Enhanced UML Methodology with New Hybrid Diagrams: An ATM Application

Sherif Magdy Mohamed Abd Elaziz Barakat, Debashish Das*
Asia Pacific University of Technology and Innovation (APU), Kuala Lumpur, Malaysia
*debashish.das@apu.edu.my

Abstract: Object oriented proved to be the most important software engineering tools. The main reason behind the importance is that, Object Oriented covers the most powerful programming languages such as Java and C++. UML is a modeling language which is a part of object oriented and it was created to provide the software development industry with analysis and design techniques based on some diagrams. The most common UML diagrams in use are Class and Activity diagram. To implement any system using object oriented UML, the programmer needs at least two or three diagrams beside the Use Case diagram in order to understand a full picture of the system. There is no diagram which can show the full picture of the system. In this research, new methodology called "Touch and Go" has been generated by the researchers to implement the system. In this methodology, analysis and design are merged in one stage called Touch. Implementation and testing stage are merged in one stage called Go. The Class and Activity diagram are merged to introduce new integrated diagram called Enhanced Class Activity (ECA) diagram and the normal Activity diagram will be modified and improved as Activity Remarked Class (ARC) diagram. Analytical analysis of the new methodology is implemented in ATM machine system which can be later implemented using Java language or any other Object Oriented language. A powerful system like ATM is chosen to apply the new technique and it has been shown clearly how the methodology is working perfectly with such system. This study helps to reduce the number of diagrams that are being used to implement the system, reducing total time of implementation and making the coding stage easier by using only few numbers of diagrams.

Keywords: *UML, Touch, Go, Class Diagram, Activity, ARC Diagram, ECA Diagram, ATM System*

1. Introduction

People in this era always try to find the easiest and fastest ways to do things. Long time ago when the computer technology was introduce to human, basic software were developed in a slow pace. Forty years ago, a bigger team was needed in order to develop simple software for calculation purposes. At this time the programmers used the machine language and Assembly language. Nowadays, everything has been changed. Life started to be faster and easier than before. It takes shorter time to develop big software, maybe few months or weeks, and sometimes just days. There are many programming languages now to develop any software and definitely much easier than Machine language and Assembly. The major works of the researchers are always to find easy and fast way to develop the system. Twenty years ago, Object Oriented came to achieve this aim with some new programming languages such as Java and C++. UML comes with object oriented to construct the system. UML comes with some diagrams that help to analyze and design the system. The problem in UML is using many diagrams to develop the system. Using many diagrams formulate the issue serious and complex. There is no diagram in UML which can show the structural and behavioral side of the system together. By this way, developing any system should use many diagrams to show full aspect of the system. In this study, we generate new methodology that will replace SDLC to use only two main stages to develop a system. To apply this methodology, the natural UML diagrams cannot be used directly. The researcher makes some modifications for two UML diagrams to make them more suitable. The new enhanced diagrams will help to understand the system easier, faster and using less possible efforts. It will also help to reduce the complexity of the system. The researchers called the new methodology as Touch and Go. The new methodology will help to reduce the development time, the number of diagrams that needed to develop the system and the documentation as well. It will make the coding stage easier and faster to understand because it does not need to read many number of diagrams but only the new diagrams will be enough to do the implementation.

Background: UML started in 1990s by Grady Booch and James Rumbaugh. Object Oriented has five main concepts. Class and Object are the main concepts of Object Oriented. The rest of the concepts are Encapsulation, Inheritance, Message and Polymorphism.

System Development Life Cycle: Any events or action in this life cycle has to follow some steps to occur. By the same way to develop any system, it has to follow some steps started by planning the system and ended by testing the system. These steps to develop the system called system development life cycle (SDLC). The first stage is Analysis to capture all the required information of the system. The second stage is Design and it considers a joint stage between analysis and implementation stage. The third stage is Implementation to translate the design into programming language code. Testing stage is the last stage to ensure that the system is working properly without any errors.

UML (Unified modeling Language): UML (Unified Modeling Language) considers a technique to analyze, design and implement the system. To develop any system using UML should use some diagrams, each one describes a particular part of the system. The most famous diagrams used in UML are Use Case diagram, Class diagram and Activity diagram.

Use Case Diagram: Use Case usually comes in early stage of system development. It is used as analysis tool to show the functions on the system and also the entities.

Class Diagrams: Class diagram shows the structural part of the system. It considers the closest diagram to the coding syntax. Class diagram usually comes in design stage and becomes the first diagram in use for implementation stage.

Activity Diagram: Activity diagram shows the activities of the system and also conditions and iterations. Activity diagram helps the programmer to easily understand the flow of the system.

2. Literature Review

The major researchers work focuses on attempting to reduce the implementation time. Some of them try to find somehow to do the development of the system automatically. Most of the studies that have been done in UML field concentrates to Class diagram, because Class diagram has a special importance in UML. Some researchers concentrate in Activity diagram because it is the closest diagram to human language. Activity diagram is very common in use especially after Use Case.

Improvement of Previous Works: In this paper, a study of the class diagram attributes has been done. The result of the study has generated a new methodology called "semantic annotation" which is a file that will replace the normal class diagram. It works on extracting the class attributes, methods and association with the other classes. The new methodology makes the attributes of the class more detailed than the normal class diagram. The new class aspect is used in coding stage to make it easier and faster. After integrating the annotation file to UML model, the implementation stage will be more flexible (Wang and Li, 2010). A new model has been devised on Activity diagram to reduce the development time and early detection of faults. The new model is called "Test Case". The main work in this study is to generate test case from the activity diagram. The test case is designed to improve the design quality and identify the faults of implementation in early stage (Debasish and Debasis, 2009). A new study is required to make a formal grammar for three UML diagrams Use Case, Activity diagram and Class diagram. This study is designed for UML 2. The main aim of this study is to make these three diagrams in the formal aspect. The future work of this study is planning to develop a compiler based on the introduced grammar (Jayeeta, Ananya, Sabnam and Swapan, 2009).

A new study focuses more on consistency and behavioral analysis of Sequence diagram. Replacing normal sequence diagram with the new model "Semantic of Sequence diagrams" makes Sequence diagram more expressive and powerful and to make it available for any aspect of the system (Jing, 2009). Nowadays, web application is very commonly used for much business purposes. Web applications have much unexpected behaviors. The unexpected behavior of web application sometimes is not compatible with UML rules. This approach is to extend the UML diagrams to be more suitable and for web applications developing. Sequence

diagram is extended by tagging related web resource on each action and object. The extended diagram further investigates to generate requirements especially the security issue (Lin, Weifeng and Stephen, 2011). The main importance of the Class diagram is to represents the structural aspect of the system. The complexity of the Class diagram considers a serious issue for researchers. New model has been generated to assume some rules for transforming complexity value of classes and the class relations into weighted class dependent graphs (Dazhou, Baowen, Jianjiang and William, 2004). The new methodology based on getting the required information to develop the system using English language then the dynamic system will auto generate the required UML diagrams for this system such as Use Case, Activity diagram, Class diagram and Sequence diagram. After generating the UML diagrams, the dynamic system can generate the programming language code. The methodology can help in generating the graphical user interface of the system (Imran et al., 2006). New approach called "Code Generator" is used to generate the programming language code from class and sequence diagram. The limitation of that approach is that complete code cannot be generated (Abilio, Eliane and Lisane, 2011). Drawing UML diagrams automatically is ambitious for many researchers. A new methodology can generate class diagram from natural language. The new methodology uses a technique called "dependency graph" to capture the required information in natural language than auto generating the class diagram. Developer must be involved in refining the set of the class and it is considered as the limitation of this methodology. The new methodology in the future work can be extended to include the class relationships (Hema and Philip, 2009). The major UML problem is that the information is separated into several diagrams. SDSUM is a new algorithm. The new algorithm include a study constructs called MDG to synthesize the information easily for generating UML diagrams. MDG study is designed to integrate the structural and behavioral aspect of the system. SDSUM algorithm uses MDG study for computing the dynamic slices (Lallchandani and Mall, 2010).

Personal Review: Authors have some notices from the previous researchers' work related by class diagram and activity diagram:

The Class diagram complexity: Class diagram is a very important diagram, because it considers a junction diagram between design and implementation stage. Class diagram shows only the structural aspects of the system. It is impossible to develop any system using only class diagram. Class diagram comes as the first concentration of the researchers, always researchers try to modify it to make it more efficient.

Activity diagram: Activity diagram is behavioral diagram which shows the main activities of the system, iterations and conditions. Activity diagram does not have any structural aspects of the system. Any system cannot be developed using only with Activity diagram.

Class diagram and Activity diagram together: Using class diagram with activity diagram together makes the system more understandable. But developing the system still needs to use more diagrams to cover the rest parts of the system.

Research Question: From the previous works in UML field, authors assume the main question of this research: "Why UML does not have a fast technique to develop the system using few numbers of diagrams that can show both structural and behavioral part of the system?"

3. Methodology

From the previous research, researchers have found that the major concentration should be in improving UML diagrams and trying to make it easier and more efficient to be used. Class diagram is almost a joint work between researchers because it has the special importance from the rest of UML diagrams. Class diagram is also the closest diagram to the programmer, simply because it can be translated direct into programming code but Class diagram only can show the structural side of the system. Activity diagram is close to the normal human language and it is the first diagram that comes after the Use Case. Activity diagram can only show the behavioral side of the system such as the activities, conditions and iterations. Most of systems need both Class and Activity diagram together to give a full picture of the system and cover both the structural and behavioral sides.

The authors can summarize the main difficult points in UML as below:
- To do a good design of any system, it needs a lot of work through analysis stage.
- To implement big system, it needs to use many diagrams and it will make the implementation time longer and the documentation will get larger as well.
- To translate any system's design into programming language code, it needs to read many diagrams to be able to do the coding of this system.

From the research question the researchers started to think about finding somehow to generate a new diagram that can reply all the implementation needs as well as to show both structural and behavioral aspects of the system in one snapshot. Authors give the new methodology a name called "Touch and Go Technique".

Define Touch and Go Methodology: Touch and Go methodology as shown in Figure (1) is a new technique to develop the system based on two main stages, Touch and Go stage. Touch stage is an integrated stage between Analysis and Design stage. Go stage is an integrated stage between Implementation and testing stage. Touch and Go methodology uses new integrated diagrams. ARC and ECA diagram are new integrated diagrams used by Touch and Go methodology.
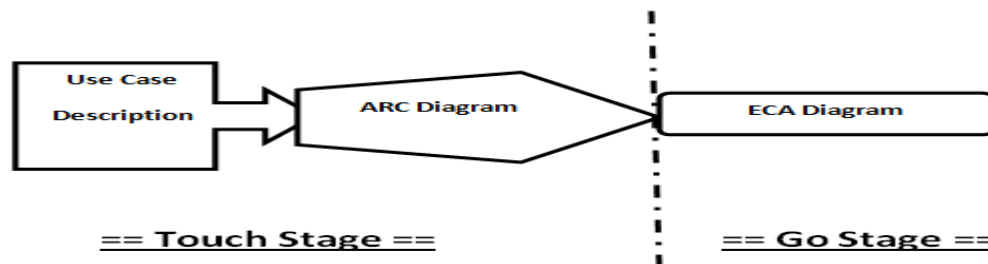
The Main Flow of the New Methodology
The new methodology based on two main steps:
- Use Case diagram and its description. Touch Stage
- Draw Activity Remarked Class (ARC) diagram
- Draw Enhanced Class Activity (ECA) diagram
- Do the coding using ECA diagram. Go Stage
- Testing the system
And the required diagrams for each stage:

Figure 1: Touch and Go Methodology

Activity Remarked Class (ARC): This diagram exactly looks like the normal Activity diagram plus only one thing that each activity has the class name.

Enhanced Class Activity (ECA): This diagram is a hybrid diagram comes from merging Class diagram with Activity Diagram. This diagram can show the structural aspect of the system exactly like normal Class diagram moreover it can show the behavioral aspect of the system acting like Activity diagram. This diagram is the main core work of Touch and Go technique.

**Apply Touch and Go Technique for ATM Machine system**

Touch Stage: The new methodology is available to use for any type of system whether small or large and available to implement using any Object oriented language such as C++, Java or C#.
The reasons why choosing ATM Machine System:
- ATM system is very common in use, popular and easy to understand.
- ATM system considers a big system and has many sub systems and requirements and by this way it is the appropriate system to test the author's new methodology.

- ATM system has many conditions and looping which will be great to be tested using the new diagrams.

Figure 2: Shows the physical hardware component of the ATM Machine
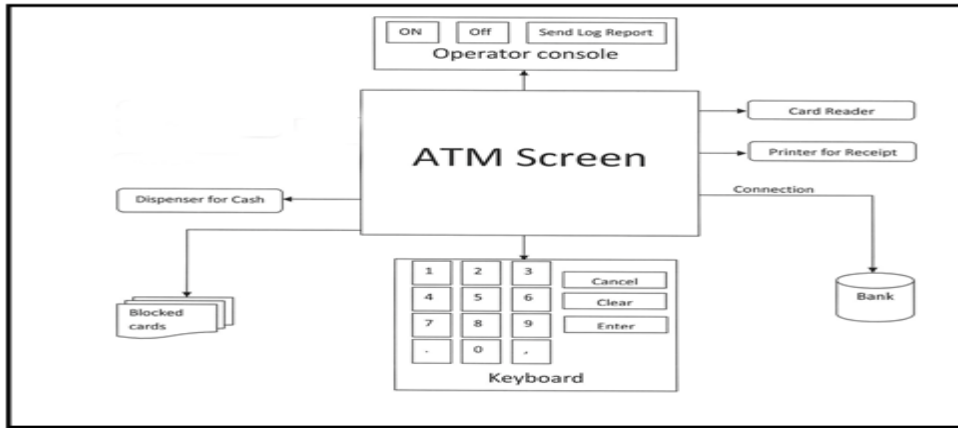
This study will focus on to the software part for ATM machine using the new designed methodology. This part is the core appliance of this study.

A. Determine the entities that involved in this system:
- The User who will do the transaction.
- The Operator who will be responsible to refill the machine and print the log file.
- The bank database will be considered as an entity.

B. Determine the functions of the system:
- Cash Withdrawal.
- Balance Inquiry.
- Update Info.
- Transfer.

C. Determine the main conditions of the system:
- Checking the validity of the card is condition to start the session.
- The maximum trials for any transaction should be three times only.
- After key-in the pin number three times wrongly the card will be blocked and held in the machine.
- In withdrawal the minimum amount can be set.

Use Case: Use Case as show in Figure (3) considers the initialization step of Touch and Go technique. Use Case must followed by the description that will be clear and accurate because it will be used later on to draw ARC diagram.
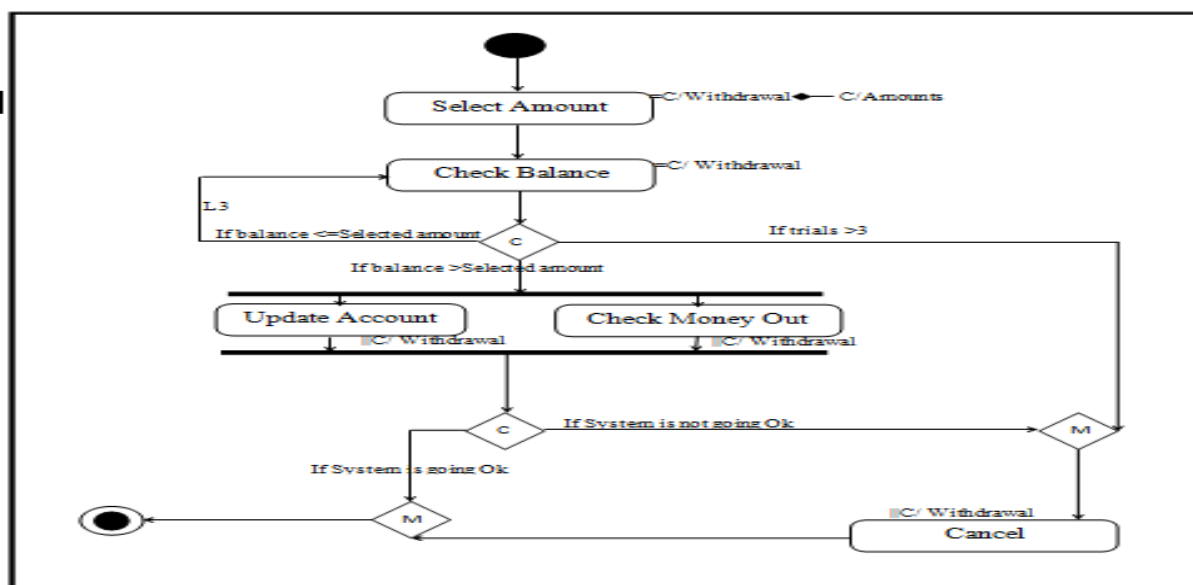
Figure 3: Use Case for ATM System

Activity Remarked Classes (ARC) For ATM System: ARC diagram can be drawn for whole the system but if the system is very large, in this case the system has to be divided into small parts then can draw ARC diagram for each part.

Figure 4: ARC Diagram for ATM System
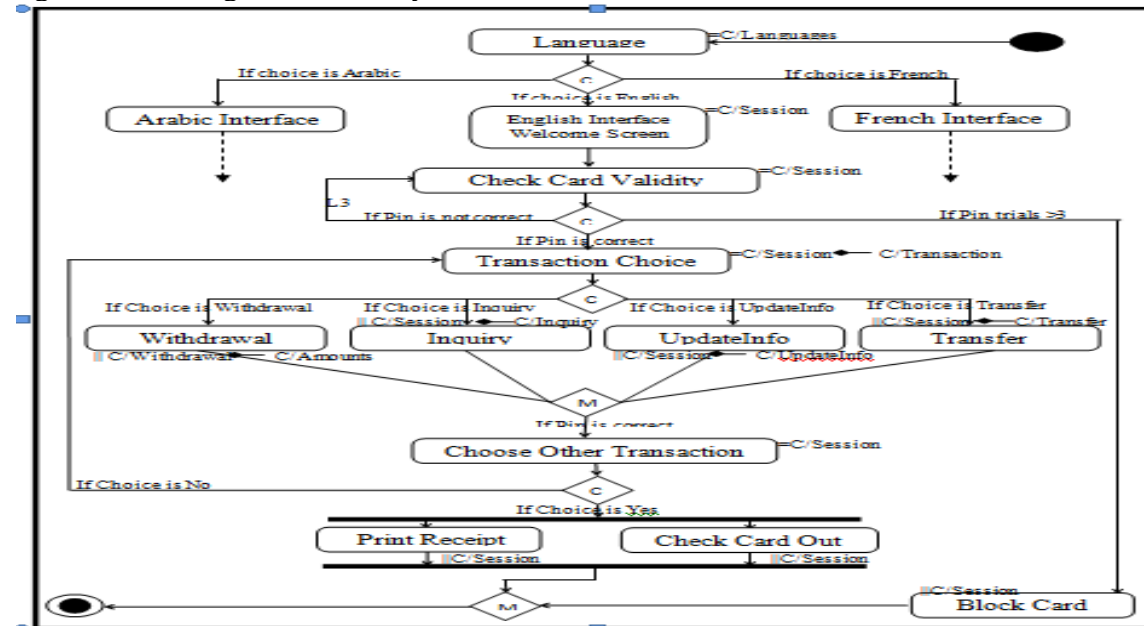
ARC diagram can be drawn for all ATM Machine System just to show the main data flow of the system. The second stage will be drawing ARC diagram for each part of the system. As shown in Figure (4), ARC diagram is acting exactly like normal Activity diagram, the only difference is that each activity is remarked by its class name. ARC for Withdrawal is shown in Figure (5):

Figure 5: ARC for Withdrawal in ATM System

Enhanced Class Activity Diagram (ECA): ECA diagram as shown in Figure (6) can be drawn for the whole system but if the system is very large, in this case the system has to be divided into small parts then can draw ECA diagram for each part.

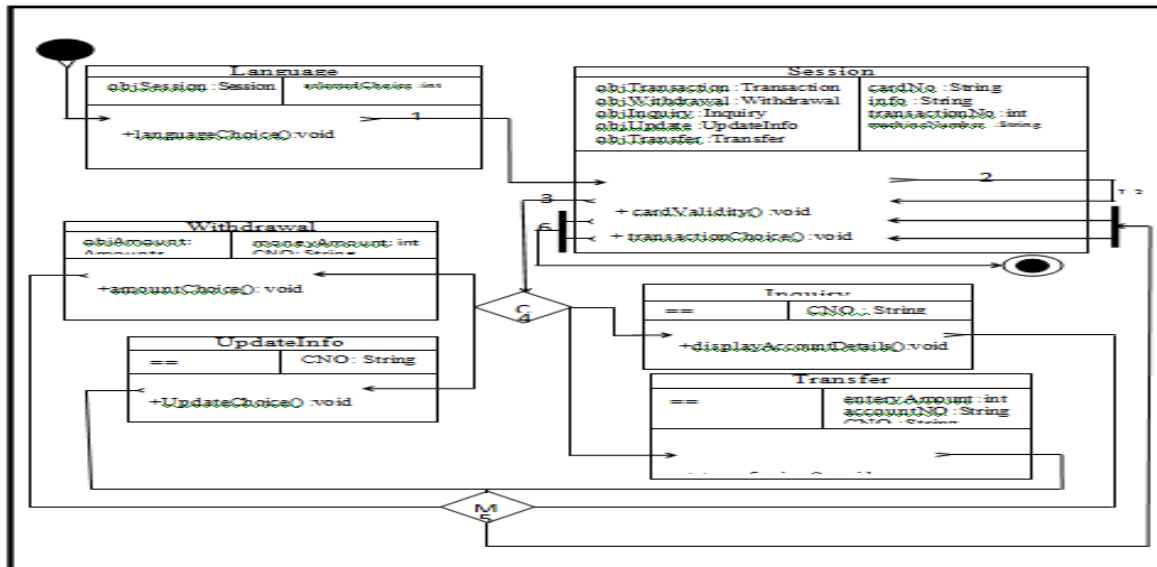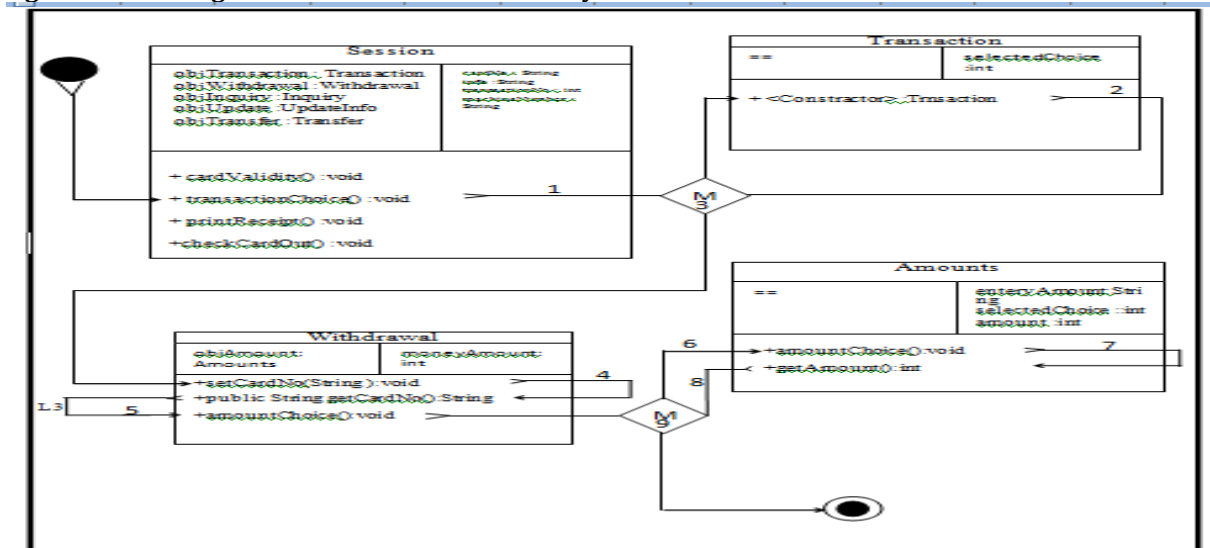**Figure 6: ECA Diagram for ATM System**

Figure (7) shows the ECA diagram for Withdrawal class. The ECA shows as well the relations between Withdrawal class, Session class and Amounts class.

**Figure 7: ECA Diagram for Withdrawal in ATM System**

Go Stage: In this stage it can be directly translated ECA diagrams into java/any other object oriented language coding to implement the system. The translation of the ECA diagram into Object Oriented code will pass three main steps:

- Zoom snapshot for ECA diagram.
- Direct implementation.
- Detailed implementation.

4. Results & Discussions

Result of the New Methodology: The main result of the new methodology is to reduce the number of the steps to develop the system from four steps into only two steps as shown in Figure (8).

66

Figure 8: SDLC and Touch & Go
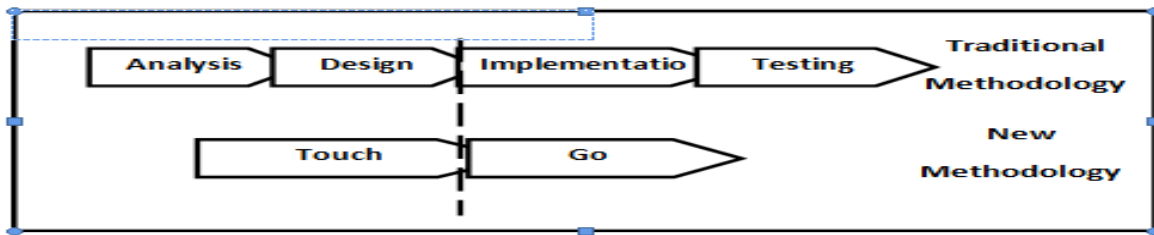
Result of New UML Integrated Diagrams: The main aim of the new methodology is to reduce the complexity and development time. To achieve this aim using Touch and Go methodology it needs to reduce the number of diagrams that is needed to develop the system. On the other hand, to develop a successful system it needs to use some of the diagrams that can give a full picture of the system. To achieve this goal, the researcher presents two new diagrams to be used in Touch and Go methodology. The new diagrams come from modifying two original UML diagrams.

Result of Integrated Activity Diagram into ARC Diagram: ARC diagram is a new diagram that has some new features to be more effective and powerful than the normal Activity diagram. ARC diagram shows clearly each activity belongs to which Class. By this way the Analyst and Designer are going to work together in Touch stage. ARC diagram presents a new way to early expect of the required classes to implement the system. ARC diagram shown in Figure (4) remarks the conditions and merge using M or C letter to make the diagram as easier to read.

Result of Integrated Class Diagram into ECA Diagram: ECA diagram is a new diagram comes from merging between Activity and Class diagram. ECA diagram (Figure (6)) shows the structural and the behavioral side of the system. ECA diagram is created in Touch stage but will be used in GO stage to be translated into programming language code. ECA diagram shows clearly the iteration and conditions. How the new diagrams are better can be explained by the following tables:

Table 1: ARC and Activity Diagram Difference

Table (1)	ARC Diagram	Activity Diagram
The number of activity	Same	
Show the class name of the activity	Show the activity belong to which class	Only shows the activity
Easy to read merge and condition	Very easy to recognize because M meaning merge and C meaning condition	Difficult to recognize unless read the all the flow of diagram
The iterations numbers	Shows how many time the loop will happen	Does not show the number of the iteration

Table 2: ECA and Class Diagram difference

Table (2)	ECA Diagram	Class Diagram
The number of activity	Same	
The order of the methods	Show the methods of the class in order based on which one will run first	The order is not important
Iterations and conditions	Show very clearly the iterations and the conditions	Does not show
The start and the end of the method	Shows the life time of the method as well as it starts it will be terminated and can show the activity of the method in sequence	Does not show

5. Conclusion

The traditional way to develop any software still has a lot of difficulties, analysis stage is normally based on human language, however implementation stage is based on programming language and design stage comes in between. In this study, the researchers come with the new methodology called Touch and Go. The main aim of that methodology is to close the distance between Analysis and Implementation stage. Touch and Go methodology comes with new technique merging Analysis and Design stage in one stage Called Touch stage. Implementation and Testing stage will be merged together in one stage called Go stage. To apply the new methodology it needs some modifications in UML diagrams to be compatible with the new methodology. The researchers come with two integrated diagrams called Activity Remarked Class (ARC) and Enhanced Class Activity (ECA) diagram. The new methodology with new UML diagrams will be more efficient and powerful than the traditional way. This study focuses to the technical side only. Applying this study for ATM machine system is just strong and powerful example to show clearly the flow of the new methodology but this methodology can be applied for any other system and it can be further implemented using Object Oriented Language like Java, C++, C# etc. The final result of this study is generating new methodology using new diagrams. The main aim is to reduce the complexity, the total implementation time and the efforts to develop the system. The future work of this study is to implement IDE called "Touch and Go". This IDE will be able to do following tasks:

- Capture the system information and the required functions using normal English language to auto generate Use Case.
- Based on generated Use Case the IDE will generate ARC and ECA diagram.
- From ECA the IDE will be able to generate Object Oriented Language code for the system classes include the graphical interface of the system.
- The IDE will be able as well to do the testing stage after code generation.

References

Abilio, G. P., Eliane, S. & Lisane, B. (2011). Generating Java code from UML Class and Sequence Diagrams. Brazilian Symposium on Computing System Engineering, 99-101.

Dazhou, K., Baowen, X., Jianjiang, L. & William, C. C. (2004). A Complexity Measure for Ontology Based on UML. IEEE International Workshop on Future Trends of Distributed Computing Systems, 417-423.

Debasish, K. & Debasis, S. (2009). A Novel Approach to Generate Test Cases from UML Activity Diagrams. ETH Zurich on Journal of Object Technology, 65-83.

Hema, K. & Philip, S. (2009). Relative Extraction Methodology for Class Diagram Generation using Dependency Graph. IEEE, 815-820.

Imran, S. B., Imran, S. M. & Abbas, C. M. (2006). Rule based Production Systems for Automatic Code Generation in Java. IEEE, 2006, 300-305.

Jayeeta, C., Ananya, K., Sabnam, S. & Swapan, B. (2009). Traceability of Requirements and Consistency Verification of UML Use Case, Activity and Class diagram: A Formal approach. International Conference on Methods and Models in Computer Science, 1-4.

Jing, Y. (2009). Analyzing UML Sequence Diagrams with UTP. International Conference on Frontier of Computer Science and Technology, 417-423.

Lallchandani, J. T. & Mall, R. (2010). Integrated state-based dynamic slicing technique for UML models. IET Software, 55-78.

Lin, D., Weifeng, X. & Stephen, F. (2011). A Resource-based Approach to Extend UML Diagrams for Web Applications. IEEE, 103-106.

Wang, Y. & Li, H. (2010). Adding Semantic Annotation to UML Class Diagram. International Conference on Computer Application and System Modeling (ICCASM), 187-190.