Construx®

Delivering Software Project Success

Applying Lean Concepts to Plan-Driven Projects

www.construx.com

VJ2.0

Executive Summary

- A one size fits all approach is flawed
- Lean applies only to “Agile” is a myth
- All software teams need to continually ensure each activity they perform is optimized to promote value and eliminate waste

Purpose

- Introduce you to applying lean thinking when circumstances require a more upfront planning approach
- Identify the lean software development principles
- Provide **examples** illustrating using lean on plan-driven projects

Lean Principles

- Build Value
- Eliminate Waste
- Build Integrity In
- Amplify Learning
- Localize Responsibility
- Delay Commitment
- Deliver Fast
- Optimize the Whole

Adapted from Mary and Tom Poppendieck, *Lean Software Development*

Principles of Lean Software Development

- Precisely specify the ***value*** of each project
- Identify the ***value stream*** for each project
- Allow value to ***flow*** without interruptions
- Let the customer ***pull*** value from the project team
- Continuously pursue ***perfection***

Ronald Mascitelli, *How to Slash Waste and Boost Profits Through Lean Project Management*

Construx®

Delivering Software Project Success

Eliminate Waste on a Plan-Driven Project

What is Waste?

Anything the customer would not agree to pay for

Ronald Mascitelli

Anything that does not add customer value

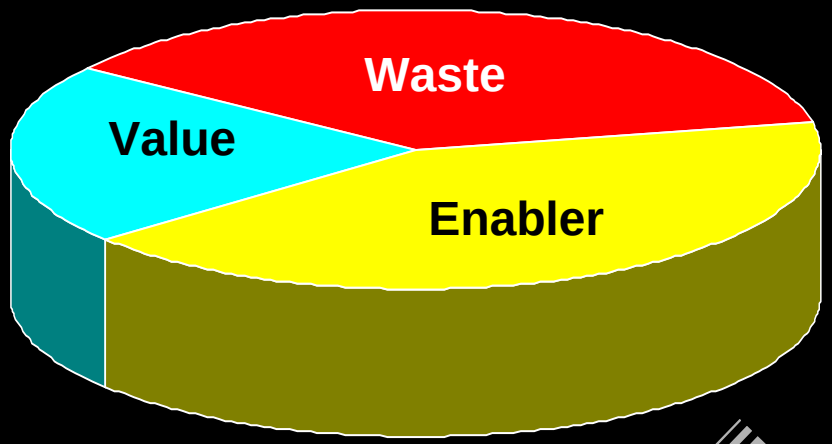
Mary and Tom Poppendieck,

What is Value?

Any activity or task is ***value-added*** if it transforms a new product design (or the essential deliverables needed to produce it) in such a way that the customer is both aware of it and willing to pay for it.

Ronald Mascitelli

Three Categories of Activities

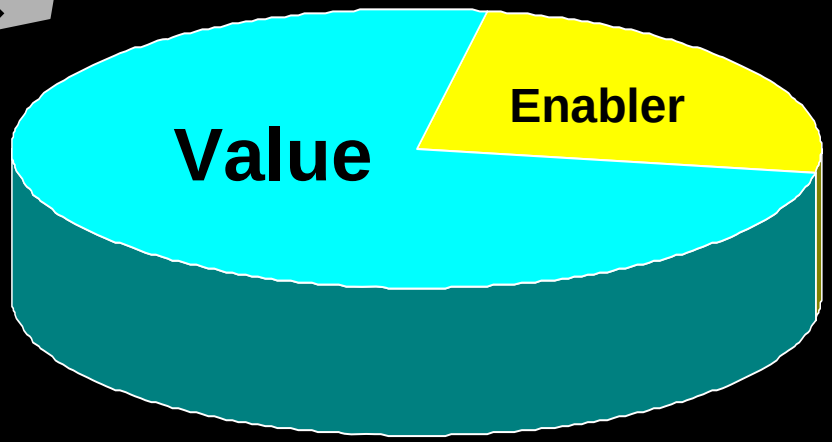


Lean's goal

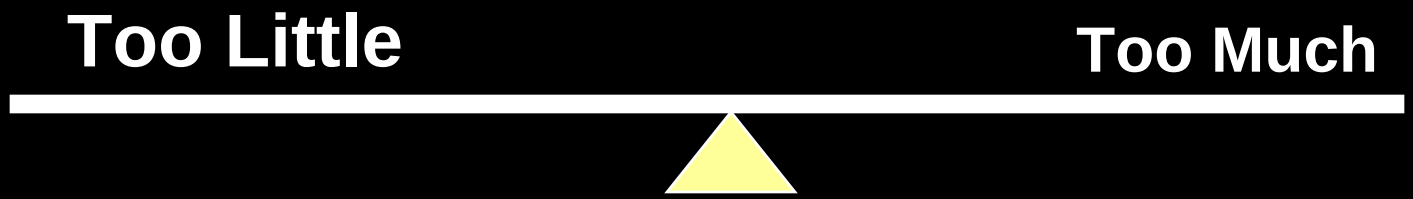
- Eliminate waste
- Reduce effort spent on enablers
- Increase value

Leads to:

- Increased development capacity
- Shorter schedules



Too Little or Too Much Leads to Waste



- Missing important tasks or information
- Error-prone
- Causes confusion, delays, and wasted work

- Includes non-value added information or tasks (aka Scrap)
- Have to filter out the noise
- Impedes efficiency, creativity, and innovation

Identifying Waste

- Review Value Stream
  - The sequence of activities that create project deliverables
  - What activities can be deleted? Streamlined? Beefed up?
  - Where are the bottlenecks?
  - Is there excessive wait-time?
- Ask your team
  - They know!
  - Can you justify each activity and deliverable?

Common Sources of Waste

- Too many projects
- Unnecessary requirements
- Random prioritization
- Inefficient meetings & status reporting
- Unrealistic schedules
- Unnecessary documentation
- Multi-tasking
- Interruptions
- Dysfunctional reviews
- Excessive wait-states
- Insufficient resources

What are common sources of waste on your projects?

Focus on the Bottlenecks

Know where your system's bottlenecks are, and make all other decisions revolve around their limitations.

Eliyahu Goldratt,
The Goal

Avoid non-value added work

- Every activity and deliverable (both what is done and how formally it is done) needs to do at least one of…
  - Help the project satisfy its charter
  - Help control a risk
  - Help maximize an asset
- Otherwise, do it less formally or don't do it at all

Construx®

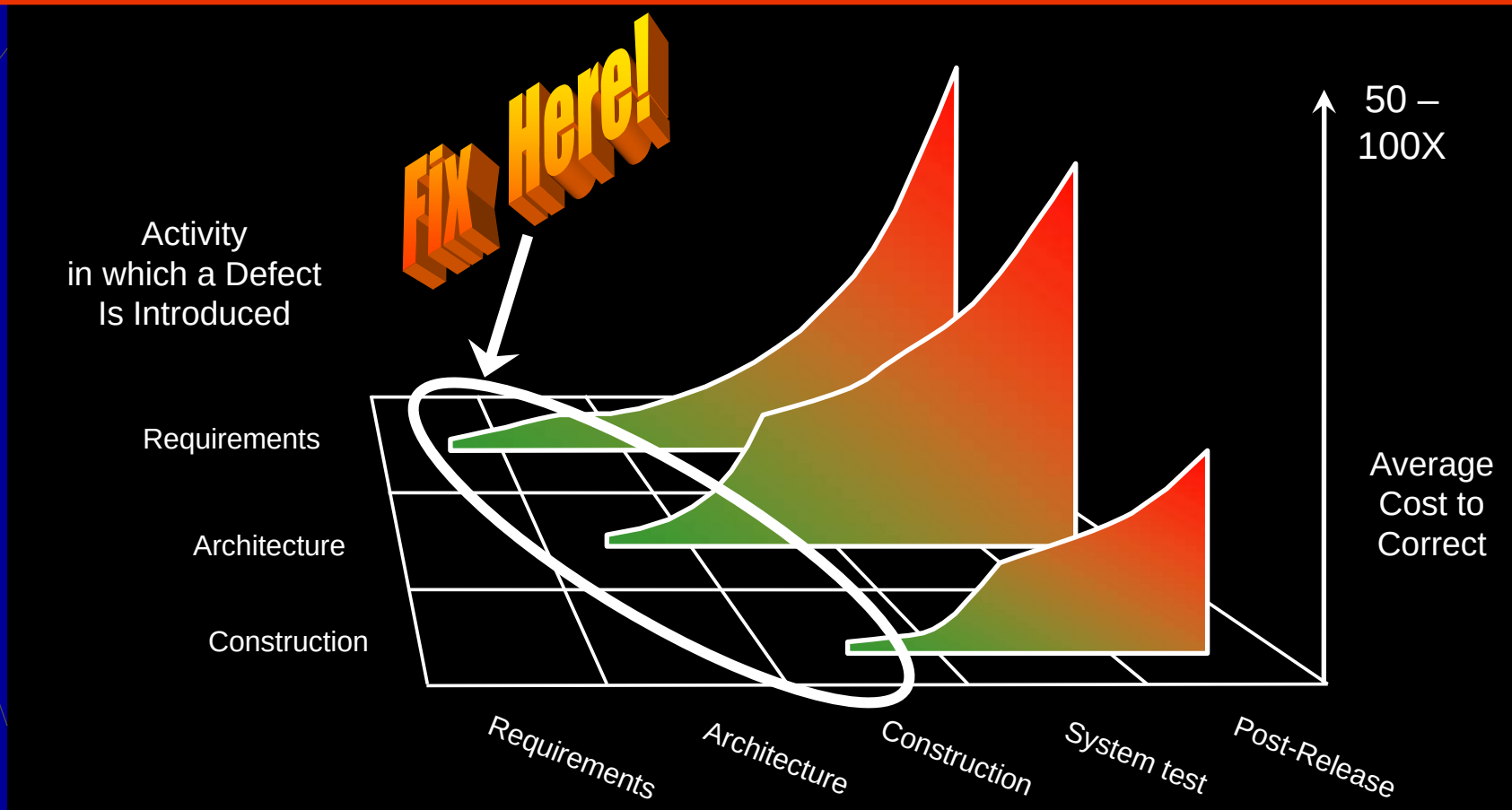
Delivering Software Project Success

Build Integrity In on a Plan-Driven Project

Quality as an Enabler

- Focus on quality reduces effort and shortens schedules
- 40-50% of the effort on typical software projects is spent on avoidable rework
- Every hour of upstream review saves up to ten hours of downstream work

Find Early—Fix Quickly



Steve McConnell, *Code Complete, 2nd Edition*

Use a Combination of Techniques

- Prevention
  - Culture
  - Professional development
  - Toolbox
  - Checklists and templates
  - Audits
  - Quality gates
  - Team structure
  - Continuous process improvement
- Detection
  - Reviews
  - Testing
  - Simulations
  - Real Use
  - Automated
  - Mathematical

Construx®

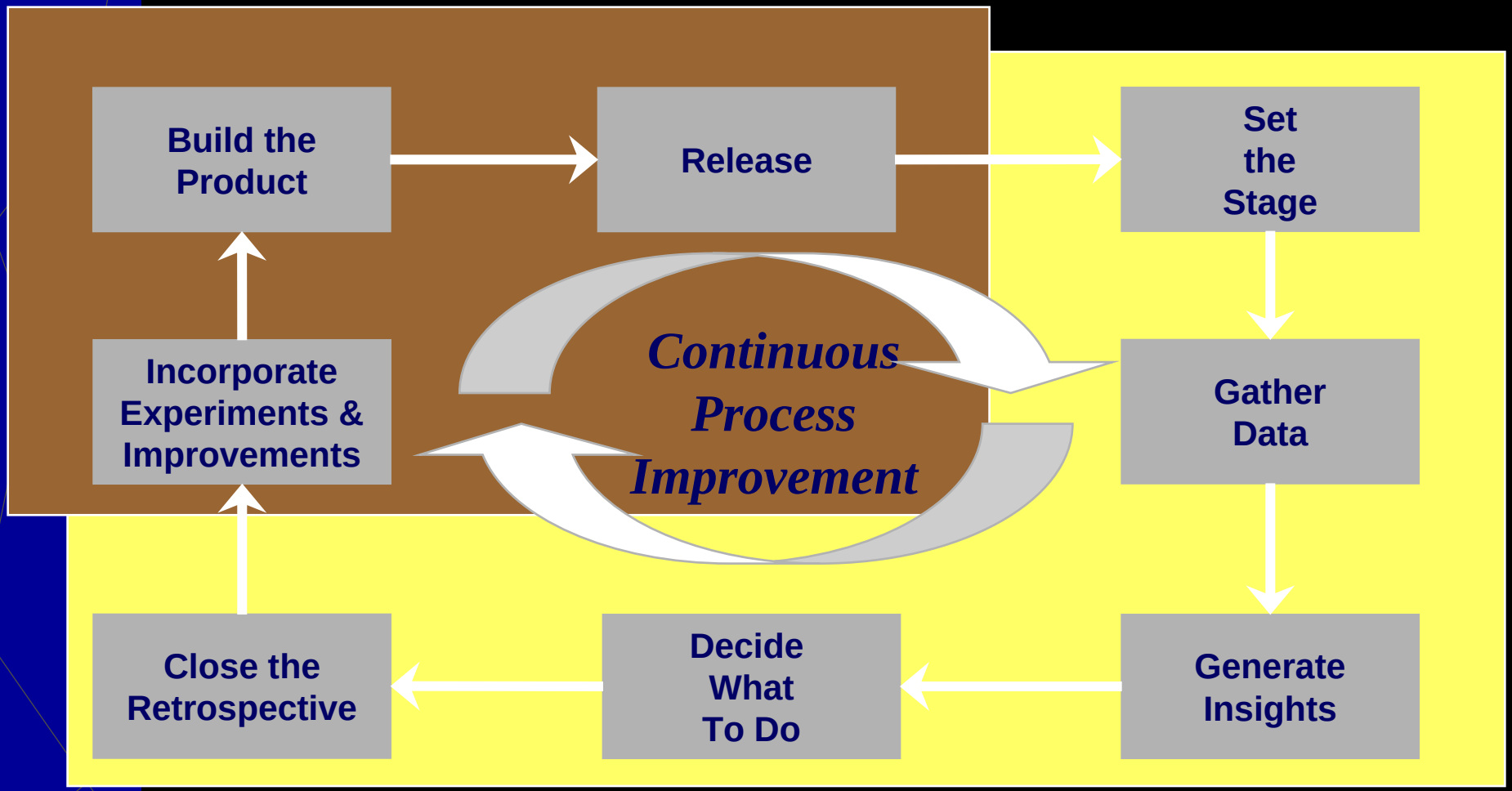
Delivering Software Project Success

Amplify Learning on a Plan-Driven Project

Interim Retrospectives

- Meet with the team to discuss successes and failures observed during the milestone
  - What did we originally think would happen?
  - What actually did happen?
  - Based on what we know today, if we were able to start over
    - What would we want to be sure to do different?
    - What would we want to be sure to do the same?
- Should the development process be changed for the next milestone?
  - Don't wait for an end-of-project retrospective

Leads to Continuous Improvement



Adapted from: Esther Derby and Diana Larsen, *Agile Retrospectives*

Construx®

Delivering Software Project Success

Localize Responsibility on a Plan-Driven Project

Empower teams

- Clearly defined roles, responsibilities, and authorities
- Push decision making as low as practical

Construx®

Delivering Software Project Success

Delay Commitment on a Plan-Driven Project

Delaying Techniques

- Focus on the process goals and intentions
  - What you need to do, not how
  - Make decisions based on coarser grained data
- Stage Freezing
  - Freeze the broad-level essentials early
  - Freeze the details later
- Last Responsible Moment
  - Ok not to know yet
  - But know when you got to know

Rolling Wave Planning

A progressive detailing of the project plan by providing the details of the work to be done in the current project phase but also providing some preliminary description of work to be done in later project phases.

Gregory Githens,
Rolling Wave Planning

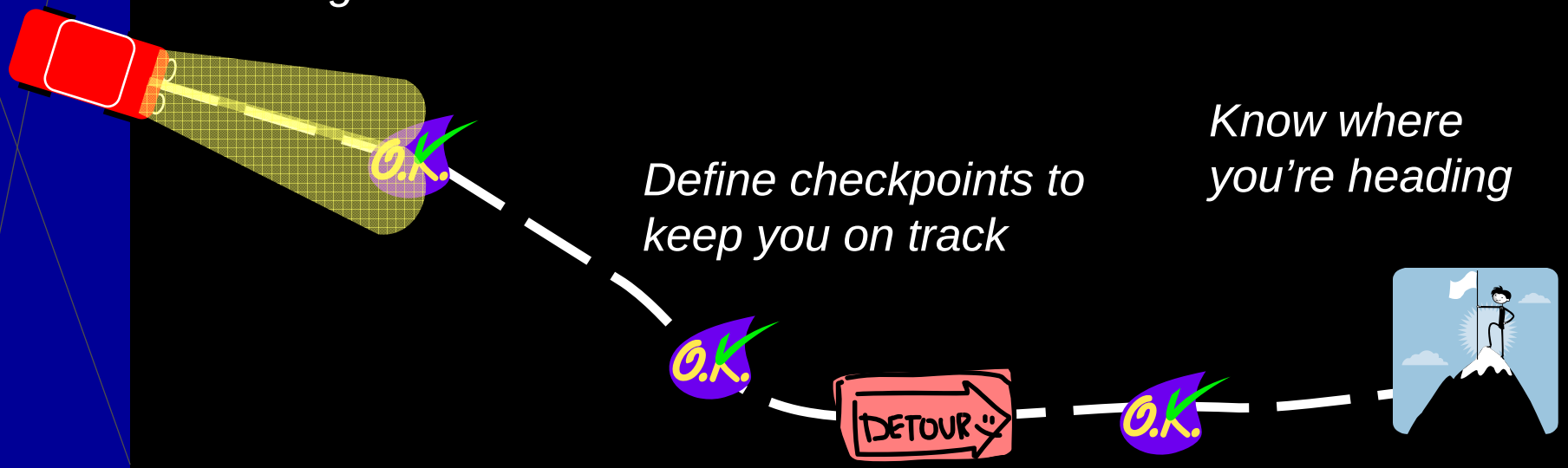
Rolling Wave Planning

Drive within your headlights

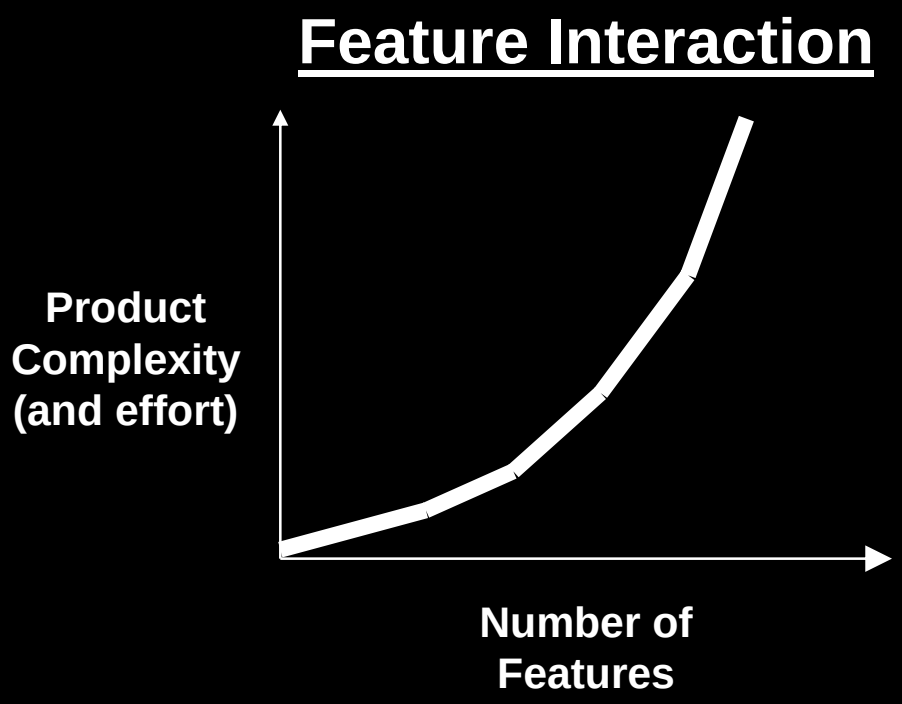
Define checkpoints to keep you on track

Know where you're heading

Make mid-course corrections when needed



Get to “NO” Quickly



- Complexity increases faster than the number of features
- Fewer features will
  - Be easier to build
  - Easier to test
  - Have less risk
- Scrub early and often
  - ~ 64% of features are rarely or never used *
  - Best case – Scrub non-value added projects before they even start!

* The Standish Group, *Extreme Chaos*

Construx®

Delivering Software Project Success

Deliver Fast on a Plan-Driven Project

Frequent Releases

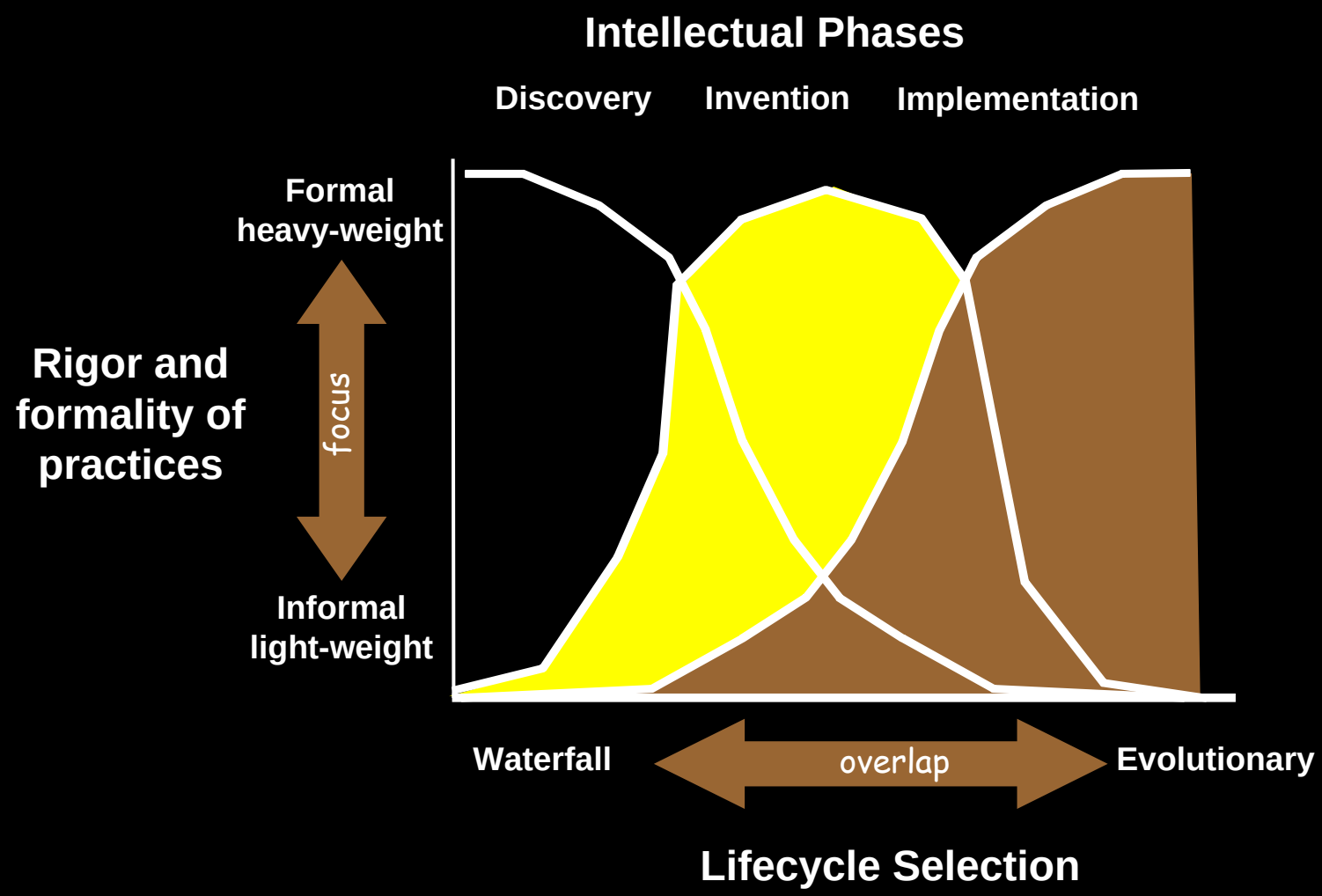
- Define releases to be no longer than 6 weeks in duration
  - OK to define interim releases that are not released outside development
  - Forces frequent convergence
  - Can be used for coarse level planning – allows you to handle fine-grain dependencies at team level
- Overall, a clear industry best practice, reduces numerous common risks—virtually always valuable

Construx®

Delivering Software Project Success

Optimize the Whole on a Plan-Driven Project

Conscience Selection of Practices



Adjust rigor of practices

	Discovery	*Invention*	*Implementation*
Low Rigor	Requirements Spec Informal Requirements Reviews	Design Document Informal Design Reviews	System Testing Informal code reviews
High Rigor	Requirements Spec Requirements Inspections User Interface Prototype Use Cases Usability Studies Incremental Delivery More Senior Requirements Developers	Architecture and Detailed Design Docs Design Inspections Proof of Concept Prototypes Outside Reviewers Incremental Delivery More Senior Designers	Automated Testing Full Regression Testing System Testing Formal Code Inspections Use of a Standard Integration Procedure Daily Builds

Myth: Overhead is Waste
Reality: It's an Enabler

Typical Outcome

Planned Work

Unplanned Work

(Missing Deliverables, Missing Tasks, Avoidable Rework, Unforeseen Risks)

Realistic Outcome

Planned Work + Overhead

(Quality Reviews, Project Management, Unavoidable Rework, Corrective Activities)

Unplanned Work

Summary

- Our teams are a constrained resource
- We must continually ensure each activity is optimized to promote value and eliminate waste
- We only had time to provide examples
  - Hopefully they illustrated that Lean Principles applies to your plan driven projects

Construx®

Delivering Software Project Success

Any questions?

Closing thoughts

Simple, clear purpose and principles give rise to complex intelligent behavior. Complex rules and regulations give rise to simple stupid behavior

Attributed to: Dee Hock,
Founder and former CEO Visa
Credit Card Association

Don’t do something stupid just because it’s written down.

Attributed to: Frank Marshall,
Former VP of Engineering, CISCO

References

- Barry Boehm & Richard Turner, *Balancing Agility and Discipline: A Guide for the Perplexed* (Addison-Wesley Professional, 2003)
- Barry Boehm & Victor Basili, “Software Defect Reduction Top-10 List”. Available at National Science Foundation, Center for Empirically Based Software Engineering. http://www.cebase.org
- Esther Derby and Diana Larsen, *Agile Retrospectives: Making Good Teams Great* (Pragmatic Bookshelf, 2006)
- Gregory Githens, “Rolling Wave Project Planning,” PMI Symposia Proceedings '98NPD Track. Available at: http://www.catalystpm.com/NP02.PDF

References

- Eliyahu M. Goldratt & Jeff Cox *The Goal: A Process of Ongoing Improvement* (North River Press, 3rd Edition, 2004)
- Ronald Mascitelli, *Building a Project-Driven Enterprise: How to Slash Waste and Boost Profits Through Lean Project Management* (Project Management Institute, 2002)
- Steve McConnell, *Software Project Survival Guide* (Microsoft Press, 1998)
- Mary and Tom Poppendieck, *Lean Software Development: An Agile Toolkit for Software Development Managers* (Addison Wesley, 2003)
- Standish Group, "Extreme CHAOS", 2001 Available at: http://www.standishgroup.com/sample_research/PDFpages/extreme_chaos.pdf

Construx®

Delivering Software Project Success

- Training
- Consulting
- Tools

sales@construx.com
www.construx.com
+1 (425) 636-0100

Construx®

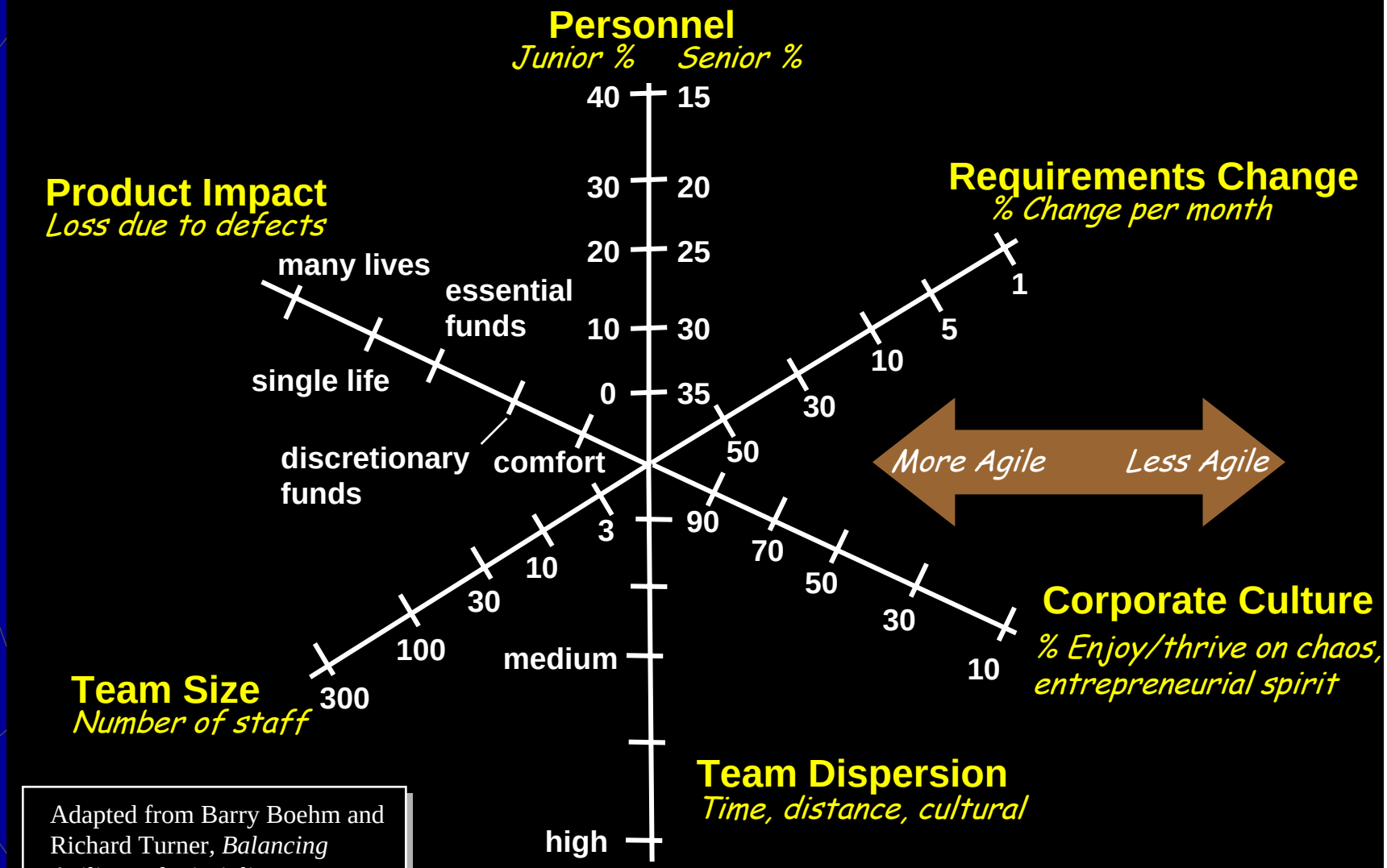
Delivering Software Project Success

Extra Material

One size doesn't fit all

- Software engineering is a multi-faceted discipline using many techniques and tools
- A one size fits all approach is flawed
  - Doesn't fit all end-product goals
  - Doesn't fit all product-lifecycle goals
  - Doesn't fit all project goals

Plan-Driven May Be Appropriate For Your Project



Adapted from Barry Boehm and Richard Turner, *Balancing Agility and Discipline*

Create Realistic Schedules