Change Impact Analysis using Python for Java Applications

T. Jalaja, T. Adilakshmi

***Abstract*: Software Applications needs to be changed constantly as per the requirements of the user or the client. This phase is known as the maintenance phase of a software application. On an average, the cost of software maintenance is more than 50% of all Software Development Life Cycle phases. The main purpose of software maintenance is to modify and update software application after delivery in order to correct faults, enhance the functionality or to improve performance. When the change request (CR) is received from the client the developers have to work upon the request. If the Change Request requires any modification, the application developers have to identify the risk of modifying the program or application before making the actual change. But there is a high chance of making errors in modifying the existing software. Change Impact analysis (CIA) is one of the error prevention technique. It is the process to find the effect of a change in a software application before the changes are made. By equipping developers with automated CIA tools to identify the risk of modifying the application we can minimize the errors.**

Impact analysis can be done based on the change request. This paper aims at developing a change impact analysis tool which can be used by the developer during the software maintenance phase. This proposed tool -Strategic Dependency Tracker (SDepTrac) helps the programmer/ developer to know "What part of the program (of a Java application) is impacted if a change is to be made to a particular variable / method / class?" with more accuracy compared to the existing tools. It displays the number of lines affected, classes and methods which are going to be affected, if we perform the requested change by considering the data dependency, control dependency and the semantic dependency. Thus the tool helps the developer to identify the impact set and minimizes the human errors and also saves time during the Maintenance phase.

***Keywords*: Bag of words, Change Impact analysis, Change request, SDepTrac Tool, Strategic dependency tracker.**

I. INTRODUCTION

During the maintenance phase, developers carry out adaptive, corrective, perfective, and preventive activities. The maintenance phase activities initiate a set of changes in the software system [1].

Software developer should be able to make the change accurately without affecting the existing workflows of the software application. A developer should be well versed in analyzing the potential impacts of the new changes. A recent survey has identified unknown impacts as the biggest impediment for implementing new changes. Thus "Change Impact Analysis" is having greater significance in the software Industries.

Software change impact analysis (CIA) is a technique for predicting the potential effects of changes before they are made, or measuring the potential effects of changes after they are made[5]. According to recent studies, software system developers are prone to fail in assessing some impacts on the systems they have built in certain cases. It would be difficult to identify the impacts of a proposed change as the software system becomes larger. Even an experienced developer may not guarantee to cover all potential impacts of a change. CIA is used to minimize the unexpected side effects of a proposed change [5,7]. So the proposed CIA tool, SDepTrac will be helpful to the developer to identify the impacted lines of code in an Java application [6].

In the next section-II the proposed algorithm and implementation steps used for identifying the final impacted set corresponding to a given change request by using the BOW concept are discussed. The Bag Of Words concept which is used to construct a knowledge base consisting of required words[4]. In section-III the results of the proposed work are discussed.

II. METHODOLOGY

A. Proposed Work

Each Change Request (CR) is represented as a BOW entity consisting of corresponding CR Summary and Description. From the data retrieved from repositories the corpus is created that represents knowledge base against which new change requests will be evaluated [4]. Using this knowledge base, data, classes and method impacts for new change requests are found by following the steps of SDepTrac Algorithm. The algorithm takes as input the change request and the path of the folder which consists of the java files of the application to be checked. The query CR represents the new CR, for which variable, class and method impact needs are to be identified [2]. The existing CRs are checked to find whether the similar CR exists, otherwise it is added to the list of existing CRs. If the CR already exists it displays the impacted classes, methods and lines of code without recalculating.

B. Proposed Algorithm (SDepTrac)

Input: CRi and the Application folder // CRi is the input change request

Output: File containing the lines and files affected due to Cri

Revised Manuscript Received on May 13, 2020.

***** Correspondence Author

T. Jalaja*, Department of Computer Science & Engineering, Vasavi College of Engineering (Autonomous), Hyderabad (Telangana), India. Email: tjalajasp@gmail.com

T. Adilakshmi, Department of Computer Science & Engineering, Vasavi College of Engineering (Autonomous), Hyderabad (Telangana), India. Email: t_adilakshmi@rediffmaill.com

Step 1: Takes the input Application folder which contains multiple files and starts the procedure by sending each file to the method bow( )
Step 2: Extract all the classes, methods and variables in the input application
Step 3: Take the CR(change request) from the developer and checks if the CRi is already there in the output file or not.
Step 4: if (CRi not in file) then
Add CRi to existing CRs
Else displays the corresponding
file contents and halts.
Step 5: Identify if any variable v is present in the CRi.
Step 6: Check whether the variable v present in the CRi is found in the input files and also identify the variables, methods and classes impacted due to the change in variable v.
Step 7: Else If CRi contains a class name c, then First extract input files containing the classname and also the inherited classes/ impacted classes.
Step 8: Else If CRi contains a method name m, then identify the variables methods and classes impacted due to m.
Step 9: Finally it opens a new file containing the lines affected with the file number and line number due to input change request CRi.

C. Implementation Steps

- Classification Of Elements:
  The Change request given to the programmer is tokenized with their respective parts of speech and these tags can be used to classify the names given in the CR into variable names, class names and method names [4].
- Getting the Impact Set:
  In this phase, we give the output of the first phase to this where variable names, class names and method names are going to be searched in the application and it finds in which files the class resides. It also displays the classes and methods which are going to be affected if we perform that change by considering the data dependency, control dependency and the semantic dependency [3,7].
- Performance Metrics:
  Accuracy, Precision and Recall are the evaluation measures used for studying the performance of the SDepTrac tool developed. Recall and Precision are widely used measures in information retrieval systems[5,7].

A confusion matrix in Fig.1 is often used to describe the performance of a classification model on a set of test data for which the true values are known. It shows how the actual values differ from the predicted values with these four attributes.

Actual Class	Predicted Class	
	Negative	Positive
Negative	TN (True Negatives)	FP (False Positives)
Positive	FN (False Negatives)	TP (True Positives)

Fig. 1: Confusion matrix

Accuracy is the proportion of correct classifications from overall number of cases.

$$Accuracy = \frac{truepositives + truenegatives}{totalexamples}$$

Recall is the proportion of correct positive classifications from the actual positive cases.

$$Recall = \frac{truepositives}{truepositives + falsenegatives}$$

Precision is the proportion of correct positive classifications from the predicted positive cases.

$$Precision = \frac{truepositives}{truepositives + falsepositives}$$

The below Table-I shows the sample Change Requests along with the TP, TN, FP, FN and Accuracy values.

Table-I: Sample CRs showing the corresponding TP, TN, FP,FN and Accuracy

CHANGE REQUEST	True Positive	True Negative	False Positive	False Negative	Accuracy
Change the variable sNo	25	506	20	0	0.963
Change the variable bookName	7	524	16	0	0.9707
Change the variable authorName	9	522	13	0	0.976
Change the variable bookQty	16	515	10	0	0.986
Change the variable searchChoice	3	528	2	0	0.992
Change the class Student	10	521	5	0	0.99
Change the class Library	2	529	1	0	0.9981
Change the class book	18	513	6	0	0.988
Change addBook	1	530	0	0	0.94
Change searchByAuthorName	1	530	0	0	0.93
Change showAllBook	2	529	0	0	0.956
Change check_prime	3	50	0	0	0.98
Change class second	4	49	0	0	0.95
Change the variable b	19	34	3	0	0.946
Change class SavingsAccount	3	49	4	0	0.928
Change variable balance	11	41	2	0	0.962
change method withdraw	1	51	0	0	0.998
Change interest	4	48	2	0	0.962
Change variable deposit	2	50	0	0	0.998
Change variable amount	4	48	1	0	0.981
change class hello	3	54	0	0	0.98
change variable x	7	50	3	0	0.95
change variable y	7	50	2	0	0.966
change variable y	1	56	0	0	0.98
change class time	32	25	0	0	0.98
change variable h	18	39	2	0	0.966
change class sample	1	56	0	0	0.98
				Average Accuracy =	0.9702519

III. RESULTS AND DISCUSSION

Three Applications were taken for testing the accuracy of the Tool developed [8, 9].

The average of the accuracy of these three applications is considered to find the accuracy for the designed Tool.

By considering the following code snippet [in Fig.2] of the program which is to be modified as per the change request, the accuracy is calculated.

```
5     int i=0;
6     int j=i;
7     if(j==0)
8     {
9         System.out.println(j);
10    }
```

Fig.2: Application - Code Snippet

Change Request : Change variable i.
Application Generated Results:
True positives=2(line 5, line 6)
True Negatives=1(Line 7)
False Positive=1(Line 9)
False Negative=2(Line 7,Line 9)
Manually Calculated Results:
True positives=2(line 5, line 6)
True Negatives=1(Line 7)
False Positive=1(Line 9)
False Negative=1(Line 7)
Accuracy= (TP+TN)/total of application generated
=(3)/4 =75%

The below figure Fig.3 shows the No of lines affected according to the above change request.

```
Enter change request: change ghe variable b
C:\Users\naish\OneDrive\Desktop\CIA PY\Demo\\demo1.java
The number of lines in the file are
57
The number of lines affected due to this variable change are
17
The affected portion due to variables are
['public', 'subtract()', 'boolean', 'public', 'numbers")', 'in.nextInt()', 'subt
ract(a,b)', 'subtract(a,b)', 'c=b', 'check_prime(b))', 'if(b-a==2', 'a-b==2)', '
be', 'numbers")', '\tpublic', 'but', 'affectedvbut']
classes affected are
['hello', 'second', 'time']


['boolean check_prime']


['class second.boolean check_prime']
>>>
```

Fig. 3: Output of the SDepTracTool showing the no of lines affected

The accuracy of the 3 applications under consideration are calculated and shown in the following figures, for a given set of change requests.

Fig.4 shows the accuracy graph for the Banking Application.

Accuracy	CRS	CHANGE REQUESTS
0.928	1	Change class SavingsAccount
0.962	2	Change variable balance
0.998	3	change method withdraw
0.962	4	Change interest
0.998	5	Change variable deposit
0.981	6	Change variable amount

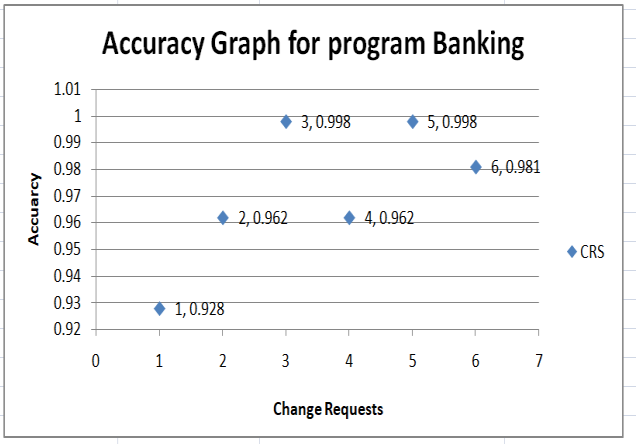

Fig. 4: Accuracy graph for Banking Application

The following figure Fig.5 shows the accuracy graph for Time Application.

CHANGE REQUESTS	CRS	Accuracy
change class hello	1	0.98
change variable x	2	0.95
change variable y	3	0.966
change class time	4	0.98
change variable h	5	0.966
change class sample	6	0.98

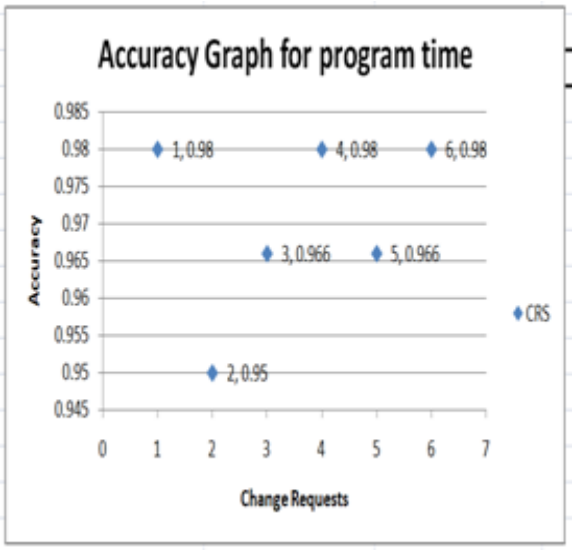

Fig.5: Accuracy graph for Time Application

The following Figure Fig. 6 shows the accuracy graph for Library Application.

Accuracy	CRS	Change Requests
0.963	1	Change the variable sNo
0.9707	2	Change the variable bookName
0.976	3	Change the variable authorName
0.986	4	Change the variable bookQty
0.992	5	Change the variable searchChoice
0.99	6	Change the class Student
0.9981	7	Change the class Library
0.988	8	Change the class book
0.94	9	Change addBook
0.93	10	Change searchByAuthorName
0.956	11	Change showAllBook

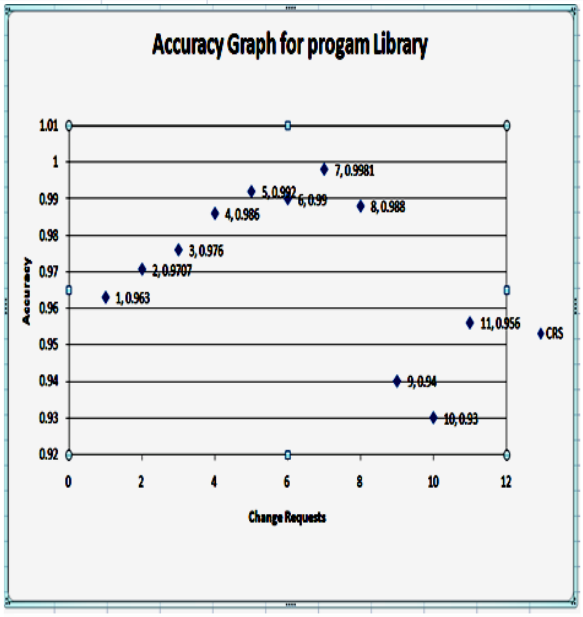

Fig.6: Accuracy graph for Library Application

The average accuracy of the above mentioned 3 Applications – Banking Application, Time Application and Library Application is calculated and shown in the below Figure Fig.7 and in Table-II.

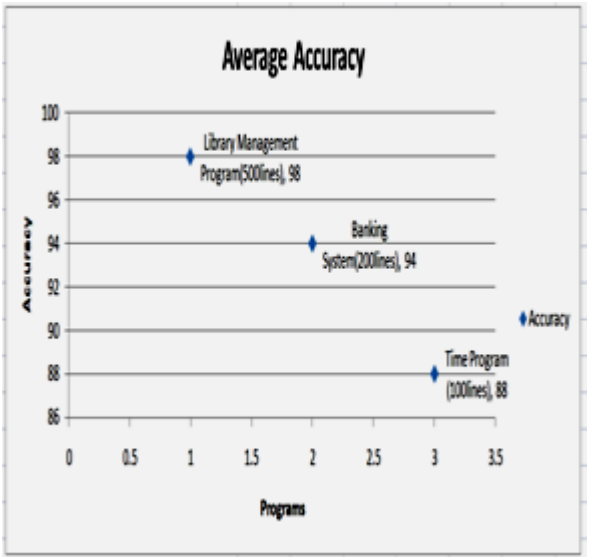

Fig.7: Average Accuracy of 3 Applications

Table-II shows the average accuracy of the applications tested

S.No	Application Tested	Average Accuracy (%)
1	Time Application	88
2	Banking Application	94
3	Library Management Application	98

IV. CONCLUSION AND FUTURE WORK

This paper describes the significance of Change Impact Analysis during the maintenance phase. The software Tool called SDepTrac is developed to assist the developer / programmer in identifying the impacted changes based on the given Change Request. The SDepTrac Tool has been implemented in python. It takes the input as a folder having java files and verifies each and every line of each file and displays the impacted lines along with the method and class names. The result shows the average accuracy of 93%. The accuracy may vary for different applications. The accuracy was calculated by using the manual result and comparing with the results generated by the Tool. The tool has been tested on three Java applications.

The limitation of this tool is that it will work only on Java Applications. The public domain applications have been used in this paper to demonstrate the working of the tool.

The tool can be further enhanced for supporting wide range of developers Eg:- implementing for multiple languages, Prioritizing the impacted files, Estimating the right resource count based on the impacted set.

REFERENCES

1. Sufyan Basri, Nazri Kama, Roslina Ibrahim and Saiful Adli Ismail, ”A Change Impact Analysis Tool for Software Development Phase”, Internal Journal of Software Engineering and its Applications, Vol. 9 No. 9 (2015).
2. Bixin Li, Xiaobing Sun, Hareton Leung and Sai Zhang,”A survey of code-based change impact analysis techniques” Published online in Wiley Online Library, Software Testing, Verification and Reliability (2012).
3. Haipeng Cai, Raul Santelices and siyuan Jiang ,“Prioritizing Change-Impact Analysis via Semantic Program-Dependence Quantification”, IEEE Transactions on Reliability, Vol.65, No.3, September 2016.
4. Shaikh Jeeshan Kabeer, “An Analogy Based Technique for Predicting the Vector Impact of Software Change Requests”, Thesis, University of CalGary, 2018.
5. Stephanie Perez Day Galbo, “A Survey of Impact Analysis Tools for Effective Code Evolution”, Thesis, 2017.
6. T. Jalaja, T.Adilakshmi,”Automation of Impact Analysis”, International Journal of New Innovations n Engineering & Technology, Vol.11 Issue 4, September 2019.
7. Tushar Sharma, Girish Suryanarayana ,– “Augur : Incorporating hidden dependencies” IEEE Paper 2016.
8. https://codereview.stackexchange.com/questions/206615/a-student-library-program-in-java.
9. https://gist.github.com/jimmykurian/1732868

AUTHORS PROFILE

T. Jalaja currently working as Assistant Professor with Vasavi College of Engineering , Hyderabad, India. She has completed her M.Tech (CSE) from Osmania University, Hyderabad in 2005. Currently pursuing Ph.D from Osmania University, Hyderabad, India. Have 10 years experience in Teaching. She is a Lifetime member in Computer society of India. The areas of interest are Programming, Software Engineering, Data Mining and Compiler Design.

T.Adilakshmi currently working as Head of the Department (CSE) & Professor in Vasav with Vasavi College of Engineering , Hyderabad, India. She completed Ph.D from HCU in 2006. Currently guiding more than 20 scholars from JNTU and OU. Two students are awarded Ph.D under her guidance.. She got more than 30 yrs of experience and more than 50 publications into her credit. Her area of interest are Image Processing, Artificial Intelligence and Machine Language.She is Lifetime member of ISTE and CSI Professional Bodies.