International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 10, October-2021

HMSC: A Visual Paradigm to Capture the Software System Requirements

Anup Acharya
School of Engineering, Pokhara University,
Nepal

Abstract:- **In this day and age, we are all living in the digital world. The digital world can be defined as the world where we live and work using digital tools as well as technologies. When we talk about digital tools, the computers are always come first. The computer contains hardware and software components. We can perform task using computers according to our needs with the help of software component for this it is necessary to develop software to meet our requirements. Although we have many software development organizations equipped with latest tools and techniques to develop software in this day, still it is very difficult to get successful product due to various factors among them lack of collection of proper software requirements that meet the actual needs of user is major one. Even we have many tools and techniques to collect requirements, still are not able to collect actual needs of user. This necessitates a kind of technique to collect software requirements that meet the user's need as far as possible.**

This article aims to identify an attractive visual technique to capture the requirement specifications of the software system that meets the needs of client to develop a successful software product in order to reduce software failure. Further, recommends a concept of visual paradigm to show; how that technique called High-level Message Sequence Chart (HMSC), ITU-T, Z.120 standardized language, is used to collect and define the specifications of the software system, which is demonstrated by means of "Automated Teller Machine (ATM)".

Keywords:- High-level Message Sequence Chart (HMSC), Requirement, visual paradigm, ATM, software system/product, task, client, developer

1. INTRODUCTION

1.1 Background

Software development task is continued to be dynamic. Although developers look as well as uses new approaches, tools and techniques for successful products, are still able to deliver very small number of successful products. Collection of system requirements to meet the client's need have always been the major problem for the development of successful software product.

Software development task has got the phases from Requirement collection/ analysis to Maintenance and is called system development life cycle (SDLC). Among them, the Requirement Analysis is the first, most vital and most difficult phase to carry out. It is interesting to note that the remarkable number of software failures have concerned with this initial phase of Software Development i.e., Requirement Analysis because of followings:

- It is difficult for clients to express their requirements in the appropriate way so that developer realizes it.
- Little numbers of clients may able to express their requirements but that may not be related to what they really want.
- Numerous clients may have the rough idea about their requirements but not specific on it.
- Developer may have little familiarity or no information about the client's domain.
- Developer does not want to express his/her weakness about knowing the domain of others because of losing reputation.

These lead the misinformation and misinterpretation between the developer and client. Due to which, a developer cannot collect the required information about the software system and this results in the consequences of client's un-satisfaction, over budget, software delay, software failure etc.

In order to develop successful software product these problems need to be addressed. Many efforts have been designed and used to resolve these problems but cannot get significant result. High technical and complicate in nature of these efforts make the client away rather than close. Hence, we need a kind of model that should be very simple and has a capacity to communicate between client and developer. Client also should be able to understand the technique and at the same time developer also should be able to extract the information from the client. The simple and powerful capability of expressing characteristics features of High-level Message Sequence Chart (HMSC) makes the possibility of playing the role to resolve these problems.

1.2 HMSC

ITU-T [1] stated High-level Message Sequence Chart (HMSC) is the graphical representation of overall concept of working scenario of a system. An HMSC contain msc name and a connected graph with in a frame; where each node may be one of the start symbols, an end symbol, an MSC reference, a connection point, a condition or a parallel frame as shown below:

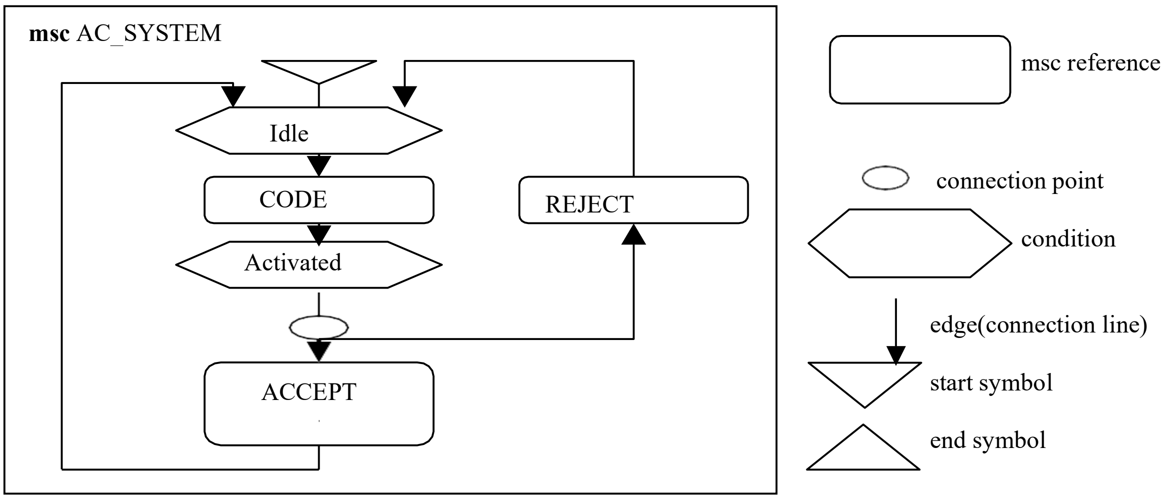

Fig 1: Introductory example of HMSC

The Fig 1 is an HMSC with its basic symbols that defines overall scenarios of access control system. Where the user provides his/her specific code to the system, in its idle state, which then move to the activated state. From that state the system either accepts the code i.e., access granted or rejects the code i.e., access not granted and then again move to the idle state.

1.3 Problem Definition

The software system development task begins with the collection of software system requirements from the client. Both, developer and the client need to take an active role in this stage. After that, the developer proceeds ahead for developing Software Requirements Specification (SRS). SRS is a document containing a complete description of requirements of the software system to be developed.

Although it is the preliminary stage but most valuable as well as significant stage of the software development process because the software is developed on the basis of the provided requirements specification of the system and is provided by the client. It is very difficult to collect actual need of the clients, if one cannot collect and interpret actual needs of the client then it is not possible to develop the system to meet the client's need and there is a chances of software failure. To reduce such type of software failure it is necessary to gather and understand the actual needs of the client, for which this visual paradigm called HMSC become a milestone because the pictorial form provide the more attractive and interactive environment than that of textual form to understand the problem. Further, this paradigm has nominal notations and rules, so easy to interpret the requirements.

2. REVIEW OF RELATED WORKS

A number of papers, Journal and articles related to the theme HMSC/MSC with its use have been examined thoroughly, such as the work of:

According to ITU-T [1], an HMSC (High-level MSC) is the graphical representation of overall concept of working scenario of a system.

Anup Acharya [2]: This is the age of Information Technology and it would not be an exaggeration to say that no any aspect of our social life is untouched by information technology because of its ability to collect, preserve and manage information in one touch. As a result, today's world moves toward this technology rigorously and this is known as digital world.

Anup Acharya [3]: Trace an HMSC for the system first. If the client satisfied with that, then give the Basic MSC because if the developer goes with the Basic MSC first, it becomes more difficult, time consuming, lengthy, boring. Due to which, client cannot understand it and is again results in software failure.

S. Leue, P. Ladkin [4]: We propose a semantics for Message Sequence Charts (MSCs). Our requirements are: to determine unambiguously which execution traces are allowed by an MSC; and to use a finite-state interpretation. Our semantics handles both synchronous and asynchronous communication.

Rajeev Alur, Gerard J. Holzmann, and Doron Peled [5] stated Message sequence charts (MSCs) are used in the design phase of a distributed system to record intended system behaviors. They serve as informal documentation of design requirements that are referred to throughout the design process and even in the final system integration and acceptance testing.

Grady Booch[6]: The problems we try to solve in software often involve elements of inescapable complexity, in which we find a myriad of competing, perhaps even contradictory, requirements. Consider the requirements for the electronic system of a multi-engine aircraft, a cellular phone switching system, or an autonomous robot. The raw functionality of such systems is difficult enough to comprehend, but now add all of the (often implicit) nonfunctional requirements such as usability, performance, cost, survivability, and reliability. This unrestrained external complexity is what causes the arbitrary complexity about which Brooks writes.

3. OBJECTIVE

The aim of this study is to propose HMSC as a visual paradigm to capture the requirements of the system according to the user's need in order to develop the successful product.

4. CONCEPTUAL FRAMEWORK

The study used the concept and notations of HMSC to evaluate the proposed paradigm, and is demonstrated by means of a real-world event an Automated Teller Machine (ATM). The basic idea about the HMSC and its notations are described hereabove in sub-section **1.2 HMSC**.

This section provides the procedure for tracing HMSC to capture system requirements:

- Find out abstract reactive sub-systems from the specification document consider them as a MSC reference.
- Find out the state of that system after each abstract reactive sub-system and considers them as a global condition.
- Give appropriate identification for the MSC reference and the condition according to the system.
- Give the system name as the MSC name then draw a connected graph within a frame.
- Connected graph always start from the initial condition with start symbol.

The procedure for tracing HMSC to capture system requirements can be summarized as:

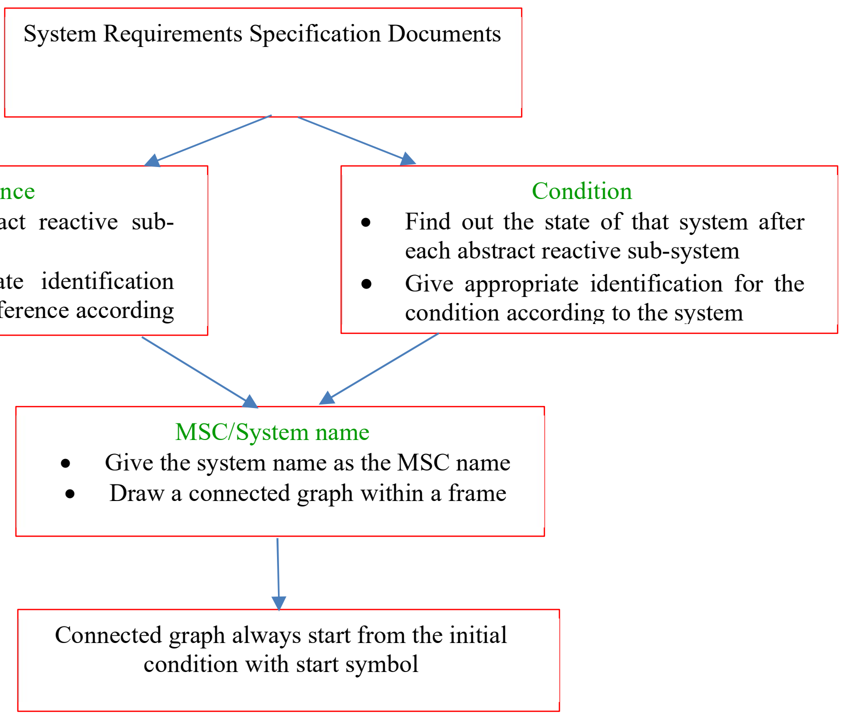
System Requirements Specification Documents
MSC reference
Find out abstract reactive sub-systems
Give appropriate identification for the MSC reference according
Condition
Find out the state of that system after each abstract reactive sub-system
Give appropriate identification for the condition according to the system
MSC/System name
Give the system name as the MSC name
Draw a connected graph within a frame
Connected graph always start from the initial condition with start symbol

Fig 2: HMSC Tracing Process Diagram

5. A PROPOSED VISUAL PARADIGM

The client meets the developer with the need and they come under dialogues with requirements. After knowing the requirements from the client, the developer makes a requirement specification document. Then the developer traces an HMSC for the system on the basis of specification and deals it with the client. If the client is satisfied, then the developer proceeds ahead on the way of development process. Otherwise, modifications are made to meet the client's need and proceed ahead.

The application of HMSC to capture and define the software requirements of a system will illustrate with a case study related to real world problem.

5.1 Case Study

The task is to develop software system for Automated Teller Machine (ATM).

5.2 An ATM

ATM stands for automated teller machine is an electronic device used for banking outlets that allows user to deposit or withdraw money, to check available balance from the bank account without human teller.

A user communicates with ATM using ATM card. An ATM equipped with a card reader slot having magnetic stripe reader for reading an ATM card, a keyboard and screen for interaction with the user, a slot for depositing envelopes, a dispenser for cash, a slot having printer for printing user receipts. Further, ATM will communicate with the bank through computer over an appropriate network.

User inserts an ATM card when the screen displays a standard Welcome Message. After verifying the card, system asks for Personal Identification Number (PIN). Then the system displays the service menu after verifying the PIN and the selected service is provided.

5.3 Preparing Software Requirements Specification

Anup Acharya [2] stated that, after collecting the system requirements from the client, the developer will try to convert this into the specification document as:
User required to insert an ATM card provided by the user's bank into the card reader slot when the screen displays a standard Welcome Message (idle state) to use an ATM. Then the machine becomes activated and the card reader attempts to read the inserted card. Unable to do so will informed the client and the card is ejected, the ATM once again becomes idle. Otherwise, information to enter the Personal Identification Number (PIN) was displayed.

The ATM verifies the PIN number from the bank when the four digited PIN is entered. The user required to re-enter the PIN number if the PIN is invalid, unable to enter the valid PIN after three tries will causes the ATM to keep the card and the information to contact the bank will be displayed. Then the machine again becomes idle.

For valid PIN, the ATM displays the menu containing list of transaction options that can be performed and the options are:

- withdraw funds
- balance inquiry
- deposit funds
- transfer funds

The user can select an option and perform the transaction. After completion of that transaction the user is asked for another transaction. If required, the system returns to the main menu. Otherwise, ejects the card and becomes idle again.
The communication with ATM can be canceled by user at any time during the entering a PIN or selecting a transaction option and is results in return the ATM in idle state after ejection of the card.
Cash can be withdrawn on selection of withdraw type, the user is asked to specify the account type from which the funds are to be withdrawn and the amount of the withdrawal. If the account contains sufficient funds, the fund is dispensed through the cash dispenser. Otherwise, the user is informed and asked to enter a different amount or cancel the transaction.

Balance of an account can be inquired on selection of balance inquiry option, the user is asked to specify the account whose balance is requested. The balance is displayed as well as printed on the receipt.

Cash can be deposited on selection of deposit option, the user is asked to specify the account to which the funds are to be deposited and the amount of deposit. Further, asked to insert a deposit envelop.

Funds transfer between any two accounts on selection of transfer option is also possible, the user is asked to specify the account from which the funds are to be withdrawn, the account to which the funds are to be deposited, and the amount of the transfer. If sufficient funds exist, the transfer is made, informed otherwise.

The ATM ejects the card and return to the idle state in the case of do not making a move within 30 seconds in each interaction by the user.

5.4 Representing Requirements with an HMSC

The software developer will try to convert that textual requirement into graphical format using High-level Message Sequence Chart (HMSC) by identifying the list of reactive sub-system from the system requirements provided by the client as:

- Insert
- Invalid
- PIN
- Check PIN
- Reject
- Menu
- Service
- Cancel
- Option
- Stop

Further, states are found as:

- Initial Condition
- Card validation
- Waiting for Pin
- PIN validation
- Waiting for user's need
- Provide user's need
- Asking about the other need

After giving the appropriate identification for the selected sub-system and the state, consider them as MSC references and the conditions respectively, an HMSC for the ATM can be traced as:

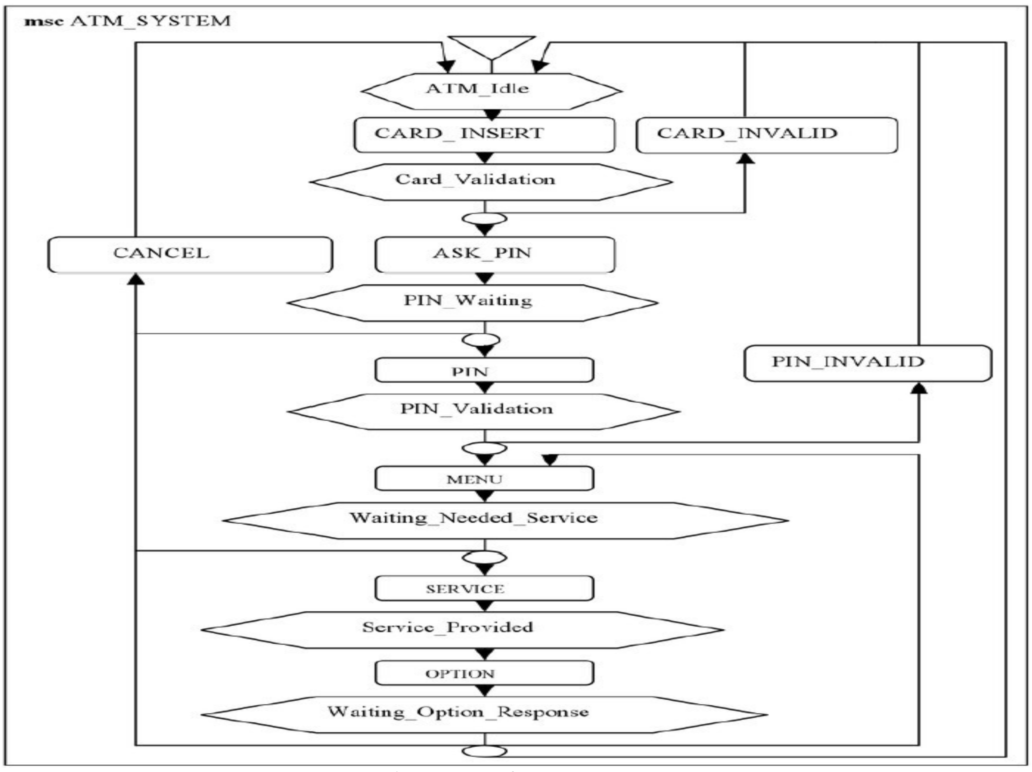

Fig 3: HMSC of ATM_SYSYEM

The developer can capture, understand and realize the clear image about the client's need with the help of this visual form. If both are satisfied, then the developer moves ahead on the way of development process considering this trace as an analytical view of the software system to be developed.

6. CONCLUSIONS

An HMSC is a graphical technique having simple and minimal set of symbol, syntax and semantics, so it is very easy to realize and use. Developer can capture, understand and define the software system requirements from the client with the help of HMSC easily.

Using HMSC developer not only able to capture the system requirements but also can verify them along with the client. Client will not be found difficult to understand the communication behavior of the HMSC and will be able to evaluate his/her requirements in a very friendly environment. If the client is not satisfied, he or she can immediately object the scenario. This makes the developer to achieve more and results in reduction of software failure. These simple features of HMSC encourage the client to take the active part in the development task, and decrease misinformation and misinterpretations about the system requirements. This helps developer to capture software system requirements that meet the client's need in pictorial form and also provide the logical view of the system to be developed in the pictorial format. Hence, an HMSC can be used as a visual paradigm to capture the system requirements as per the client's need.

REFERENCES

[1] ITU-T, TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU, *https://www.itu.int/ITU-T/studygroups/ com10/ languages/Z.120_1199.pdf.*

[2] Anup Acharya, "PRE-DIGITIZATION: A ROADMAP FOR CONTENT DEVELOPMENT TO FACILITATE DIGITIZATION", Volume. 9 Issue. 9, - 2021, Global Scientific Journals (GSJ), www. globalscientificjournal.com. ISSN - 2320-9186, PP: -1722-1729

[3] Anup Acharya, "MSC: The Bridge between the Client and Developer", Volume. 4 Issue. 1, - 2019, International Journal of Innovative Science and Research Technology (IJISRT), www.ijisrt.com. ISSN - 2456-2165, PP: -261-272.

[4] S. Leue, P. Ladkin, "What do Message Sequence Charts Mean?", Proceedings of the Sixth International Conference on Formal Description Techniques, North-Holland, 1994

[5] Rajeev Alur, Gerard J. Holzmann, and Doron Peled, "An Analyzer for Message Sequence Charts. Software / Concepts and Tools", 1996.

[6] Grady Booch, "Object-Oriented Analysis and Design. With Applications", Addison-Wesley, 2nd edition, 1994.

[7] Ekkart Rudolph, Jens Graboski, and Peter Graubmann," Tutorial on Message Sequence Chart(MSC'96).

[8] Requirements document for an automated teller machine network, (aug 5, 1996), http://www.cs.umd.edu/~mvz/cmsc435-s09/atm.pdf

Author

Anup Acharya
Assistant Professor
School of Engineering
Pokhara University
Pokhara, Nepal.