[cover sheet 1 of 2]

ASSIGNMENT 2

**DEFECT ANALYSIS REPORT**

CSE 6329 -- SOFTWARE MEASUREMENT AND QUALITY ENGINEERING

Professor Dennis J. Frailey

**Fall, 2018**

NAME \_Harshini Chandrasekar, Tharuna Kumar\_

ID Number\_\_1001586563, 1001537450\_\_

Grader Comments:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Grading Template (student should not write inside this box)** | | | | | | |
| **Section 1 – Introduction and Overview** | | | | | | |
| 8.75(/10) | 1.1 1 Purpose of This Report (1)  1.2 3 Structure of Analysis Tool (4)  1.3 2.75 Data Collection Overview (3)  1.4 2 Summary of Analyses and Graphs (2) | | | | | (10 points) |
| **Section 2 – Measures, Graphs and Analysis** | | | | | | |
| **Section 2.1 – Post Release Quality** | | | | | | |
|  | i (1 point)  **Overview**  (Purpose / Question / Definition / Collection) | ii (1 point)  **Sample Graphs** | iii **Analysis** (5 points) | | iv (3 points)  **Procedure**  (Base measures / Refinement / Compound Measures / Generation of Graph) | |
| 1. (2 pts)  **General**  **Description**  (Purpose / Graph / How to Analyze) | 2. (3 pts)  **Specific Discussion**  (What it shows / recommendations) |
| 2.1.a 9.5/10 | 1 | 1 | 2 | 2.5 | 3 | |
| 2.1.b 8.5/10 | 0.5 | 1 | 2 | 2.5 | 2.5 | |
| 2.1.c 7/10 | 0.5 | 0 | 2 | 2 | 2.5 | |
| 2.1.d 8/10 | 0.5 | 0 | 2 | 3 | 2.5 | |
| 2.1.e 8.75 /10 | 1 | 1 | 1.75 | 3 | 2 | |
| 2.1.f 10 /10 | 1 | 1 | 2 | 3 | 3 | |
| **Section 2.2 – Current Quality** | | | | | | |
|  | i (1 point)  **Overview**  (Purpose / Question / Definition / Collection) | ii (1 point)  **Sample Graphs** | iii (5 points) | | iv (3 points)  **Procedure**  (Base measures / Refinement / Compound Measures / Generation of Graph) | |
| (2 pts)  **Generic**  **Description**  (Purpose / Graph / How to Analyze) | (3 pts)  **Specific Discussion**  (What it shows / recommendations) |
| 2.2.a 8.5/10 | 1 | 0.5 | 2 | 2.75 | 2.25 | |
| 2.2.b 7.75/10 | 0.75 | 0.5 | 1 | 3 | 2.5 | |
| 2.2.c 8/10 | 0.5 | 0.5 | 1.75 | 3 | 2.25 | |
| 84.75\_/100 | ← **Total Assignment Grade** | | | | | |

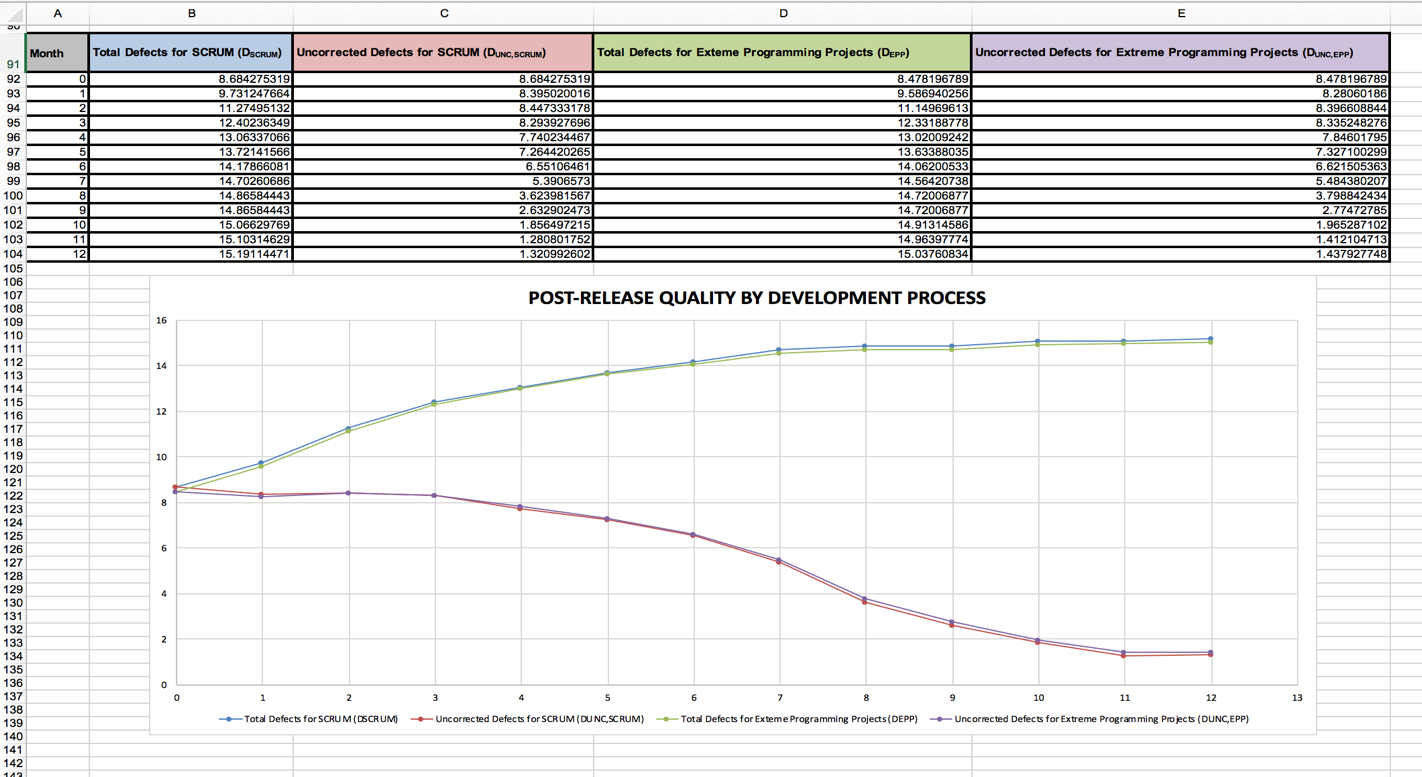
**Defect Analysis Report**

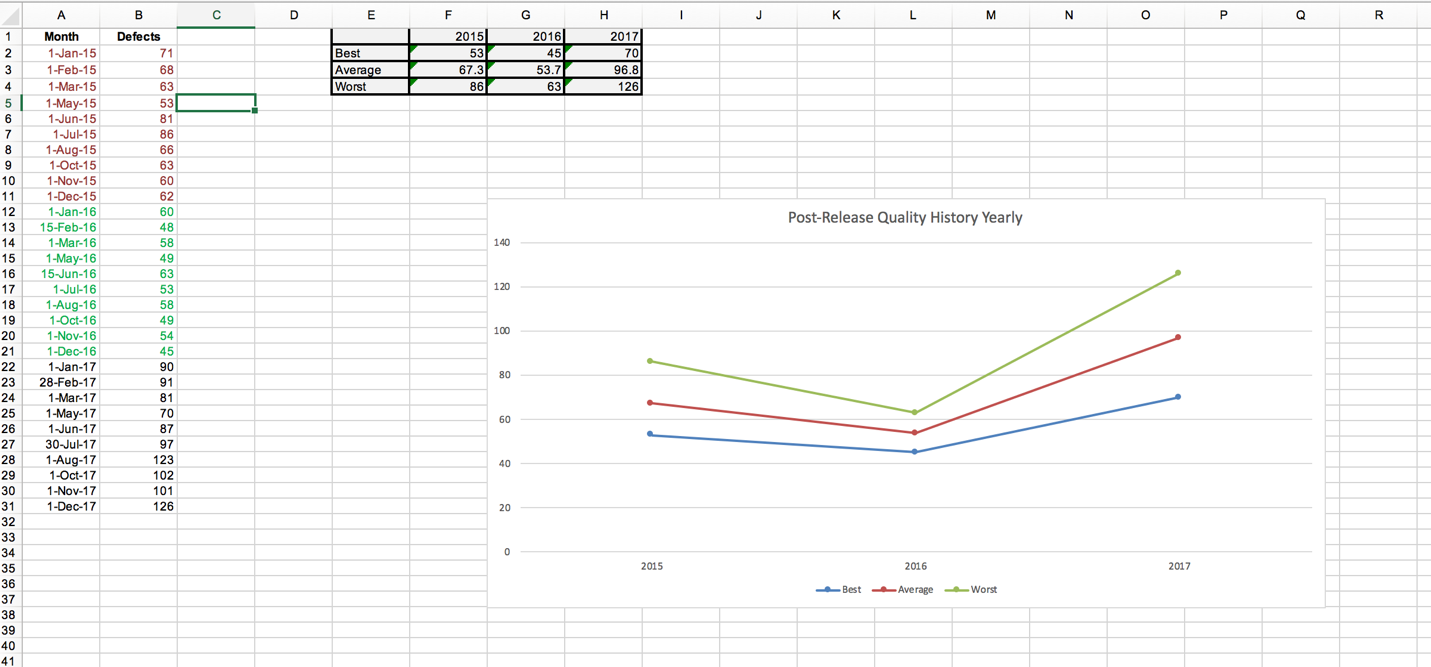
1. **Introduction**
   1. **Purpose of This Report**

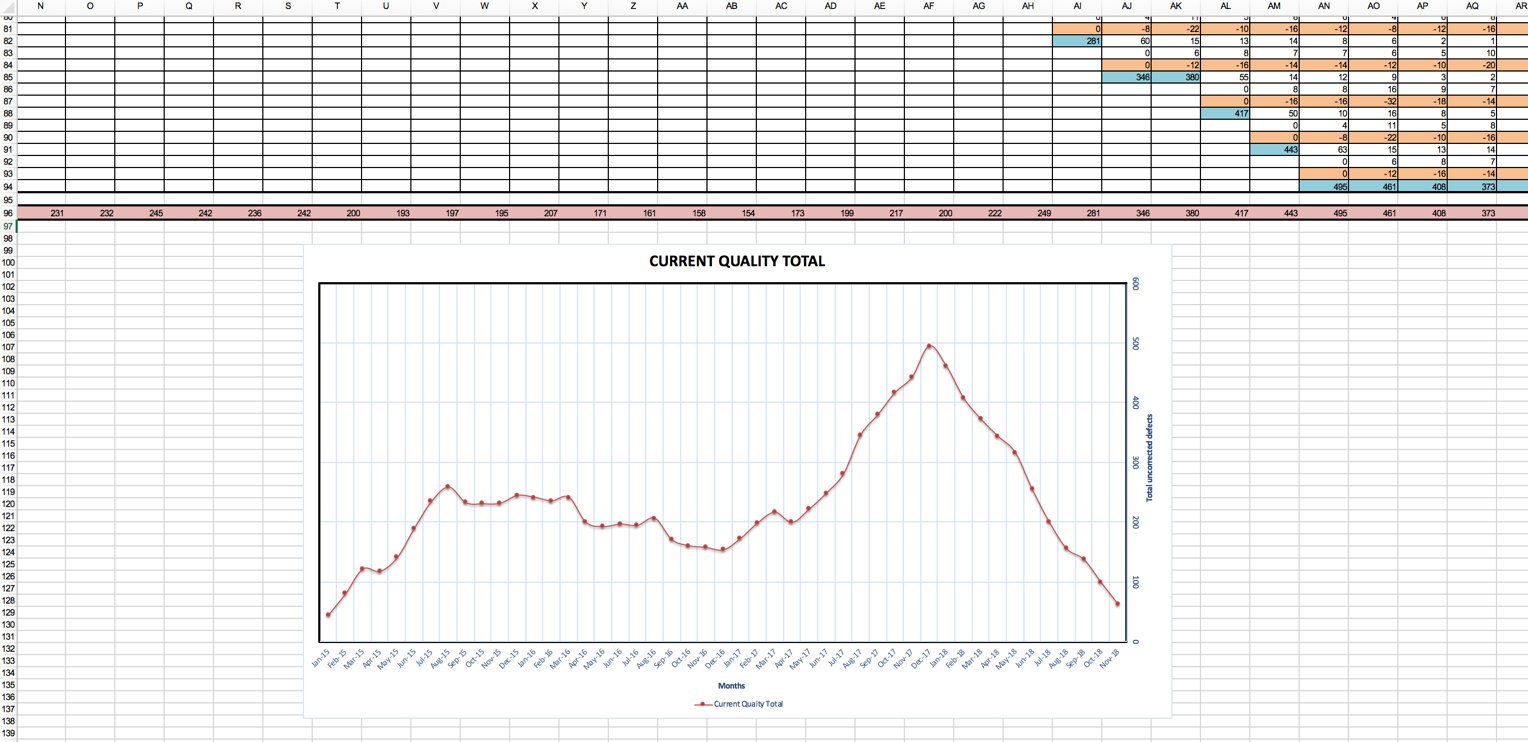
This report shows the results of analyzing three years of defect data on our released products. The purpose is to gain a greater understanding of the quality levels of our released products and to determine whether there is any correlation between software quality and other factors such as the programming language used, the development process used, or the time when the product was developed. It also helps to analyze and measure customer satisfaction and assign more resources to defect correction if necessary.

* 1. **Structure of Analysis Tool**

In order to analyze defect data, we have created a Microsoft Excel workbook in which we refine and analyze the data in different manners to depict various measures that analyze the situation. This workbook contains 12 worksheets and each of them are explained below:

* Index: This worksheet contains the list of measures examined.
* Page 1: This contains the given data that has been collected over three years (2015-2017).
* Page 2: This worksheet measures the ‘Post-release Quality’ for product ZB.
* Page 3: This contains the post-release quality average for all products and is normalized by size.
* Page 4: This contains the post-release quality average for all products by development process.
* Page 5: This contains the post-release quality average for all products by programming language.
* Page 6: This contains the post-release quality history for all products by year.



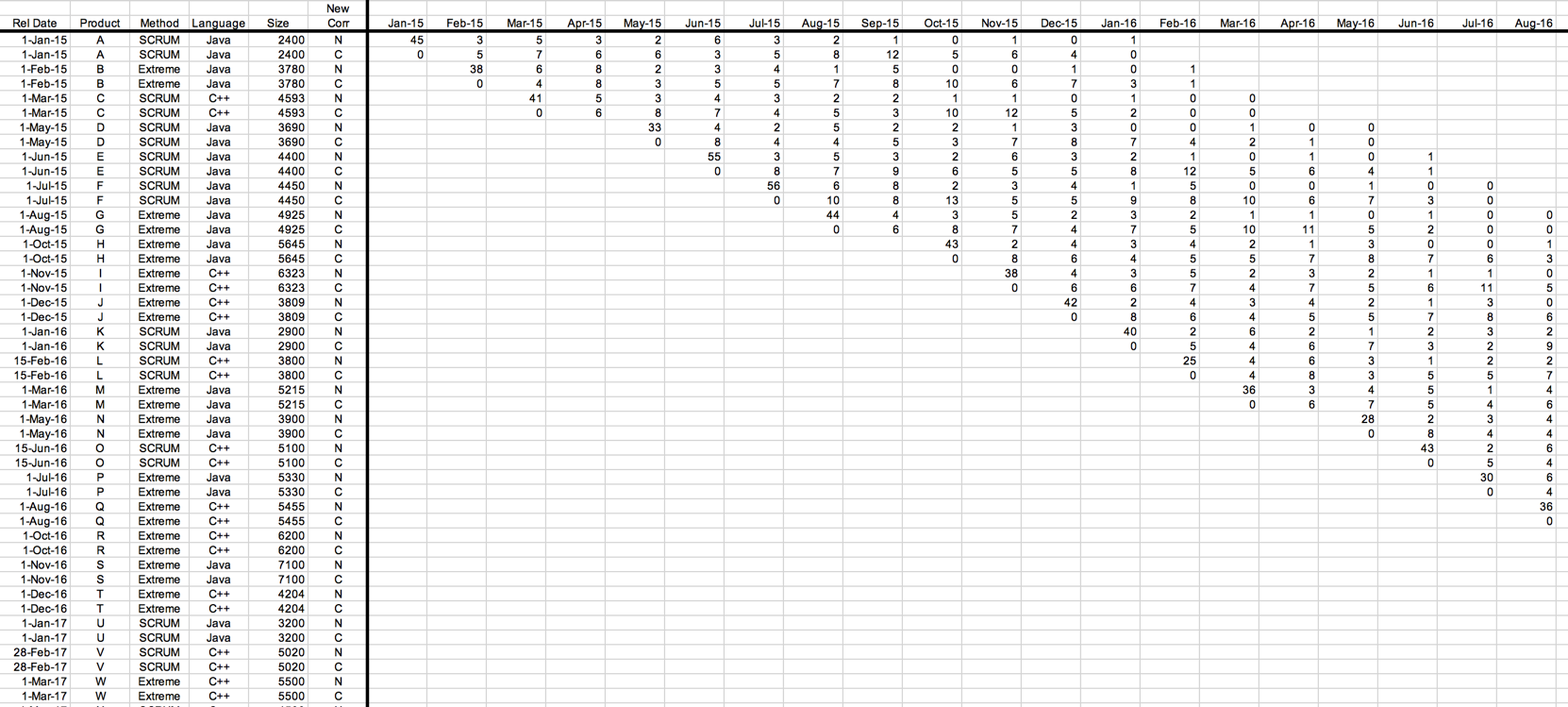
* Page 7: This contains the post-release quality history for all products by quarter.
* Page 8: This contains employee productivity measure.
* Page 9: This contains the current quality total for all products.
* Page 10: This contains the current quality total for all products, normalized by size of currently active product releases.
* Page 11: This contains the current quality total for all products, normalized by number of currently active products.
  1. **Data Collection Overview**

The data collected is available in the form of a spreadsheet. The data shows defect information for 30 products that have been released to customers over the three years from 2015 to 2017. The spreadsheet contains the following columns:

* **Rel Date**: This is the date the product was released.
* **Product**: This is the name assigned to the product during development.
* **Method**: It is the name of the development process (lifecycle) used.
* **Language**: It is the name of the programming language used to write the software.
* **Size**: This contains the number of source lines of code.
* **New Corr**: This column contains two values. ‘N’ represents the number of new defects detected during that month. ‘C’ represents the number of defects corrected during the month.

The spreadsheet has two rows for each product which represent the N and C values for its lifetime.

The data shows products that have been released during the span of three years from 2015 to 2017. Data for 2018 shows the post-release data for products that have been released in 2017. There are no new releases shown in 2018.

Defect information is collected every month for active products. For example, in the figure given below, product A, was released on 1 January 2015 and had 45 known defects. The number of corrected defects for January 2015 is 0 for this product. At the end of the first month, i.e., at the beginning of February 2015, 3 more defects had been discovered and 5 defects had been corrected.

**1.4 Summary of Analyses and Graphs**

|  |  |  |
| --- | --- | --- |
| **GRAPH** | **REPORT SECTION** | **PURPOSE** |
| Post-release Quality for product ZB | 2.1. a | The graph shows the number of defects in the product ZB each month after release for one year. |
| Post-release Quality average for all products, normalized by size | 2.1. b | The graph shows the total defects and total uncorrected defects per 1000 lines of code. |
| Post-release Quality average for all products, by development process | 2.1. c | The graph shows the total defects and total uncorrected defects per 1000 lines of code, for the development process used (SCRUM and Extreme). |
| Post-release Quality average for all products, by programming language | 2.1. d | The graph shows the total defects and total uncorrected defects per 1000 lines of code, for the programming language used (C++ and Java). |
| Post-release Quality history | 2.1. e | The graph shows the best, worst, and average case defect count for all products by year and quarter. |
| Employee Productivity Measure | 2.1. f | The graph shows the employee productivity over the years 2015, 2016, 2017, 2018. |
| Current Quality Total | 2.2. a | The graph shows total uncorrected defects for all the products released. |
| Current Quality Total for all products, normalized by size | 2.2. b | The graph shows total uncorrected defects for all the products released normalized by the size of the product. |
| Current Quality Total for all products, normalized by number of active products | 2.2. c | The graph shows total uncorrected defects for all the products released normalized by the number of active products. |

1. **Measures, Graphs and Analysis**
   1. **Post Release Quality**

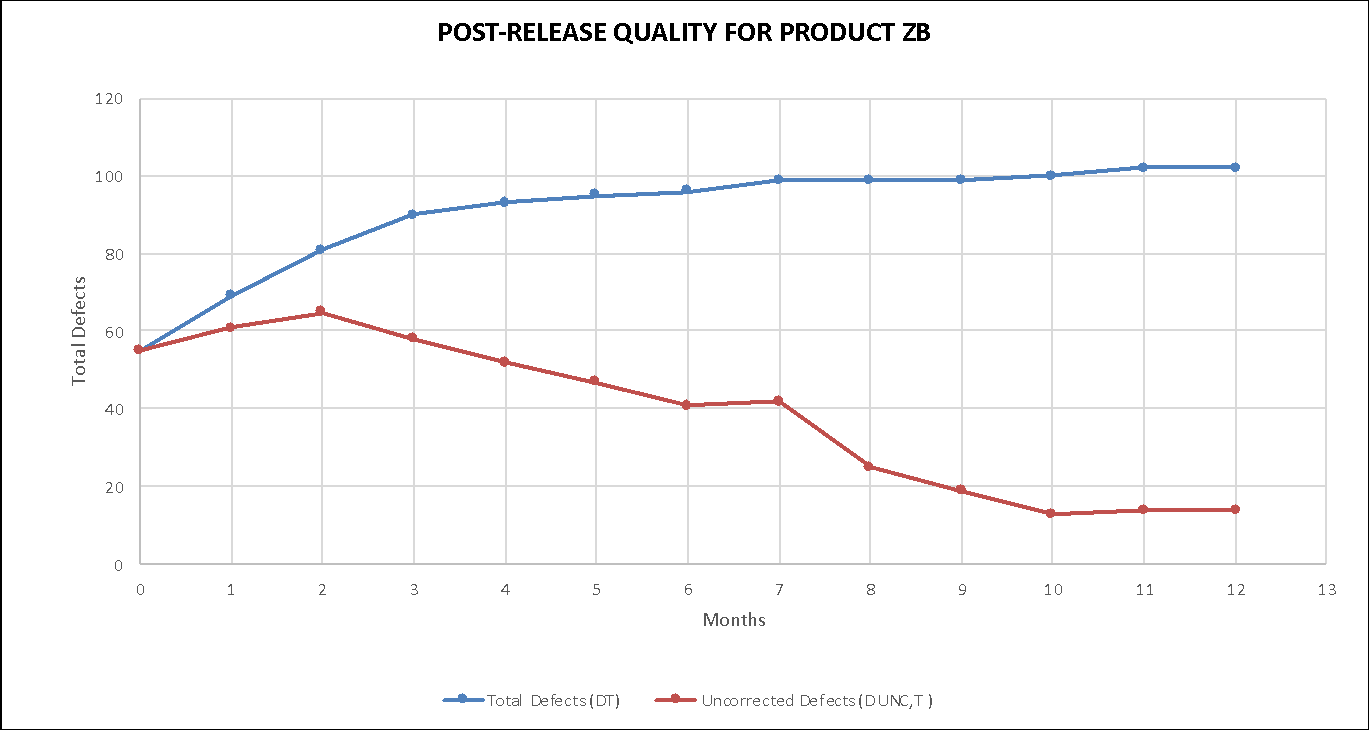
**2.1.a. Post-Release Quality for One Product**

**i. Overview**

Post-release quality measures the known defects of the released product and the total growth of the product after its release. ***Post-release quality*** is defined as the number of defects in the product or collection each month after release and is measured for an individual product or for any collection of products (such as all those written in Java). This is recorded each month for twelve months after a product is released and is displayed as a line chart using two lines: *total defects* and *total uncorrected defects*.

In the ‘Post-Release Quality’ for product ZB, the graph shows the total number of defects and the number of uncorrected defects for a period of 12 months after its release. A line chart is used to represent this measure and has two lines: total defects and total uncorrected defects. The x-axis shows the number of months whose range is from 0-12 (13 months). The y-axis shows the total defects.

**ii. Sample Graph**

The graph below shows Post Release Quality for one year of Product ZB.

**iii. Analysis and Discussion**

1. **General Discussion**

* Two lines are shown: Both DT (blue) and DUNC,T (red) are plotted monthly on a line chart for the first release of product ZB. The horizontal axis is number of months since product release and the vertical axis shows defect count. The chart shows one year of data.
* The DT line is monotonic – it will tend to grow slowly, but to flatten out over time, as the product’s defects are found. When the line flattens, it is approximately equal to the total number of defects in the product.
* The DUNC,T line should decrease over time, as defects are found and corrected. However, it may increase in any particular month if a lot of new defects are found in that month.

1. **Specific Discussion**

The DT line grows slowly and flattens out after the sixth month. The DUNC,T line increases for the first three months as a lot of new defects are found. After that, it decreases over time as the defects are corrected. The team has worked satisfactorily to correct all the defects found. There are 14 uncorrected defects for product ZB 12 months after its release.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.a.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.a.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the total defects are computed each month by adding the latest month’s defects to the previous total. * The uncorrected defects are computed each month by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total. |

|  |
| --- |
| **2.1.a.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Defects** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 1 - Total Defects**   * **Uncorrected Defects** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 2 - Uncorrected Defects** |

|  |
| --- |
| **2.1.a.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Post-release Quality for product ZB. * **Axes** – The horizontal axis is number of months since product release and the vertical axis shows defect count. * **Data** – DT (blue) represents total defects at month T using the Equation 1 in 2.1.a.iv.3. DUNC,T (red) represents the uncorrected defects at month T using Equation 2 in 2.1.a.iv.3. |

**2.1.b Post Release Quality Average for All Products, Normalized by Size**

**i. Overview**

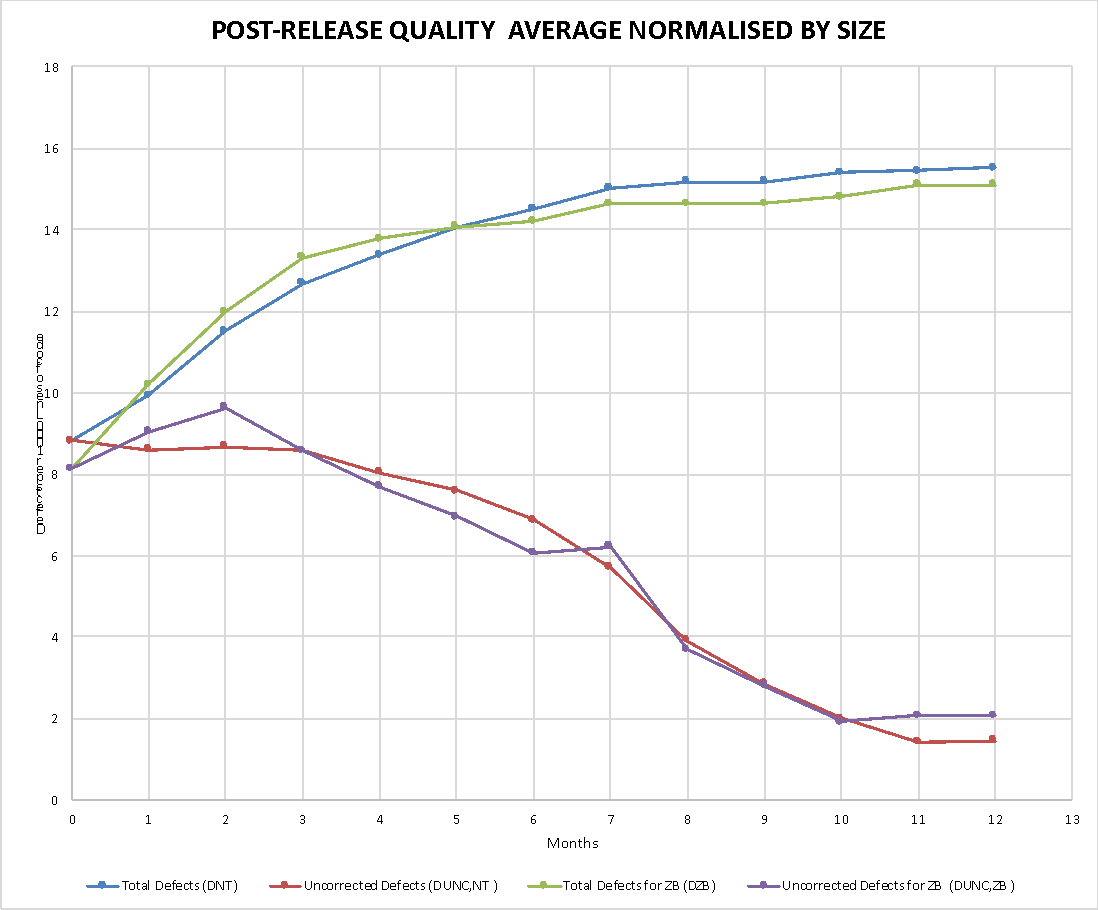
Post-release quality average for all products, normalized by size measures the average of total defects and uncorrected defects of all products per 1000 lines of code for all the months since product release.

***Post-release quality average normalized by size*** is measured by dividing each product’s defect counts by the product’s size (and then multiply by 1000) to produce “defects per 1000 lines of code” for every month. Then the average is calculated for both total defect and uncorrected defects of a product each month for twelve months after a product is released and is displayed as a line chart using two lines: *total defects* and *total uncorrected defects*. The above method is followed to calculate the Post release quality average for an individual product, normalized by size and is compared to the average value generated for all products.

In the ‘Post Release Quality Average for All Products, Normalized by Size’, the graph shows the average of total defects and uncorrected defects of all products per 1000 lines of code for all the months since product release and also the total defects and uncorrected defects for product ZB per 1000 lines of code for all the months since the release of product ZB. A line chart is used to represent this measure and has four lines: Total defects (DT), Uncorrected defects (DUNC,T) , Total defects for product ZB(DZB), Uncorrected defects for product ZB(DUNC,ZB). The x-axis shows the number of months whose range is from 0-12 (13 months). The y-axis shows the defects per 1000 lines of code.

**ii. Sample Graph**

The graph below shows Post Release Quality average for all the products and for product ZB normalized by size for one year.



**iii. Analysis and Discussion**

1. **General Discussion**

* Four lines are shown:

1. DNT (blue) shows the average total defects per 1000 lines of code for all products.
2. DZB (green) shows the total defects per 1000 lines of code for product ZB
3. DUNC,NT (red) shows the average uncorrected defects per 1000 lines of code for all products.
4. DUNC,ZB (purple) shows the average uncorrected defects per 1000 lines of code for product ZB.

* DT and DZB are monotonic. They will tend to grow slowly, but they will flatten out over time, as the product’s defects are found.
* DUNC,T and DUNC,ZB should decrease over time, as defects are found and corrected. However, it may increase if a lot of new defects are found in that particular month.

1. **Specific Discussion**

As seen from the graph above, comparison between the number of defects found in the product ZB and the total number of defects in all the products can be studied. Both DT and DZB increase over time and DZB has marginally higher number of defects than the DT after the fifth month but both flatten out after the eight month. DUNC,T and DUNC,ZB decrease over time and have a significant decrease from the eighth month. It can be seen that at the second and seventh month, the uncorrected defects for ZB is higher than the total uncorrected defects of all the products.At the end of twelve months it can be studied that the number of uncorrected defects for product ZB is marginally higher than the uncorrected defects for all the products, but the team has worked hard and corrected a significant number of defects of both ZB and Total Projects.

Data is normalized when they are not in a uniform scale. In the given data, the total and uncorrected defects are calculated for product whose sizes are different. This places the products in different scales as the number of defects may depend on the size of the product. In order to analyze the defects for all the products we require a uniform scale. To achieve that we normalize the products by their sizes and then compare their defects.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.b.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **Si –** The total number of lines of code for each product.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.b.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the total defects are computed each month by adding the latest month’s defects to the previous total. * The uncorrected defects are computed each month by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total. * The data is normalized by dividing the total defects and uncorrected defects individually by the size of each product and multiplying it by 1000. This gives the total number of defects per 1000 lines of code and total uncorrected defects per 1000 lines of code respectively. |

|  |
| --- |
| **2.1.b.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Defects** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 1 - Total Defects**  *DNT =*   * **Uncorrected Defects** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 2 - Uncorrected Defects**   * **Total Defects for ZB** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 3 - Total Defects for ZB**  *for product ZB*  *DZB =*   * **Uncorrected Defects for ZB** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 4 - Uncorrected Defects for ZB**  *for product ZB* |

|  |
| --- |
| **2.1.b.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Post-release Quality Average for all products, normalized by size. * **Axes** – The horizontal axis is number of months since product release and the vertical axis shows defect count per 1000 lines of code. * **Data** –  1. DNT (blue) shows the average total defects per 1000 lines of code for all products using Equation 1 in 2.1.b.iv.3. 2. DUNC,NT (red) shows the average uncorrected defects per 1000 lines of code for all products Equation 2 in 2.1.b.iv.3. 3. DZB (green) shows the total defects per 1000 lines of code for product ZB using Equation 3 in 2.1.b.iv.3. 4. DUNC,ZB (purple) shows the average uncorrected defects per 1000 lines of code for product ZB using Equation 4 in 2.1.b.iv.3. |

**2.1.c Post Release Quality Average for All Products, by Development Process**

**i. Overview**

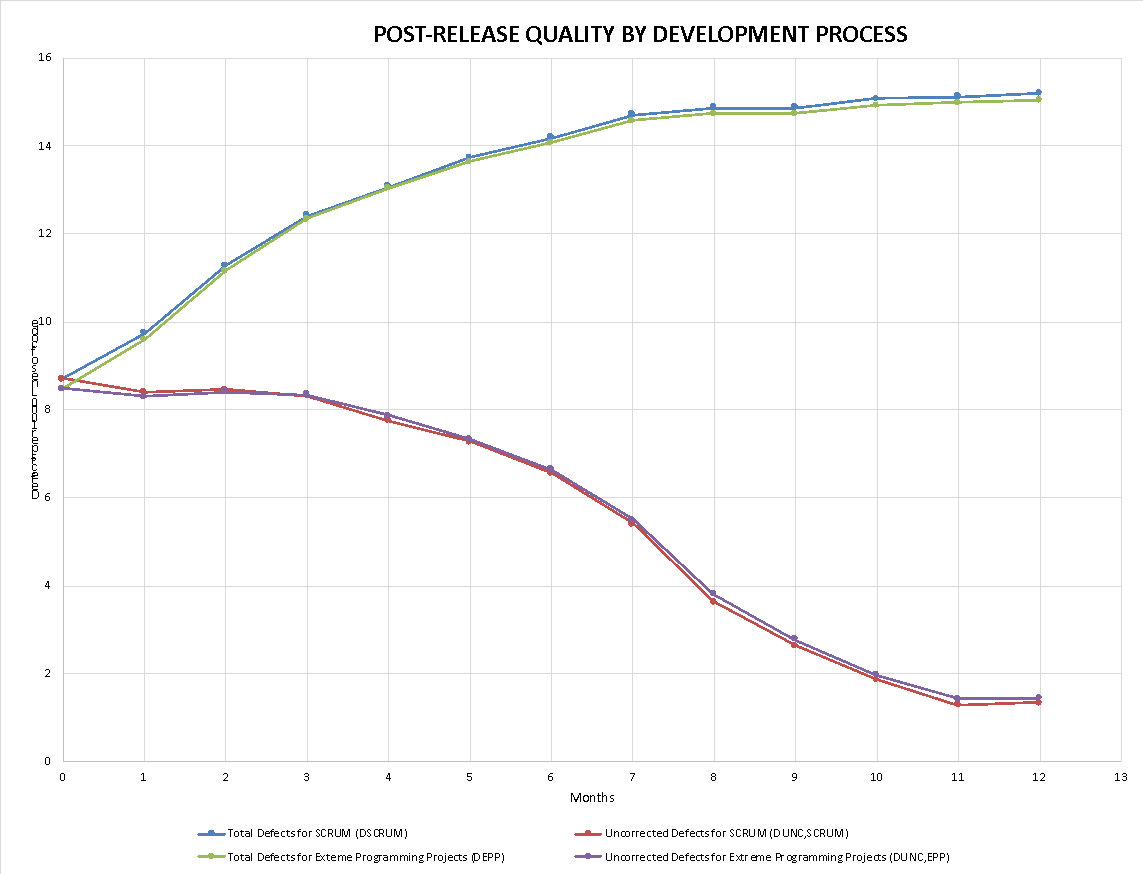
Post-release quality average for all products by Development process, normalized by size measures the average of total defects and uncorrected defects of products categorized based on their development process per 1000 lines of code for all the months since product release.

***Post-release quality by Development Process normalized by size*** is measured by dividing each product’s defect counts by the product’s size (and then multiply by 1000) to produce “defects per 1000 lines of code” for every month. Then the average is calculated for both total defect and uncorrected defects of a product based on their development process(SCRUM or Extreme Programming Projects) each month for twelve months after a product is released and is displayed as a line chart using four lines: *total defects for SCRUM* , *total uncorrected defects for SCRUM, total defects for Extreme Programming Projects* and *total uncorrected defects for Extreme Programming Projects*. The above method is followed to calculate the Post release quality for products based on SCRUM development process, normalized by size and is compared to the Post release quality for products based on Extreme Programming Project development process, normalized by size.

In the ‘Post Release Quality by Development Process, Normalized by Size’, the graph shows the average of total defects and uncorrected defects for both SCRUM and Extreme Programming Projects per 1000 lines of code for all the months since product release. A line chart is used to represent this measure and has four lines: Total defects for SCRUM (DSCRUM), Uncorrected defects for SCRUM(DUNC,SCRUM) , Total defects for Extreme Programming Projects(DEPP), Uncorrected defects for Extreme Programming Projects (DUNC,EPP). The x-axis shows the number of months whose range is from 0-12 (13 months). The y-axis shows the total defects per 1000 lines of code.

**ii. Sample Graph**

The graph below shows Post Release Quality Average for all products by development process.



**iii. Analysis and Discussion**

1. **General Discussion**

* Four lines are shown:

1. DSCRUM (blue) shows the average total defects per 1000 lines of code for all SCRUM products.
2. DEPP (green) shows the total defects per 1000 lines of code for all Extreme Programming Projects.
3. DUNC,SCRUM (red) shows the average uncorrected defects per 1000 lines of code for all SCRUM products.
4. DUNC,EPP (purple) shows the average uncorrected defects per 1000 lines of code for all Extreme Programming Projects.

* DSCRUM and DEPP are monotonic. They will tend to grow slowly, but they will flatten out over time, as the product’s defects are found.
* DUNC,SCRUM and DUNC,EPP should decrease over time, as defects are found and corrected. However, it may increase if a lot of new defects are found in that particular month.

1. **Specific Discussion**

As seen from the graph above, number of defects that have been found in SCRUM products is almost equal to the number of defects that have been found in Extreme Programming Projects. Both DSCRUM and DEPP increase over time and flatten out from the eighth month. DUNC,SCRUM and DUNC,EPP decrease over time and have a significant decrease from the eighth month. Even though SCRUM products have a marginally higher number of defects than extreme programming projects, after the sixth month, the number of uncorrected defects of for both SCRUM and Extreme Programming Projects become almost equal. This depicts that the team has worked hard and corrected a significant number of defects of both SCRUM and Extreme Programming Projects.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.c.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **Si –** The total number of lines of code for each product. * **Dp** – The development process used in different products.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.c.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the total defects by size are computed each month by adding the latest month’s defects to the previous total for each product using SCRUM and Extreme Programming methods respectively and calculating the sum. * The uncorrected defects average is computed each month by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total for each product using SCRUM and Extreme Programming method respectively and calculating the sum. * The data is normalized by dividing the total defects and uncorrected defects individually by the size of each product and multiplying it by 1000. This gives the total number of defects per 1000 lines of code and total uncorrected defects per 1000 lines of code respectively. |

|  |
| --- |
| **2.1.c.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Defects for SCRUM** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 1 - Total Defects for products using SCRUM**  *for products using SCRUM*  *DSCRUM =*   * **Total Defects for Extreme Programing** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 2 - Total Defects for products using Extreme Programming**  *for products Extreme Programming*   * **Uncorrected Defects for SCRUM** (at month T is defined as **Total Defects for SCRUM** minus the cumulative sum of all defects that have been corrected:   **Equation 3 - Uncorrected Defects for products using SCRUM**  *for products using SCRUM*   * **Uncorrected Defects for Extreme Programming** (at month T is defined as **Total Defects for Extreme Programming** minus the cumulative sum of all defects that have been corrected:   **Equation 4 - Uncorrected Defects for products using Extreme Programming**  *for products using Extreme Programming* |

|  |
| --- |
| **2.1.c.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Post Release Quality Average for All Products, by Development Process. * **Axes** – The horizontal axis is number of months since product release and the vertical axis shows defect count per 1000 lines of code. * **Data** –  1. DSCRUM (blue) shows the average total defects per 1000 lines of code for all SCRUM products using Equation 1 in 2.1.c.iv.3. 2. DEPP (green) shows the total defects per 1000 lines of code for all Extreme Programming Projects using Equation 2 in 2.1.c.iv.3. 3. DUNC,SCRUM (red) shows the average uncorrected defects per 1000 lines of code for all SCRUM products using Equation 3 in 2.1.c.iv.3. 4. DUNC,EPP (purple) shows the average uncorrected defects per 1000 lines of code for all Extreme Programming Projects using Equation 4 in 2.1.c.iv.3. |

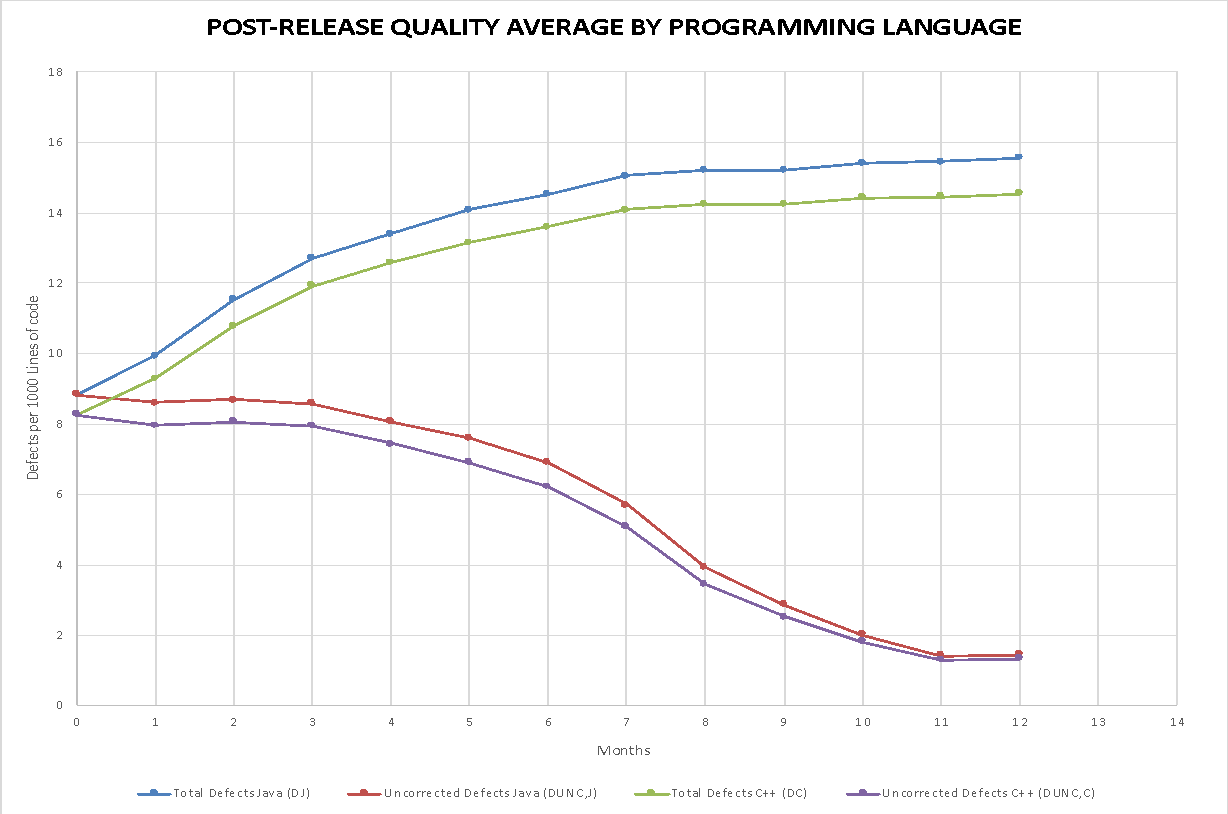
**2.1.d Post Release Quality Average for All Products, by Programming Language**

**i. Overview**

This graph shows the Post- Release Quality for all the products divided by the programming languages used to develop them, i.e. JAVA or C++ and the data is normalized per 1000 lines of code for all the months since product release. The line chart has four lines that represent the total number of defects detected and the total number of uncorrected defects of all products when JAVA and C++ were used. A line chart is used to represent this measure and has four lines: Total defects for JAVA (DJAVA), Uncorrected defects for JAVA (DUNC, JAVA) , Total defects for C++(DC), Uncorrected defects for C++ (DUNC, C). The x-axis shows the number of months whose range is from 0-12 (13 months). The y-axis shows the total defects per 1000 lines of code.

**ii. Sample Graph**

The graph below shows Post Release Quality Average for all products by programming language.



**iii. Analysis and Discussion**

1. **General Discussion**

* Four lines are shown:

1. DJ (blue) shows the total number of defects per 1000 lines of code of products that were developed using JAVA.
2. DC (green) shows the total number of defects per 1000 lines of code of products that were developed using C++.
3. DUNC,J (red) shows the total number of uncorrected defects per 1000 lines of code of products that were developed using JAVA.
4. DUNC,C (purple) shows the total number of uncorrected defects per 1000 lines of code of products that were developed using C++.

* DJ and DC are monotonic. They will tend to grow slowly, but to flatten out over time, as the product’s defects are found.
* DUNC,J and DUNC,C should decrease over time, as defects are found and corrected. However, it may increase in any particular month if a lot of new defects are found in that month.

1. **Specific Discussion**

As seen from the graph above, a greater number of defects have been found in products developed with JAVA. Both DJ and DC increase over time and flatten out from the tenth month. DUNC,J and DUNC,C decrease over time and have a significant decrease from the ninth month. Even though products developed with JAVA have a higher number of defects, after the ninth month, the number of uncorrected defects of products developed with JAVA and C++ almost become equal. This shows that the team worked hard and corrected a significant number of defects of products developed by both JAVA and C++.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.d.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **Si** – The total number of lines of code for each product. * **Li** – The programming language used for development.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.d.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the total defects by size are computed each month by adding the latest month’s defects to the previous total for each product using JAVA and C++ methods respectively and calculating the sum. * The uncorrected defects average is computed each month by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total for each product using JAVA and C++ method respectively and calculating the sum. * The data is normalized by dividing the total defects and uncorrected defects individually by the size of each product and multiplying it by 1000. This gives the total number of defects per 1000 lines of code and total uncorrected defects per 1000 lines of code respectively. |

|  |
| --- |
| **2.1.d.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Defects for JAVA** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 1 - Total Defects for products using JAVA**  *for products using JAVA*  *DJ =*   * **Total Defects for C++** at month T (is defined as the cumulative sum of all defects known at month T. This measure is computed every month, for each product. This includes both defects reported by the customer (post-release defects) and defects known to be in the product at release time (pre-release defects):   **Equation 2 - Total Defects for products using C++**  *for products using C++*   * **Uncorrected Defects for JAVA** (at month T is defined as **Total Defects for JAVA** minus the cumulative sum of all defects that have been corrected:   **Equation 3 - Uncorrected Defects for products using JAVA**  *for products using JAVA*   * **Uncorrected Defects for C++** (at month T is defined as **Total Defects for JAVA** minus the cumulative sum of all defects that have been corrected:   **Equation 4 - Uncorrected Defects for products using C++**  *for products using JAVA* |

|  |
| --- |
| **2.1.d.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Post Release Quality Average for All Products, by Programming Language. * **Axes** – The horizontal axis is number of months since product release and the vertical axis shows defect count per 1000 lines of code. * **Data** –  1. DJ (blue) shows the total number of defects per 1000 lines of code of products that were developed using JAVA using Equation 1 in 2.1.d.iv.3. 2. DC (green) shows the total number of defects per 1000 lines of code of products that were developed using C++ using Equation 2 in 2.1.d.iv.3. 3. DUNC,J (red) shows the total number of uncorrected defects per 1000 lines of code of products that were developed using JAVA using Equation 3 in 2.1.d.iv.3. 4. DUNC,C (purple) shows the total number of uncorrected defects per 1000 lines of code of products that were developed using C++ using Equation 4 in 2.1.d.iv.3. |

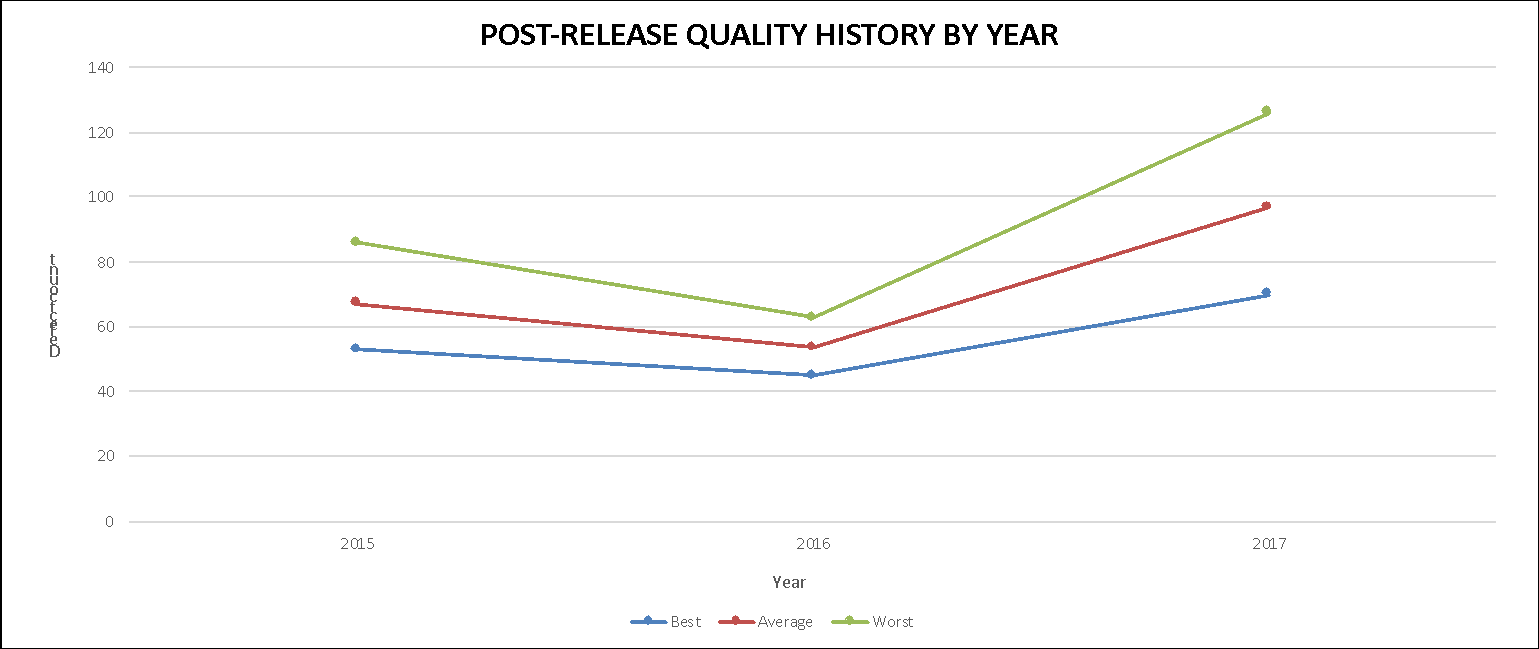
**2.1.e Post Release Quality History**

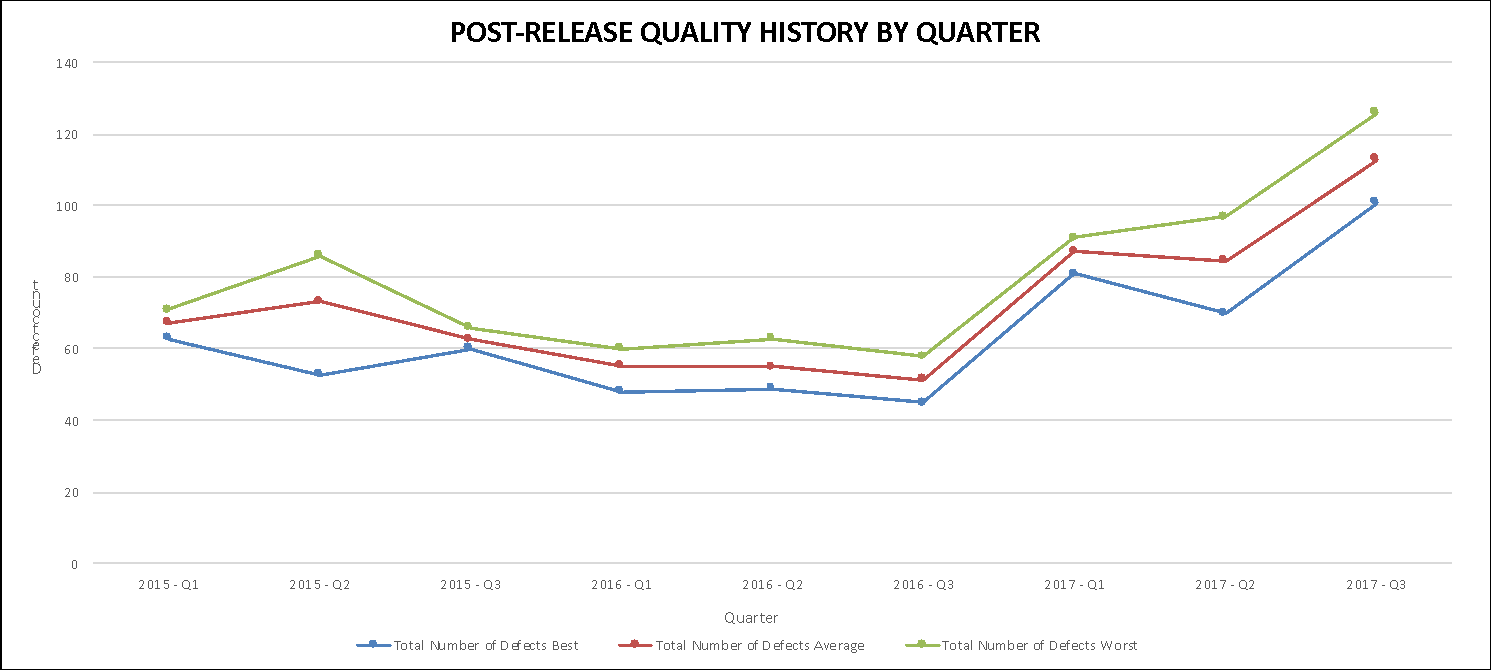
**i. Overview**

Post-release quality history indicates the total number of defects found for all product releases during their first 12 months of use. It indicates how good are the products that were shipped over several years in terms of defect rates and how have things changed over time. This is measured for all products that were released during a particular time period. It is a “lagging” indicator that can only be computed after a release has been out for 12 months. Three values are computed: best case defect count, worst case defect count, and average case defect count for all products released during a given time period. The x-axis shows the time period and the y-axis shows the total defect count.

The first graph which is a line chart shows the total number of defects found for all product releases during their first 12 months of use by year, for three years (2015, 2016, 2017). The second graph which is a line chart shows the total number of defects found for all product releases during their first 12 months of use by quarter, where each quarter represents anywhere from two to four product releases.

**ii. Sample Graph**

The graph below shows Post Release Quality History by year.



The graph below shows Post Release Quality History by quarter.

**iii. Analysis and Discussion**

1. **General Discussion**

* The following three lines are shown for each graph:

1. Best case defect count (blue) shows the minimum number of defects within a given time period.
2. Average case defect count (red) shows the average number of defects within a given time period.
3. Worst case defect count (green) shows the maximum number of defects within a given time period.

* Each year is divided into 3 quarters (Q1, Q2, Q3). Q1 and Q2 have three products releases and Q3 has four product releases.

1. **Specific Discussion**

* Post Release Quality History by Year:

This graph combines defect data of products for three years: 2015, 2016, and 2017. As seen from the graph, 2016 has the least number of defects and 2017 has the largest number of defects. This shows us that more defective products were developed during the year 2017.

By collecting data on a yearly basis helps observe trends of the released products on a long-term basis which helps improve practices for better efficiency and accuracy. One main drawback in this method is that the severity of the defect is not known and thus the quality of the product cannot be measured accurately.

* Post Release Quality History by Quarter:

The given data for each year is split into three quarters: Q1, Q2, and Q3. From the graph above, we can see that the number of defects decreases from 2015 to 2016 and significantly increases from 2016 to 2017. The products developed in 2017 were highly defective.

By collecting data on a quarterly basis, computation is faster, data is easier to model, and it is easier to identify changes occurring in trends. One main drawback in this method is that the severity of the defect is not known and thus the quality of the product cannot be measured accurately.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.e.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.e.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| 1. Post-Release Quality History by Year: 2. Total defects are calculated for 12 months for each year 2015, 2016, and 2017 by adding the latest month’s defects to the previous total. 3. Best case defect count is the minimum number of defects for the given year. 4. Average case defect count is the average number of defects for the given year. 5. Worst case defect count is the maximum number of defects for the given year.      1. Post-Release Quality History by Quarter: 2. Each year, i.e. 2015, 2016, and 2017 is divided into three quarters: Q1, Q2, and Q3. 3. Q1 and Q2 have 3 product releases. Q3 has 4 product releases. 4. Best case defect count is the minimum number of defects for the given quarter. 5. Average case defect count is the average number of defects for the given quarter. 6. Worst case defect count is the maximum number of defects for the given quarter. |

|  |
| --- |
| **2.1.e.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Defects** at time T (is defined as the cumulative sum of all defects known at time T. This measure is obtained by adding the sum of all defects of a product in 1 year.   **Equation 1 - Total Defects** |

|  |
| --- |
| **2.1.e.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Post Release Quality History by Year, and Post Release Quality History by Quarter. * **Axes** –  1. Post Release Quality History by Year: The horizontal axis is the three years, 2015, 2016, and 2017 and the vertical axis shows defect count. 2. Post Release Quality History by Quarter: The horizontal axis is the nine quarters (2015-Q1, 2015-Q2, 2015-Q3, 2016-Q1, 2016-Q2, 2016-Q3, 2017-Q1, 2017-Q2, 2017-Q3) and the vertical axis shows defect count.  * **Data** –  1. Total defects for each product are calculated for a year using Equation 1 in 2.1.e.iv.3. 2. Best case defect count (blue) shows the minimum number of defects within a given time period. 3. Average case defect count (red) shows the average number of defects within a given time period. 4. Worst case defect count (green) shows the maximum number of defects within a given time period. |

**2.1.f Employee Productivity Measure**

**i. Overview**

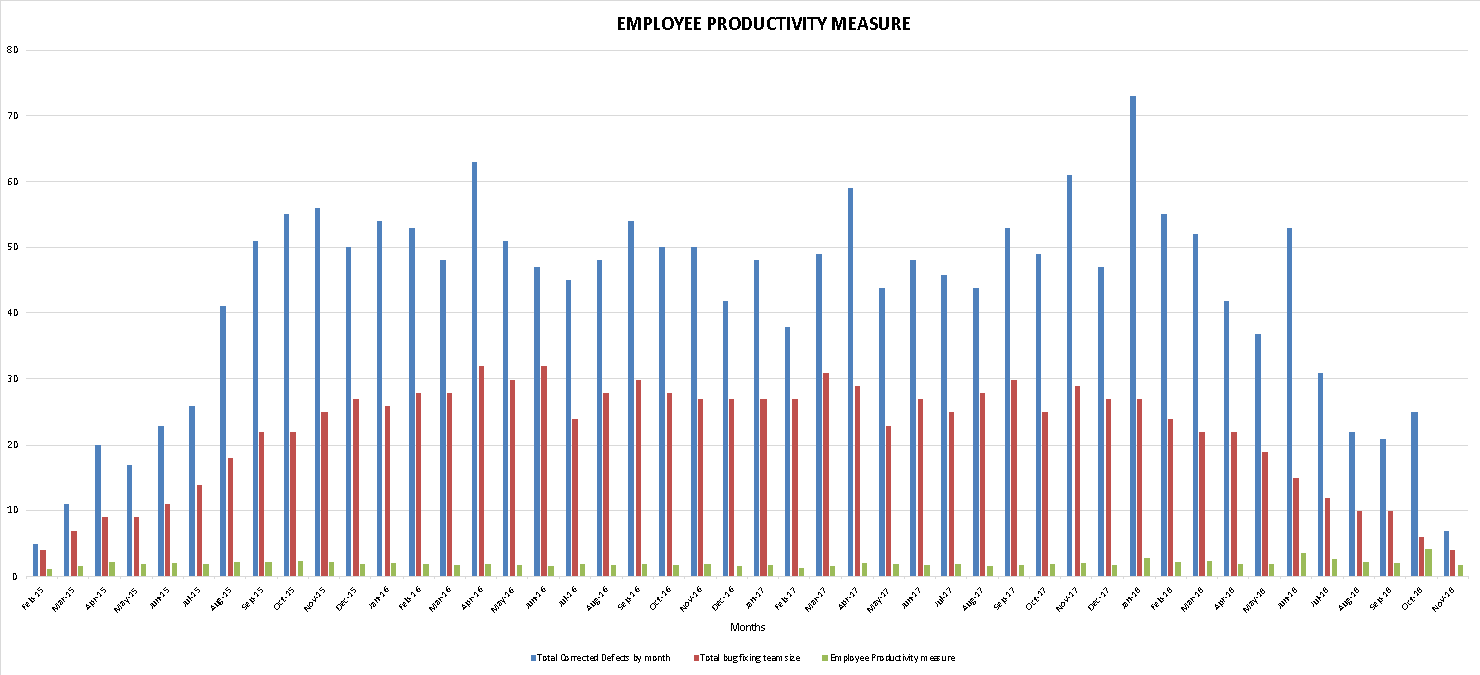
Employee Productivity Measure can be defined as the number of defects corrected by the team of employees every month. This information can be used to improve the productivity and efficiency of the company for defect correction. This measures the employee productivity relative to the team size and total defects corrected in a month.

Employee Productivity Measure can be calculated by defining the team size. For the purposes of this report, the team size allocated for every product is assumed. The total number of corrected defects is calculated every month and is divided by the corresponding total team size to obtain the employee productivity measure.

A bar chart is used to represent this measure. Each month has three bars: the total corrected defects (DC,i), total bug fixing team size (Ti), and employee productivity measure (EPM). The x-axis is the number of months for four years and y-axis shows count.

**ii. Sample Graph**

The graph below shows employee productivity measure for four years.



**iii. Analysis and Discussion**

1. **General Discussion**

* Three bars are shown:

1. DC,i (blue) shows the total corrected defects for all the products in a month for four years.
2. Ti (red) shows the total bug fixing team size.
3. EPM (green) shows the employee productivity measure.

* The employee productivity measure is corelated to the number of corrected defects occurring every month and the size of the team involved.

1. **Specific Discussion**

The graph shows that the employee productivity remains consistent for the considered four years irrespective of the fluctuation in the number of corrected defects. This is achieved by the differing team sizes that were allocated based on the defects to be corrected. This helps us understand the amount of resources that needs to be assigned to a product according the number of defects present in a month. It can also be noted that a higher team size does not always produce a higher employee productivity. The correct amount of resources needs to assigned to achieve maximum efficiency from the employees.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.1.f.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **DC** – The total number of corrected defects in a month. * **Ti** – The team size assigned to the product. (Assumption).   The above data are collected separately for each software product. |

|  |
| --- |
| **2.1.f.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the total corrected defects are computed each month from the ‘C’ in ‘New Corr’ column in the given data. * The employee productivity measure is calculated by dividing the total corrected defects by the team size every month for four years. |

|  |
| --- |
| **2.1.f.iv.3 Compound Measures (Metrics Computed)** |
| * **Total Corrected Defects** at time T (is defined as the cumulative sum of all corrected defects known at time T.   **Equation 1 - Total Corrected Defects**   * **Employee Productivity** (EPM)at time T is computed by dividing Total Corrected Defects (DC) by Team Size (Ti).     **Equation 2 – Employee Productivity Measure** |

|  |
| --- |
| **2.1.f.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A bar graph is used to display the Employee Productivity Measure. * **Axes** – The horizontal axis is number of months and the vertical axis shows count. * **Data** –  1. DC,i (blue) shows the total corrected defects for all the products in a month for four years using Equation 1 in 2.1.f.iv.3. 2. Ti (red) shows the total bug fixing team size. 3. EPM (green) shows the employee productivity measure using Equation 2 in 2.1.f.iv.3. |

* 1. **Current Quality**

**2.2.a. Current Quality Total**

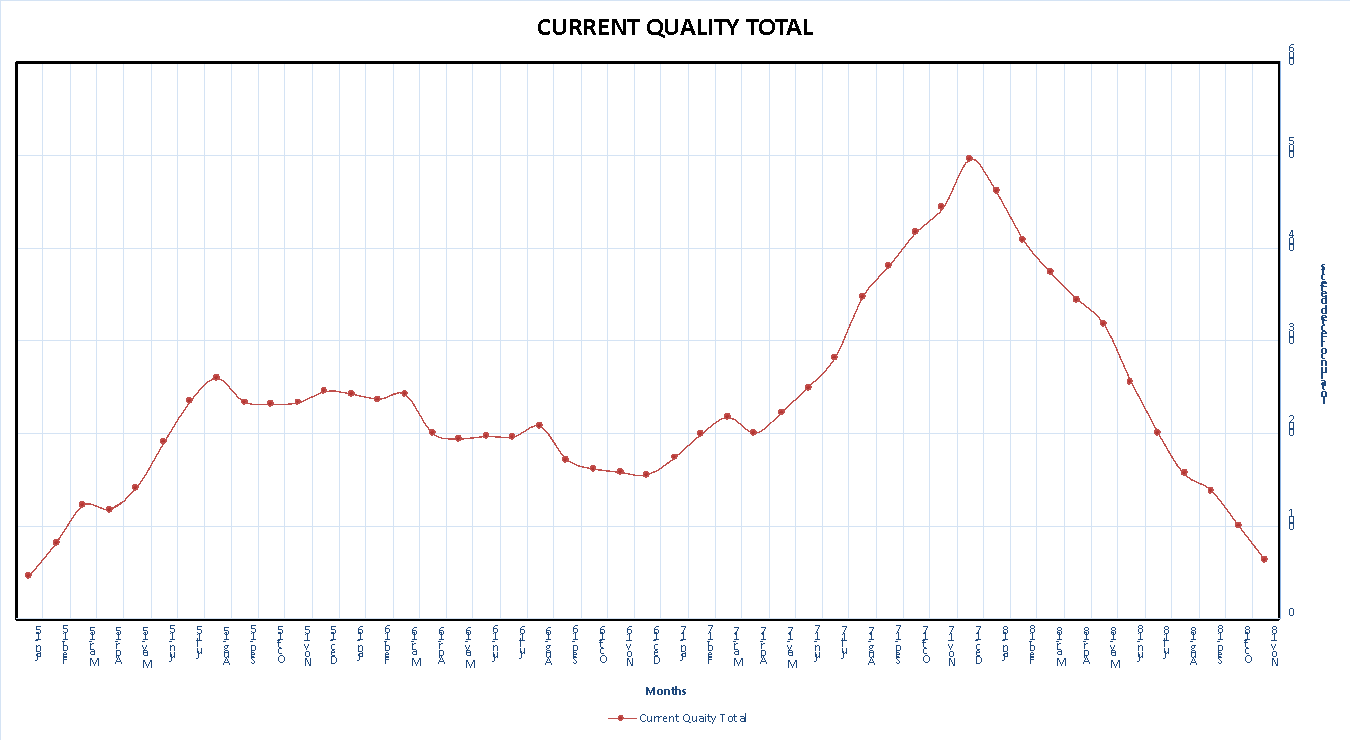
**i. Overview**

Current quality is measured for all active products and is defined as the total uncorrected defects for all active products (all products that have been released and are still active – i.e., are still within their first 12 months). This is recorded by month, for 3 years, using all active products each month and is displayed as a line chart using one line: *total uncorrected defects*.

Current quality total is graphed by month, for 3 years, using all active products each month. The x-axis shows the number of months for 3 years. The y-axis shows the total uncorrected defects.

**ii. Sample Graph**

The graph below shows Current Quality Total for all active products for 3 years.



**iii. Analysis and Discussion**

1. **General Discussion**

* One line is shown: DUNC,T (red) is plotted monthly for three years on a line chart for the total uncorrected defects for all active products. The horizontal axis is number of months for three years and the vertical axis shows total uncorrected defect count.
* The DUNC,T line should decrease over time, as defects are found and corrected. However, it may increase if a lot of new defects are found in a particular month.

1. **Specific Discussion**

The DUNC,T line increases for the first three months as a lot of new defects are found. After that, it is maintained in the same level over time as the defects are corrected. The number of uncorrected defects is at peak during the month of December 2017. The team has worked satisfactorily to correct all the defects found. There are 63 uncorrected defects for the active products at the end of 3 years.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.2.a.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.2.a.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the uncorrected defects for active products are computed each month (till their first 12 months) by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total. * The uncorrected defects of a product is subtracted from the uncorrected defects total once the product exceeds its first 12 months (no longer an active product). |

|  |
| --- |
| **2.2.a.iv.3 Compound Measures (Metrics Computed)** |
| * **Uncorrected Defects** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 1 - Uncorrected Defects**  *for active products* |

|  |
| --- |
| **2.2.a.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Current Quality Total for all active products. * **Axes** – The horizontal axis is number of months (for 3 years) since product release and the vertical axis shows total uncorrected defects count. * **Data** –DUNC,T (red) represents the uncorrected defects at month T using Equation 1 in 2.2.a.iv.3. |

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

**i. Overview**

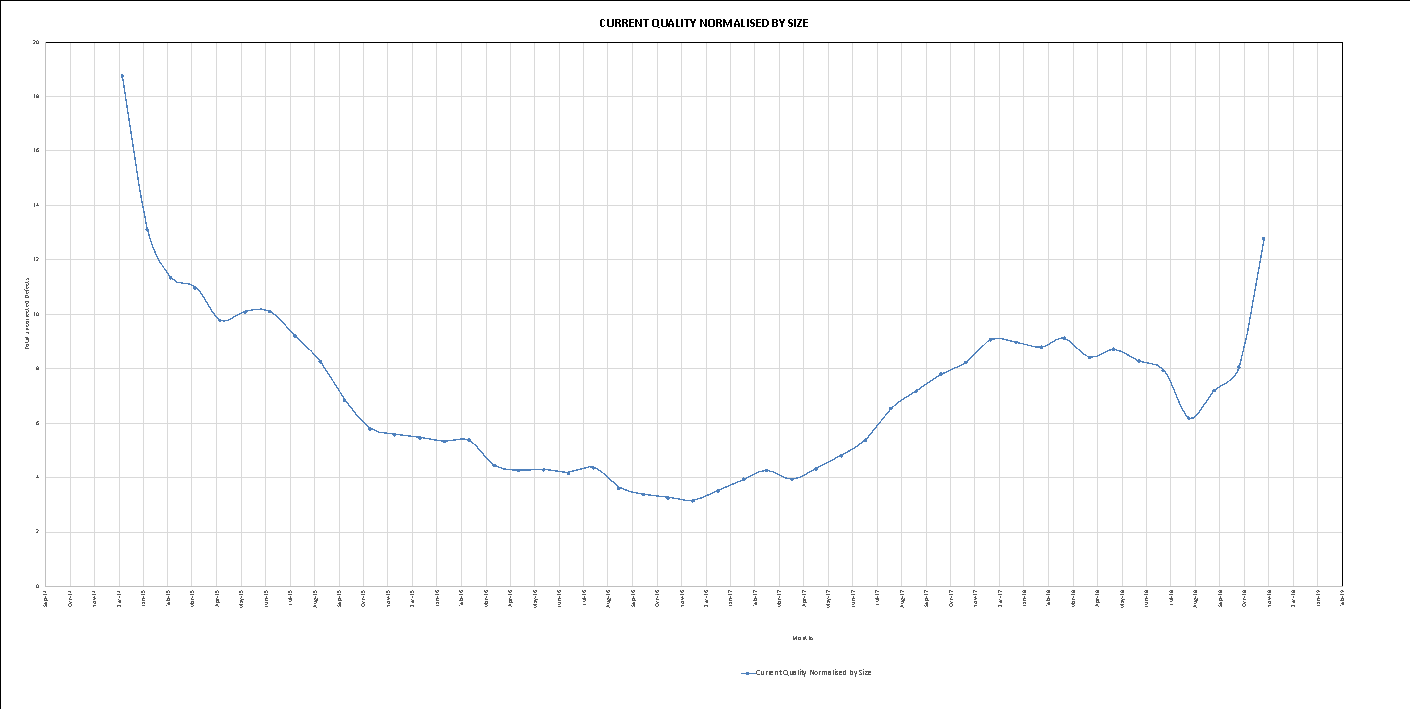
Current quality Total Normalized by size measure the total uncorrected defects for all active products (all products that have been released and are still active – i.e., are still within their first 12 months) normalized by size per 1000 lines of code for all the months for three years (2015-2017).

This is recorded by month, for 3 years, by dividing each month’s uncorrected defect count by the all active product’s size (and then multiply by 1000) to produce “defects per 1000 lines of code” for every month. This is depicted as a line chart. Using one line: *total uncorrected defects.*

The x-axis shows the number of months for 3 years. The y-axis shows the total uncorrected defects.

**ii. Sample Graph**

The graph below shows Current Quality Total Normalized by size for all active products for 3 years.



**iii. Analysis and Discussion**

1. **General Discussion**

* One line is shown: DUNC,T (blue) is plotted monthly for three years on a line chart for the total uncorrected defects for all active products normalized by size. The horizontal axis is number of months for three years and the vertical axis shows total uncorrected defect count.
* The DUNC,T line should decreases as defects are found and corrected. However, it may increase if a lot of new defects are found in a particular month.

1. **Specific Discussion**

The DUNC,T line is at its peak during the first month(Jan-2015) and it gradually decreases over time till Apr-2017. After that we can find that the DUNC,T line fluctuates and reaches a level of 12.774 at the end of 3 years.

The team has done a fairly good job over the period of 3 years but a satisfactory job during the end of the third year as we can see a significant increase in the uncorrected defects in the month Nov -2018.

Data is normalized when they are not in a uniform scale. In the given data, the uncorrected defects are calculated for product whose sizes are different. This places the products in different scales as the number of defects may depend on the size of the product. To analyze the defects for all the products we require a uniform scale. To achieve that we normalize the products by their sizes and then compare their uncorrected defects.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.2.b.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **Si** – The total number of lines of code for each product.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.2.b.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the uncorrected defects for active products are computed each month (till their first 12 months) by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total. * The uncorrected defects for active products, for 3 years, is normalized by dividing each month’s uncorrected defect count by the all active product’s size and then multiply by 1000. |

|  |
| --- |
| **2.2.b.iv.3 Compound Measures (Metrics Computed)** |
| * **Uncorrected Defects normalized by size** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 1 - Uncorrected Defects** |

|  |
| --- |
| **2.2.b.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Current Quality Total Normalized by size. * **Axes** – The horizontal axis is number of months (for 3 years) since product release and the vertical axis shows total uncorrected defect count. * **Data** –DUNC,T (blue) represents the uncorrected defects at month T using Equation 1 in 2.2.b.iv.3. |

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

**i. Overview**

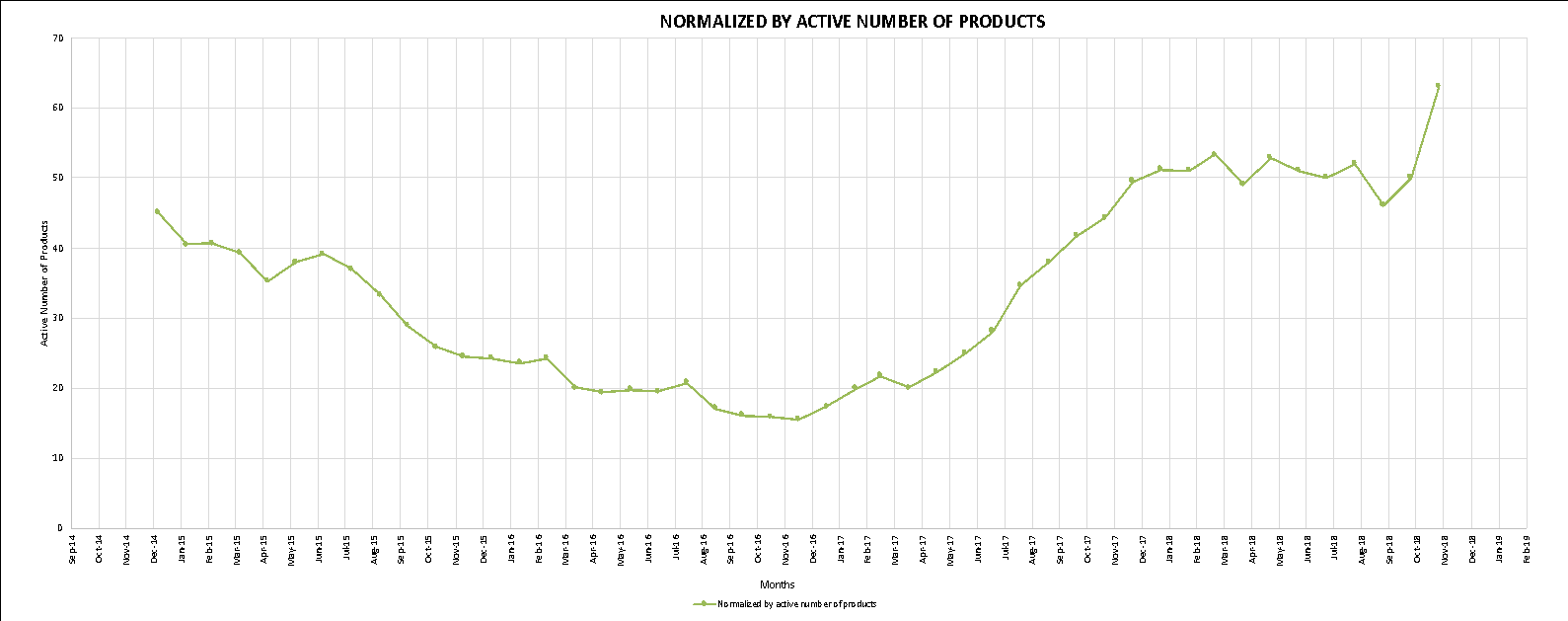
Current quality Total Normalized by Number of Products measure the total uncorrected defects for all active products (all products that have been released and are still active – i.e., are still within their first 12 months) normalized by the number of active products for all the months for three years (2015-2017).

This is recorded by month, for 3 years, by dividing each month’s uncorrected defect count by the number of active products in that month. This is depicted as a line chart using one line: *total uncorrected defects.*

Current quality total normalized by number of products is graphed by month, for 3 years, using the uncorrected defects of all active products each month which are normalized by number of products. The x-axis shows the number of months for 3 years. The y-axis shows the total uncorrected defects.

**ii. Sample Graph**

The graph below shows Current Quality Total Normalized by Number of Products for all active products for 3 years.



**iii. Analysis and Discussion**

1. **General Discussion**

* One line is shown: DUNC,T (green) is plotted monthly for three years on a line chart for the total uncorrected defects for all active products normalized by number of products. The horizontal axis is number of months for three years and the vertical axis shows total uncorrected defect count.
* The DUNC,T line decreases as defects are found and corrected. However, it may increase if a lot of new defects are found in a particular month.

1. **Specific Discussion**

The DUNC,T line increases for the first three months as a lot of new defects are found. After that, it is maintained in the same level over time as the defects are corrected. The number of uncorrected defects is at peak during the month of December 2017. The team has worked satisfactorily to correct all the defects found. There are 220 uncorrected defects for the active products at the end of 3 years.

**iv. Procedure Used to Collect and Refine Data and Produce Graph**

|  |
| --- |
| **2.2.c.iv.1 Data Collection: Base Measures Collected** |
| The data required for this measure are:   * **DPRE** – The number of known defects at the time of product release. Collected at time of product release. * **DRPT, i** – The number of defects reported in the ***ith*** customer failure report. Collected at the beginning of each month. * **N** – The total number of customer failure reports (total number of months). This is normally 12 for each product. * **DC,i** – The number of defects corrected in month ***i***, reported monthly by engineering staff. * **NAC** – The number of active products.   The above data are collected separately for each software product. |

|  |
| --- |
| **2.2.c.iv.2 Data Refinement (manipulations, extractions, sorting, etc.)** |
| * For this graph, the uncorrected defects for active products are computed each month (till their first 12 months) by subtracting the number of defects corrected in the latest month from the previous uncorrected defects total. * The uncorrected defects for active products, for 3 years, is normalized by dividing each month’s uncorrected defect count by the number of active products. |

|  |
| --- |
| **2.2.c.iv.3 Compound Measures (Metrics Computed)** |
| * **Uncorrected Defects normalized by active products** (at month T is defined as **Total Defects** minus the cumulative sum of all defects that have been corrected:   **Equation 1 - Uncorrected Defects** |

|  |
| --- |
| **2.2.c.iv.4 How to Generate the Graph** |
| * **Type of Graph** – A line graph is used to display the Current Quality Total normalized by number of products. * **Axes** – The horizontal axis is number of months (for 3 years) since product release and the vertical axis shows total uncorrected defect count. * **Data** –DUNC,T (green) represents the uncorrected defects at month T using Equation 1 in 2.2.c.iv.3. |