Someone Has To Turn This Great Idea Into Dollars

Robert Gordon, PMP

February 13, 2014

Purpose & Topics

- The purpose of this presentation is to discuss the processes central to managing a technical development and production project.
- We will discuss processes and tools that focus on these technical needs.
- We will discuss how to get the most out of these tools.

- There aren't that many slides – Hopefully we can all discuss examples from our own experiences.

R&D In A Well Functioning Organization

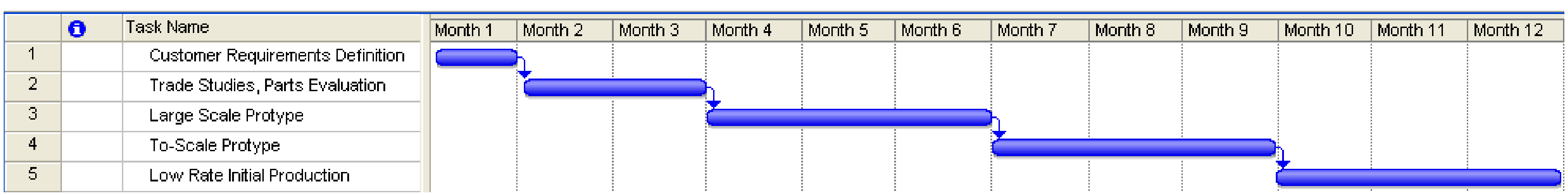
- Technical project management is extremely challenging even when the project has the organizational, management, and sponsor support it needs.
- Technical teams often create things or combinations of things that have never existed before.
- There are many great ideas and many brilliant people thinking them up – How do you get it made and shipped?
- When the going gets rough – and it will – a natural reaction is to ask if the team put enough effort into the 'Planning' process group.
  - If the plan incorporates a strategy that recognizes difficulty will arise at the unit level and defines actions to overcome these effects, the project has a better chance of success.

Focusing On The Technicalities

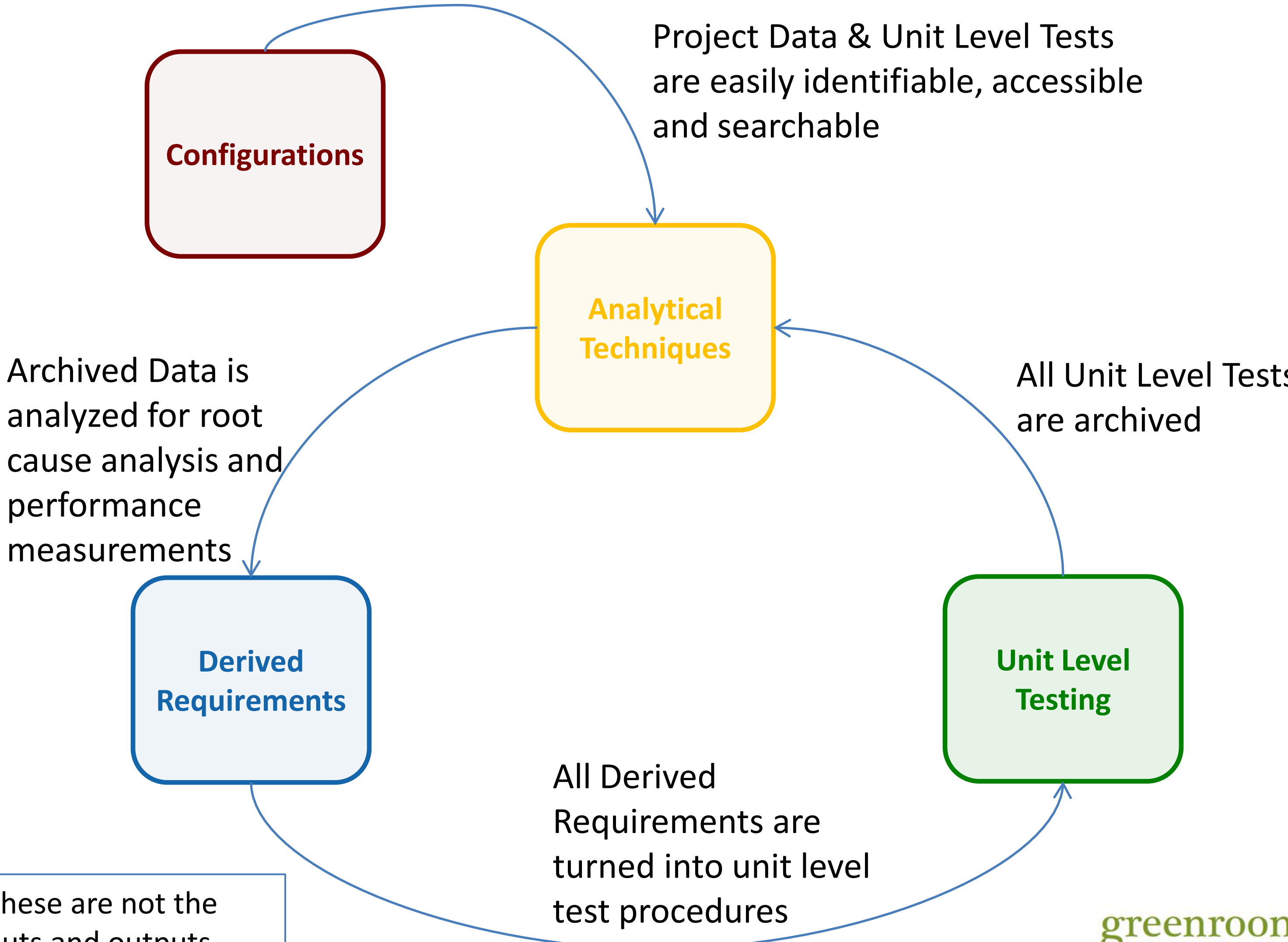
- How Does The Team Know
  - What the project is supposed to build?
    - Collect (Derived) Requirements
  - What the project is actually building?
    - Monitor & Control Project Work & Perform Integrated Change Control
      - Tools include Configuration Identification & Configuration Management
  - That what the project is building is what it thinks it is building?
    - Control (Unit Level) Quality
  - How can you learn from work already done?
    - Monitor & Control Project Work – Analytical Techniques
- Technical Projects must accept the likelihood of technical unit level problems and build a plan to mitigate the impacts.

Example: Collect (Derived) Requirements

- Top-Level requirements are typically well understood early in the development process.
- Derived requirements produce the detailed inputs necessary to create adequate unit level tests.
- The ‘Collect Requirements’ process lists prototyping as one of its tools.
- Technical projects typically spend ¾ of the overall schedule and budget producing a sequence of prototypes with each revision being an elaboration of the previous prototype.
- Successful projects put a great deal of effort into capturing derived requirements and mapping these to unit level tests.

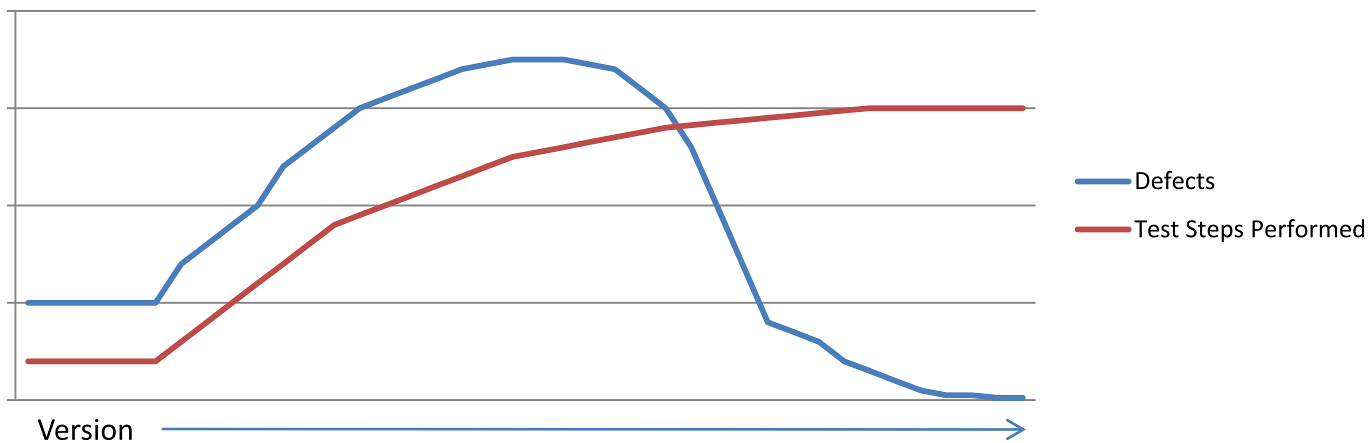


How It Fits Together



Note: These are not the only inputs and outputs

The Value of Unit Testing



- A long time ago on a project far, far away….
- Initial Testing focused on System Level Functionality.
- The team decided to focus on unit level tests – many procedures were added and elaborated over time.
- Initially, the defect rate went up as latent defects were uncovered and fixed.
- Over time, the defect rate dropped.
- The developers used the test procedures as a development guide.
- Testing costs decreased over time due to automation and a lower defect rate.
- The team was able to quickly measure testing effectiveness due to good configuration management of test procedures and records.

Isn't Testing Expensive?

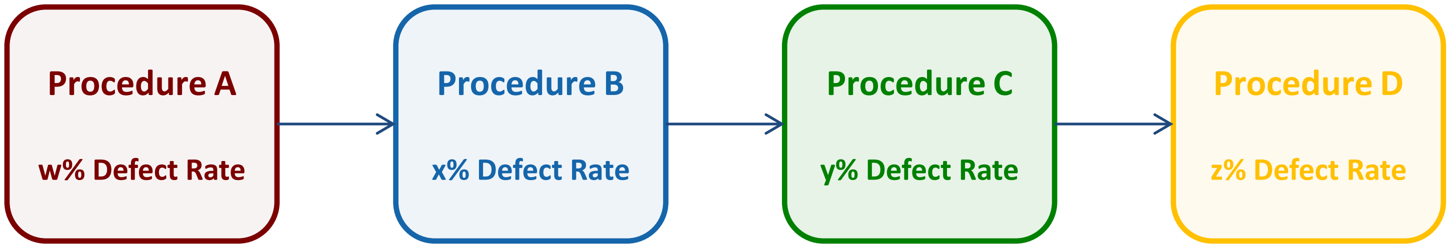
- Yes, but there are techniques to reduce costs.
- Test Driven Development is very beneficial to the engineers.
  - By definition, engineers are good at passing tests.
  - Engineers are very vested in a process that encourages them to analyze root causes, further elaborate their own unit level requirements, and develop against a set of testable requirements.
  - Many engineers find engineering a test apparatus to be as rewarding as engineering components of the deliverable system.
    - Buy-in has a positive return on investment.
- The quality of unit level testing is a good indicator of the likelihood of success for technical projects.

Cost Effective Testing

- Move tests that have a higher failure rate earlier in the test sequence.
- Now that failures are grouped up front, use all of the well managed test data to search for root causes.
  - Even better, do both at the same time!

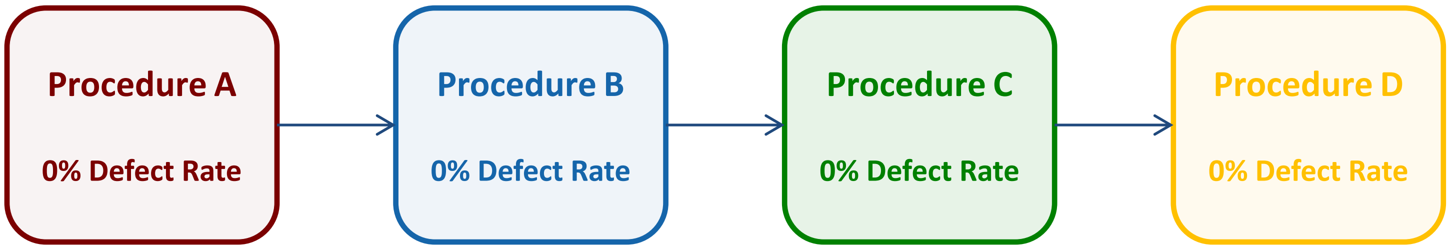
- Let's consider an example of test optimization.

Simplified Testing Scenario



- We have a production testing pipeline with four procedures.
- Assume the testing cost is only the labor required to setup the test and process the results.
  - Assume a labor rate of $100 / hour.
- Our procedures require two hours per test per item ($200).
- There are 100 test articles.
  - Manufacturing Example: 100 identical items under test.
  - Software Example: Each procedure tests 100 sub-units.
- Each defect requires 4 hours of rework ($400)
- What is the testing & rework cost per article?

Ideal Testing Scenario



Ideal Test Flow

	A	B	C	D		
Defect Rate	0%	0%	0%	0.01%	Total Tests Performed	400
Total Tests Executed Per Stage	100	100	100	100	Percent Testing Over Baseline	0.0%

- If this happens, then all of our efforts in the Planning Process Group have paid off!
- What happens when we find defects during testing?
  - (We will)
  - Risk Planning addresses the near certainty that *something* isn't going to go as planned at this point.

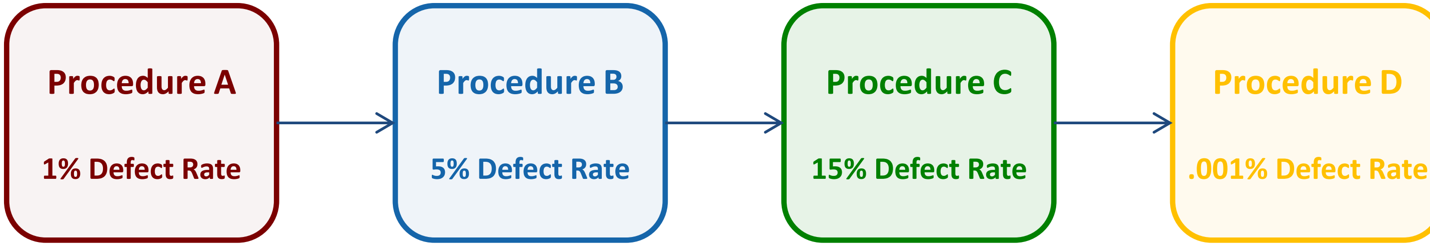
greenroom software
embedded and industrial software engineering
testing services

Ideal Testing Costs

- The ideal testing scenario provides a baseline for measuring testing costs.

	Ideal Test Flow
Testing Stages	4
Lot Size	100
Tests Performed	400
Cost Per Test	$200
Testing Cost	$80,000
Testing Cost Per Article	$800
Percent Cost over Baseline	0.0%
# Reworks	0
Rework Cost Per Incident	$400
Total Rework Cost	$0
Avg Rework Cost For The Lot	$0
Test + Rework Cost Per Article	$800
Percent Cost over Baseline	0.0%

Unoptimized Test Flow With Defects



Unoptimized Test Flow

	A	B	C	D
Defect Rate	1%	5%	15%	0.01%
# Stage 'A' Tests	100			
# Stage 'B' Tests Plus Rework	1	100		
# Stage 'C' Tests Plus Rework	5	5	100	
# Stage 'D' Tests Plus Rework	15	15	15	100
Stage 'D' Rework	0	0	0	0
Total Tests Executed Per Stage	121	120	115	100

Total Tests Performed	456
Percent Testing Over Baseline	14.0%

- Various rates of defects are found for each test procedure.
- Each item with a defect will have to be reworked at a cost of $400 per defect.
  - We will assume our rework success rate is 100%, i.e. a reworked test article will pass a retest and any previous tests.
  - This means that each stage has 100 first attempts with a variable rate of subsequent attempts depending on its order in the test flow.
- What is the testing and rework cost per article?

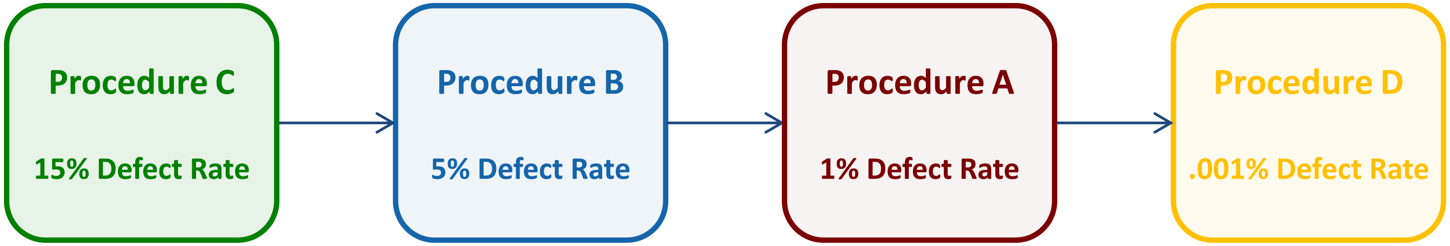
greenroom software
embedded and industrial software engineering
testing services

Unoptimized Testing Costs

- We can now compare unoptimized testing costs to the ideal baseline.

	Ideal Test Flow	Unoptimized Test Flow
Testing Stages	4	4
Lot Size	100	100
Tests Performed	400	456
Cost Per Test	$200	$200
Testing Cost	$80,000	$91,200
Testing Cost Per Article	$800	$912
Percent Cost over Baseline	0.0%	14.0%
# Reworks	0	21
Rework Cost Per Incident	$400	$400
Total Rework Cost	$0	$8,400
Avg Rework Cost For The Lot	$0	$84
Test + Rework Cost Per Article	$800	$996
Percent Cost over Baseline	0.0%	24.5%

Optimized Test Flow With Defects



Optimized Test Flow

	C	B	A	D
Defect Rate	15%	5%	1%	0.01%
# Stage 'A' Tests	100			
# Stage 'B' Tests Plus Rework	15	100		
# Stage 'C' Tests Plus Rework	5	5	100	
# Stage 'D' Tests Plus Rework	1	1	1	100
Stage 'D' Rework	0	0	0	0
Total Tests Executed Per Stage	121	106	101	100

Total Tests Performed	428
Percent Testing Over Baseline	7.0%

- Once testing reveals where defects are most likely to occur, the order of procedures can be re-arranged to move defect discovery earlier in the process.
  - This is not always possible if one test depends upon the completion of a previous test.
  - In this case, is it a test or a calibration procedure?
- All other assumptions are the same.
- What is the testing and rework cost per article?

greenroom software
embedded and industrial software engineering
testing services

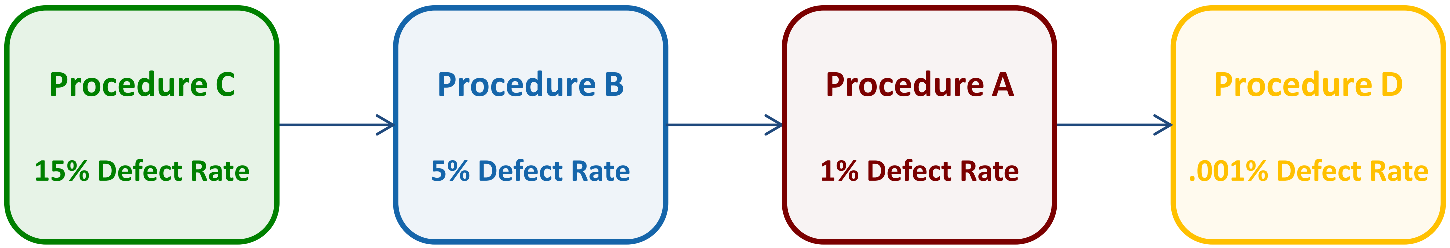
Optimized Testing Costs

- We can now compare optimized testing costs to the unoptomized testing costs and to the ideal baseline.

	Ideal Test Flow	Unoptimized Test Flow	Optimized Test Flow
Testing Stages	4	4	4
Lot Size	100	100	100
Tests Performed	400	456	428
Cost Per Test	$200	$200	$200
Testing Cost	$80,000	$91,200	$85,600
Testing Cost Per Article	$800	$912	$856
Percent Cost over Baseline	0.0%	14.0%	7.0%
# Reworks	0	21	21
Rework Cost Per Incident	$400	$400	$400
Total Rework Cost	$0	$8,400	$8,400
Avg Rework Cost For The Lot	$0	$84	$84
Test + Rework Cost Per Article	$800	$996	$940
Percent Cost over Baseline	0.0%	24.5%	17.5%

- Mileage will vary depending on test & rework costs, but there is <u>always</u> a savings when failures are caught early.

Test Procedure Elimination



- What about procedure D?
  - It has a very low defect rate.
  - Can we eliminate it?
  - Can we improve our automation to lower costs?
- This is a Risk Management Question.
  - What is the impact in terms of dollars? (Quantifiable)
  - What is the impact in terms of your company's reputation? (Might want to talk to management)
- We can use an expected monetary value analysis to determine whether the cost of a test exceeds the cost of replacing or repairing an untested item that fails after delivery.
- If **`Test Cost ≥ (Defect Rate) x (Return / Repair Cost)`**, we can consider eliminating the test procedure.

Conclusions

- Technical Projects must accept the likelihood of difficulties and plan accordingly.
- Technical issues are dealt with at the unit level.
- Good configuration management and record keeping supports analytical techniques to uncover root causes.
- Analytical Techniques are an effective source of unit level requirements and test procedures.
- Success with unit level testing is a good indicator that the overall system will perform well at system level testing.
- Testing can be organized so that it maximizes effectiveness while streamlining costs.

Thank You!