
agile

Topics

- What is Agile
- Agile Values
- Agile Principles
- Why Agile
  - Waterfall
  - Waterfall to Agile
  - Iterative Development
- Change of Mindset
- Myths

what is agile?

Agile – Values

What is Agile?

"Agile is an **Iterative and Incremental** *approach* to software development performed in a highly **collaborative** manner by **self-organizing teams** with **"just enough"** ceremony that produces high **quality** software in a **cost effective** and **timely** manner to meet the **changing needs** of its stakeholders."

** **Agile** is a **Value Center….**

Agile – Values

In 2001, a group of software development professionals met up to discuss a better way to produce software… the result was the Agile Manifesto.

The Agile Manifesto states 4 key **Values**:

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*

*"That is, while there is **value** in the items on the **right**, we **value** the items on the **left** more."*

Agile – Values Explained

1. **Individuals and interactions** - *Over processes and tools.*

While tools and processes are fundamental to delivering projects, including Agile projects *(JIRA, Backlogs, User Stories, Burn-down Charts, Scrums, Retrospectives etc…),* it is the people who actually deliver the value and are the key to success.

2. **Working software** - *Over comprehensive documentation*

Agile places the emphasis on:

Delivering **incremental**, **working** software
Delivering true **customer value** before verbose documentation...

3. **Customer collaboration** - *Over contract negotiation*

Customer needs and requirements are the driving force in delivering value. Agile recognizes this by promoting and ensuring continual customer engagement and collaboration throughout projects.

4. **Responding to change** - *Over following a plan*

Agile methodology's understand and welcome the natural evolution of requirements. By incremental delivery and adaptability, Agile products ensure systems are built on the most recent and up to date information.

*"That is, while there is **value** in the items on the **right**, we **value** the items on the **left** more."*

Agile – Principles of the Manifesto

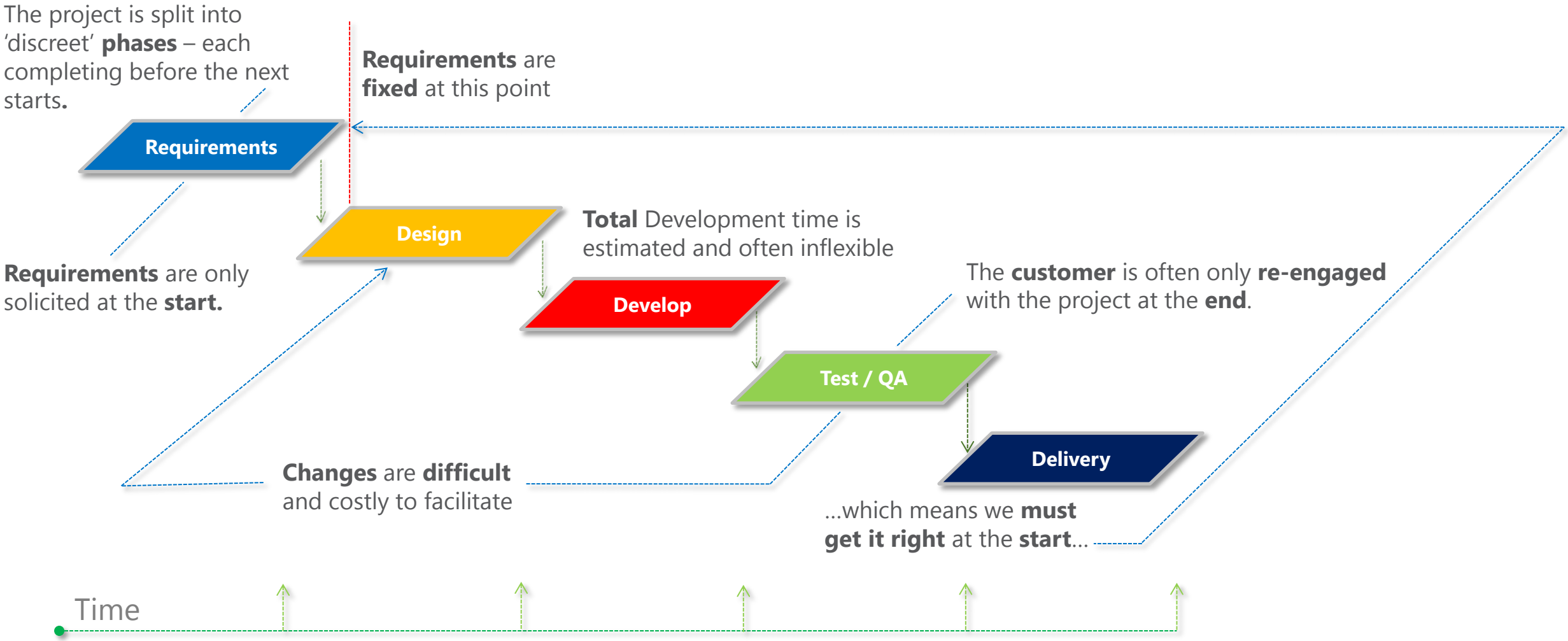
12 key **Principles**

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 maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from **self-organizing teams**.
12. At **regular intervals**, the team **reflects** on how to become more effective, then **tunes** and **adjusts** its behavior accordingly.

why agile

Waterfall Development

Traditionally, software development has followed the standard Waterfall Methodology



Waterfall Development

This approach incurs high levels of risk, is often more costly and generally less efficient than Agile approaches.

- **Quality** – How will this be assured in a fixed time but variable project?
  - If the project starts to run out of time, testing is often cut short.

- **Visibility** is often poor,
  - **Working software** comes at the **end**
  - It is hard to know where in the life-cycle the project is – *the last 20% takes 80% of the time*.

- **Risk** is high until the end –
  - Are the **requirements solicited** several months or years ahead of delivery still what the customer needs?
  - Is the **design** (based on the requirements collected at the start) still correct, taking into account latest technologies etc.?
  - Are the features delivering the right **value** – has the market changed since the Requirements phase?
  - Will the project get over the line, per **schedule** etc.?

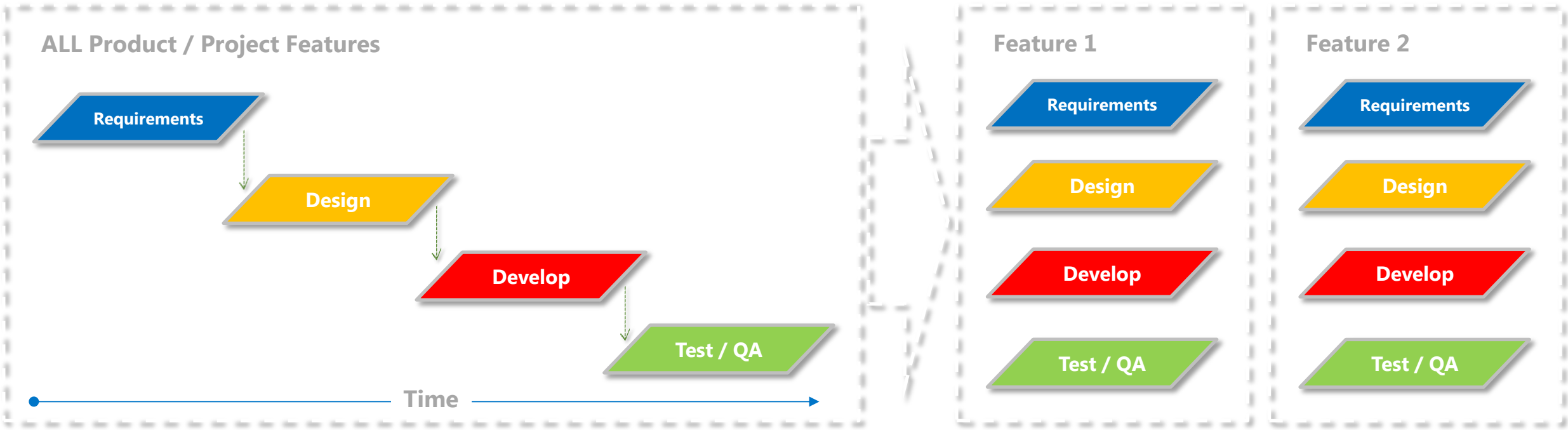
- **Changes –** are difficult to facilitate without costly (time and money) Variations and Change Requests
  - **Value** is not realized until the end.
  - **Stakeholders** are often not re-engaged until development is complete and UAT starts.
  - **Requirements** (solicited at the start of the project) may need to be updated once the Customer uses the software.

Waterfall to Agile

Instead of fixed phases, Agile methodologies consider these to be continuous activities – each 'Feature' is a '**Vertical Slice**' from *Requirements* to *Test Completion* :

Waterfall

Agile

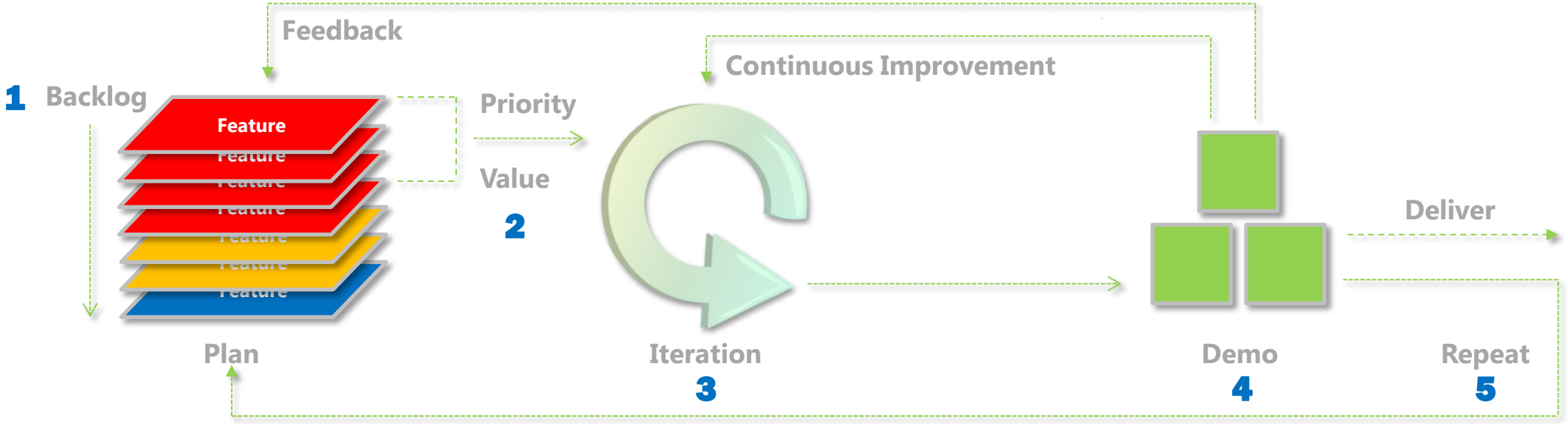


Requirements, **Design**, **Development** and **Testing** are continuous activities throughout an Agile project. *As long as there are still features requested by the customer to be built – these activities are maintained and development prioritized for each feature.*

Iterative Development

Agile methodologies are driven by the goal of delivering maximum **business value** in the least **time**. This is achieved by **iteratively** delivering high-priority / high value customer driven requirements quickly and often.

1. **Requirements** or **'Features'** are collated as '**User Stories'** and prioritized (top to bottom) in a '**Product Backlog**' The Backlog is refined throughout the project and is updated as requirements or priorities change.
2. **Priority** (highest **Business Value**) Stories are refined, planned and selected to be worked on. Priority is determined by the Customer and changes can be requested at any point, prioritized and queued to be worked on.
3. **Iterations** (or *Sprints)* are worked on based on the priorities set.
4. **Customers** are invited to view working software at the end of each Iteration in order to understand progress and to provide **Feedback**.
5. **Continuous Improvement** is achieved through inspecting and adapting processes at the end of each iteration, and the process **repeated**.

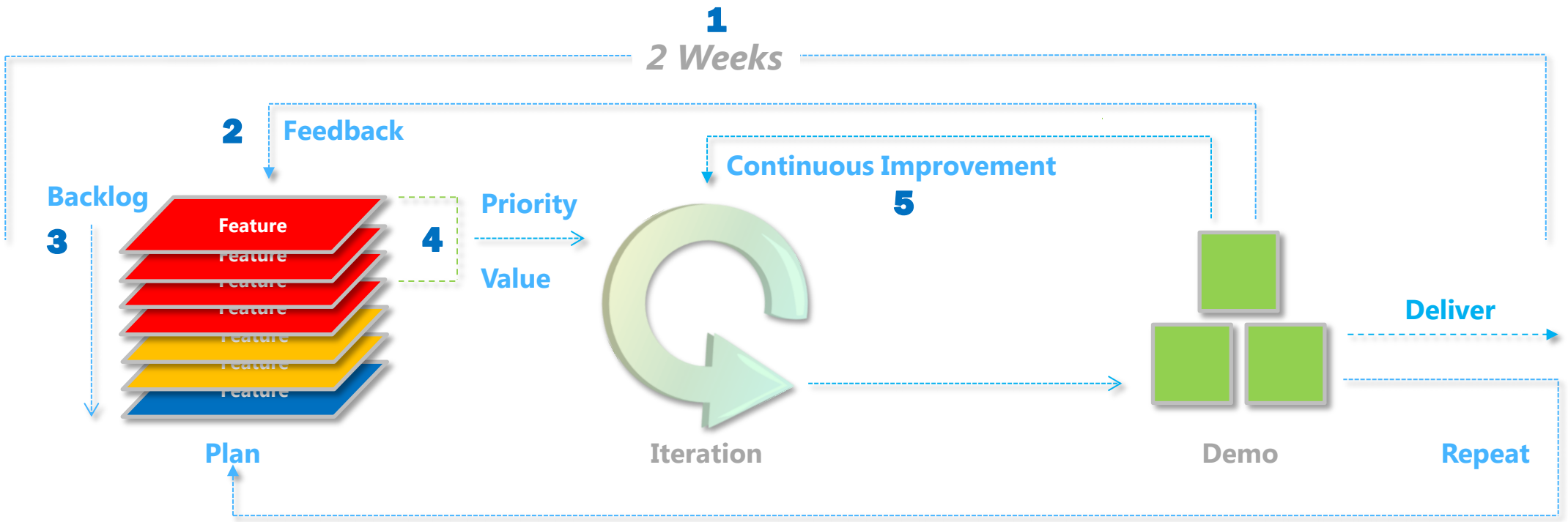


Iterative Development Benefits

Maximum **business value** in the least **time**.

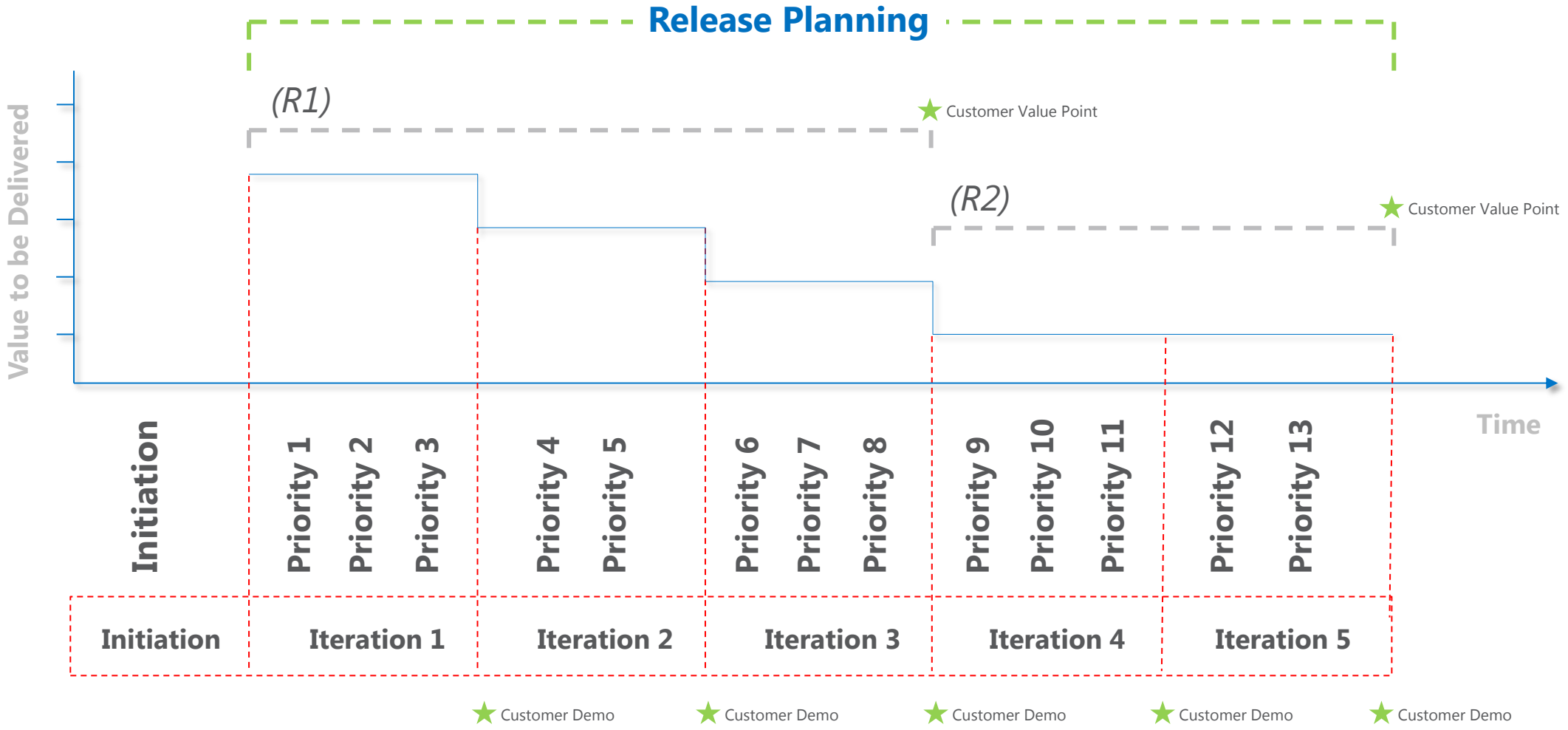
Some of the benefits of Iterative Development are:

1. **'Fail Fast'** – continual review allows for reassessment of work without wasting large amounts of time.
2. **Changes** to requirements are better accommodated – they are requested throughout development and prioritized in the backlog.
3. **Information** is updated throughout development and therefore requirements remains current.
4. **Incremental** delivery , quicker releases of value to customers – most valuable features first. Not all at once.
5. **Continual Improvement** is developed through regular reflection and retrospection.



Agile – Iterative Development / Value Delivery

Agile methodologies are driven by the goal of delivering maximum **business value** in the least **time**. This is achieved by **iteratively** delivering **high-priority** / high value customer driven requirements as often as possible.



Agile – Iterative Development

Agile methodologies are driven by the goal of delivering maximum **business value** to **customers** in the least **time**. This is achieved by **iteratively** delivering high-priority / high value customer driven requirements quickly and often.

- **Quality** – Is assessed incrementally providing time to address issues – not all at the end
  - Each prioritized requirement / features are tested through iterations.
  - The customer reviews work completed at the end of each iteration.
- **Visibility** – Is increased through Agile practices
  - Release planning facilitates a clear view of what will be released and when.
  - Each iteration is time set and openly tracked using highly visual 'task board'.
- **Risk** is assessed and mitigated through, and at the end, of each iteration
  - Requirements are iteration based and kept up to date through incremental refinement and customer involvement.
  - Value is assessed and delivered often, assuring delivery of what the customer wants.
  - 'Fail Fast' / 'Recover Quickly' – continual review allows for reassessment of work without wasting large amounts of time.
- **Changes –** are easily accommodated through customer review, prioritization and iterative development
  - Value is assessed on each incremental delivery.
  - Requirements can be changed based on each incremental delivery
  - Stakeholders are constantly engaged to re-evaluate priorities / features

Agile – Change of Mindset

Moving from a Waterfall way of working to Agile methodologies requires a change in mindset.

	Waterfall	***Agile***
Prioritization of requirements	Fixed in the project plan	*Based on business value and regularly updated*
Emphasis	Process	*People*
Domain	Predictable	*Unpredictable / Exploratory*
Documentation	Verbose	*Only as required*
Quality assurance	Process centric	*Customer centric*
Process style	Linear	*Iterative*
Organization	Managed	*Self-organised*
Upfront Planning*	High	*Lean*
Perspective toward change	Sustainability	*Adaptability*
Management Style	Autocratic	*Decentralised*
Leadership	Command and control	*Collaborative, Servant Leadership*
Performance Measurement	Plan conformity	*Business value*
Returns on Investment	End of project life	*Early / throughout project life*

Agile - Myths

There are several myths associated with Agile development, these include:

Agile is a silver bullet: Unfortunately failure can be as spectacular using Agile methodologies as it can using any other methodology. However, Agile will let you 'Fail Quicker' or 'Recover Quicker' through short iterations, transparency and visibility, and therefore let you take corrective action through inspection and adaption.

No need for documentation: This myth comes from the Agile Value "working software over comprehensive documentation". The operative word is '*over*' – there is definite value in documenting solutions.

Agile is anti-planning: Agile methodologies utilize a lot of planning, through Backlog Refinement, Daily Stand-up Meetings, Sprint, or Iteration Planning Meetings, Regular Introspection Meetings etc.

Agile is faster: There is some truth in this myth once the statement is clarified - Agile is faster at delivering incremental value through iterative development and continuous integration. That is, value can be released to customers more frequently.

Agile is anti-architecture: Agile very much values good architecture and design is key to successful delivery. As in any Software Engineering development discipline the foundations and design are crucial to the ongoing success of the application and development