# CHAPTER 3

# SYSTEM ANALYSIS AND DESIGN

## Introduction

This chapter covers the methodology involved in the analysis and design of the system. The process model chosen for the development of this system is the Rational Unified Process (RUP). The RUP model has been chosen because it is a hybrid process model that brings together all the elements from the generic process models (waterfall, incremental and reuse-oriented model). It also illustrates good software engineering practices in system specification and design, and it supports prototyping and incremental delivery. RUP is also well suited for dealing with changing requirements throughout the development of the system. And again, RUP puts emphasis on the need for proper documentation.

**Table 3.1** **Stages in the PCMCS system**

|  |  |  |
| --- | --- | --- |
| S/N | TASK | METHOD |
| 1 | Information gathering | Interview of stakeholders and observation of the existing system |
| 2 | Information capture or data collection | Gathering data (patients’ medical records) from different online medical journals and articles |
| 3 | Data cleaning and pre-processing | 1. Manual cleaning 2. Using a data pre-processing tool |
| 4 | Building the model | A set of algorithms are written to build the neural network |
| 5 | Model training | Feeding the clean and vectored data into a neural network |
| 6 | Entity Extraction | The model classifies the text according to the entities specified |
| 7 | Representation method | Conceptual Data Model |
| 8 | Tools Used | Draw.io |
| 9 | Requirement modelling | Unified Modelling Language (UML) |
| 10 | UML | Use Case, Context Diagram, Activity Diagram, Class Diagram |
| 11 | Algorithm used for entity extraction | Convolutional Neural Network (CNN) |
| 12 | Testing | Black box, Unit, Integrated and System Testing |

## System Analysis

System Analysis requires an overview of the existing system and also a detailed description of the proposed system, its architecture and its requirements.

### Overview of Jaja’s Health Record System (JHRS)

From close observation, the case study, Jaja Clinic, University of Ibadan, currently makes use of a manual (paper-based) health record system. The consultation process goes forth:

On arrival at the clinic, the patient proceeds to write down his card number in the book provided for that purpose. The card attendant then takes note of the card numbers of all the patients who have recorded them and goes to retrieve their case files from the records room. Upon the retrieval of their files, the nurses call out the patients’ names to arrange them for vital signs reading. Once the nurses have taken and recorded the vital signs of all patients whose files are available, the card attendant then transports all the files to doctor’s office after he must have arranged the patients in the new waiting room, in the order by which they would visit the doctor. Patients then take turns to see the doctor where he listens and writes down the complaints, diagnosis, lab test prescriptions, lab results (if the patient has already done the required tests) and drugs. Lab results and any external reports are attached to the patient’s file. The case file is then taken back to the records room once the patient is through with the doctor.

### Problems of JHRS

From observation of the users of JHRS, a number of challenges have been identified, some of which are highlighted below:

1. **Time consuming for the consultants:** Doctors have to go through a series of unstructured text in individual patient’s case file in order to make meaning from it and therefore understand the patient’s medical history. In times where the doctor is fatigued, this may take longer time than necessary, and might not even be as accurate as it should be. This might lead to a less productive consultation process.
2. **Lack of “Searchability”:** Finding a particular case note should not feel like a hunting experience, but with paper documentation, sometimes that is exactly what it is. A great deal of time and resources are spent searching through paper documents in an attempt to find what is needed, as paper documents are not easily searchable or able to be scanned in due time.
3. **Poor quality of service received by patients:** If a consultant spends a relatively longer time with a particular patient, the other patients waiting on the queue are going to get a bit upset as they are being delayed.
4. **Security**: Traditional paper records are vulnerable to security breaches when they get into the wrong hands and, for sensitive information like home address, individual identifying information, and so on. This is an obvious problem. This also means that individuals can access records without permission and make copies of documents without anyone knowing.

### The Proposed Patient-Centered Medical Consultation System (PCMCS)

The PCMCS takes into consideration the shortcomings of JHRS. In solving the problem of time consumption of the part of consultants, this system allows consultants view a condensed version of a patient’s medical history. The consultant still has the option of viewing the detailed history for further understanding, if need be. The system also allows consultants search for particular keywords in the course of the consultation process. This leads to a higher service quality received by the patients as they do not have to spend additional time on the waiting queue. As only authorized personnel are able to log into the system, the PCMCS ensures security of patients’ record. The PCMCS also contains graphs which allows a consultant view a further analysis of a patient’s health record.

### Architecture of the PCMCS

Figure 3.1 presents the architecture of the PCMCS. The flow begins with the consultation between a doctor and a patient. The notes taken by the doctor are saved in a consultation repository. The notes are fed into the NLP system, which is where the bulk of the work takes place. As discussed in section 2.2.1.8, the NLP system converts the unstructured text sent by the doctor, into tokens and then filters the tokens for the keywords, which are analysed and classified to give an accurate summary. The summary is then viewed by the doctor. Based on the summary, further insight is provided for the doctor by the use of charts and graphs. This is also stored in the repository. The repository therefore stores the consultation notes (input) and result of the NLP process, and feeds the requested summary to the doctor (output).

# 

**Figure 3.1 General Architecture of the PCMCS**

### System Requirement Analysis

This phase involves directly observing the users of the JHRS and using the information to determine the requirements for the PCMCS. The requirements would be structured further by using UML tools.

### 3.2.4.1 Determining System Requirements of the PCMCS

The outcome of observing the users of the JHRS has been presented in section 3.2.2 as problems of the existing system. Based on this knowledge, the functional and non-functional requirements of the PCMCS are presented below:

#### 3.2.4.1.1 Functional System Requirements

The functional system requirements entail what the system is expected to do. The requirements are mainly generated from the user or stakeholders requirements. The functional requirements of the PCMCS are:

1. The consultant is able to save the details of each consultation to the system.

|  |  |
| --- | --- |
| Function | Save details of a consultation process to the system |
| Description | This function allows the doctor or consultant to input the details of the consultation process into the text box and then save. |
| Input | Consultation notes in plain text |
| Output | A message indicating that the notes have been saved |
| Action | The notes are taken and stored in a repository. |
| Pre-condition | The doctor must be verified and logged in |
| Post-condition | The patient’s history is updated with the new date of consultation as last seen |

1. The consultant is able to view the detailed history of a patient.

|  |  |
| --- | --- |
| Function | View the detailed history of a patient. |
| Description | This function allows the doctor or consultant to view the detailed history as was originally saved by the doctor. |
| Input | A request to view the full history |
| Output | The consultation notes as written by the doctor |
| Action | The notes are retrieved from the repository and displayed to the doctor |
| Pre-condition | The doctor must be verified and logged in |
| Post-condition | The consultant can then input new notes |

1. The consultant is able to view the “condensed” history of a patient.

|  |  |
| --- | --- |
| Function | View the condensed history of a patient. |
| Description | This function allows the doctor or consultant to view the condensed/summarized history as was originally saved by the doctor. |
| Input | A request to view the summarized history |
| Output | The consultation notes as summarized by the system |
| Action | The summarized notes are retrieved from the repository and displayed to the doctor |
| Pre-condition | The doctor must be verified and logged in |
| Post-condition | The consultant can then input new notes |

1. The system presents the history of a patient in charts or graphs as suited.

|  |  |
| --- | --- |
| Function | Present’s the summary of the patient’s record in graphical format |
| Description | This function allows the doctor or consultant to view the history in graphical format. |
| Input | The summarized history of the patient |
| Output | Charts representing the history of a patient’s record |
| Action | The summarized notes are converted to graphs |
| Pre-condition | The doctor must be verified and logged in |
| Post-condition | The consultant can then input new notes |

#### 3.2.4.1.2 Non-Functional System Requirements

Non-functional requirements refer to how the system is expected to behave.

**USABILITY**

* The system should have a good interface which is appealing to all users.
* The system should be designed with great consideration on the ease of use by the various categories of users.

**RELIABILITY**

* The data stored in the system must be retained over a long period of time without data being changed by the system.
* The system shall give accurate information to users.

**PERFORMANCE**

* The system shall be able to handle large data sets
* The system should have long Mean Time Between Failures (MTBF)
* The system shall constantly update the database to include new cases stored by the doctor

**SUPPORTABILITY**

* The system should be able to work across devices as long as a browser is running on the mobile device

**SECURITY**

* The system shall provide a solid authentication process to avoid system being manipulated by third party, that is, for security purposes.

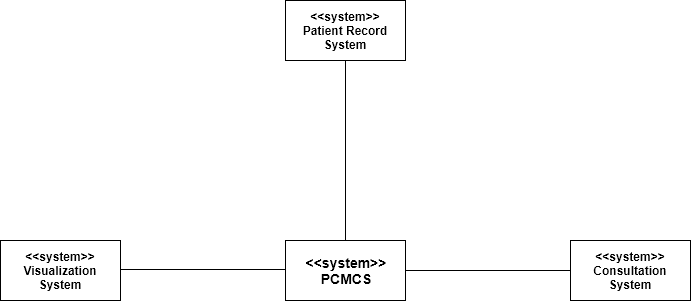
### 3.2.4.2 Structuring System Requirements of the PCMCS

This section presents the context diagram, use case diagram and activity diagram that attempt to further model and structure the system requirements.

#### 3.2.4.2.1 Context Diagram of the PCMCS

Context models are used to show system boundaries. Sometimes, the boundaries between the systems and its environment is relatively clear. Context models normally show that the environment includes several other automated systems. However, they do not show the types of relationships between the systems in the environment and the system that is being specified.

Figure 3.2 depicts the context diagram of the PCMCS. The PCMCS interacts with the patient’s records, which contains the bio-data of the patient. The summary generated from the PCMCS further becomes an element of the patient records, which is stored in the patient record system. The consultation system handles the exchange between the doctor and the patient, wherein the doctor sends the details of the consultation and also receives the summary of the patient’s record. After the summary has been generated, it can then be converted to visuals if there is a need to. This is handled by the visualization system.

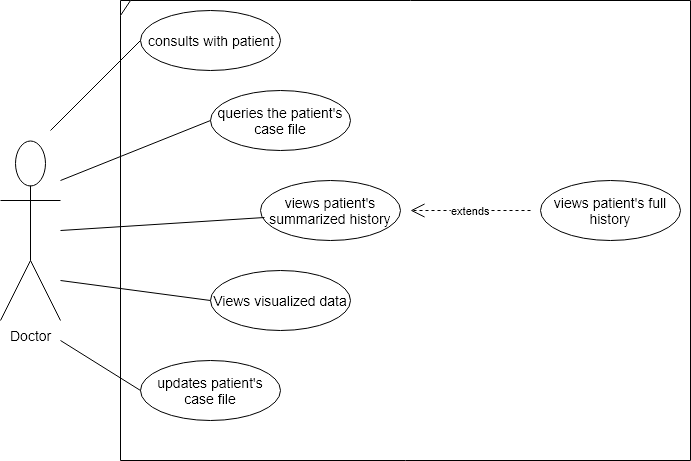


**Figure 3.2 - Context Diagram of the PCMCS**

#### 3.2.4.2.2 Use case diagram of the PCMCS

Use-case modelling is applied to analyse the functional requirements of a system without worrying about how those requirements will be implemented. Use case diagram models the interactions between the system and external actors (users or other systems).

Figure 3.3 shows the principal actor in the system (the doctor) and the different functionalities that can be carried out by the actor.



**Figure 3.3 - Use case diagram of the PCMCS**

Table 3.2 elucidates each of the functionalities.

**Table 3.2 - The descriptions of the use cases**

|  |  |
| --- | --- |
| USE CASE | DESCRIPTION |
| Consults with patient | The doctor inputs the details from the consultation process into the system.  Data: Patient’s complaints, treatment summary  Response: Confirmation that the data has been saved |
| Queries patient’s case file | The doctor can query the patient’s case file if he wants to view the history of the patient. |
| Views case history | This shows the full history of the patient for a more in-depth analysis |
| View’s patient’s summary | This shows the summarized history of the patient for a quick overview. |
| Views visualized data | This process presents the graphical data that has been generated from the summary |
| Updates patient’s case file | The doctor is able to update the patient’s case file by saving the details of the most recent consultation |

#### 3.2.4.2.3 Activity Diagram of the PCMCS

Activity diagrams are intended to show the activities that make up a system process and the flow of control from one activity to another. It is important to understand that an activity diagram gives a high level view of the system’s functionalities. It is therefore required to model the requirements of the system.

Figure 3.4 shows the flow of the different activities within the PCMCS. Upon the beginning of the consultation, the doctor requests to view the patient’s history. He can decide to view the summary if he wants the high-level history or if he wants to see the full history, he has the option of seeing that too. Both ways, he is able to view the visualized report of the patient’s history. After that, he is able to input a new consultation note base on his conversation with the patient. Once he is done with the patient, he saves his new note and thus, that particular patient’s history is updated. For subsequent consultations, the saved case note becomes a part of the summarized report to be presented to the doctor.



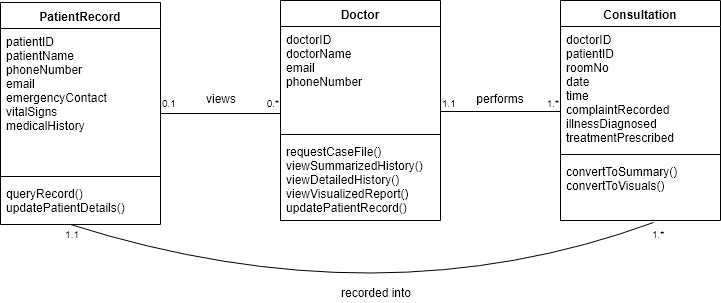
**Figure 3.4 - Activity Diagram of the PCMCS**

## 3.3 System Design

In this section, the system is first described independently of any computer platform, (logical design) and is then transformed into technology-specific details (physical design) from which all programming and system construction can be accomplished.

### 3.3.1 Class Diagram of the PCMCS

Class diagrams represent the static structure of the data and operations that act on the data. It shows an entity with a well-defined role, has state, behaviour and identity.



**Figure 3.5 - Class Diagram for the PCMCS**

Figure 3.5 presents the class diagram for this system. For the PatientRecord-Doctor relationship, a patient’s record can be viewed by as many doctors while a doctor views one record at a particular time. For the Doctor-Consultation relationship, a doctor performs as many consultations while a consultation is performed by just one doctor. The PatientRecord-Consultation shows that one consultation is recorded into the patient’s record at any point in time, and that a patient’s record contains many consultation reports.

### 3.3.2 Logic Modelling of the PCMCS

This involves representing the internal structure and functionality of the processes involved in converting the input collected by the system to the required output.

**Figure 3.6 - High-level diagram of the logic model of the PCMCS**

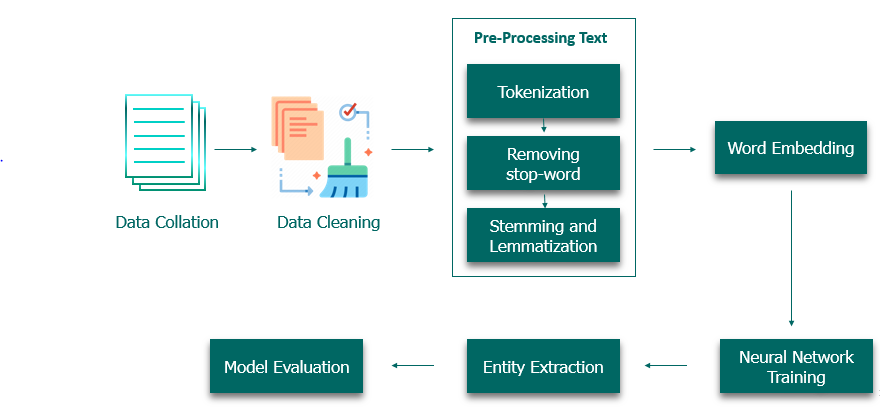
Figure 3.6 shows the high-level diagram of the logic model of the PCMCS. For this system, the input is a set of unstructured text. The category of text falls into the medical records of patients from a given hospital or groups of hospital.

For processing this unstructured text data, we have made use of a number of NLP techniques in order to convert the unstructured data to a structured summarized version than is easy to read and understand.

The main output of this system would be a summarized version of the patient’s medical history, alongside a visualized form of the data.

#### 3.3.2.1 Algorithm for PCMCS

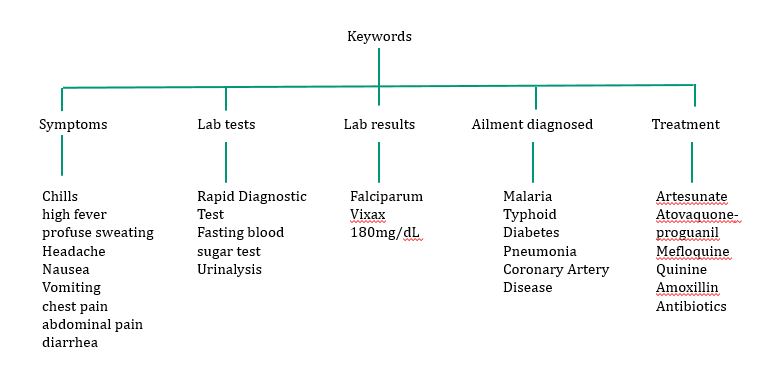
This section describes the necessary steps taken to convert the input to the output required, which is the summary of the patient’s record as a list of the different keywords which have been classified.



**Figure 3.7 showing the flow of data from input to output**

INPUT

1. Data gathering and data collation
   1. Sample case notes related to malaria, HIV, respiratory infections, diabetes and coronary artery diseases are gathered from a number of medical journals and websites.
   2. These notes are then collated in a text file.
2. Text processing and cleaning
   1. Clean data manually by correcting spelling errors and removing unnecessary content like links, diagrams, and references to tables and figures.
   2. Load data into the program.
   3. Use an NLP library such as NLTK or SpaCy to further clean the data. This involves removal of stopwords, whitespaces, filtering out punctuation, converting words to lowercase, stemming words and splitting sentences into words using the tokenizer.
3. Export the result of the text processing step to a text file
4. Labelling text for Supervised learning
   1. Create a duplicate file and then replace each keyword with the appropriate class. For example, replace ‘fever’ with ‘symptom’ and ‘quinine’ with ‘treatment’. This step is necessary for feature engineering.



**Figure 3.8 Taxonomy of the different classifications of keywords**

1. Split the clean data (from step 3) into train and test datasets: 80% of the data would be used for training and the other 20% would be used for testing.

PROCESSING

1. Training a Word Embedding model: The words are assigned vectors (or numbers) because the neural networks cannot process text
   1. Import a library for Word Embedding such as Gensim
   2. Pass the tokenized words as arrays, where each array contains the words in each sentence
   3. This results in each word being assigned a vector representation. Words similar to each other usually have vectors whose values are close to each other
2. Building a Convolutional Neural Network Model: A feature extraction model that learns to extract salient features from documents represented using a word embedding.
3. Check how features work with the model and start again until the features work properly. This means the model should be continually retrained.

OUTPUT

1. Text classification: The output shows text with keywords labelled as symptoms, treatment and disease name.
2. Result showing the accuracy of the model: The accuracy of the model is then displayed.
3. Summarized text: The keywords extracted are then placed under the defined categories.

Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O’Reilly Media Inc.