Defect Analysis Report CSE6329 – Software Measurement and Quality Engineering

**KARTHIK SOMANAHALLI MURALIDHARA-1001520125**

**SUBRAHMANYA BASAVAPATNA NAGARAJA RAO-1001540208**

**Table of contents**

1. Introduction / Description of Analysis Tool

2.Measures, Graphs and Analysis

2.1. POST RELEASE QUALITY (6 Measures)

2.1a. Post-release Quality for product ZD

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph.

2.1b Post-release Quality Average for all products, normalized by size

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.1c. Post-release Quality Average for all products, by development process

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.1d. Post-release Quality Average for all products, by programming language

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.1e. Post-release Quality History

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.1f. Graph of Your Choice (does not have to be a line chart)

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.2 CURRENT QUALITY (3 MEASURES)

2.2a. Current Quality Total

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.2b. Current Quality Total, normalized by ***size*** of currently active product releases (defects per LOC)

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

4 -> Description of how to generate the Graph

2.2c. Current Quality Total, normalized by ***number*** of currently active products (defects per active product)

1-> An Overview of the Measure.

2-> Sample Graphs.

3 -> Analysis

1 -> General Analysis

2 -> Specific Analysis

1. -> Description of how to generate the Graph

INTRODUCTION

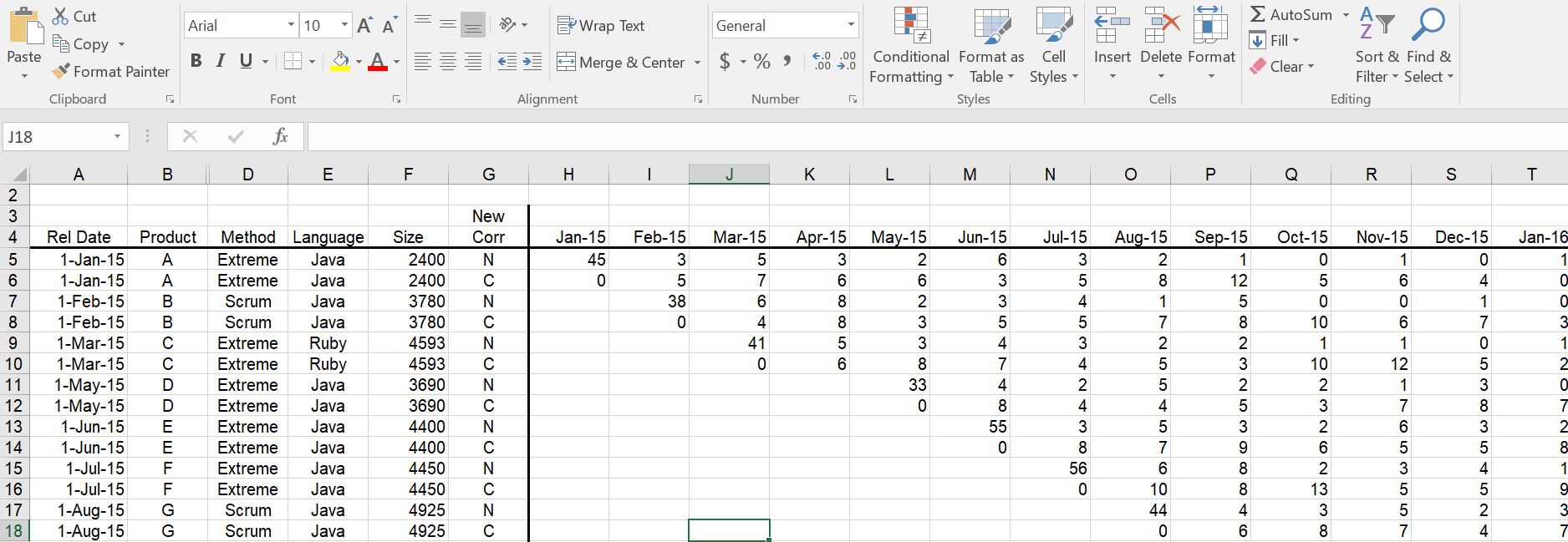
Purpose Of this Report:

* This report is intended to help us **understand which product of ours, developed using which language (java or ruby) , using which process (scrum or XP) produces better results or higher quality with respect to customers’ point of view.**
* The main intention is to measure the quality of the product released to the customer and to know what “variable measures” are being used to produce that aforementioned high-quality product.
* The measurement is calculated using raw data , that is hence refined , and then a graph is produced to let us know which product is of what quality .
* We have chosen a series of 8 measures pertaining to defects that is being found with respect to the product. 1 more measure is of our own which again calculates the product quality using defect intensity as a scale.
* We have also developed an analysis and data collection tool consisting of a Microsoft Excel® workbook that contains NINE worksheets.
* **Data are collected monthly** and entered into this tool. We have collected three years’ worth of data.
* The tool is used to analyze the data and generate graphs.
* This report explains the tool, each of the 9 measures and corresponding graphs
* It also discusses the **results of our analysis**.

STRUCTURE OF ANALYSIS TOOL

* The analysis tool is an Excel® workbook consisting of TEN worksheets:
  + The first sheet is the data collection sheet.
  + The other EIGHT sheets including the first one correspond to the nine measures and are where the data for each measure is calculated and graphed.
* On the data collection sheet, there are six columns for each of our products, product release dates , method used to develop , Language used for development , Size of the code in LOCs (lines of code) , New/Corrected status which keeps track of the status of bugs and a column for each month. Each month, a new column is added and data collected for that month are recorded there.
* Graphs are plotted on the respective sheets using the data from the 1st sheet and the analysis is done in the report here.

Picture of Typical Data Collection Sheet



Only status of the company till 1st August 2015 is shown in the above figure. We still have data worth of 3 years.

**Data Collection Overview**

* **Data are collected monthly by the quality engineering organization and recorded in the spreadsheet.**
* **Most of the data collected keeps track of the defects raised for a particular product and what process that product is built with and the language used to code. We also have been provided with the release date of the product.**
* **We also have the size of the code of the product under test.**
* **Specific data collected are:**
* Product name
* Release date of the product
* Process used to build the product
* Programming language used to build
* Size of the code(LOCs) of the product
* Category of the defect (New or corrected)

Defects raised for every product released to the customer.

**Summary of the Analyses and Graphs:**

We have analyzed the data in 2 different ways:

1. Post Release quality: The post release quality answers the "how many known defects are in this product and how does the total grow after release?" It is measured for an individual product or for any collection of products (such as all those written in Java) and is defined as the number of defects in the product or collection each month after release. This is recorded each month for twelve months after a product is released and is displayed using two lines: ***total defects*** and ***total uncorrected defects***. A value for each line is calculated each month for the 12-month period after product release.

We have 6 graphs measuring different aspects of the quality of a product and is defined as follows.

1. **Post-release Quality for Product ZD**
2. **Post-release Quality average for all Products , normalized by size.**
3. **Post-release Quality average for all Products , by development process.**
4. **Post-release Quality average for all Products , by programming language.**
5. **Post release quality History.**
6. **Graph of our choice.**
7. Current Quality: Current quality is designed to answer the question "how many known defects are out there this month in all currently-supported products?" Current quality gives you a sense of how your customers see your products. If there are a lot of defects in your products, customers may not like. This may be helpful for deciding whether you need to assign more resources to defect correction.

We have 3 graphs measuring the quality with respect to this aspect and they are as follows.

* **Current Quality Total**
* **Current quality total normalized by size**
* **Current quality total normalized by number of products**.

2.Measures, Graphs and Analysis

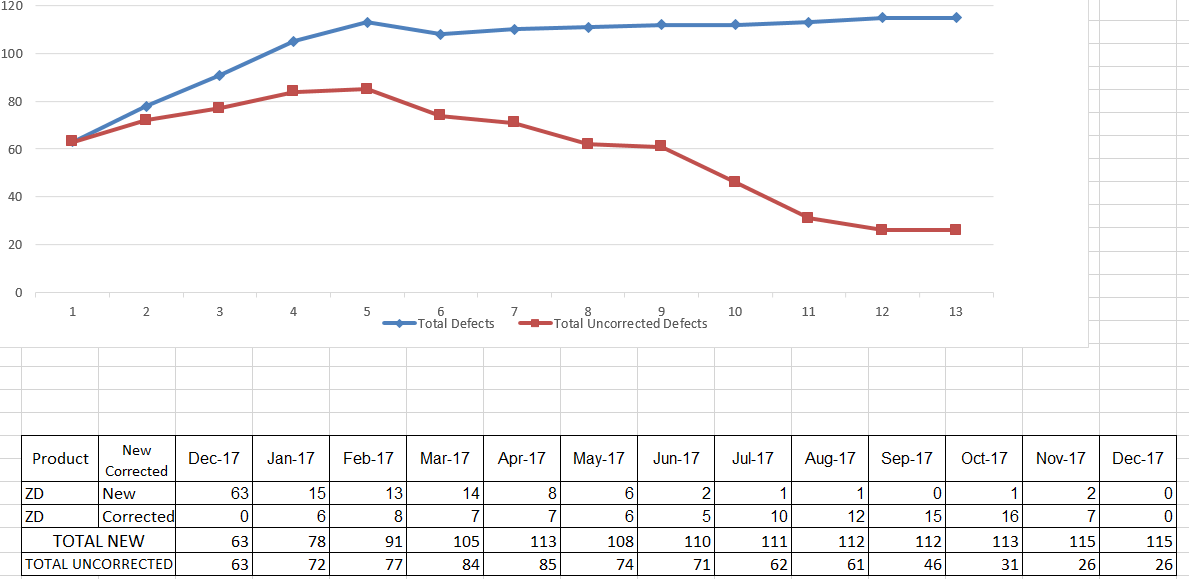
2.1.a **Post-release Quality for Product ZD:**

1.OVERVIEW-

* **Purpose**: This measure is used to determine the quality of one of our products, called product ZD , when released to the customer.
* **Question**: The question being answered is , “is the product performing well from the customer’s point of view, ie , is the customer satisfied with the quality of the product.

* **Definition**: Post release quality is defined as the total known defects of the product after its release to the customer, through its lifetime. This is the total number of defects with respect to the product.
* **Collection Frequency**: Data is collected monthly from the day of its release to the next one year. Data here is the amount of defects encountered with the product.

2. Sample Graphs:



1. Analysis – General Description:

* Purpose – The post release quality of product ZD helps us understand how our product is performing from the customers’ perspective.
* This is important because it gives us an idea of how good our most recent product is performing and analyzing which, gives us further idea about our employees’ and organization’s stance.
* A too high graphical curve means that the product is failing to do its designated job and failing miserably.
* Graph: The post release quality for product ZD is plotted by month on the X-axis and the total number of defects on the Y-axis.
* It shows the number of defects encountered by the product each month.
* 2 lines are shown; Blue, The total defects encountered and Red, the uncorrected number of defects.
* When the red line is higher than the blue line , it tells us that the product is performing worse as it progresses and our employees are not able to maintain the product .
* When the blue line is higher than the red line , it means that the product is ideal , but any line too high is a concern and demands to be looked into right away.
* How to analyze and respond to the graph:
* It is normal to have ups and downs in the curve
* Its normal to have bugs raised for a product periodically and those bugs are corrected respectively and hence it’s a cycle which shows normal behavior of the product.
* It signifies that the product and hence our employees are working fine.
* This is the ideal scenario for a product.
* The blue curve or the red curve is always preferred to stay as low as possible
* This is so because we want the number of uncorrected defects denoted by the red line to be as low as possible **since it’s a direct implication about the quality of the product**.
* This also means to say that our employees are dealing with defects and correcting them as and when they are occurring.
* A sudden burst of raise in either of the curves/lines is an immediate concern which should never happen.
* This signifies a major failure that has happened with the product all of a sudden , which wasn’t planned for.
* Scenarios such as this should almost never happen.
* It tells us that the product has failed miserably during that time.
* A line/curve periodically decreasing is a good sign whereas the other way round , where in its increasing , is of concern .
* A line gradually decreasing denotes that the quality of the product is improving with time which is always preferred .
* A line gradually increasing tells us that the quality is gradually decreasing and is a sign of concern.

**What the graph shows (Specific Description):**

* This specific graph shows that although the quality of the product was not that great at the time of release , it has gradually improved with time.
* It also denotes that our employees have resolved almost all of the defects that was/is encountered and this shows that the present quality of the product is much more improved.
* The product was launched in the month of December 2017 , and from then through middle of May 2018 , the quality was not that great .
* From May 2018 through December 2018 the quality has been increasing phenomenally.
* This shows that our employees needed time to analyze the defects with respect to that product.
* **The quality of the product is analyzed by looking at the red line (number of uncorrected defects) comparing with the blue line concurrently.**
* This graph also tells us that since June , the 6th month as shown in the graph , the total defects (blue line) almost didn’t grow up at all .
* This above behavior tells us that product has started to perform better than the previous months and is now stable.

**Specific Recommendations:**

* We should ensure that the testing team should do a much better job in order to ensure the “total defects” to come down to at least 50% of the present value.
* Since the “total defects” during the time of release is more, better Requirements Engineering on the product can be done in order to bring it down.

**4.Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The language used (Java or Ruby) to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

DATA REFINEMENT:

The uncorrected defect is calculated by subtracting C from N. This is compared with the total defects N and plotted on the graph for each month.

COMPOUND MEASURES:

Uncorrected defects is calculated using the formula “N-C”.

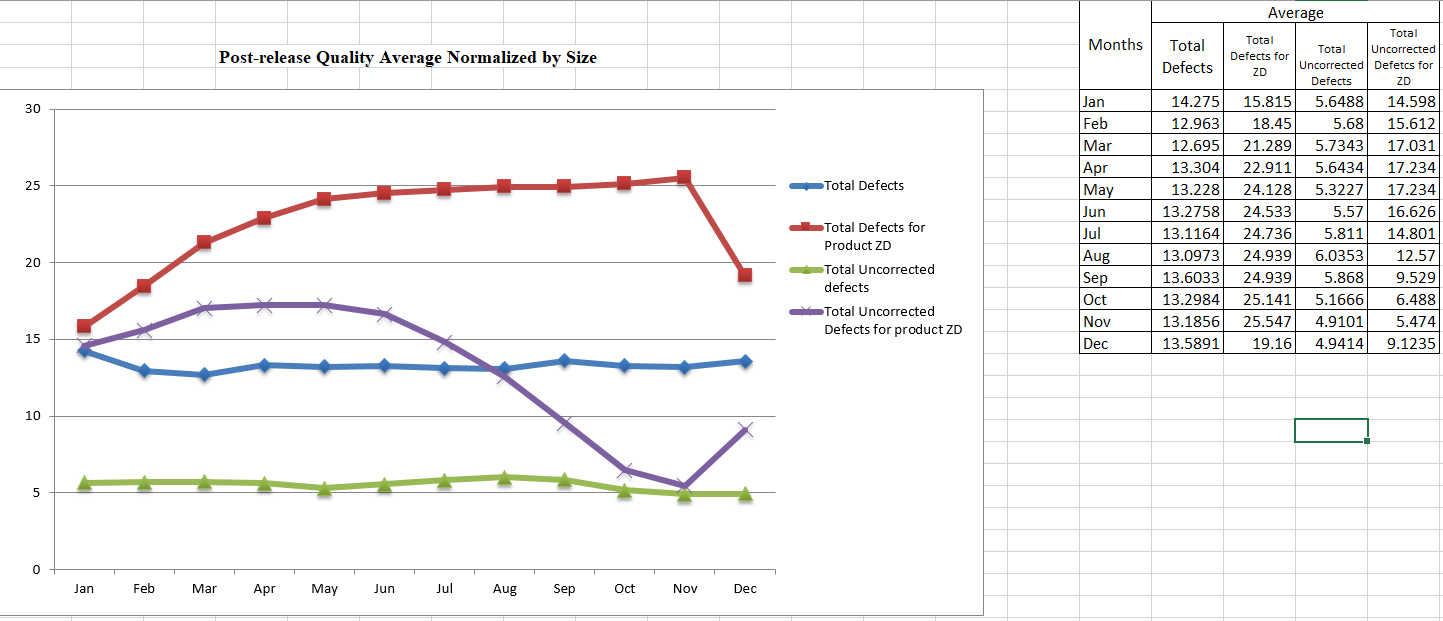
GENERATING THE GRAPH:

* The total defect is recorded using the data from column which is denoted as N. This is added for all the months for the product ZD and is denoted by BLUE line on the graph.
* The uncorrected defects is calculated using the formula N-C where C is the total of corrected results and N is the total of “total defects” tab. And is represented by Red Line.
* The horizontal axis represents the months and the vertical axis represents the “total defects”.
* This generates the graph.

**2.1.b : Post release quality Average normalized by size**

**Overview:**

* **Purpose:** The purpose of this graph is to compare the product ZD with the average of all the other products which are normalized by size.
* **Questions:** The questions being answered is , “**HOW** our most recently released product ZD compares with the average of all the previously released products? “ Is it doing any better in the market or worse.
* **Definition:** Post release quality average normalized by size is defined as the total average of defects of all the released products by month that is normalized by size compared to that of the product ZD.
* **Collection Frequency:** Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

**Sample Graph:** 

**Analysis: General Description:**

* Purpose: This graph is about the quality of product ZD compared to the previously released products.
* Since product ZD is our most recent product, the comparison between this product and the rest of them helps us understand how the present set of employees are performing when developing a product.
* This also helps us understand what are the customer needs now, in the present market and are we up to the needs of the customers.
* This also helps us understand how our team , as an organization , performed over the last few years in aspect of developing a product.
* **Graph:**
* In this graph we plot 4 lines. Red, Violet, Blue and Green.
* The horizontal axis shows the month scale and vertical axis shows “defects per 1000 lines of code” .
* It shows the comparison between the product ZD with the average of rest of the products post release by taking into account the number of defects raised per 1000 lines of code.
* When the red line is higher than the rest , it means that product ZD is worse off than the rest of products and its better if vice versa.
* When the green line is higher than the violet line , it means that the product ZD is difficult to maintain than the rest of them since the number of uncorrected defects for ZD is more.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph shows the average of total defects of all the other products compared to the product ZD.
* As a result we can see that the total defects for product ZD is much more than the total defects for the rest of the products which means to say that , quality wise , product ZD is worse off than the rest of the products.
* We can also see that, total defects immediately after the release of product ZD is also higher than the average which denotes that the initial quality of ZD was not that great.
* The graph also tells the total uncorrected defects of product ZD with respect to the average of the rest of the products.
* As we carefully examine the graph , we can see that the product ZD is difficult to maintain and our developers have a hard time fixing the defects for most part of the
* We can also conclude that product ZD has higher number uncorrected defects for most part of the year which can tell us the quality is not good compared to the rest of the products.
* We can still conclude that the total uncorrected defects for product ZD (direct implication of the quality) has been gradually improving and that is a positive development.

**Specific Recommendations:**

* A thorough analysis of SPECS need to be done the next time we are developing a product since the amount of initial defects is much more than the average.
* Proper training of QA/testing team needs to be done to ensure that the product defects are found well within the release so that the developers can fix the defects.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Defects per 1000 lines of code normalized by size: This is used to normalize the calculations since, when taking the average for all the products, each product has a different size. Hence there might be a possibility that more defects might be encountered for a product with higher size. To eliminate this drawback, we normalize the defects to 1000 lines of code and then calculate the average.

This number (uncorrected defects per 1000 lines of code) is calculated using the formula (N-C)/SIZE \* 1000 for each product with its respective size.

To calculate the number of total defects per 1000 lines of code , we use the formula N/SIZE \* 1000.

The average is then calculated for each month .

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The language used (Java or Ruby) to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

Uncorrected Defects per 1000 lines of code normalized by size is calculated using the formula (N-C)/SIZE\* 1000.

Total defects per 1000 lines of code is N/Size \* 1000.

The average is taken for every month and plotted against the same on the graph.

The month is denoted on x-axis.

COMPOUND MEASURES:

Uncorrected Defects per 1000 lines of code is calculated using the formula (N-C)/SIZE\* 1000

**GENERATING THE GRAPH:**

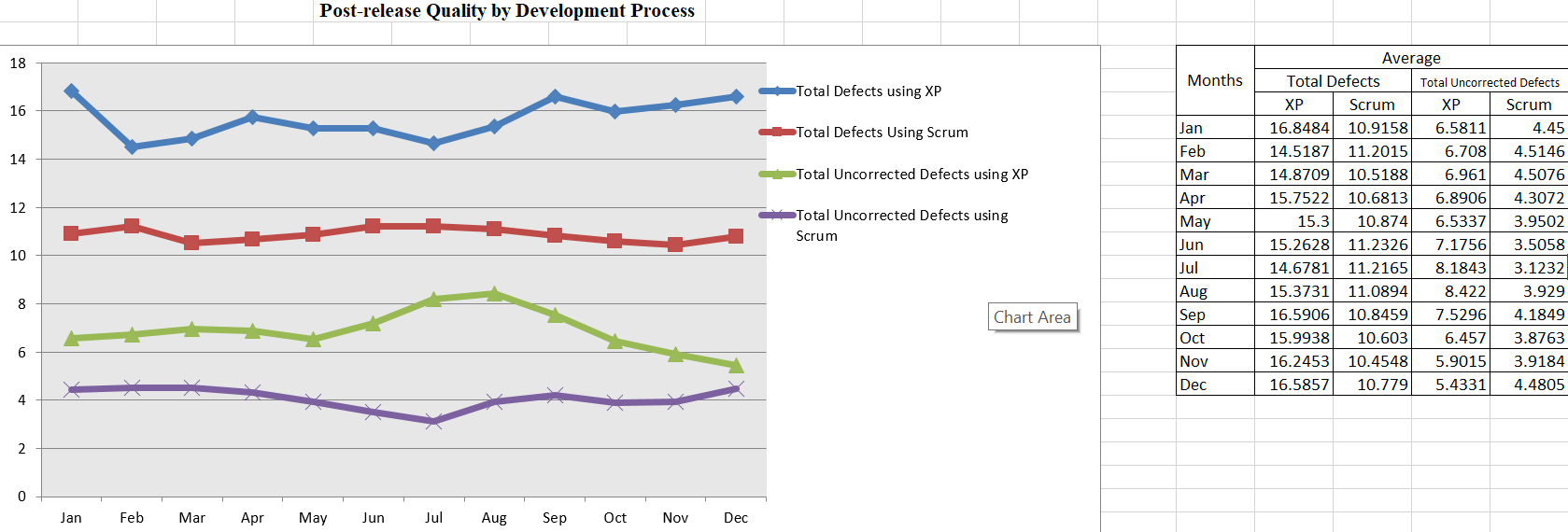
* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* The red line is the total defects for product ZD per 1000 lines of code. This is calculated using N/Size \* 1000.
* The blue line is the average total defects per 1000 lines of code for all the products.
* The violet line the number of uncorrected defects per 1000 lines of code for product ZD.
* The green line the average number of uncorrected defects per 1000 lines of code for all the products.
* The horizontal axis shows the months.
* This generates the graph.

**2.1.c : Post release quality by development process**

**Overview:**

* Purpose: The purpose of the graph is to compare which process is more effective when developing a product.
* Question: The question being answered is, when is the quality of a certain product is more when one kind of a development process is used. So in short, we can know if a particular development process has an impact on the quality of the product.
* Definition: Post release quality by development process is defined by the number of defects encountered for a product after its release for a particular process.
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

SAMPLE GRAPH:



ANALYSIS: GENERAL DESCRIPTION:

* PURPOSE: The purpose of this graph is to study and analyze if a particular development process has an effect on the quality of the product.
* To know if a development process is good enough to follow.
* This graph also tells us which products are suited for what development process.
* It also gives us a slight clue about when to use what development process.
* This graph tells us that if the development process we are using is accepted in the present market or is it rejected for having ineffective results.

GRAPH:

* In this graph, we plot 4 lines , total defects for scrum , total defects for XP , total uncorrected defects for scrum and total uncorrected defects for XP.
* It shows the defects per 1000 lines of code for a product implemented using one of the above two processes.
* A curve/line too high means that the process is ineffective and not suitable for the product to be built with.
* Months is denoted on the horizontal axis.
* Number of defects per 1000 lines of code is denoted on the vertical axis.
* This graph is basically used to compare between the two processes.
* When the violet line is lower than the green line , it means that scrum is more effective and if the other way round then its vice versa respectively.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph denotes the average of total defects per month for all the products per 1000 lines of code normalized by size for the two processes namely Scrum and XP.
* The Scrum and XP each have 2 lines namely uncorrected defects and total defects.
* Total defects are the known defects at the time of release summed up with every possible defect encountered with respect to the product each month.
* Higher the line , higher is the negativity.
* We expect to see those lines lower or tending lower.
* The number of months is denoted on the horizontal axis.
* A sudden burst of raise in any curve of the graph means that there has been a serious defect in the process followed.
* Often , it has been seen that quality of a product is directly proportional to the process followed to develop it.
* We DONOT expect to see sudden changes in the curves in this graph.
* When the violet line is lower than the green line , it means that scrum is more effective and if the other way round then its vice versa respectively.
* From the graph, it can be noted that products built using scrum are much more consistently effective and have a better quality than those products built using XP.
* Total defects initially for products built using XP is much higher than products built using scrum. This shows that products built using scrum tend to have a higher quality.
* A consistently lower bound line of the graph tells us that the process followed is effective in building the product.

Recommendations / Comments:

* If we see a curve consistently higher bound , it’s a concern for the company to look into the process implementation .
* If the above case happens repeatedly, then the higher management should come up with a process more suited and upscale to build the respective product.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Defects per 1000 lines of code normalized by size: This is used to normalize the calculations since, when taking the average for all the products, each product has a different size. Hence there might be a possibility that more defects might be encountered for a product with higher size. To eliminate this drawback, we normalize the defects to 1000 lines of code and then calculate the average.

This number (uncorrected defects per 1000 lines of code) is calculated using the formula (N-C)/SIZE \* 1000 for each product with its respective size. (Separately for each process)

To calculate the number of total defects per 1000 lines of code , we use the formula N/SIZE \* 1000.

The average is then calculated for each month with respect to each process separately.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

Uncorrected Defects per 1000 lines of code normalized by size is calculated using the formula (N-C)/SIZE\* 1000.

Total defects per 1000 lines of code is N/Size \* 1000.

The average is taken for every month and plotted against the same on the graph.

The month is denoted on x-axis.

COMPOUND MEASURES:

Uncorrected Defects per 1000 lines of code is calculated using the formula (N-C)/SIZE\* 1000

All these measures are calculated for each month for the respective processes and each process separately and then the total is done and the average is taken for each month.

**GENERATING THE GRAPH:**

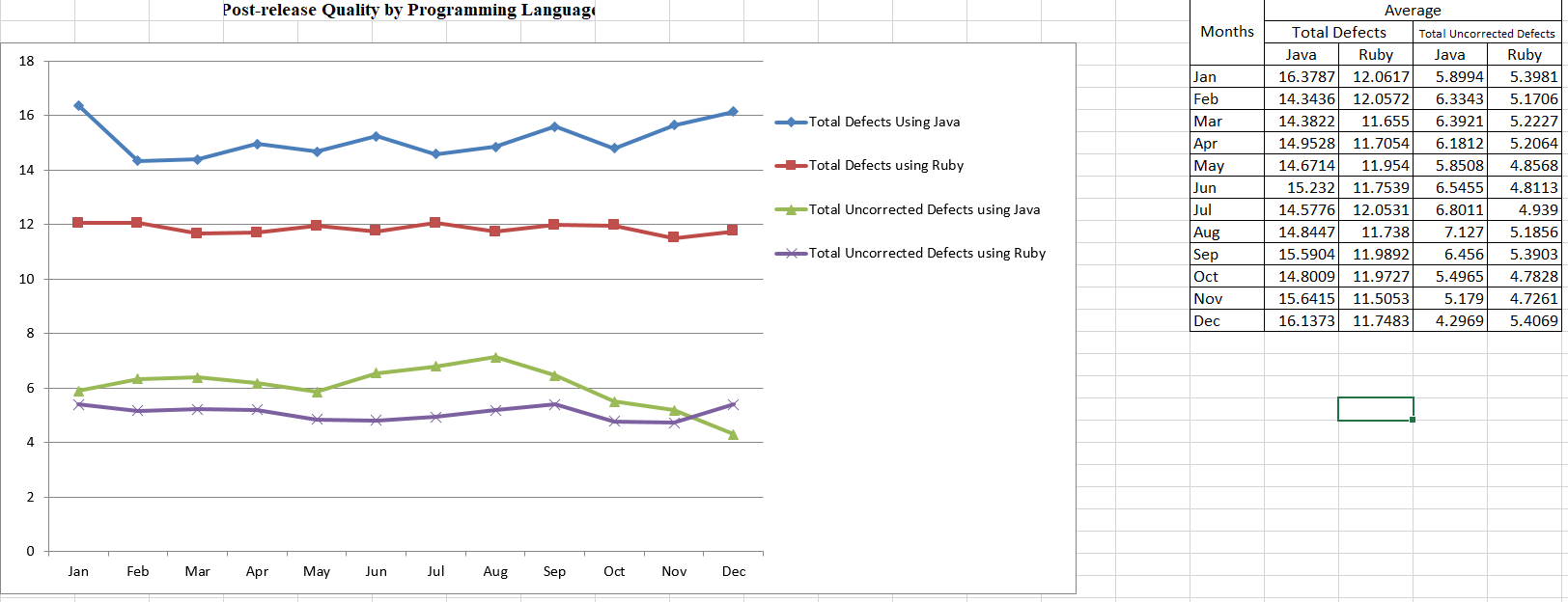
* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* The value got form computing (N-C/Size) \* 1000 for all the products individually using scrum and summing them up for each month and taking the average gives us the Violet line (total uncorrected defects with scrum)
* The value got form computing (N-C/Size) \* 1000 for all the products individually using XP and summing them up for each month and taking the average gives us the Green line (total uncorrected defects with XP)
* The value got form computing (N/Size) \* 1000 for all the products individually using scrum and summing them up for each month and taking the average gives us the Red line (total defects with scrum)
* The value got form computing (N/Size) \* 1000 for all the products individually using scrum and summing them up for each month and taking the average gives us the Blue line (total defects with XP)
* The horizontal axis is the months for which the graph is plotted for.
* This completes the graph.

**2.1.d : Post release quality by Programming language(normalized by size)**

**Overview:**

* Purpose: The purpose of the graph is to compare which programming language is more effective when developing a product.
* Question: The question being answered is, when is the quality of a certain product is more when one kind of a programming language is used. So in short, we can know if a particular language has an impact on the quality of the product.
* Definition: Post release quality by programming language is defined by the number of defects encountered for a product after its release for a particular language.
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

**SAMPLE GRAPH:**

****

ANALYSIS: GENERAL DESCRIPTION:

* PURPOSE: The purpose of this graph is to study and analyze if a particular programming language has an effect on the quality of the product.
* To know if a programming language is good enough to follow.
* This graph also tells us which products are suited for what .
* It also gives us a slight clue about when to use what programming language.
* This graph tells us that if the programming language we are using is accepted in the present market or is it rejected for having ineffective results.

GRAPH:

* In this graph, we plot 4 lines , total defects for JAVA , total defects for Ruby , total uncorrected defects for Java and total uncorrected defects for Ruby.
* It shows the defects per 1000 lines of code for a product implemented using one of the above two languages.
* A curve/line too high means that the programming language is ineffective and not suitable for the product to be built with.
* Months is denoted on the horizontal axis.
* Number of defects per 1000 lines of code is denoted on the vertical axis.
* This graph is basically used to compare between the two languages.
* When the violet line is lower than the green line , it means that ruby is more effective than java and if the other way round then its vice versa respectively in favor of Java.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph denotes the average of total defects per month for all the products per 1000 lines of code normalized by size for the two programming languages namely Ruby and Java.
* The Java and Ruby each have 2 lines namely, uncorrected defects and total defects.
* Total defects are the known defects at the time of release summed up with every possible defect encountered with respect to the product each month for
* Higher the line, higher is the negativity.
* We expect to see those lines lower or tending lower.
* The number of months is denoted on the horizontal axis.
* As we can see clearly, Ruby is more consistent.
* Ruby clearly has the lower bound curve than java for total uncorrected defects.
* This shows that its easier to maintain products developed with ruby than products developed with java.
* Total defects that occurred with products built with java is much more in number when compared to the products built with ruby.
* Clearly, from the above graph, it can be concluded that Ruby is more efficient for most part of the time compared to java.
* A sudden burst of raise in any curve of the graph means that there has been a serious defect in the programming language followed.
* Often, it has been seen that quality of a product is directly proportional to the programming language followed to develop it.
* We DONOT expect to see sudden changes in the curves in this graph.
* When the violet line is lower than the green line , it means that Ruby is more effective and if the other way round then its vice versa respectively.

A consistently lower bound line of the graph tells us that the language followed is effective in building the product

Recommendations / Comments:

* If we see a curve consistently higher bound, it’s a concern for the company to look into the programming language implementation.
* If the above case happens repeatedly, then the higher management should come up with a programming language with the consult of senior developers to come to a conclusion as to which language is more suited and upscale to build the respective product.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Defects per 1000 lines of code normalized by size: This is used to normalize the calculations since, when taking the average for all the products, each product has a different size. Hence there might be a possibility that more defects might be encountered for a product with higher size. To eliminate this drawback, we normalize the defects to 1000 lines of code and then calculate the average.

This number (uncorrected defects per 1000 lines of code) is calculated using the formula (N-C)/SIZE \* 1000 for each product with its respective size. (Separately for each programming language)

To calculate the number of total defects per 1000 lines of code , we use the formula N/SIZE \* 1000.

The average is then calculated for each month with respect to each programming language separately.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

Uncorrected Defects per 1000 lines of code normalized by size is calculated using the formula (N-C)/SIZE\* 1000.

Total defects per 1000 lines of code is N/Size \* 1000.

The average is taken for every month and plotted against the same on the graph.

The month is denoted on x-axis.

COMPOUND MEASURES:

Uncorrected Defects per 1000 lines of code is calculated using the formula (N-C)/SIZE\* 1000

All these measures are calculated for each month for the respective programming languages and each programming languages separately and then the total is done and the average is taken for each month.

**GENERATING THE GRAPH:**

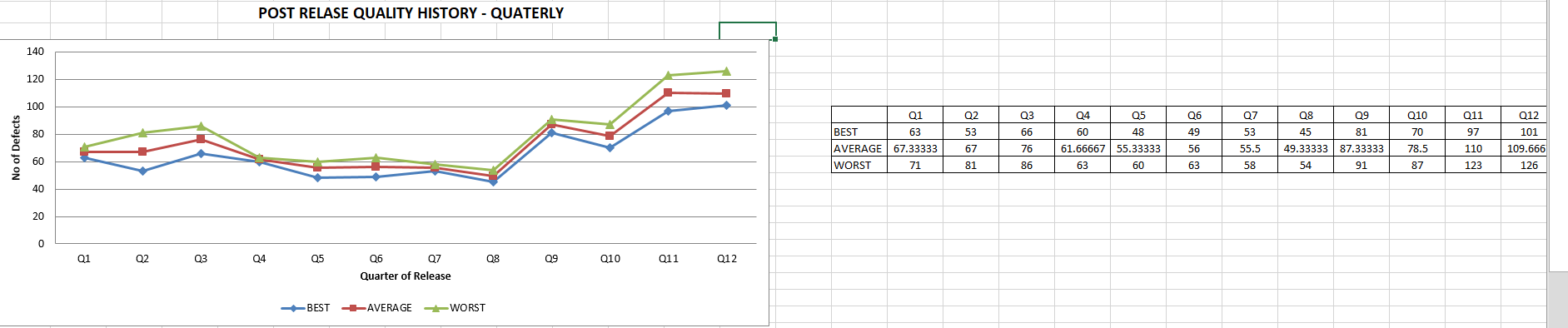
* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* The value got from computing (N-C/Size) \* 1000 for all the products individually using Ruby and summing them up for each month and taking the average gives us the Violet line (total uncorrected defects with Ruby)
* The value got form computing (N-C/Size) \* 1000 for all the products individually using Java and summing them up for each month and taking the average gives us the Green line (total uncorrected defects with Java)
* The value got form computing (N/Size) \* 1000 for all the products individually using Ruby and summing them up for each month and taking the average gives us the Red line (total defects with Ruby)
* The value got form computing (N/Size) \* 1000 for all the products individually using Java and summing them up for each month and taking the average gives us the Blue line (total defects with Java)
* The horizontal axis is the months for which the graph is plotted for.
* THIS COMPLETES THE GRAPH.

**2.1.e: POST RELEASE QUALITY HISTORY(not normalized):**

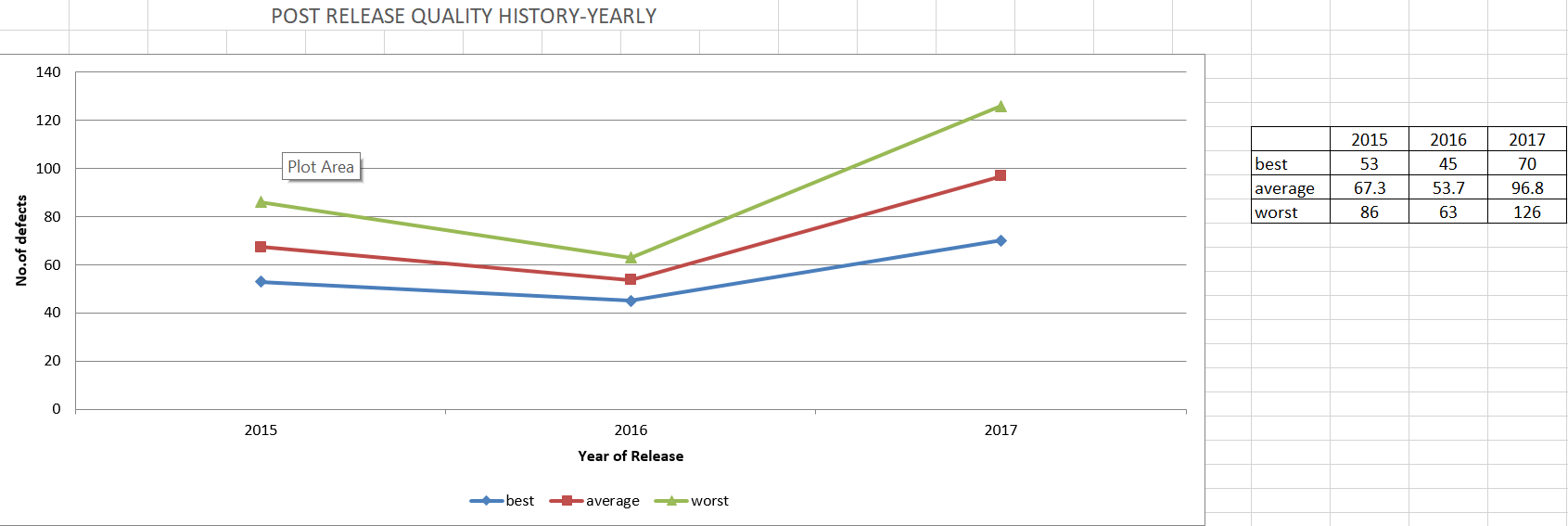
**Overview:**

* Purpose: The purpose of this graph is to “know” how our product is performing in the market post release.
* Question: The question being answered is, “overall, how good were the products we shipped over several years in terms of defect rates and have things changed over time?"
* Definition: Post release quality history is defined as “the performance of the product in the market in terms of number of defects encountered post release “
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

**SAMPLE GRAPH:**

****

**This is a graph plotted for quarterly period.**

****

**This graph is plotted for yearly period .**

ANALYSIS: GENERAL DESCRIPTION:

* PURPOSE: The purpose of this graph is to understand the usage of our product over a long period of time and how is it performing in a practical scenario.
* To know if our product has satisfied all our customers’ requirements for a longer time.
* This graph also tells us which products are suited for what customers.
* It also gives us a slight clue about when our project succeeded and when it failed.

This graph tells us about a detailed stance of our products in the industry.

* It gives us a clue if we have to continue with the same process.
* It also gives an information about the long term usage of our clients and hence the quality of our products.
* It indicates which product of ours failed and which all succeeded.

GRAPH:

* In these 2 graphs, we plot 3 lines for each graph, which shows the best case, worst case and average case over quarterly period and yearly period.
* It shows total defects of a code for a product on the vertical axis.
* A curve/line too high over a consistent period of time shows that our product hasn’t done well.
* Time period is denoted on the horizontal axis.
* This graph is basically used to check if our product has met customer expectations.
* The best case , worst case and average case is as shown in the graph.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* Careful analysis of the graph for the quarterly period of time is as follows.
* The horizontal line is denoted for quarterly period of time
* The vertical line is for the total defects.
* So as we can see, quarters 3 through 8 were our best period in terms of defects for our products. These quarters show that our products had the least defects during that period.
* Quarters 9 through 12 is our worst period since those quarters have experienced a lot of defects.
* The best case scenario had at most 40 defects
* The worst case scenario had at most 123 defects and our average case was 110 defects at most for a particular quarter.
* As we can see clearly, Quarter 8 was our most effective quarter.
* A sudden burst of raise in any curve of the graph means that there has been a serious defect in the product followed.
* Often, it has been seen that quality of a product is directly proportional to the programming language and process followed to develop it.
* We DONOT expect to see sudden changes in the curves in this graph.

Recommendations / Comments:

* Much work should be put into quality assurance to bring down the total defects to a significantly lower value and this can be ensured by proper training of developers and testers.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product in each quarter.

Summation of all the results for all the quarter results in graph 1.

Summation of all the results for all the years from 2015 through 2017 leads to graph 2.

The average is then calculated for each month with respect to each product separately.

Release date: the date on which the product is released.

Product name: the name of the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The total defects is calculated for each product’s “N” and then the average is taken for each month.

3 months make a quarter. We have 12 quarters representing 3 years’ worth of data.

The average is taken for every month and plotted against the same on the graph.

The month (quarters) is denoted on x-axis.

**GENERATING THE GRAPH:**

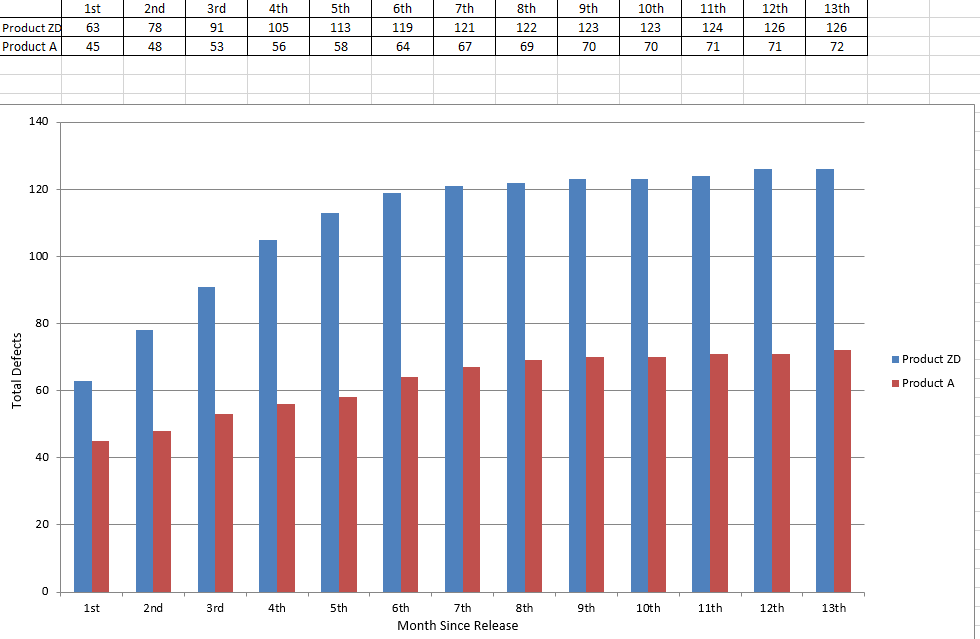
* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* The Average is got by computing the average for every month and then quarters are computed . summation of averages of 3 months.
* The Average is got by computing the average for every month and then 12 months are combined to form a year.
* **We get the best case , worst case and the average case after careful analysis .**
* Best case is when there are minimum defects for a product.
* The worst case is when there are maximum defects for a product pertaining to a quarter .
* The average case comes in between the best and the worst case.
* This completes the graph.

**2.1.f Graph of your choice( Immediate Post Release Quality of Product ZD (Most Recent Product) compared to Product A (Least Recent Product)**

**Overview:**

* Purpose: The purpose of the graph is to compare our most recent product with that of our least recent one to identify the differences in the quality that has been affected our company over a period of years.
* Question: The question being answered is, are our new employees and new procedures producing the same quality of code that used to be? Is our product any better or wirse than our 1st ever released product.
* Definition: This problem is defined as the difference between the total defects respective for both the products immediately upon release.

Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

SAMPLE GRAPH****

ANALYSIS: GENERAL DESCRIPTION:

* PURPOSE: The purpose of this graph is to study and analyze how our most recent product compares with the least recent one in terms of product quality.
* To know if we have to improve the current product or keep following what we are used to.
* This graph also tells if we are heading towards right direction.
* It also gives us a slight clue about when are we improving and when are we not.

GRAPH:

* This is a column graph of 2 columns for each month since its release
* It shows total defects dealt with the product since its release on each month .
* A curve/line too high means that the product is performing bad and might be a demerit for the company.
* Months is denoted on the horizontal axis as in , from the date of its release , to each month for 1 year.
* Total Number of defects is denoted on the vertical axis.
* This graph is basically used to compare between the two products.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph denotes the total defects dealt with both the products post release for exactly 1 year.
* As the graph shows , we have plotted product A in red column bar and product ZD in the blue column bar.
* Total defects are the known defects at the time of release summed up with every possible defect encountered with respect to the product each month for
* Higher the line, lower the quality.
* We expect to see those lines lower or tending lower.
* The number of months is denoted on the horizontal axis.
* As we can see clearly, Product A was much higher in quality than product ZD
* The worst case of Product A is almost equal to the Best case of product ZD which is a total concern with respect to ZD.
* This shows that its easier to maintain A than to maintain ZD.
* Total defects that occurred with product A is significantly lesser than product ZD. Clearly, from the above graph, it can be concluded that A is far better than ZD in all aspects.
* A sudden burst of raise in any curve of the graph means that there has been a serious defect in the product.
* We DONOT expect to see sudden changes in the curves in this graph.

Recommendations / Comments:

* We see that product A is significantly more qualitative than product ZD which suggests that the principle followed during that time was more efficient.
* If the above case happens repeatedly, then the higher management should come up with a new idea to make the business better.
* Since there is so much difference in the quality , its better to preserve employees who have been working since the inception of company by paying them better rather than adapting new resources who have minimum product knowledge.
* This calls for a much better product knowledge and hence more training should be given to resources involved in product development

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The total defects for both the products are summed up for each month and plotted as a column graph.

Total defects is N which is taken from the spreadsheet .

**GENERATING THE GRAPH:**

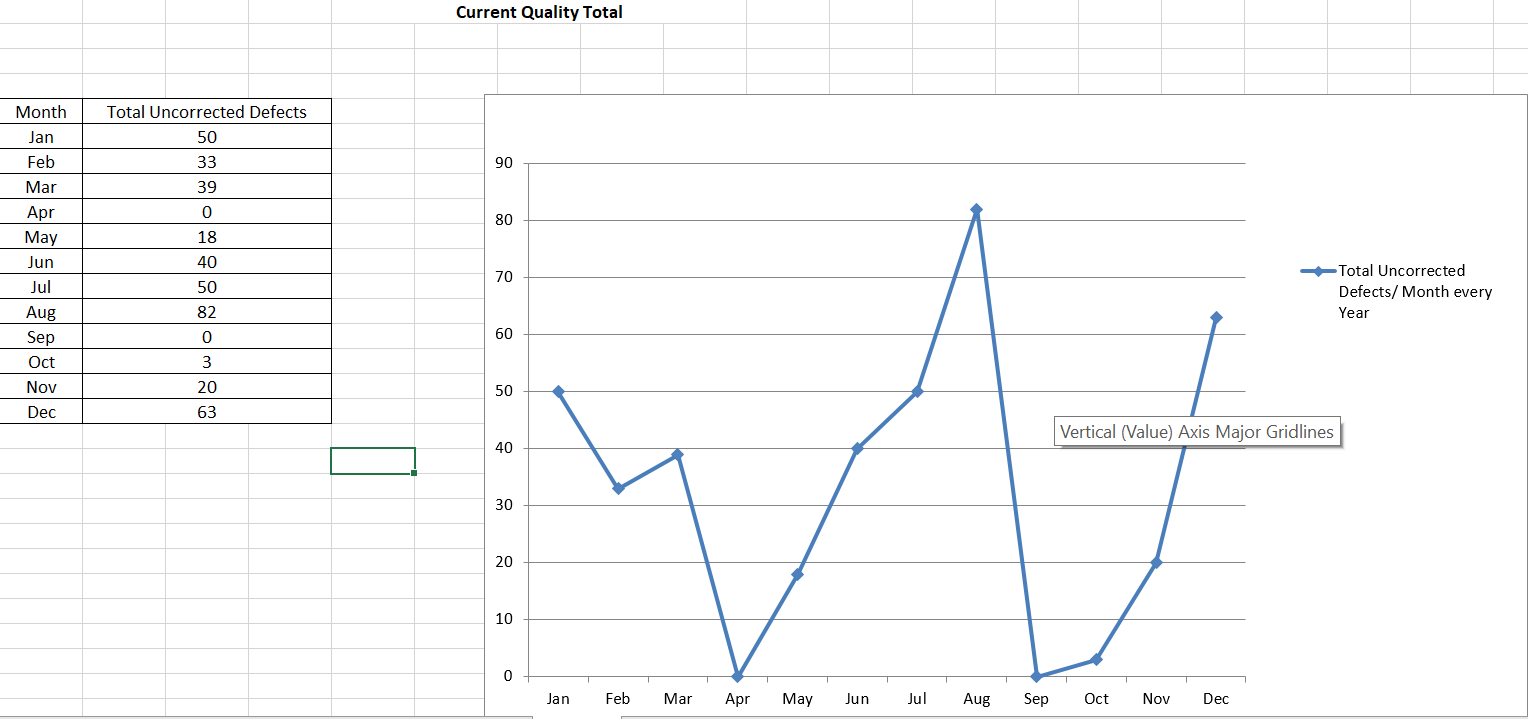
* The total defects is the value N which is taken from the spreadsheet
* We look closely , the number of defects increasing each month for the respective products.
* We then plot a column graph .
* The vertical axis is the total defects.
* The horizontal axis is the months for which the graph is plotted for.
* THIS COMPLETES THE GRAPH.

**2.2.a Current Quality Total :**

**Overview:**

* Purpose: The purpose of the graph is to study the current quality that’s corresponding for each month for all our supported products.
* Question: The question being answered is, “IS our current quality good enough to satisfy the customers”.
* Definition: Current quality is defined as current quality, by month, for 3 years, using all active products each month.
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course. Data here is the amount of defects encountered with the product.

GRAPH:



ANALYSIS: GENERAL DESCRIPTION:

* PURPOSE: The purpose of this graph is to study and analyze the current quality of all the supported products to know how well we are doing in the market NOW.
* To know if a product is good enough.
* To know when our product is working fine and when not.
* It also gives us a slight clue about when we have to take extra care for our products.
* It also suggests if we can maintain the product we have built.

GRAPH:

* In this graph, we plot a single line that describes the total uncorrected defects for each of the supported products each and every month since its release.
* It shows total uncorrected defects on the vertical axis and the months respective of it in the horizontal axis
* A curve/line consistently too high means that our products are ineffective and not suitable for the customers’ requirements
* This also gives a hint about verifying the requirements/SPECS again.
* This graph is basically used to compare the current quality for each month.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph denotes the total defects for all the currently supported products per month.
* In the month of May and October , our products do the best and hence our employees since the number of uncorrected defects is minimum during that time.
* Total uncorrected defects are the known total defects minus the corrected ones each month for every active product in the market
* Higher the line, lower is the quality.
* We expect to see those lines lower or tending lower.
* The number of months is denoted on the horizontal axis.
* During august and December our products perform at their least desired behavior exhibiting highest number of uncorrected defects.
* A sudden burst of raise in any curve of the graph means that there has been a serious defect in the respective product followed.
* When a scenario such as the one above happens , careful examination of the product has to be done and made sure that for the nest release , the defects has been fixed.
* We DONOT expect to see sudden changes in the curves in this graph.
* From April through August, there has been a sudden raise in the curve for the total uncorrected defects which is always a bad sign . This shows that our products at that time did not perform as expected.

A consistently lower bound line of the graph tells us that the products are effective quality wise and have satisfied most of the customer requirements if not all.

Recommendations / Comments:

* If we see a curve consistently higher bound for a particular month , we have to look at what product we supported is causing this curve to go high. Then after examining the product , careful engineering and further defect correction must be made sure in order to bring that value down.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

The average is taken for every month and plotted against the same on the graph.

The month is denoted on x-axis.

COMPOUND MEASURES:

Uncorrected Defects=N-C.

All these measures are calculated for each month for the respective products and each products separately and then the total is done and the average is taken for each month.

**GENERATING THE GRAPH:**

* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* The value got from computing N-C is plotted against each month.
* Only those products that are active and supported are considered.
* The horizontal axis is the months for which the graph is plotted for.

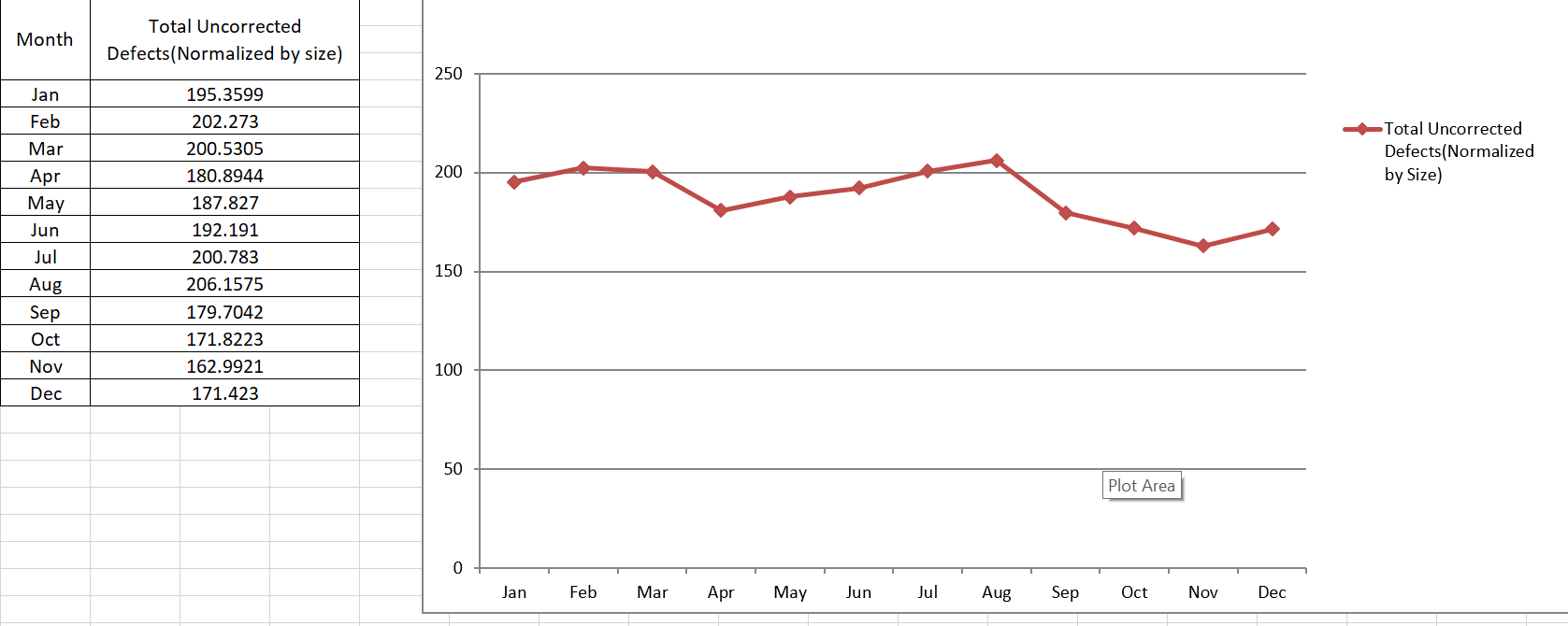
THIS COMPLETES THE GRAPH.

**2.2.b Current Quality Total Normalized by Size**:

**Overview:**

* Purpose: The purpose of the graph is to understand the current quality of all the supported products but normalized by size.
* Question: The question being answered is, “is the current quality of all of our supported products meet the expectations? should anything more to be done to improve the present quality?”
* Definition: Current quality is defined as current quality, by month, for 3 years, using all active products each month but normalized by size.
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course and still being supported. Data here is the amount of uncorrected defects encountered with the product.

GRAPH:



**Analysis: General Description:**

* Purpose: This graph is used to study the current quality of all the supported products normalized by size over its period of usage.
* This helps in knowing the current quality of the product.
* This also helps us understand if we have met our customers’ current expectations.
* This also helps us understand how our team, as an organization, performed over the last few years compared to this term.
* **Graph:**
* In this graph we plot a single line for the graph.
* The horizontal axis shows the month scale and vertical axis shows “defects per 1000 lines of code”.
* It shows the current comparison of the products which are normalized by size.
* The single line indicates the current quality for all the products normalized by size for each month.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph shows the total uncorrected defects currently for all the products
* As a result we can see that the total uncorrected defects which gives a direct insight about the quality and hence the level of maintenance at this particular point in time for all the products.
* We can also see how our products are performing currently with respect to customer point of view. So as a result, currently , products that are supported for January through August are weak in quality and needs special attention to be checked upon.
* The graph also tells the total uncorrected defects of the products from august through December has raise in quality.
* As we carefully examine the graph , we can see that the currently active products from august through December has a raise in quality since the number of uncorrected bugs has come down significantly.
* We can also conclude that all the currently supported products needs more attention to bring the total uncorrected defects to a lower value.

**Specific Recommendations:**

* A thorough analysis of SPECS need to be done the next time we are developing a product since the amount of initial defects is much more than the expected value.
* Proper training of QA/testing team needs to be done to ensure that the product defects are found well within the release so that the developers can fix the defects.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Defects per 1000 lines of code normalized by size: This is used to normalize the calculations since, when taking the average for all the products, each product has a different size. Hence there might be a possibility that more defects might be encountered for a product with higher size. To eliminate this drawback, we normalize the defects to 1000 lines of code and then calculate the average.

This number (uncorrected defects per 1000 lines of code) is calculated using the formula (N-C)/SIZE \* 1000 for each product with its respective size.

To calculate the number of total defects per 1000 lines of code , we use the formula N/SIZE \* 1000. We use this for all the currently supported products.

The average is then calculated for each month.

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The language used (Java or Ruby) to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

Uncorrected Defects per 1000 lines of code normalized by size is calculated using the formula (N-C)/SIZE\* 1000.

Total defects per 1000 lines of code is N/Size \* 1000.

The average is taken for every month and plotted against the same on the graph for all the currently supported products.

The month is denoted on x-axis.

COMPOUND MEASURES:

Uncorrected Defects per 1000 lines of code is calculated using the formula (N-C)/SIZE\* 1000

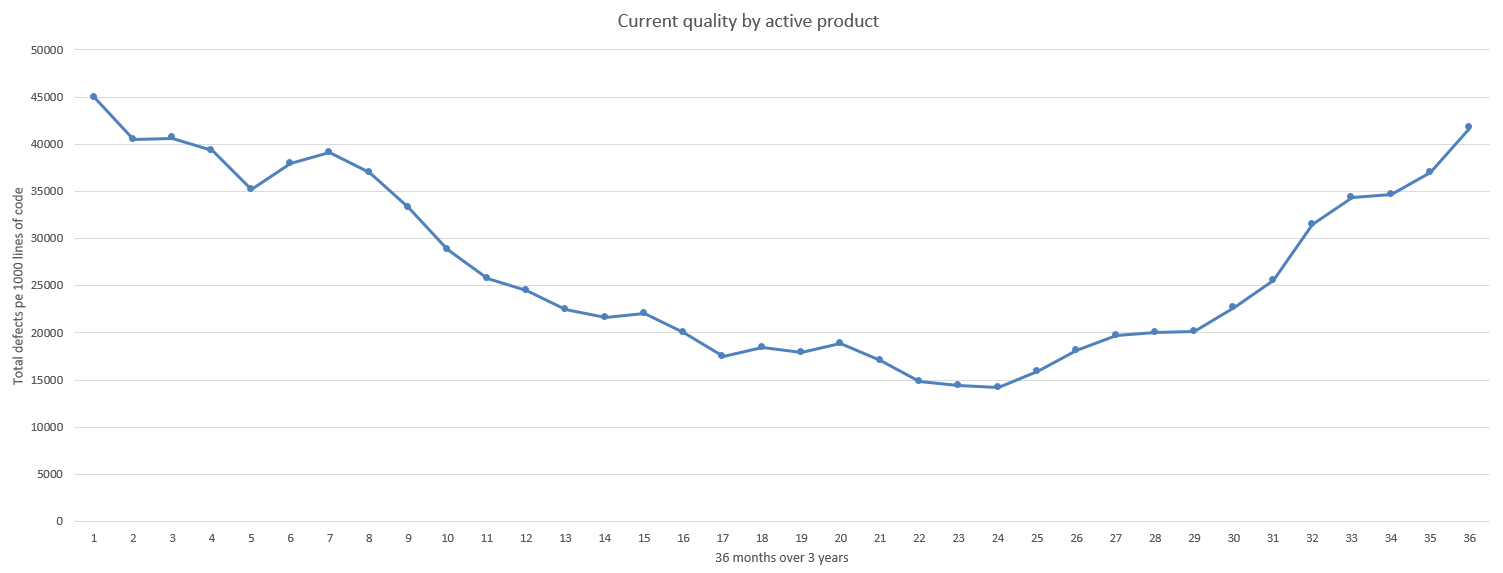
**GENERATING THE GRAPH:**

* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* **(N-C/SIZE)\*1000 is calculated for each product currently supported and plotted against each month taking the total and plotted on the graph.**
* The horizontal axis shows the months.
* This generates the graph.

2.2.c **Current Quality Total Normalized by Number of Products**

**Overview:**

* Purpose: The purpose of the graph is to understand the current quality of all the supported products but normalized by number of currently active products each month that we support. This gives us an insight if our products are doing what they are supposed to do. Helps us with the microscopic quality of each product that we are supporting.
* Question: The question being answered is, “is the current quality of all of our supported products meet the expectations? should anything more to be done to improve the present quality?”
* Definition: Current quality is defined as current quality, by month, for 3 years, using all active products each month but normalized by the number of products we support each month.
* Collection Frequency: Data is collected monthly for every released product from the day of its release through its entire course and still being supported. Data here is the amount of uncorrected defects encountered with the product.

SAMPLE GRAPH: 

**Analysis: General Description:**

* Purpose: This graph is gives the information about the quality of all the products supported currently supported over the past 3 years for every month.
* Since we have shipped and supported many products over time , we need to know , how , in general are these products performing with respect to the quality.
* This also helps us understand what are the customer needs now, in the present market and are we up to the needs of the customers.
* This also helps us understand how our team , as an organization , performed over the last few years in aspect of developing a product.
* **Graph:**
* In this graph we plot a single line against total number of defects for all the products combined for a whole period of time (3 years in our case)
* The horizontal axis shows the month scale and vertical axis shows “defects per 1000 lines of code”.
* It shows the comparison between the products shipped over time with the average of rest of the products post release and currently in use by taking into account the number of defects raised per 1000 lines of code.

**ANALYSIS: Specific Description:**

* **What the graph shows:**
* The graph shows the total defects of all the products we have shipped over the past 3 years .
* As a result, we can see that the total defects and hence the quality was bad at first, when shipped and then it started to increase. During the 2nd year, the quality of our products was best and then the quality again started to decrease.
* This means that the total number of bugs or defects is increasing with time after the 2nd year which is a bad sign.
* This shouldn’t happen because the goal was to achieve a desired quality and that was achieved during the 2nd year, After the commencement of 3rd year, the quality again started to decrease.
* This fluctuation shouldn’t happen.
* It shows that the products shipped after the commencement of 3rd year was difficult to maintain and our developers have a hard time fixing the defects for most part of the time.

**Specific Recommendations:**

* A careful analysis is to be done on all the resources and look back what exactly changed after the commencement of 3rd year where in the quality immediately started to decrease.
* Proper training of QA/testing team needs to be done to ensure that the product defects are found well within the release so that the developers can fix the defects.

**Procedure for refining and analyzing the data and producing the graph**

**Base Measures:**

Total defects: (N) The total number of New defects encountered each month with respect to the product.

Corrected defects (C): The number of defects that is corrected from N.

Uncorrected defects (N-C): These are the uncorrected defects that represent the current quality of the product. It is got by subtracting the corrected defects C from the total defects N.

Defects per 1000 lines of code normalized by the number of products: This is used to normalize the calculations since, we have a shipped and are currently supporting a wide range of products.

This number (total defects per 1000 lines of code) is calculated using the formula (N)/SIZE \* 1000 for each product with its respective size.

To calculate the number of total defects per 1000 lines of code , we use the formula N/SIZE \* 1000.

The total is then calculated for each month .

Release date: the date on which the product is released.

Product name: the name of the product.

The method used (agile or scrum) used to develop the product.

The language used (Java or Ruby) to develop the product.

The size of the product (recorded in Lines of Code).

All these data are recorded in the different columns of the spreadsheet.

**DATA REFINEMENT**:

The uncorrected defect is calculated by subtracting C from N.

Uncorrected Defects per 1000 lines of code normalized by size is calculated using the formula (N-C)/SIZE\* 1000.

Total defects per 1000 lines of code is N/Size \* 1000.

The average is taken for every month and plotted against the same on the graph.

The month is denoted on x-axis.

COMPOUND MEASURES:

Total Defects per 1000 lines of code is calculated using the formula (N)/SIZE\* 1000

**GENERATING THE GRAPH:**

* The total defects is the value N which is taken from the spreadsheet
* The corrected defects are C.
* N-C gives us the uncorrected defects.
* We compute the total defects for each product supported during each month.
* Then , the summation is done for every month and every year from then on , to compute the curve.
* This graph gives us a brief insight about the quality of products over the whole 3 years sampled with normalization of number of products.
* The previous graph gave us the insight about the yearly analysis of defects for the products normalized by size.
* This graph is helpful in analysis of current quality of our products as well as comparing it with the previous quality of the products that we had shipped.
* The previous graph doesn’t provide this facility.
* This generates the graph.