Journal of the Nigerian Association of Mathematical Physics
***Volume26*, (*March, 2014*), pp 338 – 343**
© J. of NAMP

Formal Representation of an Inventory system

Obi J. C. and Amadin F. I.

Department of Computer Science, University of Benin,
P.M.B. 1154, Benin City, Nigeria.

Abstract

An inventory system is an essential phrase in most well-meaning organization. Manual inventory processes and procedureshave for long been accepted in ascertaining stock quantities but the cumbersome nature with such approaches is time-consuming. Formal representation of an inventory system, utilizing Zed notation, fuzzy logic linguistic representation and Unified Modeling Language (UML) for design has been carried-out through this research paper; thereby reducing uncertainties associated with inventory system variables and portray a more objective approach. The system implementation was handled utilizing Microsoft Windows NT (MWNT) operating system as platform, Structured Query Language (SQL) server database as backend engine, Visual BASIC (VB) as frontend engine and Visual PROLOG (VP) for expert language representation. The result was satisfactory having been able to automate the manual processes and procedures and achieving reduced time optimization..

Keywords: Fuzzy logic, formal Specification, UML.

1.0 Introduction

Automatedinventorysystem keeps track of stockpile within an organization on a perpetual basis utilizing a computer system and a program or software. It controls the process of managing stocks with the aim of the business deriving the overall benefit from the existence of the inventory [1]. The strategy normally involves such functions as setting limits on the actual size of the stocks, while also taking care to maintain enough items on hand to allow the business to operate at maximum efficiency. When conducted responsibly, inventory control also helps businesses to manage their tax obligations more effectively and thus add to the overall profitability of the organization, ensures items are accounted for and that inflow and outflow status is updated on a continual basis. Automatedinventory may be implemented through entities like vending machines or with inventory management companies [1].

Warehouse inventory systems are usually the most common application of this technology because of the often confusing amount of stock coming in and going out of a storage facility. By scanning a barcode on each package, inventory management programs track stock in real time [2]. These programs help warehouses better understand how much stock is on hand, where each item is located, when certain items are nearing re-order points, and what orders need to be shipped to clients invariably foster easy of operation and productivity.

Logic is the study of the structure and principles of correct reasoning, and more specifically, attempts to establish the principles that guarantee the validity of deductive arguments. The central concept of validity is for logic, because when we affirm the validity of an argument, we are saying that it is impossible that its conclusion is false if its premises are true. Hence with world dynamism, and the fuzzy nature of inventory record, the objectives of our proposed system include:

a. Formalizing inventory system properties and
b. Fuzzifying the uncertainties associated with inventory variables.

The proposed system possesses the following:

a. The Inputter which receive input from the seller clerk in the form of 'stock sold' and maximum price.
b. The out-putter which display output result based on sold stock and accumulative amount.

This research paper focuses on achieving the aforementioned objective utilizing a triad approach.

Corresponding author: **Obi J. C.** E-mail: tripplejo2k2@yahoo.com, Tel.: +2348093088218

2.0 Materials and Methodology

Utilizing fuzzy logic for achieving some measure of precision among inventory system variables combined with Zed notations for formal specification and Unified modeling Language (UML) for the design phase formed our applied system techniques.

2.1 Fuzzy Logic Overview

Fuzzy systems are rule-based expert systems based on fuzzy rules and fuzzy inference. Fuzzy sets were introduced by Zadeh [3] to represent/manipulate data and information possessing non statistical uncertainties. Fuzzy sets provide a means of representing and manipulating data that are not precise, but rather fuzzy. Fuzzy logic is a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth; truth values between "completely true" and "completely false", [4, 5 and 6].

Fuzzy systems often learn their rules from experts. When no expert gives the rules, adaptive fuzzy systems learns by observing how people regulate real systems [7]. The difference between classical and fuzzy logic is something called “the law of excluded middle” [8]. In standard set theory, an object does or does not belong to a set. There is no middle ground. In such bivalent systems, an object cannot belong to both its set and its compliment set or to neither of them. This principle preserves the structure of the logic and avoids the contradiction of object that both is and is not a thing at the same time [3]. However, fuzzy logic is highly abstract, foster the law of excluded middles and employs heuristic (experiment) requiring human experts to discover rules about data relationship. Precipitating with these fuzzy rules our system was generated:

Rule 1: IF STOCK 1, STOCK 2, and STOCK 3…N = EMPTY THEN INVENTORY INITIALIZED (∅)
Rule 2: IF STOCK_NAME AND STOCK_NO = CHAR & INT RESPECTIVELY THEN AUTHENTICATED

2.2 Formal Specification Utilizing Zed Notation

Formal specifications use mathematical notation to describe in a precise way the properties which an information system must have, without unduly constraining the way in which these properties are achieved. They describe what the system must do without saying how it must be achieved. This abstraction makes formal specification useful in the process of developing a computer system, because they allow questions about what the system does to be answer confidently, without the need to disentangle the information from a mass of detailed program code, or to speculate about the meaning of phrases in an imprecisely-worded prose description [9].

A formal specification can serve as a single, reliable reference point for those who investigate the customer’s needs, those who implement programs to satisfy those needs, those who test the results, and those who write instruction manuals for the system. Because it is independent of the program code, a formal specification of a system can be completed early in its development. Although it might need to be changed as the design teams gains and understanding and the perceived needs of the customer evolve, it can be a valuable means of promoting a common understanding among all those concerned with system [9].

Z (‘zed’) 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 [9]. Formal specification (mathematical notation or Z) utilizing’s mathematical data types to model data in a system and achieved 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. We use the notation of *predicate logic* to describe abstractly the effect of each operation of our system, again in a way that enables us to reason about the behavior [10].

The other main ingredient in Z is a way of decomposing a specification into small pieces called *Schemas*. By splitting the specification into schemas, we can present it 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 [9]. Every stock is authenticated using the Stock name and numbers.

Stock
Stock_Name: seq CHAR
Stock_No: seq INT
Stock: ℙ STOCK
Access! : Boolean
(user∈user.access! = accepted) V (user ∉user.access! = denied)

Schema 1: *Stock Schema*

From Schema 1, there is no limit to the number of registered Stock piles

```
Inventory Out-putter
Inventory: ℙ Stock
Out-putter: Inventory→ Stock
Inventory = dom out-putter
```

Schema 2: *Inventory out-putter Schema*

From Schema 2, the Inventory Out-putter lists all display stocks within their records for retrieval of these Stocks.

```
Inventory In-putter
Inventory: ℙStock→Price
          In-putter: Inventory→ Stock
          Inventory = dom out-putter
```

Schema 3: *Inventory in-putter Schema*

From Schema 3, the Inventory In-putter lists all inputted stocks within their records.

```
                    FindStock
Ξ InventoryDirectory
Stock? : STOCKS
Inventorylist?ℙ INVENTROTY
(Inventorylist! = {inventory: Inventories = stocks?}∨ (stocks ∉ Inventories ∧ report! = not_known)
```

Schema 4: *FindStock Schema.*

From Schema 4, the Findstock function receives a stock type as an argument and returns all the agents stock with their identification numbers.

```
            StockNotAvailable
        Ξ InventoryDirectory
Inventorys: ℙ INVENTROY
report! : REPORT
stock? : STOCK
    ∀stock∈Inventory.directory ≠ stock
report! = not_known
```

Schema 5: *StockNotAvailable Schema*

Schema 5 shows that error which occurs when a stock requested from an inventory list is not available. An error report of 'not known' is supplied. The directory is initialized at the beginning with no stock and no input or output response in schema 6.

```
          Initialize_Directory
  InventoryDirectory
  Stocks: ℙ STOCK
  Inventories: ℙ inventory
  Inventory = ∅
  stock = ∅
```

Schema 6: *Initialize_Directory Schema*

2.3 Unified Modeling Language (UML)

Unified Modeling Language (UML) is a standard modeling language used for modeling software systems. The focus of UML is on creating simple, well documented and easy to understand software models.

UML enables system engineers to create a standard blueprint of any system. It provides a number of graphical tools that can be used to visualize a system from different viewpoints. The multiple views (user, structural, behavior, implementation and environment) of the system that is represented by using diagrams together depict the model of the system [11].

UML can be used to depict different aspects of a software intensive system through various kinds of views. The views typically used are [11 and 12]

a. The *User view* represents the goals and objective of the system form user's viewpoint.
b. The *structured view* that represent the static or idle state of the system.
c. The *behavioral view* that represents the dynamic or changing state of the state.
d. The *implementation view* that represents the distribution of the logical elements, such that as source code structure, runtime implementation structure of the system
e. The environment view that represents the distribution of the physical elements of the system.

This research paper work focuses on user, and behavior view for modeling the proposed Inventory system.

3.0 System Design; 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 Modeling Language (UML) is a standard modeling language used for modeling software systems. We use UML for design of the Inventory system because UML focuses on creating simple, well documented and easy to understand software models. UML use-casediagram shows the system from the user perspectives class diagram models the static aspect of the system whilesequence 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. For the Inventory system that has been specified, the use case, class and sequence diagrams are clearly highlighted in Figure 1, 2and 3 respectively.

4.0 Implementation

The case study of the Inventory system comprises Microsoft Windows NT operating system as platform, Structured Query Language server database as backend engine, Visual BASIC as frontend engine and Visual PROLOG for Expert language been carried out.

Utilizing the visual prolog queries are propagated as goal possess a variable or not. These variables are usually attached queries. The queries could be single or compound queries with the increasing nature of the variable determine the queries class. Some sample of prolog goals.

belong (stock 1, Inventory Inputter)
display (stock 2, Inventory out-putter)
purchase (customer 1, gold)

5.0 Findings

The system was able to achieve the following:

a. Accept input stock and catalogue within the inventory inputter.
b. The inventory system when implemented will save organizational time by eliminating manual processes.
c. Display output stock and catalogue within the inventory out-putter.

6.0 Conclusion

Formalization of a software system is an integral part of system development. Utilizing zed notation, fuzzy logic and UML, formalization of an inventory system properties, design and mitigating uncertainties within inventory system variables has been achieved. The system when fully implemented will ease the manual inventory processes utilized in most organizations.

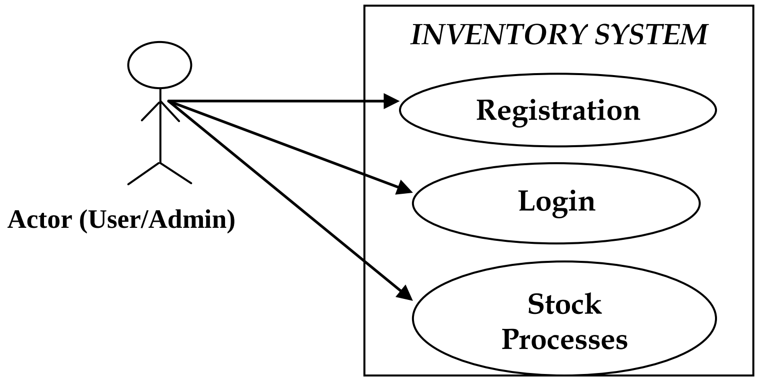

Figure 1: *Use Case Diagram modeling an Inventory system*

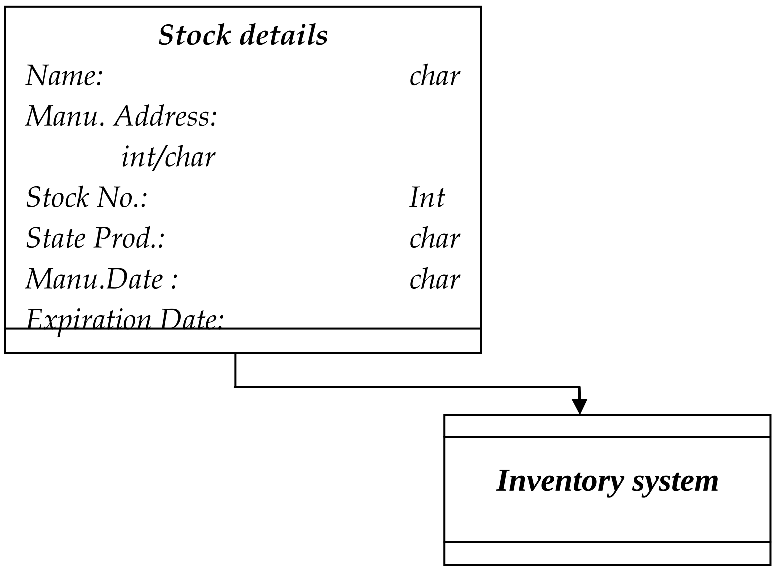

Figure 2: *Class diagram Showing Attributes, generation and composition Association for an Inventory system.*

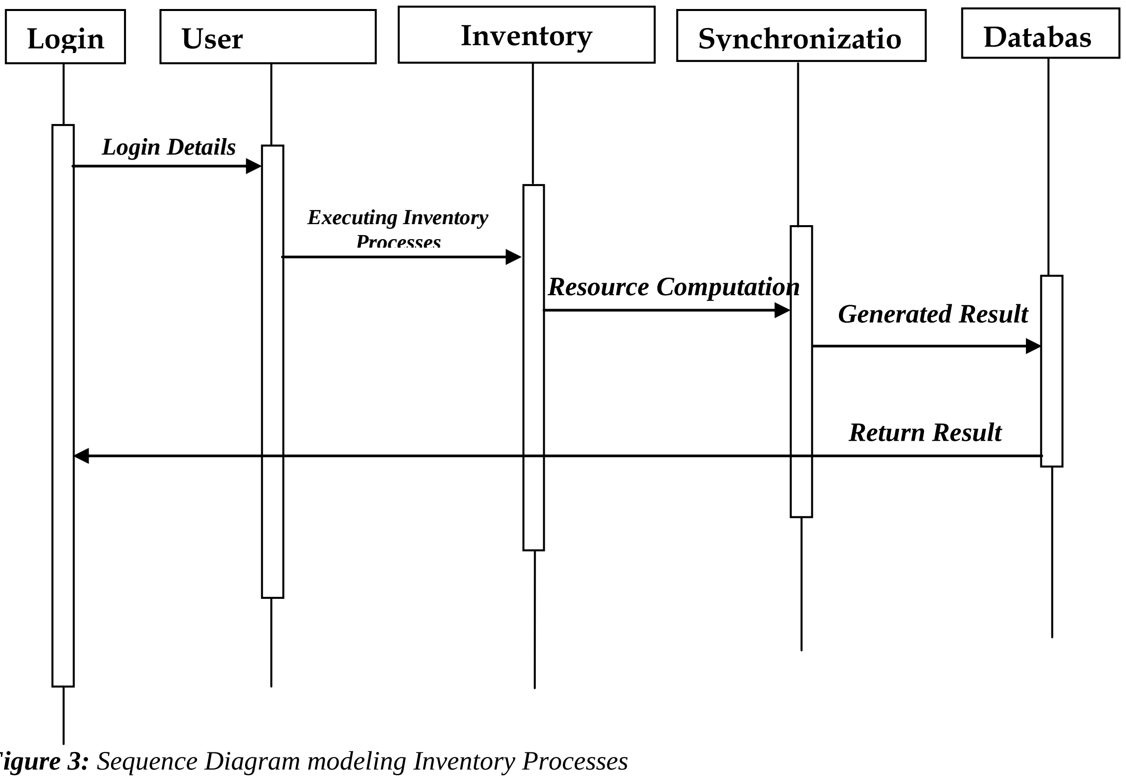

Figure 3: *Sequence Diagram modeling Inventory Processes*

7.0 References

[1] WiseGeek (2012), "Automated Inventory System", retrieved from wisegeek.com/101/5
[2] Anreas (2006), "Barcode Inventory System Process" retrieved online from match barcode.com
[3] Zadeh L.A. (1965): "Fuzzy sets and systems". In: Fox J, editor, System Theory. Brooklyn, NY: Polytechnic Press, 1965: 29–39.
[4] Christos S. And Dimitros S. (2008) "Neural Network", retrieved from http://www.docstoc.com/docs/15050/neural-networks.
[5] Kasabov N. K. (1998), "Foundations of neural networks, fuzzy systems, and knowledge engineering", A Bradford Book, The MIT Press Cambridge, Massachusetts London, England.
[6] Robert F. (2000), "Introduction to Neuro-Fuzzy Systems, Advances in Soft Computing Series", Springer-Verlag, Berlin/Heildelberg, 289 pages
[7] Leondes C. (2010), "The Technology of Fuzzy Logic Algorithm retrieved from Suite101.com/examples-of-expert-System-application-in-artificial Intelligence
[8] Bart K. and Satoru I. (1993), "Fuzzy Logic", retrieved from http//:Fortunecity.com/emachines/e11/86/fuzzylog.html.
[9] Spivey J. M. (1998), "The Z Notation: A Reference Manual", Oxford, J. M. Spivey.
[10] Spivey J. M. (1992), "The Z Notation: A Reference Manual, 2nd Edition", Prentice Hall International (UK) limited, United Kingdom.
[11] Philippe K. (2000), "Rationale Unified Process: An Introduction: Second Edition, Addison Wesley
[12] Chris M. (2000), "Enterprise Modeling With UML: Designing Successful Software through Business Analyses, Addison Wesley