

---

# Strategic Software Testing

---

Keys to Higher Impact and Less Effort

Larry Green

[Larry.Green@c2its.com](mailto:Larry.Green@c2its.com)



---

## What this talk is about

### A Goal -Question -Metric Approach

---

*Strategic planning is worthless -- unless there is first a strategic vision. - John Naisbitt*

- ❖ Visionary Goals:
  - ❖ SQA provides high value (impact) in a limited time, per project
  - ❖ SQA provides long lasting quality and productivity results
  - ❖ Provide general concepts to structure and articulate strategy

---

## What this talk is about

### A Goal -Question -Metric Approach

---

*Strategic planning is worthless -- unless there is first a strategic vision. - John Naisbitt*

- ❖ Questions to Elaborate on Goals:
  - ❖ What are example impactful SQA characteristics? (By individuals and by teams)
  - ❖ What are examples of lost or delayed SQA impact?
  - ❖ What is a quality endpoint? Can quality grow?
  - ❖ What are the intangible -Hard to Quantify- values enhanced by SQA?
  - ❖ Does impactful implementation vary by scope, by timeline?

---

# What this talk is about

## A Goal -Question -Metric Approach

---

*Strategic planning is worthless -- unless there is first a strategic vision. - John Naisbitt*

- ❖ Metrics:
  - ❖ Loss Prevention
  - ❖ Customer satisfaction and loyalty

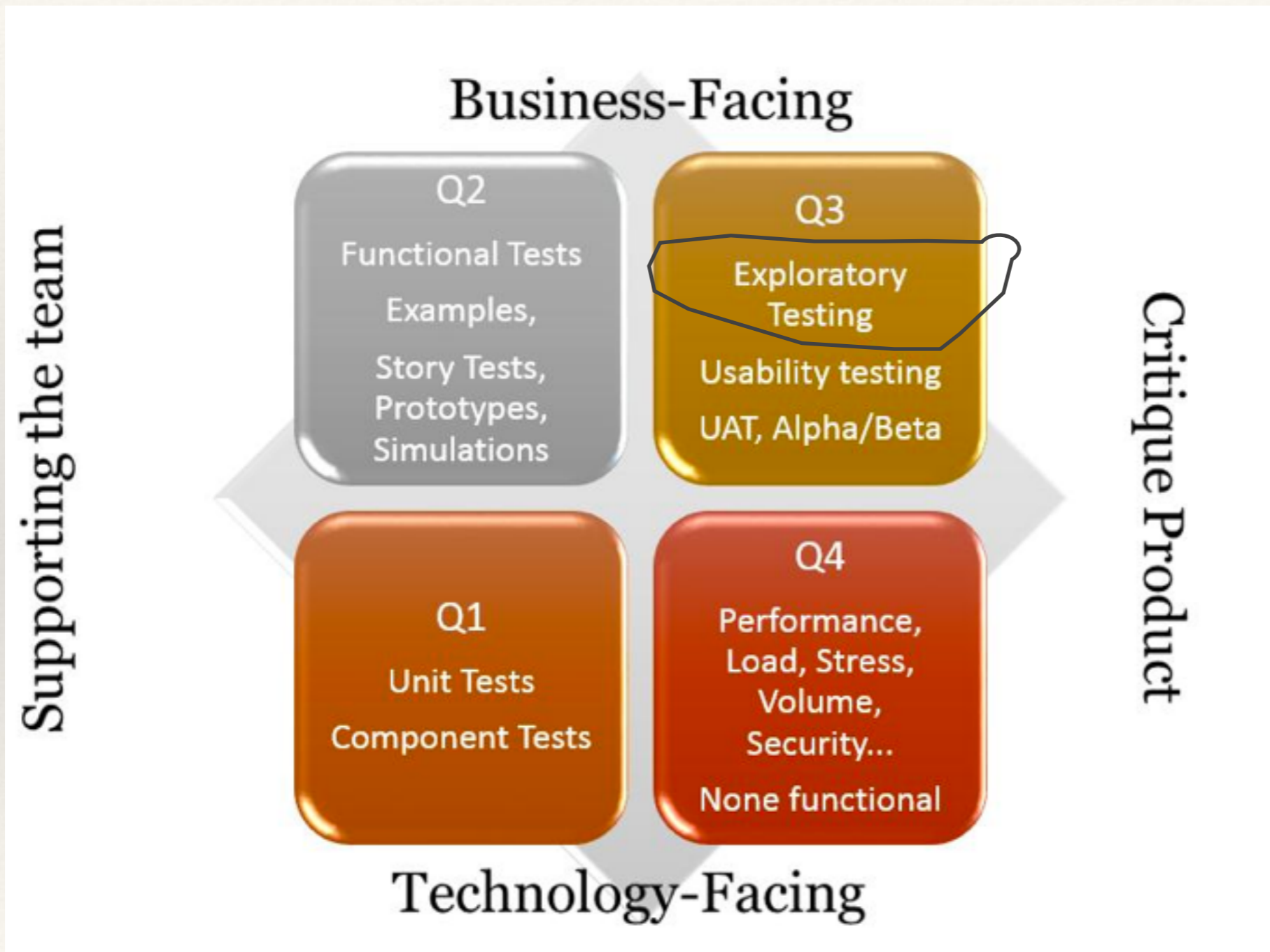
---

# Selected SQA Activities

---

- ❖ First Goal: *“SQA provides high value (impact) in a limited time, per project”*
- ❖ Typical SQA Actions:
  - ❖ Testing
  - ❖ Bug discovery and resolution
  - ❖ Demonstrate that the Product is Fit for Use

# Brian Marick's Testing Matrix



[https://  
www.linkedin.com/  
pulse/  
20141110063401-13798802  
-agile-testing-agile-  
testing-methods](https://www.linkedin.com/pulse/20141110063401-13798802-agile-testing-agile-testing-methods)

---

# Exploratory Functional Testing

---

- ❖ Exploratory Testing technique...A supplement to structural based testing.... *Also a way to jump start testing cycles!*
- ❖ Add flexibility and rapid [issue] discovery
- ❖ Rapidly learn about new features
- ❖ Augments omission and gaps by structured and automated tests
- ❖ Best with curious, focused, lateral thinking individuals as testers

Get ahead of the schedule.

Like  
Brian Marick

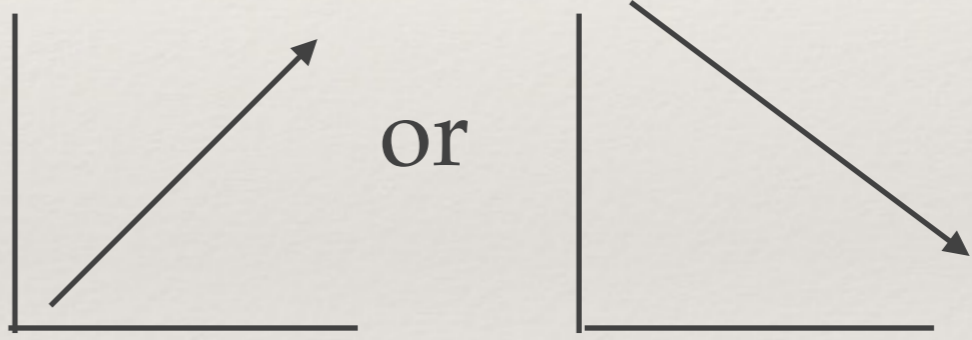
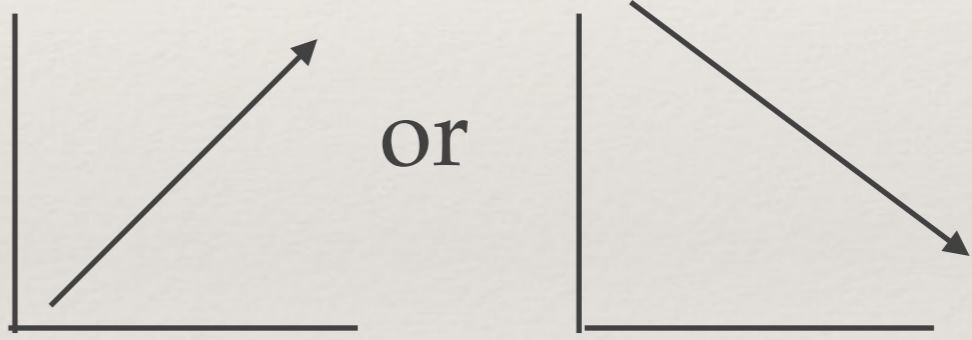
---

# Bend the Product; Bend the Schedule

---

*Planning is bringing the future into the present so that you can do something about it now -*

*Alan Lakein*

- ❖ Typical schedule metric is {Number of ~Tangible Things} per Week where the 'Things' are 'Story Points', 'Lines of Code', 'Bugs resolved', ...
- ❖ Schedule timeline looks like;  or 
- ❖ *Exploratory Testing Objective; Find bugs faster, initially, than the assumptions built into the schedule*



---

# Applied Exploratory Testing using Goal-Question-Metric

---

*“Learn the rules like a pro, so you can break them like an artist” -Pablo Picasso*

- ❖ Suppose you are asked to perform exploratory tests on a new shopping cart website. You choose a goal-question-metric approach to formulate testing approach.
- ❖ Goal: Assess web site’s functionality
- ❖ Metric: Produce an activity and observation list
- ❖ Question Generator Exercise; Ask yourself and / or others open-ended questions like Who?, What?, When?, How Much?, How Often?

---

# Goal-Question-Metric Next: Add some structure to Initial Questions

---

*Seek First to Learn...*

- ❖ ‘What?’ type questions;
  - ❖ User experience centric questions: How might a novice user use this site? A platform power user? An experienced user of the site? A Hacker? A bot? A voice command user? Does the workflow and speed of the site conform to expectations? Does the site provide shortcut alternate workflows?
  - ❖ Business centric questions: Does the site expose private data? Allow backdoors to private directories and files? Accurately present billing numbers? Correctly enter data into backend financial and order fulfillment systems? Provide suitable marketing and user tracking analytics? Provide suitable data for fraud detection? Provide logging depth sufficient to support operations?
- ❖ ‘How?’ type questions;
  - ❖ Use to develop a list of tools, techniques, locations and access rights leading to actual testing

**Refine questions into planned actions. In other words: Plan, Act, Do**

---

# Another 'Take' on Exploratory Testing

---

*In preparing for battle I have always found that plans are useless, but planning is indispensable. - General Dwight Eisenhower*

Keep it (exploratory testing) simple, flexible

The usual testing outcome (from any type of testing)

Bugs...

of various severities

---

# High Severity Bugs; High Project Impact

---

- ❖ What's the Technical Impact of High Severity Bugs?
  - ❖ Expose Private data, financials
  - ❖ Corrupt data
  - ❖ Unintended interactions
  - ❖ Slow or Stopped Responses
  - ❖ Fail to Scale

---

# High Severity Bug Impacts, Project Responses

## Language of Pain


---


### Pain Point Assessments...

 Major Project Schedule Impacts (Disruptive)

 Change from general release to beta

 Redesign product, Product component, Infrastructure

 Project Loss Prevention (Fail to satisfy user, Excessive company risk, Incur longterm maintenance and customer support costs)

 Immediate consequence is cost of change

   Creates Action!

---

# Bug Impacts, Project Responses

## Language of Gain \$\$ instead of Pain

---

SQA; A Company Asset

\$ Gain Point Assessments...

\$ Customer satisfaction and loyalty

\$ Immediately useful product (ease of install and use)

\$ Longer time between support activities (few shipped bugs)

\$ Faster product update response times (fast regression test time)

\$ Reduced Customer support cost, Higher customer approval rating

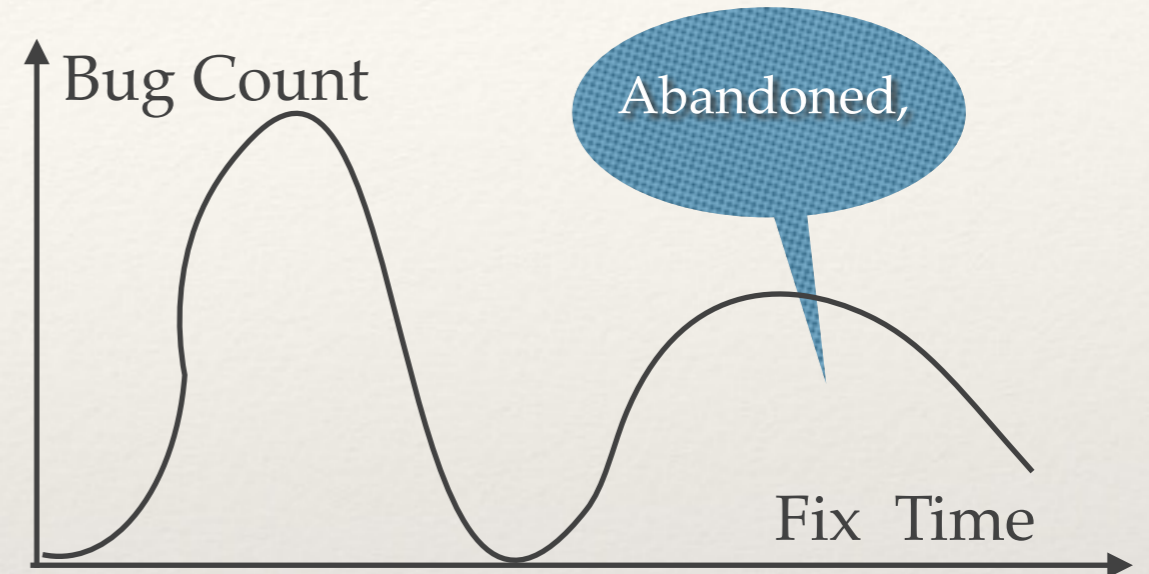
\$ Reduced Maintenance Cost

\$ Lower Liability Risk

\$ Differentiates product from competition

# Bug Fix Rate, Charting Pain and Gain

- ❖ Postponed or Cancelled Bugs; Why?
  - ❖ Found near project deadline
  - ❖ Difficult to reproduce
  - ❖ Difficult to fix (e.g. rigid architecture)
  - ❖ ‘Works as Designed’...Under appreciated qualities (company de-emphasis?)
  - ❖ Possible Postponed Bug Categories; *Usability, Maintainability, Security, Extensibility...*



---

# Positioning Abandoned Bugs and Features for Future Adoption

---

- ❖ Look for messages in abandoned / postponed bugs.
  - ❖ Pay now vs. pay latter, Initial good, eventual great Primary customer vs. secondary customer
- ❖ Emphasize **Gain Point** qualities in conversations
- ❖ Plan for long term, incremental (small) changes
  - ...or no change! Keep things in perspective
- ❖ Team with others for greater impact
- ❖ Don't loose track of bugs



---

# Bug Tracking; Verbal, Written, Both?

---

- ❖ How does Agile-Scrum emphasis on informal, rapid communication affect bug documentation? Affect bug adoption?
  - ❖ Are verbal bugs easier to ignore? Are bugs that transcend today's scrum cycle easier to ignore?
  - ❖ What's the distinction between (informal, verbal) bug resolution and (formal, written) requirements? Scope?, Time line?, Person's role?

*An observation:* The less formal and more high level the requirements, the more prevalent that detail design is directed by discovery / recovery. Bugs in role independent sense.

- ❖ How does a verbal bug become transformed into written technical debt or User Story?
- ❖ Since code is the primary tangible output of a project, how are informal, and intangible project inputs such as bugs (and the people that produce them) valued?

---

# SQA; User Acceptance, Regression, User Interface Tests

---

- ❖ After Testing, and Bug Resolution comes project rollout with End User Experience
- ❖ Demonstrate that the product is fit for use
  - ❖ Install, Login, Customize, etc.

---

# Impactful QA -Product Rollout

---

- ❖ Minimum 'Viable' Product (MVP).

Viability varies by organizational perspectives. Logistical, Financial, Technical.

- ❖ What's are 'Viable' characteristics

- ❖ Distribution Channel, Price, When is product available?

- ❖ Feature Set, Project Timeline, Initial Quality, Enhancement Delivery time

- ❖ Prioritization Perspective; No Day by Day Drama. Installs OK, Happy Path OK, Customization and configuration friendly e.g. like Apple products

# Maslow Hierarchy of Needs ...for Software Products

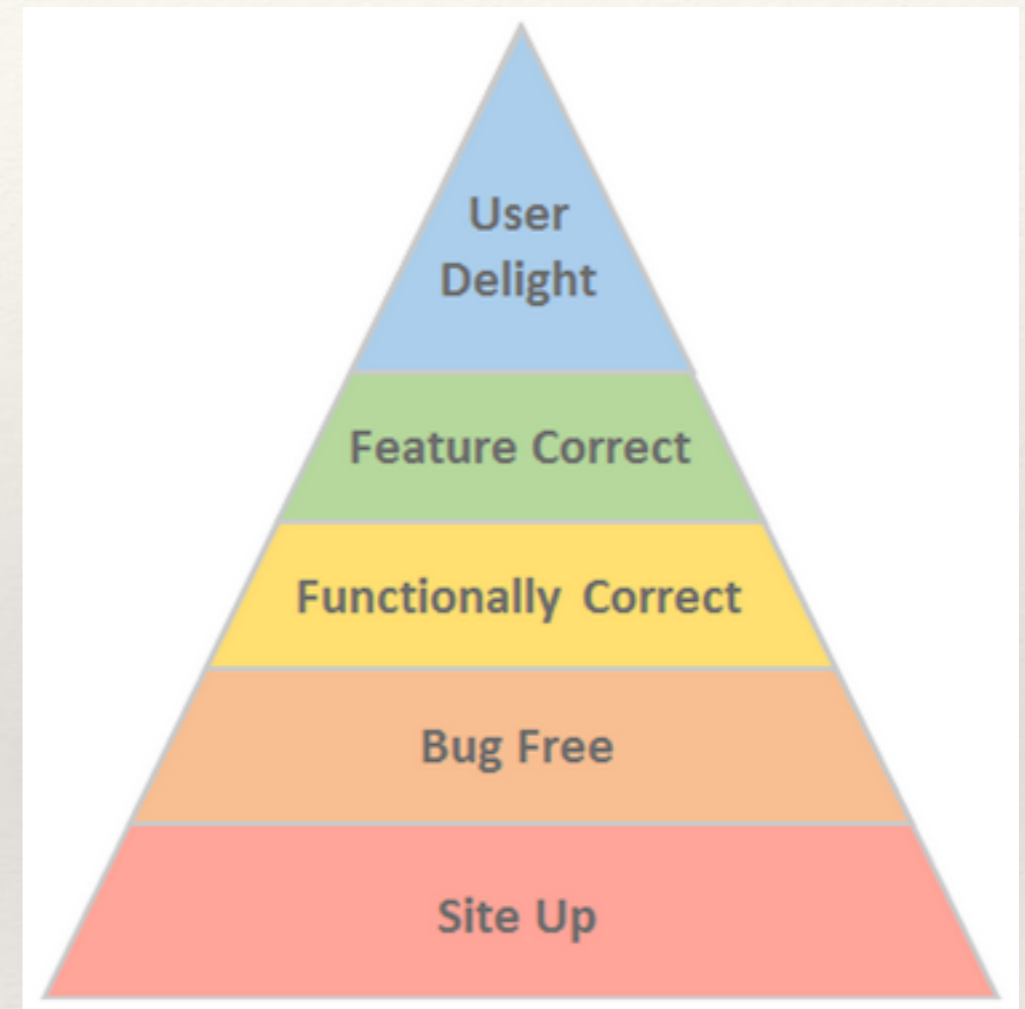
- ❖ Lowest level is No Daily Drama
- ❖ Multiple levels to satisfaction
- ❖ What's business owners 'operational' level?

- ❖ Best product;

Does the Thing Right

....and

Does the Right Thing



- ❖ Original pyramid of needs from Maslow; Physiological needs, safety, Love and belonging, Esteem , Self-actualization Reference [https://en.wikipedia.org/wiki/Maslow's\\_hierarchy\\_of\\_needs](https://en.wikipedia.org/wiki/Maslow's_hierarchy_of_needs)

---

# SQA; Seen as Driver for Productivity and Quality

---

Second Goal: *“SQA provides long lasting quality and productivity results.”*

- ❖ Challenges;
  - ❖ Long term (multi-year) payout whilst engaged in shorter duration projects
  - ❖ Finding tangible benefits and / or adding visibility to intangible or difficult to quantify benefits
  - ❖ Activities can be formal or informal
  - ❖ Activities may include communication and persuasion

---

# SQA; Seen as Driver for Productivity and Quality

---

*“Second Goal: SQA provides long lasting quality & productivity results.”*

- ❖ Potential Goal Metrics (Some with Fuzzy measures);
  - ❖ Reduce development time, support cost
  - ❖ Improve product quality consistency

---

- ❖ Improve company appreciation for quality products
- ❖ Evolve corporate culture
  - ❖ Changing culture can involve years duration but also major \$\$\$ value

---

## Question: How to Achieve the goal? Team for Greater Impact

---

*Productivity is never an accident. It is always the result of a commitment to excellence, intelligent planning, and focused effort. - Paul J. Meyer“*

*Alone we can do so little; together we can do so much“ -Helen Keller*

- ❖ Observation: Measures of the second goal include broad, long term, shared influence. DevOps and Agile movements resulted from shared needs and built from small beginnings
- ❖ Look for places where SQA skills and outputs meet the needs of other groups
- ❖ Use conversations to find and build interest across an organization

---

# Beneficial Teaming Ideas

---

- ❖ With Operations; Write test cases as operations troubleshooting guides
- ❖ With infrastructure and others; Implement new concepts with potential to improve productivity e.g. CI/CD, TDD
- ❖ With Development Team and Operations; Off-load developer and operations on load test and analysis
- ❖ With project management / scrum master for clear-eyed status and risk assessment. Keep an Eye on the (end of project, over end of scrum) Prize!
- ❖ With Tech Pubs; Write the end user documentation



---

# Communicate to Influence

---

- ❖ The Elevator pitch
  - ❖ Values, problem solving, emphasis
- ❖ Create Powerpoint first, supplement with Word
- ❖ Organize thoughts with mind maps

<https://www.ministryoftesting.com/tag/mindmap/>

---

# SQA Strategic Planning

---

*Good fortune is what happens when opportunity meets with planning. - Thomas Alva Edison*

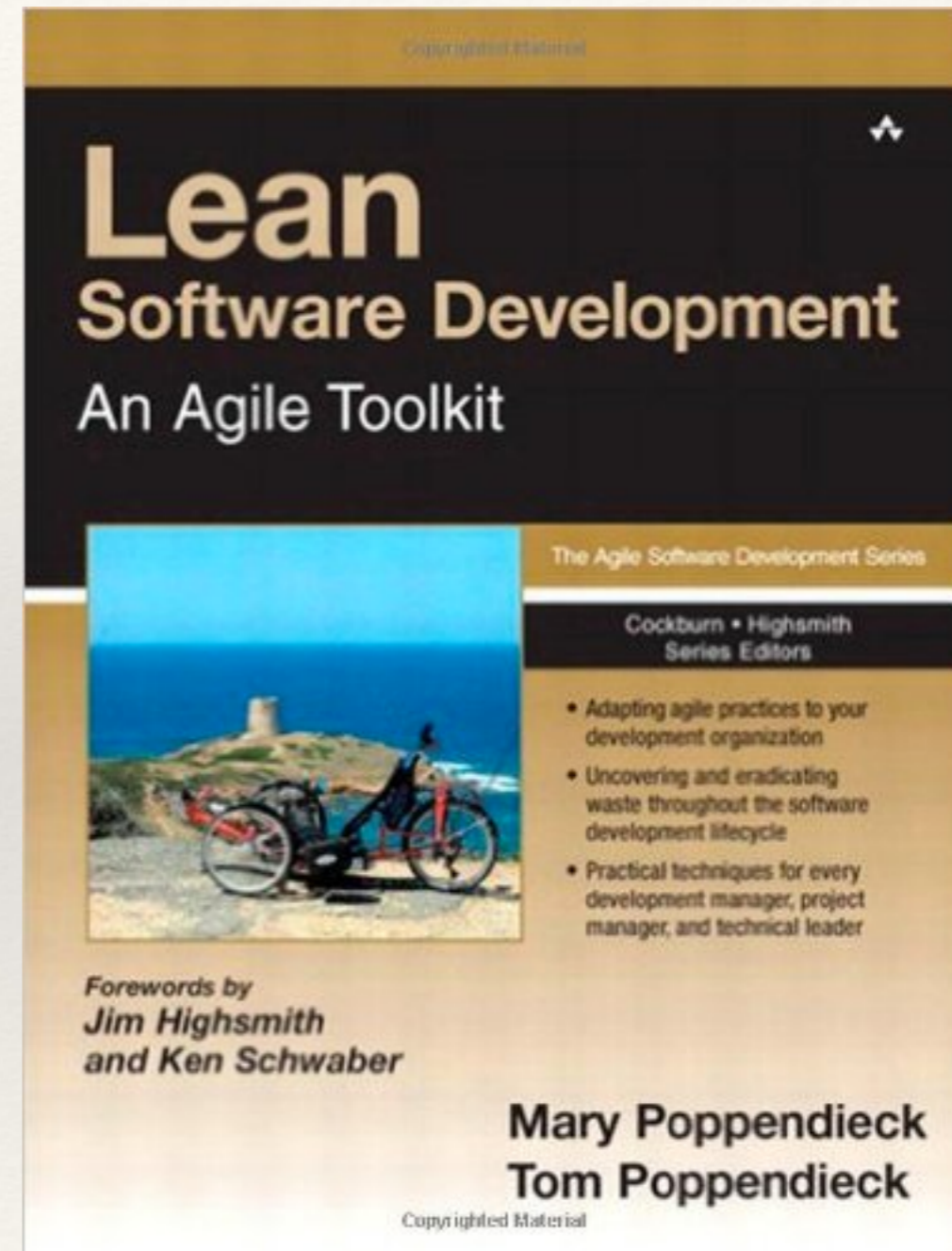
Where to find planning strategy concepts

and ...

Inspiration to create your own

# Create Your Own Impact

Lean Software  
Development created in  
the Twin Cities and now  
practiced worldwide



---

# Concept Bridges...

## Concepts from other Disciplines Applied to SQA

---

- ❖ Business Strategy
  - ❖ Minimum Viable Product
  - ❖ Structure in standards like ISO9001 (separating policies from procedures, general from specific)
  - ❖ Investment Planning (Incremental measures like Burn Down Charts, End goals)
- ❖ Philosophy and Psychology
  - ❖ Inference and Generalization, Evidence and Proof
  - ❖ Hierarchy of needs
- ❖ Software Development
  - ❖ Code Structure, Team Structure, Techniques for reliable software, Formal methods, Type Systems
  - ❖ Lean and agile
- ❖ Science , Engineering, Technology
  - ❖ Established ways to measure and predict project, product and test conditions
- ❖ Communication
  - ❖ Translate personal concepts to actionable group outcomes

---

# Thanks!

---

Questions?