International Journal of Mechanical Engineering and Technology (IJMET)
Volume 9, Issue 1, January 2018, pp. 198–206, Article ID: IJMET_09_01_024
Available online at http://iaeme.com/Home/issue/IJMET?Volume=9&Issue=1
ISSN Print: 0976-6340 and ISSN Online: 0976-6359

© IAEME Publication **Scopus** Indexed

PRIORITIZING SOFTWARE MAINTENANCE ISSUES THROUGH ACQUISITION PROCESS

Dr.B.Chaitanya Krishna, K.Rajesh Kumar Reddy, V.H.N.V.R.S.Sai Pavan and M.Shanthi

Department of Computer Science and Engineering, K L E F, Guntur, India

ABSTRACT:

When we obtain a defect list from the user the development team will decide which defect should be fixed first. We use maintenance plan which consists of list that are fixed sequentially and it is most to prioritize the defects based on the ranking till now the software maintenance has not achieved the customer needs from this research we are going with a method of prioritizing the software defect to be fixed by concentrating on the 3 impact factors. Those are severity, priority and no of users who found the same defects. We are proposing 3 models in this paper. Those are Ranking Priority Analysis, Adaptive model and Preventive model. NLP is the Natural language Processing that is used to analyze the user feedback in order to extract defect related keywords. And this type of Natural language processing helps the user to maintain the software very efficiently. We will use the ranking process to prioritize the maintenance issues. This can be achieved by the Analytical Hierarchy Process (AHP) algorithm.

Keywords: Software defects, Software Maintenance, Preventive Model, Adaptive Model, Natural Language Processing, Analytical Hierarchy Process

Cite this Article: Dr.B.Chaitanya Krishna, K.Rajesh Kumar Reddy, V.H.N.V.R.S.Sai Pavan, M.Shanthi, Prioritizing Software Maintenance Issues through acquisition process, International Journal of Mechanical Engineering and Technology 9(1), 2018, pp. 198–206.
http://iaeme.com/Home/issue/IJMET?Volume=9&Issue=1

1. INTRODUCTION

Defects produced by clients ought to be settled and conveyed as quick as could be expected under the circumstances. By and large, application support group gets client criticism reports and client input for the most part in normal dialect which clarifies what and how clients encounter the issue. On the off chance that there are many defects, the engineer group needs to create programming support intend to plan abandon reparation utilizing their experience. The product support design fundamentally comprises of the arranged rundown of deformities to be settled.

Basically at times the imperfections are accounted for over and again by a few client and each deformity has distinctive levels of seriousness. An application till now does not comprehend the quantity of client who found a similar imperfection which ought to be one the factor to organize the effect of the defects.

As of now, the product upkeep design may not work well for to client needs. Here and there similar deformities are accounted for over and over by a few clients and each imperfection has distinctive level of seriousness [1] [2]. An application group still does not have a technique to comprehend the quantity of clients who found similar deformities which ought to be the one factor to organize the effect of imperfections. With current technique dependably get defer reaction since the procedure to organize programming upkeep design isn't practiced methodically. In extra, the arrangement may not consider and utilize critical effect factors. The key individual to need the arrangement might not have top to bottom learning of framework and imperfection seriousness impacts.

This paper proposes a model in development and prioritizing the software maintenance issues though some impact factors those are the severity, priority and the number of users who found the same issues this maintenance can be calculated by the Natural language processing and the Analytical hieratical algorithm process models and the prioritizing of the cost can be done on the Preventive maintenance model for this model we are going to get the details of the optimum cost of estimation that can be handled during the maintenance of the software. By this type of the model we are going to calculate the prioritizing of the cost. The adaptive software maintenance is the process model where we are going to prioritize the issues based on our rectification of the defect w.r.t. time.

This paper is sorted out as takes after: the next section describes the severity and priority defects methodology's way to deal with organize programming defects and analysis by client criticism in section 3. Evolution of the cost model by using different models in section 4.Section 5 presents the conclusion.

2. SEVERITY AND PRIORITY DEFECTS METHODOLOGY'S

The severity and need are the properties of an imperfection and ought to be given in the bug report.

Severity:

Severity of a defect is identified with how server of a bug is. Typically the seriousness is characterized in term of budgetary misfortune, harm to condition, organization notoriety and death toll.

Defect severity can be sorted into four class

- **Critical:** This deformity shows finish close down of the procedure, nothing can continue further
- **Major:** It is a profoundly serious deformity and crumple the framework. In any case, certain parts of the framework stay utilitarian
- **Medium:** It cause some bothersome conduct, however the framework is as yet utilitarian
- **Low:** It won't cause any real separate of the framework
- **Priority:**
- Need of a defect is identified with how rapidly a bug ought to be settled and sent to the live servers.

- **Low:** The imperfection is an aggravation yet repair should be possible once the more genuine deformity have been settled
- **Medium**: Amid the ordinary course of the improvement exercises deformity ought to be settled. It can hold up until the point that another adaptation is made
- **High:** The deformity must be settled at the earliest opportunity as it influences the framework extremely and can't be utilized until the point when it is settled

There are some of the methods that can be fix defects using the severity and priority those are

Methods of fixing defects by using severity and priority:

The Settling of the bugs by utilizing seriousness and need is grouped into 4 sorts of techniques. This strategies causes us to expel the bugs from the product.

1. High severity and High priority bug:

This is when significant way through the application is broken. For instance, an online business site, each client get a mistake message on booking structure and can't put request and this is a blunder 505 reaction.

2. High severity and Low priority bug:

This happens when the bug causes real issue however it happens just in extremely uncommon conditions (or) circumstances. In an occasion, the individual who utilize exceptionally old program can't proceed with their buy of item on the grounds that the quantity of clients with extremely old program is low, it isn't a high need to settle the issues.

3. Low severity and High priority bug:

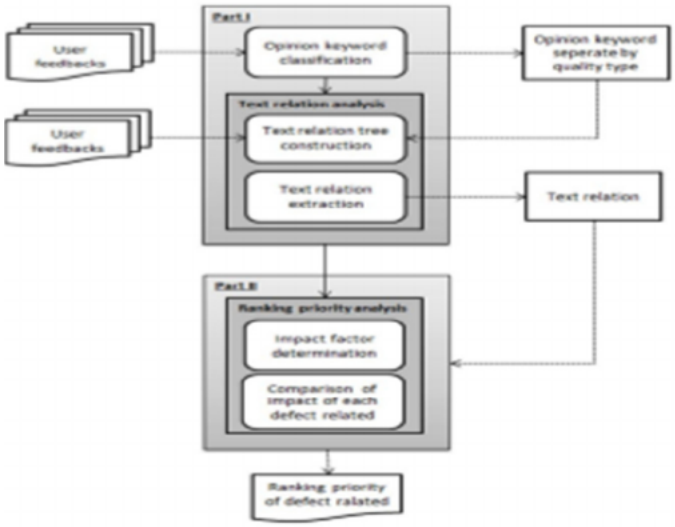
This could happen when a logo (or) name of the organization isn't shown on the site. It is vital to settle the issues at the earliest opportunity despite the fact that it may not cause a great deal of harm.

4. Low severity and Low priority bug:

For situations where the bugs doesn't cause catastrophes and just influence modest number of clients both seriousness and need are alloted low. In a case, the need arrangement page set aside a long opportunity to stack. Very few individuals see the security arrangement page and permit stacking doesn't influence the client much.

From the above Strategies and cases the group ought to choose the seriousness and need for each bug.

3. A WAY TO DEAL WITH ORGANIZE PROGRAMMING DEFECTS AND ANALYSIS BY CLIENT CRITICISM.



They are 2 Parts to implement this methods:

Part 1.The first part is a process to extract defect related keywords in the s/w through the NLP(Natural language processing) By using the grammatical relations.

Part 2.The second part is a process of find out the ranking of impact defect by using AHP (Analytical Hierarchy Process) algorithm.

Part 1:

Natural language Processing:

Definition:

The Natural Language Processing(NLP) is a Software engineering programming worried about the collaboration b/w the PC and human normal dialects.

The imperfections related watchwords are found by NLP(Natural Language Processing) by breaking down the client criticism we will extricate the deformity related catchphrases.

Sentence structure:

Lemmatization

1 .Morphological division:

Isolate words into singular morphemes and distinguish the class of the morphemes. The trouble of this errand depends incredibly on the unpredictability of the morphology (i.E. The structure of words) of the dialect being considered. English has a genuinely basic morphology.

2. Parts of discourse labeling:

Given a sentence and decide the parts of discourse for each word .Many words, particularly regular one can have a server as different parts of discourse .We lean toward English since we can have lesss uncertainty then in the contrasted with different dialects.

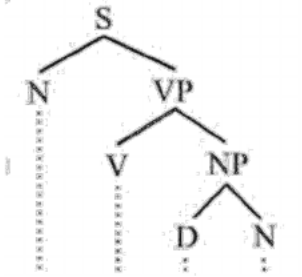
3. Parsing:

In this we will decide the parse tree and do linguistic investigation of a given sentence. The parse tree is characterized into two classes

1. Constituent based parse tree
2. Dependency based parse tree

Constituent based parse tree:

It portrays the recognize constituent linguistic use amongst terminal and non-terminal hubs



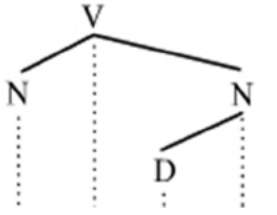
Ravi Hit the ball

The parsing begins from the S and consummation in the each of the hubs as (Ravi , hit, the, ball).

- S for sentence the best level structure in this illustration
- NP for thing phrase the main (furthest left) NP, a solitary thing "Ravi", fills in as the subject of the sentence. The second one is the object of the sentence.
- VP for verb phrase, which fills in as the predicate
- V for verb. For this situation, it's a transitive verb hit.
- D for determiner in this occasion the distinct verb "the"
- N for thing(Noun)

Dependency Based parse tree:

The reliance based parse trees of reliance punctuations see all hubs as terminal, which implies they don't recognize the qualification amongst terminal and non-terminal classes.



Ravi Hit the Ball

The above diagram is the dependency parse tree.

4. Sentence Breaking(also known as Sentence Boundary Disambiguation):

Given a piece of content, discover the sentence limits. Sentence limits are regularly set apart by periods or other accentuation checks yet these same characters can fill different needs (e.g. Checking abbreviations).

5. Text Relation extraction:

It is a procedure to extricate the relationship of each deformity related. This imperfection related catchphrases is developed from the content connection development which comprised by Nkw and Nrt. It begins connection between the from nkw with most abnormal amount with then cross the way with profundity initially seek calculation.

The new connection is esteem is come back with condition underneath and afterward deformity related terms Text Relation=(Nkw ,Quality Match,Nrt)

- Nkw is a hub speaks to every sentiment catchphrase grouping which decides the chose watchword that is coordinated every quality term.
- Nrt is a hub speaks to the related target that has a place with those imperfections from client input.

Part 2:

Ranking Priority Analysis:

Ranking prioritize analysis is a process that determine how to prioritize the defect should be fixed. We will be able to know the number of user who found the same defect can be obtained by the analysis of the text relation of the section.

This approach will apply AHP algorithm to prioritize the sequential defect. The process is divided into two parts.

1. Score of the impact factor to each defect related keyword.

2. Calculate the ranking of the user feedback based on the results of those 3 impact factors calculated.

AHP (Analytical Hierarchy Process):

It is a comprehensive mathematical framework for priority setting in complex System. According to set the developer of AHP model said that a complex system is decomposed into subsystems and represented in a hierarchical form. The element at the highest level is the goal. The elements at each level is called criteria and the elements at the bottom levels sub criteria (or) alternative. In this way the AHP organizes the basic rationality of the priority setting process by breaking down the complex system into its smaller constituent parts

We use matrix that is created to compare the impact of each pair of defects extraction based on the values obtained from considering the given factors. The process of comparison of weights by comparing the each pair of the prioritizing needs and the sum of the column in normalization.

$$\begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & & A_{2n} \\ \vdots & & \ddots & \vdots \\ A_{1n} & A_{n2} & \cdots & A_{nn} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix}$$

$$W_i = A_{i1}v_1 + A_{i2}v_2 + \cdots + A_{in}v_n = \sum_{j=1}^{n} A_{ij}v_j \qquad (1)$$

w_i is score of each criteria ranking
A_{ij} is weight of impact factor between defect i and j
v_i is weight of each criteria ranking
n is table size

By using the above framework is made to look at the effect of each match of deformity extraction, in view of the esteem got from considering the given factors, and determining the

effect weight as per the AHP strategy. The procedure will ascertain weight by looking at each match of the prioritization of requirements, and the total of every segment is standardization to be 1. The prioritization is ascertained by utilizing condition 1.

4. PRIORITIZE OF THE COST MODEL BY USING DIFFERENT MODELS

Many technical systems are subjected to aging and degradation during their lifetime. Most of these systems are repairable. Maintenance and repair problems have been widely investigated in the literature. Barlow and Proshan [3], Gertsbakh [4], Pham and Wang [6], Wang [12] survey and summarize theoretical developments and practical applications of maintenance model. Software maintenance typically accounts for at least 50 percent of the total lifetime cost of a software system [11]. Schach et al. found that 0 – 13.8% of changes made fall under the category of adaptive maintenance [7]. It is wise to design software that is maintainable since 18.2% of project time is devoted to adaptive maintenance [9]. An absence of approved, generally acknowledged, and received devices for arranging, evaluating, and performing upkeep additionally adds to this issue.

1. Adaptive maintenance of software Model:

We guess that the upkeep exertion for a product application relies upon quantifiable measurements that can be gotten from the product advancement process. We initially distinguished the measurements that could influence the exertion required for keeping up an application to enable directors to utilize the correct measurements. Next, we dealt with building up relationship be tween's support exertion and the recognized measurements. Our first arrangement of measurements was principally taken from the aftereffects of two past investigations: [12] and [10]. The information was taken from four sources: CS 499 and CS 616 courses (instructed at the College of Kentucky), the examination performed in [10] utilizing Together® [9] and the business investigate information from [1]. Table 1 shows the consequences of connection investigation: the higher the estimation of the coefficient, the more grounded the connection amongst exertion and the metric. The outcomes recommend that level of administrators changed and the quantity of lines of codes changed altered, included or erased (DLOC) were the best to predict versatile upkeep exertion.

Table 1 Correlation between metrics and effort

	Metrics	**Coefficient of Determination (R^2)**	**Significance level from ANOVA Regression (Analysis of varience)**
1.	%.Operation _Changed	0.789	0.002
2.	LOC-DELT	0.779	0.0003
3.	%Mod Change/Add	0.192	0.117
4.	No Ptr	0.152	0.4444

%Operation _Changed- Percent difference in total number of operators in the application after Maintenance

LOC-DELT-Lines of code added, editor or deleted during Maintenance

%MOD Change/Add- %Code modules Changed during Maintenance

NoPtr- Total number of Operators

After getting the Coefficient of the variance we can calculate the time that can produce now we are going to generate a data point and data source table.

Table 2 Number of data points per data source

	Date Source	Total data points	Data point with DLOC info	Data points with that info	Avg size of the app(lines of code, Min ,max)
1.	CS499	8	7	0^2	2677.2,100,602
2.	CS616	9	8^2	8^3	1934, 1192, 3425
3.	Previous In house Research Data	10	10	0^2	58.6,48,65
4.	Industry Research data	6	5	5	2725.4,652,523.5

This can be achieved by the method of least square. And formulae is that

E=-40+6.56(DLOC)

DLOC-Digital library of Caribbean

2. Preventive Maintenance Model:

In this model we are going to calculate the cost per unit time equation. The optimum preventive Maintenance time can be achieved by using the Weibull distribution, triangular distribution etc.

Lets us consider a non-repairable component with a failure rate behavior that is independent of the age of the system in which it is installed. Let us take a Weibull distribution with β=3 and η=150days. It will cost $8 each time the component is replaced after it fails, while it will costs $2 to replace the component before fails we will determine the optimum preventive maintenance time .

CPUT=X_P R(v)+X_q[1-R(v)]/$\int_0^t R(s)ds$

CPUT-Cost per Unit Time

X_q-Cost of Planned (preventive replacement)

X_{q-} Cost of unplanned(Corrective)

R(v)-Reliability Function for the component

t-Preventive Maintenance Time.

These are the some of the models where we can calculate the estimation of cost when we occur the defect Software maintenance.

5. CONCLUSION

This exploration proposes a technique to organizing the product support design by concentrating on 3 affect factors is seriousness, need and the quantity of clients who found similar deformities. We have proposed a strategy for assessing the push to perform versatile support in view of the evaluated number of lines of code to be changed as well as the quantity of administrators to be changed .A bigger scale think about with an assortment of industry extends crosswise over different spaces is required before any wide conclusions can be reached. Pearson's connection coefficient is given to approve consequence of programming imperfection organize which we accomplish score with 0.81 that implies programming deformity organizing from our approach and master judgment have a similar heading.

REFERENCES:

[1] N. Bettenburg, R. Premraj, T. Zimmermann, and K. Sunghun, Duplicate bug reports considered harmful; really?, in Software Maintenance, 2008. ICSM 2008. IEEE International Conference on, 2008, pp. 337-345.

[2] R. K. Saha, S. Khurshid, and D. E. Perry, Anempirical study of long lived bugs, in Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), Software Evolution Week - IEEE Conference on, 2014, pp. 144-153.

[3] T. L. Saaty, Decision making with the analytic hierarchy process, International Journal of Services Sciences, vol. 1, pp. 83-98, 2008.

[4] Akiyama F. An example of software system debugging, Inf Processing, Volume 71, 1971, 353-769.

[5] D. Mairiza, D. Zowghi, and N. Nurmvuliani, An investigation into the notion of non-functional requirements, presented at the Proceedings of the 2010 ACM Symposium on Applied Computing, Sierre, Switzerland, 2010.

[6] N. Jindal and B. Liu, Mining comparative sentences and relations, presented at the proceedings of the 21st national conference on Artificial intelligence - Volume 2, Boston, Massachusetts, 2006.

[7] Conte, Dundmore and Shen. Software Engineering Metrics and Models. The Benjamin/Cummings Publishing Company, Inc. 1996.

[8] http://www.esat.kuleuven.ac.be/sista/lssvmlab/tutorial/node 59.html

[9] Gefen D., and Schneberger, S. L. The Non-Homogeneous Maintenance Periods: A Case Study of Software Modifications. Proceedings of the International Conference on Software Maintenance, pages, Monterey, California, November 4-8, 1996, 134-143.

[10] https://en.wikipedia.org/wiki/Analytic_hierarchy_process_%E2%80%93_car_example Example Case Study for Software Maintenace issues through the AHP algorithm

[11] T. S. N. Group. (2015, Sept). The Stanford Pavser‖. Available: Internet: nlp.stanford.edu/software/lexparser.shtml

[12] N. Kaushik, M. amoui, L. Tahildari, L. Weiing, and L. Shimin, Defect Prioritization in the Software Industry: Challenges and Opportunities, in Software Testing, Verification and Validation (ICST), IEEE Sixth International Conference on, 2013, pp. 70-73.

[13] IEEE Standard Classification for Software Anomalies, IEEE Std 1044-1992, 1994.

[14] Dr. S. Ravichandran, Software Maintenance Metrics and Its Importance for Deriving Improvement in Software Maintenance Project: An Empirical Approach, International Journal of Information Technology and Management Information Systems (IJITMIS) Volume 1, Issue 2006, Jan–Dec 2006, pp. 01–05

[15] S. Manivannan, Dr. S. Balasubramanian, Software Metric Analysis Methods for Product Development / Maintenance Projects, International Journal of Computer Engineering and Technology, Volume 1, Number 1, May-June 2010, pp. 18-33