***Journal of the Nigerian Association of Mathematical Physics***
***Volume 30,* (May, *2015*), pp263 – 266**
***© J. of NAMP***

# Formalization of Hostel Management System.

*[1]Obi J.C. and [2]Imianvan A.A. and [3]Iyamu Iziegbe*

**[1,2]Department of Computer Science, University of Benin, P.M.B. 1154, Benin City. Nigeria.**
**[3]Department of Computer Science, Edo State Institute of Technology and Management, Usen.**

## *Abstract*

***The automation of the processing and activities of Hostel Management System (HMS) can invariably contribute greatly to the success, profitability and customer-based approach of such an organization. The use of formal specification creates a formal approach for specifying the underlying functions and properties of the system. This paper has attempted to give a formal description of the activities of HMS system Using Zed notations. The interaction within the system is visualized using Unified Modeling Language (UML) sequence diagrams.***

**Keywords:** HMS, Z-Notation, UML.

## 1.0 Introduction

In most scenarios, a hostel is seen as a budget-oriented, shared-room (dormitory) accommodation that accepts individual travellers or group of personnel's for a short period of time which invariably provides common areas and communal facilities. In some countries the word hostel can also refer to student accommodation or long-term accommodation for drug addicts or the homeless. The management and operation of the Hostel Management System (HMS) were previously handled manually, but with the passage of time automated processes were initiated to ease operational hazard and workability and were subsequently known as a Hostel Management Software System (HMSS).The Hostel Management Software System (HMSS) is geared towards maintaining all the hostel records of the customer and personnel. This software is usually designed specially to cater for the various departments operational within a well-established hostel. HMSS is usually equipped with various interactive features which are really simple for anyone to use. The HMS formalization is organization dependent. Therefore the properties (operations) of one organization usually vary in scope based on top management procedures and policies within the individual organization. It is quite possible for certain aspect of the system to been transferable to other organization systems. This includes user registration names. For all organization should been able to provide details about organizational employees names, ages gender and even age. The HMSS should be able to provide

a. Customer Registration
b. Ascertain availability of Room service or dormitory

The goal of this paper is to use the rich facilities of formal method to specify HMSS.

## 2.0 Material and Methodology

## 2.1 Materials

In achieving our objective of using formal methods in the functionalities of Hostel Management System (HMS), Z-notation and Unified Modelling Language (UML) serves as the main tools for the methodology.
Z-notation uses mathematical notation to describe in a precise way the properties a software system must possess, without unduly constraining the way in which these properties are achieved **[1 – 3]**. Formal specification (Mathematical notation or Z) uses mathematical data types to model data in a system and achieve it underlining objectives. These data types are not oriented towards computer representation, but they obey a rich collection of mathematical laws which make it possible to reason effectively about the way a specified system will behave. The notation of *predicate logic* is used to describe abstractly the effect of each operation of the system, again in a way that enables reasoning about their behaviour.
The other main ingredient in Z is a way of decomposing a specification into small pieces called *Schemas*. By splitting the specification into schemas, it can be presented piece by piece. Each piece can be linked with a commentary which explains informally the significance of the formal mathematics. In Z, schemas are used to describe both static and dynamic aspects of a system **[3]**. The static aspects include:

Corresponding author: Obi J.C., E-mail: tripplejo2k2@yahoo.com, Tel.: +234(0)8093088218 & 07069742552(I.A.A)

a. the state it can occupy;
b. the invariant (quantity that is unchanged by a set of mathematical operation) relationship that are maintained as the system moves from states to state.

The dynamic aspect include:
a. the operation aspect that are possible;
b. the relationship between their input and outputs;
c. the changes of state that happen.

The use of schema in this paper provides an avenue wherein our formal specification could be presented in fragment enabling us to associate commentaries; explaining informal the significance of the formal mathematical notation representation.

## 2.2 Methodology

The following are some of the basic types in Z
{CHAR, STRING, CURRENCY, QUERY, OBJECT, COMPONENTS,
BOOLEAN::= TRUE/FALSE, DATA and OBJECT}
The Hostel Management System (HMS); authenticate each user using his username/ID and password on the system

SystemUser
SystemUser_name/ID: seq CHAR
SystemUser_password: seq CHAR
SystemUser: ℙ System User
Access! : Boolean
SystemUser∈ SystemUser.access! = accepted) ∨
(SystemUser∉ SystemUser.access! ≠ accepted)

***Figure 1:*** *SystemUser Schema*

Figure 1 highlights clearly no frontier to the number of registered system user the Hostel Management System can ascertain and each system user can have only one authentication and authorization privilege. Logging on, each system user must register his username ID and password.

HMSystemList
Functionalities: ℙ HOSTEL FUNCTIONALITIES
Services : ℙ SERVICE
SystemList: HOSTEL_FUNCTIONALITIES → SERVICE
HMSystems = dom list

***Figure 2:*** *Hostel ManagementSystem List Schema*

Figure 2 highlights the HM*Systemlist*specifies the hostel functionalities provided by the system.

Register Hostel functionalities
Δ HMSystemList
SystemList? :HOSTEL_FUNCTIONALITIES
Service? : SERVICE
Hostel Functionalities:Hostel Functionalities
report! : REPORT
(SystemList? ∉HOSTEL_FUNCTIONALITIES∧HMSystem List′ =HMSystemList U {SystemList? →Service?}∧ report! = ok) ∨(SystemList? ∉SystemList∧HMsystemlist′ = HMsystemlist∧ Report! = already_known)

***Figure 3:*** *Register Hostel Functionalities Schema*

Figure 3 highlights; the list braces the facility in registering hostel functionalities given that the system process does not exist previously. If the process exists previously, a report of ‘already known’ is returned or vice versa as the case may be.

Locate Hostel Process
Ξ HMSystemList
Processes? : SERVICES
FaciltyList?ℙ HOSTEL_FACILITIES
(FacilityList! = {SystemList: SystemList / list (Hostel_Functionalities) = process?}∨ (system ∉Hostel_Functionalities∧ report! = not_known)

***Figure 4:*** *Locate Hostelprocess Schema*

Figure 4 highlights the *Locate Hostel Process* function obtains a service type as an argument and returns all the HMServiceList that offers those services and facilities in a service list.

```
AlreadyExisting Hostel Process
    Ξ HMSystemList
SystemList? : HOSTEL_FUNCTIONALITIES
    result! : REPORT
    process? ∈ Processes
    ─────────────────────
    result! = already_known
```

***Figure 5:*** *AlreadyExistingHostel ProcessSchema*

The *AlreadyExistingHostelProcessSchema* determines if there is a change to the systemlist in terms of new input process already. If it is the result, it will reply already_known otherwise not known. Figure 6 Exemplified schema processes

The list is initialized at the beginning with no process or hostel process as presented in Figure 6.

```
Hostel Initialize_List
    PROCESS: ℙHOSTEL PROCESSES
    ─────────────────────
HMsystemlist = ∅
    Hostel_process = ∅
```

***Figure6:*** *Initialize_List Schema*

## 2.3 System Design and Unified Modeling Language (UML)

Software design immediately follows the requirements engineering phase in a software process. Software design is the translation of the requirement specification into useful patterns for implementation. Unified Modelling Language (UML) is a standard modelling language used for modelling software systems. We use UML for design of the hostel system process because UML focuses on creating simple, well documented and easy to understand software models. UML sequence diagram shows the interaction between classes (or object) in the system for each use case. The interaction represents the order of messages that are exchanged between classes to accomplish a purpose.

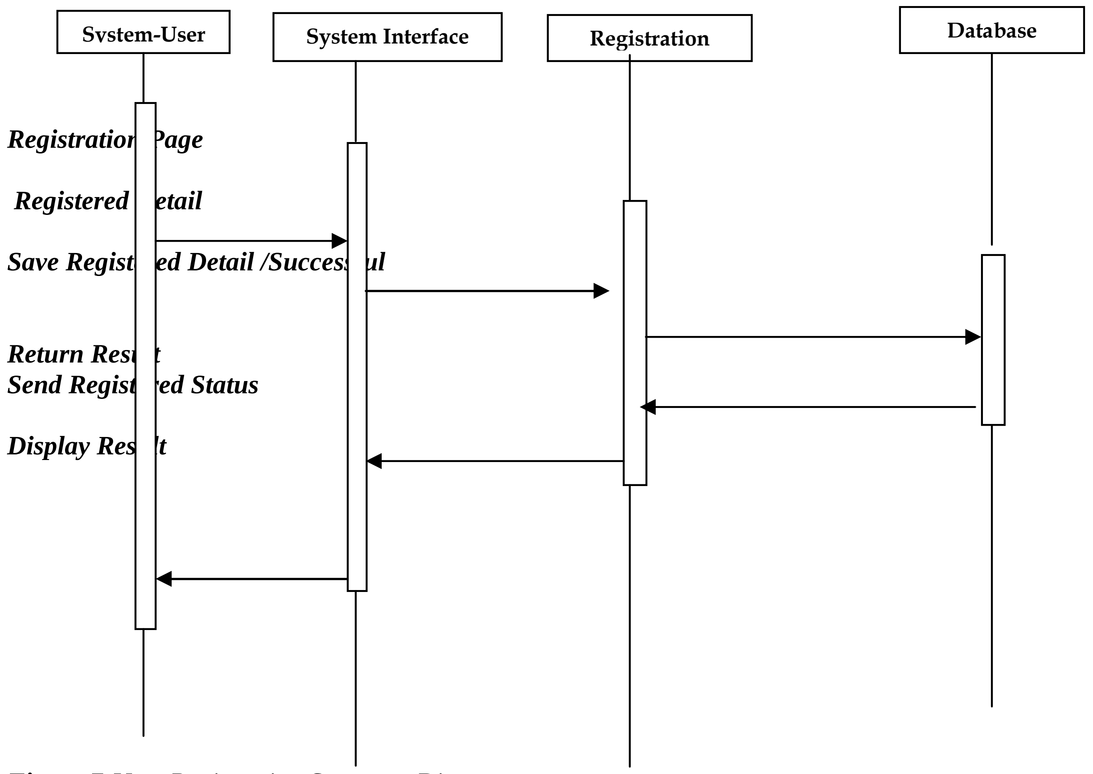

***Figure 7:*** *User Registration Sequence Diagram*

Figure 7, models the sequence of steps involved in the registration of a system user. The order of appearance of the arrows indicates the order of the actions while the arrow direction indicates the direction of flow of events / results.

## 3.0 Implementation

The implementation of the system was handled utilizing MATLAB, for several reasons:

a. Since formal specification are specified utilizing mathematical notation, MATLAB was at the frontier of all available implementation tools available to us due to its available mathematical ingredients.
b. It integrates with numerous user interfaces.
c. It has a large active community base
d. It an open source
e. Itseasy manipulation across varied numerical data.

The system accomplishes it tasks by utilizing several MATLAB tools such as:

a. ***Assignment Statements*****:** Assignment statement as a MATLAB tool was used in overriding Predecessor variable while the successor variable took over. The assignment statement were used in the system to override previous entries which have be saved to the system database
**b.** ***Case Sensitivity:***The variable names within the system are case sensitive acceding to Matrix laboratory rules. The case sensitive rules in MatLab helped us in distinctively separating our variables name
c. ***Immediate and Deferred Execution****:* When MATLAB is invoked, the user is presented with an *interactive environment*. Enter a statement, press the carriage return ("ENTER") and the statement is immediately executed. Given the power that can be packed into one MATLAB statement, this is no small accomplishment.

## 4.0 Findings

Based on the ease at which the users get information through this new system, the following are revealed:

a. Simulate hostel process thereby eliminating wastage of unusual time.
b. Eliminate and ratify unknown errors.
c. Support multiple system processes.
d. Provide user-friendliness interface.

## 5.0 Conclusion

Formal specification is the bedrock of safety critical system which uncovers ambiguities and unwanted error from system requirement. In Nigeria and Africa as a whole, to the best of our knowledge, this approach has not been implemented for most safety critical system opening the avenue for system failure with huge implication such as financial loss and untimely death. This research paper focuses on providing a sample representation using formal specification utilizing hostel processes. The system design was specified utilizing UML while implementation was handled exploring MATLAB. The results of the finding were listed assiduously. Formal specification is an avenue for safety critical system which must be explored in as much safety is involved.

## 6.0 References

[1] Kola Ayanlowo, O Shoewu Segun O. Olatinwo, OLusegun O. Omitola, Damilola D. Babalola (2014), Journal of Science and Engineering, Vol. % (1), 01-10, ORIC publication

[2] Sannella D., (1988), “A Survey of formal software development methods”, appeared in Software Engineering: A European Perspective, A. McGettrick and R. Thayer (eds.), IEEE Computer Society Press, pp 281-297, 1993.

[3] Spivey J. M. (1992), “The Z Notation: A Reference Manual, 2nd Edition”, Prentice Hall International (UK) limited, United Kingdom.

[4] Spivey J. M. (1998), “The Z Notation: A Reference Manual” 3rd Edition, Oxford, United Kingdom.