|  |  |  |
| --- | --- | --- |
| REVISION HISTORY | | |
| Description of Change | Author | Effective Date |
| Initial Draft | Michael Fetick, **619-750-7317** | March 4, 2014 |
| First release | Michael Fetick, **619-750-7317** | March 9, 2014 |

Metrics

Client(s)

The client of this metric report is named the **Comenity Bank (CB)**. Various departments at the corporate office of Comenity Bank have uniquely, vested interests in this report. This metric report presents the results and analysis of testing the proposed, **Automated Teller Machine (ATM)**’s Control Program System. The clients Chief Information Officer (CIO), the product development team, marketing team, logistics-support team, personnel-training team, and members of the board have a role in the deployment of this system.

Metrics Analyst Responsible for Producing Metric

This metric report, constructed from all the work of data extraction/data analysis, performing tests, and developing the test plan, was produced by students at Coleman University titled, the **Coleman University Software Testers (CUST)**.

This metric report was constructed from data extraction/data analysis resulted from testing by the test plan. All this work was produced by students at Coleman University, titled as the **Coleman University Software Testers (CUST)**. This metric report was constructed from data extraction/data analysis resulted from testing by the test plan. All this work was produced by students at Coleman University, titled as the **Coleman University Software Testers (CUST)**. This metric report was constructed from data extraction/data analysis resulted from testing by the test plan. All this work was produced by students at Coleman University, titled as the **Coleman University Software Testers (CUST)**. This metric report was constructed from data extraction/data analysis resulted from testing by the test plan. All this work was produced by students at Coleman University, titled as the **Coleman University Software Testers (CUST)**. This metric report was constructed from data extraction/data analysis resulted from testing by the test plan. All this work was produced by students at Coleman University, titled as the **Coleman University Software Testers (CUST)**.

Goal / Question / Metric

This metric assists with determining staffing levels. Its analysis provides an indication of the need to increase/reduce the staff of testers and programmers. The metric compares the count of defects found (by testers) and the count of defects removed (fixed, resolved by programmers) and results a difference. Depending on budget and schedule to increase or decrease staffing levels, this metric variance identifies the lesser rate that needs to be increased or the greater rate that needs to be decreased.

Gathering defect metrics at the end of the product lifecycle helps us to understand our processes.  It ensures that our decisions are based on well-documented facts.  It shows where improvements are needed and it reveals problems that longevity covers up.  We analyze the efficiency of the developer(s), the tester(s), and the process of testing, based on the bug details collected from one project.  Measurement improves management, reducing variation, self-assessment and supports continuous improvements.  The following metrics were gathered:

* Defects found and removed during requirements and specification reviews
* Defects found and removed during design reviews and code verification activities
* Defects found and removed during testing

The metric results over the entire product development life cycle is depicted in figures 1 and 2.

This metric report was developed from the IEEE 829 Standard, Software and System Test Documentation (Std. 829-2008).

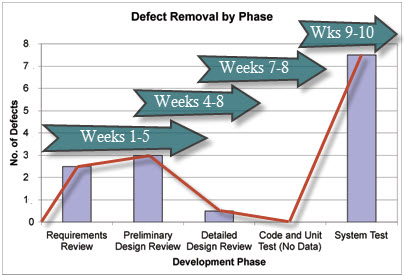


Figure 1. Defects Removal by Phase

Figure 2. Defects Accumulation Breakdown

Any test that results a failure (F) will include a description of the event observed and will try to identify the cause, possibly a software bug. This failed event will be recorded in the Mantis Bug Tracker for evaluation and analysis for management of software development. The defect will be evaluated for severity and priority as described in the textbook on page 299, and as shown below. Resources will be allocated to fixing defects to reduce the highest risks in the most efficient and effective manner.

Table 1. Defects Risk Classification

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| HIGH RISK | | MODERATE RISK | | | LOW RISK | |
| CLASSIFICATION | **Priority – 1** | | **Priority – 2** | **Priority – 3** | | **Priority – 4** |
| **Severity – 1** | 1 – 1 | | 1 – 2 | 1 – 3 | | 1 – 4 |
| **Severity – 2** | 2 – 1 | | 2 – 2 | 2 – 3 | | 2 – 4 |
| **Severity – 3** | 3 – 1 | | 3 – 2 | 3 – 3 | | 3 – 4 |
| **Severity – 4** | 4 – 1 | | 4 – 2 | 4 – 3 | | 4 – 4 |
| Severity – 1 System crash, data loss, data corruption, security breach  Severity – 2 Operational error, wrong result, loss of functionality  Severity – 3 Minor problem, misspelling, UI layout, rare occurrence  Severity – 4 Suggestion | | | | | | |
| Priority – 1 Immediate fix, blocks further testing, very visible  Priority – 2 Must fix before the product is released  Priority – 3 Should fix when time permits  Priority – 4 Would like to fix but the product can be released as is | | | | | | |
| Testers:  MVF – Michael V. Fetick, Student #84270, Coleman University | | | | | | |

Table 2. Defects found during test:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| HIGH RISK | | MODERATE RISK | | | LOW RISK | |
| CLASSIFICATION | **Priority – 1** | | **Priority – 2** | **Priority – 3** | | **Priority – 4** |
| **Severity – 1** | 4 | |  |  | |  |
| **Severity – 2** |  | | 6 | 8 | |  |
| **Severity – 3** |  | |  | 11 | |  |
| **Severity – 4** |  | | 12 | 3,10 | | 1,2,5,7, 9 |

Figure 3. Defects Risk Assessment Breakdown

Defects found during requirements and specification reviews

This metric evaluates the effectiveness of the reviews for development of the requirements and specification. The requirements and specification were developed from sequence diagrams and collaboration diagrams of the following use cases:

System startup

System shutdown

Session

Transaction

Withdrawal

Deposit

Transfer

Inquiry

Invalid PIN

Metric Requirements Statement

This metric requires analysis to evaluate the detection rate over time by severity of defects detected during requirements and specification reviews, and removed (fixed, resolved) in order to deliver a software product with the high standard of quality, requested by the client and reflects our company’s policy of the Users’ Bill of Rights for fitness-of-use.

Standardized Definitions

|  |  |
| --- | --- |
| **Attribute/Entity** | **Definition** |
| Defects found / time  (during requirements and specification reviews) | Rate of finding defects = Defects found / per workday  (during requirements and specification reviews) |
| Defects removed / time  (during requirements and specification reviews) | Rate of removing defects = Defects removed (fixed, resolved)  / per workday (during requirements and specification reviews) |

Measurement Functions & Methods

|  |  |
| --- | --- |
| **Measurement Function(s)** | |
| Not hire more : Development Leader = if (Defects Found / Time < Defects Removed / Time)  then Programming (leads)  else Testing (leads) | |
| Base Measure | Measurement Method |
| Defects Found / Time | Count of Defects Found / Time period of requirements and specification reviews |
| Defects Removed / Time | Count of Defects Removed (fixed, resolved) / Time period of requirements and specification reviews |

Decision Criteria

The metric evaluates the period of the requirements and specification reviewsrequirements and specification reviews requirements and specification reviews requirements and specification reviews. The decision to hire more testers or programmers is decided when the criteria reaches 30% difference between rates measured by this metric. Below is an example:

Step 1)

10 Defects found (by testers) / 40 tester man-hours period = 0.25 Defects/period

> 8 Defects removed (by programmers) / 40 tester man-hours period = 0.20 Defects/period

Step 2)

0.25 Defects found / 0.20 Defects removed = 1.25 = 25% difference

The result of the larger rate (count of defects found (by testers)) may indicate that maybe more programmers are needed. When it reaches the threshold of 30% difference, hire more programmers.

## Report Format

|  |
| --- |
| Example Report Format: The format of this report has been established and accepted by the project’s Program  Development Group. It includes metrics presented here and includes timely, pertinent data that substantiates the computation of the metrics. |
| Data Extraction Cycle: Weekly |
| Reporting Cycle: Weekly |
| Distribution: Distributed to the project’s Program Development Group which consist of members from our company that on the project and members from the client, Comenity Bank. |
| Availability: Hard copy of the report is distributed to members of the project’s Program Development Group after final review from the Project Leads and approval from the Project Manager.  Electronic copy of the report is posted on the company secure Intranet website.  All copies will be identified for proper secure handling with the header stating,  “Company Confidential for Internal Use Only.” |

Additional Qualifiers

This metric compares the count of defects that were found, to the count that were removed (fixed, resolved), to the count that were not removed (fixed, resolved); between phases of product development.

The following additional qualifiers are recorded and factored in for a basis of comparison.

* It accounts for the classification of defects by risk level, comprising of a severity level and a priority level.
* It accounts for the work load at the time, and the afforded to the fix of the defect.
* If there is following on work to produce a new or better release/revision of the product, then steps will be taken to maintain configuration management of the products’ development.

Data Collection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data Item | Database | Record | Data  Elements / Fields | Who Collects the Data | Data Collection Rules & Procedures  (Immediate, accurate, concise) |
| Count of Defects Found | Mantis | Defects | int cntDefFd | Lead Tstrs | Accurate, automatic |
| Count of Defects Removed | Mantis | Defects | int cntDefRem | Lead Progs | Accurate, automatic |
| Difference rule | Project | Staff | int prctDiffRule | Manager | Accurate, Immediate |
| Date of Defect Found | Mantis | Defects | int cntDefFd | Lead Tstrs | Immediate |
| Date of Defect Removed | Mantis | Defects | int cntDefRem | Lead Progs | Immediate |
| Development Phases | Project | Phases | int devPase | Manager | Accurate, automatic |
| Work week number | Project | Phases | int workWeek | Leads, Mgr | Accurate, automatic |
| Risk level identifiers | Mantis | Risks | int riskLevels | Leads | Accurate |
| Risk level rules | Mantis | Risks | int riskRules | Leads | Accurate |
| Severity level identifiers | Mantis | Risks | int sevLevels | Tstrs, Progs | Accurate |
| Severity level rules | Mantis | Risks | int sevRules | Tstrs, Progs | Accurate |
| Priority level identifiers | Mantis | Risks | int priLevels | Tstrs, Progs | Accurate |
| Priority level rules | Mantis | Risks | int priRules | Tstrs, Progs | Accurate |
| Risk level grade | Mantis | Defects | int riskGrade | Tstrs, Progs | Accurate, Concise |
| Severity level grade | Mantis | Defects | int sevGrade | Tstrs, Progs | Accurate, Concise |
| Priority level grade | Mantis | Defects | int priGrade | Tstrs, Progs | Accurate, Concise |
| Count of Programmer Staff | Project | Staff | int cntProgs | Manager | Accurate, automatic |
| Levels – Programmer Staff | Project | Staff | int lvlProgs | Manager | Accurate |
| Count of Testing Staff | Project | Staff | int cntTsrs | Manager | Accurate, automatic |
| Levels – Testing Staff | Project | Staff | int lvlTstrs | Manager | Accurate |
| Release/Revision identifier | Project | Phases | int relId | Manager | Accurate, automatic |
| Defect Tracking Number | Project | Defects | int defTrkNo | Tstrs, Progs | Accurate, automatic |
| Query Request Number | Project | Staff | int qryReqNo | Leads | Concise |
| Report Distribution Number | Project | Staff | int repDistroNo | Leads | Immediate |

Human Factors

Desired Positive Behaviors:

* Punctual, trustworthy, and responsible with deliverables and deadlines
* Follows instructions with a good understanding of best practices learned
* Contributes to the team effort by sharing responsibilities, sharing ideas, looking out for each other
* Takes an extra effort to gain product knowledge and develop leadership style
* Has determination to solve problems and evaluate circumstances for best course of actions
* Pleasant to work with without undesirable habits or behavior

Undesired Negative Behaviors:

* Not desired positive behaviors (see above)

Ways to maximize positive & minimize negative behaviors:

|  |  |  |
| --- | --- | --- |
| Action | Assignment | Due Date |
| Each team has weekly meetings to monitor progress and collaborate on issues. | Manager, Programmers, Testers | Mondays, Weekly |
| The project’s Program Development Group (weekly) meetings sustain the rhythm of work progress, disseminates news and information pertinent to the project, keeps everyone connected and focused. | Manager, Programmer Lead,  Test Lead, Client, other participants | Tuesdays, Weekly |
| Training on client security procedures | Admin, Programmers, Testers | 1st Week, Quarterly |
| Every employee has a performance track record to improve individual knowledge and skillset. | Manager, Programmers, Testers | 1st Week, Quarterly |
| Every employee has periodic reviews of individual performance and accomplishments. | Manager, Programmers, Testers | 1st Week, Quarterly |
| Training on interoperability procedures with other test participants (off-site locations) | Manager, Test Lead, Client, Other participants | 2nd Week, Quarterly |
| Training on data extraction, collection, and analysis. | Manager, Programmer Lead,  Test Lead | 2nd Week, Annually |
| Training on defect reporting and tracking. | Manager, Programmer Lead,  Test Lead | 2nd Week, Annually |
| Training on general documentation requirements | Manager, Programmers, Testers | 3rd and 4th Week, Annually |
| Training on company configuration management, software check-in/check-out procedures | Manager, Programmers, Testers | 3rd and 4th Week, Annually |
| Training on general design requirements | Manager, Programmers | 3rd Week, Annually |
| Training on software development tools, i.e. IDEs | Programmer Lead | 3rd Week, Annually |
| Training on general test requirements | Manager, Testers | 4th Week, Annually |
| Training on system under test (SUT) equipment controls | Test Lead | 4th Week, Annually |

Defects found during design reviews and code verification activities

This metric evaluates the effectiveness of the reviews for design and code verification. The design components were developed from the requirements and specifications of following use cases:

System startup

System shutdown

Session

Transaction

Withdrawal

Deposit

Transfer

Inquiry

Invalid PIN

Metric Requirements Statement

To evaluate the detection rate over time by severity of defects detected during design reviews and code verification, and removed (fixed, resolved) in order to deliver a software product with the high standard of quality, requested by the client and reflects our company’s policy of the Users’ Bill of Rights.

Standardized Definitions

|  |  |
| --- | --- |
| **Attribute/Entity** | **Definition** |
| Defects found / time  (during design reviews and code verification) | Rate of finding defects = Defects found  / per workday (during design reviews and code verification) |
| Defects removed / time  (during design reviews and code verification) | Rate of removing defects = Defects removed (fixed, resolved)  / per workday (during design reviews and code verification) |

Measurement Functions & Methods

|  |  |
| --- | --- |
| **Measurement Function(s)** | |
| Not hire more : Development Leader = if (Defects Found / Time < Defects Removed / Time)  then Programming (leads), else Testing (leads) | |
| Base Measure | Measurement Method |
| Defects Found / Time | Count of Defects Found / Time period of reviews and code verification |
| Defects Removed / Time | Count of Defects Removed (fixed, resolved) / Time period of reviews and verification |

Decision Criteria

Refer to paragraph 1.4 Decision Criteria for the same information but pertinent to this development phase.

## Report Format

Refer to paragraph 1.5 Report Format for the same information but pertinent to this development phase.

Additional Qualifiers

Refer to paragraph 1.6 Additional Qualifiers for the same information but pertinent to this development phase.

Data Collection

Refer to paragraph 1.7 Data Collection for the same information but pertinent to this development phase.

Human Factors

Refer to paragraph 1.6 Human Factors for the same information but pertinent to this development phase.

Defects found during testing

This metric evaluates the effectiveness of the testing effort. The testing was conducted under the guidance of the test plan which contains test procedures organized for unit testing of the design components. Testing proceeded per schedule in the document “COURSE SYLLABUS COM239: Software Testing.” The following test procedures were conducted to run test cases that validate the software program.

Project 1. ATM Test Case (System startup, shutdown)

Project 2. ATM Test Case (Sessions)  
Project 3. ATM Test Case (Transaction, Withdrawal)

Project 4. ATM Test Case (Deposit, Transfer)

Project 5. ATM Test Case (Inquiry, Invalid PIN)

Metric Requirements Statement

To evaluate the detection rate over time by severity of defects detected during integration and system test, and removed (fixed, resolved) in order to deliver a software product with the high standard of quality, requested by the client and reflects our company’s policy of the Users’ Bill of Rights.

Standardized Definitions

|  |  |
| --- | --- |
| **Attribute/Entity** | **Definition** |
| Defects found / time  (during testing) | Rate of finding defects = Defects found  / per workday (during integration and system tests) |
| Defects removed / time (during testing) | Rate of removing defects = Defects removed (fixed, resolved)  / per workday (during integration and system tests) |

Measurement Functions & Methods

|  |  |
| --- | --- |
| **Measurement Function(s)** | |
| Not hire more : Development Leader = if (Defects Found / Time < Defects Removed / Time)  then Programming (leads), else Testing (leads) | |
| Base Measure | Measurement Method |
| Defects Found / Time | Count of Defects Found / Time period of integration and system tests |
| Defects Removed / Time | Count of Defects Removed (fixed, resolved) / Time period of tests |

Decision Criteria

Refer to paragraph 1.4 Decision Criteria for the same information but pertinent to this development phase.

## Report Format

Refer to paragraph 1.5 Report Format for the same information but pertinent to this development phase.

Additional Qualifiers

Refer to paragraph 1.6 Additional Qualifiers for the same information but pertinent to this development phase.

Data Collection

Refer to paragraph 1.7 Data Collection for the same information but pertinent to this development phase.

Human Factors

Refer to paragraph 1.6 Human Factors for the same information but pertinent to this development phase.