ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 3, Issue 11 , November 2016

Abstraction of Design Components from Software Requirement Specifications (SRS)

Syed Naimatullah Hussain, Dr Syed Zakir Ali

Research Scholar, Computer Science and Engineering, JJTU Jhunjhunu, Rajasthan,India
Professor, Computer Science and Engineering, SECAB Engineering College, Karnataka, India

ABSTRACT: The Information Systems Development normally starts with collection of customer's requirements. Developers of the software will collect the requirements from the customers, who are also referred to as end users of in the organization, and prepares a document called the Software Requirement Specification (SRS). To visualize the design of Information Systems through SRS, Unified modeling language (UML) has been designed. However, UML suffers a low success rate. This paper abstracts the view elements from SRS for any information System in the form of Classes, Object of the Classes and Actors along with its interfaces and their characterizing attributes. And these abstractions are further refined using good database design principles and the use of Data Flow Diagrams (DFDs).

KEYWORDS: Classes, Attributes, Interfaces.

I. INTRODUCTION

The software development project normally starts with SRS of the customers [1, 2]. The customers are in general, strategic management people of the organization who work in classical ambience. So the requirements of the expected system reflect their processing mindset. These requirements are influenced by either the data oriented approach or the procedure oriented approach as with the available information of the organization. Presently, since these are not natural ways of processing the information system, these will not serve the development process effectively. Now a days, people feel the object-oriented paradigm is more towards naturalness and will survive for long time. So it is required to transform the requirements into object-oriented paradigm (Classes, Object of Classes, Actors and its interfaces) and then proceed for the development. The main objective of this paper is to develop a system of software tools, that takes SRS and then transform it into Actors, Classes and Objects and its attributes. There may also be a need to limit our ambition, as some of the sub processes may not be automated due to English as Natural Language. In such case, there is a need to provide guidelines for each of these sub processes to minimize the human dependency. The aim to develop a sequence of semi-automated software tools with embedded guidelines for inevitable subtasks at some stages. Then, the set of guidelines may give the scope to develop software agents to take up the role of semi automated processes.

Few researchers[3] have suggested some techniques for identification of object class hierarchies in procedural object oriented, but in our methodology we are using a perfect balance of procedural as well data oriented paradigm i.e., nothing but object oriented paradigm which is more towards naturalness for certain stages of the design of object classes [4, 9]. In this paper, authors are abstracting only object class and its attributes based on parsed use-case description without concentrating on all the design components from the SRS. Authors [8,10] of these papers helps in the designing only the class and subclass structures but fails to identify the object class structure i.e. object class name, attributes, actors and its interfaces. Although, these give good guidelines to the design, the authors could not derive any concrete procedure and/or guidelines to the design in its totality.

To develop a methodology that identifies Classes and Objects of the Classes, Actors and its attributes, a semi-automatic methodology has been proposed. This semi-automatic methodology comprises of a sequence of steps like feasibility analysis (for object structure identification), resolution of synonyms & homonyms issues, regrouping of attributes of entities & functionalities through the design of data flow diagrams and elimination of imbalance data & procedure selection along with authentication of correctness & completeness of the abstractions at each stage. There is manual intervention at few stages which is necessitated because of the need for human intelligence in these steps. Even for these manual intervention steps, attempt is made to provide clear-cut guidelines to streamline the design process.

II. METHODOLOGY

The algorithm presumes that, SRS of the proposed Information System is available with the developer, who can seek needy information from client's team of users (CTU).

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 3, Issue 11 , November 2016

1. Requirement gathering
   As per the SRS, the detailed requirement is gathered from CTU. The input of any information system is SRS, output is Classes, Objects of the Classes, Actors and its attributes.
2. Authentication of correctness and completeness of the process
   Now we have two sets of entities, attributes, interrelationships, business rules, work processes and business process. The developer need to establish the one-to-one and on-to correspondence separately between each pair of items of two sets.
3. Resolve synonyms/homonyms issue.
   As English is a Natural Language, in a multi-user system, though each user assigns meaningful names to attributes, the semantic flexibility in the use of English words leads synonymous words for the same attribute. The set of synonymous words of the same meaning forms a synonymy and each such synonymy is replaced by a generic name. Similarly, the use of context specific word leads to homonyms. The presence of each such homonym in attributes/entities/actors should be replaced by different names.
4. 4 Model the business process through measuring the paradigms imbalance.
   - Entities, which are modifiable within the system
   - Entities which are non modifiable within the systems form actors.
   - The identified functionalities form processes
   - The intersection of input/output attributes of a functionality with each entity attributes forms input/output dataflow from/to the concerned entity /actor to/form the process

   This logical DFD (Fig-2.2) reveals the degree of imbalance in the paradigms opted while choosing structural and behavioural components. Figures 2.51 and 2.52 depict the possible reasons tilt respectively towards Procedure oriented and data oriented paradigms

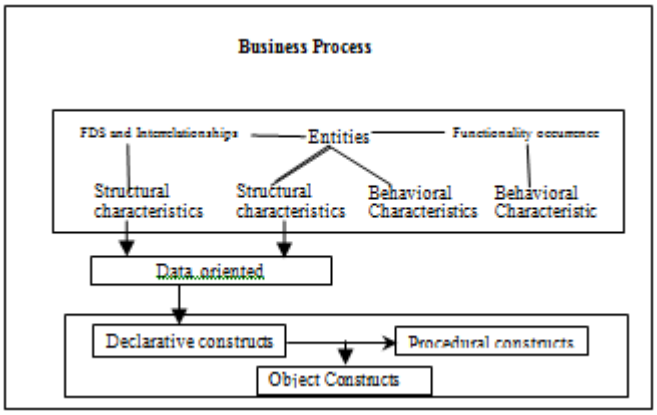

Fig. 1.1(Procedure Oriented)

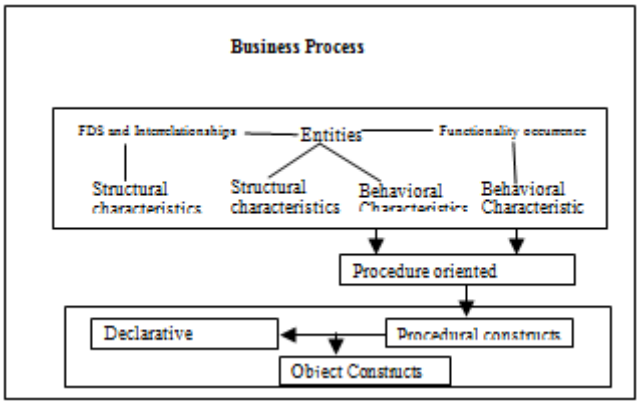

Fig.1.2 (Data Oriented)

5. Eliminate superfluity and redundancy in attributes/entities presence.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 3, Issue 11 , November 2016

Study each attributes of each entity/actor in isolation with other entities/attributes for absolute necessity of their presence in the information system. This can be identified by the participation of the attribute in any of the functionalities. Discard the attributes that are not participating in the functionalities. If an attribute or group of attributes is present in two or more entities, form a separate entity with each such group. This participation can be tested through the establishment of one-to-one and on-to correspondence between attributes referenced in data flows of logical DFD (Fig 2.0)

6. Identify keys and design extended ER-diagram
   Abstract the functional dependencies amongst attributes of each entity from the business rules of the information system. Identify the primary key and foreign keys for each entity/actor. If an attribute or group of attributes of an entity of data store is independent of the primary key, take it out and form separate entity. The entities and attributes are abstracted from the data dictionary and the interrelationships are abstracted from the business rules of the system.
7. Minimize the imbalance between the procedure and data oriented paradigms.
   [Fig-1.1 &1.2] Regroup the attributes of input data flows, based on their characterization of Person, place, thing, event or concept. This can be achieved through the grouping the attributes such that each attribute of a group establishes a functional dependence with one primary key. The input data flows may contain subset of attributes of a group so that each group of attributes may be in input data flows of one or more processes. Each such group forms a first cut object structure. [Fig-1.3] shows the perfect balance between data and procedure oriented i.e object oriented which is more towards naturalness.
8. Refine the abstracts by brining the perfect balance between the paradigms

   [5.6.7] Apply good database design principles to each first cut object structure [section 3.2].

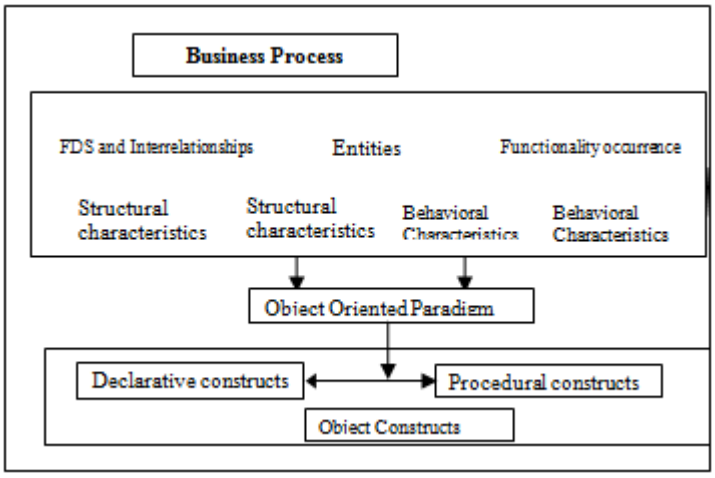

Figure-1.3 Object Oriented

Since the normalization is a way of good database design, the refined group should be normalized at least up to Boyce coded normal form. Figure-2.2 indicates the constructs to bring perfect balance between procedure oriented and data oriented paradigms.

9. Design the logical DFD with object structure
   Fig-2.2 Refine the logical data flow diagram with each object structure group as data store, maintaining attributes of total input data flow to each process and redesigning input flows such that each flow emanates from first cut object structure.
10. Decompose functionality
    If two or more data flows directed towards single process, study the possibility of decomposing the process, so that each decomposed process either receives data flow from single entity or additionally through parameter passing from other entities. Refine the logical data flow diagram into physical data flow diagram.
11. Software Requirement Specification [SRS]
    The input to the any information system is the requirements that contain the business rules of the information system along with various applications. For example, SRS for Banking Application information system may contain some of the business rules as follows.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 3, Issue 11 , November 2016

III.SRS

Automate the customer transaction in the banking system
There are two types of customer.
Deposit customer

- Each customer is provided with a unique account number
- Periodically credits/debits from his/her account.
- The customer earns interest on minimum balance of each month.
- There may be overdraft facility for some customer who has developed some kind of rapport with the bank.
- The customer can also transfer amount from his/her account to any account.

Loan Customer

- Loan customer initially applies for loan for specific purpose.
- The bank grants loan to customers after authenticating their repayments in installments.
- The bank charges interest either on floating-point rate or fixed-point rate.
- Interest rate for bank loan is generally much higher than interest rate for bank deposit
- Thus the bank earns profit by liaising between savings customer and loan customer
- Each branch manager needs to submit periodical summary report indicating the total number of transactions, the total amount deposited, the total interest paid to the customer, the total amount sanctioned, the interest earn by customer and thus the profit earn by the bank in the specific

1. Identification of Classes, attributes from SRS

Depositor (Acctno,D-name,D-address,Contact-no)
Transaction (Acctno,Pre-balance,Transtype,Transamt,New-balance,OD-facility)
Daccount (acctno, period, minibalance, int-rate, intAmt)
Transfer (acctno, minbalance,ch-amt,payee-acct,newbalance,payeebalance)
Borrower (acctno, loan-type, Amt-sanctioned, Amtavailed, newbalance, int-rate)
EMI (acctno, Balance, Rp-period, Principal, Int, EMI)
Baccount (acctno, period, maxbal, int-rate, intamt)
Report (period, no of depositor, amt deposited, int-paid, amt-sanctioned, no of borrowers, amt-disbursed, int-earned, profit)
Loan (Bacctno, Bname, Baddress, Bcontact no)

2. Synonyms and homonyms

These are Homonyms found in various entities like, Acctno, period, Int-rate, Int-amt, NewBalance.
Depositor (Acctno,D-name,D-address,Contact-no)
Transaction (Acctno,Pre-balance,Transtype,Transamt,New-balance,OD-facility)
Daccount (Dacctno, Dperiod, minibalance, Dint-rate, DintAmt)
Transfer (Dacctno, minbalance,ch-amt,payee-acct,Dnewbalance,payeebalance)
Borrower (Bacctno, loan-type, Amt-sanctioned, Amtavailed, Bnewbalance, Bint-rate)
EMI (Bacctno, Balance, Rp-period, Principal, Int, EMI)
Baccount (Bacctno, Bperiod, maxbal, Bint-rate, Bintamt)
Report (period, no of depositor, amt deposited, int-paid, amt-sanctioned, no of borrowers, amt-disbursed, int-earned, profit)
Loan (Bacctno, Bname, Baddress, Bcontact no)
I could not fine any of synonyms in this application

3. Data Flow Diagram (DFD)

Context Level Diagram [14]:
Fig-2.1 Depicts that the interaction between the banking information systems with the external agents which acts as data sources and data sinks.

Fig-2.1

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 3, Issue 11 , November 2016

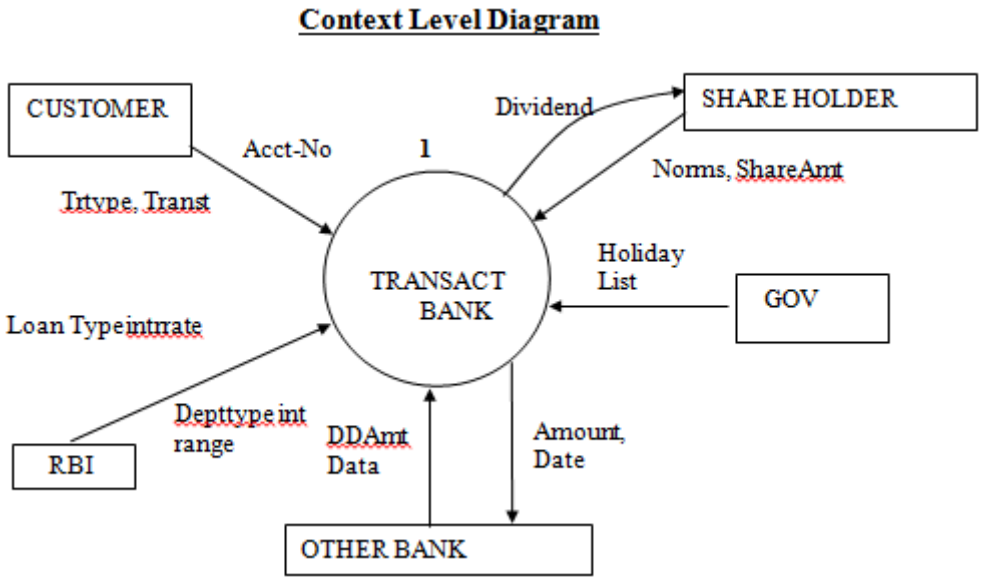

Figure-2.2 logical DFD [15] captures the data flows that are necessary for a banking information system to operate. It describes the processes that are undertaken, the data required and produced by each process.

Results:

Actors: Customer, Depositor

Attributes of Actor: report, transaction, EMI, Bank database

Classes: Sanction Loan, Transaction Amount, Transfer Amount,Compute EMI,Prepare Report, Assign Account No.

Objects of Classes: Acctno,D-name,D-address,Contact-no,Acctno,Pre-balance,Transtype,Transamt,New-balance,OD-facility) Dacctno, minbalance,ch-amt,payee-acct,Dnewbalance,payeebalance

(Bacctno, Balance, Rp-period, Principal, Int, EMI)

period, no of depositor, amt deposited, int-paid, amt-sanctioned, no of borrowers, amt-disbursed, int-earned, profit, Bacctno, Bname, Baddress, Bcontact no.

IV. CONCLUSION

UML is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-information system. But UML suffer from the correctness and completeness aspect. Developed methodology helps in designing design components (Classes, Objects and its Attributes) in a semi-automatic way incorporating correctness and completeness in each of its stages. This methodology minimizes human interventions that too giving clear-cut guidelines. In future these, interventions also can be automated.

REFERENCES

[1] Pankaj Jalote (2005). An Integrated approach to Software engineering, 3rd edition springer's publications, India.
[2] Roger S Pressman (2005). Software Engineering, 6th edition McGraw-Hill companies international edition, India
[3] Mosa Emhamed Elbendak(2005) "Framework for using a Natural Language Approach to Object Identification Framework for using a Natural Language" http://www.aclweb.org/anthology/R09-2005.
[4] Muhammed Usman, Stepahane Ducane and Marianne Huchard (IEEE-2008) 15th working conference on reverse engineering "Reconsidering classes in procedural object oriented code" page 257- 266.
[5] Jonathan & charles J Hannon (IEEE-2009) Eighth Mexican International Conference on Current Trends in Computer Science. "An algorithm for identifyingAuthors using synonyms" page 99 – 104.
[6] Alen Lovrencie and Tonirnir Kisasondi (IEEE-2007) 11th International Conference on Intelligent Engineering System "Modelling Functional Dependencies in Databases using mathematical logic" page 307 – 312.
[7] Steffen Rudle and LarsSchmidt-thieme (IEEE-2006) Sixth International Conference in Data Mining "Object Identification with constraints" page 1
[8] Sukhamay Kundu & Migel (IEEE-2005) 29th International Conference Software and Application Conference "A Formal approach to Designing a class subclass structure using a partial order on the functions" page 1 -8.

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 3, Issue 11 , November 2016

[9] Mosa Emhamed Elbendak "Framework for using a Natural Language Approach to Object Identification Framework for using a Natural Language" Student Research Workshop, RANLP 2009 - Borovets, Bulgaria, pages 23–28

[10] S. Anandha Mala and Dr.G.V.Uma "Object Oriented Visualization of Natural Language Requirement Specification and NFR Preference Elicitation. IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.8, August 2006.

[11] Lilac A. E. Al-Safadi "Natural Language Processing for Conceptual Modeling" A semi-automatic approach for designing databases. International Journal of Digital Content Technology and its Applications Volume 3, Number 3, September 2009

[12] Sunguk Lee "UML for Database System and computer application. Interantional Journal of Database System and Computer Application. Volume 5, number 3, 2012

[13] K.P Jayant, Garg,Kumar, Rana. "An approach of Software Design Testing Based on UML diagram. International Journal of Advanced Research in Computer Science and Software Engineering. Volume-4, Issue-2, February 2014

[14] Tiwari,Alpika,Dubey "Merging of Dataflow Diagrams with Unified Modeling Language" Internation Journal of Scientific and Research Publication Volume-2,issue-8, August-2012

[15] Rosziati,Yen "A Formaol Model for Data Flow Diagrams Rules" APRN Jounal of System and Software, volume-1 No.2, May-2011