Mean Time Between Failure for Predictive Maintenance Using Hadoop and PowerBI

Wiranto Herry Utomo
Information Technology Department, President University wiranto.herry@president.ac.id

Farah Yulianti
Information Technology Department, President University
farah.yulianti@student.president.ac.id

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 4 June 2021

Abstract

The use of Mean Time Between Failure in Predictive Maintenance is increasing and is in line with the number of industries that are turning to industry 4.0. Previously, predictive maintenance is being done by analyzing each machine individually and manually calculate them. This caused the predictive maintenance to become somewhat complex and a long process while it should not be. Big data helps to organize the data needed to calculate mean time between failure efficiently and PowerBI helps to visualize and analyze said data. We use data from several machines which record their runtime, downtime, and the type of downtime to get the mean time between failure. Contrary to the majority of existing implementations that mostly use complex data to schedule predictive maintenance, Our findings find that simple data is sufficient as long as it is processed in an organized environment such as using big data and visualized clearly and well using visualization applications like PowerBI.

Keyword: *MTBF, Big Data, PowerBI, Visualization*

Introduction

The increasing number of industries that have begun to switch to industry 4.0 causes a lot of data to be generated and must be processed. These data need a suitable environment so that it can be easily processed according to needs. One of the results of data processing in the industrial sector is for machine maintenance. There are many ways that have been studied before to get the prediction time in maintenance to find out when the machine will break down, one of which is the most frequently used is Mean Time Between Failure [1].

The use of Mean Time Between Failure to find out predictions of machine failure has begun to be used in the industrial world, especially the manufacturing industry. However, most of the data used cannot be processed as needed because they are not in accordance with the proper database environment.

The use of Hadoop as a big data environment to help extract and process large amounts of data and the use of PowerBI for data visualization can be an environment suitable for use in the manufacturing industry.

Maintenance data that is streamed to NoSQL in the Hadoop environment can accelerate the analysis of the Mean Time between Failure calculation for predictive maintenance. Then the dataset generated from Hado op Service can be analyzed, processed, and visualized with PowerBI to be shown to the user as a form of report.

This paper offers a comprehensive review on how to use Hadoop environment and PowerBI for a manuf acture industry in terms of finding the predictive maintenance by analyzing the Mean Time Between Failure.

Literature Review

1. Mean Time Between Failure

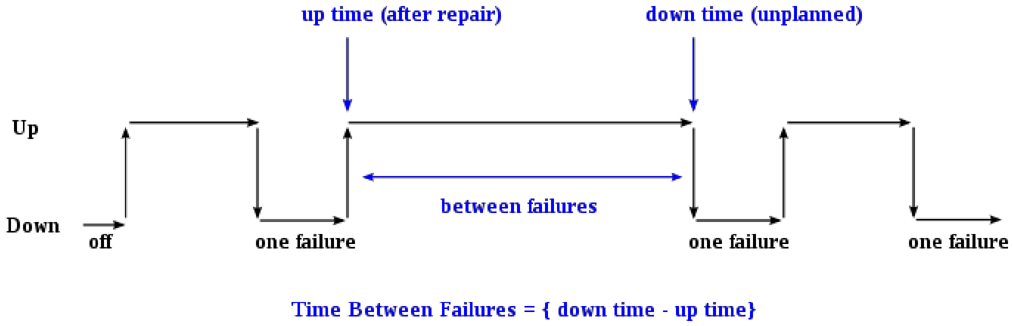


Figure 2.1 Meantime Between Failure

MTBF is a maintenance management that measures the average time lapsed between breakdowns of a system. The Mean Time Between Failures (MTBF) of a mechanical/electrical device is the expected time between one failure and the next during normal operation hours. MTBF is taken into priority when creating Predictive Maintenance. MTBF data provide the means to determine the most cost effective time to check every machine rather than waiting until a major break occurs [2].

Using MTBF calculation and predict when is a machine going into failure and can prevent that from happening. As previous studies and implementation stated, the commonly used MTBF formula is to divides an asset’s total number of runtime hours minus the downtime hours by period then divided by the number of failures that occurred on that asset in that period as shown in figure 2.2.

$$MTBF = \frac{Total\ Working\ Time\ -\ Total\ Breakdown\ Time}{Number\ of\ Breakdowns}$$

Figure 2.2 Meantime Between Failure Formula

Apache Hadoop

Hadoop is an open-source software framework for storing and processing data on commodity hardware clusters. It has a lot of storage for any type of data, a lot of computing power, and it can handle practically infinite concurrent tasks or jobs.

One of Hadoop components is HDFS. HDFS is the pillar of Hadoop that maintains the

distributed file system. It makes it possible to store and replicate data across many servers on the same time.

HDFS got a NameNode and a DataNode. DataNodes are commodities servers where the data in essence is stored. NameNode, on the other hand, contains metadata containing information about the data stored in the different nodes. The application only interacts with a NameNode which communicates with data nodes as required [3].

3. PowerBI

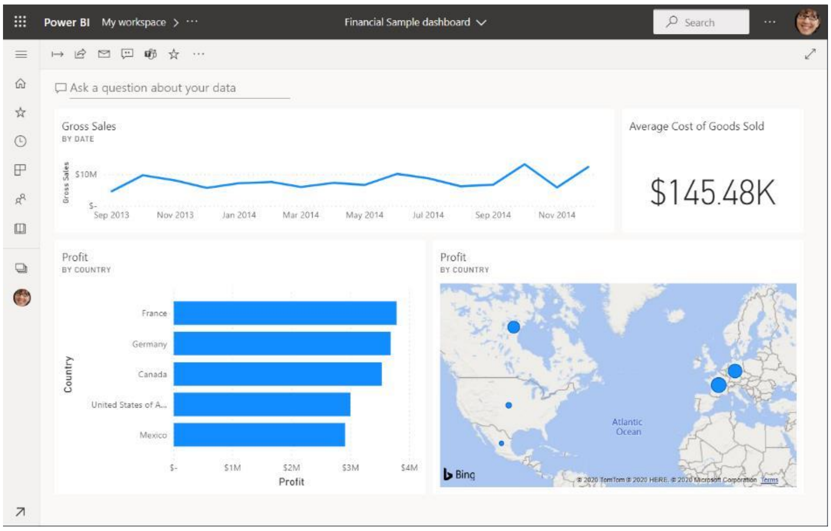


Figure 2.3 PowerBI

Microsoft's PowerBI is a software analytics service. Its goal is to provide interactive visualizations and business intelligence capabilities through an easy -to-use interface that allows end users to generate their own reports and dashboards[4].

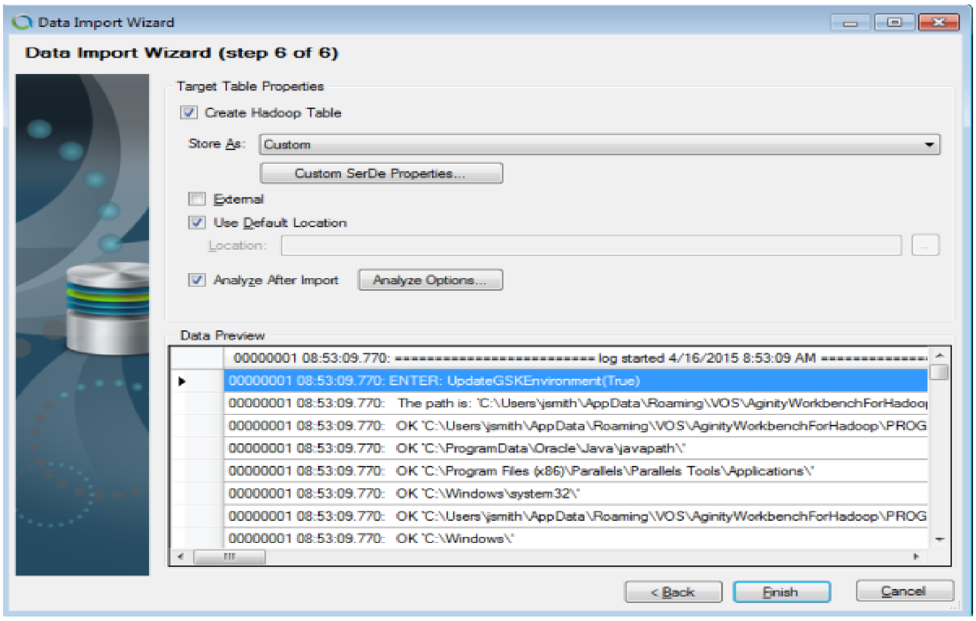
Data from HDFS will be used as a data source for PowerBI. The data acquired with

HDFS will be handled with Power Query for data engineering and data analysis which is an extension of the PowerBI.

Practical Implementation

Setting Up Hadoop

For Experimental purposes, Hadoop single node is intalled on Windows 10. Then, data from csv will be put into HDFS by either using f ilezilla or using command line to upload it to the hadoop or using Aginity Wokbench as shown in Figure 3.1 below.



Data Import Wizard (step 6 of 6)

Target Table Properties

☑ Create Hadoop Table

Store As: Custom

□ External

☑ Use Default Location

Location:

☑ Analyze After Import

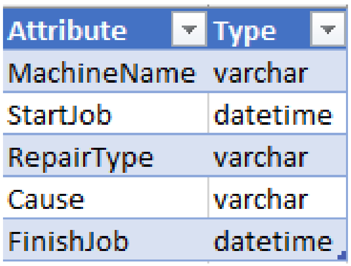
Data Preview

```
00000001 08:53:09.770: ========================== log started 4/16/2015 8:53:09 AM ==============
00000001 08:53:09.770: ENTER: UpdateGSKEnvironment(True)
00000001 08:53:09.770:   The path is: 'C:\Users\jsmith\AppData\Roaming\VOS\AginityWorkbenchForHadoo
00000001 08:53:09.770:  OK 'C:\Users\jsmith\AppData\Roaming\VOS\AginityWorkbenchForHadoop\PROG
00000001 08:53:09.770:  OK 'C:\ProgramData\Oracle\Java\javapath\'
00000001 08:53:09.770:  OK 'C:\Program Files (x86)\Parallels\Parallels Tools\Applications\'
00000001 08:53:09.770:  OK 'C:\Windows\system32\'
00000001 08:53:09.770:  OK 'C:\Users\jsmith\AppData\Roaming\VOS\AginityWorkbenchForHadoop\PROG
00000001 08:53:09.770:  OK 'C:\Windows\'
```

Figure 3.1 Uploading Data File to Hadoop

2. Database Design

The design structure for how the data can be processed for the MTBF calculation is shown below on Figure 3.2.



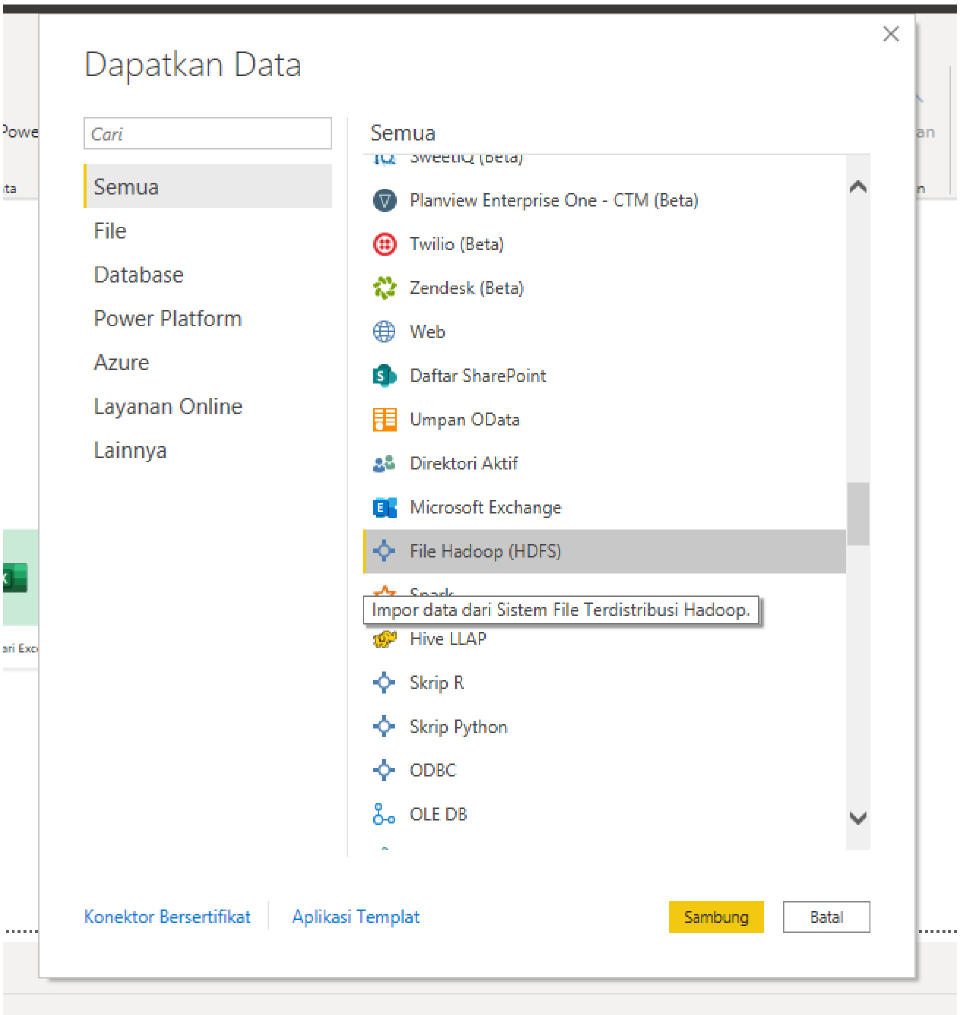
Attribute	Type
MachineName	varchar
StartJob	datetime
RepairType	varchar
Cause	varchar
FinishJob	datetime

Figure 3.2 Database Structure Design

Machine name attribute is to describe the machine name. StartJob is to describe the time started of the repair. RepairType is to describe the type of repair that was conducted. Cause is to describe the cause of repair. FinishJob is to describe the time finished of the repair.

3. PowerBI Data Engineering

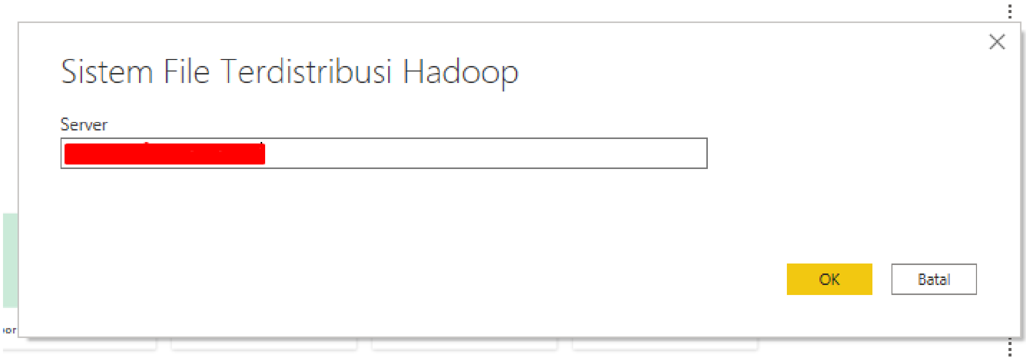
Load the data from HDFS to PowerBI as shown in Figure 3.3 below.



Impor data dari Sistem File Terdistribusi Hadoop.

Figure 3.3 Select Dataset from Hadoop

Then connect to the server where you hosted your Hadoop as shown in Figure 3.4 below, then select the table which is going to be used for reporting.



Sistem File Terdistribusi Hadoop

Server

[redacted]

Figure 3.4 Connect to Hadoop

Once the data is loaded into the PowerBI, then calculate how long the repair hours took place by using this formula below

```
Repair Hours = DATEDIFF(MTBF[StartJob], MTBF[FinishJob],SECOND)/3600
```

The formula above will result in a new column that represents the repair hours it took to complete the repair, make sure to check in the modeling type that the data type is decimal to be able to calculate it again later.

Next, calculate the uptime of the machines by only using the data. The calculation will be

using this formula below.

```
Uptime = VAR next = MINX(FILTER(MTBF,
                        MTBF[Machine        Name]=EARLIER(MTBF[Machine
                        Name]) && MTBF[StartJob]>EARLIER(MTBF[StartJob])
                        && MTBF[RepairType]<>"PM"
                ),MTBF[StartJob])
RETURN  IF([RepairType]="PM",   0,IF(ISBLANK(next),
                DATEDIFF([FinishJob],NOW(),SECON
                D),
                DATEDIFF([FinishJob],next,SECOND)
            )
        )
```

This formula is used to calculate the time between a machine entering "up" state and another machine entering "down" state. However, the current data is not organized in this manner; instead, it is based on gathering repair data. As a result, the first part of the formula, the VAR component, is used to calculate the variable that follows. It is going to be deciding the next repair after the existing one has been done. To do so, find the MIN by using a function of the StartJob column after using filter for the machines that are identical to the current machine in the row, have a StartJob that is after the recent repair in the row, and exclude "PM" f rom the row because they are preventive maintenance and not actual failures.

The RETURN portion of the formula is broken down into three parts. Since it is not included in the MTBF calculation, excluding the preventive maintenance activity, it will return 0 for uptime. If the value for next is BLANK, the most recent failure in the dataset has been identified. Finally, if none of these scenarios apply, simply measure the time difference between the start of the next repair and the end of the current repair in seconds.

Last but not least, create some measures on the table as mentioned below.

1. Repairs: This column is to show the total of repairs, minus the repair for preventive maintenance (PM), that have occurred.
   Formula :

Repairs = CALCULATE(COUNTROWS(MTBF),FILTER(MTBF,[RepairType]<>"PM"))

2. MTBF: This is the MTBF equation, which adds up the uptime in seconds, divides by the number of repairs to get an average, and then divides by 3600 seconds/hour to translate the number to hours.

Formula:

MTBF (Hours) = DIVIDE(SUM(MTBF[Uptime]),[Repairs],BLANK())/3600

3. MDT: MDT stands for Mean Down Time, or the average time it takes for a repair to be completed [5]. It is also used in conjunction with MTBF. The calculation is to divide the total number of repairs by the amount of our Repair Hours (which does not include preventive maintenance tasks).

   Formula: MDT (Hours) = SUM(MTBF[Repair Hours])/COUNTROWS(MTBF)

4. Last Repair: The column that explain when is the last date of a repair.

   Formula: Last Repair = MAX([FinishJob])

5. Next Expected Repair: Once the last repair is known, add the converted MTBF to days and then divide it by 24 to determine when the next failure is expected to take place.

Next Expected Repair = [Last Repair]+[MTBF(HOURS)]/24

Formula:

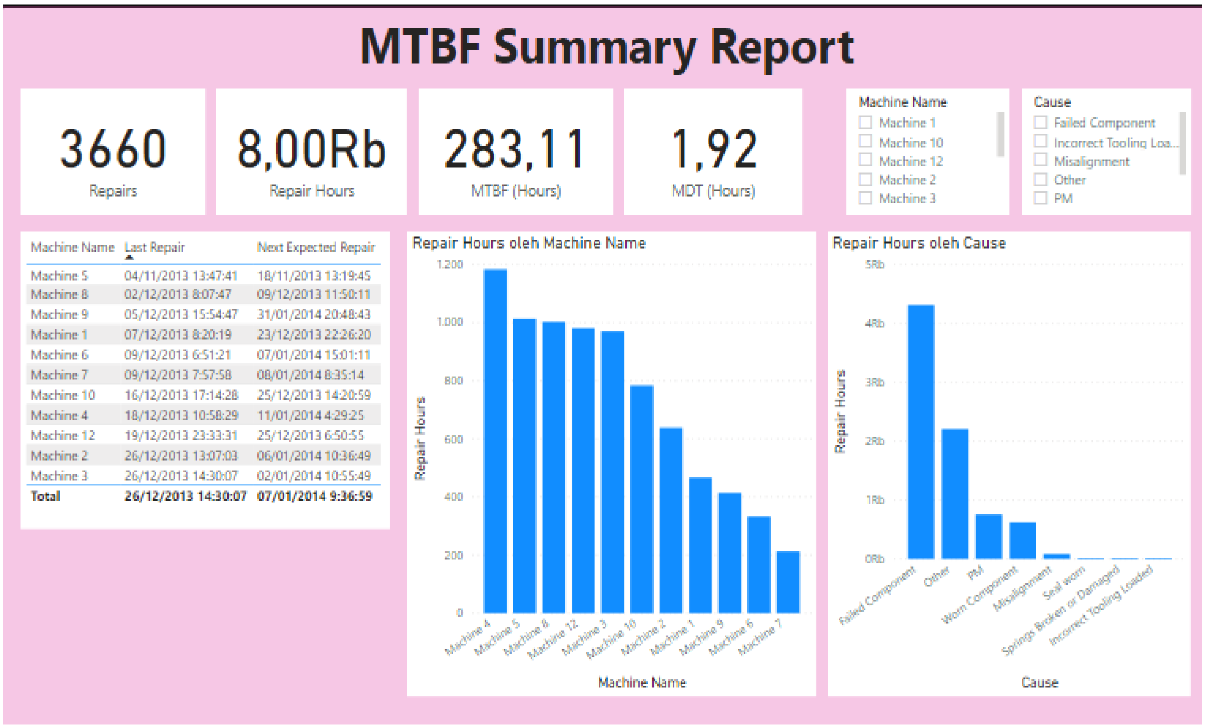
4. PowerBI Data Visualization

Visualizing the data is by dragging the chart from the visualization pane and fill in the detail box. The first chart that are build is the Repair Hours by Machine Name chart. Input the axis as Machine Name and Repair Hours while the value is Repair Hours.
The second chart is to build the Repair Hours by Cause chart. Input the axis as Cause and

Repair Hours while the value is Repair Hours.

Build filters by dragging two slicers and put Machine Name and Cause as the value. For additional information, drag card from the visualization pane and put other important values such as count of repairs, repair hours, MTBF(Hours), and MDT(Hours).
Then build a table to state the next expected repairs on each machine. Put the Machine

Name, Last Repair, and Next Expected Repair to the table. All of those combined is shown on Figure 3.5 be



MTBF Summary Report

3660 Repairs

8,00Rb Repair Hours

283,11 MTBF (Hours)

1,92 MDT (Hours)

Machine Name
□ Machine 1
□ Machine 10
□ Machine 12
□ Machine 2
□ Machine 3

Cause
□ Failed Component
□ Incorrect Tooling Loa...
□ Misalignment
□ Other
□ PM

Machine Name	Last Repair	Next Expected Repair
Machine 5	04/11/2013 13:47:41	18/11/2013 13:19:45
Machine 8	02/12/2013 8:07:47	09/12/2013 11:50:11
Machine 9	05/12/2013 15:54:47	31/01/2014 20:48:43
Machine 1	07/12/2013 8:20:19	23/12/2013 22:26:20
Machine 6	09/12/2013 6:51:21	07/01/2014 15:01:11
Machine 7	09/12/2013 7:57:58	08/01/2014 8:35:14
Machine 10	16/12/2013 17:14:28	25/12/2013 14:20:59
Machine 4	18/12/2013 10:58:29	11/01/2014 4:29:25
Machine 12	19/12/2013 23:33:31	25/12/2013 6:50:55
Machine 2	26/12/2013 13:07:03	06/01/2014 10:36:49
Machine 3	26/12/2013 14:30:07	02/01/2014 10:55:49
Total	**26/12/2013 14:30:07**	**07/01/2014 9:36:59**

Figure 3.5 Dashboard Example of MTBF Report

Conclusion

The proposed solution for calculating MTBF that we have presented suggest that it is much easier to do data engineering with PowerBI and Hadoop as the data processing and management even with a very simple data which is using machine log. While previously predictive maintenance is being done by analyzing each machine individually and manually calculate them. Using big data helps to organize the data needed to calculate mean time between failure efficiently and PowerBI helps to visualize and analyze said data.

For future implementation and work, there are several improvements needed to be done, especially if the data is much complex with additional measring such as electricity, voltage, energy consumption, etc. The data engineering operation may need additional columns formula and measures formula before being able to be visualize in the PowerBI.

References

[1] Mobley, R. K. (2002). An introduction to predictive maintenance. Elsevier.

[2] Torell, W., & Avelar, V. (2004). Mean time between failure: Explanation and standards. white paper, 78.

[3] Shvachko, K., Kuang, H., Radia, S., & Chansler, R. (2010, May). The hadoop distributed file system. In 2010 IEEE 26th symposium on mass storage systems and technologies (MSST) (pp. 1-10). Ieee.

[4] Azvine, B., Cui, Z., Nauck, D. D., & Majeed, B. (2006, June). Real time business intelligence f or the adaptive enterprise. In The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conf erence on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06) (pp. 29-29). IEEE.

[5] Rahman, C. M. (2015, March). Assessment of total productive maintenance implementation in a semiautomated manufacturing company through downtime and mean downtime analysis. In 2015 International Conference on Industrial Engineering and Operations Management (IEOM) (pp. 1-9). IEEE.