Agile and Why It Works

David Starr
PRINCIPLE SOLUTIONS ARCHITECT

@ElegantCoder www.elegantcode.com

Overview/ Summary

Comparing development processes

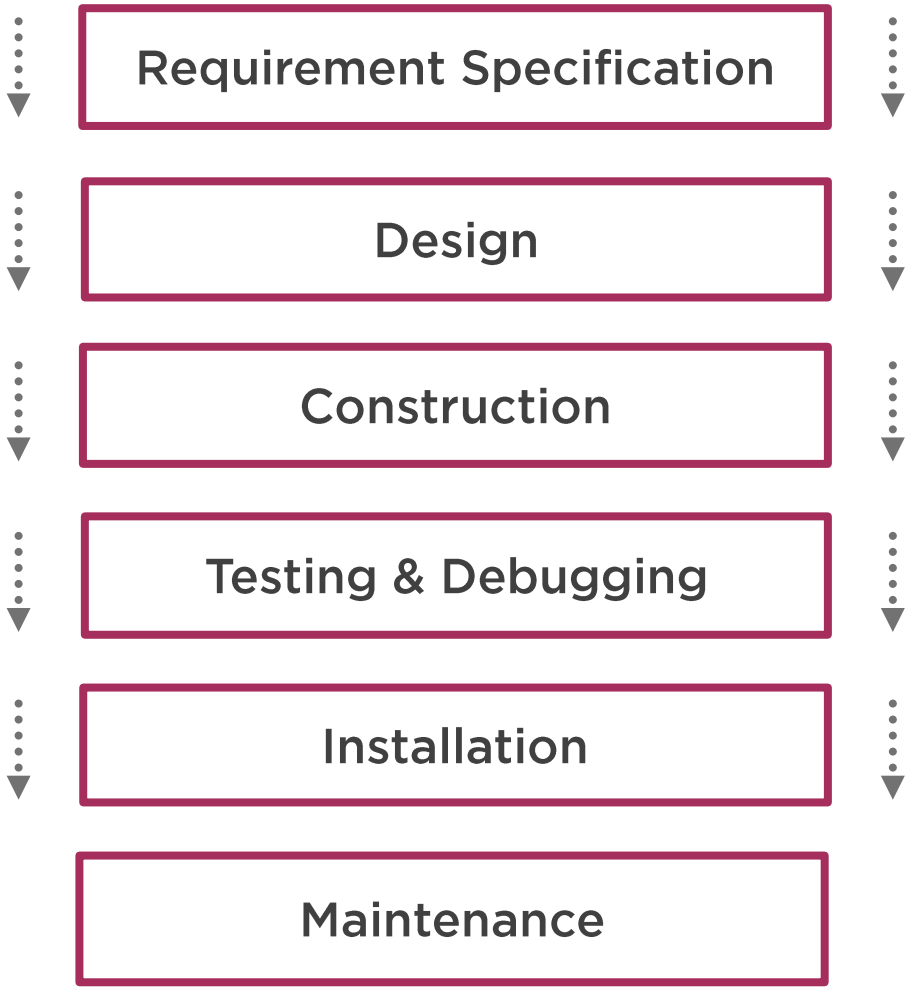
What Agile is and is not

Contemporary Agile Methodologies

Comparing Development Processes

Agile and Plan Driven

Winston Royce's Waterfall Model



From the 1970 IEEE paper "Managing the Development of Large Software Systems"

Each phase should pass iteratively to the next

The entire process should be exercised twice before release

Royce knew that a single pass will fail

Unfortunately, for the process illustrated, the design iterations are never confined to the successive steps.

The Agile Manifesto

Original Signatories

Kent Beck
Mike Beedle
Arie Van Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler
James Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick
Robert C. Martin
Steve Mellor
Ken Sutherland
Dave Thomas

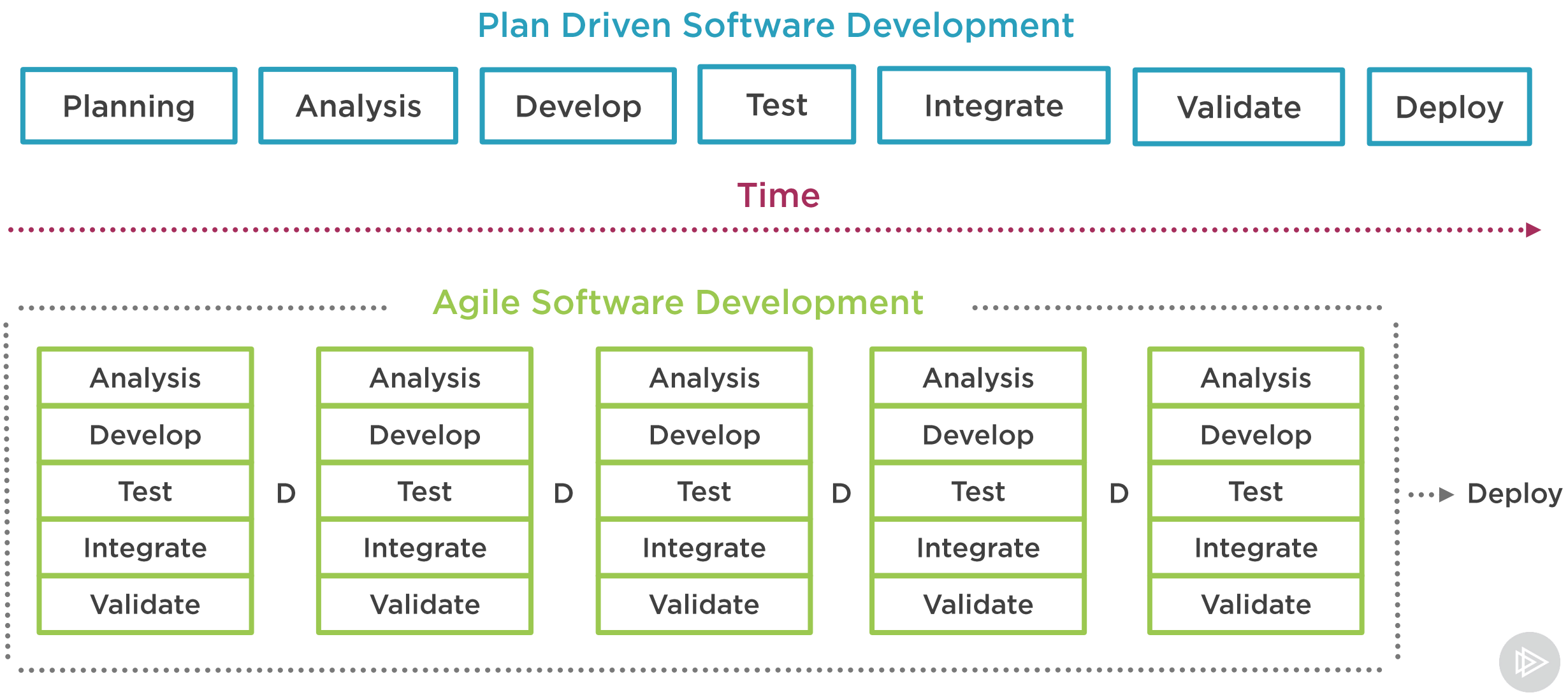
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and Interactions	Processes and Tools
Working Software	Comprehensive Documentation
Customer Collaboration	Contract Negotiation
Responding to Change	Following a Plan

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

The Big Difference



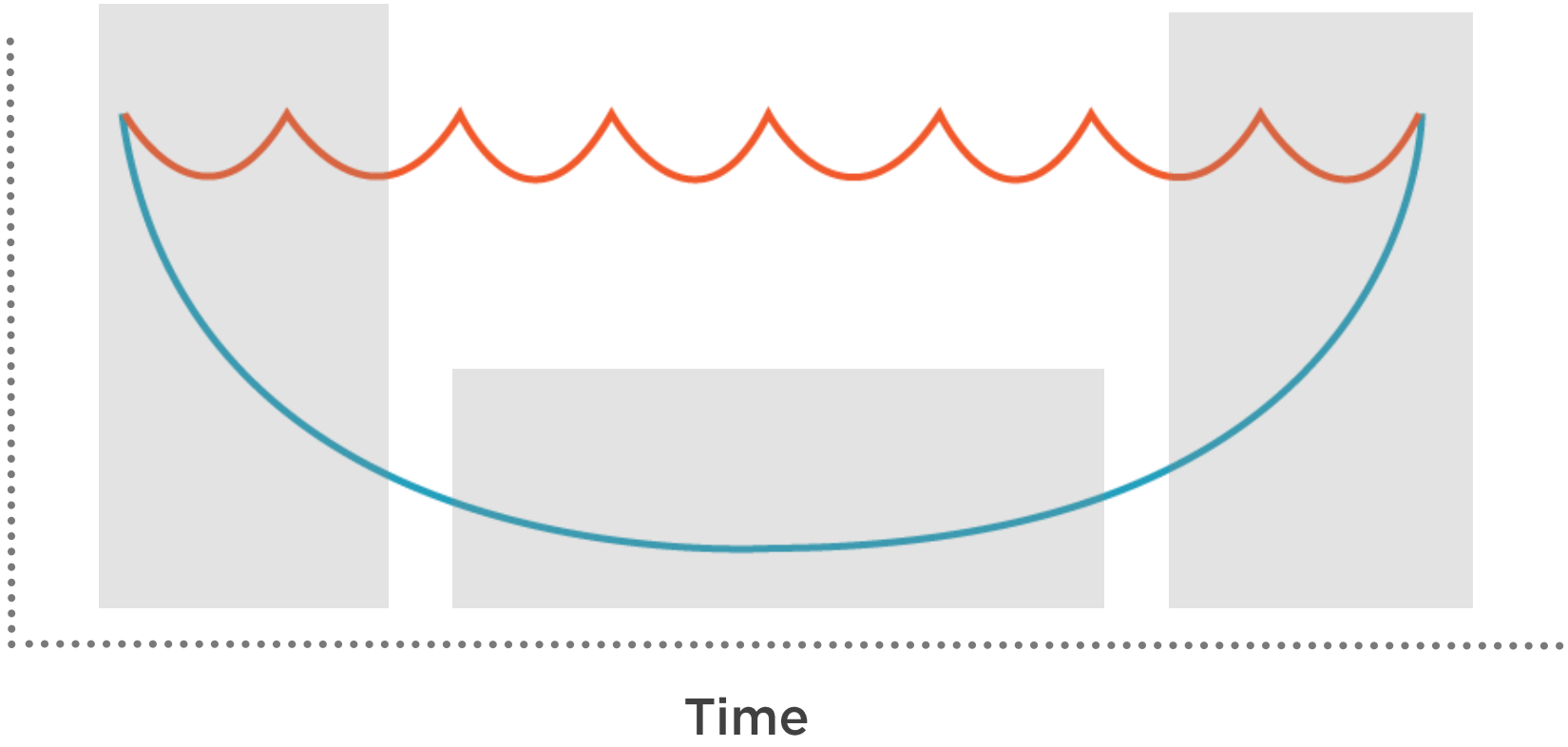
Comparing Methodologies

Plan Driven Methodologies	Agile Methodologies
Change is bad, therefore discouraged and actively controlled	Change is inevitable and valuable, therefore encouraged and embraced
Adherence to the plan determines success or failure	Incentives are often based on customer satisfaction and ROI
I am done when my part of the plan is signed off	I am done when the customer is happy

Comparing Methodologies

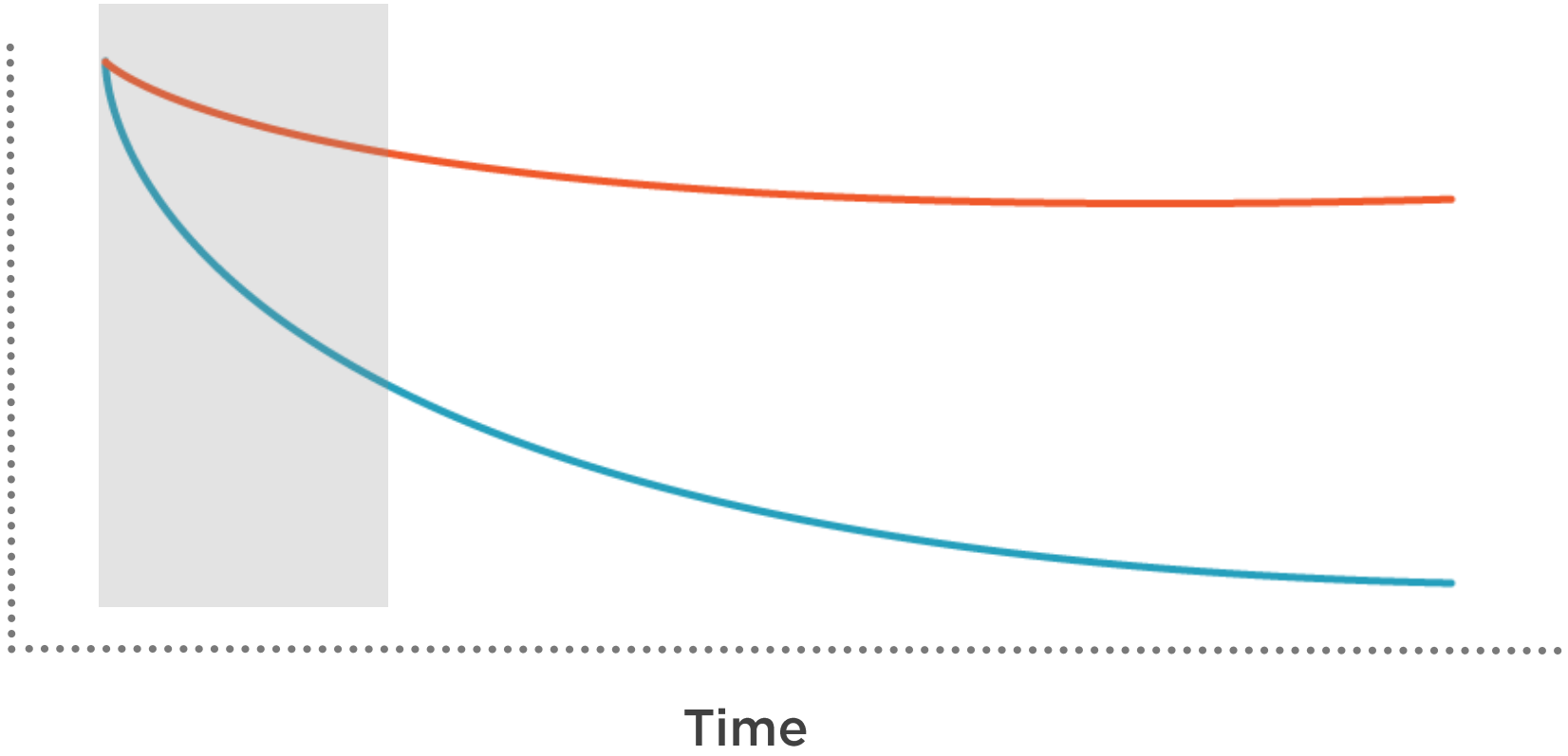
Plan Driven Methodologies	Agile Methodologies
Lots of gates to control quality	Highly iterative to achieve quality
Inspect product when it is complete	Inspect work as it is being done
Start by predicting what will be delivered	Start with a goal of filling a need

Visibility

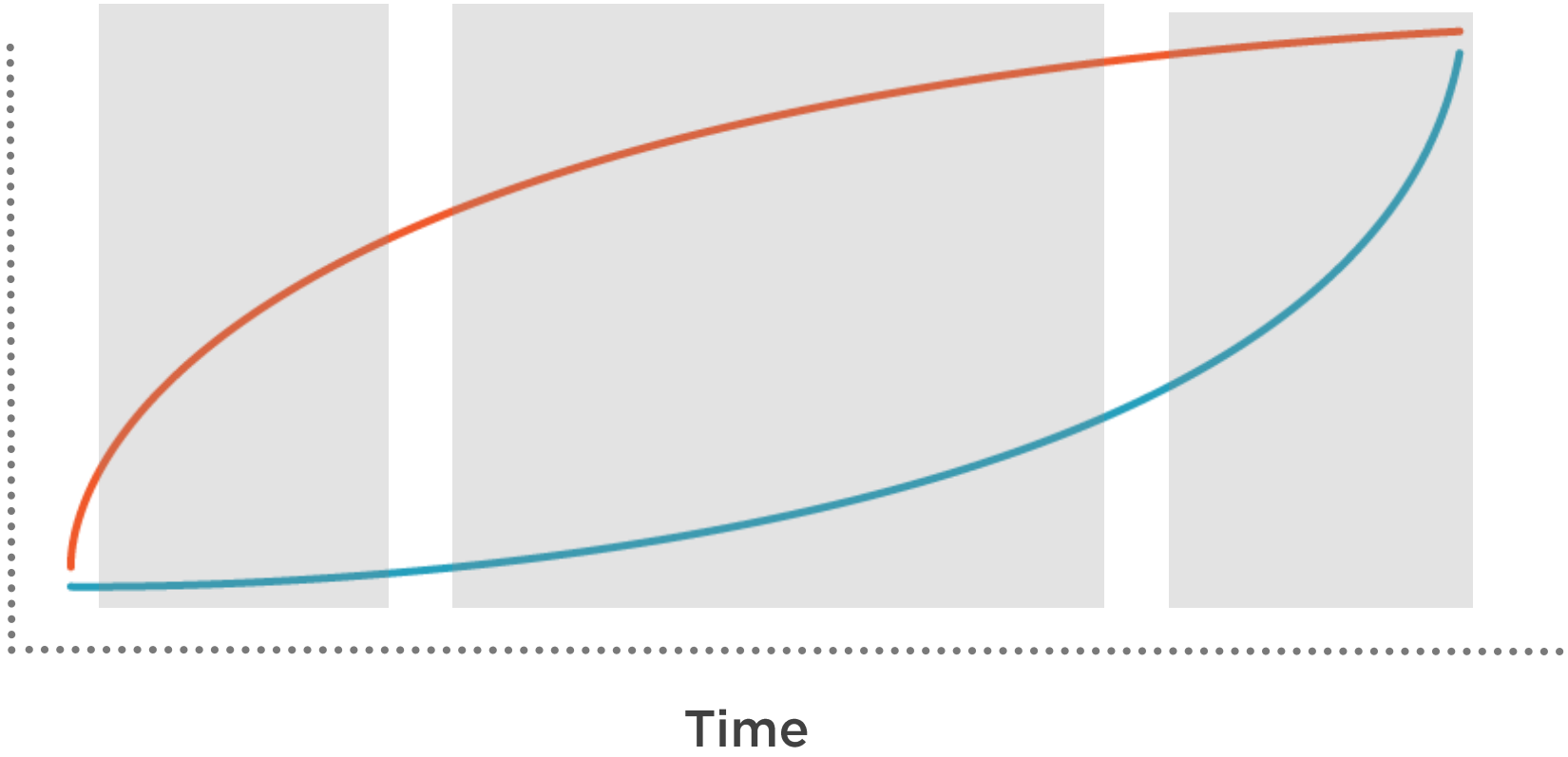


Ability to Change

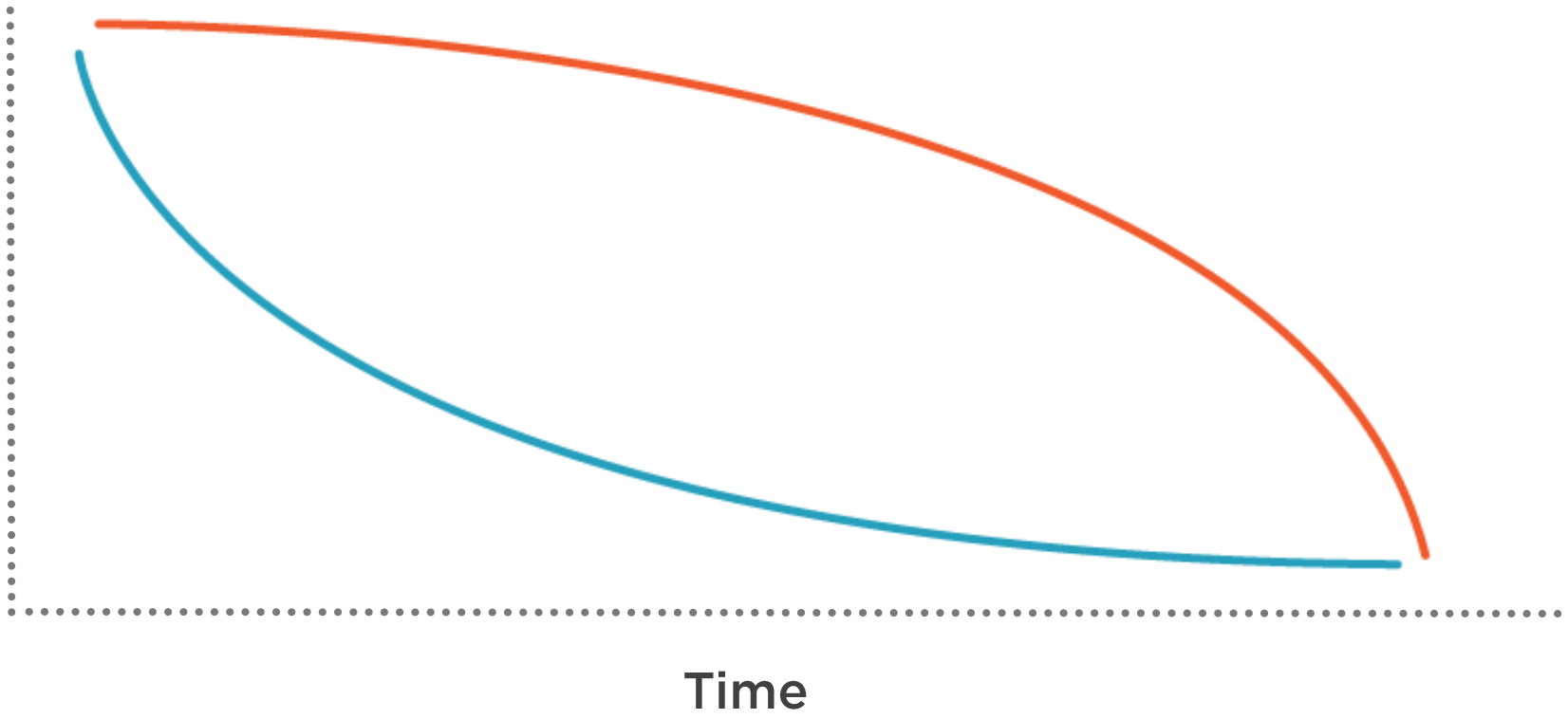
This is true of software produced and business decisions made



Business Value



Risk



What Agile is and is not

Dispelling Common Myths

Touchstones of Agility

Come back to these things to recall what we are trying to accomplish.

Documents	Principles	
The Manifesto	Iterative Delivery	Transparency and Openness
The Declaration for Interdependence	Frequent Feedback	Lean Principles

The Two Parts of the Agile Discussion

Processes and Methodologies

Working with People

- Planning
- Teamwork
- Engaging customers
- Providing leadership
- Collaboration
- Learning

Techniques and Practices

Working with Software

- Design
- Coding
- Testing
- Deploying
- User experience

Processes and Methodologies

Working with People

- Planning
- Teamwork
- Engaging customers
- Providing leadership
- Collaboration
- Learning

The Two Parts of the Agile Discussion

Processes & Methodologies

- Adaptive Software Development
- Agile Modeling
- Agile Unified Process
- Crystal Clear
- Dynamic Systems Development Method
- eXtreme Programming
- Feature Driven Development
- Lean Driven Development
- Scrum
- User Stories

Techniques & Practices

- Automated Regression
- Behavior Driven Development
- Continuous Integration
- Design Patterns
- Domain Driven Design
- DRY, SoC, Reuse, Testability
- Incremental Design
- JIT Architecture
- Seperation of Concerns
- Pair Programming
- Refractoring
- Test Driven Development

Agile Is

Iterative

Adaptive

Value Based

Easy to Understand

Hard to Implement

Agile Is Not

Just about writing code

Undisciplined

Unstructured

Whatever you want it to be

A placebo for pesky developers

Contemporary Agile Methodologies

Proven Practices

Extreme Programming (XP)

- Ancestor of most Agile methodologies
- Originated with Kent Beck in 1999
- Blends processes and practices
- Found success in smaller teams
- XP became controversial early on due to advocates teaching it as dogma

Extreme Programming (XP)

1. Take observed effective team practices 2. Push them to extreme levels

Good Practice	Pushed to the Extreme
Code Reviews	Pair Programming
Testing	TDD and constant regression
Software Design	Relentless Refactoring
Simplicity	The simplest thing that could work
Integration Testing	Continuous Integration
Short Iterations	The Planning Game

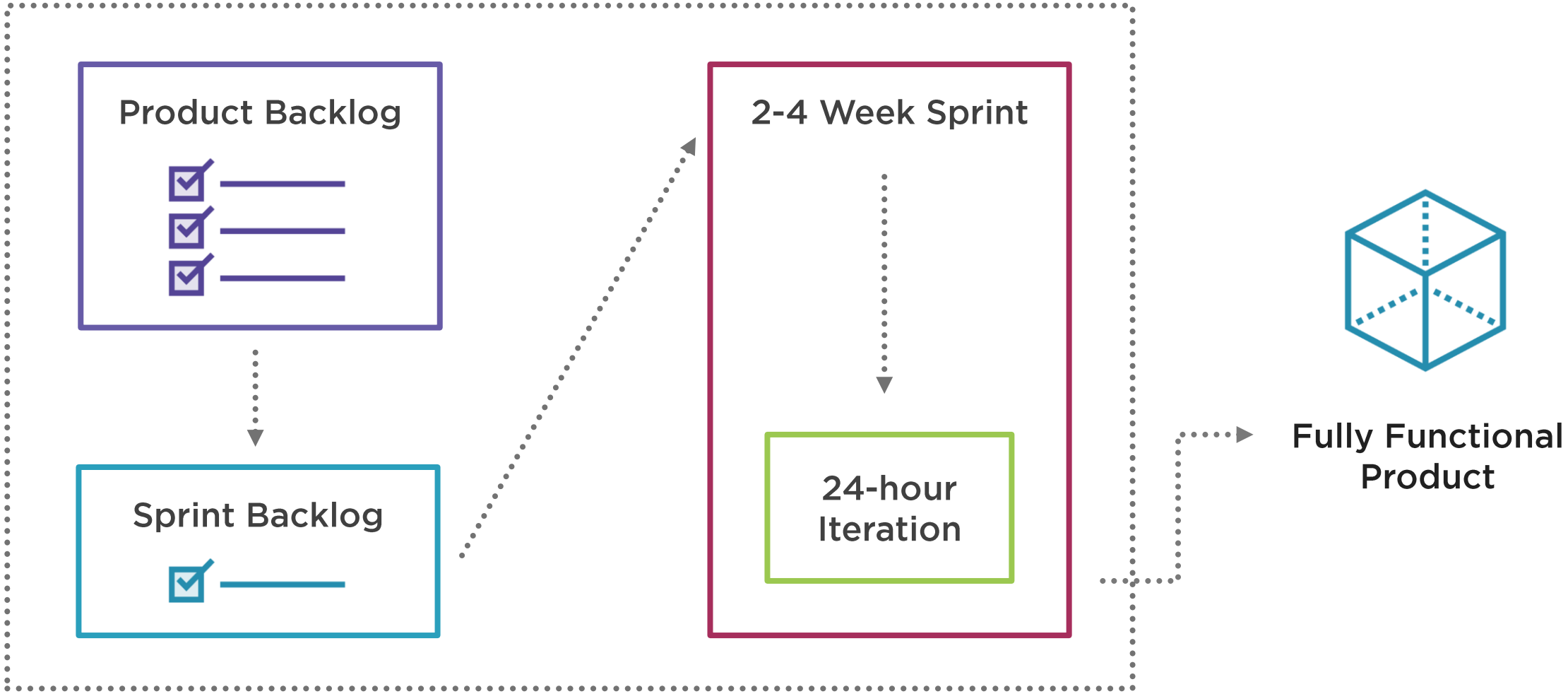
XP's 12 Practices

1. The Planning Game
2. Small Releases
3. Metaphor
4. Simple Design
5. Testing
6. Refactoring
7. Pair Programming
8. Continuous Integration
9. Collective Ownership
10. On-site Customer
11. The 40-hour Week
12. Coding Standards

Scrum

- An iterative project management process
- Originated with Ken Schwaber and Jeff Sutherland in 1990s
- Does not advocate any specific engineering techniques
- Used beyond software development
- Blends well with XP development
- Simple

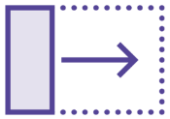
Scrum


Product Backlog
Sprint Backlog
2-4 Week Sprint
24-hour Iteration
Fully Functional Product

Lean Software Development

More a set of guidelines than a formal methodology

Originally applied in manufacturing, now used in software development



Can be applied to improve any process

Focuses on Continuous Improvement (Kaizen) and value flow

Principles of Lean Software Development

Eliminate waste

Amplify learning

Respect people

Build Quality

Defer commitment

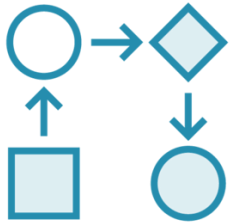
Deliver Fast

Recognize and optimize the whole

Feature Driven Development

Originated with Jeff De Luca while working on a 50-person project at Singapore bank in 1997 and adheres to strict processes.

Based on time-honored engineering practices



Advocates modeling as the base currency of process

Some reports claim FDD scales more effectively than Scrum

Feature Driven Development

Activities

1. Develop Overall Model
2. Build Feature List
3. Plan by Feature
4. Design by Feature
5. Build by Feature

Milestones

1. Domain Walkthrough
2. Design
3. Design Inspection
4. Code
5. Code Inspection
6. Promote to Build

<action> <result> <object>

Sum the total for monthly sales

Validate the password of the user

Feature Driven Development

Activities

1. Develop Overall Model
2. Build Feature List
3. Plan by Feature
4. Design by Feature
5. Build by Feature

Milestones

1. Domain Walkthrough
2. Design
3. Design Inspection
4. Code
5. Code Inspection
6. Promote to Build

FDD Practices

Domain Object Modeling

Developing by Feature

Individual Code Ownership

Feature Teams

FDD Practices

Inspections

Configuration Management

Regular Builds

High Visibility Progress and Results

Summary

Agile is a set of Principles

- Individuals and interactions **over** processes and tools
- Working software **over** comprehensive documentation
- Customer collaboration **over** contract negotiation
- Responding to change **over** following a plan

Agile is not a distinct process or methodology, though it has distinct attributes

- Frequent and iterative product delivery
- Adaptive and responsive to feedback
- Close customer relationships
- Highly transparent

References

The Manifesto for Agile Software Development
AgileManifesto.org

Agile Alliance
AgileAlliance.org