

# L&T Infotech Insurance Practice

## Measures of Test Effectiveness

21<sup>st</sup> September, 2007

Anita Kahate

Gaurav Oberoi

Reshma Mane

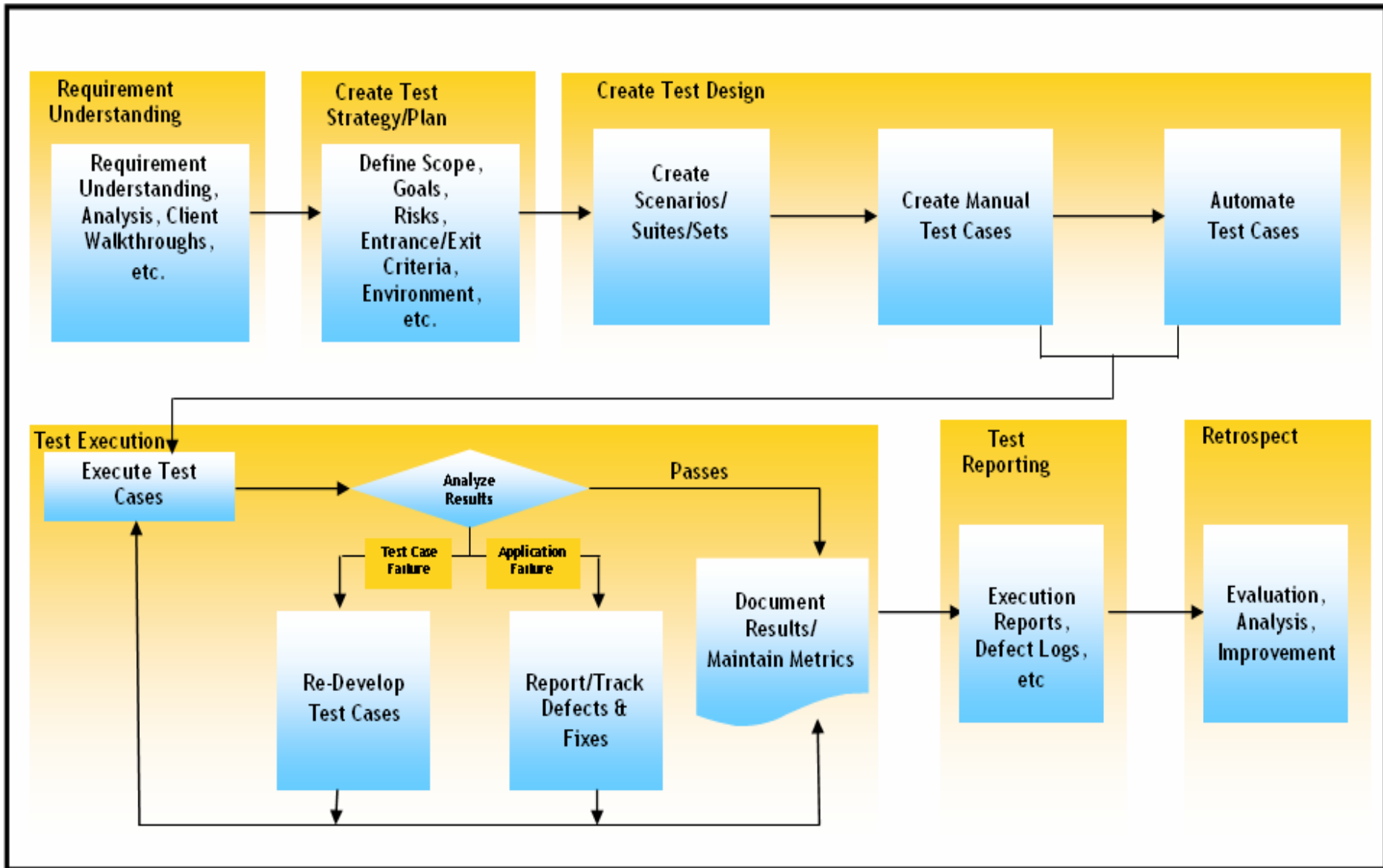


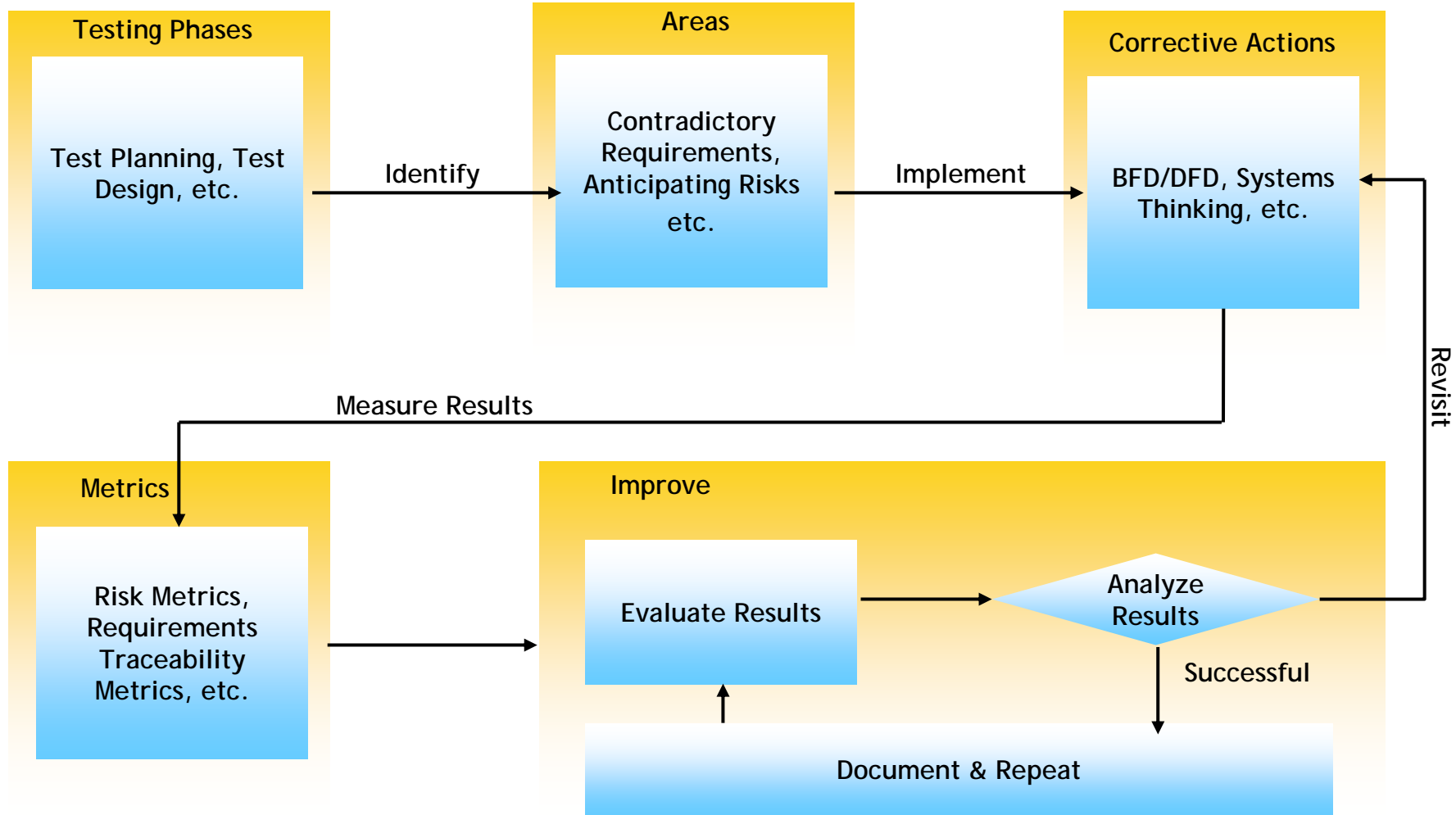
***L&T Infotech***

*You cannot control what you cannot measure. Measurement is the prerequisite of Management Control*

- Tom DeMarco (American Consultant, 1982)

- Introduction
- Testing Process Model
- Contributors Towards Test Effectiveness
- Case Study
- Conclusion



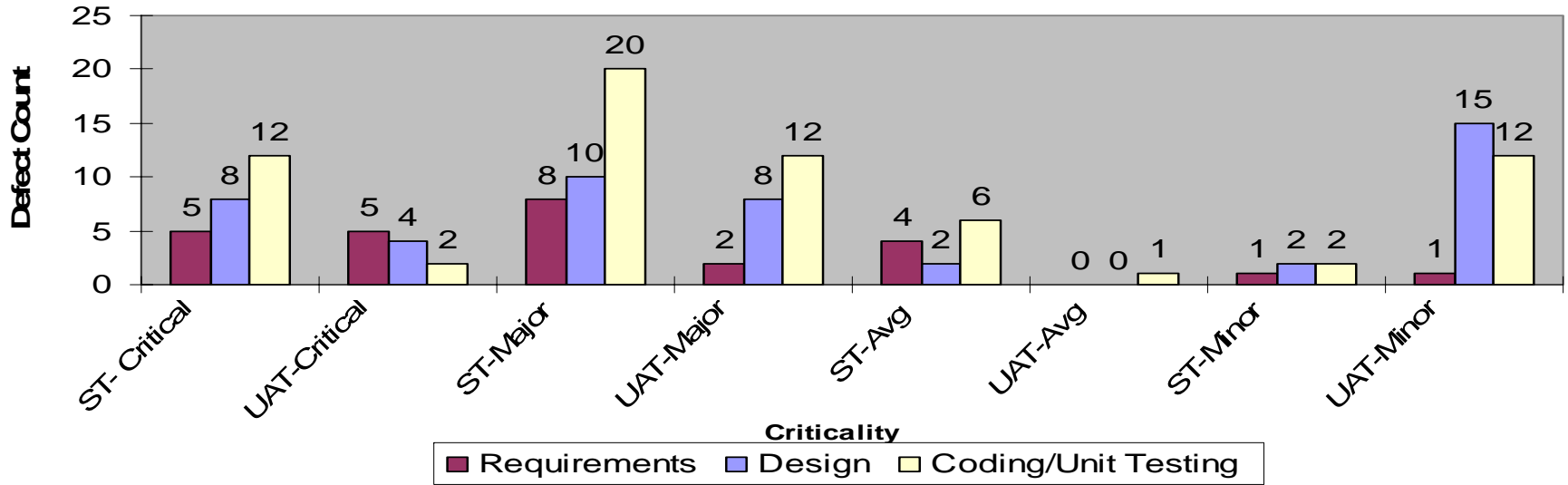


## Phase-wise System Testing Versus User Acceptance Testing Defects

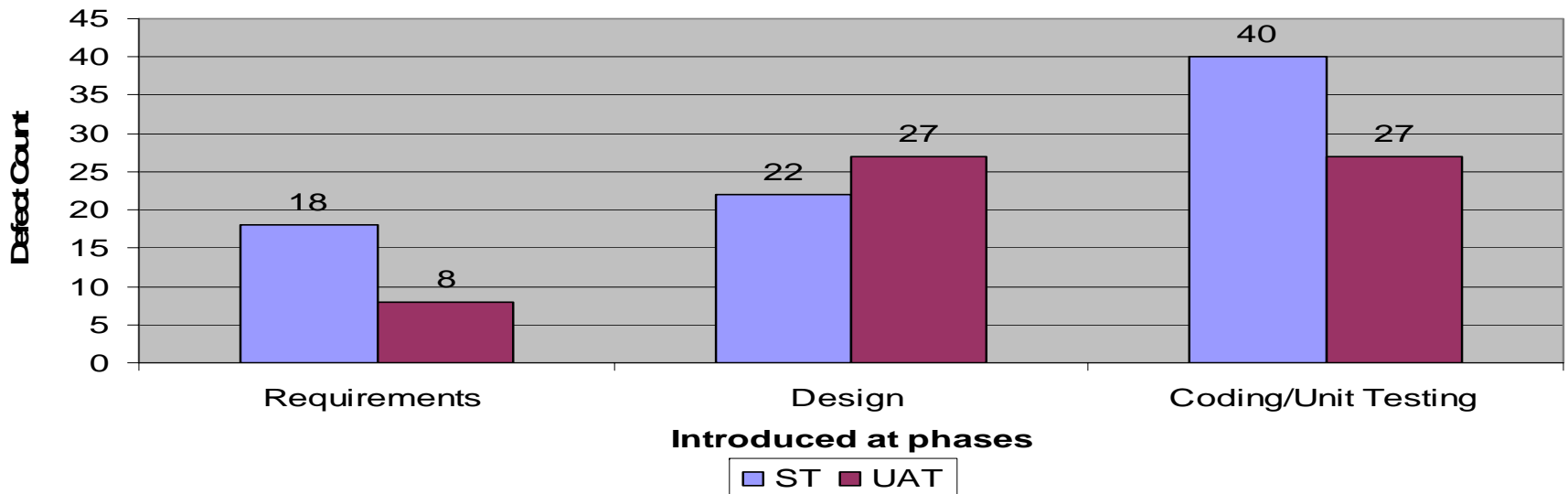
Defects found in System Testing phase					
Phase Introduced	Critical	Major	Average	Minor	Total
Requirements	5	8	4	1	18
Design	8	10	2	2	22
Coding/Unit Testing	12	20	6	2	40
<b>Total</b>	<b>25</b>	<b>38</b>	<b>12</b>	<b>5</b>	<b>80</b>

Defects found in UAT phase					
Phase Introduced	Critical	Major	Average	Minor	Total
Requirements	5	2	0	1	8
Design	4	8	0	15	27
Coding/Unit Testing	2	12	1	12	27
<b>Total</b>	<b>11</b>	<b>22</b>	<b>1</b>	<b>28</b>	<b>62</b>

### Criticality wise Defects in ST and UAT



### Total ST and UAT Defects introduced at different phases



$$\text{System Test Effectiveness} = \frac{\text{Defects found in ST}}{\text{Defects found in ST} + \text{Defects found in UAT}} * 100$$

$$\text{System Test Effectiveness for Release 2} = \frac{80}{80+62} \times 100 = 56.35 \%$$

Observation - Test Effectiveness is 56.35%

**Requirements Analysis/Understanding**

**Incomplete Requirements**

**Intermediate Changes**

**Test Planning & Strategy**

**Anticipating Risks**

**Deviation from Plan**

**Quality level expectation from each cycle**

**Number of Test Cycles**

**Test Design**

**Test Case Coverage**

**Productive & Non-Productive Test Cases**

**Test Execution**

**Test Execution Time**

**Prioritization of Test Cases**

**Test Reporting**

**Reporting Effectiveness**

Consider this specific scenario:

*A requestor sends an ECO request to a project manager. The project lead enters the ECO into the ECO system. If the ECO is incomplete, the ECO manager sends a notice to the requestor. The requestor calls the ECO manager. If the CRB determines that the ECO is urgent, then an immediate meeting is scheduled.*

- ECO - Electronic Change Order
- CRB - Change Request Board

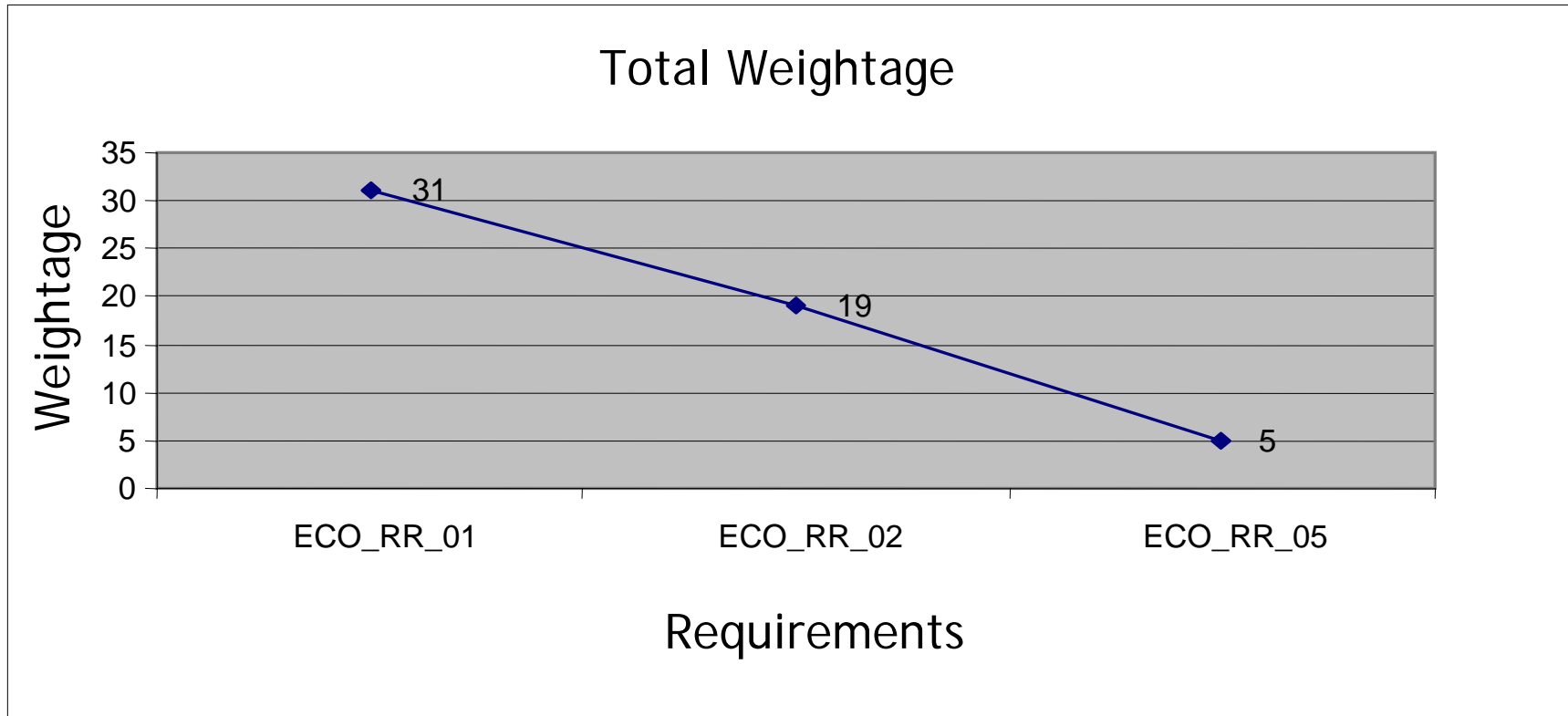
Let us try to put it in Business Flow Diagram (BFD)/Data Flow Diagram (DFD) to catch the problems in the flow



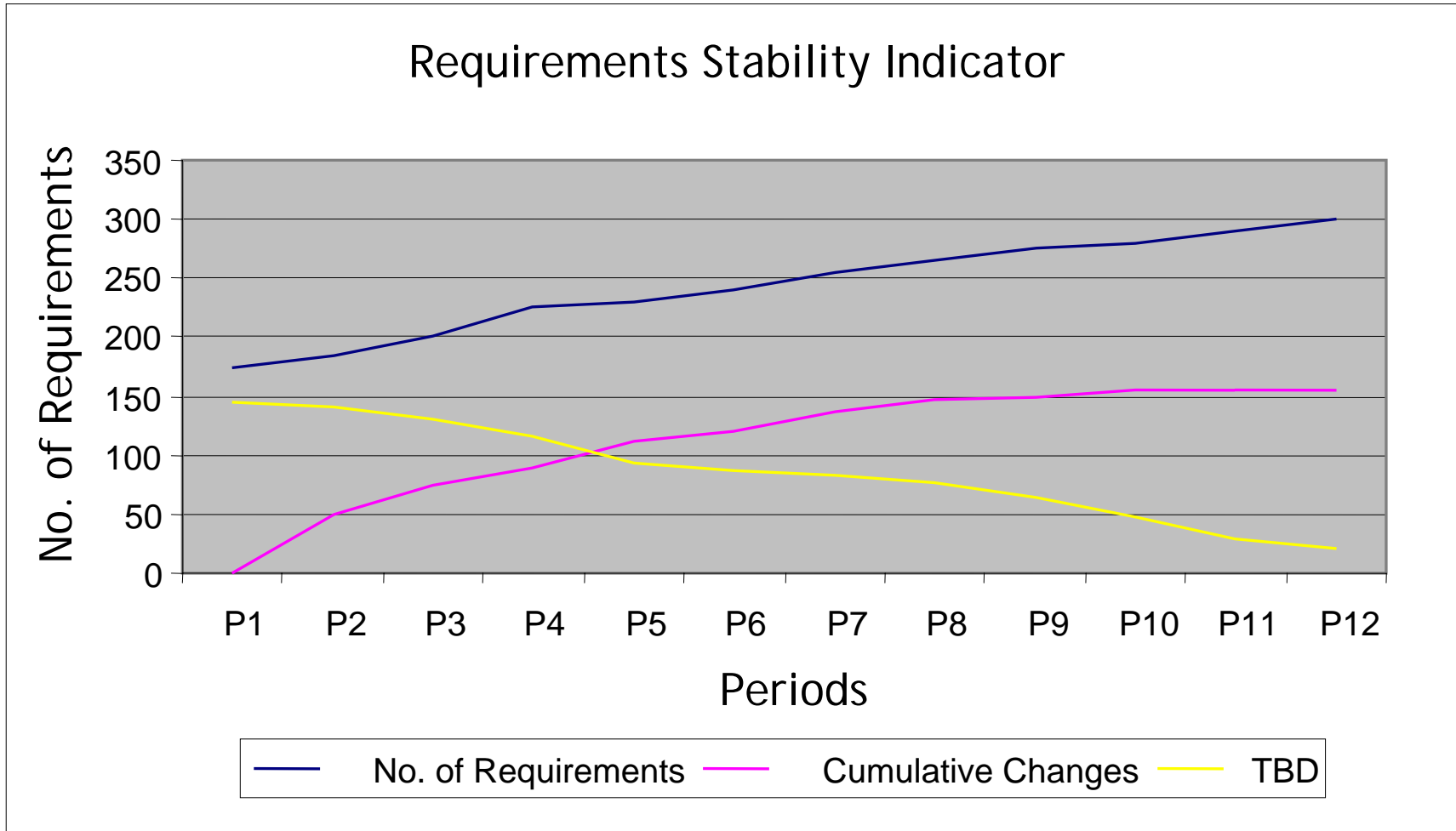
Depicting the above defects in Requirements Defect Metrics (RDM):

Req. Ref.	Flow Break	Inconsistent Naming	Conditional Actor	Incomplete Conditional Output	Non-Specific Transfer of control	No Initiating action	Requirements Language Error
ECO_RR_01	1	1	1	1	1	1	1
ECO_RR_02	0	0	0	0	2	1	1
ECO_RR_05	0	0	0	1	0	0	0
<b>Weightage</b>	<b>6</b>	<b>1</b>	<b>4</b>	<b>5</b>	<b>4</b>	<b>7</b>	<b>4</b>

Req. Ref.	Flow Break	Inconsistent Naming	Conditional Actor	Incomplete Conditional Output	Non-Specific Transfer of control	No Initiating action	Requirements Language Error	Total Weightage
ECO_RR_01	6	1	4	5	4	7	4	31
ECO_RR_02	0	0	0	0	8	7	4	19
ECO_RR_05	0	0	0	5	0	0	0	5



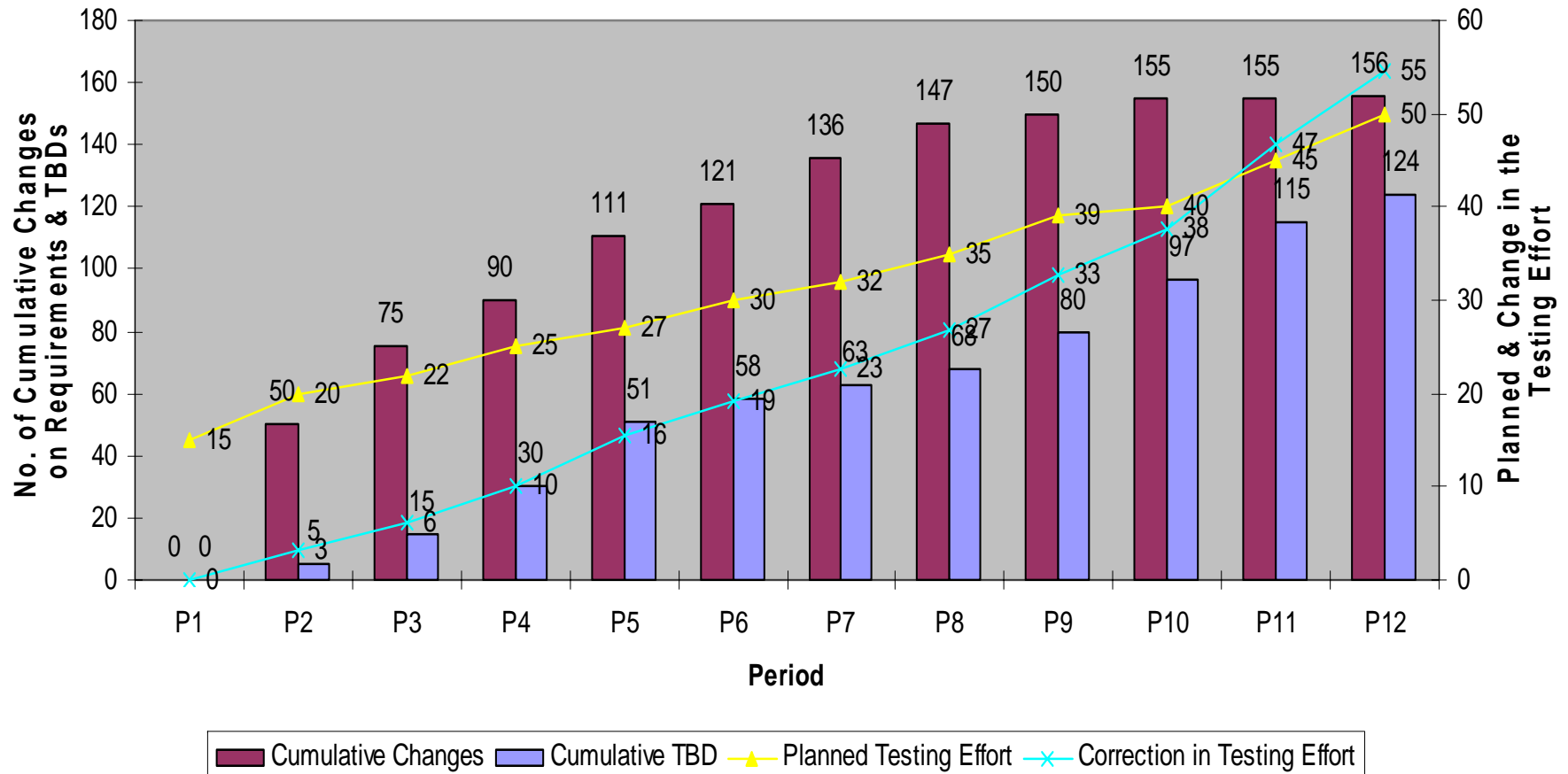
Observation - Req. ECO\_RR\_01 is most incomplete



Periods	No. of Requirements	No. of Changes	Cumulative Changes	TBD	Cumulative TBD	Planned Testing Effort	Correction in Testing Effort
P1	175	0	0	145	0	15	0
P2	185	50	50	140	5	20	3
P3	200	25	75	130	15	22	6
P4	225	15	90	115	30	25	10
P5	230	21	111	94	51	27	16
P6	240	10	121	87	58	30	19
P7	255	15	136	82	63	32	23
P8	266	11	147	77	68	35	27
P9	275	3	150	65	80	39	33
P10	280	2	155	48	97	40	38
P11	290	3	155	30	115	45	47
P12	300	1	156	21	124	50	55

• TBD - To Be Decided

### Impact of Requirement Changes on Testing Effort



**Requirements Analysis/Understanding**

**Incomplete Requirements**

**Intermediate Changes**

**Test Planning & Strategy**

**Anticipating Risks**

**Deviation from Plan**

**Quality level expectation from each cycle**

**Number of Test Cycles**

**Test Design**

**Test Case Coverage**

**Productive & Non-Productive Test Cases**

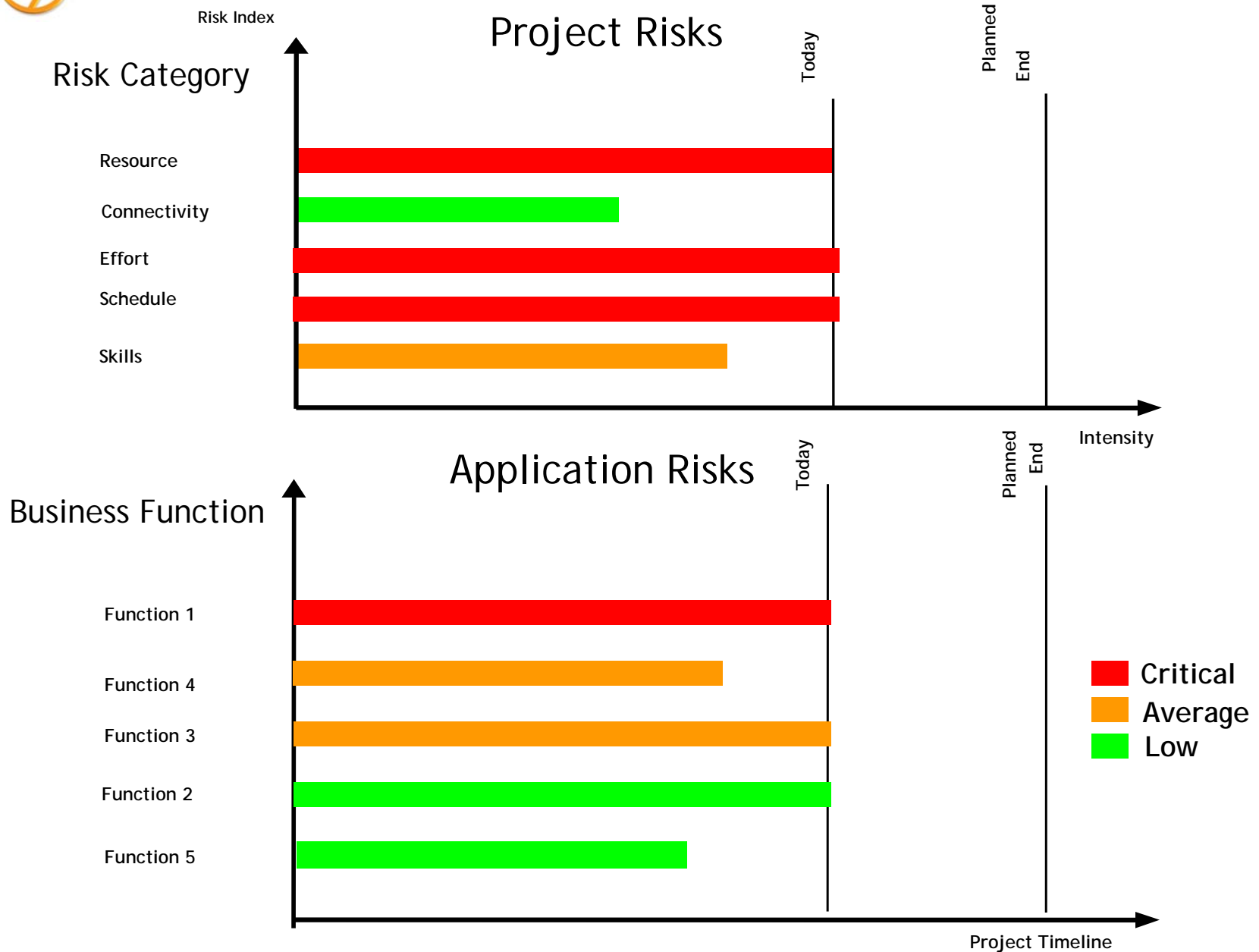
**Test Execution**

**Test Execution Time**

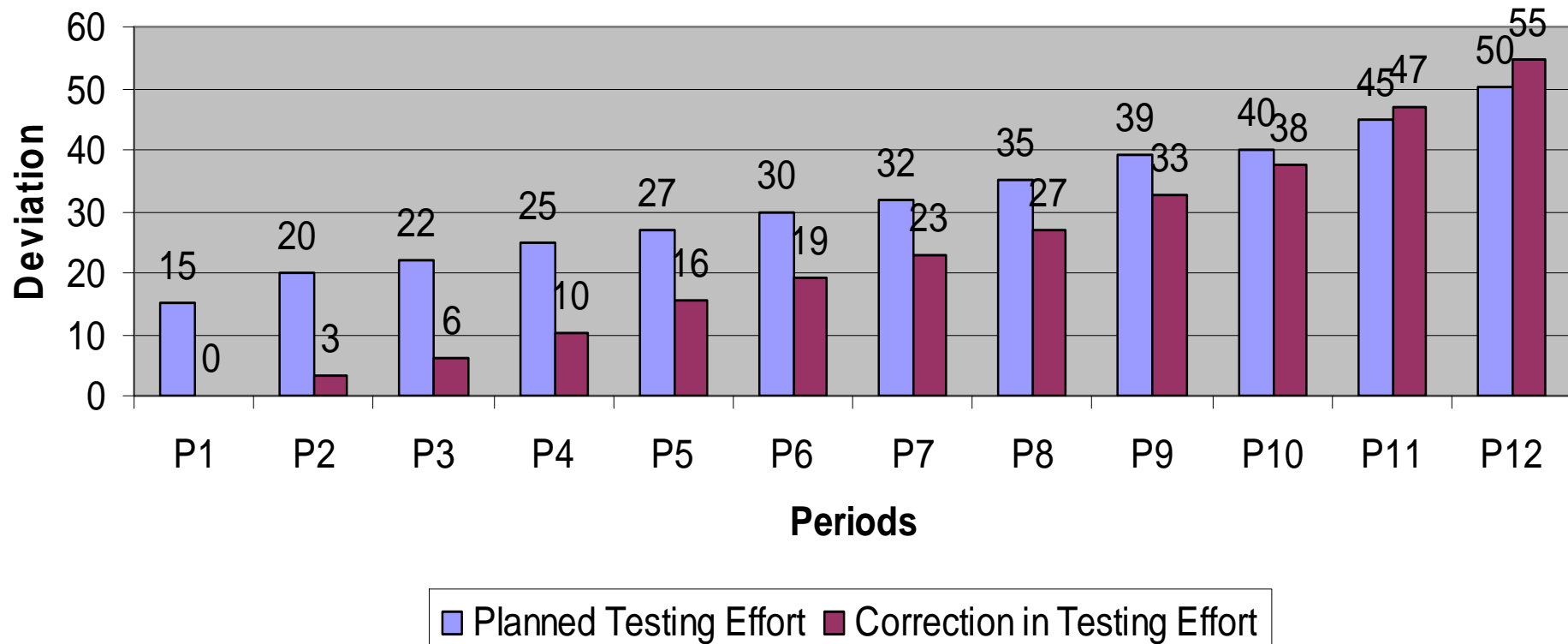
**Prioritization of Test Cases**

**Test Reporting**

**Reporting Effectiveness**

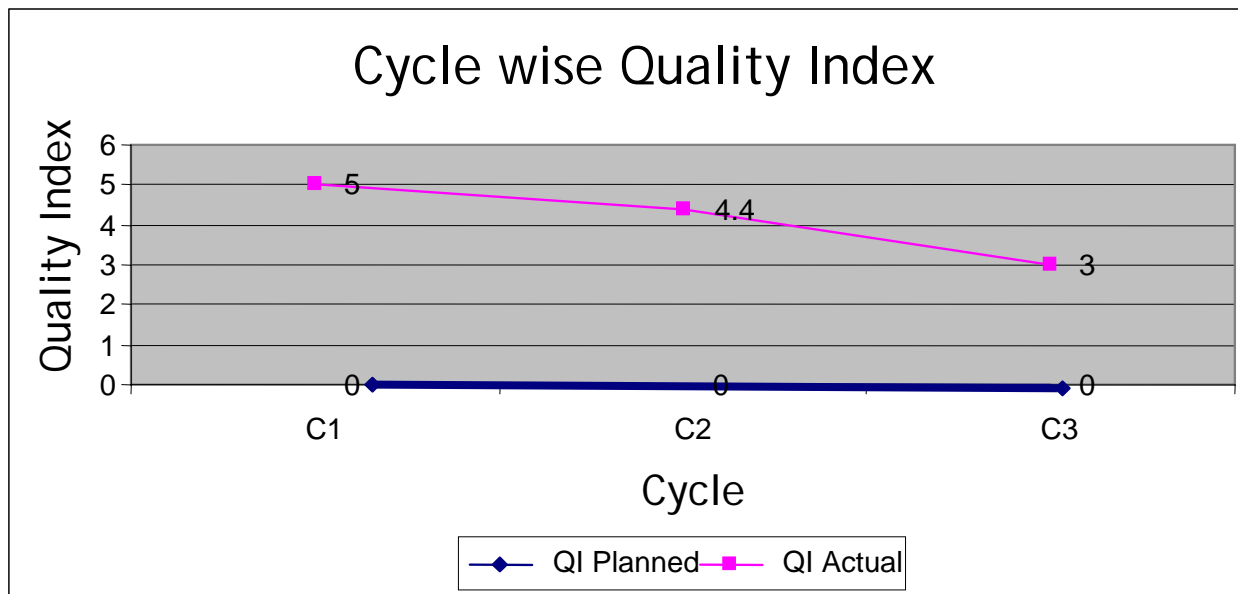


## Deviation from Planned Test Effort



<b>Cycle Ref.</b>	<b>Critical Defects</b>	<b>Major Defects</b>	<b>Average Defects</b>	<b>Minor Defects</b>
Cycle 1	20% should be fixed	20% should be fixed	10% should be fixed	10% should be fixed
Cycle 2	80% should be fixed	60% should be fixed	40% should be fixed	40% should be fixed
Cycle 3	100% should be fixed	100% should be fixed	100% should be fixed	100% should be fixed

Cycles	Critical planned	Critical actual	Major planned	Major actual	Avg planned	Avg actual	Minor planned	Minor actual	Quality Index Planned	Quality Index Actual
C1	20	15	20	15	10	5	10	5	0	5
C2	80	70	60	60	40	40	40	30	0	4.4
C3	100	97	100	97	100	97	100	97	0	3



**Observation - Desired quality level is not achieved even in cycle 3**

## Determining Number of Test Cycles

Indicative values: Based on past Data we have derived following:

Bug Finding Rate	Bug Fix Rate	Bug Insertion Rate
20 defects/cycle	15 defects/cycle	5 defects/cycle

Cycle ref.	Bugs at the beginning of the cycle	Number of bugs fixed	Bugs found in the current cycle	Bugs remaining	Bugs inserted due to fixes	Bugs at the end of cycle
Cycle 1	0	0	25	25	0	25
Cycle 2	25	20	15	20	5	25
Cycle 3	25	25	5	5	5	10

**Requirements Analysis/Understanding**

**Incomplete Requirements**

**Intermediate Changes**

**Test Planning & Strategy**

**Anticipating Risks**

**Deviation from Plan**

**Quality level expectation from each cycle**

**Number of Test Cycles**

**Test Design**

**Test Case Coverage**

**Productive & Non-Productive Test Cases**

**Test Execution**

**Test Execution Time**

**Prioritization of Test Cases**

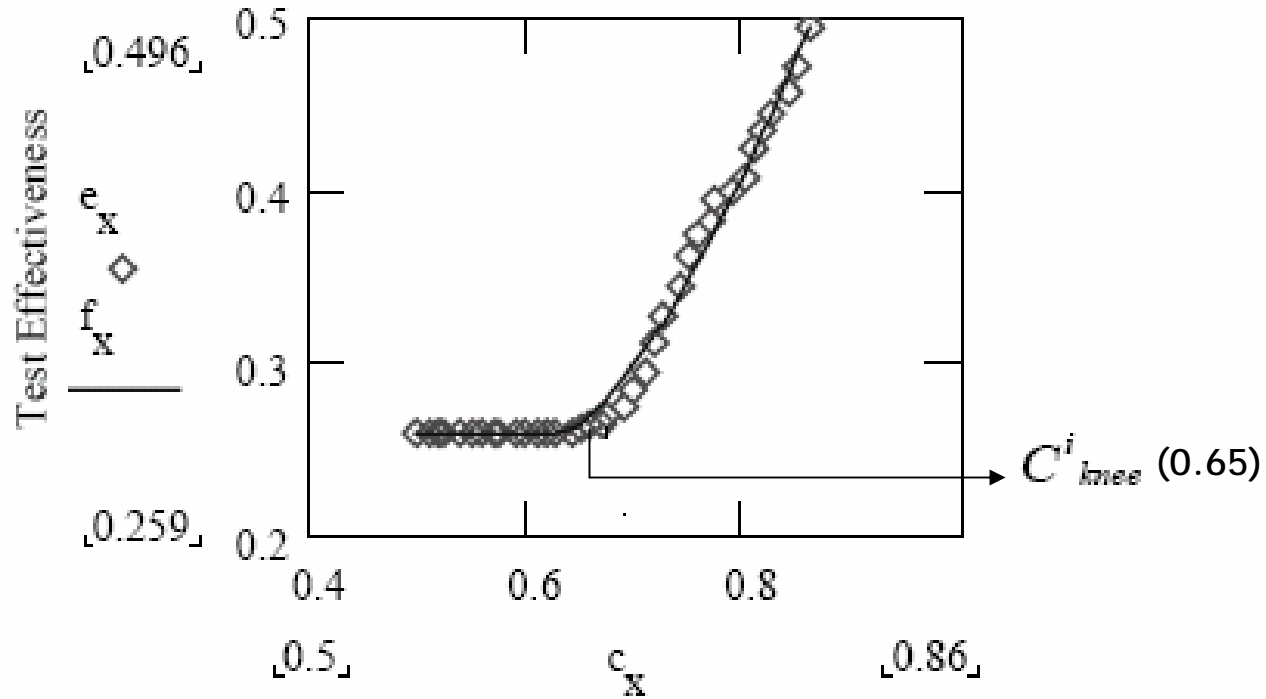
**Test Reporting**

**Reporting Effectiveness**

Defect finding capability of sets of tests with different coverage values has recently been studied by **Frankl and lakounenko**. They have defined measure *test effectiveness*, and have evaluated it for programs containing a single fault. Here we explore the applicability of **Malaiya et al's model** to the test data reported by Frankl and lakounenko.

- The model by Malaiya et al is based on the Logarithmic Poisson reliability growth model, which assumes that the number of defects, found grows logarithmically with the number of tests applied
- They also assumed that the number of coverage enumerables (like decisions) exercised also grows according to a similar logarithmic expression.
- If we eliminate the number of tests and express defect coverage  $C^0$  directly as a function of test coverage  $C^i$ , we can write:

$$C^0 = a_0^i \ln[a_1^i (\exp(a_2^i C^i))] = -A + BC^i \quad C^i > C_{free}^i$$



 Experimental Data  
 Fitted Curve

Defects found by Test Cases					Defects found by other means				
Cycles	Critical	Major	Average	Minor	Cycles	Critical	Major	Average	Minor
Cycle 1	1	3	5	5	Cycle 1	5	2	14	9
Total Defects				14	Total Defects				30

$$\text{Effectiveness of Test Cases} = \frac{\text{Defects found by Test Cases}}{\text{Total Number of Defects found}} \times 100$$

$$\text{Effectiveness of Test Cases} = \frac{14}{14+30} \times 100 = 31.81 \%$$

Observation - Productivity of Test Cases is low,  
need to improve test cases to achieve higher productivity

**Requirements Analysis/Understanding**

**Incomplete Requirements**

**Intermediate Changes**

**Test Planning & Strategy**

**Anticipating Risks**

**Deviation from Plan**

**Quality level expectation from each cycle**

**Number of Test Cycles**

**Test Design**

**Test Case Coverage**

**Productive & Non-Productive Test Cases**

**Test Execution**

**Test Execution Time**

**Prioritization of Test Cases**

**Test Reporting**

**Reporting Effectiveness**

Technique described in "Combinatorial Design Approach to Test Pattern Generation" Bell Labs New Jersey, published in IEEE Software September 1996. Given a set of parameters (X; Y; Z) having values (x1, x2, ...; y1, y2 ...; z1, ...) it is assured that each pair of values (e.g. x1-y1, x1-z1, y2-z3 etc.) is tested at least once with the minimum number of test cases. E.g. with 5 parameters having 4 values each a full systematic combination would require  $4*4*4*4*4 = 1024$  of test data sets. With the optimized 'All pairs' approach the same is reduced to 52.

No. of test data Combinations to execute	No. of test data Combinations to execute with all Pair	Defects found without All Pair	Defects found with All Pair
<b>1024</b>	<b>52</b>	<b>27</b>	<b>27</b>

**Requirements Analysis/Understanding**

**Incomplete Requirements**

**Intermediate Changes**

**Test Planning & Strategy**

**Anticipating Risks**

**Deviation from Plan**

**Quality level expectation from each cycle**

**Number of Test Cycles**

**Test Design**

**Test Case Coverage**

**Productive & Non-Productive Test Cases**

**Test Execution**

**Test Execution Time**

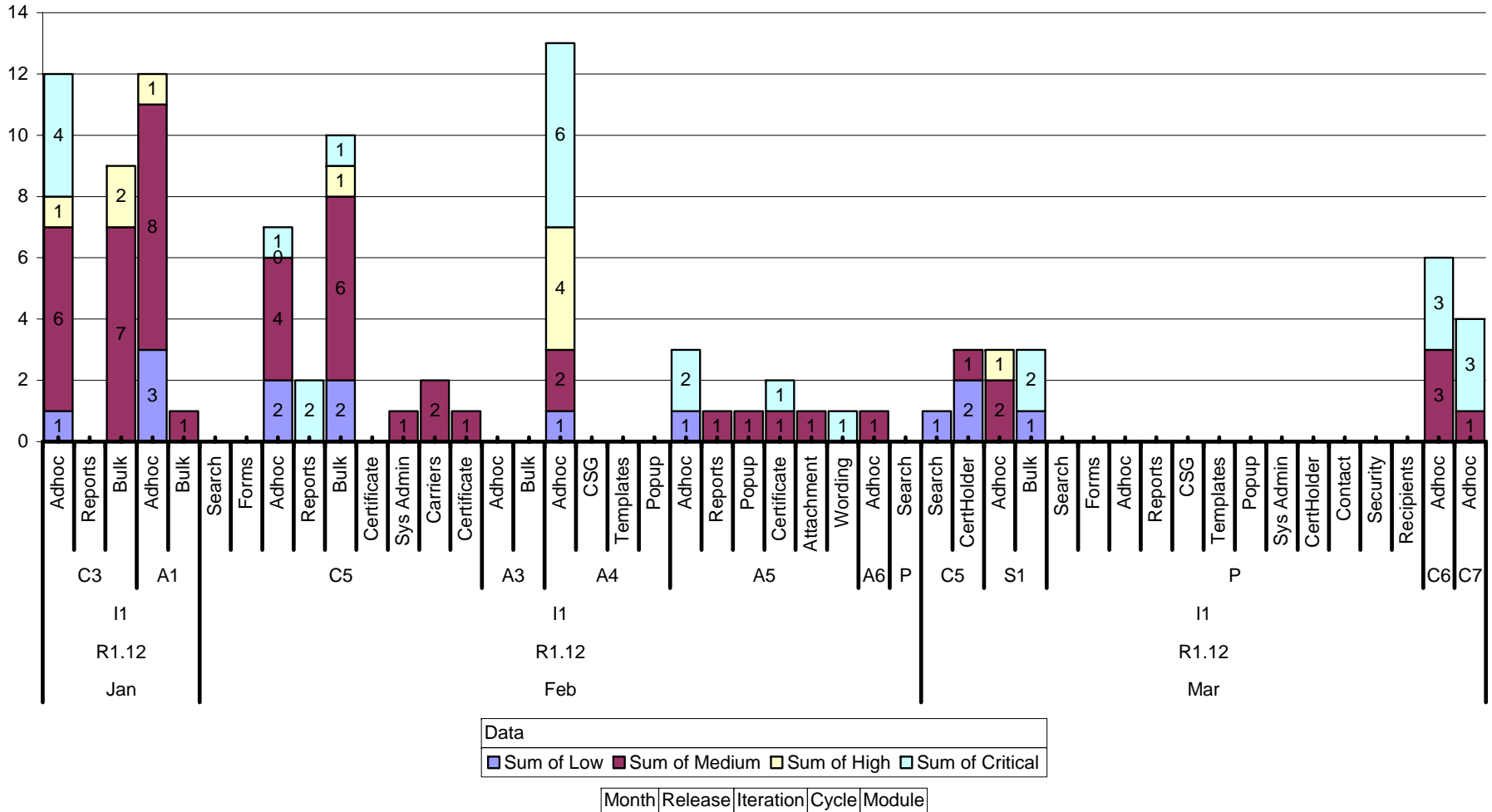
**Prioritization of Test Cases**

**Test Reporting**

**Reporting Effectiveness**

- Different Types of reports:
  1. Release vs Severity Chart (Combination of Month, Release, Iteration, Cycle & Module)
  2. Module vs Severity Chart (Combination of Month, Release, Iteration, Cycle & Module)
  3. Test Case Creation Chart – Shows productivity
  4. Test Case Execution Chart – Shows productivity
  5. Bug Rejection Rate / Ratio chart
  6. Bugs Reported Per Build
  7. Size & Complexity based reports

## Release VS Severity Chart



Data				
■	■	■	■	
Sum of Low	Sum of Medium	Sum of High	Sum of Critical	
Month	Release	Iteration	Cycle	Module

Cumulative History of defects:

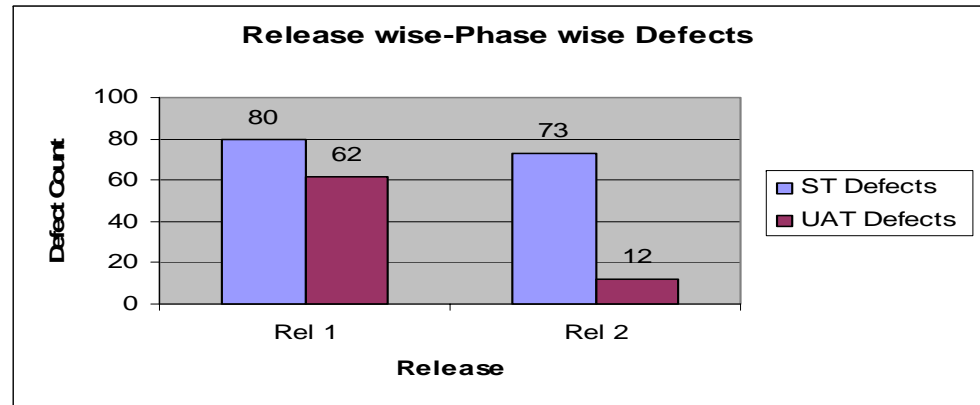
Bug Ref.	Cycle 1	Cycle 2	Cycle3
Bug 1	Y	Y	N
Bug 2	-	-	Y
Bug 3	Y	N	Y

## Phase-wise System Testing Versus User Acceptance Testing Defects

Defects found in System Testing phase					
Phase Introduced	Critical	Major	Average	Minor	Total
Requirements	2	4	8	10	24
Design	3	4	8	5	20
Coding/Unit Testing	5	7	8	9	29
<b>Total</b>	<b>10</b>	<b>15</b>	<b>24</b>	<b>24</b>	<b>73</b>

Defects found in UAT phase					
Phase Introduced	Critical	Major	Average	Minor	Total
Requirements	1	0	0	2	3
Design	1	0	1	1	3
Coding/Unit Testing	0	1	0	5	6
<b>Total</b>	<b>2</b>	<b>1</b>	<b>1</b>	<b>8</b>	<b>12</b>

	Rel. 1	Rel. 2
ST Defects	80	73
UAT Defects	62	12



$$\text{System Test Effectiveness} = \frac{\text{Defects found in ST}}{\text{Defects found in ST} + \text{Defects found in UAT}} * 100$$

$$\text{System Test Effectiveness for Release 2} = \frac{73}{73+12} * 100 = 85.88 \%$$

Observation - Test Effectiveness is 85.88% as compared to earlier 56.35 %

Measures of Test Effectiveness is not mere collating metrics but to effectively identify the metrics for each and every problem areas in every phase of testing, analyzing the metrics and continuously improving them.

#### Visible Benefits

- Organization is more oriented towards customers expectations
- Excellence in Process and Operation with focus on continuous improvement
- Through retrospection one can understand What-could-improve
- Repeat Business through operational excellence
- Less Risks in running defined and predictable process
- Appropriate corrective and preventive actions can be taken to correct the deficiencies in process

- How to detect Requirement errors – A guide to slashing Software Costs & Shortening Development Time – By Adam Frankl & Tom King (2005)
- White Paper presented in IEEE by Ran Ye and Yashwant K. Malaiya on Relationship Between Test Effectiveness and Coverage

White Papers presented during organization level conferences

- Test Case Design Techniques – By Deepesh Belani, Anita Kahate, Rachana Gandhi & Damandeep Kaur (2007)
- Functional Testing – Finding the Right Defects – By Gaurav Oberoi (2007)

Thank You