

GENERAL DYNAMICS

Land Systems

Put Your Defect Data to Work: Using Process Performance Models and Root Cause & Corrective Action

Margaret Corr
Dawn Jaskolski
David Sobetski, PMP

November 17, 2010

Agenda

- GDLS Overview
- Software Quality System Overview
- Peer Review and Software Problem Reporting Tools
- Defect Containment Effectiveness Model (DCE)
- Process Performance Models (PPM)
- Process Performance Baselines (PPB)
- Root Cause and Corrective Action (RCCA)
- Benefits & Challenges

Land Systems Products



Software Quality System Overview

Origin Phase	Phase Found In					Total Defects	%Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	84%
Design		88	5	0	0	93	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	12%	96%

Total Defects 715

Metric	Value
PCE	68%
DCE	12%

PCE = Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
 DCE = Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
 TCE = Total Containment Effectiveness (TCE = Pre-Release Defects / Total Defects * 100%)
 Contained Defects = Defects that are found during the phase that created them
 Escaped Defects = Defects that escape to a subsequent development or delivery phase

Software Systems Metrics

Abrams Requirements with Meeting Optimization Model

Number Reviewers Invited: [Min: 5] [Max: 15]

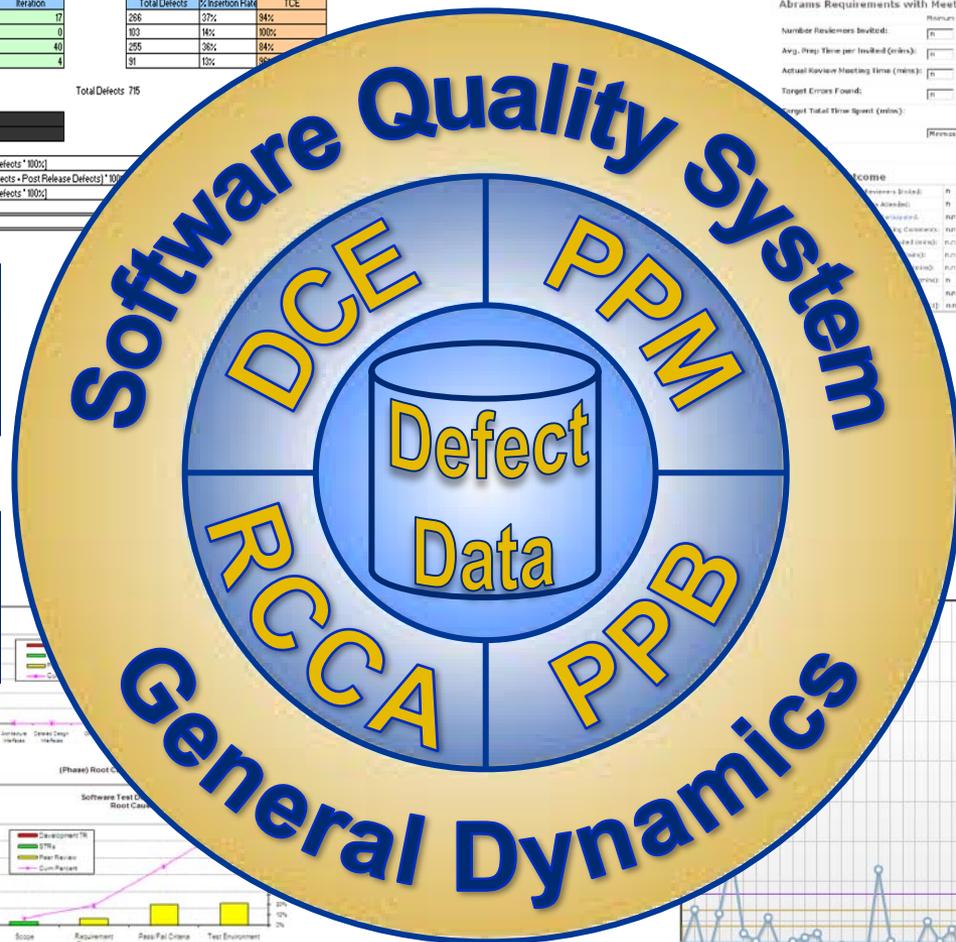
Avg. Prep Time per Invited (min): [Min: 15] [Max: 30]

Actual Review Meeting Time (min): [Min: 15] [Max: 30]

Target Errors Found: [Min: 1] [Max: 5]

Target Total Time Spent (min): [Min: 15] [Max: 30]

Previous Cost (Total Time Spent): [Min: 15] [Max: 30]

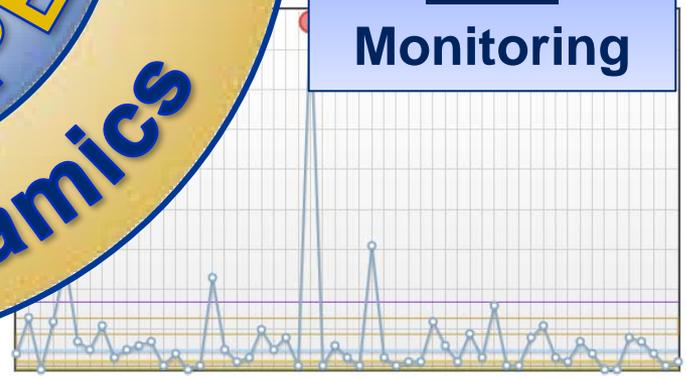
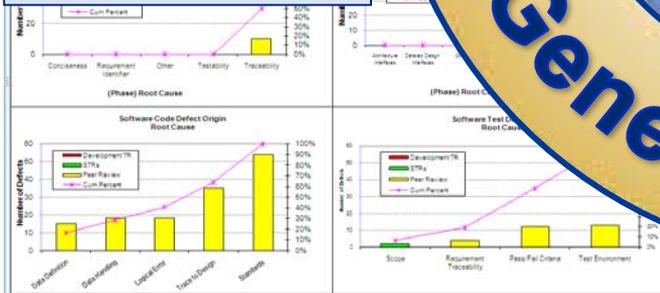


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



Defect Data

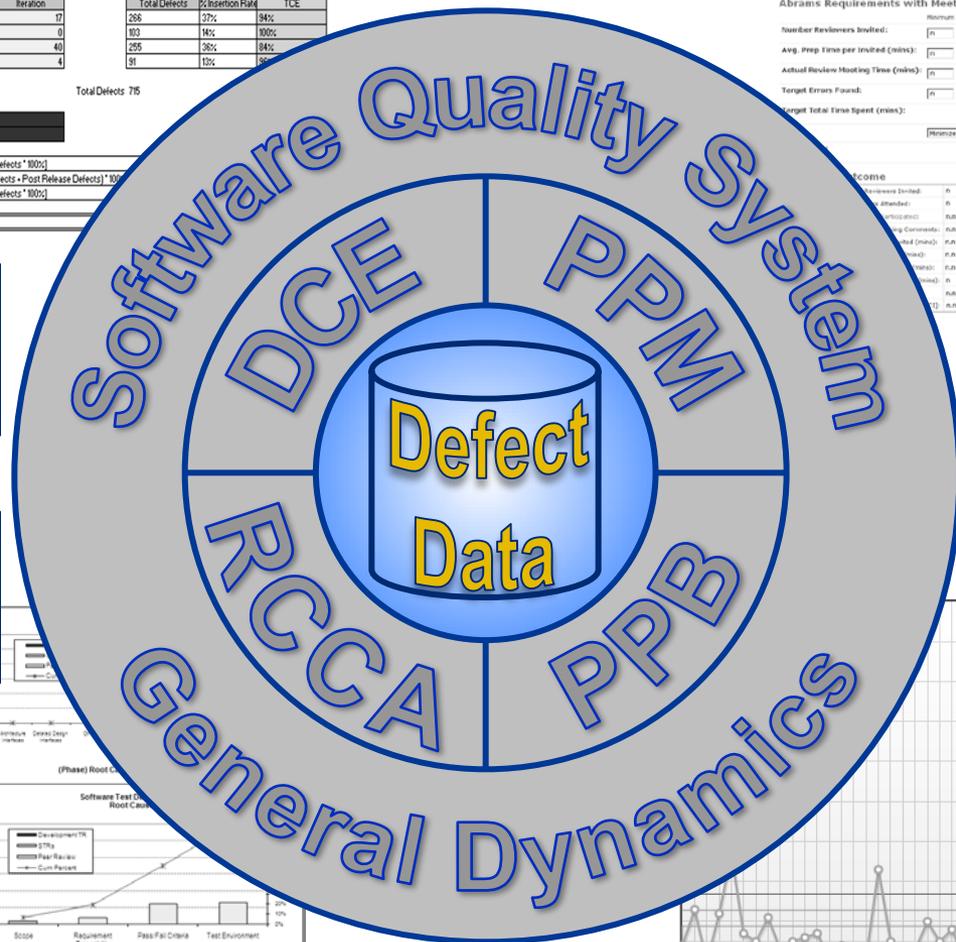
Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	101	10	43	15	17	286	37%	84%
Design		88	5	0	0	103	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	12%	98%

Total Defects 715

PCE	68%	95%	70%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
 DCE = Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
 TCE = Total Containment Effectiveness (TCE = Pre-Release Defects / Total Defects * 100%)
 Contained Defects = Defects that are found during the phase that created them
 Escaped Defects = Defects that escape to a subsequent development or delivery phase

Software Systems Metrics	
Abrams Requirements with Meeting Optimization Model	
Number Reviewers Invited:	Minimum: <input type="text"/> Maximum: <input type="text"/>
Avg. Prep Time per Invited (mins):	<input type="text"/> <input type="text"/>
Actual Review Meeting Time (mins):	<input type="text"/> <input type="text"/>
Target Errors Found:	<input type="text"/> <input type="text"/>
Target Total Time Spent (mins):	<input type="text"/> <input type="text"/>
	Minimize Cost (Total Time Spent)

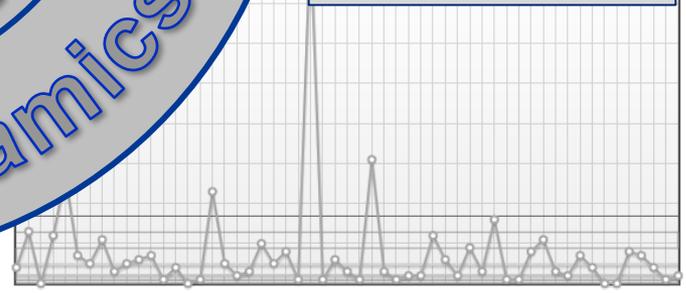
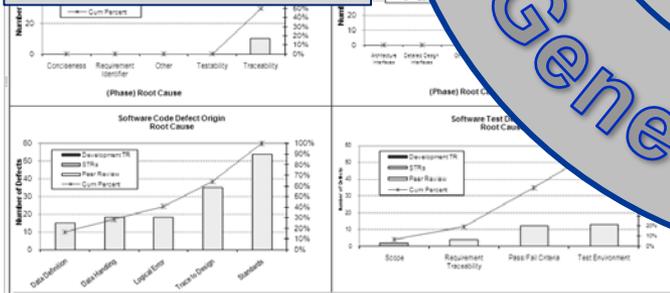


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



Defect Data

- Defect Origin
- Root Cause
- Defect Severity
- Peer Review Data
- Software Problem Reporting Data

Defect Origin and Root Cause

The Root Cause

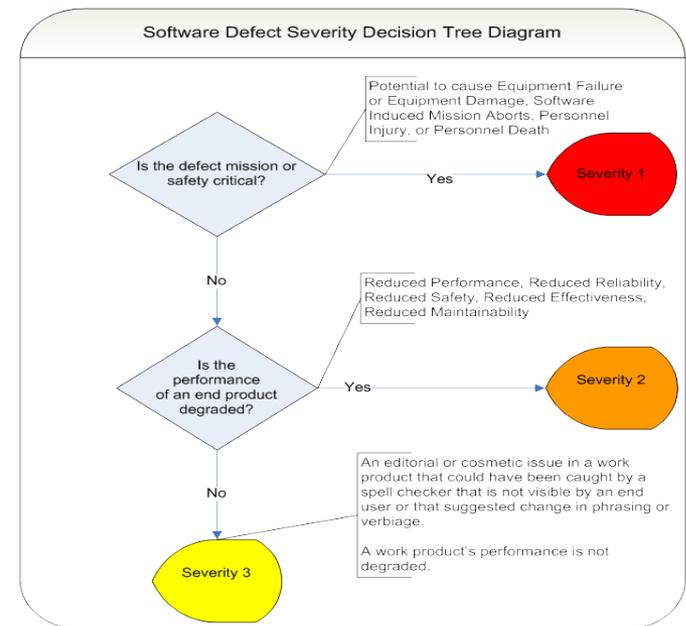
- Selection changes based upon the Defect Origin selected.
- Chosen based on what is determined to be the root cause of the problem and where it originated (Defect Origin).
- Not applicable for editorial issues.

<u>Issue Category (Defect Origin)</u>	<u>Root Cause</u>
Planning	Clarity
	Completeness
	Scope Change Per Higher Level Documents
	Trace to Higher Level Documents
Requirements	Ambiguity
	Testability
	Traceability
Design	Architecture
	Detailed Design
	Interfaces
	Traceability
Code	Data Definition
	Data Handling
	Initialization
	Logical Error
	Standards
	Trace to Design
Test	Pass/Fail Criteria
	Requirement Traceability
	Test Environment
	Scope

Defect Severity

- It is important to categorize the severity of the issue correctly. This categorization helps the project and organization identify where process, tool, and training improvements are needed.
- Major and Minor issues, Severity 1 and 2 respectively, are considered technical defects that generally require further analysis and prevention measures.
- Editorial issues, Severity 3, are considered non-technical defect and are not used in the generation of project defect prevention metrics.

Defect Origin:	Planning
Root Cause:	Clarity
Issue Severity:	3 - Editorial
Location in Item:	[Select One]
Issue:	3 - Editorial 2 - Performance Degraded (Minor) 1 - Mission Critical / Safety (Major)



Peer Review Tool

viewing review # 15257 | logged in as jaskolsk

Peer Review Tool

[home](#) | [create peer review](#) | [search & reports](#) | [edit groups](#) | [ips process](#) | [user guide](#) | [feedback](#)

[Main](#) | [Items](#) | [Participants](#) | [Schedule](#) | [Issues](#) | [Prep Time](#) | [Close Meeting](#) | [Close Review](#)

Excel Report: Export Issues (Note: Report may contain technical data - ITAR rules apply)

'NEW' [Click here to hide the 'Add Review Issue' and 'Upload Issues' forms to make it easier to view and scroll through issues \(any filters will remain\).](#)

Add Review Issue	
* Item:	Software Design Document
Submitted By:	Jaskolski, Dawn
Accept:	Undecided
Defect Origin:	[Select One]
Issue Severity:	[Select One]
Location in Item:	Code EN-GL-1.19.4, Software Defect Severity Decision Tree Guideline Design Editorial Planning Requirements Test
* Issue:	ex: Requirement#, Page #, Section

This is the lifecycle phase that the work product was in when the defect was injected into it.

Note: This is *not* when it was *found*. Example: A Software Requirements problem found in Software Design has a Defect Origin = Requirements.

Add Review Issue	
* Item:	Peer Review Training Material
Defect Origin:	Requirements
Root Cause:	[Select One]
Issue Severity:	[Select One]
Location in Item:	Ambiguity (Each requirement shall be unambiguous) Testability (Each requirement shall be testable) Traceability (Each requirement shall be traceable to a higher-level requirement)
* Issue:	Decision Tree Guideline

Root Cause
• Dependent on the selected Defect Origin

Software Problem Reporting Tool

Phase Found In
This is the lifecycle phase that the work product was in when the defect was identified or found

Severity

- Safety/Mission Critical
- Performance Degraded
- Performance NOT Degraded

The screenshot shows a web-based form for reporting software issues. The form is divided into several sections:

- Issue Type:** STR (dropdown)
- Problem Type:** SW (dropdown)
- Phase Found In:** Independent Software Test (dropdown)
- Phase Text:** *REQUIRED if Phase Found In = "Other Field".
- Related System Requirement:** (text field)
- Responsible Engineer:** CORR, M. (dropdown)
- Defect Origin:** Requirements (dropdown)
- Root Cause:** [Select One] (dropdown menu with options: [Select One], Ambiguity, Other, Testability, Traceability)
- Root Text:** *REQUIRED if Root Cause = [Selected]
- Issue Title:** Demonstration CMMI - San (text field)
- Items:** (text field)
- Date Created:** 2010-10-15 08:09:29
- Status:** REVIEW/ANALYSIS
- Version Identified In:** 1.1 (dropdown)
- Originator:** CORR, M.
- Severity:** 2. Performance Degraded (dropdown)
- Version Injected In:** [Select One] (dropdown)
- Test Name:** (text field)
- Regression Problem:** No (dropdown)
- Test Engineer:** CORR, S. (dropdown)
- Deferred:** No (dropdown)

Defect Origin
This is the lifecycle phase that the work product was in when the defect was injected into it

Root Cause
Based on the defect origin. Identifies the root cause of the defect

Defect Data – Benefits and Challenges

● Benefits:

- Automation
 - Tools enforce a more consistent process
 - Simpler data collection
 - Real-time metrics
- Consistent data between the Peer Review and Problem Reporting Tools
- Guidelines and Help linked into the tools for easy access

● Challenges:

- Data Integrity
 - Consistently identifying defect severities between technical and editorial
 - Size data is often entered incorrectly
 - Root cause data is often inconsistent
- Identifying definitions for root causes
- Educating employees on definitions

Defect Containment Effectiveness (DCE) Model

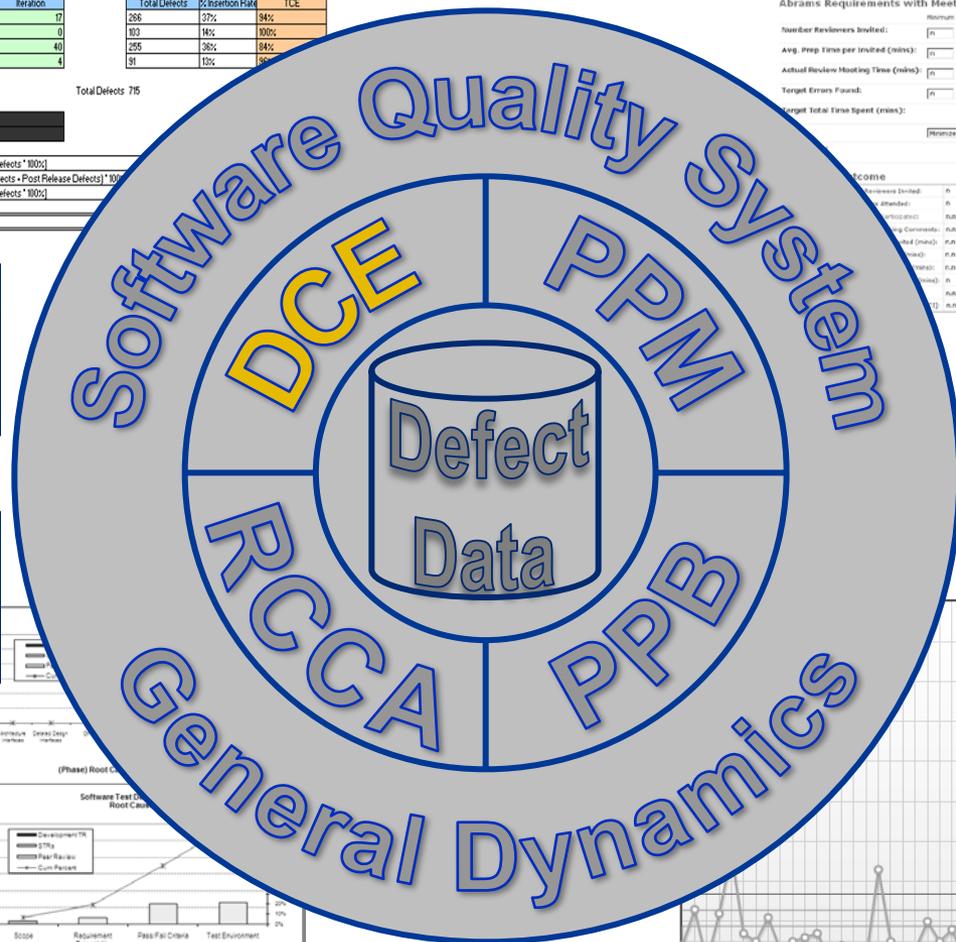
Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	84%
Design		88	5	0	0	93	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	12%	100%

Total Defects 715

Phase	PCE	DCE	TCE
Requirements	68%	95%	78%
Design	12%	60%	34%

PCE Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
DCE Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
TCE Total Containment Effectiveness (TCE = Pre-Release Defects / Total Defects * 100%)
 Contained Defects: Defects that are found during the phase that created them
 Escaped Defects: Defects that escape to a subsequent development or delivery phase

Abrams Requirements with Meeting Optimization Model	
Number Reviewers Invited:	Minimum: [n] Maximum: [n]
Avg. Prep Time per Invited (mins):	[n] [n]
Actual Review Meeting Time (mins):	[n] [n]
Target Errors Found:	[n] [n]
Target Total Time Spent (mins):	[n] [n]
Review Cost (Total Time Spent):	[n] [n]

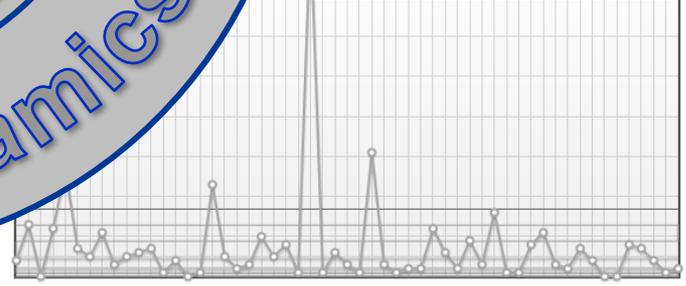
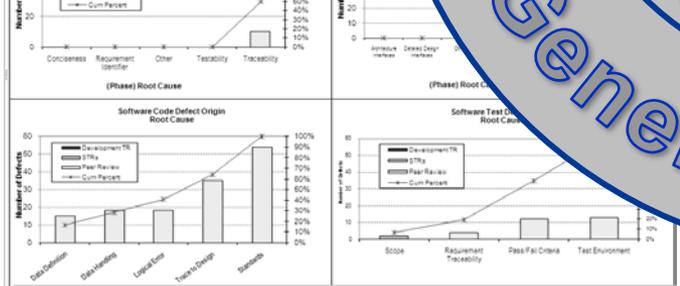


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



DCE Model - Introduction and Definitions

Software Defect Prevention Containment Effectiveness Model

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	94%
Design		98	5	0	0	103	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	13%	96%

Total Defects 715

PCE	68%	95%	78%	96%
DCE		12%	60%	34%

PCE =	Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE =	Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE =	Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects =	Defects that are found during the phase that created them.
Escaped Defects =	Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

- **Phase Containment Effectiveness (PCE)**: tracks the ability of each phase to find defects before they escape that phase
- **Defect Containment Effectiveness (DCE)**: tracks the ability of each phase to find defects passed to it by upstream phases
- **Total Containment Effectiveness (TCE)**: tracks the ability of the project to find defects before they are released (post-release)

- Source data is Major and Minor defects collected from the peer review database and problem reporting database
- **Phase Defects**: The defects that were found in a specified software development lifecycle phase
- **Post Release Defects**: The defects that were found in a phase outside of the software development lifecycle phase
- **PreRelease Defects**: Defects that were found in a software development lifecycle phase in the current software version

DCE Model - Goals

Software Defect Prevention Containment Effectiveness Model

1. Early Defect Detection

Origin Phase	Phase Found In						Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration				
Requirements	181	10	43	15	17	266	37%	94%	
Design		98	5	0	0	103	14%	100%	
Code			200	15	40	255	36%	84%	
Test				87	4	91	13%	96%	
						Total Defects: 715			

	Requirements	Design	Code	Software Test	Post Software Iteration
PCE	68%	95%	78%	96%	
DCE		12%	60%	34%	

2. Defect Prevention

PCE =	Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE =	Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE =	Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects =	Defects that are found during the phase that created them.
Escaped Defects =	Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

Goals of the Model:

1. Early Defect Detection

- Increase the number of defects found within the phase they originated in
 - Phase Containment Effectiveness (PCE)
- Reduce the number of defects escaping to later development phases
 - Defect Containment Effectiveness (DCE)

2. Defect Prevention

- Reduce total number of defects originating within a specific phase
 - Total Defects
 - % Insertion Rate

DCE Model - Usage

When Used:

- Data collection and analysis occurs after a software release

Used For:

- Identifying organizational process and technology improvement needs
- Planning of subsequent product release(s)
 - Consider if a significant number of defects are detected in any particular life cycle phase
 - Ensure common causes of defects within a phase or phases have been identified and actions to further investigate causes of defects are assigned and tracked to completion
 - Understand the implications of not addressing the common causes of defects
 - Ensure implementation of preventive corrective actions (e.g. notify developers of common defects, increase focus on suspect or problem areas)

DCE Model – Analysis Examples

Software Defect Prevention Containment Effectiveness Model

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	94%
Design		98	5	0	0	103	14%	100%
Code			200	15	40	265	36%	84%
Test				87	4	91	13%	96%

Total Defects: 715

PCE	68%	95%	78%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE = Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE = Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects = Defects that are found during the phase that created them.
Escaped Defects = Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

Is the number of post release defects significantly higher than expected?

Is the number of post release defects significantly lower than expected?

What is the phase with the lowest PCE percentage? Consider how early in the life cycle this phase is.

How does the PCE, DCE and TCE compare to the last release or other projects?

Defect Containment Effectiveness Model Considerations for Analysis

Software Defect Prevention Containment Effectiveness Model

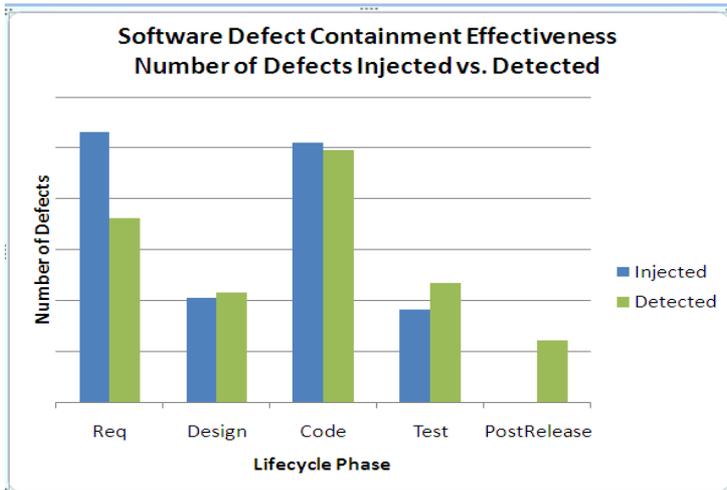
Origin Phase	Phase Found In						Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration				
Requirements	181	10	43	15	17	266	37%	84%	
Design		98	5	0	0	103	4%	100%	
Code			200	15	40	255	36%	14%	
Test				87	4	91	12%	98%	

Total Defects: 715

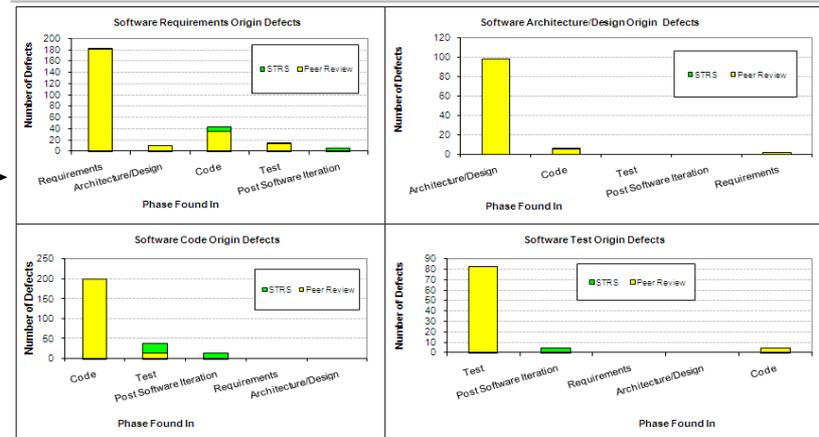
PCE	68%	95%	78%	98%
DCE		12%	69%	34%

PCE = Phase Containment Effectiveness [(PCE = Contained Defects / Total Defects) * 100%]
 DCE = Defect Containment Effectiveness [(DCE = Phase Defects / (Phase Defects + Post Release Defects)) * 100%]
 TCE = Total Containment Effectiveness [(TCE = Pre-Release Defects / Total Defects) * 100%]
 Contained Defects = Defects that are found during the phase that created them.
 Escaped Defects = Defects that escape to a subsequent development or delivery phase.

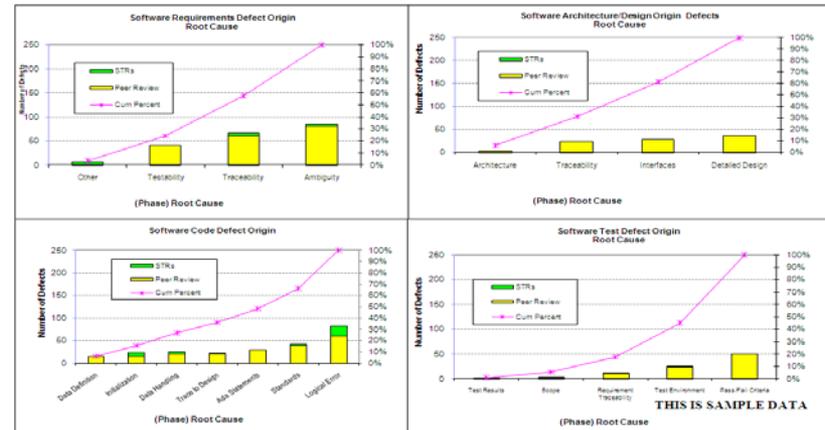
*THIS IS SAMPLE DATA



Software Defect Prevention Escaped Defects by Origin



Software Defect Prevention Root Cause by Origin



DCE Model – Benefits and Challenges

● Benefits

- One model that can be used as a basis of identifying areas for both early defect detection and defect prevention
- Provides an overall view of a project's quality with many ways to slice and dice the data
- Similar project data can be rolled up as appropriate

● Challenges

- The model can be complex to understand, so time is needed to educate users
- Formula variables need to be well defined and understood before generating the model
- Shows the macro view of defects, so a change in the model cannot directly be linked to a specific improvement
 - Not measuring a sub-process

Process Performance Models (PPM)

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	94%
Design		88	5	0	0	93	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	13%	98%

Total Defects 715

PCE	68%	95%	70%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
 DCE = Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
 TCE = Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
 Contained Defects = Defects that are found during the phase that created them
 Escaped Defects = Defects that escape to a subsequent development or delivery phase

Software Systems Metrics

Abrams Requirements with Meeting Optimization Model

Number Reviewers Invited: Minimum: [5] Maximum: [15]

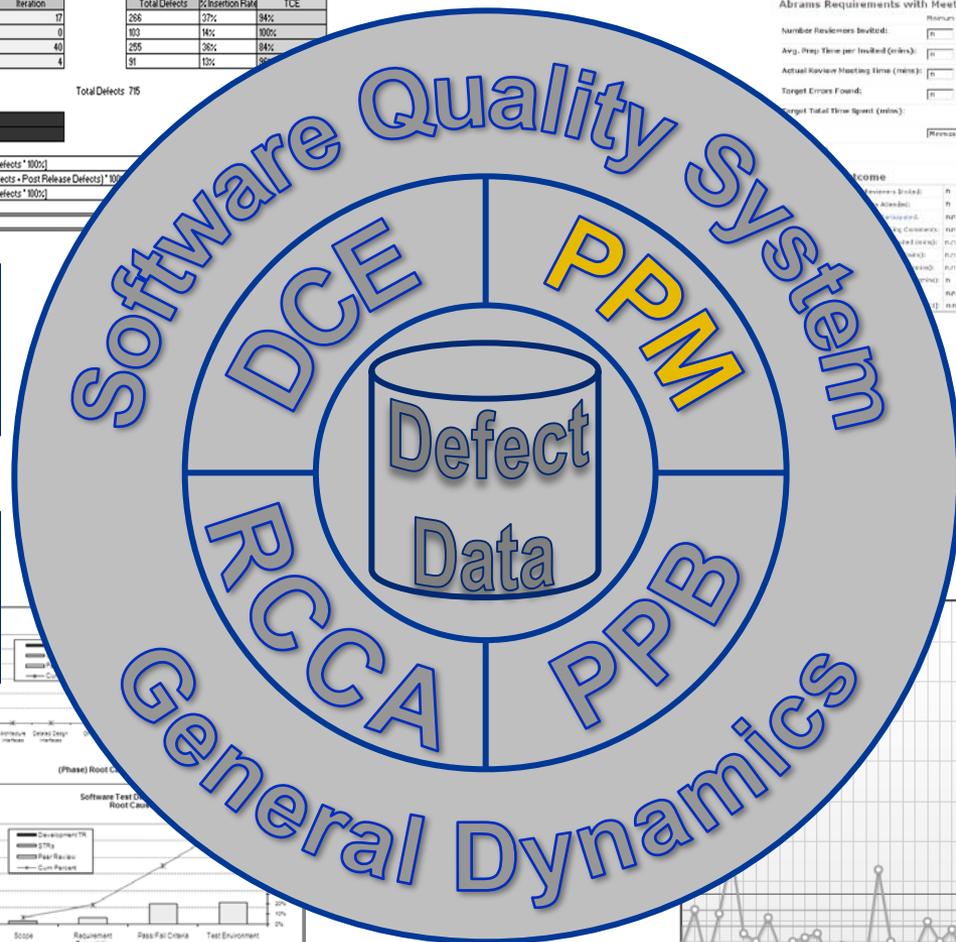
Avg. Prep Time per Invited (min): [15] [15]

Actual Review Meeting Time (min): [15] [15]

Target Errors Found: [15] [15]

Target Total Time Spent (min): [15] [15]

Previous Cost (Total Time Spent): [15]

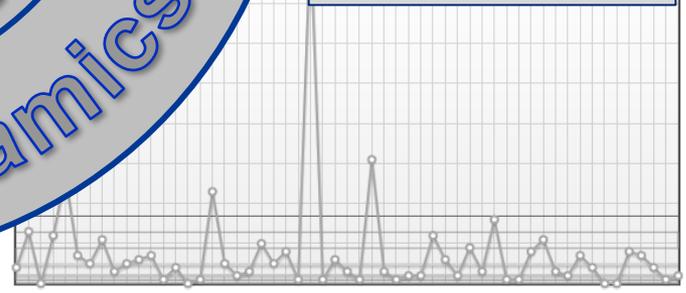
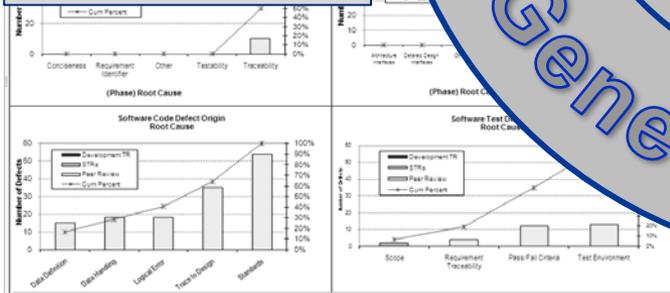


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



PPM – Early Defect Detection

Software Defect Prevention Containment Effectiveness Model

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	37%	94%
Design		98	5	0	0	103	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	13%	96%

Total Defects: 715

PCE	68%	95%	78%	96%
DCE		12%	60%	34%

PCE =	Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE =	Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE =	Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects =	Defects that are found during the phase that created them.
Escaped Defects =	Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

Peer Review Process Performance Models were first created for Requirements in order to improve the Requirements PCE through Early Defect Detection

Software Metrics Tool – PPM Main Page

Code Selection Criteria Averages

With Meeting	Without Meeting
Major/Minor Defects per Review: 0.00	Major/Minor Defects per Review: 0.00
Total Peer Review Time: 0 min	Total Peer Review Time: 0 min
Major/Minor Defects Found per Total Peer Review Time: 0.0 defects/hour	Major/Minor Defects Found per Total Peer Review Time: 0.0 defects/hour
Preparation Recommendations with Meeting:	
» Number of Participants Entering Issues: 0	

Requirements Selection Criteria Averages

With Meeting	Without Meeting
Major/Minor Defects per 100 Requirements per Review: 0.00	Major/Minor Defects per 100 Requirements per Review: 0.00
Total Peer Review Time: 0 min	Total Peer Review Time: 0 min
Major/Minor Defects Found per 100 requirements per Total Peer Review Time: 0.0 defects/hour	Major/Minor Defects Found per 100 requirements per Total Peer Review Time: 0.0 defects/hour
Preparation Recommendations with Meeting:	
» Average Prep Time per Participated: 0 min	
» Software Requirements Reviewed: 0	

Peer Review Model

Action:

Program:

Type:

Meeting:

Families of models based upon:

- Prediction / Optimization / Comparison
- Program (New development & maintenance)
- Requirements / Design / Code
- Meetings / No Meetings

Abrams Requirements Selection Criteria Averages

With Meeting	Without Meeting
Major/Minor Defects per Review: 0.0	Major/Minor Defects per Review: 0.0
Total Peer Review Time: 0 min	Total Peer Review Time: 0 min
Major/Minor Defects Found per Total Peer Review Time: 0.0 defects/hour	Major/Minor Defects Found per Total Peer Review Time: 0.0 defects/hour
Preparation Recommendations with Meeting:	
» Hold Overviews for all Peer Reviews	
» Average Prep Time per Invited: 0 - 0 min	
» Number of Reviewers Entering Issues: 0	

PPM Tool – Prediction

Abrams Requirements with Meeting Prediction Tool

Estimated Number of Reviewers Invited:

Estimated Avg. Prep Time per Invited (mins):

Estimated Review Meeting Time (mins):

Predictions

Number of Reviewers Attended:	n.n
Number of Reviewers Participated:	n.n
Number of Reviewers Making Comments:	n.n
Avg. Prep Time per Attended (mins):	n.n
Avg. Prep Time per Participated (mins):	n.n
Technical Errors [95% CI]:	n.n [+/- n.n]
Total Peer Review Time (mins) [95% CI]:	n.n [+/- n.n]

User can enter the tool on their own or is automatically routed here at the start of a peer review.

User can predict peer review results based upon number of reviewers invited, preparation time spent by the reviewers, and meeting time.

PPM Tool – Optimize Time

Abrams Requirements with Meeting Optimization Model

	Minimum	Maximum
Number Reviewers Invited:	<input type="text" value="n"/>	<input type="text" value="n"/>
Avg. Prep Time per Invited (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Actual Review Meeting Time (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Target Errors Found:	<input type="text" value="n"/>	
Target Total Time Spent (mins):		<input type="text" value="n"/>
Goal:	<input type="text" value="Maximize Errors Found"/>	



Optimized Outcome

Recommended Number of Reviewers Invited:	n
Expected Number of Reviewers Attended:	n
Expected Number of Reviewers Participated:	n.n
Expected Number of Reviewers Making Comments:	n.n
Recommended Avg. Prep Time per Invited (mins):	n.n
Expected Avg. Prep Time per Attended (mins):	n.n
Expected Avg. Prep Time per Participated (mins):	n.n
Recommended Actual Review Meeting Time (mins):	n
Expected Technical Errors [95% CI]:	n.n [+/- n.n]
Expected Total Peer Review Time (mins) [95% CI]:	n.n [+/- n.n]

PPM Tool – Post Peer Review Comparison

Peer Review Tool logged in as sobetski

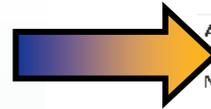
home | create peer review | search & reports | edit groups | ips process | user guide | feedback

Main | Items | Participants | Schedule | Issues | Prep Time | **Close Meeting** | **Close Review**

Excel Report | Export Issues (Note: Report may contain technical data - ITAR rules apply)

NEW! Click here to hide the 'Add Review Issue' and 'Upload Issues' forms to make it easier to view and scroll through issues (only filters will remain).

Add Review Issue	
* Item:	NDA CMMI Presentation Abstract
Submitted By:	Corr, Margaret E
Accept:	Undecided
Issue Category:	[Select One]
Issue Severity:	[Select One]
Location in Item:	ec Requirement#, Page#, Section
* Issue:	



Software Systems Metrics

Home Projects Manpower Resources Efficiency

Abrams Requirements with Meeting Comparison Tool

	Actual	Var (%)
Review Meeting Time (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Avg. Prep Time per Invited (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Number Reviewers Invited:	<input type="text" value="n"/>	<input type="text" value="n"/>
Number Reviewers Making Comments:	<input type="text" value="n"/>	<input type="text" value="n"/>
Technical Errors:	<input type="text" value="n"/>	
Total Peer Review Time (mins):	<input type="text" value="n"/>	

Comparison

	Predicted	Actual
Technical Errors [95% CI]:	n.n [+/- n.n]	n
Total Peer Review Time (mins) [95% CI]:	n.n [+/- n.n]	n

1. Based on the comparison data was this peer review effective?
2. Will a re-review be held?
3. Describe decisions or analysis made as a result of using the Comparison Tool.

Users are directed at peer review closure from the Peer Review Tool to the Metrics Tool.

Data is automatically filled in for the user from the Peer Review Tool to determine if the review was within the baseline.

Users are required to provide analysis information on their results.

PPM – Benefits and Challenges

● Benefits

- ↗ Optimize resource utilization
- ↗ Better able to predict future project performance
- ↗ PPM improvements returned as high as 400% increase in technical defect detection
- ↗ Exceeded our defect Phase Containment Effectiveness (PCE) goal of 90%.

● Challenges

- ↗ PPM Model usage
 - Making it user-friendly and easily accessible
 - Documenting usage
- ↗ Stakeholder buy-in
- ↗ Keeping it from being personal, i.e. measuring the process & product and not the person performing the work
- ↗ Ensuring data integrity in the models

Process Performance Baselines (PPB)

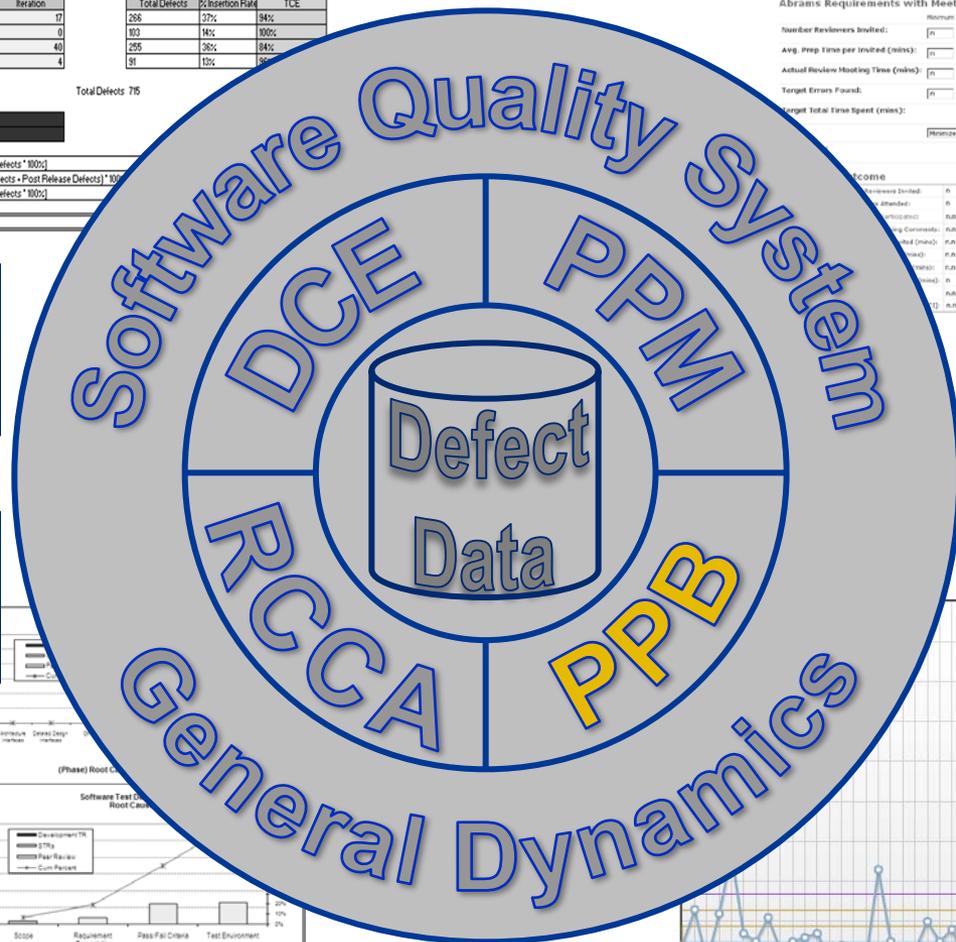
Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	33%	84%
Design		88	5	0	0	93	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	12%	98%

Total Defects 715

PCE	68%	95%	70%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
 DCE = Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
 TCE = Total Containment Effectiveness [TCE = Pre-Release Defects / Total Defects * 100%]
 Contained Defects = Defects that are found during the phase that created them
 Escaped Defects = Defects that escape to a subsequent development or delivery phase

Abrams Requirements with Meeting Optimization Model	
Number Reviewers Invited:	Minimum: [n] Maximum: [n]
Avg. Prep Time per Invited (mins):	[n] [n]
Actual Review Meeting Time (mins):	[n] [n]
Target Errors Found:	[n]
Target Total Time Spent (mins):	[n]
Review Cost (Total Time Spent):	[n]

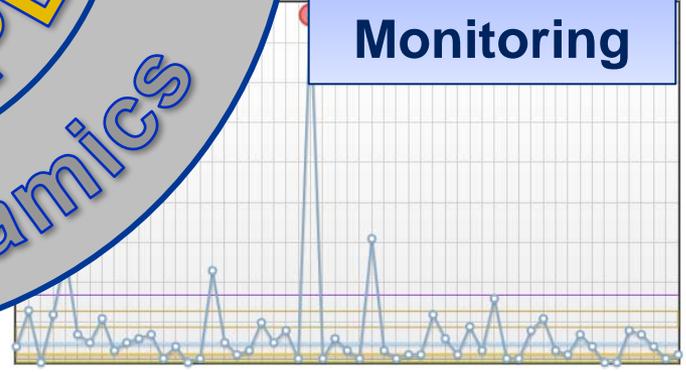
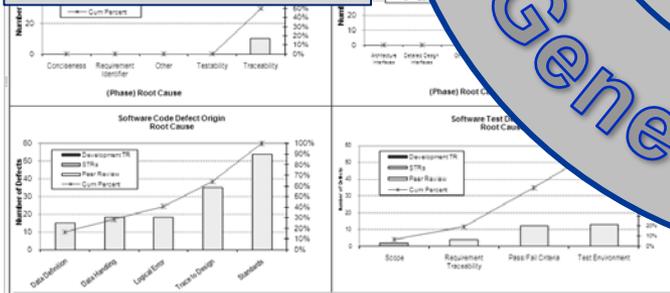


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



Process Performance Baselines (PPBs) – Using PPM Tool Post Peer Review Comparison Analysis

Software Systems Metrics

Home Projects Manpower Resources Efficiency

Abrams Requirements with Meeting Comparison Tool

	Actual	Var (%)
Review Meeting Time (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Avg. Prep Time per Invited (mins):	<input type="text" value="n"/>	<input type="text" value="n"/>
Number Reviewers Invited:	<input type="text" value="n"/>	<input type="text" value="n"/>
Number Reviewers Making Comments:	<input type="text" value="n"/>	<input type="text" value="n"/>
Technical Errors:	<input type="text" value="n"/>	
Total Peer Review Time (mins):	<input type="text" value="n"/>	

Comparison

	Predicted	Actual
Technical Errors [95% CI]:	n.n [+/- n.n]	n
Total Peer Review Time (mins) [95% CI]:	n.n [+/- n.n]	n

1. Based on the comparison data was this peer review effective?

2. Will a re-review be held?

3. Describe decisions or analysis made as a result of using the Comparison Tool.

Data is automatically provided to the PPBs when the peer review is completed.

Users provide analysis information on their results. Special causes of variation can be identified and explained at the time the data was created.

PPBs – Automated Metrics

Peer Review Process Performance Baseline

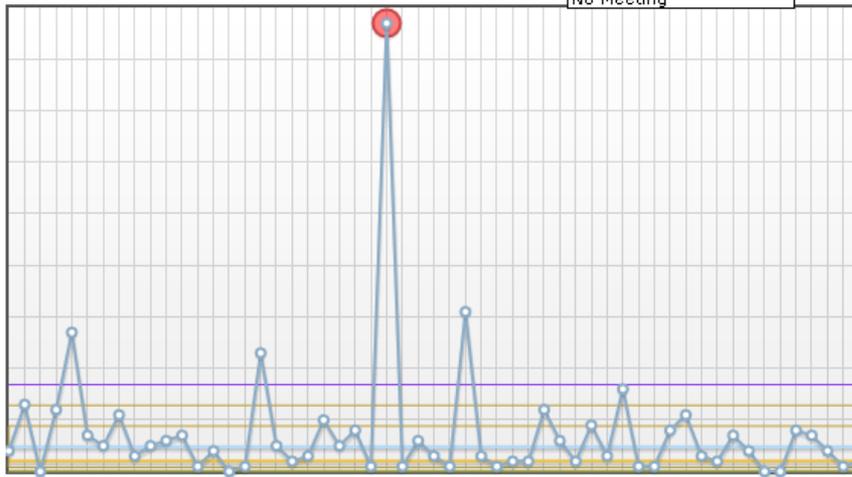
Project: myDefault

Product: Type:

- Preparation (Meeting)
- Meeting
- No Meeting

T
e
c
h
n
i
c
a
l

I
s
s
u
e
s



Peer Review (Ordered by Date)

Show

Check/Uncheck All

Goal

Avg Tech Issues (\bar{X})

Deviation

Lower Control Limit

Upper Control Limit

Sigma Lines (σ)

Technical Issues

Legend

Goal

Avg Tech Issues (\bar{X})

Deviation

Lower Control Limit

Upper Control Limit

Sigma Lines (σ)

Technical Issues

Statistical Process Control charts are generated live from Peer Review Data

User selects

- Project
- Work product type
- Meeting information
- Technical Defects or Cost (hours)
- Properties to display

PPBs – Analysis

Peer Review Process Performance Baseline

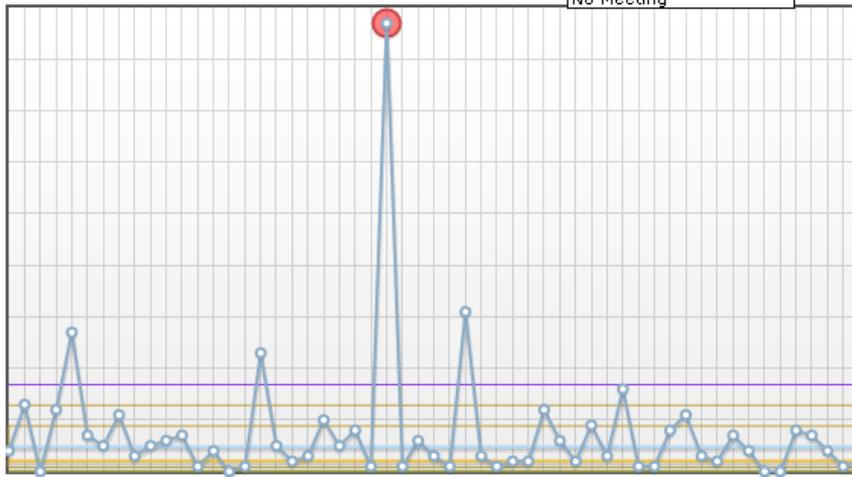
Project: myDefault

Product: Type:

- Preparation (Meeting)
- Meeting
- No Meeting

T
e
c
h
n
i
c
a
l

I
s
s
u
e
s



Peer Review (Ordered by Date)

Show

- Check/Uncheck All
- Goal
- Avg Tech Issues (\bar{X})
- Deviation
- Lower Control Limit
- Upper Control Limit
- Sigma Lines (σ)
- Technical Issues

Legend

- Goal
- Avg Tech Issues (\bar{X})
- Deviation
- Lower Control Limit
- Upper Control Limit
- Sigma Lines (σ)
- Technical Issues

Analysis

- Look for special cause variation
- Monitor for trends, shifts or drifts that may highlight process changes
- Determine if re-review should be performed

PPBs – Benefits and Challenges

● Benefits

- ↗ Peer reviews automatically feed PPBs
- ↗ Control charts are automated to generate from live peer review data
- ↗ Process is monitored to ensure process is performing as expected
- ↗ Quickly deal with process deviations
- ↗ Special causes of variation can be identified and documented as they occur

● Challenges

- ↗ Educating employees on the use of the PPBs
- ↗ Determining when to set new process baselines
- ↗ Determining where to segregate the data
- ↗ Ensuring data integrity
- ↗ Upkeep of statistical data (UCL, Mean, Std Dev)

Root Cause & Corrective Actions

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Requirements	181	10	43	15	17	266	33%	84%
Design		88	5	0	0	93	14%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	12%	98%

Total Defects 715

PCE	68%	95%	70%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
 DCE = Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
 TCE = Total Containment Effectiveness (TCE = Pre-Release Defects / Total Defects * 100%)
 Contained Defects = Defects that are found during the phase that created them
 Escaped Defects = Defects that escape to a subsequent development or delivery phase

Software Systems Metrics

Abrams Requirements with Meeting Optimization Model

Number Reviewers Invited: Minimum Maximum

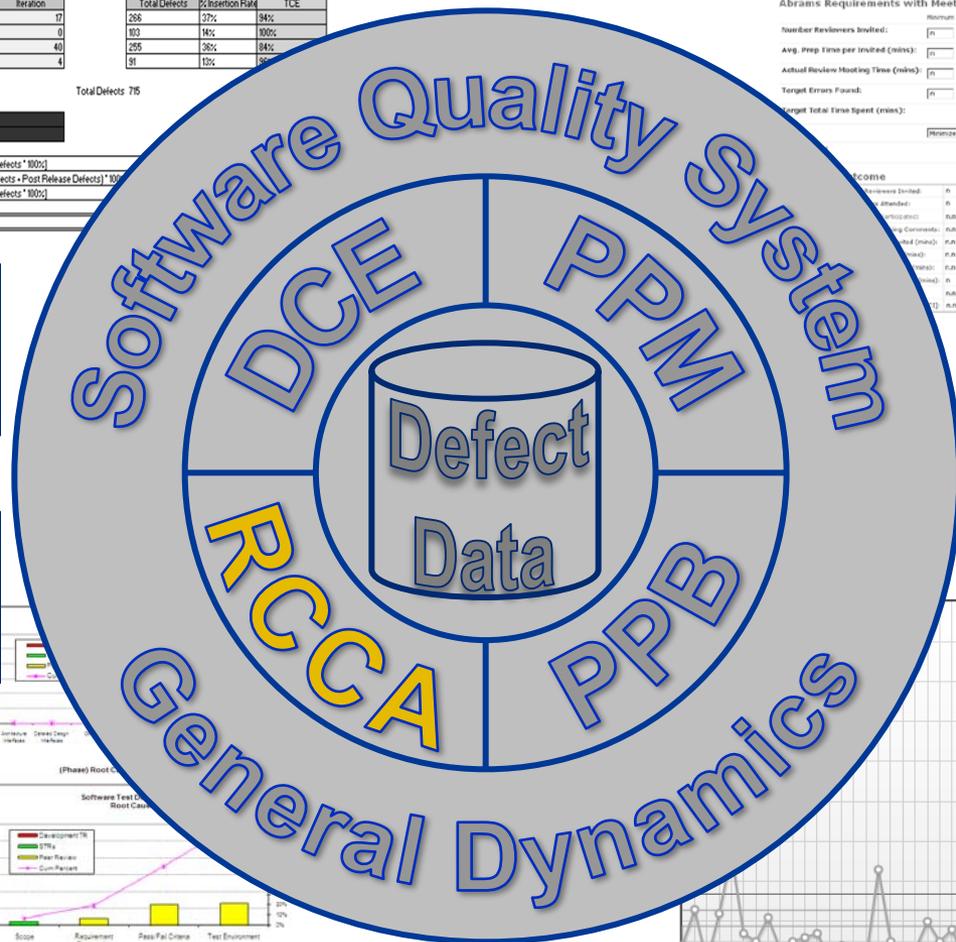
Avg. Prep Time per Invited (mins): Minimum Maximum

Actual Review Meeting Time (mins): Minimum Maximum

Target Errors Found: Minimum Maximum

Target Total Time Spent (mins): Minimum Maximum

Minimize Cost (Total Time Spent)

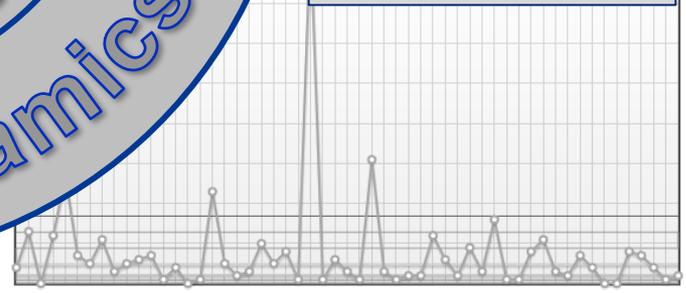
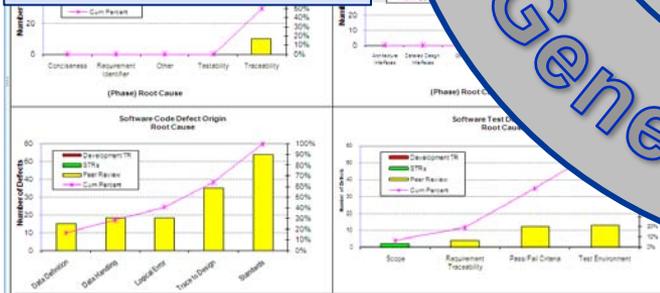


DCE
Identify

PPM
Optimization

RCCA
Prevention

PPB
Monitoring



Root Cause and Corrective Action (RCCA) – Defect Prevention Using the DCE Model

- Considerations for analysis
 - Common Cause Defects
 - Critical Defects

Software Defect Prevention Containment Effectiveness Model

Origin Phase	Phase Found In						Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration				
Requirements	181	10	43	15	17	266	37%	94%	
Design		98	5	0	0	103	14%	100%	
Code			200	15	40	255	36%	84%	
Test				87	4	91	13%	96%	

Total Defects 715

PCE	68%	95%	78%	96%
DCE		12%	60%	34%

PCE =	Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE =	Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE =	Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects =	Defects that are found during the phase that created them.
Escaped Defects =	Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

Root Cause and Corrective Action (RCCA) – Steps For Performing RCCA

1. Identify a Problem
2. Defect Root Cause Analysis
3. Fishbone Analysis
4. Identify Corrective Actions
5. Verify Corrective Actions

Root Cause and Corrective Action (RCCA)

Step 1: Identify a Problem - Example

Software Defect Prevention Containment Effectiveness Model

Origin Phase	Phase Found In						Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration				
Requirements	181	10	43	15	17	266	37%	94%	
Design		98	5	0	0	103	14%	100%	
Code			200	15	40	255	36%	84%	
Test				87	4	91	13%	96%	

Total Defects 715

PCE	68%	95%	78%	96%	
DCE		12%	60%	34%	

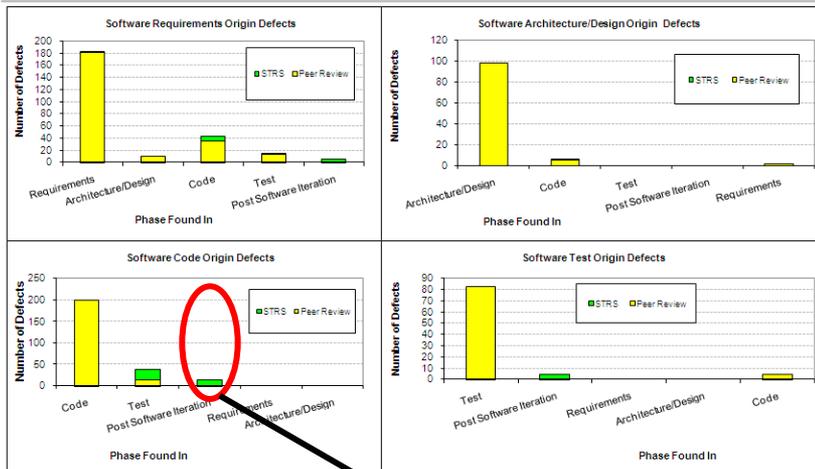
PCE =	Phase Containment Effectiveness [PCE = Contained Defects / Total Defects * 100%]
DCE =	Defect Containment Effectiveness [DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%]
TCE =	Total Containment Effectiveness [TCE = PreRelease Defects / Total Defects * 100%]
Contained Defects =	Defects that are found during the phase that created them.
Escaped Defects =	Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

The DCE Model is showing that 40 Code defects were found in the Post Software Iteration Phase

Root Cause and Corrective Action (RCCA) – Step 2: Defect Root Cause Analysis - Example

Software Defect Prevention
Escaped Defects by Origin

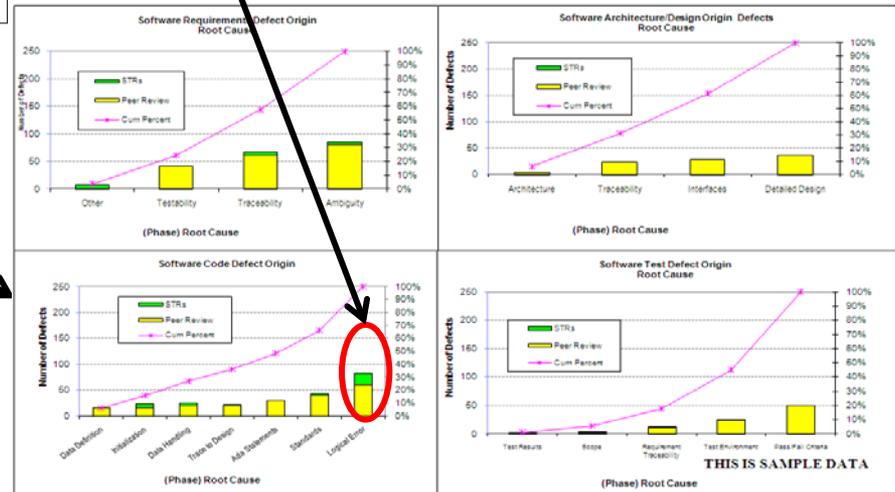


THIS IS SAMPLE DATA

Need to identify the Root Cause of why the defect escaped.

-The Code Defect Origin by Root Cause chart shows that the root cause of the Post Software Iteration Phase defects was Logical Error.

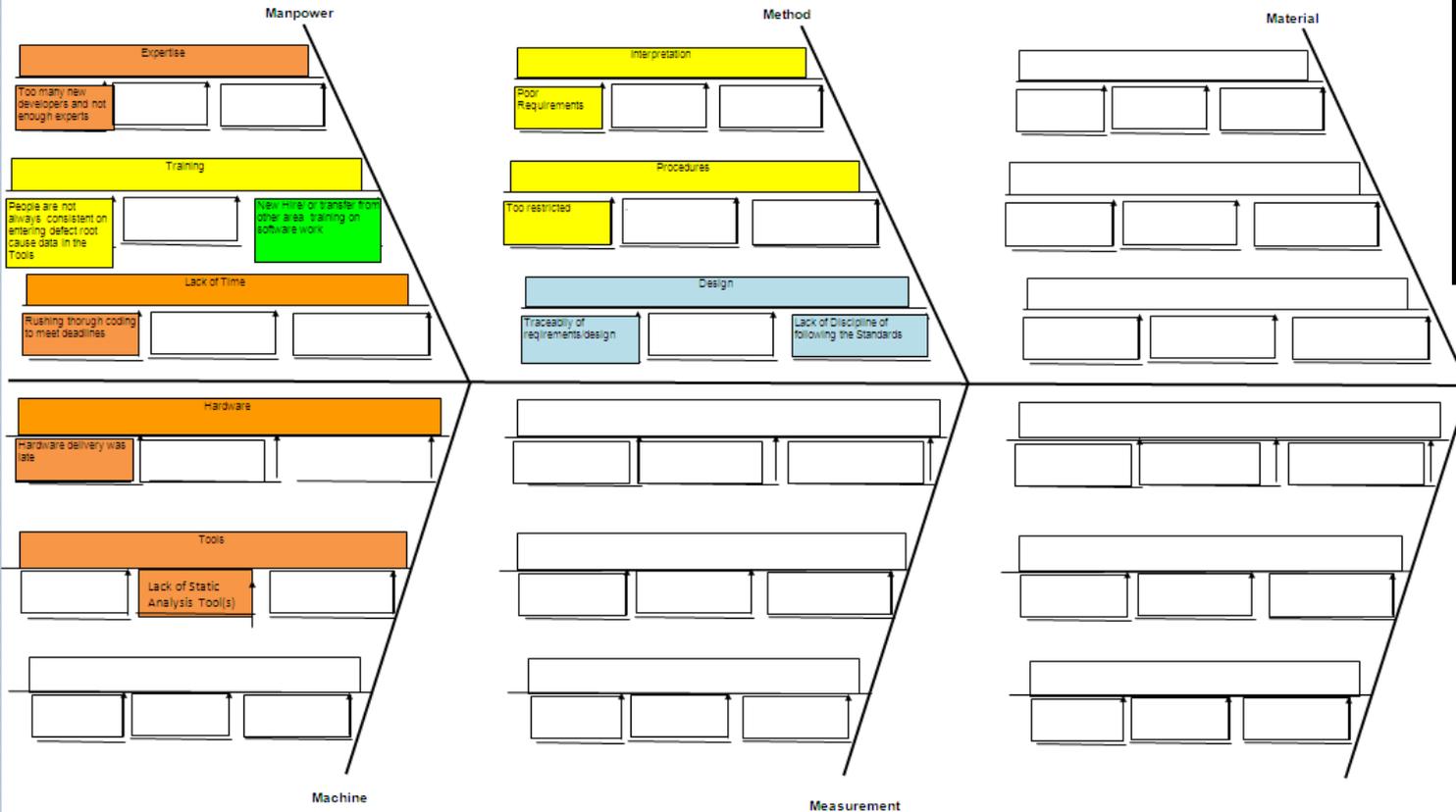
Software Defect Prevention
Root Cause by Origin



THIS IS SAMPLE DATA

Root Cause and Corrective Action – Step 3: Fishbone Analysis - Example

Documenting Cause and Effect ("The Fishbone Diagram")



- 6 Ms**
- Manpower
 - Method
 - Material
 - Machine
 - Measurement
 - Mother Nature

Why were the Logical coding defects injected?

*THIS IS SAMPLE DATA

Likelihood of reoccurring

- Higher Probability
- Medium Probability
- Low Probability
- RuleOut

Root Cause and Corrective Action – Step 4: Identify Corrective Actions - Example

Root Cause: (*usually High Probability)	Corrective Action:
Too many new developers and not enough experts	Add a review of the resource allocation plan
Rushing through coding to meet deadlines	Need to plan more time for peer reviews
Incomplete requirements	Pilot agile development methods
Lack of static analysis tool(s)	Allocate effort and budget to identifying and purchase static analysis tool(s)

*THIS IS SAMPLE DATA

Root Cause and Corrective Action – Step 5: Verify Corrective Actions - Example

Before:

Software Defect Prevention
Containment Effectiveness Model

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Origin								
Requirements	181	10	43	15	17	266	37%	94%
Design	98	5	0	0	0	103	7%	100%
Code			200	15	40	255	36%	84%
Test				87	4	91	13%	96%

Total Defects 715

PCE	68%	95%	78%	96%
DCE		12%	60%	34%

PCE = Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
DCE = Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
TCE = Total Containment Effectiveness (TCE = PreRelease Defects / Total Defects * 100%)
Contained Defects = Defects that are found during the phase that created them.
Escaped Defects = Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

After:

Software Defect Prevention
Containment Effectiveness Model

Origin Phase	Phase Found In					Total Defects	% Insertion Rate	TCE
	Requirements	Design	Code	Software Test	Post Software Iteration			
Origin								
Requirements	181	10	43	15	17	266	38%	94%
Design	98	5	0	0	0	103	13%	100%
Code			200	15	20	235	34%	91%
Test				87	4	91	13%	98%

Total Defects 695

PCE	68%	95%	85%	96%
DCE		12%	60%	45%

PCE = Phase Containment Effectiveness (PCE = Contained Defects / Total Defects * 100%)
DCE = Defect Containment Effectiveness (DCE = Phase Defects / (Phase Defects + Post Release Defects) * 100%)
TCE = Total Containment Effectiveness (TCE = PreRelease Defects / Total Defects * 100%)
Contained Defects = Defects that are found during the phase that created them.
Escaped Defects = Defects that escape to a subsequent development or delivery phase.

*THIS IS SAMPLE DATA

- This model alone cannot be used to verify the results of the corrective action implemented.
- PPBs need to be created/modified at the sub-process level

Root Cause & Corrective Actions – Benefits and Challenges

● Benefits

- Able to eliminate defects and make process improvements to prevent reoccurrence
- DCE Model showing fewer total defects and lower insertion rates

● Challenges

- Focusing on prevention versus a correction
- New methods require training
- Identifying where the best return on investment is
- Maintaining the scope of the effort
- Ensuring involvement of SMEs

Challenges

- Stakeholder buy-in
 - ↗ Understanding the intent and purpose of the quality system
 - Getting past, “Why do I have to do this?”
 - ↗ Keeping it from being personal, i.e. measuring the processes & products and not the person performing the work
- Data Integrity
 - ↗ Trusting the data
- Documenting savings / cost benefit

What Worked Well

- Automation – keep it simple
 - Web-based tools (easy access)
 - Integrated tools
 - Automation leads to data and process consistency
 - Users able to focus on products, not process
- Communication and education
 - Strong and repeated communication with users
 - Educating key stakeholders on the quality system and its benefits to help others buy-in
 - Updating processes and guidelines to identify usage of quality system
 - Incorporating user feedback

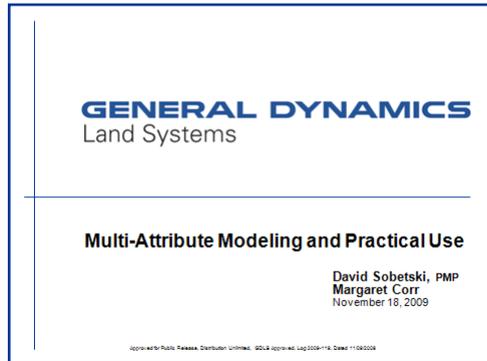
Contact Information

Margaret Corr
Software Engineering Process Group Lead
Section Manager, Software Process, Tools & Environment
38500 Mound Road
Sterling Heights, MI 48310
586-825-5787
corr@gdls.com

Dawn Jaskolski
Software Engineering Process Group Lead
Senior Software Engineer
38500 Mound Road
Sterling Heights, MI 48310
586-825-4418
jaskolsk@gdls.com

David Sobetski, PMP
Sr. Specialist Business Processes, Systems Engineering Process Excellence
38500 Mound Road
Sterling Heights, MI 48310
586-825-5362
sobetski@gdls.com

References



- David Sobetski and Margaret Corr, “Multi-Attribute Modeling and Practical Use” General Dynamics Land Systems November 18, 2009
<http://www.dtic.mil/ndia/2009CMMI/9407WednesdayTrack4Sobetski.pdf>

GENERAL DYNAMICS