ATTRIBUTES SANITIZATION IN OBJECT-ORIENTED DESIGN TO IMPROVE DATABASE STRUCTURE

MOHD ZAINURI BIN SARINGAT

A thesis submitted in fulfillment of the requirement for the award of the Doctor of Philosophy in Information Technology

Faculty of Computer Science and Information Technology
Universiti Tun Hussein Onn Malaysia

JUNE 2014
ABSTRACT

Modelling using the Entity Relationship Model was introduced more than thirty years ago. Until late 1990’s, object-oriented introduced class diagram. However, designing a good database is still a serious issue. Some of the issues are very difficult to handle such as consistency checking between system design and database design, redundancy of data, mismatch of the data structure with the user’s needs in the database and unused data in the database. In this thesis, a new technique called UIInData is introduced as an alternative method for designing database based on attribute sanitization. The proposed technique will extract class behaviour from class diagram to produce schema table which will then be compared with the user interface to normalize the structure. Attributes sanitization is introduced to remove the unused attributes and to provide final schema table. An experiment using three case studies has shown that some improvements of designing optimal database have been achieved in term of data sanitization and data accessibility. Attribute sanitization was applied in LAS, SPKS and MPBP database. Data sanitizations have removed 2.2%, 14.1% and 24.5% from defined attributes which are not used by user interface. Meanwhile the results shown in data accessibility for these three cases have shown that LAS was reduced by 50%, SPKS have not reducing of data accessibility and MPBP was reduced by 20% when UIInData is used as compared to using ordinary object-oriented. Therefore, the UIInData is a good alternative technique to improve database structure.
ABSTRAK

Permodelan menggunakan Model Hubungan Entiti telah diperkenalkan sejak lebih tiga puluh tahun. Sehingga lewat 1990-an, kaedah berorientasikan objek memperkenalkan *class diagram*. Walaubagaimanapun, proses mereka bentuk pangkalan data yang baik masih menjadi isu yang besar. Antara isu yang dikenalpasti adalah sukar mempastikan keserasian diantara rekabentuk sistem dan pangkalan data, pertindanan data, stuktur pengkalan data tidak memenuhi kehendak pengguna dan atribut yang tidak digunakan dalam pengkalan data. Dalam tesis ini, satu teknik baru yang dinamakan UInData telah diperkenalkan sebagai kaedah alternatif untuk menghasilkan pangkalan data dengan membuang atribut. Teknik ini akan mengambilkira ciri-ciri *kelas* bagi menghasilkan jadual skema dan membandingkannya dengan antara muka penguna bagi melaksanakan struktur pernormalan. Pembuangan atribut diperkenalkan bagi membuang atribut yang tidak digunakan bagi menghasilkan skema jadual yang terkini. Pengujuan terhadap tiga kajian kes mendapati penambahbaikan rekabentuk pangkalan data telah dicapai dalam penghapusan data dan capaian data. Penghapusan atribut telah diaplikasikan terhadap pangkalan data LAS, SPKS dan MPBP. Penghapusan data telah memadam 2.2%, 14.1% dan 24.5% dari atribut yang dikenalpasti bilamana ia tidak digunakan dalam antaramuka pengguna. Sementara itu dalam capaian data bagi ketiga-tiga kajian kes menunjukkan capaian data bagi LAS telah berkurangan sebanyak 50%, SPKS tiada perbezaan dalam kadar capian data dan MPBP telah berkurangan sebanyak 20%. Ini menunjukkan UinData adalah salah satu teknik yang berguna dalam menambahbaik rekabentuk pangkalan data.
## CONTENTS

<table>
<thead>
<tr>
<th>TITLE</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECLARATION</td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td>ABSTRAK</td>
<td>vi</td>
</tr>
<tr>
<td>CONTENTS</td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF PUBLICATIONS</td>
<td>xiii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>xiv</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xvi</td>
</tr>
<tr>
<td>LIST OF SYMBOLS AND ABBREVIATIONS</td>
<td>xviii</td>
</tr>
<tr>
<td>LIST OF APPENDICES</td>
<td>xx</td>
</tr>
</tbody>
</table>

### CHAPTER 1 INTRODUCTION

1. **Background of Study**  
2. **Problem Statement**  
3. **Aim**  
4. **Objectives of study**  
5. **Scope and Limitation**  
6. **Report Organization**
CHAPTER 2 LITERATURE REVIEW

2.1 Introduction 11

2.2 Overview of Data 11

2.2.1 Data Sanitization 12

2.2.2 Attribute Sanitization 13

2.3 Structured Approach 16

2.3.1 Data Flow Diagram 16

2.3.2 Entity Relationship Diagram 16

2.4 Object-Oriented Approach 17

2.4.1 Class Diagram 19

2.4.2 User Interface 21

2.5 Normalization Technique 22

2.6 Database Design 23

2.6.1 Database Design Using Structured Technique 24

2.6.2 Design Database using Object-oriented Technique 26

2.6.3 Optimal Database 27

2.7 The Issues in Designing Database 28

2.8 Comparison Structure and Object-oriented Techniques 31

2.9 Summary 33

CHAPTER 3 RESEARCH METHODOLOGY 34

3.1 Introduction 34

3.2 Methodology 34
CHAPTER 4  FORMALIZATION OF RULES IN DESIGNING OPTIMAL DATABASE

4.1 Introduction

4.2 Attribute Sanitization

4.2.1 Attribute Sanitization in Class Diagram

4.2.2 Generating Schema Table

4.2.3 User Interface Normal Form (UINF)

4.2.4 Attribute Sanitization in User Interface

4.2.5 Optimal Schema Table

4.3 Rules for Designing Database Using Attribute Sanitization Technique

4.3.1 Rules of Attribute Sanitization in a Class Diagram

4.3.2 Rules of Generating Schema Table

4.3.3 Rules of User Interface Normal Form (UINF)

4.3.4 Rule of Attribute Sanitization in User Interface Normal Form (UINF)

4.4 Algorithm for Designing Database
4.4.1 Algorithm for Attribute Sanitization in Class Diagram

4.4.2 Algorithm for Generating Schema Table

4.4.3 Algorithm for User Interface Normal Form (UINF)

4.4.4 Algorithm for Attribute Sanitization in User Interface Normal Form (UINF)

4.5 Formalization for Designing Database

4.5.1 Formal Model of Class Diagram

4.5.2 Formal Model in Schema Table

4.5.3 Formal Model in User Interface Normal Form (UINF)

4.6 Consistency Rules for Designing Database Using Attribute Sanitization

4.6.1 Consistency Rules between Class Diagram and Schema Table

4.6.2 Consistency Rules between Schema Table and User Interface

4.7 Summary

CHAPTER 5 IMPLEMENTATION OF UInData

5.1 Introduction

5.2 Environment for UInData Tool

5.3 Work Instruction of Create Schema Table
5.4 Work Instruction of Create Optimal Schema

Table 70

5.5 Summary 75

CHAPTER 6 EVALUATION OF UiData 76

6.1 Introduction 76

6.2 Lecturer Assessment System (LAS) 79

6.2.1 Designing LAS Database Using Object-oriented Technique 81

6.2.2 Designing LAS Database Using UiData 81

6.2.3 Results Comparison on LAS Database Design 88

6.3 Sistem Pengurusan Klinik Sejahtera (SPKS) 90

6.3.1 Designing SPKS Database Using Object-oriented Technique 91

6.3.2 Designing SPKS Database Using UiData 92

6.3.3 Results Comparison on SPKS Database Design 99

6.4 Sistem Pengurusan Kompaun Majlis Perbandaran Batu Pahat (MPBP) 101

6.4.1 Designing MPBP Database Using Object-oriented Technique 102

6.4.2 Designing MPBP Database Using UiData 103
6.4.3 Results Comparison on MPBP Database Design

6.5 Summary

CHAPTER 7 CONCLUSIONS AND FUTURE WORKS

7.1 Introduction

7.2 Summary of Contributions

7.2.1 Attribute Sanitization Technique

7.2.2 Formalization of Rules for Attributes Sanitization

7.2.3 UInData Tool

7.3 Future Work

7.4 Summary

REFERENCES

APPENDIX

VITA
LIST OF PUBLICATIONS

Journals:


Proceedings:


LIST OF TABLES

2.1 Current Designing Database Techniques 29
2.2 Comparisons of Structure and Object-oriented Techniques 32
4.1 Overview of Research Elements and Formal Logical Specification 53
4.2 Mapping Between Consistency Rules and Formal Logical Specification 57
6.1 LAS-OOT Schema Table 81
6.2 LAS-UlnData Schema Table 82
6.3 UINF in Table lasStudentAttendance 83
6.4 UINF in Table lasSubjectMark 84
6.5 UINF in Table lasStudentMark 85
6.6 UINF in Table lasSubjectPercentage 86
6.7 Result on LAS Database Design 88
6.8 LAS Matrix Accessing Table 89
6.9 SPKS-OOT Schema Table 92
6.10 SPKS-UlnData Schema Table 93
6.11 UINF in Table pekerja 94
6.12 UINF in Table temujanji 94
6.13 UINF in Table doktor 95
6.14 UINF in Table pesakit 96
6.15 UINF in Table ubat 96
6.16 UINF in Table stokUbat 97
6.17 UINF in Table rawatan 97
6.18 Result on SPKS Database Design 99
6.19 SPKS Matrix Accessing Table 100
6.20 MPBP-OOT Schema Table 103
6.21 MPBP-UInData Schema Table  
6.22 UINF in Table staf  
6.23 UINF in Table letakKereta  
6.24 UINF in Table jawatan  
6.25 UINF in Table undangUndang  
6.26 UINF in Table kesalahan  
6.27 UINF in Table penguatkuasaan  
6.28 Result on MPBP Database Design  
6.27 MPBP Matrix Accessing Table
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>System Design: Specify How System Works</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>The System Design Components</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>UML Class Diagram</td>
<td>20</td>
</tr>
<tr>
<td>2.2</td>
<td>Structured Technique in Designing Database</td>
<td>25</td>
</tr>
<tr>
<td>2.3</td>
<td>Object-oriented Technique in Designing Database</td>
<td>27</td>
</tr>
<tr>
<td>3.1</td>
<td>The Flow Chart Research</td>
<td>36</td>
</tr>
<tr>
<td>3.2</td>
<td>Idea of the Research</td>
<td>38</td>
</tr>
<tr>
<td>3.3</td>
<td>The Research Framework</td>
<td>39</td>
</tr>
<tr>
<td>4.1</td>
<td>Attribute Sanitization in Class Diagram Algorithm</td>
<td>50</td>
</tr>
<tr>
<td>4.2</td>
<td>Relationships Algorithm</td>
<td>50</td>
</tr>
<tr>
<td>4.3</td>
<td>User Interface Normal Form Algorithm</td>
<td>51</td>
</tr>
<tr>
<td>4.4</td>
<td>Attribute Sanitization in User Interface Algorithm</td>
<td>52</td>
</tr>
<tr>
<td>5.1</td>
<td>First Page of Visual Studio 2010</td>
<td>65</td>
</tr>
<tr>
<td>5.2</td>
<td>The C# Code Editor</td>
<td>65</td>
</tr>
<tr>
<td>5.3</td>
<td>Implementing the LAS Class Diagram</td>
<td>67</td>
</tr>
<tr>
<td>5.4</td>
<td>Sample Error in Validate Process</td>
<td>68</td>
</tr>
<tr>
<td>5.5</td>
<td>Menu Options in Generate Schema Table</td>
<td>68</td>
</tr>
<tr>
<td>5.6</td>
<td>Text File for LAS Schema Table</td>
<td>69</td>
</tr>
<tr>
<td>5.7</td>
<td>Start Page</td>
<td>70</td>
</tr>
<tr>
<td>5.8</td>
<td>Opening Text File</td>
<td>70</td>
</tr>
<tr>
<td>5.9</td>
<td>Display Tables and Attributes</td>
<td>71</td>
</tr>
<tr>
<td>5.10</td>
<td>Marking Intersection between Tables and User Interfaces</td>
<td>71</td>
</tr>
<tr>
<td>5.11</td>
<td>Calculating Intersections</td>
<td>72</td>
</tr>
<tr>
<td>5.12</td>
<td>Designing Table Structure</td>
<td>73</td>
</tr>
<tr>
<td>5.13</td>
<td>Optimal Table Structure</td>
<td>73</td>
</tr>
<tr>
<td>5.14</td>
<td>Text File of Optimal Table Structure</td>
<td>74</td>
</tr>
<tr>
<td>5.15</td>
<td>Text File of Attribute Sanitization</td>
<td>74</td>
</tr>
<tr>
<td>6.1</td>
<td>LAS Class Diagram</td>
<td>80</td>
</tr>
<tr>
<td>6.2</td>
<td>LAS Optima Schema Table</td>
<td>87</td>
</tr>
<tr>
<td>6.3</td>
<td>SPKS Class Diagram</td>
<td>91</td>
</tr>
<tr>
<td>6.4</td>
<td>SPKS Optima Schema Table</td>
<td>98</td>
</tr>
<tr>
<td>6.5</td>
<td>MPBP Class Diagram</td>
<td>102</td>
</tr>
<tr>
<td>6.6</td>
<td>MPBP Optima Schema Table</td>
<td>110</td>
</tr>
</tbody>
</table>
LIST OF SYMBOLS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1NF</td>
<td>First Normal Form</td>
</tr>
<tr>
<td>2NF</td>
<td>Second Normal Form</td>
</tr>
<tr>
<td>3NF</td>
<td>Third Normal Form</td>
</tr>
<tr>
<td>4NF</td>
<td>Fourth Normal Form</td>
</tr>
<tr>
<td>ANSI/IEEE</td>
<td>American National Standards Institute/ Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>AS</td>
<td>Attributes Sanitization</td>
</tr>
<tr>
<td>CD</td>
<td>Class Diagram</td>
</tr>
<tr>
<td>DBA</td>
<td>Database Administrator</td>
</tr>
<tr>
<td>DBMS</td>
<td>Database Management System</td>
</tr>
<tr>
<td>DFD</td>
<td>Data Flow Diagram</td>
</tr>
<tr>
<td>ERD</td>
<td>Entity Relationship Diagram</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>K</td>
<td>Primary Key</td>
</tr>
<tr>
<td>KDD</td>
<td>Knowledge Discovery in Database</td>
</tr>
<tr>
<td>LAS</td>
<td>Lecturer Accessing System</td>
</tr>
<tr>
<td>MPBP</td>
<td>Pengurusan Kompaun Majlis Perbandaran Batu Pahat</td>
</tr>
<tr>
<td>OCL</td>
<td>Object Constraint Language</td>
</tr>
<tr>
<td>OOD</td>
<td>Object-oriented Design</td>
</tr>
<tr>
<td>OOT</td>
<td>Object-oriented Technique</td>
</tr>
<tr>
<td>S</td>
<td>Sanitization</td>
</tr>
<tr>
<td>SMP</td>
<td>Student Information System</td>
</tr>
<tr>
<td>SPKS</td>
<td>Sistem Pengurusan Klinik Sejahtera</td>
</tr>
<tr>
<td>SRS</td>
<td>System Specification Document</td>
</tr>
<tr>
<td>SSADM</td>
<td>Structure System Analysis and Design</td>
</tr>
<tr>
<td>UlnData</td>
<td>Attributes Sanitization in Object-oriented Design to Improve Database Structure</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>UML</strong></td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td><strong>UINF</strong></td>
<td>User Interface Normal Form</td>
</tr>
<tr>
<td><strong>UTHM</strong></td>
<td>Universiti Tun Hussein Onn Malaysia</td>
</tr>
</tbody>
</table>
# LIST OF APPENDICES

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Listing A: Part of Source Code for Generating Schema Table from Class Diagram</td>
<td>127</td>
</tr>
<tr>
<td>B</td>
<td>Text File of Schema Table from Class Diagram</td>
<td>149</td>
</tr>
<tr>
<td>C</td>
<td>Listing B: Part of Source for Designing Optimal Schema Table</td>
<td>151</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Background of Study

Living in this modern era is interesting but complex. Since the early 1980’s, when computer gains its popularity, the human social interaction dramatically changes. Furthermore, with the increase in the number of human population in the world, life has become more challenging. People are migrating from their country to other countries for many reasons such as study, work, travel and business. This phenomenon encourages the establishment of good platform for communications through which people can interact to share information, do banking transactions, pay bills and purchase goods. All these involve huge data. The data must always be available and free from errors. Incorrect data will lead to wrong information and inaccurate decision making [1, 2]. The characteristics of good quality information are that the information must be accurate, available and accessible right on time and in right form [3]. To ensure this, good technique in designing database must be well defined. Another reason for a good database design is to increase the speed of accessing data. Bad database design can slow down the data transfer. In a worst condition, the data will not be accessible and no information can be provided. In other words, the database must be precise and concise. Using those applicable systems, there are a series of important data that must be stored for future use. The data can also be used to generate information such as monthly reports, bills and receipts. The place to store all these data is known as database. However, current technique focuses on designing the database structure based on user requirement, but there is no consent whether the data store is used or not.
Before the computer system was introduced, people used files to keep all records. The same thing happens with the filing systems; the data must be well-organised to avoid misplace, error, wasting of time and redundancy. Designing the database is closely related to user needs. It involves a long process and starting to identify when the user requests for the system.

In producing a system or software application, there are four fundamental process activities: software specification, software design and implementation, software validation and software evolution [1-3]. Software specification is a process of writing the user requirements to get system specification [4, 5]. People can use existing standards technique to write documents such as using the natural technique as proposed by Heninger [6], the US Department of Defence and the IEEE. Another technique in writing system documentation that has been discussed by Pohl and Rupp [5] includes the famous and widely used technique, IEEE/ANSI 830-1998. Normally, the System Specification Document (SRS) will be produced as the deliverable for this phase to verify the user requirements with the system developer. However, in an established company, they usually use their own form as the standard way to show the user requirements. These user requirements will be used in the next phase to the software design and implementation. The system design is part of the software fundamental process as illustrated in Figure 1.1.

![System Design: Specify How System Works](image-url)
After that, the process of collecting the requirements is completed. Modelling or designing a software application in software design and developing phase is an important part in order to make sure that the development is congruent with the specification. A model can specify, visualise and document models of software system including their structure and design. During system design, the software engineer is needed to complete three main tasks, which are designing the system logic, user interface and database. These system logic, user interface and database will be used to counter back with the users before starting the coding. The counter back process assures that business functionality is complete and correct, end-user needs are met and program design supports the requirements for scalability, robustness, security and extendibility. The components involved in system design are illustrated as in Figure 1.2.

![Figure 1.2: The System Design Components](image)

Designing database is very important in system development [7]. It involves storing data and generating information for a long term process. Currently, there are two well-known techniques in designing database, which are the structured technique and the object-oriented technique. Even though the techniques have been introduced for more than 30 years, there are still some issues in designing database. One of the issues is designing optimal tables.
This thesis is a study that defines an alternative approach in designing optimal database system based on object-oriented concept. Optimal database means all defining attributes table in database are used. The discussion is made to present the technique on how attributes sanitization is applied in proposed technique which is not implemented in the current defining database technique, whether using the structure or object-oriented technique. The discussion includes process flow of the proposed technique, rules, example of implementation and the comparisons with current techniques to show that this technique is better than previous techniques.

Normally, in object-oriented approaches, Unified Modelling Language (UML) is used to represent the system requirements that consist of 13 notations such as use case, sequence diagram and class diagram [8-10]. This study has been done to consider all the features and the constraints in the class diagram to be used in producing the first schema table. Attributes sanitization is a technique to remove duplicated class or attributes in constructing the first schema table. Then, this schema table will be compared with the user interface to apply the second step which is to normalize the structure based on user interfaces. Upon completion of the process of normalization in the structure of schema table, the attributes sanitization technique is again applied to remove unused data in producing the final schema table. Then, the schema table is available to create the structure to implement the optimal database.

This chapter presents the background of this study including the related issues, aim, research objectives, scope and the limitation of study and lastly the organization of the thesis.

The third process in system design as illustrated in Figure 1.1 is software validation. Software validation was involved with verification and validation processes are used to ensure the development product are confirm to the requirements [11, 12]. The last process in system design is software evaluation. The evaluation is involved with the activities improvement the system after deliver to the used. Both last processes was not involved in this research and not used in verified the attributes sanitization in designing object-oriented to improve database design.
1.2 Problem Statement

System application consists of software design and database design. Software design will show the business process of the system. Software design will do the system logical and sketch the user interface. Another activity is database design to describe the flow of data involving in the system [13]. We cannot deny that software design and database design are tightly coupled. However, until now, both these designs are not congruent. Furthermore, the practitioners have distinguished these two activities in many ways.

In universities, students learn different subjects. System analysis and design subject represents software design. Another subject is database system or database design. This phenomenon also happens in organizations. Most of the companies break up into two positions, which are software engineers or analyst programmers for developing the system and database administrators (DBA) for managing the database. This situation is supported with strong reasons such as for security or to focus on particular jobs that can increase the quality of tasks. However, to break the tasks into two groups, there must be mutual understanding in order to get high quality system.

The situation is more chaotic when the development approaches in software development are broken into two techniques, which are the structured and the object-oriented approaches. The structured approach is also known as functional, procedural and imperative. This approach consists of two techniques, which are Data Flow Diagram (DFD) for software design [13-15] and Entity Relationship Diagram (ERD), which are used for data modelling to assist during the development of the database [16, 17]. The approach was popular in the 1970’s. Some programming languages that apply structures or procedures concept are Pascal, C and COBOL. The relational data model concept was introduced by E. F. Codd in 1970 [18]. It is a popular model and easy to implement. Entity Relationship (ER) diagram is a famous model to present the relation of the data [19]. Furthermore, Codd introduced normalization technique to avoid data redundancy [20]. He introduces three techniques of normal form to ensure that the database design is free from anomalies. However, the result of this technique has decomposed into too many tables.
The second approach is object-oriented. This technique, which uses Unified Modelling Language (UML), consists of various notations such as use case, sequence diagram and class diagram. Normally, during system analysis and design, the developer will produce use case diagram, state diagram and sequence diagram to confirm the understanding of the developer with the users regarding the system [5, 21, 22]. The class diagram is produced to show the interaction among the entities. In class diagram, it will show the behaviours of the class such as generalization, association and multiplicity. The class diagram also contains details about the class such as the attributes, name and data type [23]. PHP, C+, C# and Java are among the programming languages that support object-oriented concept. However, the UML does not have specific notation for database modelling. Normally, class diagram is used to design the database because these diagrams represent the semantic data [1].

In current practice for software development using object-oriented methodology, there are two famous ways on how people establish the database from designing the problem domain. The first technique is by transferring the class diagram to the appropriate tables after considering the class behaviour. Objects of the class are used without any translation between data structures in the design system and the program data structures [24]. This object-oriented approach seems to be a very attractive solution for modelling system based on object in system design and database design. It can be more meaningful especially if it is combined with the object-oriented programming technique and applied in programming language to manage persistence object. This technique gains its popularity when many references such as books and papers have been published. However, this system is not mature. In addition, it seems to lead to data redundancy in the object-oriented database design. Many researchers have given suggestions to consider the relationships principle in data model to avoid redundancy [25-28]. Since this technique is very young and not mature, further researches have to be done.

The second technique, which is a famous practice in object-oriented software development process, is hybrid technique. This technique allows the software engineer to use different approaches in designing the software. Software engineers use object-oriented notation to represent the system design by producing the diagrams such as use case, state diagram, sequence diagram and class diagram. Then, they use Entity Relationship Diagram (ERD), which is a relational data model to build the database design and Relation Database Management System (RDBMS) as
the repository [2, 16]. Developer will use normalisation technique to reduce redundancy. The developer combines the object-oriented technique in system design, while structured approach is used in database design. This object-oriented technique is workable to produce the system. However, there is lack of consistency checking from system design to database design. The consistency checking cannot be applied from design system to design database. Furthermore, the possibility to lose data while transferring diagram from object to ERD is very high. This technique is also lacking references because most of the references discuss structure technique or object-oriented technique.

Both structure technique and object-oriented technique are commonly used by the software developers. However, there was no consistency checking applied from system design to database design. Without consistency checking, it will lead to storing useless data in the database.

Therefore, in this research, an alternative technique to transfer the design system to database design based on object-oriented approach is identified. The class diagram produced in the system design will be used to create the schema table. This schema table will be compared with the user interface for normalization process. Lastly, attributes sanitization process will be applied to produce an optimal database structure. Optimal database structure referred to use of defining attributes in data model. This technique will apply continuous process from system design until the creation of database. Furthermore, the normalization technique using interface will ensure that only dedicated data are stored in database based on the user’s needs. The technique will remove the passive data to avoid the process of inserting and keeping the data. Furthermore, the technique will provide a platform to consistently check the system design and database design. On the other hand, this technique will provide better database compared to the current techniques. This technique will be compromised with both fundamental techniques to resolve the drawbacks.

1.3 Aim

The outcome of this research is to define the appropriate technique to produce the optimal database from the object-oriented design. The optimal database is achieved
when all defining attributes in schema table are used in displaying the user interface. The result must show the improvement in terms of design and storage consumption.

Well-designed technique will clearly show the flow in software development process. It starts from using defining class diagram until producing schema table. This research is primarily concerned with how efficiently the new alternative algorithms solve software development problem. It is hoped that this research would encourage the use of object-oriented technique in software development process including software specification and software design by using the Unified Modelling Language (UML) as a tool to draw notation to show that the class diagram produce in system design is harmonious with the database design.

1.4 Objectives of study

The objectives of this research are as follows:

i. To design the framework for producing an improved database structure based on attribute sanitization from class diagram.

ii. To formalize the rules for producing an improved database structure based on attribute sanitization from class diagram.

iii. To develop the tool based on (ii) in getting an improved database structure.

1.5 Scope and Limitation

The followings are the scope of this research:

i. Schema Table Focus on Class Diagram
   The class diagram is used as the base diagram to produce optimal schema table. The class diagram was considered in designing database because it contained semantic rules and represents data model [1, 5, 21, 23, 24, 29-31].

ii. Optimal Database
   In this research, the optimal database is considered achieved when the structure of schema table is used in the user interface. The defining attributes that are not
used by the user interface are considered as not useful and should be removed [32-34].

iii. Formalization of the rules
The entire transferring of rules from class diagram to schema table and comparing schema table with user interface is specified using mathematics such as logic, sets, and quantifiers in order to prove the concept of the ideas [35, 36].

iv. Proving the consistency rules
Two ways will be used to prove the technique in producing schema table from class diagram. First is using the defining rules. The idea was transferred to the rules as will be described in Chapter 4. The rules are defined to prove it consistent. Then, the rules are implemented in the proposed tool. Developing an application is done to prove that the research idea is workable. The software is developed using Microsoft Visual Studio 2010 to apply the rules in producing the schema table and the detailed discussion about the tool is in Chapter 5. The second technique is by using a selective existing system, which is chosen to verify that the technique follows the defining rules as discussed in Chapter 6. The discussion compares the result between the research technique and object-oriented technique with normalize technique.

1.6 Report Organization

The next chapter deals with related work and discusses features of the existing system design algorithm.

Chapter 2 also discusses current issues in designing database. There are two techniques in designing database, namely the structured technique and the object-oriented technique. The structured technique is defined as the database that is based on two diagrams, which are data flow diagram and entity relationship diagram. On the other hand, the object oriented is introduced nearly 30 years after the structured technique has been introduced. The object-oriented is used as class diagram to
produce structure of database. Chapter 2 elaborates further regarding the structured and object-oriented techniques.

Next, Chapter 3 illustrates the methodology used for the proposed Attributes Sanitization in Object-oriented Design for Relational Database System. This chapter will give clear picture about the research framework. The discussion details up the process involved in creating schema table from class diagram and in producing optimal schema table.

Then, the process flow is transferred to formalize the rule in Chapter 4. Four primary topics, which are the rules, algorithm, formalization and consistency rules, in creating optimal database schema from class diagram will be discussed.

It is followed by discussion on the tool in Chapter 5. To ensure that the defining process in Chapter 3 and defining rules in Chapter 4 are workable, the tool has been created using Visual Studio 2010 using C# language. Chapter 5 will give an example of Lecture Assessment System (LAS) to show the process in producing schema table using tools.

In Chapter 6, the manual implementation in the process of designing database design using object-oriented technique and the research technique will be discussed. Then, a comparison using LAS will be analysed and discussed. In Chapter 5 and Chapter 6, the application of the process of attributes sanitization in designing optimal database system will also be shown.

Lastly, the conclusion and the future work will be discussed in Chapter 7.
CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter covers the related and pertinent issues to this research. The literature review will be discussing the spectrum of research done in this area for supporting the research, specifically to find the gaps in producing schema table or designing the database structure. There are two techniques in designing database: structured technique and object-oriented technique. Structured technique is defined as the database is based on two diagrams, which are data flow diagram and entity relationship diagram. On the other hand, object-oriented used class diagram to produce the structure of database. This chapter will start explaining the differences between the data and attribute as discussion is based in database design, before discussing the techniques of designing database.

2.2 Overview of Data

Data is known as any facts that contain numbers or texts. It is a raw fact and need to be processed to produce information [37-39]. Data was started identify in requirement phase when studies the business process. Eliciting the requirement will show the flow of data involved in the system. Context diagram and data flow diagram are used in structured technique to illustrate this business process. Meanwhile in object-oriented, use case and sequence diagram are commonly used. Data is an element that describe things, events, activities and transaction [39, 40].
The next phase is design phase where data was analysed and organized as the system need.

In this phase, data model was established to show the conceptual structure [7, 41]. There are three basic elements used in the data modelling language which are entities, attributes and relationships. Class diagram or entity relationship diagram is an example of data model which is frequently used. In the design phase, each entity was described in detail. Entity is the basic symbol and depicted by a rectangle. Entity is a person, place, thing or event about which data is collected. A research done by Yangbo Xu et al. [42] showed how to extract table properties from the web to store as entity attributes. The entity name was taken from the table name. Each entity has name and details about what it is representing is called attribute. An attribute is a property of an entity and each entity may have many attributes. Each attribute will hold value and is called data. The data value can be changed but once the attribute is removed, the data value will be gone. In designing phase, Yuto Nakamura et al. [43] are suggested to use similarity words in giving the name for entity and attributes with the daily words used. Their research has shown that higher validity and understandability of the diagram if used same structure of class diagram.

2.2.1 Data Sanitization

Data sanitization is the process of making sensitive information in non-production database safe for wider visibility. The data should be sanitized or removed in order to protect valuable business information [44-46]. Data sanitization is also referred to as the process of removing resistant or worthless data. Some researchers use the terms data cleansing or data reduction [47]. The objectives of doing data sanitization are to get clean unused data and to produce quality data. Moreover, sometimes data sanitization is used to hide important data [46]. The technique can be applied before or after the data is stored in the database.

Formerly, database designer produces database structure for the system operation to fulfil user needs. The term data sanitization is not familiar in database design. However, the concept has been hidden applied since relational data model was introduced by Codd in 1970 [20]. To avoid redundancy in the tables, Codd
introduced normalization technique. Codd focused on data redundancy but ignored the usefulness of the data in the database.

The term data sanitization is formerly used in Knowledge Discovery in Database (KDD). Data mining is one of the activities in the process to discover previously ambiguous, duplicated and scattered data in database to define the technique to mine the data for gaining useful information. Many techniques have been proposed to mine the data including classification, clustering, association rule mining using rough set and neural network [47, 48]. Then rough set and neural network was improved by soft set theory that introduced by Molodtsov [49]. Other researchers have proposed the improvement techniques in soft set theory such as Herawan who proposes the alternative way to consider multi-value attributes reduction in information system [50].

Besides that, data sanitization concepts have also been introduced in other fields such as network data. Bishop et al [46] propose the idea of removing the selected information such as password and information that can be identified by individuals to hide sensitive data from passing through the network. Another research done by Defit [47] uses data cleansing technique to identify and fill the missing values in Jakarta Stock Exchange. Defit [47] combines several intelligent techniques, which are rough set, neural network, knowledge based and statistic to claim that the techniques offer better results in handling uncertain and missing value.

Literature study found that data sanitization or data cleansing was a very popular technique in removing the available data in the database but no research was done on attribute or column sanitization on the table structure in the database.

2.2.2 Attribute Sanitization

Data sanitization is a process of deleting data in records. It can be one data in a record or a set of data for a few records. Meanwhile attribute sanitization is a process of deleting a set of data for particular column. That means when an attribute is deleted, all the data referring to that column will permanently be deleted. Inspired by the fact that database may consist of unused attributes; this research proposed a technique to apply attribute sanitization in designing database with reference to object-oriented system design concept. Literature study found that attribute
sanitization is not popular among the researchers but attribute reduction which has a similar meaning was used by many researchers in their areas of research.

Long Giang Nguyen [51] did a research on attribute reduction in decision table using metric technique. This research proposed a new method for attribute reduction in information system. It was based on the subset of attributes which determines the knowledge structure on the set of objects. A metric on knowledge structure will be defined on the attribute sets to measure the similarity of knowledge structures. The researcher has published the technique which proves this metric method is more effective based on Shannon entropy. A research done by Xu Qisheng et al. [52] used mapping matrix according to the mapping relation between an attribute value and a decision value from decision table. They claimed that by using this mapping matrix optimal reduction set can be obtained compared to other techniques such as discrimination matrix, decision matrix or heuristic reduction. Their paper provides a detailed technique to apply mapping matrix with an example to prove their idea is workable.

Mingquan Ye and Chanrong Wu [53] did a research on attribute reduction using discernability matrix. Both researchers have proposed a decision table decomposition model to solve the attribute reduction problem based on discernibility matrix for large decision table. In their technique, they have introduced attributes partition. The large decision table was divided into a number of decision sub-tables and transferred the computing discernibility matrix in original decision table into sub-tables. They have attached a complete algorithm and example in their paper for reviewers. Their paper shows how this technique is applied and they claimed that their technique is efficient.

A research done by Jin Zhou et al [54] deals with attribute reduction in information system. They proposed a direct method of attribute relative reduction on rough set theory which is called tolerance relationship. The given algorithm is based on the concept of tolerance relationship similar matrix to calculate the attributes of incomplete information system. It applied attribute significance based on attribute frequency in the tolerance relationship similar matrix as the heuristic knowledge. To calculate the candidate attribute, they used binary search heuristic algorithm. Their paper also includes the algorithm, definition rule and examples of the proposed technique. Another research done on information system is proposed by Yuan Keyong et al. [55] using discernibility matrix. This research is based on the analysis of
the core attributes in information system and the cost of getting core attributes to identify whether a set contains the core or not. By using this technique, it takes less time and easier to achieve the smallest reduction.

Attribute reduction was also used by Xiongbin Wang et al. [56] in their area which was rough set theory. Their research was based on the relationship of many deferent attribute reduction definitions. It was known as relative attribute reduction. Their research paper established the algorithm of attribute reduction and the rules acquired from the decision table. They did a research to identify the relationship between attribute reduction based on the system entropy and on the database system.

Wang Yanbo et al. [57] have done a research on attribute reduction for network security. The research done is based on the network situation where the network equipment running status, network behaviour and user behaviour was saved in a log file. The information is very large, scrambled, repetitive and imperfect. There are some attributes in the log file which are useless. These researchers produced a paper that proposed a rough set reduction algorithm based on adaptive genetic algorithm. This algorithm was improved on cross probability, variation probability and fitness function. They claimed that their proposed technique has reduced time and enhanced the calculation far better than the traditional algorithm.

In this research, attribute sanitization process was applied in two processes. First sanitization process was done during the transfer from class diagram to schema table. Attribute sanitization was applied to remove same attributes in a same class. The second process was done after normalization process based on the user interface. Cross checking between schema table and user interface identified the attributes that were not retrieved in the system. This idea can improve the usage of storage through sanitizing of unused attributes in the database. Current practices in analysis and design system only focus on gathering user requirements to identify the flow of data to be stored in database. These current techniques do not consent with the data in the database. It has led the data to be stored that is used for nothing in the database for a long life of the system. The research has done with proposed the rules as in Chapter 4 to produce database structure.
2.3 Structured Approach

Modelling of a system is an essential process in software development lifecycle to produce a system model. A system model can provide the structure for problem solving. It can also be used as an experiment to explore multiple solutions and abstractions to manage complexity. It is not just only to understand the requirements and business process, but more than that, which is to suggest improvement process when the system is implemented. The models used should have traceability links among them. In other words, a system model must be consistent.

Structured technique was introduced in 1970. The old technique for showing the flow of data in structure system analysis and design (SSADM) is known as data flow diagram (DFD) \([15, 16]\). Using the data flow diagram, users are able to show how the system operation and data involved are used as the input and output at each process. It was introduced by Edward Yourdon and Larry Constantine in 1974 based on Martin and Estrin data flow graph model of computation \([58]\). Then, in 1977, Gane and Sarson introduced different notation for data flow diagram, which is more efficient \([15]\).

2.3.1 Data Flow Diagram

Data flow diagram is very popular and easy for end users to understand the physical idea of where the data they input has an effect to the process in the whole system. Data flow diagram can show the flow of data from external entities into the system and involvement of the data with the process. Data flow diagram has four notations: input/output, process, data flow and data storage \([2, 15]\). A data flow diagram mainly focuses on the processes or activities performing in the system. This diagram was established to represent the system process in a diagram.

2.3.2 Entity Relationship Diagram

In structured technique, the Entity Relationship Diagram (ERD) was used to produce database. Entity Relationship Diagram was popular a long time ago since Peter Chan wrote in ACM Journal in 1976 \([19]\). ERD is categorised as a top-down approach that
represents a concept of data. The diagram contains entity and relationships between entities. Chan defines entity as a thing that can be a person, company or event, while relationship is an association among the entities. There are three types of association, which are one-to-one, one-to-many or many-to-one, and many-to-many. The ERD is a basic for advanced entity relationship as implemented in object-oriented concepts [2, 16].

2.4 Object-Oriented Approach

Using object-oriented for software development, the system model can be developed by using Unified Modelling Language (UML). Unified Modelling Language (UML) is a modelling language that contains notations to represent objects-oriented concepts. It was established in the works of Grady Booch, James Rumbaugh, Ivar Jacobson and Rational Software Company. Since 1995 until now, UML has passed through many evolution processes for improvement such as from UML 0.8 until UML 2.3, which was released in 2010 [21, 29]. UML has many diagrams in view of system modelling. However, Class Diagram is the backbone of a modelling system because it represents the structure of the system that shows the system classes, attributes and relationship between classes [9, 16]. Furthermore, the graphical notation in a class diagram can be used to produce database structure.

Dobing and Parsons [59] in their interviews-based research with almost 2,700 UML practitioners and clients through the web find that use case diagrams and class diagrams have the highest usage levels followed by sequence, state and activity diagrams. Most of the diagrams are used to present the requirements; however, the class diagram is one of the diagrams available to use in producing the database design. The research study is based on class diagram as the initial stage to produce schema table as the diagram contains semantic data [1]. Furthermore, class diagram is also used to show entities, element of analysis and design such as associations, aggregations and generalizations among entities [1, 21, 23].

Currently, process modelling using object-oriented approach is already well-known [1, 9, 29]. This technique, which has thirteen notations, uses Unified Modelling Language (UML) that consists of various notations such as use case, sequence diagram and class diagram. UML is a language for specifying,
constructing, visualizing and documenting the software system and its components. It is an object modelling and specification language used in software engineering. It includes a set of graphical notation techniques to create abstract models of specific systems. Normally, during system analysis and design, the developer will produce use case diagram, state diagram, class diagram and sequence diagram to confirm the understanding of the developer with the users regarding the system [59]. When the users agreed with the design, the class diagram will be produced to show the interaction among the entities. In class diagram, it will show the behaviours of the class such as association, composition, aggregation and generalization [16, 60]. It also contains details about the class such as the attributes, name and data type [21, 29]. Association in object-oriented is the same as in structure technique was known as multiplicity. Multiplicity in object-oriented shows the relationship between two classes. Composition, aggregation and generalization are not defined in structured technique. Composition is shown as a class which is a whole of another class and has a strong relationship. Aggregation is described as a week relationship where a class is part of another class. Generalization is shown in a class diagram that one class shares the structure defined in one or more other classes. These four types of relationships were considered in this research as defined in the rules in Chapter 4.

Practitioners use class diagram for the development team member as their internal design [23, 29, 61]. This class diagram will be used as the initial part of designing the database system. The UML does not contain dedicated diagram for modelling database design. However, the class diagram is the only one which is the most suitable diagram to consider as the base for database conceptual modelling. A class diagram visually represents the semantic data [1, 23, 24].

Many tools [10, 31, 62-64] are available to represent UML notations to design system such as Rational Rose, Poseidon, Power Designer and MagicDraw. However, neither tool provides the facilities to generate optimal schema table from the class diagram nor the facilities to check consistency from the class diagram to the schema table. In the tools, the user must indicate which class will be the permanent or transient. In this case, the permanent class will be transferred to the table in schema table. Another class will transfer to the template program code such as C++. In this research the idea is to create a good schema table from a class diagram. This will help the database designer especially the novice database designer or those who are not familiar with database concept. Database is the heart of any system
application and designing database is a critical part. Even though it is hidden from the user, the database design must be fulfilled the user’s needs.

2.4.1 Class Diagram

A class in a class diagram [10, 65] represents a set of object with common features. A class graphically shows a rectangle that is divided into three parts: names, attributes and operations. The names for each class must be unique in the whole diagram and the other parts are optional. Among the classes, there are four types of relationships: associations, composition, aggregation and generalization. Association is shown as the relationship of the object among the classes. It is called multiplicity class diagram or cardinality on entity-relationship diagram. There are three types of multiplicities which are one-to-one, one-to-many and many-to-many. Composition and aggregation represent the whole and part relationships, when one class represents the whole object and other classes represent parts. The generalization is also known as inheritance, which is a concept in which the derived class can contain attributes and operation as based class. The derived class can also contain additional attributes and operations or less than the number of attributes or operations from based class. The class is defined based on the user requirements and classified based on persons, things, events or places. The class diagram is used to represent model data with ignoring the operations. The UML does not have specific notation for database modelling. However, the class diagram can be used to represent semantic data [1].

As shown in Figure 2.1, there are four (4) classes which are Subject, Staff, SubjectFTMM and Student. The figure also shows two relationships which are generalization and association. The generalization was shown between SubjectFTMM and Staff. Meanwhile the associations between those classes are shown between SubjectFTMM and Staff, and SubjectFTMM and Student.
The class diagram will be the input for process generating schema table. The standard concept in the class diagram as class behaviour is described as follows:

i. Class diagram consists of classes and relationships.
   - A class is defined as the set of shared attributes and behaviours in each object in the class.
   - Relationships are associations between entities. They are referred to as data associations.

ii. A class consists of name, attributes and operations.
   - Class name is used to distinguish the defining class from all other classes in a class diagram.
   - Attributes are used to describe the class.
   - Operation is defined as the process to invoke on the attributes based on system requirement. However, the operations are not used in designing the structure of table.

iii. Among the classes, there are four types of relationships namely associations, composition, aggregation and generalization.
   - Associations show the relationship of the object among the classes. It is called as multiplicity class diagram or cardinality on entity-relationship
diagram. There are three types of multiplicities which are one-to-one, one-to-many and many-to-many.

- Composition relationship is when one class represents the whole object of other classes.
- Aggregation relationship is when one class represents the part object of other classes.
- Generalization is also known as inheritance, which is a concept where the derived class can contain the attributes and operations as based class. It can also contain additional attributes and operations or contain less than the number of attributes or operations from based class.

2.4.2 User Interface

An interface is defined as a point of interaction between two systems or work groups. It is very important as a reference point of interaction between human to system and system to human. There are many types of interface including natural language, question answer, menus, form fill, command language, graphical user interface (GUI) and a variety of web interfaces [16]. Users use the interface to supply data for store in the database and to retrieve it back. The information will be displayed after the system retrieves the data and processes it for the user’s needs. Most of the time, users are not familiar and do not concern with the structure in the database. They do not bother with the database design. However, users are really consent to have information in the interface to fulfil what they want. In other words, the interface helps users to get information they need in and out of the system [16, 66]. Generally, there are two purposes of the interface. Firstly, interface is used as presentation language, which is the computer to human part of transaction, and secondly, it is used as action language, which is the human to computer part of transaction.

Many researchers work on how to create the interface with the focus on users’ satisfaction. Many books have also been written focusing on a good interface according to the user’s likes preferences [1, 67]. No authors write about the impact of interface to database structured. Most of them work on retrieving the information from database or report to attract the users’ attention. However, a research done by Dzafic et al. [68] proposed a technique to design database and user interface in
object-oriented with the aim of reducing time maintenance. Reducing the time in design database will significantly reduce the cost. Dzafic et al. has presented their work with an example and found that the technique proposed has increased the number of database table and increased the response time.

In this research, the attribute is referred to the data which is channelled through the interface to be utilized but does not consider the graphics such as colour. The user interface was an important input in this research because it was used to create the final database structure after making comparison among the schema table from the user requirement and the user interface. During the process of gathering user requirements, a lot of data are requested to be stored in the database. The users believed that these storing data will be used in the future. However, after the system was completed and delivered to the user, not all the dreams came true. There are many cases where the data stored in the database is never used. To avoid this, the proposed method as suggested in this research is by tuning the structure of the database through removal of the unused data in the database design. This method can reduce the storage and increase the database performance. It is based on the computer principle dealing with input and output concepts. This technique will be further described in Chapter 3.

2.5 Normalization Technique

Normalization is a formal technique to reduce redundancy by analysing the relationships based on primary key or candidate keys and functional dependencies [18]. It is implemented from lower steps to higher steps: First Normal Form (1NF), Second Normal Form (2NF), Third Normal Form (3NF), Boyce-Codd Normal Form, Fourth Normal Form (4NF) and others. Each step corresponds to a specific normal form that defined the properties to reduce data redundancy. The critical part in this normal form is to define first normal form. Even though there is a series of normal form, the users are encouraged to apply to at least Third Normal Form to avoid update anomalies [7]. All the normalize forms are based on the relational database. This technique is mature and well-formed. However, the drawback of this normalize of forms is that when a higher normal form is applied, the database structure is becomes less vulnerable and the number of tables increases [7, 13, 41]. Decomposing
the tables will increase the number of tables in displaying information, decrease accessing time and make the process of updating, accessing and programming much more complex.

2.6 Database Design

Database in general is known as allocation storage in computer to store data. Database is a shared collection of logically related data that contains description of the data defined according to the organization’s needs [2, 41, 69, 70]. Usually, database put in a computer act as a server and is accessible by other computers recognised as clients. Since a server is needed to serve many clients, all efforts must be made when designing the database during the system design phase. Normally, there are three stages in designing database, which are conceptual, logical and physical design. However, this research focuses on logical database design only.

Logical database design focuses on identifying the data flow during eliciting user’s requirements. Normally, the designer establishes the entity, attributes and relationship between the entities. Designing logical database requires the software engineer to comprehend thoroughly the overall business process. There are two main approaches in designing database that are referred as bottom-up and top-down. The bottom-up approach starts with identifying the attributes and relationships. This approach is quite suitable for a simple database with a limited number of attributes.

This research focused on top-down approach because this approach is appropriate for designing any type of database whether it’s simple or complex database [7]. This approach starts with designing data model that contains entities and relationships to show data semantic. In structured technique, the Entity Relationship Diagram is usually used to represent the entities, relationships and attributes. In UML technique, the data semantic can be derived from class diagram.

In current practice, a schema table is produced either from entity relationship diagram or class diagram, depending on the design approach. A schema table is a description of tables and its attributes [25]. This structure table will be used to create physical database. However, normally, before the schema table is implemented in physical database, the normalization technique that was introduced by Codd will be applied to avoid the anomalies or data redundancy [20].
In this research, the schema table is constructed from class diagram with considering the semantic data. Attribute sanitization in class diagram will eliminate redundant of class name and redundant attributes in class diagram. Then the user interface is used as normalized process to structure the table as wanted by users. Finally, the fields in user interface will be compared with the schema table. If any attributes in the schema table are not accessible from user interfaces, then it was suggested for the schema table to be removed.

2.6.1 Database Design Using Structured Technique

In structured technique approach, the user defines the database based on two diagrams, which are data flow diagram (DFD) and entity relationship diagram (ERD). Data flow diagram is used to represent the user’s requirements. System analysts will attempt to understand the user’s needs and business process in both diagrams. Then, these diagrams are used to get confirmation from the users.

Through the structured analysis technique with the Data Flow Diagram, the four symbols which are process, data flow, data store and entity are used to create a pictorial depiction of process that will eventually provide system documentation. When the system documentation is accepted by the user, the next step is to collect the data stored in Data Flow Diagram to visualise in Entity Relationship Diagram. Then, the schema table will be produced based on the defined Entity Relationship Diagram.

After producing schema table, there are two steps to implement the schema table to physical database system, done by system analysts. The first step is that they will strictly create the database structure based on the schema table. Another step is the systems analyst will perform current normalization technique, as introduced by Codd, before the implementation. Although, applying normal form is optional, it is recommended to do the process at least up to third normal form [7]. When, the schema table were normalized until Third Normal Form, it has complied with the Codd basic rule to set of functional dependencies for each relation and each relation has a designated primary key. The structured technique in designing database system is illustrated in Figure 2.2.
REFERENCES


