An Improved Fuzzy Rule-Based System for the Diagnosis of Software Rot Metrics

Adedokun Ajibike Mariam
Department of Computer Science
University of Port Harcourt, Nigeria

Onyecjegbu Laeticia Nneka
Department of Computer Science
University of Port Harcourt, Nigeria
Email: neka2k [AT] yahoo.com

Abstract--- Software rot has affected the productivity of various organizations due to the fact that our modern computer facilities are not immune to Software and Architectural rot. Numerous algorithms are made available in order to diagnose related Software rot issues. Some of these algorithm application for diagnosing Software rot issues generate problems such as slow service delivery, which results from non-application of Open-Source fuzzy logic tools, and limited storage capacity for datasets due to non-usage of Relational Database Management System (RDBMS). However, this study addresses the mentioned problems by reducing the complexity involved in diagnosing Software rot metrics through the adoption of Structured System Analysis and Design Methodology (SSADM), interfaced with FuzzyTech (FT) algorithm, and also adopts MySQL Database for adequate storage of datasets. Furthermore, the Proposed System is divided into two phases. The first phase involves the recognition of Software rot symptoms; the second phase diagnoses and represent the outputs in Triangular Form Membership Function format through the application of FuzzyTech(FT) open-source Software and C# programming language. FuzzyTech(FT) was used as a platform to improve the fuzzy logic system. In addition, this study mainly solves the problem of slow service delivery in diagnosing Software rot issues, and limited storage capacity for datasets used. Our result also shows that an essential feature of the fuzzy system is that a numerical value does not have to be fuzzified using only a membership function. In other words, a value can belong to multiple sets at the same time. For instance, Software Symptom value can be considered as negative, zero and positive at the same time with different degree of membership.

Keywords--- Fuzzy Logic, FuzzyTech, Membership Function, Metrics, Open-Source, Software rot, Triangular Form.

I. INTRODUCTION

As stated in Lehman’s first law [1] to maintain the usefulness or suitability of software in real-world applications, the software must have the ability to evolve with changes in requirements. Unfortunately, most of the software systems used in companies are susceptible to software rot and ageing. As such existing software systems may become less maintainable as time progresses. Software rot in software products is a common problem associated with software systems. Software Rot is a situation where the software system acquires flaws or bugs and other issues which makes it incompatible with the present situation because of factors such as changes in technology, disagreement amongst stakeholders. These changes may force the company to abandon the software application. A detailed study of the existing work in this area reveals that the cost or difficulty in terms of money, skill and effort involved in the maintenance of a software increases with increase in time. That is as the software ages, it becomes more difficult to effect changes. Obviously, it is more viable to replace the system or application with a new or redesigned system than to keep on maintaining the existing system. The design decision considered during development system may interfere with requirements that are to be introduced as a result of system evolution. Software Rot is the aftereffect of repeated unrestrained maintenance reduces the system’s quality. The consequence of uncontrolled maintenance overtime which degrades quality of system. In that case it becomes mandatory to replace the existing one old system. [1] Notwithstanding, complete system replacement is dangerous because it comes with great effect on technology, skill and organizational financial state. Often, replacement involves, retraining of user(s) and operators and the new system may not have the core functionalities present in the old system. These factors may place unbearable financial responsibility on the organization. Also, developing a new software usually involve writing of programs with lines of code running into several millions. These codes are grouped into several classes. To be sure that the new software will not also move into software Rot, there is need for a machine learning algorithm which can take care of voluminous dataset for the finding and discovering of pattern such that the system will have a learning ability.

Machine Learning algorithms consist of methods and procedures which enables learning in machines such as computer. To handle the challenges of software rots, it is better to adopt the evolutionary maintenance approach instead of complete replacement of the existing system. This way the impact of software rot could be minimized. [2]

II. REVIEW OF RELATED WORK

Software rot is also known as “code rot”, “bit rot”, “software erosion”, “software decay” or “software entropy” describes the perceived "rot" which is either a slow deterioration of software performance over long period of time or its...
III. ANALYSIS OF EXISTING SYSTEM

The influence of Software Architecture Rot has increased tremendously in Software Systems. However, the methodology of the existing system for diagnosing Software Architecture Rot is known as the Architectural Pattern Decision System (APDS). During the process of Software Architecture Design, a lot of architectural patterns are being implemented by the architect. [3] Architectural Pattern Decision System (APDS) is a unique system that aids in the diagnosis of Software Architecture Rot. This is because the major reason for the existence of Architectural Rot is the absence of useful design decisions. This decisions are often made during the design of the systems architecture and are part of the resulting architecture. Several types of Architectural Pattern Decision System (APDS) include: Archium, AREL, ArchDesigner and AQUA. Most of these decision system examples use different methodology to implement design decisions.

i) How the Architectural Pattern Decision System (APDS) works:
The Architectural Pattern Decision System (APDS) detects and diagnose Software Architecture Rot by gathering the most necessary and useful design decisions relating to the system’s architecture. Furthermore, most developers make use of most used architectural models in their design. Hence, the Architectural Pattern Decision System (APDS) captures what and how the architectural patterns are used, and then generate a set of corresponding output [3]. In Software Architecture Design of the existing system, the architects usually adopt some architectural patterns.

ii) Process Diagram of an Architecture Pattern Decision System (APDS)

From figure1, horizontally, analysis of system requirements, architectural design, implementation and deployment make up the stages of the software life cycle. Vertically, the segments; manual and tool support represent the manual activities and the finished events in the system. Furthermore, in the first phase, the architects will be able to arrange the entire Architectural Significant Requirements (ASR) of the System. [11] The Proposed System for the diagnosis of Software Rot Metrics is known as FuzzyTech System. The FuzzyTech System is a unique and efficient way for developing tools for fuzzy logic and neural fuzzy solutions. Furthermore, it comprises of an Open-Source FuzzyTech Software that enables adequate fuzzy logic designs. The Methodology of this study further covers the analysis and development of the Proposed System.

The Methodology used for the Proposed System is Structured System Analysis and Design Methodology (SSADM). This Methodology uses the system’s analysis and design strategy, the algorithm used is fuzzyTech algorithm.
Fig. 1: Process Diagram of an Architecture Pattern Decision System (APDS)
(Source: Perry et al, 2011)

Fig. 2: Sample Design of a FuzzyTech System (Proposed System)
Table 1: Basic Software Rot Diagnosis for MariamLaeticia Fuzzy Tech Systems.

<table>
<thead>
<tr>
<th>SYMPTOM</th>
<th>METRICS</th>
<th>DIAGNOSIS</th>
<th>POSSIBLE SOLUTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing Documentation</td>
<td>0.8 unit</td>
<td>Missing knowledge</td>
<td>Re-document through reverse engineering.</td>
</tr>
<tr>
<td>Dead Code</td>
<td>0.75 unit</td>
<td>Dead Code</td>
<td>Remove Dead Code</td>
</tr>
<tr>
<td>Missing Functionalities</td>
<td>0.8 unit</td>
<td>Missing Knowledge</td>
<td>Create, Split or modify programs to support those functionalities.</td>
</tr>
<tr>
<td>Useless Data</td>
<td>1.0 unit</td>
<td>Anomalous Data</td>
<td>Remove programs that create obsolete data.</td>
</tr>
<tr>
<td>Poor Lexicon</td>
<td>0.6 unit</td>
<td>Missing Knowledge</td>
<td>Rename and Refactor</td>
</tr>
<tr>
<td>Source less Programs</td>
<td>0.8 unit</td>
<td>Pollution</td>
<td>Rewrite Source code by means of reverse engineering</td>
</tr>
<tr>
<td>Pathological Files</td>
<td>0.55 unit</td>
<td>Coupling Issues</td>
<td>Re-factor by means of reverse engineering.</td>
</tr>
</tbody>
</table>

Fig.3: FuzzyTech 5.8: Proposed Sample Fuzzy Rules Software Rot Diagnosis System
Fig. 4: FuzzyTech 5.8: Triangular form Membership Function of SYMPTOM = {negative, zero, positive}

Fig. 5: FuzzyTech 5.8: Triangular form Membership Function of METRICS = {negative, zero, positive}
IV. RESULT DISCUSSION

An essential feature of the fuzzy system is that a numerical value does not have to be fuzzified using only a membership function. In other words, a value can belong to multiple sets at the same time. For instance, Software Symptom value can be considered as negative, zero and positive at the same time with different degree of memberships.

V. CONCLUSIONS

The existing system employs the use of predefined and frequently used architectural patterns. This process is not compatible with some Operating Systems (O.S) and often results to slow service delivery in the diagnoses of Software Rot metrics. Hence, there is need to appreciate the existing system, but as well develop an easy and efficient Fuzzy-base system for the diagnoses of Software Rot metrics. The proposed system highlights the efficiency in Software Rot metrics diagnosis through Fuzzy Logic applications. Input datasets are gathered and channeled to a fuzzy set using fuzzy linguistic variables, fuzzy linguistic terms and membership functions (Fuzzification). Thereafter, an inference is made based on a set of rules. Finally, the resulting fuzzy output is mapped to crisp outputs using the membership function in the defuzzification step.

REFERENCES


