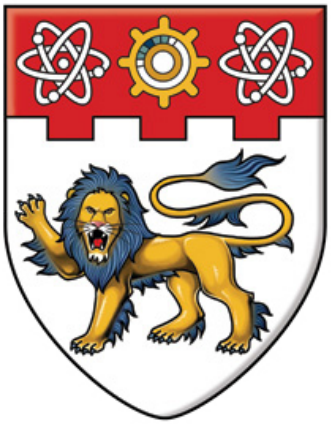

NANYANG TECHNOLOGICAL UNIVERSITY

COURSE CODE	**:**	**CZ3003**
COURSE NAME	**:**	**SOFTWARE SYSTEM ANALYSIS AND DESIGN**
TEAM NAME	**:**	**911 TEAM**
TEAM MEMBERS	**:**	**1) DENISE (U1521697H)** **2) AIQING (U1520734F)** **3) HTET NAING (U1620683D)** **4) ERIC (U1420135F)**
LAB GROUP	**:**	**SSP1**
COURSEWORK	**:**	**ASSIGNMENT 1**

Content

1) System Thinking

"System thinking" is studying and understanding kinds of assemblages of a system. It is usually applied top-down. The whole assemblage is taken apart or decomposed into constituent sub-assemblies, which are subsequently taken apart, and so on, until the remaining entities are indissoluble. System thinking not only helps cover information about requirements of a system but also design of a system.

In this assignment, our team belongs to 911 subsystem. To help us elucidate requirements of a 911 subsystem and how it should be designed, system thinking should be applied with the assistance of an artist illustration. This is important from the perspective of system thinking because it is useful for all stakeholders as an ideation of the new software as it is being designed.

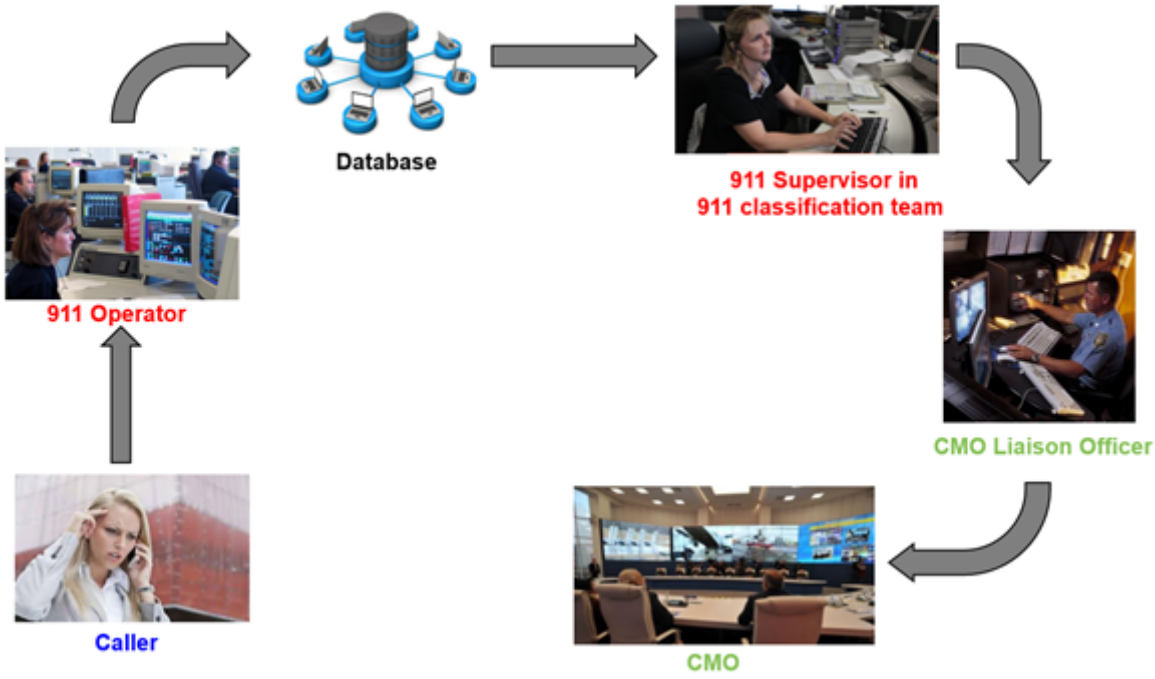

Figure 1. Artist Illustration for 911 subsystem

From the above figure, it can be surmised that 911 subsystem consists of different parts such as people, software, hardware, etc. The people include 911 operators, a 911 supervisor in 911 classification team and a CMO Liaison Officer. The software may include web graphical user interfaces and highly secured cloud database whereas the hardware may comprise computers with fast processors, telephones, monitors, etc.

When a 911 operator receive an emergency call from a caller, they will record all the critical information concerning the caller and the nature of an emergency call. After that, they will store all the incident reports in the database. The 911 supervisor will be notified whenever a new incident report is available in the database for classifying the type of the incident. In case of a suspected crisis, the CMO Liaison Officer will be notified and prompted to determine the authenticity of the crisis. Eventually if it turns out to be a legitimate crisis, the case will be forwarded to Crisis Management Office(CMO) to tackle the crisis immediately.

Thus, merely by looking at the above Artist Illustration and applying system thinking to it, we can proceed to define the requirements and to formulate the design of a 911 subsystem in terms of user-friendliness, holisticness and meaningfulness.

2) Architectural Thinking

The opposite of system thinking, architectural thinking is a way of composing a meaningful solution to a problem given the abstract assemblages. After disassembling the possible subsystems for this 911 project using system thinking, the architectural thinking combines those subsystems into what would be a fully functioning 911 system.

From the artist's illustration shown in Fig. 1, we need to determine whether data and/or control will be passed, to whom they will be passed to in order to make the system work according the requirement. We could see from that figure that the 911 system will include the operator manager, supervisor manager, database, and an API to connect to the CMO system, which is separately build.

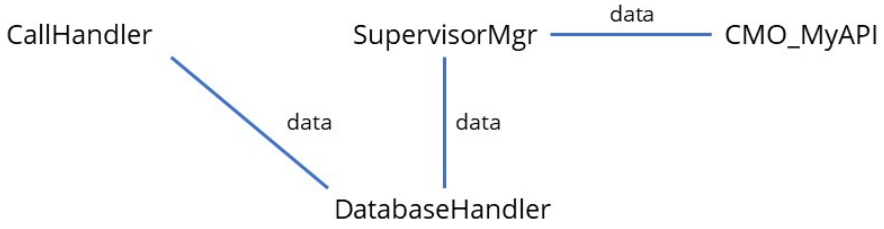

Figure 2. Initial Architecture for 911 subsystem

The call handler will be manned by the operators, who will directly liaise with the caller and gather the data from the conversation to be pushed into the database. Upon new cases, the DatabaseHandler will notify the SupervisorMgr via an action listener The supervisor will receive the data from the database, analyze the data using current and past reports, and should the crisis be legitimate, send the data to the CMO system using MyAPI. The database handles all requests from the call handler and supervisor manager including requests to select, enter, or update data.

3) Software System Composition

Software system is a system whose development effort is majorly software in accordance with the Software Development Life Cycle(SDLC) and functionality is commanded and controlled by one or more software applications (product, or developed subsystem), all working to a common mission. Furthermore, the system likely bears the name of the leading software application. Therefore, in this assignment, the software system of our 911 subsystem can be defined — in EBNF language — as follows:

911subsystem ::= <swproduct>⁺ + <operator>⁺ + <supervisor>⁺ + <phonetracking-MyAPI>⁺
+ [<CMO liaison officer>⁺ + <CMO-MyAPI>⁺];

swproduct ::= <compiledcode>⁺ + <API>⁺ + [<doc>*];

<API> ::= <HTTP>⁺ + <JSON>⁺;

Figure 3. EBNF Statement for 911 subsystem

Legend:
::= means "can be broken down into"
⁺ means multiplicity from 1 to infinity
* means multiplicity from 0 to infinity
<X> means mandatory part
[<X>] means optional part

There are three EBNF statements in the above figure. The interpretation of the first statement is that 911 subsystem is composed of at least one part from each of the following:

(i) software product
(ii) 911 operator
(iii) 911 supervisor
(iv) phone tracking MyAPI
(v) CMO liaison officer
(vi) CMO MyAPI

The phone tracking MyAPI is used to send digital signal to local commercial telephone communication carrier upon receiving emergency calls so that it can be helpful in detecting prank calls and locating where they are being made. In case of an authenticated crisis, CMO MyAPI will be necessary to provide CMO with all the critical information and data in the form of incident report. The above two API will be named appropriately as MyAPI since the data is being passed from 911 subsystem to others (i.e. we are the givers or providers of data).

In the second statement, the software product is said to be broken down into the compiled code that make use of relevant APIs for implementing the 911 subsystem software and optional storage of documents or incident reports. Finally, as described by the last statement, APIs that will be used in our software consist of a HTTP API for implementing

web graphical user interface and a JSON API for passing required data to the other subsystem in JSON format. Therefore, EBNF statements are effective in examining composition of a software system.

4) Initial Architecture Structure with Functions

Initial architecture structure with functions is essential for a software designer to understand how each part of the system connects with each other. As a result, initial architecture structure with functions require more details than the usual initial architecture even though the diagrams are derived from it.

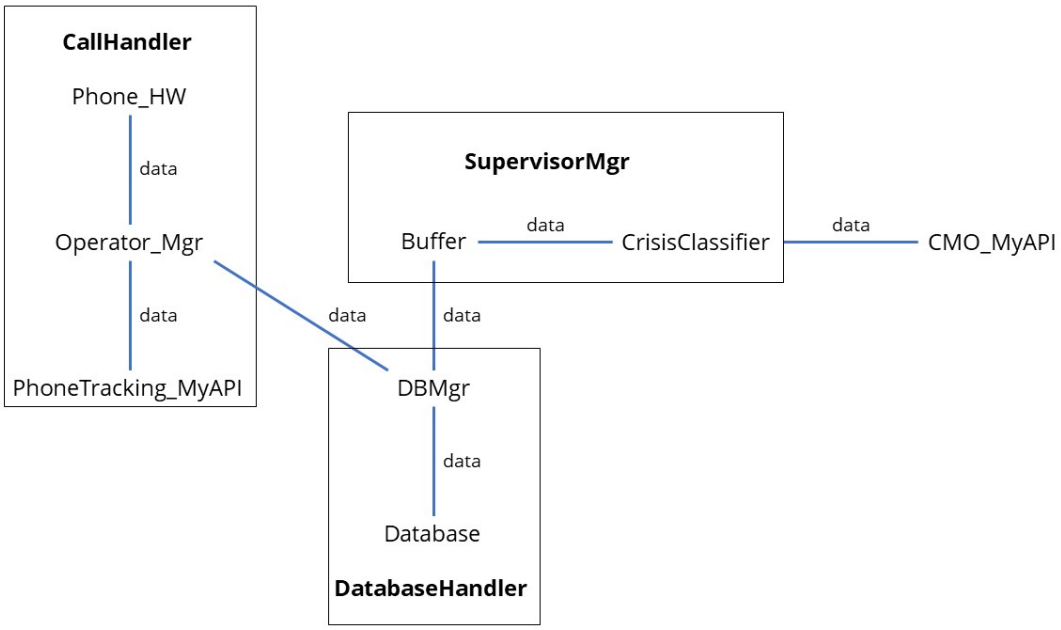

Figure 4.1. Initial architecture structure with functions in 911 subsystem

The architecture above contains all 3 components present in the system, namely CallHandler, SupervisorMgr, and DatabaseHandler. We will show a breakdown of the different subsystems below:

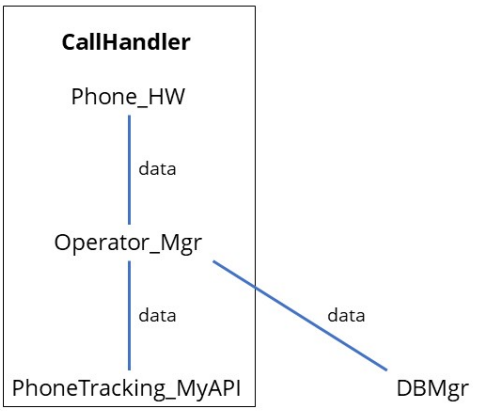

Figure 4.2. Initial architecture structure with functions for call handler in 911 subsystem

The example above shows how the components in CallHandler relate with others and within itself. Calls are received by specially trained operators through a hardware phone, who then creates an incident report. During the call, a background process will be initiated to determine the caller's location through the API provided by the telecommunication operators. Data gathered by the operator will be entered to the database manager.

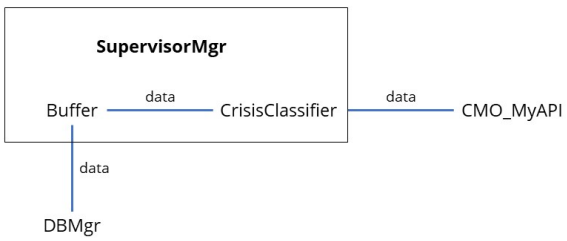

Figure 4.3. Initial architecture structure with functions for supervisor manager in 911 subsystem

The data from the database manager will be temporarily stored in the buffer, pending for the supervisor's review. This is to ensure that the CrisisClassifier (supervisor) will not be flooded with multiple reports at the same time and the cases with higher priority will be shown to the supervisor first. As the supervisor reads the incident report, he then analyses the report using data from the caller and potentially linked cases or past data retrieved through the database manager. After the supervisor decides the outcome for the assessment, he will

send the updated outcome to the database manager for recording and if the report is about a crisis, the crisis classifier will to send the data through CMO's MyAPI.

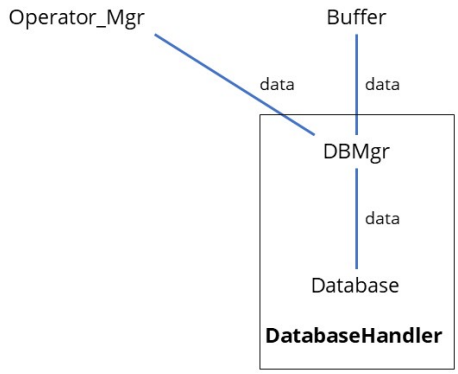

Figure 4.4. Initial architecture structure with functions for database manager in 911 subsystem

The DBMgr acts as a Data Access Object (DAO) which provides specific data operations without exposing details of the database. After there is a new case in the DBMgr, it searches for cases that may be linked and sends it to the buffer. As a result, the database manager controls the data flow between the operator and database or supervisor and the database.

5) Initial architecture behaviour

For initial architecture structure with behaviour, we will use dialog map to illustrate. Dialog map is a specific form of state machine diagram which is used to show discrete behaviour of a part of designed system through finite state transition, and focus more on user interface. The dialogue map for 911 subsystem is shown as below:

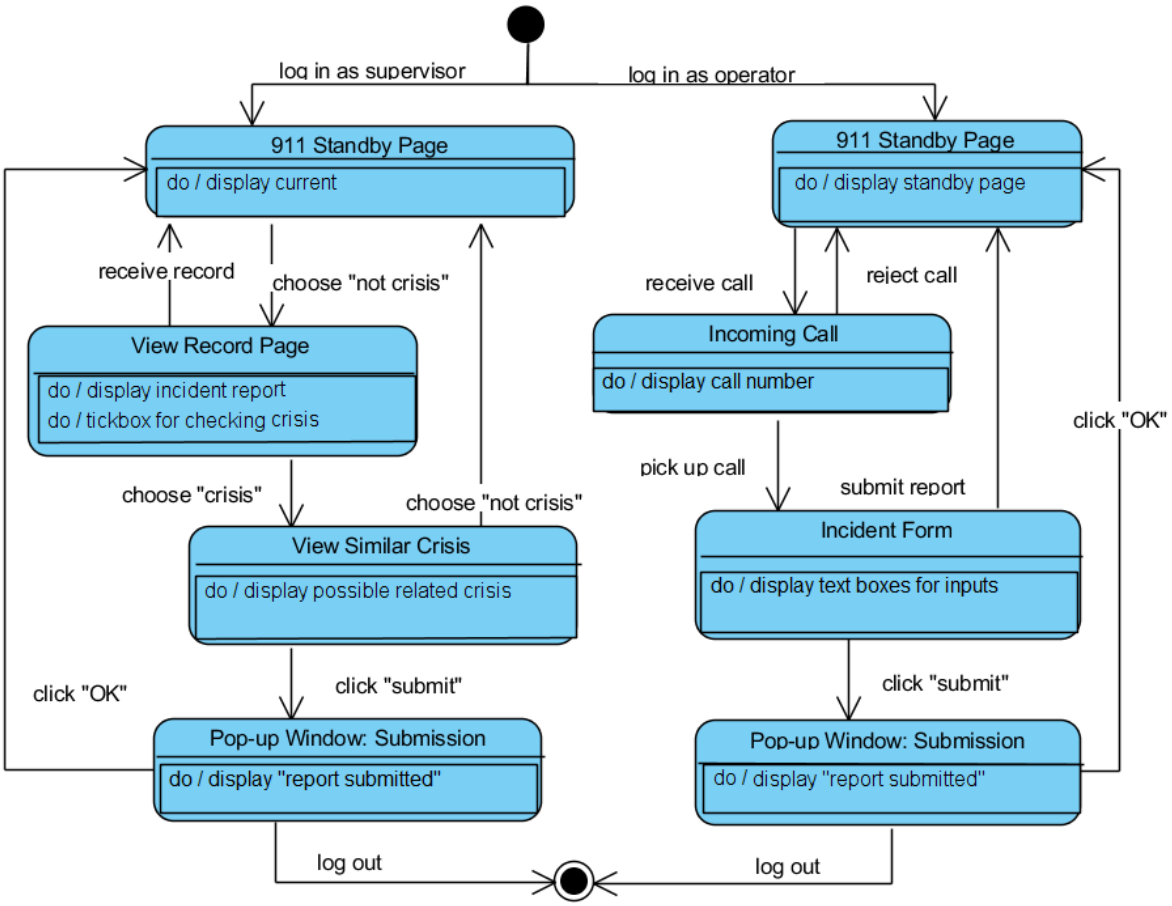

Figure 5. Dialog map for 911 subsystem

6) Imprinted Software Quality

Quality is the measurable degree to which the software satisfies requirements, conforms to specifications, and meets vendor's expectations. The 911 subsystem has implanted following McCall quality attributes to strive for a better performance of the subsystem:

Integrity

Add SignOn component to ensure that control is only passed after authentication. This is to prevent unauthorized access to data and control in the system. The following diagram shows the structural implementation of this quality attribute:

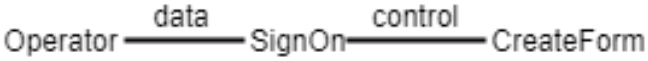

Figure 6.1. Integrity quality attribute

As seen from the diagram, operators need to pass the authentication process before use the system. They need to log in with their own userID and password, and this information will be then passed to SignOn component for verification. Only upon successful authentication will the control be passed to the next process, in this case, CreateForm.

Efficiency

In order to improve on efficiency, function accelerator is used by creating customized keyboard shortcuts for operators during form filling process. This is to reduce the hassle they have when try to answer the phone call and fill the form simultaneously.

The system will come with a set of default setting for keyboard shortcuts such as using "ctrl+C" for copy and "ctrl+V" for paste. Besides that, the system will also have a set of customized setting for keyboard shortcuts for better efficiency during form filling process. Examples will be allowing the operators to use "ctrl+alt+N" to directly go to name field of the reporter, and "ctrl+alt+D" to directly go the case description field. Besides that, during form filling process, details such as operatorID will be auto-filled for all log-ins operators. This will reduce the time they spent on repetitively key in their own detailed such as operatorID during the working hours.

Flexibility

The system should be able to handle exceptions risen from the system. For example, in the 911 subsystem, if a user fails to log in for a few times, some exception handling process will be triggered. The following diagram illustrates our idea:

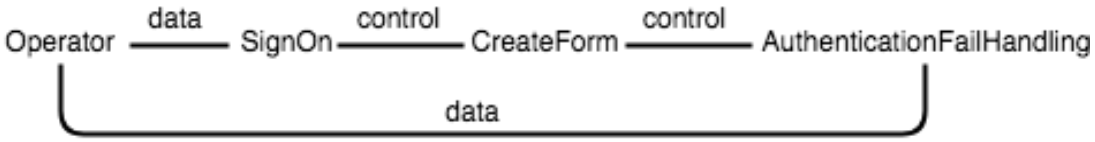

Figure 6.2. Flexibility quality attribute

In the case of several consecutive times of log-in failure performed by the user, the system will go to AuthenticationFailHandling component to work on this exception.

7) Analysis Modelling

Structured Systems Analysis & Design Method (SSADM) is a hefty set of structured analysis and structured design techniques and graphical tools for identifying and transforming business requirements into software design specifications. For the purpose of this portion, our group will be making use of the Data Flow Diagram (DFD) to process mapping of system.

DFD is a behaviour diagram which shows (from the process perspective) inputs, outputs, and data storage. It does not indicate timing between design objects inside a process, nor sequence of distinct processes. Fig 7 below shows our DFD.

Figure 7. Data Flow Diagram for 911 subsystem

Referring to the data diagram, the caller first passes the data of incident report to the our *Receive call and create a case* process. In this process, the operator creates a case based on this information and this data is passed into our incidents reports database. Once there is an update in the database, it will notify the supervisor in-charge of classifying if the case is a crisis. Hence, it passes the incident record as well as data of similar cases to the *Identify suspected crisis* process. Upon classifying the case, the process passes the data of suspected crisis to the *Authenticate the crisis by CMO liaison officer* process, where the CMO liaison officer truly determines if this is a case to be passed to the CMO. For cases which are authenticated, it will be passed as data to the CMO.