CHAPTER 6

RESEARCH

The aim of this research is to evaluate the effectiveness of artificial intelligence in the design phase of software development and thus draw a conclusive answer on whether artificial intelligence is preferred over the traditional method and to deduce a viable explanation over the findings of the research. The basis of comparison are on the following fields:

* 1. **Time:** The research will attempt to find the time saved by incorporating artificial intelligence in the software development process.
  2. **Accuracy:** The research will attempt to calculate the accuracy of the generated diagrams, which have been generated using artificial intelligence.

As has been stated, the aim of this research is to use class diagrams as an example of an essential design diagram involved in the design phase of software development, and in extension, a class diagram generator which uses natural language processing has been developed to compare against the traditional pen and paper method of making class diagrams.

# 6.1 RESEARCH METHODOLOGY

## 6.1.1 DETERMINING THE ACCURACY OF THE CLASS DIAGRAM GENERATOR

The accuracy of the class diagram generator is measured by subjecting it to a number of test problem statements and calculating the number of generated classes, methods and attributes. The metrics used for this purpose are given below: (Add reference to paper)

* 1. **True Positive (TP):** Number of classes, methods or attributes are correct and are generated correctly.
  2. **False** **Positive (FP):** Extra classes, methods or attributes are generated and are incorrect.
  3. **False** **Negative (FN):** No classes, methods or attributes generated.

From the accuracy metrics, two factors are then calculated:

* 1. **Precision:** It is a measure of how close the resultant diagram is to the correct diagram.
  2. **Sensitivity:** It is a measure of how sensitive the tool is in detecting classes, attributes and methods.

The test problems used for testing the accuracy are as follows:

| Table 6.1: Table of problem statements | |
| --- | --- |
| **Statement Number** | **Problem Statement** |
| S1 | A system shall allow customers to create and login to their accounts using their username and password. |
| S2 | A mall has a collection of perishables and non-perishables. Perishables have fruits, vegetables and meat and can expire. Non-perishables have plates, spoons and forks and can degrade. |
| S3 | A library consists of books, novels, magazines, journals and has a librarian and an assistant. The librarian has a unique username and password to login to the system and maintain and add new records. The assistant also has a unique username and password to access the system and update the records. Members have a member-card to borrow and return the items. The member-card contains information like member-id, name, age, email and address. |
| S4 | A system shall allow the customers to register and login by entering their username and password, in order to get access to the store and buy different products. The store consists of phones, laptops, tablets and desktops to sell and has an administrator. The administrator has a username and password to login and maintain the system. Payment methods include cash, card, cheque, UPI etc. |
| S5 | An online grocery store should allow user to login to the system with their username and password to browse through a list of groceries and buy them. Payment can be done using card, net-banking, UPI. |

## 6.1.2 COMPARISON WITH TRADITIONAL METHOD

For comparing the proposed class diagram generator with the traditional method of generating class diagrams, the speed at which the diagrams are generated and their accuracy are taken into account.

**6.1.2.1 Experiment using the traditional method**

The research has been performed on a population of 10 final year students from the Department of Computer Science & Engineering, Assam Don Bosco University. The class diagrams and the problem statement were presented in English. Each subject performed the experiment in a controlled and supervised environment and each subject was instructed to not use any external aid to generate the class diagram from the given problem statement.

The test subjects were given the following problem statement:

*“A library consists of books, novels, magazines, journals and has a librarian and an assistant. The librarian has a unique username and password to login to the system and maintain and add new records. The assistant also has a unique username and password to access the system and update the records. Members have a member card to borrow and return the items. The member card contains information like member-id, name, age, email and address.”* (S3 from Table 6.1)

Once the problem statement has been received by the subjects, a timer has been set to calculate the following times:

* 1. **Analysis Time (AT):** This is the time taken by the subjects to read and understand the statement completely before proceeding to draw the diagram.
  2. **Diagram Generation Time (DGT):** This is the time spent by the subjects to draw the diagram.
  3. **Overall Time (OT):** This is the overall time spent by the subjects to fully complete making the class diagram.

In addition to the time taken, the accuracy of the generated diagrams is also calculated using the metrics below:

* 1. **True Positive (TP):** Number of classes, methods or attributes are correct and are generated correctly.
  2. **False** **Positive (FP):** Extra classes, methods or attributes are generated and are incorrect.
  3. **False** **Negative (FN):** No classes, methods or attributes generated.

From the accuracy metrics, two factors are then calculated:

* 1. **Precision:** It is a measure of how close the resultant diagram is to the correct diagram.
  2. **Sensitivity:** It is a measure of how sensitive the test subjects are in detecting classes, attributes and methods.

**6.1.2.2** **Experiment using the proposed class diagram generator**

The problem statement (S3 from Table 6.1) was also used as an input statement to the class diagram generator, and the following times were recorded:

* 1. **Analysis Time (AT):** This is the time taken for the software to analyse the input statement and generate the candidate classes, methods and attributes. This also includes the time spent by the user to make any necessary modifications before the final diagram is generated.
  2. **Diagram Generation Time (DGT):** This is the time taken by the class diagram generator to generate the diagram, after having generate the candidate classes, methods and attributes and after having the user make any necessary changes, i.e., the time taken to generate the diagram after analysis. This also includes the time spent by the user to rearrange the diagram in a suitable manner.
  3. **Overall Time:** This is the total time taken by the tool to fully generate the class diagram.

In addition to the time taken, the accuracy of the generated diagrams is also calculated using the metrics below:

* 1. **True Positive (TP):** Number of classes, methods or attributes are correct and are generated correctly.
  2. **False** **Positive (FP):** Extra classes, methods or attributes are generated and are incorrect.
  3. **False** **Negative (FN):** No classes, methods or attributes generated.

From the accuracy metrics, two factors are then calculated:

* 1. **Precision:** It is a measure of how close the resultant diagram is to the correct diagram.
  2. **Sensitivity:** It is a measure of how sensitive the class diagram generator is in detecting classes, attributes and methods.

# 6.2 DATA OBTAINED

For determining the accuracy of the class diagram generator using the methods stated above, the following data has been obtained:

| Table 6.2: Accuracy values of the proposed class diagram generator | | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test Statement | Expected Output | | | Output Generated Correctly (True Positives, TP) | | | Output Generated Incorrectly (False Positives, FP) | | | Output not Generated (False Negative, FN) | | |
| No. of Classes | No. of Attributes | No. of Methods | No. of Classes | No. of Attributes | No. of Methods | No. of Classes | No. of Attributes | No. of Methods | No. of Classes | No. of Attributes | No. of Methods |
| 1 | 1 | 2 | 2 | 1 | 2 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 3 | 8 | 2 | 3 | 8 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 5 | 15 | 5 | 5 | 15 | 5 | 0 | 3 | 0 | 0 | 0 | 0 |
| 4 | 4 | 12 | 6 | 4 | 11 | 6 | 0 | 1 | 0 | 0 | 1 | 0 |
| 5 | 2 | 5 | 3 | 1 | 4 | 3 | 1 | 2 | 0 | 1 | 1 | 0 |

Where:

* **Expected Output:** Denotes the expected no. of classes, attributes, methods.
* **Output Generated Correctly (True Positives, TP):** Denotes the no. of classes, attributes and methods that match with the expected output.
* **Output Generated Incorrectly (False Positives, FP):** Denotes the expected no. of classes, attributes and methods that does match with the expected output or has not been generated.

For calculating the precision, the following formula is used:

For calculating the average precision, the following formula is used:

| Table 6.3: Precision percentage of the proposed class diagram generator | | | |
| --- | --- | --- | --- |
| Test Statement | Precision (%) | | |
| Classes | Attributes | Methods |
| 1 | 100 | 100 | 100 |
| 2 | 100 | 100 | 100 |
| 3 | 100 | 83.3 | 100 |
| 4 | 100 | 91.6 | 100 |
| 5 | 50 | 66.6 | 100 |
| Average Precision (%) | 90 | 88.3 | 100 |

For calculating the sensitivity, the following formula is used:

For calculating the average sensitivity, the following formula is used:

| Table 6.4: Sensitivity percentage of the proposed class diagram generator | | | |
| --- | --- | --- | --- |
| Test Statement | Sensitivity (%) | | |
| Classes | Attributes | Methods |
| 1 | 100 | 100 | 100 |
| 2 | 100 | 100 | 100 |
| 3 | 100 | 100 | 100 |
| 4 | 100 | 91.6 | 100 |
| 5 | 50 | 80 | 100 |
| Average Sensitivity (%) | 90 | 94.32 | 100 |

For comparison with the traditional method of drawing class diagrams from a problem statement by using the method stated above, the following data was obtained:

| Table 6.5: Time values for test subjects using the traditional method | | | |
| --- | --- | --- | --- |
| Test Subject | Analysis Time (s) | Diagram Generation Time (s) | Overall Time (s) |
| T1 | 114 | 43 | 157 |
| T2 | 246 | 603 | 849 |
| T3 | 85 | 182 | 267 |
| T4 | 76 | 68 | 144 |
| T5 | 40 | 207 | 247 |
| T6 | 110 | 576 | 686 |
| T7 | 120 | 433 | 553 |
| T8 | 118 | 432 | 550 |
| T9 | 75 | 89 | 164 |
| T10 | 140 | 152 | 292 |

From Table 6.5, the following calculations can be made:

**Average Analysis Time:** ATavg =

ATavg = 112.4s

**Average Diagram Generation Time:** DGTavg =

DGTavg = 278.5s

**Average Overall Time:** OTavg =

OTavg = 390.9s

The problem statement was also passed as input to the class diagram generator and the following timings were recorded:

**Analysis Time (AT):** 12.58s

**Diagram Generation Time (DGT):** 10.50s

**Overall time (OT):** 23.08s

It is to be noted that the recorded time is taken to be the average of worst case scenario, which is when when the tool is ran for the first time.

Consequently, the diagrams generated by the traditional method were analysed and compared with the following data:

Expected No. of Classes: 5

Expected No. of Attributes: 15

Expected No. of Methods: 5

And thus the data given below was obtained:

| Table 6.6: Accuracy values of test subjects using the traditional method | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test Subject | Output Generated Correctly (True Positive, TP) | | | Output Generated Incorrectly (False Positive, FP) | | | No Output Generated (False Negative, FN) | | |
| No. of Classes | No. of Attributes | No. of Methods | No. of Classes | No. of Attributes | No. of Methods | No. of Classes | No. of Attributes | No. of Methods |
| T1 | 1 | 3 | 0 | 2 | 0 | 0 | 4 | 12 | 5 |
| T2 | 4 | 9 | 2 | 4 | 10 | 12 | 1 | 6 | 3 |
| T3 | 4 | 8 | 0 | 1 | 6 | 0 | 1 | 1 | 5 |
| T4 | 3 | 2 | 0 | 1 | 4 | 0 | 2 | 13 | 5 |
| T5 | 4 | 8 | 4 | 0 | 4 | 2 | 1 | 7 | 1 |
| T6 | 5 | 11 | 4 | 1 | 4 | 2 | 0 | 4 | 1 |
| T7 | 4 | 4 | 5 | 4 | 14 | 9 | 1 | 11 | 0 |
| T8 | 3 | 5 | 4 | 5 | 7 | 3 | 2 | 10 | 1 |
| T9 | 2 | 6 | 2 | 4 | 6 | 0 | 3 | 9 | 3 |
| T10 | 4 | 10 | 5 | 2 | 6 | 3 | 1 | 5 | 0 |

Therefore, with reference to Table 6.6, the precision and sensitivity of the test subjects could be calculated using the given formulas and were found to be as follows:

| Table 6.7: Precision percentage and Sensitivity percentage of test subjects using the traditional method | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Test Subject | Precision (%) | | | Sensitivity (%) | | |
| Classes | Attributes | Methods | Classes | Attributes | Methods |
| T1 | 33.33 | 100 | 0 | 20 | 20 | 0 |
| T2 | 50 | 47.36 | 14.28 | 80 | 60 | 40 |
| T3 | 80 | 57.14 | 0 | 80 | 53.33 | 0 |
| T4 | 75 | 33.33 | 0 | 60 | 13.33 | 0 |
| T5 | 100 | 66.66 | 50 | 80 | 53.33 | 80 |
| T6 | 83.33 | 73.33 | 66.66 | 100 | 73.33 | 80 |
| T7 | 50 | 22.22 | 35.71 | 80 | 26.66 | 100 |
| T8 | 37.5 | 41.66 | 57.14 | 60 | 33.33 | 80 |
| T9 | 33.33 | 50 | 100 | 40 | 40 | 40 |
| T10 | 66.66 | 62.52 | 62.5 | 80 | 66.66 | 33.33 |

With reference to Table 6.7, the average precision and sensitivity of the test subjects were found to be as:

| Table 6.8: Average Precision and Average Sensitivity of test subjects using traditional method | | | |
| --- | --- | --- | --- |
|  | Classes | Attributes | Methods |
| Average Precision (%) | 60.92 | 55.42 | 38.63 |
| Average Sensitivity (%) | 68 | 44 | 45.33 |

# 6.3 INFERENCES

From the data collected, it can be concluded:

1. The class diagram generator showed the following average precision values:
   * For Classes: 90%
   * For Attributes: 88.3%
   * For Methods: 100%
2. The class diagram generator showed the following average sensitivity values:
   * For Classes: 90%
   * For Attributes: 94.32%
   * For Methods: 100%
3. For generating a class diagram from a problem statement (S3 from Table 6.1), the test subjects showed an average overall time of 390.9s while the class diagram generator took 23.08s. This amounts 367.82s of time saved which equates to the class diagram generator being 94.10% faster than the traditional method.
4. The test subjects the following average precision values:
   * For Classes: 60.92%
   * For Attributes: 55.42%
   * For Methods: 38.63%

With reference to serial no. 3 from Table 6.3, for the same statement, the class diagram showed the following precision values:

* + For Classes:100%
  + For Attributes: 83.3%
  + For Methods: 100%

1. The test subjects the following average sensitivity values:
   * For Classes: 68%
   * For Attributes: 55.42%
   * For Methods: 45.33%

With reference to serial no. 3 from Table 6.4, for the same statement, the class diagram showed the following precision values:

* + For Classes:100%
  + For Attributes: 100%
  + For Methods: 100%

1. The disparity seen between the outputs generated by the testers and the tool is assumed to be the subjective assessment of the problem statements by the testers unlike the objective and rule based assessment done by the tool.

**Table 6.9: Inferences table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Characteristic** | **Traditional Method** | **Class Diagram Generator Method** | **Inferences** |
| Accuracy - Classes | 60.92 (%) | 90 | The tool shows higher accuracy in identifying classes. |
| Accuracy - Attributes | 55.42 | 88.3 | The tool shows higher accuracy in identifying attributes classes. |
| Accuracy - Methods | 38.36 | 100 | The tool shows higher accuracy in identifying methods of classes. |
|  |  |  |  |
| Sensitivity - Classes | 68 (%) | 90 | The tool has higher sensitivity when identifying classes. |
| Sensitivity - Attributes | 44 | 94.32 | The tool has a higher sensitivity when identifying attributes. |
| Sensitivity - Methods | 45.33 | 100 | The tool has higher sensitivity when identifying methods. |
|  |  |  |  |
| Diagram Analysis Time | 112.4 (seconds) | 12.58 (seconds) | The tool analyzes input statements faster than an average tester. |
| Diagram Generation Time | 278.5 | 10.50 | The tool generates output faster than an average tester. |
| Overall Time | 390.9 | 23.08 | The overall time required by this tool is far smaller than what is required by an average tester. |

**Table 6.10: Comparison Table between similar tools**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Support** | **CM-Builder** | **LIDA** | **GOOAL** | **NLOOML** | **DC-Builder** | **RACE** | **RAUE** | **RAPID** | **ACDG** |
| Classes | Yes | User | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Attributes | Yes | User | Yes | Yes | Yes | Yes | Yes | No | Yes |
| Methods | No | User | Yes | Yes | No | Yes | No | No | Yes |
| Associations | Yes | User | Semi-NL | No | Yes | Yes | Yes | Yes | Yes |
| Multiplicity | Yes | User | No | No | No | No | Yes | No | No |
| Aggregation | No | No | No | No | Yes | Yes | Yes | Yes | No |
| Generalization | No | No | No | No | Yes | Yes | Yes | Yes | No |
| Instances | No | No | No | No | No | No | No | No | No |