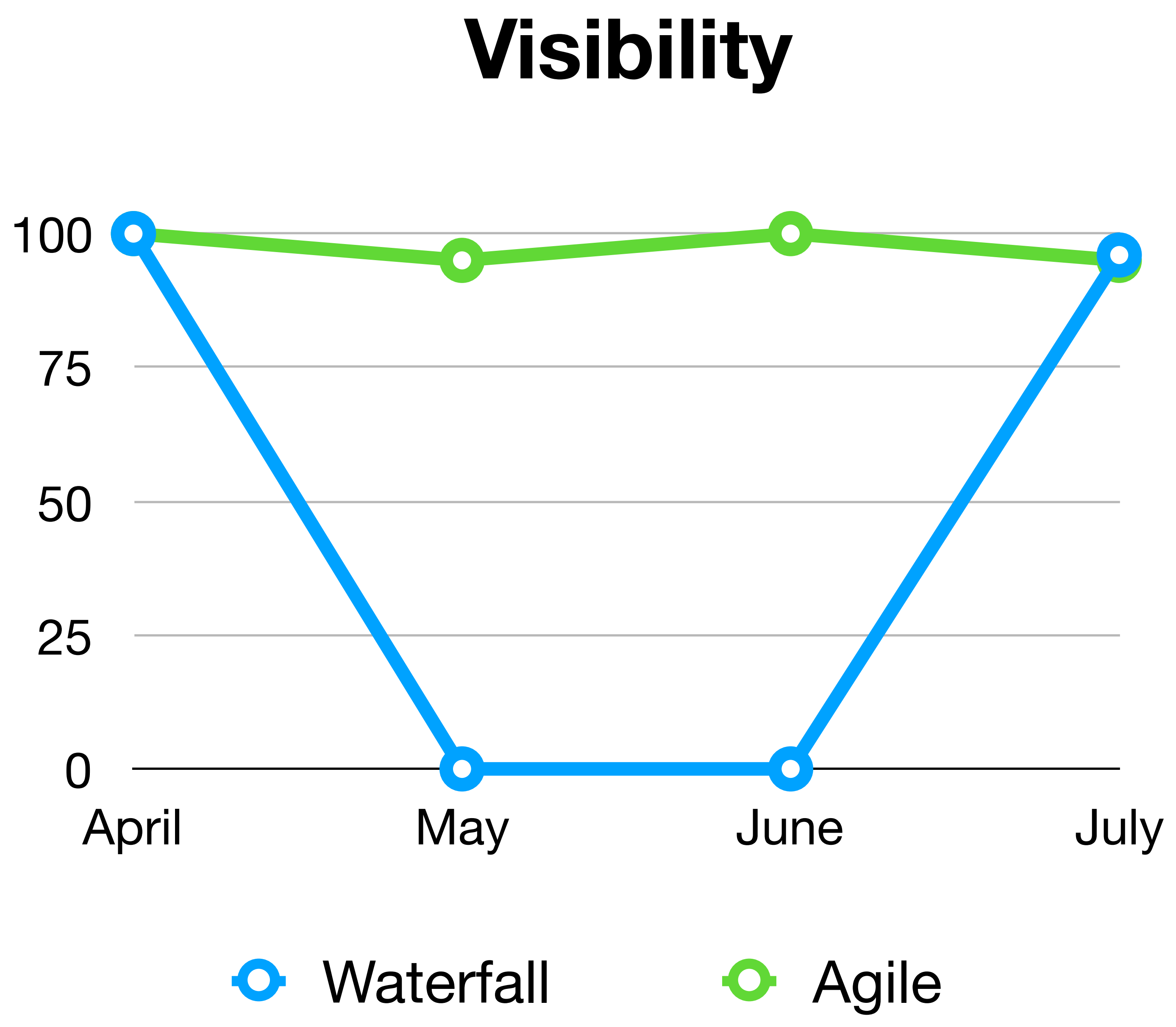
IBM circa 2001

Original member of Agile manifesto

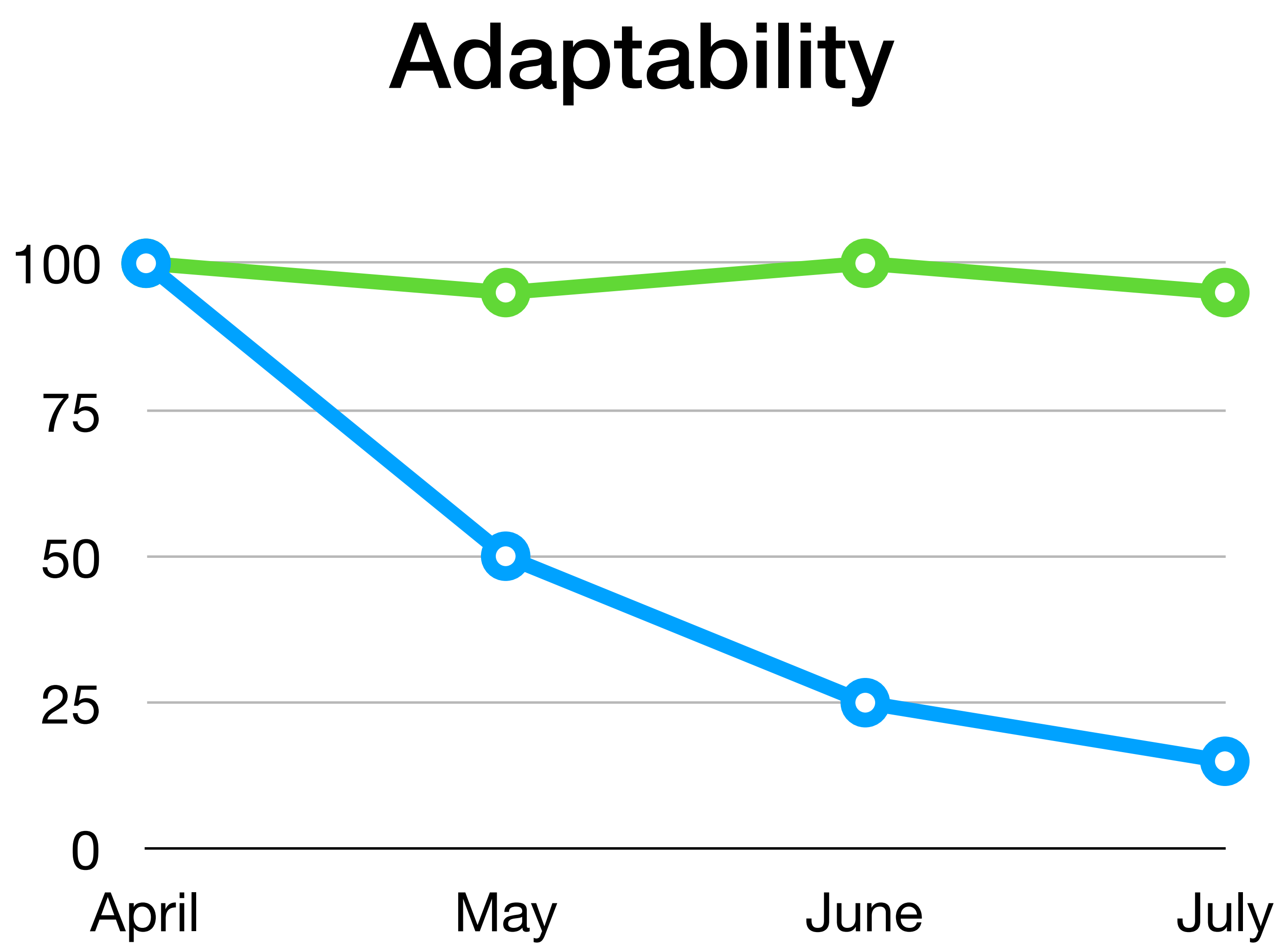
Benefits of Agile

- Lightweight framework for helping **teams**
- Enables rapid delivery of business **value**
- Failing projects fail faster
- Reduces Risk
- Is a more robust approach, adapting to change requirements - **feedback** loop
- More likely to create a system that is more suited to either the business of customer needs

Agile vs Waterfall

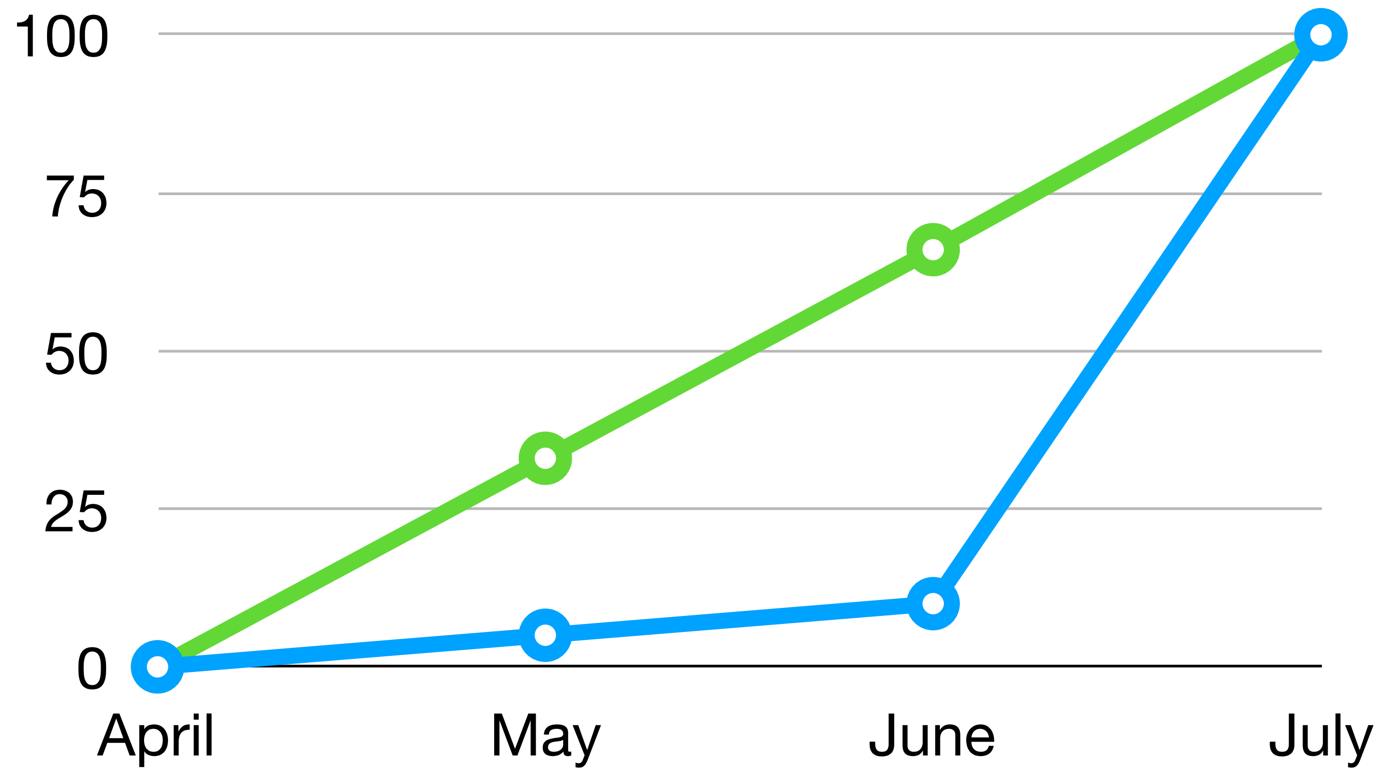


Agile vs Waterfall

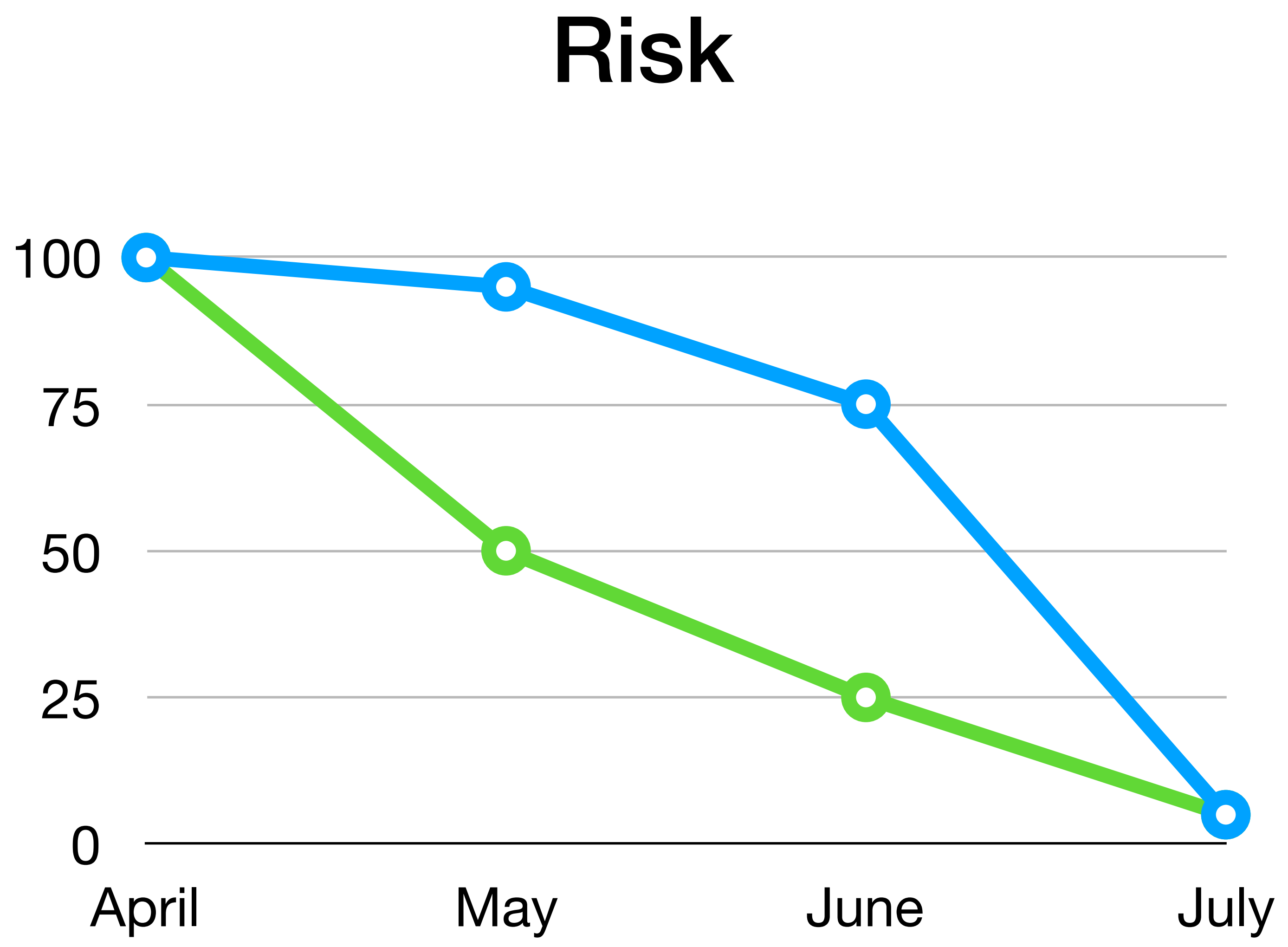


Agile vs Waterfall

Value



Agile vs Waterfall



Challenges

- The organisation isn’t ready for Agile - Lack of agile culture
- We can’t mess up or the Scrum Police will come and take us away
- Code Broken Accidentally due to Frequent Builds – Since code is changed and compiled daily, the likelihood of code breaking existing features is much higher.
- Inadequate Test Coverage – With continuous integration and changing requirements, it can be easy to miss critical tests for any requirement.
- Early Detection of Defects - Development time spent fixing defects as a priority can impact on other deliverables in sprint
- Ensuring the right side of the Agile Manifesto truly enables and strengthens the left side is the #1 challenge with agile projects