ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 2, February 2015*

# An Enhanced Tool for Migration Testing of Web Applications

**S. Geetha[1], Dr. S. Iyakutti[2]**

Research Scholar in Computer Science, Madurai Kamaraj University, Madurai, Tamil Nadu, India[1]

Professor, Dept. of Physics & Nanotechnology, SRM University, Chennai, Tamil Nadu, India[2]

**Abstract:** Software organizations often need to migrate applications from one platform or technology to another for a variety of reasons. The software engineering research community is trying to find out techniques by which such migration projects can be carried out efficiently. The present paper proposes a technique called e-Splitter, to address the challenge of testing of migrated web application by enhancing an approach called Splitter. The paper attempts to draw on the power of genetic algorithms in addressing complex problems. Through an empirical study, we show that e-Splitter performs better than Splitter in testing of migrated web applications.

**Keywords**: Splitter, e-Splitter, Web Application migration, Testing

## 1. INTRODUCTION

The novel advancements in hardware and software technologies necessitate organizations to resort to migration of applications to exploit the advantages presented by these advancements. The Software Engineering research community is working hard to discover techniques, tools and principles by which such migration projects can be handled efficiently.

One of the challenges associated with migration projects is testing. Testers need to ensure that the original and the migrated applications produce the same results when presented with test cases. The test case generation becomes a baffling problem as testers are often uncertain about the precise nature of inputs that would occur in the production application. Migration projects are fraught with problems like incorrect settings in configuration files, incorrect security settings and a host of other issues that need to be uncovered by the presented test cases.

## 2. BACKGROUND

Ding et al propose a tool called Splitter that can be used in testing of migrated web applications [1]. The basic idea behind Splitter is as follows: A Proxy is put in front of the Web Server and this proxy intercepts HTTP requests and forwards them to both the original and the migrated applications. The outputs of the original and the migrated applications are compared detecting a flaw when a mismatch occurs. The obvious question that arises is the likely impact on the production environment but Ding et al prove that the overhead is within tolerable limits. The basic architecture is diagrammed below. It is taken from [1] and is presented here for clarity.

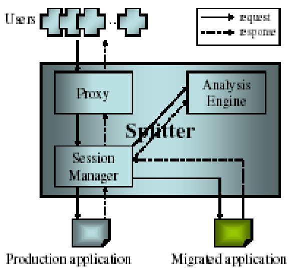

Figure 1 – Basic Architecture of Splitter (from [1])

The major components of Splitter are:

**Proxy**

Ding et al use Squid [2] which is a commonly used Web Proxy for replicating requests. The proxy forwards incoming requests to the production application without any delay and also sends the same to Session Manager Component.

**Session Manager**

HTTP requests often contain state information in cookies and direct forwarding of these into the migrated application will cause problems. The session manager component takes care of this by substituting a cookie in place of its occurrence. The session manager also addresses the issues of dealing with URL parameters often found with HTTP GET requests and HTTP POST parameters.

**Analysis Engine**

This component compares the responses from the production and migrated applications and reports issues to the test engineer. A major challenge on the comparison is imposed by the presence of dynamic contents in the response. If the response contains only static contents, the responses form the production and migrated applications will be the same. But in the presence of dynamic contents like timestamps and advertisements, the production and migrated application might yield different responses. To differentiate between the case where the different responses indicate an issue in the migrated application and the case where different responses are solely due to the presence of dynamic contents, Ding et al propose an algorithm that uses heuristics. Simple text responses are compared using the SES algorithm [3 Chawathe1996]. HTML responses are converted to DOM trees [4 Wood20000] and compared using heuristics like Structure heuristic which captures the difference is the structure of DOM trees with high importance given to nodes closer to the root and the distribution heuristic which examines the value distribution of leaf nodes. In the case of distribution heuristic, KS Test [5 Young 77] is used to quantify the differences. In order to avoid, too much attention being

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 2, February 2015*

given to little differences, Ding et al categorize related problems into one group.

## 3. E-SPLITTER

We propose some modifications in the Analysis Engine component of splitter with the objective of improving the accuracy of the performance of Splitter. The task of the Analysis Engine is the complex of all the three components. The analysis engine must be able to detect if differences in responses of the production and migration applications are different and if they are different decide whether the difference is due to an issue in the migrated application or acceptable difference due to presence of dynamic contents. It must also be able to rank the differences in the order of importance when presenting the differences to the test engineer who can choose to ignore or inspect the lower ranked differences. It must also categorize related problems into one group.

For all these tasks of the Analysis Engine, we propose to use Genetic Algorithm as opposed to the heuristic based algorithm used by Ding et al. Genetic Algorithms mimic the natural process of evolution in uncovering solutions to problems and have been applied successfully in a variety of domains to solve complex problems. Therefore, we examine if the power of GA can be exploited to improve the performance of Splitter.

**Applying GA**

The skeleton of the basic GA is shown below:

```
Initialize a population of solutions at random
Repeat
Evaluate fitness of each individual
Based on the fitness, select 2 solutions at random
Cross-over the selected solutions to generate an off-spring solution
Mutate the Off-Spring Solution at random
Replace the less fit solutions with the newly generated off-spring solutions
Until termination-criterion
```

The Genetic Algorithm for our problem is presented with 2 DOM trees representing HTML responses form the production and migrated applications and is expected to output the difference between the documents. These differences may not be straight forward differences in textual content. For applying GA, the choice of the fitness function which represents the fitness of a particular solution in solving the problem, is critical. The fitness function for our case is obtained using the Linear Discriminant Analysis and the leave-one-out method of training and testing. In the leave-one-out method, the system is trained with all but one pair of responses to be compared and it is examined if the system is able to predict the difference for the left out pair.

Our proposed enhancement also uses GA to cluster the erroneous responses found by the analysis engine component, this is desirable as otherwise it is likely that the test engineer will be flooded with a large number of erroneous responses that need to be examined. In this GA is provided with the erroneous responses uncovered by the previous step as input and the GA outputs clusters of responses where each cluster contains a group of related erroneous responses, probably triggered by the same fault in the migration application.

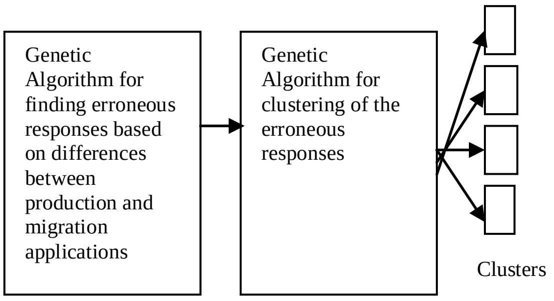
Genetic Algorithm for finding erroneous responses based on differences between production and migration applications
Genetic Algorithm for clustering of the erroneous responses
Clusters

Figure 2 – Sketch of the Analysis Component enhanced with GA

## 4. EVALUATION OF E-SPLITTER

To demonstrate the superiority in performance achieved with s-Splitter we conducted an empirical study with 3 moderately sized web sites which were migrated recently by a local software organization. The web sites were migrated from PHP to ASP.NET and the databases were migrated from mySQL to SQL Server. We first used Splitter to find out the number of erroneous responses uncovered and then used e-Splitter. A manual inspection was used to find out the exact number of erroneous responses. Splitter was able to find out all the erroneous differences but it reported a moderate number of false positives, which were correct responses in actuality, but reported by Splitter as erroneous.

We then used e-Splitter for the same purpose. E-Splitter also was able to identify all the erroneous responses and reported a smaller number of false-positives which justified our assumption that GA is likely to be more accurate in tackling the issue compared to the heuristic based algorithm adopted by Ding et al.

Table 1: False Positives reported by Splitter and e-Splitter

| Application | False Positives reported by Splitter | False Positives reported by e-Splitter |
|---|---|---|
| A1 | 97 | 35 |
| A2 | 134 | 89 |
| A3 | 72 | 29 |

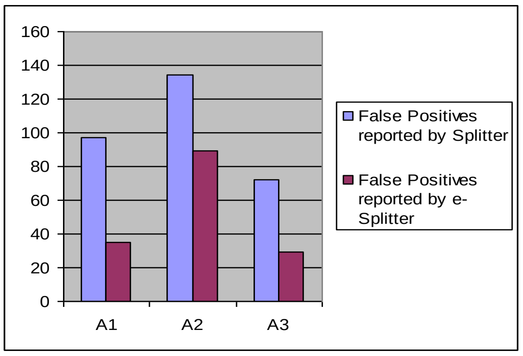

Figure 3 – Comparison of false positives reported by Splitter and e-Splitter

To evaluate the power of GA for clustering erroneous responses we used the Cluster purity measure which is the ratio of the responses within a cluster that were not due to the same fault as the other responses in the cluster. Here again, e-Splitter was able to arrive at more pure clusters and in fact on an average, 92% of the clusters thus found, contained erroneous responses triggered by the same fault in the migrated application.

Table 2: Average Cluster Purity Splitter and e-Splitter

| Application | Average Cluster Purity of Splitter | Average Cluster Purity of e-Splitter |
|---|---|---|
| A1 | 83% | 90% |
| A2 | 88% | 94% |
| A3 | 81% | 92% |

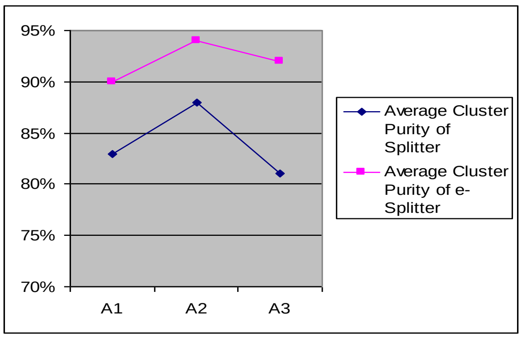

Figure 4 – Comparison of Cluster Purity of Splitter and e-Splitter

## 5. CONCLUSION AND FUTURE WORK

There is an imperative necessity for tools that support testing of migrated applications in the endeavor of efficient handling of migration projects. The paper presented an enhancement using GA to a tool called Splitter that can be used for testing migrated web applications. the enhanced tool christened e-Splitter was able to uncover all the errors while at the same time maintaining a lesser number of false positives in an empirical study conducted with 3 web applications. e-Splitter was also found to cluster the erroneous responses more accurately than Splitter. This clustering can greatly save the time and effort of the test engineer.

As a part of future work, we plan to develop other similar tools that can be used for testing non-web applications as well. We also envision development of tools that can greatly aid in the design and coding phase of migration projects.

## REFERENCES

[1] Ding, Xiaoning, H. Huang, Y. Ruan, A. Shaikh, B. Peterson, X. Zhang, Splitter: A Proxy-Based Approach for Post-Migration Testing of Web Applications,
[2] Squid Proxy Cache, http://www.squid-cache.org
[3] Chawathe, Sudarshan S, A. Rajaraman, H. Garcia-Molina, J. Widom, Change Detection in hierarchically structured information, SIGMOD, 1996.
[4] Wood, Lauren, V. Apparao, S. Byrne, M. Champion, S. Isaacs, Document Object Model (DOM) Level 1 specification, Retreieved from: http://www.w3.org/TR/2000/WD-DOM-Level-1-20000929/
[5] Young, Ian T, Proof without prejudice: use of the kolmogorov-smirnov test for the analysis of histograms from flow systems and other sources, The Journal of Histoichemistry and Cytochemistry, 1977.
[6] Benedikt, Michael , J. Freire, and P.Godefroid, VeriWeb: Automatically testing dynamic Web sites, 2002.