Usability Log Analysis for Healthcare Devices with Software Process Mining

Atthia Abrar

Helsinki 06.02.2020

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta/Osasto – Fakultet/Sektion –Faculty/Section	Laitos – Institution – Department
Faculty of Mathematics and Natural Sciences	Department of Computer Science

Tekijä – Författare – Author

Atthia Abrar

Työn nimi – Arbetets titel – Title

Usability Log Analysis for Healthcare Devices with Software Process Mining

Oppiaine – Läroämne – Subject

Computer Science

Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä – Sidoantal – Number of pages
	06.02.2020	63 pages + 22 appendix pages

Tiivistelmä – Referat – Abstract

With the new trends in advanced healthcare equipment and innovation, the healthcare industry is now focused more on efficiency and improving quality. Devices record events in event log files that represent the program or application's actual usage. The log file of the event is like an operation history which shows what occurred in the program. Since devices have the log of real-time events, real-time processes can be tracked, and data analysed from different aspects. The details about the event log file can be used to create a process model and analyse the data to know its strengths and weaknesses.

This thesis aimed to develop a tool for usability analysis used for GE healthcare. The design science approach has been used as an overall research method. To achieve the research goal, ideas was taken from already developed process mining algorithms and used in making algorithms that solves the problem of software process mining. In this thesis, already used process mining algorithm techniques were examined that can be used to answer the problem of software process mining. Software process mining was used to analyse the deployed software behaviour.

The study focused on making the process discovery algorithm along with structured algorithm. The outcome of the thesis was the tool that was used by the GE Healthcare to do the usability analysis on log files. The tool produces the events in the form of flow chart diagram.

Avainsanat – Nyckelord – Keywords

usability analysis, event log, software process mining, alpha algorithm, fuzzy mining algorithm, heuristic mining algorithm, process discovery

Säilytyspaikka – Förvaringställe – Where deposited

Muita tietoja – Övriga uppgifter – Additional information

Contents

1. Introduction

In this chapter background and motivation of the research is presented. Furthermore, this chapter also discusses the research problem, research goal, research questions, research scope and research methodology. At the end this chapter present the skeleton of the thesis and briefly explain the next chapters and its content.

1.1 Background and Motivation

Now-a-days information systems are recording information about the processes in the form of event log files. Systems like ERP (Enterprise Resource Planning), CRM (Customer relationship management), SCM (Supply Chain management), PDM (Product Data Management) and WFM (Workflow Management) are example of systems, which are recording actual data of the events that are taking place [GRWU08]. Moreover, Embedded Systems (ESs) like copier machines, imaging devices, scanners etc. has also started recording the events. Healthcare industry is now more attentive on recoding the actual usage of devices.

With the new trends of advanced healthcare equipment and innovation, healthcare industry is now more focused on performance and quality improvement. Devices can record events in the event log files, which has the reflection of actual usage of the software or application. The event log file is like a history of activities which shows what happened in the system. As devices have the real-time event log, it is possible to monitor real time processes and analyse the data from various aspects. The event log file information can be used to derive a process model and analyse the data to learn its strengths and weakness. A proper and intelligent log analysis can help organization in increasing the reliability and usability of deployed devices. Log analysis can also help to monitor the real-life processes systematically.

Healthcare industry emphasis more on efficiency and effectiveness of their medical equipment and control their expenses by giving quality service at low cost. In this situation, it is very important to analyse and evaluate the existing systems. Here software process mining technique can be applied to extract the process models from the event log files [WBJL03]. These process models give you insights of the reality and actual usage.

1.2 About GE Healthcare

GE Healthcare is one of the largest industries having expertise in medical imaging, medical diagnostics, patient monitoring systems, drug discovery and biopharmaceutical manufacturing technologies. GE Healthcare is a unit of General Electric Company (NYSE:GE), and is headquartered in Chicago, USA. GE Healthcare is serving healthcare professionals and their patients in more than 100 countries.

GE has developed in Finland for more than 45 years. GE headquartered in Vallila, Helsinki Finland is one of GE's leading patient monitoring product development and marketing centres. Vallila campus designed and developed the solutions related to anaesthesia administration and health information systems. Vallila is also GE Healthcare's domestic sales, marketing, maintenance and service operations.

GE Healthcare employs around 800 people, of whom 250 works in production. Health Innovation Village, start-ups, as well as several medium-sized companies are also part of Vallila campus, in addition to our campus they are total of about 200 people.

GE's premises in Vallila generates ultrasound equipment, patient monitoring solutions in hospitals, clinical information systems and IT solutions, as well as provides services and maintenance. With GE Healthcare extensive medical imaging and information systems, clinical diagnostics, patient monitoring systems, drug development, production technologies to produce biological drugs and a variety of solutions to improve performance helps their customers provide better care to more people around the world at lower cost. GE Healthcare don’t have the access to the real hospital and patient data, they can get the real data from hospital on demand for debugging and fixing issues and bugs.

1.3 Research Problem

GE Healthcare’s equipment in hospitals are recording real time usage of the devices in the event log files named clinical log files. Currently GE Healthcare has usability test process with test users for usability analysis. In those usability tests, they make the videos of the users while using the application and observe their behaviour. There is a need to have the analysis on the log files with the actual events coming from the hospitals to improve usability and reliability of devices. They don’t have the usability analysis on the real-time data under different circumstances. Real time data analysis will help them analyse the data more frequently and assist them to understand which feature of the device is

operating well and which need improvements. Perhaps they have the picture of device usage more clear and real.

1.4 Reserach Goal

The goal of the research proposed in this thesis was to develop a tool for GE healthcare, which allows them to do the usability analysis on event log's file data, Additionally, the tool also present useful information about the errors and failures in the GE Healthcare applications.

1.5 Reserach Questions

The research problem was performing the usability analysis on the data extracted from the event log files. To address the research problem following research questions were proposed:

RQ1 How process mining technique can be used to analyse healthcare log file data?

RQ2 What was already used process mining algorithms?

RQ3 How the algorithms can be applied in software process mining for usability analysis?

1.6 Reserach Scope

To prevent the research from getting too broad, scope of the research is encircled here. The main idea of thesis was to develop a tool for GE Healthcare to perform the usability analysis on data in event log files. GE healthcare produces multiple log files to record the events, this thesis research focuses only on clinical log files. Information needed to be extracted from the clinical log files are:

- Actual usage of devices
- List of features that are being used
- Data Statistics

The tool should transform the log data into graphical and presentable form. The tool should give the possibility to the user to see the data and do the analysis on different features e.g. errors, alarms etc. The tool also gives the facility to export reports of the

data.

1.7 Reserach Approach

The theoretical part of this research is based on the literature review. With the literature review answers of the "What is/are" questions were found. This was served as a guideline to further answer the research questions and do the study. The main study in the theoretical part was about the process mining and software process mining. I also learnt about the existing algorithms used in software process mining technique.

After the outcome of theoretical part, the current GE Healthcare patient monitor's software and clinical log files was analysed. This was done with the help of discussion with different people who developed them and from the documentation of the software and their processes. Next, the requirements were gathered from the usability team to find out what they need and expect in the tool to analyse from the clinical log files.

I also developed the algorithms by getting ideas from the existing algorithms to fulfils the requirements of the GE Healthcare tool. As GE Healthcare have some special requirements for the tool so I developed a new algorithm for software process mining by taking the ideas and features from the existing algorithms.

The main idea of this thesis research is to develop the tool for GE Healthcare to do the usability analysis. So, the research was done via design science approach per the Hevner et al. [HMP04]. Design science research approach can be done via several ways, either you can create an artefact by improving and extending the design science knowledge base or by developing and evaluating the design science research methodology as per Hevner et al. [HMP04]. For my thesis, the research lies clearly in creating the design artefact.

For evaluation, I asked the ability team to work on the tool and evaluate that. This helped to check whether tool fulfils the requirements. Also, other people who can use the tool also check the tool for their usage and information extraction. Additionally, it was ensured that when there is a need of further improvements and changes the code and algorithm is flexible to add new changes.

1.8 Thesis Outline

The remainder of the thesis is structured as follows; section 2 provide the preliminary and

detailed theoretical knowledge of event logs, process mining techniques and already developed algorithms. These are necessary to understand this thesis. Section 3 provides the details of methodology used in thesis research which is design science and describe its mapping on the research problem. Section 4 focuses on results of the developed artefact. Section 5 gives you the evaluation of the developed artefact. In Section 6 there is discussion about the research development and conclusion.

2. Theoretical Framework

2.1 Event Logs

Event logs are the essential part of process mining. Event logs file consist of number of events in it. Event log contains information about the events in the specific process. Event logs files are stored in various format like text file, xml format, database etc. But the attributes an event contains is almost the same with every event log files.

An event is defined as an activity, generated by the information system. An event is always unique instance. There is always a sequential relation between event's occurrence. Event consists of many attributes. These attributes are; timestamp of the event i.e. when the event is occurred, originator of the event i.e. who generated or executed the event or from where event is generated.

Table 2.1 : Event log example

Event ID	Timestamp	Message	Originator
1	15-02-2016 11:18:02	Data Inserted	Data View
2	15-02-2016 11:20:15	Print Request	Data View
3	15-02-2016 11:21:20	Enter Credential	Verification View
4	15-02-2016 11:21:54	Verified	Verification View
5	15-02-2016 11:22:10	Prining	Data View
6	15-02-2016 11:24:04	Printing Success	Data View
7	15-02-2016 11:26:32	Close Form	Data View

Event log in Table 2.1 represents the example of data in an event log file. The first column event id is a unique event id column. Second column is the timestamp on which the event is generated. Third column have the activity being executed at that time and fourth column is the corresponding place from where the event is generated, that is the specific view of the software.

2.2 Process Mining

Business process mining or in short form process mining is an evolving research area. The main purpose of process mining is to bring new ways of analysing and improving the business processes by giving the big picture of processes using event logs. Process mining aims to develop understanding of processes behaviour so that the decision makers can analyse them and try to find ways to improve processes behaviour. Process mining gets the information from event log files, which are recorded by the organization's information

system [SS14]. Process mining always starts by collecting information about the processes as they take place instead of making the process design.

Process mining highlights the problems of most the organization that they do not have the

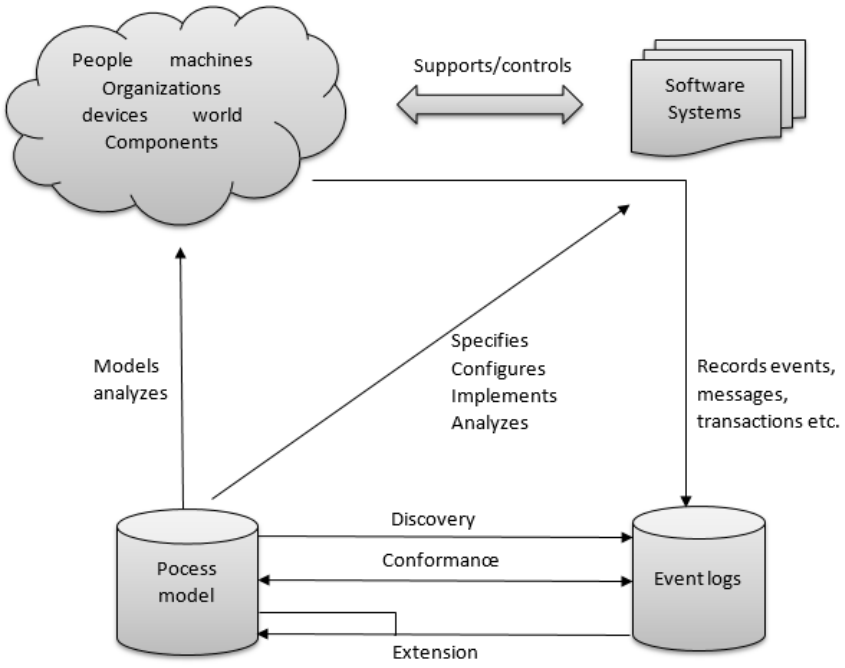

Figure 2.1: Process mining types [SS14]

knowledge of what is going on in the organization. There is always a gap between the what is supposed to be happen with the deployed application and what happen.

Process mining deals with process modelling, process analysis, business intelligence and data mining. There are two ways that process mining can be used by any organization. First way is that process mining is used as a tool to fetch data from event logs and process it for providing information about the actual usage of procedures. One of the main aspects of process mining is to discover control flow, i.e. construct the process model automatically by looking at the real activities and apply dependencies between them. Second way is comparing the actual process with the predefined process to find the differences and loopholes in organization's procedures. If there is already a process model, then process mining can be used to find the differences between the existing model and new discovered model and extend the model with additional information to highlight bottle neck between processes. Process mining helps people to identify how the procedures work for the application, and to compare the predefined process with the actual process [SS14]. Figure 2.1 shows the process mining types along with the structure of processing mining, that

how it works. Process mining types will be discussed in detail in later section.

Process mining can be characterised into three phases. These phases are pre-processing, processing and post-processing. In the pre-processing phase, event log file is read and scanned, and relation between the activities are inferred. In the processing phase, mining algorithm applied, log data and ordering relation serves as input. The post-processing phase output the discovered process model into graphical representation [MDAW04].

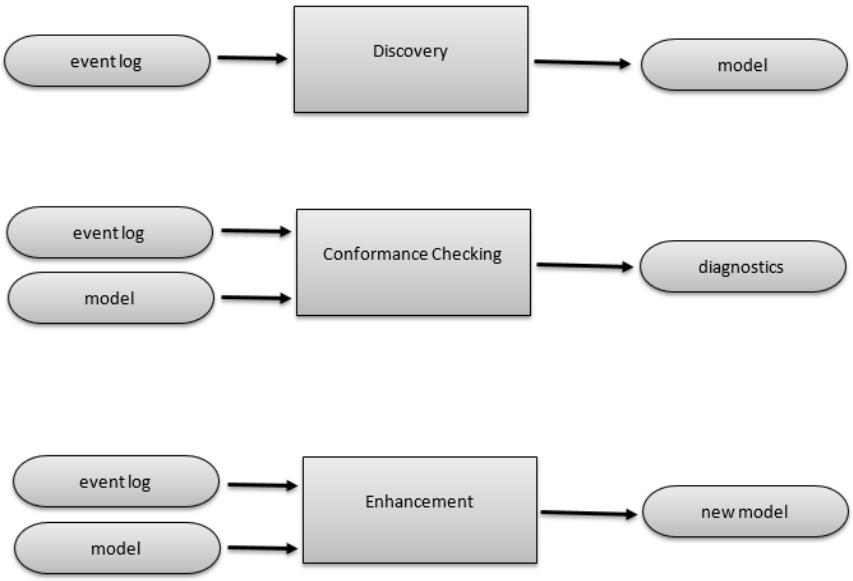

Figure 2.2: Process mining types in terms of input and output [A12]

Process mining has three basic types: discovery, conformance, and extension. Figure 2.2 shows the process mining types in terms of input and output. In discovery part a process model is derived from the observed event log files, and petri net model is constructed based on observed behaviour from the log files. In conformance part the observed model is checked against the already developed model from the predefined process and find out the deviations. In extension part the predefined model is extended by using the facts of conformance and deviation with the observed model. This helps to get the better model with real time usage [MSSAB08].

2.2.1 Process Discovery

The main purpose of process discovery is to identify the model from the event log. In process discovery, different algorithms can be applied to produce process model. Mostly the models are represented in petri net form. The mined model after applying the process

discovery techniques are showed in Figure 2.3. Some common constructs used in mined model are sequence, splits, joins, loops, non-free choice, invisible tasks, duplicate tasks.

2.2.2 Conformance Checking

Process discovery is the starting point of process mining. After process discovery the deeper analysis is possible with the help of produced model. In conformance checking the real model is developed by hand and is compared with the observed model generated from event log. The conformance checking has two angles about the results, first angle I that model is wrong and doesn't have the real behaviour, and second angle is event log is wrong and in real life the events are different, means reality differs from desired model.

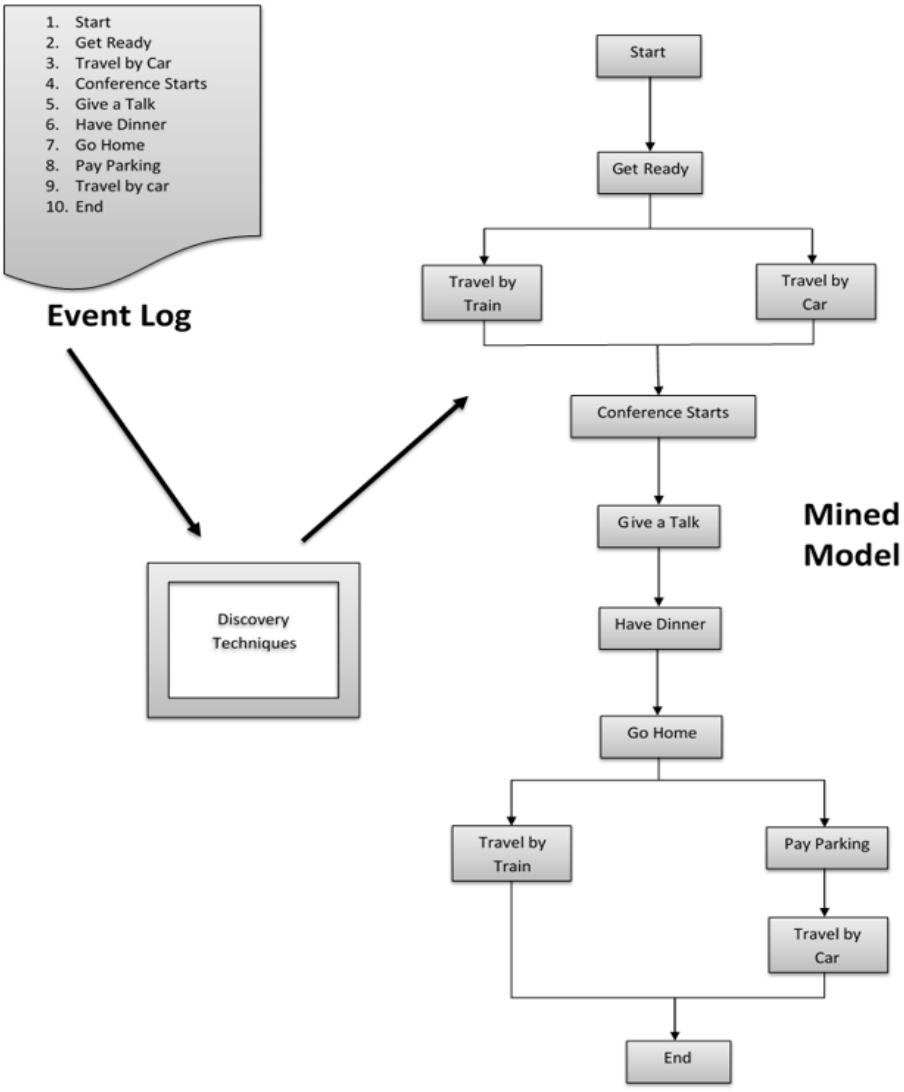

Figure 2.3: Process mining discovery technique [A16]

2.2.3 Model Enhancement

Process models can be improved and extended with the help of observed process from the event log files. Different analysis on observed behaviour can help to find the bottleneck. This helps the organization to correct their process per the analysis [A12].

2.3 Software Process Mining

Process mining technique is not used so far for analysing the deployed application and software behaviour. We use the term software process mining to construct the process specification from the event log files as they occurred and extracting the structure of the process from the real executions.

Process mining technique can be applied in the software [RMLA14]. When a system or application is used, and utilized, the system records the user interaction into the log files. With the help of process mining technique, we can derive the user interface flow models, and this gives the real usage of the application and helps improve the usability of the software.

The focus of software process mining is user workflow. Companies and organizations always face challenges while analysing the user behaviour and user workflow for their innovative software projects. Usability and user acceptance are complicated and thus require the robust solution for usability analysis [OB91]. The focus of this thesis research is on software process mining.

2.4 Algorithms

There are various number of algorithms that are used in process mining. But I will discuss the highly used algorithms. There are two largely used tools named ProM toolkit (www.promtools.org/) and Fluxicon - Disco (https://fluxicon.com/disco/). These frameworks most commonly use fuzzy mining and heuristic mining algorithms to fulfil the needs of the user. Fluxicon - Disco basically used fuzzy mining algorithm. On the other hand, ProM toolkit contain more than 600 plugins and it also have wide selection of mining techniques. Generally utilized techniques include fuzzy mining and heuristic mining. These algorithms are used in process discovery and called control-flow mining algorithms [A16].

2.4.1 α -algorithm

Alpha mining algorithm reads log files and get activities from it to infer the ordering relations. Based on ordering relations alpha mining algorithm build petri net diagram. The core step of this algorithm is ordering relation [PM16]. By looking at the example of log file in Table 2.2, it can be identified that the case 1 has task sequence of ABCD, case 2 has ACBD, case 3 has ABCD, case 4 has ACBD and case 5 has EF. So, the possible sequences in this log files are ABCD, ACBD and EF.

Table 2.2: Sample event log

Case 1	Task A
Case 2	Task A
Case 3	Task A
Case 3	Task B
Case 1	Task B
Case 1	Task C
Case 2	Task C
Case 4	Task A
Case 2	Task B
Case 2	Task D
Case 5	Task E
Case 4	Task C
Case 1	Task D
Case 3	Task C
Case 3	Task D
Case 4	Task B
Case 5	Task F
Case 4	Task D

In alpha algorithm, there are some relation that should be followed when making the petri net diagram. These are as follow:

- Direct Succession: x > y if for some cases x is directly followed by y. For our sample case (Table 2.2) A > B, A > C, B > C, B > D, C > B and E > F.

- Causality: x → y if x > y and not y > x. For our sample case (Table 2.2) A → B, A → C. B → D, C → D and E → F. Consider the case x → y, its notation is shown in Figure 2.4 in column a.
- Parallel: x || y if x > y and y > x. For our sample case (Table 2.2) B || C and C || B. Consider the case x → y, x → z and y || z, its notation is shown in Figure 2.4 in column b. The other case with x → z, y → z and x || y, its notation is shown in Figure 2.4 in column c.
- Choice: x # y if not x > y and not y > x. For our example case (Table 2.2) A →D. Consider the case x → y, x → z and y # z, its notation is shown in Figure 2.4 in column d. The other case with x → z, y → z and x # y, its notation is shown in Figure 2.4 in column e.

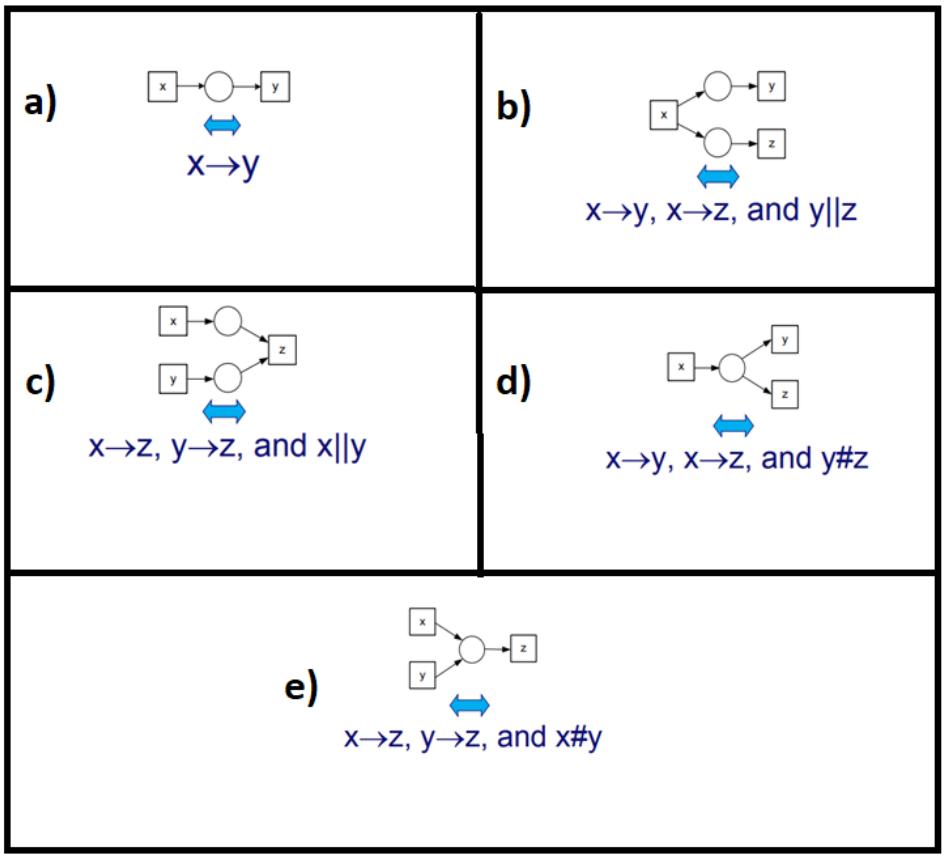

Figure 2.4: Alpha algorithm relations [PM16]

So, the petri net diagram based on the sample case (Table 2.2) is shown in Figure 2.5 using the notations discussed above, here the start node is specified with a circle with black hole and the end state as hollow circle.

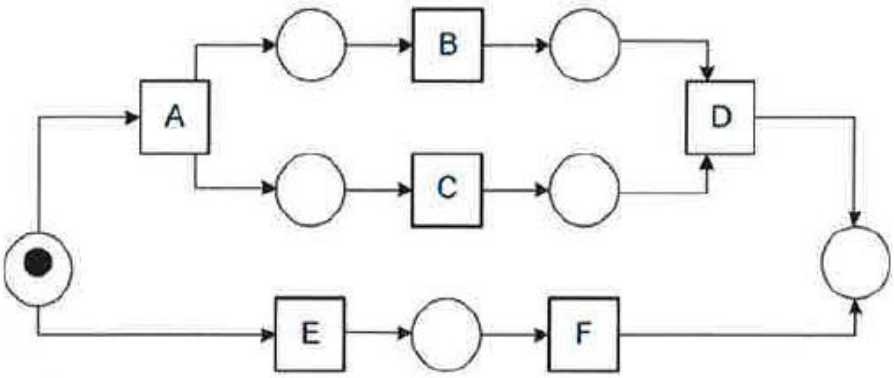

Figure 2.5: Petri net diagram [PM16]

Figure 2.6 shows the formalization of alpha algorithm. The overview of the alpha algorithm at high level is as follows [PM16]:

Step 1: Define all the events occurred in the event log.

Step 2: Define all the possible start events.

Step 3: Define all the possible end events.

Step 4: Calculate possible sets A and B such that all events within A and within B are independent of each other and all events in A should be causally related to events in B.

Step 5: Drop the non-maximum sets from the sets obtained in Step 4.

Step 6: Create places for all derived Sets and add start + end state.

Step 7: Draw the connections between places created according to the sets.

Step 8: Return the petri net.

Let W be a workflow log over T. $\alpha(W)$ is defined as follows.

1. $T_W = \{ t \in T \mid \exists_{\sigma \in W} t \in \sigma \}$,
2. $T_I = \{ t \in T \mid \exists_{\sigma \in W} t = first(\sigma) \}$,
3. $T_O = \{ t \in T \mid \exists_{\sigma \in W} t = last(\sigma) \}$,
4. $X_W = \{ (A,B) \mid A \subseteq T_W \wedge B \subseteq T_W \wedge \forall_{a \in A} \forall_{b \in B} a \rightarrow_W b \wedge \forall_{a1,a2 \in A} a_1 \#_W a_2 \wedge \forall_{b1,b2 \in B} b_1 \#_W b_2 \}$,
5. $Y_W = \{ (A,B) \in X \mid \forall_{(A',B') \in X} A \subseteq A' \wedge B \subseteq B' \Rightarrow (A,B) = (A',B') \}$,
6. $P_W = \{ p_{(A,B)} \mid (A,B) \in Y_W \} \cup \{i_W, o_W\}$,
7. $F_W = \{ (a,p_{(A,B)}) \mid (A,B) \in Y_W \wedge a \in A \} \cup \{ (p_{(A,B)},b) \mid (A,B) \in Y_W \wedge b \in B \} \cup \{ (i_W,t) \mid t \in T_I \} \cup \{ (t,o_W) \mid t \in T_O \}$, and
8. $\alpha(W) = (P_W, T_W, F_W)$.

Figure 2.6: Alpha algorithm formalization [PM16]

2.4.2 Heuristic Mining Algorithm

Heuristic mining algorithm is an improvement of alpha algorithm. It differs in three aspects from the alpha algorithm, firstly it considers the frequency of the events, secondly it is able to detect short loops, and lastly it detects the skipping of single activities [PM16]. The heuristic miner algorithm can be expressed in the following steps [PM16]:

1. Read a log file.
2. Get the set of tasks/events.
3. Infer the ordering relations based on their frequencies.
4. Build the net based on inferred relations.
5. Output the net.

Steps 1 and 2 are simple and so is the 4rth and 5th. The main step of the algorithm is 3rd which states that we need to create ordering relations based on the frequencies. To better understand this let's take a simple example.

1. Sequence a,b,c,d is repeated 6 times
2. Sequence a,c,d,b is repeated 2 times
3. Sequence a,b,d,c is repeated 12 times
4. Sequence a,d,c,b is repeated 4 times

As a first step it builds a directly follows frequency matrix i.e. directly follows a > b (a is directly followed by b). $|a > b|$ is the number of times a > b occurs in the log. Matrix created using the following information will be shown in Table 2.3.

Table 2.3: Heuristic mining matrix example

	A	**b**	**C**	**D**
A		18	2	4
B			6	12
C		4		8
D		2	16	

By looking into the sequences 1 and 3 we see that |a > b| value is 18. The rest of the matrix is filled using the same technique. For the second step it creates the dependency matrix whose formula is as followed:

$$|=>| = \frac{|a > b| - |b > a|}{|a > b| + |b > a| + 1}$$

It's a simple calculation formula where we will use information from the frequency matrix to fill out our dependency matrix shown in Table 2.4.

Table 2.4: Heuristic mining dependency matrix example

	a	**b**	**C**	**D**
A		0.95	0.66	0.8
B	-0.95		0.18	0.67
C	-0.66	-0.18		-0.32
D	-0.8	-0.67	0.32	

Let's try to fill a to b and b to a place in the matrix. Values for |a>b| is 18 and |b>a| is 0 based on the frequency matrix. So,

a to b $=> \frac{18-0}{18+0+1} = 0.95$

b to a $=> \frac{0-18}{0+18+1} = -0.95$

Similarly, other places of the matrix can be filled accordingly. The values we will get will be in range of -1 and 1. Positive value means that the relation is strong and negative means the relation is weak. Using the information of the above two matrices the petri net can be created.

2.4.3 Fuzzy Mining Algorithm

The above-mentioned algorithms are good to find processes within structured log data files or processes, they provide you impressive results when the process in consideration is well structured and are enforced to be systematic. But these algorithms failed when the

process under discussion is not well structured.

When applied to unstructured data the result they produce is entirely correct, but output is a mess and termed that as “spaghetti models” [GCW07]. The problem is that the messy output is hard to understand, and the amount of information displayed on the page is useless to the reader as it shows too much information. Hence, in order to cope this problem, the mining algorithm which is considering less structured data should provide a high-level view with abstractions for the reader to understand it at a higher level. This will be of importance for the end users as they will be able to get out of it instead of just a messy network like output where there is no prominent pattern to identify.

Günther et al. [GCW07] identified two elementary metrics Significance and Correlation that can be used to create such an output. They specified that frequency could be to measure the significance of events. As the events occurring more are for sure significant and should be given high priority. For correlation they defined it as how closely related two events following one another are, which can be measured based on different techniques that will vary on the given problem. Following these two metrics the simplification process developed by Günther et al. [GCW07] is:

- Highly significant behaviour is preserved as it is in the model.
- Less significant and highly correlated behaviour is aggregated (combined) in a cluster.
- Less significant and low correlated behaviour is abstracted (removed) from the model.

To Conclude, it is important to note that Fuzzy Mining does not create a petri net but instead creates a graph based on the above-mentioned points that can be characterized as high-level graph.

3. Methodology and Research Design

In this chapter design of the research is elaborated. This chapter first provide the brief description of the design science research approach. Next this chapter describe the methods used for solving the problem of this thesis. Further this chapter provide the methods used for validation the artefact.

3.1 Design Science

Design science approach is originally from the domain of engineering and has gained attention in information systems (IS) research [KGM12]. Nunamaker et al. [NJCP90] present the idea that systems development could produce the valuable results in the field of information systems (IS), if considers and view it as research methodology. Nunamaker et al. [NJCP90] have also presented the process of systems development research with the processes consist of construct a conceptual framework, develop a system architecture, analyse and design system, build the prototype, and observe and evaluate system.

Orlikowski et al. [OB91] research work shows the study of information technology, organization's research approaches and most of their research focus is on differentiating the design science research with behavioural science (also known as natural science) and social science. In March et al. [MS95] research, they combine the knowledge of natural science research with design research and proposed a framework with activities half taken

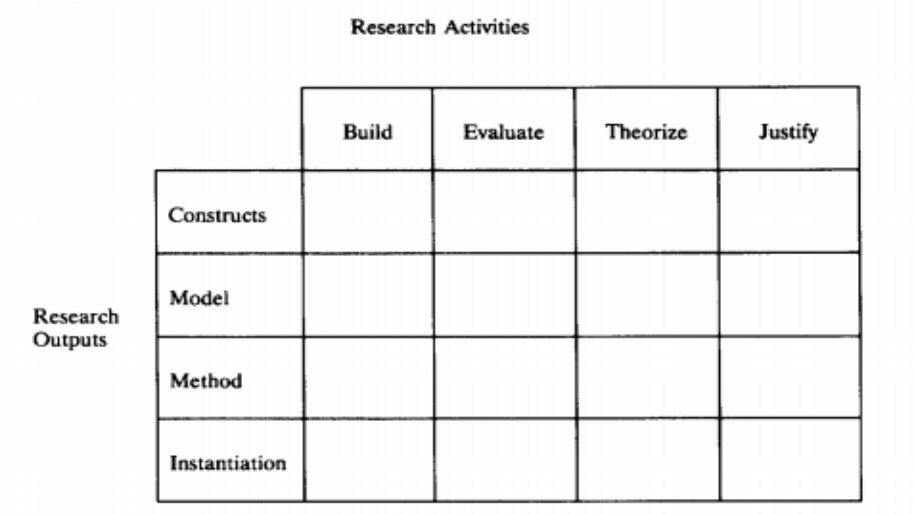

Figure 3.1: Design science research framework by March et al. [MS95]

from natural science research and half taken from design research. Figure 3.1 shows the research framework by March et al. [MS95]. The framework is not very innovative but

unpolished and indefinite as it represents the characteristics of the early researches of design science articles.

After this, a lot of work has been published that defines the paradigm of design science research in information systems. These researches include ontology of design science and including the artefact in design science by Orlikowski et al. [OI01], Benbasat et al. [BZ03], Iivari et al. [I07], Baskerville et al [PBV08]. There are also other researches that focuses on methodology of design science to create and evaluate design with some proposed methods Hevner et al. [HMP04] and Sein et al. [SHPRL11].

Hevner et al. [HMP04] continues the idea of March et al. [MS95] and combine the natural science research with design science research. Hevner et al. [HMP04] takes the theory from Silver et al. [SMB95] to add up in the research. So far, several design science research frameworks have been proposed and existed. These frameworks have several phases and each phase has some set of milestones to achieve. Most of them are with iterative approach with several cycles of design process.

To understand the context of this thesis research, five steps taken from Takeda et al. [TVY90] methodology is provided to present the development of project in a natural way. These steps are shown in Figure 3.2.

Awareness of problem came through the discussions with the GE Healthcare personnel and with academic supervisor. It was identified that most of the people in GE healthcare uses the clinical log files for different purposes and it is difficult for them to analyse the data and find out the required messages from the clinical log files. GE Healthcare keen to have a tool that gives the statistics on overall data of clinical log file as well as provide the usability analysis. The outcome of this process step was an agreement with GE Healthcare and candidate for a master thesis work sponsored by GE Healthcare.

In the suggestion process step, there was ideas of tools on how to present the clinical log file in human readable form to understand the data better and quickly. This step required the understanding of clinical log files in depth and finding out ways to present the features in tool. The output of this process step was a thesis description that list down the features of the tool that was going to be in the tool.

Development and evaluation process step contain the empirical work done for this thesis. The output of this process was artefact in the form of tool that fulfils the requirements of the data statistics and usability analysis of clinical log files. The output of the evaluation

process step was assessment of the tool based on recommended criteria.

Finally, the conclusions along with research process description, artefact, results and evaluation was embodied in this thesis publication.

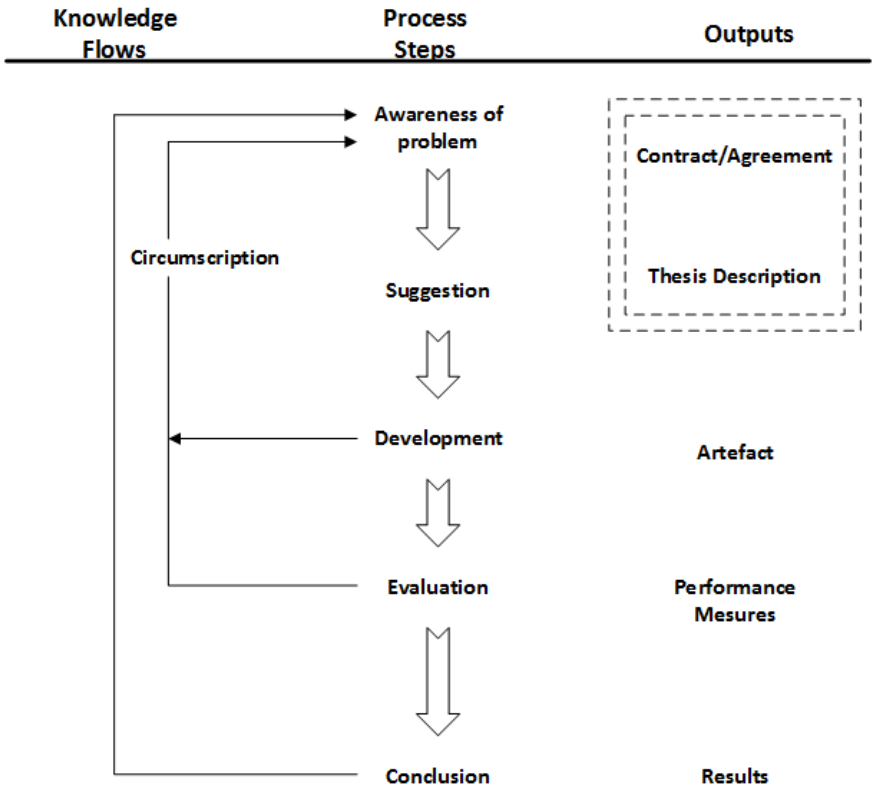

Figure 3.2: Design science research process by Takeda et al. [TVY90]

3.2 Design Science Research Methodology

For this thesis end users were interviewed to understand the current tools used in GE Healthcare, and perceive the domain knowledge as well to better understand their business needs, processes and end user's expectations. For this purpose, a framework is needed that would take care of the research part also. Unfortunately, the design research process by Takeda et al. [TVY90] and design Science research framework by March et al. [MS95] did not offer this dilemma. That is why the framework by Hevner et al. [HMP04] is used.

To provide the usability analysis using the clinical log files, it is very important to understand what form a suitable solution might take and how it would be used in practice. The primarily concern of this thesis was producing a tool useful to practitioners of GE Healthcare as well as the researchers.

The specific model of design science selected to use in this thesis research is that presented by Hevner et al. [HMP04]. This model is selected because it is well-developed, also latest and published in the top journal of Information Systems. This makes it as of high quality, accepted by researchers and likely to be a reference source for several future projects. It also presents several guidelines for critically evaluating design science research.

Figure 3.3 shows the design science model proposed by Hevner et al. [HMP04]. This model explicitly has two modes which is develop/build and justify/evaluate. These modes are linked with business needs (relevance) and application knowledge (rigor). As Hevner et al. [HMP04] argue, IS research needs to be rigorous to provide an "addition to the knowledge base", and relevance allows for "application in the appropriate environment". This thesis proceeds with identifying the key elements from the model, and then applying these elements into thesis research.

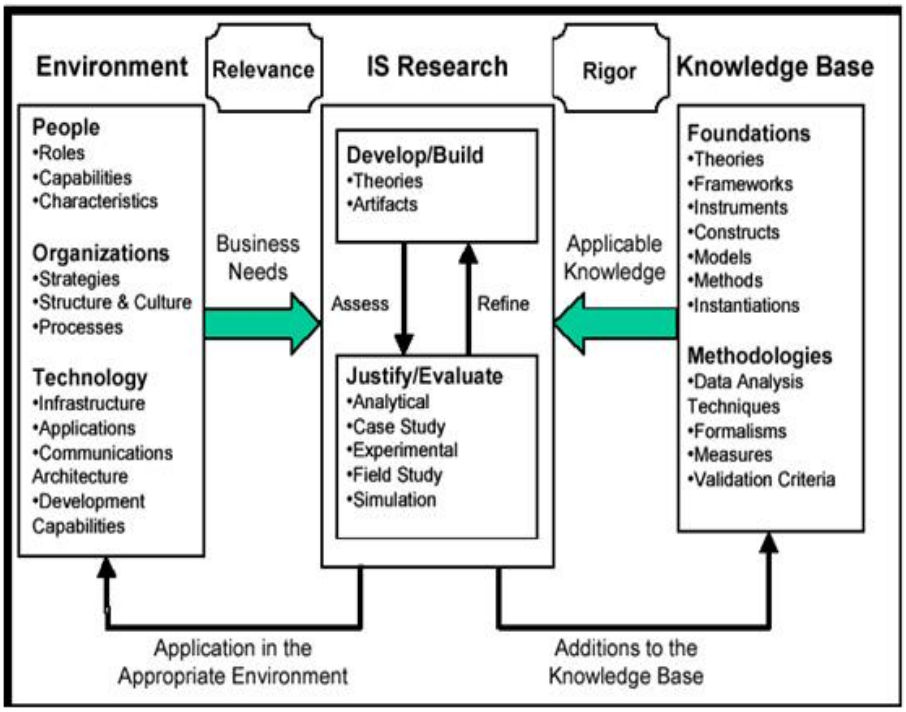

Figure 3.3: Design science research framework by Hevner el al. [HMP04]

Figure 3.4 presents the thesis research mapping on Hevner et al. [HMP04] framework of design science research. The figure shows the artefact to be developed, evaluation, foundations in the knowledge base, and applied methodologies. The main pattern of this framework approach is to understand the requirements and needs first, then develop artefact and evaluate them.

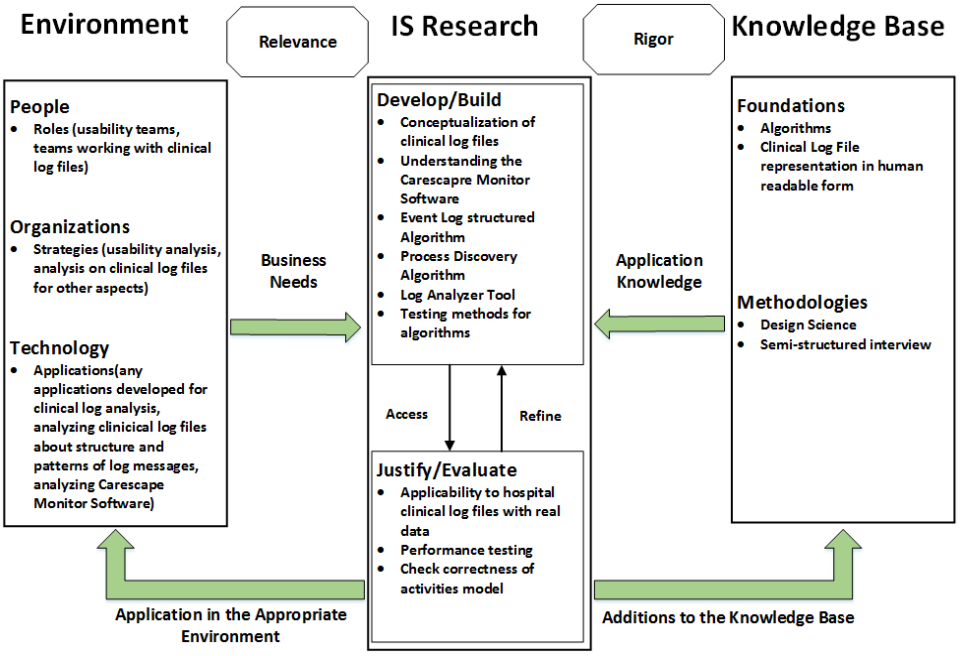

Figure 3.4: Mapping of Hevner el al. [HMP04] framework on thesis research

3.2.1 Business Needs

The mapping of design science framework begins with business needs. Business needs ensures that the design science research meets the goals of relevance. Per the Hevner et al. [HMP04] business needs are assessed within the context of organizational strategies, structures, culture and existing business processes. Therefore, for understanding the business needs for the usability analysis tool, there is a need to look out the elements of business in the design science model.

3.2.1.1 People

From the thesis research problem, it is clear that target people and audience are the usability teams and the teams using the clinical log files. Information from the usability team was collected that how they were doing the usability analysis and what were their expectations as well. Also, the other teams and persons who were using the clinical log files, it was good to ask them and know what information they were using from the clinical log files and how they were extracting them.

3.2.1.2 Organization

In this research thesis, I got the strategies from the organization about the usability analysis. Without the proper knowledge, it was not possible to produce the tool in the desired form. There should also be the analysis on the clinical log files. How they were created, how messages are logged, pattern of messages, time of messages and how I would extract the information from them.

3.2.1.3 Technology

From the technology point of view, clinical log files were saved into the carescape patient monitors and would be extracted if the hospital gave permission or if there was any errors or bugs while using the patient monitors. GE healthcare save those data got from the hospital in their network drives with permissible person using them. I used those clinical log files for the tool.

3.2.2 IS Research

For a work body to count as design science it must produce and evaluate a novel artefact Hevner et al. [HMP04]. The research thesis seeks to achieve the artefact by developing a new tool which will take the clinical log file as an input and do the usability analysis.

3.2.2.1 Develop/Build

The artefact that is produced for this thesis was a system which has the functionality of importing the clinical log file into your local system database. It also gives some data statistics on the log file messages and gives you the user activities chart for usability purpose. The things covered into the development of artefact was as follows:

- Conceptualization of clinical log file
- Understanding the carescape patient monitor software
- Develop event log structure algorithm
- Develop process discovery algorithm
- Develop log analyzer tool
- Prepare test methods for log analyzer tool

3.2.2.2 Justify/Evaluate

To ensure that the artefact is useful for organization's targeted people and as well as for knowledge base, it should go under rigorous evaluation and justification. The artefact not only provide the numerical results, but it also gives the user confidence that the output is valid. The goals of this thesis research evaluation are as follows:

- Tool should be applicable and work with the hospital clinical log files with real data.
- Test the performance of tool in term of time taking to run the algorithms.
- Test the tool in terms of correctness of data and activities model.

All the evaluation methods describe by Hevner et al. [HMP04] was considered for evaluation the tool developed for the thesis research. Table 3.1 shows the evaluation methods used to evaluate this research thesis artefact. In the Table 3.1 discussion is given about which evaluation method is used in thesis artefact evaluation.

Table 3.1: Evaluation methods

Evaluation Method	Description	Discussion
Observational	**Case Study:** Study artefact in depth in business environment	This is applicable for my research thesis because the real hospital data will be used and inserted in the tool to monitor its behaviour.
Experimental	**Controlled Experiment:** Study artefact in controlled environment for qualities (e.g., usability)	This is applicable for my research thesis because the tool will be given to the usability team for the analysis and they will report the results after experimentation.
	Simulation - Execute artefact with artificial data	This is applicable for my research thesis because for initial testing purpose I will use the in-house development data to test my algorithms functionality.

Testing	**Functional (Black Box) Testing:** Execute artefact interfaces to discover failures and identify defects	This is applicable for my research thesis. End Users will be the ones testing the functionality and reporting their results and feedback.
	Structural (White Box) Testing: Perform coverage testing of some metric (e.g. Execution paths) in the artefact implementation	This is applicable for my research thesis. I will be the one responsible to test my application with valid and invalid inputs to verify the outputs.

3.2.3 Applicable Knowledge

In design science research methodology, to complete the artefact the research gets the knowledge from the existing knowledge base. This information can be from anywhere; it can be from previous work and papers as well.

3.2.3.1 Foundations

In the foundation part of the design science research, the idea of process mining algorithm research is taken from the previous work. As they were applicable for my thesis problem, but they were not fully utilized per the GE Healthcare scenarios because their requirements and business needs were different. Therefore, those algorithms were studied, and new algorithms was developed for the thesis research artefact.

There were also some tools already developed in the field of process mining. Those tools were studied, and observation is done on how they were representing the event log files in human readable form. After the tool's algorithms were studied, development of the functionality of making the clinical log file's user activities is done which was represented in user readable form.

3.2.3.2 Methodology

The next area to focus was the design science research method usage and the semi-structure interview. That is studied from previous researches which shows that how to conduct them and sae is done for this thesis.

3.3 Overall Research Design

Now design science research approach and its needs are understood, here is the overall

research design for this thesis.

1 **Literature Review**: The first phase consists of gathering the knowledge form the domain of process mining literature. For this thesis research was carried from scholarly journals, conference proceedings, technical reports, and seminar papers.

2 **Interviews**: The next phase is understanding the business needs of GE healthcare. Therefore, some interviews were conducted with the usability teams and the persons who were working and generating the log files to fully understand everything. All the conversations were face-to-face.

3 **Conceptual Study and Development**: In this phase concepts were built about what was going to be in the tool and then start developing it per the plan.

4 **Evaluation**: In the last, tool was evaluated by testing the functionality using the in-house clinical log files as well as with the real hospital clinical log files.

4. Research Results

In this chapter the solution of the research problem is presented.

4.1 Process Mining Methodology

Process discovery method was used to show the workflow of the clinical event log files in the tool. Process discovery method and algorithms are already discussed in chapter 2 with detail. Process mining has three parts, discovery, conformance checking and enhancement. For this thesis research, only the process discovery method is used to find the workflow of the clinical event log file and do the usability analysis.

While analysing the clinical event log files of GE healthcare, it was identified that it did not had any structure that identified the cases uniquely. Algorithms that was discussed in section 2.4 take the event log file in some structure form, in which the processes were uniquely identified. There was a need of algorithm that first make the structure of the event log files. After that process discovery algorithm was developed and applied on structured data.

Two algorithms were developed in this thesis; one was structure algorithm and the second was discovery algorithm. In the following section the algorithms are discussed in detail how they were created, and the pseudo code of the algorithms are provided.

4.2 Technology used to Develop Log Analyzer Tool

As the thesis research follows the design science research methodology. There should be a tool to fulfils the research problem in the form of artefact. To develop the tool, following technology was used:

- MS Visual Studio 2015: MS Visual Studio 2015 with C# was used to develop the tool. The tool was a window-based application which runs on windows.
- Infragistics for Windows Forms: This is a third-party tool used in MS Visual Studio to represents the data in different types of charts.
- Mind Fusion Diagramming for Windows Forms: This is a third-party tool used in MS Visual Studio to represents the activities flow diagrams.
- MYSQL Database: A MYSQL 5.5 database was used to store the data into the

database and then show them into the Log Analyzer Tool (name of the artefact developed for this thesis).

4.3 Structure Algorithm Development

Structure algorithm was developed to identify the processes and their sub processes. The clinical pattern had a unique log number, date and time, message type, message, generated from, and thread. Without identifying the processes and sub process thee process discovery algorithm cannot be applied and developed. The raw clinical event log files should be converted to the pattern which contains a case id (which uniquely identified the process and its steps) and other required columns. So that the algorithm identifies that case 1 process has two steps; case 2 process has four steps and so on.

For this purpose, it was necessary to look at the carescape patient monitor and analyse the menu and sub menus and identified what message was generated in the event log file

```
6735|2015-May-19 21:16:13|<ALR>|_i+Lead off|AlarmLog.cpp:649|AlarmEng
6736|2015-May-19 21:16:16|<ALR>|La+Leads off|AlarmLog.cpp:649|AlarmEng
6737|2015-May-19 21:16:43|<ALR>|_i+Leads off|AlarmLog.cpp:649|AlarmEng
6738|2015-May-19 21:16:46|<ALR>|La+Lead off|AlarmLog.cpp:649|AlarmEng
6739|2015-May-19 21:16:57|<ALR>|_i+Lead off|AlarmLog.cpp:649|AlarmEng
6740|2015-May-19 21:17:09|<ALR>|_i+Arrhythmia paused|AlarmLog.cpp:649|AlarmEng
6741|2015-May-19 21:34:46|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6742|2015-May-19 21:34:54|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6743|2015-May-19 21:35:03|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6744|2015-May-19 21:35:12|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6745|2015-May-19 21:35:21|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6746|2015-May-19 21:35:30|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6747|2015-May-19 21:35:39|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6748|2015-May-19 21:35:48|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6749|2015-May-19 21:35:56|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6750|2015-May-19 21:36:05|<NFO>|WLAN Connected, RTClinicalDSCP 34, NonRTClinicalDS(
6751|2015-May-19 21:40:42|<NFO>|1 message(s) above has been repeated 31 time(s)|Lo
6752|2015-May-19 21:47:57|<USR>|Trends:MainMenuScr1|ServiceDataCtrl.cpp:338|GUI
6753|2015-May-19 21:47:57|<NFO>|Removed 15 expired events from event archive, max
```

Figure 4.1: Clinical event log

when the specific menu or submenu is clicked. By doing this the pattern of message generated in the event log file is analysed. Figure 4.1 shows the clinical event log file.

As thesis research majorly focused on usability analysis, thus the focus of getting the only user clicked messages from event log files. GE clinical event log files contains different types of messages which include errors, information, alarms, users etc. The focus was on <USR> type messages. The raw clinical log files converted to a pattern with case ID look like Table 4.1.

Table 4.1: Event log with case id

Case ID	Timestamp	Message
1	15-02-2016 11:18:02	Data Inserted
1	15-02-2016 11:20:15	Print Request
2	15-02-2016 11:21:20	Open Form
2	15-02-2016 11:21:54	Enter Credentials
2	15-02-2016 11:22:10	Check Information
2	15-02-2016 11:24:04	Close Form
3	15-02-2016 11:26:32	Open Document

4.3.1 Carescaer Patient Monitor Analysis

Figure 4.2 shows the carescape patient monitor software. In Figure 4.2 the bottom part contains the user menus which have submenus as well. The right side of the figure is the

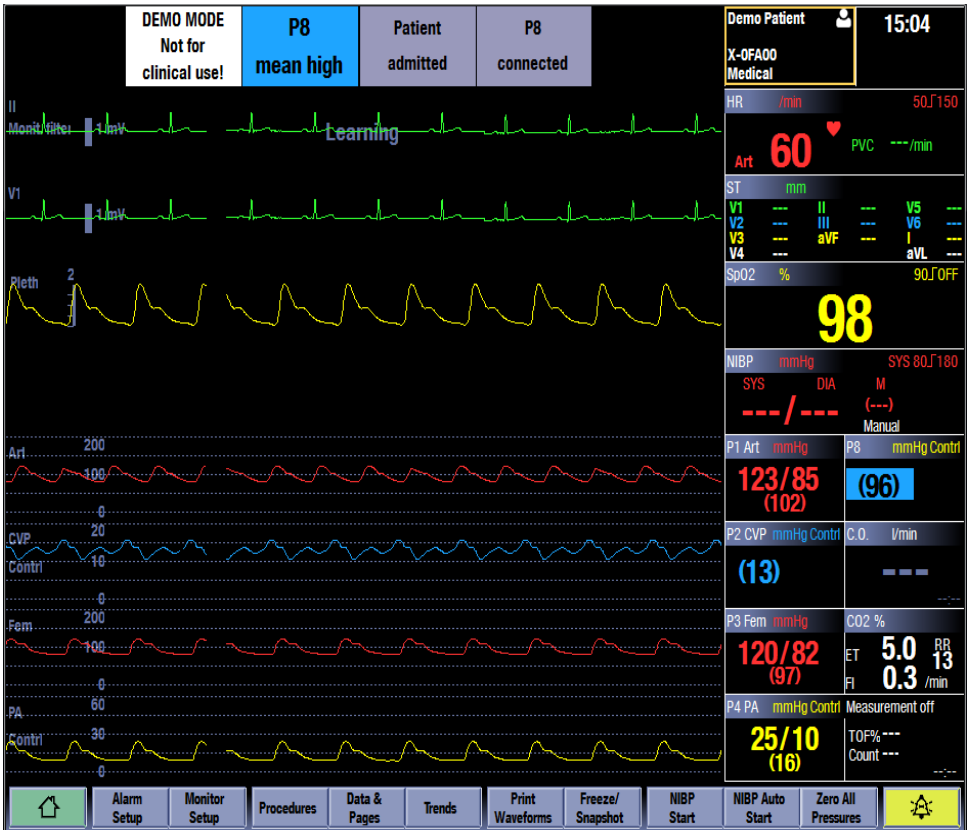

Figure 4.2: Carescape patient monitor software

parameter window where all the shortcuts of monitor functionality is displayed. It also has shortcut to the specific menu.

As the focus is on user messages, it is identified that the pattern for all the menus in the bottom and the parameter window on the right side should be checked. It is obvious that whenever something from these areas are clicked a process has started and previous should be ended as everything worked in sequence. There are menus which has sub menus and some of the menus have just stand-alone functionality and it doesn't have any sub menus. For this a table was created in the database to identify that which menu was single and which menu was extended. The parameter window button on the right side was also considered as menu because these were also the shortcuts of some menus. The table structure of menu information is discussed in section 4.3.3.

4.3.2 Event log Message Pattern Recognition

For identifying the message pattern generated when any menu is clicked, there is a need to check and analyse the messages and find out what is common in main menus and sub

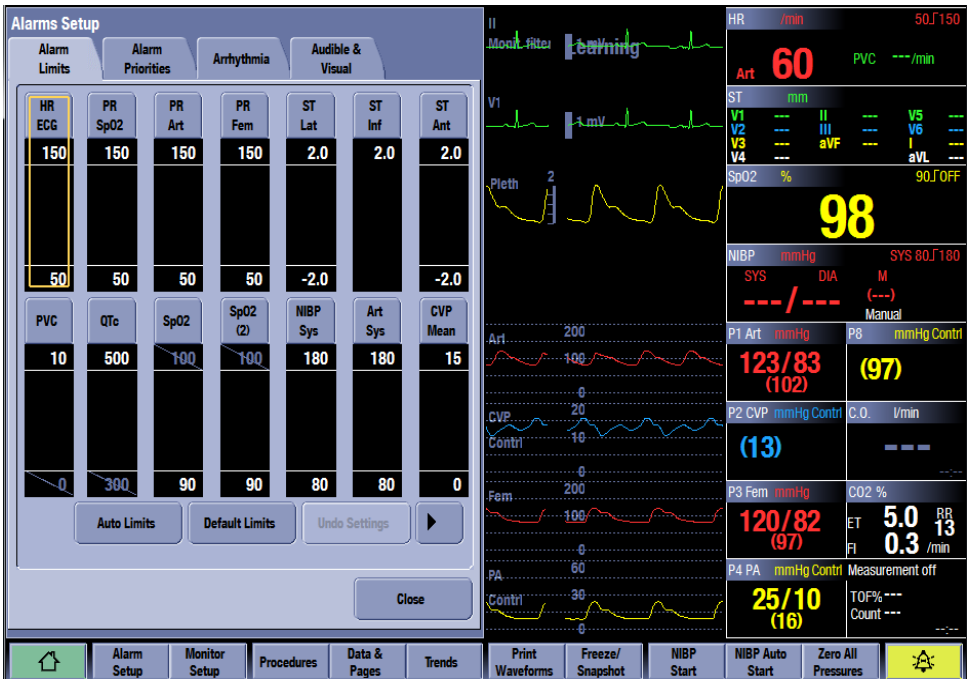

Figure 4.3: Carescape patient monitor alarm setup menu

menus. Let's take the example of "Alarm Setup" in bottom of the menu strip of carescape patient monitor. Figure 4.3 shows the Alarm Setup Submenu. And Figure 4.4 shows what message was generated in the event log file when the menu was clicked. The message is highlighted in the figure. The message is "Alarm Setup:MainMenuScr1" and when Close

button was clicked, “Close:Alarm Limits:Alarms Setup” message is generated. It looks like for all the menus in the bottom the pattern of the main menu message is name of the menu, then a colon and after that keyword that identified it was clicked from main menu screen area which is “MainMenuScr1”. Same thing goes with the parameter window area except it has the keyword as “Horizontal Screen 1” and “Vertical Screen 1”. So, it is extracted that if we take the keywords “MainMenuScr1”, “Horizontal Screen 1”, “Vertical Screen 1” as a process start activity and previous process close activity because in this scenario activities do not run in parallel, so previous activity must close before new activity start.

```
451|2017-Jan-10 15:56:18|<NFO>|Enabled automerge timer: start 1484063320, fire at 1484149720|AutomergeTimer.cpp:77|PdmNetApp
452|2017-Jan-10 15:56:18|<NFO>|processUpdateCase sets netCasePresentM=1, was 1|PdmConsumer.cpp:903|PdmNetApp
453|2017-Jan-10 15:56:18|<NFO>|Network down, won't start case saving|PdmConsumer.cpp:1727|PdmNetApp
454|2017-Jan-10 15:56:18|<NFO>|Failed in processUpdateCase:  no conn|PdmConsumer.cpp:920|PdmNetApp
455|2017-Jan-10 15:56:19|<USR>|Alarm Setup:MainMenuScr1|ServiceDataCtrl.cpp:338|GUI
456|2017-Jan-10 15:56:22|<NFO>|Enabled automerge timer: start 1484063320, fire at 1484149720|AutomergeTimer.cpp:77|PdmNetApp
457|2017-Jan-10 15:56:22|<NFO>|processUpdateCase sets netCasePresentM=1, was 1|PdmConsumer.cpp:903|PdmNetApp
458|2017-Jan-10 15:56:22|<NFO>|Network down, won't start case saving|PdmConsumer.cpp:1727|PdmNetApp
459|2017-Jan-10 15:56:22|<NFO>|Failed in processUpdateCase:  no conn|PdmConsumer.cpp:920|PdmNetApp
460|2017-Jan-10 15:56:24|<ERR>|ESP CPU usage 97 % > 90 % (esp:97, X:0, pulseaudio:0)|IdleTimeCalc.cpp:195|Diagnostic
461|2017-Jan-10 15:56:25|<USR>|Close:Alarm Limits:Alarms Setup|ServiceDataCtrl.cpp:338|GUI
462|2017-Jan-10 15:56:26|<NFO>|Enabled automerge timer: start 1484063320, fire at 1484149720|AutomergeTimer.cpp:77|PdmNetApp
463|2017-Jan-10 15:56:26|<NFO>|processUpdateCase sets netCasePresentM=1, was 1|PdmConsumer.cpp:903|PdmNetApp
464|2017-Jan-10 15:56:26|<NFO>|Network down, won't start case saving|PdmConsumer.cpp:1727|PdmNetApp
465|2017-Jan-10 15:56:26|<NFO>|Failed in processUpdateCase:  no conn|PdmConsumer.cpp:920|PdmNetApp
```

Figure 4.4: Event log generated by Alarm Setup Menu

4.3.3 Database Table for Menu Information

To make the algorithm dynamic it’s better to have a table in the database that have all the information of menus. User can add new menus and change the names as the software revolve and changes done into the carescape patient monitor software. The structure of the database table looks like in Figure 4.5. For fields description see Appendix A.

4.3.4 Database Table for Condition

During testing it was seen that there were some menus whose click were not logged into the log file, considering those an exception and to cater this problem a table was created to save such conditions in the Condition table. User can add the condition to let the algorithm know that this is the start of the process. The structure of the database table looks like in Figure 4.6. For fields description see Appendix A.

menuinfo_id	menu_type	menu_name	menu_status	type
1	MainMenuScr1	Alarm Setup	Extended	NORMAL
2	MainMenuScr1	Monitor Setup	Extended	NORMAL
3	MainMenuScr1	Procedures	Extended	NORMAL
4	MainMenuScr1	Data & Pages	Extended	NORMAL
5	MainMenuScr1	Trends	Extended	NORMAL
6	MainMenuScr1	Print Waveforms	Single	NORMAL
7	MainMenuScr1	Freeze/ Snapshot	Single	NORMAL
8	MainMenuScr1	NIBP Start	Single	NORMAL
9	MainMenuScr1	NIBP Auto Start	Single	NORMAL
10	MainMenuScr1	Zero All Pressures	Single	NORMAL
11	MainMenuScr1	Silence Alarms	Single	NORMAL
12	Vertical DigitField Screen 1	HR DigitField	Extended	NORMAL
13	Vertical DigitField Screen 1	SpO2 DigitField	Extended	NORMAL
14	Vertical DigitField Screen 1	NIBP DigitField	Extended	NORMAL
15	Vertical DigitField Screen 1	P1 Parameter Window	Extended	NORMAL
16	Vertical DigitField Screen 1	P8 Parameter Window	Extended	NORMAL
17	Vertical DigitField Screen 1	P2 Parameter Window	Extended	NORMAL
18	Vertical DigitField Screen 1	P3 Parameter Window	Extended	NORMAL
19	Vertical DigitField Screen 1	CO2 Parameter Window	Extended	NORMAL
20	Vertical DigitField Screen 1	P4 Parameter Window	Extended	NORMAL
21	Vertical DigitField Screen 1	NMT	Extended	NORMAL
22	Vertical DigitField Screen 1	EEG1	Extended	NORMAL
23	Vertical DigitField Screen 1	EEG2	Extended	NORMAL
24	Vertical DigitField Screen 1	EEG3	Extended	NORMAL
25	Vertical DigitField Screen 1	EEG4	Extended	NORMAL
26	Horizontal DigitField Screen 1	O2 Parameter Window	Extended	NORMAL
27	Horizontal DigitField Screen 1	Paw MVTV	Extended	NORMAL
28	Horizontal DigitField Screen 1	EEG Numeric	Extended	NORMAL
29	Horizontal DigitField Screen 1	Temp DigitField	Extended	NORMAL
30	Horizontal DigitField Screen 1	EEG Graphical	Extended	NORMAL
31	Horizontal DigitField Screen 1	NIBP DigitField	Extended	NORMAL

Figure 4.5: Database table for Menu Information

condition_id	condition_index	condition_text	condition_type	active
1	1	MainMenuScr1	Single	1
2	1	MainMenuScr2	Single	1
3	1	Vertical DigitField Screen 1	Single	1
4	1	Vertical DigitField Screen 2	Single	1
5	1	Horizontal DigitField Screen 1	Single	1
6	1,0	Admit/Discharge,Tab_Standby	Multiple	0
7	1,0	Admit/Discharge,Tab_Load Patient	Multiple	0
8	1,0	Sisään-/Uloskirjaus,Tab_Standby	Multiple	1
9	1,0	Continue,Continue Current	Multiple	1
10	1,0	Continue,Poista PDM-tiedot	Multiple	1
11	1,0	Patient,Réinitialiser le cas	Multiple	1
12	1,0	Patient,Discharge Patient	Multiple	0
13	1,0	Patient,Uloskirjaa potilas	Multiple	1
14	1,0	Patient,Reset Case	Multiple	1
15	1	PatientDataAreaScr1	Single	1
16	1	B2BView	Single	1

Figure 4.6: Database table for Condition

```
6758|2015-May-19 21:48:01|<USR>|NIBP Start:MainMenuScr1|ServiceDataCtrl.cpp:338|GUI
```

Figure 4.7: Log entry

Look at the message in Figure 4.7 to understand how to add condition. Figure 4.7 shows the log entry for USR type message. The highlighted message is parsed which is "NIBP Start:MainMenuScr1" as shown in Figure 4.8.

In this case, we know that all the messages which has "MainMenuScr1" keyword are the message from the parameter window of carescape patient monitor, and thus are the main messages to start the new activity or process. So, this is added into our condition list. Now look at the message again, the message must further split on the bases of ":" sign to get the exact MainmenuScr1 keyword. If string is split, the indexer starts with 0. So, while adding the new condition the condition Indexer is "1", the message is "MainMenuScr1".

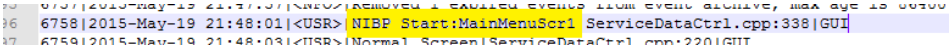

Figure 4.8: Parse log entry

Look at the messages in Figure 4.9 to understand the scenario of multiple condition. In Figure 4.9 the purple highlighted text is the log entry for USR type message. It has been checked that for some reasons the carescape patient monitor software was not logging the shortcut patient many click. This Close:Patient:Admit/Discharge is after that shortcut click which was not logged. So, it should be made as activity start as there is no other option. The message is parsed and the indexer for multiple condition as 0,1 is added as

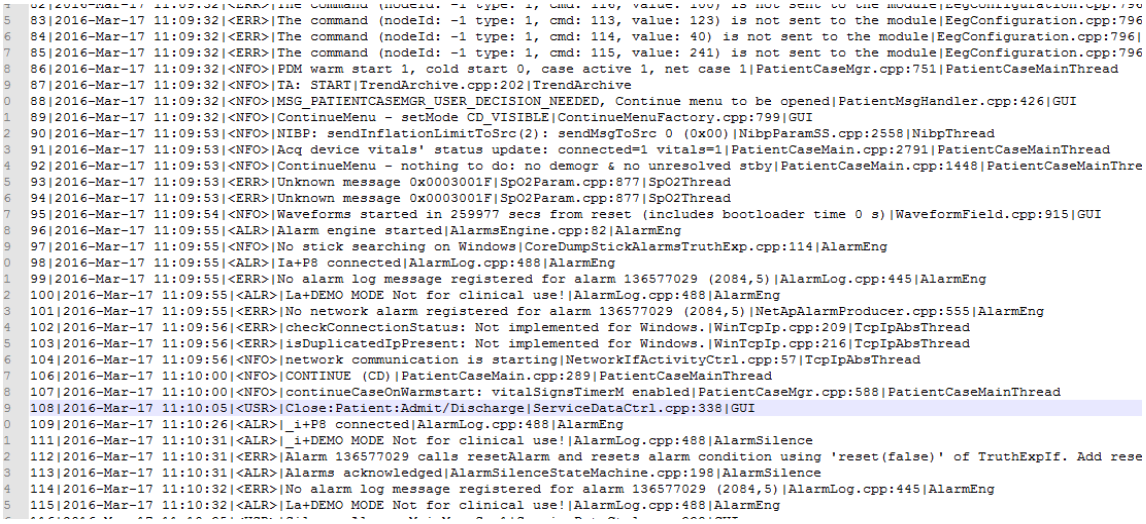

Figure 4.9: Log entry for multiple Condition

now there is a need of two AND conditions like "Where message1 = "Close" AND message2 = "Patient"". In the Condition message the messages is added with comma separated, and condition type is added as Multiple.

4.3.5 Pseudocode of Structure Algorithm

The structure algorithm was used to create the proper sequence of log data that could be used for creating the activities.

```
1   procedure make_structure(messages: array)
2   begin
3   integer structurecount, oldstruct = 0
4   for each row on the messages
5       messagesplit = split message by (':')
6       if (splitstring length > 1)
7           if (check in conditions table if message is a start process)
8               status = check_process_status(Single or Extended) from menu table
9               structurecount = structurecount + 1;
10              add structurecount and message info in tbl_structure table
11              if (status == Extended)
12                  oldstruct = structurecount
13          else
14              add oldstruct and message info in tbl_structure table
15      else
16          if ("Silence Alarms"|| "Normal Screen")
17              structurecount = structurecount + 1;
18              add structurecount and message info in tbl_structure table
19          else
20              add structurecount and message info in tbl_structure table
21  end for
22  end
```

Line 4 starts by having a loop that will traverse through all the messages. The message is then split based on delimiter colon (:) (Line 5). If the split length is greater than 1 then it is checked that if the message in consideration is a start process. In case if it's a start process, the structure order count is added with other message info into the table 'tbl_structure' otherwise algorithm jumps to line 16 where there are conditions for singleton messages. In line 11 there is a condition to check if process status is 'extended'. This can be checked from menu table so that the upcoming messages can have the same structure order count that is being maintained in variable oldstruct (Line 12).

4.4 Process Discovery Algorithm Development

Process discovery algorithm was developed to make the activity flow of the user interaction with menus. For this algorithm, structured data from the database is used after applying the structured algorithm. The data after applying the structured algorithm looks like in Figure 4.10. In Figure 4.10 structure_order field represent the process relation. The

fields with id 1 belong with the same process. Field with id 2 belongs to the other process and so on.

structure_id	case_id	data_id	structure_order	structure_date	structure_type	structure_message	structure_lineno	structure_thread
1	1	273	1	2016-Jan-12 15:13:47	P1 Parameter Window:Vertical DigitField Screen 1	<USR>	ServiceDataCtrl.cpp:338	GUI
2	1	275	1	2016-Jan-12 15:13:49	Button:Zero:Setup:P1:Invp Menu	<USR>	ServiceDataCtrl.cpp:338	GUI
3	1	276	1	2016-Jan-12 15:13:50	Close:Setup:P1:Invp Menu	<USR>	ServiceDataCtrl.cpp:338	GUI
4	1	321	2	2016-Jan-12 15:20:18	Resp DigitField:Horizontal DigitField Screen 1	<USR>	ServiceDataCtrl.cpp:338	GUI
5	1	322	2	2016-Jan-12 15:20:20	Tab_Alarms:Impedance Respiration	<USR>	ServiceDataCtrl.cpp:338	GUI
6	1	323	2	2016-Jan-12 15:20:21	Alarm On/Off Resp Rate (Impedance):Alarms:Impedance...	<USR>	ServiceDataCtrl.cpp:342	GUI
7	1	326	2	2016-Jan-12 15:20:22	Alarm On/Off Resp Rate (Impedance):Alarms:Impedance...	<USR>	ServiceDataCtrl.cpp:342	GUI

Figure 4.10: Structured data

For process discovery algorithm, a table was made to add the activities with their occurrences and time. The message occurrences were calculated, its incoming and outgoing links and time taken on that activity as well. Figure 4.11 shows the table view of process discovery algorithm output. After that this table was used to view the activities in the form of activities chart in the Log Analyzer tool.

4.4.1 Pseudocode of Process Discovery Algorithm

The process discovery algorithm is used to find the activity count that how many times a user had clicked a menu so that it can be visualized by the users.

```
1   procedure make_dependency(structured messages: array)
2   begin
3   for each row on the structured messages
4       integer link = 0
5       for each orderdata on the structured messages where structure_order = row
6           check if message already exists in tbl_activities, save activity_link also
7           if (message exist)
8               check if the link = activity_link
9               if link is same
10                  calculate activity_time
11                  increment the activity_count
12                  update this information in tbl_activities to the same row
13              else
14                  add new row in tbl_activities with activity_time = 0, activity_count = 1
15          else
16                  add new row in tbl_activities with activity_time = 0, activity_count = 1
17      end for
18  end for
19  end
```

Line 3 starts with a loop that traverses through the tbl_structure. In line 5 the rows are

taken from tbl_structure that have the structure_order equal to the row number in consideration. After that in Line 6 and 7 it is checked if the message already exists in the tbl_activities table, if not algorithm jump to line16 where a new row is added in table tbl_activities, otherwise algorithm move to line 9 where algorithm checks if the current link is same as found in the table, if yes then update it with required information otherwise create a new row table tbl_activities.

activity_id	case_id	case_activity_id	activity_message	activity_link	activity_count	activity_time
1	1	1	P1 Parameter Window:Vertical DigitField Screen 1	0	8	0
2	1	2	Button:Zero:Setup:P1:Invp Menu	1	1	2000
3	1	3	Close:Setup:P1:Invp Menu	2	1	1000
4	1	4	Resp DigitField:Horizontal DigitField Screen 1	0	6	0
5	1	5	Tab_Alarms:Impedance Respiration	4	3	11000
6	1	6	Alarm On/Off Resp Rate (Impedance):Alarms:Impedance...	5	1	1000
7	1	7	Alarm On/Off Resp Rate (Impedance):Alarms:Impedance...	6	1	1000
8	1	8	Normal Screen:MainMenuScr1	0	6	0

Figure 4.11: Process discovery algorithm output

4.5 Database Structure Development

Structure and tables information of the database is shown in this section in detail.

4.5.1 tblcases

This table is used to name the clinical log files in the form of cases. While importing any new file into the database there is a need to create the case name into the table to represent the file. Figure 4.12 shows the structure of the table. In the figure case_id represent as the primary key of the table. case_name is used to save the name of the case and case_date represent the date when the case is added.

#	Name	Datatype	Length/Set	Allow NULL	Default
1	case_id	INT	11	☐	AUTO_INCREMENT
2	case_name	VARCHAR	500	☑	NULL
3	case_date	VARCHAR	500	☑	NULL

Figure 4.12: tblcases structure

4.5.2 tbldata

This table is used to save the raw data of a clinical log file which is imported under the case. Figure 4.13 shows the structure of the table. In the figure data_id represent as the primary key of the table. case_id is the foreign key of the case, where the data belongs. log_id is the unique id that is generated in the clinical log file and we save them as it is

from the clinical log file. Log_date comes from the clinical log file which is the date and time of the log message. log_type comes from the clinical log files which represent which type of log entry is that. log_message contains the message. log_lineno represent the line no of code from where the message is generated. log_thread represent the thread of the log message.

#	Name	Datatype	Length/Set	Allow NULL	Default
1	**data_id**	**INT**	**11**	☐	**AUTO_INCREMENT**
2	case_id	INT	11	☑	NULL
3	log_id	VARCHAR	1000	☑	NULL
4	log_date	VARCHAR	1000	☑	NULL
5	log_type	VARCHAR	1000	☑	NULL
6	log_message	VARCHAR	1000	☑	NULL
7	log_lineno	VARCHAR	1000	☑	NULL
8	log_thread	VARCHAR	1000	☑	NULL

Figure 4.13: tbldata structure

4.5.3 tblstructureddata

This table is used to save the structured data from the raw data taken into tbldata table, and by applying the structure algorithm. Figure 4.14 shows the structure of the table. In the figure structure_id represent as the primary key of the table. case_id and data_id are foreign keys taken from tblcases and tbldata respectively. Structure_order represent the order number of the processes. Structure_date is same as log_date in the tbldata. structure_type is same as log_type, structure _message is same as log_message, structure_lineno is same as log_lineno and structure_thread is same as log_thread.

#	Name	Datatype	Length/Set	Allow NULL	Default
1	**structure_id**	**INT**	**11**	☐	**AUTO_INCREMENT**
2	case_id	INT	11	☑	NULL
3	data_id	INT	11	☑	NULL
4	structure_order	INT	11	☑	NULL
5	structure_date	VARCHAR	500	☑	NULL
6	structure_type	VARCHAR	500	☑	NULL
7	structure_message	VARCHAR	500	☑	NULL
8	structure_lineno	VARCHAR	500	☑	NULL
9	structure_thread	VARCHAR	500	☑	NULL

Figure 4.14: tblstructureddata structure

4.5.4 tblactivities

This table is used to save data after applying the discovery process algorithm on to the structured data. Figure 4.15 shows the structure of the table. In the figure activity_id represent as the primary key of the table. case_id is the foreign key from tblcases. case_activity_id represent the unique order key within the case. activity_message represent the

log message generated into the clinical log file. activity_link represent the link between the other activity message. activity_count represent the no of occurrence of that activity message and activity_time represent the time taken by that activity message.

#	Name	Datatype	Length/Set	Allow NULL	Default
1	**activity_id**	**INT**	**11**	☐	**AUTO_INCREMENT**
2	case_id	INT	11	☑	NULL
3	case_activity_id	INT	11	☑	NULL
4	activity_message	VARCHAR	500	☑	NULL
5	activity_link	VARCHAR	50	☑	NULL
6	activity_count	VARCHAR	500	☑	NULL
7	activity_time	VARCHAR	500	☑	NULL

Figure 4.15: tblactivities structure

4.5.5 tblmenuinfo

This table is used to add information about the menus in carescape patient monitors. Figure 4.16 shows the structure of the table. In the figure menuinfo_id represent as the primary key of the table. menu_type represent the name of the menu in clinical log file. menu_name represent the name of the menu shown in carescape patient monitor display. menu_status shows whether the menu is Single or Extended. type column shows that whether this entry in the table is normal or exceptional.

#	Name	Datatype	Length/Set	Allow NULL	Default
1	**menuinfo_id**	**INT**	**11**	☐	**AUTO_INCREMENT**
2	menu_type	VARCHAR	500	☑	NULL
3	menu_name	VARCHAR	500	☑	NULL
4	menu_status	VARCHAR	500	☑	NULL
5	type	VARCHAR	500	☑	NULL

Figure 4.16: tblmenuinfo structure

4.5.6 tblconditions

This table is used to apply the structure algorithm condition which shows when there a

#	Name	Datatype	Length/Set	Allow NULL	Default
1	**condition_id**	**INT**	**11**	☐	**AUTO_INCREMENT**
2	condition_index	VARCHAR	500	☑	NULL
3	condition_text	VARCHAR	1000	☑	NULL
4	condition_type	VARCHAR	500	☑	NULL
5	active	INT	11	☑	NULL

Figure 4.17: tblconditions structure

process start and end. Figure 4.17 shows the structure of the table. In the figure condition_id represent as the primary key of the table. condition_index represent the index position of the condition. condition_text represent the text of the condition. condition_type represent that whether the condition is single of multiple and active is used to make condition active or inactive.

4.6 Log Analyzer Tool Features

As this research thesis followed the design science research approach and the artefact was in the form of tool. A tool was developed to fulfils the requirements of GE Healthcare and to solve the research problem. Name of the tool is “Log Analyzer”. The structure algorithm and process discovery algorithm were implemented. The Log Analyzer tool's features are discussed in detail below.

4.6.1 Import File

The main idea of import file view is to add the log files data into the database table. As a database has created to automate the analysing of log files data, therefore there is a need of a view in the tool which import the log files. Also, the users wanted a view that can be used to import raw data (log files) into the system with a unique name. For this purpose, a view is created with options to add a case name (unique) first, then adding a single file

Figure 4.18: Import File view

to the selected case. There is also an option to add multiple files at the same time with on click. This is done by using the bulk insertion feature in the tool. It is used when several files need to be imported under one case. Figure 4.18 shows the view of import file.

4.6.2 Data Statistics

The user wanted to have a view where he/she would be able to visualize the statistics of a log file which is imported in the case based on the “Message type” and “Date”. The key thing about this view is that it shows the overall statistics of data. With this view the users have the idea what is inside the log files. Data statistics view is categorized into two types which are discussed below.

4.6.2.1 Data Statistics based on Message Type

Figure 4.19 shows the data statistics view based on message type. From this view, user can view and analyse the data per the message type of the selected case. User can see that in the clinical log file how many user type messages are generated, and how many errors are there and vice versa. This will give an overall view of the data. User can go further to view the data count of selected message by clicking on the desired bar of message type.

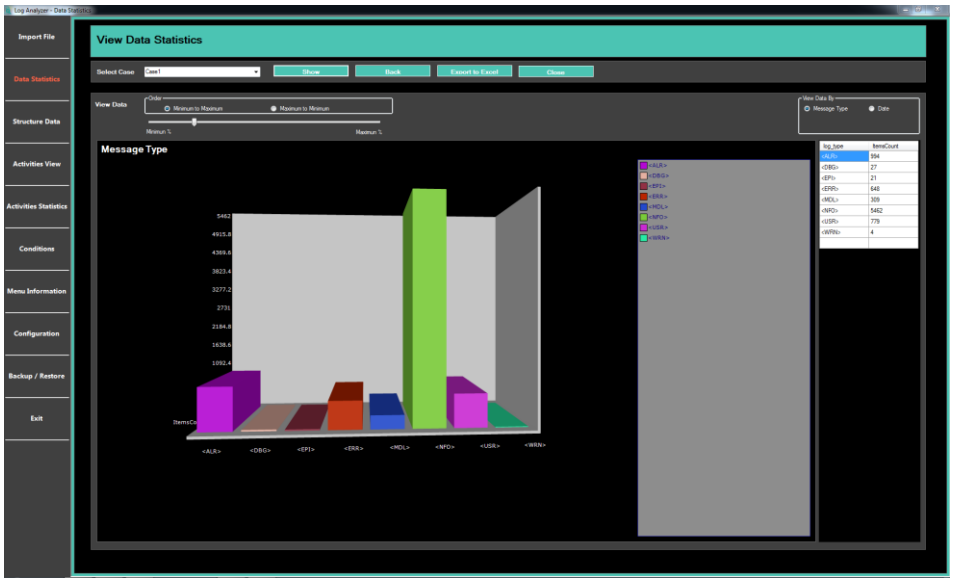

Figure 4.19: Data statistics view base on message type

When user click on any bar of message type it will give another view of specific message type and give further details of messages and their count. For example, if user click on

user type messages it will give the view displaying the user messages and its count. Figure 4.20 represent the further details of the message type statistics. This behaviour was intended by the user so that they can easily visualize the count of log that they are obtaining for the different message types and then to have a further drill down to a certain message type to see further details.

Other than that, the user wanted to have a filter where they would be able to filter the data set based on the criteria of maximum or minimum count. For example, user can use the filter bar to show only top 10 highest count or top 10 lowest count messages. Minimum to maximum and maximum to minimum radio button represents to show data sorting. There is a track bar which is used to manage the number of messages to show.

Just like the visual form, the user also wanted to have the data in tabular form and for this reason the same information is shown in a tabular format that can be seen on the right-side panel of Figure 4.20. With the help of this user would be able to easily find information.

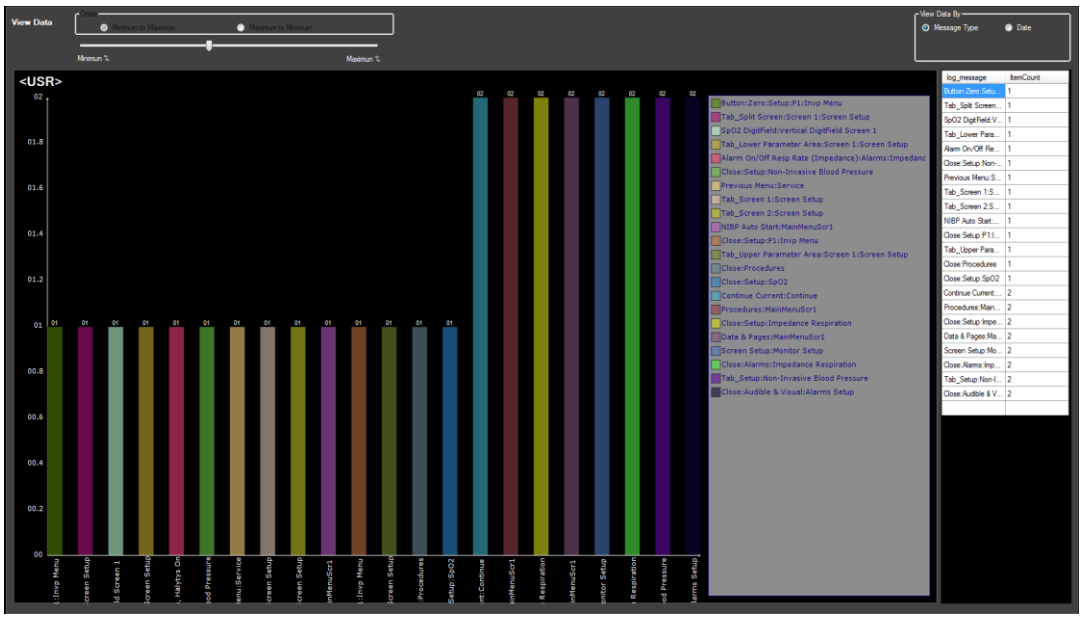

Figure 4.20: Message type view next level messages

4.6.2.2 Data Statistics based on Date

The other view that user wanted was the statistics based on the date. Figure 4.21 shows the data statistics view based on date. From this view, user can view and analyse the data based on date of the selected case. The same view could then be drilled down to a specific date as the user wanted to see the view with analysis based on a single date. Further details

are message type's occurrence on the selected dates, moreover it will provide the time of the messages and then following is messages as in the message type-based view.

Thus, if user click on the specific date slice in the pie chart, user will be redirected to the bar chart view with message types occurred on specific date. See Appendix C for figures which represent the message type bar chart with the date filter and the views that users intended to have.

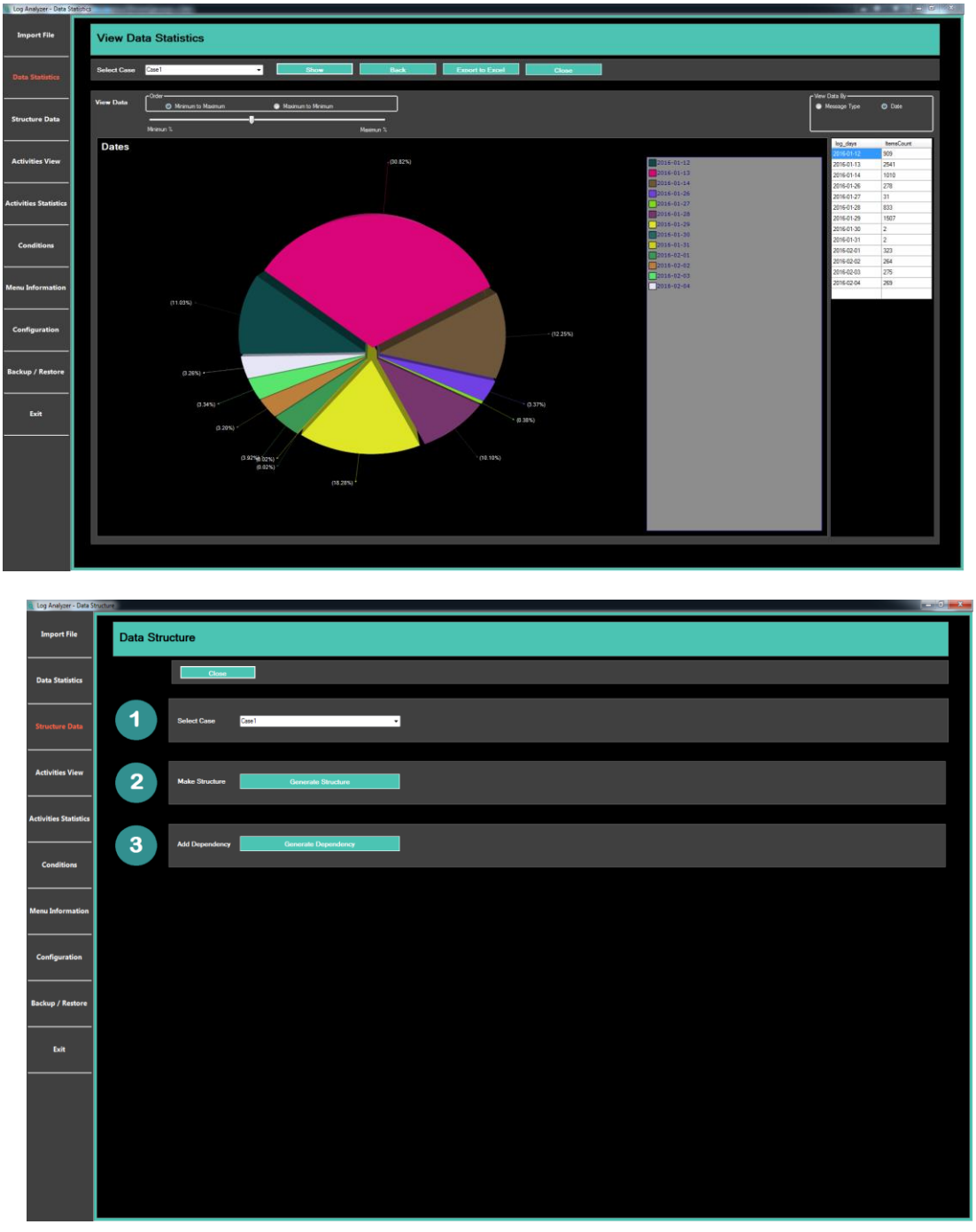

Figure 4.22: Structure data view

4.6.3 Structure Data

An activity is need for the tool that has the information of date, from where it is generated and duration. For activity creation, there is a need to run the algorithms which are structure algorithm that make the structure on the log file and process discovery algorithm that will discover the process in the log file. for this purpose, structure data view is created to apply both the structure algorithm and process discovery algorithm on the raw log data imported under a case. First, structure algorithm will run on the raw clinical log file after that process discovery algorithm will run on the structured data. Figure 4.22 shows the structure data view.

4.6.4 Activities View

All the data imported in the log file is in tabular form. It is a good idea to represent the data in a graphical form to make it more readable and understandable. The user also wanted to have a view that can show the user activities in the form of processes extracted from the clinical log file. For this purpose, activities view is created. Figure 4.23 shows the activities view in the tool.

This view has many features. From this the data can be shown as per frequency, which means that activities show the number of occurrences between the links. Figure 4.24 shows the activities with frequency.

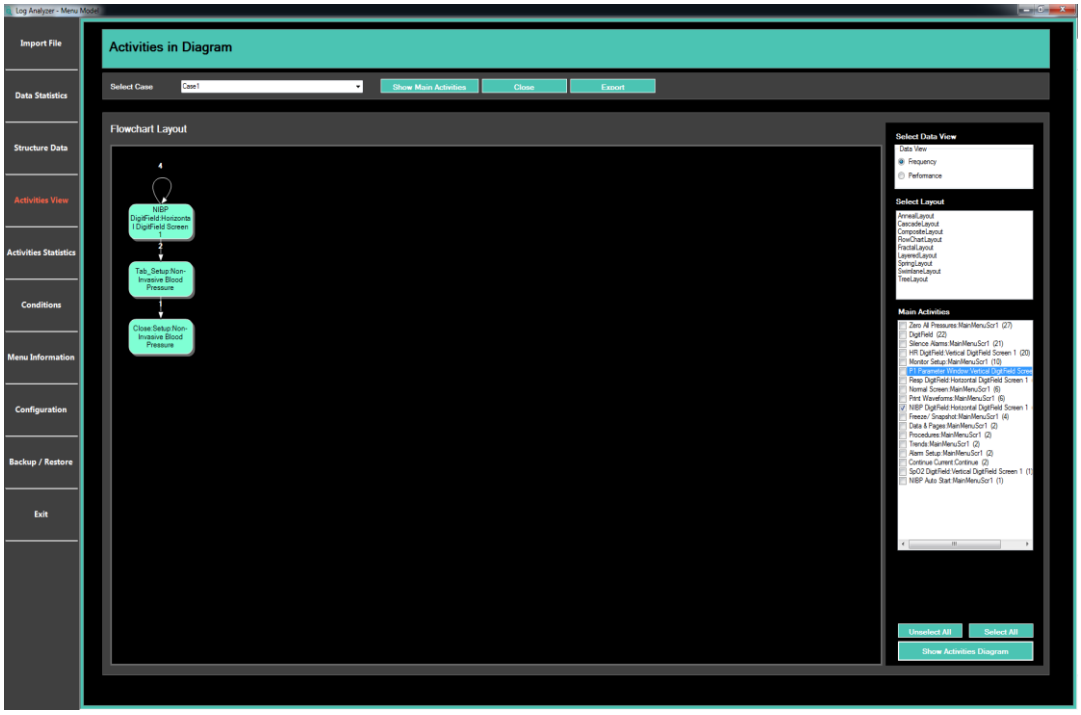

Figure 4.23: Activities view

The data can also be shown as per Performance, which means that activities shows the total time spent between the corresponding activities. Figure 4.25 shows the activities with frequency.

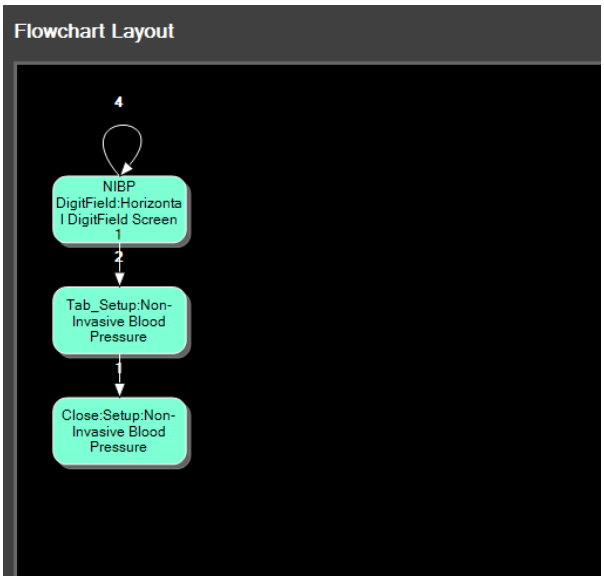

Figure 4.24: Activities view with frequency

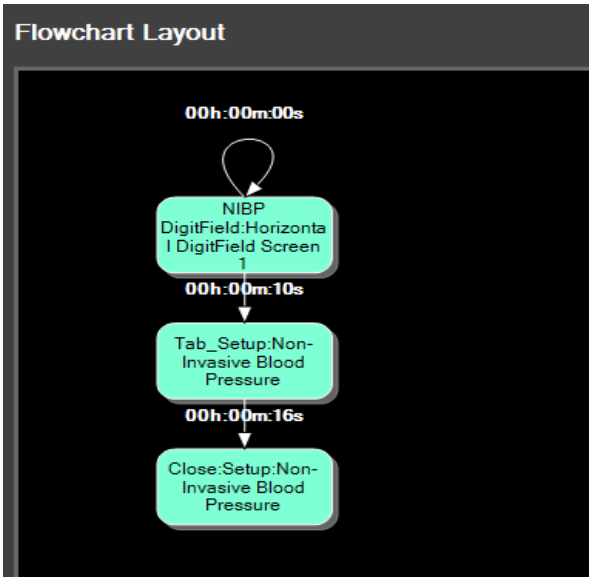

Figure 4.25: Activities view with performance

User can jump to the raw log data to see the other messages with different message type other than USR type messages by clicking on the specific activity. The process discovery algorithm is only applied on the USR type messages, so sometimes there is a need to see the other messages with that specific activity to understand the scenario. Figure 4.26 shows the data lookup with highlighted messages.

There is also a feature to export the activity diagram to specific location in the form of pdf file for further use and analysis.

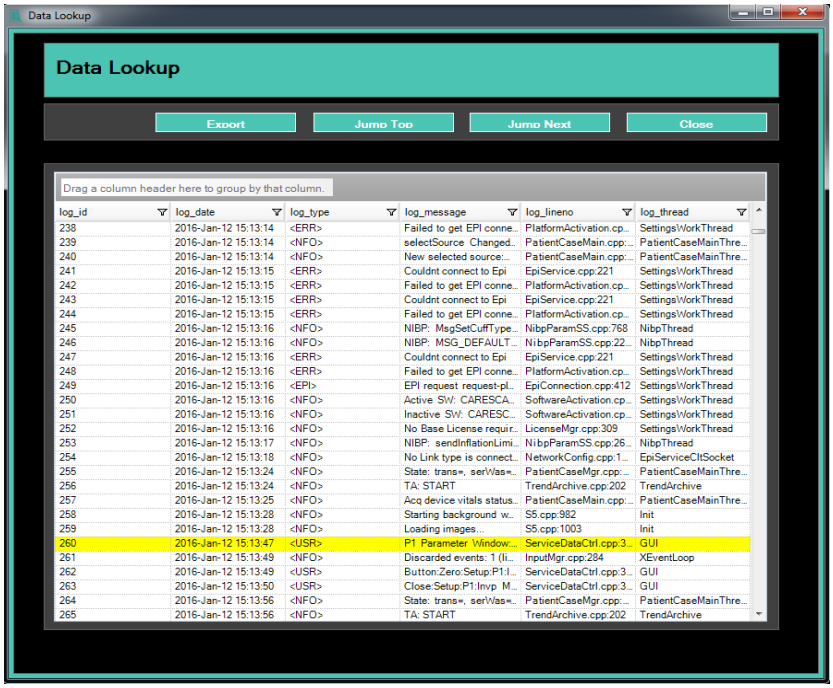

Figure 4.26: Activities view data lookup

4.6.5 Activities Statictics

Just like the activities view there is a need for a view where user show the activity statistics in the form of charts and numbers. The activity view provides the chart in the form of processes but cannot show that which menu is used more frequently, or which is used less. For this reason, activity statistics view is created. In this view activities are shown according to the case and the activities name and its occurrence count is seen in the chart. The data can also be shown in tabular form on the right-side panel. Figure 4.27 shows the activities statistics view and Figure 4.28 shows sub activities. The view is used to see specifically which feature or activity has more occurrence, and which is not used at all. The user can also export the data in excel for further analysis. There is also a limitation of not seeing the very long messages names. Hence, for this reason the export functionality is used to see the long messages.

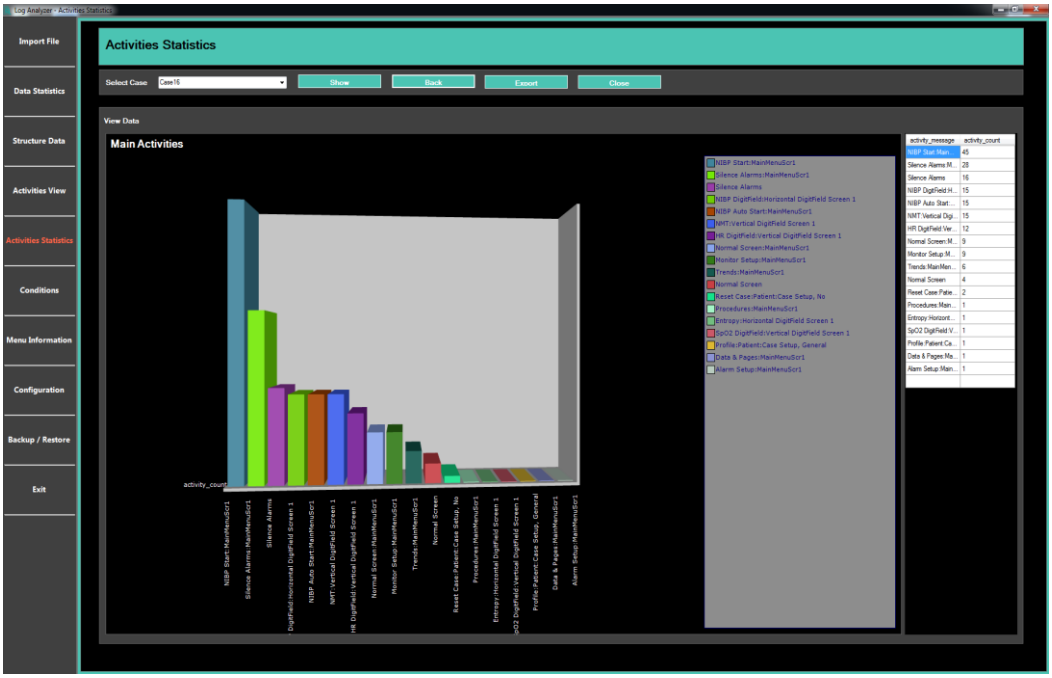

Figure 4.27: Activities statistics view

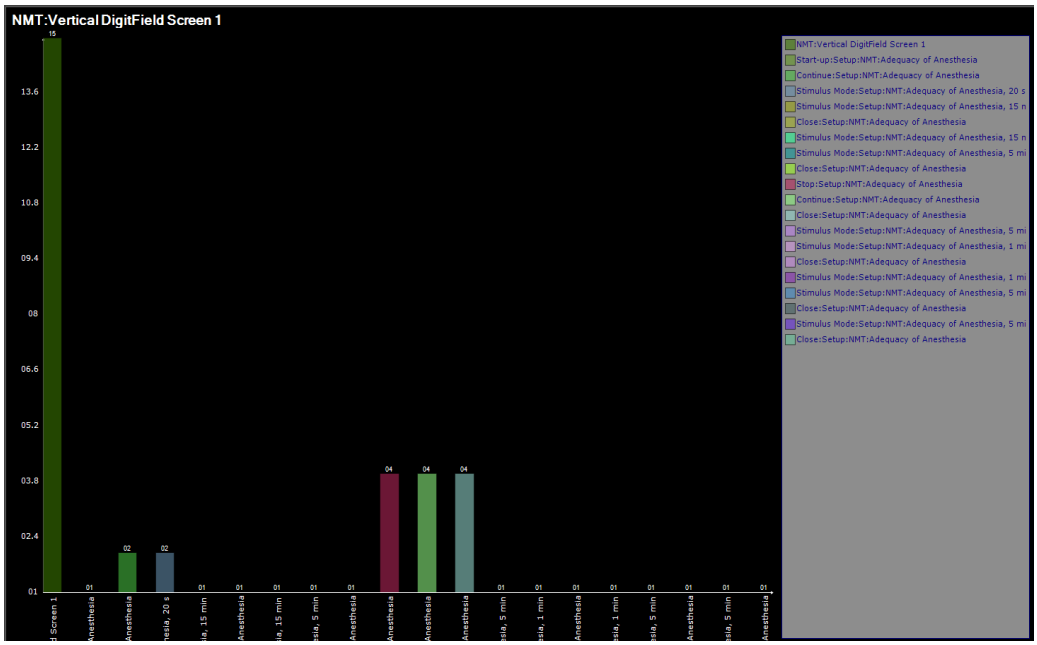

Figure 4.28: Activities statistics view with sub activities

4.6.6 Condition

As discussed in section 4.3.4 there are some exception where the carescape patient monitor is not generating the log message in the log files. For that purpose, a condition mechanism is created where these types of exceptions are added into the database to let the algorithm know that this is an exception scenario and process start or end is calculated. Hence there is also a need for a view where these conditions can be added as the proposal was to make the tool dynamic. In order to do this a view is created with the name condition to add all the exception conditions. The writing or the condition is discussed in detail in section 4.3.4. Figure 4.29 shows the condition view.

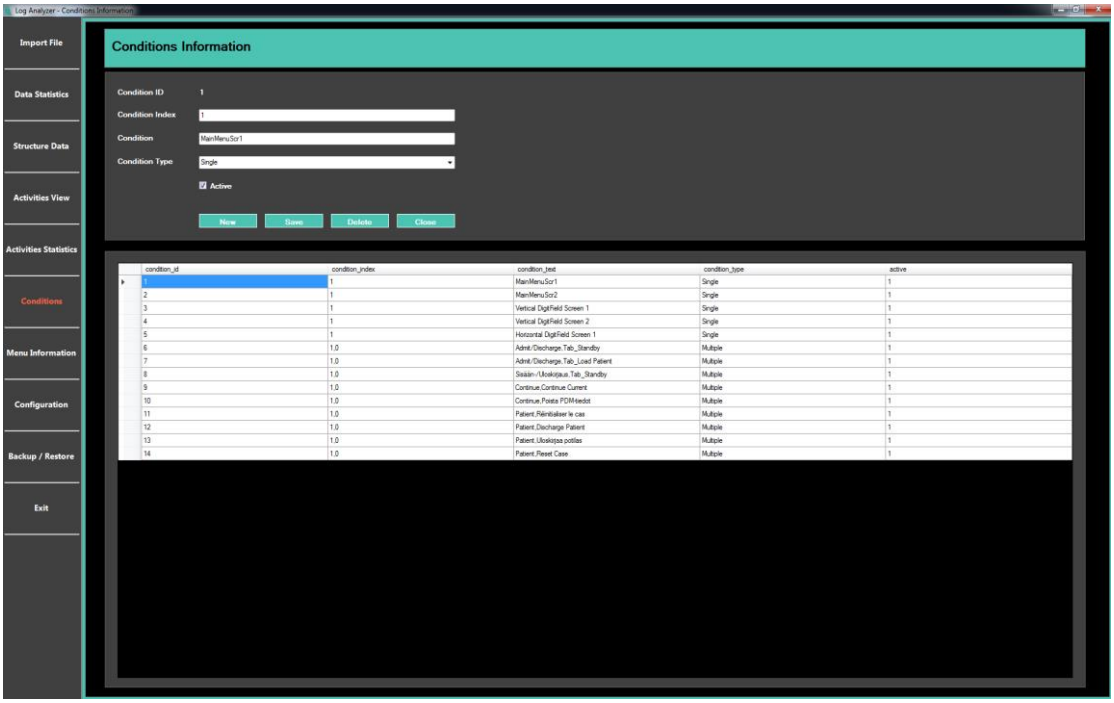

Figure 4.29: Conditions view

4.6.7 Menu Information

As proposed the tool should be dynamic, it means if there are any changes in the carescape patient monitor menu the tool should work. For this purpose, there is a need of a view where user can add or update menu information which is used in the structure and process discovery algorithm. This menu view is serving this purpose. It is used to add the menu information of the carescape patient monitor. This menu information will provide the functionality to structure algorithm about menu is single or extended. Figure 4.30 shows the menu information view.

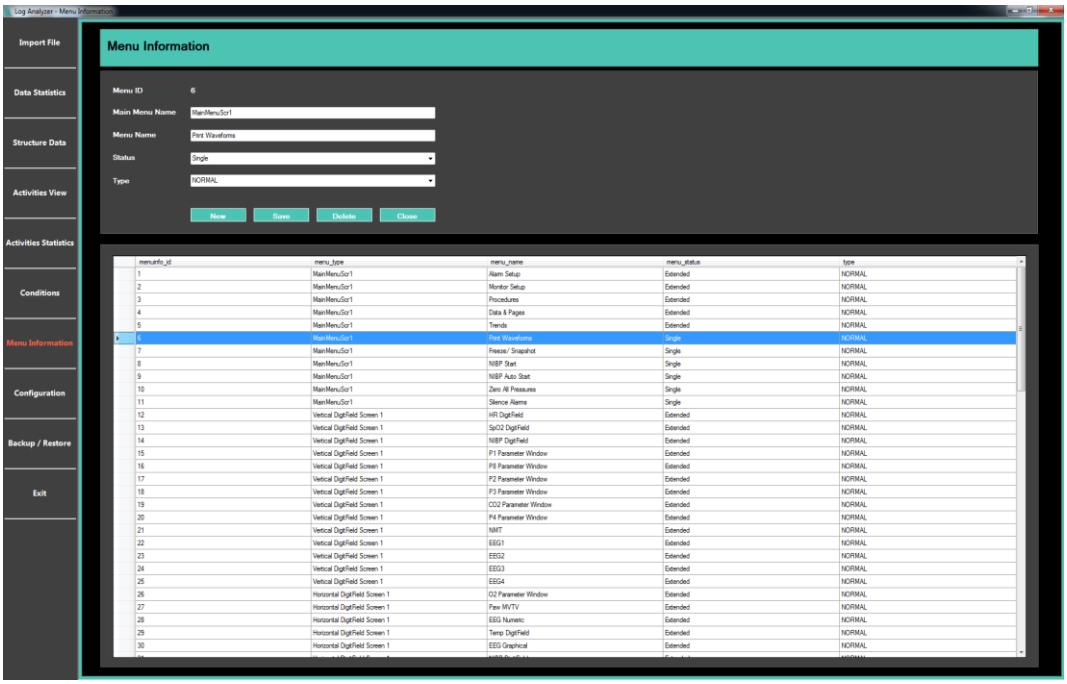

Figure 4.30: Menu information view

4.6.8 Configuration

As a database is used with this tool to store information, there is a need of view where the

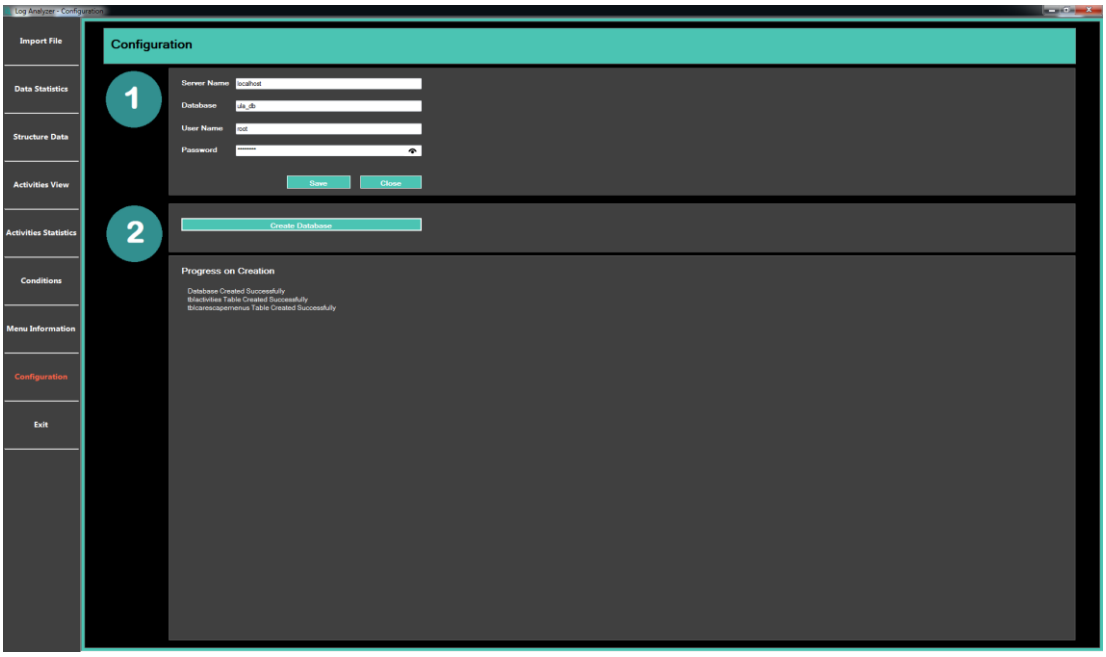

Figure 4.31: Configuration view

credential of the local MySQL database is stored to use the tool. For this reason, configuration view is created. Figure 4.31 shows the configuration view. In this view, user can add the credential of the database and create the database structure when using the tool very first time.

4.6.9 Backup & Restore Database

In order to protect the work performed in the tool and to avoid any loss of data there is always a need of backup and restore functionality. The backup and restore database view are provided to fulfils this purpose. The backup of database is stored in the local hard drive and restored from that drive. Figure 4.32 shows the backup & restore database view.

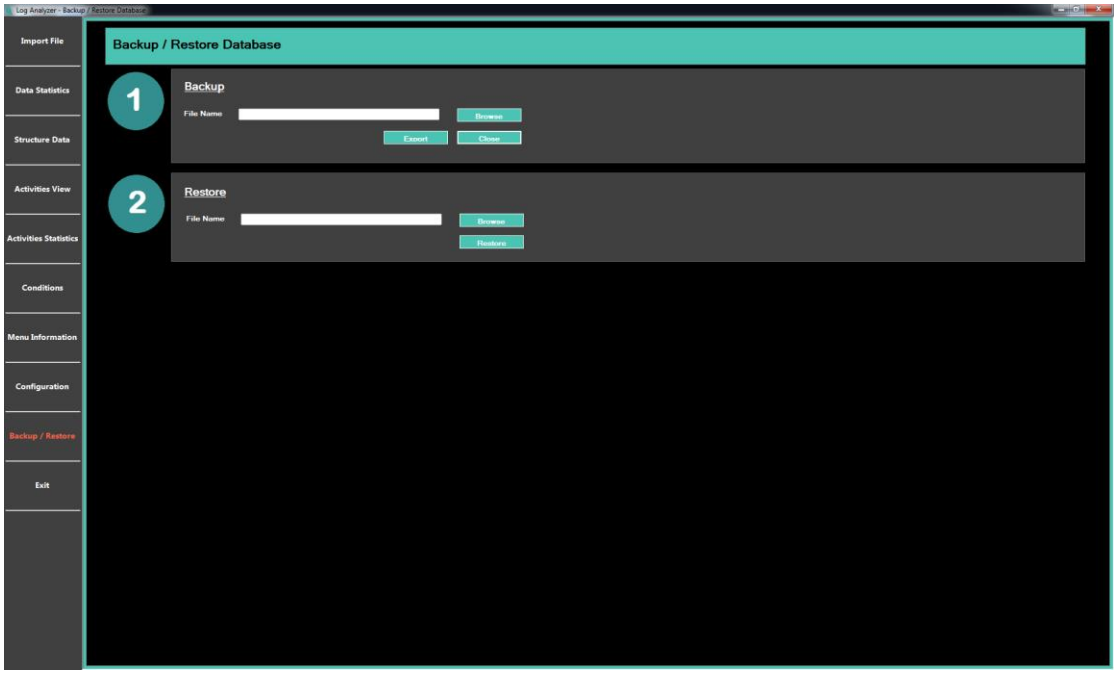

Figure 4.32: Backup and Restore Database view

4.6.10 Guide for using the Tool

The help file on how to use the Log Analyzer tool has also beeen created for the end users. It has all the informatuon on how to use the views and analyze the data. The screenshots of help file can bee seen in Appendix E.

5. Evaluation

Following the description and explanation of the experimental results, this chapter aims at evaluating and analysing the usefulness of the Log Analyzer tool implemented to obtain process discovery activities in the form of activities chart.

5.1 Settings

For testing Log Analyzer, carescape patient monitor software version 2 was used. All the information about menu and condition was added after looking at the event log files. The configuration of menu information is done using the version 2. In that version, there was some areas in the parameter window menu where the entry is not logged and in some cases the pattern of the log entry message was not correct as per the pattern other messages are generated. For those cases the exceptions menu is used to enter information into the database and use the multiple condition to make the process discovery algorithm output more accurate.

5.2 Experimental Results

Log Analyzer tool was tested with 20 different clinical log cases. All of them were the real data got from the hospitals. Before adding the cases into the tool, there was some exceptions in the log files that should be fixed before applying the algorithm. The exceptions are described in the next section. They were handled by adding some conditions and fixing them into the algorithm and tables to make the activity diagram as accurate as possible.

5.2.1 Exceptions

The exceptions are divided into three categories which are discussed below.

5.2.1.1 Menus with no log entry

There were some menus in parameter window of carescape patient monitor software, where the event entry was not logged in event log file. So, in this case there was an exception that it's possible that the sub activities may come under the previous process, because the main process entry of those activities were missing.

For fixing this, the possible sub activities was added that can come after the main process

activity in the condition under exception. With this condition, the accurate results were generated but eventually this would be fixed in future in the carescape patient monitor software to make the algorithm more generic.

5.2.1.2 Menus with wrong pattern messages

There were some menus in the carescape patient monitor software, which were recording the wrong patterned log messages. For example, for the main activity or process there is a pattern of "ActivityName: ActivityMainArea". If the message is split, it is known that the main activity message length is always greater than 1. Same goes with sub activity which have length greater than 2. But in some menus, the main activity message length is only 1. This would cause the algorithm to skip the message and hence there were not correct results of the activities.

To fix this, exceptional conditions in the algorithm were added. Those were the fixed conditions and were hard coded.

5.2.1.3 Menus with same message in log file

There were some menus in the carescape patient monitor software, which were recording the same message into the event log file. For example, for one main menu it is recoding "DigitField:MainMenu" and another main activity it is recording the same message. In this case, there is a possibility that the sub activities of these main processes will be merged, and which is not correct. Unfortunately, it cannot be fixed but this will be fixed in the carescape patient monitor software next version to make the algorithm work correctly.

5.2.2 Results

Below have the results of structured and discovery algorithm used in the Log Analyzer tool. Some of the cases have described here.

5.2.2.1 Case 1

After adding the clinical log file in case 1, data statistics view is used to see what type of messages were there and count of them. Figure 5.1 shows the data statistics. The clinical log file of case 1 consist of various message type. These are <ALR> type messages which has 994 records. <DBG> type messages have 27 records, <EPI> type messages have 21

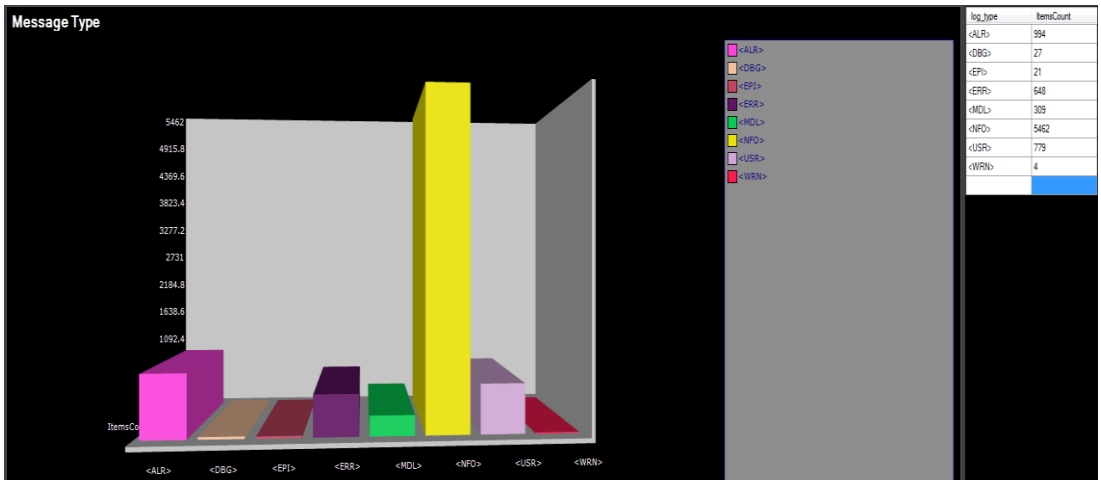

Figure 5.1: Case 1 data statistics

records, <ERR> type messages have 648 records, <MDL> type messages have 309 records, <NFO> type messages have 5462 records, <USR> type messages have 779 records, and <WRN> type messages have 4 records.

For structured algorithm and process discovery algorithm Log Analyzer tool choose the <USR> type messages. Figure 5.2 shows the further details of <USR> type messages. Let's choose the order of the messages from maximum to minimum and 50% of records to show from the trackbar.

Figure 5.2: Data statistics for user type messages

Figure 5.2 shows that most of the <USR> type message has "Character key" with the count of 595. Then there are 27 "Zero All Pressures:MainMenuScr1" messages , 22

“”DigitField” messages, 21 “Silence Alarms:MainMenuScr1” messages and 20 “HR DigitField:Vertical DigitField Screen 1” messages and so on.

After applying the algorithms take a look at the main activities generated by the algorithms. Figure 5.3 shows the main activities or processes for the case 1. The activities are shown with the occurrences count. User can first check the activities and analyse the data and check if there are some odd menu names exist. In this scenario, all the main activities look good.

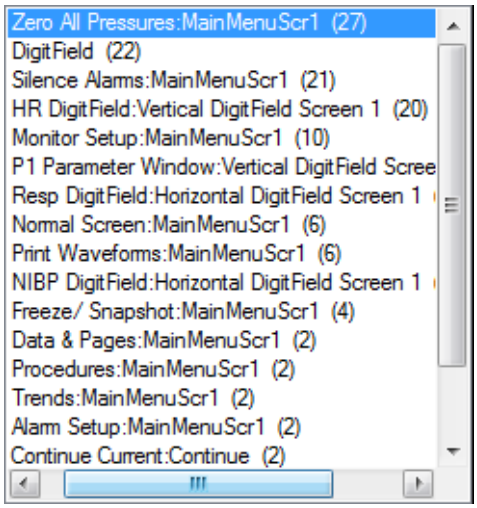

Figure 5.3: Case 1 main activities

Now let’s check the sub activities of “Monitor Setup:MainMenuScr1” which occurred 10 times in the log file with sub activities. Figure 5.4 shows the sub activities of this main menu with the Frequency Data View. The sub activities show that 10 times “Monitor Setup:MainMenuScr1” is clicked which is the main activity, in which user goes to “ScreenSetup:MonitorSetup” 2 times and “Service:MonitorSetup” 6 time. Which gives us the total of 8. But the main activity is clicked 10 times. Now there is a possibility that user click the normal screen button which redirects the user to the main screen. This is shortcut menu which is used to directly comes to the main screen of carescape patient monitor software by closing every menu. And normal screen is another single activity.

To verify this, jump to the log messages of “Monitor Setup:MainMenuScr1” and check the activities. Figure 5.5 shows the data lookup with “Monitor Setup:MainMenuScr1” log entry highlighted in it. By looking at the log files it seems that 1 time user clicked on the other main activity “DigitField” after clicking the “Monitor Setup:MainMenuScr1” and

2nd time user click on the “Normal Screen:MainMenuScr1” which is another main activity to jump directly to main screen.

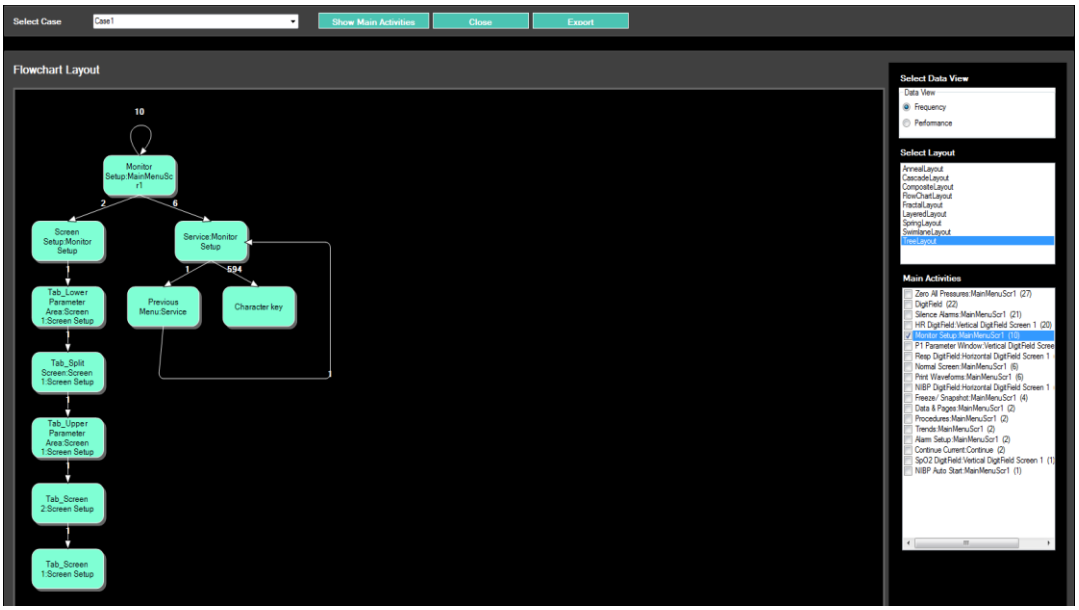

Figure 5.4: Case 1 frequency view

log_id	log_date	log_type	log_message	log_lineno	log_thread
262	2016-Jan-28 14:29:24	<ALR>	Alarms audio paused	AlarmSilenceStateM...	AlarmSilence
263	2016-Jan-28 14:29:26	<USR>	Alarm Setup:MainMenuScr1	ServiceDataCtrl.cpp:...	GUI
264	2016-Jan-28 14:29:27	<USR>	Monitor Setup:MainMenuScr1	ServiceDataCtrl.cpp:...	GUI
265	2016-Jan-28 14:30:53	<ALR>	_i+No printer selected	AlarmLog.cpp:488	AlarmEng
266	2016-Jan-28 14:31:16	<USR>	DigitField	ServiceDataCtrl.cpp:...	GUI
267	2016-Jan-28 14:31:23	<ALR>	Acknowledge/audio pause period ex...	AlarmSilenceStateM...	AlarmSilence
268	2016-Jan-28 14:32:30	<NFO>	Discarded events: 1 (limit: 2 events /...	InputMgr.cpp:284	XEventLoop
269	2016-Jan-28 14:32:30	<USR>	Alarm Setup:MainMenuScr1	ServiceDataCtrl.cpp:...	GUI
270	2016-Jan-28 14:32:31	<USR>	Monitor Setup:MainMenuScr1	ServiceDataCtrl.cpp:...	GUI
271	2016-Jan-28 14:32:31	<USR>	Normal Screen:MainMenuScr1	ServiceDataCtrl.cpp:...	GUI
272	2016-Jan-28 14:36:16	<DBG>	Memory usage: VmSize=303632 kB...	MemoryUsageLogge...	WorkThread

Figure 5.5: Case 1 data lookup

Now let’s see the same menu with Performance Data View which shows the time spent on the activities. Figure 5.6 shows the “Monitor Setup:MainMenuScr1” sub activities with Performance View. In the view, it is seen that the main activity has time show as 00h:00m:00s, the main activity doesn’t show the time because this view shows the time spent between 2 activities. By looking at the view we can see there is a wired time shown between the “Service:MonitorSetup” and “PreviousMenu:Service” which is

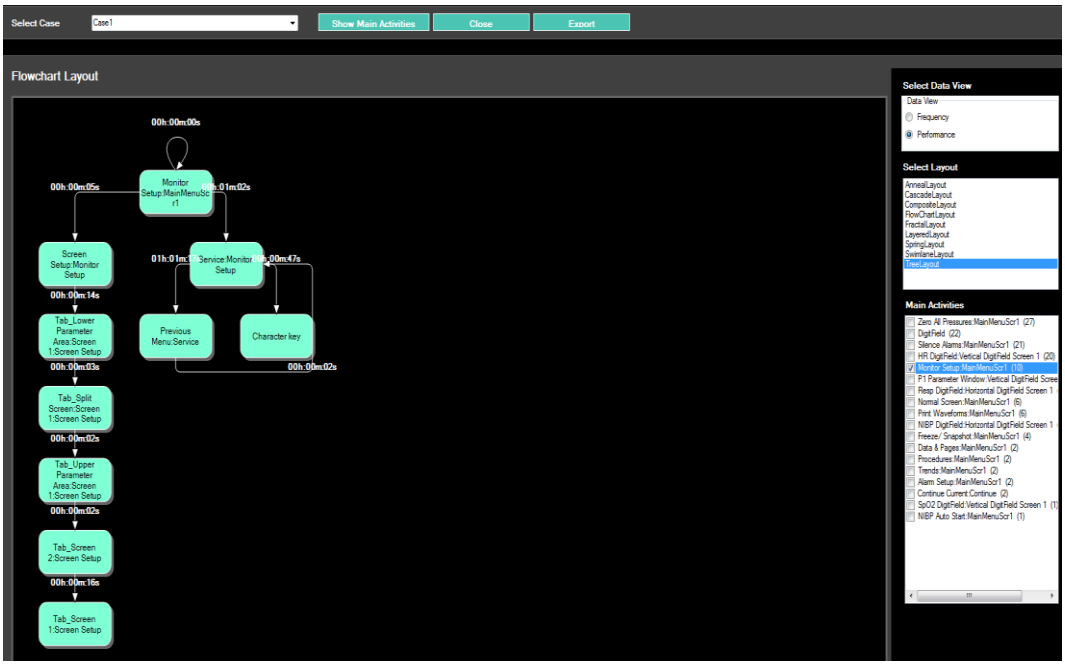

Figure 5.6: Case 1 performance view

01h:01m:17s. To verify this lets have a look at the data lookup of "Service:MonitorSetup". Figure 5.7 shows the data lookup for "Service:MonitorSetup". It is clearly seen that user have spent that much time in them by looking at the date selected with purple rectangle. The data can be imported in the pdf file. Appenxix G show that how the activities look in the pdf file.

log_id	log_date	log_type	log_message	log_lineno	log_thread
365	2016-Jan-29 08:28:12	<NFO>	Discarded events: 1 (li...	InputMgr.cpp:284	XEventLoop
366	2016-Jan-29 08:28:18	<NFO>	Mouse was found. /dev...	MouseVisibilityHandl...	GUI
367	2016-Jan-29 08:28:21	<USR>	Service:Monitor Setup	ServiceDataCtrl.cpp:3...	GUI
368	2016-Jan-29 09:29:38	<USR>	Previous Menu:Service	ServiceDataCtrl.cpp:3...	GUI

Figure 5.7: Case 1 data lookup for performance view

5.2.2.2 Case 2

More clinical log file has been added in the Log Analyzer tool under the name case 14. After importing the clinical log file, data statistics view was used to see what type of messages were there and count of them. Figure 5.8 shows the data statistics of log file. The log file of case 2 contain several message types. These are <ALR> type messages which has 12959 records. <ERR> type messages have 1080 records, <MDL> type messages have 6 records, <NFO> type messages have 843 records, and <USR> type messages have 464 records.

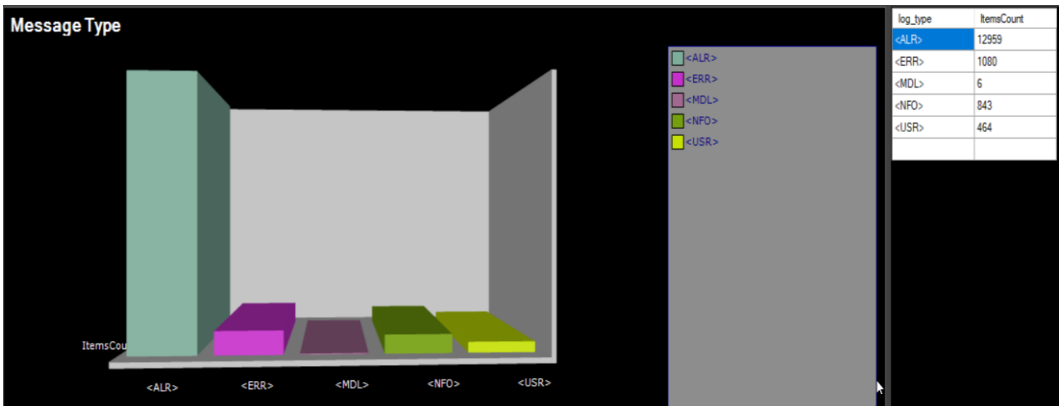

Figure 5.8: Case 2 data statistics

Figure 5.9 illustrates the additional details of <USR> type messages. Let's choose the order of the messages from maximum to minimum and 40% of records to show from the trackbar. It is seen that the majority of the <USR> type message are "Silence Alarms:MainMenuScr1" with the count of 128. Then there are 31 "Silence Alarms:MainMenuScr2" messages , 17 "Monitor Setup:MainMenuScr2" messages, 16 "Service:Monitor Setup" messages and 14 "HR DigitField:Vertical DigitField Screen 1" messages and so on.

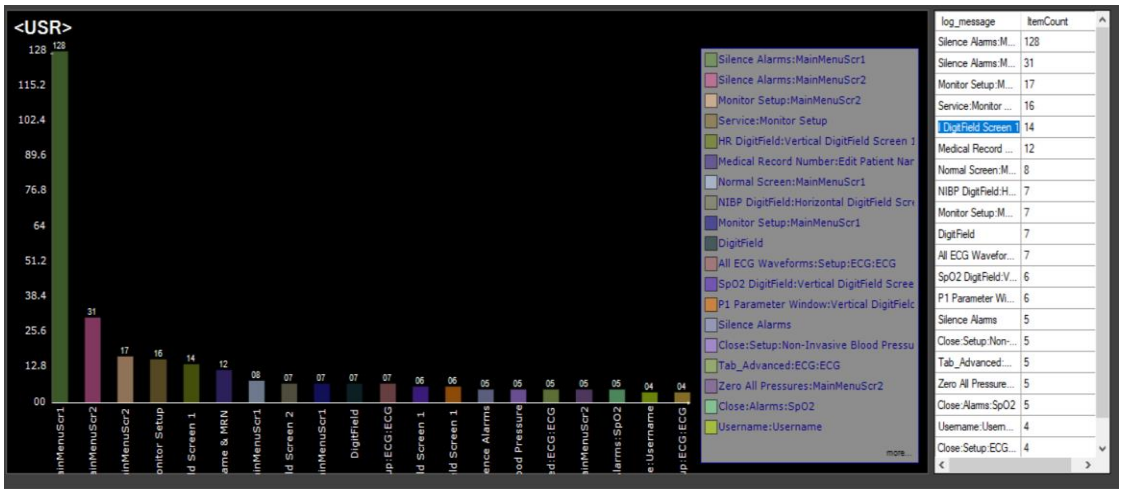

Figure 5.9: Case 2 data statistics for user type messages

Let's have a look at the major activities created by the tool after applying the algorithms. Figure 5.10 shows the main activities for case 2. The activities are presented with the count of occurrences. Next, the user should monitor the behaviour and analyse the data to test whether there is any odd menu name. All the main activities look good in this scenario. In this scenario we have main activity named "DigitField". As we have dis-

cussed in section 5.2.1.3 that there is an exception with the menu name in log file. Multiple menus are generating the same name “DigitField” in the log file. Which is an error in the carescape patient monitor and will be fixed in the next version.

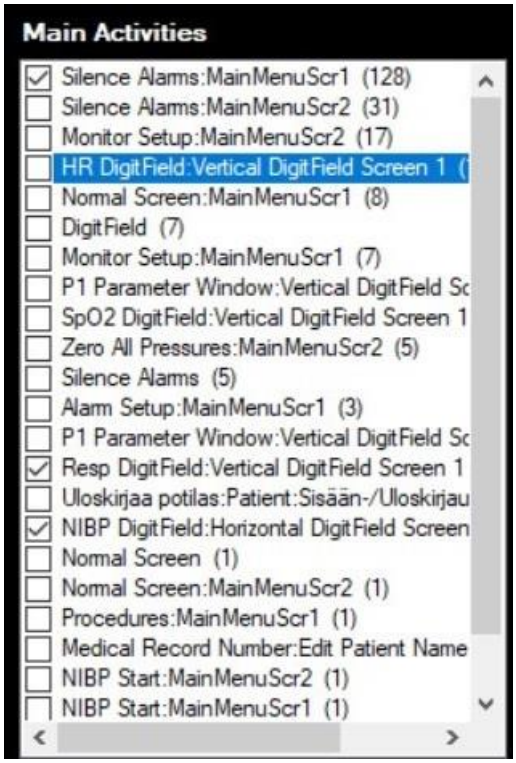

Figure 5.10: Case 2 main activities

Right now, let’s see what sub activities are presented in the “DigitField” main menu. Figure 5.11 shows the “DigitField” sub activities with frequency which occurred 7 times in the log file with sub activities. The sub activities show that 7 times “DigitField” is clicked which is the main activity, in which user goes to “SaveReference:RealTimeView:ST:ECG” once, “Close:Alarms:ST:ECG” once, “Close:Timers” once and “Tab_Setup:ST:ECG” once. It gives us a total of 5. But the main activity is clicked 7 times. Now there is a possibility that user clicked the normal screen button which redirects the user to the main screen. This is shortcut menu which is used to directly jumps to the main screen of carescape patient monitor software by closing every menu. And normal screen is another single activity.

To verify this, lets jump to the log messages of “DigitField” and check the activities. Figure 5.12 shows the data lookup with “DigitField” log entry highlighted in it. By looking at the log files it seems that 2 time user clicked on the “NormalScreen:MainMenuScr2” which is another main activity to jump directly to main screen.

Now let's see the same menu with Performance Data View which shows the time spent

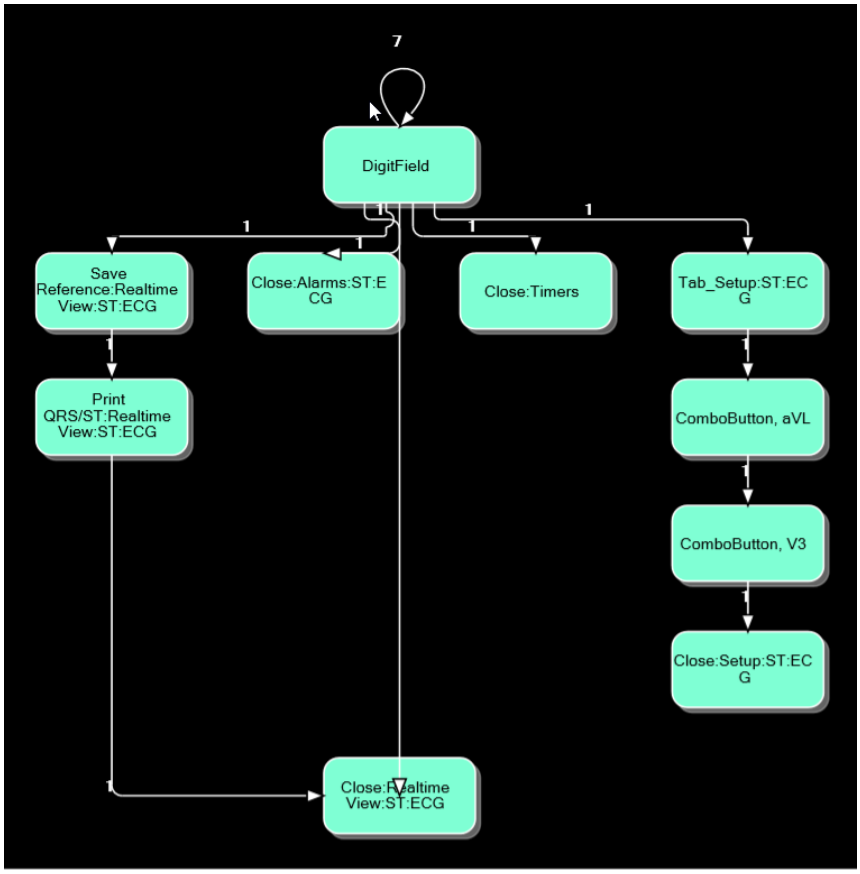

Figure 5.11: Case 2 "DigitField" frequency view

59648	2015-Dec-17 15:...	<NFO>	Discarded events: 1 (limit: 2 events / 60 m...	InputMgr.cpp:284	XEventL
59649	2015-Dec-17 15:...	<USR>	DigitField	ServiceDataCtrl.c...	GUI
59650	2015-Dec-17 15:...	<USR>	Normal Screen:MainMenuScr2	ServiceDataCtrl.c...	GUI
59651	2015-Dec-17 15:...	<USR>	Uloskirjaa potilas:Patient:Sisään-/Uloskirj...	ServiceDataCtrl.c...	GUI
59652	2015-Dec-17 15:...	<NFO>	DISCHARGE	PatientCaseMain....	PatientC
59653	2015-Dec-17 15:...	<NFO>	Request to discharge the Tram received	PdmTramRacSer...	PatientC

Figure 5.12: Case 2 data lookup for "DigitField"

on the activities. Figure 5.13 shows the "DigitField" sub activities with Performance View. In the view, it is seen that the main activity has time show as 00h:00m:00s, the main activity doesn't show the time because this view shows the time spent between 2 activities. By looking at the view it is clearly seen that the maximum time spent in the sub activities is between "DigitField" and "Close:Timers" which is 00h:07m:37s. This looks good and there is no odd timing in the subactivities.

Other data on main activities are presneted in the Appendix D.

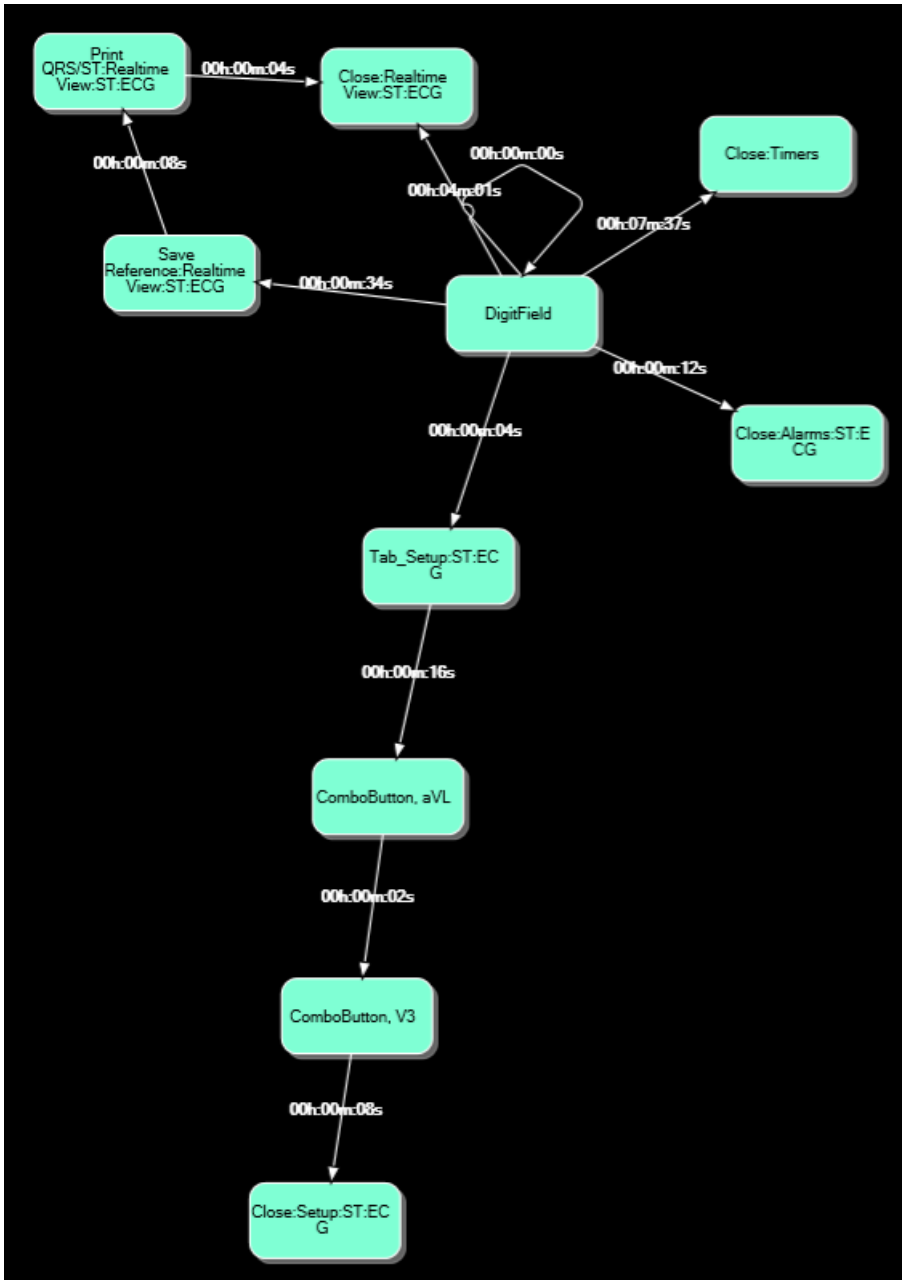

Figure 5.13: Case 2 performance view of “DigitField”

5.3 *Feedback form Usability Team*

The Log Analyzer tool was demonstrated to the usability team after the completion of the development. The end-users of the tools were mainly usability team and some other teams as well who were using the log file. The feedback collected from the usability team was as follows:

- The tool fulfils the purpose of usability analysis on log files as it has view where

the activities were seen clearly instead of a raw log file with the number of messages.

- The tool was good for inhouse usability tests for the software as well, where you can have set of desired tasks or sequence of activities.
- The tool has the ability to show you data instant and very fast. This feature also works great when using it for usability test. You can view very fast in labs instead of looking through the recorded videos.

5.4 Assessment of Research Design

Table 5.1 shows the guidelines of design science research by Hevner et al. [HMP04] and comments to assess tis thesis research under every guideline. As in Fig 3.4 the mapping of Hevner framework in thesis is presented and the comments on Table 5.1 against every guideline is according to it.

Table 5.1: Design science research guidelines for assessment by Hevner el al. [HMP04]

Evaluation Method	Description	This Thesis Work
Design an Artefact	Design-science research must produce a viable artefact in the form of a construct, a model, a method, or an instantiation.	An artefact was developed in the form of tool which has the functionality of importing the clinical log files and generating activities on which the usability analysis is done.
Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.	Usability analysis on clinical log files was the important component for GE healthcare, as their product quality and validation work through this.
Design Evaluation	The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods.	The artefact was evaluated with different real hospital files and the results were verified by checking them with the clinical log files.

Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations, and/or design methodologies.	This tool will help the GE healthcare to fill the gap in the analysis and it also gives the new researchers the path to extend the research with more functionalities.
Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact.	The theories from the literature of process mining was effectively used in the algorithms which were developed for the tool.
Design as a Search Process	The search for an effective artefact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.	The research was conducted by properly searching the methods to apply in the tool. The method used in making the algorithm was process discovery method.
Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.	The main parts of the functionality of tool was well communicated into this document. This will help the other researcher to understand the research and helps them to find new research areas in this field.

6. Discussion and Conclusion

6.1 Main Findings

The main goal of the thesis was to develop a tool for GE healthcare that could be used for usability analysis on event log files. The main purpose of the tool was to find menus that were not frequently used by staff in the carescape patient monitor, so that they can report and ask for the reasons and to find information about the errors and failures.

The first step of the thesis was to study the already designed algorithms for process mining and then analyzing the GE healthcare log file. After that the algorithms were created to apply the process mining technique in the clinical log file. For the thesis two algorithms were created, structure algorithm and process discovery algorithm.

The objective of the thesis was achieved in the form of reserach questions and how the reserach questions were answered in the thesis is described below. The answer to the research question will be provided on the basis of research results which were described in the previous chapter. First the sub research questions will be answered and after that the main research problem will be answeres.

RQ1 How process mining technique can be used to analyse healthcare log file data?

The concept of process mining was discussed in the theoratical framework section. In that section there was discussion about different process mining types and how they could be used in any organization.

It was also discussed that there is a clear difference between process mining and software process mining. Software process mining was used to analyze the deployed software behaviour. GE healthcare was using a deployed software and there were algorithms for process mining and not for software process mining. So by using the concept of process mining algorithms which were already developed new algorithm for process discovery was created.

RQ2 What are already used process mining algorithms?

There were various algorithms that were used for process mining. Mainly three of them were discussed that were commonly used. The algorithms were discussed in theoratical framework section which were alpha algorithm, fuzzy mining algorithm and heuristic

mining algorithm.

RQ3 How the algorithms can be applied in software process mining for usability analysis?

Ideas from fuzzy mining algorithm were taken into account which was basically upgraded version of alpha algorithm. The key concepts taken was to take the count of the process and then create an activity diagram just like the petri net diagram that was discussed in the fuzzy mining algorithm.

The main research problem was **"Perform the usability analysis on the data extracted from the event log files".** Two algorithms and Log Analyzer tool has developed to perform the usability analysis on clinical event log file. The tool provides the user with activity diagrams that depict the information about used menus. It also provides chart about the errors and failures in the software.

6.2 Limitations

The file size used has the size of 2 MB which works fine, but it takes more and more time depending on the size of the file for larger files.For this purpose the implementation of the algorithms can be reviwed and performance loop holes can then be tuned.

As the algorithm is taking only USR type messages, the process discovery algorithm gets slower when there are more than 3000 USR type messages in a single take. It can be optimized in future work.

It is seen in the log file that it contains a lot of messages that are not useful for the problem. The example of these kind of messages are "Shutting down", "Restarting" etc. For developing the algorithm, these types of messages are ommited and has been taken only the messages with a proper format. Hence the rest of the messages without the format is will be filtered down during the algorithm application. The format that is considered to be a proper message is with colon (':') and when they splitt with colon (':') the length should be greater than 1. For exception cases only some of the messages are considered in format, these are "Silence Alarms", "Normal Screen", "DigitField", "Button" and "Character key" that does not contain any colon (':'). The rest of the messages will be discarded.

There were some exceptions that needed to be considered. One of them is that there are some menus clicked which are not logged in the log file due to unforseen behaviour of

the system. Initially , the user will run the algorithm as he/she is supposed to do in the normal scenario and then see the activities as if there is any exception (any activity start that cannot be a possible start activity). In order to fix this problem, the user will add the appropriate activity information as an exception into the conditions table.

Process discovery algorithm can be optimized for fast processing as it was seen that it is taking more time when the size of the file gets bigger than 2 MB or when the USR messages exceeds 3000.

6.3 Conclusion

This thesis research was done using the design science research approach by Hevner et al. [HMP04]. All the reserach questions were answerd and discussed in this section. An arteface was produced in the form of Log Analyzer tool for GE Healthcare that fullfills the need of usability analysis on clinical log files. Two algorithms were developed in the tool structure algorithm and process discovery algorithm. Structure algorithm helps to make the structure of raw clinical log file and process discovery algorithm was used to get activities in the form of processes. Only USR type messages in the clinical log files were used as there was only the need of user activities evaluation. Log Analyzer tool was tested and evaluated with real clinical log files and it was working well. There were some exception cases in the log file which were handeled in the code but those will eventually corrected in the new versions of clinical log file by GE Healthcare.

.

References

A12 Aalst, Wil Van Der. Process Mining. Commun. ACM, 55(8):76–83, Aug. 2012.

A16 Aalst, Wil Van Der. Process Mining Data Science in Action. 2nd ed., Springer, 2016.

BZ03 Benbasat, I. and Zmud, R.W., 2003. The identity crisis within the IS discipline: Defining and communicating the discipline's core properties. MIS quarterly, pp.183-194.

GCW07 Günther, Christian W., and Wil MP Van Der Aalst. "Fuzzy mining–adaptive process simplification based on multi-perspective metrics." International conference on business process management. Springer, Berlin, Heidelberg, 2007.

GRWU08 Günther, C., Rozinat, A., van der Aalst, W.M.P. and van Uden, K., 2008. Monitoring deployed application usage with process mining. BPM Center Report BPM-08-11, pp.1-8.

HMP04 Hevner, S. and March, P., 2004. J., and Ram, S.," Design Science Research in Information Systems,". Management Information Systems Quarterly, 28, pp.75-105.

I07 Iivari, J., 2007. A paradigmatic analysis of information systems as a design science. Scandinavian journal of information systems, 19(2), p.5.

KGM12 Koppenhagen, N., Gaß, O. and Müller, B., 2012. Design Science Research in Action-Anatomy of Success Critical Activities for Rigor and Relevance.

MDAW04 de Medeiros, A.K.A., van Dongen, B.F., van der Aalst, W.M. and Weijters, A.J.M.M., 2004, June. Process mining for ubiquitous mobile systems: an overview and a concrete algorithm. In International Workshop on Ubiquitous Mobile Information and Collaboration Systems (pp. 151-165). Springer Berlin Heidelberg.

MS95 March, S.T. and Smith, G.F., 1995. Design and natural science research on information technology. Decision support systems, 15(4), pp.251-266.

MSSAB08 Mans, R.S., Schonenberg, M.H., Song, M., van der Aalst, W.M. and Bakker,

P.J., 2008, January. Application of process mining in healthcare–a case study in a dutch hospital. In International Joint Conference on Biomedical Engineering Systems and Technologies (pp. 425-438). Springer Berlin Heidelberg.

NJCP90 Nunamaker Jr, J.F., Chen, M. and Purdin, T.D., 1990. Systems development in information systems research. Journal of management information systems, 7(3), pp.89-106.

OB91 Orlikowski, W.J. and Baroudi, J.J., 1991. Studying information technology in organizations: Research approaches and assumptions. Information systems research, 2(1), pp.1-28.

OI01 Orlikowski, W.J. and Iacono, C.S., 2001. Research commentary: Desperately seeking the "IT" in IT research—A call to theorizing the IT artifact. Information systems research, 12(2), pp.121-134.

PBV08 Pries-Heje, J., Baskerville, R. and Venable, J., 2008. Strategies for design science research evaluation. ECIS 2008 proceedings, pp.1-12.

PM16 Process Mining Group, Math&CS department, Eindhoven University of Technology
"http://www.processmining.org/_media/courses/processmining/lecture3_controlflowminingalgorithms.pdf "

RMLA14 Rubin, V.A., Mitsyuk, A.A., Lomazova, I.A. and van der Aalst, W.M., 2014, September. Process mining can be applied to software too. In Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (p. 57). ACM.

SHPRL11 Sein, M., Henfridsson, O., Purao, S., Rossi, M. and Lindgren, R., 2011. Action design research.

SMB95 Silver, M.S., Markus, M.L. and Beath, C.M., 1995. The information technology interaction model: A foundation for the MBA core course. MIS quarterly, pp.361-390.

SS14 Saylam, R. and Sahingoz, O.K., 2014. A process mining approach in software development and testing process: a case study. In Proceedings of the World Congress on Engineering (Vol. 1).

TVY90 Takeda, H., Veerkamp, P. and Yoshikawa, H., 1990. Modeling design process. AI magazine, 11(4), p.37.

WBJL03 W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering, 47, 2(Nov. 2003), 237-267

Appendix A. Database tables structure

Menu Table

Menu Information table has the following fields:

- **menuinfo_id**
  This is the primary key in the table for menu. It is auto generated.
- **menu_type**
  This filed is used to add the process start keyword generated by the menu in event log files. In our case these are MainmenuScr1, Vertical DigitField Screen 1, Horizontal DigitField Screen 1.
- **menu_name**
  This files represent the name of the menu displayed in the Monitor Software.
- **menu_status**
  This files represent whether the menu is “Single” or “Extended”.
- **type**
  This filed is used whether the menu information is “Normal” or “Exceptional”. There are some cases where carescape patient monitor software don’t log all the user activities. In those scenarios, we add some exceptions.

Condition Table

Condition table has the following fields:

- **condition_id**
  This is the primary key in the table for condition. It is auto generated.

- **condition_index**
  This filed is used to index of the condition location into the string message. The detail of how to add condition is further described. Multiple condition indexes can be added with comma separated string.

- **condition_text**
  This field represent the condition text. Multiple condition text can be added with comma separated string.

- **condition_type**
  This field represent whether the condition has is “Single” or “Multiple” strings.

- **active**
  This field is used to active or inactive any condition for further use.

Appendix B. Data after applying the structure algorithm

structure_id	case_id	data_id	structure_order	structure_date	structure_type	structure_message	structure_lineno	structure_thread
44	1	2,483	26	2016-Jan-13 16:52:23	HR DigitField:Vertical DigitField Screen 1	<USR>	ServiceDataCtrl.cpp:338	GUI
45	1	2,488	27	2016-Jan-13 16:54:00	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
46	1	2,505	28	2016-Jan-13 16:54:13	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
47	1	2,507	29	2016-Jan-13 16:54:17	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
48	1	2,509	30	2016-Jan-13 16:54:22	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
49	1	2,528	31	2016-Jan-13 16:54:37	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
50	1	2,539	32	2016-Jan-13 16:54:42	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
51	1	2,554	33	2016-Jan-13 16:54:52	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
52	1	2,556	34	2016-Jan-13 16:54:54	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
53	1	2,576	35	2016-Jan-13 16:55:05	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
54	1	2,578	36	2016-Jan-13 16:55:05	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
55	1	2,680	37	2016-Jan-13 16:57:38	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
56	1	2,693	38	2016-Jan-13 16:57:46	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
57	1	2,710	39	2016-Jan-13 16:58:08	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
58	1	2,735	40	2016-Jan-13 16:58:20	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
59	1	2,752	41	2016-Jan-13 16:58:36	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
60	1	3,175	42	2016-Jan-13 17:56:09	Monitor Setup:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
61	1	3,176	42	2016-Jan-13 17:56:11	Screen Setup:Monitor Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
62	1	3,179	43	2016-Jan-13 17:57:23	Monitor Setup:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
63	1	3,180	43	2016-Jan-13 17:57:26	Screen Setup:Monitor Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
64	1	3,181	43	2016-Jan-13 17:57:40	Tab_Lower Parameter Area:Screen 1:Screen Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
65	1	3,182	43	2016-Jan-13 17:57:43	Tab_Split Screen:Screen 1:Screen Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
66	1	3,183	43	2016-Jan-13 17:57:45	Tab_Upper Parameter Area:Screen 1:Screen Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
67	1	3,184	43	2016-Jan-13 17:57:47	Tab_Screen 2:Screen Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
68	1	3,186	43	2016-Jan-13 17:58:03	Tab_Screen 1:Screen Setup	<USR>	ServiceDataCtrl.cpp:338	GUI
69	1	3,800	44	2016-Jan-14 10:29:16	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
70	1	3,830	45	2016-Jan-14 10:33:16	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI
71	1	3,831	46	2016-Jan-14 10:33:21	Zero All Pressures:MainMenuScr1	<USR>	ServiceDataCtrl.cpp:338	GUI

Figure B.1: Structured data in table tblstructureddata

Appendix C. Detailed Data View

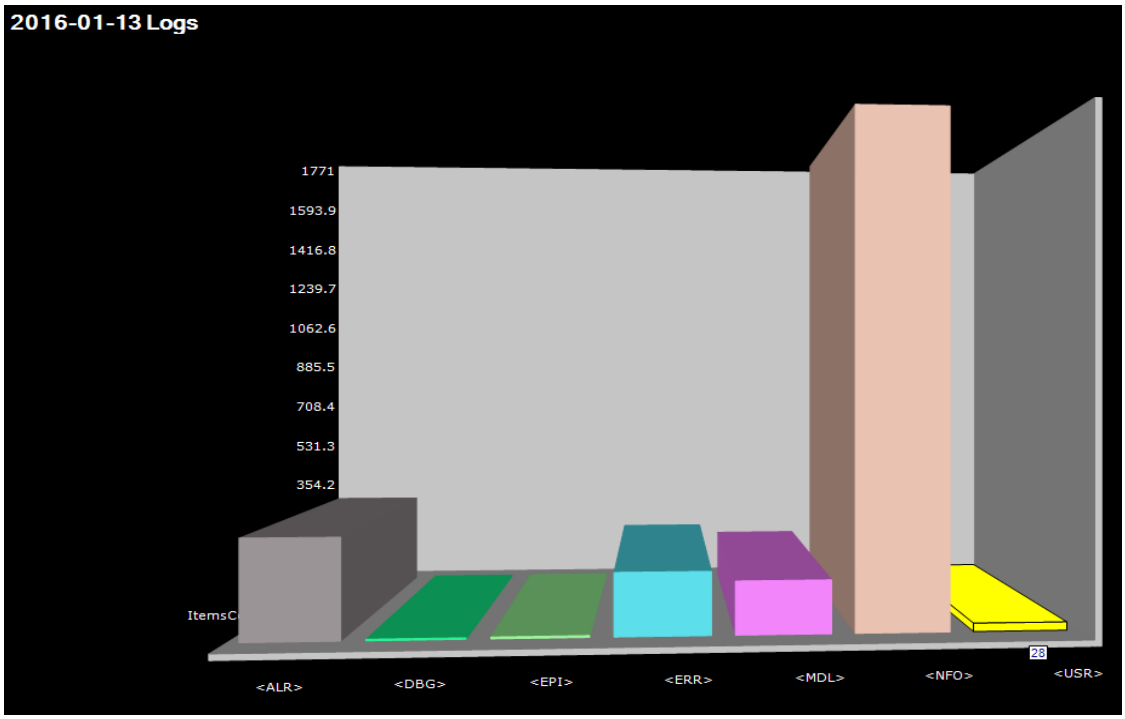

Figure C.1: Detailed data view for message type

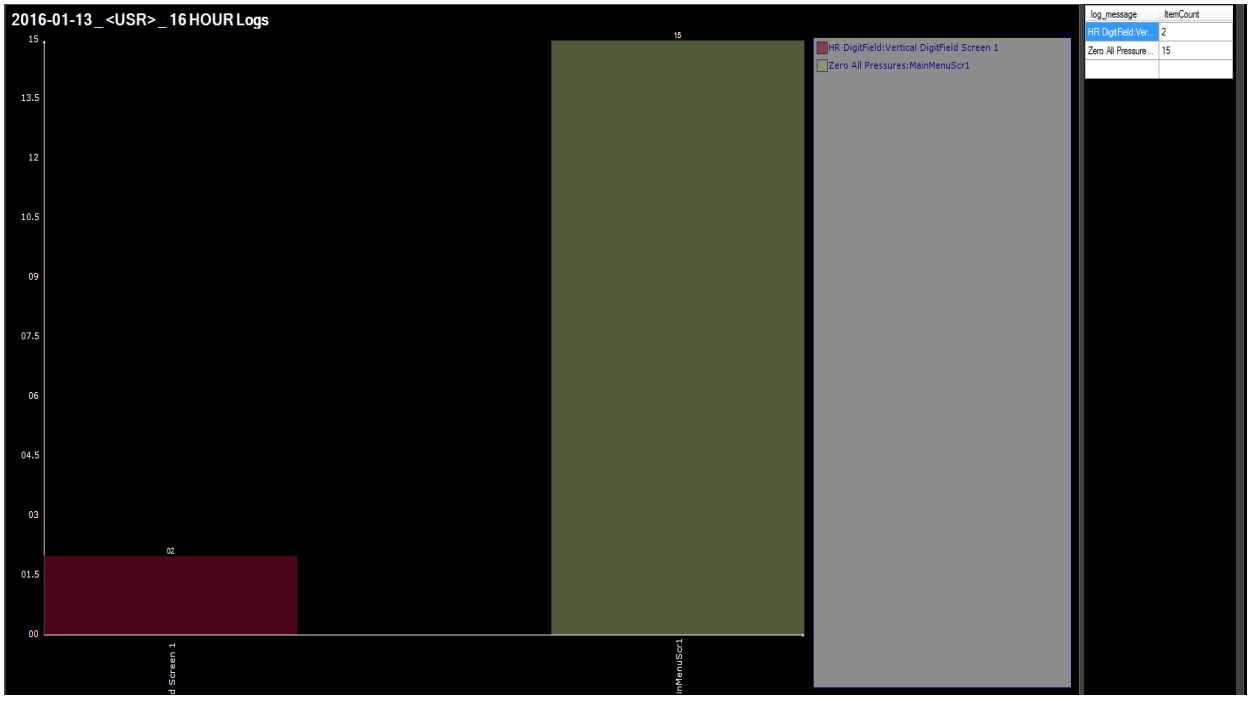

Figure C.2: Detailed data view with messages

Appendix D. Sub activities flow diagrams

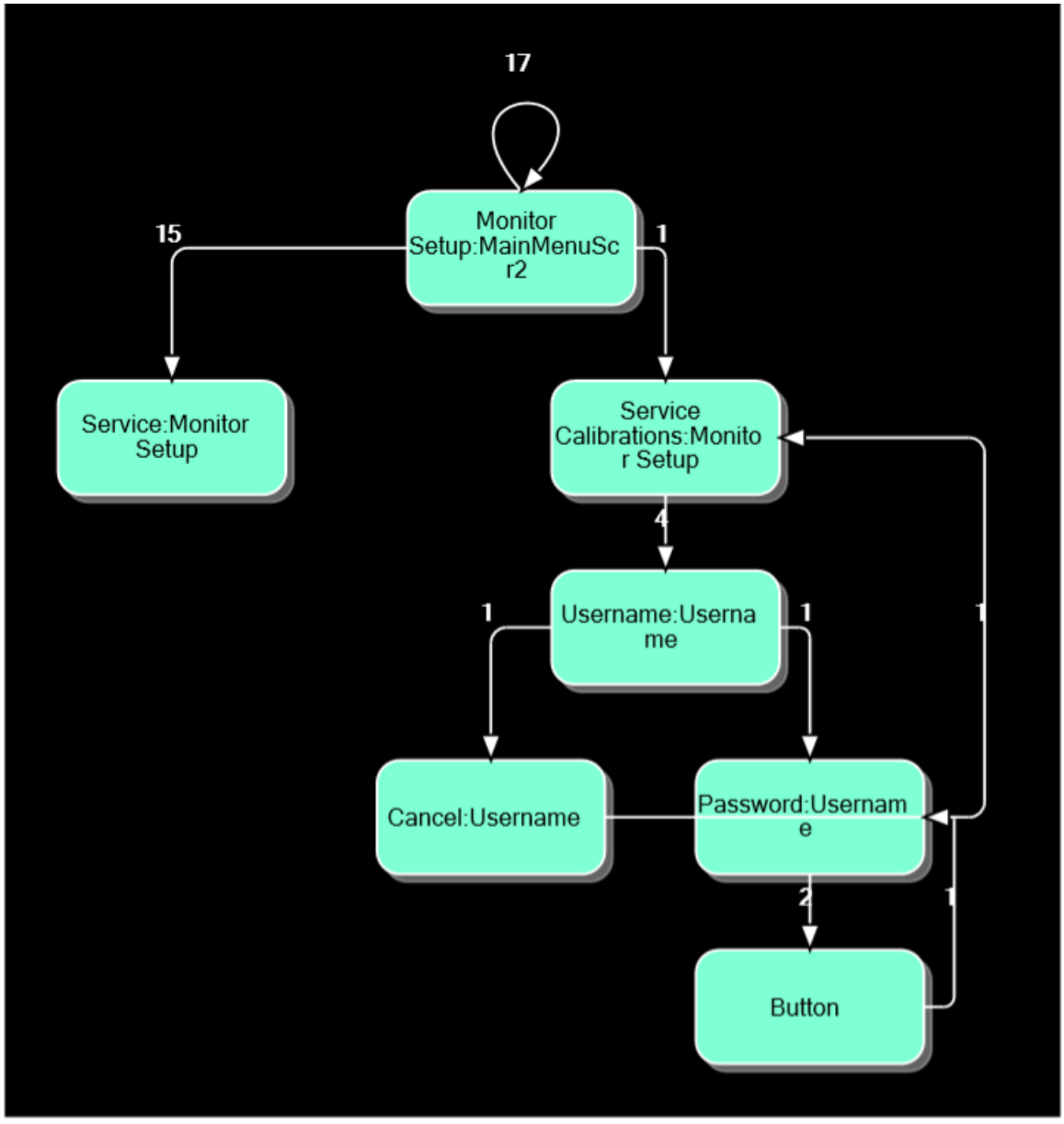

Figure D.1: Sub activities for “MonitorSetup:MainMenuScr2”

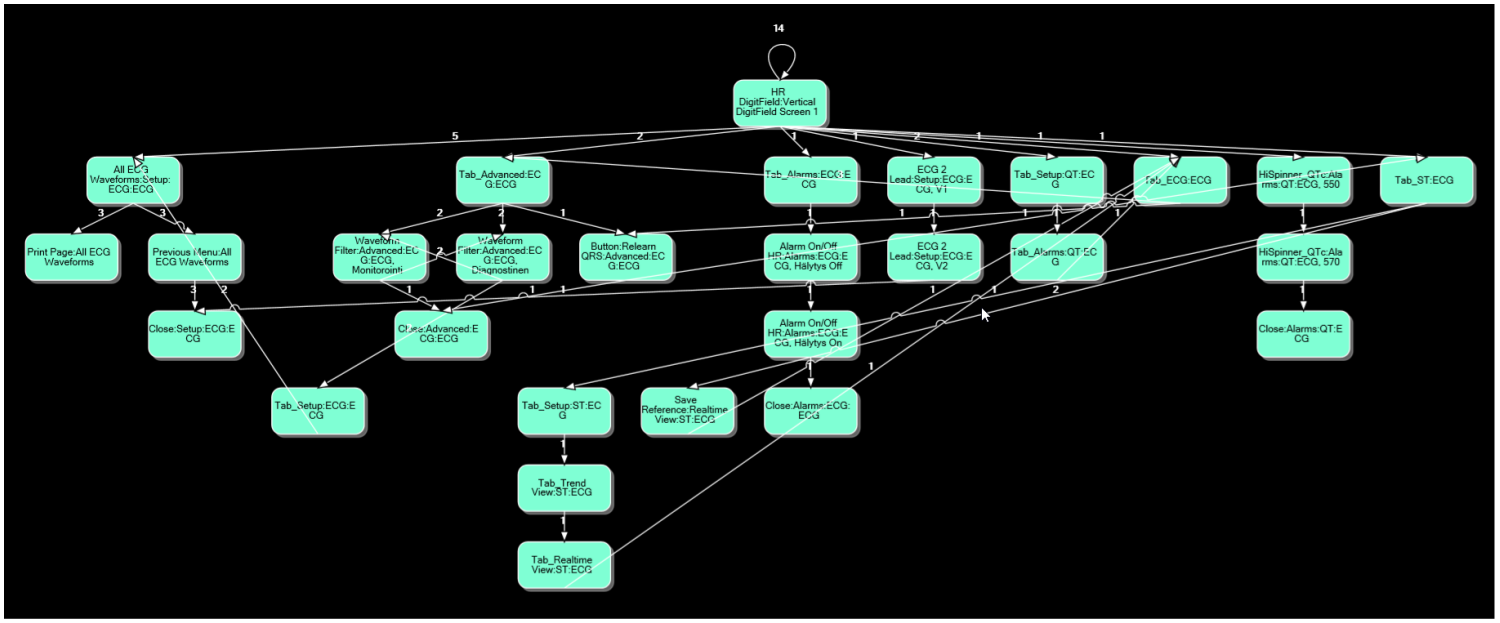

Figure D.2: Sub activities for “HRDigitField:VerticalDigitfieldScr1”

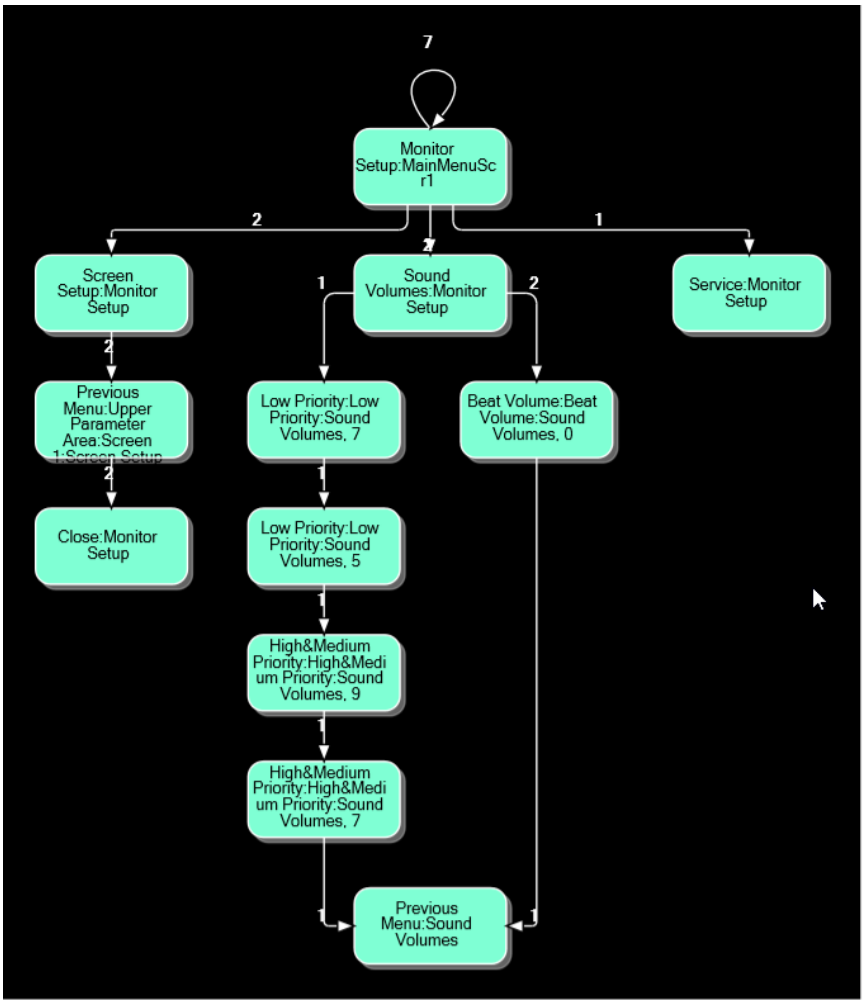

Figure D.3: Sub activities for “MonitorSetup:MainMenuScr1”

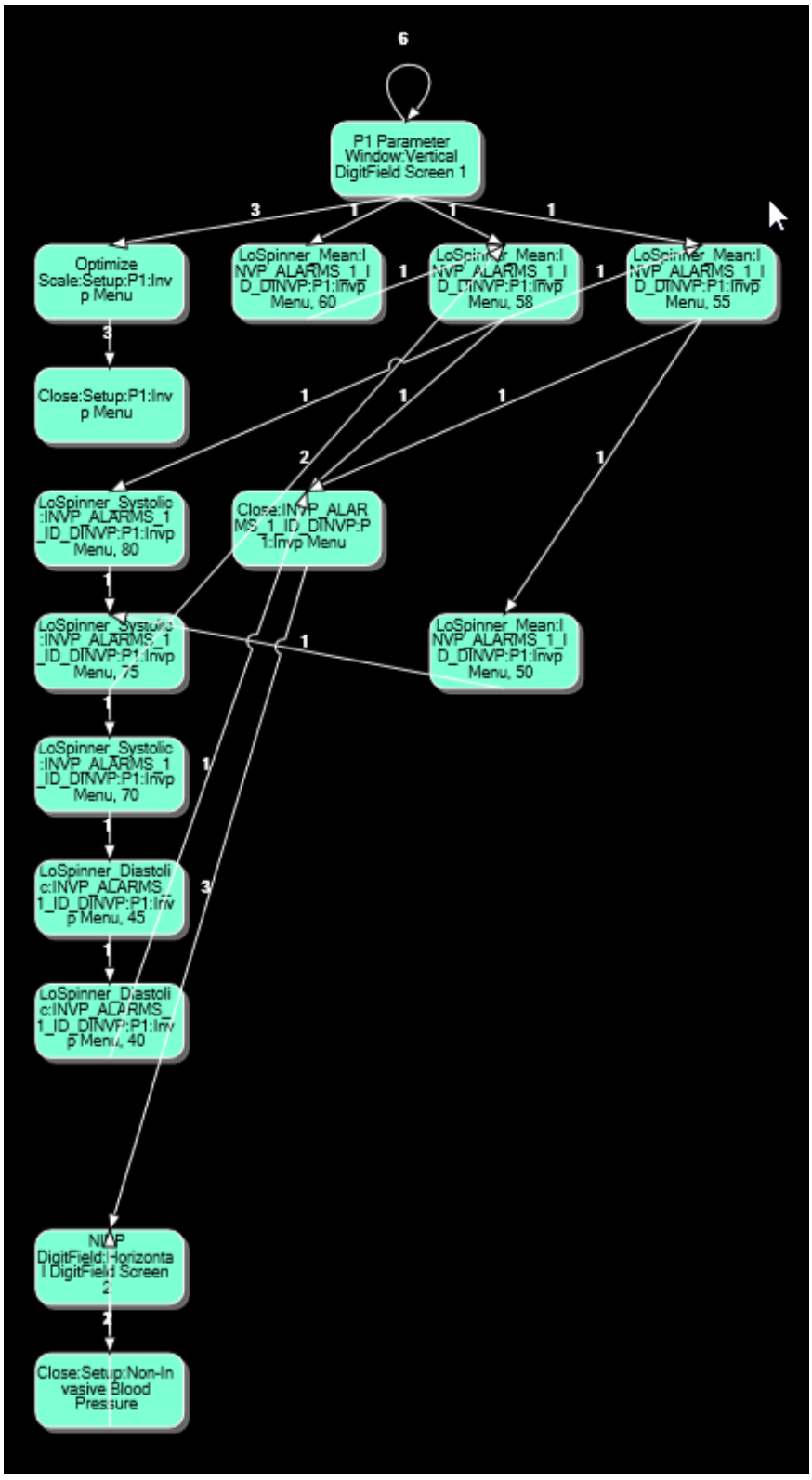

Figure D.4: Sub activities for “P1 Parameter Window:Vertical DigitField Screen”

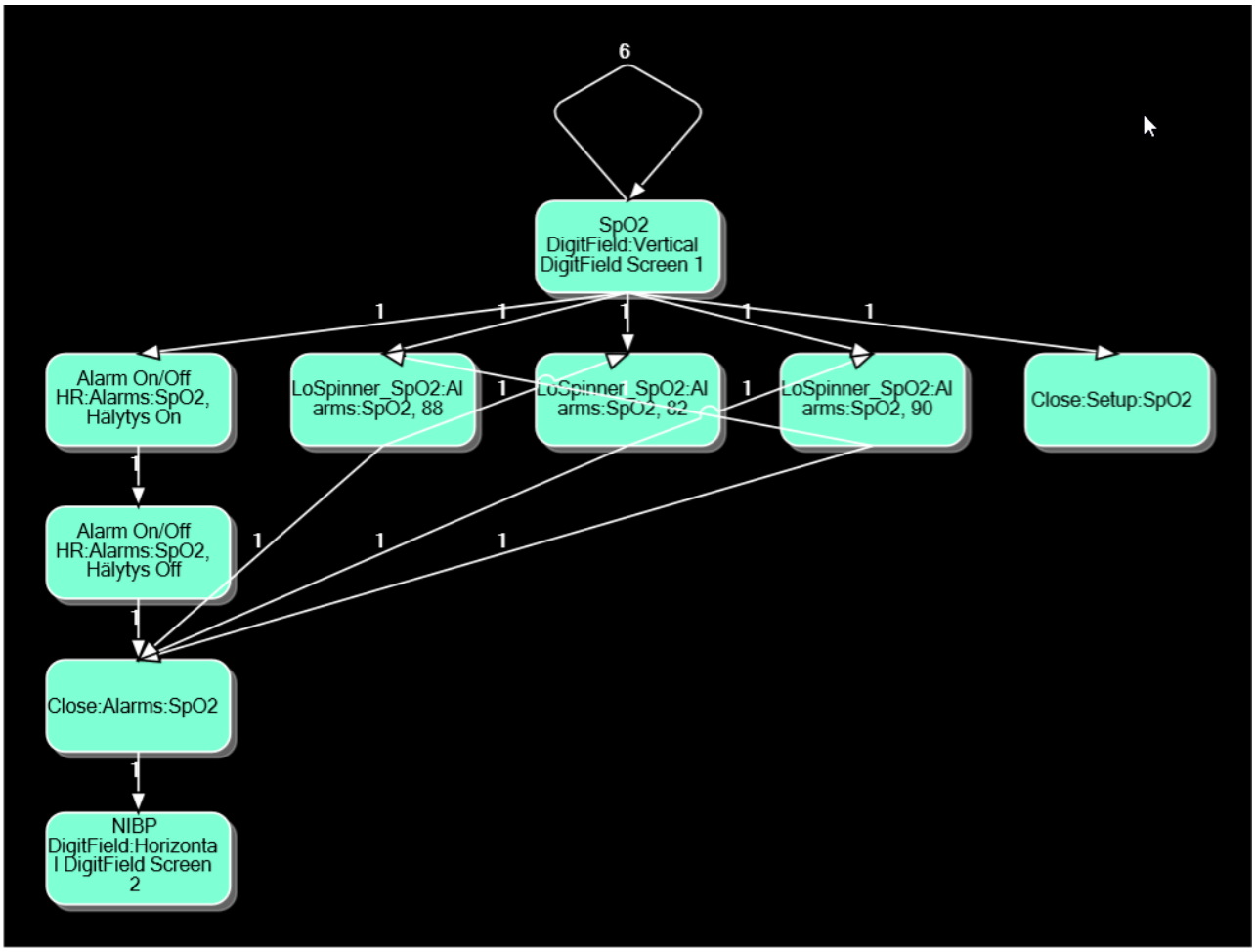

Figure D.5: Sub activities for “SpO2 DigitField:Vertical DigitField Screen 1”

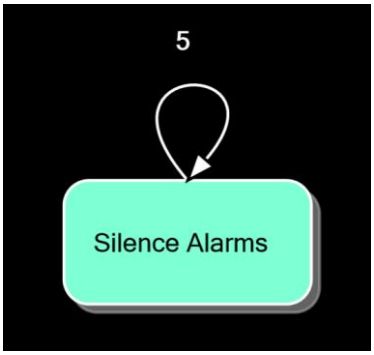

Figure D.6: Main activity “Silence Alarms” with no subactivities

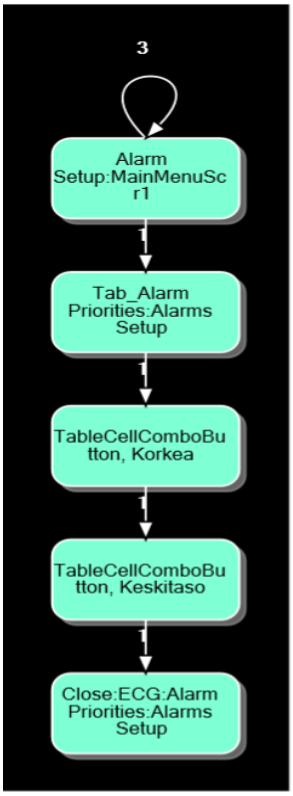

Figure D.7: Sub activities for “Alarm Setup:MainMenuScr1”

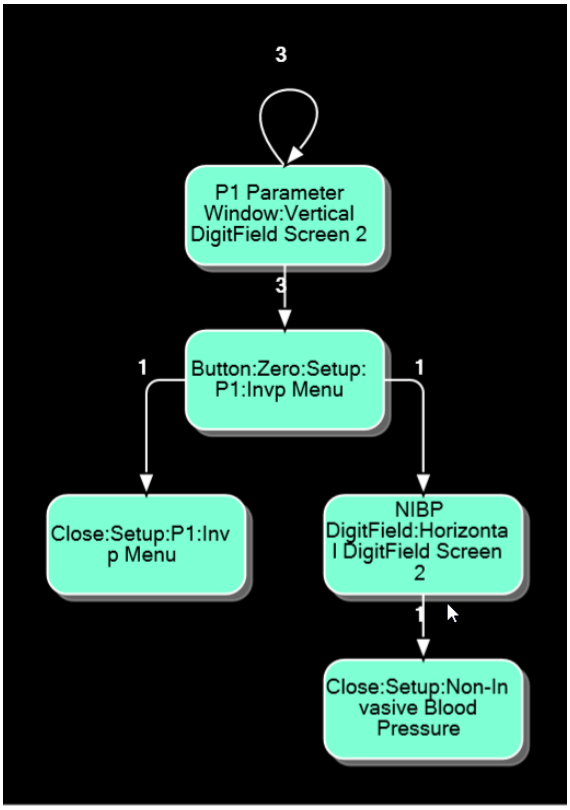

Figure D.8: Sub activities for “P1 Parameter Window:Vertical DigitField Screen 2”

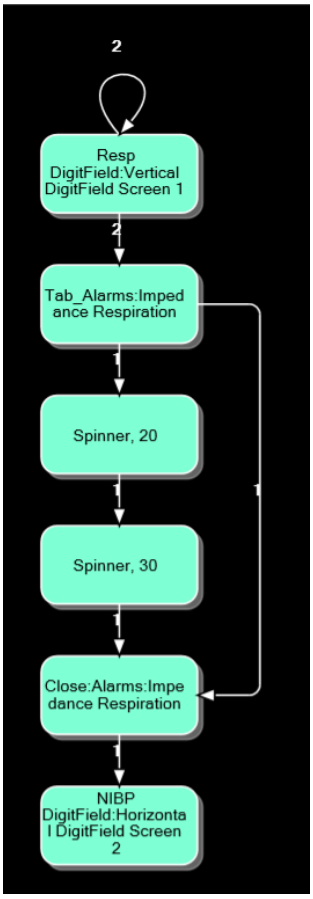

Figure D.9: Sub activities for “Resp DigitField:Vertical DigitField Screen 1”

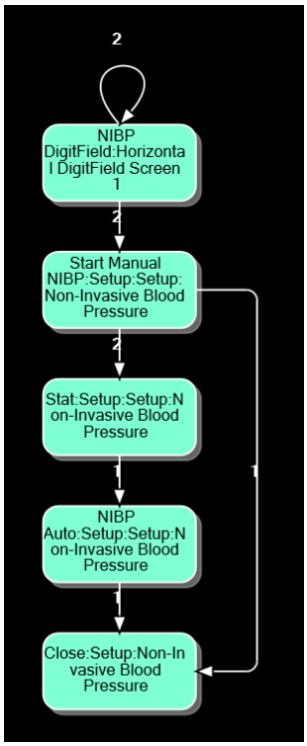

Figure D.10: Sub activities for “NIBP DigitField:Horizontal DigitField Screen 1”

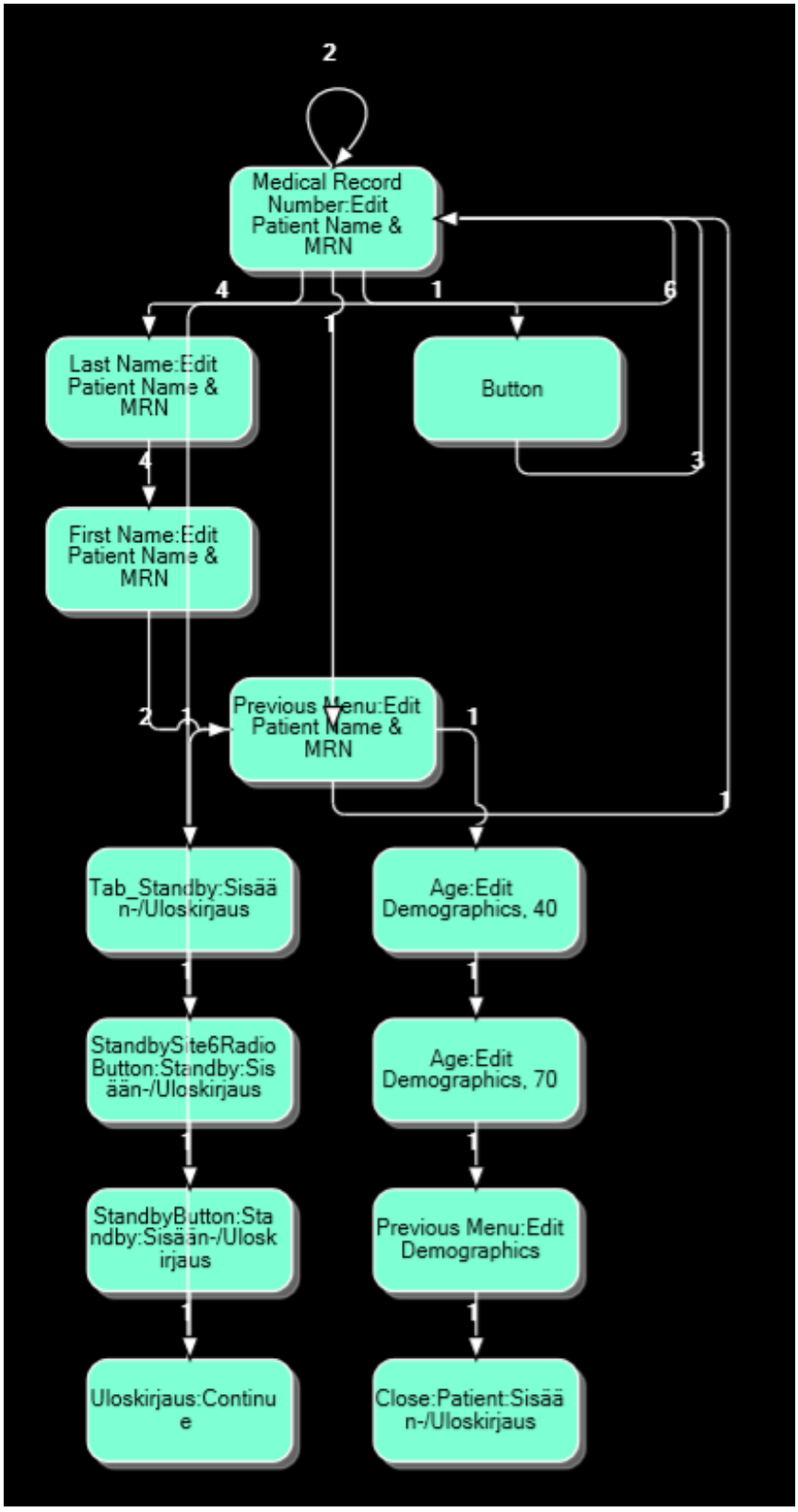

Figure D.11: Sub activities for “Medical Record Number:Edit Patient Name & MRN”

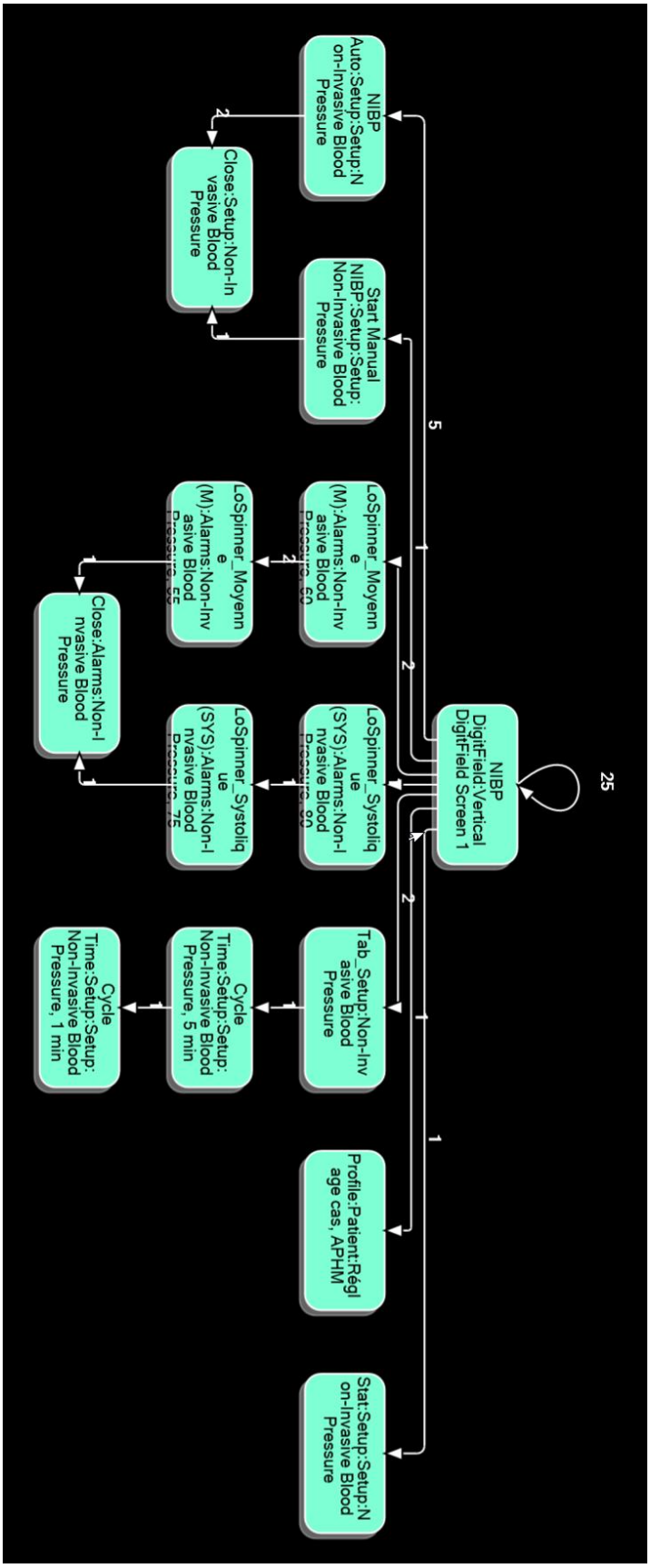

Figure D.12: Sub activities for “NIBP DigitField:Vertical DigitField Screen 1”

Appendix E. Help File for Log Analyzer

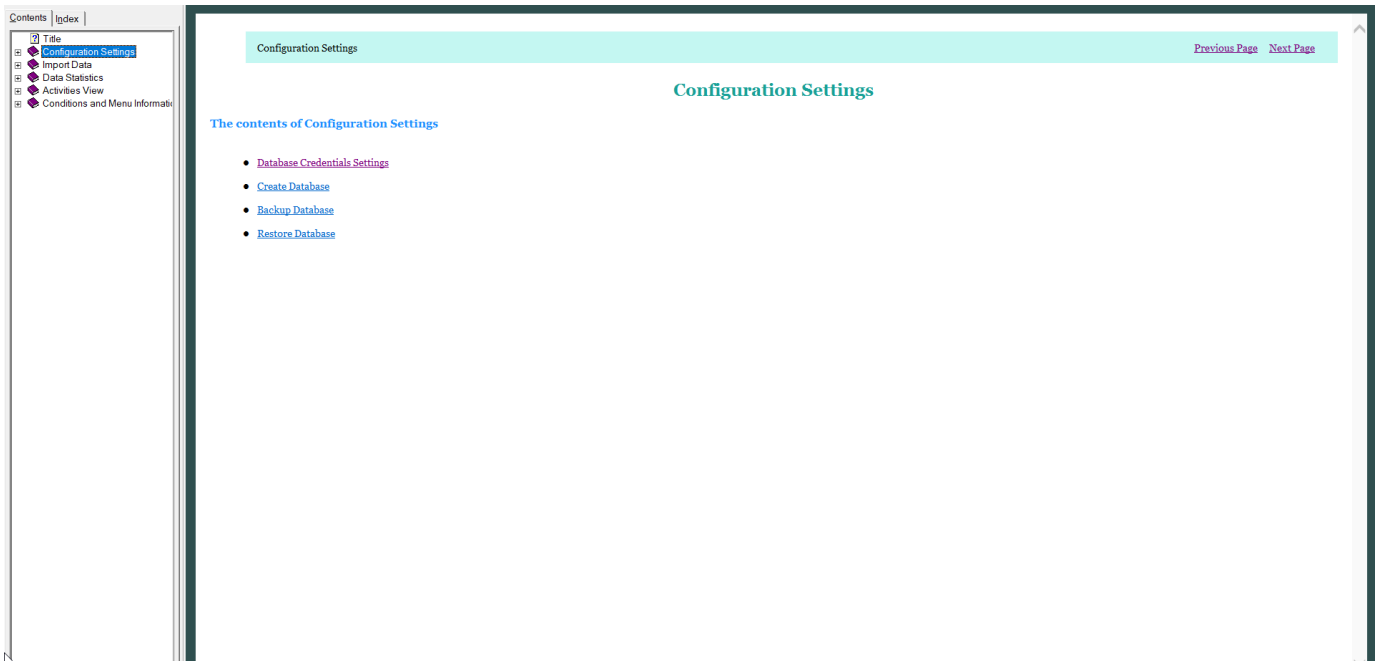

Figure E.1: Help file main page

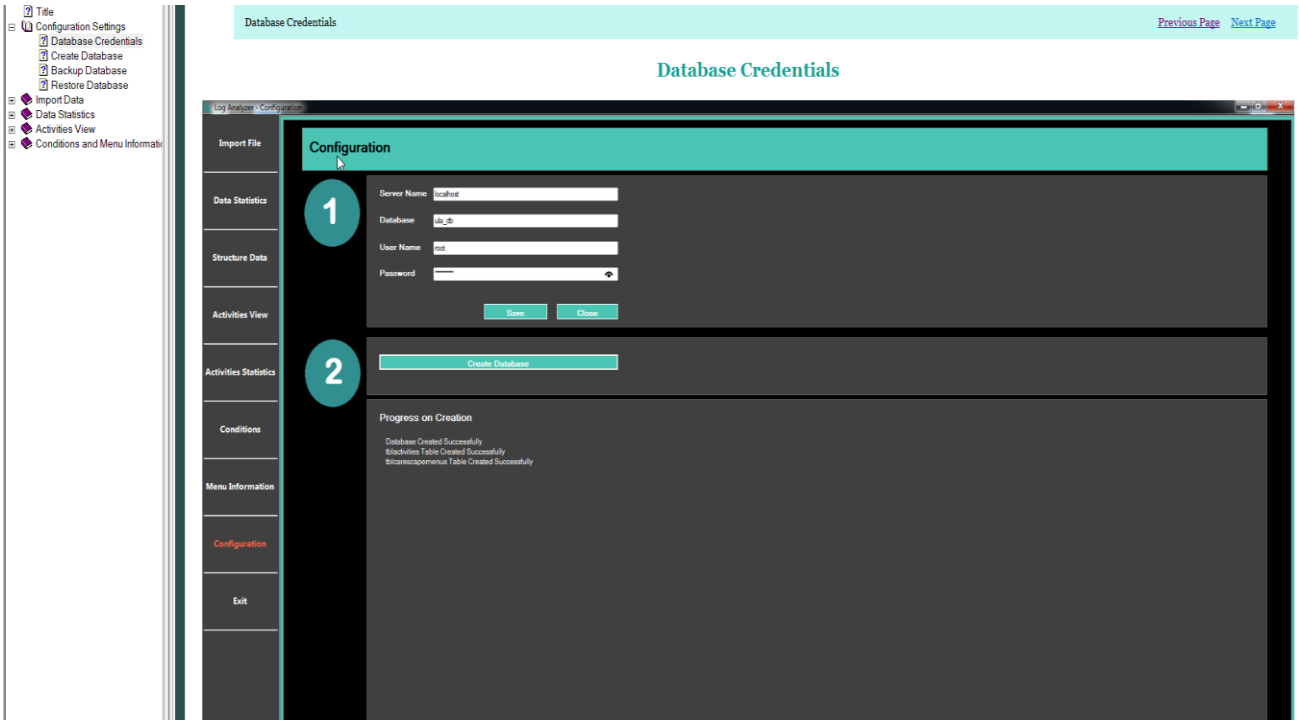

Figure E.2: Help instructions for adding database credentials

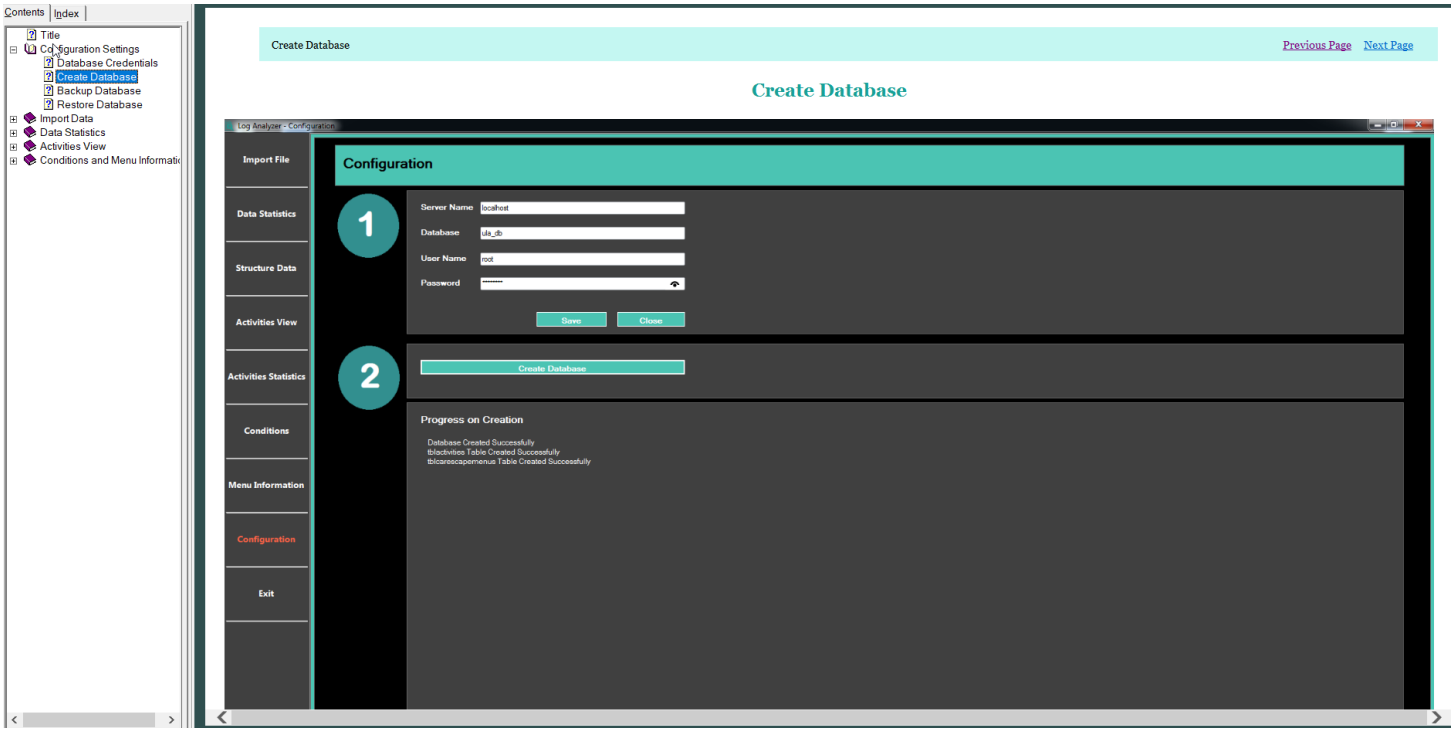

Figure E.3: Help instructions for creating database

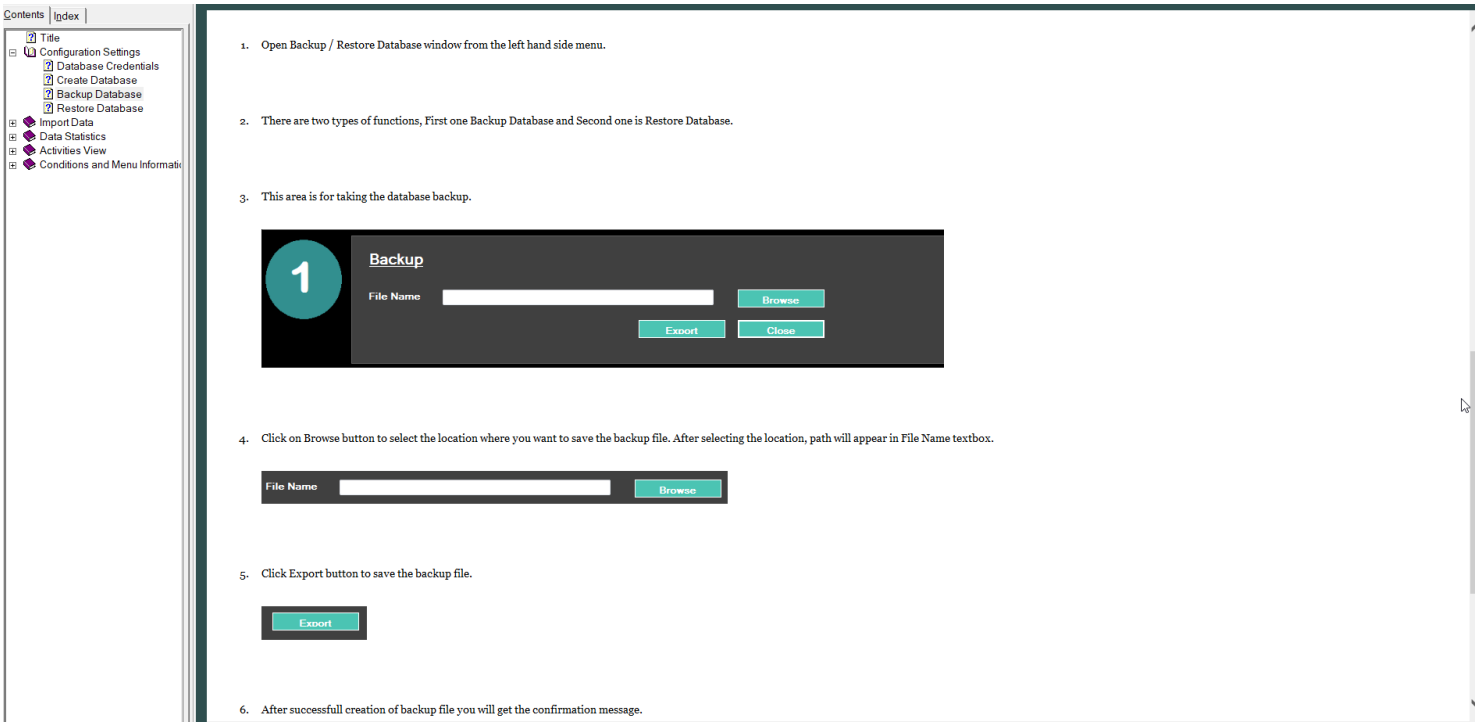

Figure E.4: Help instructions for database backup

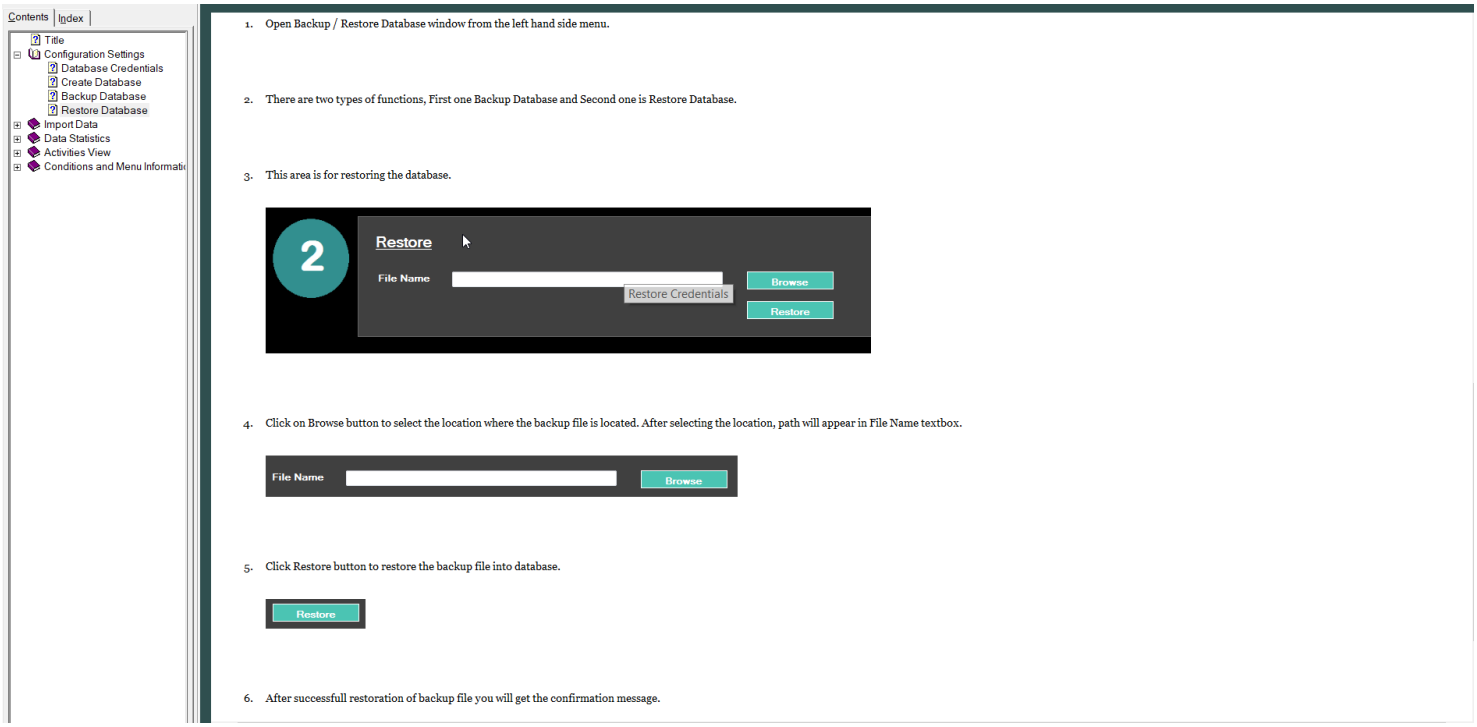

Figure E.5: Help instructions for restore database

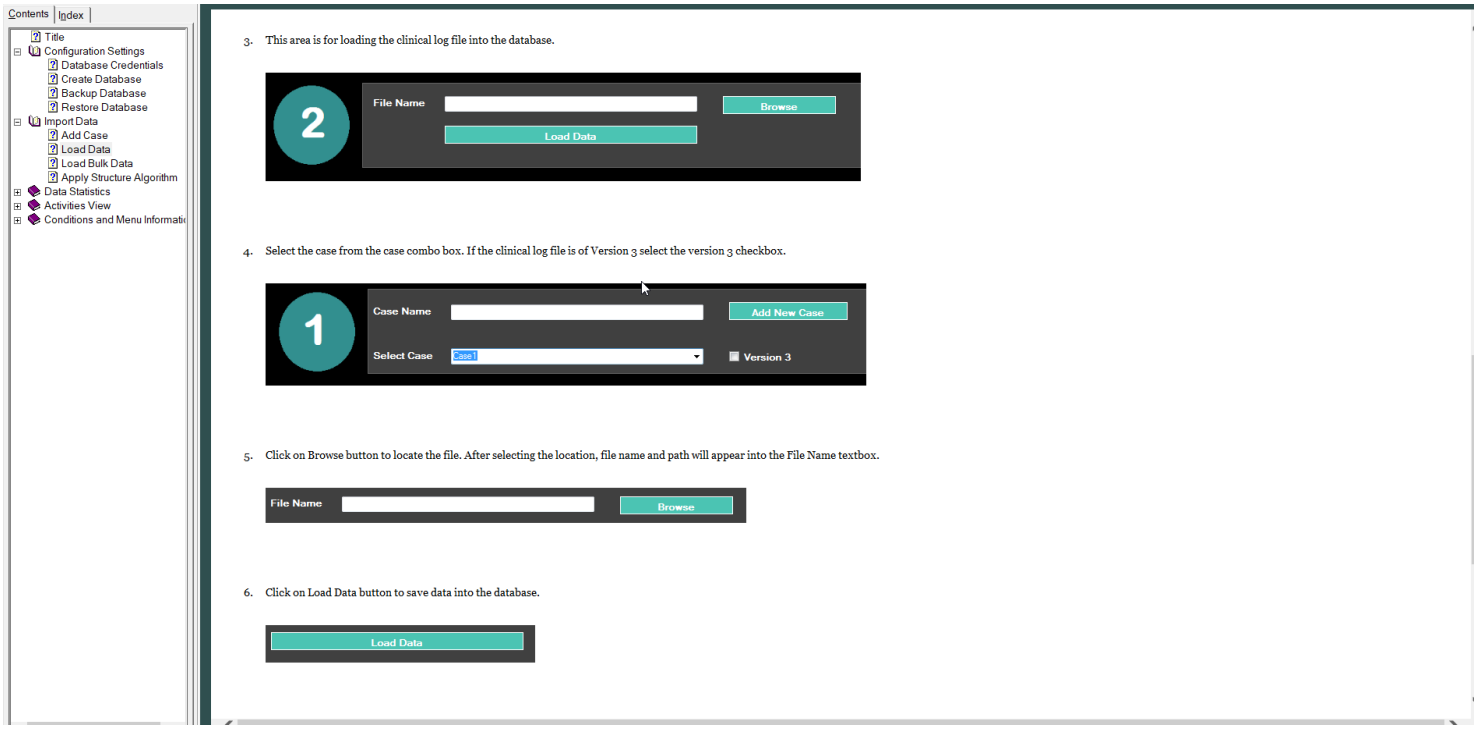

Figure E.6: Help instructions for load data into tool

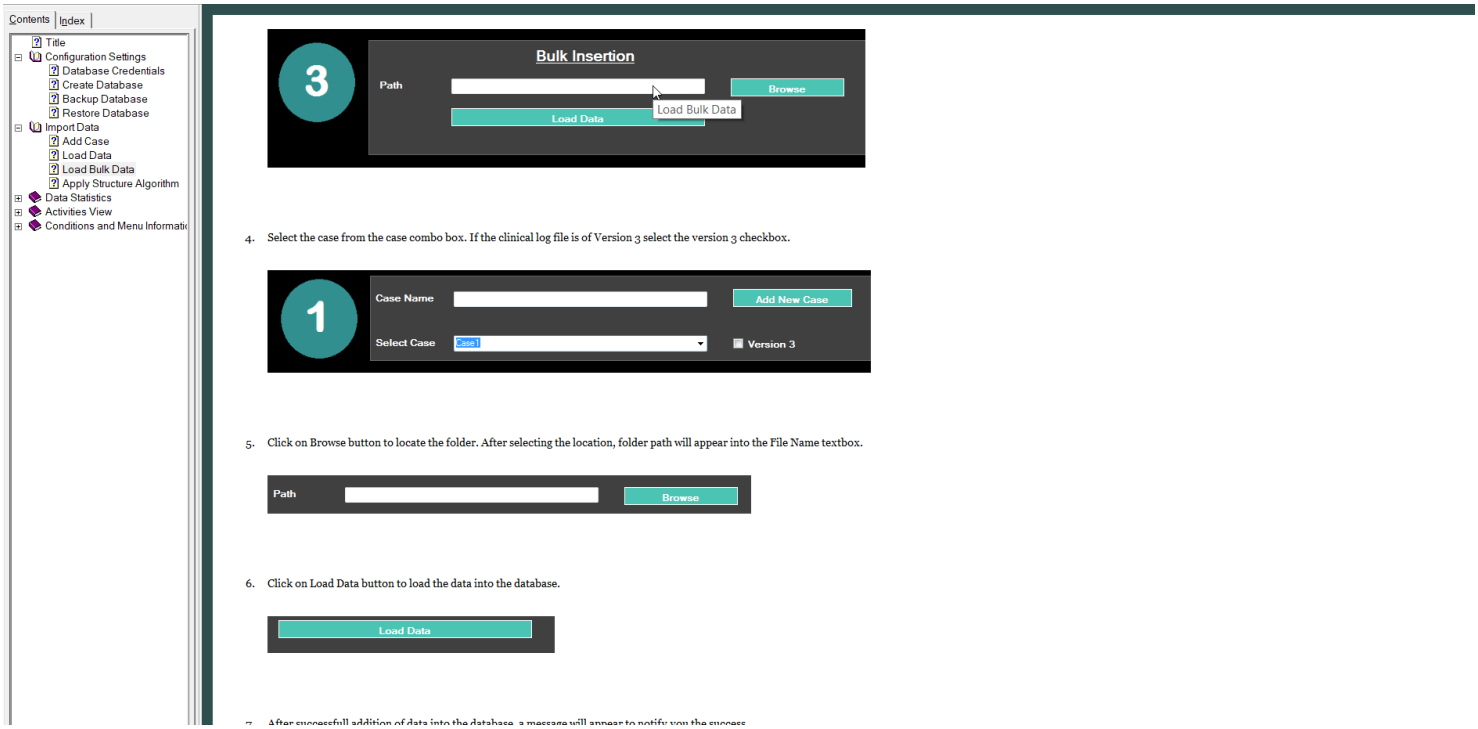

Figure E.7: Help instructions for load bulk data into tool

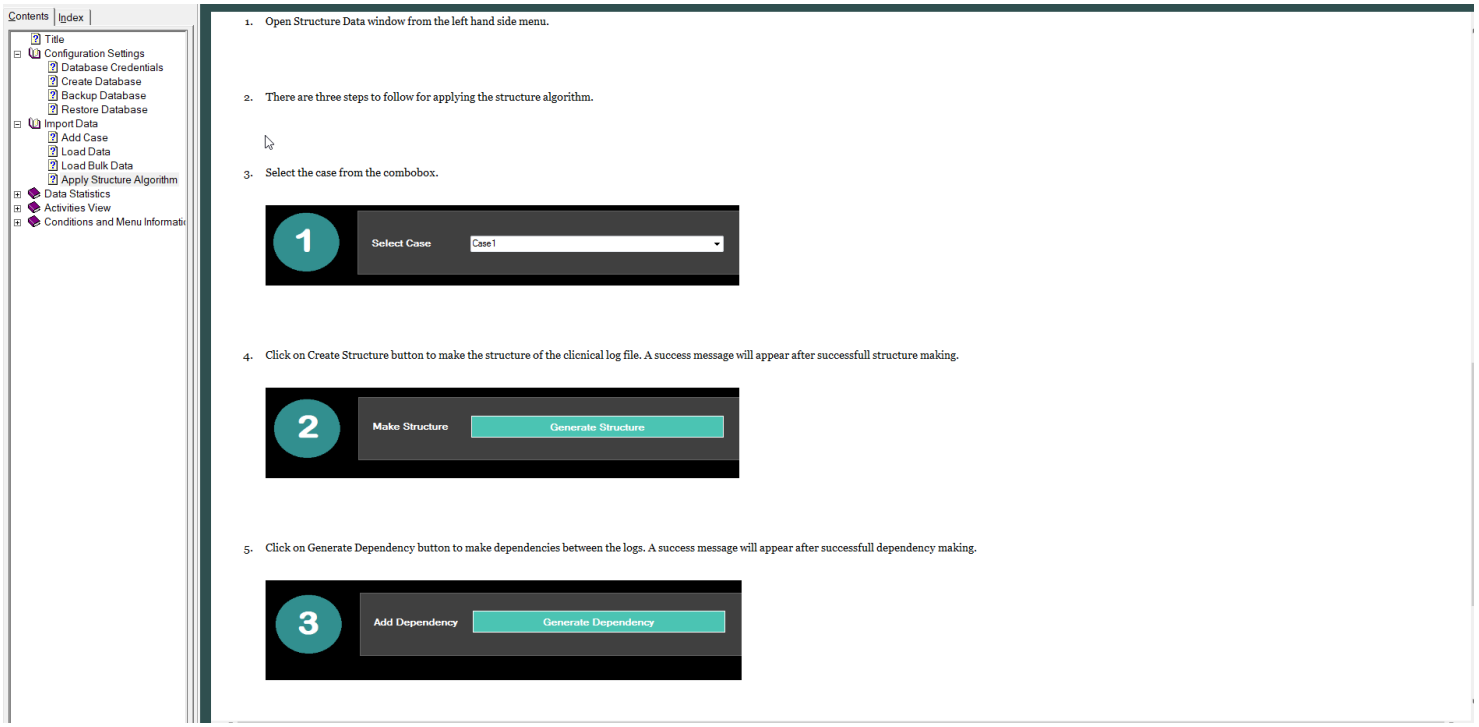

Figure E.8: Help instructions for running algorithms

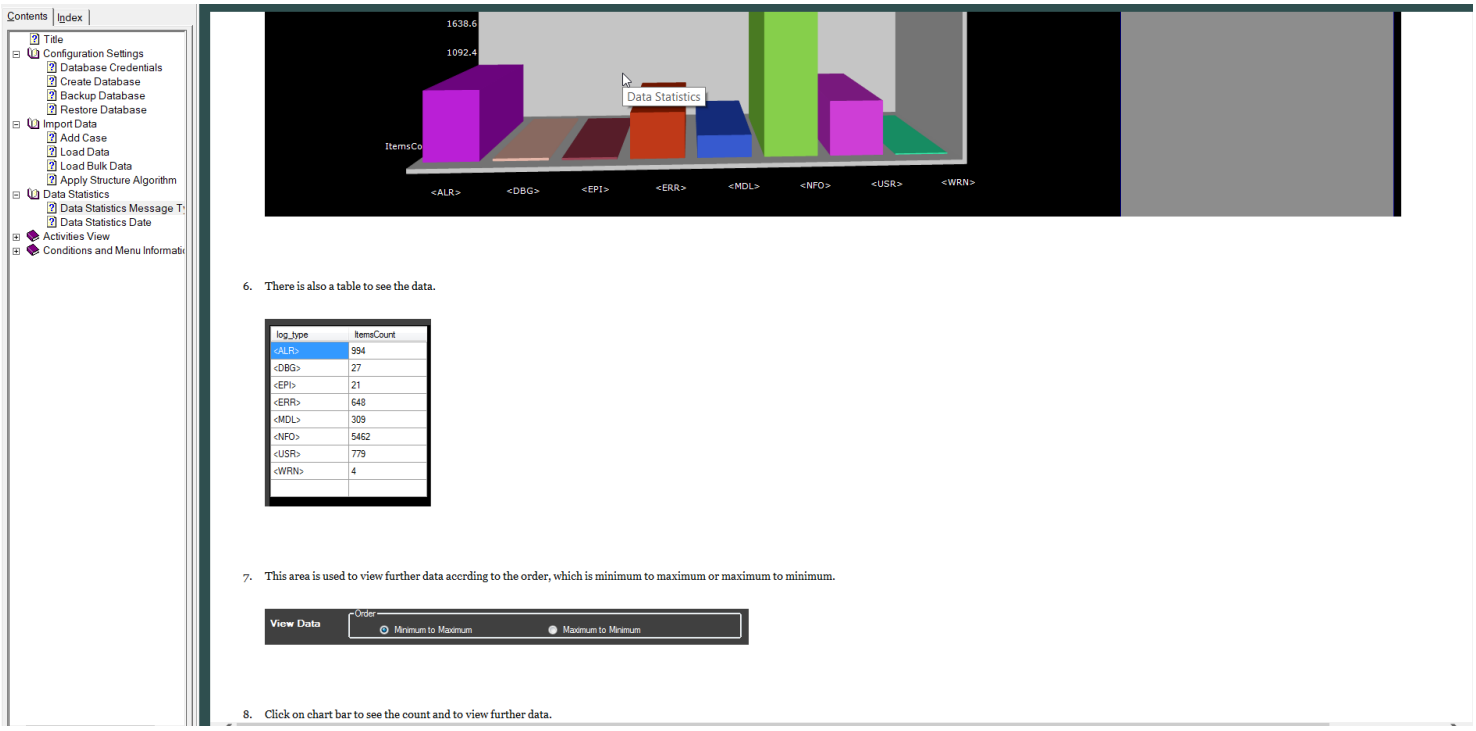

Figure E.9: Help instructions for viewing data statistics

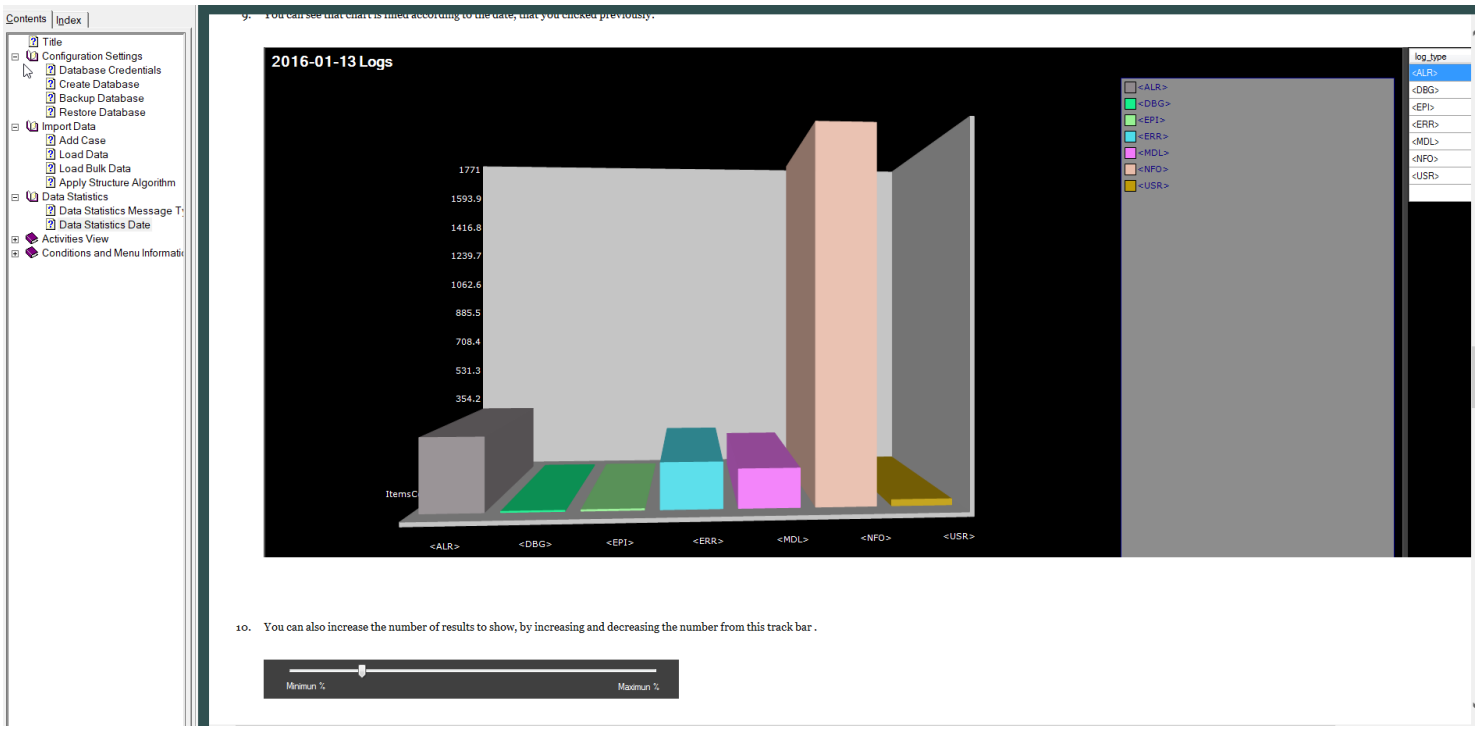

Figure E.10: Help instructions for viewing data statistics according to date

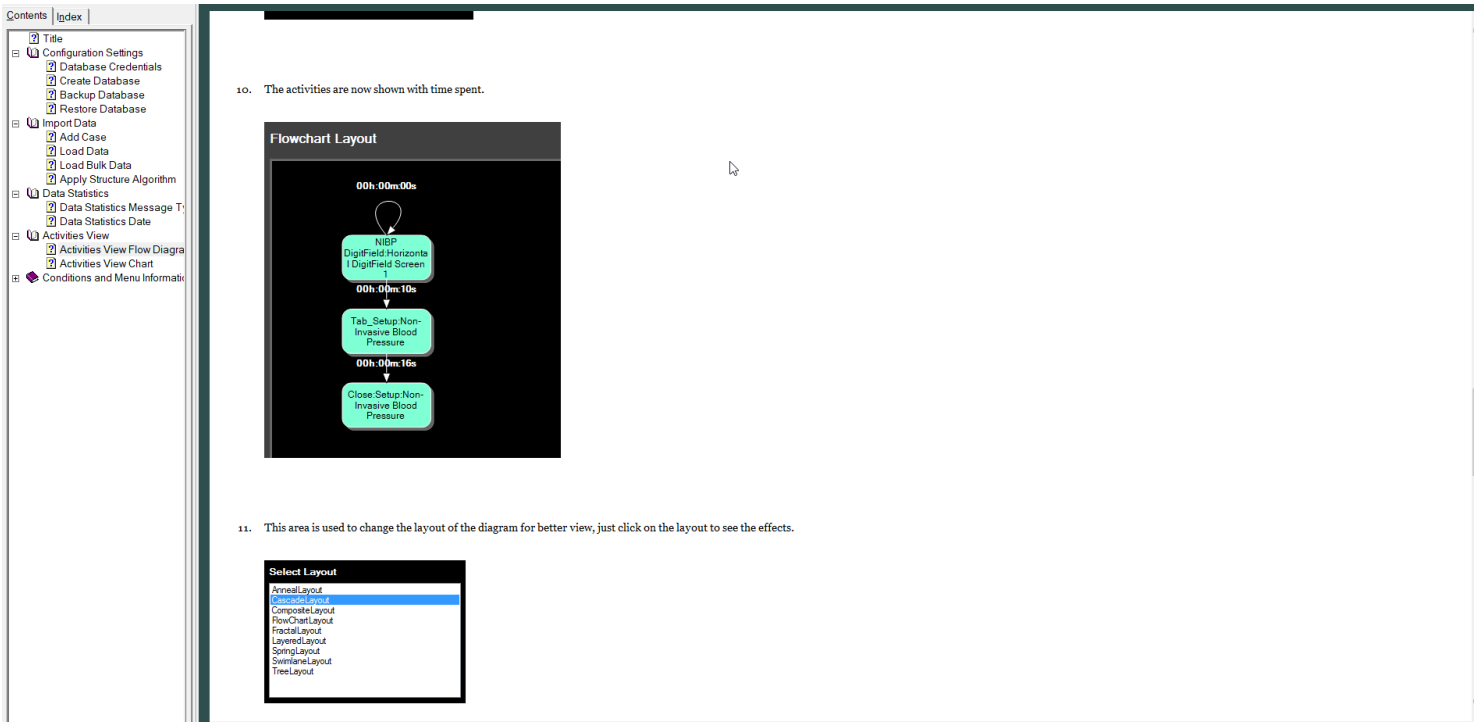

Figure E.11: Help instructions for showing activities

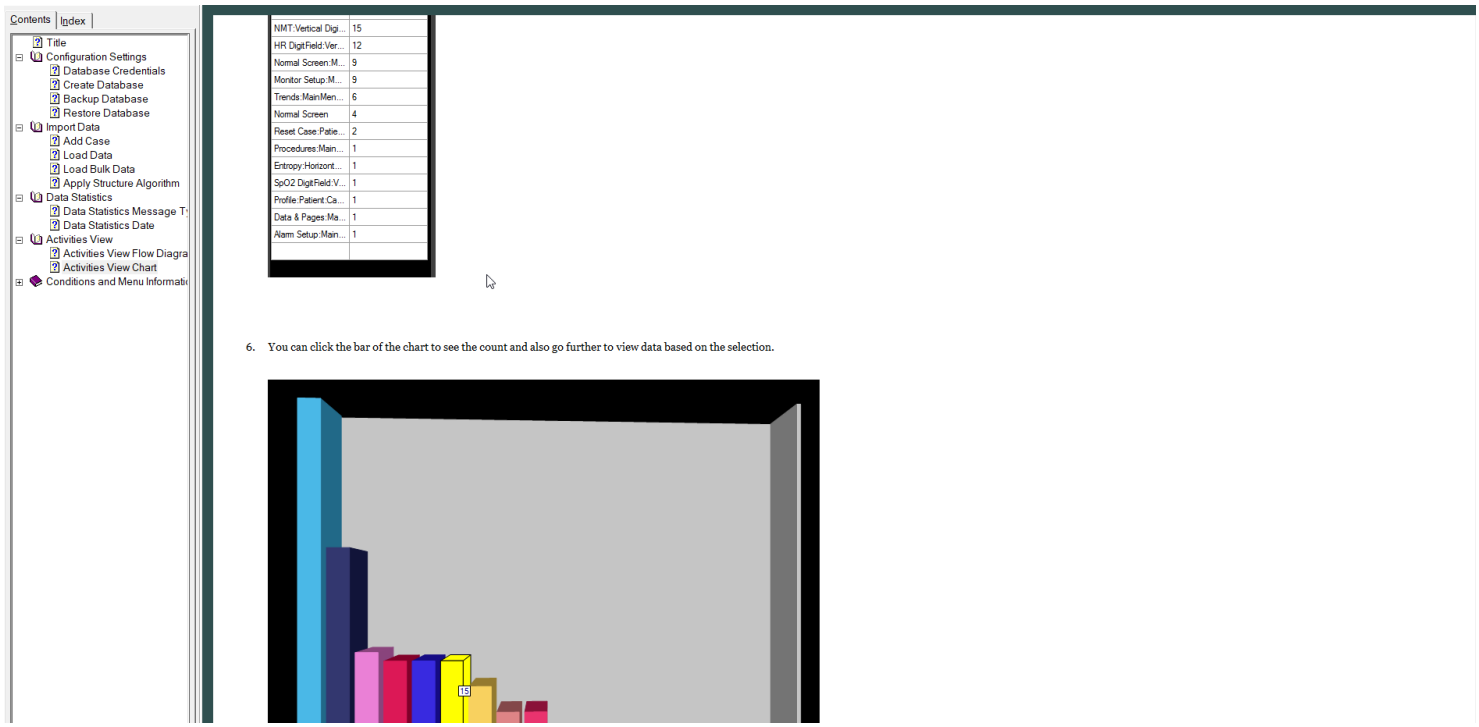

Figure E.12: Help instructions for viewing activities chart

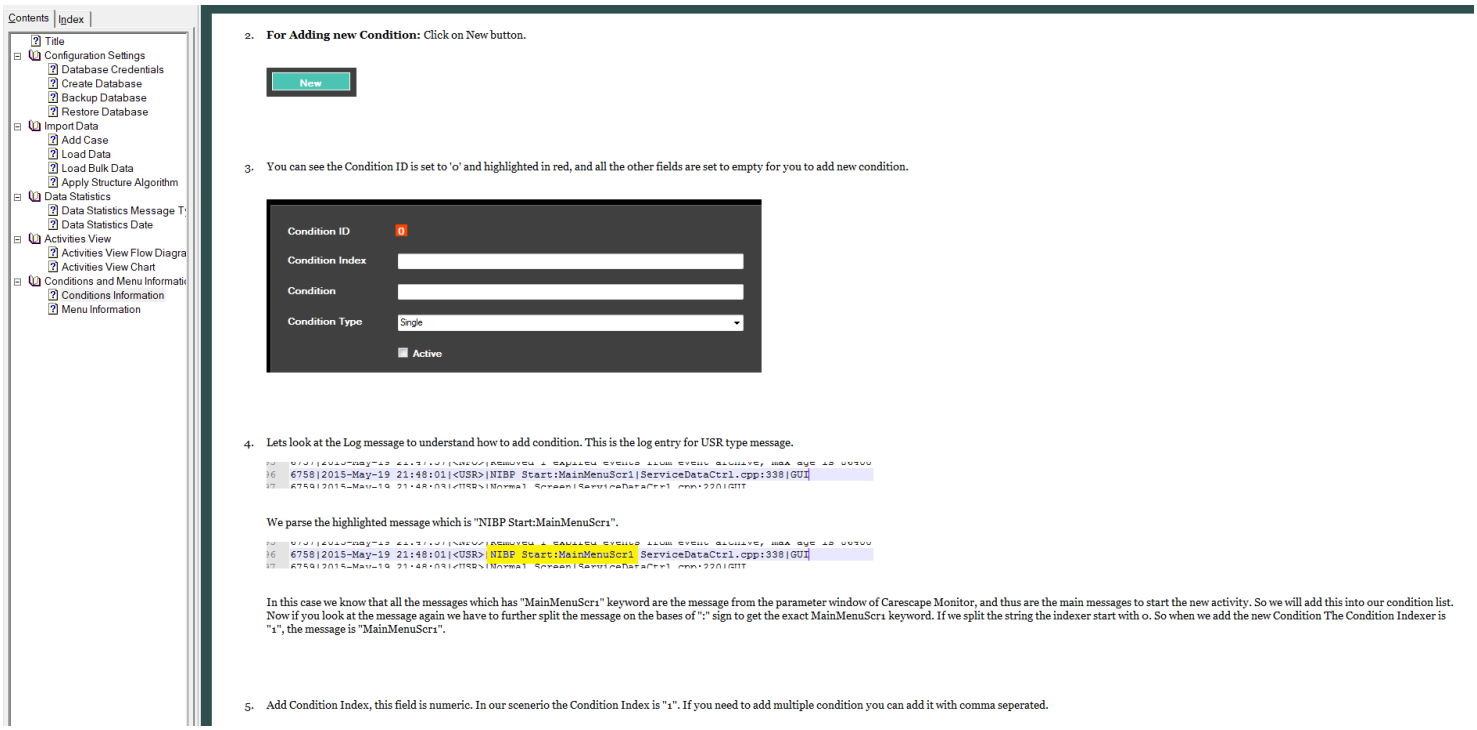

Figure E.13: Help instructions for adding exceptions into conditions table

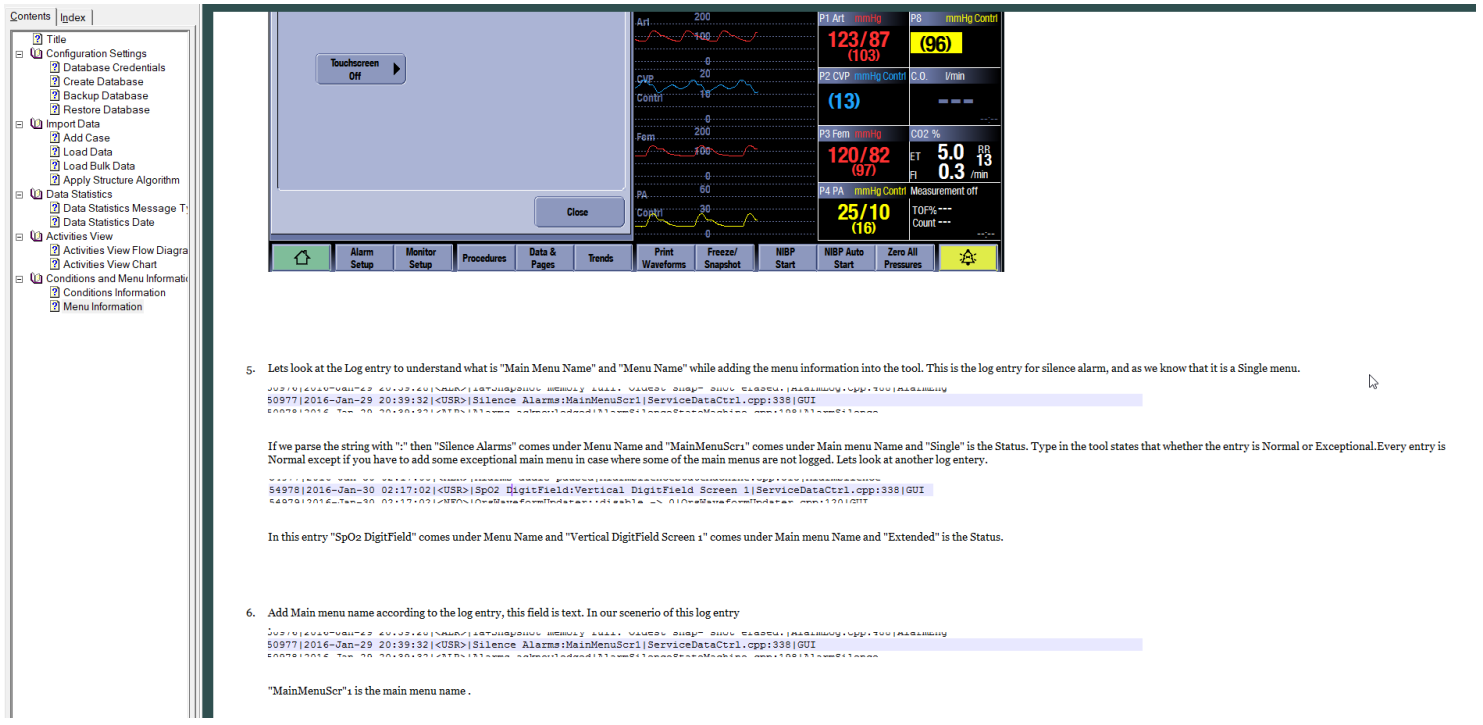

Figure E.14: Help instructions for adding/editing menu information

Appendix F. User instructions for adding exceptional cases into tblcondition

1. Open Conditions window from the left-hand side menu of Log Analyzer tool.

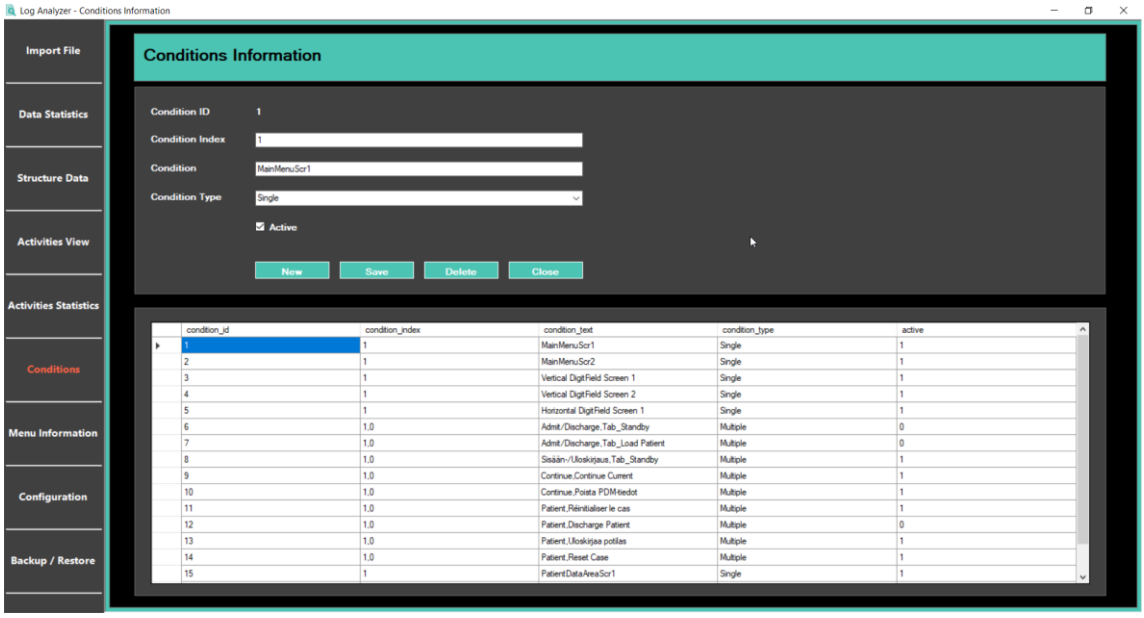

Figure F.1: Condition View

2. For adding new condition, click on **New** button.
3. You can see the Condition ID is set to '0' and highlighted in red, and all the other fields are set to empty for you to add new condition.

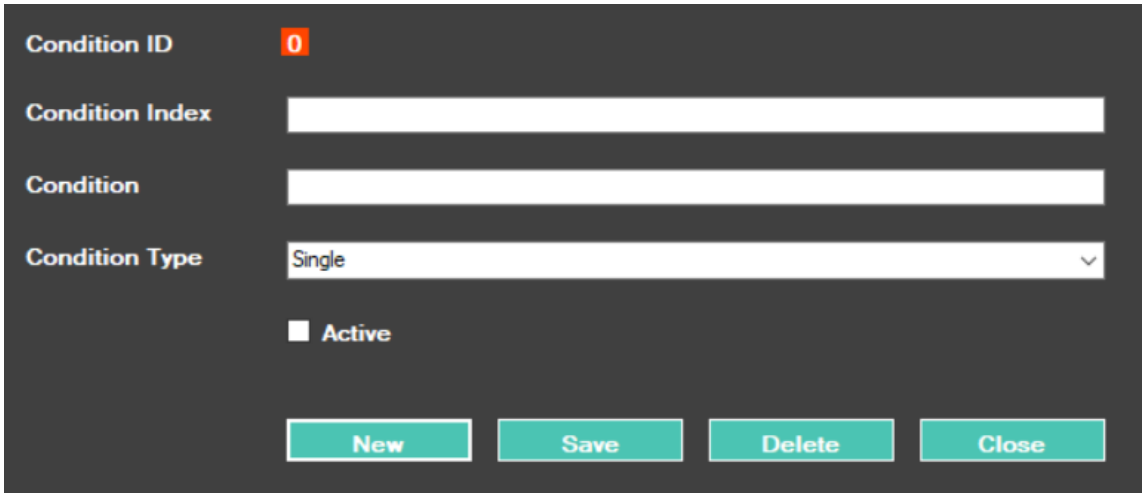

Figure F.2: Condition View for adding new condition

4. Lets look at the Log message to understand how to add condition. Figure F.3 shows the log entry for USR type message. Message "NIBP Start:MainMenuScr1" will be parsed in this case. From this case it is known that all the messages which has "MainMenuScr1" keyword are the message from the parameter window of Carescape Patient Monitor, and thus are the main messages to start the new activity. So this will be added into the condition list. Now by looking at the message again it is seen that there is a need to further split the message on the bases of ":" sign to get the exact MainMenuScr1 keyword. If string is split the indexer start with 0. So when new condition is added ,the condition index is "1", and message is "MainMenuScr1".

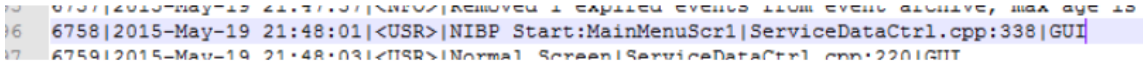

Figure F.3: Log entry

5. Add Condition Index, this field is numeric. In the case from Figure F.3 the condition index is "1". If there is a need to add multiple condition, it can be added with comma seperated.

Figure F.4: Condition index

6. Add Condition message , this field is text and in the case from Figure F.3 the condition name is "MainMenuScr1". If you need to add multiple condition you can add it with comma seperated.

Figure F.5: Condition name

7. There is a possibility to add multiple conditions in exceptional cases to make the start of the activity. This filed differenciate the type of condition as single or multiple.

Figure F.6: Condition type

8. Lets look at the log messages in Figure F.7 to understand the scenerio of multiple condition. The purple highlighted text is the log entry for USR type message. It is seen that for some reasons carescape patient monitor software is not logging the shortcut of patient menu click. The highlighted message in Figure F.7 shows that Close:Patient:Admit/Discharge is after that shortcut click which is not logged. So it should be activity start as there is other option. According to this scenerio the message is parsed and the index is added for multiple condition as 0,1 as now there is a need of two AND conditions like "Where message1 = "Close" AND message2 = "Patient"". So in the condition message is added with comma and contidion type as Multiple.

```
83|2016-Mar-17 11:09:32|<ERR>|The command (nodeId: -1 type: 1, cmd: 113, value: 123) is not sent to the module|EegConfiguration.cpp:796|EegThread
84|2016-Mar-17 11:09:32|<ERR>|The command (nodeId: -1 type: 1, cmd: 114, value: 40) is not sent to the module|EegConfiguration.cpp:796|EegThread
85|2016-Mar-17 11:09:32|<ERR>|The command (nodeId: -1 type: 1, cmd: 115, value: 241) is not sent to the module|EegConfiguration.cpp:796|EegThread
86|2016-Mar-17 11:09:32|<NFO>|PDM warm start 1, cold start 0, case active 1, net case 1|PatientCaseMgr.cpp:751|PatientCaseMainThread
87|2016-Mar-17 11:09:32|<NFO>|TA: START|TrendArchive.cpp:202|TrendArchive
88|2016-Mar-17 11:09:32|<NFO>|MSG_PATIENTCASEMGR_USER_DECISION_NEEDED, Continue menu to be opened|PatientMsgHandler.cpp:426|GUI
89|2016-Mar-17 11:09:32|<NFO>|ContinueMenu - setMode CD_VISIBLE|ContinueMenuFactory.cpp:799|GUI
90|2016-Mar-17 11:09:53|<NFO>|NIBP: sendInflationLimitToSrc(2): sendMsgToSrc 0 (0x00)|NibpParamSS.cpp:2558|NibpThread
91|2016-Mar-17 11:09:53|<NFO>|Acq device vitals' status update: connected=1 vitals=1|PatientCaseMain.cpp:2791|PatientCaseMainThread
92|2016-Mar-17 11:09:53|<NFO>|ContinueMenu - nothing to do: no demogr & no unresolved stby|PatientCaseMain.cpp:1448|PatientCaseMainThread
93|2016-Mar-17 11:09:53|<ERR>|Unknown message 0x0003001F|SpO2Param.cpp:877|SpO2Thread
94|2016-Mar-17 11:09:53|<ERR>|Unknown message 0x0003001F|SpO2Param.cpp:877|SpO2Thread
95|2016-Mar-17 11:09:54|<NFO>|Waveforms started in 259977 secs from reset (includes bootloader time 0 s)|WaveformField.cpp:915|GUI
96|2016-Mar-17 11:09:55|<ALR>|Alarm engine started|AlarmsEngine.cpp:82|AlarmEng
97|2016-Mar-17 11:09:55|<NFO>|No stick searching on Windows|CoreDumpStickAlarmsTruthExp.cpp:114|AlarmEng
98|2016-Mar-17 11:09:55|<ALR>|Ia+P8 connected|AlarmLog.cpp:488|AlarmEng
99|2016-Mar-17 11:09:55|<ERR>|No alarm log message registered for alarm 136577029 (2084,5)|AlarmLog.cpp:445|AlarmEng
100|2016-Mar-17 11:09:55|<ALR>|La+DEMO MODE Not for clinical use!|AlarmLog.cpp:488|AlarmEng
101|2016-Mar-17 11:09:55|<ERR>|No network alarm registered for alarm 136577029 (2084,5)|NetApAlarmProducer.cpp:555|AlarmEng
102|2016-Mar-17 11:09:56|<ERR>|checkConnectionStatus: Not implemented for Windows.|WinTcpIp.cpp:209|TcpIpAbsThread
103|2016-Mar-17 11:09:56|<ERR>|isDuplicatedIpPresent: Not implemented for Windows.|WinTcpIp.cpp:216|TcpIpAbsThread
104|2016-Mar-17 11:09:56|<NFO>|network communication is starting|NetworkIfActivityCtrl.cpp:57|TcpIpAbsThread
106|2016-Mar-17 11:10:00|<NFO>|CONTINUE (CD)|PatientCaseMain.cpp:289|PatientCaseMainThread
107|2016-Mar-17 11:10:00|<NFO>|continueCaseOnWarmstart: vitalSignsTimerM enabled|PatientCaseMgr.cpp:588|PatientCaseMainThread
108|2016-Mar-17 11:10:05|<USR>|Close:Patient:Admit/Discharge|ServiceDataCtrl.cpp:338|GUI
109|2016-Mar-17 11:10:26|<ALR>|_i+P8 connected|AlarmLog.cpp:488|AlarmEng
111|2016-Mar-17 11:10:31|<ALR>|_i+DEMO MODE Not for clinical use!|AlarmLog.cpp:488|AlarmSilence
112|2016-Mar-17 11:10:31|<ERR>|Alarm 136577029 calls resetAlarm and resets alarm condition using 'reset(false)' of TruthExpIf. Add reset method to your alarm condi
113|2016-Mar-17 11:10:31|<ALR>|Alarms acknowledged|AlarmSilenceStateMachine.cpp:198|AlarmSilence
114|2016-Mar-17 11:10:32|<ERR>|No alarm log message registered for alarm 136577029 (2084,5)|AlarmLog.cpp:445|AlarmEng
115|2016-Mar-17 11:10:32|<ALR>|La+DEMO MODE Not for clinical use!|AlarmLog.cpp:488|AlarmEng
```

Figure F.7: Log file

9. Click on Save button to add new condition or update condition.

Appendix G. Activities View based on frequency

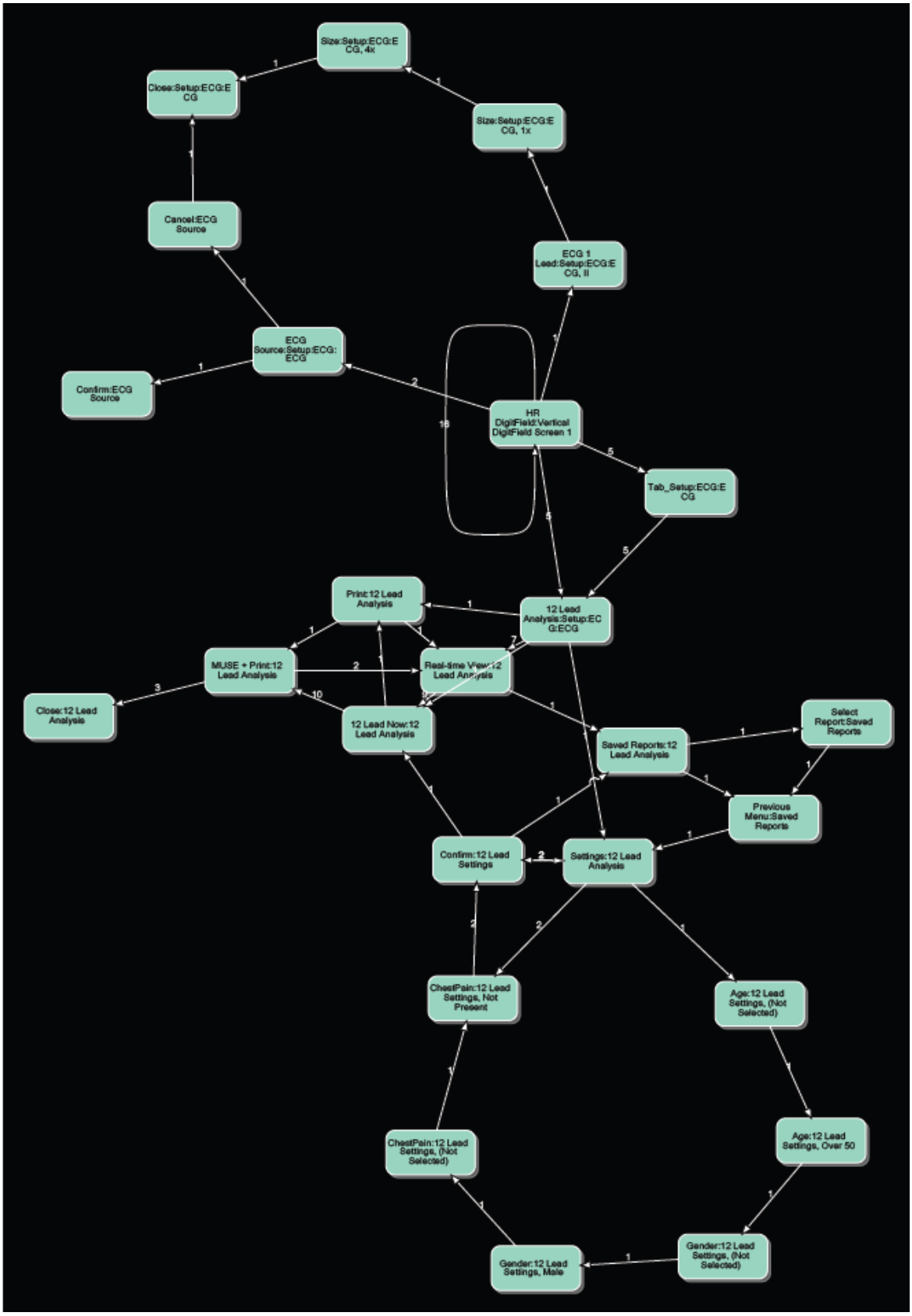

Figure G.1: Activities based on frequency imported in pdf