StickyMinds.com and *Better Software* magazine presents…

Using Lean Thinking to Improve Agile Testing:

The New Role of QA

Sponsored by SQE Training

Non-streaming participants should call 1-866-761-8643

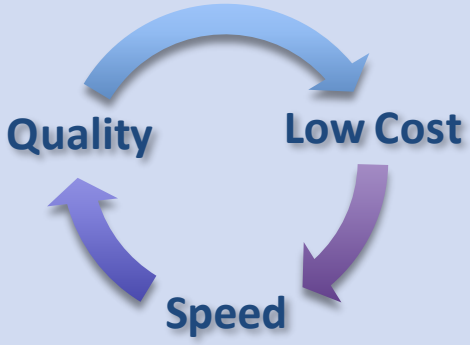
International Non-streaming participants should call 1-904-596-2362

Using Lean Thinking to Improve Agile Testing

The New Role of QA

Net Objectives

info@netobjectives.com
www.netobjectives.com



Net Objectives: Who We Are

Vision	Effective software development without suffering
Mission	To assist companies in maximizing the business value returned from their efforts in software development and maintenance. We do this by providing training, coaching, and consulting that directly assists and empowers our customers to create and sustain this ability
Services	Training in sustainable product development Assessments Lean-Agile coaching and mentoring
Expertise	Lean Software Development Agile Methods (Scrum, XP, RUP) Agile Analysis Design Patterns Test-Driven Development / Quality Assurance

Net Objectives

Alan Shalloway

alshall
@netobjectives.com

- CEO and Founder, Net Objectives
- Courses delivered:
  - Lean Software Development
  - CSM
  - Agile Analysis
  - Design Patterns Explained
  - Advanced Software Design
  - Effective Object-Orientation Analysis and Design
- Coach in all areas above
- Author, Design Patterns Explained
- MS in Computer Science from M.I.T.

Lean ain't mean. Lean is the key to "effective software development without suffering."

Poll Question 1

- To what extent are you familiar with Lean?
  - Not at all
  - Somewhat
  - Quite a bit

Poll 1 Answers

Poll Question 2

- To what extent are you familiar with Agile and/or Scrum
  - Not at all
  - Somewhat
  - Quite a bit

Poll 2 Answers

Poll Question 3

- What is your role?
  - Executive
  - First line to Mid management
  - Product / Program Manager
  - Project Manager / Scrum Master
  - Analyst
  - Developer
  - Tester
  - Other

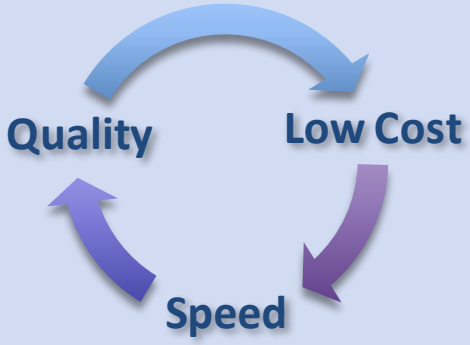
Poll 3 Answers

Lean Foundations

Paradigm of Lean

Net Objectives

info@netobjectives.com
www.netobjectives.com



Lean and Lean Software Development?

- Lean is the name given Toyota's Production and Product Development System
- Lean Software Development are Lean principles applied to software development

Principles and Practices

"Principles are underlying truths that don't change over time or space, while practices are the application of principles to a particular situation.

Practices can and should differ as you move from one environment to the next, and they also change as a situation evolves."

Mary and Tom Poppendieck . ***Implementing Lean Software Development: From Concept to Cash***, Addison-Wesley. 2006

Net Objectives

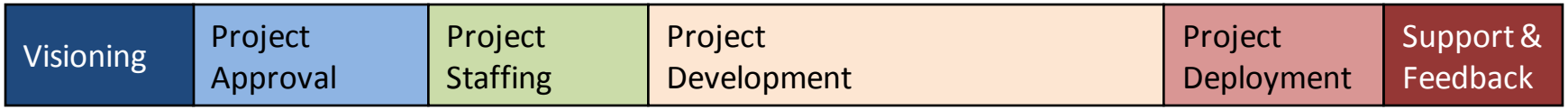
Essence of Lean

- Foundation
  - Respect People
  - System causes errors
  - Continuously Improve the Process To Support People
- Attitude
  - When errors happen fix the system – no work arounds
- Guidance
  - Focus on time, not resource utilization
  - Delays cause waste
    - From when get information until use it
    - From when an error occurs, until it is detected
- Principles
  - Optimize the whole
  - Eliminate waste
  - Build quality in
- Practices
  - Agile/Scrum Methods

Goal of Lean-Agile Software Development

- Lean Software Development enables the discovery, prioritization, and delivery of highest business value

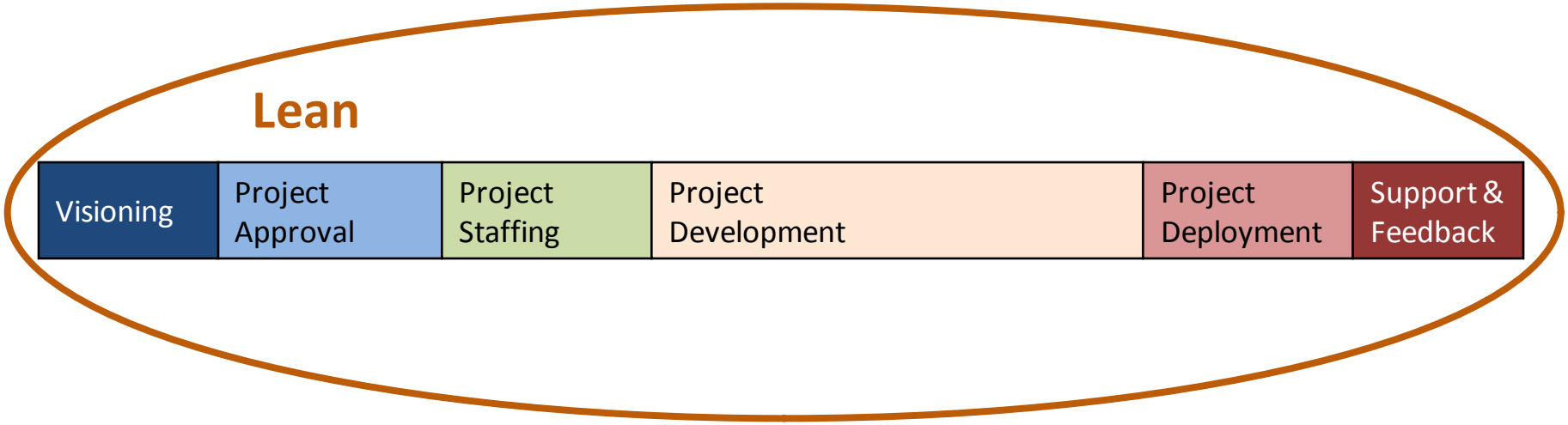
- Agile methods enable the incremental delivery of business value based on the team's capabilities



Goal of Lean-Agile Software Development

- Lean Software Development enables the discovery, prioritization, and delivery of highest business value

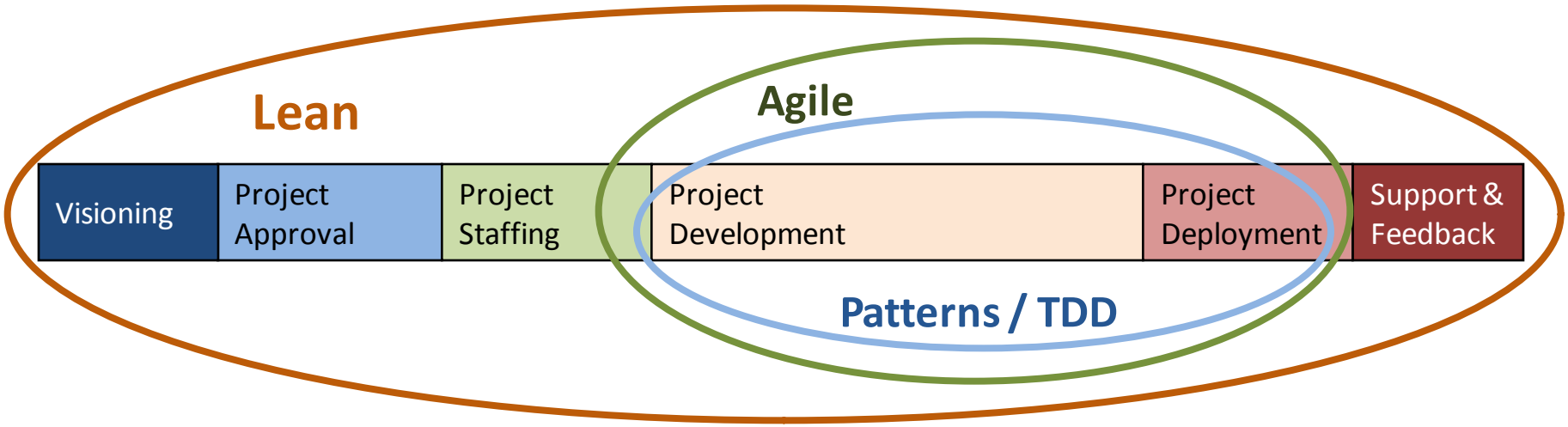
- Agile methods enable the incremental delivery of business value based on the team's capabilities



Goal of Lean-Agile Software Development

- Lean Software Development enables the discovery, prioritization, and delivery of highest business value

- Agile methods enable the incremental delivery of business value based on the team's capabilities



Lean Software Development

- Goal of delivering value to customer quickly
- Speed and quality results in low cost
- Commitment to continuous process improvement
- Process supports team
- Team adjusts the process within the context of business needs
- Management facilitates this

- *Focuses on the ability to add value quickly now, while improving the ability to add value quickly in the future*

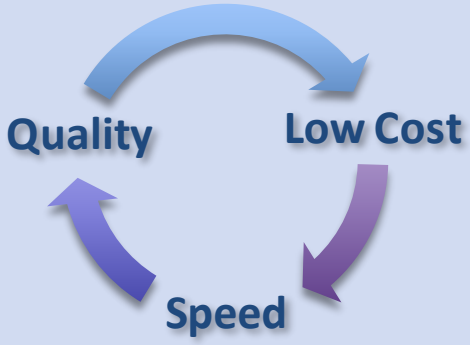
What Is Scrum?

- Scrum was originally a framework for wrapping whatever implementation process your team had
- It is guided by:
  - Develop in iterations to:
    - Add value quickly to the customer
    - Provide feedback about what the customer needs
    - Improve your development process
  - Remove impediments to your team
  - Self-motivated team figures out what to do
- Has come to include some "standard work" not originally defined in Scrum
  - XP Engineering Practices
  - Scrum-of-Scrums for team collaboration
- Scrum was inspired by Lean principles

Lean's Approach To Software Challenges

Net Objectives

info@netobjectives.com
www.netobjectives.com



The Risks of Software Development

- Building more than you need
- Building lower priority items
- Building the right thing wrong
- Poor quality of software
  - Software is buggy
  - Software is not maintainable
- Architectural risks
- Having the wrong resources
- Discovering functional needs late in the project* but being unable to build them

* Could this be a good thing?

Why Do Errors Happen in Software Development?

- Developer didn't write the code correctly
- Developer misunderstood what the customer wanted
- Customer realized later that they mis-spoke the requirement
- Customer spoke it properly, but realized they asked for the wrong thing once they saw what was delivered
- Customer spoke it properly, realized they got what they originally wanted, but they now have a better idea

What can We do About Them?

- Tell developers not to write bugs
- Implore customers to be clearer
- Find better customers
- Have people talk more clearly so we don't have miscommunications!

Work harder!

Work smarter!

What can We do About Them?

- Tell developers not to write bugs
- Implore customers to be clearer
- Find better customers
- Have people talk more clearly so we don't have miscommunications!

Work harder!

Work smarter!

Ignores that fact that people are probably already working as hard and as smart as they can!

Net Objectives

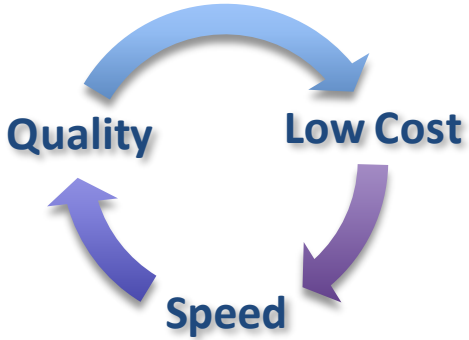
Change the Process

- Developer didn't write the code correctly
  - Write Unit Tests
- Developer misunderstood what the customer wanted
  - Specify acceptance tests early
- Customer realized later that they mis-spoke the requirement
  - Specify acceptance tests early
- Customer spoke it properly, but realized they asked for the wrong thing once they saw what was delivered
  - Don't take a long time to show customers what you've done
- Customer spoke it properly, realized they got what they originally wanted, but they now have a better idea
  - Don't take a long time to show customers what you've done

NetObjectives

Optimize the Whole

- Eliminate Waste
- Build Quality In
- Deliver Fast
- Defer Commitment
- Respect People
- Create Knowledge



Principle: Optimize the Whole

Vicious Cycle #1:

1. A customer wants some new features yesterday
2. Developers hear: Get it done fast, at all costs!
3. Result: Sloppy changes are made to the code base
4. Result: Complexity of code base increases
5. Result: Number of defects in code base increases
6. Result: Exponential increase in time to add features

Principle: Optimize the Whole

Vicious Cycle #1:

1. A customer wants some new features yesterday
2. Developers hear: Get it done fast, at all costs!
3. Result: Sloppy changes are made to the code base
4. Result: Complexity of code base increases
5. Result: Number of defects in code base increases
6. Result: Exponential increase in time to add features

Vicious Cycle #2:

1. Testing is overloaded with work
2. Result: Testing occurs long after coding
3. Result: Developers don't get immediate feedback
4. Result: Developers create more defects
5. Result: Testing has more work. Systems have more defects
6. Result: Feedback to developers is delayed further. Repeat cycle

Myth: Optimize Through Decomposition

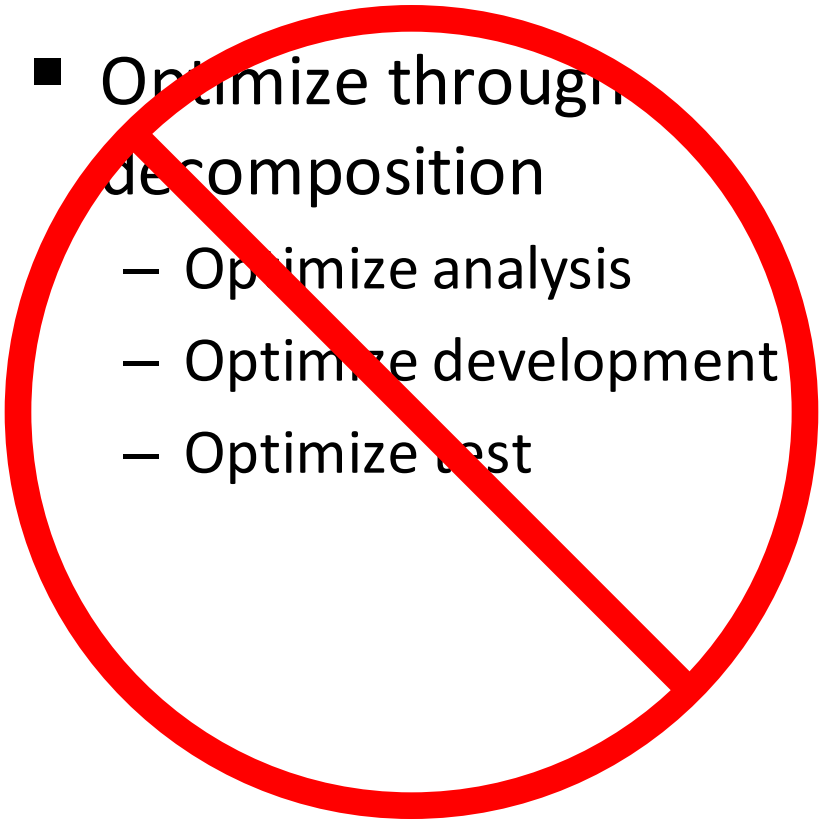
Not Useful

- Optimize through decomposition
  - Optimize analysis
  - Optimize development
  - Optimize test

Useful

- Optimize the whole
  - Have analysts, developers and testers work together to make high quality code

Myth: Optimize Through Decomposition



Not Useful

- Optimize through decomposition
  - Optimize analysis
  - Optimize development
  - Optimize test

Useful

- Optimize the whole
  - Have analysts, developers and testers work together to make high quality code

Measure **UP**

Not Useful

Span of Control

- Hold people accountable for what they can ***control***
- Measure at the individual level
- Fosters competition

Development Manager

- "We measure developers on function points delivered"

Testing Manager

- "We measure testers on bugs found"

Useful

Span of Influence

- Hold people accountable for what they can ***influence***
- Measure at the team level
- Fosters collaboration

Product Manager

- "Everyone in the company knows that their job depends upon delighting customers"

Net Objectives

Measure **UP**

Not Useful

Span of Control

- Hold people accountable for what they can ***control***
- Measure at the individual level
- Fosters competition

Development Manager

"We measure developers on function points delivered"

Testing Manager

"We measure testers on bugs found"

Useful

Span of Influence

- Hold people accountable for what they can ***influence***
- Measure at the team level
- Fosters collaboration

Product Manager

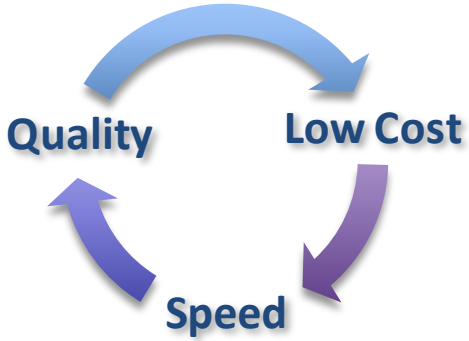
"Everyone in the company knows that their job depends upon delighting customers"

NetObjectives

- Optimize the Whole
- Eliminate Waste

Build Quality In

- Deliver Fast
- Defer Commitment
- Respect People
- Create Knowledge



Quality Is Outside and Inside

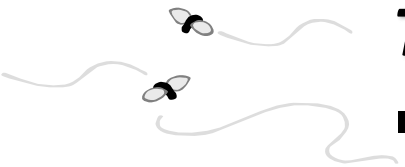
- There is both perceived integrity
  - system works the way the customer expects
- And conceptual integrity
  - system works the way the developers expect

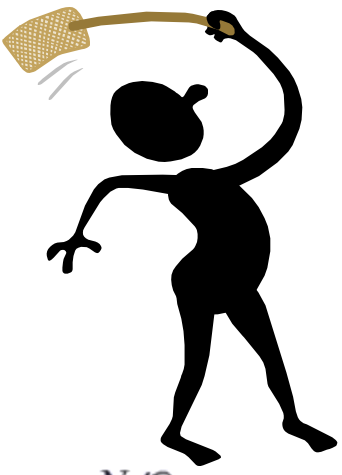
Move Inspection Forward

Two Kinds of Inspection

- Inspection to Find Defects – is WASTE
- Inspection to Prevent Defects – is Essential

If you are focused on finding defects you are not doing your job





The Role of QA

- The job of Testing is ***not*** to *find* defects
- The job of Testing is to *prevent* defects
- A quality process builds quality into the code
  - If you routinely find defects during verification
    - your process is defective

Net Objectives

Poppendieck LLC

Testing Is Validation

- We have to validate
  - That we understand what is needed
  - That we did what we wanted
  - That the product is of sufficient quality
- We must push testing up early
  - Tests improve the conversation between customers and developers
  - Tests become executable specifications

Why Test?

The job of tests, and the people that develop and run tests, is to *prevent* defects, not to find them. A quality assurance organization should champion processes which build quality into code from the start, rather than test quality in later.

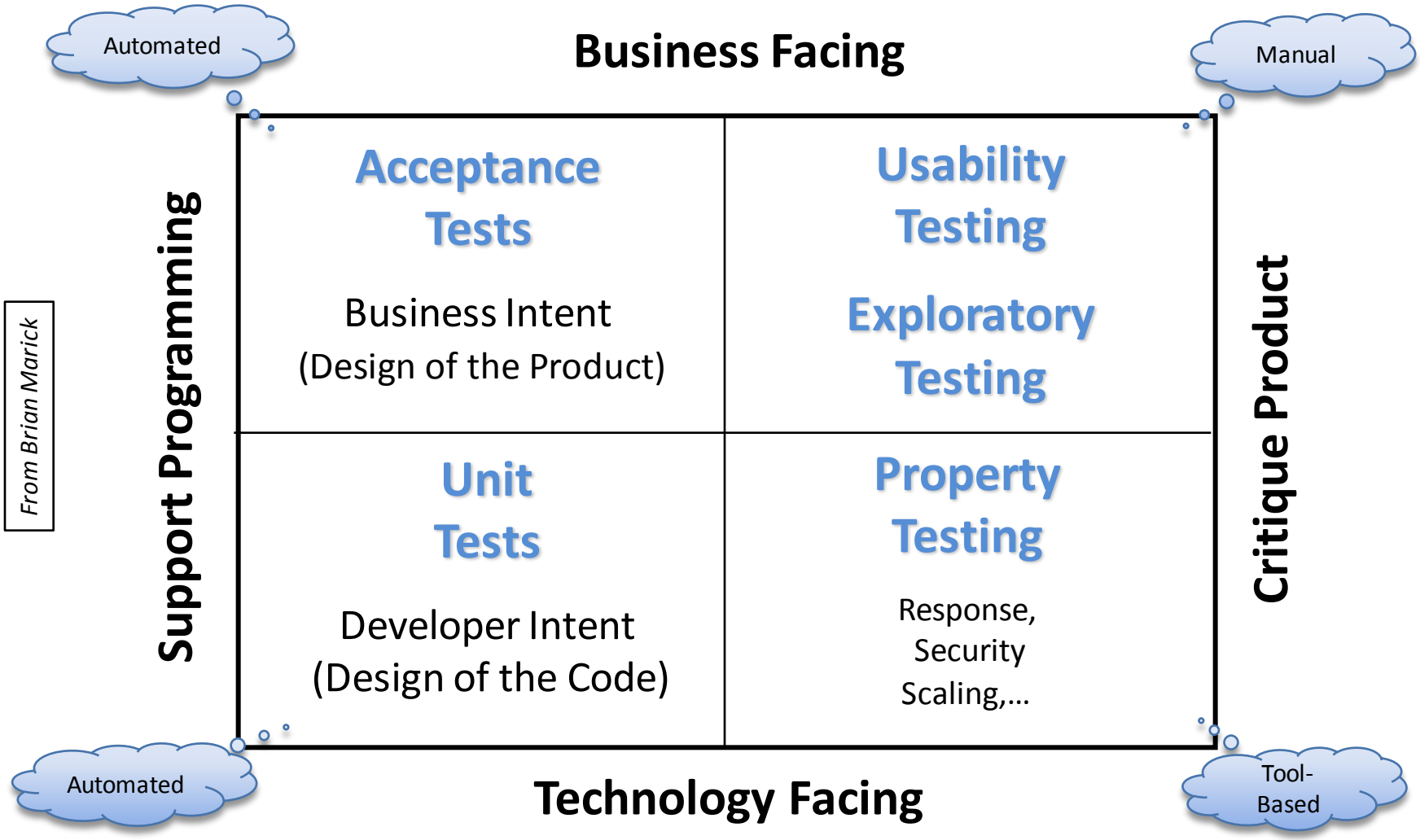
This is not to say that verification is unnecessary. Final verification is a good idea; it's just that finding defects should be the exception, not the rule, during verification.

If verification routinely triggers test-and-fix cycles, then *the development process itself* is defective.

Implementing Lean Software Development: From Concept to Cash, by Mary and Tom Poppendieck. Addison-Wesley. 2006

Net Objectives

Types of Testing



An Example of Framework For Integrated Test (FIT)

Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

An Example of Framework For Integrated Test (FIT)

Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

Payroll.Fixtures.WeeklyCompensation			
StandardHours	HolidayHours	Wage	Pay()
40	0	20	$800
45	0	20	$950
48	8	20	$1520

An Example of Framework For Integrated Test (FIT)

Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

Payroll.Fixtures.WeeklyCompensation			
StandardHours	HolidayHours	Wage	Pay()
40	0	20	$800
45	0	20	$950
48	8	20	$1520

Note the examples help in understanding.
$800 for the first line is clear.
$950 may not be clear, but it is 40 @ $20/hr + 5 @ $30/hr
The $1520 may not be clear. I could think: 40 @ $20/hr + 8 @ $30/hr + 8 * $40 = $1360. But *that is wrong.*
The 8 Holiday Hours get paid time and a half of double time. So it should be:

40 @ $20/hr + 8 @ $30/hr + 8 * $60 = $1520

When FIT Runs

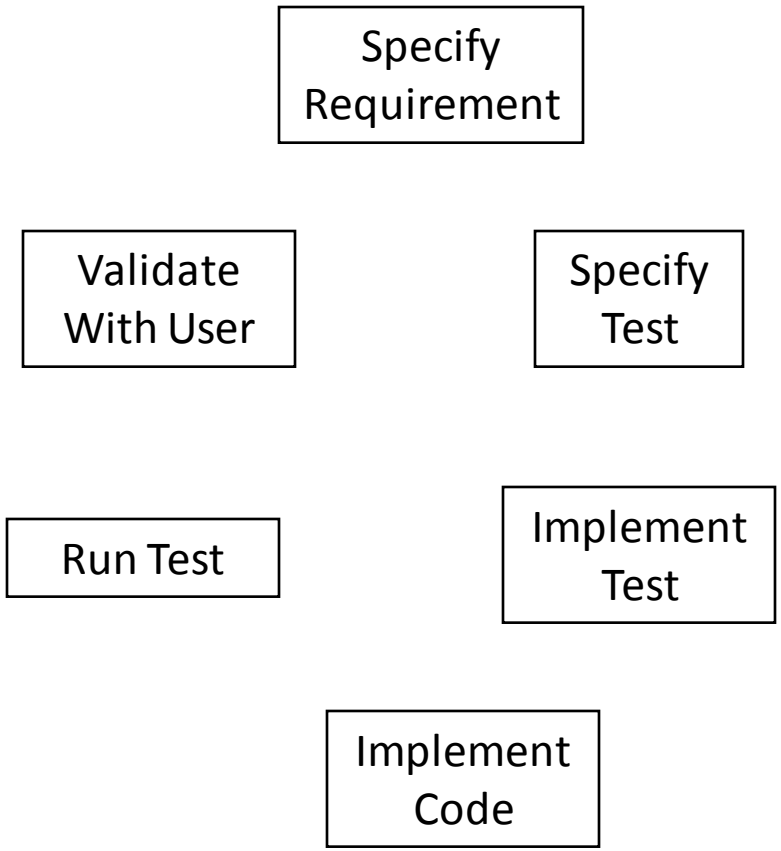
Basic Employee Compensation

For each week, hourly employees are paid a standard wage per hour for the first 40 hours worked, 1.5 times their wage for each hour after the first 40 hours, and 2 times their wage for each hour worked on Sundays and holidays.

Here are some typical examples of this:

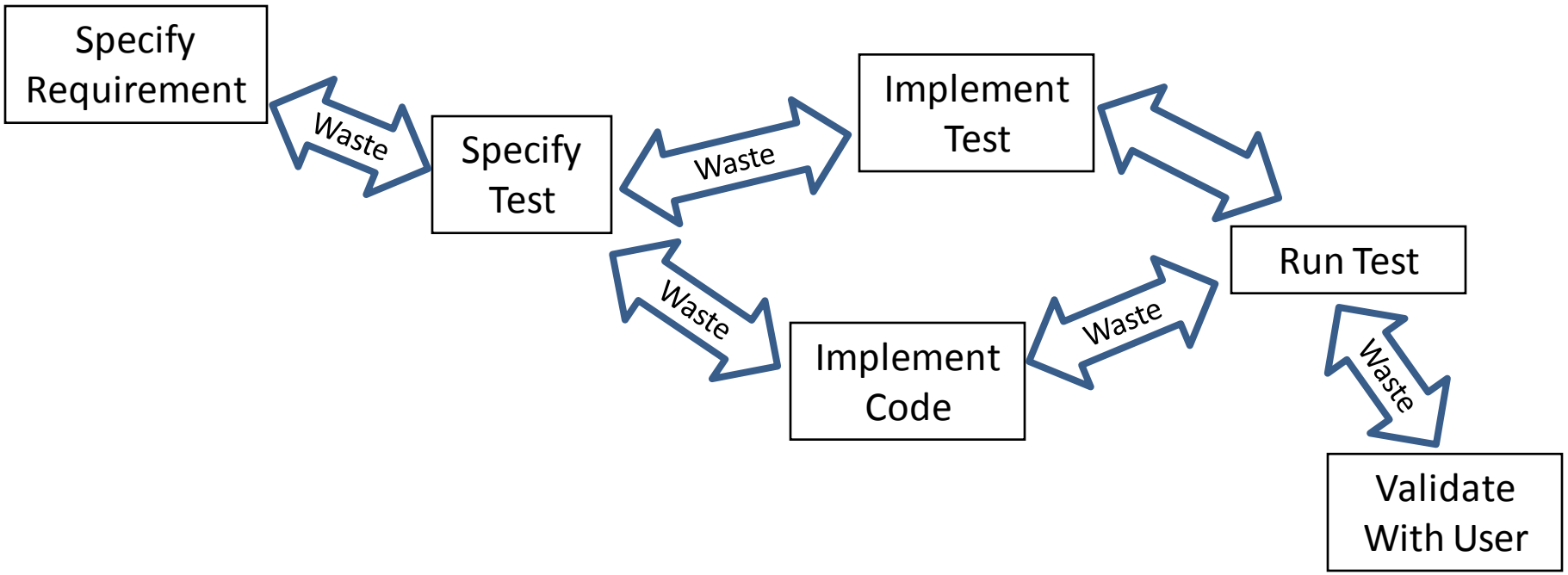
Payroll.Fixtures.WeeklyCompensation			
StandardHours	HolidayHours	Wage	Pay()
40	0	20	$800
45	0	20	$950
48	8	20	$1520 *expected* $1360 *actual*

How To Decide the Timeline For Testing



The tasks required to implement a particular story after it has been analyzed.

Illustrating Dependencies and Waste



Laying Out the Flow and Identifying Wastes

The Ideal Sequence of Events

Specify Requirement	Specify Test	Implement Test	Implement Code	Run Test	Validate With User

Optimally at the same time

Identifying the Optimal Ordering of the Steps

Timeline for Acceptance Tests

- Product Champion refines stories and acceptance tests from Release Planning meeting, a few days before the Iteration Planning meeting
- Developers/Testers add more detail tests in the Iteration Planning meeting
- Developers/Testers continue to flesh out in the Iteration – failing tests until code is implemented
- Developers get tests to pass
- Becomes part of the Regression test suite when story is accepted

Value of Tests to the Customer

- Allows for iterative development which enables the team to avoid building the wrong feature sets
- Retains the ability to add functionality quickly in the future
- Speeds the development cycle up by clarifying the needs of the customer

Resources

Lean Software Development
Agile / Scrum
Design / Testing / Programming Skills for Agile Developers
Lean-Agile Testing
Agile VSTS
Certification

All Ezines
All Streamzines
All Webinars
All Net Objectives Articles
Recommended Articles
Recommended Links

Tools

Bibliography by Topic
Book: Design Patterns Explained
Book: Emergent Design
Blog: Net Objectives Thoughts
User Groups

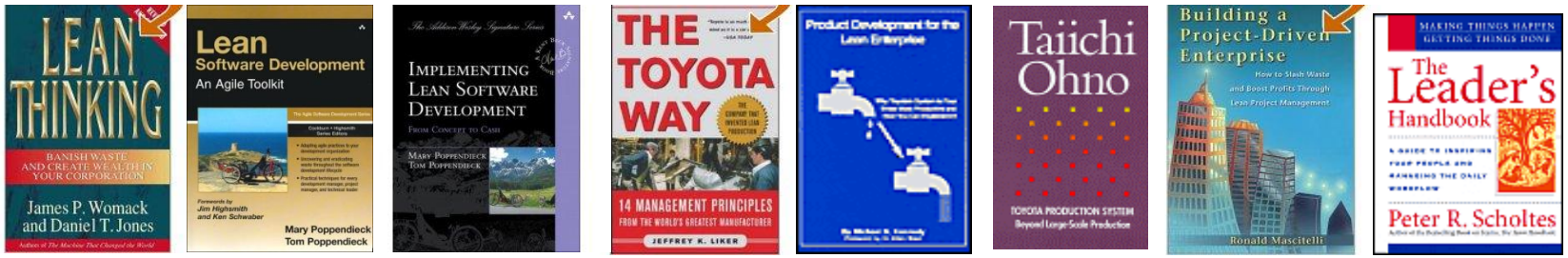
- Resources: www.netobjectives.com/resources
  - Ezines (regular online magazine)
  - Streamzines (PowerPoint with audio)
  - Articles and whitepapers
  - Pre/post course support Supporting materials
  - Quizzes
  - Recommended reading paths
- Blogs and podcasts: blogs.netobjectives.com
- Annotated Bibliography
- After-Course Support (students only)
- Additional Training
- Two User Groups
  - http://tech.groups.yahoo.com/group/leanagilescrum
  - http://tech.groups.yahoo.com/group/leanprogramming

Join our e-mail list to receive regular updates and information about our resources and training of interest to you

Net Objectives

A Short List of Books - Lean Related

- Womack and Jones: *Lean-Thinking*
- Mary & Tom Poppendieck
  - *Lean Software Development*
  - *Implementing Lean Software Development: From Concept to Cash*
- Jeff Liker: *The Toyota Way*
- Michael Kennedy: *Product Development in the Lean Enterprise*
- Taiichi Ohno: *Toyota Production System*
- Ronald Mascitelli: *Building a Project-Driven Enterprise: How to Slash Waste and Boost Profits Through Lean Project Management*
- Peter Scholtes: *The Leader's Handbook: Making Things Happen, Getting Things Done*



See http://www.netobjectives.com/resources/bibliography for a full bibliography

Other Relevant Books

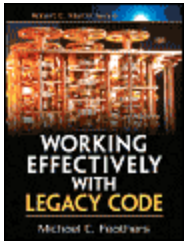
- William Bridges: *Managing Transitions*
- Weick and Sutcliffe: *Managing the Unexpected: Assuring High Performance in an Age of Complexity*

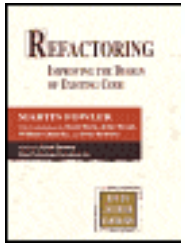
See http://www.netobjectives.com/resources/bibliography for a full bibliography

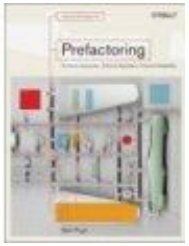
Net Objectives

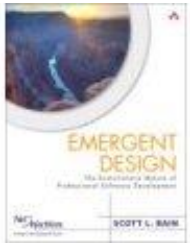
A Short List of Books - Technical

- Mugridge & Cunningham: *Fit for Developing Software*
- Michael Feathers: *Working Effectively with Legacy Code*
- Shalloway & Trott: *Design Patterns Explained, A New Perspective on Object-Oriented Design*
- Bob Martin: *Agile Software Development: Principles, Patterns and Practices*
- Freeman, Freeman, Bates, Sierra: *Head First Design Patterns*
- Martin Fowler, *Refactoring: Improving the Design of Existing Code*
- Ken Pugh, *Prefactoring*
- Scott Bain, *Emergent Design: The Evolutionary Nature of Professional Software Development*









See http://www.netobjectives.com/resources/bibliography for a full bibliography

SCRUM MASTER IMPLEMENTATION WORKSHOP

Applying Lean–Agile Software Development Practices with Scrum

- **Learn, experience, and practice the Scrum Master approach to managing development**
- **Apply lean-agile principles to software development projects**
- **Build a cohesive agile team to deliver high quality software more quickly**

Upcoming Training Dates & Locations

Washington, DC	September 8-10, 2008
Denver, CO	September 22-24, 2008
Chicago, IL	October 6-8, 2008
Seattle, WA	October 20-22, 2008
San Diego, CA	November 3-5, 2008

On-site Training is also always available. For additional information or to receive a free quote call 888.268.8770 or 904.278.0524, or email onsitetraining@sqe.com

PRACTICAL TEST-DRIVEN DEVELOPMENT

A Revolutionary Approach to Software Design and Programming

- **Practice using test-first design development methods**
- **Gain experience developing programs in small verifiable steps for better designs**
- **Learn how to refactor (re-design) existing applications to make them more maintainable**

Upcoming Training Dates & Locations

Washington, DC	September 8-10, 2008
Denver, CO	September 22-24, 2008
Chicago, IL	October 6-8, 2008
Seattle, WA	October 20-22, 2008
San Diego, CA	November 3-5, 2008

On-site Training is also always available. For additional information or to receive a free quote call 888.268.8770 or 904.278.0524, or email onsitetraining@sqe.com

USER STORIES AND ESTIMATION IN AGILE DEVELOPMENT

How to Write User Stories and Estimate Development Time

- **Create user stories that describe what the user really needs**
- **Learn to accurately estimate story development time**
- **Manage user stories over the life of the project**

Upcoming Training Dates & Locations

Washington, DC	September 11-12, 2008
Denver, CO	September 25-26, 2008
Chicago, IL	October 9-10, 2008
Seattle, WA	October 23-24, 2008
San Diego, CA	November 6-7, 2008

On-site Training is also always available. For additional information or to receive a free quote call 888.268.8770 or 904.278.0524, or email onsitetraining@sqe.com

DESIGN PATTERNS EXPLAINED

Principles, Practices, and Qualities of Good Design

- **Learn what design patterns are and which are most common and useful**
- **Discover how design patterns work in an agile environment**
- **Strengthen your design and programming abilities**

Upcoming Training Dates & Locations

Washington, DC	September 11-12, 2008
Denver, CO	September 25-26, 2008
Chicago, IL	October 9-10, 2008
Seattle, WA	October 23-24, 2008
San Diego, CA	November 6-7, 2008

On-site Training is also always available. For additional information or to receive a free quote call 888.268.8770 or 904.278.0524, or email onsitetraining@sqe.com

LEAN-AGILE TESTING PRACTICES

Rapid Delivery of High Quality Software

- **Apply lean principles to quality and testing**
- **Deliver value to customers quickly with agile testing practices**
- **Discover opportunities for lean-agile improvements**

Upcoming Training Dates & Locations

New York, NY	September 11-12, 2008
Washington, DC	September 18-19, 2008
San Francisco, CA	October 23-24, 2008
Tampa, FL	November 20-21, 2008

On-site Training is also always available. For additional information or to receive a free quote call 888.268.8770 or 904.278.0524, or email onsitetraining@sqe.com

Q & A

Have a question for the speakers?
Ask now.