Volume 5, Issue 5, May 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Clone Detection in UML Sequence Diagrams Using Token Based Approach

Balwinder Kaur
M.Tech Scholar, University College of Engineering,
Punjabi University,
Patiala, Punjab, India

Er. Harpreet Kaur
Assistant Professor, University College of Engineering,
Punjabi University,
Patiala, Punjab, India

Abstract: Model Based Development appears to progress extremely in large scale software companies. UML (Unified Modeling Language) is raising as an utility in software development. In object oriented development, the complete details for the lifecycle are provided by UML. UML is a standard modeling language, so that it is used for analysis, design and implementation of software based systems. Clone detection in UML models examines duplicated parts, which can be reused and will reduce maintenance cost. The prime prospect of this paper is to demonstrate an innovative contribution of novel technique and profitable strategy designed to find clones in UML Sequence Diagrams. UML sequence diagrams (SDs) are mainly used to model the behaviour of web applications and other interactive applications where the sequencing of interactions over time needs to be specified. In this paper an approach suffix array is implemented using arraylist to detect model clones. The advantage of this approach is that clone detection in sequence diagrams helps us to inspect the duplicate parts at earlier stage in the software development process, suffix array consumes less memory and executes faster.

Keywords: Model clones, UML models, Code clones, Model clone detection.

I. INTRODUCTION

Software Clones are indistinguishable portions of designs and codes. Copying and pasting in programming is one of the main sources for similar documents in software systems [15]. The copied part is called the clone of the original and the process of creating such duplicate fragments is called code cloning. Like code cloning with the advent of model based development, a need arises to detect the occurrence of clones in UML models. Clones detected at the design phase are called model clones and clones detected at the implementation phase are called code clones.

Various reasons for software clones are: Programmer's less knowledge and time limits, complexity of the system, language limitations, fear of making fresh code, lack of abstraction [4]. It is very important to detect clones because of good maintainability, compact code, good patterns and to avoid copy paste approach [15].

Model Driven Development (MDD) defines domain models also called the conceptual models which mainly focuses on modeling rather than computer programming. Various UML tools like StarUML are used that provides multiple views of the models. There are various UML models such as class diagram, activity diagram and sequence diagram etc [3]. Use of UML makes modeling more effective and efficient.

This paper presents an approach to detect clones in UML diagrams. UML diagrams contain redundant elements which increase the complexity and hence need to be removed. In this paper we propose a technique that finds the similarity between model elements. UML diagrams are created using tool like StarUML that is encoded as XML document. The XML document is parsed to extract the tokens. Java programming is used for token matching and matched tokens are reported as clones. Clone detection in such models will help us to identify the duplicate part within the models and also helps in reusability.The applicability of our approach is demonstrated by taking UML class diagram and UML sequence diagram examples.

1.1 Motivation

1.1.1 Higher level of abstractions and a number of modeling languages has made modeling a key industry practice in different domains and different phases of software development life cycle.

1.1.2 Models attain substantial size [11] thereby increasing complexity and higher rate of duplications.

1.1.3 There is a dearth of studies suggesting techniques for detection of clones in UML models.

II. MODEL CLONING

A) Model Clone

A model fragment is a set of model elements that is closed under some closure property of similarity. Model Clone is a pair of model fragments that contains high degree of similarity.

B) Clones in Sequence Diagram

Sequence diagrams show interactions between classes arranged in a time sequence. Interactions are the message exchanges that take place between classes to accomplish a specific purpose.

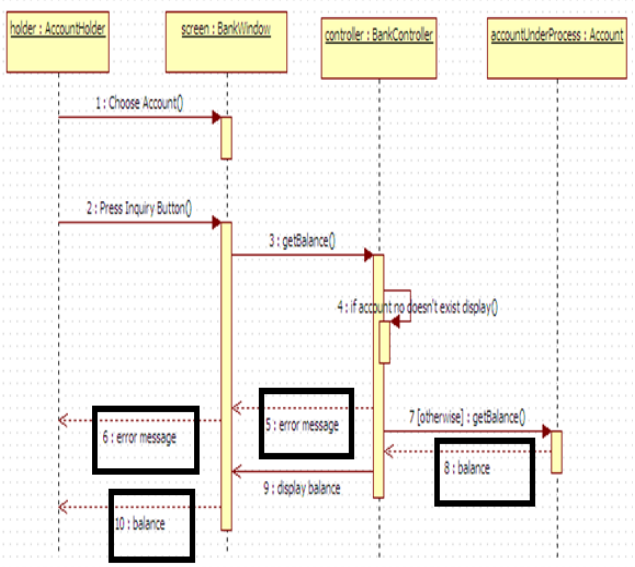

Fig. 1: Clones in sequence diagram

For Example: In figure 1 Sequence diagram of InquiryBalance have classes BankWindow, BankController and Account. Interaction between the classes occurs through sequence of messages i.e. ChooseAccount, Press Inquiry Button, getBalance, if account no. does not exist display, error message, balance, displayBalance. In this sequence diagram error message and balance messages exist two times, so that in sequence diagrams clones exist and there is need to identify clones. Clones are highlighted with solid boundary.

III. LIERATURE REVIEW

S. Mythili et al.[1] proposed Efficient Weight Assignment method for clone detection in state flow diagrams. Clone detection in state flow diagrams takes query model as input. The weight identification of the query model is done. The weight is compared with the weights of all the models available in the database. If both the weight matches then the whole model is said to be cloned. Clone is displayed when weight of the query model matches with one of the weight in the database.

Pamilpreet Singh et al.[2] proposed Detection of clones from UML Diagrams Unified Modeling Language. This paper introduces an approach to detect clones in UML Class diagrams. UML Class diagrams are encoded as XML files. Tokens are extracted and matched using programming technique. This approach detect only exact matching classes which are 100% similar to one another in terms of attributes and operations.

Harjot Kaur et al.[3] proposed Detecting Clones in Class Diagrams using Suffix Array. This paper describes an approach to detect clones in class diagrams. Firstly class diagrams are encoded as XML files. Tokens are extracted and matched using Suffix array technique. This approach is based on finding similarities in tokens known as clones. Class diagrams contains redundant elements. Similar attributes or operations present in two different classes are known as clones.

Elizabeth P. Antony et al.[5] proposed Clone Detection in Behavioral Models. In this paper an approach has been introduced for reverse engineered UML behavioral models to detect near-miss interaction clones. Behavioural models are represented as XMI file. TXL is used to do transformation then normalization is performed. NiCad is used to do clone detection analysis. This approach has been detected type 3-1(exact near-miss) clones. The result obtained from NiCad Clone detector are presented in XML and HTML text formats.

Dhavleesh Rattan et al.[7] proposed Model Clone Detection based on Tree Comparison. This paper describes an approach to detect clones in UML diagrams. UML class diagrams encoded as XMI files. XMI file is filtered and formed as a tree. Then subtree comparison is applied to detect similarity between two class diagrams in a model and reported as a clone.

Harald Storrle et al.[8] proposed Towards Clone Detection in UML Domain Models. Clone detection algorithm is implemented as MQlone tool, a plugin in Magic Draw UML CASE tool which reports the clones to the user. Formal definition of models, model clones and implemented approach in the MQlone tool have been provided. The clone detection quality and runtime of algorithm were validated experimentally.

James R Cordy et al.[9] proposed The NiCad Clone Detector. This paper introduces that the NiCad Clone Detector is a flexible, scalable clone detection tool designed to implement the clone detection method in a useful manner. NiCad Clone detector involves three stages: 1) Parsing 2) Normalization 3) comparison. It provides output results in both XML and HTMl forms. NiCad is new clone detection method that has been used in detecting near-miss clones with high precision and high recall.

N. H. Pham et al.[12] proposed Complete and Accurate clone detection in Graph-based Models. This paper introduces ModelCD, a novel clone detection tool for Graph-based models that is able to detect both exact and approximate clones. The core of ModelCD is two clone detection algorithms i.e. eScan and aScan that are used to detect clones with a high degree of accuracy, scalability and completeness. Two algorithms detect clones through three steps: generating, grouping, and filtering. Empirical evaluation on large-scale Simulink systems has shown that it is able to handle both similar and exact-matched clones. Compared to CloneDetective tool ModelCD gives detection results with a much more quantity and high quality in reasonable running time.

Chanchal K. Roy et al.[15] presented A Survey on Software Clone Detection Research. This paper first, describes the clone types. Second, provide a review of the existing clone detection approaches and experimental evaluations of clone detection tools. Applications of clone detection research to other domains of software engineering and in the same time how other domain can assist clone detection research have also been pointed out.

IV. METHODOLOGY

Steps to detect clones from class diagrams and sequence diagrams:

A) Creating Class diagrams and Sequence Diagrams using UML tool.
B) XML file is generated from Class Diagrams and Sequence Diagrams.
C) Analysis of XML file &detection of all the tokens (i.e. classes, attributes, operations and messages).
D) Decryption of properties of found tokens (e.g. data type, visibility).
E) Use of data structure approach to compare retrieved tokens to detect clones.

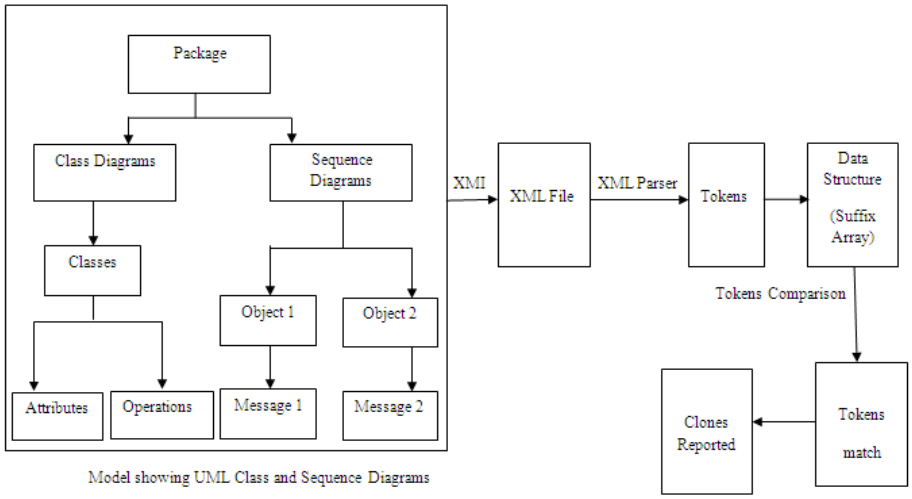

Fig. 2: Block diagram of clone detection in UML Sequence diagrams

V. EXPERIMENTATION

The technique is based on token based matching. Diagrams are exported into XML files. XML is used to represent large amount of data in expressive and flexible way. Every element of XML has an XMI Id identified by "xmi:id" and a set of attributes that shows the relationship of this element with the others.These XMI id's are basically used to extract the tokens. These extracted tokens are then compared with each other and clones are found.

V.I. Clone Detection in UML Class Diagrams

A) Class diagrams are generated using reverse engineering by the use of StarUML tool.

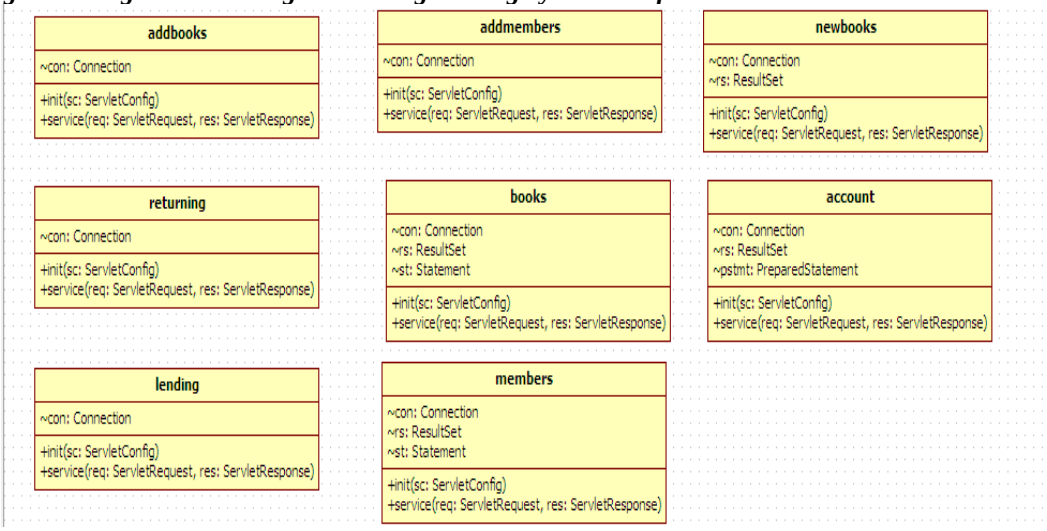

Fig. 3. Class Diagrams

B) Class diagrams are converted to XML file by using StarUML tool.

Fig. 4: XML file

C) Extraction of each class including their attributes and operations from the XML file.

```
addbooks
Attr: con
Operations: init
Operations: service

addmembers
Attr: con
Operations: init
Operations: service

books
Attr: con
Attr: rs
Attr: st
Operations: init
Operations: service

lending
Attr: con
Operations: init
Operations: service

members
Attr: con
Attr: rs
Attr: st
Operations: init
Operations: service

returning
Attr: con
Operations: init
Operations: service

newbooks
Attr: con
Attr: rs
Operations: init
Operations: service

account
Attr: con
Attr: rs
Attr: pstmt
Operations: init
Operations: service
```

Fig. 5: Classes with attributes and operations

D) Comparison and list of each class including their attributes and operations.

```
Cloned Classes Are
addbooks
addmembers
lending
returning

Cloned Classes Are
books
members


Non Cloned Classes Are:
newbooks
account



Attributes              Instance.
con             : 8
rs              : 4
st              : 2
pstmt           : 1



Operations              Instance.
init            : 8
service         : 8
```

Fig. 6. List of Cloned and non cloned classes

V.II. Clone Detection in UML Sequence Diagram

A) Sequence diagram is created through StarUML tool for Library Management System.

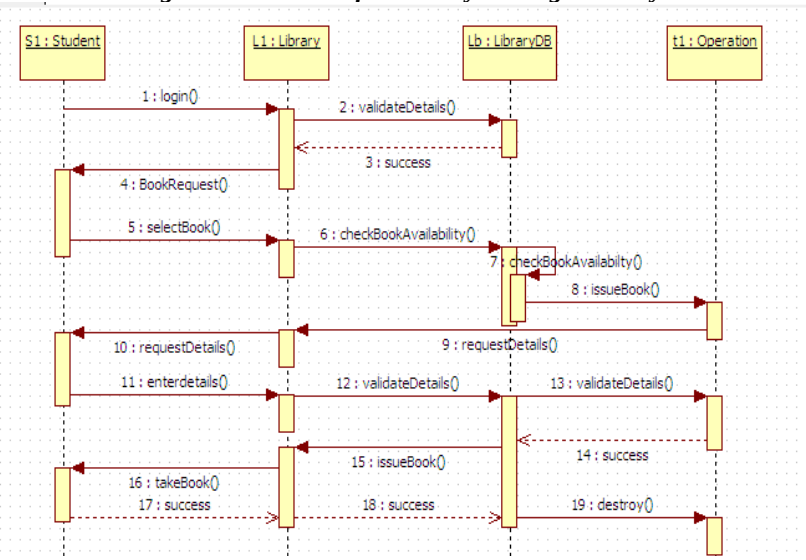

Fig. 7: Sequence diagram

B) Sequence diagram is converted to XML file by using StarUML tool.

Fig. 8: XML File

C) Extraction of each class including their messages from the XML file.

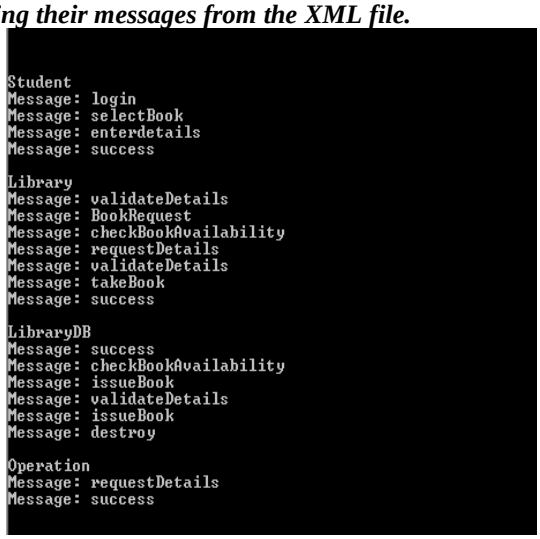

Fig. 9: Classes with messages

D) Comparison and list of messages

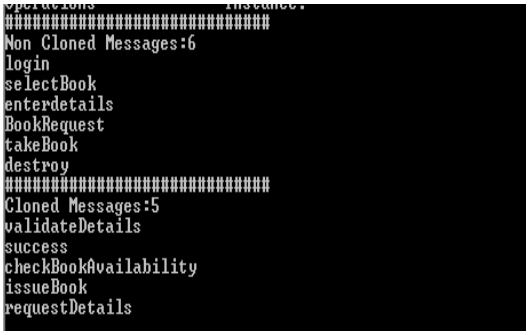

Fig. 11: List of cloned and non cloned messages

VI. RESULTS AND DISCUSSION

A) Detection of model clones in class diagrams

After formation of XML file , this file is compiled in Java Development environment by using Java programming. Programming is done in such a way that it extracts the list of Classes including attributes and operations from Class diagrams and gives comparison of each class with each other. If the Classes including the attributes and operations are same, they are listed as Cloned classes as shown in figure 6.

Here is the table 1 that shows the number of cloned and non-cloned classes that are found in Class Diagrams of Library Management System.

Table 1: List of Cloned and Non-Cloned Classes

Cloned Classes	**Cloned Classes**	**Non Cloned Classes**
addbooks	books	newbooks
addmembers	members	account
lending		
returning		

Instances of each attribute and operation are also found by using java programming as shown in table 2 and table 3.

Table 2**:** Instances of each attribute

Attributes	Instances
con	8
rs	4
st	2
pstmt	1

Table 3: Instances of each operation

Operations	Instances
init	8
service	8

B) Detection of model clones in sequence diagram

After formation of XML file, this file is compiled in Java Development environment by using Java programming. Programming is done in such a way that it extracts the list of Classes and messages from sequence diagrams and gives comparison of each message with each other. If the messages are same, they are listed as Cloned messages as shown in figure 11.

Here is the table 4 that shows the number of cloned and non cloned messages that are found in Sequence diagrams of Library Management System.

Table 3: List of Cloned and Non-Cloned Messages

Cloned Messages	**Non-Cloned Messages**
validateDetails	login

success	selectBook
checkBookAvailability	enterDetails
issueBook	BookRequest
requestDetails	takeBook
	destroy

Instances of each message are also found by using java programming as shown in table 5.

Table 5: Instances of each message

Messages	**Instances**
login	1
selectBook	1
enterdetails	1
success	4
validateDetails	3
BookRequest	1
checkBookAvailability	2
requestDetails	2
takeBook	1
issueBook	2
destroy	1

VII. CONCLUSION AND FUTURE SCOPE

Large adaptability of model based development in software field is promoting model based clone detection. In this paper we presented a way to detect clones in UML diagrams. The present work reports that class diagrams and sequence diagrams contains number of redundant elements. Similar attributes or operations present in two different classes are repoted as clones. Similar messages in sequence diagrams are also reported as clones. Our approach will work for all object oriented diagrams with the facility to export those diagrams to xml files using modeling tools. Our result has shown that there are number of clones that occur at multiple places hence it will support maintainability and data structure suffix array consumes less memory.

In future we will try to apply our approach to activity diagrams and state chart diagrams. Further we will compare this technique with data mining technique to detect clones in UML diagrams.

REFERENCES

[1] S. Sarala, S. Mythili, "Efficient weight assignment method for detection of clones in state flow diagrams", International Journal of Software Engineering Research & Practices, vol. 4, Issue 2, October 2014.

[2] Pamilpreet Singh, Harpreet Kaur, "Detection of clones from UML Diagrams Unified Modeling Language", International Journal for Scientific Research & Development, ISSN: 2321-0613, vol. 2, Issue 05, 2014.

[3] Harjot Kaur and Manpreet Kaur, "Detecting clones in class diagrams using suffix array", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958, Vol. 3, Issue 4, April 2014.

[4] D. Rattan, R. Bhatia, M. Singh, "Software clone detection: A systematic review", Information and Software Technology, pp. 1165-1199, 2013.

[5] Antony.E.P., Alafi.M.H., Cordy.J.R., "An approach to clone detection in behavioural models", Queen's University, Kingston, Canada, AAC-WCRE, 2013.

[6] M.H. Alafi, J.R. Cordy, M. Stephan, T.R. Dean and A. Stevenson, "Models are code too: Near –miss clone detection for Simulink models", IEEE, in ICSM, pp. 295-304, 2012.

[7] Rattan.D, Bhatia.R, and Singh.M, "Model clone detection based on tree comparison", India Conference (INDICON), IEEE, ISBN: 978-1-4673-2270-6, pp. 1041-1046, Dec. 2012.

[8] H. Storrle, "Towards clone detection in UML domain models", Software and Systems Modeling, doi 10.1007/s10270-011-0217-9, pp. 39, 2011.

[9] J.R. Cordy and C.K. Roy, "The NiCad clone detector", in Proceedings of the Tool Demo Track of the 19th International Conference on Program Comprehension (ICPC 2011), IEEE, Kingston, Canada, pp. 219-220, June 2011.

[10] Deissenboeck.F, Hummel.B, Pfaehler.M and Schaetz.B., "Model clone detection in practice", IWSC'10, Cape Town, South Africa, pp. 37-44, 2010.

[11] H. Storrle, "Towards clone detection in UML domain models", Proceedings of European Conference on Software Architecture (ECSA'10), Copenhagen, Denmark, pp. 285-293, 2010.

[12] Nam H. Pham, Hoan Anh Nguyen, Tung Thang Nguyen, Tien N. Nguyen, Jafar M. Al-Kofahi, "Complete and accurate clone detection in Grap-based Models", Proceedings of the IEEE 31st International Conference on Software Engineering (ICSE 2009), pp. 276-286, 2009.

[13] C.K. Roy, R. Koschke and J. R. Cordy, "Comparison and evaluation of code clone detection techniques and tools: A qualitative approach", Science of Computer Programming, vol. 74, no. 7, pp. 470-495, 2009.

[14] Lin.H.J. and Peng.L.F., "Quick similarity measurement of source code based on suffix array", IEEE, International Conference on Computatinal Intelligence and Security", DOI 10.1109/CIS.2009.175, 2009.

[15] C.K. Roy, R. Koschke and J.R. Cordy, "A survey on software clone detection research", Technical Report 2007-541, Queen's University at Kingston Ontario, Canada, pp. 115, 2007.

[16] S. Bellon, E. Merlo, J. Krinke, G. Antoniol and R. Koschke, "Comparison and evaluation of clone detection tools", IEEE, Transactions on Software Engineering, vol. 33, no. 9, pp. 577-591, 2007.

[17] F. Deissenboeck, B. Hummel, E. Juergens, B. Schatz, S. Wagner, S. Teuchert and J. F. Girard, "Clone detection in automotive model based development", Proceedings of 30th International conference on Software Engineering, Leipzig, Germany, pp. 603-612, 2008.

[18] Abdul.H.B., Puglisi.S.J., Smyth.W.F., Turpin.A. and Jarjabek.S., "Efficient token based clone detection with flexible tokenization", ESEC/FSE'07, ACM, Cavtat Croatia, 2007.

[19] H. Liu, W. Shao, L. Zhang and Z. Ma, "Detecting duplications in sequence diagrams based on suffix trees", IEEE CS, Proceedings of 13th Asia- Pacific Software Engineering Conference (APSEC'06), Bangalore, India, pp. 269-276, 2006.

[20] Abdul.H.B. and Jarzabek.S., " Detecting higher-level similarity patterns in programs", ESEC-FSE'05, ACM, Lisbon, Portugal, 2005.

[21] I.D. Baxter, L. Bier, M. Sant'Anna, L. Moura and A. Yahin, "Clone detection using abstract syntax trees", in ICSM'98, IEEE Computer Society, pp. 368-377, 1998.

