CHAPTER 3

RESEARCH METHODOLOGY

3.1 Introduction

This chapter describes the research methodology applied in the current research. It further describes the selected research methodology to achieve the defined research aims as presented in chapter 1. This chapter includes an illustration of the methods undertaken.

Three phases are illustrated: the literature review phase, the DQM development phase, and the evaluation phase for the proof of concept. Figure 3.1 presents the critical activities used for this research corresponding to the different phases.

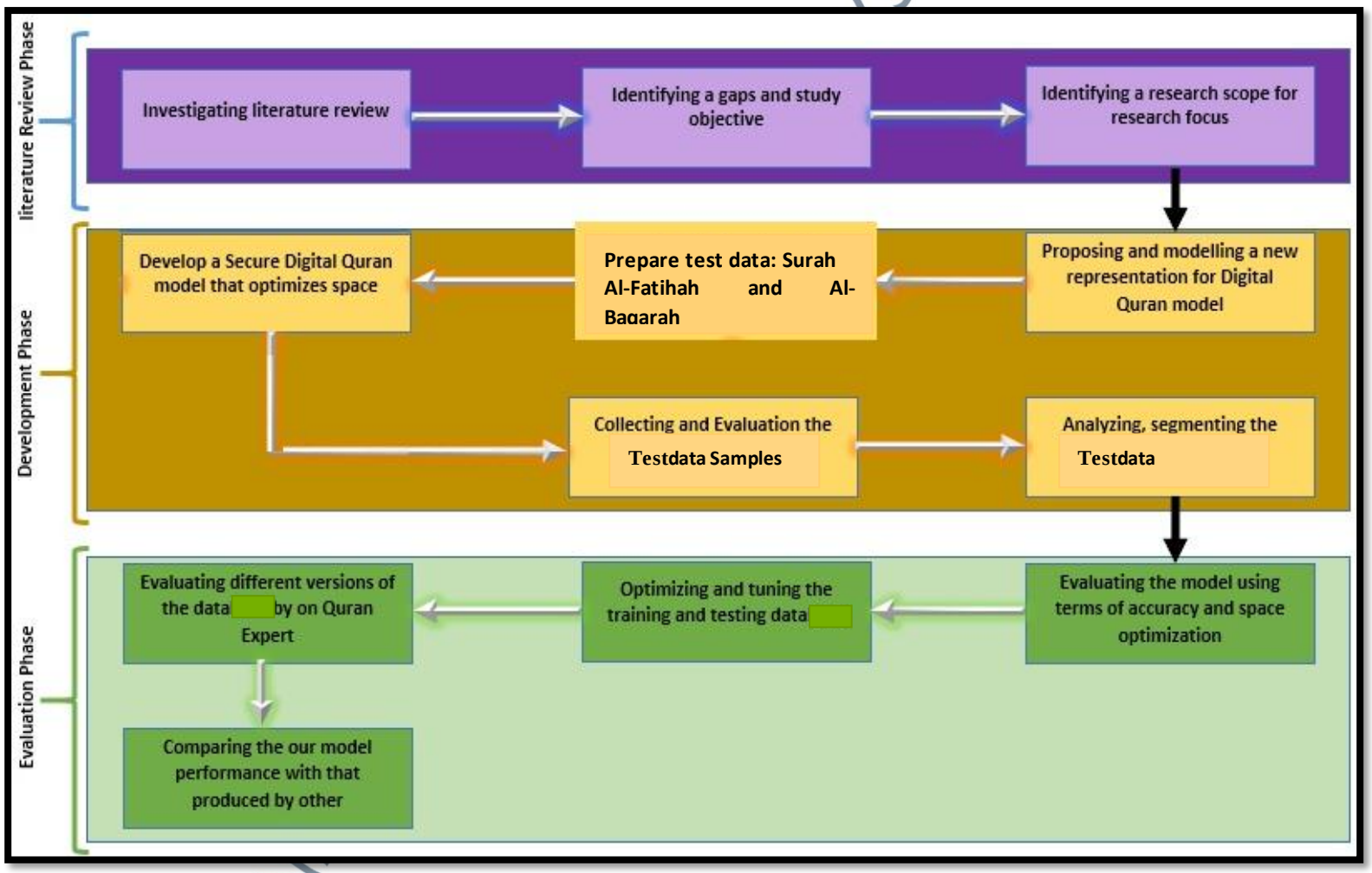


Figure 3.1: Different Stages of the Research Activities

3.2 Proposed Digital Quran Text-Based Format Using Hexadecimal and Compressed Matrix Structure

This study proposed a novel approach for representing Quranic words using Unicode transformation format (UTF) that utilizes the UTF-8 character encoding, which is backwards compatible with ASCII code. Unicode is the worldwide character coding standard for representing characters, whereas UTF-8 is an alternate coded representation format for all Unicode characters that maintains ASCII compatibility (Kurniawan et al., 2014). Three algorithms need to be developed as a mechanism to optimize the storage space:

a. **Quranic word representation in Hexadecimal representation**.

   Instead a memory is allocated to each letter in words, and an algorithm is formulated to represent a Quranic word in Hexadecimal that requires only one memory location.

b. **Compressed sparse matrix for digital Quran content structure**

   The Quran content is stored using a matrix-like structure or two-dimensional array. Since the length of each verse varies from one letter to 129 words, each row of the array representing the verses is not equal in length which resembles a sparse matrix. A compressed algorithm for this content structure needs to be developed.

c. **Lookup up table for unique Quran words with a unique ID**

   The lookup table stores the unique ID of each unique Quran word, acting as the indexing list for the structure of the digital Quran content arranged according to sequences of verses and words in the actual text. Content will be organized using

this unique ID, thus reducing repeating words that have a larger size than the unique numbers.

The details of this mechanism approach is explained in the following sections.

3.2.1 Arabic Letters and Words Conversion to Hexadecimal Representation

Letters conversion is the backbone of the proposed technique in this study using UTF-8 Arabic Presentation Forms (A Range: (FB50–FDFF) and B Range: (FE70–FEFF)). Unicode Standard 7.0 for Arabic characters for character encoding, taking into consideration the position of the letters on the word, which are either initial, medial, final or isolated. Figure 3.2 show the Unicode for Arabic Letters in Hexadecimal according to the position of the letters.

Figure 3.2 The Unicode Standard 7.0, Copyright © 1991-2014 Unicode, Inc., Arabic Presentation Forms-A

The general context of the orthographic Arabic word consists of a sequence of Arabic characters, and Arabic word is a finite set of Arabic characters that can be denoted as Equation 3.1:

woe $W(h_n) = L(h_1 + h_2 + \cdots + h_n)$

(3.1)

Where W is the Arabic word, h is each letter (half) in the word with its hexadecimal value with a finite length, and L is the conversion formulation for the Arabic word. The representation of an Arabic word W in hexadecimal can be denoted as equation 3.2:$h_n = \sum_i^n h_i$ (3.2)

h_n Represents an Arabic word with a finite length $n$whereit combines each character value in hexadecimal form into a new hexadecimal value. For example, the word (الرحيم) AL RAHIM; this token is represented orthographically as follows, where 6 characters re converted to 6 hexadecimal values computed into hexadecimal word format, refer to formula (3.3).W (الرحيم) $= L($ ا + ل + ر + ح + ي + م $)$

$$= L\,(FE8D + FEDD + FEAE + FEA3 + FEF4 + FEE2)$$

$$= 5F893 \qquad (3.3)$$

The word representation is calculated according to formulas 3.1 and 3.2 referring to Unicode Table according to the letter's position as in Figure 3.2. The algorithm for the word conversion was developed as discussed in Chapter 4. Table 3.1 shows the letter conversion and word conversion that is proposed in this study. The storage size reduces

from 6 locations to only 1 location that stores the new code of the word itself instead of letters.

Table 3.1: Comparison between letter and word conversion in terms of storage.

Letters		ا	لـ	ـر	ـحـ	ـيـ	ـم
Unicode		$FE8D$	$FEDD$	$FEAE$	$FEA3$	$FEF4$	$FEE2$
Size (bytes)		2	2	2	2	2	2

Word	الرحيم
New code	5F893
Size (bytes)	5

Total storage size: 12 bytes for the word الرحيم with the letters approach and 5 bytes with the word approach.

In the previous approaches, Al-Mazrooei et al. (2020) and Hakak et al. (2017) use string concatenation where the word **الرحيم** will be represented by string concatenation as in FE8DFEDDFEAEFEA3FEF4FEE2 which requires 12 bytes as compared to 5F893 which occupies only 5 bytes. Thus, word representation in Hexadecimal reduces the spaces required for each letter in the word.

3.2.2 Compressed Matrix Representation for Quran Content

For the Holy Quran, there are 6236 verses where the shortest verse has one word only, and the most extended verse has 129 words which are in Al-Baqarah verse 282 (https://qurananalysis.com/analysis/basic-statistics.php). Thus, if we present the whole Quran in a table or matrix, the dimension is 6236 rows, and a maximum of 129 columns where each row represents one verse and the column represents the word in the verse as in Equation 3.4 illustrated in Figure 3.3. $Memory\ Size = 6236\ row \times 129\ column \times Max\ size\ in\ Bytes\ for\ word$

$$= 4.022\ MB\ (Assumes\ max\ size = 5B) \quad (3.4)$$

word / Verse	w_1	w_2	w_3	....	w_i
v_1	ID_{11}	ID_{12}	ID_{13}	....	ID_{1i}
v_2	ID_{21}	ID_{22}	ID_{23}	....	
v_3	ID_{31}	ID_{32}	ID_{33}	....	
⋮	⋮	⋮	⋮	⋮	
v_i	ID_{i1}	ID_{i2}			ID_{ii}

Figure 3.3: Sparse Matrix Represent Words and Verses of the Quran

There are two concerns for space optimization

a. The space without word entry in the matrix is a waste of space.

b. Duplication of words that reoccur in other verses (rows).

The next space optimization method proposed to represent Quran text is the sparse matrix which is table matrix populated with many zeros. In the mathematical subfield of numerical analysis, a sparse matrix is a matrix populated primarily with zeros (Stoer & Bulirsch, 2002). For instance, the matrix has $\boldsymbol{v}$ columns $\boldsymbol{w}$ rows, if the number of non-zero elements is *s*, *total-s* is much larger than *s* where *total* elements in the matrix are $\boldsymbol{v} * \boldsymbol{w}$, and we will say the matrix is sparse. Figure 3.4 demonstrated the space requirements for the sparse matrix which (4x4) x (4x1) = 64 memory allocations.

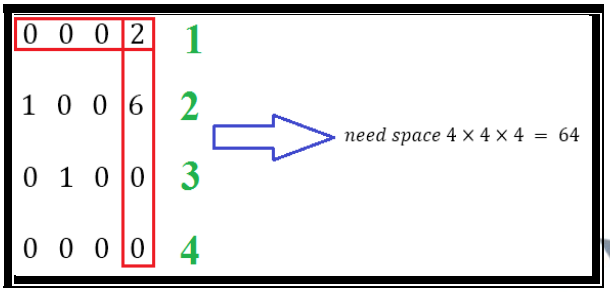


Figure 3.4: Sparse Matrix Populated Primarily with Zeros

This matrix is a typical sparse matrix; the total number of elements for the first vector is 16. It has 12 zero (0) elements, which is primarily with zeros. Using a two-dimensional array to store a v row w column sparse matrix, if every array element needs L binary to store it, then the total storage need for this sparse matrix is $V * W * L$. Thus, in this matrix, most of the space used to store zero elements is wasted while taking huge spaces to store item zero.

Sparse Matrix computing includes so many different operations, for example, addition, scalar multiplication, transpose, and vector multiplication which is one of the most important computations and implementation (Dakuan, 2009). Due to the low computation and higher communication characteristics of a sparse matrix, a simple compression algorithm with double offset indexing was also introduced to further optimize the space for the matrix representation. Therefore, to optimize the space, a better approach is to only store non-zero items (Norwawi, 1994). To find an item in the matrix is easy if we know which row and column the item is located. Hence, in a sparse matrix, to save space, a new data structure that only defines non-zero items only via the row and column is required as the compression mechanism.

3.2.3 Sparse Matrix with Double Off Set Indexing

A wide variety of Markov chain and grid structured models can be created using variable Arrays. The basic idea is that when looping over a range ***for each*** block, the loop counter *i* can be accessed and use expressions of the form (*i-k*) or (*i+k*) where *k* is a constant integer. This is called "offset indexing" and allows you to link the array element at index *i* with index *i-k* of the same array or another array (Minka et al., 2018). The 2-dimension array structure is compressed to only store the non-zero element. For example, Figure 3.5 shows how the matrix is represented in double offset indexing.

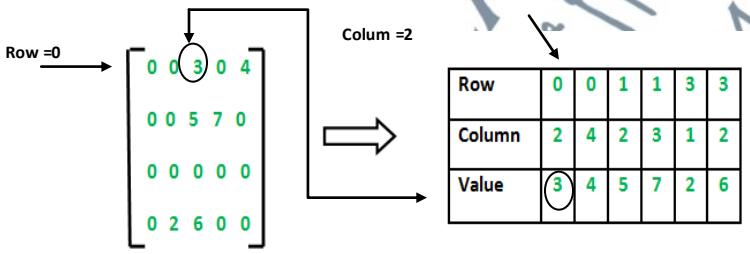
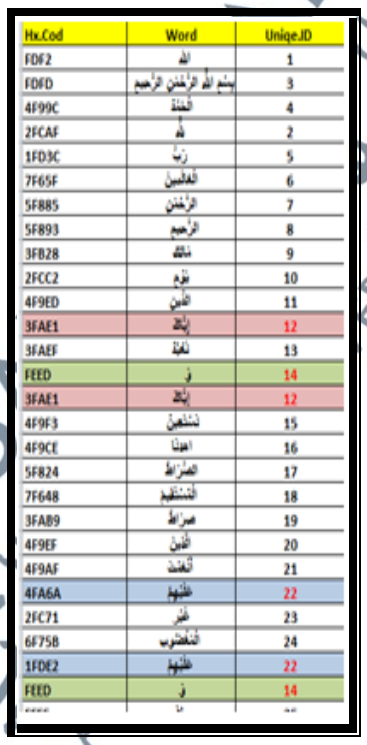


Figure 3.5: Sparse Matrix with Double Offset Indexing Representation.

The compression technique called double-offset indexing will reduce the zero elements in the matrix thus minimizing the storage requirements of a sparse table by eliminating default entries. However, the indices of a non-default entry must be stored in the compact table to facilitate non-default entry lookup as shown in Figure 3.5. A given row or column index can be repeated multiple. An algorithm was developed for the compression method that tries to eliminate such redundancy and take advantage of default entries as presented in Chapter 4.

3.2.4 Handling Word Duplication of the Digital Quran Content Structure in the Matrix Representation

In order to handle the repeating words in the digital Quran content structure, all unique words in the Quran is listed in a look up table with a unique ID for each word. Thus, duplications were handled by creating an index listing of all unique words in the Quran. A word representation lookup table was created with three columns which are the Hexadecimal code of the word, the Arabic text and the unique ID as shown in Figure 3.6.



Hx.Cod	Word	UniqeJD
FDF2	الله	1
FDFD	بسم الله الرحمن الرحيم	3
4F99C	الحمد	4
2FCAF	لله	2
1FD3C	رب	5
7F65F	العالمين	6
5F885	الرحمن	7
5F893	الرحيم	8
3FB28	مالك	9
2FCC2	يوم	10
4F9ED	الدين	11
3FAE1	إياك	12
3FAEF	نعبد	13
FEED	و	14
3FAE1	إياك	12
4F9F3	نستعين	15
4F9CE	اهدنا	16
5F824	الصراط	17
7F648	المستقيم	18
3FAB9	صراط	19
4F9EF	الذين	20
4F9AF	أنعمت	21
4FA6A	عليهم	22
2FC71	غير	23
6F75B	المغضوب	24
1FDE2	عليهم	22
FEED	و	14

Figure 3.6 Example of a Lookup Table for Indexing Unique Quran Words with Unique ID

The Quran content consisting of the chapters, verses and words are arranged according to the unique ID of the words instead of the Arabic characters. Repeated words will appear in the content structure with its unique ID. For example, Figure 3.7 illustrate the use of look up tables as in Figure 3.6, unique ID to represent the content of surah Al-Fatihah

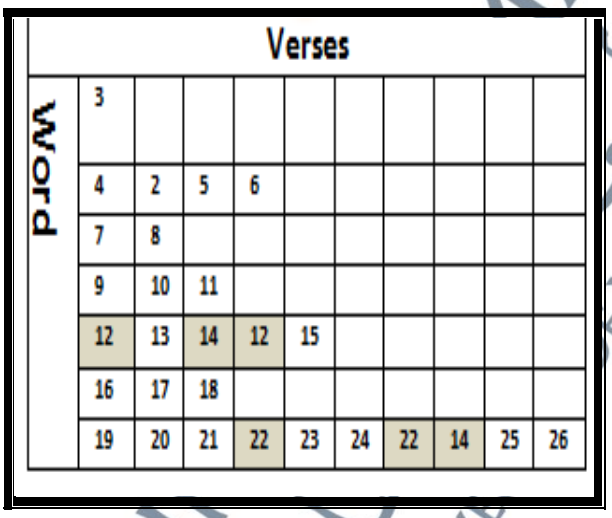


Figure 3.7: Al-Fatihah Content Structure with Look Up Table and Unique ID

For example, number 12 is repeated in verse 5 representing the word اياك that occurs twice in verse 5 of Al-Fatihah إِيَّاكَ نَعْبُدُ وَإِيَّاكَ نَسْتَعِينُ

.

3.3 Development Phase: Digital Quran Model Implementation

The development phase is one of the standard processes in the methodology flow, which aims to identify the essential procedures and steps to construct the model, design, and analyze the proof of concept. The development life cycle is defined as a process of building the conceptual DQM model, implementing the proof of concept, and modifying and manipulating the prototype to achieve the desired output or performance. The research used a design-based approach with a development method (construction of techniques) to produce the conceptual model of a new representation for the digital Quran by handling word duplications to optimize space. The model is enhanced with an authentic digital Quran model with tamperproof and content integrity features to bridge the research gap for digital Quran research and development.

3.3.1 Creating Quran Test case Data: Surah Al-Fatihah and Al-Baqarah

For proof of concept of the proposed Digital Quran model with the 3 mechanisms developed for storage optimization discussed earlier, Surah Al-Fatihah, being the most frequent Surah recited in a day and Al-Baqarah being the longest Surah, were chosen as the test cases. In this regard, the Quran data is collected from *tanzil.net* which is a reliable and credible source for the Digital Quran source under the University of Leeds Quran research group. A Quran expert was also referred to during the data collection, experimentation, and evaluation of the proposed model. Through these measures, the current research established a testcase data that will address the research objectives and questions.

3.3.2 Consultation with Domain Expert

Consultation with the Quran expert was conducted to get their insight on the issue. They also have a broader viewpoint of the whole problem that helped improve the proposed solution significantly. This further allowed the researcher to better structure the research and its approach and gain an in-depth understanding of the topic at hand. These semi structured interviews that are qualitative approach have been conducted by the researcher to yield comprehensive results that can be beneficial for both scholars and practitioners alike. The report of consultation is provided in the next chapter of this study.

3.3.3 Design and Develop the Digital Quran Proof of Concept

In this thesis, two conceptual models of Digital Quran are proposed: (i) new representation for digital Quran using Hexadecimal UTF-8 Arabic Presentation Unicode Standard 7.0 and (ii) enhancement of the model which was further developed into a Digital Quran model to further optimize space with antitampering and content integrity, executed in three phases as shown in Figure 3.8.

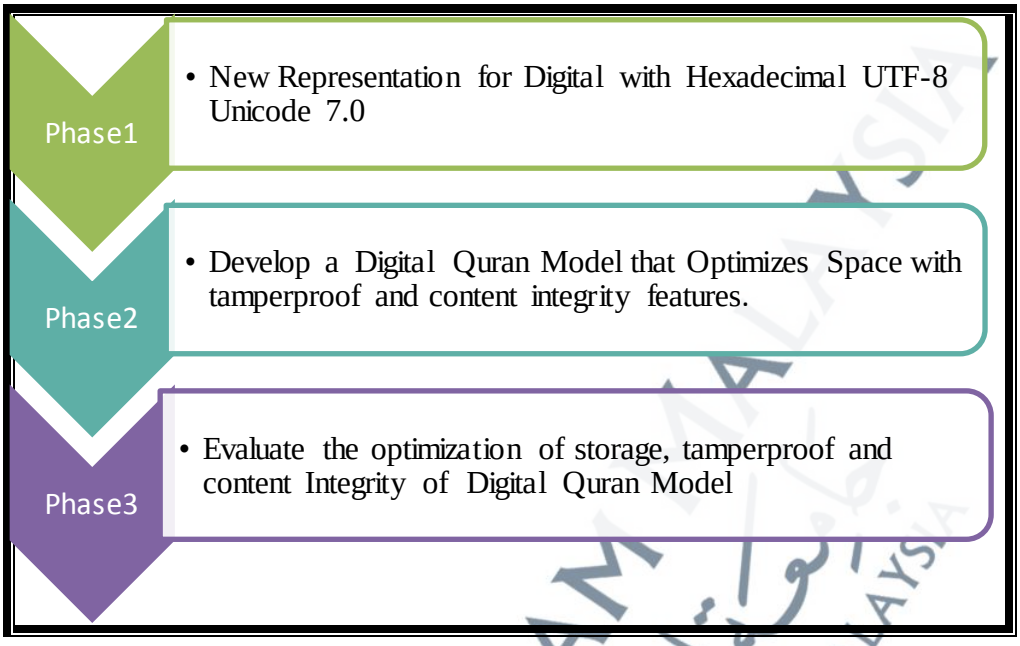


Figure 3.8: Phases for Digital Quran Prototype Development

The key emphasis in this research is on mapping of words of verses in Al Quran into Unicode 7.0 in Hexadecimal format. Al-Quran words are mapped into an array of letters and matched each letter with its current position in that word either on the beginning, middle, isolate or last with its corresponding hexadecimal Unicode. The word representation in Hexadecimal will be computed.

As mentioned earlier, the storage of words was optimized with one memory space for that word rather than one memory space for each Arabic character. For example, for the word الرحيم is represented in separated space but only 1 space in hexadecimal shown in Figure 3.9 as described also in Table 3.1.

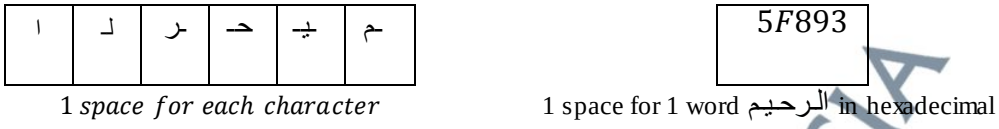


Figure 3.9 Comparison of Memory Space Allocated with Original Text and Hexadecimal Transformation.

On the other hand, all verses are represented in a matrix form consisting of ayat (row) and words in the ayat (column). Due to the variable length of each ayat, the matrix becomes a Sparse Matrix, each row and column contains the words in each verse. To address the waste of space in the sparse matrix, a double offset indexing compression algorithm has been used to optimize the size and storage of the matrix. Furthermore, duplications are handled by indexing all unique words in the al-Quran and restructuring the verses based on the indexed ID.

The backbone of the prototype was written in Visual Basic 6.0 and Java programming language taking advantage of VB 6.0 to reduce the cost of deployment per user. End-users of DQM can run the application using only a browser; no special software aside from the appropriate browser needs to be installed on their computers.

VB is a component integration language that is attuned to Microsoft's Component Object Model (COM), COM components can be written in different languages and then integrated using VB; hence the interfaces of COM components can be easily called remotely via Distributed COM (DCOM), which makes it easy to construct distributed applications, as a result, COM components can be embedded in linked to your applications user interface and also into stored documents (Object Linking and Embedding "OLE", Compound Documents).

In the earlier stage of the study, DQM was developed in Java programming language taking advantage of abstract classes and threading used for various pre-processing steps. The prototype was designed and develop according to the flow show in Figure 3.10.

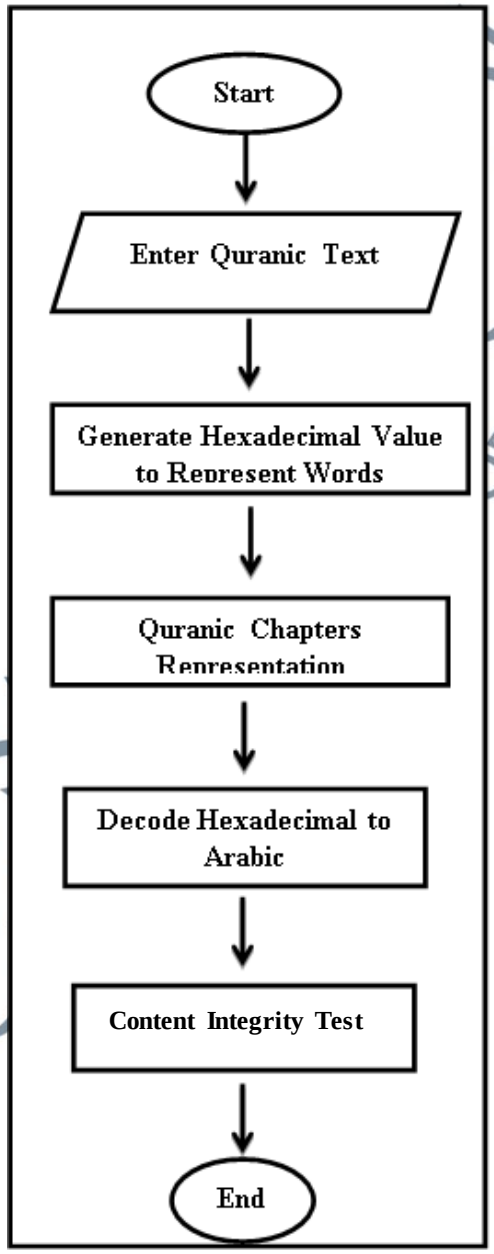


Figure 3.10 Flowchart of the Prototype of DQM functions

3.4 DQM Evaluation Phase

In the evaluation phase, an experimental approach was used to measure and evaluate the proposed model and compare the digital Quran model's performance result with the related work, which served as the performance evidence of the proposed model. Digital Quran Model (DQM) implementations were conducted based on the three parameters mentioned (space, tamperproof, content integrity).

The main factor in the evaluation process is the solution quality, which was measured by comparing the size of the file before applying the DQM algorithm. For example, the word (الرحيم) AL RAHIM, the new hexadecimal representation is 5F893 which yield about 58.33% of space reduction per word which will spill over to 148 times occurrences in the whole Quran.

In the proposed DQM, the Quranic code is created at three levels: character or letter, word, and verse. At the character level, letters are translated into its UTF-8 character encoding for the Arabic characters in the Al- Quran which is the basis of DQM, and then compute the word representation in hexadecimal in the verses.

The evaluation on the performance of the proposed DQM is based on

a. Storage Space

Comparison with its original size in bytes, reduction due to word conversion into hexadecimal, content structure of verses using compressed sparse matrix as in Table 3.2.

Table 3.2 : Evaluation on the Storage Optimization

Storage Size / Test Case	Original Text in Arabic	With Words Conversion into Hexadecimal	Sparse Matrix Content Structure	Sparse Matrix with Double Offset Indexing
Selected Quran words				
Al-Fatihah				
Al-Baqarah				

b. Content Integrity

Besides the actual Quran text, three types of text were used to evaluate

i. Quran text with missing letter

ii. Non-Quranic text

iii. Non-Quranic text embedded with some words exists in the Quran.

c. Content Validity by Quran Experts

Quran experts were involved in recommending the database collection, verify the experimentation, and evaluation of the proposed model. The experts were:

i. Dr. Bashar Igried Deb Alkhawaldeh, Department Chair Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, The Hashemite University, Zarqa, Jordan.

ii. Dr. Mohammad Al Khaldy, assistant professor at Khwarizmi University Technical College (KUTC) in Jordan.

3.5 Summary

This chapter provides a summary of the research activities employed in this thesis. DQM key space optimizations algorithms for (i) representing Quran text using hexadecimal representation for letters followed by words and verses conversion intelligently reduce memory usage using the presentation layer format of Unicode Hexadecimal UTF-8 for character encoding which is backward compatible with ASCII code and (ii) compressed Sparse Matrix with double offset indexing to further improve the memory management and finally (iii) handle repeating words with Lookup table of all unique Quran words to cross reference the content structure with unique ID. Then a proof of concept is implemented to as workable demo of the proposed model which then be evaluated with certain protocols. The next chapter shall describe the model design and implementation used in this research.